Revista Innovación y Software
Vol. 7, No. 2, Mes Septiembre - Febrero, 2026
ISSN: 2708-0935
Pág. 28-49
https://revistas.ulasalle.edu.pe/innosoft

Esta obra está bajo una Licencia Creative Commons Atribución 4.0 Internacional.

Tipo de artículo: Artículos originales
Temática: Inteligencia artificial. Redes y seguridad informática
Recibido: 29/05/2026 | Aceptado: 14/07/2026 | Publicado: 30/09/2026

Identificadores persistentes:
DOI: 10.48168/innosoft.s30.a369
ARK: ark:/42411/s30.a369
PURL: 42411/s30.a369

Una Variante Reducida del Esquema Salsa20 Ruidoso Aleatorio con Rendimiento Mejorado

A Reduced Variant of the Random Noisy Salsa20 Scheme with Improved Performance

Edgar Rangel Lugo[1][0000-0002-9611-8323]*, **Kevin Uriel Rangel Ríos**[2][0009-0001-7029-7183], **Cinthya Maybeth Rangel Ríos**[3][0009-0007-3503-5157], **Leticia Roque Correa**[4][0009-0006-3913-0816], **Simón Marcelo Peñaloza**[5][0009-0008-7122-8340], **César Del Ángel Rodríguez Torres**[6][0000-0001-9198-4372], **Aracely Salgado Mendoza**[7][0009-0002-6247-0991], **Mauricio De La Cruz Ramos García**[8][0009-0002-5381-908X]

[1]Tecnológico Nacional de México. I.T. de Ciudad Altamirano. Av. Pungarabato Ote. S/N, Col. Morelos, Ciudad Altamirano, Guerrero, México. erangel_lugo@hotmail.com
[2]Tecnológico Nacional de México. I.T. de Ciudad Altamirano. Av. Pungarabato Ote. S/N, Col. Morelos, Ciudad Altamirano, Guerrero, México. kgvppro@gmail.com
[3]Tecnológico Nacional de México. I.T. de Ciudad Altamirano. Av. Pungarabato Ote. S/N, Col. Morelos, Ciudad Altamirano, Guerrero, México. maybethrangelrios@gmail.com
[4]Tecnológico Nacional de México. I.T. de Ciudad Altamirano. Av. Pungarabato Ote. S/N, Col. Morelos, Ciudad Altamirano, Guerrero, México. leticia.rc@cdaltamirano.tecnm.mx
[5]Tecnológico Nacional de México. I.T. de Ciudad Altamirano. Av. Pungarabato Ote. S/N, Col. Morelos, Ciudad Altamirano, Guerrero, México. simon.mp@cdaltamirano.tecnm.mx
[6]Tecnológico Nacional de México. I.T. de Ciudad Altamirano. Av. Pungarabato Ote. S/N, Col. Morelos, Ciudad Altamirano, Guerrero, México. cesardelangel.rt@cdaltamirano.tecnm.mx
[7]Tecnológico Nacional de México. I.T. de Ciudad Altamirano. Av. Pungarabato Ote. S/N, Col. Morelos, Ciudad Altamirano, Guerrero, México. aracely.sm@cdaltamirano.tecnm.mx
[8]Tecnológico Nacional de México. I.T. de Ciudad Altamirano. Av. Pungarabato Ote. S/N, Col. Morelos, Ciudad Altamirano, Guerrero, México. mauriciodelacruz.rg@cdaltamirano.tecnm.mx

*Autor para correspondencia: erangel_lugo@hotmail.com

Resumen
Los métodos de cifrado dinámico, pueden mitigar algunos riesgos financieros derivados de la pérdida de datos al generar múltiples textos cifrados por cada texto plano. En este trabajo aplicamos estrategias aleatorias ruidosas (random noisy) y reducidas ruidosas (reduced noisy), como alternativas para mejorar el esquema de cifrado: random noisy Salsa20, un área no estudiada previamente. Los esquemas ruidosos aleatorios (random noisy) pueden mejorar la seguridad de datos digitales pero incrementan la sobrecarga de almacenamiento. Proponemos Salsa20 ruidoso reducido (reduced noisy Salsa20), una variante que incorpora el camuflaje de texto cifrado para resolver esta limitación. El método se evaluó frente a el estándar de Salsa20 y la alternativa ruidosa aleatoria: random noisy Salsa20. Los resultados muestran una reducción del tamaño del texto cifrado de hasta 33 % y una mayor eficiencia de almacenamiento en disco, sin pérdida de rendimiento dinámico. Este equilibrio hace que el esquema sea viable para organizaciones que implementan Salsa20. El enfoque ruidoso reducido (reduced noisy) mantiene la seguridad frente al criptoanálisis y reduce los costos de almacenamiento.

Revista Innovación y Software
Vol. 7, No. 2, Mes Septiembre - Febrero, 2026
ISSN: 2708-0935
Pág. 28-49
https://revistas.ulasalle.edu.pe/innosoft

Este artículo detalla el algoritmo, la configuración experimental y la comparación de rendimiento, introduciendo reduced noisy Salsa20, como una alternativa eficiente de cifrado dinámico.

Palabras claves: Aplicaciones de criptografía, cifrado de datos dinámico, reducción del cifrado dinámico ruidoso.

Abstract
Dynamic encryption methods can mitigate financial risks from data loss by generating multiple ciphertexts per plaintext. We apply the reduced noisy and random noisy strategies to enhance Salsa20 scheme, an area not previously studied. Random noisy strategies can improve digital data security but increase storage overhead. We propose reduced noisy Salsa20, a variant that incorporates ciphertext camouflaging to address this limitation. The method was evaluated against standard Salsa20 and random noisy Salsa20. Results show a ciphertext size reduction of up to 33 % and improved disk storage efficiency, with no loss of dynamic performance. This balance makes the scheme viable for organizations deploying Salsa20. The reduced noisy approach maintains security against cryptanalysis while lowering storage costs. This paper details the algorithm, experimental setup, and performance comparison, introducing the reduced noisy Salsa20 as an efficient dynamic encryption alternative.

Keywords: *Applications of cryptography, dynamic encryption methods, reduced noisy injection strategies.*

Introduction

A cybersecurity strategy [1]- [3], is deemed insufficient if any of its methods is susceptible to cyberattacks [4]- [8] leading to theft of sensitive digital data [8]- [9]. In practice, this results in substantial financial losses for organizations [4]- [5], [8], [10]- [11]. Typical cases [12]- [17] involve phishing, fraudulent calls, and financial scams targeting social networking platforms, banking systems, large markets, energy networks, and e-commerce [4]- [5]. Three main strategies address digital data theft: Regular updates of cybersecurity strategies [4], *dynamic encryption* methods [5], and *noisy injection* strategies on ciphertext [5]- [7], [9]- [10], which have shown promise in practical domains. This study focuses on encryption methods [8] that utilize *random noisy* and *reduced noisy* strategies [5]- [7].

Data encryption converts plaintext (Si) to ciphertext (Ci). Decryption is the reverse process [8]- [10], [14], [18]. These methods compute statistics or mathematical equations to transform plaintext into ciphertext, decipherable only by authorized entities [8], [10]- [11]. Here, Si denotes original input data and Ci the encrypted output. An encryption method is *dynamic* if it produces unique results for the same plaintext input each time, *static methods* produce identical outputs [8]. *Symmetric algorithms* use a single secret key, whereas *asymmetric algorithms* use a private-public key pair [8]. *Asymmetric algorithms* include *RSA (Rivest-Shamir-Adleman)* [3], [5]- [6], [8], [13]- [14], [19]- [27], *ECC (Elliptic Curve Cryptography)* [5]- [6], [8], [19]- [23], [28]- [31], and *ElGamal* [13], [19]- [21], among others. These *asymmetric algorithms* have shown *dynamic encryption* results [5], [8], [11]. *Symmetric algorithms* include *DES (Data Encryption Standard)* [3], [13]- [14], [22]- [23], [27], [32]- [33], *3DES* or *TripleDES (Triple Data Encryption Standard)* [14], [21]- [23], [34], *Blowfish* [22]-

[23], [35]- [38], *Salsa20* [2], [38], *ChaCha20* [38]- [39], *GOST (GOsudarstvennyy STandart)* [6], and *AES (Advanced Encryption Standard)* [13], [21]- [23], [26]- [27], [40]- [41], to mention a few. Some studies [5], [8], reveal that *symmetric algorithms* often report *static* ciphertext results. This does not imply all are vulnerable schemes [5].

Dynamic encryption methodologies offer an alternative for increasing noisy and redundant information in *ciphertext* outputs [4]- [5], [9]- [11]. Therefore, *noisy injection* strategies are recommended as an encryption measure [5]- [7]. *Noisy injection* means that *plaintext* or *ciphertext* contains additional ASCII or UTF-8 characters not part of the original message, which achieve camouflaging or a special operation with the *plaintext* or *ciphertext* as an obfuscation measure. A key *hypothesis* is that this scheme can create confusion for *cybercriminals* [5], [8]. *Noisy injection* strategies utilize *random methods* with or without replacement. These methodologies can be employed in AI-based intelligent models [42]- [55], like *genetic algorithms* (*GAs*) [1], [9]- [10], [16]- [17], [56]- [62], *Monte Carlo (MC1)* [59], and *artificial neural networks* [57], [63]- [64], to mention few. Several authors have explored AI-based *cryptography* alternatives [1]- [4], [8], [10]- [11], [14], [16]- [17], [24], [27], [48]- [50], [60]- [62]. For a comprehensive review of AI applications in *cryptography*, see [1], [3], [16], [24], [27], [48]- [49], [50], [58], [60]- [62].

In this context, *noisy injection* strategies [4]- [11] group into several categories. First, *pseudo-hexadecimal* format is used. The *"Noised" random pseudo-hexadecimal GAs* and *"Noised" random pseudo-hexadecimal* without *GAs* methodologies were presented in [9], [10]. Later, *four dynamic* alternatives based on a *pseudo-hexadecimal* schema, known as *noisy random pseudo-hexadecimal* strategies, were presented in [8]. Their goal is to confuse *cybercriminals* by injecting noise using a novel *pseudo-hexadecimal* format. Second, AI-based *noisy injection* combines with the *nearest neighbor (1-NN)* rule [46]- [47], [65]- [67]. The *"Noised" random 1-NN* with *hexadecimal* encoding was introduced in [11]. Similarly, combining *"Noised" random pseudo-hexadecimal* format with *1-NN* was explored in [68]. These *pseudo-hexadecimal* schemes were limited to *plaintext* applications and its implementation is based on *Windows* [69] plataforms with *Python* [70] language, as well as *Android* [71] environment, using *PyDroid3* [72]. Both cases, *Python's encryption* libraries [73]- [75] were employed. Third, *random Caesar II mod 120* [4]- [7], [11], and *reduced random Caesar mod 105* [4] are utilized for *noisy injection* and *camouflaging text*, applied to *plaintext* [4]- [5], [11], and *ciphertext* [5]- [7] obtained through *standard encryption* algorithms. These methods are referred to as the *random noisy* and *reduced noisy* strategies [5]- [7], respectively.

The purpose of the *reduced* scheme is to camouflage and reduce *ciphertext* size by at least one-third. Studies on *pseudo-hexadecimal* and *random noisy* approaches show that *double noisy injection* on *ciphertext* outputs improves data safety, though it sacrifices disk storage space [8]. Reference [5] presents eight strategies integrating the *random noisy* approach with *standard encryption*: *random noisy DES*, *random noisy 3DES, random noisy*

RC4, random noisy Blowfish, random noisy WEP, random noisy AES, random noisy RSA-2048, and *random noisy ECIES SECP-256-R1.* Similarly, *random noisy GOST (R 34.12-2015)* was proposed in [6], and *random noisy Camellia* in [7]. Studies regarding *random noisy* implementations for *Salsa20, AES-256, ChaCha20*, and *CAST-128* were reported in [2], [5], [76]- [78]. Moreover, references [79] and [80] explore an improved *random noisy IDEA-128* scheme and analyze the *asymmetric random noisy RSA* algorithm, respectively. These approaches integrate standard ciphers with *noisy injection* techniques based on *random Caesar II mod 120* [4]- [5]. Various *reduced noisy* strategies [4]- [5], [77] have proven effective at obfuscating *ciphertext* [4]- [5], such as combining *reduced random Caesar mod 105* [5], [77], with conventional algorithms. Reference [5] details eight specific strategies: *reduced noisy DES, reduced noisy 3DES, reduced noisy RC4, reduced noisy Blowfish, reduced noisy WEP, reduced noisy AES-256, reduced noisy RSA-2048*, and *reduced noisy ECIES SECP-256-R1.* Finally, additional research on *ciphertext* camouflaging via *reduced noisy AES-256* can be found in [4] and [77].

Random Caesar and *reduced random Caesar* [4]- [5] are based on *traditional Caesar* [4], [9], [13]- [14]. They stand out because they enable *dynamic encryption* [5]. *Traditional Caesar* uses a fixed shifting value K, given by Ci = Si + K *mod 26* [13]- [14]. *Random Caesar* and *reduced random Caesar* utilize varying shifting values Ki for each character Si, chosen randomly with replacement: Ci = Si + Ki *mod N*, where N =105 for *reduced random Caesar* and N =120 for *random Caesar II.* Here, *mod N* indicates the *encryption alphabet* size and the maximum value in vector Ki [4], [6], [9], [11]. With N = 95 (*random Caesar II* version 1), *ASCII* characters from *space (32)* to *'˜' (126)* are used. N = 120 (*random Caesar II* version 2) covers 30 to 150 in the *ASCII* table. N = 105 (*reduced random Caesar*) corresponds to selected *ASCII* characters with a range of 0 to 105 (variant version 1) and 0 to 135 (variant version 2). A novel variant with a range of 30 to 105 for *ASCII* characters has only been applied to the *reduced noisy CAST-128* [81]. For different N values *(random Caesar I)*, Ki falls between 0 and N . *Random Caesar* and *reduced random Caesar* include a second phase [5] to confuse attackers, calculating the final package as RandomFinalPackage = Ci & Ki & OrdChr(Ci), or ReducedFinalPackage = Ci & Char(Ki), where & denotes concatenation, Char(Ki) returns the *ASCII* character for value Ki, and OrdChr(Ci) appends character Ci when N =120 or converts it to its *ordinal* value depending on the mode. The *reduced random Caesar* and *random Caesar II mod 120* have been employed as randomness mechanisms in [4]- [5], [7]. Applying *random Caesar II mod 120* to *ciphertext* is known as the *random noisy* strategy, whereas applying *reduced random Caesar* to *ciphertext* is known as the *reduced noisy* strategy. Both approaches combine standard encryption with *noisy injection* into *ciphertext.*

We assessed the *random noisy* strategy utilizing *standard symmetric* algorithm *Salsa20.* A novel alternative, *reduced noisy Salsa20*, was developed and is introduced in this study. This research explores *reduced noisy* and *random noisy* schemes in *cybersecurity*, specifically applying *noisy injection* to *ciphertext* from *standard algorithm* to potentially deceive *cybercriminals* [8]. The scope covers two situations. First, evaluating standard

algorithm *Salsa20* as a *static encryption* scheme to facilitate comparison with related work [5]- [7]. Second, assessing the proposed *random Caesar* and *reduced random Caesar* variants on *ciphertext* as *dynamic encryption* schemes to demonstrate improvements in randomness and security over traditional methods. Thus, *random noisy Salsa20* and *reduced noisy Salsa20* were developed as comparative benchmarks for *random noisy* and *reduced noisy* strategies, respectively. These approaches were previously assessed using five-fold *cross-validation*, employing a novel *modification* [4]- [5], [10]. This work expresses performance in terms of *global average* or *accuracy* [46]- [47], [65]- [66].

We present findings of an extensive project investigating digital data theft, including experimental results. The study analyzed cases where inadequate *static encryption* caused security concerns. Experiments focused on replacing *static encryption* with *reduced noisy* and *random noisy* strategies as *dynamic* alternatives [4]. Several *ciphertext* exemplars from *random noisy* and *reduced noisy* schemes are shown here. Based on observations, *noisy injection* on *ciphertext* output is recommended due to excellent performance as a *dynamic encryption* indicator. This paper introduces *reduced noisy Salsa20*, a novel *dynamic encryption* approach. We compare it against *random noisy Salsa20* and the *standard Salsa20* cipher. The study is motivated by the scarce research on *reduced noisy Salsa20* and the absence of documented vulnerabilities to *cyberattacks*.

Computational methodology

The implementation of *static encryption* algorithms as a substitute for current *cybersecurity* measures fails to ensure adequate data protection. In this context, a *dynamic encryption* strategy is proposed [4]. To address the issue of data theft, this study assesses two *dynamic* approaches utilizing *random noisy* and *reduced noisy* alternatives [4]- [5]. The present work is characterized as *experimental* and *exploratory*, given that it presents an innovative solution based on a modified *random noisy Salsa20* scheme incorporating *reduced noisy Salsa20* proposal, which is introduced herein for the first time.

The present research required *hardware*, *software*, and *data* resources. Experimental evaluations were executed on a personal computer configured with a 2 GHz CPU, 4 GB of *RAM*, and 32 GB of storage capacity. *Software* development for the *standard Salsa20* algorithm and its innovative *random noisy* and *reduced noisy* approaches [5] was performed in a *Windows 10* environment [69] using *Python 3* [70]. Furthermore, to ensure consistency with literature [5]- [7], trials were extended to a mobile platform running *Android 9* [71] and *PyDroid3* [72]. The analysis demonstrated comparable performance across both environments.

The *datasets* consisted of *training samples (TS)* [6]- [7], [46]- [47], [65], comprising 1000 randomly generated exemplars. Each entry is defined by five features. The data includes *encryption* and *decryption* parameters, such as the *ciphertext* (Ci) represented by the pair (Test1, Test2), along with metrics like *encryption time*

(TC), *decryption time* (TD), *error rate*, and class *label*. The pattern follows the format TP = [(Test1, Test2), TC, TD, Error, Label], enabling direct comparison with existing literature [5]- [7].

Table 1 presents two *ciphertext* examples for the *label* '▒Welcome▒?'. This *label* represents the *plaintext* (Si), simulating a *password* and containing noise characters '▒' and '?' (ordinal values 9619 and 65533, respectively). *Encryption* and *decryption* times were measured in milliseconds, while TC, TD, and Error rate were stored as double-precision values. The *encryption* process transforms the *plaintext* into a *ciphertext* tuple: (Test1, Test2), and computes TC, revealing the *dynamic* behavior of the output. Similarly, TD is recorded during the *decryption* phase. All data, including *times* and *error* rates, is stored in the *TS* following a standardized structure. The *error* rate is calculated based on character mismatches. If the resulting *ciphertext* does not correspond correctly to the original *plaintext* (Si), the Error value represents the proportion of incorrect characters. Both *plaintext* and *ciphertext* are encoded as character sequences using *ASCII* or *UTF-8*, with a maximum length of 255 characters.

This information was restructured using a *modified cross-validation* techniques [4]- [7]. The updated format was then applied individually to each encryption strategy. To facilitate result analysis, *Table 1* presents the *arithmetic mean* and *standard deviation* obtained for the *plaintext* '▒Welcome▒?'. Finally, distinct *datasets* were generated for each algorithm, including those implementing *random* noisy and *reduced noisy* strategies.

The first strategy involved the use of the *standard symmetric encryption* algorithm: *Salsa20*. This method was tested as a *static encryption* scheme applied directly to the *plaintext*, allowing for comparison with findings reported in existing literature [5]- [7]. The experiments were conducted on five *training samples* using a *modified cross-validation* approach [4]- [5], evaluating each algorithm individually. The *standard encryption* algorithm was implemented in *Python* [70] using libraries such as *Cryptography* [73] and *PyCryptodome* [74]- [75], which were installed and subsequently imported into the source code as follows: from Crypto.Cipher import Salsa20. In the case of *ciphertext* produced by the *Salsa20* algorithm, the computation source-code is: from Crypto.Cipher import Salsa20 ; Si = ''.join([chr(9619), 'W', 'e', 'l', 'c', 'o', 'm', 'e', chr(9619), chr(65533)]) ; Key = "00000000000000000000000000000001" ; Num = 32 ; Nonce = "00000001" ; PlainText = Si.encode() ; A = Salsa20.new ; Mode = None ; Ci = ((A(Key.encode() , Nonce.encode())).encrypt(PlainText)).hex(); print(Ci). Where: A refers to the specific algorithm instance. Key, the encoded *secret key*. Mode, a valid operational mode for the algorithm. Nonce, a valid value specific to the *Salsa20* algorithm. The PlainText is encoded Si, while encrypt() returns a *ciphertext* object. Num, the maximum number of bytes allowed in the sequence. Ci is the *ciphertext* and hex() function that converts byte values into their *hexadecimal* string representation.

Second adopted approach involves the separate application of *random noisy* and *reduced noisy* schemes [5] for

dynamic encryption. Reference [4] highlights these methods as suitable solutions for enhancing output randomness. Following the framework in [5], the noise layers were applied to the results of standard algorithms. This work presents *random noisy Salsa20* and *reduced noisy Salsa20* as innovative contributions. Implementation was performed in *Python* [70]. The tests involved combining the *standard Salsa20* ciphertext with noise generators, specifically *random Caesar II mod 120* and *reduced random Caesar mod 105*, applied independently. This setup ensures valid comparison with previous studies [5]- [7].

Each *encryption algorithm* was evaluated individually on the five *TS* using an iterative process with five *cross-validation* repetitions. A modified cross-validation [4]- [5] approach was employed to compute the overall *mean* and *standard deviation* for each strategy. As outlined in [5], the proposed methods involve noise injection into the *ciphertext*. The procedures for calculating the *random noisy* and *reduced noisy* strategies are detailed in [5]- [7], and are computed as follows: RandomNoisyi = Char (Ord (StandardEncryptioni) + Ord(Ki)) & Char (Ki) & Char (StandardEncryptioni), with *mod 120*, and ReducedNoisyi = Char (Ord (StandardEncryptioni) + Ord(Ki)) & Char (Ki), with *mod 105*, respectively.

Replacing the *plaintext* with *ciphertext* streamlined the operation, as demonstrated in [5]. The adjustments to *mod* N = 120 and *mod* N = 105 were implemented because the FinalPackage (RandomNoisyi or ReducedNoisyi) stores only character types. As discussed in [5], various versions of *random noisy* and *reduced noisy* schemes have been explored. In this study, only the *Salsa20* algorithm was used to generate the StandardEncryptioni *ciphertext*, as previously mentioned. Similarly, the *random noisy Salsa20* and *reduced noisy Salsa20* variants were implemented and calculated as follows: RandomNoisySalsai = Char (Ord (StandardSalsaEncryptioni) + Ord(Ki)) & Char (Ki) & Char (StandardSalsaEncryptioni) *mod 120*, and ReducedNoisySalsai = Char (Ord (StandardSalsaEncryptioni) + Ord(Ki)) & Char (Ki) *mod 105*, respectively. Where: The + *operator* denotes the *sum* function, while the & operator represents the *concatenation* function. The *shifting* vector Ki contains random integer values. The StandardEncryptioni parameter refers to the *ciphertext* obtained using a *standard encryption* algorithm, as described in [2], [4]- [5]. Specifically, StandardSalsaEncryptioni corresponds to the *ciphertext* generated by the *Salsa20* algorithm. Each strategy was applied independently. The RandomNoisyi and ReducedNoisyi sequences represent the FinalPackage derived from general *random noisy* and *reduced noisy* strategies, according to [2], [4]- [5]. In contrast, RandomNoisySalsai and ReducedNoisySalsai denote the FinalPackage obtained by applying the *random noisy Salsa20* and *reduced noisy Salsa20*, respectively. The Ord() function returns the *ordinal* value of a character or integer, while the Char() function converts a numerical value into its corresponding character representation using *ASCII* or *UTF-8* encoding.

In the *random noisy* and *reduced noisy* approaches, the *plaintext* is first encrypted using a *standard algorithm*. Subsequently, the *random Caesar II mod 120* and *reduced random Caesar mod 105* transforms are applied separately to the *ciphertext*, adding an extra layer of security [5]- [7]. As highlighted in [5], this method differs

from *double encryption* and avoids the repeated use of *standard algorithms*, a practice that could potentially be exploited by attackers using computational *cryptanalysis*. According to [5], *noisy injection* should be applied selectively to prevent attackers from easily predicting its location. By introducing noise only in specific sections, malicious actors face the complex challenge of locating the modified characters, significantly increasing the difficulty of decryption.

For result evaluation, a *modified cross-validation method* [4]- [7] was implemented to introduce internal bias into the discrimination process, as previously discussed. These findings support the recommendation of using *noisy injection*. Applying *random noisy* or *reduced noisy* schemes, it results in *dynamic ciphertext*, significantly strengthening security protection for organizations [4]- [5].

Results and Discussion

All experiments were conducted using the *training samples (TS)* as previously described. The estimated *error*, *encryption time* (TC), and *decryption time* (TD) were computed separately for each encryption strategy. A *modified* five-fold *cross-validation* method [4]- [5] was employed to ensure direct comparability with existing studies [5]- [7]. According to the literature [7], *traditional cross-validation* [46]- [47], [56], [59], [63], [65]- [67], typically divides the *dataset* into five subsamples (approximately 20 % each). One subset is used for testing *(MC)*, while the remaining four (80 %) form the *training set*. The process is repeated five times to calculate *averages* and *standard deviations* [66]. However, this *traditional* procedure is not directly applicable to *encryption evaluation*, as this research does not involve *model training* or *testing phases*. Instead, the process consists of *encrypting* the *plaintext*, *decrypting* the result, and storing both vectors for comparison of the TP *pattern*. For the *random noisy* and *reduced noisy* strategies, the *encryption* method is applied to the *ciphertext* to generate two *noisy* outputs (Test1 and Test2).

The novel *cross-validation modification* [4]- [5] was updated to adopt an approach that does not rely on the *MC* for assessing *global accuracy* [66]. The *encryption/decryption* process uses this modified method [4]- [7] that omits the evaluation step typically performed with the *test set*. Instead, the Error is computed using four-fifths of the *TS*, sequentially omitting approximately 20 % in each of the five iterations. This schema simulates performance in different scenarios by excluding subsets of data, ensuring convergent results with an optimistic bias (i.e., practical performance is expected to be equal or better) [2], [7]. This approach allows for the assessment of the *estimated error* and other numerical features. If the *decrypted* text does not match the original, the error percentage is calculated based on the proportion of incorrectly decoded characters. As in traditional methods [5], the process is repeated five times, extracting a different subsample each time. This allows calculating the *mean* and *standard deviation* for numerical attributes such as *encryption time* (TC),

decryption time (TD), and *error* percentage. Since the analysis was performed globally without distinguishing between classes, as an ambiguity noted during experimentation, class-specific results were not considered. Although this topic could be addressed in future studies, it did not affect the overall *accuracy* of the evaluated strategies.

Consequently, experimentation was limited to two main scenarios. First, the *standard Salsa20* algorithm was compared with its *random noisy* variant. The *Salsa20 static* scheme was analyzed to allow comparison with findings reported in the literature [5]- [7]. Second, the *random noisy* and *reduced noisy* strategies [2], [5] were evaluated using the *ciphertext*, as previously described. Specifically, *random noisy Salsa20* [2] was compared against *reduced noisy Salsa20*, which is introduced here as a novel alternative. These two strategies were tested separately to inject noise into the *ciphertext* following a *dynamic encryption* approach. After processing all samples for each strategy, *global average* results were computed using the *updated cross-validation* method [4]- [7] described above. These results are presented in *Table 1*, with *standard deviations* shown in parentheses. For consistency with previous studies [5], [7], *encryption* and *decryption* times are reported in milliseconds in the columns labeled TC and TD, respectively. The procedure was repeated five times for each method, and the values shown correspond to the *averages* obtained using the *modified cross-validation* approach [5]- [11].

The parameters for *Salsa20* included a 32-bit *secret key* ('00000000000000000000000000000001'), a *nonce* value ('00000001'), and an *operation mode* set to 'None'. Other requirements were defined using default values via the *PyCryptodome* library [74]- [75], with outputs generated in *hexadecimal* format. The *Table 1* presents the calculated *times* and *error* rates, structured to match the formats used in [5], [7]. For the *noisy* strategies, the processes are inherently *random*, meaning they generate different, *dynamic encryption* results even with identical inputs. The table displays *average* times and *deviations*, alongside two *ciphertext* examples (Test1 and Test2) for each strategy. Using the *plaintext* "▓Welcome▓?", experiments occasionally yielded varying results, although the majority were successful.

Compared to *noisy injection* strategies, the *standard symmetric Salsa20* algorithm proved non-substantially faster, achieving speeds between 1.0003 and 1.0201 *times* greater, representing a time difference of 0.0004 to 0.0202 milliseconds. Furthermore, *Salsa20* demonstrated the ability to handle inputs exceeding 255 characters, making it a robust choice when properly validated. Although *Table 1* shows *static* results for the *standard algorithm*, this does not imply that these schemes are inherently vulnerable.

Revista Innovación y Software
Vol. 7, No. 2, Mes Septiembre - Febrero, 2026
ISSN: 2708-0935
Pág. 28-49
https://revistas.ulasalle.edu.pe/innosoft

Tabla 1. Performance Metrics: Timing and Error Rates

Encryption Strategies	**TC**	**TD**	**% Error**	**Ci Length**	**Test 1**	**Test 2**
Salsa20 [2]	1.0030 (0.4828)	0.0882 (0.0058)	0 (0)	34	c35d5f4be432 84e385d7773 45fdf61d243	c35d5f4be432 84e385d7773 45fdf61d243
random noisy Salsa20 [2]	1.0232 (0.3491)	0.2717 (0.0311)	0 (0)	102	±N±ºº}H}6{ F{4ÇÇ*©D ©±}±¿¿Ps;s ··ÛvÛW$W³ {³q<qÖrÖˆl 5lÉÉ¦s¦ÅÅj5j* ÇcÇææ··h7h ääl:l[T	Â_ÂzGzhÌhÌ ±|±Á[Á\ãã7 Xo<oo=o»»· ·åån;nk3k]2P ¹¹⁄₄₄ÂÂUc/c¯z ¯ÊdÊ:àzà¬v ¬S£?£QQ_+ _ÁÁ
reduced noisy Salsa20	1.0026 (0.3983)	0.2426 (0.0054)	0 (0)	68	¥Bs@\'£?c² LeËi¨Cin;c1¡i iÎig¡i|G&[$Ng iˆ*c¨B´P²Lq ;i:uCm9S	ÌiX %|GÍiX# ©CuA²P¹Tii if‘,»V‘-¡iL± M i i iUiLÏi0 ÏiieÍiiixE
random Caesar II (mod 120) [5]	0.1461 (0.0108)	0.0518 (0.0011)	0 (0)	30	◕B◕AïïÇ[Ç ¬I¬øøĂĂ±L ±⁑₱□|□g]g	☀m☀ààÝxÝ¸ L¸´Q´ÝnÝ2 ÓnÓ◓@◓□A □_U_
reduced random Caesar (mod 105) [5]	0.1491 (0.0081)	0.0559 (0.0012)	0 (0)	20	■i¦O+¥9Êg(◁.□hbX	◒?G¿Z-ÌiÔe È[¼W□X□isi
Average	**0.6648 (0.4224)**	**0.1420 (0.0953)**	**0.0 (0.0)**	**50.8 (30.2681)**	**- - - -**	**- - - -**

Notably, the *random Caesar II* and *reduced random Caesar* approach applied directly to *plaintext* achieved an excellent balance. Specifically, *random Caesar II* with *mod 120* significantly improved encryption times.

Experiments yielded an *average encryption* time of just 0.1461 ms (± 0.0108) and *decryption time* of 0.0518 ms (± 0.0011), while *reduced random Caesar* with *mod 105* yielded an *average encryption time* of 0.1491 ms (± 0.0081) and *decryption time* of 0.0559 ms (± 0.0012). Regarding to reduce storage spacing disk. When applied to *ciphertext*, the *reduced noisy Salsa20* strategy–which combines *standard Salsa20* with *reduced random Caesar mod 105*–outperformed all other approaches evaluated on *ciphertext*. It consistently delivered the most balanced results. Compared to *random noisy Salsa20*, the novel *reduced noisy Salsa20* alternative showed speed improvements from 1.0205 *times* faster, with time differences from 0.0206 ms. Interestingly, the performance gap between *standard Salsa20* and its *noisy* variants was relatively small. For instance, *traditional Salsa20* is only 1.0201 *times* faster, a marginal difference of 0.0202 ms. Both variants supported the maximum input length of 255 characters without generating errors, which significantly increases confidence in their performance. However, it is important to note that experiments were limited to this length, and further testing is required. Each *encryption strategy* was evaluated using independently designed samples.

The *reduced random Caesar* scheme has proven effective in enhancing data security [4]- [5], [9], [11], and the *reduced noisy* strategy has demonstrated similar efficacy. *Table 1* confirms this behavior in the *global averages*. Notably, the *random noisy* alternative produces slightly longer *ciphertexts*, whereas the *reduced noisy* strategy compresses the data to approximately one-third of the original storage size. Both strategies generate *dynamic ciphertexts* that vary with each execution. However, as shown in *Table 1* (Test1 and Test2), the time required to generate the FinalPackage *ciphertext* was longer compared to *standard algorithms*. Consequently, execution times for *standard Salsa20* were shorter than for its *noisy* variants. These schemes present significant obstacles for potential attackers, who would need to determine each random Ki shift value beforehand, information that remains hidden. *Decryption* becomes even more complex when using the PartialNoisy proposal, a novel partial *noisy injection* scheme introduced in [5]. Such complexity makes unauthorized data discovery extremely challenging. The exploration of PartialNoisy was beyond the scope of this study and is suggested as a direction for future research.

Repeated application of these strategies yields more *dynamic* and secure results compared to *static* methods. These alternatives effectively increase the security level of both *plaintext* and *ciphertext*, thereby enhancing organizational data protection [5]. Nevertheless, the resilience of these schemes against specific *cyberattacks* remains to be fully evaluated. Comparing our findings with [4]- [5], the *reduced random mutation* method [5] appears to be an effective indicator of improved security.

This scheme achieves a balance between efficiency and security, producing short *ciphertexts* and fast processing times, while the Ki shifting applied to partial data ensures robust protection. According to [5], *mutation* strategies can mitigate the risk of data theft by introducing substantial noise (double) into the *ciphertext*. Consequently, approaches based on *reduced* randomness and *mutation* scheme show great promise [5]. However,

this study focuses on other methods, leaving *mutation*-based alternatives for future investigation. The *aims* and *hypotheses* were successfully fulfilled. Specifically, *reduced noisy Salsa20* is presented here as a novel *noisy injection* proposal that enables *dynamic encryption* and reduces *ciphertext* size to approximately one-third. It generates varying outputs even with identical inputs and parameters, effectively misleading potential attackers. These findings are supported by the data presented in *Table 1.*

Comparing *standard algorithms* with *reduced noisy* strategies suggests that *noisy injection* is a promising approach for organizations seeking enhanced security. This is particularly relevant where *traditional Salsa20* is already implemented, as the novel *reduced noisy Salsa20* can be easily integrated. According to [2], [5], Implementing a noisy scheme is recommended to strengthen data protection, offering organizations a reliable solution and opening new possibilities for improving digital security. Furthermore, modifications to the *dynamic encryption* methodology presented here warrant further exploration. By employing *reduced random mutation* scheme [4]- [5], *ciphertext* size in the FinalPackage can be significantly reduced–up to 33 %–compared to *random noisy* proposals. Other potential avenues include different *noisy injection* methods, particularly those combining simultaneous randomness with *artificial intelligence* using the *nearest neighbor* rule [8], [10]- [11], [46]- [47], [65], [67] and *pseudo-hexadecimal* encoding [8]- [10]. This indicates a wide range of opportunities for future research.

Conclusions

Regular updates to *cybersecurity* strategies are vital, yet they do not inherently guarantee absolute data protection. As noted in the literature [4], [8], well-known or inadequate *encryption* methods become vulnerable over time. To address this challenge, various *dynamic* approaches have been proposed [4]- [11]. This paper introduces a novel *noisy injection* strategy, termed *reduced noisy Salsa20*, which combines *standard encryption* with a *reduced random Caesar* cipher, specifically designed for practical organizational use. Moreover, we evaluated two *dynamic encryption* alternatives that utilize *noisy* strategies, to inject noise into *ciphertext* through *random methods*. The proposed methodology successfully overcomes limitations found in *standard algorithm*, offering a robust defense against cyber threats. Experimental results confirm that the *objectives* and *hypotheses* were successfully met. The *reduced noisy Salsa20* strategy generates *dynamic ciphertexts*, being compressing until one-third, producing variable outputs even with identical inputs, thereby increasing security and performance compared to traditional methods (see Table 1). Future work will focus on reducing size and *camouflaging ciphertext*, through *reduced random mutation* [5] schemes, as well as integrating *noisy injection* techniques with *artificial intelligence* and *pseudo-hexadecimal* [8]- [10] encoding, offering promising avenues for further research.

Acknowledgment

This work was partially supported by the Tecnológico Nacional de México (I.T. Ciudad Altamirano, project 24321, pending approval).

Contribución de Autoría

Edgar Rangel Lugo: Investigation, Methodology, Project administration, Software, Validation. **Kevin Uriel Rangel Ríos:** Conceptualization, Investigation, Software, Validation, Writing – review & editing. **Cinthya Maybeth Rangel Ríos:** Investigation, Validation, Writing – review & editing. **Leticia Roque Correa:** Investigation, Supervision. **Simón Marcelo Peñaloza:** Investigation, Validation. **César Del Ángel Rodríguez Torres:** Investigation. **Aracely Salgado Mendoza:** Validation. **Mauricio De La Cruz Ramos García:** Investigation.

Referencias

[1] B. Delman, "Genetic Algorithms in Cryptography," M.S. thesis [Thesis for the Degree of Master of Science in Computer Engineering], Department of Computer Engineering, Rochester Institute of Technology, RIT Scholar Works, Rochester, New York, 2004.(https://scholar.google.com.mx/scholar_url?url=https://repository.rit.edu/cgi/viewcontent.cgi%3Farticle%3D6460%26context%3Dtheses&hl=es&sa=X&ei=cbaZZoadNt246rQPnd604AU&scisig=AFWwaeaMfCM5ORUFQN6DU4LA3aEG&oi=scholarr, 23/03/2024).

[2] E. Rangel, K. U. Rangel, C. A. Bernal, L. González, C. A. Rodríguez, L. J. Ascencio, and R. I. Reynoso, "A Case Study of the Salsa20 Encryption Algorithm Using Random Noisy Injection Enhanced by Artificial Intelligence," Revista Innovación y Software – Innovation and Software Journal (ULASALLE), vol. 6, no. 2, pp. 134-158, Sep. 2025, doi: 10.48168/innosoft.s24.a317. (https://revistas.ulasalle.edu.pe/innosoft/article/view/317/413, 30/09/2025).

[3] J.C. Mendoza, "Demostración De Cifrado Simetrico Y Asimétrico," [Universidad Politécnica Salesian. Cuenca, Ecuador. ISSN: 1390-650X], Ingenius, Revista de Ciencia y Tecnología, núm. 3, 2008, pp. 46-53. (http://www.redalyc.org/articulo.oa?id=505554806007, 09/11/2024).

[4] E. Rangel and K.U. Rangel, "Novel Random Encryption Methods Based On Mutation Strategies Of Artificial Intelligence," SPCSJ: Scientific and Practical Cyber Security Journal [Published by Scientific Cyber Security Association in Tbilisi, Georgia], ISSN: 2587-4667, vol. 8, no. 3, pp. 84-91, septem-

ber issue 2024. (https://journal.scsa.ge/papers/novel-random-encryption-methods-based-on-mutation-strategies-of-artificial-intelligence/, 05/11/2024).

[5] E. Rangel, K.U. Rangel, and L. González, "Dynamic Encryption Methods Based On Noisy Injection And Camouflaging Ciphertext Strategies With Artificial Intelligence," SPCSJ, Scientific and Practical Cyber Security Journal [Published by Scientific Cyber Security Association in Tbilisi, Georgia, ISSN: 2587–4667], vol. 9, no. 1, pp. 82-104, march issue 2025. (https://journal.scsa.ge/papers/dynamic-encryption-methods-based-on-noisy-injection-and-camoufla ging-ciphertext-strategies-with-artificial-inte lligence/, 23/04/2025).

[6] ——, "Inyección De Ruido Para Encriptado De Datos Dinámico Con Inteligencia Artificial. Caso De Estudio: Algoritmo GOST R 34.12-2015," Revista Electrónica de Divulgación de la Investigación del SABES [Revista de la universidad del SABES, Editada por Sistema Avanzado de Bachillerato y Educación Superior en el Estado de Guanajuato. México, ISSN: 2007-3542], vol. 29, pp. 11-36, edición junio — 2025. (https://sabes.edu.mx/revista-electronica/28/docs/2_inyeccion_de_ruido.pdf, 01/08/2025).

[7] E. Rangel and K.U.Rangel, "Mejorando la seguridad del algoritmo Camellia, mediante la inyección de ruido sobre textos cifrados utilizando procesos basados en inteligencia artificial," INTELETICA, Revista de Inteligencia Artificial, Ética y Sociedad [Edited & Published by IBERAMIA, Iberoamerican Society of Artificial Intelligence – Sociedad Iberoamericana de Inteligencia Artificial in Valencia, España, ISSN: 3020-7444], vol. 2, no. 4, pp. 75-101, september 2025. DOI: https://doi.org/10.4114/INTELETICA.vol2iss4pp75-101. (https://inteletica.iberamia.org/index.php/journal/article/view/45, 03/09/2025).

[8] E. Rangel, K.U. Rangel, L. González, A. Ortiz, and C.A. Rodríguez, "Four Dynamic Encryption Alternatives With Artificial Intelligence Based On Pseudo-Hexadecimal Noisy Injection Schema For Handling The Theft Of Digital Data Problem," SPCSJ, Scientific and Practical Cyber Security Journal [Edited by SCSA - SPCSJ - BOAI, Published by Scientific Cyber Security Association in Tbilisi, Georgia, e-ISSN: 2587-4667], vol. 9, no. 3, pp. 59-77, june 2025, https://journal.scsa.ge/issue/june-2025/, (https://journal.scsa.ge/papers/four-dynamic-encryption-alternatives-with-artificial-intelligence-based-on-pseudo-hexadecimal-noisy-injection-schema-for-handling-the-theft-of-digital-data-problem/, 27/07/2025).

[9] E. Rangel, K.U. Rangel, J. Medrano, C.A. Bernal, and L. González, "Algoritmo Genético Para Cifrado De Datos, Basado En Un Nuevo Concepto Pseudo-Hexadecimal Con Inteligencia Artificial," Tecnológico Nacional De México, Instituto Tecnológico de Ciudad Altamirano, Sexto (VI) Congreso Nacional De Investigación En Ciencia E Innovación De Tecnologías Productivas, Cd. Altamirano, Estado De Guerrero, México. Noviembre, 2023. (https://www.cdaltamirano.tecnm.mx/index.php/17-vi-congreso-nacional-de-investigacion-en-ciencia-e-innovacion-de-tecnologias-productivas/140-tecnm-40, 23/03/2024).

[10] E. Rangel, K.U. Rangel, and L. González, "Cifrado De Datos Dinámico Con Inteligencia Artificial, Utilizando El Nuevo Formato Pseudo-Hexadecimal," Revista Electrónica de Divulgación de la Investigación del SABES [Revista de la Universidad del SABES, Editada por Sistema Avanzado de Bachillerato y Educación Superior en el Estado de Guanajuato. México, ISSN: 2007-3542], vol. 28, pp. 46-73, edición diciembre 2024. (https://sabes.edu.mx/revista-electronica/27/docs/4_Cifrado%20de_Datos_Dinamico.pdf , 17/12/2024).

[11] E. Rangel and K.U. Rangel, "La Regla Del Vecino Más Cercano Como Alternativa Para Inyectar Ruido A Mensajes Encriptados Por El Algoritmo: Noised Random Hexadecimal," INTELETICA, Revista de Inteligencia Artificial, Ética y Sociedad [Editado por IBERAMIA, Iberoamerican Society of Artificial Intelligence – Sociedad Iberoamericana de Inteligencia Artificial, España, ISSN: 3020-7444], vol. 1, num. 2, Dec. 2024, pp. 1–15, doi: https://doi.org/10.4114/INTELETICA.vol1iss2pp1-15, (https://inteletica.iberamia.org/index.php/journal/article/view/16, 16/12/2024).

[12] D. Álvarez, "Algunos Aspectos Jurídicos Del Cifrado De Comunicaciones," Derecho PUCP [Pontificia Universidad Católica del Perú], núm. 83, 2019, pp. 241-264, 2019. DOI: https://doi.org/10.18800/derechopucp.201902.008, (http://www.redalyc.org/articulo.oa?id=533662765008, 09/11/2024).

[13] F. Barranco and C. Galindo, "Criptografía básica y algunas aplicaciones," Universidad Jaume I, Departamento de Matemáticas, Castellón, España, octubre, 2022. URI: http://hdl.handle.net/10234/ 201359, (https://repositori.uji.es/items/35da2f29-ee4a-4dbc-a82f-c450a81cf9be,13/04/2025).

[14] S. Gómez, J.D. Arias, and D. Agudelo, "Cripto-Análisis Sobre Métodos Clásicos De Cifrado," Scientia Et Technica, [Universidad Tecnológica de Pereira Pereira, Colombia, ISSN: 0122-1701, Año: XVII], vol. 2, no. 50, pp. 97-102, abril 2012, DOI: https://doi.org/10.22517/23447214.6681, (https://revistas.utp.edu.co/index.php/revistaciencia/article/view/6681, 16/04/2025).

[15] B. Javidi and J.L. Horner, "Optical Pattern Recognition for Validation and Security Verification," Optical Engineering [ISSN: 0091-3286], vol. 33, issue: 6, pp. 1752-1756, june, 1994, DOI: https://doi.org/10.1117/12.170736,(https://www.spiedigitallibrary.org/journals/optical-engineering/volume-33/issue-6/0000/ Optical-pattern-recognition-for-validation-and-security-verification /10.1117/12.170736.short, 14/ 04/2025).

[16] B. Reddaiah, "A Study on Genetic Algorithms for Cryptography," International Journal of Computer Applications, [Department of Computer Applications, Yogi Vemana University Kadapa, A.P, India, ISSN: 0975-8887], vol. 177, no. 28, pp. 1-4, december, 2019. DOI: http://dx.doi.org/10.5120/ijca2019919509, (https://www.researchgate.net/publication/338012809_A_Study_on_Genetic_Algorithms_for_Cryptography, 23/03/2024).

[17] C. Sebas, "¿Qué son los Algoritmos Genéticos en las Inteligencias Artificiales?" Manuales y Tutoriales de Informatica, 2023. (https://aprendeinformaticas.com/ia/, 23/03/2024).

[18] S. Paul, P. Dasgupta, P. K. Naskar, and A. Chaudhuri, "Secured image encryption scheme based on DNA encoding and chaotic map," Review Of Computer Engineering Studies, [IIETA: International Information and Engineering Technology Association, ISSN: 2369-0755, e-ISSN: 2369-0763], vol. 4, no. 2, pp. 70-75, june issue, 2017. DOI: 10.18280/rces.040206, (https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://iieta.org/sites/default/files/Journals/RCES/04.02_06.pdf&ved=2ahUKEwjVlYfordOMAxVVLUQIHXknBWsQFnoECB4QAQ&usg=AOvVaw3yvem4PsuKbMM9009owuAJ, 12/04/2015).

[19] R. Oppliger, "Contemporary cryptography," 1ra. ed., Artech House Computer Security Library, [Boston/London, ISBN: 1-58053-642-5, 978-8189265038], 510p, 2005. (https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://theswissbay.ch/pdf/Gentoomen%2520Library/Cryptography/Contemporary%2520Cryptography%2520-%2520Rolf%2520Oppliger.pdf&ved= 2ahUKEwiT1tm1xNWMAxUAmO4BHWYLNa0QFnoECBoQAQ&usg=AOvVaw3GvxLI6QzJhvEXqcl9efPz, 13/04/2025).

[20] D.R. Stinson and M.B. Paterson, "Cryptography: Theory and Practice," (4ta ed.). Chapman and Hall Book/CRC Press [Taylor & Francis Group, ISBN: 978-1-1381-9701-5], 2019. (https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://www.ic.unicamp.br/~rdahab/cursos/mo421-mc889/Welcome_files/Stinson-Paterson_CryptographyTheoryAnd Practice-CRC %2520Press %2520 %25282019 %2529.pdf&ved=2ah UKEwi8_euUtdWMAxWzh-4BHShGAaQQFnoECCUQAQ&usg=AOv Vaw1DCBGkDbaZMeHuyR45xZZC, 13/04/2025).

[21] H.C.A. Van-Tilborg, "Encyclopedia Of Cryptography And Security," TUE Research portal. Springer, pp. 114-115, 201-202, 2005. DOI: https://doi.org/10.1007/0-387-23483-7, (09/11/2024).

[22] L. Baklaga, "Leading The Way In Quantum-Resistant Cryptography For Everyday Safety," SPCSJ, Scientific and Practical Cyber Security Journal [Scientific Cyber Security Association (SCSA).Tbilisi, Georgia, ISSN: 2587-4667], vol. 8, no. 3, pp 65-73, issue 2024. (https://journal.scsa.ge/papers/leading-the-way-in-quantum-resistant-cryptography-for-everyday-safety/, 23/03/2025).

[23] R. Bavdekar, C. Eashan-Jayant, A. Ankit, and K. Tiwari, "Post Quantum Cryptography: A Review of Techniques, Challenges, and Standardizations," In 2023 International Conference on Information Networking (ICOIN), 2023.

[24] Q.H. Dang and H.Q. Le, "Improved cryptanalysis of the RSA algorithm using side-channel attacks," Journal of Information Security and Applications, vol. 65, no. 103313, 2022.

[25] D. Luciano and G. Prichett, "Cryptology: From Caesar Ciphers To Public-key Cryptosystems," The College Mathematics Journal, vol. 18, pp. 2-17, 1987. (http://www.jstor.org/stable/2686311, 06/06/2025).

[26] M.S. Rahman and M.S. Hossain, "A Secure Private Key Cryptography Scheme Using RSA and AES," Journal of Cybersecurity, 1(1), 2021, pp. 1-9.

[27] J. Rodríguez, "Operadores Genéticos Aplicados A La Criptografía Simétrica," [Proyecto De Grado], Universidad Distrital Francisco José De Caldas. Facultad De Ingeniería. Ingeniería De Sistemas. Bogotá, Colombia, 2020. (https://repository.udistrital.edu.co/ handle/11349/28192, 06/05/2024).

[28] M. Baker and J. Schiller, "ECIES: Elliptic Curve Integrated Encryption Scheme," In Cryptography and Network Security, Springer, pp. 245-263, 2015. (https://medium.com/asecuritysite-when-bob-met-alice/elliptic-curve-integrated-encryption-scheme-ecies-encrypting-using-elliptic-curves-dc8d0b87eaa, 23/03/2025).

[29] D. Hankerson, J.L. Hernandez, and A.J. Menezes, "Software implementation of elliptic curve cryptography over binary fields," Springer-Verlag Berlin Heidelberg 2000, [K. Ko¸c and C. Paar (Eds.) – CHES 2000, LNCS 1965], pp. 1–24, 2004. (https://idp.springer.com/authorize?response_type=cookie&client_id=springerlink&redirect_uri=https%3A%2F%2Flink.springer.com%2Fcontent%2Fpdf%2F10.1007%2F3-540-44499-8_1.pdf, 23/03/2025).

[30] P.L. Montgomery, "Speeding up the Pollard rho method," Mathematics of Computation, vol. 48, no. 177, 1987, pp. 453-456.

[31] NIST, "Special Publication 800-56A Revision 2: Recommended methods for key establishment using public key cryptography," NIST Special Publication 800-56A, 2013. (https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-56ar2.pdf&ved=2ahUKEwio1eCz3KGMAxWgJ0QIHY c4BXQQFnoECBsQAQ&usg=AOvVaw367-qADImRilvhabe1UtQr), (https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://csrc.nist.gov/pubs/sp/800/56/a/r2/final&ved= 2ahUKEwio1eCz3KGMAxWgJ0QIHYc4BXQQFnoECBwQAQ&usg= AOvVaw183JO5CLTxYuENINrMwqgL, 23/03/2025).

[32] H.W. Dhany, F. Izhari, H. Fahmi, M. Tulus, and M. Sutarman, "Encryption and Decryption using Password Based Encryption, MD5, and DES," Published by Atlantis Press, [ISSN: 2352-5398. Open Access: CC BY-NC], 2018, license: http://creativecommons.org/licenses/by-nc/4.0/.

[33] A. Kumar and S. Sharma, "A Study on Historical Cryptographic Techniques: Caesar Cipher to DES," International Journal of Advanced Science and Technology, 30(2), 2021, pp. 555-564.

[34] H.C. Van and S. Jajodia, "Encyclopedia Of Cryptography And Security," Springer Science & Business Media, [ISBN: 978-14419-5907-2], 1416p, 2011.

[35] B. Schneier, "Description of a new variable-length key, 64-bit block cipher (Blowfish)," In Fast Software Encryption, 1994.

[36] ——, "Secrets and lies: Digital security in a networked world," Wiley, 2000.

[37] E.A. AL-Maqtari and E.A. AL-Maqtari, "Performance Evaluation for AES, Blowfish, DES, and 3DES Cryptography Algorithms," PUIRP: Partners Universal Innovative Research Publication (ISSN: 3048-586X, Tamilnadu, India), vol. 2, no. 5, pp. 86-95, october, 2024, doi: https://doi.org/10.5281/zenodo.13974870. Available at: (https://www.puirp.com/index.php/research/article/view/81, 27/04/2025).

[38] R. K. Muhammed, R.R. Aziz, A.A. Hassan, A.M. Aladdin, S.J. Saydah, T. A. Rashid, and B.A. Hassan, "Comparative Analysis of AES, Blowfish, Twofish, Salsa20, and ChaCha20 for Image Encryption," Kurdistan Journal of Applied Research (Published by Sulaimani Polytechnic University, ISSN: 2411-7684), vol. 9, no. 1, pp. 52–65, may issue, 2024, doi: https://doi.org/10.24017/science.2024.1.5, URL: https://doi.org/10.48550/arXiv.2407.16274. Available at: (https://arxiv.org/abs/2407.16274, 01/06/2025).

[39] H. K. Garai and S. Dey, "A multi-step key recovery attack on reduced round Salsa and ChaCha," Cryptologia [Taylor & Francis Group Eds., ISSN: 1558-1586], vol. 49, no. 3, pp. 252–267, june issue 3, 2024, doi: https://doi.org/10.1080/01611194.2024.2342918. (https://jackgiffin.com/main/pdfs/A-multi-step-key-recovery-attack-on-reduced-round-Salsa-and-ChaCha-Hirendra-Garai-and-Sabyasachi-Dey.pdf, 10/06/2024).

[40] A. Saini, A. Tsokanos, and R. Kirner, "CryptoQNRG: a new framework for evaluation of cryptographic strength in quantum and pseudorandom number generation for key-scheduling algorithms," The Journal of Supercomputing (e-ISSN: 1573-0484), vol. 79, pp. 12219–12237, july, 2023, doi: https://doi.org/10.1007/s11227-023-05115-4. Available at: (https://link.springer.com/article/10.1007/s11227-023-05115-4, 14/04/2025).

[41] J. Daemen and V. Rijmen, "The Design of Rijndael: AES - The Advanced Encryption Standard," Springer, 2002. DOI:10.1007/978-3-662-04722-4, (23/03/2025).

[42] M. Iavich, T. Kuchukhidze, and A. Gagnidze, "Post-quantum Digital Signature Using Verkle Trees And Lattices," SPCSJ, Scientific and Practical Cyber Security Journal [SCSA, Scientific Cyber Security Association, Tbilisi, Georgia, ISSN: 2587-4667], vol. 8, no. 3, pp. 35-52, issue 2024. (https://journal.scsa.ge/issues-archive/, 23/03/2025).

[43] P. Fuegner, "Are RSA and AES Both at Risk From the Quantum Threat?" QuSecure, Inc., 2024. (https://www.qusecure.com/are-rsa-and-aes-both-at-risk-from-the-quantum-threat/#:~:text=The%20emergence%20of%20quantum%20computers,efficiently%20factoring%20large%20prime%20numbers, 2025-02-08).

[44] S. Sengupta and S. Ghosh, "Quantum Computing Encryption Threats: Why RSA and AES Are at Risk," Journal of Cryptographic Research, 2023.

[45] J. Thakur and N. Kumar, "DES, AES and Blowfish: Symmetric Key Cryptography Algorithms Simulation Based Performance Analysis," International Journal of Emerging Technology and Advanced Engineering, 2011, pp. 6-12.

[46] E. Rangel, "Vecinos Envolventes para Variantes de la Regla del Vecino más Cercano," [MSc.Thesis], Instituto Tecnológico de Toluca, Metepec, México, 2002.

[47] ——, "La Regla De Los k Vecinos Más Cercanos (k-NN) Basada En Distancia De Manhattan (City-Block) Para Mejorar La Clasificación De Patrones," In Quinto (V) Congreso Nacional De Investigación En Ciencia E Innovación De Tecnologías Productivas. Tecnológico Nacional De México, campus: Instituto Tecnológico de Cd. Altamirano, Estado De Guerrero, México, noviembre, 2022. (http://erangel.coolpage.biz/pappers/edgarrangel2022.pdf, 06/06/2025).

[48] S. S. Iyengar, R. Kannan, and S. Ganapathi, "Inteligencia artificial y criptografía: Tendencias y desafíos," IEEE Transactions on Emerging Topics in Computing, vol. 9, no. 2, pp. 833-844, 2021.

[49] X. Liu and X. Wang, "Quantum cryptanalysis of lattice-based cryptographic protocols," Physical Review X, vol. 12, no. 2 (021004), 2022.

[50] A.K. Singh, P. Kumar, and R. Singh, "Aplicación de la inteligencia artificial en la criptografía: Una revisión," Journal of Intelligent Information Systems, vol. 67, no. 2, 2021, pp. 257-275.

[51] T.M. Mitchell, "Machine learning," (2ª ed.), McGraw-Hill, 2020.

[52] J. Ross-Quinlan, "C4.5: Programs for Machine Learning," Morgan Kaufmann, San Mateo, CA., 1993.

[53] S.J. Russell and P. Norvig, "Inteligencia artificial: Un enfoque moderno," (4ª ed.), Pearson, 2020.

[54] A.K. Hartmann, "Heuristic search in graphs," Journal of Artificial Intelligence Research, vol. 68, 2020, pp. 1-33.

[55] J.S. Sánchez, F. Pla, and F. J. Ferri, "Prototype selection for the nearest neighbor rule through proximity graphs," Pattern Recognition Letters 18, 1997, pp. 507-513.

[56] L.I. Kuncheva and L.C. Jain, "Nearest Neighbor Classifier: Simultaneous editing and feature selection," Pattern Recognition Letters, vol. 20, pp. 1149-1156, 1999.

[57] K.P. Murphy, "Probabilistic machine learning: An introduction," MIT Press, 2022.

[58] B. Reddaiah, "A Study on Pairing Functions for Cryptography," IJCA (0975-8887), vol. 149, no. 10, pp. 4-7, september, 2016.

[59] D.B. Skalak, "Prototype and Feature Selection by Sampling and Random Mutation Hill Climbing Algorithms," In ML94, Proceedings of the Eleventh International Conference on Machine Learning, [Morgan Kaufmann], pp. 293-301, 1994.

[60] A. Clark, "Modern optimisation algorithms for cryptanalysis," In Proceedings of the 1994 Second Australian and New Zealand Conference on Intelligent Information Systems, November 29 - December 2, 1994, pp. 258-262.

[61] W. Griindlingh and J.H. Van-Vuuren, "Using Genetic Algorithms to Break a Simple Cryptographic Cipher," (submitted 2002), Retrieved March 31, 2003, unpublished. (http://dip.sun.ac.za/~vuuren/abstracts/abstr_genetic.htm).

[62] R.A.J Matthews, "The use of genetic algorithms in cryptanalysis," Cryptologia, vol. 17, no. 4, pp. 187-201, 1993.

[63] L. Bruzzone and S.B. Serpico, "Classification of Imbalanced remote-sensing data by neural networks," Elsevier Science B.V. , [0167-8655, 97, PH S0167-8655 (97) 00109-8], 1997. DOI: https://doi.org/10.1016/S0167-8655(97)00109-8. (https://www. sciencedirect.com/science/article/abs/pii/S0167865597001098, 01/06/2025).

[64] I. Goodfellow,.Y. Bengio, and A. Courville, "Deep learning," MIT Press, 2021.

[65] R. Barandela, J.S. Sánchez, V. García, and E. Rangel, "Strategies for Learning in Class Imbalance Problems," Pattern Recognition [Rapid and Brief Comunication, Pergamon, ISBN: PII: S0031-3203(02)00257-1.0031-3203/02/], vol. 36, no. 3, pp. 849-851, 2003. DOI: https://doi.org/10.1016/S0031-3203(02)00257-1, (01/06/2025).

[66] D. Lewis and J. Catlett, "Heterogeneous Uncertainty Sampling for Supervised Learning," Proceedings of the 11th International Conference on Machine Learning, ICML'94, [New Brunswick, New Jersey, Morgan Kaufmann], pp. 148-156, 1994.

[67] T.M. Cover and P.E. Hart, "Nearest Neighbor Pattern Classification," IEEE Transactions on Information Theory, [e-ISSN: 1557-9654], vol. IT-13, january 1967, pp. 21-27. DOI: 10.1109/TIT.1967.1053964. (https://ieeexplore.ieee.org/abstract/document/ 1053964/, 23/03/2024).

[68] E. Rangel and K.U. Rangel, "Novel Pseudo-Hexadecimal Encryption Strategies For Camouflaging Ciphertext Based On Nearest Neighbor With Artificial Intelligence," IJCOPI, International Journal of Combinatorial Optimization Problems and Informatics, ISSN: 2007-1558, manuscript in review since 2024. https://ijcopi.org/ojs/authorDashboard/submission/529, unpublished.

[69] Microsoft, "Descarga de software," Microsoft, 2025. (https://www.microsoft.com/es-mx/software-download, 01/06/2015).

[70] Python.org, "The Python Network," Python.org, 2024. (https://www.python.org/downloads/, 18/11/2024).

[71] Google and Android 12, "Sistema operativo para dispositivos móviles," Google, 2024. (https://www.android.com/intl/es_es/android-12/, 01/06/2025).

[72] Pydroid3 versión 7.4_arm64, "IDE for Python 3. Lenguaje de programación y compilador," Google Play Store, 2025. (https://play.google.com/store/apps/ details?id=ru.iiec.pydroid3&hl=en&pli=1, 01/06/2025).

[73] Python and Cryptography, "Cryptography 45.0.4," Python Software Foundation, 2025. (https://pypi.org/project/cryptography/, 01/06/2025).

[74] PyCryptodome, "Crypto.Cipher package. Introduction," Readthedocs.io, 2025. (https://pycryptodome.readthedocs.io/en/ latest/src/cipher/cipher.html, 30/03/2025).

[75] PyPI, "Pycryptodome 3.21.0," Python Software Foundation, 2024. (https://pypi.org/project/pycryptodome/, 13/12/2024).

[76] E. Rangel, K. U. Rangel, L. González, C. A. Bernal, C. M. Rangel, R. I. Reynoso, C. A. Rodríguez, and L..J. Ascencio, "ChaCha20 Encryption Algorithm Security Enhancement through Artificial Intelligence-Based Random Noisy Injection: A Case Study," CULCYT. Cultura Científica y Tecnológica. Revista de investigación en ingeniería e innovación tecnológica, vol. 22, no. 3, pp. 14-33, Sept.-Dec. 2025, doi: 10.20983/culcyt.2025.3.2.2, (https://erevistas.uacj.mx/ojs/index.php/culcyt/article/view/6916/8918, 31/12/2025).

[77] E. Rangel, C. M. Rangel, and K. U. Rangel, "Camuflaje de Textos Cifrados Utilizando Inteligencia Artificial para Mejorar la Seguridad del Algoritmo AES-256, Convirtiéndolo en un Esquema de Encriptado

Dinámico," Rev. Electron. Divulg. Investig., vol. 30, pp. 39-75, Dec. 2025, (https://sabes.edu.mx/revista-electronica/29/#, 19/12/2025).

[78] E. Rangel, K. U. Rangel, L. González, C. A. Bernal, L. J. Ascencio, R. I. Reynoso, and C. A. Rodríguez, "Novel Dynamic CAST-128 Encryption Scheme And Comparison With Five Random Noisy Injection Strategies With Artificial Intelligence," Sci. Pract. Cyber Secur. J., vol. 9, no. 4, pp. 32-49, Sep. 2025, (https://journal.scsa.ge/papers/novel-dynamic-cast-128-encryption-scheme-and-comparison-with-five-random-noisy-injection-strategies-with-artificial-intelligence/, 13/10/2025).

[79] E. Rangel, K. U. Rangel, C. M. Rangel, C. A. Bernal, C. A. Rodríguez, and L. González, "A Novel IDEA-128 Enhancement Using Noisy Injection And Artificial Intelligence For Increasing Ciphertext Security," Sci. Pract. Cyber Secur. J., vol. 10, no. 1, pp. 1-20, Mar. 2026, (https://journal.scsa.ge/papers/a-novel-idea-128-enhancement-using-noisy-injection-and-artificial-intelligence-for-increasing-ciphertext-security/, 03/04/2026).

[80] E. Rangel, C. M. Rangel, K. U. Rangel, L. Roque, A. Salgado, C. A. Rodríguez, L. González, and M. C. Ramos, "An Analysis Of The Asymmetric Random Noisy RSA Encryption Scheme," Sci. Pract. Cyber Secur. J., vol. 10, no. 1, pp. 121-140, Mar. 2026, (https://journal.scsa.ge/papers/an-analysis-of-the-asymmetric-random-noisy-rsa-encryption-scheme/, 03/04/2026).

[81] E. Rangel, C.M. Rangel, K.U. Rangel, L. Roque, C.A. Rodríguez, A. Salgado, and M.C. Ramos, "Improving CAST-128 Performance for Efficient Dynamic Data Encryption. A Comparative Evaluation of CAST-128 and AES-256 Ciphertext Camouflaging Schemes," Manuscript in review since 2026. CULCYT. Cultura Científica y Tecnológica. Revista de investigación en ingeniería e innovación tecnológica, ISSN: 2007-0411, vol. 23, num. 1, Enero-Abril 2026, doi: https://doi.org/10.20983/culcyt.2026.1, (https://erevistas.uacj.mx/ojs/index.php/culcyt/issue/view/897, 06-03-2026), unpublished.