Journal of Information Hiding and Multimedia Signal Processing ©2019 ISSN 2073-4212
Ubiquitous International Volume **10**, Number **3**, September **2019**

A Data Hiding Algorithm Based on DNA and Elliptic Curve Cryptosystems

M. I. Moussa
Computer Science Department, Faculty of Computers and Informatics
Benha University, Benha, Egypt
Email: mahmoud.mossa@fci.bu.edu.eg

E. M. Badr
Scientific Computing Department, Faculty of Computers and Informatics
Benha University, Benha, Egypt
Email: alsayed.badr@fci.bu.edu.eg

Sultan Almotairi
Department of Natural and Applied Sciences, Community College
Majmaah University, Al-Majmaah, 11952, Saudi Arabia
Email: almotairi@mu.edu.sa

Received January 2018; revised June 2018

Abstract. *This paper introduces a new encryption and hiding algorithm based on a tribble selection from the open usable DNA sequences and elliptic curve cryptosystem with key size level between 160-bit and 512-bit. A set of three DNA reference sequences was randomly selected from NCBI database. Two DNA sequences were selected to work with a mathematical system, which consists of two distinct function to generate two keys. These keys were used to encrypt the message in the first encryption phase. Another encryption phase used the elliptic curve cryptosystem to run the second encryption on the message. Then, the message was hidden in the third DNA sequence. In addition, the paper describes a new encoding and decoding approach to convert between integer numbers and elliptic curve points. The proposed algorithm keeps the length of the DNA sequence unchanged. The simulation results showed the effectiveness of the algorithm with respect to some parameters such as capacity(C), payload and bit per nucleotide (bpn).*

Keywords: DNA Steganography, Cryptography, Hiding messages, Elliptic Curve.

1. Introduction.

The DNA sequence consists of four bases, which are thymine T, cytosine C, adenine A, and guanine G. To get the complete nucleotide these bases are attached to the sugar/phosphate. Total number of the 3-letter combinations of units T, C, A and G equals 64 possible. The 3-letter combinations are called codons. The realistic DNA sequences possess some properties which make it a perfect media for hiding data. One special property of the DNA sequence is that, there is almost no difference between a fake DNA sequence and a real one. Another useful key element is that there are 163 million DNA sequences openly usable in the National Center for Biotechnology Information (NCBI) database [16]. Based on that, many research works on hiding data using DNA sequence were carried out[5, 6, 7]. The idea of using DNA computing in the fields of steganography and cryptography is a possible technology that may bring forward a new advance for powerful, or even unbreakable, algorithms. Steganography technologies represent a very important part in the future internet security [1, 3].

The most commonly used approaches for data hiding are the substitution method, the complementary pair method and the insertion method. These entails embedding the cipher message into a DNA sequence induceing new reference sequence with data hidden [11]. However, these methods have many limitations. For example, the insertion method expands the length of the DNA sequence and increases the redundancy, the resulting DNA sequence from the complementary pair method expands the DNA sequence in the process of embedding the secret message while the substitution method has a highly modification rate [2, 8]. Another approach used the complementary pair rules of DNA to hide data was described in [13]. A modification of the original substitution method, where the capacity of the updated version has been doubled was proposed in [1]. The original substitution method hides one secret bit per nucleotide where the modified one hides two secret bits per nucleotide. The modified algorithm increases the capacity; however, it still suffers from a high modification rate in DNA sequence, the modification attracts the attention of attackers. Another improvement of the original substitution method is proposed to minimize modification rate in DNA reference sequence [9, 17]. Other algorithms have used both data hiding and encryption together to communicate data securely. Firstly; the given message is encrypted using Amino Acids-Based Playfair cipher and the DNA. Secondly; the encrypted data is hidden into a DNA sequence using an insertion method. To recover the embedded data, the receiver executes the inverse process with the support of both the secret key and the reference DNA sequence [9]. Injective mapping is established between two secret bits and one complementary rule . Based on this mapping mechanism, the hiding scheme hides two secret bits by replacing only one character. The improved algorithm maintains the length of the resulting faked DNA sequence; these issues ensure robustness and security of the hiding algorithm [14].

A different data hiding algorithm used word document as host file and induced DNA coding, was introduced in [12]. The plaintext converted to DNA coding and encrypted to induce the cipher sequence, which was concealed in random primer DNA sequences. Then, the resulting sequence was hidden into a word document by substituting the least significant 2-bit in a three-color component. A reversible data hiding method used the histogram technique for transforming the DNA sequence to a binary string. Afterwards, it used a histogram mechanism to embed secret data. This scheme has some key features, which preserve the length of the DNA sequence and has a small modification rate [4].

Different scheme based on 8-bit binary string was described in [15]. Another scheme used a truly random mathematical system such as chaos system to generate artificial DNA sequences. These sequences are used to encrypt the plaintext and hide it into a realistic DNA [10].

This paper describes in subsection 3.1 a new encoding and decoding approach to convert between integer numbers and elliptic curve points. Subsection 3.2 presents a new encryption algorithm based on a triple selection from the open usable DNA sequences and elliptic curve cryptosystem with key size level between 160-bit and 512-bit. A set of three DNA reference sequences was randomly selected from NCBI database. Two DNA sequences are selected to work with a mathematical system (described in section 2), which consists of two distinct functions to generate two secret keys. Section 3.2 describes an encryption phase based on the elliptic curve cryptosystem. This runs the first encryption on the secret message. Then the two generated secret keys are used to encrypt the secret message again in the auxiliary encryption phase. The encrypted message is hidden in the third DNA sequence as appeared in section 3.3. The proposed algorithm keeps the length of the DNA sequence unchanged. The security analysis of the current work in section 4 shows that the probability of successfully guessing the hidden message is less than the probability of successfully guessing the hidden message in [1, 2, 5, 7-12, 15]. This increases the robustness of hidden data against modification attacks compared with these previous works. The simulation results in section 5 shows the effectiveness of the described algorithm with respect to three parameters such as bit per nucleotide (bpn), capacity(C) and payload. Section 6 shows the main advantage of the current work compared to related works.

2. The Mathematical System.

In this work, the auxiliary conversion function (2) was added to convert data between integers and elliptic curve points as described in section 3.1. Let E be an elliptic curve defined over a finite prime field F_p by the equation (1)

$$y^2 = x^3 + ax + b \bmod p \tag{1}$$

And the points on E with coordinates in F_p are;

$$E(\mathrm{F}_p) = \{(x, y) : x, y \in F_p,\ y^2 = x^3 + ax + b\} \cup \infty.$$

The auxiliary conversion function is used,

$$f_i(x_i, y_i) = \exp\left(-\frac{|x_i|^3 - |y_i|^2}{2c^2}\right) \tag{2}$$

The elliptic curve domain parameters over F_p are $(p,\ a,\ b,\ G,\ n,\ h)$ where p is a prime number, $\lceil \log_2 p \rceil = 2\gamma$, let $80 \le \gamma \le 256,\ c > 1,\ a,\ b \in F_p,\ G = (x_G,\ y_G)$ is a base point, n is the order of G, and h is the cardinality number of $|E(F_p)|/n$, subject to the constraints;

$p^\alpha \neq 1\,|(mod)n\ \forall\ 1 \le \alpha < 100;\ \#E(F_p)\,|\neq p|;\ 4a^3 + 27b^2 \neq 0(mod\,p);\ h \le 2^{\gamma/8}\ and\ n - 1\ and\ n + 1\ should\ all\ have\ a\ large\ prime\ factor.$

3. The proposed Scheme.

A triple simultaneous random selection (S, S', S'') were taken from the NCBI database. From these, two keys $(K_1,\ K_2)$ were derived using the above mathematical model which was presented in section 2. Generate a distinct integer sequence $Z = \{z_i\}$ by using the equations(1) and (2), for each number in Z in nondecreasing order extract a nucleotide from S and add it consequently in (K_1). The second key (K_2) was induced from applying the complementary rules on the nucleotides of (K_1). The following pseudocode shows how does it generates the keys$(K_1,\ K_2)$.

Algorithm 1 GET-KEYS($p, a,\ b$)

```
1. Generate all the pairs $P = (x_i, y_i) \in E(F_p)$.
2. $z_i \leftarrow$ INTEGERS $(P = (x_i, y_i))$, to convert Elliptic Curve Point-to integers
3. Select a DNA sequence $S_1$ from DNA database.
4. For each integer $z_i$ select the element in i-th position in $S_1$ and get the first key
       $K_1 = S_{1,z_1}.S_{1,z_2}.S_{1,z_3} \ldots .S_{1,z_i}.$
5. Apply the complementary $(AT)\,(CA)\,(GC)\,(TG)$ on $K_1$ the resulting sequence is the
   second key
       $K_2 = S'_{1,z_1}.S'_{1,z_2}.S'_{1,z_3} \ldots .S'_{1,z_j}.$
```

Algorithm-1 generated two encryptions keys

3.1. **Data Types Conversions.** Elliptic Curve Point should be mapped to integer numbers using the following algorithm;

```
Algorithm 2 GET-INTEGERS ( $P = (x_i, y_i)$ )
Input: A point $P = (x_i, y_i) \in E(F_p)$ defined by the domain parameters over $F_p$
Output: A non-negative integer $z_i$
  1. Compute $f_i(x_i, y_i) = exp(-\frac{|x_i|^3 - |y_i|^2}{2c^2})$.
  2. Induce the sequence of integers
        $h_i = 10^{14} * f_i(x_i, y_i) \quad mod \quad p$
  3. Sort the sequence $h_i$ and get the sorted sequence $h'_i$.
  4. Generate the sequence $z_i = h'_i + i$.
```

Algorithm-2 Converted the Elliptic Points to integer numbers

A non-negative number should be mapped to an elliptic curve points using the following algorithm;

```
Algorithm 3 ELLIPTIC-POINTS ($z_i$)
Input: A non-negative integer $z_i$
Output: A point $P = (x_i, y_i) \in E(F_p)$ defined by the domain parameters over $F_p$
  1. For each $i = 1, 2, ...$ the induced sorted sequence of integers is $h'_i = z_i - i$,
  2. Compute $( f_i(x_i, y_i) )^{-1} = 10^{14} * (h'_i)^{-1} \quad mod \quad p$
  3. Solve the elliptic $y_i^2 = x_i^3 + \; 2c \, ln f_i(x_i, y_i) \; mod \; p$.
  4. Let the set of solution of the elliptic curve in step 3 is
        $E(F_p) = \{(x, y) : x, y \in F_p \; satisfy \;\; y^2 = x^3 + ax + b\} \cup \infty$
  5. Mark all points in $E(F_p)$ as free
  6. For each $i$ do
      - $x_i \leftarrow \; z_i \; mod \; P$
      - Try to solve the next equation for $y_i$
            $y_i^2 = x_i^3 + \; 2c \, ln f_i(x_i, y_i) \; mod \; p$
      - Mark the solution point as unfree
      - If it does not solve delay $x_i$ and mark it free
  7. Assign the free points $x_i$ to the smallest free point in $E(F_p)$
  8. The set of integer points in the sequence $z_i$ now converts to the set of points on the
     above curve.
```

Algorithm-3 Converted the integer numbers to Elliptic Points

3.2. **Encryption.** The encrypted algorithm has two phases. In the first phase, it represented the plaintext message denoted (P) in binary bit string and mapped every 8-bits to one DNA codons, the plaintext message converted to a DNA nucleotide form denoted (DP) Algorithm-4 sorts the 4^3 DNA codons in lexicographical order$\{c_{000}, \; c_{001}, \; \dots, \; c_{333}\}$, then the index value NUM_FORMAT combines each three digits and transfers the resulting number to the corresponding DNA codons (ex. 122 converts to CGG the sender and the receiver agree on this). In other words, it described a one -one and onto map from all possible permutations between these 64 codons and the English alphabets (26 small letters, 26 capital letters), the numbers (0,1, ...,9) and some punctuation marks. Every codon c_{ijk} on the message (DP) had 8-bit integer denoted ijk. Algorithm-4 converted the number ijk to an elliptic point using algorithm-2.

Algorithm 4 GET-DNA-CODONS (DP)

```
1. Sort all DNA codons in lexicographical order $\{c_{000},\ c_{001},..,c_{333}\ \}$
2.  For i = 0 to 3 do
3.      For j = 0 to 3 do
4.          For k = 0 to 3 do
5.              NUM_FORMAT = i j k
6.              $c_{ijk} \longleftarrow$ NUM_FORMAT
7.              $DP = (\ x_i^{DP}, y_i^{DP}\ ) \longleftarrow$ ELLIPTIC-POINTS $(c_{ijk}\ )$
```

Algorithm-4 assigned one 9-bit integer number to each DNA codon

Algorithm-5 was used to encrypt the secret message DP, which was given the ECC public key cryptosystem and it's resulting cipher message was DP$' = m_1^{'} m_2^{'} m_3^{'} ... m_l^{'}$.

Algorithm 5 ECC-CRYPTOSYSTEM (p, a, b, G, n, h)

```
1. The receiver and the sender come to an agreement on the parameters $(p, a, b, G, n, h)$
2. The sender chooses his owner private key, $\kappa_s < n$ and compute his public key,
   $P_s = \kappa_s * G$.
3. The receiver chooses his owner private key, $\kappa_R < n$ and compute his public key
   $P_R = \kappa_R * G$.
4. The sender generates the ciphertext $\{kG, DP + k_R\}$ of the message points
     $DP = (x_i^{DP}, y_i^{DP})$.
5. $z_i \longleftarrow INTEGERS\ DP = (x_i^{DP}, y_i^{DP})$
6. $z_i \longleftarrow c_{ijk}$,
7. convert the sequence $c_{ijk}$ to $DP'$ then go to AUXILIARY-PHASE $(DP', K_1', K_2')$.
```

Algorithm-5 showed the elliptic curve cryptosystem steps

In the second phase, it applied the exclusive or operator XOR (denoted $\oplus$) on the nucleotides bases $A, C, G,$ *and* T according to the rules $A \oplus C = 00 \oplus 01 = 01 = C$ and $G \oplus C = 10 \oplus 01 = 11 = T$; continuing in this fashion to end up with all the possible XOR values between every two nucleotides bases. The requirements regarding the values in S-box are shown in Table-1; the distributions of outputs must be checked for uniformity to protect against the Davies' Attack, the outputs have no linearity in their function to the input, and there have to be individual values in every row of the S-box.

TABLE 1. S–box of DNA nucleotides

S_0–box	A	C	G	T
A	C	A	T	G
C	T	G	C	A
G	A	G	C	T
T	T	C	A	G

The two keys (K_1', K_2') which were generated by algorithm-1 and the encrypted message DP' which was got from algorithm-5 were passed as the arguments in the next pseudocode denoted AUXILIARY-PHASE $(DP',\ K_1', K_2')$.

Algorithm 6 AUXILIARY-PHASE (DP', K_1', K_2')

```
1. $Divide\ DP',\ into\ DP'(L)\ and\ DP',(R)$
2. $Compute\ IC_1 = DP'(R) \oplus K_1\ and\ IC_2 = S_0\text{-}box\ (IC_1)$
3. $IC_3 = expand\ \ IC_2$
4. $IC_4 = DP'(L) \oplus IC_3$
5. $IC_5 = Concatenate\ \{IC_4,\ DP'(R)\}$
6. $Divide\ IC_5 into\ IC_5(L)\ and\ IC_5(R)$
7. $IC`_2 = IC_5(R) \oplus K_2$
8. $IC`_3 = S_0 - box\,(IC`_2)$
9. $IC`_4 = Expand\ IC`_3$
10. $IC`_5 = IC`_4 \oplus IC_5(L)$
11. $DP'' = Concatenate\ \{IC`_5\ , IC_6(R)\} = m_1''m_2''m_3''...m_l''$
```

Algorithm-6 described addition step to encrypt the DNA message DP'

3.3. **Hiding the Encrypted Message** Dp''**.** Third sequence S'' was used to hide the encrypted message DP''. The letter $m_i^{''} \in DP''$ is substituted with the z_i position of the third DNA sequence $S^{''}$ to get a new sequence and send it to the receiver. Some DNA sequences contain non-labeled nucleotides (N) as well as the other nucleotides. The number of these nucleotides is low. We avoid hiding DP'' in N nucleotides to keep on the properties of the real DNA sequence. The replacement rules used to hide DP'' in S'' by replacing the second repeated letter in S'' with one of the four letters $\{A,\ C,\ G,\ T\}$ of the encrypted message DP''. Accomplishing those replacements was done based on the following replacement rules;

(i) No replacement for the second repeated letter in S'' when A in (DP'') is found.
(ii) The replacement in S'' is between A&C, G&T when C in DP'' is found.
(iii) The replacement in S'' is between A&G, C&T when G in DP'' is found.
(iv) The replacement in S'' is between A&T, C&G when T in DP'' is found.

The replacement rules are used in hiding processes and their inverses are used in recovery processes respectively. The IDs of the sequences ($S,\ S',\ S''$) were appended at the beginning of the resulting sequence to generate IDs$''$ (IDs$'$ (IDsDP'')) before sending the fake sequence to the receiver.

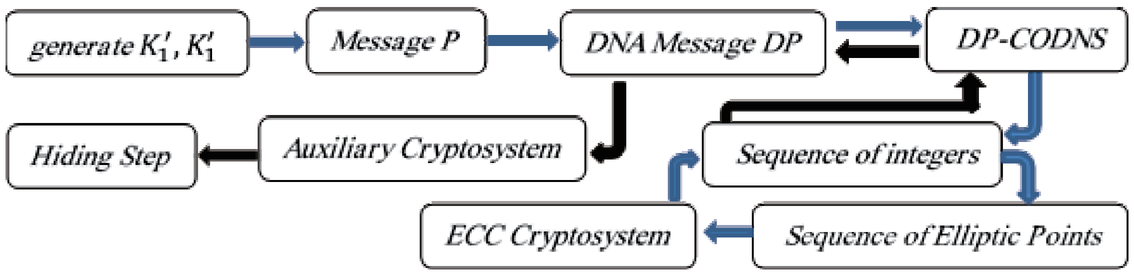

Figure 1. Data encryption and hiding diagram

The sender uses Algorithm-7 as the main algorithm to hide the encrypted message P.

Algorithm 7 ENCRYPT-HIDING-MESSAGE (*P, (S, S′, S″*)

```
Preprocessing:
  1. The set of all publicly DNA sequences $S = \{S_i \mid 1 <= i <= 163 \times 10^8\}$
  2. Consider the set of all unordered selections $\{(S, S', S'') | S, S', S'' \in S\}$
  3. Extract all non-labeled nucleotide (N) if exist.
Input: Three DNA sequences (S, S′, S″) and the secret message P
Output: A non-realistic DNA sequence within the secret message P hidden
  1. Encode the secret message P to DNA sequence DP using GET-DNA-CODONS
     (Algorithm-4).
  2. Generates the secret keys $K_1'$ and $K_2'$ using $GET-KEYS(p, a, b)$ (Algorithm-2)
  3. $DP' \longleftarrow$ ECC PUBLIC KEY CRYPTOSYSTEM $(p, a, b, G, n, h)$
  4. $DP'' \longleftarrow$ AUXILIARY-PHASE $(DP', K_1, K_2)$
  5. Hide the encrypted message $DP''$ in $S''$ using the replacement rules.
  6. Insert the extracted non-labeled nucleotide (N) if exist.
  7. Append the IDs of S, S', and $S''$ at the beginning of resulting $DP''$
  8. Send the faked sequence to the receiver.
```

Algorithm-7 was the main proposed algorithm which described encryption steps and hiding the secret message using three DNA sequences

3.4. Numerical Example.

Step1: Keys Generation

1. Use Algorithm-1 to generate the two secret keys (K_1, K_2) such that $a = 5$, $b = 5$, $p = 17$, and the elliptic curve equation over F_{17} is;
$$y^2 = x^3 + 5x + 5$$
2. The set of points satisfy the above equation are: (3,8) (3,9) (4,2) (4,15) (5,6) (5,11) (6,8) (6,9) (7,3) (7,14) (8,8) (8,9) (10,1) (10,16) (12,5) (12,12) (15,2) (15,15) (16,4) (16,13)
3. When $c = 2$ *and* $q = 37$, we get
   h_i = {16,8,14,30,5,8,14,18,23,9,26,5,20,8,9,26,19,26,3,18} and
   z_i= {4,7,8,12,13,14,16,17,23,24,27,30,31,33,35,39,43,44,45,50}
4. Choose S_1=ATCGAATTCGGGCTGAGTCACAATTCGCG CTGAGTGAACC
   S_2=GATATCGACTGAGGCACG CTGAGTGAAATTCGCACC
   S_3=TCGCATCGAATTCTGAGTCACAATTCTATTCGCG CTTGTTTCG-GAA
5. K_1= S_1 (z_i) = GT**T**GCTAGATGCTATC and K_2 = AT**A**GGGGGCAACCGGC

Step2: Data Types Conversions:

Encoding the secret message using the data types conversions Algorithm−1 and Algorithm−2

1. Let the secret message (m) to encode is 'Hello'
2. The secret message m in the binary form is:
   P =0100100001100101011011000110110001101111001000000110010001100 0000110001001101000010111 0
3. The binary secret message (P) must convert to DNA sequence form (DP) using Algorithm-4
   DP= ACTTAAGAGGAGGGT= ACT TAA GAG GAG GGT =c_{101} c_{220} c_{233} c_{233} c_{301}

4. Use Algorithm-2 to get the elliptic curve point corresponding to the index of the DNA codons, which are: (6, 9) = c_{101}, (3, 8) = c_{220}, (3, 9) = c_{233}, (4, 2) = c_{233}, (4, 15) = c_{301}

Step3: Encryption-1 (ECC-CRYPTOSYSTEM)

1. These points are encrypted using ECC- system (Algorithm-5) resulting the following cipher points: (12, 5) = $c_{220,}$ (15, 15) = $c_{210,}$ (16, 4) = $c_{312,}$ (5, 11) = $c_{030,}$ (8, 9) = c_{131}
2. Algorithm-4 used again to induce the DNA condones corresponding to the cipher ECC points and the resulting encrypted secret message is
   $DP' = c_{220}c_{210}c_{312}c_{030}c_{131} = GCCTGAAGCAAGCAT$

Step4: Encryption-2 (AUXILIARY-PHASE)

1. Encrypt the ECC encrypted message DP' using the two secret keys K_1 & K_2
2. Split DP' into left $DP'(L)$ = GCCTGAAG and right $DP'(R)$ = CAAGCATN
3. Apply the XOR and S_0-*box* on DP' (R) and DP' (L) consequently using K_1 K_2 to get the twice encrypted message $DP" = TCAG\ AGTACGGC\ AGTN$

Step 5: Hide the message $DP'' = TCAG\ AGTACGGC\ AGTN$

1. Select the third DNA sequent
   S''=TCGCATCGAATTCTGAGTCACAATTCTATTCGCG CTTGTTTCGGAA
2. Call the replacement rules in subsection 4.2 to hide the encrypted message DP'' in the third selected sequence S''.
3. *Append the IDs of S, S', and S'' at the beginning of* S''.
4. The sender sends to the receiver the following fake DNA sequence
   S'' =AAGTACGGCGCGCCCAGTCACAATTCTATTCGCG CTTGTTTCGGAA.

3.5. **Data Recovery and Decryption.** The receiver got the sequence IDs''(IDs$'$ (IDsDP'')) without any other DNA sequences, DNA-like sequences, or any sequences of numbers from the sender. The receiver processes the *DECRYPT-RECOVR-MESSAGE* pseudocode denoted Algorithm-8 to recover the original message P.

4. **Security Analysis.** The strength and robustness of the described algorithm are based on the following:

1. The reference DNA sequence S used in encryption, there are nearly 163 million DNA sequences available publicly. Consequently, the probability of making a successful guess for (S, S', S'') is $\left(\frac{1}{1.63\times10^8}\right)^3$.
2. The replacement rules are used in hiding (recovery), there are totally 24^4= $(4!)^4$ possible situations used for hiding (recovery), so the probability of having a successful guess is $\frac{1}{24^4}$. There are totally $(4!)^2$ possible XOR and S-box table. The probability of a successful guess is $\frac{1}{24^2}$.
3. There are six possible complementary rules. The probability of a successful guess for the complementary rule is $\frac{1}{6}$.
4. There are total $p_{64}^{64} = 1.27E + 89$ kinds of permutation between 64 codons which are generated from algorithm-4 and 52 the English alphabets, 10 digits (0,1...9), and two punctuation marks.
5. Therefore, the probability of guessing the secret message without implement ECC cryptosystem is: $\left(\frac{1}{1.63\times10^8}\right)^3\times\frac{1}{24^4}\times\frac{1}{24^2}\times\frac{1}{6}\times\frac{1}{p_{64}^{64}}\approx 2.5\times10^{-121}$.

6. The security strength level of ECC between 80-bit string and 256-bit string need elliptic curve over $F_p$where $2^{160} \leq p \leq 2^{512}$ which resultes the same level of security in RSA with key size K_{RSA}, where $1024 - bit \leq K_{RSA} \leq 15360 - bit$.

Algorithm 8 *DECRYPT-RECOVR-MESSAGE (P, (S, S′, S″)*

Input: A non-realistic DNA sequence IDs″(IDs′ (IDsDP″)) with encrypted message hidden.
Output: The given message P.

```
1. Extract the first bases that represent IDs of S, S′, and $S''$.
2. Extract all non-labeled nucleotide (N) from S″ if exist.
3. Find out the second repeated nucleotide in S″.
4. Extract $DP''$sequence from S″ using the replacement inverse rules.
5. Decrypt DP″ as follow:
      Divide DP″ into two halves DP″ $(R)$, DP″ $(L)$
            $IC`_2 = K_2 \oplus \mathrm{DP}''(R)$
            $IC`_3 = Apply\ \boldsymbol{S_0} - box\ in\ IC`_2$
            $IC`_4 = Expand\ IC`_3$
            $\mathrm{DP}\,(R) = IC`_4 \oplus \mathrm{DP}''\,(L)$
            $IC_1 = K_1 \oplus \mathrm{DP}\,(R)$
            $IC_2 = S_0 - \mathrm{box}\ (IC_1)$
            $IC_3 = Expand\ IC_2$
            $DP(L) = IC_3 \oplus \mathrm{DP}''\ (R)$
6. Convert each codon in $DP'$ to 8-bit integer values.
7. Use Algorithm-2 to convert 8-bit integer number to Elliptic Curve Point
   $DP = (\ x_i^{DP}, y_i^{DP}\ )$.
8. Decrypt the elliptic curve encrypted message as;
   $DP + k\ P_{\mathcal{R}} - \kappa_{\mathcal{R}}(kG) = DP + k\ \kappa_{\mathcal{R}}G - \kappa_{\mathcal{R}}(kG) = DP$.
9. Use Algorithm-1 to convert Elliptic Curve Point $DP = (\ x_i^{DP}, y_i^{DP}\ )$ to 8-bit integer
   number.
10. Convert each 8-bit integer values to DNA codon to get $DP$ then get the plaintext
    $P$.
```

Algorithm-8 was the main proposed algorithm which described the decryption steps and recovering the secret message using three DNA sequences

5. **Simulation Results.** The experiments were conducted and evaluated based on some parameters such as bpn, capacity,and payload. Table-2 shows an eight sequences from National Center for Biotechnology Information (NCBI) database [16] which were used in the experimental study of the proposed algorithm. Table 3 appears the measurements of the number of bit per nucleotide (bpn), the capacity, and the payload. The third column of the table gives the number of nucleotides before hiding the given message, and the fourth column shows the number of nucleotides after hiding the secret message. The number of nucleotides after hiding the secret message didn't changed. The sixth and the seventh columns show the results of using the algorithm in [8,10] on the eight sequences, the eight column shows results of using the proposed algorithm. The average bpn for our proposed algorithm is 0.61, where the average bpn for the algorithm in [8,10] were 0.57 and 0.44.

Table 2. The tested DNA sequences with experimental results

DNA Sequence	**Capacity C**	**No. of (N) nucleotides**	**The embedding capacity (bit)**
AC168901	191,456	250	111,704
AC166252	149,884	0	868,96
AC168908	218,028	918	127,184
AC168874	206,488	1300	115,560
AC168897	200,203	5186	113,184
AC153526	200,117	0	115,392
AC167221	204,841	0	115,392
AC168907	194,226	809	112,648

Table 3. Determines in detail the embedding capacity for each DNA sequence represented in bits

Locus	**Specifies definition**	**No. of nucleotides**	**Capacity C**	**Pay-load**	**bpn [10]**	**bpn [8]**	**bpn The proposed algorithm**
AC168901	Bos taurus clone CH240-18511	191,456	191,456	0	0.583	0.439	0.615
AC166252	Mus musculus6 BAC RP23-100G10	149,884	149,884	0	0.580	0.442	0.732
AC168908	Bos taurus clone CH240-95K23	218,028	218,028	0	0.583	0.443	0.539
AC168874	Bos taurus clone CH240-209N9	206,488	206,488	0	0.560	0.446	0.565
AC168897	Bos taurus clone CH240-190B15	200,203	200,203	0	0.565	0.451	0.638
AC153526	Mus musculus10 BAC RP23-383C2	200,117	200,117	0	0.577	0.434	0.619
AC167221	Mus musculus10 BAC RP23-3P24	204,841	204,841	0	0.563	0.424	0.588
AC168907	Bos taurus clone CH240-19517	194,226	194,226	0	0.580	0.444	0.596

6. **Advantages of Proposed Algorithm.** In comparison with previous related work in [1,2,5,7-12,15], the encryption algorithm can be compared by the following metrics:

1. Key Size
   The current work provides a key size level of ECC between 160-bit and 512-bit which gives the same level of security in RSA with key size K_{RSA}, where $1024 - bit \leq K_{RSA} \leq 15360 - bit$. The proposed algorithm depends upon several secret keys $(K_1^{'}, K_2^{'}, p, a, b, G, n, h)$ that increase the strength of the encryption, however only two secret keys were employed by [10].
2. Algorithm Complexity (encryption, decryption, and key setup)
   Playfair techniques was used by [1] for data encryption but the ECC cryptosystem, used in our work create faster, smaller and more efficient keys.

3. Best Methods of Attack
   The probability of successfully guessing the hidden message in the previous works [1, 2,5,7-12, 15, 17] is at most $\left(\frac{1}{1.63\times10^{8}}\right)\times\frac{1}{24}\times\frac{1}{6}$. Security analysis of the current work shows that the probability of successfully guessing the hidden message is $\left(\frac{1}{1.63\times10^{8}}\right)^{3}\times\frac{1}{24^{4}}\times\frac{1}{24^{2}}\times\frac{1}{6}\times\frac{1}{\mathrm{P}_{64}^{64}}\approx2.5\times10^{-121}$ This increases the robustness of hidden data against modification attacks compared with these previous works.

4. Rounds of the algorithm
   A unique feature of the proposed algorithm is the double data type conversions and the two phases of encryption on data.

5. Payload and Bit Per Nucleotide (bpn)
   The proposed algorithm has many features as shown in Table3 like not expanding the length of the DNA reference sequence, no change in the non-labeled nucleotides (N), a low modification rate compared to the previous works [8, 10]. The average of bpn is higher than bpn of previous works [8, 10].

7. Conclusions.

A hiding algorithm based on DNA sequences with double phases of encryption and new approach for encoding and decoding data has been proposed. The proposed algorithm used a 6-bit DNA coding instead of 8-bit ASCII codes, where each character of plaintext was denoted by three nucleotides. This increased the embedding capacity and minimized the modification rate. The plaintext message was converted to a DNA sequence, which was encrypted using the elliptic curve cryptosystem. Two secret keys were induced from the proposed mathematical system and the two selected DNA sequences. These keys were used to encrypt again the plaintext by using the auxiliary encryption phase. Finally, the double encrypted message was hidden into other DNA reference sequence. According to the security analysis, it is very difficult for attacker to identify the secret message, it takes long time if he considered the brute force approach. The simulation results showed an enhanced performance for the proposed data hiding algorithm.

REFERENCES

[1] A. Khalifa, S. Z. Reda, DNA-based data encryption and hiding using playfair and insertion techniques, *Journal of Communications and Computer Engineering*, vol. 2, no. 3, pp.44-49, 2012.

[2] A. Leier, C. Richter, W. Banzhaf, H. Rauhe, Cryptography with DNA binary strands, *BioSystems*, vol. 57, pp. 13-22, 2000.

[3] B. Shimanovsky, J. Feng, M. Potkonjak, Hiding data in DNA, *Revised Papers from the 5th International Workshop on Information Hiding, Lecture Notes in Computer Science 2578*, pp.373-386, 2002.

[4] C.C. Chang, T.C. Lu, Y.F. Chang, R.C.T. Lee, Reversible data hiding schemes for deoxyribonucleic acid (DNA) medium, *International Journal of Innovative Computing, Information and Control*, vol. 3, pp. 1-16, 2007.

[5] C. Guo, C. C. Chang, Z. H. Wang, A new data hiding scheme based on DNA sequence, *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 1(A), pp. 136-149, 2012.

[6] Chen, J, A DNA-based biomolecular cryptography design, *2003 IEEE International Symposium on Circuits and Systems*, vol. 3, pp. 822-825, 2003.

[7] D. Bhattacharyya, S. K. Bandyopadhyay, Hiding secret data in DNA sequence, *International Journal of Scientific & Engineering Research*, vol. 4, no. 2, 2013.

[8] F.E Ibrahim, M.I. Moussa, H.M. Abdalkader, Enhancing the security of data hiding using double DNA sequences, *Industry Academia Collaboration Conference (IAC)*, 6-8 April, Cairo, Egypt, 2015.

[9] F. E. Ibrahim, M. I. Moussa, H. M. Abdulkadir, A symmetric encryption algorithm based on DNA computing, *International Journal of Computer Applications,* vol. 97, no. 16, 2014.

[10] Eman I. Abd El- Latif, and M. I. Moussa, Chaotic information- hiding algorithm based on DNA, *International Journal of Computer Applications*, vol. 122, no. 10, pp. 41-45, 2015.

[11] H. J. Shiu, K. L. Ng, J. F. Fang, R. C. T. Lee and C. H. Huang, Data hiding methods based upon DNA sequences, *Information Sciences*, vol. 180, no. 11, pp. 2196-2208, 2010.

[12] H. Liua, D. Lin, A. Kadir, A novel data hiding method based on deoxyribonucleic acid coding, *Computers and Electrical Engineering*, vol. 39, pp. 1164-1173, 2013.

[13] J. S. Taur, H. Y. Lin, H. L. Lee, C. W. Tao, Data hiding in DNA sequences based on table lookup substitution, *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 10(A), pp. 6585-6598, 2012.

[14] TY Liu, WH Tsai, A new steganographic method for data hiding in microsoft word documents by a change tracking technique, *IEEE Trans Inf Forensics Secure*, vol. 2, no. 1, pp. 24-30, 2007.

[15] M. R. Abbasy, P. Nikfard, A. Ordi, M. R. Najaf Torkaman, DNA base data hiding algorithm, *International Journal on New Computer Architectures and Their Applications (IJNCAA)*, vol.2, no.1, pp. 183-192, 2012.

[16] Website, NCBI Database: <http://www.ncbi.nlm.nih.gov/>.

[17] Y. H. Huang, C. C. Chang, C. Y. Wu, A DNA-based data hiding technique with low modification rates, *Springer Science Business Media*, LLC 2012.