SAKARYA UNIVERSITY JOURNAL OF COMPUTER AND INFORMATION SCIENCES
http://saucis.sakarya.edu.tr/
e-ISSN: 2636-8129
Publisher: Sakarya University
Vol. 7, No. 3, 2024, 389-403
DOI: 10.35377/saucis...1483300

RESEARCH ARTICLE

Application of Classical and Genomic Cryptography on Textual Dataset

Alev Kaya[1*], **İbrahim Türkoğlu**[1]

[1] Fırat University, Department of Software Engineering, Elazığ, Türkiye

Corresponding author:
Alev Kaya, Fırat University,
Department of Software Engineering,
Tekirdağ, Türkiye,
alev.kaya@firat.edu.tr

Article History:
Received: 14.05.2024
Accepted: 30.07.2024
Published Online: 10.12.2024

ABSTRACT

Cryptology is one of the methods used when sharing confidential or private data over any communication network that poses a security risk. It is applied to restrict access, minimize or completely prevent dangerous situations. Cryptographic algorithms use a combination of mathematical operations and applications to protect information. It strives to ensure the confidentiality, integrity, availability and non-repudiation of information. In other words, it aims to keep data safe against all kinds of threats. However, the performance of these objectives depends on various factors. These factors include the file format used, the volume and complexity of the data. Additionally, the key system and application platform (software and hardware) also affect performance. These variables determine the effectiveness of cryptographic algorithms. In fact, existing cryptographic algorithms may be inadequate or ineffective in the face of new requirements. Therefore, new techniques need to be designed to meet such needs. This study, one of the new generation cryptographic techniques, includes a symmetric key genomics (DNA)-based application. The aim is to test the suitability of genomic encryption on artificial data sets (100 and 500 KB, 1 and 5 MB) generated from the content named "Siyasetname" in the Turkish textual data type. The usability of the genomic encryption technique, which has not been applied before in Turkish data sets, was tested by comparing it with classical algorithms such as AES (symmetric) and ECDH (asymmetric). Performance criteria are determined as encoding and decoding times (seconds), memory consumption (MB) and processor usage (%), which are accepted in the literature for textual data type. It is supported by different indicators according to dimensions and more successful outcomes compared to similar studies in the literature. These findings suggest that DNA/genomic encryption techniques can be considered as an alternative solution to cryptographic requirements.

Keywords: Genomic (DNA) encryption, Textual data, Siyasetname, AES, ECDH

1. Introduction

Cryptography has changed from past to present with different techniques depending on the conditions. It is one of the methods used to share data over unsecured communication networks [1-4]. Existing cryptographic systems, with their keyed (symmetric/asymmetric) or keyless structures, cannot be fully functional in some cases. Technological advances, the requirements of the digital age and the results of other situations direct the trend towards new solution alternatives [5-13]. One of these new fields is Deoxyribonucleic Acid (DNA) encryption techniques, inspired by its bio-molecular structure and designed with a hybrid of cryptographic algorithms [10-14]. The huge storage capacity of DNA in its natural form and the role it plays in transferring information from generation to generation have been a source of inspiration for cryptosystems. Advantages of DNA encryption techniques include multiple security options, large storage capacity and efficient memory usage. Additionally, extensive key generation and storage facilities are a significant plus. However, limitations such as high experimental costs, complex processing processes and lack of knowledge in biotechnology procedures are the disadvantages of these techniques. Due to these limitations, the actual application of DNA encryption techniques cannot go beyond the laboratory environment. In order to be used in the digital environment, conversion processes are carried out with a hybrid of existing cryptographic methods and mathematical coding techniques. These hybrid approaches aim to provide greater efficiency in practical applications. [10, 12-15].

The aim of this study is to test the suitability of cryptographic use of the genomic-based DNA (gDNA) encryption technique on Turkish text datasets of different sizes. In practice, among the classical/existing cryptographic algorithms, Advanced Encryption Standard (AES) was used symmetrically and Elliptic-Curve Diffie-Hellman (ECDH) was used asymmetrically. As a new generation cryptographic algorithm, the gDNA technique is in symmetric form. The performance of the DNA cryptographic technique in the coding and decoding processes was compared with existing classical algorithms and its use in cryptographic systems was evaluated. The second part of the article provides literature information of related (similar or close) studies. The third chapter is titled materials and methods; it covers the data set, cryptographic algorithms used and performance metrics. Additionally, this section is presented enriched with visuals. Fourth part; it includes graphical representation of experimental results with tabular content and discussion sections of these results. In addition, in this section, the contribution of the study and the information of similar or close studies are concretized in a different table. Thus, a more striking comparison was aimed. In the last section, the results are analyzed and their pros and cons are discussed. Suggestions are given with the aim of contributing to future studies. The highlights of the study are listed below.

• Coding and decoding of artificially produced textual data sets in different sizes from the content named Siyâsetnâme (Nizamülmülk) [16], which is a Turkish content data set,

• Evaluation of time (s), memory (MB) and CPU (%) consumption rates as performance metrics of symmetric (current: AES and new generation: DNA) and asymmetric (ECDH) algorithms used for the application,

• The detailed content of the DNA encryption algorithm, which is one of the new generation techniques in cryptosystem, is given to the Turkish literature. In addition, it is aimed to present not only research but also practical use.

2. Related Works

To date, significant progress has been made in the development of cryptographic algorithms. In this process, basic encryption methods such as single-alphabet ciphers, multi-alphabet substitution ciphers, transfer ciphers and block ciphers were initially developed [5-15]. Following the early techniques, more complex and powerful cryptographic algorithms such as AES, Data Encryption Standard (DES), Rivest-Shamir-Adleman (RSA) and Secure Hash Algorithm (SHA) have emerged [6-8]. Monoalphabet ciphers replace letters of plaintext with other letters using a specific alphabet. Multi-alphabet substitution ciphers perform a more complex substitution process by using more than one alphabet. Transfer ciphers provide security by reversing the letters of the text. Block ciphers are for higher security by encrypting text in blocks of specific sizes. Among these algorithms, AES has become a prominent standard among symmetric encryption algorithms. It offers superior features in terms of both performance and security. DES has been widely used before. However, due to its insufficient key length, more secure algorithms like AES eventually replaced it. RSA is one of the most well-known asymmetric encryption algorithms. It enabled secure data transmission using public and private key pairs. SHA is among the secure hashing algorithms. It is used to protect the integrity of data. These developments have led to the emergence of more robust and effective encryption methods in the field of information security. It has become one of the alternatives to better meet the security needs of the digital world. The evolution of cryptographic algorithms, together with constantly improving technology, has resulted in the design of more efficient information security solutions [4-15]. The foundation of DNA cryptography, one of these possibilities, was laid [17, 18]. It originated from the quest to find a new computational model to meet the requirements for large amounts of processing power and storage. The biological functioning of the DNA molecule inspired this new computational model. In particular, Adleman's [17] attempt to find a solution to the Hamilton Path Problem with DNA computation in 1994 aroused great interest in the field of DNA computation. Then, in 1995, researchers such as Lipton [18] published about DNA computers and (non-deterministic Polynomial) NP-complete problems. The timing problem of 2-bit complex numbers was solved in the test tube using DNA molecules. Success has been achieved with this method. DNA cryptography is inspired by the fact that data can be biologically encoded in DNA strands. This approach is seen as a new source of hope for unbreakable algorithms. Because the nature of DNA provides a very complex and secure storage environment. Therefore, DNA computation and cryptography may offer a new and effective approach to secure data. Based on the application of the article, some of the similar or close studies [19, 20, 22-29] conducted in the last five years are mentioned below.

Before uploading textual data to the cloud environment, a multi-layered symmetric key DNA cryptography technique has been proposed [19]. This study was proposed in 2018; it was compared with classical symmetric structured AES, DES and Blowfish algorithms. Performance criteria of the experimental application; it was evaluated in terms of efficiency, namely cipher text size, encryption time and transfer speed. It has been observed that the proposed technique is more successful. To ensure the confidentiality and security of the textual dataset, another algorithm based on asymmetric key DNA cryptography, consisting of three steps, was proposed in 2019. The result of the study [20] was compared with existing techniques. It has been tested with statistical tests according to the National Institute of Standards and Technology (NIST) [21] to analyze the security analysis and the randomness of the generated cipher text. Another study [22] in which DNA-based cryptography was applied on textual data type was published in 2020. Cryptography has been applied so that some of it is symmetrical and the other part is asymmetrical. First, the combination of DNA coding with algorithms called One Time Pad (OTP) was taken. Secondly, DNA codes in the RSA algorithm were applied. The efficiencies of DNA coding in OTP, RSA and other algorithms are given. As a result, it was seen that the calculation time of the RSA algorithm combined with DNA coding was longer. To solve this problem, data redundancy is reduced by enabling the GZIP compressed algorithm. Current experimental results show that DNA symmetric cryptography works quite well in both time and dimension analyses. Compressed DNA has shown to be less efficient than asymmetric cryptography. Another study [23] evaluating the performance of genetic (DNA) and classical (DES, AES and RSA) encryption algorithms is published in 2021. The application was made on textual data sets of different sizes (58 KB, 100 KB, 1 MB and 5 MB). It is encoded by generating 64 or 128 bit random keys. Additionally, an interface platform is provided for the coding process. Time, CPU and RAM were evaluated as performance metrics in encryption and decryption. It has been observed that symmetric algorithms give better results in large-sized data sets, and genetic encryption algorithms give better results in small-sized data sets. Another study [24] on textual datasets of different sizes is in 2021. Hybrid structured models based on symmetric (Blowfish and DES) and DNA-based encryption algorithms were used. Their performances were compared in terms of time, memory and processor complexity. General evaluations have been made for the application, which gives different result indicators according to different platforms. The current non-hybrid Blowfish algorithm has been shown to be more successful. However, it was found that it slowed down even more in terms of time after the hybrid was made with DNA-based techniques. It resulted in significant performance loss in the time display of the Blowfish algorithm on large data sets. However, the DES algorithm gave better performance on big data. In addition, DES and DNA hybrid algorithms have been shown to be better than studies in the literature. Study in 2022 [12]; it was about image, text and video data types. The process of transforming DNA as a carrier and a means of applying modern biological techniques has been carried out. The structure of the proposed DNA cryptography was created by integrating DNA operators into the Feistel network structure. The simulation software developed in the experimental application and the synthesized DNA content were integrated with biotechnical hardware. Evaluation criteria; the capacity is about 100%, the brute force attack is about 12 × 106 years, the key space is 2, and the entropy analysis is close to 2. It was interpreted as effective for cryptographic requirements and the results were confirmed by in vitro experiments. The study in 2023 [25] focused on DNA-based new generation cryptographic techniques. Similar or close studies in the literature are mentioned and the general process of biological structure is given. The process steps of multiple formats in the encoding and decoding processes are explained with detailed explanations. There is plain data (original form) consisting of a string of characters of textual type as words. A small bio-molecular based application has been made for the key index created by a hybrid of words and numbers. The encryption and decryption stages were implemented without any problems. It has been shown that the infrastructure can be adapted to storage studies along with cryptographic requirements. Another study in 2023 [26] is the application of different symmetric encryption algorithms on textual data type. The application again consists of data sets of different sizes. Performance metrics in comparing algorithms; CPU, RAM and time. When the results are analyzed, it has been observed that the performances of Blowfish, Salsa20, 3DES, Cast, AES and DES algorithms are better in all data sets. A similar study [27] was also conducted in 2023. The only difference from the previous source is the cryptographic application with asymmetric algorithms. Again, coding and decoding processes were carried out on textual data sets with different volumetric content. Performance criteria are again RAM, time and CPU. When performance analysis was performed, ECDH, El-Gamal and RSA rankings were obtained in all volumetric data sets. Finally, another different study [28] was conducted in 2023. This study was again carried out on textual data sets of different sizes. In the cryptographic process, symmetric (AES, Blowfish, Cast-128) and asymmetric (ECDH, El-Gamal, RSA) encryption algorithms have been implemented. Performance criteria were evaluated based on encryption and decryption speed (s), memory (MB) and CPU (%) usage rates. Thus, the performances of symmetric and asymmetric encryption algorithms were compared on the same data sets. As a result, it was observed that the performance results of symmetric encryption algorithms were better in all volumetric data sets. Based on the studies and gaps in the literature, the goal of this application is given below.

- As far as researched, DNA or other (RNA, amino acid, etc.) genomic-based cryptographic applications have not been used in converting a Turkish data set. Only character transformations have been mentioned in the studies in the literature. Additionally, it has not been tested on data sets containing Turkish characters of different sizes. For this reason, the study is the first to use Turkish content datasets of different sizes. It is aimed to present the proposed algorithm as one of the alternatives in cryptographic requirements.

3. Material and Method

Information called confidential or private may require additional processing in environments where there is a security risk. These transactions; this may include restricting access from unauthorized persons, minimizing or completely preventing dangerous situations. One of the methods used for this is cryptology. Cryptographic algorithms; the transaction is made with keyed (symmetric, asymmetric), keyless or hybrid components of the original data. Changing, or locking, this original data with the help of mathematical functions is called encryption. Transforming it to its original state, that is, opening the lock, with the help of reverse mathematical operations is decryption. Cryptographic systems, which have changed with different techniques according to conditions from past to present, are not fully functional in some cases. Technological breakthroughs, requirements of the digital age and other situations, etc. The results lead to new solution alternatives. This is one of the new areas; they are genomic-based (DNA, RNA, amino acid ...) encryption techniques inspired by the bio-molecular structure and designed with a hybrid of cryptographic algorithms [1-15]. In this study, which is one of the new generation cryptographic techniques, a DNA-based application with a symmetric key was made. This paper; it includes a comparative application of current and next generation cryptographic systems. The purpose of the application; it is a test of the suitability of the cryptographic use of the gDNA encryption technique in data sets of different sizes (100 and 500 KB, 1 and 5MB) artificially created from the Turkish content named "Siyasetname" in the textual data type. It has been observed that the gDNA encryption technique has not been extensively applied on a Turkish data set before. Artificial data sets with different volumetric contents were created. Comparison with existing/classical (AES (symmetric), ECDH (asymmetric) algorithms supported application results. Performance criteria; accepted criteria in the literature for textual data type were taken into account. These are the time (seconds), memory (Megabytes) and processor (%) usage rates in encoding and decoding. In Figure 1, the course of the study is visualized as the main flow chart. It is explained and detailed in the following sections.

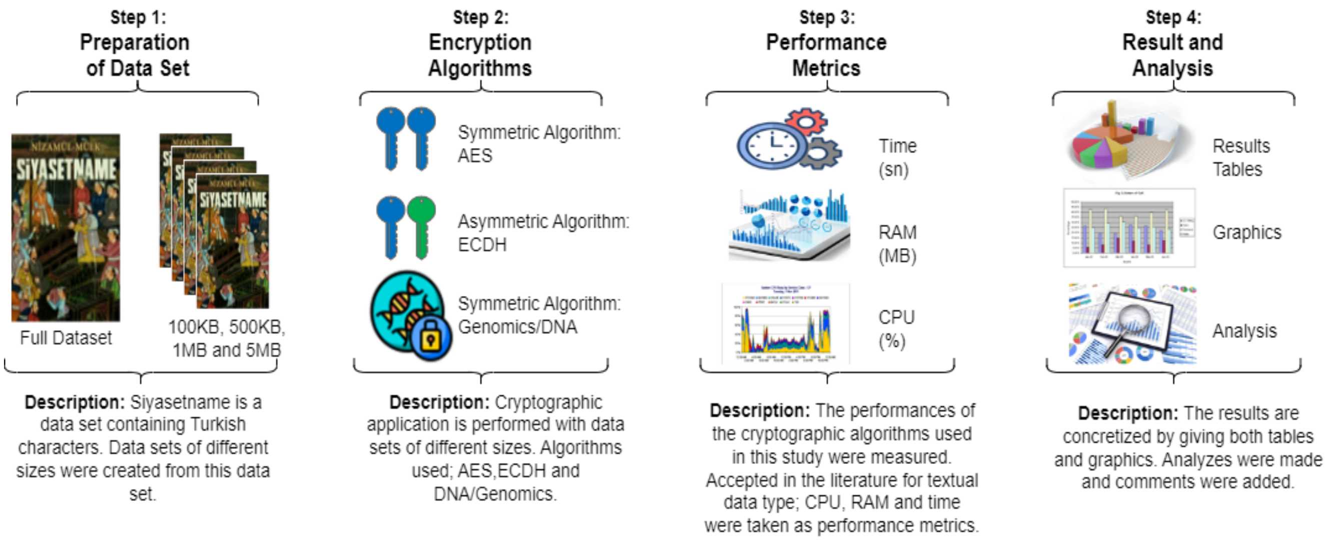

Step 1: Preparation of Data Set
Full Dataset
100KB, 500KB, 1MB and 5MB
Description: Siyasetname is a data set containing Turkish characters. Data sets of different sizes were created from this data set.
Step 2: Encryption Algorithms
Symmetric Algorithm: AES
Asymmetric Algorithm: ECDH
Symmetric Algorithm: Genomics/DNA
Description: Cryptographic application is performed with data sets of different sizes. Algorithms used; AES,ECDH and DNA/Genomics.
Step 3: Performance Metrics
Time (sn)
RAM (MB)
CPU (%)
Description: The performances of the cryptographic algorithms used in this study were measured. Accepted in the literature for textual data type; CPU, RAM and time were taken as performance metrics.
Step 4: Result and Analysis
Results Tables
Graphics
Analysis
Description: The results are concretized by giving both tables and graphics. Analyzes were made and comments were added.

Figure 1. Main Flow Chart of the Study

3.1. Data Set Preparation

In this study, a data set with Turkish content called "Siyasetname" or "Siyeru'l-mülk" [16] written by Nizamülmülk was used. Artificial data sets with four different volumetric contents were created from this original data set. The actual size of the original content is 1.60 MB. Volumes of artificially generated new datasets; it is between 100 KB, 500 KB, 1 MB and 5 MB. Data duplication was done for the 5 MB data set.

3.2. Encryption Algorithms

DNA-based coding and decoding application, one of the new generation cryptographic systems, has a symmetric key structure. The performance of this structure was evaluated by comparing it with existing/classical algorithms. AES was used as the classical existing symmetric encryption algorithm and ECDH was used as the asymmetric encryption algorithm. In particular, the selection of these two algorithms is based on three previous studies [26]-[28]. Because of these three studies, the symmetric and asymmetric encryption algorithms that gave the most successful outputs were included in this application. The purpose of these three studies is to determine the performance of classical encryption algorithms on a Turkish data set of different sizes. Because of the studies, the classical encryption (symmetric and asymmetric) algorithms that give the best performance are determined.

3.2.1. Figures Symmetric Encryption Algorithm: AES

A symmetric encryption system is designed by using the same public key to be secret during the encoding and decoding phases. Since only a single secret key is used in these algorithms, they are easier to break than asymmetric algorithms. However, since there is only one key (secret), the encryption and decryption time is advantageous compared to asymmetric or other hybrid structures. Therefore, it is faster. This also provides superiority over asymmetric structures in encrypting large data. It consumes less system resources (CPU, RAM, etc.). It works compatible with the hardware. The main disadvantage here is that all security depends on the key and its size. The key size can be created in longer directories than asymmetric ones [29]-[32].

AES, one of the popular symmetric encryption algorithms, is used in this application. The working architecture of the AES cryptographic system is visualized in Figure 2. In addition, the working logic of the AES algorithm is given in Algorithm 1, "Pseudo code of the AES algorithm". This pseudo code; it includes key generation, encryption process and decryption processes. AES was designed by NIST in 2001 to be a standard for encrypting digital data. It has a block encryption structure. Key lengths consist of 128 (10 rounds)/192 (12 rounds)/256 (14 rounds) bit options. However, the coding process is done with bytes instead of bits. It takes all data as input, that is, the original contents in 128 bits (16 bytes). It gives the encrypted form as output, again as 128-bit blocks. Each block evaluates as a grid of 4 row bytes x 4 column bytes = 128 (16 byte) bits in main column order. Manipulating and shuffling input data through a series of linked steps is known as the substitution-permutation network principle. Each round consists of 4 steps. The AES instruction set is now integrated into the CPU to increase the speed and security of applications that use AES for encryption and decryption. This offers a transfer speed of several GB/s. Although it has been 20 years since its launch, the AES algorithm, which is not possible even with current technology, has not been cracked. To date, the only vulnerability remains in the implementation of the algorithm [29]-[33].

```
Key Generation
Input: Key
Output: Round keys
Method: Generate the round keys by expanding the initial key.
Step 1. Check Key Length:
        o If the key length is not 128 bits, the algorithm returns an error.
Step 2. Initialization:
        o Convert the key into 4-byte words (32-bit).
        o Determine Nk, Nb, and Nr:
            ▪ Nk is the key length divided by 32 (4 words for 128-bit key).
            ▪ Nb is the number of columns in the state matrix (4).
            ▪ Nr is the number of rounds (10 for 128-bit key).
Step 3. Place Initial Keys:
        o Place each byte of the key into the round key matrix sequentially.
Step 4. Key Expansion:
        o Expand the remaining Nb * (Nr + 1) words:
            ▪ temp = round_keys[i - 1]
            ▪ If i is a multiple of Nk:
                ❖ temp = SubWord(RotWord(temp)) ^ Rcon[i / Nk]
            ▪ If Nk > 6 and i % Nk == 4:
                ❖ temp = SubWord(temp)
            ▪ round_keys[i] = round_keys[i - Nk] ^ temp
```

```
Encryption
Input: Plaintext
Output: Ciphertext
Method: Encrypt the plaintext using the AES algorithm.
Step 1. Initialize State:
        o Convert the plaintext into the initial state matrix.
Step 2. Add Initial Round Key:
        o Add the initial round key to the state matrix.
Step 3. Rounds (1 to 9):
        o SubBytes: Transform each byte in the state matrix using the S-Box.
        o ShiftRows: Shift each row in the state matrix to the left by the row number.
        o MixColumns: Multiply each column in the state matrix by a fixed matrix.
        o AddRoundKey: Add the corresponding round key to the state matrix.
Step 4. Final Round:
        o SubBytes: Transform each byte in the state matrix using the S-Box.
        o ShiftRows: Shift each row in the state matrix to the left by the row number.
        o AddRoundKey: Add the final round key to the state matrix.
Step 5. Output:
        o Extract the state matrix as the ciphertext.
```

```
Decryption
Input: Ciphertext
Output: Decrypted text
Method: Decrypt the ciphertext using the AES algorithm.
    Step 1.     Initialize State:
            o Convert the ciphertext into the initial state matrix.
    Step 2.     Add Final Round Key:
            o Add the final round key to the state matrix.
    Step 3.     Rounds (1 to 9):
            o InvShiftRows: Shift each row in the state matrix to the right by the row number.
            o InvSubBytes: Transform each byte in the state matrix using the inverse S-Box.
            o AddRoundKey: Add the corresponding round key to the state matrix.
            o InvMixColumns: Multiply each column in the state matrix by the inverse fixed matrix.
    Step 4.     Final Round:
            o InvShiftRows: Shift each row in the state matrix to the right by the row number.
            o InvSubBytes: Transform each byte in the state matrix using the inverse S-Box.
            o AddRoundKey: Add the initial round key to the state matrix.
    Step 5.     Output:
            o Extract the state matrix as the decrypted text.
```

Algorithm 1. Pseudo code of AES algorithm

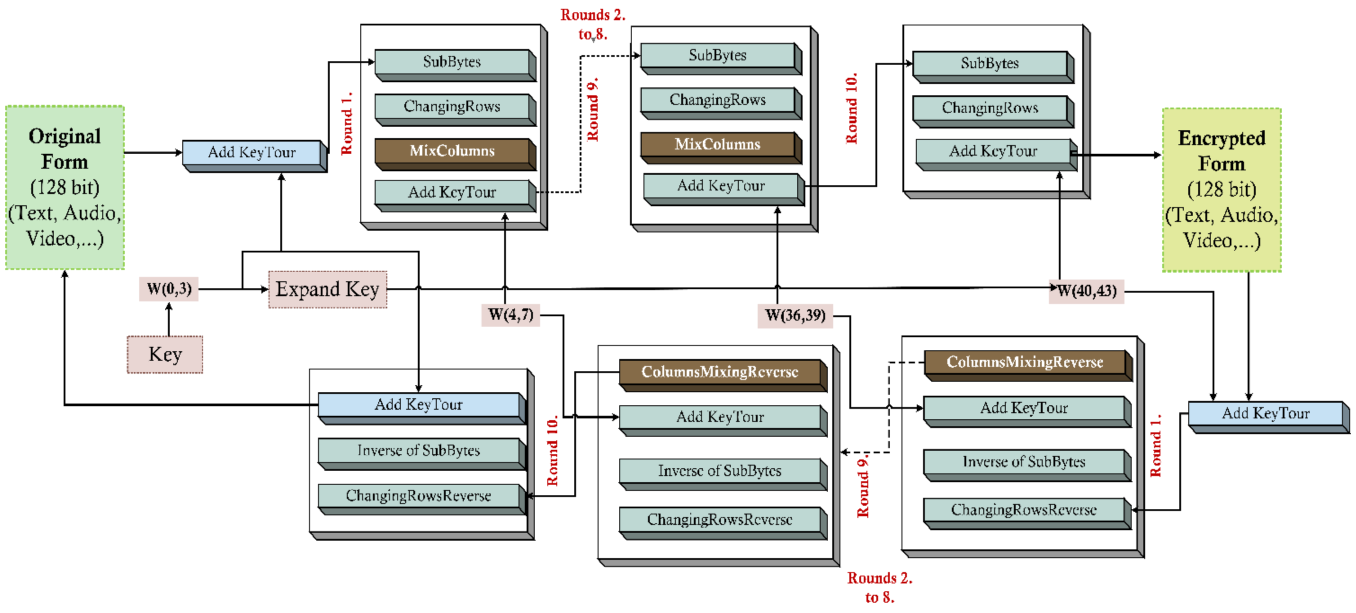


Figure 2. Working Architecture of the AES Cryptographic Algorithm

3.2.2. Asymmetric Encryption Algorithm: ECDH

Asymmetric encryption system forms the infrastructure of digital signatures. While encoding and decoding, two different keys called, secret and open are used. The public key is used in the encryption process. It is common between the parties (message sender-receiver). The secret key is used for decryption. It is special only in decryption. Since there is no single key structure, there is no need for frequent key changes as in symmetric systems. The reason is that the public key has no effect on the decryption phase. This situation also has another disadvantage. Encrypted contents are more difficult to crack than symmetric systems. They can also manage their processes with shorter length key sequences. However, due to the secret key that the parties cannot share, the decryption time is higher (slower) than in the symmetric structure. Additionally, the size of the file to be encrypted increases this situation exponentially. Another disadvantage is that it requires difficult mathematical functions to ensure data/information security. Because the calculation content includes large prime numbers. This consumes the processor and other system resources more than symmetric algorithms. It forces integration with hardware structures. For such reasons, data size is an important constraint in the encryption process.

In other words, it can be applied on lower dimensional data sets compared to symmetrical structures [29]- [32]. Asymmetric crypto systems are aimed at two main uses: asymmetric encryption and digital signatures. Digital signatures are a way to verify that any message comes from the owner of a specific private key and that the information has not been tampered with in transit. In this application, ECDH encryption algorithm, one of the popular asymmetric encryption algorithms, is used. The ECDH cryptographic system is visualized in Figure 3 under the name of its working architecture. In addition, the working logic of the ECDH algorithm is given in Algorithm 2, "Pseudo code of the ECDH algorithm". This pseudo code; it includes key generation, encryption process and decryption processes. It is based on the Diffie-Hellman (DH) algorithm, which dates back to 1976 [34] and is one of the first public key protocols. ECDH algorithm is known as an encryption algorithm in crypto systems. However, it is more commonly known as the agreement protocol for all transactions of the key up to a certain level. These transactions; how to produce, how to share between mutual parties, etc. Contains. The actual applications of the encryption process also depend on how these key operations are used. ECDH itself is a key exchange protocol. It is not designed for direct message encryption operations. ECDH is used to securely generate a shared key between parties. This shared key is then used to encode and decode the actual messages, often using it with a symmetric encryption algorithm (e.g. AES). Encryption process; it covers the protections made to prevent the third party named Oscar from gaining access while communicating between the parties named Alice and Bob. Elliptic algorithms perform operations on a curve. Alice and Bob agree on a curve with a starting point. Let us call this P. Alice has a private key a and a public key A = a * P. Bob has a private key b and public key B = b * P. Combining the equations; a * B = a * b * P = b * A. Thus a * b * P becomes the shared secret [35].

Key Generation

Input: *Private key*
Output: *Public key*
Method: *Generate the public key using the private key and elliptic curve parameters.*

Step 1. Check Private Key Length:

- *Ensure the private key is of the correct bit length according to the elliptic curve used.*

Step 2. Initialization:

- *Select the elliptic curve parameters (curve equation, base point G, order n).*

Step 3. Generate Public Key:

- *Alice and Bob each form a private key (a and b). These keys are usually random numbers and are kept secret during the transaction.*
- *Calculate the public keys:*
  - *Public key of Alice (Qa): $Qa=a\cdot G$*
  - *Public key of Bob (Qb): $Qb=b\cdot G$*
- *The result Qa and Qb are the public keys.*

Key Exchange

Input: *Private key (own), Public key (peer)*
Output: *Shared secret*
Method: *Generate the shared secret using the own private key and peer's public key.*

Step 1. Initialize:

- *Use your private key (a or b) and peer's public key (Qb or Qa).*

Step 2. Calculate Shared Secret:

- *Alice computes the shared private key (K) by using her private key (a) and Bob's public key (Qb):*
  - *$K=a\cdot Qb$*
- *Bob computes the same shared secret key (K) by using his private key (b) and Alice's public key (Qa):*
  - *$K=b\cdot Qa$*
- *The result K is the shared secret.*

Encryption

Input: *Plaintext, Shared secret*
Output: *Ciphertext*
Method: *Encrypt the plaintext using the shared secret.*

Step 1. Derive Symmetric Key:

- *Use a key derivation function (KDF) to derive a symmetric key K from the shared secret K, if necessary.*

Step 2. Encrypt Plaintext:

- *Alice creates the encrypted message (C) using the shared secret key (K) and the message (M):*
  - *$C=M+K$*

Step 3. Output:

- *The result is the ciphertext (C).*

Decryption

Input: *Ciphertext, Shared secret*
Output: *Decrypted text*
Method: *Decrypt the ciphertext using the shared secret.*

1. ***Derive Symmetric Key:***

- *Use the same key derivation function (KDF) to derive the symmetric key K from the shared secret K, if necessary.*

2. ***Decrypt Ciphertext:***
    - *Bob decrypts the encrypted message (M) using the same shared secret key (K) and encrypted message (C):*
        - $M=C-K$
3. ***Output:***
    - *The result is the decrypted text (M).*

Algorithm 2. Pseudo code of ECDH algorithm

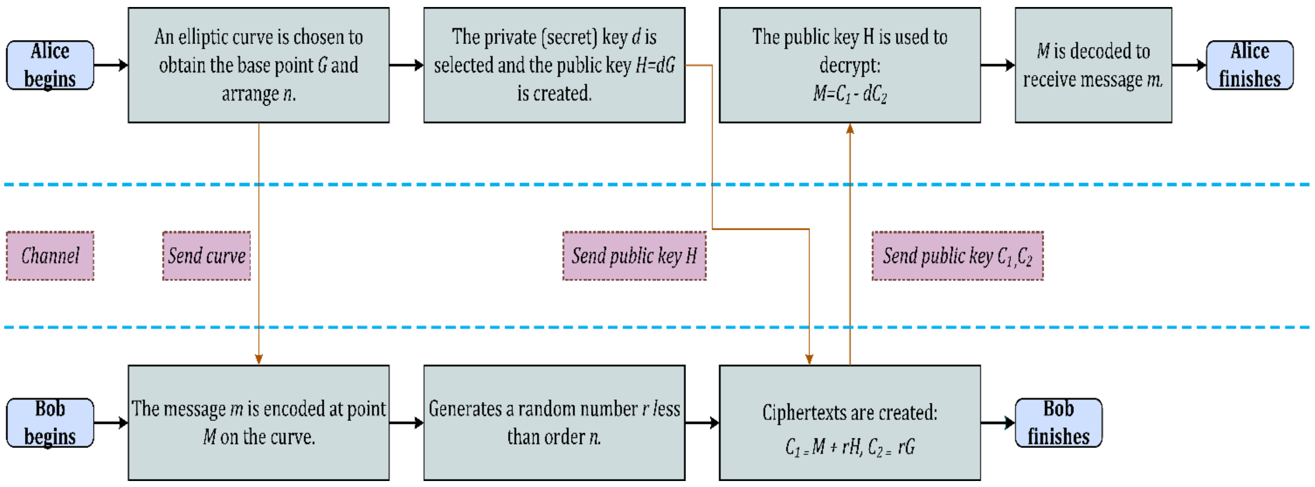

Alice begins
An elliptic curve is chosen to obtain the base point G and arrange n.
The private (secret) key d is selected and the public key H=dG is created.
The public key H is used to decrypt: $M=C_1 - dC_2$
M is decoded to receive message m.
Alice finishes
Channel
Send curve
Send public key H
Send public key C_1, C_2
Bob begins
The message m is encoded at point M on the curve.
Generates a random number r less than order n.
Ciphertexts are created: $C_1 = M + rH$, $C_2 = rG$
Bob finishes

Figure 3. Working Architecture of the ECDH Cryptographic Algorithm

3.2.3. New Generation Encryption Algorithm: DNA

One of the methods used to ensure information security is current/classic cryptographic algorithms. Crypto systems, which have been developed to date with different add-ons according to needs, can sometimes be insufficient or ineffective. One of the new methods proposed to find solutions to such problems is genomic-based crypto systems [1-5]. Genomics-based crypto systems are inspired by bio-molecular structures such as DNA, RNA and amino acids. These systems are inspired by natural functionality such as transcription and translation in living form during the encryption and decryption stages. Then, the inspired form tries to digitize it by transforming it with mathematical techniques. In other words, it plays the role of a hybrid implementation tool of living functionality with mathematical techniques. It carries out its role as a carrier medium through biological processes. Genomics-based algorithms as actors have major roles to play in cryptographic systems. Large-volume data storage and high-end security capabilities are a priority [1-4, 8, 9, 11-13]. However, biotechnology has disadvantages such as limited knowledge of procedures, complex structure of processes and cost of empirical environments. Due to such disadvantages, real applications are limited to the laboratory level. Based on this restriction, the bio-molecular structure is transformed by coding techniques. Thus, it is digitized with a hybrid of existing cryptographic algorithms [10, 12-15, 19].

3.2.3.1. Genomics-based DNA (gDNA) Encryption and Decryption Process

DNA is the structure that undertakes hereditary duties as the information carrier of life. A long biopolymer resembles the interlocking teeth of a zipper. These biopolymers consisting of sequences are called nucleotides. In the mutual chain of each zipper, there are nucleobases Adenine: A, Guanine: G, Cytosine: C and Thymine: T. These nucleobases are ordered by pairing A = T with G = C. These different sequential sequences (combinations) within each other are natural coding. It is in the form that stores and transmits information. This four-character (A, T, G, C) language form of life must be transformed into a language form (0 and 1) that digital systems can understand. This requires a hybrid of existing cryptographic systems and mathematical transformation techniques [10-15, 17-21, 22-25, 29, 30]. Additionally, a pseudo code explaining the encryption and decryption stages of the DNA algorithm is given in Algorithm 3. Its detailed explanation is given below.

```
Data Set Preparation

Input: Text data
Output: Binary representation of text data
Method: Convert text data to its binary representation.

Step 1.  Convert Characters to ASCII:
        o  Convert each character of the text data to its ASCII equivalent.
        o  If there are Turkish characters, use a format such as UTF-8 or UTF-16 for wide character set conversion.
Step 2.  Map to Binary Representation:
        o  Map ASCII or other numeric conversions to their binary representation counterparts.

Key Preparation

Input: None (or a predefined key)
Output: Symmetric key
Method: Generate a symmetric key for the encryption system.

Step 1.  Generate Symmetric Key:
        o  Generate a symmetric key (secret) for the encryption system.
        o  If you are not going to use a key, skip this step.

Key Expansion

Input: Symmetric key, Original data
Output: Expanded key
Method: Expand the binary format of the secret key to the size of the original data.

Step 1.  Expand Key:
        o  Expand the binary format of the secret key to match the size of the original data.

Encryption

Input: Binary representation of text data, Expanded key
Output: Encrypted form
Method: Encrypt the binary data using the expanded key and a suitable transformation.

Step 1.  Concatenate Data and Key using XOR:
        o  Concatenate the original data in binary form and the expanded key using the XOR technique.
Step 2.  Transform Binary Form:
        o  Transform the final combined binary form with A (00), G (01), C (10), and T (11) or another suitable
           transformation.
Step 3.  Save Encrypted Form:
        o  Save the encrypted form.

Decoding

Input: Encrypted form
Output: Original text data
Method: Decrypt the encrypted form to retrieve the original text data.

Step 1.  Convert Encrypted Data to Binary:
        o  Convert the recorded encrypted 00 (A), 01 (G), 10 (C), and 11 (T) data to the original binary structure.
Step 2.  Reverse XOR Operation:
        o  Obtain the original binary data by performing a reverse XOR operation using the expanded key.
Step 3.  Convert Binary to ASCII:
        o  Convert binary data to ASCII characters to get the original text data.
```

Algorithm 3 Pseudo code of DNA algorithm

The working steps are visualized in Figure 4 under the name "Working architecture of the DNA cryptographic algorithm". Detailed explanation is given below.

1. Each character in the data type in the original form (text, image, signal … etc.) is converted to its ASCII equivalent. If there are Turkish characters, use UTF-8 or UTF-16 for wider character set conversion, etc. is used.

Recommended in the study: The work named Siyâsetnâme (Nizamülmülk) [16], which is a data set containing Turkish characters, was used. Textual datasets of different sizes (100 KB, 500 KB, 1MB and 5MB) were created.

2. Translated ASCII or other numeric conversions (UTF, etc.) are matched to their binary base counterparts. The aim is to transform the digital form so that the digital form can understand it.
3. If a crypto system with a key (symmetric, asymmetric or hybrid) is designed, all operations applied in the first and second steps are applied exactly within the key. If it is a keyless structure, the key steps are skipped.

Recommended in the Study: A crypto system with a symmetric key was designed. The secret key used is the 32-character (256 bit = 32 byte) “1086_NizamülmülkSiyasetname_1092” index.

4. The key index in binary form is expanded by the size of the original data. The goal is to achieve compliance in the fifth step.
5. The original data in the binary structure and the extended key index in the binary structure are combined with the XOR [36, 37] technique.
6. This final combined binary form is completed by being converted into A (00), G (01), C (10) and T (11) structure, which is the language of life. The order of coding nucleobases can be changed. In the literature, this order is listed as eight basic coding rules [36-38]. This encoding order is irrelevant in the accepted performance metrics (time, RAM and CPU) for textual data type in encryption and decryption processes. The decryption process consists of the exact reverse process of these six steps. In the visual, the fifth work package explains these steps.

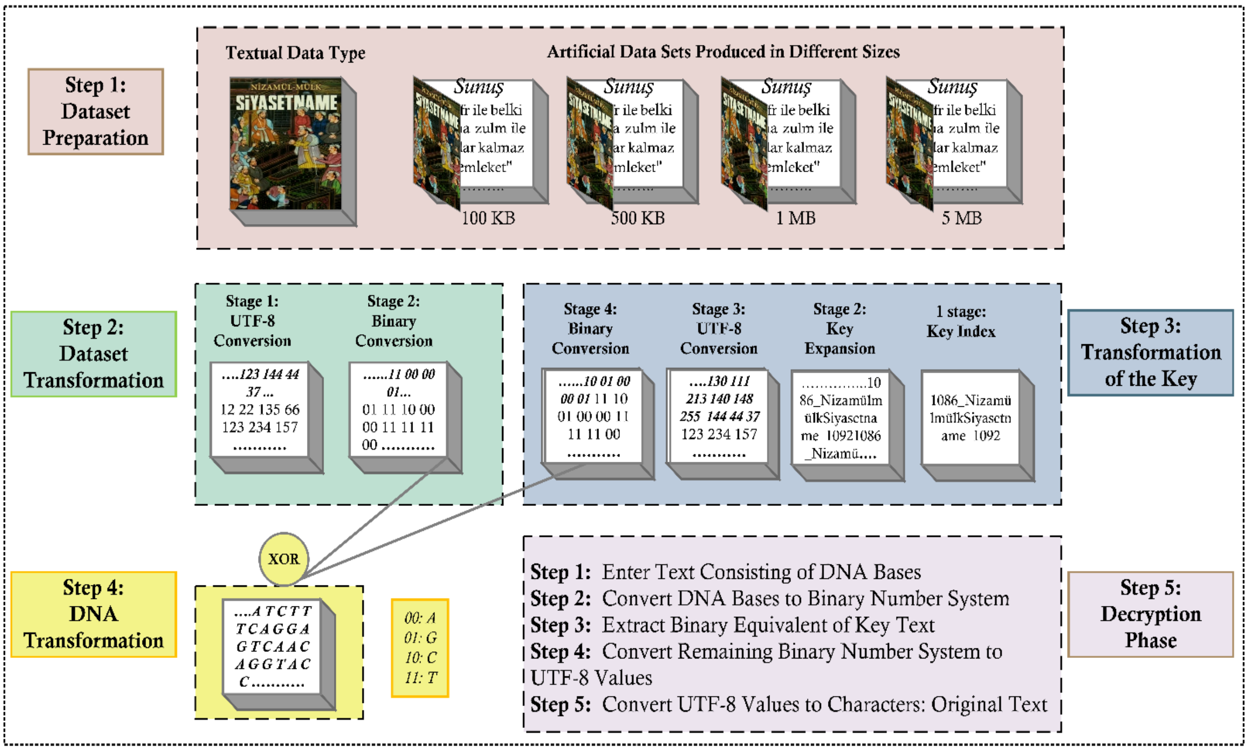

Step 1: Dataset Preparation
Textual Data Type
Artificial Data Sets Produced in Different Sizes
100 KB
500 KB
1 MB
5 MB
Step 2: Dataset Transformation
Stage 1: UTF-8 Conversion
Stage 2: Binary Conversion
Stage 4: Binary Conversion
Stage 3: UTF-8 Conversion
Stage 2: Key Expansion
1 stage: Key Index
Step 3: Transformation of the Key
XOR
Step 4: DNA Transformation
00: A
01: G
10: C
11: T
Step 1: Enter Text Consisting of DNA Bases
Step 2: Convert DNA Bases to Binary Number System
Step 3: Extract Binary Equivalent of Key Text
Step 4: Convert Remaining Binary Number System to UTF-8 Values
Step 5: Convert UTF-8 Values to Characters: Original Text
Step 5: Decryption Phase

Figure 4. Working Architecture of the DNA Cryptographic Algorithm

3.3. Performance Metrics

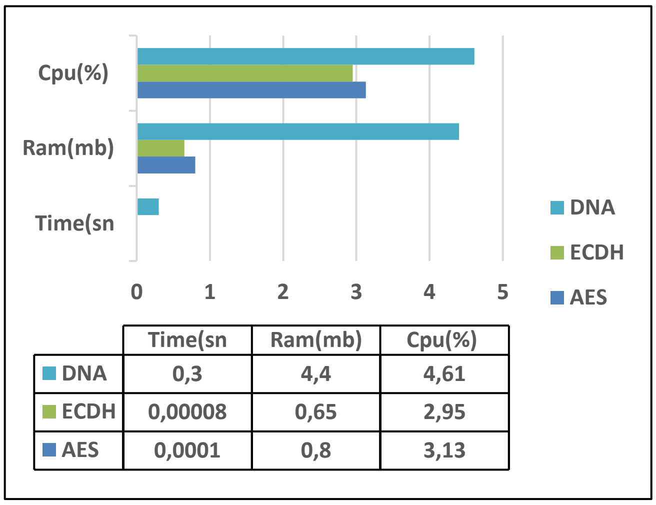
This work is for the application of gDNA encryption in symmetric form. It is aimed to evaluate the use of the DNA encryption algorithm in cryptographic requirements. For this reason, two classical algorithms were used. These algorithms; they are AES (symmetric) and ECDH (asymmetric). In the application on textual data type, criteria accepted in the literature were taken into account as performance metrics. These criteria; duration/processing time (seconds), processor (CPU :%) and memory/memory (RAM: Megabyte (MB)) consumption amount [23],[24]. Analysis of performance metrics “Hardware and software features of the computer that uses computing power in practice.” It was concluded according to the structure in Table 1. Consumption speeds of algorithms may vary depending on different computer features.

Table 1. Hardware and Software Features of the Computer Whose Computing Power is Used in Practice

Requirements	**Feature**
Processor	Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz (12 CPUs), ~2.6GHz
Memory	16384MB RAM
Operating System	Windows 11 Pro 64-bit
Software	Python 3.9.13, Vs Code 1.78.0.

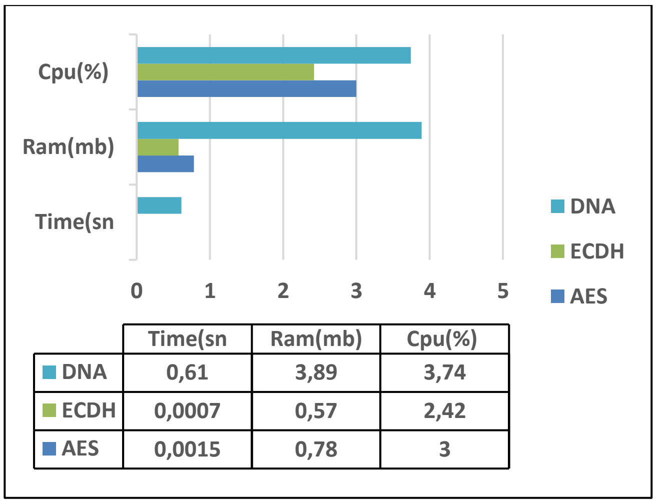
4. Experimental Results and Discussion

This study involves the application of gDNA cryptography on a textual Turkish content dataset. gDNA cryptography is accomplished with techniques inspired by bio-molecular processes. A (mixed) character sequence consisting of numbers and letters is used as a symmetric keying system. The secret key used is the 32-character (256 bit = 32 byte) index “1086_NizamülmülkSiyasetname_1092”. The purpose of the application is to test the suitability of using the gDNA encryption technique in cryptographic requirements. Four new data sets of different sizes were artificially created from the original data set called "Siyasetname". Sizes are between 100 and 500 KB, 1 and 5MB. Comparison with existing/classical (AES (symmetric), ECDH (asymmetric) algorithms support application results of DNA-based encryption techniques. Performance metrics are criteria accepted in the literature for the textual data type. These are time (seconds), memory (Megabytes) and processor (%) usage rates in encoding and decoding. The images from Figure 5 to Figure 12 show the outputs of the application, that is, the encoding and decoding results.



	Time(sn	Ram(mb)	Cpu(%)
DNA	0,3	4,4	4,61
ECDH	0,00008	0,65	2,95
AES	0,0001	0,8	3,13

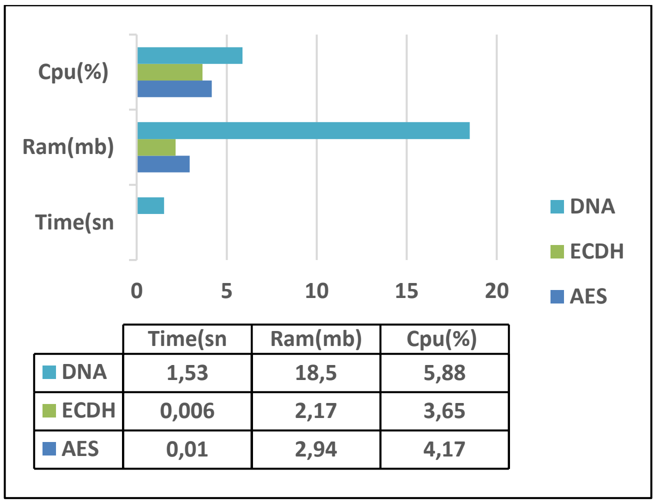
Figure 5. Encryption Performance Results of 100 KB Data Set



	Time(sn	Ram(mb)	Cpu(%)
DNA	0,61	3,89	3,74
ECDH	0,0007	0,57	2,42
AES	0,0015	0,78	3

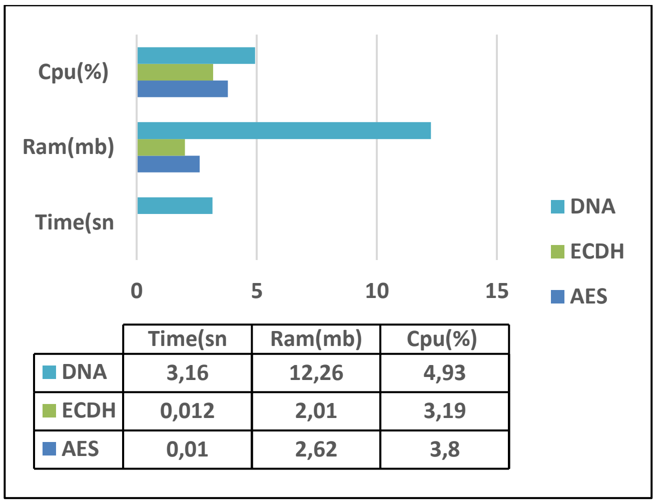
Figure 6. Decryption Performance Results of 100 KB Data Set

Figure 5 and Figure 6 are the encryption and decryption graphs of the 100 KB text dataset, respectively. The results in these two images, which appeared in a small data set, are close to each other. When the conversion of gDNA is calculated, the result of encryption and decryption is at an acceptable level. AES and ECDH algorithms work very fast and efficiently, especially on small data sets. Genomic (DNA) encryption, on the other hand, takes longer in terms of both coding and decoding time. Additionally, RAM and CPU usage is also higher. This is because DNA-based encryption requires more complex algorithms and data structures when processing biological data



	Time(sn	Ram(mb)	Cpu(%)
DNA	1,53	18,5	5,88
ECDH	0,006	2,17	3,65
AES	0,01	2,94	4,17

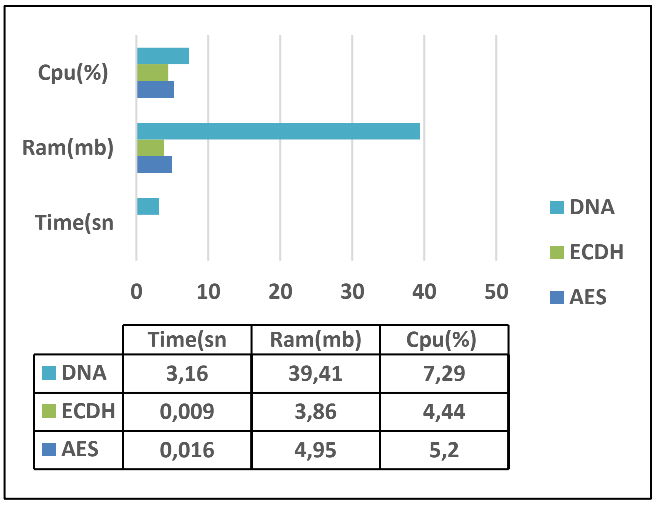
Figure 7. Encryption Performance Results of 500 KB Data Set



	Time(sn	Ram(mb)	Cpu(%)
DNA	3,16	12,26	4,93
ECDH	0,012	2,01	3,19
AES	0,01	2,62	3,8

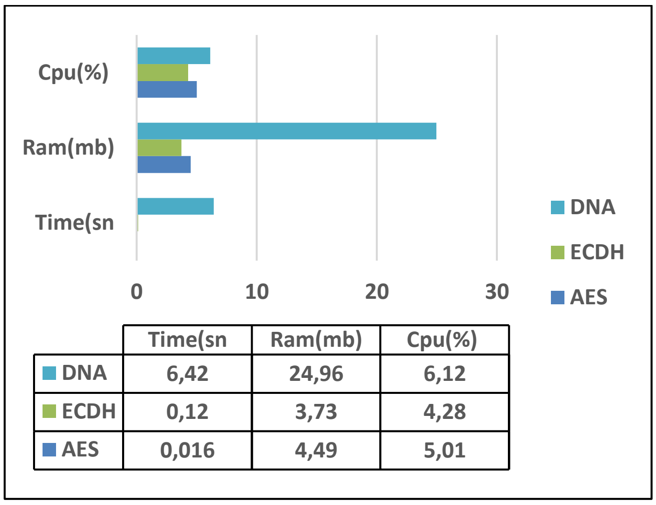
Figure 8. Decryption Performance Results of 500 KB Data Set

Figure 7 and Figure 8 are the encryption and decryption graphs of the 500 KB text dataset, respectively. In these two images, which appeared in a medium volume data set, the results gradually became clearer. Here again, when the conversion of gDNA is calculated, the result obtained in encryption and decryption is at an acceptable level. On medium-sized data sets, AES and ECDH algorithms remain fast and efficient. Genomic (DNA) encryption, on the other hand, takes longer and consumes more resources. DNA-based encryption algorithms cause processing time and resource consumption to increase as data grows.



	Time(sn	Ram(mb)	Cpu(%)
DNA	3,16	39,41	7,29
ECDH	0,009	3,86	4,44
AES	0,016	4,95	5,2

Figure 9. Encryption Performance Results of 1 MB Data Set

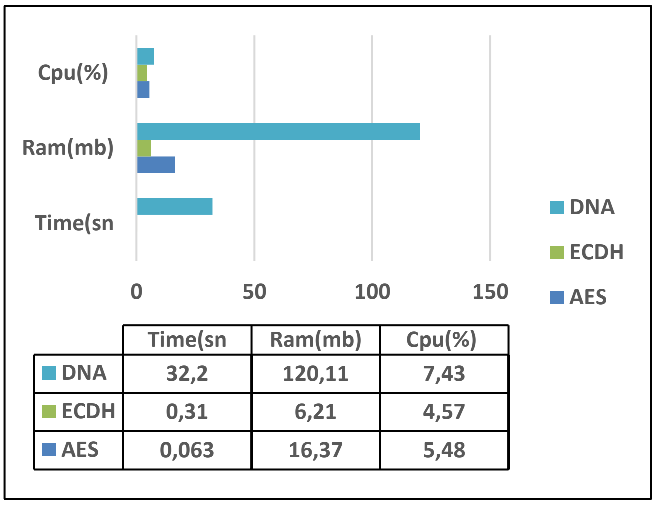


	Time(sn	Ram(mb)	Cpu(%)
DNA	6,42	24,96	6,12
ECDH	0,12	3,73	4,28
AES	0,016	4,49	5,01

Figure 10. Decryption Performance Results of 1 MB Data Set

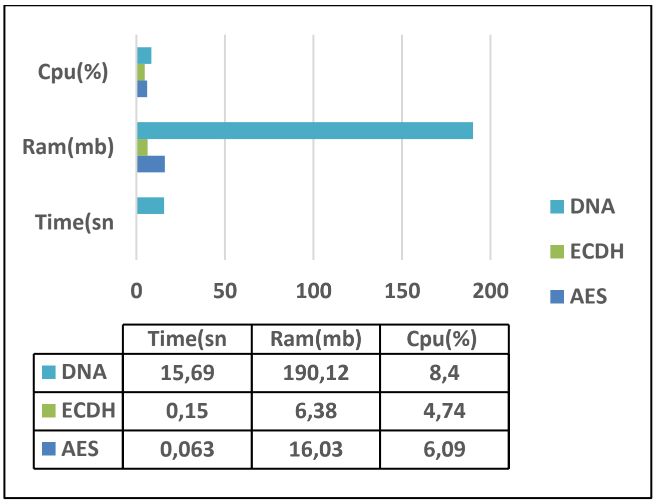
Figure 9 and Figure 10 are the encryption and decryption graphs of the 1 MB text dataset, respectively. In these two images, which appear in a large data set, the range of results is widened. On large data sets, AES and ECDH still work quite efficiently. Genomic (DNA) encryption, on the other hand, requires much more time and resources. As the data size increases, the computational complexity of DNA-based encryption and the resources required to process the data structures further increase.

Finally, Figure 11 and Figure 12 are the encryption and decryption graphs of the 5 MB text dataset, respectively. In these two images, which are another large data set, the range of results is widened. AES and ECDH algorithms also perform with acceptable times and resource usage on large data sets. gDNA encryption, on the other hand, works very slowly and resource intensive on large data sets. This is because DNA-based encryption requires complex biological data structures and algorithms.



	Time(sn	Ram(mb)	Cpu(%)
DNA	32,2	120,11	7,43
ECDH	0,31	6,21	4,57
AES	0,063	16,37	5,48

Figure 11. Encryption Performance Results of 5 MB Data Set



	Time(sn	Ram(mb)	Cpu(%)
DNA	15,69	190,12	8,4
ECDH	0,15	6,38	4,74
AES	0,063	16,03	6,09

Figure 12. Decryption Performance Results of 5 MB Data Set

Based on these results, the advantages of the DNA-based cryptographic algorithm are listed below.

- **High Security:** DNA-based encryption provides high security due to the complexity and uniqueness of biological data. This is an important advantage, especially in protecting biometric data.
- **Resistance:** DNA-based encryption may be more resistant to traditional cryptographic attacks. This can increase data security. It can be used in situations with special security requirements.
- **Originality:** DNA-based encryption can offer unique solutions regarding the processing of biotechnological and biometric data. This increases its usability in specialized areas such as medical data, genetic information or biological research.

In addition, the disadvantages of the DNA-based cryptographic algorithm are given below.

- **High Resource Usage:** DNA-based encryption requires higher resources in terms of both RAM and CPU usage. This difference becomes more evident especially in large data sets.
- **Long Processing Time:** The encoding and decoding time is much longer compared to algorithms such as AES and ECDH. This can be a significant disadvantage in large data sets.
- **Complexity:** DNA-based encryption algorithms require more complex data structures and algorithms when processing biological data. This can create additional challenges in terms of implementation and management.

AES and ECDH algorithms are very advantageous, especially in large data sets, with their high efficiency, low resource usage and fast processing times. However, DNA-based encryption can be preferred in certain cases with its high security, authenticity and advantages in processing biometric data. For small to medium-sized data sets, gDNA encryption may be appropriate. It can be used in situations where security and authenticity requirements are high. In large data sets, performance and resource usage should be taken into consideration. More efficient algorithms such as AES or ECDH may be preferred. The advantages of DNA-based encryption may be important for certain security and biotechnological requirements. However, in general use, careful evaluation should be made in terms of performance and efficiency.

The appearance may differ when this application is run on different platforms. Therefore, it is possible to show different results depending on the factors. Some of these factors; It is the type of file, its volume, complexity, keyed or non-keyed structure, or software/hardware platform.

5. Result and Discussion

In this study, the textual data type called Siyâsetnâme (Nizamülmülk), which is a data set containing Turkish characters, was used. The original size of the dataset is 1.6 MB. Artificial data sets with different volumetric content were created from this data set. Sizes of the created data sets; they are 100 KB, 500 KB, 1 MB and 5 MB. Volumetric replication was performed for the 5 MB data set. Symmetric-key gDNA encryption was implemented on these artificial data sets. The secret key used is the 32-character (256 bit = 32 byte) “1086_NizamülmülkSiyasetname_1092” index in symmetric form. The purpose of the application is to test the suitability of using the gDNA encryption technique in cryptographic processes. For this reason, a gDNA crypto system has been designed. To determine the suitability of the proposed cryptosystem, existing/classical encryption algorithms were compared. AES, one of the symmetric algorithms, and ECDH, were used as the asymmetric encryption algorithm. The selection of these two classical algorithms is a reference to their most successful outcomes in previous studies. Performance metrics used for the application; time (s), memory (MB) and CPU (%) consumption rates. These criteria are widely accepted in the literature for cryptographic processes of textual data. When the results of the application were analyzed, the results of the gDNA encryption technique were evaluated in 4 different data sets. When performance was compared with existing algorithms, it was seen that there was not much difference in small and medium volume data sets. However, it has been confirmed by the results that the range widens significantly in large volume data sets. In addition, it has been observed that the cryptosystem we propose gives the best results from similar or close studies in the literature. In conclusion, gDNA encryption techniques can be considered as one of the alternatives that can be used in cryptographic requirements. Future research may be directed towards accelerating the algorithms of genomics-based cryptosystems. Studies can be carried out to improve the algorithms of genomic cryptosystems in large volume data sets. Applications can be focused on video, images, signals or multi-format data sets, not just textual data types. Especially symmetrical, asymmetrical or etc. Hybrid studies of coding techniques may be at the forefront. Hybrid artificial intelligence architectures can be added, especially in the optimization part of genomic-based algorithms. It can be used to improve performance not only in cryptosystem applications but also in stages such as receiving, storing, sharing and storing data.

References

[1] Md M.A. Aziz, Md N. Sadat, D. Alhadidi, S. Wang, X. Jiang, C.L. Brown, N. Mohammed, "Privacy-preserving techniques of genomic data—a survey", *Briefings in Bioinformatics, vol. 20, 887–895,* 2019. https://doi.org/10.1093/bib/bbx139

[2] L. Bonomi, Y. Huang &, L. Ohno-Machado, "Privacy challenges and research opportunities for genomic data sharing", *Nat Genet 52, 646–654,* 2020. https://doi.org/10.1038/s41588-020-0651-0

[3] Z. Wan, J.W. Hazel, E.W. Clayton, et al., "Sociotechnical safeguards for genomic data privacy", *Nat Rev Genet 23, 429–445*, 2022. https://doi.org/10.1038/s41576-022-00455-y

[4] Y. Jiang, T. Shang, & J. Liu, "SM algorithms-based encryption scheme for large genomic data files", *Digital Communications and Networks, 7(4), 543-550,* 2021. https://doi.org/10.1016/j.dcan.2020.12.004

[5] H. Feistel, "Cryptography and computer privacy", *Scientific american, 228(5), 15-23,* 1973. https://www.jstor.org/stable/24923044

[6] G.J. Simmons, "Symmetric and asymmetric encryption", *ACM Computing Surveys (CSUR), 11(4), 305-330,* 1979. https://doi.org/10.1145/356789.356793

[7] P. Mahajan & A. Sachdeva, "A study of encryption algorithms AES, DES and RSA for security", *Global journal of computer science and technology, 13(15), 15-22,* 2013.

[8] H. Handschuh, L.R. Knudsen. M.J. Robshaw, 2001, "Analysis of SHA-1 in encryption mode*", In Cryptographers' Track at the RSA Conference, 70-83,* 2001. https://doi.org/10.1007/3-540-45353-9_7

[9] M. Kantarcioglu, W. Jiang, Y. Liu, B. Malin, "A cryptographic approach to securely share and query genomic sequence", *IEEE Transactions on information technology in biomedicine, 12(5), 606-617,* 2008. https://doi.org/10.1109/TITB.2007.908465

[10] S. Kalsi, H. Kaur & V. Chang, "DNA cryptography and deep learning using genetic algorithm with NW algorithm for key generation", *Journal of medical systems, 42, 1-12,* 2018. https://doi.org/10.1007/s10916-017-0851-z

[11] S. Basu, M. Karuppiah, M. Nasipuri, A. K. Halder, N. Radhakrishnan, "Bio-inspired cryptosystem with DNA cryptography and neural networks", *Journal of Systems Architecture, 94, 24-31,* 2019. https://doi.org/10.1016/j.sysarc.2019.02.005

[12] E. Şatir & O. Kendirli, "A symmetric DNA encryption process with a biotechnical hardware", *Journal of King Saud University-Science, 34(3),* 2018. https://doi.org/10.1016/j.jksus.2022.101838

[13] S. Namasudra, D. Devi, S. Kadry, R. Sundarasekar & A. Shanthini, "Towards DNA based data security in the cloud computing environment", *Computer Communications, 151, 539-547,* 2020. https://doi.org/10.1016/j.comcom.2019.12.041

[14] GZ Cui, "New direction of Data storage: DNA molecular storage technology", *Computer Engineering and Applications, vol. 42, 29-32,* 2006.

[15] S. Pramanik, S.K. Setua, "DNA cryptography", *2012 7th International Conference on Electrical and Computer Engineering, Dhaka, Bangladesh, 551-554,* 2012.https://doi.org/10.1109/ICECE.2012.6471609

[16] Türkiye Yazarlar Birliği (TYB). "nizamulmulk.pdf - TYB KİTAP". https://kitap.tyb.org.tr/kitap/nizamulmulk.pdf (Access Date: 29.10.2023).

[17] L M. Adleman, "Molecular computation of solutions to combinatorial problems", *Science, 266(5187), 1021-1024,* 1991. https://doi.org/10.1126/science.7973651

[18] R.J. Lipton, "DNA solution of hard computational problems", S*cience, 268(5210), 542-545,* 1995. https://doi.org/10.1126/science.7725098

[19] M. Sohal, S. Sharma, "BDNA-A DNA inspired symmetric key cryptographic technique to secure cloud computing", *Journal of King Saud University-Computer and Information Sciences, 34(1), 1417-1425,* 2018. https://doi.org/10.1016/j.jksuci.2018.09.024

[20] M.R. Biswas, K.M.R. Alam, S. Tamura, Y. Morimoto, "A technique for DNA cryptography based on dynamic mechanisms", *Journal of Information Security and Applications, 48, 102363,* 2019. https://doi.org/10.1016/j.jisa.2019.102363

[21] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, S. Vo, "A statistical test suite for random and pseudorandom number generators for cryptographic applications". *US Department of Commerce, Technology*

Administration, National Institute of Standards and Technology, vol(21), 2001. https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906762

[22] B.T. Hammad, A.M. Sagheer, I.T.Ahmed, N. Jamil,"A comparative review on symmetric and asymmetric DNA-based cryptography", *Bulletin of Electrical Engineering and Informatics, 9(6), 2484-2491,* 2020. https://doi.org/10.11591/eei.v9i6.2470

[23] M. Zengin, "Genetik kod yöntemiyle kriptoloji uygulaması", Yüksek Lisans Tezi (MSc Thesis), Karabük Üniversitesi, Karabük, Türkiye, 2021.

[24] F. Talo, "DNA tabanlı kriptoloji uygulaması", Yüksek Lisans Tezi (MSc Thesis), Düzce Üniversitesi, Düzce, Türkiye, 2021.

[25] A. Kaya, İ. Türkoğlu, "Yeni nesil güvenlik sistemleri: biyo-ilhamlı kriptografi", *2. International Uludağ Scientific Researches Congress, Bursa/Türkiye, 419-430,* 4-5 Kasım 2023.

[26] A. Kaya, İ. Türkoğlu, "Evaluation of symmetric cryptography algorithms in terms of performance analysis", *Cukurova 10th Internatıonal Scıentıfic Researches Conference, Adana/Türkiye, 4048-462,* 2-4 April 2023.

[27] A. Kaya, İ. Türkoğlu, "Evaluation of asymmetric cryptography algorithms in terms of performance analysis", *4. International Cappadocia Scientific Research Congress, Nevşehir-/Türkiye, 1056-1070,* 16-17 April 2023.

[28] A. Kaya, İ. Türkoğlu, "Simetrik ve asimetrik şifreleme algoritmalarının performans karşılaştırılması", *Fırat Üniversitesi Müh. Bil. Dergisi, 35(2), 891-900,* 2023. https://doi.org/10.35234/fumbd.1296228

[29] R.M. Indrasena, K.A.P. Siva, R.K. Subba, "A secured cryptographic system based on dna and a hybrid key generation approach", *Biosystems, 197: 1-10,* 2020. https://doi.org/10.1016/j.biosystems.2020.104207

[30] G. Rahman, C.C. Wen, "Omega network pseudorandom key generation based on dna cryptography", *Applied Sciences, 12(16), 1-19,* 2022. https://doi.org/10.3390/app12168141

[31] SSL2, "Simetrik ve asimetrik şifreleme – farklar nelerdir?. https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences, (Access Date: 10.12.2023).

[32] H. Kodaz, F.M. Botsalı "Simetrik ve asimetrik şifreleme algoritmalarının karşılaştırılması", *Selçuk-Teknik Dergisi 9(1), 10-23,* 2010.

[33] Geeksforgeeks, "Advanced encryption standard (AES)" https://www.geeksforgeeks.org/advanced-encryption-standard-aes/ , (Access Date:10.11.2023).

[34] W. Diffie and M. Hellman, "New directions in cryptography", *in IEEE Transactions on Information Theory, vol. 22, no. 6,644-654,* 1976. https://doi.org/10.1145/3549993.3550007

[35] G.N. Krishnamurthy, V. Ramaswamy, "Encryption quality analysis and security evaluation of cast-128 algorithm and its modified version using digital images", *International Journal of Network Security & Its Applications, 1(1):28-33,* 2009. https://doi.org/10.48550/arXiv.1004.0571

[36] H. Wen, S. Yu & J. Lü, "Breaking an image encryption algorithm based on DNA encoding and spatiotemporal chaos", *Entropy, 21(3), 246,* 2019. https://doi.org/10.3390/e21030246

[37] M. Şahin, "Memristor-based hyperchaotic system and DNA encoding based image encryption application on LabVIEW", *Uluslararası Muhendislik Arastirma ve Gelistirme Dergisi, 15. 269-276,* 2023. https://doi.org/10.29137/umagd.1239725

[38] A. Arı"CDIEA: Chaos and dna based image encryption algorithm", *Turkish Journal of Science & Technology, 18(1), 261-273,* 2023. https://doi.org/10.55525/tjst.1250419

Author(s) Contributions

Alev Kaya: Methodology, software, writing, editing, material preparation, data collection and analysis
İbrahim Türkoğlu: Supervision, writing, reviewing, editing

Acknowledgments

Firat University Scientific Research Projects Coordination Unit (FÜBAP) supported this study with project number ADEP.22.06.

Conflict of Interest Notice

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Ethical Approval and Informed Consent

It is declared that during the preparation process of this study, scientific and ethical principles were followed, and all the studies benefited from are stated in the bibliography.

Availability of data and material

[16] https://kitap.tyb.org.tr/kitap/nizamulmulk.pdf

Plagiarism Statement

This article has been scanned by iThenticate ™.