ISSN 1816-6075 (Print), 1818-0523 (Online)
Journal of System and Management Sciences
Vol. 12 (2022) No. 4, pp. 412-438
DOI:10.33168/JSMS.2022.0425

A Technique of Data Security using DNA Cryptography with Optimized Data Storage

M.K. Padmapriya[1], Pamela Vinitha Eric[2]

[1]Department of CSE, New Horizon College of Engineering, Bengaluru, Karnataka, India 560103
[2]Presidency University, Bengaluru, Karnataka, India 560064

padmapriyabhat@gmail.com (Corresponding author)

Abstract. A technique for data security with improved data storage using the concepts of DNA cryptography and lossless compression is proposed in this work. The problem with the existing cryptographic methods are the increased size of the cipher text which results in an increased storage and data transfer cost. The proposed technique uses a DNA OTP method to change ordinary text into a DNA cipher text, further on the basis of occurrence of the DNA codons a binary code is assigned for each of the DNA nucleotides. This method resulted in a cipher text whose size is smaller than the corresponding plain text. The experimental results of the proposed method showed that the cipher text to plain text ratio is 0.99. Avalanche effect of the proposed method is 63% for a single bit change in the plaintext thus satisfies the strict avalanche criteria and also the algorithm passes the NIST statistical test suites for randomness.

Keywords: ASCII, avalanche effect, cryptography, DNA, encryption, NIST, OTP, security.

1. Introduction

These days, the quantity of information created and saved on computing gadgets is growing at an alarming rate. Following the pandemic, the rapid and widespread adoption of remote working drastically increased the need for cloud-based services and infrastructure, posing security concerns for businesses. Tremendous quantities of essential and sensitive statistics are transmitted between most of these gadgets. Data saved on local PCs or cloud servers is susceptible to various threats (Corallo et al., 2020; Bin et al., 2021; Guo 2018; Macnish et al., 2020). When sensitive, confidential, or protected information is leaked, seen, stolen, or used by someone who is not authorized to do so, it is called a data breach. A data breach might be caused by human error, application flaws, or insufficient security procedures, or it can be the consequence of a targeted attack. The visualization of the total records breached during 2014 to 2018 as given by the Gemelto survey is represented in Figure 1. The census only included records broken in the first half of 2018.

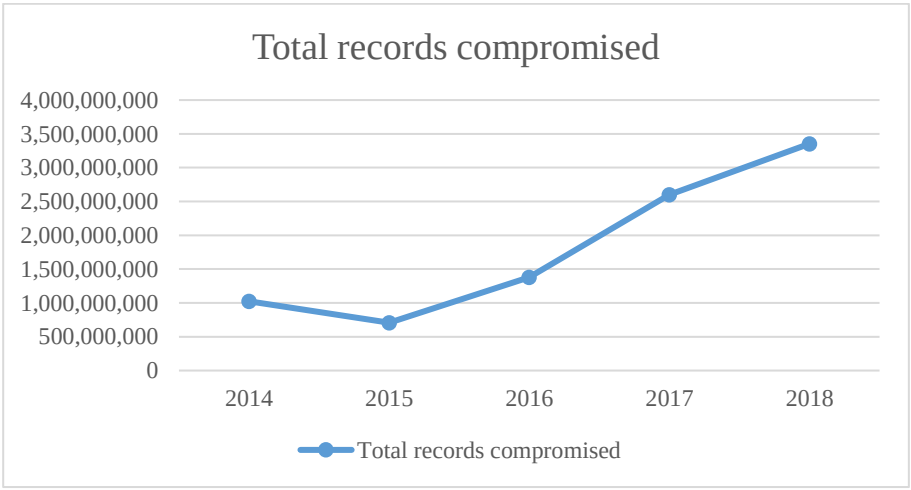


Fig. 1:Total records compromised during the year 2014-2018.

Despite the presence of powerful cryptographic technology, the number of data breaches is increasing every year. Technological advancements have also provided a forum for cybercrime. This clearly demonstrates how data security concerns affect every industry. Thus, it's very vital to assure the safety of most of these indispensable records. Cryptography is a widely used method for ensuring information security. The essential goal of cryptography is to transmit information from the sender to the receiver in the steadiest way possible, so that an attacker is not able to extract the unique information content (Kaur et al., 2021; Zaki et al.). Hackers and security providers compete to outsmart one another in cyber security, which is a fast-paced

industry. New threats and creative strategies to counteract them appear on a regular basis (kilincer et al., 2021; Wiafe et al., 2020).

DNA cryptography is one of the most exciting emerging disciplines in cryptography that arose from the study of DNA computing. The use of DNA computing in cryptography brings up new possibilities. DNA cryptography enhances traditional cryptography with the advantages of molecular biology. The development of DNA cryptography represents a big step forward in cryptography. In fact, the huge parallelism and incredible information density of the DNA molecule are unique qualities that can be employed for cryptography (Gehani et al., 2004). The nucleotide bases have the ability to form self-assembly structures with superior computational capabilities. Numerous DNA computer methods are currently used in encryption, cryptanalysis, key generation, and steganography, making DNA as an extremely powerful tool in terms of cryptography (Elmogy 2018; Popovici 2010). Several algorithms have been developed using the principles of DNA cryptography to protect text as well as image files (Akkasaligar et al., 2020; Roy et al., 2011; Yunpeng et al., 2011; Zhang et al., 2017; Basu et al., 2019).

DNA cryptography combined with finite automata theory uses randomly generated mealy machine to code the DNA sequences to make the ciphertext more secure (Pavithran et al., 2021). Dynamic techniques are used in DNA cryptography. Randomly, 256 ASCII characters are allocated to DNA base sequences. Positions of DNA base sequences are rearranged iteratively to achieve dynamism and encrypted them using an asymmetric cryptosystem and finally ciphertext chunks were merged through dynamic DNA encoding (Biswas et al., 2019). DNA techniques are augmented with mathematical cryptosystem to enhance the security. This method creates the private key from the blood analysis result and applies mathematical formulas for encryption and decoding. The substitution method was used in DNA cryptography, which is a classic cryptographic technique (Devi et al., 2016). To reduce time complexity, a new DNA cryptography system is suggested that uses Symmetric Key Exchange, an OTP mechanism, and DNA hybridization. The encryption mechanism is based on the XOR operation with the OTP DNA sequence. A secure key generation mechanism is presented by Symmetric Key Exchange (Paul et al., 2016). DNA cryptography is combined with one time pad and Caesar cipher. The resulting ciphertext passes most of the NIST statistical test cases (Samwal et al., 2021). Text Encryption Based on DNA Cryptography, RNA, and Amino Acid converts the plain text into ASCII, DNA codons, RNA sequences and finally amino acids. This method maps a DNA codon to multiple amino acid values, which leads to an ambiguity while decrypting (Rashid 2021).

The creation of unbreakable algorithms has been facilitated by the concept of DNA computing in the field of DNA cryptography (Chen 2003). The DNA principle has been used to construct a number of symmetric key encryption, asymmetric key encryption, and steganography methods. Existing DNA based symmetric and

asymmetric encryption algorithms have been studied and the experimental results of this study showed that DNA symmetric algorithms work best in terms of size and time (Hammad et al., 2020). Hence, the motivation is to build a secure data storage system that achieves data confidentiality while maintaining storage efficiency using symmetric DNA cryptography concepts.

This article is organized as follows. Section 2 gives an overview of state-of-the-art DNA cryptography, Section 3 describes the proposed algorithm, Section 4 gives the performance analysis of the proposed method and the last section concludes the work.

2. Related Work

DNA cryptography: a novel paradigm for secure routing in Mobile Ad hoc Networks (MANETs) (Verma et al., 2008), is basically a pseudo DNA cryptography approach, which is based on molecular biology's central dogma principle. The approach duplicates the central dogma's transcription and translation process, as well as some artificial elements to make the cypher texts difficult to crack. Theoretical studies demonstrate that this strategy is effective against brute force attacks. The plaintext is not particularly complete in terms of the confusion and the length of the ciphertext is more than the length of the plain text. The confusion property of plain text is not adequate. Encryption is not reversible since numerous RNA codons map to a single amino acid.

An improved Symmetric key cryptography with DNA based strong cipher (Roy et al., 2011) is a symmetric key cryptography method. It uses a level1 key to encode the plaintext and to generate a primary ciphertext. The primary ciphertext is divided into three uneven parts. Additional parameters like primer code, file type code, integrity code, and authentication code are added in between parts of the ciphertext to obtain the final ciphertext. This method is capable of encrypting large files as well. The ciphertext size has almost doubled, which is the major concern of this method.

Secure Data Communication and Cryptography Based on DNA Based Message Encoding (Majumder et al., 2014) is a combination of conventional cryptography and DNA cryptography. It is a block cipher with a 256 bit key. The encryption process includes four rounds of coding where each round uses the cipher block chaining coding method. This method can be used to encrypt text, audio, video, or any type of file. The size of the ciphertext is increased with this approach.

Extending Feistel structure to DNA Cryptography (Kaundal et al., 2015), is a symmetric encryption method that employs DNA cryptography as well as feistel-inspired structure. This approach uses a key length equal to the 5*length of binary plain text. On the basis of key length, a pseudo random generator will generate the random DNA sequence as a key. In the encryption process, plain text is converted into ASCII and then to binary. This binary is reordered by passing it in to a FEISTEL

inspired structure. To apply confusion, reordered binary is scanned from left to right and the key is from right to left. If 1 occurs in binary text, a 5-mer oligonucleotide is picked up from the key and placed in a ciphertext. If 0 occurs, then only a 5-mer nucleotide is left from the key sequence. This process continues till the end of binary text to obtain the final ciphertext. The reverse process is carried out at the receiver. In this method, plain text messages are converted into DNA forms of ciphertext. It has increased security due to dynamic key selection using a one-time pad. The shortcomings of this method are the comparatively high time requirements for encryption and decryption and increased ciphertext size. The comparatively long encryption and decryption times, as well as the higher ciphertext size, are disadvantages of this approach.

Secure Data communication and Cryptography based on DNA based Message Encoding (Javheri et al., 2015) uses a secure symmetric key generation technique to generate a primary cipher, which is subsequently turned into a final cipher by using DNA sequences. This method is capable of encrypting large files like image or video. The drawback of the method is increased ciphertext size.

Deoxyribonucleic Acid (DNA) for a Shared Secret Key Cryptosystem with Diffie Hellman Key sharing technique [29] makes use of a shared secret key. The secret key is shared using the Diffie-Hellman key sharing mechanism. Plain text is converted into DNA using a DNA code set table. The forward and end primers are attached to this intermediate DNA sequence. The entire string is converted back to binary and it will undergo a DNA hybridization process using a shared secret key which is in DNA form. The resultant ciphertext is transmitted to the receiver, who does the reverse process using the same shared secret key. In this method, plaintext is converted into an ATGC DNA nucleotide sequence. This method is reliable and scalable, with robust performance. But it has the following shortcomings: The possibility of a man in the middle attack while using the Diffie-Hellman approach to share the secret key. The usage of static keys and static tables also gives the opponent more chances. Manipulating DNA sequences takes more time compared to the traditional method. Limited to encrypt only capital letters, numbers, and punctuation marks as the DNA code set contains mapping only for these characters.

An innovative DNA cryptography technique for secure data transmission [30] is based on symmetric key exchange, XOR and matrix operations. Random OTP DNA sequences are generated randomly by using a symmetric key exchange technique. Randomly generated OTP DNA sequences are used only once for the encryption and decryption process. It finds the occurrences of 1s in sequences, gets their positions, and obtains the corresponding DNA sequence from the OTP DNA sequence. The algorithm is compatible with being used on mobile devices. However, the sender has to send the key, random DNA sequence, and OTP DNA sequence along with the ciphertext. This method has a significant drawback in that the encrypted text is approximately 16 times larger than the plain text.

BDNA-A DNA inspired symmetric key cryptographic technique to secure cloud computing (Sohal et al., 2018) adopts client-side encryption. i.e., data is encrypted at the client side before being sent to the cloud. It requires two tables; a random number and a 14-bit key for encryption. Table 1 contains the 7-bit binary code for 96 ASCII characters, and this table changes dynamically based on the random number. A 7-bit key is generated from the 14-bit key based on the initial bit. This 7-bit key is used for encryption. Table 2 is used for converting 7-bit binary to final ciphertext. Experimental results show that this method performs better when compared with Blowfish, DES, and AES in terms of encryption time, ciphertext size, and throughput. But still, the ciphertext is slightly larger than the plaintext.

Enhanced DNA and ElGamal cryptosystem for secure data storage and retrieval in cloud (Thangavel et al., 2018) is a hybrid cryptosystem. It uses a DNA symmetric cryptosystem to encrypt the data and an asymmetric cryptosystem to authenticate the user based on the Enhanced ElGamal cryptosystem. The security of the system depends on randomness and the discrete logarithm problem. The dynamic coding table and introns enhance the security of the system there by reducing the chances of cryptanalytic attacks. The system is resistant to brute force attack as well. From the experimental results, it can be seen that there is a drastic hike in the encryption and decryption times as the input size increases. Also, the size of the ciphertext is double that of the plaintext.

An improved DNA Based Security Model using Reduced Ciphertext Technique (Gupta et al., 2019) uses DNA-based multi-layer encryption with a 128-bit key. It also includes a round key selection technique, a random series of DNA-based coding and modified DNA-based coding, followed by a unique method of substitutions. This technique increases the size of the ciphertext by 33%.

Text Encryption Based on DNA Cryptography, RNA, and Amino Acid [22], encompasses six steps to convert the plain text to ciphertext. Encryption steps are converting plain text to ASCII, ASCII to binary, binary to DNA, complement DNA, DNA to RNA, and finally from RNA to amino acid. The main drawback of this method is ambiguity while decrypting. There is a 1: n mapping between the amino acid codes and RNA codons. Also, the size of the ciphertext is increased in this method.

A new data security algorithm for the cloud computing based on genetics techniques and logical-mathematical functions (Thabit et al., 2021) provides two-layer security. First layer security is obtained through Shannon's theory of diffusion and confusion by the use of logical operations such as XOR, XNOR, and shifting. The second layer of security is based on the principles of the central dogma of molecular biology. It translates binary to DNA bases, converts DNA to mRNA using the transcription process, and mRNA to protein through the translation process. This

approach is extremely secure, but the power consumption is high and increases the size of the ciphertext.

A symmetric DNA encryption process with a biotechnical hardware (Satir et al., 2022) integrates DNA coding and DNA XOR with a feistel structure to increase the confusion property using traditional S boxes. A novel F function has been developed by combing DNA coding and DNA XOR. The implementation of the proposed method has been verified by vitro experiments. The proposed method is resilient against brute force attack. Like all other methods, this also has the problem of increased ciphertext size.

The methodology followed, merits, and demerits of the previous DNA encryption systems are summarized in Table 1.

Table 1: DNA cryptographic methods.

Title	Methodology	Merits	Demerits
DNA cryptography: a novel paradigm for secure routing in Mobile Ad hoc Networks (MANETs) (Verma et al., 2008)	Central dogma of molecular biology	Theoretically secure	Encryption is not reversible Length of ciphertext is increased
An improved Symmetric key cryptography with DNA based strong cipher (Roy et al., 2011)	Primer and intron insertion	Capable to encrypt large files as well	Ciphertext size is almost doubled
Secure DataCommunication and Cryptography Based on DNA Based Message Encoding (Majumder et al., 2014)	DNA XOR	Applicable to any file type	Increased ciphertext size
Extending Feistel structure to DNA Cryptography (kaundal et al., 2015)	One time pad	Dynamic key improves the security	Increased file size
Secure DataCommunication and Cryptography based on DNA based Message Encoding (Javheri et al., 2015)	OTP and DNA digital coding	Able to encrypt any type of file	Increased ciphertext size
Deoxyribonucleic Acid (DNA) for a Shared Secret Key Cryptosystem with Diffie Hellman Key sharing technique (Aieh et al., 2015)	DNA coding table and DNA hybridization	Simple method and is scalable	Increased ciphertext size Encryption limited to capital letters, numbers and punctuation marks

An innovative DNA cryptography technique for secure data transmission (Sanchita et al., 2016)	DNA OTP	Compatible for mobile devices	Limited to encrypt small files Ciphertext is 16 times larger than the plaintext
BDNA-A DNA inspired symmetric key cryptographic technique to secure cloud computing (Sohal et al., 2018)	Dynamic encoding	High throughput	Ciphertext size is larger than the plaintext
Enhanced DNA and ElGamal cryptosystem for secure data storage and retrieval in cloud (Thangavel et al., 2018)	DNA coding, Transcription and Translation process	Resistance to brute force attack	Plainttext to ciphertext ratio is 1:2
An improved DNA Based Security Model using Reduced Ciphertext Technique (Gupta et al., 2019)	128 bit block cipher	Reduced decryption time for large data set	33% increase in the ciphertext size
Text Encryption Based on DNA Cryptography, RNA, and Amino Acid (Rashid 2021)	Central dogma of molecular biology	Simple multilayer security	Ambiguity while decrypting Increased ciphertext size
A new data security algorithm for the cloud computing based on genetics techniques and logical-mathematical functions (Thabit et al., 2021)	Feistel structure and Central dogma of molecular biology	Secure against brute force attack and cryptanalytic attacks	High power consumption and increased ciphertext size
A symmetric DNA encryption process with a biotechnical hardware (Satir et al., 2022)	DNA synthesis	Robust and Dynamic	High cost

A number of DNA cryptographic algorithms have been proposed in the literature to improve the encryption process over existing methods. The larger ciphertext size is a typical problem experienced by almost all methods. The cost of storing and transmitting rises as the size of the ciphertext grows. As a result, most businesses will either use partial encryption or leave data exposed, giving cyber criminals a foothold. This means that shrinking the size of a ciphertext is critical [36]. This paper presents a storage-optimized secure DNA-based crypto system. The experimental results reveal that the size of the encrypted text is smaller than plaintext while maintaining security.

3. Methodology

The suggested encryption system is a symmetric stream cipher. It has two stages: encryption and decryption, as shown in Figure 2.

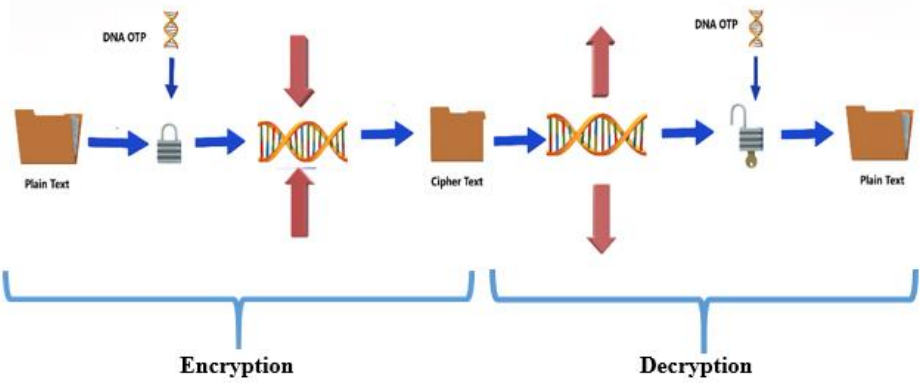


Fig. 2: Graphical representation of the proposed system.

The proposed system adopts the DNA OTP methodology and a frequency-based compression method. A DNA key is selected from the publicly available database. During the encryption process, the text form of the input file is converted into intermediate DNA ciphertext and then into a binary representation of the ciphertext. Encryption steps are performed in reverse order to decrypt the input file. The detailed encryption and decryption processes are discussed below.

Encryption system

The encryption method comprises two major phases. The first phase translates plain text into DNA codons, and the second phase converts DNA codons into variable length bit patterns based on their frequency. The ASCII values of the characters are used to transform plaintext to binary, which is then stored in the variable M. The DNA key is selected from the GenBank dynamically and converted into binary using the 2 bit digit code 00, 01, 10, and 11 for A, C, G, and T respectively. An XOR operation is performed between the plaintext M and the DNA key. This intermediate result is converted into DNA sequences. The number of occurrences of A, T, G, and C are calculated from this DNA form of ciphertext. The second phase includes the construction of an optimal prefix code for DNA nucleotide bases using prefix coding. It removes the two nucleotides with the lowest frequencies from the set and creates a subtree with these two characters as leaves. The root of this subtree is assigned a frequency that is a summation of the frequencies of these two characters. The root of the subtree along with frequency is added to the nucleotide set. This procedure is repeated until only one symbol is left in the set. In order to obtain the prefix code, the left edge is labelled 0; the right edge is labelled 1. The prefix code for the nucleotide

base is "path from root to leaf." The final ciphertext is obtained by encoding A, T, G, and C as per their prefix code words. The encryption algorithm is given below.

Algorithm: DNA Encryption E (Input_file, DNAKey)

$M \leftarrow$ Convert input_file to binary

$Key \leftarrow$ Binary form of DNAKey

$l \leftarrow length(M)$

$k \leftarrow length(Key)$

$Repeat\ the\ following\ for\ i = 0\ to\ l$:

$Ci \leftarrow Mi \oplus Keyi$

$Z \leftarrow []$

a$\leftarrow$t$\leftarrow$g$\leftarrow$c$\leftarrow 0$

$Repeat\ the\ following\ for\ i = 0\ to\ l$:

$if\ Ci == 0\ and\ Ci + 1 == 0\ then$

$Z.append('A')$

$a \leftarrow a + 1$

$else\ if\ Ci == 0\ and\ Ci + 1 == 1\ then$

$Z.append('C')$

$c \leftarrow c + 1$

$else\ if\ Ci == 1\ and\ Ci + 1 == 0\ then$

$Z.append('G')$

$g \leftarrow g + 1$

$else$

$Z.append('T')$

$t \leftarrow t + 1$

We have an alphabet $U \leftarrow \{A|a, T|t, G|g, C|c\}$ where a, t, g, c are the frequencies of A, T, G, C respectively.

Repeat the following until there is only one symbol left in the alphabet:

$x \leftarrow newnode()$

$left(x) \leftarrow Extract\_Min\_freq(U)$

$right(x) \leftarrow Extract\_Min\_freq(U)$

$f(x) \leftarrow f\big(left(x)\big) + f(right(x))$

$U.insert(x)$

$root \leftarrow Extract\_Min\_freq(U)$

Assign code 0 to the left edge and 1 to the right edge by traversing the tree from $root$

Cipher_text=Encoded value of A, T, G, C with their optimal code words.

An illustration of the encryption process for the plaintext "*Have a great day*" is given in Table 2. It can be observed that the English form of plaintext undergoes several transformations and is finally converted into a bit array pattern based on the prefix code of the nucleotide bases.

Table 2: Sample encryption process.

step	*Operation*	
1	*Plain Text*	*Have a great day*
2	*Binary Value* (M)	*01001000011000010111011001100101001000000110000100100000011001110111001001100101011000010111010000100000011001000110000101111001*
3	*DNA OTP* $(DNAKey)$	*ATCGACCGGATCACACAACAATCGACCGGATCACACAACAATCGACCGGATCACACAACAATCG*
4	*Binary form of OTP* (Key)	*00110110000101101000110100010001000001000011011000010110100011010001000100000100001101100001011010001101000100010000010000110110*
5	$Ci \leftarrow Mi \oplus Ke$	*01111110011101111111101101110100001001000101011100110111010100110001101100001010101110110001010101101110101011001010100111*
6	*Convert C to DNA with frequency*	*CTTGCTCTTTGTCTCAAGCACCCTATCGTGGGCGATCGACCCCTCGAGGGTCCTCCCGCCCATT {'C': 25, 'T': 17, 'G': 14, 'A': 8}*
7	*Phse2 Operation*	*[[25, ['C', '']], [17, ['T', '']], [14, ['G', '']], [8, ['A', '']]] [[8, ['A', '']], [17, ['T', '']], [14, ['G', '']], [25, ['C', '']]] right = [8, ['A', '']] left = [14, ['G', '']] right = [17, ['T', '']] left = [22, ['A', '0'], ['G', '1']] right = [25, ['C', '']] left = [39, ['T', '0'], ['A', '10'], ['G', '11']] [['C', '0'], ['T', '10'], ['A', '110'], ['G', '111']]*
8	*Prefix code for nucleotide bases*	*{'C': bitarray('0'), 'T': bitarray('10'), 'A': bitarray('110'), 'G': bitarray('111')}*
9	*Final Ciphertext*	*bitarray('0101011101001010101111001001101101110110000101101010011110111111111100010000111000110101011111111111000100001110001101010')*

Decryption system

Decryption is the reverse process of encryption. The frequency tree is required to decode the encoded data. The decryption method finds the character corresponding to the current bits by iterating the tree. If the current bit is 0, the algorithm moves to

the tree's left node. If the bit is 1, it moves to the tree's right node. If it is a leaf node, the algorithm replaces the bit string with the corresponding nucleotide base. This intermediate result is converted to binary by assigning the codes 00, 01, 10, and 11 to A, C, G, and T, respectively. An XOR operation is performed between this binary code and the DNA key. The plain text is generated by converting the result of the XOR operation to ASCII letters.

Algorithm: DNA Decryption D (Cipher_text, root, Key)

$x \leftarrow root$

$Z \leftarrow []$

$Repeat\ the\ following\ for\ i = 0\ to\ length(Cipher\_text)$

$if\ Cipher\_text[i] == '0'\ then$

$x \leftarrow x\text{-}\rangle left$

$else$

$x \leftarrow x\text{-}\rangle right$

$if\ x\text{-}\rangle left == NULL\ and\ \ x\text{-}\rangle right == NULL\ then$

$x \leftarrow x\text{-}\rangle left$

$Z.append(x\text{-}\rangle data)$

$x \leftarrow roo$t

$Repeat\ the\ \ following\ for\ i = 0\ to\ length(Z)$:

$if\ Zi == 'A'$

C. $append(0)$

C. $append(0)$

$if\ Zi == 'C'$

C. $append(0)$

C. $append(1)$

$if\ Zi == 'G'$

C. $append(1)$

C. $append(0)$

else

C. $append(1)$

C. $append(1)$

$M \leftarrow []$

$Repeat\ the\ \ following\ for\ i = 0\ to\ length(C)$:

$Mi \leftarrow Ci \oplus Keyi$

$Plaintext \leftarrow Covert\ M\ to\ ASCII$

An example decryption process is shown in Table 3. Since all phases in the encryption process are reversible, decryption is carried out in the reverse order of encryption.

Table 3: Sample decryption process.

Step	*Operation*	
1	*Ciphertext*	*bitarray('0101011101001010101111001001101101110110000010110100111101111111110111101000111100000100111110111111111110001000011100011010101')*
2	*DNA OTP* (Key)	*ATCGACCGGATCACACAACAATCGACCGGATCACACAACAATCGACCGGATCACACAACAATCG*
3	*Corresponding Nucleotide*	*CTTGCTCTTTGTCTCAAGCACCCTATCGTGGGCGATCGACCCCTCGAGGGTCCTCCCGCCCATT*
4	*Binary* Ci	*0111111001110111111110110111010000100100010101110110110111010100110001101100001010101110110001010101110101001011001101100001010101110110001010*
5	$Mi \leftarrow Ci \oplus Ke$	*0100100001100001011101100110010100100000011000010010000001100111011100100110010101100001011101000010000001100100011000010111100*
6	*Plain text*	*Have a great day*

Mathematical Proof:

Hypothesis: The resulting algorithm produces an optimal prefix code for any alphabet U with frequencies f.

Proof: Let us consider an alphabet $\{U_i\} i = 2,3, \ldots, |n|$ with probabilities P_i $=\{ p_1 , p_2 ,\ldots, p_{i-1} , p_I, p_{i+1,}+\ldots+ p_{|n|}\}$ and C_i be the prefix code on the set of source symbols U_i with probabilities P_i

We prove the hypothesis by mathematical induction on the given alphabet.

For the base case n= 2, then there are only two characters in the alphabet and it maps one to each. Hence, the encoding is optimal. Hence the encoding is optimal.

Let us assume that the hypothesis is true for all the alphabets with n-1 characters.

We will show that the hypothesis is true for all the alphabets with n characters as well.

The prefix code for n-1 characters is formed by taking the common prefix of the longest code words in the alphabet U and assigning it to a symbol with expected length $p_{i-1}+p_i$. Let l_k signifies the length of the code word for symbol k in C_i and l'_k denote the length of the symbol k in C_{i-1}. Then

$$L(C_i) = \sum_{k=1}^{i-2} p_k\, l_k + p_{i-1}\, l_{i-1} + p_i\, l_I$$

$$= \sum_{k=1}^{i-2} p_k\, l'_k + (p_{i-1}+p_i)\, l'_{i-1} + (p_{i-1}+p_i) \qquad (1)$$

$$L(C_{i-1})$$

Assume, on the other hand, that Code C_i is not optimal. Let $\hat{C}_i$ be optimal. We get $\hat{C}_{i-1}$ by combining the two least occurrence symbols with same length. But then

$$L(\hat{C}_i) = L(\hat{C}_{i-1}) + (p_{i-1}+p_i) \geq L(C_{i-1}) + (p_{i-1}+p_i) = L(C_i) \quad (2)$$

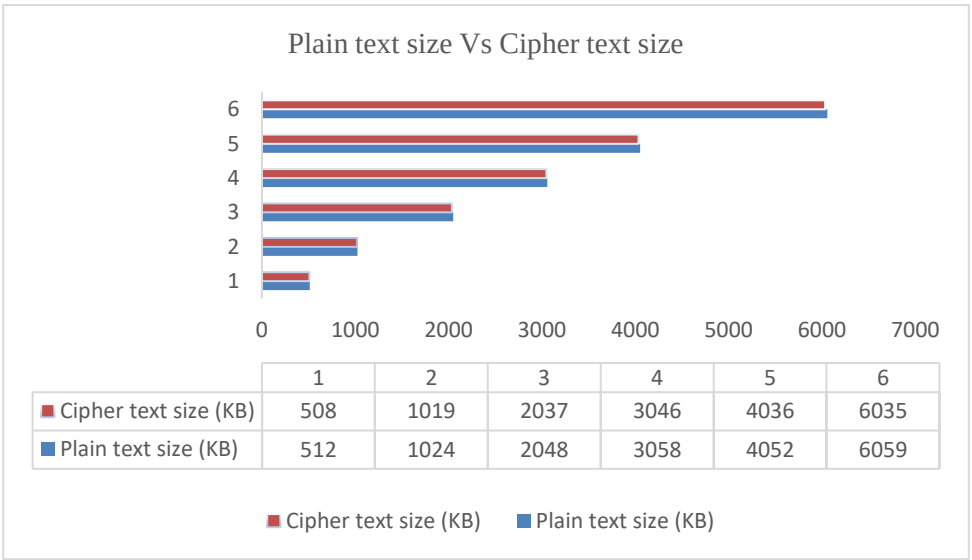
Where the inequality holds since C_{i-1} is optimal. Hence prefix code C_i had to be optimal.

4. Results and Discussion

The suggested system was simulated using Python 3.7 on a system with a 2.4 GHz processor and 4 GB of RAM. The performance of the system is analysed based on ciphertext size, time complexity, execution time, throughput, one-time pad, avalanche effect and randomness using NIST test suites.

Ciphertext size:

Ciphertext size determines the cost of storage and transmission. The ciphertext size of the proposed method is measured by taking 25 samples for each of the 512KB, 1MB, 2MB, 3MB, 4MB, 5MB, and 6MB file size categories. Finally, the average ciphertext size is taken and is presented in Figure 3. When compared to other DNA cryptography methods and traditional methods, the proposed method improves storage performance and transmission costs. In previous techniques, the encrypted text was always larger than or equal to the plain text. The experimental results of the proposed technique show that the size of the ciphertext is always less than the plain text.



	1	2	3	4	5	6
Cipher text size (KB)	508	1019	2037	3046	4036	6035
Plain text size (KB)	512	1024	2048	3058	4052	6059

Fig. 3: Plain text size Vs Ciphertext size for the proposed method.

The comparison of ciphertext size with other DNA cryptography methods (Gupta et al., 2019; Majumder et al., 2014) and the traditional blowfish algorithm is shown in Figure 4 and Table 4. The results show that the suggested method improves memory utilisation and lowers transmission costs in comparison to alternative encryption approaches.

Table 4 Ciphertext size comparison of proposed method Vs. other methods

Plain text size(MB)	Ciphertext size (MB)			
	Majumder	Gupta	Blowfish	Proposed
1	2	1.33	1.80	0.99
2	4	2.67	3.70	1.98
3	6	4	5.58	2.97
5	10	6.65	9.23	4.9
10	20	13.3	18.43	9.85

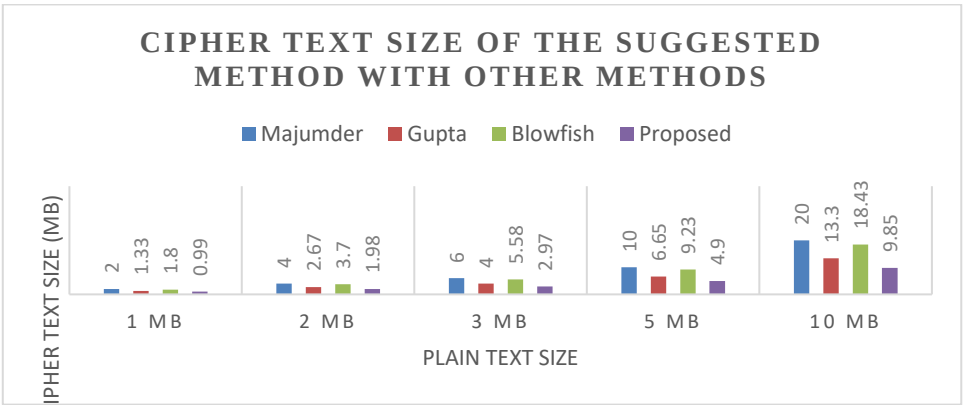


Fig. 4: Comparison of Ciphertext size of the proposed method with other methods.

The ciphertext size to plaintext size proportion has been analysed for the proposed method as well as previous DNA cryptographic approaches and other traditional cryptographic methods. The size of the ciphertext for AES, DES, and Blowfish is referred to in[31]. The findings of the study are given in Table 5 and visualized in Figure 5. It is clear that the proposed method optimizes the storage and transmission costs compared to the other methods.

Table 5: Ratio of ciphertext size to plain text size.

Reference	Units	Plain text size	Ciphertext size	Ciphertext size/ Plain text size
(Roy et al., 2011)	KB	1126.4	2252.8	2.00
		2304	4608	2.00
		5990.4	11980.8	2.00
		14950	29900.8	2.00
		35840	71680	2.00

		57856	115712	2.00
(Kaundal et al., 2015)	Byte	10	160	16.00
		20	324	16.20
		40	635	15.88
		80	1281	16.01
(Sanchita et al., 2016)	Byte	40	642	16.05
		80	1289	16.11
		100	1560	15.60
		500	8016	16.03
(Devi et al., 2016)	KB	147	588	4.00
		384	1536	4.00
		768	3072	4.00
		3105	12420	4.00
		4028	16112	4.00
(Rashid 2021)	Characters	1001	1335	1.33
		1327	1769	1.33
		3000	4000	1.33
		5115	6820	1.33
		10229	13639	1.33
		20442	27256	1.33
AES	KB	5	6.86	1.37
		10	13.7	1.37
		15	20.55	1.37
		20	27.39	1.37
		25	34.25	1.37
		30	41.09	1.37
DES	KB	5	6.86	1.37
		10	13.7	1.37
		15	20.55	1.37
		20	27.39	1.37
		25	34.25	1.37
		30	41.09	1.37
Blowfish	KB	5	6.76	1.35
		10	13.52	1.35
		15	20.27	1.35
		20	27.03	1.35
		25	33.78	1.35
		30	40.5	1.35
(Sohal et al., 2018)	KB	5	5.83	1.17
		10	11.66	1.17
		15	17.5	1.17
		20	23.34	1.17
		25	29.17	1.17
		30	35	1.17
(Satir et al., 2022)	Bits	40000	40320	1.01
(Thabit et al., 2021)	KB	5	15	3.00
		10	30	3.00

		20	60	3.00
		30	90	3.00
		40	120	3.00
(Majumder et al., 2014)	MB	1	2	2.00
		2	4	2.00
		3	6	2.00
		5	10	2.00
		10	20	2.00
(Thangavel et al., 2018)	KB	4	8.0	2.00
		8	16.1	2.01
		16	32.2	2.01
		32	64.6	2.02
		64	129.0	2.02
(Gupta et al., 2019)	MB	1	1.33	1.33
		2	2.67	1.34
		3	4	1.33
		5	6.65	1.33
		10	13.3	1.33
Proposed Method	MB	1	0.99	0.99
		2	1.98	0.99
		3	2.97	0.99
		5	4.9	0.98
		10	9.85	0.99

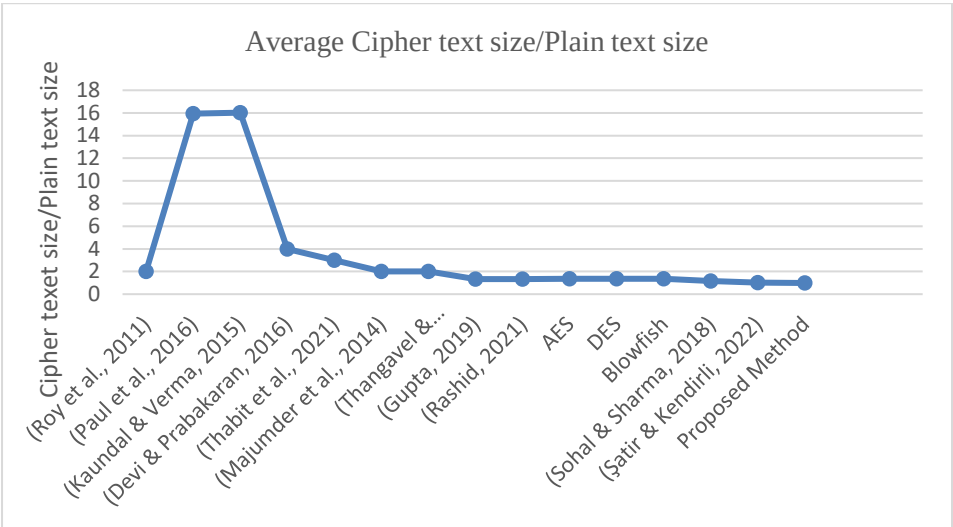


Fig. 5: Average ratio of ciphertext size to plain text size.

Time Complexity:

An algorithm's complexity analysis is important because it discloses its overall effectiveness for practical uses. The processing complexity of the algorithm was

evaluated in this work using complexity theory methodologies. The obtained conclusions were validated by the experimental results.

The runtime of an algorithm is defined as the sum of all operations. The time required to convert plaintext to ciphertext is referred to as encryption time. The time complexity of an encryption algorithm is the sum of the time required at phase 1 and phase 2. Phase 1 depends on the size of the input n and phase 2 runs in constant time. It can be represented as follows:

$$T_{(Encryption)} = T_{(Phase\ 1)} + T_{(Phase\ 2)} \quad (3)$$

$$= O(n) + O(3) \quad (4)$$

$$= O(n) \quad (5)$$

Similarly decryption algorithm also works with a complexity of $O(n)$.

Execution time:

Execution time corresponds to the time taken to convert a plaintext message to ciphertext and vice versa. Execution time is inversely proportional to the performance of the system. The encryption and decryption time required for the proposed method are calculated for different file sizes and compared with the other state-of-art methods (Pavithran et al., 2020; Kaundal et al., 2014; Sanchita et al., 2016). The encryption time and decryption times are measured by taking data sets of different sizes. The average encryption and decryption times are given in Table 6 and Table 7 respectively. It can be seen that the encryption and decryption time of the proposed method is far less compared to other methods. There is a sudden hike in the execution time for other methods as the input size increases. The cost of decryption is another important factor that every user takes into account. For the methods (Kaundal et al., 2014; Sanchita et al., 2016), decryption time is more than the encryption time, whereas the proposed method's decryption time is less than or equal to the encryption time. The visual representation of the execution times are shown in Figure 6 and Figure 7. The experimental results validate the estimated complexity of the algorithm.

Table 6: Comparison of the encryption time.

Plain text length(char)	Encryption Time (ms)			
	Kaundal	Paul	Pavithran	Proposed
10	15	18	11.31	2
20	27	28	16.42	4
40	38	42	31.11	3
80	70	84	62.27	5
100	96	112	85.45	5
500	420	486	312.45	8

Table 7: Comparison of the decryption time (ms).

Plain text length(char)	Decryption Time (ms)			
	Kaundal	Paul	Pavithran	Proposed
10	37	42	5.87	2
20	44	57	9.57	2
40	110	123	24.75	3
80	270	310	49.111	3
100	365	415	61.25	3
500	1660	1896	245.75	7

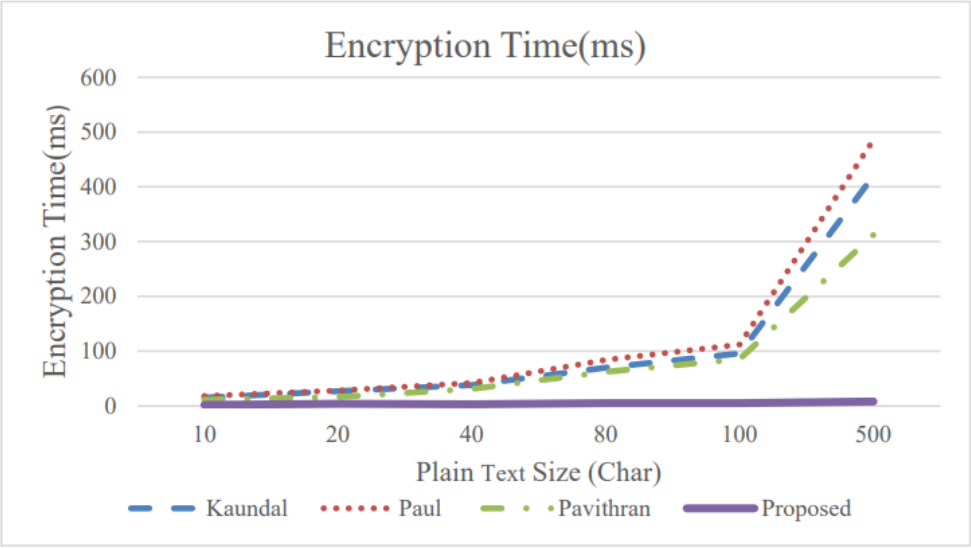


Fig. 6: Encryption time of the proposed method Vs. Other.

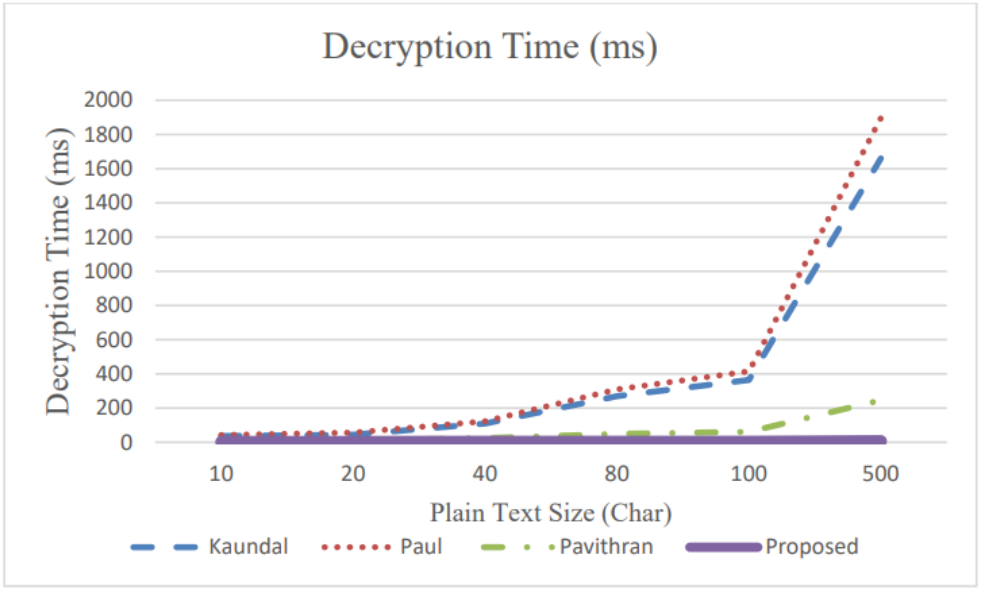


Fig. 7: Decryption time of the proposed method Vs. Other.

Throughput:

Throughput is another factor that indicates the efficiency of a system. The performance of a system is directly related to its throughput, meaning that the higher the throughput, the higher the performance. The execution time and throughput are inversely proportional, with the execution time being lower and the throughput being larger. Throughput is calculated using the following formula:

$$Throughput = \frac{Plain\ text\ size}{Encryption\ Time} \tag{6}$$

The throughput of the suggested method is estimated, and a comparison to comparable state-of-the-art methods (Roy et al., 2011; basu et al., 2019;; Pavithran et al., 2020; Paul et al., 2016; Majumder et al., 2014; Gupta et al., 2019; Thabit et al., 2021) is shown in Figure 8. The results show that the throughput is higher than that of alternative methods. As a result, the proposed approach outperforms other DNA cryptography algorithms.

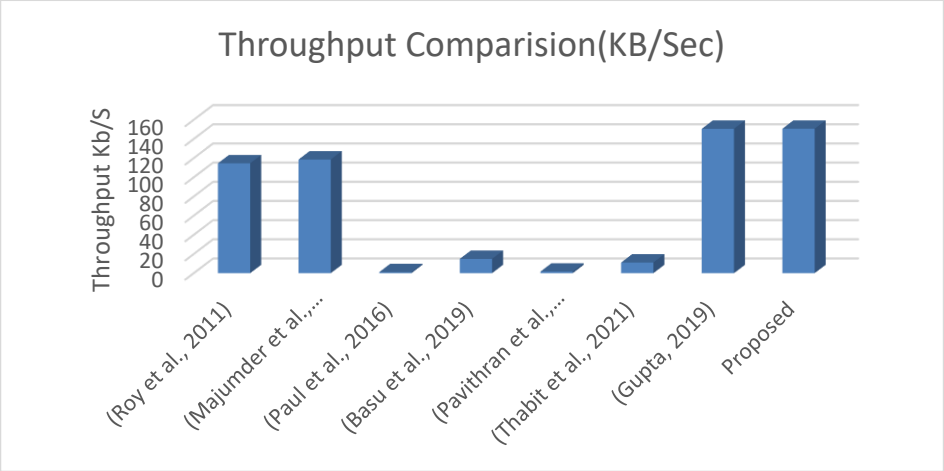


Fig. 8: Throughput of different DNA cryptography systems.

Avalanche effect:

To be considered secure, a cryptographic algorithm must have a high avalanche effect. There must be a radical change in the ciphertext for any minor change in the plaintext or key. We say an algorithm meets strict avalanche conditions if a single bit change in the input causes a 50% change in the output. An avalanche of more than 50% will always be satisfied by a good cypher. The proposed algorithm exhibits an avalanche effect of 63% for a single bit change in ciphertext. The subsequent formula is used to compute the avalanche effect.

$$Avalanche\ effect = \frac{No.of\ flipped\ bits\ in\ the\ cipher\ text}{No.of\ bits\ in\ the\ cipher\ text} \quad (7)$$

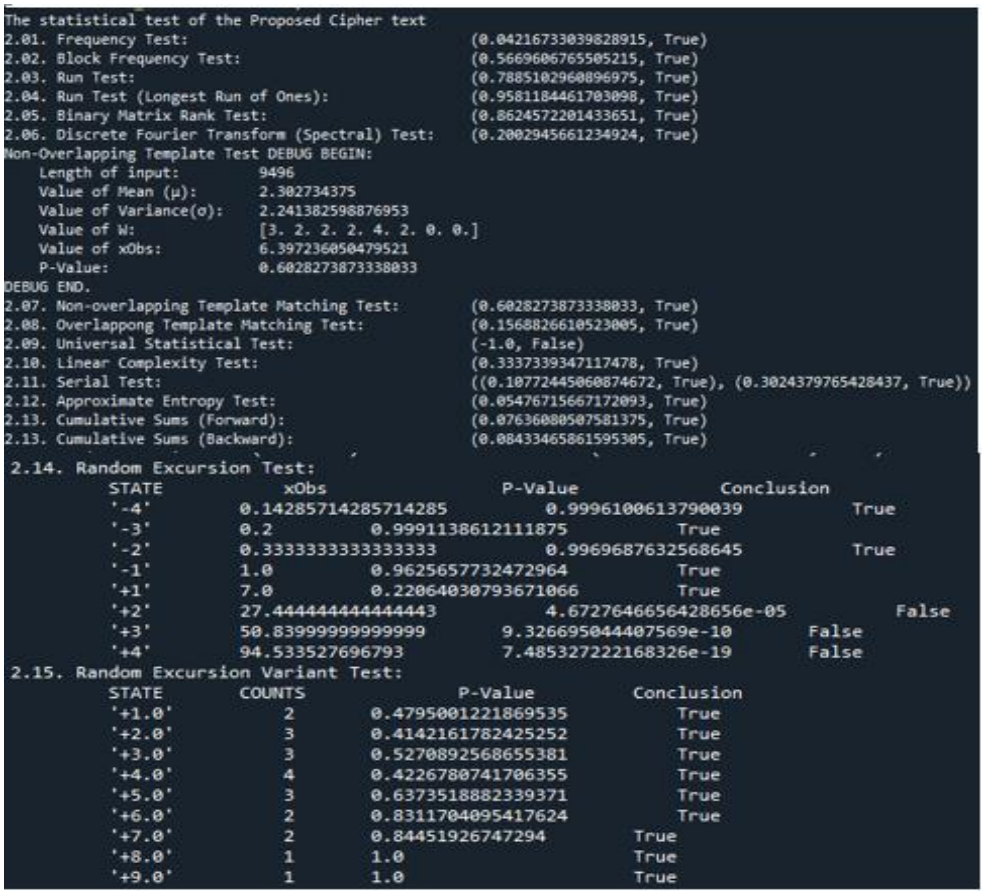
Security Analysis:

The proposed algorithm is based on the one-time pad technique. Theoretically, a one-time algorithm is considered unbreakable (Tantau 2011; Hirabayashi et al., 2009; Babaei 2013) if it possesses the following features: Each OTP sequence should be used only once, Length of the pad must be at least as long as the plain text message and OTP must be random. As an OTP, the proposed method employs a DNA sequence. The number of DNA sequences on the planet is practically limitless. Thus, there will be no need to reuse the DNA string as OTP. There are around 163 million DNA sequences of different sizes available in various public databases, and DNA sequences are random in nature (Chen 2003; Roy). The proposed method chooses a "$4 \times length\ of\ the\ input$" long key as one time pad and produces a random output. As a result, it meets the requirements of the one-time pad. This makes it resilient against cryptanalytic attacks.

NIST Statistical test:

NIST stands for the National Institute of Standards and Technology. It is one of the most important testing suites for determining the randomness of an encrypted

message. This covers 15 statistical tests that look for different sorts of non-randomness in a sequence. The tests look at the unpredictability of data using various bit statistics or bit block statistics. All of the NIST STS (Statistical Test Suite) tests look at the randomness of the entire bit stream. Local non-randomness can also be detected using multiple tests, which break the bit stream into several relatively large parts and compute a bit characteristic for each component. The test statistic is then computed using all of these partial attributes (Rukhin et al., 2010). NIST STS was applied to the resultant ciphertext with a level of significance α=0.01. A sample test result is shown in Figure 9. The test result indicates that the proposed method passes the majority of the NIST test suites. If less than 7 NIST STS tests fail, the data can be considered random at the significance level of 0.01 (Sys et al., 2015). This result demonstrates that the generated cipher texts have good statistical properties: they are unpredictable, random, independent, and uniformly distributed. Hence, the proposed system is highly secure.



```
The statistical test of the Proposed Cipher text
2.01. Frequency Test:                                (0.04216733039828915, True)
2.02. Block Frequency Test:                          (0.5669606765505215, True)
2.03. Run Test:                                      (0.7885102960896975, True)
2.04. Run Test (Longest Run of Ones):                (0.9581184461703098, True)
2.05. Binary Matrix Rank Test:                       (0.8624572201433651, True)
2.06. Discrete Fourier Transform (Spectral) Test:    (0.2002945661234924, True)
Non-Overlapping Template Test DEBUG BEGIN:
    Length of input:        9496
    Value of Mean (μ):      2.302734375
    Value of Variance(σ):   2.241382598876953
    Value of W:             [3. 2. 2. 2. 4. 2. 0. 0.]
    Value of xObs:          6.397236050479521
    P-Value:                0.6028273873338033
DEBUG END.
2.07. Non-overlapping Template Matching Test:        (0.6028273873338033, True)
2.08. Overlappong Template Matching Test:            (0.1568826610523005, True)
2.09. Universal Statistical Test:                    (-1.0, False)
2.10. Linear Complexity Test:                        (0.3337339347117478, True)
2.11. Serial Test:                                   ((0.10772445060874672, True), (0.3024379765428437, True))
2.12. Approximate Entropy Test:                      (0.05476715667172093, True)
2.13. Cumulative Sums (Forward):                     (0.07636080507581375, True)
2.13. Cumulative Sums (Backward):                    (0.08433465861595305, True)
2.14. Random Excursion Test:
        STATE           xObs                P-Value             Conclusion
        '-4'        0.14285714285714285         0.9996100613790039          True
        '-3'        0.2         0.9991138612111875          True
        '-2'        0.3333333333333333          0.9969687632568645          True
        '-1'        1.0         0.9625657732472964          True
        '+1'        7.0         0.22064030793671066         True
        '+2'        27.444444444444443          4.6727646656428656e-05          False
        '+3'        50.83999999999999       9.326695044407569e-10       False
        '+4'        94.533527696793         7.485327222168326e-19       False
2.15. Random Excursion Variant Test:
        STATE       COUNTS              P-Value         Conclusion
        '+1.0'          2       0.4795001221869535          True
        '+2.0'          3       0.4142161782425252          True
        '+3.0'          3       0.5270892568655381          True
        '+4.0'          4       0.4226780741706355          True
        '+5.0'          3       0.6373518882339371          True
        '+6.0'          2       0.8311704095417624          True
        '+7.0'          2       0.84451926747294        True
        '+8.0'          1       1.0                     True
        '+9.0'          1       1.0                     True
```

Fig. 9: Result of NIST STS on ciphertext

5. Conclusion

A secure and optimized data storage system is proposed using DNA cryptography and frequency-based compression algorithms. The suggested approach produces an optimal length ciphertext, according to a mathematical induction proof. An experimental study shows that the intended ciphertext size is always smaller than the plaintext size. The resulting method has a throughput of 150 KB/S and an avalanche effect of 65%, which meets the stringent avalanche requirement. The proposed method is based on a one-time pad that is unbreakable in principle. The proposed technique is both random and secure, according to the NIST statistical test findings. Hence, the performance of the proposed system is better in terms of space and time compared to other DNA cryptography methods. It is a simple yet secure approach. The proposed method can now only encrypt text files, but it can be modified to encrypt other types of data like audio, video, and image files.

Reference

Aieh, A., Sen, A., Dash, S. R., & Dehuri, S. (2015). Deoxyribonucleic acid (DNA) for a shared secret key cryptosystem with Diffie hellman key sharing technique. *Proc. 2015 3rd Int. Conf. Comput. Commun. Control Inf. Technol. C3IT 2015*. DOI:10.1109/C3IT.2015.7060130.

Akkasaligar, P. T. & Biradar, S. (2020). Selective medical image encryption using DNA cryptography. *Inf. Secur. J.*, 29(2), 91–101, DOI:10.1080/19393555.2020.1718248.

Babaei, M. (2013). A novel text and image encryption method based on chaos theory and DNA computing. *Nat. Comput*., 12(1), 101–107. DOI:10.1007/s11047-012-9334-9.

Basu, S., Karuppiah, M., Nasipuri, M., Halder, A. K., & Radhakrishnan, N. (2019). Bio-inspired cryptosystem with DNA cryptography and neural networks. *J. Syst. Archit*., 94, 24–31. DOI:10.1016/j.sysarc.2019.02.005.

Bin, S. & Yongjie, W. (2021). Computer network security under the cloud computing technology environment. *in Journal of Physics: Conference Series*, 1982(1), DOI:10.1088/1742-6596/1982/1/012204.

Biswas, M. R., Alam, K. M. R., Tamura, S., & Morimoto, Y. (2019). A technique for DNA cryptography based on dynamic mechanisms. *J. Inf. Secur. Appl*., 48. DOI:10.1016/j.jisa.2019.102363.

Chen, J. (2003). A DNA-based, biomolecular cryptography design. *IEEE*, 822–825.

Corallo, A., Lazoi, M., & Lezzi, M. (2020). Cybersecurity in the context of industry 4.0: A structured classification of critical assets and business impacts. *Computers in Industry*, 114. Elsevier B.V.,DOI:10.1016/j.compind.2019.103165.

Devi, K. R. & Prabakaran, S. (2016). An enhanced bilateral information security towards a conventional cryptographic system using DNA sequences. 9. DOI:10.17485/ijst/2016/v9i39/102067.

Elmogy, M. M. (2018). DNA-based cryptography: Motivation, progress, challenges, and future. *J. Softw. Eng. Intell. Syst*., 3(1).

Gehani, A., Labean, T., & Reif, J. (2004). DNA-based cryptography. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, 2950, 167–188. DOI:10.1002/9783527678679.dg03229.

Guo Li Dong Hua da Xue and Institute of Electrical and Electronics Engineers, WOCC 2018 : the 27th Wireless and Optical Communication Conference (WOCC 2018) : April 30-May 1, 2018, National Dong Hwa University, Hualien, Taiwan.

Gupta, L. M., Garg, D. H., & Samad, D. A. (2019). An improved DNA based security model using reduced cipher text technique. *I. J. Comput. Netw. Inf. Secur*., 13–20. DOI:10.5815/ijcnis.2019.07.03.

Hammad, B. T., Sagheer, A. M., Ahmed, I. T., & Jamil, N. (2020). A comparative review on symmetric and asymmetric dna-based cryptography. *Bull. Electr. Eng. Informatics*, 9(6), 2484–2491DOI:10.11591/eei.v9i6.2470.

Hirabayashi, M., Kojima, H., & Oiwa, K. (2009). 80 Nucleic Acids Symposium Series, (53).

Javheri, S. & Kulkarni, R. (2014). Secure data communication and cryptography based on DNA based message encoding. *Proc. 2014 IEEE Int. Conf. Adv. Commun. Control Comput. Technol. ICACCCT 2014*, 98(16), 360–363, DOI:10.1109/ICACCCT.2014.7019464.

Kaundal, A. K. & Verma, A. K. (2015). Extending feistel structure to DNA cryptography. *J. Discret. Math. Sci. Cryptogr*., 18(4), 349–362, DOI:10.1080/09720529.2014.995975.

Kaur, D. H. & Kaur, P. A. (2021). Cryptography in cloud computing .*Indian J. Cryptogr. Netw. Secur*, 1(1), 1-2. DOI:10.35940/ijcns.A1402.051121.

Kilincer, I. F., Ertam, F., & Sengur, A. (2021). Machine learning methods for cyber security intrusion detection: Datasets and comparative study. *Comput. Networks*, 188, DOI:10.1016/j.comnet.2021.107840.

Macnish, K. & van der Ham, J. (2020). Ethics in cybersecurity research and practice. *Technol. Soc.*, 63, DOI:10.1016/j.techsoc.2020.101382.

Majumder, A., Majumdar, A., Podder, T., Kar, N., & Sharmas, M. (2014). Secure data communication and cryptography based on DNA based message encoding. *IEEE Int. Conf. Adv. Commun. Control Comput. Technol.*, 978, 360–363.

Paul, S., Anwar, T., & Kumar, A. (2016). An innovative DNA cryptography technique for secure data transmission. *Int. J. Bioinform. Res. Appl.*, 12(3), 238–262. DOI:10.1504/IJBRA.2016.078235.

Pavithran, P., Mathew, S., Namasudra, S., & Lorenz, P. (2021). A novel cryptosystem based on DNA cryptography and randomly generated mealy machine. *Comput. Secur.*, 104, 102160. DOI:10.1016/j.cose.2020.102160.

Popovici, C. (2010). Aspects of DNA cryptography. *Ann. Univ. Craiova, Matheatics Comput. Sci. Ser.*, 37(3), 147–151.

Rashid, O. F. (2021). Text encryption based on DNA cryptography, RNA, and amino acid. *E- Proc. 5th Int. Multi-Conference Artif. Intell. Technol. (MCAIT 2021) Artif. Intell. 4th Ind. Revolut.*, no. 2017, 167–173.

Roy, P. DNA cryptography. 775–777. DOI:10.4018/978-1-5225-0058-2.ch031.

Roy, B., Rakshit, G., Singha, P., Majumder, A., & Datta, D. (2011). An improved symmetric key cryptography with DNA based strong cipher. *2011 Int. Conf. Devices Commun. ICDeCom 2011 - Proc.*, 2–6. DOI:10.1109/ICDECOM.2011.5738553.

Rukhin, A., Soto, J., & Nechvatal, J. (2010). Nistspecialpublication800-22R1a.Pdf.

Samwal Idris, M. A. H. & Mathur, A. (2021). Novel enhanced algorithm based on DNA cryptography to secure data transfer over network. *J. Phys. Conf. Ser*. DOI:10.1088/1742-6596/2040/1/012047.

Sanchita, P., Tausif, A., & Abhishek, K. (2016). An innovative DNA cryptography technique for secure data transmission. 12(3), 238–262.

Şatir, E. & Kendirli, O. (2022). A symmetric DNA encryption process with a biotechnical hardware. *J. King Saud Univ. - Sci.*, 101838DOI:10.1016/j.jksus.2022.101838.

Sohal, M. & Sharma, S. (2018). BDNA-A DNA inspired symmetric key cryptographic technique to secure cloud computing. *J. King Saud Univ. - Comput. Inf. Sci.* DOI:10.1016/j.jksuci.2018.09.024.

Sýs, M., Říha, Z., Matyáš, V., Márton, K., & Suciu, A. (2015). On the interpretation of results from the NIST statistical test suite. *Rom. J. Inf. Sci. Technol.*, 18(1), 18-32.

Tantau, T. (2011). The one-time pad algorithm – The simplest and most secure way to keep secrets. DOI:10.1007/978-3-642-15328-0.

Thangavel, M. & Varalakshmi, P. (2018). Enhanced DNA and elgamal cryptosystem for secure data storage and retrieval in cloud. *Cluster Comput.*, 21(2). DOI:10.1007/s10586-017-1368-4.

Thabit, F., Alhomdy, S., & Jagtap, S. (2021). A new data security algorithm for the cloud computing based on genetics techniques and logical-mathematical functions. *Int. J. Intell. Networks*, 2, 18–33. DOI:10.1016/j.ijin.2021.03.001.

Verma, A. K., Dave, M., & Joshi, R. C. (2008). DNA cryptography: A novel paradigm for secure routing in mobile ad hoc networks (MANETs). *J. Discret. Math. Sci. Cryptogr.*, 11(4), 393–404. DOI:10.1080/09720529.2008.10698192.

Wiafe, I., Koranteng, F. N., Obeng, E. N., Assyne, N., Wiafe, A., & Gulliver, S. R. (2020). Artificial intelligence for cybersecurity: A systematic mapping of literature. *IEEE Access*, 8, 146598–146612. DOI:10.1109/ACCESS.2020.3013145.

Yunpeng, Z., Yu, Z., Zhong, W., & Sinnott, R. O. (2011). Index-based symmetric DNA encryption algorithm. *Proc. - 4th Int. Congr. Image Signal Process. CISP*, 5(2), 2290–2294. DOI:10.1109/CISP.2011.6100690.

Zaki, W. & Al-Humadi, M. (2020). Solid state technology. Cryptography in cloud computing for data security and network security. 63(4), 6965.

Zhang, Y., Liu, X., & Sun, M. (2017). DNA based random key generation and management for OTP encryption. *BioSystems*, 159, 51–63DOI:10.1016/j.biosystems.2017.07.002.

Zhang, Z., Li, C., Gupta, B. B., & Niu, D. (2018). Efficient compressed Ciphertext length scheme using multi-authority CP-ABE for hierarchical attributes. *IEEE Access*, 6(c), 38273–38284DOI:10.1109/ACCESS.2018.2854600.