Design and Implementation of a New DNA Based Stream Cipher Algorithm using Python

Kamel H. Rahouma[1], Faten M. AbdelGhany[1], Lamiaa N. Mahdy[2], Yahiya B.E. Hassan[3]

[1] Faculty of Engineering, Minia University, Minia, Egypt
[2] Higher Technology Institute in Tenth of Ramadan City, Cairo, Egypt
[3] Higher Institute of Engineering, Minia, Egypt
kamel_rahouma@yahoo.com

Abstract

This paper proposes a design and implementation of a new DNA - Based stream cipher algorithm. We implement this algorithm using python language and django framework. The algorithm is presented and its cryptanalysis is explained. The main purpose of this algorithm is to utilize one of the 1.6×10^8 real DNA sequences that are available in the online database (NCBI database) as a suitable real random DNA sequence (i.e., a reference sequence) to generate true random keys with different lengths and different data to overcome the problems of key lengths in some algorithms. By using the four bases of DNA (A< C, G, T), we generate dynamic DNA tables to replace message characters by a dynamic DNA sequence. The proposed technique is also based on using a second order chaotic equation to generate pseudo random sequences which are used for different purposes such as: generating one-time pad (OTP) keys, changing the message block lengths, generating dynamic coding tables for the ASCII characters of the message. The OTP key is XOR-ed with the message block. Then, the result is converted to a DNA sequence. Finally, we apply steganography concept by hiding the resulted cipher text in a randomly selected file from the used database. The obtained cipher text contains the information that will provide enhanced security against the intruder's attack.

Keywords: *DNA Cryptography; Symmetric Cryptography; Asymmetric Cryptography; One Time Pad (OTP);Steganography,Deoxyribo Nucleic Acid (DNA), Bioinformatics.*

1. Introduction

In recent years, many Internet applications are developed such as on-line shopping, internet banking and electronic bill payment etc. Many transactions processed over wire or wireless public networks require end-to-end secure connections. Some transactions should be private. There are a huge evolution in the number and type of attacks that should be dealt with by data security specialists to protect sensitive data from any unauthorized disclosure or unknown modification during transmissions or during storage[1].Most systems have several security requirements that can be met through the use of cryptographic techniques when implemented properly. Here, some examples security services, that a system might need, are given:

1) **Confidentiality**: It is the property by which information is encrypted (changed into unintelligible information) such that unauthorized parties can not disclose it.

2) **Data Integrity:** It is the property by which it is checked up if unauthorized people have changed the information or not. This can be done by using hash functions and digital signature algorithms.

3) Authentication services: refers to the problem of confirming or denying a person’s claimed identity [11]. This can be done using several cryptographic mechanisms such as, digital signature or message authentication codes; and key agreement techniques.

However, selecting a cryptographic method depends on the application demands such as the response time, bandwidth, confidentiality, integrity and etc.. Cryptography is a method of coding/decoding data to become unreadable or accessible by unauthorized users, to protect data during transmission and/or storage. Cryptography can be divided into two main components: a cryptographic algorithm and a cryptographic key. The algorithm is a mathematical function, and the key is a parameter used by that function[1].

Cryptographic algorithms can be classified into symmetric and asymmetric algorithms. Symmetric algorithms use the same key to encrypt and decrypt data. Thus,the key must be kept secret and only known to authorized parties. Hence, it is referred to as private key. Symmetric algorithms can be classified into block cipher and stream cipher. A block cipher (such as DES, 3DES, AES, etc.) applies a fixing and computable function repeatedly to encrypt a block of data at once as a group using different fixed-length key for each round. A stream cipher combines a plaintext stream with a key stream in a way to produce a cipher stream. The keys are generated using logical procedures or mathematical functions, which normally need an initial value or password[1].

Asymmetric algorithms use two related keys. One of these keys is referred to as a public key (it is made public), and the other one is referred to as a private key (it is kept secret).In such algorithms, data encrypted with any of these two keys can only be decrypted using the other key. This depends on the required security service (confidentiality or authentication). Examples of asymmetric algorithms include: RSA, Diffie-Hellman (DH), Al-Gamal, etc. They use mathematical functions for encryption/decryption and key generation. Therefore, they are slow. Thus, they are used for securing key-exchange over unsecure communication channels [1].

Hash functions are a cryptographic service that converts a large message into a block called the message digest (MD). They are used for detection of error and modification that may occur during transmission. The MD is sent along with the message. An MD is computed from the received message. If the new MD equals the received MD, then, the message is genuine and no changes have happened. The hash function is a One-Way Function that converts an arbitrary length message M to a fixed length message digest MD and it may be denoted as H(M)[2].

Steganography is another that can be used with cryptography to increase the security of data during transmission. Steganography is a technique of hiding information and/or scrambling it. Information may be embedded inside some medium (such as a text, or an image, or an audio or video clip, etc.) or scrambled in a way that it will be unintelligible. In the received the hidden information is extracted from the medium or descrambled. Many algorithms have been proposed for that[3].

Deoxyribo Nucleic Acid (DNA) cryptography is emerging as a new favorable cryptographic field, where DNA is used to carry the information or to be used as a data encoding approach. Recently, many DNA- based algorithms have been developed for data cryptography and cryptographic key generation [1]. DNA is a molecule that represents the genetic material for all living organisms. DNA molecules consist of two long chains held together by complementary base pairs, twisted around each other to form a double-stranded helix with the bases on the inside. A DNA sequence consists of four nucleic acid bases A (adenine), C (cytosine), G (guanine), T (thymine), where A and T are complementary, and C and G are complementary [10], as shown in figure 1.[4]

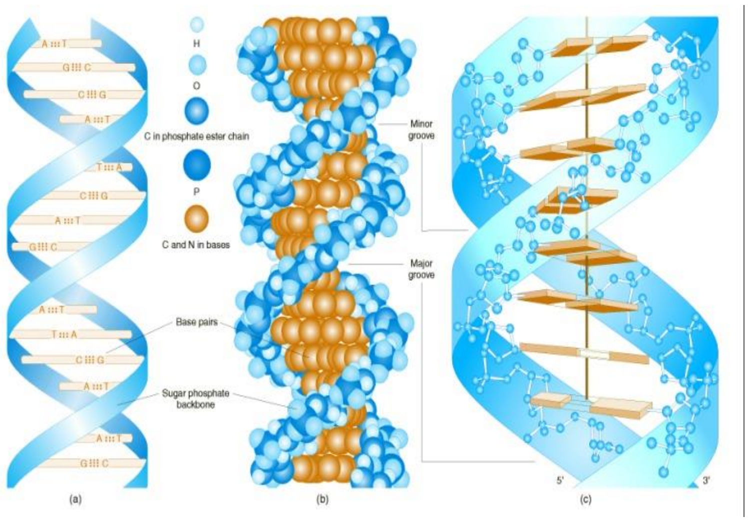

Figure 1: Three representations of the DNA double helix

DNA molecules have huge storage capacity. A gram of DNA molecules consist of 10^{21} DNA bases which is roughly about 10^{8}tera-bytes. So, it can be concluded that, a few grams of DNA can restrain all the data stored in the world. An important feature of the sequence of DNA is that it is completely random. Thus, we can use these data to get random keys with different lengths or hide data in DNA. These DNA advantages have motivated the ideas of DNA cryptography[5].

It may bring a new hope for powerful and unbreakable algorithms for the following reasons:

1) It is a new field of cryptography arising with the research of DNA computing in recent years.
2) It stores a huge amount of random data (1 gm of DNA has 10^21 DNA bases, i.e, 1 gm = 10^8tera bytes.) in a small volume with the combination of only these four letters A, C, G, and T.
3) It can be utilized in conventional cryptography by choosing random keys with different lengths as one time pad sequences for encryption and decryption.
4) It achieves the higher level of security while sending data over network.
5) This increases the efficient for DNA cryptography because of having the random keys, the long key size, the changing key size and data, the high speed in computations and the high quality of services.

This paper proposes a design and implementation of a new DNA - Based stream cipher algorithm. We implement this algorithm using python language and django framework. We use the python library “Pycryptodome” with two added features. The first feature is using random One-Time Pad keys with different lengths which we choose from a free DNA sequence database. This is done by using a simple chaotic equation to generate complex un-correlated sequences which refer to the positions of keys in the DNA file. The second feature is hiding the result of the encryption in another DNA sequences database. In addition to that, we implement an encryption stream algorithm, which uses the simple chaotic equation to randomly divide the message into blocks. Then, we, dynamically, generate a DNA sequence table (as a code book) to express each block of data using a corresponding DNA sequence. Also, the key streams are generated with lengths equals to the message block length. Each block from the message is XORed with the corresponding key stream to obtain the cipher block. Then, we hide the cipher block inside a DNA file using the same techniques.

The rest of the paper is organized as follows: **Section** 2 introduces the related work and **Section 3** presents our proposed algorithm. Section **4** presents the results and discusses them while Section **5** gives some cryptanalysis of the proposed algorithm. Section **6** highlights some conclusions. A list of the used references is given at the end of the paper.

2. Related Work

In [6], authors introduced the genomic DNA method as a hiding medium, known as DNA steganography. They proposed a new technique called DNA-based cryptographic method for data hiding in DNA media. The algorithm includes two phases. The first phase is to use a preprocessing Key as a seed for generating 16 random English letters to form a 4 × 4 play fair cipher grid. Their play fair cipher is enhanced in a way that made it the most satisfying suitable encryption technique to be mutual with DNA steganography. It also increases the hiding capacity to 2 bits per nucleotide rather than 1.5 bpn achieved by the current DNA-based play fair cipher and 1.992 bpn achieved by the AES. Also, it provides a suitable security level with lower speed than all the discussed ciphers with no restrictions on the data size or the key size.

In[7], authors presented the implementation of DNA cryptography based on dynamic DNA sequence table using cloud computing. Securing data in cloud has basic issues such as processing, compression and speedup computation. There are many cryptographic techniques to strengthen security of the stored data on cloud computing. Authors aim to overcome the security problem. Their technique considers dynamic sequence table to assign ASCII characters initially and then, a finite number of iterations are applied to get on the modified binary values and later On One-Time-Pad (OTP) is applied on these binary values. The OTP Cipher text is again processed through genomic conversion. Finally, it is converted into a compressed cipher text by using amino acid table which consists of protein sequences that increases the confusion of the cipher text. The resulted the cipher text will contain the information that will provide enhanced security against the intruder’s attack.

In[8],authors introduced many steganography techniques based on DNA sequences from worldwide databases such as the Gen- Bank. With the current available computational powers, brute force attacks on those DNA data sequences can easily be carried out. So, proposed a new technique called hiding secret messages using artificial DNA sequences. These sequences are generated by integer chaotic maps. They developed two steganography algorithms for hiding a cipher message in artificial DNA sequences. They used this type of chaotic maps to allow the compatibility between devices with different architectures, such that the extraction and decryption of the hidden message can be carried out correctly. The first algorithm has low computational requirements but it is vulnerable to statistical attacks when the cipher message is divided into a large number of sub-messages. The second algorithm deals with this issue.

3. The Proposed Work

This paper aims to design and implement a new stream cipher algorithm using python language and django platform. A simple chaotic equation is used to generate complex un-correlated pseudo random sequences. Using these sequences, the message block lengths are determined and OTP DNA keys are randomly selected with different lengths equal to the block lengths. Then, we generate the dynamic DNA sequence table that we use to express each block of the message in DNA sequence. The OTP DNA key is XORed with the obtained representation of the message block to give the cipher text. To encrypt, the same chaotic equation is used under the same conditions to obtain the pseudo random sequences to determine the blocks of cipher text and the dynamic DNA keys. The cipher text blocks are XORed with the DNA keys to give the representation blocks of the message locks. These blocks are used to obtain the original message blocks from the DNA sequence table used in the sender. Note That: for each message block we generate different dynamic sequence table, also we generate different OTP DNA key with the same length as the message block length. The algorithm goes into the following steps:

A- DNA Digital Coding

DNA digital coding can be encoded by four kinds of bases.These are Adenine (A), Cytosine (C), Guanine (G) and Thymine (T). The four bases can be encoded using 2 binary bits. The coded bases can be as: {A, C, G, T} = {00, 01, 10, 11}.

It is clear that Adenine is the complement of Thymine and Cytosine is the complement of Guanine. This is what happens in real, where strands of the DNA are connected at the points where each complementary bases are tied together. In this work, we can generate keys with different random lengths (for instance, “ATCGATCGAGT”).By using the DNA digital coding, the result of conversation is “0001111000011110001001”,and also this conversation is applied on the plain text after converting it into a DNA sequence.

B- Generating Dynamic DNA Sequence table

The message is read as characters. We divide this message (plaintext) to blocks of random lengths ranged from 64 to 128 characters. Then, we convert each block of characters to DNA sequence using a dynamic DNA sequence table. We generate this DNA sequence table using the following python Figure2.

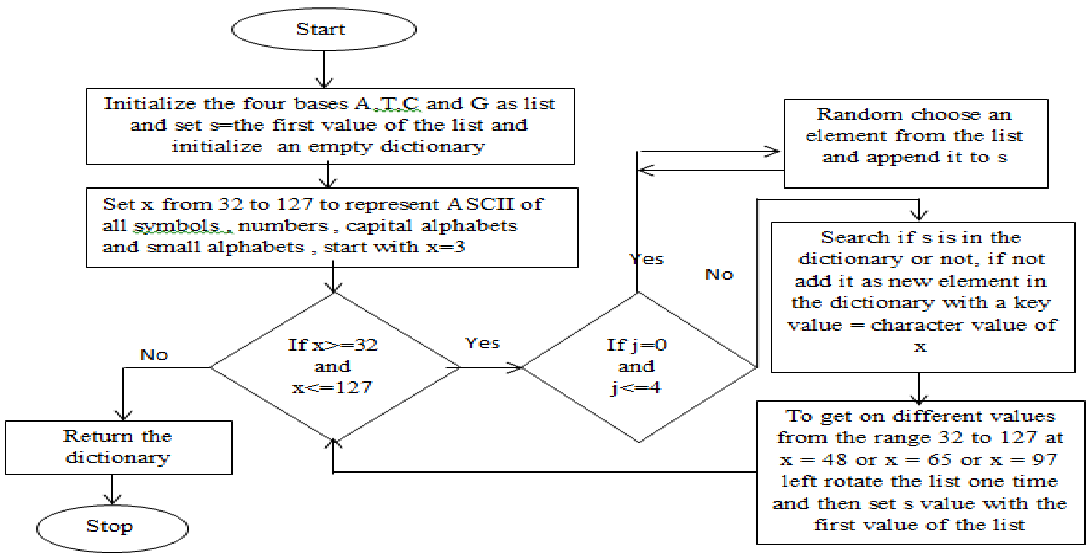
Start
Initialize the four bases A,T,C and G as list and set s=the first value of the list and initialize an empty dictionary
Set x from 32 to 127 to represent ASCII of all symbols , numbers , capital alphabets and small alphabets , start with x=3
If x>=32 and x<=127
No
Yes
Return the dictionary
Stop
If j=0 and j<=4
Yes
No
Random choose an element from the list and append it to s
Search if s is in the dictionary or not, if not add it as new element in the dictionary with a key value = character value of x
To get on different values from the range 32 to 127 at x = 48 or x = 65 or x = 97 left rotate the list one time and then set s value with the first value of the list

Figure 2: Flowchart of Generating the Dynamic DNA Sequence Table

Note that, we use the random function generator with different seeds. Each time we generate dynamic a DNA sequence table, we use a different seed value. Thus, the following table is not fixed, but it changes dynamically with each block of message. Both the sender and the receiver have the same dynamic DNA sequence tables. Table 1 gives an example of these tables.

Table 1: A code book of the ASCII symbols (S) and corresponding DNA sequence (DNAseq)

S	DNAseq	S	DNAseq	S	DNAseq	S	DNAseq	S	DNAseq	S	DNAseq
' '	ACAT	0	CAAA	@	CCAC	P	TTCA	‘	TCCG	p	GACA
!	AGGT	1	CACC	A	TACT	Q	TTTA	a	GAGC	q	GACT
"	AAAG	2	CCGT	B	TCCT	R	TAGA	b	GTGC	r	GGAT
#	AGAC	3	CGAG	C	TACG	S	TGAG	c	GACG	s	GGTG
$	AAGC	4	CCTT	D	TGCC	T	TAAA	d	GTAA	t	GCTT
%	AACT	5	CCGT	E	TCTA	U	TGAC	e	GTAC	u	GACC
&	AGAA	6	CTGT	F	TAGT	V	TGAG	f	GCCT	v	GACT
'	AATC	7	CTCT	G	TTAA	W	TAAC	g	GCTA	w	GCCC
(	ATTG	8	CCGT	H	TGGC	X	TCCT	h	GAGT	x	GATC
)	AATT	9	CTCA	I	TGTT	Y	TGAA	i	GATG	y	GTCG
*	AATG	:	CTAG	J	TTCC	Z	TAAG	j	GATT	z	GTGA
+	AAGA	;	CCGC	K	TACT	[	TCAT	k	GGGC	{	GGCT
,	AGAG	<	CACA	L	TATG	\\	TAAG	l	GTTG	|	GGTG
-	AAGC	=	CATA	M	TAGT	]	TCCA	m	GTGA	}	GAAC
.	ACAC	>	CTAC	N	TGTC	^	TGTT	n	GACT	~	GATG
/	ACGT	?	CCAG	O	TATT	_	TCCG	o	GCCG	\x7f	GAGT

C- The Encryption Process

1) Chaotic Function

Chaotic sequence can produce apparently independent sequences which are examined in the context of communication applications. On the contrary a pseudo-random sequence generator, if an adversary gets a copy of that generator and the master key; he can create the same keys and destroy your system. Also the random number generator's sequence cannot be reproduced. We use this simple chaotic equation[9]:

$$y(n+1)=r * x(n) * \{1-x(n)\} \qquad (1)$$

We initially repeat this equation at least twenty times to get on unexpected random value by using initial values for r=3.6 and x(0)=0.3.
The following flowchart shows how we can generate random value by using the previous chaotic equation:

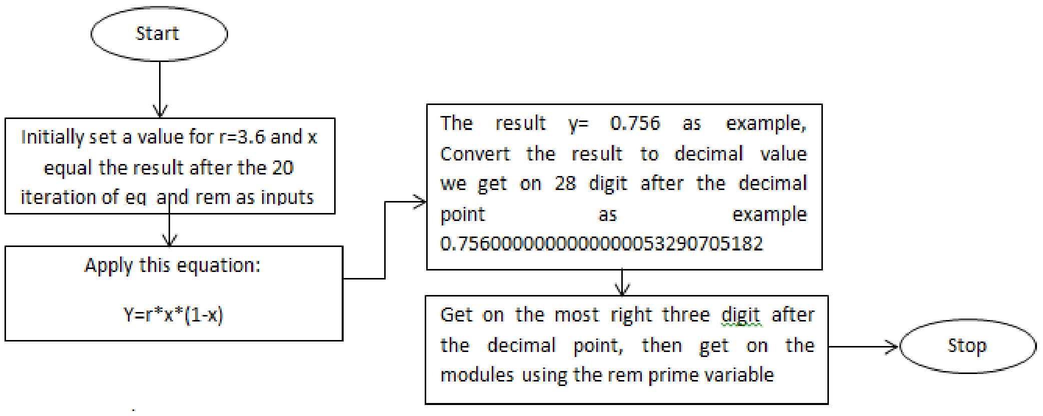
Start
Initially set a value for r=3.6 and x equal the result after the 20 iteration of eq and rem as inputs
Apply this equation:
Y=r*x*(1-x)
The result y= 0.756 as example, Convert the result to decimal value we get on 28 digit after the decimal point as example 0.7560000000000000053290705182
Get on the most right three digit after the decimal point, then get on the modules using the rem prime variable
Stop

Figure 3: Flowchart of Generating Random Value by Using the Previous Chaotic Equation

2) OTP Key Generation

There are almost 1.6 × 108real DNA sequences available on the online database (NCBI database). We can use one of them as a suitable real random DNA sequence (i.e., a reference sequence) to generate true random keys with different lengths and different data. Then, we apply the following steps to get on OTP keys:

Step 1: Collect all the files of the DNA sequence database.
Step2: Randomly select only one file from these files using the chaotic function (It is preferred that the number of files is prime and used for finding the modulus. If not, then choose the nearest bigger prime number).
Step 3: Open the file and then count the number of lines in the file, by using the number of lines (preferred to be a prime number also, or else choose the nearest bigger prime number for modulus) we can select only one random line using the chaotic function.
Step4: Count the line size and use it for selecting the position in this line (using the chaotic equation) to start reading the key string.
Step5: The key length must be equal the block length, by using this length we can define the key starting from the selecting position in the selected line. If the length of the required key is bigger than the line length, we can complete the key value from the next lines.
Step 6: The value of the key is converted as a DNA sequence, using the coding process mentioned in section III.I.
Step 7: We repeat all the previous steps when we need to generate a new OTP key.

3) Encryption Algorithm

We apply the following steps to encrypt a message:

Step 1: Divide the message into blocks of different lengths from 64 to 128 characters using the chaotic function. Note that the block length is preferred to be a prime number or else the nearest bigger prime number is used for the modulus. as prime number and it value near by the maximum range of the length.
Step 2: An initial dynamic DNA sequence table of ASCII characters is generated and known by both the sender and the receiver. This table is dynamically changing for each message block. Each time we call the dynamic DNA sequence table function, a new table will be reproduced.
Step 3: The selected block is organized according to thetable to convert it to DNA format.
Step 4: The resulted DNA format of the block is converted to binary using the following mapping: A-00, C-01, G-10 and T-11.
Step 5: True random OTP DNA key is generated with a key length equal to the length of the block in a binary format.
Step 6: An XOR operation is applied between the obtained key and the message DNA sequence from the table.
Step 7: The result of the XOR operation is arranged into a DNA format.
Step 8: The XOR output DNA format is hidden into the free DNA sequence database which is similar to the one we used to get the OTP keys. We follow two techniques to hide data as shown in the flowchart (a, b) in figure 4.

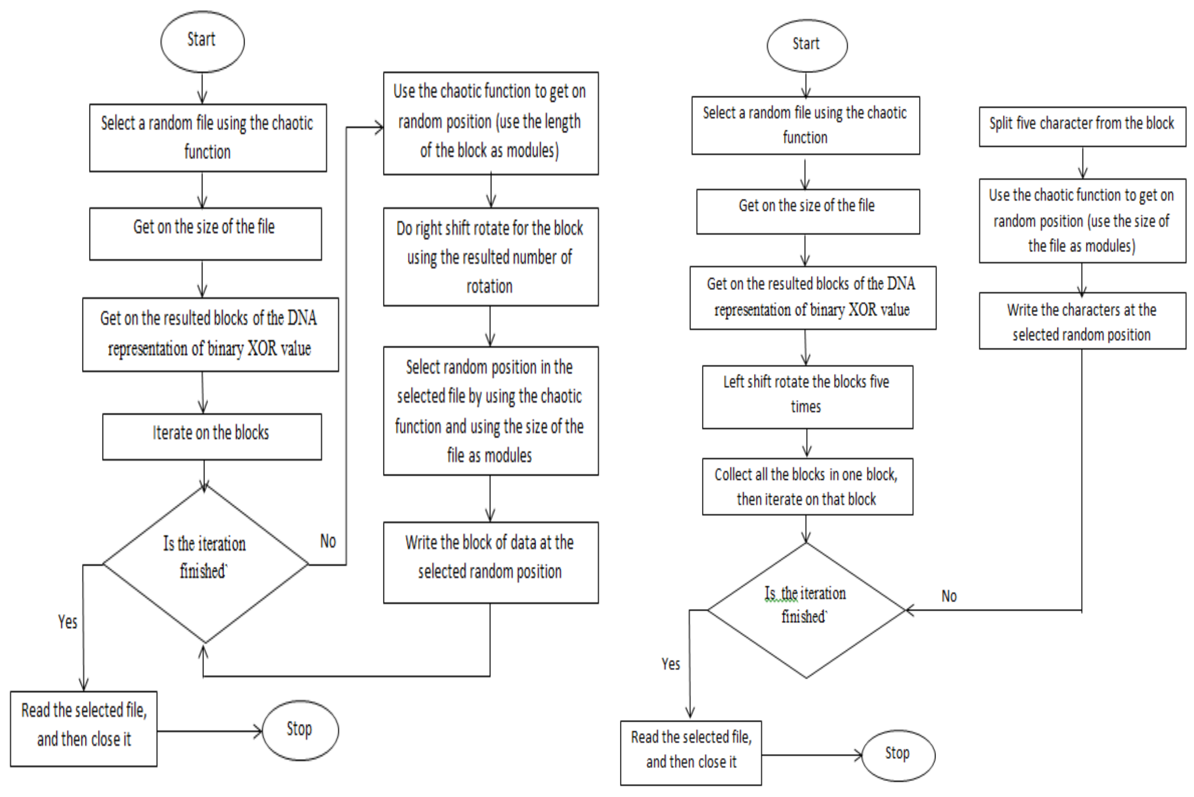
Start
Select a random file using the chaotic function
Get on the size of the file
Get on the resulted blocks of the DNA representation of binary XOR value
Iterate on the blocks
Is the iteration finished`
No
Yes
Use the chaotic function to get on random position (use the length of the block as modules)
Do right shift rotate for the block using the resulted number of rotation
Select random position in the selected file by using the chaotic function and using the size of the file as modules
Write the block of data at the selected random position
Read the selected file, and then close it
Stop
Start
Select a random file using the chaotic function
Get on the size of the file
Get on the resulted blocks of the DNA representation of binary XOR value
Left shift rotate the blocks five times
Collect all the blocks in one block, then iterate on that block
Is the iteration finished`
No
Yes
Split five character from the block
Use the chaotic function to get on random position (use the size of the file as modules)
Write the characters at the selected random position
Read the selected file, and then close it
Stop

Figure 4: (a): First flowchart for Hiding Data (b) Second flowchart for Hiding Data

4.Results and Discussion

Example: We apply the algorithm to a message. The steps and results are explained as follows:

4.1. The message plain text:

This is a simple example to illustrate the steps and results of (new stream encryption algorithm), that is based on DNA sequence. Made by eng. faten Mohammed

4.2. Obtaining the message metadata:

Message length = 154 symbol
Number of blocks= 2
Block lengths = 92, 86
Needed padding zeros = 92 + 86 – 154 = 24 zeros.
The padded message is:

This is a simple example to illustrate the steps and results of (new stream encryption algorithm), that is based on DNA sequence. Made by eng. faten mohammed000000000000000000000000

4.3. Encryption of the first block:

1) Generate the dynamic DNA sequence table for this block as in table (2).

Table 2: A code book of the ASCII symbols of the first message block

S	DNAseq	S	DNAaeq	S	DNAseq	S	DNAsequ	S	DNAseq	S	DNAseq
{	ACAA	0	CCTG	@	CCTC	P	TACG	`	TTAA	p	GATC
!	ATTA	1	CGCT	A	TATG	Q	TTGA	a	GTTA	q	GAAT
“	ATTT	2	CACG	B	TAGC	R	TGGG	b	GTAT	r	GTGG
#	AATC	3	CTAA	C	TGAA	S	TATT	c	GTTA	s	GTTA
$	ACCT	4	CACA	D	TAGG	T	TTGT	d	GTGG	t	GGCT
%	ACAC	5	CATG	E	TACG	U	TCCA	e	GTCT	u	GCCA
&	AGGT	6	CGGG	F	TTCG	V	TTTT	f	GCGG	v	GTAC
"	AAAT	7	CTAC	G	TAGT	W	TTAT	g	GCTG	w	GCAC
(	AATA	8	CTAC	H	TTAC	X	TATA	h	GCCT	x	GCCA
)	ACAA	9	CTAT	I	TATA	Y	TCAT	i	GATA	y	GGCT
*	ATAT	:	CGAA	J	TTCC	Z	TCAT	j	GTCT	z	GTTG
+	AGGC	;	CGGA	K	TCCA	[	TGCG	k	GGTT	{	GAGA
,	AAGG	<	CCGA	L	TGTA	\\	TCAG	l	GTTT	|	GTGA
-	ATGG	=	CCAC	M	TGCG	]	TTAA	m	GGTG	}	GTAC
.	ACTG	>	CGCT	N	TGGT	^	TTTG	n	GCGG	~	GCAA
/	AGTC	?	CTTT	O	TAGC	_	TCAG	o	GTGG	\x7f	GCTT

2) Convert the message to DNA sequence by using the previous table. This gives 92* 4 =368 DNA characters as follows:
TTCCGTGCGCAAGAATACCGGCAAGAATACCGGGAGACCGGAATGCAAGTTC
GTTAGGACGTGTACCGGTGTGGGGGGAGGTTCGTTAGGACGTGTACCGGCGTG
GATACCGGCAAGGACGGACGGTGGAATGCGTGGTCGGAGGCGTGTGTACCGGCGT
GTGCGTGTACCGGAATGCGTGTGTGTTAGAATACCGGGAGGGTGGTCCACCGGGT
CGTGTGAATGGTGGGACGCGTGAATACCGGGATGAAGACCGAGGCGGTGGTG
TGAATACCGGAATGCGTGGTCGTGTGGAGGTTCACCGGTGTGGTGGCCGGGTC
GCAGGTTAGCGTGCAAGGATGGTGACCGGGAGGGACGACCGGATGGTC

3) Extract the OTP key with the same length of the message block from the DNA files as:
TGGGGAAAATTCCTAGGCAACCTTGGGGAAATAGCCATGTTTAGCTTAAGGCC
TAGCGGATAGCCAGGCCTAGCGGAATTCCTAGGCAACCTTGGGGAAACGGAAT
TCCTAGGCAACCTTGGGGAAAATTCCTAGGCAACCTTGGGGAAATAGCCATGT
TTAGCTTAAGGCCTAGCGGATAGCCTAGGCAACCTTGGGGAAACGGAATTCCT
AGGCAACCTTGGGGAAAATTCCTAGGCAACCTTGGGGAAATAGCCATGTTTAG
CTTAAGGCCTAGCGGATAGCCATGTTTAGCTTAAGGCCTAGTAGCCATGTTTAG
CTTAAGGCCTAGCGGAATTCCTAGGCAACCTTGGGGAAACGGAATTCCT

4) Convert the message block and the OTP key into binary bits (76 bits each)

5) XOR the message block and the OTP key to give:
0001101000010111110010110100000011110000011110000010110000010001110111100
0001101100000000100100111101011010000010000101100000001001111011101000001
0010011100010111000100101010110100001011001111010000010000111110001110000
0110011110101011100000001101101010110001001101100000110111101110011000000
1111001000111110001000101010011001100111001110101010011101000011101111011
0100111010101000110000111000110001111111000011001011110011101001111000110
1000110000100101111111000101101100101011001110011100010100000110101011110
1000010110100001001010011110111001011010000111100111011000100110000011001
1101110110010110111110011111100000011101100101101111100111111000000111011
0000111101000111011111000101000101100010010110110101111101110101001001001
1100110000010111001011001101001000010O

6) Apply the DNA encoding onto the cipher block to obtain:
ACCAACTAGTTTTCGGGGCATCTCAAAGATCTGATGAATTCCGGTTTGGACGTA
CCTATCCTAGCTTGTGCTATAAGGCATTATACATGTAGCCACGTCATTGTGTTT
CACGACAGTTGGAATTGTTGCCTCCCGCTGAATCCCATGTGTCTGGAACTTACC
CCCAGTGCACCGACATAAGCTGCTTCTGGCCAGGAAATGAAAAATTTGATGAG
AGTTGCCAGACTGGGGGTATTAATAAGATCGGAACCCTTTCCGTACTGGAGCA
GGTTCGGAATAGGGACTTAGAACAGATACGTCGGATGCCTACAGCTATCGGAC
TCAACAGCCGGAGAGCTCTCATACTCCTCCCAATCGAAAAAACCTCG

4.4. Encryption of the second block:

Carry out the steps of the first message block.

5. Cryptanalysis of the algorithm

The algorithm has elegant features that make cryptanalysis really hard. Some of these features are as follows:

1) Changing the DNA code tables representing the ASCII characters of the plaintext.
2) Changing the block length.
3) Changing the encryption keys for each block which is known as one time key system.
4) Using the chaotic equations in conjunction with the last changes.
5) Hiding the encrypted characters in a DNA text.

However, these features make the cryptanalysis really hard. This can be explained as follows:

1) It is hard to apply the known (or chosen) cipher text attack. This attack depends on sending many plain texts to the system and receiving their ciphers texts. Then, the cryptanalyst tries to guess the key(s) used in the encryption. Changing the block length during the encryption means that the plaintext may be divided (in some situations) into multiple blocks. This means that more than one key may be used in the encryption. If no multiple encryption blocks are needed, then some padded zeros may be added to the attacker plain text block because it is very rare that the attacker plain text block will have the same length planned by the system.Thus, the attacker cannot guess the used to keys as well as the number of padded zeros. Hence, the attacker cannot get the encryption key(s) and consequently the corresponding plain text of the use cipher text will be obtained.

2) It is also hard to apply the known (or chosen) plain text attack. This attack depends on using common and repeated plain text phrases and blocks. However, changing the block length and the encryption key(s) makes it hard to figure out the cipher texts corresponding to the common repeated plain texts. Also, the padded zeros will be missing by the attacker. Thus, the attacker will not be able to guess any encryption keys.

3) It is hard to apply the man-in-the-middle attack. This attack depends on standing between the two communicating parties to get the message from the sender, change and add to it and then send it forward to the receiver. This needs that the attacker knows the encryption keys to be able to encrypt the added parts to the message. Again, the one-time pad keys and changing the block lengths make the attacker not able to succeed to play the man-in-the middle role.

4) It is finally hard to apply other attacks such as the side channel attacks and brute force attacks. These attacks depend on the weakness of the system and they may need to know the internal structure of the encryption/decryption system to determine the weakness points.

6. Conclusion and future work

A new DNA based stream cipher algorithm with elegant features and properties has been introduced. The system uses chaotic equations to generate pseudo random sequences. These sequences are used for different purposes. From these purposes: generating one-time keys, changing the message block lengths, generating dynamic coding tables for the ASCII characters of the plain text. Also, the system uses DNA sequences as keys. We used the python library "Pycryptodome" with two added features. The encryption/decryption system has been explained and its cryptanalysis is discusses. The proposed algorithm overcomes many of the restrictions in other algorithms such as it is not related to the lengths of certain keys nor the length of a particular plain text, the plain text is divided into blocks with random lengths. This is unlike some other algorithms where block length is fixed. Also, each block is encoded to a DNA sequence using a different dynamic DNA codebook table and encrypted using a different OTP key. Added to these advantages, the encrypted block DNA characters are hidden in the DNA file characters using steganography concepts. This makes it hard for the attacker to apply any one of the cryptanalysis methods such as Brute-force attack, Birthday attack and Man-in-the-middle attack. Also, the main advantage of this algorithm is that it uses available source (the 1.6×10^8 real DNA sequences that are available in the online database (NCBI database) to extract the OTP keys without using difficult algorithms to generate it. The encryption process is carried out by using the simple XOR operation. This gives a high speed system which can be used in many applications.

References

[1]. Hussain, S.M. and H. Al-Bahadili. A DNA-Based Cryptographic Key Generation Algorithm. The 2016 International Conference on Security and Management (SAM'16), Las Vegas, USA.pp.1-6. Last Accessed on: Nov. 23,2019. [Online]. Available at: https://www.researchgate.net/publication/30836247_A_DNABased_Cryptographic_Key_Generation_Algorithm.

[2]. Bakhtiari, S., R. Safavi-Naini, and J. Pieprzyk, Cryptographic hash functions: A survey. 1995, Citeseer.

[3]. Reddy, M.I.S. and A.P.S. Kumar. Secured Data Transmission Using Wavelet Based Steganography and Cryptography by Using AES Algorithm. International Conference on Computational Modeling and Security (CMS 2016).vol .85. p. 62-69. Last Accessed on: Nov. 23,2019. . [Online]. Available at: www.sciencedirect.com.

[4]. Griffiths AJF, Gelbart WM, Miller JH. Modern Genetic Analysis: Three representations of the DNA double helix. New York: W. H. Freeman; 1999. [Online]. Available at: https://www.ncbi.nlm.nih.gov/books/NBK21261/. [Accessed .Nov. 12, 2019].

[5]. Mondal, M. and K.S. Ray. Review on DNA Cryptography. arXiv preprint arXiv:1904.05528, 2019.

[6]. Marwan, S., A. Shawish, and K. Nagaty. DNA-based cryptographic methods for data hiding in DNA media. Biosystems, 2016.vol. 150. p. 110-118. [Online]. Available at: https://www.ncbi.nlm.nih.gov/pubmed/27634362.

[7]. Vinay S, A.P., Ankith, H.Akshay Kedlaya, Vasudev S Shahapur. Implementation of DNA Cryptography based on Dynamic DNA Sequence Table using Cloud Computing. INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) RTESIT, 2019. Vol.7(08).pp.1-4.ISSN (Online) : 2278-0181.

[8]. Eihab Bashier, Ghaidaa Ahmed, Hussam-Aldeen Othman and Rayan Shappo. Hiding secret messages using artificial DNA sequences generated by integer chaotic maps. International Journal of Computer Applications (0975 - 8887), 2013.Vol. 70,No.15,PP.1-5.

[9]. Rahouma, K.H. A chaos-based stream cipher algorithm for high speed networks and real time applications. SIMULATION SERIES, 2000. Vol.32.no.4.p. 95-99.

[10]. Amany Abdel Aziz El_Sebaai, Mohamed Abdel Fatah Belal, Hala Abdel Galil El_Sayed. A New Algorithm for Overlap Graph Reduction. Egyptian Computer Science Journal, ECS, Vol. 37 .No. 3. May 2013. ISSN-1110-2586.

[11]. Silvia Parusheva. A comparative study on the application of biometric technologies for authentication in online banking. Egyptian Computer Science Journal, Vol. 39. No. 4 .September 2015 .ISSN-1110-2586.