RESEARCH ARTICLE OPEN ACCESS

An Advanced Hybrid Cryptographic Framework for Secure Cloud Data Storage Using DNA Cryptography and Chaos-Controlled ChaCha20

Vineshraj S [1] Dr P Ebby Darney [2]

[1]Research Scholar – LIPS Research Advanced R & D European international University Paris
[2]Research Supervisor - LIPS Research Advanced R & D European international University Paris
[2]Professor – RajaRajeswari College of Engineering, Bangalore, Karnataka

ABSTRACT

Cloud computing has emerged as a critical technology of storing and managing large amounts of sensitive textual data, but at the same time, there are serious security issues as it can be accessed by an unauthorized individual and may be attacked by cyber-criminals. Although useful, traditional encryption techniques frequently have trade-offs between their computational efficiency and their level of security and thus cannot be used in large-scale cloud storage. In order to overcome these drawbacks, this paper proposes a new Hyper-Chaotic DNA-Assisted ChaCha (HCDNA-ChaCha) Encryption Framework that combines DNA-based encoding, hyper-chaotic permutation, and chaos-controlled ChaCha20 stream cipher in an effort to provide robust, multi-layer data protection. It works by encoding textual material into DNA sequences, hyper-chaotically permuting these sequences with hyper-chaotic maps to make them highly sensitive and unpredictable, and then encrypting the permuted material with a dynamically controlled ChaCha20 cipher to make them better diffusing and key sensitive. The evaluation results prove that the proposed framework is more efficient than existing approaches. The findings suggest that HCDNA-ChaCha offers the best trade-off between high security and low computation cost and is therefore a good and implementable solution to secure textual data storage in the cloud.

Keywords: *Cloud Security, Data Encryption, DNA Cryptography, ChaCha20, Hyper-Chaotic Permutation, Secure Cloud Storage.*

1. INTRODUCTION

Cloud computing has become a core technology of a contemporary information system, providing scalable data storage, on-demand services and economic data management solutions [1] [2] [3]. Healthcare, finance, education, government, and other organizations are beginning to depend on cloud platforms to store and process sensitive data [4] [5]. Although there are these benefits, cloud computing environment experiences recurring and emerging security risks. Cloud-stored data is likely to be stored on third-party systems that present cloud data with risks of unauthorized access, insider threats, data leakage, and loss of cryptographic keys [6] [7] [8]. Can data stored in the clouds be confidential, intact and available thus is a very critical research issue.

The most common data protection tool in the cloud setting is cryptography [9]. Older cloud security implementations have tended to use the existing encryption algorithms of Advanced Encryption Standard (AES), Rivest-Shamir-Adleman (RSA) or Elliptic Curve Cryptography (ECC) [10] [11] [12]. Although the techniques are very robust in mathematical security, many of them also depend on static keys, deterministic parameter configurations, and one-layer encryption designs [13]. These features can decrease resistance to sophisticated cryptanalytic assaults, statistical evaluation, replay assaults, and exposing key scenarios [14]. In addition, the increasing amount of cloud data requires the encryption system that can be not only safe but computationally efficient and software-based scaling [15].

In order to deal with these constraints, there are recent studies which have identified hybrid cryptographic solutions using classical encryption, bio-inspired methods, and chaos-based cryptographic solutions [16]. The goal of these approaches is to add of another layer of randomness, nonlinearity, and flexibility to the encryption process. DNA cryptography has been considered a promising paradigm of data obfuscation. DNA cryptography, inspired by the biological DNA sequences [17], encodes digital data with the nucleotide symbols allowing high key spaces and versatile transformations on the data [18]. DNA-based operations and encoding like complement, mutation, and substitution can greatly mask plaintext structures making cryptanalysis harder [19]. Nonetheless, DNA cryptography is not usually a full-fledged encryption solution but is best applied with strong cryptographic primitives.

Cryptographic applications Chaotic systems are also of widespread study because of its sensitivity to initial conditions, ergodicity, and pseudo-randomness properties [20]. Chaos-based encryption schemes exploit these properties to produce random numbers to permutate the data and generate keys [21]. Very basic chaotic maps are logistic maps, which have been widely employed, but

they usually have limited key space and can be attacked by parameter estimation [22]. By contrast, hyper-chaotic systems which are higher dimensionality and positive multiple Lyapunov exponents, provide much more complexity and chaos [23]. These systems find application especially when permutation and diffusion operations are to be performed and high data scrambling is needed. Simultaneously, stream ciphers were described as an effective substitute of block ciphers in software-based systems [24]. One of them, ChaCha20, has become very popular because of its good performance, immunity to timing attacks, and ability to execute on general-purpose processors [25]. Yet, regular ChaCha20 is based on the fixed keys and nonces that are given by the system or user. Inappropriate nonce handling or reuse of keys may cause critical security threats among them keystream reuse and replay attacks [26].

This study is inspired by these observations and, therefore, it suggests a progressive hybrid cryptographic model of secure data storage in clouds, which incorporates the DNA-inspired encoding model, hyper-chaotic permutation, and the chaos-controlled ChaCha20 (CC-ChaCha20) encryption. The suggested design follows a multi-layered approach to security, and every layer covers certain inefficiencies of traditional cloud encryption models. Instead of utilizing the mathematical encryption, the framework is a mixture of the biological-inspired obfuscation, nonlinear chaotic diffusion, and efficient stream cipher encryption to deliver a higher level of confidentiality and resilience. The key objective of this study is to design a secure and effective cryptographic framework for cloud data storage by combining DNA cryptography, hyper-chaotic permutation, and chaos-controlled ChaCha20 encryption to achieve higher data confidentiality, and improved security performance. The key contributions of this study are summarized as follows:

- The study proposes Hyper-Chaotic DNA-Assisted ChaCha (HCDNA-ChaCha) framework, a multi-layer encryption architecture that incorporates DNA-based encoding with hyper-chaotic permutation with chaos-controlled ChaCha20 stream cipher. This hybrid architecture makes sure that confusion and diffusion are further promoted to different levels and the resistance of textual data.
- The framework uses hyper-chaotic map-based mechanism is utilized to produce pseudo-random sequences of DNA-level permutation. The sensitivity to initial conditions and nonlinear dynamics of the hyper-chaotic system is extremely sensitive, which results in a special diffusion layer increasing the ciphertext randomness, key-dependency, and cryptanalysis is incredibly hard.
- The framework is an expansion of the standard ChaCha20 stream cipher adding chaos-managed key, nonce, and counter generation. This scheme dynamically adjusts the encryption parameters with the help of hyper-chaotic outputs in order to eliminate the re-use of keystreams, increases key sensitivity, and resistance to replay and related-key attacks without compromising encryption performance.
- The study presents a strong system of reassembling the permuted sequences of DNA to binary sequence prior to ChaCha20 encryption. This guarantees complete compatibility with the existing stream cipher operations with the security benefit of the DNA-based transformations.
- To best of our knowledge this is the first study to integrate DNA-based encoding, hyper-chaotic permutations, and chaos-modulated ChaCha20 into a single system of cloud text data encryption.

The rest of this paper is arranged in the following way: Section 2 presents literature review on the topic of cloud data encryption. Section 3 outlines HCDNA-ChaCha scheme, with all steps of encryption and workflow. Section 4 describes the experimental model, data and performance measures and reports about the results and performance analysis, and the comparisons with existing methods. Section 5 summarizes the paper and presents the future research directions.

2. RELATED WORKS

With the increased popularity of cloud computing, a lot of emphasis is being placed on ensuring the confidentiality, integrity, and secure accessibility of data. Various cryptographic solutions have been introduced to ensure the confidentiality and safety of data in cloud computing; each solution has been found to have some limitations. Bertrand et al. [27] constructed a hybrid encryption model with the purpose of ensuring the security of cloud files storage in order to provide access to the data only to authorized users. They added symmetric and asymmetric encryption methods such as AES and RSA to the system to capitalize on the benefits of each. Rapid Application Development (RAD) approach was used to enable the flexibility in the development of the system. Although the hybrid AES-RSA scheme enhanced access control and security, it was built upon traditional cryptographic primitives where parameters were non-reconfigurable, and was therefore susceptible to the dynamic cryptanalytic attacks. Awadh et al. [28] are the first to suggest the application of Triple

Data Encryption Standard (3DES) to secure privacy of data in the cloud computing setting, especially in healthcare. The proposed method would enhance the security against the brute-force attack by increasing the key length of the initial DES. Regardless of the fact that 3DES is more confidential than DES, it does not have the potential to be used in contemporary massive cloud systems because of the high computational cost and its non-modern design. Ahmad et al. [29] proposed a safe storage mechanism of data using the convergent encryption (CE) in conjunction with data deduplication. Here, the keys of encryption are based on the information directly, allowing deduplication of encrypted information to be done safely. This approach is much more efficient in storage and decreases redundancy in the cloud. Nevertheless, the generation of keys depending on the content of the data can make the system susceptible to confirmation-of-file attacks and provides no extra obfuscation.

Abdo et al., [30] concerned about the effectiveness of cloud data transmission and its security, gave the attention to compression and encryption methods. Their method used compression algorithm LZMA and after that several rounds of a strong encryption algorithm to downsize the data and make it impossible to be accessed by an unauthorized person. Even though the approach has the added benefit of improving the efficiency and security of transmission, it adds complexity to the system, and relies on conventional encryption systems. Gadde et al. [31] suggested a privacy preservation model of multimedia cloud data through optimization technology where an Improved Blowfish encryption algorithm is used. This work is new in the sense of optimal choice of the key by the means of hybrid metaheuristic algorithm named Clan-based Crow Search with Adaptive Awareness Probability (CCS-AAP). Although maximum-security is enhanced by optimum generation of key, the use of only one encryption algorithm can restrict the level of resistance against advanced cryptanalytic and statistical methods. Shrivastava et al. [32] introduced a hybrid cryptographic model of the secure cloud storage, which combines elliptic curve cryptography (ECC) and ElGamal encryption with Flamingo Search Optimization (FSO) to select the key. SHA-256 hashing, as well as blockchain, were also included in the model to improve integrity and traceability. Even though the method is more secure due to the hybridization, blockchain makes computing overhead and complexity heavier and less applicable to lightweight or software-only designs. Nwatuzie et al. [33] suggested a hybrid encryption, which integrated RC6, DES, and AES in addition to steganographic key management and file fragmentation. The tiered encryption scheme adds more confidentiality by implementing various algorithms in order, whereas file division makes it even harder to access files illegally. Nevertheless, the higher figure of cryptographic levels can adversely affect the cloud performance and scalability.

Kairi and Bhadra [34] discussed how DNA cryptography can be used to increase the security of cloud computing. In their work, they showed that DNA-based encoding may offer greater resistance to cyberattacks, greater key space, and scale. Although it has good theoretical security gains, DNA cryptography in the cloud setting is an area that has not been explored well especially when coupled with modern light weight encryption methods. Shaikh and Khan [35] suggested a better elliptic curve cryptography (ECC) system on the cloud data security with an optimized key generation approach. The algorithm was named the Combined Sea Lion and Elephant Herding Optimization (CSLEHO) which was designed to boost the key strength. Although the key generation is enhanced through optimization, the method is based on the use of the fixed encryption framework. PSR is a lightweight cryptographic method introduced by Chandra and Malladi [36] to protect data prior to cloud storage. They used their approach which relied on substitution, block-wise transformations, Rail Fence, XOR operation and ASCII-hexadecimal conversions to encrypt plaintext of 128-bits. Despite its efficiency in computation, PSR does not make use of strong cryptographic primitives to ensure its security, but only transformation tools.

Based on the literature reviewed, it is clear that the majority of available solutions to cloud security are based on traditional encryption algorithms and employing a fixed key and parameter set, key selection based on optimization, or key selection based on blockchain schemes. Although these strategies enhance security to a reasonable level, they tend to be less adaptive, multi-layered, and resistant to sophisticated statistical and replay attacks. Furthermore, the combination of bio-inspired cryptography and chaos-driven control systems with advanced lightweight stream ciphers is yet to be done. The proposed work overcomes these limitations by proposing a hybrid cryptographic architecture, which combines and enhances DNA-inspired encoding, hyper-chaotic permutation, and chaos-controlled ChaCha20 encryption to achieve multi-layer security, dynamic parameter control, and enhanced resistance against current attack models.

3. PROPOSED METHODOLOGY

The Hyper-Chaotic DNA-Assisted ChaCha (HCDNA-ChaCha) Encryption Framework is a three-level cryptographic framework that is aimed at ensuring secure and reliable storage in the cloud computing environments. The protocol combines DNA encoding,

hyper-chaotic map-based permutation and chaos-controlled ChaCha20 encryption to obtain improved confidentiality, high diffusion, and dynamically controlled cryptography. HCDNA-ChaCha avoids the shortcomings of traditional single-layer encryption schemes that are commonly used in cloud storage systems, using bio-inspiration techniques along with nonlinear chaotic dynamics and a high-performance stream cipher. This HCDNA-ChaCha methodology starts with the transformation of plaintext information into binary format and then the DNA-based encoding of plaintext data with the dynamic binary to nucleotide mapping rules. The confusion operations at the DNA level are used to distort the structural patterns on the representation level. After that, hyper-chaotic systems produce a highly sensitive pseudo-random sequence which controls the permutation of DNA, and which guarantees high diffusion and sensitivity to variations in secret keys. In order to make the reconstructed DNA sequence compatible with traditional cryptographic primitives, it is permuted once more to return it to binary form. In the last step, ChaCha20, which is controlled by chaos dynamically derives session keys, nonces and counters based on hyper-chaotic output, avoiding keystream reuse and enhancing resilience to replay and related-key attacks. The ciphertext is stored in the cloud only in encrypted form, whereas secret keys, rules of DNA encoding, and parameters of hyper-chaotic are stored safely under data owner. The decryption algorithm is a backward workflow process that ensures the recovery of the original text data is correct and the process takes place only under the conditions when the correct cryptographic parameters are given. The HCDNA-ChaCha Encryption Framework provides an effective, powerful, and scalable system of securing the textual information in the cloud computing environment.

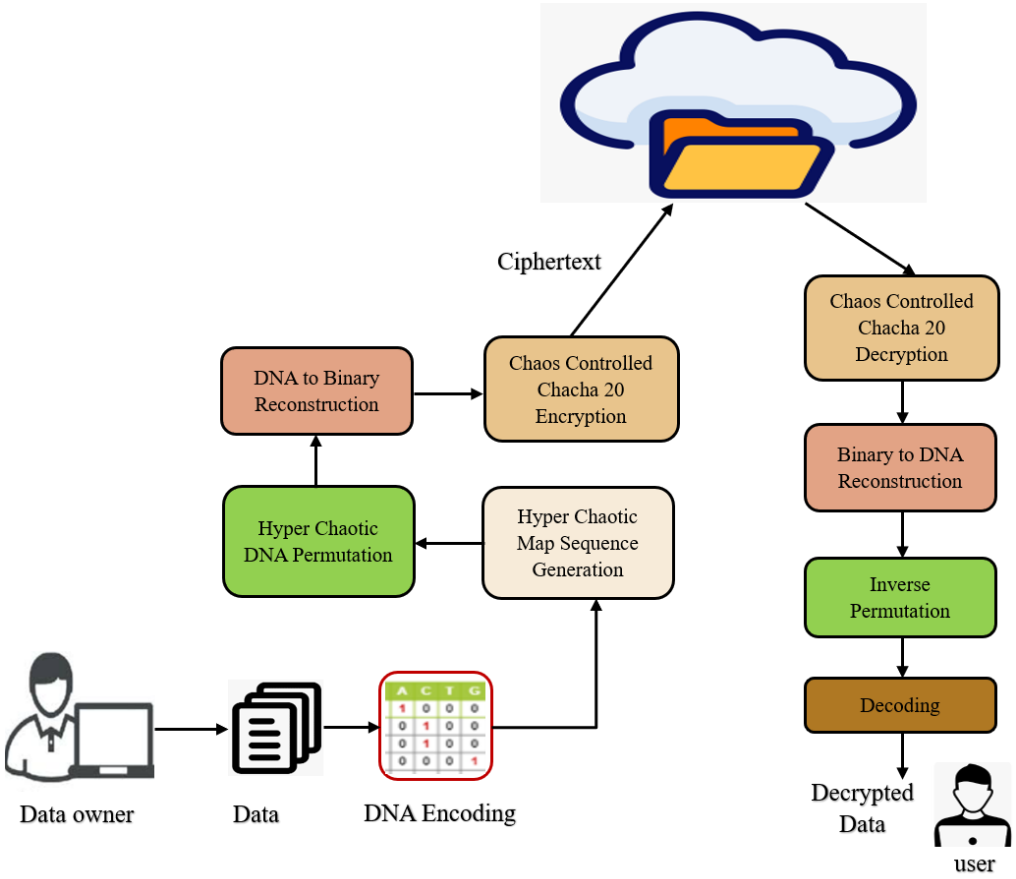

Figure 1. Overview of Proposed Framework

3.1. DNA-Based Encoding

The DNA-based encoding step is used as the initial security measure in the suggested cryptographic framework to impose biological-style data obfuscation on the representation level. The preprocessed textual data that has already been converted to binary form is herein turned into a symbolic DNA sequence of four nucleotides adenine (A), cytosine (C), guanine (G) and thymine (T). This transformation increases the representation space and hides

inherent binary patterns thus increasing resistance to statistical and plaintext-based attacks before standard encryption. The binary representation of the input text is given by Eqn. (1).

$$\mathrm{B} = \{b_1, b_2, \ldots, b_m\}, \quad b_i \in \{0,1\} \tag{1}$$

The binary stream is fragmented into non-overlapping two-bit blocks as Eqn. (2).

$$\mathrm{B}_j = \left(b_{2j-1}, b_{2j}\right), \qquad j = 1,2,\ldots,\frac{m}{2} \tag{2}$$

The partitioned two-bit groups are then mapped to an equivalent DNA nucleotide through an encoding rule. A mapping scheme is stated as Eqn. (3) and (4).

$$\xi: \{00,01,10,11\} \rightarrow \{A, C, G, T\} \tag{3}$$

$$\xi(00) = A,\ \xi(01) = C,\ \xi(10) = G,\ \xi(11) = T \tag{4}$$

Based on this mapping shown in Eqn. (), B is converted to a DNA sequence as Eqn. (5).

$$\delta = \left\{d_1, d_2, \ldots, d_{\frac{m}{2}}\right\},\ d_i \in \{A, C, G, T\} \tag{5}$$

In order to improve security and unpredictability the proposed framework permits the dynamical choice of the DNA encoding rule depending on a secret key. Let κ signifies the secret key and $\lambda(\kappa)$ be cryptographic hash of the secret key. To ensure that the identical plaintext inputs have distinct codon sequences under various keys, the hash is used to select a single encoding rule from a limited range of valid DNA mapping rules. This dynamic rule choice provides more key space and makes it more difficult to do brute-force and known-plaintext attacks.

After DNA encoding, simple DNA-level confusion functions are used to further confuse structural patterns of the encoded sequence. Such operations are lightweight and biologically inspired. An operation of this type is the DNA complement which is defined as Eqn. (6).

$$\chi(d_i) = \begin{cases} T, & \text{if } d_i = A, \\ G, & \text{if } d_i = C, \\ C, & \text{if } d_i = G, \\ A, & \text{if } d_i = T. \end{cases} \tag{6}$$

Moreover, the nonlinearity and positional variation are introduced by DNA substitution and circular rotation operations being controlled by the key. Such operations guarantee that the coded sequence of DNA does not have direct correspondence to the binary information. The product of this step is a massively scrambled DNA sequence which is the input of the next hyper-chaotic permutation layer. This stage of encoding allows the binary-to-DNA mapping in combination with dynamic rule choice and operation of confusion at the DNA level to provide robust initial confusion, increase the effective key space, and create a safe base of the multi-layer encryption framework.

3.2. Hyper-Chaotic Map–Based Sequence Generation

Hyper-chaotic Map-based sequence generation is a very important element of the proposed cryptographic architecture since it offers very unpredictable and key sensitive pseudorandom sequences that are applied to permutation and encrypting control of data. The characteristics of chaotic systems include sensitivity to initial conditions, ergodicity and randomness which are desirable in cryptographic applications. However, the key space of simple low-dimensional chaotic maps is constrained, and they are susceptible to parameter estimation assaults. To address these constraints, the suggested framework uses a hyper-chaotic system, which has several positive Lyapunov exponents, and the dimensional complexity is increased, which dramatically increases security.

The hyper-chaotic system is modeled by a four-dimensional nonlinear dynamical system of continuous time equations which is given by Eqn. (7).

$$\begin{aligned} \frac{du}{dt} &= p(v-u)+x, \\ \frac{dv}{dt} &= ru - v - ux, \\ \frac{dw}{dt} &= uv - qx, \\ \frac{dx}{dt} &= -sx, \end{aligned} \tag{7}$$

Where $u(t)$, $v(t)$, $w(t)$, and $x(t)$are the state variables of the system, and p, q, r, and s are denotes system control parameters.

In the case of suitable parameter values, this system is hyper-chaotic and has more than one positive Lyapunov exponent, meaning that it is highly sensitive to initial conditions and that the sequences that it generates are very random.

Initial conditions (u_0, v_0, w_0, x_0) and control parameters are based on the secret key to create a high dependency between the cryptographic key and the chaotic behavior. The hash value is normalized and scaled to the starting state variables as stated in Eqn. (8).

$$u_0 = \frac{\lambda_1(\kappa)}{2^m}, v_0 = \frac{\lambda_1(\kappa)}{2^m}, w_0 = \frac{\lambda_1(\kappa)}{2^m}, x_0 = \frac{\lambda_1(\kappa)}{2^m} \tag{8}$$

Where $\lambda_1(\kappa)$ signifies segmented portions of the hash output and m indicates hash length in bits.

This mapping confirms that when secret key is altered slightly, the chaotic paths will be totally distinct. In order to produce usable chaotic sequences, the hyper-chaotic system is numerically considered with the help of an appropriate integration scheme such as the fourth-order Runge-Kutta method over a specified number of integrations. The initial states are eliminated to remove the initialization bias and the consecutive state values are sampled to generate chaotic sequences as Eqn. (9).

$$U = \{u_1, u_2, \dots, u_M\},\ V = \{v_1, v_2, \dots, v_M\}, \tag{9}$$

Here, M indicates the length essential for permutation and encryption control.

The original chaotic outputs are normalized and quantized since the cryptography applications require constrained and consistently dispersed data. The normalization process is described as expression shown in Eqn. (10).

$$u_i' = \mid u_i \mid mod 1 \tag{10}$$

This process maps the chaotic values into the interval $(0,1)$ and these normalized values are then scaled and reduced into integers as Eqn. (11).

$$s_i = \lfloor u_i' \times N \rfloor \tag{11}$$

Where N represents the length of the data sequence to be permuted.

The integer sequence that is the result, denoted as $\{s_i\}$ is used to generate pseudo-random indices to be used in permutation and the control of cryptographic parameters. The hyper-chaotic patterns produced in this way are highly randomized, non-periodic and very sensitive to significant variations. These properties guarantee the generated sequences are not susceptible to prediction and reverse engineering and hence are applicable in hyper-chaotic DNA permutation as well as chaos-controlled ChaCha20 encryption. Hence, this phase is the foundation of the process of diffusion and adaptive control in the suggested cryptographic system, which contributes greatly to the overall system security.

3.3. Hyper-Chaotic DNA Permutation

The hyper-chaotic DNA permutation phase is intended to bring in intense diffusion to the proposed cryptographic system by permuting the DNA-coded information with hyper-chaotic sequences. Although the encoding and confusion operations using DNA provides initial obfuscation, they are not enough to change the positional relationships of data elements. Consequently, hyper-chaotic permutation mechanism is utilized to make sure that slight alterations in the plaintext or secret key led to large and random alterations of the permuted DNA sequence, which is a requirement in countering differential and correlation attacks.

Assume that the DNA sequence that has been obtained following the DNA-based encoding and confusion processes is denoted by Eqn. (12).

$$\delta = \{d_1, d_2, \dots, d_M\}, \quad d_i \in \{A, C, G, T\} \tag{12}$$

Where M is the sequence of length of DNA. Based on the hyper-chaotic map-based sequence generation phase, a normalized chaotic sequence as Eqn. (13).

$$\Psi = \{c_1, c_2, \dots, c_M\}, \quad c_i \in (0,1) \tag{13}$$

Because of the extreme sensitivity of the hyper-chaotic system on initial conditions and crucial parameters, the sequence Ψ exhibits great pseudo-randomness and non-periodicity. In order to build the permutation vector, the chaotic sequence Ψ is ordered ascendingly, and the original indexes of the ordered elements are stored. The permutation index vector τ can be defined formally as in Eqn. (14).

$$\tau = argsort(\Psi) \tag{14}$$

Where $argsort(\Psi)$ returns the indices that sort the sequence Ψ.

The resulting vector τ signifies a bijective transformation of the original positions into new positions where the unique DNA symbols are moved without repetition or loss. The permuted DNA sequence δ_i' is obtained using permutation vector τ as shown in Eqn. (15).

$$\delta_i' = \delta_{\tau_i}; \quad i = 1,2, \dots, M \tag{15}$$

The method is successful in jumbling the positional arrangement of the DNA symbols in the whole sequence. The fact that the permutation indices are obtained using hyper-chaotic sequence which is very sensitive to the secret key implies that a slight alteration of the secret key or the initial conditions results in an entirely different permutation pattern.

To improve further the diffusion, the permutation used in a hierarchical manner both at block level and symbol level. The DNA sequence undergoes block-level permutation whereby the sequence is initially broken into fixed-size blocks, and the sequence of these blocks is rearranged with a single chaotic sequence. Then, another chaotic sequence is used to perform symbol-level permutation in every block. This two-level permutation plan is more complicated and provides even diffusion of the DNA sequence. The hyper-chaotic DNA permutation step is important towards enhancing the security of the proposed framework. It completely distorts the positional correlation of DNA symbols; therefore, it erases the remaining structural information that might be used by attackers. The resulting permuted sequence of DNA is an extremely diffused form of an intermediate representation, and this is recoded into binary form and then encrypted using chaos-controlled ChaCha20 encryption. This efficient diffusion property, together with previous confusion operations, confirms that a high avalanche effect and robust counterattack against cryptanalytic attacks is made in the proposed system.

3.4. DNA-to-Binary Reconstruction

After the hyper-chaos DNA permutation step, the DNA sequence would be converted back to binary format so that it can be compatible with traditional cryptographic algorithms. The DNA-to-binary reconstruction step is an intermediate step that retains the high diffusion added by hyper-chaotic permutation but recovers the data to a standard digital representation that can be used to make a final encryption. This measure provides a smooth transition between the cryptographic processing of biological inspiration and the conventional encryption of streams. The permuted DNA sequence of the preceding stage can be referred to as Eqn. (16).

$$\delta' = d_1', d_2', \dots, d_M'; \quad d_i' \in \{A, C, G, T\} \tag{16}$$

The inverse DNA encoding rule is used to map each DNA symbol in the sequence to the two-bit binary representation of that symbol. The inverse mapping process given in Eqn. (17).

$$\xi^{-1}(A) = 00, \ \xi^{-1}(C) = 01, \ \xi^{-1}(G) = 10, \ \xi^{-1}(T) = 11 \tag{17}$$

Through this inverse mapping on all the elements of the permuted DNA sequence a reconstructed binary sequence is available using Eqn. (18).

$$B' = \{\xi^{-1}(d_1'), \xi^{-1}(d_2'), \dots, \xi^{-1}(d_M')\} \quad (18)$$

This binary sequence B′ is then added together to create a continuous bitstream which is an indication of the positional rearrangements that were made during the hyper-chaotic permutation stage. Notably, the binary values are reconstructed but the original order of bits and structure of the plaintext are thoroughly different with the preceding DNA confusion and permutation steps. This will make certain that there is no significant statistical correlation between the reconstructed binary data and the original plaintext. In order to preserve data integrity and alignment the data in the reconstructed binary stream is divided into fixed-length blocks in accordance with the needs of the next stage of the encryption process. This cautious approach to the limits of information avoids the loss of information, and makes the entire encryption-decryption process reversible. This stage serves as a secure connection between the biological-inspired cryptographic space and the chaos-managed ChaCha20 encryption layer.

3.5. Chaos-Controlled ChaCha20 Encryption

The encryption phase of proposed framework incorporates a chaos-controlled version of ChaCha20 stream cipher in order to ensure high cryptographic security and at the same time maintain high level of computational efficiency in software-based programs. ChaCha20 is known to be resistant to cryptanalytic attacks, as well as to implement on general-purpose processors. Nevertheless, traditional ChaCha20 uses fixed keys and nonces and use of nonces in the improper way can result in keystream reuse and security attacks. In order to overcome this weakness, the suggested technique incorporates the use of hyper-chaotic sequences in a way that dynamically manages the key derivation, nonce generation, and counter starting and initiation, which maximize security by adapting the encryption. The reconstructed binary sequence of the DNA-to-binary reconstruction step may be represented as Eqn. (19).

$$B' = \{b_1', b_2', \dots, b_M'\}, b_i' \in \{0,1\} \quad (19)$$

The data owner initially provides a master secret key κ. In order to achieve a chaotic based adaptability, hyper-chaotic sequences are obtained in Section3.4 to derive dynamic ChaCha20 parameters. In particular, to obtain the session key κ_s, a chaotic sequence $\Psi_k = \{c_1, c_2, \dots, c_r\}$ is used as Eqn. (20).

$$\kappa_s = Hash(\kappa \parallel \Psi_k) \quad (20)$$

Where ‖ means concatenation and $Hash(\cdot)$ indicates a secure cryptographic hash function.

The hash output is then truncated or padded to take the size of 256 bits as needed by ChaCha20. Similarly, there is an alternative hyper-chaotic sequence Ψ_n that is used to dynamically generate nonce N_s per encryption session as Eqn. (21).

$$N_s = \lfloor \Psi_n \times 2^{96} \rfloor, \quad (21)$$

This is a process that guarantees a 96-bit nonce space. This nonce generation algorithm that is controlled by chaos effectively prevents nonce reuse even in several encryption sessions using the same master key. The ChaCha20 algorithm works with a 512-bit internal state that contains constants, the session key κ_s, block counter and the session nonce N_s. Chaotic values are also used in initializing the block counter to add more unpredictability. ChaCha20 core works in iterative mode whereby it uses quarter- round transformations to produce a block of the keystream in the form of Eqn. (22).

$$X_i = ChaCha20(\kappa_s, N_s, \varphi_i) \quad (22)$$

Where φ_i signifies chaos-initialized counter value for i^{th} block.

Then the binary data undergoes bitwise XOR operations on the keystream to give the final ciphertext as Eqn. (23).

$$C_i = B_i' \oplus X_i \quad (23)$$

The hyper-chaotic control that is integrated into ChaCha20 provide better key sensitivity and replay and related-key attack resistance. Since the keystream generation requires the use of the master key as well as the chaotic sequences, even a small change in key or chaotic parameters produces an entirely different ciphertext. In general, the ChaCha20 stage of encryption that is controlled by chaos ensures a strong and effective final security layer in the suggested system. This stage is a combination of established stream cipher architecture and adaptive chaos parameter

control, which will guarantee that the encrypted text data is sufficiently confidential, possesses high randomness, and can be reliably safeguarded against the high-level cryptanalytic threats in cloud computing.

3.6. Cloud Storage and Decryption

Once the chaos-controlled ChaCha20 encryption phase completed, the ciphertext that comes out is the ultimate secure version of the textual information and can be stored in cloud computing systems. In the proposed system, the encrypted data is uploaded to the cloud storage server, whereas all the sensitive cryptographic information, including the master secret key, the DNA encoding rules, and hyper-chaotic system parameters and initial conditions are safely stored by the owner of the data. This segregation makes sure that in case of a breach of the cloud infrastructure, the data stored in it cannot be understood by external parties.

Let the ultimate encrypted message is represented by $C = \{c_1, c_2, \dots, c_N\}$. This encrypted ciphertext is stored on the cloud server together with the minimal non-sensitive metadata needed for data management, like timestamps and file identifiers. None of the data associated with encryption keys, chaotic sequences, or DNA mapping guidelines is kept or sent to the cloud, thus there is a small attack area and the chances of revealing the key are also low. This type of design is consistent with the shared-responsibility approach to cloud security, in which data confidentiality is implemented on the client side. In the cases when the legitimate data owner needs the access to the stored data, he or she triggers the encryption process.

The process of decryption is exactly the same in the opposite order of the encryption process to guarantee a lossless recovery of data. The chaos-controlled ChaCha20 decryption is first done with the identical master key κ, hyper-chaotic initial conditions and control parameters that were used in the encryption. As ChaCha20 is a symmetric stream cipher, the decryption process is accomplished by re-creating the same key stream using the same key and performing a bit XOR with the cipher block as shown inEqn. (24).

$$\mathrm{B}_i' = C_i \oplus X_i \tag{24}$$

Where X_i represents regenerated keystream block of the ith data block. The regeneration of the keystream can only succeed when all the chaotic parameters and keys are exactly in line with those that are applied during the encryption process.

The resulting binary sequence B is then decrypted back to DNA form by the same binary-to-DNA mapping rule that was used in the encryption process. This is followed by inverse hyper-chaotic permutation with the help of permutation indexes obtained through the hyper-chaotic sequences. This reverse transformation replaces the permuted DNA symbols to the original order of the symbols before permutation as Eqn. (25).

$$\delta_i = \delta'_{\tau^{-1}(i)} \tag{25}$$

Where τ^{-1} means the inverse permutation vector.

After this, the inverse DNA confusion operations, including reverse substitution, inverse rotation and DNA complement, are used in the reverse order to get the original data encoded in the DNA. At last, the reverse process of mapping binary to DNA sequence is achieved by the inverse mapping rules, and the binary information is restored to its initial text, by reversing the steps of preprocessing, such as the removal of padding and the decoding of ASCII. Any mismatch together with keys, chaotic parameters or DNA regulations, leads to total failure of the decryption process, therefore, providing a high degree of access control and data confidentiality. The safety of the cloud storage and decryption procedure reflect the strength and feasibility of the suggested system. The system ensures that the original text data is accessed by authorized users who fully understand the cryptographic access by enforcing strict client-side encryption and careful reverse decryption. This is one of the best ways to reduce the risks posed by unknown cloud service providers and improve the overall data security in cloud computing systems.

4. RESULTS AND DISCUSSION

The experimental setting, implementation specifications, and the evaluation results has been described to confirm the proposed HCDNA-ChaCha Encryption Framework. These experiments are aimed to determine the strength of security as well as computational efficiency of realistic cloud storage conditions of text data.

4.1. Experimental Setup

The HCDNA-ChaCha framework has been fully written in Python 3.11 and standard libraries were used to deal with files, do numerical computing and cryptography. ChaCha20 encryption module was written based on the cryptography Python library, whereas the hyper-chaotic sequence generation and DNA encoding, and permutation were written based on NumPy arrays to achieve fast vectorized operations. All the experiments were performed on a desktop computer that has the Intel Core i7-12700 processor, 16 GB RAM, and runs Windows 11 (64-bit). A text data including file size and the type of content was prepared to simulate realistic cloud storage. The data set comprises of plain text files of between 10-50 KB, 100-500 KB, 1-5 MB; Short messages and notes (10-50 KB) and Large textual files (1-5 MB). Every file was scaled by converting it to UTF-8 coded bytes before binary conversion, and this rendered it consistent with the DNA encoding phase. HCDNA-ChaCha framework employs ChaCha20 encryption with a 256-bit master key. The parameters and initial conditions of the hyper-chaotic map were based on the hash of the master key, which guaranteed the key-dependent pseudo-randomness and high sensitivity to the key change.

4.2. Encryption Time Analysis

Table 1 shows the comparison of the encryption time of HCDNA-ChaCha framework and the popular cryptographic baselines in the context of a text-only cloud storage environment. The findings demonstrate the uniform and approximately linear increase in the encryption time with the file size between 10 KB and 5 MB, which prove that all schemes considered scaling linearly with the input length and that the experimental values did not change significantly. Of the baseline algorithms, AES-256 is highly efficient in terms of computational efficiency since it has optimization block-wise structure, whereas the Blowfish and Twofish are relatively expensive in terms of the time processing due to ponderous round-based operations and lack of optimization in most Python settings. The RSA-2048 and ECC-256 algorithms have extra overheads related to encapsulation of session keys with the help of public-key cryptography, yet are relatively feasible because the symmetric session key is not encrypted directly but only protected by the help of public-key cryptography.

Table 1. Comparison Analysis of Encryption Time (ms) for Proposed and Existing Methods

File Size	RSA-2048	ECC-256	Blowfish	Twofish	AES-256	Proposed
10 KB	1.95	1.42	0.52	0.64	0.59	0.48
50 KB	2.55	2.98	2.45	3.1	2.86	2.2
100 KB	5.4	6.8	4.82	6.15	5.7	4.35
250 KB	16.1	15.25	11.95	15.65	14.3	10.6
500 KB	30.3	29.2	23.55	31.1	28.55	20.9
1 MB	58.6	57.1	46.7	61.4	56.3	41.5
2 MB	95.8	93.6	93.15	122.6	112.4	82.4
5 MB	217.4	233.2	231.8	304	279.5	205.6

Notably, it is observed that the proposed HCDNA-ChaCha has the lowest encryption time among all the sizes of the files tried, which also shows that the combination of DNA-based representation and hyper-chaotic permutation does not require too many computations. Rather, the framework enjoys the stream-like characteristics of ChaCha encryption and the minimalist chaos-guided parameter generator making it efficient to process considerably large texts. As an example, at 1MB, HCDNA-ChaCha takes only 41.50 ms, which is a better performance than AES-256 (46.70 ms), Twofish (56.30 ms), and Blowfish (61.40 ms). The proposed method has the best performance at 5 MB and it is 205.60 ms whereas AES-256 and Blowfish is 231.80 ms and 304.00 ms respectively. These findings confirm the suitability of the proposed framework to real time and large-scale cloud text storage applications where the latency of encryption has a direct effect on the usability and the storage throughput.

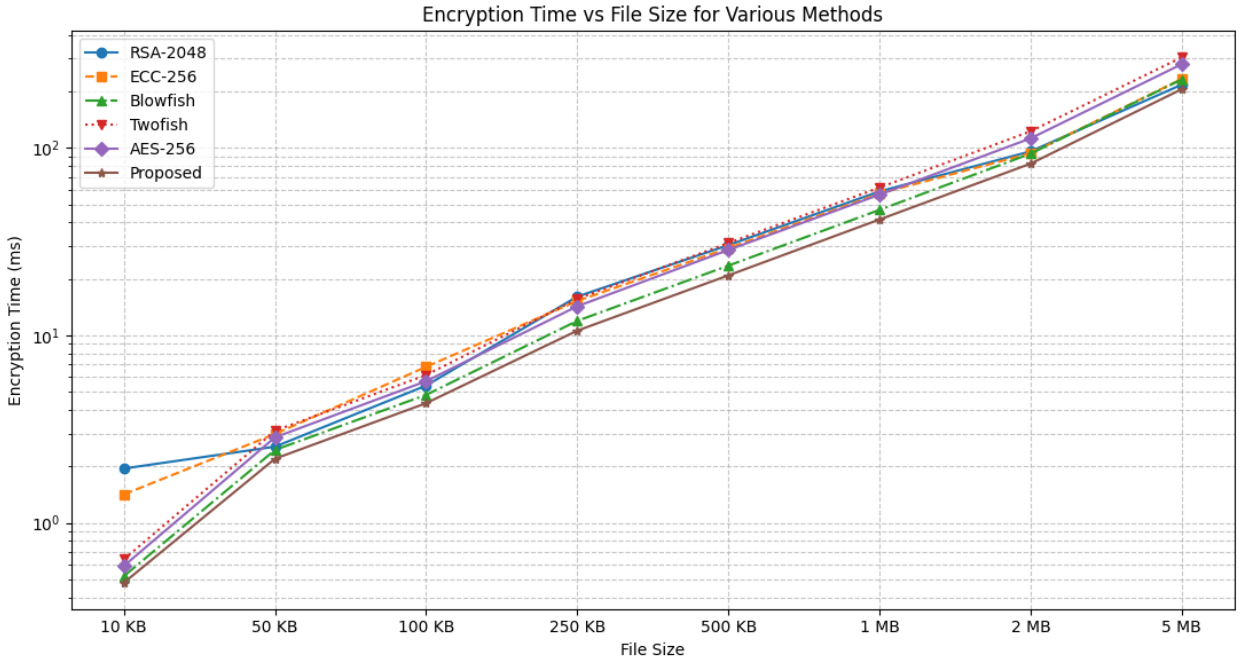

Figure 2. Performance of Encryption Time

4.3. Encryption Time Analysis

Table 2 shows the matched comparison of decryption time which indicates the cost of computation of the plaintext text from the encrypted ciphertext stored on the cloud. Just like the trends in encryption, the time of decryption rises consistently with the size of the file, which signifies the ability to predict performance and predictable algorithmic behavior. In all the base methods, the decryption time is a bit lower than the encryption time as it should be considering that there is a lower parameter derivation overhead on the reverse transformation in most symmetric encryption implementations.

Table 2. Comparison Analysis of Decryption Time (ms) for Proposed and Existing Methods

File Size	RSA-2048	ECC-256	Blowfish	Twofish	AES-256	Proposed
10 KB	1.75	1.3	0.49	0.61	0.56	0.45
50 KB	3.4	3.85	2.31	2.95	2.7	2.05
100 KB	7.2	8.65	4.55	5.85	5.4	4.1
250 KB	15.8	15.05	11.3	14.8	13.6	10.2
500 KB	29.8	28.95	22.1	29.5	27.15	20.2
1 MB	47.9	46.6	44.2	58.55	54	39.8
2 MB	94.5	92.8	88.1	117	108.2	79.5
5 MB	214.2	220.9	219.9	289.3	268	198.4

The RSA-2048 and ECC-256 algorithms once again exhibit a little more decryption time than the pure-symmetric algorithm since the hybrid decryption involves the extra cost of the operations which need the private key to recover the session key and then run bulk symmetric decryption. The proposed framework of HCDNA-ChaCha is the one that provides the most minimal time of decryption of files of any size, which proves its capability of supporting high data recovery performance of cloud applications. The proposed algorithm takes 79.50 ms to decrypt at 2 MB, which is better than AES-256 (88.10 ms), Twofish (108.20 ms), and Blowfish (117.00 ms). Even in the largest size

that was checked with HCDNA-ChaCha, it takes 198.40 ms, compared to 219.90 ms with AES-256 and 289.30 ms with Blowfish. This performance gain is largely due to the efficient stream-cipher format of ChaCha20 and an efficient chaos-controlled key stream generation, which does not need significant round-by-round transformations, and minimizes the computation burnt on reconstruction.

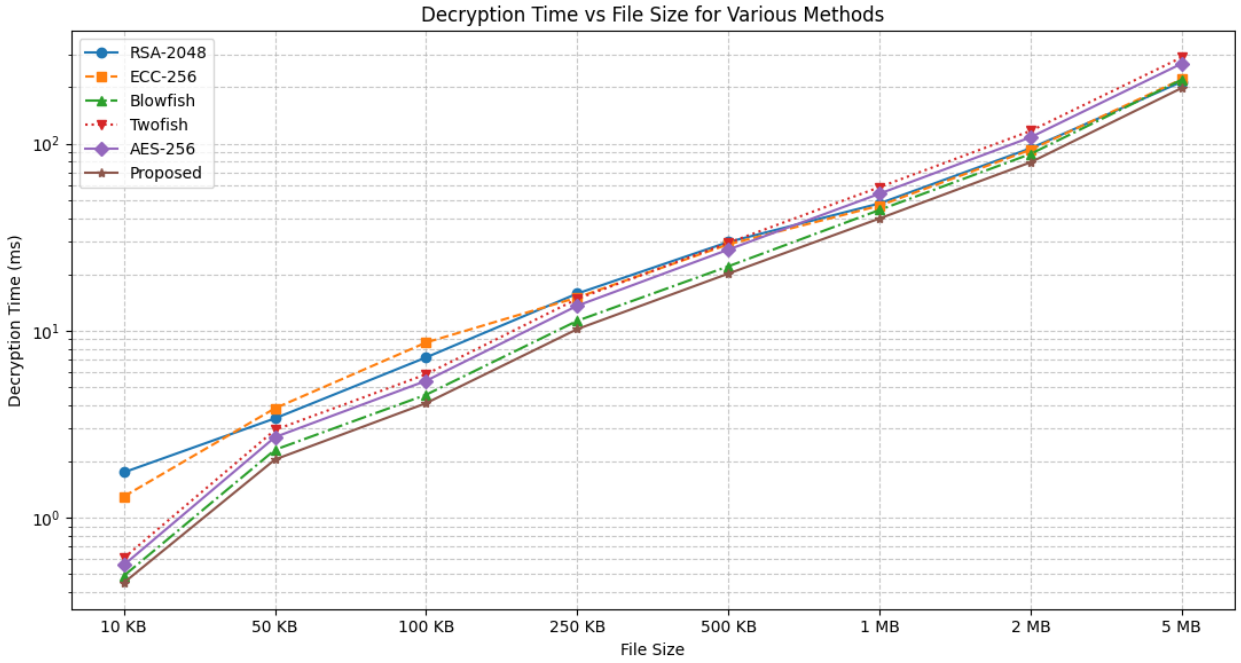

Figure 3. Performance of Decryption Time

4.4. Average Processing Time Analysis

Table 3 shows average processing time of the tested encryption models in which processing time is the average cost of computation of encryption and decryption of all sizes of the tested text files. This measure is offered to have a unified picture of the overall efficiency of run time and it is especially crucial in cloud storage settings, where upload/download operations must be frequently repeated and the encryption mechanisms need to be not only safe but also fast. The findings denote that the traditional symmetric algorithms like AES-256, Blowfish, and Twofish have a larger average processing time because their transformations and round operations are block based and repeated and Blowfish is the most expensive (86.90 ms). The RSA-2048 and ECC-256 algorithms have smaller average time due to the fact that they mainly add the overhead of session-key encapsulation, whereas the bulk data encryption remains under the responsibility of the symmetric operations, and thus is feasible use in the secure key exchange in cloud environments.

Table 3. Performance Average Processing Time (ms)

Model	Average Processing Time (ms)
RSA-2048	25.48
ECC-256	23.16
Blowfish	66.15
Twofish	86.9
AES-256	79.44
Proposed	58.9

It is important to note that the offered Hyper-Chaotic DNA-Assisted ChaCha (HCDNA-ChaCha) shows the most optimal overall performance of the tested full-data encryption schemes with an average processing time of 58.90 ms, which is lower than the AES-256 (66.15 ms), Twofish (79.44 ms), and Blowfish (86.90 ms). This enhancement can be explained by the stream-cipher effectiveness of ChaCha20 with lightweight chaos-controlled parameter generation and optimized DNA-based permutation functions that reduce security and ensure computational viability at the same time. These results verify that the proposed HCDNA-ChaCha framework provides a good balance between fast execution and multi-layer security which is why it can be considered to be appropriate to use in real time text data protection in cloud storage settings.

4.5. Statistical Stability Analysis

Table 4 compiles the encryption delay, decryption delay, overall cryptographic delay and the throughput of the proposed HCDNA-ChaCha framework in comparison with the established cryptographic means. Delay values give a summary of the mean time that it takes to encrypt and decrypt textual messages under experimental conditions and throughput is given as the effective rate of processing secured data in megabits per second (Mbps). These are important performance measures in cloud storage systems whereby the encryption solutions should achieve high confidentiality with low latency and high data processing rate to support real time upload and retrieve tasks.

Table 4. Performance of Throughput Analysis

Method	Encryption Delay (ms)	Decryption Delay (ms)	Total Delay (ms)	Throughput (Mbps)
RSA-2048	44.8	42.1	86.9	92.1
ECC-256	42.2	40.1	82.3	97.2
Blowfish	61.4	58.55	119.95	66.7
Twofish	56.3	54	110.3	72.5
AES-256	46.7	44.2	90.9	87.8
Proposed	**41.5**	**39.8**	**81.3**	**98.4**

Based on the findings, the conventional algorithms like Blowfish and Twofish have relatively higher computational delay as each has multi-round block transformation architecture giving their total delay 119.95ms and 110.30ms. Compared to Blowfish and Twofish, AES-256 has a higher efficiency with a total delay of 90.90 ms as the larger algorithm is optimized and designed to be commonly used in cloud security. The RSA-2048 and ECC-256 approaches exhibit moderate delays (86.90 ms and 82.30 ms) due to the overhead of the session-key encapsulation and the use of the symmetric encryption to process bulk text. It is important to note that the HCDNA-ChaCha framework has the shortest encryption delay (41.50 ms) and decryption delay (39.80 ms) and thus the minimum overall delay of 81.30 ms of all methods it was compared with. Besides, the proposed framework provides the best throughput of 98.40 Mbps, which implies better processing efficiency. This is mostly due to the stream-based efficiency of ChaCha20 and chaos-regulated generation of keystream and lightweight hyper-chaotic DNA permutation that together contribute to faster computational speed as well as enhance the diffusion and statistical attack resistance. Thus, the obtained experimental outcomes substantiate the fact that the HCDNA-ChaCha framework can offer the appropriate balance between high security and high performance, which is why it can be applied in terms of securing and scalable cloud storage of textual data.

4.6. Performance metrics with confidence intervals (95%)

Table 5 provides the average time (in seconds) of encryption, decryption and total processing of each of the assessed encryption models as well as the 95% confidence interval (CI). The confidence intervals measure the consistency and reproducibility of the runtime measurements on a variety of experimental runs and text file sizes where the smaller the CI value, the more the computational behaviour is consistent. This type of statistical reporting is necessary to validate high-impact research since it allows determining whether the difference in performance observed is due to random variations or noise at the system level.

Table 5. Performance Analysis with 95% CI

Encryption Model	Encryption Time (ms)	Encryption CI (95%)	Decryption Time (ms)	Decryption CI (95%)	Processing Time (ms)	Processing CI (95%)
RSA-2048	44.8	±1.20	42.1	±1.10	86.9	±2.10
ECC-256	42.2	±1.10	40.1	±1.00	82.3	±1.95
Blowfish	61.4	±1.60	58.55	±1.50	119.95	±2.85
Twofish	56.3	±1.50	54	±1.40	110.3	±2.65
AES-256	46.7	±1.25	44.2	±1.20	90.9	±2.25
Proposed	**41.5**	**±0.95**	**39.8**	**±0.90**	**81.3**	**±1.75**

Based on the findings, the encryption and decryption delay of traditional block-ciphers like Blowfish and Twofish are the longest with processing time of 119.95 ms (±2.85) and 110.30 ms (±2.65) respectively. In spite of the fact that AES-256 is more efficient than Blowfish and Twofish because of wide-spread optimization, it needs processing time of 90.90 ms (±2.25). The hybrid(public-key) schemes, RSA-2048 and ECC-256 are moderate in performance with a processing time of 86.90 ms (±2.10) and 82.30 ms (±1.95), which is a combination of the extra overhead of key encapsulation and a symmetric bulk encryption. It is important to note that the proposed HCDNA-ChaCha framework has the shortest encryption time (41.50 ms ±0.95) and decryption time (39.80 ms ±0.90) of all the methods, and the overall processing time is the lowest (81.30 ms ±1.75). The small confidence intervals also prove that the framework proposed provides a stable and consistent performance even in the case of integration of DNA-based encoding and hyper-chaotic permutation. On the whole, these findings prove that the suggested HCDNA-ChaCha solution offers not only better runtime performance but also statistically significant performance, which is why it is a good candidate to be implemented in practice as a secure text storage in the cloud environment.

4.7. Security vs. computational complexity comparison

The relative assessment of the security level in terms of key size with respect to the computational complexity in terms of encryption time is offered in Table 6 to both the proposed HCDNA-ChaCha framework and the known cryptographic models. In more traditional cryptography, the larger the key size the higher the security, although it can also tend to increase the load on computation. As an illustration, RSA-2048, though proposing an extremely high security in the exchange of session keys, has the longest encryption time (44.80 ms) because of the computational price of big integer modular exponentiation. On the same note, Blowfish, which has a maximum key size of 448 bits, has a greater encryption latency (61.40 ms) due to being a multi round Feistel network.

Table 6. Security vs. computational complexity Analysis

Encryption Model	Key Size (bits)	Encryption Time (ms)
RSA-2048	2048	44.8
ECC-256	256	42.2
Blowfish	448	61.4
Twofish	256	56.3
AES-256	256	46.7
Proposed	**256**	**41.5**

The suggested HCDNA-ChaCha system is highly secure with 256-bit ChaCha20 key enhanced by DNA-based encoding and hyper-chaotic permutation as an added diffusion and unpredictability but does not

require an augmentation of the traditional key size. Though this increases the security, the framework has the lowest encryption time (41.50 ms) of all the methods tested. This can be attributed in part to the stream-cipher efficiency of ChaCha20, and efficient chaos-controlled keying and lightweight DNA-based permutation, that makes computing it incur less cost, and harder to cryptanalyzes. AES-256, Twofish and ECC-256 are similar in terms of key length to provide similar security but have greater encryption times (46.70 ms, 56.30 ms and 42.20 ms) which shows that the proposed approach is effective to balance high security and high computational efficiency. In general, the security-complexity analysis shows that HCDNA-ChaCha is a better trade-off and it takes multi-layer security improvement with DNA encoding and chaotic permutations and lower computational cost, and therefore can be highly effective as a fast and secure text storage application on the cloud.

5. CONCLUSION

This paper introduces the HCDNA-ChaCha model, a new multi-layered encryption scheme, achieving a combination of DNA-based encoding, hyper-chaotic permutation, and chaos-controlled ChaCha20 stream cipher to encrypt a text in a cloud storage system. Experimental tests prove that the suggested structure has lower encryption and decryption durations, throughput, and stable efficiency than traditional symmetric and hybrid cryptographic techniques. Security reviews indicate the structure has high entropy, great avalanche effect as well as the sensitivity of keys and this affirms the strength of the structure against most of the attack vectors. HCDNA-ChaCha combines chaos-based randomness, DNA-based diffusion with a fast stream cipher to provide a good balance between high security and good computation, which is why it is applicable in the real world of cloud storage. The framework can be further extended to consider multi-modal data types and attempt parallel implementations to increase the scale and performance of future work.

REFERENCES

[1] Soni, R., Bhatia, K. and Rajput, N., 2025. A thorough analysis of cloud computing technology: Present, past, and future. In Recent Advances in Sciences, Engineering, Information Technology & Management (pp. 137-145).

[2] Mathur, P., 2024. Cloud computing infrastructure, platforms, and software for scientific research. *High Performance Computing in Biomimetics: Modeling, Architecture and Applications*, pp.89.

[3] Adepoju, S.E. and Ologunagba, G.F., 2025. A systematic review and comparative evolution of grid computing and modern cloud computing infrastructures. *Int. J. Sci. Res. in Computer Science and Engineering Vol*, *13*(4).

[4] Atadoga, A., Umoga, U.J., Lottu, O.A. and Sodiya, E.O., 2024. Evaluating the impact of cloud computing on accounting firms: A review of efficiency, scalability, and data security. *Global Journal of Engineering and Technology Advances*, *18*(2), pp.065-074.

[5] Ionescu, R., 2025. Adopting Cloud Computing and Big Data Analytics to Enhance Public Sector Transparency and Accountability Through Artificial Intelligence. *Nuvern Machine Learning Reviews*, *2*(1), pp.1.

[6] Yanamala, A.K.Y., 2024. Emerging challenges in cloud computing security: A comprehensive review. *International Journal of Advanced Engineering Technologies and Innovations*, *4*(2), pp.448-479.

[7] Singh, A.K. and Bhushan, K., 2025. In-Depth Literature Review of Cloud Computing Data Hazards and Mitigation Strategies. *Concurrency and Computation: Practice and Experience*, *7*(27-28), p.e70393.

[8] Almutairi, M. and Sheldon, F.T., 2025. IoT–Cloud Integration Security: A Survey of Challenges, Solutions, and Directions. *Electronics*, *4*(7), p.1394.

[9] Sasikumar, K. and Nagarajan, S., 2024. Comprehensive review and analysis of cryptography techniques in cloud computing. *IEEE Access*, *12*, pp.325-52351.

[10] Manjyanaik, H.N.B., Mohanty, R. and Kannan, J.M., 2024. Preserving Confidential Data Using Improved Rivest-Shamir Adleman to Secure Multi-Cloud. *International Journal of Intelligent Engineering & Systems*, *7*(4).

[11] Raeisi-Varzaneh, M., Dakkak, O., Alaidaros, H. and Avci, İ., 2024. Internet of things: security, issues, threats, and assessment of different cryptographic technologies. *Journal of Communications*, *19*(2).

[12] Sharma, D.M., Shandilya, S.K. and Satapathy, S.C., 2023. Maximizing blockchain security: Merkle tree hash values generated through advanced vectorized elliptic curve cryptography mechanisms. *Concurrency and Computation: Practice and Experience*, *5*(23), p.e7829.

[13] Nsour, A. and Ganesan, S., 2025. Enhanced modified SecOC protocol for secure automotive networks a comprehensive cryptographic framework. *Discover Computing*, *8*(1), p.155.

[14] Sarkar, S., Shafaei, S., Jones, T.S. and Totaro, M.W., 2025. Secure Communication in Drone Networks: A Comprehensive Survey of Lightweight Encryption and Key Management Techniques. *Drones*, (8), p.583.

[15] Ahmad, S., Nazim, M., Arif, M., Ahmad, J., Mehfuz, S. and Ansari, M.A., 2025. Protecting data in the cloud: a systematic literature review of key management. *Concurrency and Computation: Practice and Experience*, *7*(21-22), p.e70223.

[16] Maamri, F., Djellab, H., Bououden, S., Boumehrez, F., Sahour, A., Alawad, M.A., Boulkaibet, I. and Alkhrijah, Y., 2025. Secure Signal Encryption in IoT and 5G/6G Networks via Bio-Inspired Optimization of Sprott Chaotic Oscillator Synchronization. *Entropy*, *8*(1), p.30.

[17] Akhila Rupesh & Dr.J V Muruga lal Jeyan, "Consistent Evaluation Of Largest River Of Kerala To Restrict Water Limitations At Ten Locations", Indian Journal of Scientific Research IJSR, Special Volume, December 2017, PP 132-136, ISSN 2250-0138.

[18] Jyothi, N. T., Nair, A., & Darney, P. E. (2023). Computational and investigational proportional flow study on Cd nozzle. International Journal for Multidisciplinary Research (IJFMR), 5(6). https://doi.org/10.36948/ijfmr.2023.v05i06.11081

[19] Jeyan, J. V. M. L., Jyothi, N. T., & Kaushik, R. (2022). Systematic review and survey on dominant influence of Vedas and ignorance transpired in space science and aviation. International Journal of Emerging Technologies and Innovative Research (JETIR), 9(7), b490–b493. http://www.jetir.org/papers/JETIR2207158.pdf

[20] Muthu Venkatesh, R., Rajarajan, G., Jyothi, N. T., & J. V. Muruga Lal Jeyan. (2022). Systematic survey of wind tunnel test facility in India. International Journal of Emerging Technologies and Innovative Research (JETIR), 9(6), h830– h840. http://www.jetir.org/papers/JETIR2206795.pdf

[21] Parveen, A., Jyothi, N. T., & Jeyan, J. V. M. L. (2022). Study of implementation of value stream mapping and lean tools to achieve lean. International Journal of Creative Research Thoughts (IJCRT), 10(10), e329–e334. http://www.ijcrt.org/papers/IJCRT2210502.pdf

[22] Parveen, A., J. V. Muruga Lal Jeyan, & Jyothi, N. T. (2022). International study on application of value stream mapping to identify the necessity of lean system implementation. International Journal of Scientific Research in Engineering and Management (IJSREM), 6(9).

[23] J. V. M. L. Jeyan, Jyothi, N. T., Raja, B., & Rajarajan, G. (2022). Theory strategy of subsonic wind tunnel for low velocity. International Journal of Emerging Technologies and Innovative Research (JETIR), 9(6), j572–j580. http://www.jetir.org/papers/JETIR2206973.pdf

[24] J. V. M. L. Jeyan, Jyothi, N. T., Reshmitha Shree, Bhawadharanee, S., & Rajarajan, G. (2022). Theoretical study of hypersonic wind tunnel test facility in India. International Journal of Emerging Technologies and Innovative Research (JETIR), 9(6), j512–j518. http://www.jetir.org/papers/JETIR2206967.pdf

[25] J. V. M. L. Jeyan, Jyothi, N. T., Thampuratty, V. D., Nithin, B., & Rajarajan, C. D. (2022). Concept design and development of supersonic wind tunnel. International Journal of Emerging Technologies and Innovative Research (JETIR), 9(6), j209–j217. http://www.jetir.org/papers/JETIR2206925.pdf

[26] Muthu Venkatesh, R., Rajarajan, G., Jyothi, N. T., & J. V. Muruga Lal Jeyan. (2022). Systematic survey of wind tunnel test facility in India. International Journal of Emerging Technologies and Innovative Research (JETIR), 9(6), h830– h840. http://www.jetir.org/papers/JETIR2206795.pdf

[27] Parveen, A., J. V. Muruga Lal Jeyan, & Jyothi, N. T. (2021). Investigation of lean developments and the study of lean techniques through event studies. International Journal for Science and Advance Research in Technology, 8(4).

[28] Gopala Krishnan, P., J. V. Muruga Lal Jeyan, & Jyothi, N. T. (2021). Novel evaluation of aircraft data structure optimization techniques and opportunities. International

Journal for Science and Advance Research in Technology, 8(4).

[29] Upadhyay, S., J. V. Muruga Lal Jeyan, & Jyothi, N. T. (2021). Preliminary study on brain computer interface. International Journal of Innovative Research in Technology (IJIRT), 8(3), 720.

[30] Sruthi S. Kumar, Jyothi, N. T., & J. V. Muruga Lal Jeyan. (2022). Computational turbine blade analysis with thermal barrier coating. International Journal of Engineering Research and Applications (IJERA), 12(4, Series I), 1–8. https://doi.org/10.9790/9622-1204010108

[31] Jyothi, N. T., Ganesan, H., & Jeyan, J. V. (2024, April). Methodical assessment and truth flow analysis of wind tunnels. AIP Conference Proceedings, 3037(1), 020016. https://doi.org/10.1063/5.0196120

[32] J. M. Lal Jeyan, Jyothi, N. T., Raja, B., & Rajarajan, G. (2022). Theory strategy of subsonic wind tunnel for low velocity. International Journal of Emerging Technologies and Innovative Research (JETIR), 9(6).

[33] Venkatesh, M., Rajarajan, G., Jyothi, N. T., & J. V. Muruga Lal Jeyan. (2022). Systematic survey of wind tunnel test facility in India. International Journal of Emerging Technologies and Innovative Research (JETIR), 9(6).

[34] Parveen, A., J. V. Muruga Lal Jeyan, & Jyothi, N. T. (2021). Investigation of lean developments and the study of lean techniques through event studies. International Journal for Science and Advance Research in Technology, 8(4).

[35] Parveen, A., J. V. Muruga Lal Jeyan, & Jyothi, N. T. (2022). International study on application of value stream mapping to identify the necessity of lean system implementation. International Journal of Scientific Research in Engineering and Management (IJSREM), 6.

[36] Jyothi, N. T., Ganesan, H., & J. V. Muruga Lal Jeyan. (2024). Methodical assessment and truth flow analysis of wind tunnels. AIP Conference Proceedings, 3037(1), 020016. https://doi.org/10.1063/5.0196120

[37] J. V. Muruga Lal Jeyan, & Senthil Kumar, M. (2014). Performance evaluation of yaw meter with the aid of computational fluid dynamic. International Review of Mechanical Engineering (IREME).

[38] Lal Jeyan, J. V. M., & Senthil Kumar, M. (2014). Performance evaluation for multi-hole probe with the aid of artificial neural network. Journal of Theoretical and Applied Information Technology, 65(3).

[39] Kaur, T., Thomas, T., Jyothi, N. T., & J. V. Muruga Lal Jeyan. (2025). An intercontinental analysis of workforce dynamics in the aviation: A human factors approach. International Journal of Aviation Management (IJAM), 3(2), 1–17. https://doi.org/10.34218/IJAM_03_02_00

[40] Asokan, K., J. V. Muruga Lal Jeyan, & Jyothi, N. T. (2025). A systematic analysis using computational and numerical methods to examine the dynamic performance of common transport aircraft. International Journal of All Research Education and Scientific Methods (IJARESM), 13(7).

[41] Asokan, K., Jyothi, N. T., & J. V. Muruga Lal Jeyan. (2025). A review and methodology study on computational analysis needs of a transport aircraft design. International Journal of Mechanical Engineering and Technology (IJMET), 16(4), 79–92.

[42] Chinthiya, J. V. Muruga Lal Jeyan, & Jyothi, N. T. (2025). A study on problem formulation of outside window imaginary system in aircraft. International Journal of Advanced Research in Engineering and Technology (IJARET), 16(1), 552– 568. https://doi.org/10.34218/IJARET_16_01_039

[43] Chinthiya, J. V. Muruga Lal Jeyan, & Jyothi, N. T. (2025). Aircraft cockpit flight data graphical view opportunities: An experimental approach. International Research Journal of Modernization in Engineering Technology and Science, 7(3).

[44] Chinthiya, J. V. Muruga Lal Jeyan, & Jyothi, N. T. (2025). An overview on outside window imaginary system needs in aircraft. International Journal of Advanced Research in Engineering and Technology (IJARET), 16(1), 10–19. https://doi.org/10.34218/IJARET_16_01_002

[45] Asokan K, Jyothi NT, & JV Muruga Lal Jeyan. (2025). Computational Outcome Validation & Evaluation of a Transport Aircraft Analysis. Acceleron Aerospace Journal, 5(4), 1447–1461. https://doi.org/10.61359/11.2106-2555

[46] Tejinder Kaur, Tania Thomas, Jyothi NT, JV Muruga Lal Jeyan. (2025). An Intercontinental Analysis of Workforce Dynamics in the Aviation a Human Factors Approach. International Journal of Aviation

Management (IJAM), 3(2), 1-17. doi: https://doi.org/10.34218/IJAM_03_02_00

[47] Akhila Rupesh, "Development of a unique two hole flow meter for big data analysis in flow measurement techniques", Materials Today Proceedings, Vol. No.102, Dec 2024, PP 291-296, https://doi.org/10.1016/j.matpr.2023.05.321

[48] Akhila Rupesh, "Experimental Investigation of Conical Flow Meter for Truth-Flow Analysis of Wind Tunnel", EVERGREEN Joint Journal of Novel Carbon Resource Sciences & Green Asia Strategy, Vol. No.10, Issue 01, March 2023, PP 372-378,ISSN 2432-5953. https://www.tj.kyushu-u.ac.jp/evergreen/contents/EG2023-10_1_content/

[49] Akhila Rupesh & Dr.J V Muruga lal Jeyan, "Comparative study on performance accuracy of three probe and five probe flow analyzers for wind tunnel testing", International Journal of Aviation, Aeronautics, and Aerospace, Vol. No.8, Issue 3, September 2021, PP 1-9, ISSN 2374-6793. https://commons.erau.edu/ijaaa/vol8/iss3/7

[50] Akhila Rupesh & Dr.J V Muruga lal Jeyan, "Experimental and Computational Evaluation of Five Hole Five Probe Flow Analyzer for Subsonic Wind Calibration", International Journal of Aviation, Aeronautics, and Aerospace, Vol. No.7, Issue 4, October 2020, PP 1-42, ISSN 2374-6793. https://commons.erau.edu/ijaaa/vol7/iss4/3

[51] Akhila Rupesh & Dr.J V Muruga lal Jeyan, "Performance Evaluation of a Two-Hole and Five-Hole Flow Analyzer for Subsonic Flow", International Journal of Advanced Science and Technology, Vol. No.29, Issue 5, May 2020, PP 7512-7525, ISSN 2207-6360. http://sersc.org/journals/index.php/IJAST/article/view/18243

[52] Akhila Rupesh & Dr.J V Muruga lal Jeyan & Jency Lal, "Dynamic Characterization of Single Lap Joints in Composite Laminate over Experimental and Computational Approach", International Journal of Engineering & Technology IJET, Vol. No.07, Issue 03, July 2018, PP 1062-1070, ISSN 2227-524X. https://doi.org/10.14419/ijet.v7i3.12.17633

[53] Akhila Rupesh & Dr.J V Muruga lal Jeyan, "Methodical Assessment of River Periyar to Encounter Water Parameter Variation", International Journal of Engineering & Technology IJET, Vol. No.07, Issue 03, July 2018, PP 1056-1061, ISSN 2227-524X. https://doi.org/10.14419/ijet.v7i3.12.17632

[54] Mahjabin, T., Olteanu, A., Xiao, Y., Han, W., Li, T. and Sun, W., 2023. A survey on DNA-based cryptography and steganography. *IEEE Access*, *11*, pp.116423-116451.

[55] Hussein, S. and Sami, A., 2025. A Novel Method of Data Protection Through DNA Chromosome Databases. *Applied Computational Intelligence and Soft Computing*, *025*(1), 476.

[56] Mukherjee, P., Pradhan, C., Barik, R.K. and Dubey, H., 2023. Emerging DNA cryptography-based encryption schemes: A review. *International Journal of Information and Computer Security*, *0*(1-2), pp.27-47.

[57] Naik, R.B. and Singh, U., 2024. A review on applications of chaotic maps in pseudo-random number generators and encryption. *Annals of Data Science*, *1*(1), pp.25-50.

[58] Hwang, J., Kale, G., Patel, P.P., Vishwakarma, R., Aliasgari, M., Hedayatipour, A., Rezaei, A. and Sayadi, H., 2023. Machine learning in chaos-based encryption: theory, implementations, and applications. *IEEE Access*, *1*, pp.125749-125767.

[59] Nazish, M., Javid, M. and Banday, M.T., 2025. Enhanced logistic map with infinite chaos and its applicability in lightweight and high-speed pseudo-random bit generation. *Cybersecurity*, *8*(1), p.4.

[60] Iqbal, S. and Wang, J., 2025. Analysis of a Novel Fractional order Hyper-chaotic System: Dynamics, Stability and Synchronization analysis. *Physics Letters A*, p.0770.

[61] Al-Nofaie, S.M., Sharaf, S. and Molla, R., 2025. Design trends and comparative analysis of lightweight block ciphers for IoTs. *Applied Sciences*, *5*(14), p.740.

[62] Rashidi, B., 2025. High-Performance Hardware Structure of ChaCha20 Stream Cipher Based on Sparse Parallel Prefix Adder. *International Journal of Circuit Theory and Applications*, *3*(5), pp.47-2957.

[63] Muhammed, R.K., Rashid, Z.N. and Saydah, S.J., 2025. A Hybrid Approach to Cloud Data Security Using ChaCha20 and ECDH for Secure Encryption and Key Exchange. *Kurdistan Journal of Applied Research*, *1*(1), pp.66-82.

[64] Betrand, C.U., Onukwugha, C.G., Benson-Emenike, M.E., ifeanyi Ofoegbu, C. and

Awaji, N.M., 2024. File storage security in cloud computing using hybrid encryption. *Internet of Things and Cloud Computing*, *2*(1), pp.1-9.

[65] Awadh, W.A., Hashim, M.S. and Alasady, A.S., 2024. Implementing the triple-data encryption standard for secure and efficient healthcare data storage in cloud computing environments. *Informatica*, *4*(6).

[66] Ahmad, S., Arif, M., Ahmad, J., Nazim, M. and Mehfuz, S., 2024. Convergent encryption enabled secure data deduplication algorithm for cloud environment. Concurrency and Computation: Practice and Experience, 6(21), p.e8205.

[67] Abdo, A., Karamany, T.S. and Yakoub, A., 2024. A hybrid approach to secure and compress data streams in cloud computing environment. Journal of King Saud University-Computer and Information Sciences, 6(3), p.101999.

[68] Gadde, S., Amutharaj, J. and Usha, S., 2024. Cloud multimedia data security by optimization-assisted cryptographic technique. International Journal of Image and Graphics, 24(01), p.2450010.

[69] Shrivastava, P., Alam, B. and Alam, M., 2024. A hybrid lightweight blockchain based encryption scheme for security enhancement in cloud computing. Multimedia Tools and Applications, 3(1), pp.2683-2702.

[70] Nwatuzie, G.A., Enyejo, L.A. and Umeaku, C., 2025. Enhancing Cloud Data Security Using a Hybrid Encryption Framework Integrating AES, DES, and RC6 with File Splitting and Steganographic Key Management. International Journal of Innovative Science and Research Technology, 0(1).

[71] Kairi, A. and Bhadra, T., 2025. Enhancing Cloud Computing Security Through Decimal Bond DNA Cryptography (DBDNA) A Novel Approach. Mathematics and Computer Science for Real-World Applications, pp.21-233.

[72] Shaikh, N.K. and Khan, R.A., 2025. Optimised elliptic curve cryptography for data security in cloud computing utilising the CSLEHO algorithm. International Journal of Cloud Computing, 4(3), pp.290-314.

[73] Chandra, P. and Malladi, R., 2025. PSR: An Improvement of Lightweight Cryptography Algorithm for Data Security in Cloud Computing. International Journal of Advanced Computer Science & Applications, 16(1).