Improved 2-Level Data Security Approach using DNA Cryptography

I. M. Awwal[1], J. A. Ojekunle[2], O. A. Abisoye[2], G. A. Babalola[2], A. B. Olatunde[2], A. O. Shadrack[2]

[1]Department of Computer Science, College of Education, Minna, Nigeria.
[2]Department of Computer Engineering, Federal University of Technology, Minna, Nigeria.

Keywords: Cryptography, DNA Cryptography, 2-level data security approach, DNA One Time Pad.

Abstract

Cryptography is the process of transforming the meaning of information with the aid of an encryption key. DNA cryptography is a new rapidly evolving technology that uses DNA computing principles: Adenine (A), Guanine (G), Cytosine (C), and Thymine (T) to encrypt or hide the meaning of information such that only the intended recipient can understand. This research aims to improve on a two-level text data encryption system using DNA cryptography. To achieve the research goal, 2-level data security algorithms for encryption, decryption, and key generation were designed using Shift Cipher encryption technique at the first security level and One Time Pad encryption technique at the second security level of the encryption and decryption algorithms. The algorithms were implemented using the PHP programming language. The research findings revealed that the improved 2-level data security approach has better encryption and decryption execution time compared to the 2-level Text Data encryption using DNA cryptography and the encryption algorithms can be used to encrypt and decrypt alphanumeric and special symbol information.

1 Introduction

Cryptography is the science of using mathematics and logic operations to encrypt and decrypt information. It is the ability to transmit information in a mangled form between participants that prevents others from reading it. This allows for versatile storage of confidential information or transmits it through vulnerable networks (such as the internet) so that nobody can read it except the intended user [1].

[[2] describes cryptography as the technique developed by applying mathematics and logic to store and transmit data in coded and protected form so that it can be read and processed only by the intended recipient. The method of securing the meaning of an information or data is known as encryption. Encryption is the process of generating ciphertext from a plaintext. Cryptography protects data from unauthorized third-party applications and other adversaries. The science or technique for decrypting a ciphertext is known as cryptanalysis. The cryptography basic model is depicted in Figure 1.

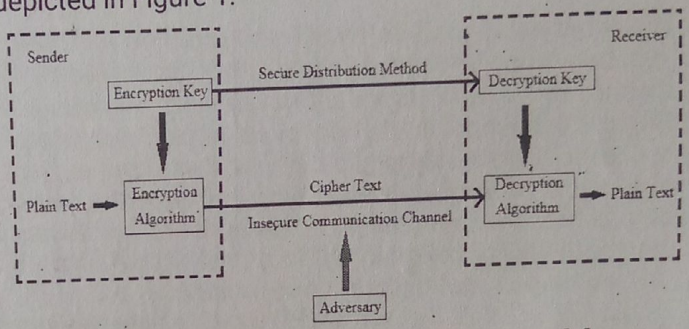

Figure 1: Basic model of cryptography, [2].

1.2 Components of Cryptography

The basic components of cryptography are, [3]:

- Plaintext: Original data that has not been computationally coded.
- Cipher Text: It is plaintext encoded or encrypted in the form of unreadable human language.
- Encryption Algorithm: This is a mathematical procedure or algorithm which takes plaintext as the input and generates ciphertext as the output using the encryption key.
- Decryption Algorithm: A mathematical method to transform ciphertext to plaintext using a decryption key. It is the inverse form of an algorithm for encryption.
- Encryption Key: This is the parameter specifically generated to produce the usable output, i.e., using an encryption algorithm to obtain ciphertext.
- Decryption Key: This is the parameter required to convert the encrypted data, i.e., the ciphertext, to its original form, i.e., the plaintext of the intended recipient.

2 DNA Cryptography

DNA cryptography is another rapidly evolving approach within the cryptographic domain that focuses on sequences of DNA. The concept of DNA cryptography is inspired by the DNA molecule, which has the capacity to store, process, and transmit information in

a mangle format using DNA digital coding. It operates on the DNA computing principle which uses 4 bases to conduct computation. The DNA computing principles are Adenine (A), Guanine (G), Cytosine (C), and Thymine (T) [4]. DNA encryption methods are:

DNA Random One Time Pad Based

In this technique, a series of randomly ordered, non-repeating characters are used to enforce a one-time pad. If an input ciphertext is used once, it will not be used again to increase security. The size of the plaintext in this scheme may be equivalent to the size of a one-time pad. The DNA One-Time Pad method is used to transform the short parts of plaintext messages to ciphertext. To encrypt the plaintext, a DNA digital coding is used. DNA digital coding is the process of using the principles of DNA, namely: Adenine (A), Guanine (G), Cytosine (C), and Thymine (T) to substitute binary digits using the combination of 0s and 1s, [5].

DNA Chip-Based Cryptography

The Microarray is also called the DNA chip. This DNA chip, made of semiconductor-designed nucleic acid and an electronic circuit. This technology represents excellent development in the field of cryptography based on DNA. A DNA chip is used to store, handle, and maintain a significant volume of the genome and biological information. The text and photographs are encrypted using biochemical processes. The drawback of this strategy is the sudden physical factor shift that is used to produce negative outcomes, [2].

DNA Fragmentation

This approach is used in the DNA sequence for library building. This splits the DNA sequence into tiny bits. Most encryption algorithms use it as a second protection layer. It's also being applied in key encryption, [1].

DNA Steganography

DNA Steganography is used to conceal one message inside another. Images, audio, and video are reused to preserve vast volumes of data, but data can be lost when there is a sudden environmental change, [6].

3 Methodology

This research adopted DNA cryptographic method to store, process, and transmit information in a mangle form using shift cipher and DNA One-Time Pad encryption techniques. To implement the 2-level data security approach, a Shift Cipher technique was used at the first security level and DNA One Time Pad (OTP) encryption technique for the second security level.

3.1 Shift Cipher Encryption Technique

The general idea of shifting plaintext by k letters (or adding k modulo 26) is called a shift cipher. A Shift Cipher is a Symmetric-key cipher where the plaintext and the pseudo-random key (keystream) are combined by XOR operation. In the encryption, each byte of plaintext is encrypted with the corresponding character of the keystream. An alternate name is the cipher state, where the encryption of each character depends on the current state, [7].
The mathematical notation of Shift Cipher is $C \equiv P + k$ modulo 26. Where P stands for any number between 1 and 26 that represents a plaintext letter, C represents a ciphertext letter, and k is the encryption key.

3.2 DNA One Time Pad (OTP) Encryption Technique

The DNA Random One-Time Pad Encryption technique is implemented using a set of randomly arranged non-repeating characters as the input. This set of random characters works as a pad. It is considered highly secure because if an input ciphertext is used once, it is never used for any other message, due to which it is named as One-time-pad. The length of the pad should be equal to the length of the plaintext in One-time-pad, [8]. The one-time-pad encryption equation is generalized as $Ci = E(Pi, Ki)$ for I=1, 2, 3,....., n, where E is the encryption operation, Pi is the I^{-th} character of the plaintext, Ki is the I^{-th} byte of the key used for the information, Ci is the I-th character of the resulting ciphertext, and n is the length of the keystream. Both the keystream K and the encryption operation must be kept secret.

4 Results and Discussions

The proposed 2-level data security approach was designed to eliminate the limitations of the existing information security methods of low confidentiality and high execution time.

4.1 2-Level Data Security Algorithm Designs

Table 1 presents the encryption algorithm using the Shift Cipher technique at the first security level and the OTP technique at the second security level.

Table 1: Encryption Algorithm.

Step	Algorithm
	// First security level
1	Enter input P_I (where i=1, 2, 3, ...N characters) and the input could be (alphabet, numbers and special symbols)
2	Compute the length **N** of P_I
3	For each P_i, shift P_i using the shift key value and convert to its corresponding decimal value P_{ASCII}
4	Convert P_{ASCII} to its equivalent binary string P_B
5	Compute P_B length **L** obtained from step 4
6	If **L** is less than 8-bits, convert P_B to 8-bits

string by
inserting 0s at the prefix of P_B to generate 8-bit string P_{Bytes}, else goto step 7

$P_{Byte} = [L / 8]$ if $L \bmod 8 = 0$
$P_{Byte} = [L / 8] + 0$ if $L \bmod 8 \neq 0$

	//Second security level
7	Generate key K_{OTP} using P_{Byte} and a pseudo random number **R**
8	Apply XOR operation on P_{Bytes} and its respective K_{OTP} obtained from step 7 to get the DNA binary form
9	Convert the result of the DNA binary form to DNA sequence P_{DNA}
10	Generate ciphertext C_{DNA}

In Table 1, the input P represents the plaintext while I is the Ith position of a character (symbol, number, alphabet). In P, each character is converted to its corresponding decimal value, denoted as P_{ASCII}, and further converted to binary numbers, denoted as P_B respectively. The binary number for each character P_B is transformed into an 8-bit string, denoted as P_{Byte}. KOTP represents the key variable. It is generated using the key generation function that accepts two parameters: the P_{Byte} of each character and a generated pseudo-random number. The encryption algorithm applied an XOR operation on the P_{Byte} and K_{OTP}. The binary result obtained is converted to a DNA sequence and the ciphertext is generated. The decryption algorithm is presented in Table 2.

Table 2: Decryption Algorithm.

Step	Algorithm
	// Security level I
1	Enter input (DNA sequence **Ci**, where i=1, 2, 3,.....**n** characters)
2	Calculate the length of the DNA sequence **N**
3	For i=1 to **N**
4	Shift C_i using shift key value and Convert C_i to its corresponding binary string C_B
5	C_{Bits} = concat(C_{Bits}, C_B)
6	Increment i
7	if **i** less than or equal **N**, repeat steps 4 to 6, else goto step 8
8	Compute C_{Bits} length **L**
9	Divide **L** by 8 to get new length **m** Where **m** = **L** / 8
10	For j=1 to m, divide C_B into m groups of 8-bit binary string C_{Bytes} C_{Bytes} = substr(C_B, j, 8)
	//Security level II
11	Generate key_{OTP} using **m** and the input corresponding pseudo random number (i.e. obtained from the database)
12	Perform **XOR** on C_{Bytes} and key_{OTP}
13	Convert **XOR** 8-bit binary string result to its equivalent decimal value C_{ASCII}
14	Substitute C_{ASCII} to its corresponding plaintext character
15	Increment j
16	if **j** less than equal **m**, repeat steps 11 to 15, else concatenate the plaintext characters (i.e. obtained from step 10) and generate the plaintext **P**

In Table 2, the DNA sequence C is received from the sender and each character of the DNA sequence position C_i is transformed using the Shift key value, then converted into its corresponding binary number C_B. K_{OTP} is generated using the length of the resulting binary strings m and the input corresponding pseudo-random number (i.e. obtained from the database). The decryption function performs an XOR operation on the C_{Bytes} with the K_{OTP}. The resulting binary strings are converted to an ASCII number. The ASCCII numbers are transformed into their corresponding characters as plaintext P. Table 3 depicts the Key Generation Algorithm used in generating keys.

Table 3: Key Generation Algorithm.

Step	Algorithm
1	Enter input **N** and **R** (where **N** is the input length and **R** is a pseudo random number)
2	Convert **N** and **R** to their corresponding binary sequence (i.e. N_B and R_B respectively)
3	Compute the length of N_B and R_B
4	Find the difference **D** of N_B and R_B
5	If **D** is not equal zero and N_B length is greater than R_B length then add zero(s) of size **D** to the prefix of R_B else, add zeros of size **D** to the prefix of N_B
6	Apply XNOR operation for the result of N_B and R_B (i.e. the result obtained in step 5)
7	Generate key K_{OTP}

In Table 3, the key generating function requires two parameters to process the key. The first parameter is the length of the input to be encrypted, and the second parameter is a generated pseudo-random number. Both inputs (parameters) are converted to their corresponding binary strings. The length of both strings is compared by finding the difference and then converted to 8-bit strings. After converting the two strings to an 8-bit string, the XNOR operation is applied to generate the key denoted as K_{OTP}.

4.2 Experimental Results

The interfaces presented in Figure 2 and Figure 3 were developed using the PHP programming language. Figure 2 displays the experimental result for the encryption of "DNA ENCRYPTION FOR TMS" as input.

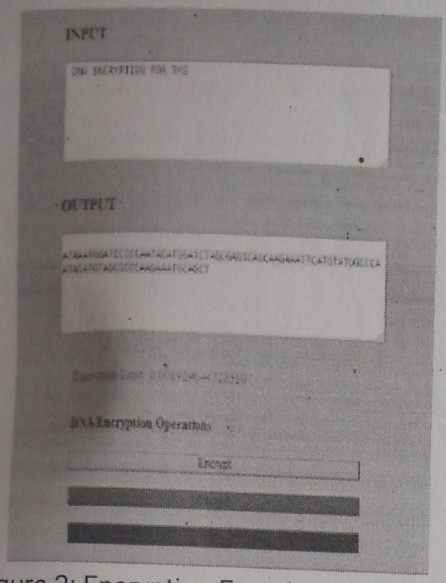

Figure 2: Encryption Experimental Result.

The encryption result of the input "DNA ENCRYPTION FOR TMS" was executed in 0.00193 and the output of the encryption process was displayed at the output field as "ATAAATGGATCCCCCAATACATGGATCTAGCGAGTCA GCAAGAAATTCATGTATGGCCCAATAGATGTAGCGCC CAAGAAATGCAGCT". Figure 3 depicts the decryption result of the input (DNA sequence) and a plaintext was displayed as the output of the encrypted information.

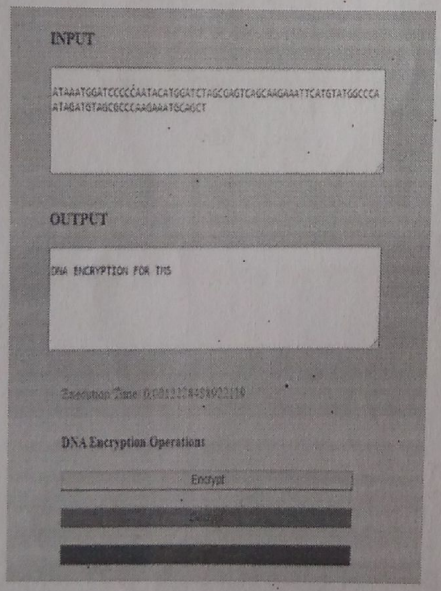

Figure 3: Decryption experimental result.

The software experiment was performed at an increasing number of characters in order to see the progression of the runtime. Table 4 presents the measurements of the execution time for encryption.

Table 4: Measurements of the encryption runtime.

Plaintext size (chars)	Runtime (ms)
3000	0.42
2500	0.37
2000	0.27
1500	0.19
1000	0.14
500	0.08
400	0.06
300	0.05

As presented in Table 4, the measurements of the encryption runtime for plaintext characters of size 100 to 3000 characters show that plaintext with 100 characters was executed within 0.02 milliseconds and plaintext with 3000 characters was executed within 0.42 milliseconds. The encryption runtime of plaintext characters between 0 to 500 characters is presented in Figure 4.

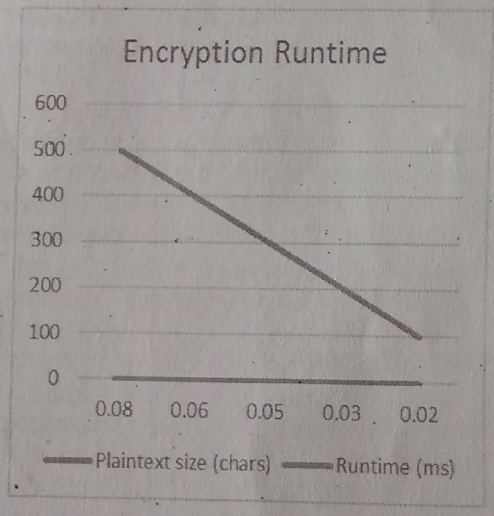

Figure 4: Growing rate of the encryption runtime.

Figure 4 depicts the encryption runtime against the plaintext character size is linear. That is, the higher the number of characters of a plaintext the higher the execution time. Table 5 presents the measurements of the decryption runtime for plaintext characters of size 100 to 3000 characters.

Plaintext with 100 characters' length was encrypted within 0.02 milliseconds and plaintext with 3000 characters' length was executed within 0.19 milliseconds. The result shows that the higher the number of characters of a DNA ciphertext, the higher the execution time.

Table 5: Measurements of the decryption runtime.

Ciphertext size (chars)	Runtime (ms)
3000	0.19
2500	0.14
2000	0.12
1500	0.09
1000	0.06
500	0.03
400	0.03
300	0.03
200	0.02
100	0.02

The finding in Table 5 reveals the decryption algorithm execution time is faster compared to the encryption algorithm execution time. The encryption runtime of the plaintext characters between 0 to 500 characters is presented in Figure 5.

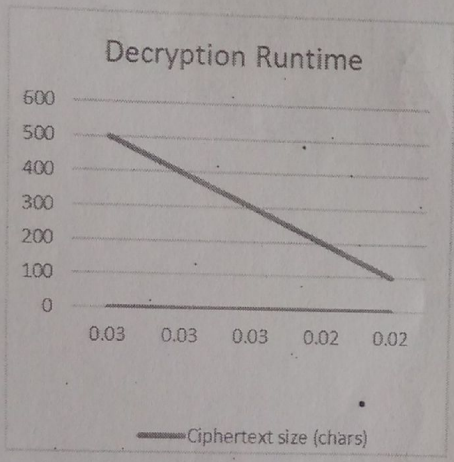

Figure 5: Growing rate of the decryption runtime.

Figure 5 shows that the encryption runtime against the plaintext characters' size is linear.

5 Conclusions

In the current era of technology, protecting the confidentiality of stored information as well as transmitted data is extremely crucial. The concept of DNA cryptography is inspired by DNA molecules that can store, process, and transmit information in a mangle form. This research used Shift Cipher in reshuffling the position of the text and OTP in generating a pseudo-random number encryption key, which makes the technique strong enough to protect information from all attacks, including brute force attacks. The experimental result of the encryption and decryption runtime indicates that the developed 2-level data security approach has a better execution runtime compared to [9] research on Two Level Text Data Encryption using DNA Cryptography.

References

[1] M. Darbari and V. Prakash, "A new framework of distributed system security using DNA cryptography and trust based approach." International Journal of Advancements in Research and Technology, 2014, vol. 3, no. 3, pp. 1-4.

[2] S. R. Kumar, "Review on DNA Cryptography." Research at Electronics and Communication Science Unit Indian Statistical Institute, International Journal of Engineering and Technology (IJET'15). 2014.

[3] N. Gulati, and S. KalyaniS. "Pseudo DNA cryptography technique using OTP key to secure data transfer." International Journal of Engineering Science and Computing, vol. 6, 2016, no. 5, pp. 5657-5663.

[4] B. B Raj, and V. Panchami, "DNA-based cryptography using permutation and random key generation method." International Journal of Innovative Research in Science, Engineering and Technology, vol. 3, 2015, no. 5, pp. 263-267.

[5] G. Singh, and K. Kamaljit, "A Review to an Invincible Cryptographic Approach: DNA Cryptography.", 4(1), 327–331, 2015. https://doi.org/10.17148/IJARCCE.2015.4175

[6] T. Anwar, A. Kumar, and S. Paul, "DNA cryptography based on symmetric key exchange." International Journal of Engineering and Technology (IJET'15), vol. 7, 2015, no. 3, pp. 938-950.

[7] P. T. Sudhakara, S. Aditi, K. Chahat, and B. Prantik, "An Extended Hybridization of Vigenere and Caesar Cipher Techniques for Secure Communication." 2nd International Conference on Intelligent Computing, Communication & Convergence (ICCC-2016), 2016.

[8] A. Hazra, S. Ghosh, and S. Jash, "A Review on DNA Based Cryptographic Techniques.", 20(6), International Journal of Engineering and Advanced Technology (IJEAT), 2018, 1093–1104. https://doi.org/10.6633/IJNS.201811

[9] E. Vidhya, and R. Rathipriya, "Two Level Text Data Encryption using DNA Cryptography.", International Journal of Computational Intelligence and Informatics, Vol. 8: No. 3, 8(3), 106–118, 2018.