PERFORMANCE ANALYSIS OF HARDWARE IMPLEMENTED DNA ALGORITHM FOR SECURITY APPLICATIONS

A. Ruhan Bevi, Kathan Patel and S. Malarvizhi
Department of Electronics and Communication Engineering, SRM University, Kattankulathur, India
E-Mail: ruhan.b@ktr.srmuniv.ac.in

ABSTRACT
DNA Cryptography is used for secure end to end communication over a network by encrypting the messages. DNA is a well-known information carrier from one generation to another. DNA cryptography is preferred due to vast parallelism and information density that are inherent in any DNA molecule. In this paper, a new algorithm is proposed based on DNA cryptography, which enhances the security aspects of the data which is sent over a network. This is achieved by introducing feistel inspired structure and adding complex operations to it. Furthermore, this paper discusses DNA cryptosystem concepts based on the classic Vigenere cipher for substitution. A random function is used for generating One Time Password, which is unique for every transaction. This makes the algorithm complex and prevents the attackers to perform any brute force attacks. The results indicates that the confidentiality and integrity of the data is maintained and the feistel inspired structure for DNA cryptography using one time pad for key generation achieves a better encryption rate.

Keywords: DNA, Cryptography, DNA sequences, random function, one time pad, encryption.

1. INTRODUCTION
With the growing pace of Internet and network technology day by day, the security threats are also increasing due to lot of information flow on the network. There are various kinds of attackers/adversaries who always try to break into the system either to retrieve the crucial information or to destroy the integrity of data. So, the information security becomes necessary for modern computing systems. Generally, the secret data hiding techniques are used to protect the data from the adversaries. Cryptography and steganography are most common and widely used methods to prevent data from invaders. Cryptography performs the encryption of the data whereas Steganography hides the data from the hacker. In Cryptography, the encryption and decryption of data/plaintext is done with the help of key which may be shared public/private.

Increasing the bit size of encryption reduces the risk of being attacked. A 512 bit encryption seemed to be safe compared to 64/128 bit encryption. So, with the failure of modern cryptographic algorithm like DES and MD5, new methods of information security are needed to protect the data.

Efforts are taken continuously to improve the encryption methods while staying within the limits of available technology.

In the process of cryptography, the algorithm and the key play vital role to ascertain the secrecy of data while saving or passing it over the unsafe networks like internet. This is done in order to secure the data from the black hat hackers/adversaries and make it understandable only to its intended receiver. The general process of cryptography involving both encryption and decryption is shown in the Figure-1.

Figure-1. Flow diagram of cryptography.

In this paper, section 2 deals with the related works. Section 3 deals with DNA cryptography where DNA sequences, its properties and the concepts of DNA coding are discussed. The proposed method is conferred in section 4 with experimental results listed in section 5. Discussions on performance and security analysis is done in section 6. Conclusions are drawn in section 7.

2. RELATED WORK
In 1994, Adleman [1] proposed solution of Hamiltonian path problem using DNA. This resulted into the discovery of new field of research known as bio-computing. In 2006, Sherif T. Amin et al. [2] proposed the DNA cryptographic approach based on symmetric key, where key sequences are obtained from the genetic database and remain same at both ends while sending and receiving. Message/plaintext is first converted into binary format and then into a DNA format using substitution. Once the substitution has been performed and message is in the form of DNA sequence, a quadruple is chosen from the sequence obtained and a match is done with the key sequence. The position of match is used for encrypting/retrieving the message. Random position for each character in the plaintext are obtained this way and the file which contains these positions are defined ciphertext which is send to the receiver where decryption is performed in reverse order.
Deepak Kumar and Shailendra Singh [4] in 2011 proposed a new secret data writing techniques based on DNA sequences. A simple string is transmitted and a ssDNA One-time pad (OTP) key of 350 bits is generated which is 70 times longer than the plaintext and is used to perform encryption and decryption on the plaintext using symmetric key. 4^{350} different ssDNA strings should be addressed by the attacker to explore the key.

Bibhash Roy et al. [5] in 2011 proposed an improved symmetric key cryptography with DNA based strong cipher. A DNA computational logic is discussed for encrypting, storing and transmitting the data. In 2013, Wang Zhong et al. [6] proposed an Index based DNA encryption algorithm. Block cipher and Index of string is used for encrypting message into DNA sequences, which is send to the receiver by a secure communication medium. The message was converted into ASCII and binary which is further converted into DNA sequence to perform searching in the key sequence.

Ashish Kumar Kaundal and A.K Verma [8] in 2014 proposed a DNA cryptographic approach based on symmetric key using OTP. A fiestel inspired structure for DNA cryptography was proposed using genetic database. Mohammadreza Najaftorkaman, Nazanin Sadat Kazazi [11] in 2015 proposed a new DNA cryptography algorithm based on the classic Vigenere cipher. OTP key was generated and plain text was encrypted in DNA format using substitution method.

In our proposed method based on DNA cryptography, the plaintext is hidden in the DNA digital form. DNA cryptography uses DNA for storing data. DNA cryptography enables the confidentiality of data more high than the modern methods with the use of OTP keys and its size.

3. DNA CRYPTOGRAPHY

In this section, basic concepts of DNA sequences and its properties are discussed first. These concepts are essential to understand the DNA coding techniques which are discussed later in this section.

3.1 DNA sequences and it’s properties

Deoxyribonucleic Acid (DNA) is the heredity unit and also a information carrier of all living organism ranging from small viruses, chromosomes to complex human beings. DNA is a long polymer of small units called nucleotides. Each nucleotide consists of the following three components: a nitrogenous base, a five carbon sugar and a phosphate group. There are four different nucleotides classified upon the type of nitrogenous base which are A, C, T, G called Adenine, Cytosine, Thymine and Guanine respectively. Watson-Crick proposed a complimentary rule for DNA sequences where "A combines with T through double bound A=T and C combines with G through triple bound C≡G. Adenine and Guanine are called purines and Thymine and Cytosine are called pyrimidines in biological terms. DNA is a double helical structure with two strands running anti parallel as shown in Figure-2.

3.2 DNA coding

DNA cryptography is either based on molecular theory or on conventional approach. DNA Coding is a new way for storing large amount of data in the small fragment of DNA and providing security to it. DNA structure provides vast parallelism, exceptional energy efficiency and extra ordinary storage capacity. A 1 gm of DNA can store about 10^8 tera bytes of digital data [9]. It provides security by using the properties of DNA and good number of arithmetic operations.

The proposed method employs DNA cryptography consisting of regular key generation, encryption and decryption process [3, 10]. The DNA cryptography differs from the conventional cryptography in following ways:

i) Key generation: The key sequence is in a DNA format say ATCGCCAG which is purely based on OTP.

ii) Cipher text: The cipher text produced during encryption process by converting plaintext is also in DNA form.

iii) Decryption process converts the DNA cipher into its original plaintext.

DNA cryptography is based on both symmetric and asymmetric key, but it is easier to realize with symmetric key rather than asymmetric key. DNA Cryptography based on symmetric key [7] is shown below:

![Figure-3. DNA cryptography: An overview.](http://www.microbe.net)

The key generation is based on the OTP and according to the Shannon’s theory, $K_{size} \geq P_{size}$
where $K_{size}$ => Size of the key.
$P_{size}$ => Size of the plaintext.

The encryption block involves the process of shifting, confusion and substitution which are performed on the feistel structure of the message. The decryption block involves the process which is just the reverse of the encryption process and it uses the symmetric DNA key for decryption of the cipher text to obtain the plain text. The methodology for DNA encryption is described below:

1) Set fixed number of nucleotides adaptive for any length of plaintext $P$.
2) Devise an algorithm inspired on feistel inspired structure and Vigenere cipher in order to make the ciphertext.
3) MATLAB is used for simulating the proposed algorithm.
4) Variable length plaintexts consists of numeric, alphanumeric and alphabets are simulated to deliver corresponding ciphertext.
5) Validation of the algorithm with the conventional cryptographic algorithms for parameters like security, encryption and decryption time.

4. THE PROPOSED METHOD

The input to the encryption algorithm will be a plaintext which is entered by the user. Encryption algorithm converts the plaintext into ciphertext where it is coded as DNA sequence using a DNA key. The obtained ciphertext is given as a input to the decryption algorithm which converts it back into plaintext using the same DNA key. There are five main steps in implementing the proposed DNA cryptography algorithm: data pre-processing, key selection, encryption, decryption, and data post-processing.

4.1 Data pre-processing

1. The plaintext $P$ is converted into ASCII, $P_{ASCII}$ and then to binary plaintext, $P_B$. For simplicity let length of plain text be 16 bits.
2. Re-ordering of binary plaintext is done as follows:
   a) The first byte of $P_B$ is shifted left by 2 bits and the last byte is shifted right by 4 bits to obtain $P_{B1}$.
   b) After shifting, XOR operation is performed on $P_{B1}$ to obtain $P_{B2}$ as shown in Figure-4.

   Figure-4. Reordering of the plaintext.

4.2 Key selection

A pseudo-random generator is used to provide a seed of 32 bytes. Using this, a DNA sequence is provided as an input from the genetic database (Genbank). This pseudo-random generator generates a high quality OTP sequence based on the seed and is much secure than the other random functions. For the implementation of the proposed algorithm, the Bioinformatics Toolbox provided by the MATLAB is used. The getgenbank function is used to retrieve sequence information from GenBank database. This database is maintained by the National Center for Biotechnology Information (NCBI) [12].

NCBI bank is the master bank of all human genome and search for different kinds of DNA strings are performed. It provides a sample database of DNA strings and MATLAB is used to extract from the NCBI database. The following example extracts the DNA sequence from the NCBI bank:

\[
M = \text{getgenbank}('NC_001807','SequenceOnly',true);
\]

where the variable $M$ is returned with 32 bytes of DNA sequence. In this paper, the length of DNA secret key is chosen to be 64 (128 bits).

4.3 Encryption

The encryption process involves following steps:

i. Data in DNA form

The binary form of the plain text is converted into DNA form as follows:

<table>
<thead>
<tr>
<th>Binary</th>
<th>DNA</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>A</td>
</tr>
<tr>
<td>01</td>
<td>T</td>
</tr>
<tr>
<td>10</td>
<td>C</td>
</tr>
<tr>
<td>11</td>
<td>G</td>
</tr>
</tbody>
</table>

Hence 16 bit binary data, $P_{B2}$ is converted into DNA form of 8 characters ($P_{DNA}$).

ii. Combination with the key

Using a random function which gives the startindex, 8 character key is selected from long DNA key sequence of 64 characters. The template of startindex is shown,

\[
\text{startindex} = \text{fix}(10\times\text{rand})+1; \\
\text{startindex} =\text{startindex}\times5; \\
\]

The partial key (K) obtained is combined with $P_{DNA}$ to get $P_{\text{comb}}$ of length 16 characters.

Startindex acts as One Time Password which will be required for successful decryption.

iii. Substitution

The $P_{\text{comb}}$ obtained from step 2 is processed using Vigenere cipher table to get $P_{\text{all}}$ as shown below.
Table-1. DNA-Vigenere Table.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>T</th>
<th>C</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td>T</td>
<td>C</td>
<td>O</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>C</td>
<td>G</td>
<td>A</td>
</tr>
<tr>
<td>C</td>
<td>C</td>
<td>G</td>
<td>A</td>
<td>T</td>
</tr>
<tr>
<td>G</td>
<td>G</td>
<td>A</td>
<td>T</td>
<td>C</td>
</tr>
</tbody>
</table>

For example, let \( P_{comb} = \text{GGTCTCCGACTCGATA} \).
- The first character and the last character of \( P_{comb} \) is compared using the DNA Vigenere table. The row number represents the first character and the column number represents the last character.
- The first character of \( P_{comb} \) is ‘G’ and the last character of \( P_{comb} \) is ‘A’. Therefore the first character of \( P_{sub} \) will be ‘G’.
- In the next iteration the second character and the second last character will be compared. Therefore second character ‘G’ and second last character ‘T’ gives ‘A’ which is the second character of \( P_{sub} \).
- The process continues for the entire length of \( P_{comb} \) and the obtained \( P_{sub} \) is GATTGGAG.

The obtained \( P_{sub} \) of length 8 characters is given as input to step 2 for another round of encryption.

iv. Cipher text

Thus after \( n \) rounds, the cipher text of length 8 characters is obtained which is given to decryption algorithm.

Pseudocode: Encryption

```plaintext
for i:=1 to length(P_B2) do
  if P_B2(i) = '0' && P_B2(i+1) = '0'
    then
      P_DNA = 'A';
    else if P_B2(i) = '0' && P_B2(i+1) = '1'
      then
        P_DNA = 'T';
    else if P_B2(i) = '1' && P_B2(i+1) = '0'
      then
        P_DNA = 'C';
    else P_B2(i) = '1' && P_B2(i+1) = '1'
      then
        P_DNA = 'G';
  
  for j=1 to n do
    P_comb = P_DNA + K;
    
  for i:=1 to length(P_comb) do
    Vigenere Cipher substitution
  
end
```

The process flow of A, B, C is illustrated in Figure-5.

![Figure-5. Proposed DNA encryption flow diagram.](image-url)

4.4 Decryption

Decryption algorithm to decrypt the message at receiver side will consists of the following steps:

1. Receiver receives the packets, arranges them and obtains the ciphertext (C).
2. Receiver also obtains and enters the correct value of OTP which was generated in the encryption algorithm.
3. Combination with the key
   Using OTP, correct 8 character key is selected from long DNA sequence. This key is combined with C to get \( C_{comb} \) (16 characters).
4. Vigenere Cipher Operation
   \( C_{comb} \) is then processed using DNA-Vigenere Table to get \( C_{vig} \) (8 characters). This process is same as that explained in encryption process.
   Thus obtained is given as an input to Step 3 for further round of decryption.
5. Data in binary form
   Thus after \( n \) rounds, \( C_{vig} \) obtained is converted into binary form as follows:
   A=>00
   T=>01
   C=>10
   G=>11
   This binary form of data is denoted as \( C_{bin} \) whose length is of 16 bits.
Pseudocode: Decryption
for i:=1 to n do
{ 
C\text{comb} = C+K
for i:=1to length(C\text{comb}) do
{ 
Vigenere Cipher operation;
Obtain C\text{vig}.
}
}
for i:=1to length(C\text{vig}) do
{ 
if C\text{vig}(i) == 'A'
then
C\text{bin} = '00';
else if C\text{vig}(i) == 'T'
then
C\text{bin} = '01';
else if C\text{vig}(i) == 'C'
then
C\text{bin} = '10';
else C\text{vig}(i) == 'G'
then
C\text{bin} = '11';
end
}

4.5 Data post-processing
XOR operation is performed on C\text{bin} as shown below in Figure-6.

Figure-6. Reordering of ciphertext.

The first byte of D_{\text{shift}} is then shifted right by 2 bits whereas the last byte is shifted left by 4 bits to get D_{\text{shift}}. Finally, D_{\text{shift}} is converted into ASCII format and then to the desired plaintext.
The process flow of D and E is illustrated in Figure-7.

5. EXPERIMENTAL RESULTS
Validation for the encryption and decryption of the proposed method is demonstrated below. The entire simulation was done using MATLAB (R2015a). The Bioinformatics toolbox provided by MATLAB is used for the generation of the key.

Here 8 character plain text ‘SRM-EST1’ is encrypted and decrypted using the proposed algorithm.

5.1 Key generation
In our proposed algorithm, the length of Key is chosen to be 64.
Let the obtained key be 
Key=ACTCGATACATGACATAGACAGATACAGATACAACATAGAGGATACAGATACATAGACCCATAG

This DNA key is shared between both the sender and the receiver for this session. The partial key (K) used in both encryption and decryption process is chosen from the above long DNA key. This is done by using the random function. The startindex or the OTP helps in selecting the correct partial key in the decryption algorithm. If the value of OTP entered is wrong then the decrypted text and the original plain text would not match.

5.2 Encryption
The simulation of encryption in MATLAB is shown in Figure-8. Figure-8 shows the entered plaintext, DNA key, length of the DNA key, startindex, cipher text and the time required for encryption. The startindex acts as OTP.
5.3 Decryption

The simulation of decryption in MATLAB is shown in Figure-9. Figure-9 shows the obtained ciphertext, the DNA key, length of the DNA key, entered startindex, obtained plaintext and the time required for decryption.

Entire encryption and decryption process is secure and if the adversary wants to apply brute force method in order to compute the key sequence from ciphertext then $4^{64}$ different computations should be performed for DNA key sequences.

5.4 Hardware implementation

After successful implementation of the algorithm on MATLAB, the algorithm was implemented using Arduino Uno R3 board involving Atmega 328p microcontroller. The code was written using Arduino IDE. The encryption and decryption codes were loaded on two different Arduino boards. The plain text was given as an input to the Arduino board by the user using the serial window present in Arduino IDE. The cipher text obtained is given as an input to the decryption algorithm which deciphers and outputs the plain text on the LCD.

Figure-10 shows the encryption serial window module of Arduino Uno. It shows the entered plain text, generated OTP and the generated cipher text.

Figure-10. Serial Window for encryption module.

Figure-11 shows the decryption serial window module of Arduino Uno. It shows the entered OTP, obtained cipher text and the decrypted text.

Figure-11. Serial Window for decryption module.

Figure-12 shows the hardware implementation using Arduino Uno R3 boards.

Figure-12. Hardware implementation using Arduino Uno R3 boards.
6. DISCUSSIONS

6.1 Security analysis

**Friedman test (Kappa test)**

Friedman test is used to find the correct key length. One of the limitations associated with the Vigenere cipher is the repetition of its secret key where the key length is short. If the length of the key is predicted, the entire cipher text can become vulnerable. So, this is not desired. The algorithm should be secure enough that this test does not reveal the length of the secret key. According to the test, Key Length is given by:

\[
\text{Key Length} = \frac{K_p - K_r}{K_o - K_r}
\]

Where,
- \(K_p\) = The probability that two randomly chosen cipher text elements are the same.
- \(K_r\) = Probability of coincidence for random selection from the alphabets. In DNA cryptography, we have the four letters A, C, T and G. Therefore, the value is 0.25 and
- \(K_o\) = \(\frac{\sum c_{\text{act}} ni(ni-1)}{N(N-1)}\)

Where, \(c\) = size of alphabets.
- \(N\) = length of cipher text.
- \(ni\) = letter frequency.

One example depicting the test is shown. For the plain text ‘GO’ shown earlier, length of plain text = 16 bits.

Ciphertext = GCTCGGTT

Therefore, \(N = 8, c = 4, ni(A) = 0, ni(C) = 2, ni(G) = 3, ni(T) = 3\).

So, \(K_o = 0.25\).

Therefore, Key Length = \((K_p - K_r)/(K_o - K_r) = \infty\).

This test was performed on various combinations of the cipher text and the results showed that it was impossible to predict the length of the key.

**7.1 Brute force attack**

If the adversary wants to apply brute force method in order to compute the key sequence from cipher text than \(4^{160}\) different computations should be performed. This is very time consuming and almost impossible task.

**7.2 Frequency analysis**

The Vigenere cipher uses the English alphabets, so with frequency analysis we can guess the correct letter of the cipher text. The cipher text is in the form of ‘ATCG’. The frequency of occurrence of each of the letter in the cipher text keeps on changing according to the plain text. So, no relationship can be established with the letters together to guess the correct plain text.

**7.3 Performance analysis**

For an Intel(R) Core(TM) i5 CPU @ 1.6 GHz, the timing performance of encryption and decryption algorithms is shown in Table-5.

<table>
<thead>
<tr>
<th>Plain Text (Size)</th>
<th>Time for Encryption (secs)</th>
<th>Time for Decryption (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 bit</td>
<td>0.0938</td>
<td>0.1005</td>
</tr>
<tr>
<td>32 bit</td>
<td>0.1074</td>
<td>0.1116</td>
</tr>
<tr>
<td>64 bit</td>
<td>0.1135</td>
<td>0.1198</td>
</tr>
</tbody>
</table>

7. CONCLUSIONS

In this paper, a DNA cryptographic approach was a Feistel network with Vigenere cipher substitution. The addition of OTP in DNA cryptography makes the technique strong enough to protect from brute force attacks. So, if the attacker wants to know the exact key sequence then the attacker has to search \(4^{\text{key length}}\) different DNA key sequences which are very difficult and time consuming.

The shifting, confusion and substitution concepts used in the approach makes the algorithm secure and easy to use. Further the invention of energy efficient DNA nanochip for computers opens new horizons for the...
researchers in the field of DNA computing and information security.

REFERENCES


