Camuflaje de Textos Cifrados Utilizando Inteligencia Artificial para Mejorar la Seguridad del Algoritmo AES-256, Convirtiéndolo en un Esquema de Encriptado Dinámico.

[1]Edgar Rangel Lugo , [1]Cinthya Maybeth Rangel Ríos, [1]Kevin Uriel Rangel Ríos

[1]Tecnológico Nacional de México. I. T. Ciudad Altamirano. Depto. Sistemas y Computación
erangel_lugo@hotmail.com

Recibido*: 26 de noviembre de 2025*
Aceptado*: 19 de diciembre de 2025*

RESUMEN

El *robo digital de datos* es un problema que puede ocasionar importantes pérdidas financieras en las organizaciones. Una de las causas puede ser el uso de estrategias de *ciberseguridad* inadecuadas. Por ejemplo, el uso de *algoritmos de encriptación* obsoletos o muy conocidos por las herramientas de los *ciberdelincuentes*. La llegada de la *computación cuántica* también supone un riesgo para la seguridad de los algoritmos de cifrado estándar. Para amortiguar este problema, suelen utilizarse métodos de *cifrado dinámico*, porque producen diferentes resultados encriptados. La importancia de esta investigación se centra en la fusión de *camuflaje de textos* utilizando el método: *reduced random mutation*, siendo aplicado a *textos cifrados* por el algoritmo *AES-256 (Advanced Encryption Standard 256)*, debido a que ello ha sido poco estudiado. El *objetivo* es evaluar la efectividad de esta combinación, denominada aquí como: *noisy mutation AES-256*, y compararla con otras estrategias de *cifrado dinámico*, específicamente, las estrategias: *random noisy* y *reduced noisy*. El alcance de este estudio no implica el uso de *computación cuántica*, ya que se experimenta un caso particular de *AES-256* con *camuflaje de textos cifrados* para convertirlo en un esquema *dinámico*. Se introduce una nueva propuesta para el *camuflaje de textos* basada en *procesos de mutación* con inteligencia artificial *(IA)* y se comparan resultados con estrategias basadas en *inyección de ruido*. El propósito es evitar que los *textos cifrados* sean descifrados con facilidad por los *ciber-criminales,* y así, poder demorar posibles ataques futuros basados en *computación cuántica*.

Palabras claves: *Cifrado dinámico, inteligencia artificial, camuflaje de textos basado en mutación, criptografía*.

ABSTRACT

The *theft of digital data* is a problem that can produce significant financial losses in organizations. One of the causes may be the use of inadequate *cyber-security* strategies. For example, the employment of *encryption algorithms* that are well-known to *cybercriminals*. The advent of *quantum computing* also poses a risk to the security of *standard encryption algorithms*. *Dynamic encryption methods* are often used for handling this problem because they produce different *ciphertext* results. This research focuses on the fusion of *camouflaging strategy* using the *reduced random mutation* method, applied to *ciphertext* by the *AES-256* algorithm *(Advanced Encryption Standard 256),* because it has been rarely studied. The *aim* of this research is to evaluate the effectiveness of this combination, referred to as *noisy mutation AES-256*, and compare it with other *dynamic encryption* strategies, specifically, the strategies: *random noisy* and *reduced noisy*. The *scope* of this study does not involve the use of *quantum computing*, but rather explores a particular case of *AES-256* with a *camouflaging strategy* to make it a *dynamic* scheme. A novel proposal for *camouflaging ciphertext* based on *mutation* processes with artificial intelligence *(AI)* is also introduced. Besides, experimental results are compared with other strategies based on *noisy injection* schemes. The purpose is to prevent *ciphertext* from being easily decrypted by *cyber-criminals* and to delay possible future attacks based on *quantum computing*.

Keywords: *Dynamical encryption methods, artificial intelligence, camouflaging ciphertext, cryptography.*

1. INTRODUCCIÓN

Un problema que está recibiendo gran atención, debido a que puede ocasionar importantes pérdidas financieras en las organizaciones, es precisamente, el *robo digital de datos* (Rangel & Rangel, 2024a; Rangel et al., 2025d). De acuerdo con Rangel & Rangel (2025a), se ha observado que esta situación puede ocurrir en dominios prácticos, siendo ocasionada por el uso de *estrategias de ciberseguridad* inadecuadas (Rangel et al., 2023, 2025b, 2025d), o al menos, porque alguno de los métodos empleados resulta vulnerable a *ciberataques* (Rangel et al., 2024, 2025c). La *ciberseguridad* es un campo de las ciencias computacionales e informática, que estudia aspectos concernientes con la protección de los datos, su confidencialidad, integridad y disponibilidad, así como, de los sistemas informáticos, redes de computadoras, prevención contra ataques y accesos no autorizados. En este ámbito, existen diversas estrategias destinadas a proteger sistemas computacionales, redes y datos digitales. Entre ellas se encuentran la evaluación de riesgos, políticas de seguridad, herramientas de detección de ciberataques y prevención de intrusiones, segmentación de redes, planes de respuesta a incidentes, software de protección de datos (como cortafuegos, antivirus y antimalware), medidas de autenticación, pruebas de penetración (pentesting), actualizaciones periódicas de sistemas, respaldos de información y métodos de *cifrado de datos*, por mencionar algunas (Rangel et al., 2025b, 2025d). Por consiguiente, el *cibercrimen* es considerado una actividad que involucra el uso de *internet* o *ciberespacio* como medio para llevar a cabo conductas deshonestas, previstas para obtener ganancias financieras u otras formas de deshonestidad (Drzani, 2014; Mamman et al., 2024). En este mismo contexto, en dependencias dónde se ha optado por el uso de *cifrado de datos* (Rangel et al., 2025b; Rangel et al., en revisión 2025d), como estrategia de *ciberseguridad*, se considera vulnerable la seguridad de la información, cuando los métodos o *algoritmos de encriptación* (Rangel & Rangel, 2024a, 2025a; Rangel et al., en revisión 2025a) que están utilizando, han quedado obsoletos o son muy conocidos por las herramientas que manejan los *ciber-delincuentes* (Rangel et al., 2023, 2025a). Se entiende por *encriptado* o *cifrado de datos*, a la ocultación de la información, mediante la traducción de un mensaje original convirtiéndolo en un tipo de lenguaje o código, utilizando un *alfabeto* para *cifrado/descifrado*, que solamente podrá ser capaz de entender el *software especializado* o la *persona autorizada.* El proceso inverso, se conoce como *descifrado de datos*, el cual, consiste en convertir el *texto cifrado* a *texto claro* (Gómez et al., 2012; Liu & Wang, 2021; Rangel et al., 2023, 2024, en revisión 2025b).

De acuerdo con Rangel et al. (2025b), el área que estudia lo relacionado con *cifrado de datos*, permitiendo resolver algunos tipos de problemas en dicho contexto, se conoce como: *criptografía* (Barranco & Galindo, 2022; Delman, 2004; Granados, 2006; Kalsi et al., 2018; Mendoza, 2008). Dicha área, a pesar que se desprende de la *ciberseguridad*, en la actualidad, ya se considera un nuevo campo independiente. Lo anterior, está sustentado porque la *criptografía* proporciona métodos, técnicas, algoritmos, estrategias y herramientas que permiten realizar el *cifrado y descrifrado de datos*, siendo capaz de resolver una gran variedad de problemas, relacionados con el *robo digital de datos*, ayudando a salvaguardar la información de las organizaciones, así como, de usuarios particulares, que manejan grandes volúmenes de datos (Oppliger, 2005; Rangel et al., 2024, 2025b). Empero, para lograr el *cifrado de datos*, existen una gran variedad de alternativas (Barranco & Galindo, 2022; Oppliger, 2005; Stinson & Paterson, 2019; Van & Jajodia, 2011; Van-Tilborg, 2005), las cuales, siguen diferentes caminos en su desarrollo (Rangel et al., 2023, 2024, 2025b). Por ejemplo, existen propuestas basadas en *alfabetos* que permiten el *encriptado de datos* usando *cifrados de flujo*, *cifrado por bloques*, *por permutación*, *por desplazamiento* y/o *sustitución* de carácteres (Barranco & Galindo, 2022; Rangel et al., 2024, 2025a, 2025b; Van & Jajodia, 2011; Van-Tilborg, 2005); así como, algoritmos basados en *exponenciación modular, aritmética de curvas elípticas* y *logaritmo discreto* (Baker & Schiller, 2015; Barranco & Galindo, 2022; Hankerson et al., 2004; Montgomery, 1987; NIST, 2013; Rivest et al., 1978; Rodríguez 2020; Susilo et al., 2021; Van & Jajodia, 2011; Van-Tilborg, 2005), por mencionar algunas metodologías. En este mismo contexto, Rangel et al. (2025b) también señala que existen otras alternativas, como la *encriptación de datos* mediante el uso de la *tecnología óptica* (Goodman, 1968; Javidi & Horner, 1994; Linfei & Daomu, 2005; Rueda & Lasprilla, 2002; Rueda et al., 2005; Wang & Chen, 2022), así como, el uso de *cripto-sistemas* (Barranco & Galindo, 2022; Centellas et al., 2022; Clark, 1994; Gómez & Díaz, 2021; Gómez et al., 2012; Luciano & Prichett, 1987; Mendoza, 2008; Oppliger, 2005; Rivest et al., 1978;

Stinson & Paterson, 2019), basados en *métodos* o *algoritmos simétricos* y *asimétricos*, que comúnmente son empleados en el *encriptado de archivos* o *cifrado de texto plano*, mientras otros tipos de *cripto-sistemas* basados en *sistemas discretos caóticos* (Chong et al., 2011; Jiménez et al., 2015; Pisarchik & Flores-Carmona, 2006; Pisarchik & Zanin, 2008; Rajan & Saumitr, 2006), están enfocados en el *encriptado de imagen plana* o en el *cifrado de datos* para la *transmisión en tiempo real*, entre otras aplicaciones. Además, existen otras metodologías aplicables al cifrado de datos, que emplean *modelos criptográficos* basados en *inteligencia artificial* (Baklaga, 2024; Griindlingh & Van-Vuuren, 2002; Iyengar et al., 2021a, 2021b; Kalsi et al., 2018; Rangel, 2002; Rangel, 2022; Rangel & Rangel, 2024a, 2024b, 2025a, 2025b; Rangel et al., 2023, 2024, 2025a; Reddaiah, 2019). Algunos de estos modelos, hacen uso de *algoritmos genéticos* o *GAs* (Clark, 1994; Delman, 2004; Griindlingh & Van-Vuuren, 2002; Kalsi et al., 2018; Kuncheva & Jain, 1999; Matthews, 1993; Rangel & Rangel, 2024a, 2024b; Rangel et al., 2023, 2024, 2025a, 2025d; Reddaiah, 2019; Rodríguez, 2020; Skalak,1994), los cuales, han destacado, incluso en el área del *cripto-análisis* (Barker & Roginsky, 2020; Dang & Le, 2022; Gómez et al., 2012; Katz.& Lindell, 2019; Liu & Wang, 2022; Nagaraj & Srinadth, 2015; Susilo et al., 2021).

Uno de los algoritmos de *cifrado de datos,* que fue muy popular en sus inicios, por ser sencillo de implementar (Rangel et al., 2023, 2024, 2025a), es el *cifrado del César* (Barranco & Galindo, 2022; Gómez et al., 2012; Luciano & Prichett, 1987; Rangel et al., 2023, 2024), también conocido como: *algoritmo de César* o *algoritmo Caesar*. Se trata de un *algoritmo de desplazamiento* o *sustitución* (Rangel & Rangel, 2024a, 2024b; Rangel et al., 2025b), según la forma de su implementación. Este tipo de procedimiento, ya ha sido estudiado en otras investigaciones (Barranco & Galindo, 2022; Rangel & Rangel, 2024a, 2024b). Debido a vulnerabilidades encontradas, este algoritmo ha sufrido modificaciones, dando lugar a la existencia de diversas variantes. Una de ellas, corresponde al caso del *cifrado Vigenère* (Gómez et al., 2012), mientras que otras modificaciones, utilizan *inteligencia* artificial *(IA),* para proporcionar resultados con *cifrado dinámico* (Rangel et al., 2023, 2024; Rangel & Rangel, 2024a), estrategias que se describen más adelante. Por consecuencia, surgen nuevas propuestas para el *cifrado de datos*, siendo mucho más seguras que *Caesar* y *Vigenère.* En la actualidad, según Rangel et al. (2025b), los casos más populares para cifrado de *texto plano* son los *métodos de cifrado estándar* que están basados en algoritmos: *simétricos* o *asimétricos* (Centellas et al., 2022; Gómez & Díaz, 2021; Mendoza, 2008; Oppliger, 2005; Stinson & Paterson, 2019; Van & Jajodia, 2011; Van-Tilborg, 2005). Algunos de estos algoritmos, mediante una pequeña adaptación, también han sido utilizados para *cifrado de archivos e imágenes*.

Los algoritmos de *encriptación de datos*, que utilizan para el *cifrado* y *descrifrado*, una misma *clave* o *llave secreta* como parte de la entrada, son considerados *algoritmos simétricos,* porque realizan el *cifrado de llave simétrica* (Rangel et al., 2025b). Algunos ejemplos de este tipo de esquemas, son las alternativas que trabajan *por bloques,* como es el caso de: *RC (Rivest Cipher), particularmente RC1, RC2, RC5 y RC6;* así como, los algoritmos *DES (Data Encryption Standard), 3DES (Triple Data Encryption Standard), AES-256 (Advanced Encryption Standard - 256 bits), Blowfish, Twofish, CAST-128 (por Carlisle Adams & Stafford Tavares - 128 bits), IDEA-128 (International Data Encryption Algorithm - 128 bits), Camellia, GOST (Gosudarstvennyy STandart)*, por mencionar algunos. Otros ejemplos de *algoritmos simétricos*, pero que trabajan por el cifrado *por flujo*, los más populares son: *RC4 (Rivest Cipher 4), WEP (Wired Equivalent Privacy), Salsa20, ChaCha20,* entre otros. Empero, si para realizar operaciones de *cifrado/descrifrado* se utilizan *dos claves* distintas: una *pública* y otra *privada*, además de la *llave secreta*, entonces son considerados algoritmos *asimétricos* (Centellas et al., 2022; Gómez & Díaz, 2021; Mendoza, 2008; Rangel et al., 2025b). Algunos algoritmos populares de esta categoría son: *ElGamal* (basado en la dificultad del problema del *logaritmo discreto*), el algoritmo *RSA (Rivest-Shamir-Adleman)* que utiliza la *exponenciación modular* y *ECC (Elliptic Curve Cryptography)* basado en *aritmética de curvas elípticas,* destacando la versión *RSA-2048* y *ECIES-SEC (Elliptic Curve Integrated Encryption Scheme of Standards for Efficient Cryptography) P-256-R1,* respectivamente. En la presente investigación, solamente se realizan experimentos con el algoritmo *simétrico AES,* usando el estándar *AES-256*, con el propósito de llevar a cabo un estudio respecto a la *inyección de ruido* y *camuflaje de textos cifrados*, lo cual, se describe más adelante.

De acuerdo con Rangel et al. (2025b), una descripción rápida sobre los *algoritmos simétricos* más populares, fue realizada por: Baklaga (2024), quien señala que, *AES* (Barranco & Galindo, 2022; Daemen & Rijmen, 2002; NIST, 2001; Rahman & Hossain, 2021; Rodríguez, 2020; Van-Tilborg, 2005), es un *algoritmo simétrico* de *cifrado por bloques* que opera con distintos tamaños y soporta diversas longitudes de *clave* (con *128, 192* y hasta *256* bits). En contraste, el algoritmo *DES* (Barranco & Galindo, 2022; Dhany et al., 2018; Kumar & Sharma, 2021; Mendoza, 2008; Rodríguez, 2020), procesa *texto plano* a *64 bits*, produciendo también, *64 bits* de *texto cifrado*. Esta operación involucra una serie de *rondas* basadas en *sustitución* y *permutación* que incluyen operaciones para *descifrados* en *inversa* o reversa. Sin embargo, estos *64 bits* son considerados insuficientes para ambientes seguros, porque el esquema de seguridad, lo hace relativamente fácil de romper. Como resultado, fue desarrollado el *algoritmo 3DES* (Mendoza, 2008; Van & Jajodia, 2011; Van-Tilborg, 2005), como una versión mejorada del *DES*. Otro *algoritmo simétrico* popular es conocido como: *Blowfish* (Schneier, 1994, 2000), el cual, utiliza *bloques* de cifrado de longitud variable y claves con rango de longitud entre *32* y *448 bits,* mientras que su sucesor: *Twofish* (AL-Maqtari & AL-Maqtari, 2024; Muhammed et al., 2024; Rangel & Rangel, en revisión 2025), también es basado en *bloques* de *longitud variable*, pero trabaja con *claves* de 128 hasta 256 bits. A pesar que, dicho esquema se utiliza comúnmente con 128 bits. Esta alternativa resalta porque presenta nuevas características en la seguridad de datos, así como, un mayor rendimiento, en comparación con su predecesor (Rangel & Rangel, en revisión 2025). En lo concerniente con *Camellia* (Aoki et al., 2000; Oppliger, 2005; Sami-Sulaiman & Hammood, 2025), según Rangel & Rangel (2025a), es un *algoritmo simétrico* que trabaja por bloques de 128 bits y *encripta datos* usando *claves* de 128, 192 y 256 bits en *llave secreta*, empleando un número variable de *rondas*, de acuerdo con la *longitud de la clave*, es decir, se realizan 18 *rondas* con *claves* de 128 bits, mientras que, para *claves* de 192 y 256 bits, se ejecutan 24 rondas. En el cifrado, combina operaciones matemáticas, basadas en *permutación* y *sustitución*, empleando operador *XOR* (Barranco & Galindo, 2022; Van-Tilborg, 2005). El propósito del diseño de este algoritmo, fue adoptar un esquema altamente eficiente y seguro. En cambio, el *algoritmo simétrico: CAST5* o *CAST-128* (Wang et al., 2016), trabaja con bloques de longitud fija. De acuerdo con Wang et al. (2016), esta alternativa presenta buena resistencia a varias formas de *criptoanálisis*, operando con *bloques* de 64 bits, utilizando *claves* de 40 hasta 128 bits, muy parecido al *criptosistema SPN* (Wang et al. 2016) del algoritmo *DES*. La versión *CAST-128* ofrece buen rendimiento para la seguridad de datos y según Rangel et al. (2025c), la alternativa *estándar CAST-128,* implementada en lenguaje *Python 3*, proporciona resultados de *cifrados dinámicos* (Rangel & Rangel, 2024a, 2024b), ya que, se observó que el *vector de inicialización (iv),* se generaba automáticamente de manera aleatoria, lo que proporcionaba resultados *cifrados* distintos en cada ejecución de *CAST-128*, aún utilizando la misma secuencia de *texto plano*, en comparación con otras propuestas de algoritmos *simétricos estándar*, cuyo resultado *cifrado* fue del tipo *estático* (es decir, siempre generaba la misma *secuencia cifrada* para un mismo *texto plano*). Otro *algoritmo simétrico* basado en bloques es conocido como: *IDEA-128* (Gundaram 2024; Rangel et al., en revisión 2025d; Van-Tilborg 2005). Este esquema es sucesor del estándar *IPES* (Gundaram 2024), por sus siglas en inglés: *Improved Proposed Encryption Standard*. Por lo tanto, *IDEA-128* trabaja con *bloques* de 64 bits usando 8.5 rondas, empleando *claves* de 128 bits. Ello proporciona alto nivel de seguridad, a pesar de hacer uso de *bloques* más pequeños que su *clave*. En cada *ronda* envuelve distintas *subclaves* derivadas de la *llave* maestra (principal), mediante una serie de operaciones que incluyen: *sustitución, permutación, adición, modularizacion*, y el uso del operador *XOR* (Van-Tilborg 2005). Para *desencriptado* se aplica el proceso en orden inverso (Gundaram 2024; Rangel et al., en revisión 2025d). Con respecto a *GOST* (Courtois, 2012; Dunkelman et al., 2023; Fulgueira et al., 2015; Ishchukova et al., 2020; Rangel et al., 2024, 2025b), existen varios estándares. Por ejemplo, el estándar *GOST R 34.11-94*, que opera *cifrado por bloques* con tamaño de *64 bits,* a través de un número invariable de *rondas* (comúnmente *32 rondas*), manejando *claves* de *256 bits*. Otra variante conocida como: *Streebog,* corresponde a la versión *GOST R 34.11-2012*, en este caso, no se involucra en el *cifrado de datos*, solamente se enfoca en tareas de *autenticación* e *integridad de datos*, ya que, trabaja con el uso de *funciones hash* (Alvarez, 2019; Courtois, 2012; Dhany et al., 2018; Fulgueira, et al., 2015; Gómez et al., 2012; Nagaraj & Srinadth, 2015; Oppliger, 2005; Stinson & Paterson, 2019), que puede utilizar desde *256* hasta *512 bits*. Sin embargo, el estándar *GOST 28147-89*, puede manejar tamaños de bloque con *64 bits* y *claves* de *256 bits,* mientras que, el estándar *GOST R 34.12-2015,* específicamente *GOST R 34.11-94 y R 34.12-2015*, han sido optimizadas para proporcionar mayor seguridad y eficiencia, mediante el uso de *claves* de *longitud variable.* Finalmente, el estándar *GOST R 34.12-2015*, tiene definidos al menos dos algoritmos para el *cifrado*

simétrico por bloques: *Magma* y *Kuznyechik* (Кузнечик). El primer caso, trabaja *por bloques* con tamaño de *64 bits* con *claves* de *256 bits*. El segundo caso, utiliza *bloques* con tamaño de *128 bits*, empleando *claves* de longitud de *256 bits* (Rangel et al., 2025b). De la misma forma que *GOST* dispone de varios estándares, según Rangel et al. (2025b), la familia de algoritmos: *RC (Rivest Cipher)*, también provee varias versiones (Mendoza 2008; Rambe, 2024; Van-Tilborg 2005). Entre las alternativas que corresponden al *cifrado de datos por bloques* solo se encuentran: *RC1, RC2, RC5* y *RC6*. Los esquemas: *RC1* y *RC2*, utilizan tamaño de *bloque fijo* de *64 bits*, mientras que, la versión: *RC5*, puede usar tamaños de *bloque variable*, empleando entre *32* y *128 bits*. Empero, *RC6* puede usar *bloques* con *tamaño fijo* de *128 bits*, y acepta *claves* con longitud de *128, 192* y *256 bits*. También, el referido *RC*, tiene otra variante, que es conocida como: *RC4* (Mendoza 2008; Mohammed et al., 2025; Van-Tilborg 2005), la cual, no trabaja *por bloques*, sino que, es considerado un *cifrador de flujo*, característica de algunos algoritmos, que requieren de poca memoria, cuando realizan la *encriptación de datos* en contextos de comunicaciones, con el propósito de proteger la confidencialidad de la información transmitida, principalmente, las que son realizadas en *tiempo real* (Rangel et al., 2025b). Este *algoritmo de flujo (RC4)*, según Rangel et al. (2025b), trabaja también con el uso de una *clave secreta*, pero con *longitud variable*, empleando un rango entre *40* y *2048 bits*. Esta *clave*, se utiliza para la inicialización (*pseudoaleatoria*) de un *vector* denominado: *IV* o *vector de estado (S),* el cual, contiene todos los posibles valores de un byte, dentro de un rango comprendido entre *0* y *255*. Lo anterior, es utilizado para el *cifrado*, haciendo uso de operaciones tipo *XOR* (Mendoza 2008; Mohammed et al., 2025; Van-Tilborg 2005), combinado con el *texto plano*. Otro *cifrador de flujo*, derivado del *RC4*, es el protocolo de seguridad: *WEP* (Anderson, 2008; Mohammed et al., 2025; Stubblefield et al., 2002), diseñado para proteger redes inalámbricas, el cual, hace uso de: *RC4*, como su principal *método de cifrado*. La alternativa: *WEP*, utiliza *claves* con longitud de *40* o *104* bits. Sin embargo, lo más común es llevar a cabo la implementación con *64 bits*, siendo empleados *40 bits* para el *cifrado* y *24 bits* para el *vector de inicialización*. Estas alternativas, están vinculadas con la seguridad en redes, presentando diferencias contundentes en el diseño y su aplicación. De acuerdo con Rangel et al. (2025b), las alternativas *WEP* y *RC4*, al igual que: *DES* y *3DES*, ya no son recomendadas, debido a que, se han detectado vulnerabilidades o han revelado problemas en la seguridad digital de datos. En la actualidad, se recomienda en la protección de redes inalámbricas, reemplazar el uso de: *WEP* por otras medidas basadas en: *WPA2* (Sandoval & Santamaría, 2024) y/o *WPA3* (Alghisi & Gringoli, 2024). Para el caso de *RC4*, se recomiendan otras variantes más seguras, por ejemplo, el uso de: *RC5* (Rambe, 2024) y/o *RC6* (Mohammed et al., 2025; Rambe, 2024), aclarando que estas dos últimas propuestas no son *cifradores por flujo*. Por último, existen otros *cifradores por flujo*, que en la actualidad, son muy populares porque se ha reportado tienen un buen margen de resistencia a *ataques criptográficos*, tal es el caso de los algoritmos: *Salsa20* (Garai & Dey, 2024; Rangel et al., en revisión 2025a) y *ChaCha20* (Garai & Dey, 2024; Rangel et al., en revisión 2025b). El primero, es considerado un esquema *criptográfico sencillo*, basado un modelo primitivo o rudimentario, por el uso de operaciones tipo *ARX* (Addition/Rotation/XOR), ya que, trabaja con *adiciones* usando *módulo 232* (⊞), *rotaciones* de bits por la izquierda *(n)* y uso de operaciones basadas en *XOR*, para la generación del *keystream* (Garai & Dey, 2024). Por consecuencia, es capaz de lograr gran velocidad y alto margen de seguridad. *Salsa20*, toma una clave de 256 bits *(k)*, usando una constante *(c)* y un *vector inicial (v)* o *IV*, cada uno de ellos con 128 bits, generando una salida (*keystream*) de 512 bits. La clave *(k0, k1, :::, k7)*, el *vector de inicialización: IV (v0, v1, t0, t1),* y la *constante (c0,c1,c2,c3),* son divididas en palabras de 32 bits, respectivamente. Debido a que se utilizan *20 rondas*, haciendo uso de claves con 256 bits, este algoritmo ha sido llamado: *Salsa20*. En contraste, su sucesor: *ChaCha20,* también basado en operaciones *ARX* (Garai & Dey, 2024), tal como lo hace *Salsa20*. Empero, *ChaCha20* toma una *clave* de 128 o 256 bits, según sea el caso, empleando una *constante* de 128 bits, agregando a la operación 128 bits para el uso del *nonce (number used once)*, el cual, debe ser un *valor único aleatorio* cada vez, para producir una salida *(keystream)* de 512 bits. La aportación de *ChaCha20* a este esquema criptográfico, además del uso de un *nonce* con 128 bits (porque *Salsa20*, lo trabaja con 64 bits), se introduce una ligera modificación al *estado interno* de la *matrix*, haciéndolo más rápido en implementación del software y más resistente a ciertos tipos de ataques, en comparación con su predecesor *Salsa20* (Garai & Dey, 2024).

Por otra parte, dentro de este mismo contexto, según Baklaga (2024), existen los *algoritmos asimétricos*, los cuales, son matemáticamente complejos. Uno de los más populares es conocido como: *RSA* (Barranco & Galindo, 2022; Dang & Le, 2022; Luciano & Prichett, 1987; Mendoza, 2008; Rahman & Hossain, 2021; Rivest et al., 1978;

Rodríguez 2020; Susilo et al., 2021). Se trata de un *cripto-sistema* basado en *clave pública*, la cual, es compartida públicamente, mientras que otra *clave privada* es guardada como secreto. Otro *algoritmo asimétrico* que ha emergido como una alternativa muy prometedora es conocido como: *ECC* (Baker & Schiller, 2015; Hankerson et al., 2004; Montgomery, 1987; NIST, 2013), particularmente en *aplicaciones criptográficas*, debido a su eficiencia en el tratamiento del *problema logarítmico* en campos finitos. Cabe mencionar que, los algoritmos: *RSA* y *ECC*, son muy similares. Sin embargo, difieren en que *ECC*, utiliza un *algoritmo criptográfico* distinto, el cual, provee una capacidad de solución más rápida (Baklaga, 2024; Rangel et al., 2025b).

De acuerdo con Rangel et al. (2025b), la mayoría de los *algoritmos simétricos* descritos previamente, presentan resultados basados en *cifrados estáticos* (excepto *CAST-128*). Sin embargo, los algoritmos *asimétricos*, como es el caso de: *RSA* y *ECC* (específicamente, las versiones: *RSA-2048* y *ECIES SECP-256-R1*), se ha revelado que tienen la capacidad de arrojar siempre resultados con *cifrados dinámicos*. Aunque dicha situación, no implica que esté garantizada la seguridad de los datos digitales, incluso estudios recientes (Fuegner, 2024; Rangel et al., 2025a, 2025d; Sengupta & Ghosh, 2023; Thakur & Kumar, 2011), advierten que no solamente los *algoritmos simétricos*, sino también, los de tipo *asimétricos*, pueden estar amenazados por la llegada de la *computación cuántica* (Baklaga, 2024; Iavich et al., 2024), señalando específicamente dos casos: *AES* y *RSA*, situación considerada como otro aspecto más, que puede influir en la vulnerabilidad de los algoritmos de encriptación estándar.

A pesar que, en la presente investigación no ha sido utilizada la *computación cuántica*, otras medidas han sido propuestas. Lo anterior, porque siguiendo la común práctica, de acuerdo con Rangel et al. (2025b), una alternativa para amortiguar un poco el problema del *robo digital de datos*, consiste en utilizar *métodos de cifrado dinámico*, debido a que, con dichas estrategias se puede confundir un poco a los *ciber-criminales,* ya que, estas alternativas son capaces de producir diferentes resultados *cifrados*, incluso cuando se utiliza la misma entrada de *texto plano,* empleando los mismos parámetros. Según Rangel et al. (2024, 2025b), algunas alternativas que permiten generar resultados *cifrados dinámicos* son: la *tecnología óptica,* los *sistemas discretos caóticos,* así como, la *criptografía* basada en *inteligencia artificial (IA)*, que de acuerdo con Rangel (2002, 2022), esta área de estudio pertenece al campo de las *Ciencias Computacionales*, donde uno de sus principales propósitos es hacer *emular* (o *simular*) que "*la máquina piense*". Un caso muy particular, concerniente con el *cifrado de datos* utilizando *IA*, es el uso de *algoritmos genéticos (GAs)*. A pesar que estos mecanismos, pueden utilizarse para el *cifrado de datos dinámico*, solamente en determinados contextos. En este respecto, existen varias alternativas. Sin embargo, ha sido poco estudiado el *cifrado de datos* basado en la *inyección de ruido* (Rangel et al., 2023, 2024, 2025d) y/o *camuflaje de textos* (Rangel & Rangel, 2024a; Rangel et al., 2025a). Un precedente sobre el impacto del ruido, como afecta en el tratamiento de la información, lo podemos encontrar en algunas investigaciones (Barandela et al., 2001, 2003a, 2003b, 2003c; Rangel 2002, 2022; Rangel & García, 2002), que han demostrado en el área de *inteligencia artificial (IA)* y *reconocimiento de patrones (RP),* como la presencia de *"ruido"* y/o *patrones atípicos* almacenados en una *muestra de entrenamiento (ME)*, pueden afectar de manera significativa la *precisión* (Lewis & Catlett, 1994) de un *sistema de reconocimiento de patrones,* resultando esta situación muy costosa para las organizaciones. En este mismo orden de ideas y ubicándonos en el contexto del *robo digital de datos,* es bien sabido, que las herramientas de desencriptado que utilizan los *ciber-delincuentes* suelen estar muy actualizadas, es por ello importante, diseñar nuevos *algoritmos de cifrado*, o en su defecto, realizar modificaciones, a los algoritmos ya existentes, para que los *ciber-criminales* desconozcan el procedimiento de descifrado, y al menos, tener nuestra información segura, durante un breve lapso de tiempo (Rangel et al., 2023, 2024, 2025b). Por ende, el hecho de *inyectar ruido* o *camuflaje* sobre el *texto cifrado*, resulta un buen indicador para confundir a los *ciber-criminales*. Lo anterior, sustentado en que hasta la misma *IA*, ha tenido que *"lidiar"* con el problema de presencia de ruido o patrones atípicos en *ME* o *conjuntos de datos*.

Para llevar a cabo el *cifrado de datos* basado en *inyección de ruido* y/o *camuflaje de textos*, existen varias propuestas. El primer término (conocido como: *noisy injection* o *random noisy*), significa que una secuencia de *texto plano* (S_i) o *texto cifrado* (C_i), contiene algunos caracteres adicionales (*ASCII* o *UTF-8)*, los cuales, no son parte del texto de entrada original (S_i). El segundo término (conocido como: *camouflaging text/ciphertext*, *reduced*

noisy o *reduced random mutation*), significa que se ha inyectado a la secuencia de *texto plano* o *encriptado*, algunos valores "disfrazados", los cuales, parecen caracteres *ASCII* o *UTF-8,* pero en realidad son otros tipos de datos. Por ejemplo, valores enteros u *ordinales*, *hexadecimales*, *pseudo-hexadecimales* (Rangel et al. 2023, 2024, 2025d), que están camuflados, simulando ser un texto *ASCII*, todo con el propósito de confundir a los ciber-delincuentes (Rangel et al., 2025a, 2025b, 2025d). En la presente investigación, solamente nos hemos enfocado en el estudio de *métodos de cifrado de datos*, particularmente, en la *inyección de ruido* (Rangel & Rangel, 2024a, 2024b, 2025a; Rangel et al., 2023, 2024, 2025a, 2025b, 2025c, 2025d) y *camuflaje de textos* (Rangel & Rangel, 2024a; Rangel et al., 2025a), empleando estrategias para *cifrado de datos dinámico* (Rangel & Rangel, 2025a; Rangel et al., 2025b), siendo combinado con el *algoritmo simétrico estándar AES-256*. Con respecto al estudio del *cifrado de datos* basado en *inyección de ruido* y/o *camuflaje de textos*, existen diferentes tipos de propuestas, que pueden ser agrupadas en varias categorías, todas ellas, basadas en *IA*.

La primer categoría consiste en *inyectar ruido,* utilizando los formatos: *hexadecimal* y *pseudo-hexadecimal*. Uno de los primeros trabajos en este sentido, propone el uso de un *algoritmo genético (AG)*, para seleccionar los alfabetos más óptimos "ruidosos" y poder emplear durante el proceso de cifrado/descrifrado de datos basado en un nuevo formato: *pseudo-hexadecimal*. La propuesta fue denominada: "*Noised" random pseudo-hexadecimal GAs* (Rangel et al., 2023). Esta alternativa presentaba errores o vulnerabilidades, y por el uso del *AG*, se observaba muy lento su procesamiento en el *cifrado de datos*, razón por la cual, se dió origen al diseño de una nueva metodología, omitiendo el uso completo del *AG,* aplicando solamente la *etapa de selección* del *AG*, durante la fase de aprendizaje del método, para poder generar los *alfabetos* de *cifrado/descifrado*, mediante la ejecución de un *método heurístico*. Esta nueva alternativa fue denominada como: "*Noised" random pseudo-hexadecimal I, versión estándar* (Rangel et al., 2024), la cual, omite ahora el calificativo *GAs*. En este mismo contexto, con el propósito de confundir a los ciber-delincuentes, Rangel et al. (2025d) proponen cuatro variantes de implementación para el *cifrado dinámico* con el mismo formato *pseudo-hexadecimal*, dando lugar a las nuevas propuestas: *Noisy random pseudo-hexadecimal I, noisy random pseudo-hexadecimal (by substitution) o por sustitución, noisy random pseudo-hexadecimal (by shifting) o por desplazamiento y noisy random pseudo-hexadecimal II (o versión 2).* Definitivamente, se trata de cuatro adaptaciones al procedimiento *aleatorio ruidoso* basado en formato *pseudo-hexadecimal*. Una última propuesta (Rangel, 2024), relacionada con el cifrado de datos basado en formato *pseudo-hexadecimal*, aún no ha sido publicada, pero se encuentra en proceso de revisión, la cual, ha sido denominada como: *"Noised" random 1-NN pseudo-hexadecimal* (Rangel, 2024; Rangel & Rangel, en revisión 2024). Este estudio propone el uso de la *regla del vecino más cercano* (Cover & Hart, 1967; Rangel, 2022; Rangel & Rangel, 2024b), también conocida como: *regla 1-NN*, siendo empleada en combinación con el *cifrado* basado en *pseudo-hexadecimal*, para *inyectar ruido*, haciendo más robusto este esquema, aunque sacrifica espacio en almacenamiento y tiempos de ejecución, se logra incrementar la seguridad de los datos, logrando confundir a los *ciber-delincuentes*. Se supone que existen al menos dos variantes al respecto, ya que, la alternativa enviada para su revisión, fue denominada: *"Noised" random 1-NN pseudo-hexadecimal II* (Rangel & Rangel, en revisión 2024). Por lo tanto, se entiende que la primera versión, es la presentada por Rangel (2024), como reporte parcial de un proyecto de investigación registrado con financiamiento en su dependencia. En lo concerniente con la *inyección de ruido* basada en codificación *hexadecimal*, existen al menos dos alternativas relacionadas (Rangel & Rangel, 2024b). La primera denominada como: *"Noised" random hexadecimal*, que fue presentada como una alternativa "ruidosa" rápida, mientras que la segunda propuesta, denominada como: *"Noised" random 1-NN hexadecimal*, sacrificaba tiempos en ejecución, pero fue presentada como una nueva alternativa *doblemente ruidosa*, por un lado mediante el esquema del procedimiento *"Noised"* basado en *alfabetos ruidosos* con codificación *hexadecimal*, mientras que, por otro lado, proyectaba el uso de la *regla del vecino más cercano* (Cover & Hart, 1967; Rangel, 2022; Rangel & Rangel, 2024b), que permitía la inyección de un patrón "*ruidoso*" al paquete final del encriptado, lo cual, llevaba la intención de confundir a los *ciber-delincuentes* (Rangel et al., 2025b). Estás propuestas surgen casi de manera paralela a las alternativas basadas en *pseudo-hexadecimal*, cuyo propósito también perseguía la *inyección de ruido*, pero sin hacer uso de este mismo formato denominado: *pseudo-hexadecimal*. Cabe mencionar, que estas propuesta basadas en *inyección de ruido hexadecimal* y *pseudo-hexadecimal*, solo han sido experimentadas utilizando *texto plano*.

Una segunda categoría, consiste en la aplicación de variantes del *cifrado del César*, siendo utilizado para *inyectar ruido* y *camuflajear texto plano*. Estas propuestas fueron presentadas por Rangel & Rangel (2024a), siendo denominadas como: *random Caesar, reduced random Caesar* y *reduced random mutation*. La primera, empleada para *inyección de ruido*, y las denominadas *reduced random*, fueron aplicadas para *camuflajear textos*. Según Rangel et al. (2025b), esta adaptación del *cifrado Caesar*, utiliza *métodos aleatorios* (Kuncheva & Jain, 1999; Rangel et al., 2025a; Reddaiah, 2016, 2019; Skalak, 1994), haciendo uso de *inteligencia artificial* (Iyengar et al., 2021a, 2021b; Rangel, 2002; Rangel, 2022; Rangel & Rangel, 2024a, 2024b), para conseguir el *cifrado de datos dinámico*. Dicha propuesta, denominada como: *random Caesar* (Rangel & Rangel, 2024a, 2024b), aunque ha sido poco estudiada, es recomendada para la *inyección de ruido* sobre *textos cifrados*, asegurando poder salvaguardar la información, e incluso, estar prevenidos contra futuros ataques basados en *computación cuántica* (Baklaga, 2024; Iavich et al., 2024) a través del uso de estrategias denominadas como: *random noisy* y *reduced noisy* (Rangel et al., 2025a, 2025b), las cuales, se describen más adelante. Según Rangel et al. (2025b), la metodología *random Caesar* (Rangel et al., 2023, 2025a; Rangel & Rangel, 2024a, 2024b), se puede aplicar usando diferentes modalidades, de acuerdo con la extensión del *módulo* (mod) empleado en los alfabetos de *cifrado/descrifrado*. Por ejemplo, *random Caesar I mod 255 & mod 9* (Rangel et al., 2023, 2025b), *random Caesar II mod 95* (Rangel & Rangel, 2024a; Rangel et al., 2023) y *random Caesar II mod 120* (Rangel et al., 2023, 2025a). Estas alternativas, a pesar de estar basadas en el procedimiento clásico de: *cifrado del César* o *algoritmo Caesar* (Barranco and Galindo 2022; Gómez et al. 2012; Rangel & Rangel, 2024b), se han considerado como un buen indicador en la seguridad de datos digitales, porque sus procedimientos están basados en *inteligencia artificial*, ya que, se utiliza un *método heurístico* que incluye el uso de *métodos aleatorios* para simular un alfabeto de *cifrado/descifrado*, a través del uso de un vector denominado K_i, siendo seleccionado mediante un proceso parecido a la *fase de selección* de un *algoritmo genético*, siendo empleado este valor como *desplazamiento*, aplicado a cada caracter del *texto plano,* con el propósito de *cifrar* los datos e incorporar en el paquete de cifrado, duplicando cada caracter en formato *ASCII* (American National Standards Institute, 1963) y/o como entero u *ordinal*, según sea el caso, para inyectar "*ruido*" en el *paquete final* encriptado, cuyo resultado *cifrado* es *dinámico*, bajo la hipótesis que ello confundiría a los *ciber-delincuentes* (Rangel et al., 2025b). Por ende, es sabido que el *algoritmo Caesar tradicional*, trabaja con un solo valor *K* para *desplazamiento* estático, definido por: $C_i = S_i + K \bmod 26$ (Barranco and Galindo 2022; Gómez et al. 2012), mientras que la implementación *Caesar* por *sustitución*, (Rangel & Rangel, 2024a, 2024b), está definida por: $C_i = Z_i \bmod 26$. Donde: $Z_i = D_t$, solamente cuando el caracter del *texto plano*: S_i , es igual que A_t , de otro modo se asigna: $Z_i = S_i$, porque el vector: A_t , corresponde al *alfabeto* original con *26* caracteres y el valor: D_t , es el *alfabeto* después de la operación de *desplazamiento*, el cual, puede ser obtenido como: $D_t = A_{(t+K)}$, solo cuando el valor de: $(t+K)$ es mayor que el módulo *26* , porque ello obliga a reiniciar la cuenta desde el principio otra vez (Rangel et al., 2025b). En cambio, tal como se ha mencionado previamente, *random Caesar* aplica un *desplazamiento* distinto (K_i) para cada caracter del *texto plano* (S_i), porque puede emplearse un *método heurístico* (similar al proceso de selección de un *AG*) para seleccionar el vector de desplazamientos más óptimo (K_i), o simplemente, utilizar K_i generado con *métodos aleatorios (con reemplazo)*, definiendo un *módulo N (mod N)*, para evitar que el alfabeto exceda el rango de la tabla *ASCII*. Una definición formal de *random Caesar,* puede ser la siguiente: *RandomCaesar <- FinalPackage* = C_i *&* K_i *& OrdChr(*C_i*)*. De acuerdo con Rangel et al. (2025), el valor de C_i corresponde al *cifrado parcial* o primera fase del procedimiento, el cual, puede ser obtenido por: $C_i = S_i + K_i \bmod N$. Donde: El vector S_i, es el *texto plano*, mientras que el vector K_i, son los *desplazamientos* seleccionados *aleatoriamente con reemplazo*, uno para cada caracter en S_i . Se considera el uso de *inteligencia artificial* en esta fase, porque el proceso de selección de K_i, es muy similar a la *fase de selección aleatoria* de un *algoritmo genético*. La función *OrdChr*, regresa un valor entero u ordinal cuando es empleado *random Caesar I*, mientras que, regresa un valor caracter *ASCII*, cuando se utiliza *random Caesar II*. El valor de *FinalPackage,* es el resultado del *paquete final* que contiene el *cifrado de datos*. El operador + indica la función *suma de enteros*, mientras que el operador *&*, indica la función *concatenación*. El valor *N*, se refiere al *módulo* (*mod*) utilizado, es decir, el número de caracteres a emplear en el *alfabeto*. Un valor de *N=9* o *N=255,* define el uso de *random Caesar I*, y en este último caso, se usan *255 tipos* de *desplazamientos*, que incluye todos los valores de la tabla *ASCII.* En cambio, el módulo 9, es fácil *descifrar* porque está muy limitado el rango de desplazamientos. Por ejemplo: $S_i \pm 9$, que equivale a: $(S_i - 9) \leq S_i \leq (S_i + 9)$. Por lo otra parte, un módulo con *N=95* y/o *N=120*, implica el uso de *random Caesar II*. Según Rangel et al. (2023, 2025b), un *módulo* 95 *(mod*

95*),* define un *alfabeto* de *95* caracteres que incluye valores enteros u ordinales desde *32 hasta 126* de la tabla *ASCII*. Por lo tanto, el *alfabeto* incluye solamente caracteres *imprimibles en pantalla* (desde el *espacio* hasta la *tilde*: ~). Para el caso del uso del *módulo 120 (mod 120)*, se incluyen *120* caracteres en el *alfabeto*, con valores ordinales dentro del rango entre *30* y *150*, correspondientes a la tabla *ASCII*, cuyos caracteres *no imprimibles en pantalla*, son considerados "*ruidosos*", los cuales, están dentro del rango entre *30 y 31,* así como, entre *127 y 150*, ambos casos, valores ordinales de la tabla *ASCII*. Este esquema, debido a que se utilizan *alfabetos* con distintos *módulos* (*mod*), podría confundir a los ciber-criminales, porque no podrían saber a simple vista, cuál de las tres variantes *random Caesar* ha sido empleada en un *FinalPackage*. Dentro de este mismo contexto, existe el *camuflaje de textos* (Rangel & Rangel, 2024a), que se trata de una adaptación realizada al *FinalPackage* del *random Caesar II mod 120*. La primera modificación, denominada como: *reduced random Caesar*, puede ser definida por: $ReducedRandomCaesar <- FinalPackage = C_i + ((char)(K_i)) \ mod \ 105$. Mientras que, la segunda adaptación presentada por Rangel & Rangel (2024a), denominada como: *reduced random mutation*, incluye un proceso basado en "mutación" (muy similar al empleado en *AG*), siendo definido de la siguiente manera: $ReducedRandomMutation <- FinalPackage = (C_1 + ((char)(K_1))) + (((char)(K_2)) + C_2) + ... + (C_{i-1} + ((char)(K_{i-1}))) + (((char)(K_i)) + C_i) \ mod \ 105$. En ambos casos, el vector C_i , contiene el *cifrado parcial* y está definido por: $C_i = S_i + K_i \ mod \ 105$, mientras que la función *char*, permite camuflajear (convertir) un valor entero u ordinal como caracter tipo *ASCII* o *UTF-8*. Según Rangel et al. (2025b), un *módulo 105*, solamente toma valores *aleatorios con reemplazo* dentro del rango entre *0* y *105*, para evitar que el caracter *cifrado con desplazamiento*, exceda el valor *ordinal 255* de la tabla *ASCII*. Lo anterior, considerando que el valor máximo para un *módulo 120,* es el *ordinal 150*. Por lo tanto: $(C_i + K_i) = 150 + 105$ (máximo valor de C_i y K_i , respectivamente), se obtiene como resultado el valor máximo de la tabla *ASCII*. Otra explicación que Rangel et al. (2025b) señalan respecto al uso de *IA*, es la siguiente: « *Se presume que ambos esquemas utilizan inteligencia artificial, ya que, sus procedimientos de cifrado, son equivalentes a los empleados en las fases de un algoritmo genético abreviado, porque solo se emplean las fases de selección aleatoria y mutación por intercambio, respectivamente, para cada método. Estos esquemas, fueron diseñados con el propósito de aumentar la velocidad de encriptado y reducir hasta ⅓ de espacio de almacenamiento en los resultados cifrados que genera: random Caesar II mod 120. Por lo tanto, se elimina el caracter repetido "ruidoso" del paquete final encriptado, y en su lugar, se define un nuevo concepto de camuflaje de textos, que en caso de: reduced random Caesar, se refiere a "disfrazar" cada desplazamiento del vector* K_i *como caracter ASCII. En cambio, con reduced random mutation, el concepto de camuflaje de textos, se refiere a intercambiar de lugar en el paquete final encriptado, a un caracter de texto cifrado por su correspondiente desplazamiento* K_i*. En otras palabras, un caracter ASCII cifrado se "disfraza" como desplazamiento* K_i*, y viceversa. De acuerdo con Rangel & Rangel (2024a), esta situación puede confundir a los ciber-delincuentes y no pone en riesgo la seguridad de los datos digitales en las organizaciones* ».

Una tercer categoría, es la aplicación de *random Caesar II mod 120* y *reduced random Caesar mod 105*, siendo empleadas para *inyectar ruido* y *camuflajear textos,* ambos casos*,* siendo aplicados sobre *textos cifrados* por algún algoritmo estándar, estrategias conocidas como: *random noisy* y *reduced noisy*, respectivamente (Rangel et al., 2025a). De acuerdo con Rangel et al. (2025b), las primeras investigaciones realizadas respecto al uso de estas metodologías: "*ruidosas aleatorias*" y "*reducidas ruidosas*", fueron experimentadas formalmente, mediante un procedimiento de fusión con los *algoritmos estándares*: *DES, 3DES, AES-256, Blowfish, RC4, WEP, ECIES-SECP-256-R1* y *RSA-2048*. Reportando al respecto, un conjunto de nuevas alternativas, que han sido denominadas como: *Random Noisy DES, Reduced Noisy DES, Random Noisy 3DES, Reduced Noisy 3DES, Random Noisy AES (AES-256), Reduced Noisy AES (AES-256), Random Noisy Blowfish, Reduced Noisy Blowfish, Random Noisy RC4, Reduced Noisy RC4, Random Noisy WEP, Reduced Noisy WEP, Random Noisy ECIES SECP-256-R1, Reduced Noisy ECIES SECP-256-R1, Random Noisy RSA-2048* y *Reduced Noisy RSA-2048*. Esta *fusión de técnicas*, en recientes investigaciones (Rangel et al., 2025a), ha sido empleada, primero aplicando el *algoritmo de cifrado estándar* sobre el *texto plano*, y posteriormente, se *inyecta ruido* al resultado encriptado, a través de la aplicación de *random Caesar II mod 120*, para el caso de estrategias: *random noisy*; mientras que, para el caso de estrategias: *reduced noisy*, se aplica *camuflaje de textos* con: *reduced random Caesar*. En este contexto, no se han encontrado estudios sobre la aplicación de *camuflaje* sobre de *textos cifrados* usando: *reduced random mutation*. Sin embargo, existen más investigaciones concernientes con el uso de las estrategias basadas en *random noisy*.

Por ejemplo, mediante la aplicación de inyección de ruido para encriptado de datos dinámico con el algoritmo GOST R 34.12-2015 (Rangel et al., 2025b). Un caso de estudio del algoritmo Salsa20, basado en inyección de ruido mejorado (Rangel et al., en revisión 2025a). El nuevo esquema dinámico CAST-128 y su comparación de resultados con cinco estrategias: random noisy (Rangel et al., 2025c). Otro caso de estudio concerniente con incrementar la seguridad de los textos cifrados empleando inyección de ruido combinado con el algoritmo ChaCha20 (Rangel et al., en revisión 2025b). En este mismo contexto, se logró mejorar la seguridad del algoritmo Camellia, mediante la inyección de ruido sobre textos cifrados utilizando procesos basados en inteligencia artificial (Rangel & Rangel, 2025a). También, se introduce una nueva propuesta basada en el algoritmo IDEA-128 combinando con inyección de ruido para incrementar la seguridad de los textos cifrados (Rangel et al., en revisión 2025d). Del mismo modo, se introduce otro caso de estudio con el algoritmo Twofish, combinado con estrategias: random noisy (Rangel & Rangel, en revisión 2025). Finalmente, se introduce la comparación de resultados de dos esquemas de cifrado asimétricos, combinando la inyección de ruido (Rangel et al., en revisión 2025c).

La presente investigación, es continuidad de dichos trabajos (Rangel & Rangel, 2024a; Rangel et al., 2025a), ya que, se ha considerado el estudio de las estrategias: *reduced random mutation,* siendo empleada por primera vez, sobre *texto cifrado* por el estándar *AES-256*. Del mismo modo, se evalúan las estrategias: *random noisy* y *reduced noisy*, siendo aplicadas exclusivamente, en combinación con el *algoritmo estándar: AES-256*, con el propósito de comparar resultados con la nueva alternativa aquí diseñada, que ha sido denominada como: *noisy mutation AES-256*, debido a que está compuesta por la fusión de la aplicación del algoritmo estándar *AES-256* combinado con *camuflaje de textos cifrados* basado en la estrategia: *reduced random mutation*. Lo anterior, sustentado porque en otras investigaciones (Rangel et al., 2025a, 2025b), aseguran que este tipo de alternativas *dinámicas*, pueden incrementar la seguridad de los datos, mejorando los esquemas de *encriptación*. Además, se observó que *reduced random mutation*, solo se ha experimentado con *texto plano*, pero no se han incluido pruebas sobre *texto cifrado* basado en el algoritmo *AES-256*. Esta situación no se considera buena para las organizaciones, particularmente aquellas que han optado por el uso de *AES-256*, considerando que las estrategias basadas en: *reduced noisy* y *random noisy*, así como, *reduced random mutation*, han revelado resultados muy prometedores. Por ello, la importancia de esta investigación, porque se llevaron a cabo experimentos basados en *inteligencia artificial* para la *inyección de ruido* y *camuflaje de textos cifrado* por el algoritmo *AES-256*, abriendo de este modo, un gran abanico de oportunidades a las organizaciones, respecto al uso de *AES-256* con *inyección de ruido* y *camuflaje de textos,* según sea el caso (Rangel et al., 2025b). Cabe aclarar que en el presente trabajo, solamente se ha evaluado el algoritmo *AES-256*, aplicando *inyección de ruido* con estrategias: *random noisy*. Del mismo modo, se hace uso de *camuflaje de textos* mediante la estrategia: *reduced noisy*. Finalmente, se introduce el nuevo concepto de *camuflaje de textos cifrados* empleando la estrategia: *reduced random mutation*. Dicha fusión, tal como se ha mencionado previamente, ha sido aquí denominada como: *noisy mutation AES-256*. Además, se considera importante el presente trabajo, para poder dar aportación empírica a favor de las metodologías relacionadas con *inyección de ruido* y *camuflaje de textos* cifrados*,* debido a que, dichos esquemas no han sido considerados aún metodologías estándar, y por ende, han sido poco estudiados en la literatura. Del mismo modo, las propuestas aquí presentada, tampoco han sido estudiadas sus vulnerabilidades mediante el uso de algoritmos basados en *computación cuántica*. En lo concerniente con el *objetivo* y *alcances* de la presente investigación, ello consiste en realizar el estudio de un caso particular, basado en el *cifrado de datos* con el algoritmo *AES-256*, siendo empleado en combinación de *inyección de ruido* (Rangel et al., 2025a) y *camuflaje de textos* (Rangel & Rangel, 2024a; Rangel et al., 2025a), cada una de manera separada durante el *cifrado de datos* para convertirlo en un esquema *dinámico*, bajo la *hipótesis* de que es posible mejorar el esquema de seguridad del denominado *algoritmo AES-256*, mediante el *camuflaje de textos* basado en procesos de *mutación*, y por ende, poder evitar que los *textos cifrados* sean descifrados con facilidad por parte de los *ciber-delincuentes*, e incluso, para estar prevenidos contra ataques basados en *computación cuántica* (Baklaga, 2024; Iavich et al., 2024). El método propuesto: *noisy mutation AES-256*, a pesar que se observó durante los experimentos, que demora más tiempo en el proceso de encriptado, este retardo no se comprobó ser sustancial en comparación con el resto de las estrategias aquí experimentadas. Por lo tanto, *noisy mutation,* es aquí recomendado como nueva alternativa para el *cifrado dinámico,* porque permite confundir a los ciber-delincuentes, incrementando la seguridad de los datos digitales en

las organizaciones. Es por tal razón, la *justificación* de la presente investigación, debido a que, las alternativas sobre *inyección de ruido* y *camuflaje de textos,* según Rangel et al. (2025b), se consideran muy prometedoras, pero han sido poco estudiadas en la literatura, y por ende, la *computación cuántica* no ha estudiado sus vulnerabilidades (Rangel et al., 2025a). Por lo tanto, en este trabajo, se propone dar aportación empírica a favor de la *inyección de ruido y camuflaje* de *textos cifrados*, presentando un caso de estudio basado sobre la estrategia: *reduced random mutation*, siendo aplicado a *textos cifrados* por el *algoritmo AES-256*. Además, la comparación de resultados, también es presentada en términos de *precisión del sistema* (Barandela et al., 2003), conocida también como: *global average* o *global accuracy* (Lewis & Catlett, 1994). Ello ha sido calculado, empleando una nueva modalidad de *validación cruzada* para poder contrastar la información con otras investigaciones relacionadas (Rangel et al., 2023, 2025a; Rangel & Rangel, 2024a, 2024b). Por último, se introduce la comparación de resultados de esta nueva propuesta: *noisy mutation AES-256*, con los obtenidos por el algoritmo *AES-256* estándar. También, se comparan resultados generados por la *inyección de ruido* y *camuflaje de textos,* basados en estrategias: *random noisy AES-256* y *reduced noisy AES-256*, respectivamente.

2. METODOLOGÍA

Esta investigación es de tipo *exploratoria* y *cuantitativa*, la cual, se llevó a cabo en las instalaciones del Instituto Tecnológico de Ciudad Altamirano, campus del Tecnológico Nacional de México, ubicado en el Municipio de Pungarabato, Estado de Guerrero, México. El principal enfoque de la investigación se considera *exploratoria*, ya que, las estrategias de *camuflaje de textos* cifrados, que fueron empleadas en la experimentación, han sido poco estudiadas en la literatura.

2.1. Diseño de la investigación. Nueva propuesta: Noisy Mutation AES-256

El método de *cifrado de datos dinámico*, diseñado como nueva propuesta en la presente investigación, es una fusión de técnicas basada en la aplicación del algoritmo *AES-256* (Barranco & Galindo, 2022; Daemen & Rijmen, 2002; NIST, 2001; Rahman & Hossain, 2021; Rangel & Rangel, 2025a; Rodríguez, 2020; Van-Tilborg, 2005), siendo combinado con la metodología: *reduced random mutation* (Rangel & Rangel, 2024a). Esta metodología ha sido aquí denominada como: *noisy mutation AES-256 (algoritmo aleatorio reducido-ruidoso AES-256 basado en procesos de mutación, o bien, algoritmo AES-256 con camuflaje de textos cifrados aleatorio).* Este esquema fue diseñado con el propósito de obtener un *dinámico* resultado en el *proceso de encriptación de datos*, con el algoritmo *AES, usando el* estándar *AES-256*, ya que, se ha observado que los resultados cifrados de este esquema estándar son del tipo estático (ver *Tabla 2*), aunque ello no significa que sea vulnerable el algoritmo en todos los casos. Sin embargo, en previas investigaciones (Rangel et al., 2023, 2024, 2025a, 2025b, 2025c), se han destacado inconvenientes sobre esta situación, respecto a los esquemas de *cifrado estáticos*, señalando que pueden estar propensos a posibles *ciberataques* por parte de los *ciber-delincuentes*.

Al respecto, en el presente trabajo, se implementaron dos alternativas para *camuflaje de textos.* La primera, basada en metodología *reduced noisy* (Rangel et al., 2025a) que usa el método *reduced random Caesar* (Rangel & Rangel, 2024a), mientras que la segunda estrategia, utiliza el método *reduced random mutation* (Rangel & Rangel, 2024a), siendo aplicada a *textos cifrados* por el algoritmo *AES-256*, fusión que ha sido aquí denominada como estrategia: *noisy mutation (ver Figura 1).* También, fue implementada una tercer estrategia, basada en *inyección de ruido,* utilizando el esquema de: *random noisy* (Rangel et al., 2025a), siendo combinadas por la previa ejecución del proceso de cifrado del algoritmo: *AES-256,* con el propósito de hacer frente al problema de ciberataques y amortiguar un poco o disminuir el *robo digital de datos* en las organizaciones.

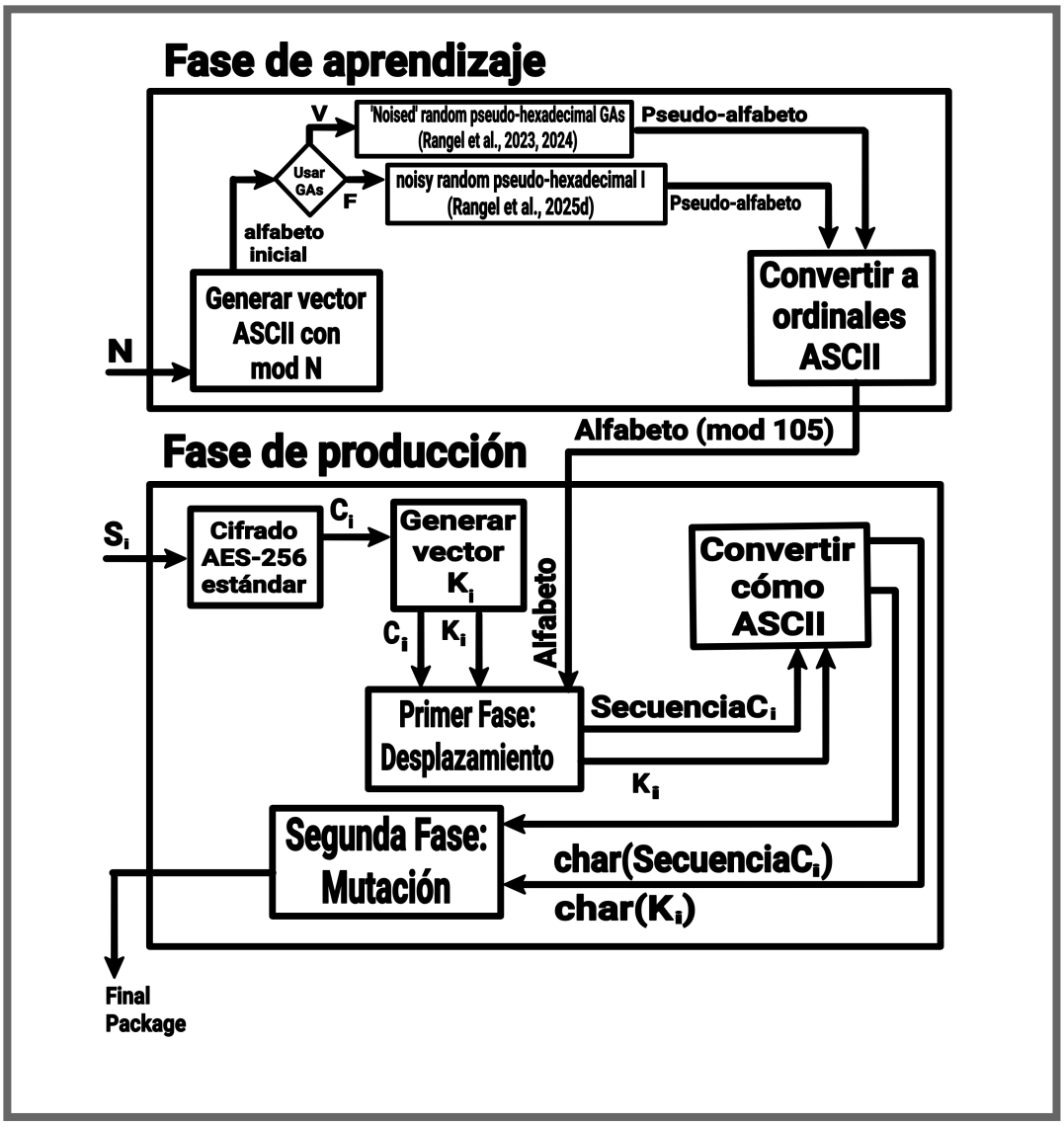

Figura 1: Diagrama de procesos para cifrado de datos usando: noisy mutation AES-256

La forma en que ha sido empleada la primer estrategia: *reduced noisy AES-256,* fue utilizando el mismo modo que describe: Rangel et al. (2025a). Primero, se hizo el cifrado de datos sobre el texto plano, empleando el algoritmo estándar *AES-256*, obteniendo un *texto cifrado estático*. Posteriormente, se hizo uso de la estrategia: *reduced random Caesar*, siendo aplicada sobre el *texto cifrado*, utilizando un *módulo 105*. Del mismo modo, la segunda estrategia: *noisy mutation AES-256*, se utilizó primero cifrando el *texto plano* con algoritmo *AES-256* estándar, y a este *texto cifrado* resultante, se le aplicó el método: *reduced random mutation*, también con *módulo 105*, siguiendo la forma que describe: Rangel & Rangel (2024a), con la excepción que, en la presente investigación ha sido aplicada la estrategia sobre *texto cifrado*. Debido a que no había sido empleada está fusión de métodos o algoritmos, en previas investigaciones, se considera nueva propuesta, y por ende, se tuvo que diseñar una definición formal para dicha estrategia, la cual, se describe más adelante. Por último, la tercer estrategia: *random noisy AES-256*, fue empleada con el propósito de comparar resultados con otras investigaciones (Rangel, 2024; Rangel & Rangel, 2024a, 2024b, 2025a, en revisión 2024, en revisión 2025; Rangel et al., 2023, 2024, 2025a,

2025b, 2025c, 2025d, en revisión 2025a, en revisión 2025b, en revisión 2025c, en revisión 2025d), utilizando un *módulo 120*, siguiendo el mismo procedimiento que describe: Rangel et al. (2025a). Es decir, iniciando el cifrado de datos con algoritmo *AES-256* sobre el *texto plano*, y luego, se aplica *random Caesar II* (Rangel & Rangel, 2024a), a este resultado de *texto cifrado*. En forma adicional, y con la finalidad de comparar resultados y poder observar si los tiempos o *estimación del error* pueden ser considerados significativos o no, fue necesario realizar experimentos con los métodos o algoritmos estándar: *AES-256, random Caesar II módulo 120, reduced random Caesar* y *reduced random mutation*, aclarando que en estos casos, cada estrategia fue aplicada exclusivamente a *texto plano*. Cabe mencionar que, para poder obtener los *promedios* y *desviación estándar* de cada una de las estrategias aquí experimentadas, se procede en todos los casos, a aplicar una variante de *validación cruzada* (Rangel & Rangel, 2024a; Rangel et al., 2023, 2024, 2025a), la cual, ha sido diseñada específicamente para trabajar con cifrado/descifrado de datos. Se aplicó este método como *estimación de error/tiempo*, a cada una de las estrategias, en forma separada, utilizando una *muestra de entrenamiento* independiente, diseñada específicamente para cada experimento. Más adelante se describen detalles sobre el diseño de la *muestra de entrenamiento*.

En lo que concierne con la definición formal, utilizada en la presente investigación, para implementar la estrategia: *noisy mutation AES-256*, se puede definir como sigue: $NoisyMutationAES256 <- FinalPackage = (char(ord (StandardEncryptionAES256_1) + ord (K_1)) \& char(ord (K_2) + ord (StandardEncryptionAES256_2)) \& ... \& char(ord (StandardEncryptionAES256_{i-1}) + ord (K_{i-1})) \& char(ord (K_i) + ord (StandardEncryptionAES256_i)) \bmod N$. En la Figura 1, se muestra el diagrama de procesos para obtener el cifrado dinámico utilizando: noisy mutation AES-256. Una forma de calcular esta secuencia cifrada, puede ser obtenida implementando el siguiente pseudocódigo:

```
FUNCIÓN NoisyMutationAES256
INICIO
 cont = 0
 SecuenciaCi = ""
 PseudoAlfabeto = crear_con_noisy_random_pseudohexadecimal()
 Alfabeto = convertir_pseudohexa2ordinalASCII(PseudoAlfabeto)
  PEDIR Si
  txt = StandardEncryptionAES256(Si)
  t = numero_de_caracteres_en (txt)
  S  = arreglo[0..t]
  Ki = arreglo[0..t]
  Ci = arreglo[0..t]
  PARA J = 0 HASTA (t-1) HACER
    S[k] = ord(txt[J])
  FIN-PARA
  PARA I = 0 HASTA (t-1) HACER
    Ki [ I ] = numero_aleatorio(0, 105)
  FIN-PARA
  PARA I = 0 HASTA (t-1) HACER
     p = buscar_posicion_en_Alfabeto ( S[ I ] ) + Ki [ I ]
     Ci [ I ] = Alfabeto [ p ]
  FIN-PARA
```

```
  PARA J = 0 HASTA (t-1) HACER
    SI (J ES PAR) ENTONCES
       SecuenciaCi = SecuenciaCi & ("" & ((char(Ci [J]))) & "" & ((char(Ki [J]))) & "")
     OTRO CASO
       SecuenciaCi = SecuenciaCi & ("" & ((char(Ki [J]))) & "" & ((char(Ci [J]))) & "")
      FIN-SI
   FIN-PARA
 FinalPackage = SecuenciaCi
 regresar  (FinalPackage)
FIN
```

Donde: El operador *&* , se refiere a la función *concatenación* de caracteres, mientras que el operador +, refiere a la *suma* de números ordinales, que en realidad efectúa el *desplazamiento* de K_i, sobre el *alfabeto* de cifrado/descifrado. La función *ord* convierte un número entero o caracter de la tabla *ASCII (American National Standards Institute, 1963)* o *UTF-8 (Unicode Consortium, 2022)* a su valor ordinal correspondiente, mientras que la función *char*, convierte un valor entero u ordinal a caracter *ASCII* o *UTF-8*, según sea el caso. La cadena de caracteres o vector*:* $StandardEncryptionAES256_i$, es el resultado de *texto cifrado* obtenido por el algoritmo estándar *AES-256* (es decir, se utiliza el *texto plano* de entrada para ser *cifrado* por: *AES-256*. Dicho resultado, contiene el vector: $StandardEncryptionAES256_i$). El vector K_i , representa el *desplazamiento en el alfabeto con mod N,* respecto a cada uno de los caracteres almacenados en el *texto cifrado* ($StandardEncryptionAES256_i$). ***Ello*** significa que K_i , contiene valores enteros u ordinales, delimitados por el módulo N (mod N), cuyo alfabeto de cifrado/descifrado ha sido generado con inteligencia artificial (Iyengar et al., 2021a, 2021b; Rangel, 2002; Rangel, 2022; Rangel et al., 2023), utilizando el mismo procedimiento basado en un método heurístico descrito en otras investigaciones (Rangel & Rangel, 2024a, 2024b, 2025a; Rangel et al., 2024, 2025a, 2025b, 2025c, 2025d). Este método heurístico, se apoya en el uso de métodos aleatorios con reemplazo (Kuncheva & Jain, 1999; Rangel & Rangel 2024a; Rangel et al., 2023, 2024, 2025a, 2025b; Reddaiah, 2016, 2019; Skalak, 1994), mediante un procedimiento muy similar a la fase de selección de un algoritmo genético (Clark, 1994; Delman, 2004; Griindlingh & Van-Vuuren, 2002; Kalsi et al., 2018; Kuncheva & Jain, 1999; Matthews, 1993; Rangel et al., 2023, 2024, 2025a, 2025b; Reddaiah, 2019; Skalak,1994), siendo aplicado del mismo modo que señala Rangel et al. (2025d), cuando describe el procedimiento del denominado noisy random pseudo-hexadecimal por desplazamiento, con la excepción de que en la presente investigación se omite el uso del formato pseudo-hexadecimal, y en su lugar, se emplean los caracteres en formato ASCII o UTF-8, e incluso, los valores ordinales también son empleados para realizar la operación de los desplazamientos, tal como se ha descrito previamente. Por lo tanto, el proceso basado en mutación, que se aprecia en la definición formal de: noisy mutation AES-256 (NoisyMutationAES256), se aplica en la segunda etapa del cifrado para la generación del empaquetado final, al momento de intercambiar los valores K_i de posición en el FinalPackage, se puede deducir que utiliza una mutación por intercambio. En la presente investigación, el vector K_i , ha sido delimitado con módulo 105 (mod 105). Es decir, solamente toma valores aleatorios (con reemplazo) dentro del rango entre 0 y 105, para evitar desplazamientos con valores ordinales fuera de la tabla ASCII. Ello está sustentado de ese modo, porque si consideramos que un alfabeto generado con un módulo 120, su valor máximo es el ordinal 150. Por lo tanto, es el valor ordinal mayor que debería tener un caracter seleccionado en el vector: $StandardEncryptionAES256_i$ (aunque en la práctica, ello depende de los resultados arrojados por el algoritmo AES-256 estándar, lo cual, puede hacer migrar un caracter de la tabla ASCII convirtiéndolo en un valor correspondiente a la codificación: UTF-8). Por consiguiente, continuando con el mismo orden de ideas, la suma de los ordinales que corresponden al caracter cifrado y su desplazamiento, al ser calculado como: ($StandardEncryptionAES256_i + K_i$) = (150 + 105) = 255, denota la delimitación con el máximo valor de la tabla ASCII. Cabe aclarar, que el alfabeto final para cifrado/descrifrado de datos, con dimensión establecida por el mod N, no contiene sus caracteres ordenados de acuerdo con los valores ordinales de la tabla ASCII, ya que, estos han sido seleccionados de manera aleatoria, a

través de una fase de aprendizaje (Rangel et al., 2023, 2025d). Por ende, la operación: ord ($StandardEncryptionAES256_i$) + ord (K_i), no implica obligatoriamente una suma de valores ordinales, ya que, este esquema refiere más bien, a un desplazamiento del valor seleccionado en K_i , aplicado sobre el alfabeto de cifrado/descrifrado, mediante la sustitución de su carácter y ordinal correspondiente, según sea el caso. Enseguida, se muestra un ejemplo, sobre este procedimiento, considerando un caso particular generado con un texto plano, cuyo valor, corresponde a la palabra: "▓Welcome▓□", simulando una contraseña que incluye en la entrada de texto plano, al menos un caracter fuera del rango de la tabla ASCII.

En este ejemplo, se considera la información del diagrama de procesos que muestra la Figura 1, así como, las instrucciones mostradas en el pseudocódigo, descrito previamente. Suponiendo que la etapa o fase de aprendizaje inicial basada en inteligencia artificial, nos ha generado un alfabeto de cifrado/descifrado, que al ser convertido a valores ASCII queda del siguiente modo: $Alfabeto_i$ = ['5', '6", 7', '8', '9', '0', '1' , '2', '3', '4', 'a', 'b', 'c', 'd', 'e', 'f', '▓', '□', '~', '|', '•', '√', '£', 'M', 'N' , 'Ñ', 'O', 'P', 'g', 'h', 'i', 'j', 'k', 'l', 'm', '@', '#', '$', '_', '&', 'n', 'ñ', 'o', 'p', 'q', 'r', 's', 'Q', 'R', 'S', 'T', 'U', '-', '+', '(', ')', '/', 'A', 'B', 'C', 'D', 'E', 't', 'u', 'v', 'w', 'x', 'y', 'z', '*', ':', ';', '!', '?', 'F', 'G', 'H', 'I', 'K', 'L', ***... HASTA COMPLETAR EL RANGO DEL mod 105 ...], y considerando que la cadena de texto plano ha sido encriptada con el estándar AES-256, obteniendo el siguiente resultado cifrado (ver Tabla 2): txt =*** "30303030303030303030303030303030319421780fc40c59f14796a598115d8e5139f6f9666f67cb30fcda01f22739eeda". ***Por lo tanto, una vez que ha sido convertida a vector, quedaría como:*** txt = StandardEncryptionAES256i = ['3', '0','3','0','3','0','3','0','3','0','3','0','3','0','3','0','3','0','3','0', '3','0','3','0','3','0','3','0','3','0','3','1','9','4','2','1','7','8','0', 'f','c','4','0','c','5','9','f','1','4','7','9','6','a','5','9','8','1','1', '5','d','8','e','5','1','3','9','f','6','f','9','6','6','6','f','6','7','c', 'b','3','0','f','c','d','a','0','1','f','2','2','7','3','9','e','e','d','a']. ***En este caso, se considera necesario generar aleatoriamente un vector K_i de la misma dimensión, respetando los rangos del mod 105. Suponiendo que este vector, se ha generado con los números: 33, 34 y 35, para sus desplazamientos, quedando como sigue:*** Ki = [33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 33, 35, 33, 34,35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35, 33, 34, 35]. ***Por lo tanto, el vector cifrado, después de aplicar los desplazamientos sería:*** Ci = ['ñ', '&', 'p', '_', 'o', 'n', 'ñ', '&', 'p', '_', 'o', 'n', 'ñ', '&', 'p', '_', 'o', 'n', 'ñ', '&', 'p', '_', 'o', 'n', 'ñ', '&', 'p', '_', 'o', 'n', 'ñ', 'n', '&', 'o', 'n', 'ñ', 0, '$', 'n', 'R', 's', 'q', '_', 's', '#', '$', 'S', 'ñ', 'o', 0, '&', 0, 'q', '#', '$', '$', 'ñ', '&', '@', 'R', '#', 'R', '#', '&', 'o', '&', 'R', 0, 'T', '$', 0, 0, 0, 'S', 0, 0, 's', 's', 'ñ', '&', 'T', 'r', 'Q', 'r', '_', 'n', 'T', 'n', 'ñ', 0, 'ñ', '_', 'S', 'Q', 'Q', 'r']. ***Pero, si consideramos que, los valores de los desplazamientos: 33, 34 y 35, corresponden a los caracteres ASCII:*** '!", '"', ***y*** '#', ***respectivamente. Entonces, el vector K_i camuflajeado como ASCII sería:*** Ki = ['!', '"', '#', '!', '"', '#', '!', '"', '#', '!', '"', '#', '!', '"', '#', '!', '"', '#', '!', '"', '#', '!', '"', '#', '!', '"', '#', '!', '"', '#', '!', '"', '#', '!', '!', '#', '!', '"', '#']. ***En estos términos,***

después de aplicar mutación del 50%, por intercambio a cada par, tendríamos un empaquetado del siguiente modo: **`FinalPackage = 'ñ!"&p#!_o"#nñ!"&p#!_o"#nñ!"&p#!_o"#nñ!"&p#!_o"#nñ!"n&#!on!#ñ0!" $n#!Rs"#q_!"s##!$S"#ño!"0&#!0q"##$!"$ñ#!&@"#R#!"R##!&o"#&R!"0T#!$0"#00!"S0#! 0s"#sñ!"&T#!rQ"#r_!"nT#!nñ"#0ñ!"_S#!QQ"#r'`. *Para validar esta información, solo vamos a comparar las primeras cuatro secuencias del cifrado, usando:* **`Ci = [ 'ñ', '&', 'p', '_', ...]`**, *así como,* **`Ki = [ '!' , '"' , '#', '!' , ... ]`**. *Entonces,* **`Tested = FinalPackage = (C1=ñ , K1=!) & (K2= " , C2= & ) & ( C3=p , K3=# ) & (K4= ! , C4= _ )`**. *Simplificando tenemos:* **`[ ( ñ , ! ) & ( " , & ) & ( p , # ) & ( ! , _ ) ]`**. *Reduciendo por concatenación:* **`[ ñ , ! , " , & , p , # , ! , _ ]`**, *lo cual, coincide perfectamente con los primeros ocho caracteres de la secuencia final empaquetada:* **`FinalPackage = " ñ!"&p#! ...."`**.

Por otra parte, para implementar el resultado cifrado, de este tipo de operación con *AES-256*, fue necesario utilizar algunas bibliotecas o librerías del lenguaje *Python 3* (Python.org, 2024), como es el caso de: *Crypto.Cipher* de *pycryptodome* (PyCryptodome, 2025), así como, *cryptography* (Cryptography, 2025) y *Pycryptodome* de *PyPI* (PyPI, 2024). Por lo tanto, para trabajar el *cifrado* de datos con este estándar *AES-256*, se puede obtener ejecutando el siguiente código fuente escrito en *Python 3*: `from cryptography.hazmat.primitives import padding ; from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes ; from cryptography.hazmat.backends import default_backend ; textoplano = "Welcome".encode() ; IV= b"0000000000000001" ; Key = b"00000000000000000000000000000001" ; KeySize=256 ; Ci = Cipher(algorithms.AES(Key), modes.CBC(IV), backend = default_backend()).encryptor(); Ri = padding.PKCS7(KeySize).padder(); Ri = Ri.update(textoplano) + Ri.finalize(); Ci = (IV + (Ci.update(Ri) + Ci.finalize())).hex() ; print(Ci)`. Dónde: El parámetro *textoplano*, es el *texto plano* de entrada *(S_i)* codificado usando el método *encode()*, mientras que, el parámetro: *Key*, es la *llave secreta* (en bytes, por ejemplo: *b"00000000000000000000000000000001"),* la cual, tiene un tamaño máximo definido por: *KeySize* (por ejemplo: 256). La variable *IV* (también definida en bytes: *b"0000000000000001"*), es el *vector de inicialización* que es requerido por *AES-256*, al utilizar el *modo CBC* (Stinson & Paterson, 2019). La variable *Ci*, guarda el *texto cifrado* por: *AES-256,* mientras que, para obtener dicho resultado, es requerido un desplazamiento o relleno (*padding)*. Es por ello, necesario el uso de *Ri*, porque permite *instanciar* un *padding* empleando el método de relleno: *PKCS7* (Van & Jajodia, 2011; Van-Tilborg, 2005). Cabe recordar que, la nueva propuesta: *noisy mutation AES-256* , trabaja de manera separada, cifrando primero el *texto plano* con el estándar: *AES-256*, y posteriormente, se aplica el método: *reduced random mutation con módulo 105,* sobre el *texto cifrado* resultante.

Por último, de acuerdo con Rangel et al. (2025b), esta fusión de técnicas que hacen uso de algoritmos *estándar*, combinados con estrategias basadas en *inyección de ruido* o *camuflaje de textos* (por ejemplo: *random noisy* o *reduced noisy*), podrían ser mal interpretadas como un *doble cifrado de datos* utilizando diferentes algoritmos, lo cual, en los experimentos presentados, no ha sido enfocado de esa manera, ya que, el hecho de aplicar el cifrado de datos usando dos algoritmos estándar repetidamente, sobre un *texto plano*, y posteriormente, aplicado sobre el resultado de *texto cifrado*, ello no garantiza la seguridad de la información, debido a que, podría ser *desencriptado* fácilmente por un *ciber-delincuente* utilizando algún tipo de herramienta de software o *estrategia cibercriminal*, o mediante la *fuerza bruta*, un *diccionario*, o desarrollando algún algoritmo basados en *computación cuántica*, o incluso, haciendo uso de un lenguaje de programación (como: *Python)*, y mediante una serie de ciclos (*for* o *while*), a través de sus distintas y muy amplia gama de bibliotecas que proveen procedimientos para el *cifrado de datos* usando algoritmo estándar. Esta situación no ocurre con la *inyección de ruido* y/o *camuflaje de textos*, ya que, dichas estrategias, no son muy conocidas por los *ciber-delincuentes*, y además, los lenguajes de programación tradicionales, aún no tienen librerías con soporte para este tipo de estrategias. Finalmente, cabe recordar que, este tipo de esquemas "ruidosos" aleatorios, puede adaptarse para que solamente sea aplicado a ciertos fragmentos del *texto cifrado,* ya sea, *inyectando ruido* o *camuflajeando textos,* empleando la notación descrita por otros autores (Rangel et al., 2025a). En la presente investigación, se pudo observar que también es posible realizarlo con el método: *reduced random mutation*, incorporando esta nueva opción a la estrategia: *noisy mutation AES-256*, que en términos generales, puede ser definida formalmente como sigue: *PartialNoisyMutation = [char(N) ,*

char(RandPos$_i$) , char(KVector$_i$) , NoisyCiphertext$_i$]. Donde: *PartialNoisyMutation* es el resultado de aplicar *noisy mutation* en forma parcial, solamente a una parte del *FinalPackage*. El valor de *N* corresponde al número de elementos que será aplicado el *camuflaje de textos*. El vector *RandPos$_i$* guarda las posiciones de *texto cifrado* que se ha aplicado el *camuflaje de textos*, mientras que el *KVector$_i$* contiene los valores aleatorios de K_i que serán utilizados, siendo delimitados por *N* y *RandPos$_i$*. El vector *NoisyCiphertext$_i$* refiere al *texto cifrado* que ha sido obtenido por algún método o algoritmo de cifrado de datos. Teniendo en cuenta que *KVector$_i$* es calculado solamente en la posición *RandPos$_i$* con un tamaño definido por *N*. La función *char*, permite *camuflajear* un vector o algún otro tipo de datos convirtiéndolo como un caracter de la tabla *ASCII* o *UTF-8*. En estos términos, los *ciber-delincuentes* prácticamente tendrían que "*adivinar*", primero dónde fue realizada la *inyección de ruido* o *camuflaje de textos*, y posteriormente, determinar cuál es el valor de la *llave secreta* utilizada en el cifrado con el algoritmo estándar, para poder aplicar el *descrifrado de datos*. Dicha tarea, puede resultar muy compleja para los *ciber-criminales*, e incluso, puede retardar los procesos de algoritmos basados en *computación cuántica*.

2.2. MATERIALES, EQUIPO O RECURSOS UTILIZADOS

La unidad experimental utilizada para realizar la implementación y pruebas del presente trabajo de investigación, fue principalmente, la *Sala de Investigación* o *Sala De Educación a Distancia.* También, algunos experimentos y validaciones se llevaron a cabo en las salas de cómputo: *SC1* y *SC3* del *Edificio 700*, espacios ubicados dentro de las instalaciones del Instituto Tecnológico de Ciudad Altamirano, campus del Tecnológico Nacional de México. En dichos espacios, se utilizaron computadoras disponibles, cuidando que, en todos los casos, los equipos presentaran las mismas características, específicamente, con una capacidad de 8 GB de memoria, espacio disponible en disco de 500 GB y una velocidad procesador con 2 Ghz. Estos equipos de cómputo, tenían instalado el sistema operativo *Windows 10* (Microsoft, 2024), y se tuvo que instalar el lenguaje de programación *Python 3* (Python.org, 2024). En forma adicional, y como medida de validación de las pruebas realizadas, se llevaron a cabo algunos experimentos usando un *equipo de cómputo móvil* con sistema operativo *Android 9* (Android, 2024), con 4 GB de capacidad de memoria y espacio de almacenamiento disponible de 64 GB. Este equipo móvil contaba con 2 Ghz de velocidad en el procesador. En este caso, tuvo que instalarse la aplicación *Pydroid3* (Pydroid3, 2024), para poder ejecutar el código fuente escrito en lenguaje de programación *Python 3*. La finalidad del uso de este último equipo móvil, fue para corroborar resultados. En nuestra experimentación, no se observó diferencia sustancial en los tiempos de procesamiento con los equipos de cómputo empleados.

En lo que concierne con los *datos* utilizados, se trata de *muestras de entrenamiento (ME),* que contienen mil filas de patrones cuya estructura cuenta con cinco atributos de distintos tipos de datos, así como, una columna adicional, que guardar la *etiqueta de clase*. ***Por lo tanto, el tamaño de la muestra de entrenamiento (ME) es de mil ejemplares con cinco columnas. Cada ME fue diseñada específicamente para cada método evaluado, ya que, en dos de sus columnas son almacenados diferentes ejemplares de texto cifrado, que solamente puede corresponder al método de cifrado experimentado.*** Estos dos *atributos*, están específicamente diseñados para almacenar secuencias de *texto cifrado* (*Test 1* y *Test 2*), que pueden guardar valores del tipo *hexadecimal* o *cadena de texto* ("ruidosa"), o incluso, pueden instanciar un *vector* de tipo *entero* u *ordinal*. Lo anterior, depende del método o algoritmo de *cifrado*, el cual, haya sido empleado para la generación de estos resultados *cifrados*. Esta es la razón, del porqué, para cada estrategia o método de encriptado, aquí evaluados, fue necesario preparar una *ME* específica para evaluar sus propios experimentos. También, se incluyen tres columnas (atributos) de tipo de datos *numéricos* (valor real o doble precisión), diseñados para guardar los tiempos de cifrado *(TC)* y descrifrado *(TD)*, así como, el porcentaje de error ocurrido durante la operación de descifrado. Otra columna de la *ME*, es la *etiqueta de clase (Label),* atributo del *patrón de entrenamiento* (Rangel, 2002), que permite guardar valores del tipo cadena de texto, ya sea, en formato *ASCII* (American National Standards Institute, 1963) o en codificación *UTF-8* (Unicode Consortium, 2022). En esta investigación, este valor corresponde al mismo *texto plano (S$_i$)*, que consiste en *palabras de prueba* o secuencias aleatorias de máximo 255 caracteres, que simulan ser *claves* o *"password"*, a los cuales, se le aplicó dos veces el proceso de *cifrado de datos*, dando lugar a los atributos *Test 1* y *Test 2*, que son

incluidos en cada *ME* (ver *Tabla 2)*. Estás cadenas de *texto plano*, en ocasiones fueron convertidas a *vector entero u ordinal,* para poder llevar a cabo su procesamiento con cada algoritmo o *método de cifrado* aquí experimentados. Debido a que las condiciones de prueba fueron planteadas por el uso de una nueva variante de validación cruzada (Rangel et al., 2023), y con el propósito de poder comparar resultados con otras investigaciones (Rangel et al., 2025a; Rangel & Rangel, 2024a, 2024b), se ha empleado la validación cruzada modificada usando cinco iteraciones por experimento, de manera separada, para cada uno de los métodos de cifrado aquí experimentados. Lo anterior, utilizando los equipos de cómputo e infraestructura, descritos previamente.

2.3. PROCEDIMIENTO

El presente estudio, se ha llevado a cabo, siguiendo el mismo orden de ideas, como se ha puntualizado en la metodología, tal como se describe a continuación: En primer lugar, fueron preparados los datos (reitero, una *ME* específicamente para cada experimento). En lo que concierne a las *variables* evaluadas durante la experimentación, fueron guardadas en la *ME* como información básica. Dichos datos, ya comentados previamente, incluyen una *etiqueta de clase* o *texto plano* (S_i), dos ejemplares de cadenas de *texto cifrado (Test 1 y Test 2),* los tiempos de encriptado *(TC)* y descifrado *(TD)* medidos en milisegundos, así como, el porcentaje de *error ocurrido.* Posteriormente, usando la *ME* correspondiente, se procede a la aplicación del método o estrategia de cifrado de datos, utilizando una variante del método *validación cruzada*, con cinco repeticiones, tal como se ha empleado en estudios previos relacionados con la *encriptación de datos*, todo con el propósito de comparar resultados con otras investigaciones (Rangel et al., 2023, 2024, 2025a; Rangel & Rangel, 2024a, 2024b, 2025a). Este proceso permite obtener el *promedio* y la *desviación estándar,* para cada caso*,* que incluye cálculo del *error* ocurrido y de los *tiempos* de respuesta de cada *método* o estrategia aquí evaluada*.* Considerando que, si un *texto encriptado*, al ser *descrifrado* por el método o algoritmo, este resultado no coincide con la entrada de *texto plano*, entonces es anotado como *error*. La aplicación de la variante de *validación cruzada* (Rangel & Rangel, 2024a; Rangel et al., 2025a), se utiliza de manera separada para cada experimento, teniendo en cuenta los cinco fragmentos o repeticiones que fueron empleados por cada método o estrategia (Rangel et al., 2024, 2025a, 2025b). Lo anterior, se realiza siendo extraído un *20%* de la *ME*, omitiendo el uso de esta *muestra de control,* al mismo tiempo que es empleado el restante 80% de la *ME*, para evaluar la estrategia o método de *cifrado* de datos. Al finalizar este proceso de *validación cruzada modificada*, se obtienen los resultados finales, que incluyen *promedio global* y *desviación estándar,* de cada experimento realizado (ver *Tabla 1*). Cabe aclarar que los tiempos de cifrado y descrifrado fueron medidos en milisegundos, por cada ejemplar en su ME correspondiente. Se inicia la medición del tiempo, antes de aplicar la estrategia de cifrado sobre el texto plano (columna Label) y se termina de medir, cuando se ha obtenido el par de textos cifrados (Test 1 y Test 2). Estos valores se almacenan en las columnas TC y TD, respectivamente. Posteriormente, se intenta descifrar ambos Test 1 y Test 2, para guardar el resultado en la columna Error, siendo obtenido como se ha descrito previamente. Una vez procesados mil ejemplares, se considera creada la ME, que corresponde a un método o estrategia de cifrado específico. Posteriormente, se aplica la validación cruzada modificada, con cinco iteraciones. En cada iteración, se guarda el promedio parcial de las columnas: TC, TD y Error. Al concluir con la quinta repetición se calcula la precisión del sistema, que en este caso, coincide con el promedio global de los promedios parciales, de cada iteración. Ello da lugar, al cálculo de la desviación estándar, que puede servir para evaluar hipótesis. Se observó que el aspecto de aleatoriedad de las alternativas de cifrado dinámico, puede afectar la velocidad del procesamiento. Es por ello, que ***en la Tabla 1, aparece el promedio de todas las iteraciones, después de concluir la validación cruzada modificada***.

Con respecto a la implementación del *cifrado de datos dinámico* utilizando la nueva propuesta: *noisy mutation AES-256*, se llevó a cabo empleando un procedimiento similar, al descrito previamente, incorporando la fusión de técnicas que primero hace uso de la aplicación del algoritmo estándar *AES-256* sobre el *texto plano*, combinando posteriormente con la aplicación del método *reduced random mutation* sobre este resultado de *cifrado estático*, proporcionado por *AES-256*. Esta combinación de estrategias, permite conseguir un resultado *dinámico*, a través del *camuflaje de textos*. Para facilitar la comparación de resultados, los experimentos realizados con esta fusión de estrategias, aquí denominada como: *noisy mutation AES-256*, también fueron sometidos al proceso de *validación*

cruzada modificada, usando cinco repeticiones con la *ME* diseñada específicamente para evaluar la nueva propuesta *noisy mutation AES-256*. Cabe recordar, que las *ME* en cada experimento deben ser preparadas de acuerdo con el método de cifrado a utilizar, debido a que, se incluyen dos ejemplares de *texto que han sido cifrados o camuflados* por la misma estrategia de encriptado que se está evaluando. Por último, se obtiene el global *promedio* (*global accuracy*) y *desviación estándar* de todos los experimentos *(*ver *Tabla 1).*

Por otra parte, en lo concerniente con los parámetros empleados para el cifrado de datos con el algoritmo: *AES-256*, se ha utilizado una *clave secreta (Key)* con longitud *(KeySize)* de 256 bits y un valor de "*00000000000000000000000000000001*", que corresponde al uso de 32 bytes. También, se utilizó el formato o modo de encadenamiento de bloque de cifrado *CBC: Cipher Block Chaining Mode* (Stinson & Paterson, 2019), con *OpenSSL* (Van & Jajodia, 2011; Van-Tilborg, 2005), como "*default_backend*", haciendo uso de un *vector de inicialización (IV)* con valor de "*0000000000000001*", empleando 128 bits, mediante el *estándar de criptografía de llave pública PKCS7: Public Key Cryptography Standard #7* (Van & Jajodia, 2011; Van-Tilborg, 2005), que fue usado como método de relleno *(padding)*. La salida del resultado *cifrado* se generó en formato *hexadecimal*. Este algoritmo: *AES-256*, fue implementado, también en lenguaje *Python 3* (Python.org, 2024), empleando el paquete o librería: *Cryptography* (Cryptography, 2025), haciendo uso del conjunto de métodos de la clase: *Cipher*, importando el componente: *algorithms.AES*.

De acuerdo con Rangel et al. (2025b), la aplicación del *método validación cruzada* descrito por Rangel (2002, 2022), puede ser empleada con una pequeña adaptación al procedimiento, para poder utilizarlo como método de estimación de error en tareas relacionadas con el cifrado de datos. Esta modificación, ya ha sido descrita en otras investigaciones (Rangel & Rangel, 2024a, 2024b, 2025a; Rangel et al., 2023, 2024, 2025a), que en términos generales consiste en lo siguiente: Para cada estrategia o método o algoritmo de *cifrado/descifrado* a emplear: *(1) Primero, se aplica el cifrado de datos, utilizando las cinco submuestras de entrenamiento, cada una de ellas, con 20% de tamaño en filas. (2) Una submuestra se extrae de manera secuencial y sin reemplazo. Posteriormente, se aplica el procedimiento de desencriptado al resto de las submuestras, debe corresponder al 80% de la ME original. (3) Este proceso es repetido cinco veces, siendo extraída en cada iteración, una submuestra diferente con 20% de tamaño. Después de procesar todas las submuestras con cada alternativa de cifrado/descrifrado, en forma separada, se calculan los promedios globales de los resultados obtenidos en cada iteración* (Rangel et al., 2024, 2025b). En la presente investigación, este procedimiento iterativo fue detenido cuando cinco repeticiones han sido finalizadas con cada estrategia *de cifrado de datos* empleada, lo cual, reitero, se ha llevado a cabo de manera separada en cada experimento.

2.4. ANÁLISIS DE LOS RESULTADOS

Los experimentos fueron realizados considerando dos categorías. Primero, la aplicación de estrategias sobre *textos cifrados* por el algoritmo *AES-256*, así como, el uso de métodos de cifrado sobre *texto plano*. Ello se llevó a cabo para poder comparar resultados entre las propuestas estándar y las basadas en *camuflaje de textos* e *inyección de ruido,* según sea el caso.

Cabe recordar que, cada experimento fue evaluado de manera separada, utilizando una *muestra de entrenamiento* (*ME*) con mil *patrones* (filas) almacenados y una dimensión de seis columnas (cinco atributos y una etiqueta de clase), siendo diseñada específicamente para cada estrategia a evaluar. Lo anterior, porque entre los *atributos* de la *ME*, se incluyen *dos* ejemplares *(Test 1 y Test 2)* de *texto cifrado* por la estrategia correspondiente, para efectos de observar si fueron generados resultados con *cifrado dinámico*, y de este modo, poder calcular la estimación de los tiempos de cifrado *(TC)* y descrifrado *(TD)* en *milisegundos,* así como, la *estimación del error*, utilizando como referencia la *etiqueta de clase* (*Label*), que en realidad es el mismo *texto plano* (S_i). En términos generales, los *textos planos* de las

muestras de entrenamiento, fueron generados en forma *aleatoria (con reemplazo)*. Aunque hubo algunos casos que fueron incorporados de manera supervisada, simulando ser "contraseñas o password", por la razón de que incluían caracteres "con ruido", cuyos valores ordinales estaban fuera del rango de la tabla *ASCII*. Un ejemplo de ello, es la secuencia: "▒Welcome▒□", cuyos resultados se observan en la *Tabla 2*. Esta situación fue proyectada de ese modo, con el propósito de poder observar presuntas vulnerabilidades que pudiera tener una secuencia de caracteres al momento de solicitar una *entrada de texto plano*, con longitud máxima de 255 caracteres. Por lo tanto, si el *texto cifrado*, al ser *desencriptado*, no coincide con la entrada de *texto plano*, es anotado como *error*. Para calcular el *error* y estimar la velocidad promedio del *cifrado/descrifrado* de datos, se ha utilizado una variante de *validación cruzada* con cinco repeticiones, la cual, fue propuesta en otras investigaciones (Rangel et al., 2023; Rangel & Rangel, 2024; Rangel et al., 2025a), y que ha sido descrita previamente, todo con el propósito de poder comparar resultados.

Con respecto al procedimiento de experimentación, se llevó a cabo como se describe enseguida: Primero, se realizaron pruebas con las *muestras de entrenamiento* de *texto plano*, utilizando el algoritmo estándar *AES-256,* así como, los métodos: *random Caesar II mod 120, reduced random Caesar mod 105* y *reduced random mutation* con *mod 105,* cada grupo de experimentos fue llevado a cabo en forma separada. Lo anterior, para obtener el *promedio* y la *desviación estándar* de los *tiempos* de cifrado/descifrado, en cada categoría, donde se observó la ausencia de errores durante el proceso de descifrado de datos (ver *Tabla 1*). Posteriormente, se aplicaron las estrategias: *random noisy AES-256, reduced noisy AES-256* y *noisy mutation AES-256, siendo* experimentadas, cada una de ellas de manera separada. Del mismo modo, fueron empleadas cinco repeticiones de *validación cruzada*, utilizando la misma variante de procedimiento, descrito previamente. Finalmente, se obtuvo el *promedio global* y *desviación estándar* de los *tiempos de cifrado/descrifrado,* con cada categoría de experimentos, donde se observó también la ausencia de errores en el proceso de descrifrado de datos.

Tabla 1: Resultados preliminares de los métodos y/o estrategias de cifrado dinámico aleatorio que incluyen la inyección de ruido y camuflaje de textos, según sea el caso, siendo seleccionada para obtener el tiempo promedio, un ejemplar o muestra con el texto plano: "▒Welcome▒□". Se presentan los tiempos promedio de dicha muestra seleccionada, así como, los tiempos del rango promedio mínimo y máximo (entre paréntesis) que corresponden al cálculo con la muestra de entrenamiento completa. Todo ello concerniente con el cifrado/descrifrado de datos medido en milisegundos, incluyendo el porcentaje de la estimación de error, considerando todos los ejemplares incorporados en la muestra de entrenamiento. Aparece señalado con ± , el cálculo de la desviación estándar para cada caso. La columna H es la entropía obtenida por el método de Shannon (1948).

Estrategia de cifrado de datos	**H: Entropía**	**TC: Tiempo promedio de cifrado en milisegundos ± Desviación estándar (Rango mínimo..máximo=promedio global de cifrado en milisegundos ± Desviación estándar)**	**TD: Tiempo promedio de descifrado en milisegundos ± Desviación estándar (Rango mínimo..máximo=promedio global de descifrado en milisegundos ± Desviación estándar)**	**Error: Porcentaje de error (Desviación estándar)**
Random Caesar II (mod 120)	6.9 .. 8 bits.	0.1461±0.0109 (0.1390±0.0072 .. 0.1563±0.0136 = 0.1461±0.0064)	**0.0518**±0.0012 (0.0537±0.0022 .. 0.0592±0.0002 = 0.0547±0.0027)	0 (0)
Reduced Random Caesar	6.7 bits	0.1491±0.0082 (0.1293±0.0098 .. 0.1349±0.0073 = *0.1366*±0.0075)	0.0559±0.0013 (0.0501±0.0021 .. 0.0544±0.0032 = *0.0535*±0.0021)	0 (0)
Reduced Random Mutation	**13.4 .. 2040 bits**	**0.1439**±0.0090 (0.1461±0.0047 .. 0.1523±0.0111 = 0.1472±0.0031)	0.0619±0.0013 (0.0596±0.0018 .. 0.0743±0.0017 = 0.0654±0.0056)	0 (0)
AES-256	127.9 bits	7.3117±3.7094	0.6166±0.01699	0 (0)

estándar		(0.8224±0.02845 .. 7.5745±3.9946 = 4.1463±3.2982)	(0.3838±0.01239 .. 0.7319±0.0572 = 0.6138±0.1403)	
Random Noisy AES-256	6.9 bits	8.4638±3.9351 (1.3502±0.0389 .. 8.1772±3.7392 = 4.8933±3.4297)	1.0022±0.0119 (0.9986±0.0306 .. 1.2122±0.0122 = 1.0999±0.0999)	0 (0)
Reduced Noisy AES-256	13.4 bits	**7.7470**±3.5379 (1.5055±0.0304 .. 8.2181±3.7822 = *4.7558*±3.2311)	1.1280±0.0181 (1.1925±0.0202 .. 1.2751±0.0295 = 1.2084±0.0549)	0 (0)
Noisy Mutation AES-256	6.7 .. 13.4 bits	8.7409±4.1845 (1.2589±0.0334 .. 9.2999±4.3854 = 5.1466±3.8789)	**0.9399**±0.01964 (0.9847±0.0308 .. 1.1793±0.0208 = *1.0350*±0.0899)	0 (0)
Promedio Global		**4.6718±3.9425 (0.7645±0.5753 .. 4.8162±4.07014 = 2.7674±2.2895)**	**0.5509±0.4513 (0.5319±0.4728 .. 0.6552±0.5384 = 0.5901±0.4914)**	**0 (0)**

En la *Tabla 1*, se observa que, en ninguno de los experimentos hubo presencia de error. Aclarando que dicha información incluyen secuencias de *texto plano* "ruidosas", como es el caso de la cadena de *texto plano*: "▓Welcome▓□". Este tipo de situaciones fueron controladas durante la etapa de experimentación por cada uno de los métodos, utilizando *módulos* con *105* o*120* caracteres en los alfabetos, según sea el caso, para evitar que fueran seleccionados valores fuera del rango de la tabla *ASCII* o *UTF-8*. En cambio, en la *Tabla 2* aparecen *dos* muestras (*Test 1* y *Test 2*) de resultados cifrados para cada estrategia evaluada. En la mayoría de los casos, se observan *resultados dinámicos*, excepto los cifrados obtenidos por el algoritmo estándar *AES-256*, cuando fueron aplicados exclusivamente sobre *texto plano*. Sin embargo, cabe aclarar que las secuencias de texto ahí presentadas, son solamente aproximaciones al resultado cifrado original, ya que, algunas de esas secuencias, incluían caracteres no imprimibles en pantalla y/o algunos elementos que estaban fuera del rango ordinal de la tabla *ASCII* o *UTF-8*, debido a que fueron incorporados "con ruido", de manera supervisada, desde la misma *entrada de texto plano*, dichos caracteres probablemente hayan cambiado o desaparecido, al momento de ser incorporados en el documento de formato *MS Word* (.docx), o también, al ser convertido este artículo a formato *PDF*, posiblemente varios caracteres se hayan perdido. Esta situación, nos hace resaltar, la confiabilidad de las estrategias basadas en *inyección de ruido* y/o *camuflaje de textos*, porque solamente tienen *integridad* utilizando el *archivo original,* que en la presente investigación, fueron generados utilizando el lenguaje de programación *Python 3*.

Tabla 2: Algunos ejemplares de muestra seleccionados de los textos cifrados por las alternativas de cifrado dinámico basadas en inyección de ruido: random noisy AES-256, así como, camuflaje de textos usando: reduced noisy AES-256 y noisy mutation AES-256 (cada secuencia corresponde a un solo experimento, seleccionado aleatoriamente, cuando ha sido utilizado como texto plano: "▓Welcome▓□"). Se presentan dos test de prueba. En todos los casos se obtuvo cifrado dinámico con las propuestas: random noisy, noisy mutation y reduced noisy. En cambio, el estándar AES-256, ha presentado cifrados estáticos en todos los casos, aunque esta situación no significa que sea vulnerable en todas las situaciones. El tamaño del texto cifrado es la longitud de caracteres (en promedio) que contiene la secuencia de texto encriptada, considerando que algunos caracteres se han perdido por no ser imprimibles en pantalla.

Estrategia de cifrado de datos	**Número de muestra**	**Tamaño del texto cifrado**	**Texto cifrado (muestra seleccionada)**
Random Caesar II (mod 120)	Test #1	30	★r★¯X¯ÈcÈì ³³P³ó óÊ]ÊÎìÎ◊7◊□□□qgq
	Test #2	26	◘U◘ ? ô ô«?«ÄaÄ) ¦A¦◕B◕□q□ZPZ
Reduced Random Caesar	Test #1	22	◄1«TÎìÕìÌìØìÖìÌg◈5□i9/
	Test #2	19	◙F IÎì¥9 6) -Îì◘X□iJ@
Reduced	Test #1	21	▸ &iÀÎigÓ -3¢ÖiiÎ⌒Mi□si

Random Mutation	Test #2	21	◒?iÀÍhdÐÌi. ÖiR·◨U:□_U
AES-256 estándar	Test #1	96	30303030303030303030303030303030319421780fc40c59f14796a598115d8e5139f6f9666f67cb30fcda01f22739eeda
	Test #2	96	30303030303030303030303030303030319421780fc40c59f14796a598115d8e5139f6f9666f67cb30fcda01f22739eeda
Random Noisy AES-256	Test #1	218	©v©Z*Z¯⌈-}-›h›[+[a.a˜h˜tAt»‹» j šjšj7j»‹»² ²k;k,O,i9iZ'Z[+[Z —g— M Q!Qf3fuEu' 'yIy³ ³ªV†g4gf5fr9r¾Š¾¢p¢V%V°ƒº^&^Ã"Ãë…ë–3– »‡»¢r¢ÜyÜ](]ÃŠÃ * Â'Â³ ³j3j²y²m7m⁸I⁸t?t¬s¬£k£‹Z‹,Q,²}²Œ(ŒÍ•ÍÊeÊ}H} ^ uBux?xÐjÐt>w>wc-cÁ‹ÁÁk5k†"^" I ã ãí‹ír?r~N~ÙsÙ"0"‹×s×Œ+Œ-}- . ¢<¢U#U¼Š¼ g [([¬s¬†!†é„éê†ê¬K¬
	Test #2	209	£p£Z*Z°}°|v|Z'ZyIy[([¾ ¾|I|R"R] µ…µ¶ƒ¶℗SSh8h«x«¯ ¯³ ³¿ ¿È•ÈPP»ˆ»`0`¿Œ¿PPq>q»‹»a.am=m]*]¨w¨^%^¨t¨"b"†U†b+b©q©½ ½Š$Š¯E¨l8lšjšØuØ‰T‰—^— -G-˜W^r>r®w®["["[\"¨G¨^)^a(a I rAr¥t¥TT"0"†N†Š%Š‡R‡šiš•b•a(a††©s©ÍgÍ®u®ŸiŸÁ‹Ád.d³M³Æ Æ Y ¡>¡Þ|Þf3f†V† (ˆ %ˆâ ãíŒi–f–„S„Ÿ9Ÿ¯V¯¥s¥ f a.a¥I¥ÖqÖá|á•1•Ü{Û
Reduced Noisy AES-256	Test #1	141	]* Q™is@]^+™i~Kj:œi™if3,RwD™i`]l9_/›hV&]*g7d1™is@™im:™iœi…T H ixFšiœe¡ivFÏìÌi i ^Ìi i¢ìÏi]ƒO{D¢iŸi«J iœc¡iši—f ZÍiV¥@ išiœi¢i (ŸiÀZ Va+Ÿi hÏiŸil5¦CÊi N™i¿Ÿli¯KÂac3ši 'ŠX›i i{H¢i„®IÉeÊi
	Test #2	147	c0{Kg4,R\)yIœi Pœi™i[(R"œi–f^‰Yœi]œi™ixE™iœi™i^+_/œih8uB™i|I`/k2j6zHši ik3]Ïi^%{GN³P'\ƒJÏi‡V— c i†MšdÊiq<¢i¡ii8U$ iÍi¡ÌÌi išiT!s:".Ÿiœ6¢it>™cŸiÏi•_ i³P'/–cT$ÏìÌi *¯N"cf5Éc›i'_yBn;˜_‡"Îi¥AÉh
Noisy Mutation AES-256	Test #1	148	‚OZŠ}Ji™j7&Vœi4dX%Nœii™|IT„– cR‚œi8hd1i™ˆUi™R$TœiXˆœi#Sj7i™œi>oZ!\ šhiš i)aU%;¡ˆ%i U%E¨wBH ¡;c"d0i ¢iiŸÊii _&6nl;O LíÍc+iÎ Y,]wD0i— 1%[Š$V ŸiAw hiÏŸii À]iË^++[Â\ìÍÍi:›"cS„¾X1c›i'^ŒYi¢ÍihÍËg<
	Test #2	153	]*>nœiPœiK{šgb'g44dœiJz'^:kœi8hœi:jxEi™ _,N'_i™œiEu'^R.„QHxœiDu¢i2fi7f— ´i6n™i.”½Zi a1 i*cÏi:k iQ˜s:iŸ»Zi ¢ii¡c2#Te0eÉ•]G¬ ia'ŠWG »U4jœ6i¢k5iŸ]'OµŸii ¦CfÈd10`ÇaF©ÍiiÊ `*[Ïi1cW%R‰ vC/hÏiiÎ¢>iÊ

Sin embargo, las estrategias basadas en inyección de ruido y camuflaje de textos, no han sido aún evaluadas sus vulnerabilidades, por lo tanto, no puede especularse el grado de resistencia a ataques. En nuestros experimentos, se detectó también ambigüedad en la parte inicial del cifrado de datos, lo cual, dificulta la operación del descrifrado, que corresponde a la evaluación del texto plano usando: $C_i = S_i + K_i$, así como, para texto cifrado: $C_i' = E_i + K_i$. Dónde: S_i es el texto plano, mientras que E_i , es el texto cifrado por AES-256 y K_i implica el desplazamiento, considerando que guarda valores diferentes en cada posición, siendo seleccionados aleatoriamente usando mod 120 o mod 105, según la estrategia de cifrado empleada. Teniendo en cuenta que el alfabeto de cifrado/descrifrado con mod N, es seleccionado anticipadamente, mediante una fase de aprendizaje usando inteligencia artificial, del modo que ha sido descrito en otras investigaciones (Rangel et al., 2023, 2024, 2025a, 2025d). Por ende, tanto C_i como C_i', pueden tener ambigüedad, es decir, dependen de su vector K_i, ya que, dos textos planos distintos, pueden tener el mismo C_i o C_i '. Por ejemplo, S_i-1 = "123" con K_i-1 = [1, 2, 1], generan un C_i-1 = "244", suponiendo que el alfabeto contiene: ABCD = [1, 2, 3, 4, 5, .. mod N]. Del mismo modo, S_i-2 = "231", con K_i-2 = [0, 1, 3], generará el mismo C_i-2 = "244". Sin embargo, al cambiar el orden en el alfabeto, por ejemplo: ABCD = [1, 5, 4, 2, 3, .. mod N], en este caso, se obtendrían otros resultados. Esa es la razón, por la cuál, los métodos: random Caesar, reduced random Caesar y reduced random mutation, requieren de una segunda fase, para incluir en un FinalPackage, la información que permita el descrifrado de datos. Por lo anterior, en la presente investigación no se ha estudiado el aspecto de ambigüedad en el cifrado de datos, debido a que, requiere de un conjunto exhaustivo de experimentos y/o explicaciones, que podrían ser consideradas para un trabajo futuro. En estos términos, el ataque del diccionario (Alkhwaja, 2023) y el ataque de fuerza bruta (Bošnjak et al., 2018), podrían ser inadmisibles por la concurrencia de la ambigüedad.

Otro factor que garantiza la confianza en las estrategias de cifrado dinámico aquí propuestas, es el valor de la entropía real (Shannon, 1948), que aparece en la columna H de la Tabla 1. La entropía de salida (H) de cada algoritmo, se refiere a la cantidad de incertidumbre o aleatoriedad en los datos cifrados. Empleando el método de Shannon (1948), puede ser obtenida del siguiente modo: $H = -\sum p(x) \log 2\, p(x)$. Donde: H es la entropía, p(x) es la probabilidad de cada valor posible x, esta suma se realiza sobre todos los valores posibles. Por ejemplo, AES-256 es un algoritmo de cifrado simétrico que produce un texto cifrado de 128 bits (16 bytes) para cada bloque de entrada de 128 bits (16 bytes). Se estima que la entropía real es de aproximadamente 127.9 bits por bloque de 128 bits. En este caso, la entropía del texto cifrado AES-256 es muy alta, lo que significa que es muy difícil predecir el contenido del texto cifrado sin la clave de cifrado. Por lo tanto, el método random Caesar con mod 255, usando un texto plano con maximo 255 caracteres, equivale a: $H(C_i) = \log2(256) = 8$ bits. Sin embargo, hay un problema: el rango de K_i es de 0 a 105, lo que significa que el desplazamiento: $C_i = S_i + Ki$ no cubre todo el rango de 0 a 255. En realidad, C_i solo puede tomar valores de 0 a 255 + 105 = 360, lo que se reduce a 0 a 104 (módulo 255, que en

realidad son 256 valores, porque es incluido el 0). La entropía real de C_i será menor que 8 bits, ya que no se cubre todo el rango de posibles valores. Para calcular la entropía real, debemos considerar la distribución de probabilidad de Ci, que no es uniforme. Una aproximación es calcular la entropía utilizando la fórmula de Shannon: $H(C_i) \approx - \sum p(x) \log2\ p(x)$. Donde p(x) es la probabilidad de cada valor posible x de C_i o C_i '. Por ende, podemos deducir que la entropía de C_i será menor que 8 bits, probablemente alrededor de 7-7.5 bits, debido al rango limitado de K_i. Sin embargo, si el texto plano S_i contiene caracteres ASCII con ordinales desde 0 a 255, entonces la distribución de probabilidad de S_i es uniforme sobre 256 posibles valores. En ese caso, el desplazamiento de $C_i = S_i + K_i$, donde K_i es un valor aleatorio con reemplazo dentro del rango de 0 a 105, producirá una distribución de probabilidad de C_i que es la convolución de las distribuciones de S_i y K_i. La entropía de C_i se puede calcular como: $H(C_i) = H(S_i + K_i)$. Debido a que S_i y K_i son independientes, la entropía se puede calcular como: $H(C_i) = H(S_i) + H(K_i) - I(S_i; K_i)$. Donde: $I(S_i; K_i)$ es la información mutua entre S_i y K_i. Teniendo en cuenta que: $H(S_i) = \log2(256) = 8$ bits, ya que S_i es uniforme sobre 256 posibles valores. Del mismo modo: $H(K_i) = \log2(106) \approx 6.7$ bits, ya que K_i es uniforme sobre 106 posibles valores (porque también incluye el 0). Por lo tanto, la entropía de C_i es: $H(C_i) = 8$ bits + 6.7 bits - 0 = 8 bits, lo que significa que el texto cifrado C_i tiene una entropía máxima y es difícil de predecir sin la clave K_i. Sin embargo, es importante destacar que la seguridad del algoritmo no solo depende de la entropía del texto cifrado, sino también de la seguridad de la clave K_i y del proceso de cifrado en general. De manera similar, para el caso del random Caesar II con mod 120, el texto cifrado C_i se calcula como: $C_i = (S_i + K_i) \bmod 120$. Por ende, la distribución de probabilidad de C_i dependerá de la distribución de S_i y K_i. Entonces, cuando S_i es uniforme sobre 256 posibles valores, la distribución de C_i será uniforme sobre 120 posibles valores (ya que 256 > 120). Además, K_i es uniforme sobre 106 posibles valores, pero solo 105 de estos valores producirán un resultado diferente en C_i (ya que 106 > 120). Por lo tanto, la entropía de C_i se puede calcular como: $H(Ci) = \log2(120) \approx 6.9$ bits. Por otra parte, en caso de reduced random Caesar, el texto cifrado C_i se calcula como: $C_i = (S_i + K_i) \bmod 105$. Donde la distribución de probabilidad de C_i dependerá de la distribución de S_i y K_i. Por lo tanto, si S_i es uniforme sobre 256 posibles valores, entonces la distribución de C_i no será uniforme sobre 105 posibles valores (ya que 256 > 105), mientras K_i es uniforme sobre 106 posibles valores, pero solo 105 de estos valores producirán un resultado diferente en C_i (ya que 106 > 105). En este caso, la entropía de C_i se puede calcular como: $H(C_i) = \log2(105) \approx 6.7$ bits. Esta entropía es menor que en el caso original (8 bits) debido a la reducción del rango de posibles valores de C_i. Por último, el método: reduced random mutation, que también utiliza un mod 105, pero además se agrega el aspecto de la mutación por intercambio, suponiendo que se aplica solamente a los conjuntos pares del siguiente modo: $C_i = (C_1 + K_1), (K_2 + C_2), (C_3 + K_3), ..., (C_n + K_n)$. Donde n es el número de pares de números. En esta situación, la entropía del texto cifrado Ci dependerá de la distribución de probabilidad de los valores de K_i y S_i. Si los valores de K_i y S_i son uniformemente distribuidos en el rango [0, 255], entonces la entropía de cada par de números (C_i, K_i) es de 8 bits (porque existen 256 posibles valores para cada número). Sin embargo, debido a la mutación en cada par de números, la entropía total del texto cifrado C_i no es simplemente la suma de las entropías de cada par de números. Para calcular la entropía del texto cifrado Ci, podemos utilizar la misma fórmula de la entropía de Shannon: $H(C_i) = - \sum p(x) \log2\ p(x)$. Debido a la mutación en cada par de números, la probabilidad de cada valor posible x de C_i es uniforme en el rango [0, 255], por lo que la entropía de cada par de números es de 8 bits. La entropía total del texto cifrado C_i es la suma de las entropías de cada par de números, que es: $H(C_i) = 8$ bits x 255 = 2040 bits. En este caso, la entropía del texto cifrado C_i es de 2040 bits, lo que indica que el texto cifrado es muy seguro y difícil de descifrar. Es importante destacar que la seguridad del texto cifrado C_i depende no solo de la entropía, sino también de la complejidad del algoritmo de cifrado y de la calidad de la clave K_i. En estos términos, las estrategias: random noisy AES-256, reduced noisy AES-256 y noisy mutation AES-256, no solamente ***tienen la complejidad de su método predecesor, sino que también, se le suma la entropía del algoritmo AES-256.***

En lo concerniente con los tiempos del procesamiento de *cifrado* de datos, basados en alternativas *AES*, se observó que son más rápidos cuando se aplica al *texto plano,* utilizando la versión estándar del algoritmo *AES-256*, pero la diferencia no es muy significativa, ya que, obtiene una convergencia en promedio entre 0.45 y 1.43 milisegundos de diferencia. Sin embargo, en relación a los métodos que fueron aplicados a *texto cifrado* basado en *AES-256*, el *mejor balance* fue obtenido con la estrategia: *reduced noisy AES-256*, que demostró ser entre 1.03 y 1.13 veces más rápida, con una diferencia entre 0.28 y 0.99 milisegundos, lo cual, se considera no es muy sustancial, si se

aplica en dominios prácticos donde se utilicen contraseñas o textos planos menores a 255 caracteres. Empero, en términos generales, el *mejor balance* fue obtenido cuando se aplicó *reduced random mutation* sobre *texto plano*, resultando ser entre 1.02 y 60.74 veces más rápido que el resto de las estrategias aquí evaluadas.

En este mismo contexto, aunque la mayoría de los experimentos fueron exitosos, se reconoce la necesidad de profundizar más en la experimentación, debido que, solamente se utilizaron cadenas de *texto plano* con longitudes máximas de 255 caracteres, aunque los *textos cifrados* en algunos casos rebasaron esta limitante (ver *Tabla 2)*. En consecuencia, se pudo cumplir con la *hipótesis*, ya que, la nueva propuesta: *noisy mutation AES-256,* permite mejorar el esquema de seguridad del denominado algoritmo *AES-256*, mediante el *camuflaje de textos* para convertirlo en un método de *cifrado dinámico*, siendo difícil descifrar, debido a que, puede llegar a confundir a los *ciber-delincuentes*. Está situación también puede ser corroborada en los resultados presentados en la *Tabla 2*. Además, en la *Tabla 1*, podemos apreciar que la nueva propuesta: *noisy mutation AES-256,* aunque no demostró ser más rápida en el *cifrado* de datos, al ser comparada con el resto de las estrategias aquí evaluadas, se observó que la diferencia no es muy sustancial, ya que, estriba entre 0.99 y 0.23 milisegundos de diferencia. Sin embargo, *noisy mutation* resultó ser más rápida en el *descrifrado* de datos entre 1.06 y 1.20 veces mayor que el resto de las técnicas o métodos aquí experimentados, lo que deduce también, que la diferencia no es muy significativa.

Por otra parte, de acuerdo con Rangel et al. (2024, 2025b): « *en el contexto que concierne con la inyección de ruido, a pesar que dicha situación puede ser difícilmente controlada, se debe ser cuidadoso en la selección de caracteres que formarán parte del vector K_i, debido a que, si se incorporan elementos que estén fuera del rango de la tabla ASCII, puede producir errores en el descrifrado de la información. Por ejemplo, el uso del caracter '▓' que se supone su correspondiente ASCII es 178 (pero al ser convertido a entero u ordinal regresa el valor: 9619). Otro caracter, que se ha utilizado, con el propósito de experimentar "ruido" en el cifrado/descifrado de datos, es el caso de: '□'. Dicho caracter, fue extraído del nombre de un documento que en sistema operativo Android no reconoce al migrar datos de Linux a Windows sobre la plataforma Android. Al parecer, ese caracter no tiene un correspondiente en la tabla ASCII. Esta información se pudo obtener, porque se realizaron pruebas en lenguaje de programación Python, donde se obtuvo el valor entero u ordinal: 65533. El código fuente utilizado para dicho propósito fue el siguiente: c1 = '▓' ; c2 = '□' ; print("->" + c1 + " = " + str(ord(c1))) ; print("->" + c2 + " = " + str(ord(c2))) ».* A pesar que dicha situación se presentó en esta investigación, no se observó ningún problema en alguna de las estrategias aquí evaluadas.

Finalmente, la evaluación de resultados mediante el uso de la *validación cruzada modificada* (Rangel & Rangel, 2024a; Rangel et al., 2025a), en este estudio proporciona información valiosa para determinar en qué situaciones la *inyección de ruido* y/o *camuflaje de textos*, presenta una alternativa segura y recomendable para las organizaciones.

3. DISCUSION

En recientes investigaciones (Rangel & Rangel, 2024a; Rangel et al., 2024, 2025d; Rangel et al., en revisión 2025c), se ha referido que en dominios prácticos el *robo digital de datos* puede causar pérdidas financieras significativas a las organizaciones. Mismos estudios sugieren el uso de *métodos de encriptación de datos* como alternativa efectiva para amortiguar un poco este problema. Empero, uno de los principales desafíos de estas *estrategias de ciberseguridad,* a saber, consiste en que comúnmente se basan en *algoritmos estándar* de cifrado, que en su mayoría, suelen producir *resultados estáticos*. Está situación, puede generar incertidumbre, debido a que, para una misma entrada de *texto plano*, usando parámetros idénticos, siempre se obtiene la misma salida cifrada. Aunque ello no implica que dicha secuencia encriptada resulte vulnerable a *ciberataques* en todos los casos. Sin embargo, algunos autores (Rangel & Rangel, 2024b, 2025a; Rangel et al., 2023, 2024, 2025b), consideran que esta situación puede comprometer la seguridad de los datos en las organizaciones, debido a que, si una misma secuencia de texto cifrado corresponde a una misma cadena de texto plano, se podrían almacenar los

resultados en un *diccionario* (Alkhwaja, 2023), siendo entrenado con *inteligencia artificial*, lo cual, en un corto periodo de tiempo, podría ser capaz de resolver secuencias cifradas o contraseñas de al menos 16 caracteres de longitud. Esta situación, de los *cifrados estáticos*, también puede facilitar su descifrado a otros tipos de algoritmos, por ejemplo, el uso de *fuerza bruta* (Bošnjak et al., 2018), o incluso, algoritmos basados en *computación cuántica* (Baklaga, 2024; Bavdekar et al., 2023; Iavich et al., 2024), como es el caso de: *Kyber* (Iavich et al., 2024; Rangel et al., 2025a) y *Shor* (Bavdekar et al., 2023; Rangel et al., 2025d). En la presente investigación se exponen algunas alternativas *dinámicas* para el cifrado de datos, con el propósito de poder abordar este problema, teniendo en cuenta que, Rangel & Rangel (2024a) refieren que el uso de este tipo de esquemas basados en *procesos de mutación*, constituyen un buen indicador para incrementar la seguridad de los datos digitales en las organizaciones. Lo anterior, debido a que, se observó que los resultados basados en *cifrados dinámicos aleatorios*, pueden producir siempre, secuencias cifradas distintas, incluso empleando la misma entrada de texto y usando los mismos parámetros, situación que según Rangel & Rangel (2024a, 2024b, 2025a, en revisión 2025), esta alternativa se considera prometedora para proteger la información, ya que, puede confundir a los ciber-criminales.

En este contexto, existen propuestas que sugieren el uso de *inyección de ruido* y/o *camuflaje de textos*, siendo aplicado sobre *texto cifrado* por algoritmos estándar, presentando al menos dos alternativas: *random noisy* y *reduced noisy* (Rangel et al., 2025a), respectivamente. Empero, el uso del *cifrado de datos dinámico* basado en *procesos de mutación*, conocido como: *reduced random mutation* (Rangel & Rangel, 2024a), no ha sido estudiado con textos cifrados, solamente se han publicado experimentos basados en *texto plano*, sugiriendo resultados muy prometedores. La presente investigación, es continuidad de dicho trabajo, porque se observó que las estrategias basadas en: *procesos de mutación (reduced random mutation)*, no han sido experimentadas en combinación con alguno de los *algoritmos de cifrado estándar*, o al menos no se han incluido pruebas sobre en el uso ese tipo de *camuflaje de textos* empleando secuencias cifradas por el algoritmo: *AES-256 (Advanced Encryption Standard 256)*. Esta situación no se considera buena para las organizaciones, particularmente aquellas que han optado por el uso de *AES-256* (Barranco & Galindo, 2022; Daemen & Rijmen, 2002; NIST, 2001; Rahman & Hossain, 2021; Rangel & Rangel, 2025a; Rodríguez, 2020; Van-Tilborg, 2005). Ello ha dado lugar al diseño de la nueva propuesta, aquí denominada como: *noisy mutation AES-256*, la cual, combina los procesos basados en *mutación* aplicados sobre textos cifrados por el algoritmo *AES-256*. Lo anterior, contribuye a responder la *pregunta de investigación* que sienta sus bases en la hipótesis, de que: *¿Es posible mejorar el esquema de seguridad del denominado algoritmo AES-256, mediante el camuflaje de textos para convertirlo en un método de cifrado dinámico, que sea difícil descifrar para los ciber-delincuentes, e incluso, retardar un poco el descrifrado del procesamiento de algoritmos basados en computación cuántica?*. Al respecto, en la presente investigación, aunque no se realizaron pruebas con *computación cuántica*, se logró cumplir con el propósito de obtener resultados de *cifrado dinámico (ver Tabla 2)*.

Del mismo modo, en la *Tabla 1*, podemos observar que la nueva propuesta: *noisy mutation AES-256 con módulo 105,* logró alcanzar buena convergencia, mostrando un valor de *0* en porcentaje de error, y a pesar que dicha alternativa no es más rápida en el cifrado de datos, al ser comparada con el resto de las propuestas aquí evaluadas, la diferencia en velocidad no es muy significativa. Empero, se reivindica en los procesos de descrifrado de datos, donde demuestra un comportamiento más rápido, comparado con las estrategias: *random noisy* y *reduced noisy,* aunque la diferencia en velocidades tampoco es muy sustancial. Sin embargo, lo anterior, permite dar aportación empírica a favor de los *métodos de cifrado de datos dinámicos*, que hacen uso de las estrategias basadas en *camuflaje de textos,* alternativas que han sido poco estudiadas en la literatura. Además, para poder facilitar la comparación de resultados con otras investigaciones (Rangel & Rangel, 2024a,.2025a; Rangel et al., 2025a), el presente estudio incluye el uso de una adaptación del método de validación cruzada (Rangel et al., 2023; Rangel & Rangel, 2024a, 2025a), que ha sido utilizado para estimar el error en el *cifrado de datos dinámico*, ya que, permite un análisis crítico que incluye situaciones no exploradas, tal como se señala por Rangel & Rangel (2025a), del siguiente modo: "*Por lo tanto, aplicar el método sin contemplar un grupo de cada clase, asegura resultados con sesgo positivo u optimista. Es decir, garantiza que el resultado en una situación real, pueda ser mejor o igual, que el valor reportado por el método de estimación de error*".

En lo concerniente con las ventajas y desventajas de la metodología aquí empleada, cabe destacar que el algoritmo *AES-256*, en su versión estándar, ofrece mayor velocidad en el cifrado y descifrado de datos, pero tiene la limitación de producir resultados de *cifrado estáticos*. Aunque ya se ha comentado previamente, que esta situación no implica vulnerabilidad en todos los casos. En contraste con la nueva propuesta basada en: *noisy mutation AES-256*, a pesar de comportarse más lenta que el estándar *AES-256*, ofrece resultados dinámicos en todos los casos *(ver Tabla 2)*. Otro aspecto a considerar, es que el uso de: *random noisy* y *reduced noisy*, en la presente investigación se obsevaron más rápidos sus procesos de cifrado, comparados con: *noisy mutation*, pero si consideramos que dicha diferencia no es muy sustancial, y tenemos en cuenta que *reduced random mutation* provee una capa adicional de seguridad, que es precisamente: el *proceso de mutación*, de este modo, podemos apreciar que un *FinalPackage* obtenido por *reduced noisy*, es prácticamente el mismo archivo que un *FinalPackage* producido por *noisy mutation*, excepto que en este último caso, se incluye *mutación*, situación que puede ayudar a confundir a los *ciber-delincuentes*, logrando de este modo alcanzar los propósitos de la presente investigación, debido a que, se puede incrementar la seguridad de los datos, al inyectar *camuflaje de textos* en los paquetes cifrados.

Por otra parte, de acuerdo con Rangel & Rangel (2024a), destacan que su propuesta basada en "mutación", muestra que el *camuflaje de textos* es efectivo para el *cifrado de datos dinámico*. Señalando que, aunque la reducción del tiempo de ejecución en el cifrado y descifrado podría facilitar el trabajo de los *ciber-delincuentes*, entonces necesitarían "adivinar" prácticamente todos los valores de desplazamiento aleatorio: K_i, que también han sido *camuflados*. Por ende, esas tareas de descubrimiento de dichos valores podrían ser muy difíciles. Por lo tanto, el uso de un *diccionario* sería inadmisible, ya que, cada vez que se cifra un *texto plano* usando *noisy mutation*, se obtienen resultados distintos, y en algunos casos, ocurre *ambigüedad* en el vector C_i, dado que sorpresivamente puede resultar ser el mismo valor cifrado (C_i) para dos secuencias distintas de texto plano (S_i), cifradas separadamente. Esta situación no afectó a los resultados de la presente investigación, porque se observó que el vector K_i obtenía siempre distintos valores en cada ejecución del algoritmo, y al ser guardado en el *FinalPackage*, se obtiene un paquete cifrado distinto en todos los casos (ver *Tabla 2*). En este mismo contexto, el uso de la *fuerza bruta* y algoritmos basados en *computación cuántica*, podrían demorar mucho tiempo, dado que la *ambigüedad* detectada en el proceso de cifrado de datos, cuando fue aplicada la variante de extraer el vector K_i del *FinalPackage*, para ser guardado posteriormente como si se tratara de una *clave pública* o *privada*, simulando un proceso equivalente a los *algoritmos de cifrado asimétricos*. Aclarando, que no se realizaron este tipo de pruebas, porque implica una gran variedad de experimentos, lo cual, podría considerarse en un trabajo futuro. Solamente se observó que, realizar el proceso de extracción del vector K_i del *FinalPackage*, separando los archivos, expone a la concurrencia de *ambigüedad*, lo cual, puede hacer más difícil la tarea de descubrimiento, dificultando más el proceso de descifrado por parte de los *ciber-delincuentes*.

4. CONCLUSIONES Y/O TRABAJOS FUTUROS

La *encriptación de datos,* como estrategia de *ciberseguridad*, constituye una buena opción para proteger la información en las organizaciones frente a *ciberataques*. Empero, los algoritmos de *cifrado simétrico estándar* suelen generar resultados estáticos, a pesar que su velocidad de procesamiento es superior, en comparación con algunas alternativas *dinámicas*. Sin embargo, optar por alternativas estáticas de *cifrado de datos* no es recomendable para las organizaciones, debido a que, dichos esquemas pueden generar resultados predecibles y comprometer la seguridad de la información, convirtiéndose vulnerables a ataques *cibernéticos*. Esta situación, ya ha sido revisada en otras investigaciones (Rangel et al., 2023, 2024; Rangel & Rangel, 2024b), que en términos generales, refieren que si un lenguaje de programación permite el cifrado de datos utilizando *algoritmos estándar*, solo basta con entrenar un *modelo supervisado* (Rangel, 2002, 2022), durante un período prolongado, para poder construir un *diccionario* (Bošnjak et al. 2018, Rangel et al., 2025d), basado en *inteligencia artificial,* con capacidad de descifrar textos o claves de al menos 16 caracteres. Por lo tanto, los métodos de *cifrado dinámico* (Rangel et al., 2025a, 2025d; Rangel & Rangel, 2024a, en revisión 2025), presentan una buena alternativa en solución de este tipo de problemas, especialmente los basados en *métodos aleatorios* (Rangel et al., 2025a), ya

que, han sido presentados en la literatura, como alternativas seguras, debido a que generan una gran variedad de resultados distintos, incluso con una misma entrada de *texto plano*, tal como se muestra en la *Tabla 2*. El uso de este tipo de estrategias, puede reducir la vulnerabilidad a *ataques cibernéticos* y podría prevenirnos para futuros ataques basados en *computación cuántica* (Baklaga 2024; Bavdekar et al., 2023; Fuegner 2024; Iavich et al. 2024; Rangel et al., 2025a, en revisión 2025b, 2025d; Sengupta and Ghosh 2023; Thakur and Kumar 2011). Lo anterior, se relaciona con la presente investigación, debido a que, está enfocada en la evaluación de variantes para el *cifrado dinámico,* con el propósito de mejorar la seguridad de la información en las organizaciones. En este contexto, debido a que, fue posible realizar de manera satisfactoria, una adaptación al procedimiento estándar del algoritmo *AES-256*, siendo combinado con estrategias del tipo: *reduced random mutation* (Rangel & Rangel, 2024a), muy similares a las experimentadas en otras investigaciones (Rangel et al., 2025a; Rangel & Rangel, 2024a), dando lugar al diseño de la nueva estrategia, aquí presentada como: *noisy mutation AES-256,* con *módulo* de *105* caracteres en alfabeto, utilizando valores de la tabla *ASCII* y/o *UTF-8*. Esta nueva propuesta se recomienda como una estrategia alternativa para el *cifrado de datos dinámico*, ya que, durante la experimentación se observó que no presentaron errores, alcanzando una convergencia aceptable durante la *etapa de aprendizaje* (Rangel, 2002, 2022; Rangel et al., 2023, 2024, 2025a), basada en *procesos de mutación* (Rangel & Rangel, 2024a; Rangel et al., 2024, 2025a) con *IA*, produciendo resultados cifrados diferentes en cada ejecución, incluso empleando la misma secuencia de *texto plano (ver Tabla 2)*, situación que se presenta como alternativa para confundir a los *ciber-delincuentes*, logrando de este modo cumplir con la *hipótesis* de la presente investigación.

En resumen, de acuerdo con lo observado durante la experimentación (ver *Tabla 1)*, y con el propósito de presentar un *balance final* de la investigación, se considera que la nueva propuesta *noisy mutation AES-256* con *módulo 105*, es entre 1.06 y 1.20 veces más rápida en el *descrifrado de datos* comparado con el resto de las estrategias basadas en AES-256, aquí evaluadas, con una diferencia promedio entre 0.06 y 0.19 milisegundos. Sin embargo, a pesar que para el *cifrado de datos*, la propuesta *noisy mutation AES-256,* se observó más lenta que las estrategias: *random noisy* y *reduced noisy*, su ventaja tampoco es muy sustancial, siendo obtenido en los experimentos una diferencia entre 0.23 y 0.99 milisegundos en promedio. Por ende, se propone el uso del esquema *noisy mutation*, debido a que, utiliza procesos basados en *mutación* (Rangel & Rangel, 2024a), estrategia bien conocida en *algoritmos genéticos* (Rangel & Rangel, en revisión 2024; Rangel et al., 2023, 2024, 2025a, 2025d; Reddaiah, 2016, 2019), que tiene la capacidad de confundir a los *ciber-criminales*, porque un *FinalPackage* (Rangel et al., 2025a) generado por estrategias del tipo: *reduced noisy* (Rangel & Rangel, 2024a; Rangel et al., 2025a), es prácticamente del mismo tamaño que los generados por: *noisy mutation* (ver *Tabla 2)*, dando lugar no solamente a confundir a los *ciber-delincuentes*, sino que además, puede retrasar presuntos ataques de *algoritmos basados en computación cuántica*. Empero, se considera importante aclarar, que los resultados fueron obtenidos empleando secuencias de *texto plano* con longitud máxima de 255 caracteres. Por lo tanto, se sugiere realizar pruebas adicionales con archivos en diferentes formatos, lo cual, implica realizar ajustes al procedimiento empleado, situación que podría ser considerada como objeto de futuras investigaciones.

Con respecto a las dificultades del cifrado de datos dinámico, se tuvo que hacer frente a problemas relacionados con la aleatóriedad, porque en ocasiones alguno de los experimentos generaba valores fuera del rango de la tabla *ASCII*, o en su defecto, fueron seleccionados en el alfabeto, algunos caracteres *no imprimibles*, lo que pudo haber resultado en pérdida de información y/o errores durante el *descifrado*, lo cual, representa una desventaja significativa. Sin embargo, este tipo de situaciones pudieron evitarse usando *módulos 120* para estrategias del tipo *random noisy*, así como, empleando un *módulo 105* en los *alfabetos aleatorios*, al momento de utilizar los métodos: *reduced noisy* y *noisy mutation*. Finalmente, al comparar nuestros resultados con otras investigaciones (Rangel et al. 2025a, 2025d; Rangel & Rangel, 2024a, 2024b, en revisión 2024), se pudo observar que existen varias alternativas para *inyección de ruido* y/o *camuflaje de textos*, propuestas basadas en variantes de *algoritmos genéticos* y/o haciendo uso de la *regla 1-NN* (Cover & Hart, 1967; Rangel, 2002, 2022, *2024*; Rangel & Rangel, 2024b, en revisión 2024), ya sea, con formato *hexadecimal* o *pseudo-hexadecimal*, investigaciones que presentan métodos muy similares a las propuestas aquí experimentadas. Situación que puede dar lugar a una posible línea de investigación futura, la cual, puede consistir en adaptar el uso del formato *pseudo-hexadecimal* a las variantes

basadas en *mutación* con *AES-256,* lo que podría ofrecer nuevas perspectivas y mejoras en la seguridad del *cifrado de datos dinámico*, e incluso, se podría realizar un estudio a profundidad sobre el tema de la *ambigüedad* en el cifrado de datos, convirtiendo las propuestas aquí experimentadas, en algoritmos del tipo *asimétricos*. Por supuesto, ello tendría que ser considerado en uno o varios trabajos futuros.

5. AGRADECIMIENTOS

Este trabajo fue parcialmente soportado por el Tecnológico Nacional de México, campus Instituto Tecnológico de Ciudad Altamirano.

6. REFERENCIAS BIBLIOGRÁFICAS

Alghisi, G.A., & Gringoli, F. (2024). "An Experimental Analysis of the WPA3 Protocol in IoT Devices". 22nd Mediterranean Communication and Computer Networking Conference (MedComNet), pp. 1-4. Recuperado de: https://iris.unibs.it/handle/11379/614913, (01/06/2025).

Alkhwaja, I., Albugami, M., Alkhwaja, A., Alghamdi, M., Abahussain, H., Alfawaz, F., Almurayh, A., & Min-Allah, N. (2023). "Password Cracking with Brute Force Algorithm and Dictionary Attack Using Parallel Programming". Applied Sciences 13 (10), 5979. E-ISSN: 2076-3417. DOI: https://doi.org/10.3390/app13105979, may., 2023. Recuperado de: https://www.mdpi.com/2076-3417/13/10/5979, (14/04/2025).

AL-Maqtari, E.A., & AL-Maqtari, E.A. (2024). "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 issue. DOI: https://doi.org/10.5281/zenodo.13974870. Disponible en: (https://www.puirp.com/index.php/research/article/view/81, 27/04/2025).

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

American National Standards Institute (1963). "American Standard Code for Information Interchange (ASCII)". ANSI X3.4-1963. New York: ANSI. Recuperado de: https://www.ascii-code.com/, (11/11/2024).

Anderson, R. (2008). "Security engineering: A guide to building dependable distributed systems". Second Edition, Wiley.

Android 9 (2024). [Sistema operativo para dispositivos móviles]. Google. Recuperado de: https://www.android.com/intl/es_es/android-12/, (18/11/2024).

Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., & Tokita, T. (2000). "Camellia: A 128-Bit Block Cipher Suitable for Multiple Platforms—Design and Analysis". Proceedings of the Seventh Annual Workshop on Selected Areas in Cryptography (SAC 2000), Springer-Verlag, LNCS vol. 2012, August Issue, eds. D.R. Stinson and S.E. Tavares. Springer-Verlag, Berlin, pp. 39–56.

Baklaga, L. (2024). "Leading The Way In Quantum-Resistant Cryptography For Everyday Safety". Scientific and Practical Cyber Security Journal (SPCSJ), 8(3): 65-73. ISSN: 2587-4667. Scientific Cyber Security Association (SCSA).Tbilisi, Georgia. Recuperado de: https://journal.scsa.ge/papers/leading-the-way-in-quantum-resistant-cryptography-for-everyday-safety/, (23/03/2025).

Baker, M., & Schiller, J. (2015). "ECIES: Elliptic Curve Integrated Encryption Scheme". In Cryptography and Network Security, pp. 245-263. Springer. Recuperado de: https://medium.com/asecuritysite-when-bob-met-

alice/elliptic-curve-integrated-encryption-scheme-ecies-encrypting-using-elliptic-curves-dc8d0b87eaa, (23/03/2025).

Barker, E., & Roginsky, A. (2020). "Recommendation for key management: Part 1 - General". National Institute of Standards and Technology.

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

Barandela, R., Rangel, E., Sánchez, J.S., & Ferri FJ. (2003a). "Restricted Decontamination for the Imbalanced Training Sample Problem". In 8th Iberoamerican Conference on Pattern Recognition, Havana (Cuba). Sanfeliu, A., Ruiz-Shulcloper, J. (eds), Progress in Pattern Recognition, Speech and Image Analysis. CIARP 2003. Lecture Notes in Computer Science, vol 2905. Springer, Berlin, Heidelberg. Springer-Verlag, pp. 424-431, ISBN: 978-3-540-20590-6. Recuperado de: https://link.springer.com/chapter/10.1007/978-3-540-24586-5_52. DOI: https://doi.org/10.1007/978-3-540-24586-5_52, (01/06/2025).

Barandela, R., Sánchez, J.S., García, V., & Rangel, E. (2003b). "Strategies for Learning in Class Imbalance Problems". Pattern Recognition (Edited & Published by Elsevier, Rapid and Brief Comunication, Pergamon in United Kingdom). ISSN: 0031-3203, vol. 36, no. 3, pp. 849-851, march issue 2003. PII: S0031-3203(02)00257-1.0031-3203/02/. Recuperado de: https://www.sciencedirect.com/science/article/pii/S0031320302002571. DOI: https://doi.org/10.1016/S0031-3203(02)00257-1, (14/11/2002).

Barandela, R., Sánchez, J.S., García, V., & Rangel, E. (2001). "Fusion of techniques for handling the imbalanced training sample problem". In Procedings of 6th Ibero-American Symposium on Pattern Recognition, Florianópolis, Brazil, pp. 34-40, 2001. Recuperado de: https://erangel.coolpage.biz/pappers/p2001.jpg, (01/12/2001).

Barandela, R., Sánchez, J.S., & Rangel, E. (2003c). "Two Modifications of the Decontamination Methodology". IASTED, International Conference On Artificial Intelligence and Applications, pp. 391-396, Benalmádena, Spain (Edited & Published by ACTA Press in Calgary, Alberta, Canada), ISBN: 0-88986-390-3. Recuperado de: https://www.actapress.com/PaperInfo.aspx?PaperID=15031 &reason=500, (01/06/2025).

Barranco, F., & Galindo, C. (2022). "Criptografía básica y algunas aplicaciones". Universidad Jaume I, Departamento de Matemáticas, Castellón, España. Recuperado de: https://repositori.uji.es/xmlui/bitstream/handle/10234/201359/TFM_2022_Barranco_Bl%C3%A1zquez_FranciscoMiguel.pdf?sequence=1, (09/11/2024). URI: http://hdl.handle.net/10234/201359. Recuperado de: https://repositori.uji.es/items/35da2f29-ee4a-4dbc-a82f-c450a81cf9be, (13/04/2025).

Bošnjak, L., Sres, J., & Brumen, B. (2018). "Brute-force and dictionary attack on hashed real-world passwords". Conference: 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). DOI: 10.23919/MIPRO.2018.8400211. Recuperado de: https://www.researchgate.net/publication/326700354_Brute-force_and_dictionary_attack_on_hashed_real-world_passwords, (23/05/2025).

Centellas, L.S., Blanco, L., & Sandoval, J.P. (2022), "Estudio comparativo de los algoritmos de criptografía simétrica AES, 3DES y ChaCha20". Revista ActaNova (ISSN: 1683-0768, e-ISSN: 1683-0789), vol. 10, no. 3, Epub 31-Mar-2022, Cochabamba, marzo. Recuperado de: http://www.scielo.org.bo/scielo.php?script=sci_arttext&pid=S1683-07892022000100283, (13/04/2025). Recuperado de: https://dialnet.unirioja.es/servlet/articulo?codigo=10071717, (16/04/2025).

Chong F., Bib, L., Yu, S.M., Xiao, L., & Jun, J. (*2011*). "A Novel Chaos-based Bit level Permutation Scheme For Digital Imagen Encryption". Optics communications, volumen 284, august [en línea]. Recuperado de: www.elsevier.com/locate/optcom, (09/11/2024).

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

Courtois, N.T. (*2012*). "Security Evaluation Of GOST 28147-89". In: View Of International Standardisation. Cryptologia, Vol. 36, no. 1, 2012, pp. 2-13.

Cover, T.M., & Hart, P.E. (1967). "Nearest Neighbor Pattern Classification". IEEE Transactions on Information Theory (e-ISSN: 1557-9654), Volume IT-13, January 1967, pages 21-27. DOI: 10.1109/TIT.1967.1053964. Recuperado de: https://ieeexplore.ieee.org/abstract/document/ 1053964/ , (23/03/2024).

Cryptography (2025). "Cryptography 45.0.4. Python Software Foundation". Recuperado de: https://pypi.org/project/cryptography/, (01/06/2025)..

Dang, Q. H., & Le, H. Q. (*2022*). "Improved cryptanalysis of the RSA algorithm using side-channel attacks". Journal of Information Security and Applications, 65, 103313.

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

Delman, B. (2004). "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. Recuperado de: 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).

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

Drzani, M. (2014). "Securing E-learning platforms". 2014 International Conference on Web and Open Access to Learning (ICWOAL), Dubai, United Arab Emirates, 1-4. Recuperado de: https://doi.org/10.1109/ICWOAL.2014.7009237, (23/03/2025).

Dunkelman, O., Keller, N., & Weizmann, A. (2023). "Practical-Time Related-Key Attack on GOST with Secret S-boxes", In: Handschuh, H., Lysyanskaya, A. (eds) Advances in Cryptology - CRYPTO 2023. CRYPTO 2023. Lecture Notes in Computer Science (Annual International Cryptology Conference. Publisher by Springer, Cham., eBook Packages: Computer Science R0), ISBN: 978-3-031-38547-6, e-ISBN 978-3-031-38548-3, vol. 14083, no. 1, pp. 177-208, august, 2023, doi: https://doi.org/10.1007/978-3-031-38548-3_7.

Fuegner, P. (2024). "Are RSA and AES Both at Risk From the Quantum Threat?". QuSecure, Inc. Recuperado de: 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).

Fulgueira, M., Hernández, O.A., & Henry, V. (2015). "Paralelización Del Algoritmo Criptográfico GOST Empleando El Paradigma De Memoria Compartida". Lámpsakos, núm. 14, pp. 18-24. Fundación Universitaria Luis Amigó Medellín, Colombia. E-ISSN: 2145-4086; julio-diciembre. DOI: http://dx.doi.org/10.21501/21454086.1633. Recuperado de: http://www.redalyc.org/articulo.oa?id=613965326004, (09/11/2024).

Garai, H. K., & Dey, S. (2024). "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. 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).

Gómez, H., & Díaz, J. (2021). Análisis de algoritmos de cifrado simétricos y asimétricos para la protección de datos. Revista de Ingeniería y Tecnología, 20(2), 1-12.

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

Goodman, J. (1968). "Introduction To Fourier Optics". New York: McGraw-Hill.

Granados, G. (2006). "Introducción A La Criptografía". Revista Digital Universitaria, 7(7), s.p. Recuperado de: http://www.revista.unam.mx/vol.7/num7/art55/int55.htm, (09/11/2024).

Griindlingh, W., & Van-Vuuren, J. H. (*2002*). "Using Genetic Algorithms to Break a Simple Cryptographic Cipher". Retrieved March 31, 2003. Recuperado de: http://dip.sun.ac.za/~vuuren/abstracts/abstr_genetic.htm, (06/06/2024).

Gundaram, P.K. (2024). "Algebraic Cryptanalysis of Reduced-Round International Data Encryption Algorithm". Research Square, ISSN: 2693-5015 (online), PREPRINT (Version 1), august. DOI: https://doi.org/10.21203/rs.3.rs-4785692/v1. (https://www.researchsquare.com/article/rs-4785692/latest.pdf, 23/08/2024).

Hankerson, D., Hernandez, J.L., & Menezes, AJ. (2004). "Software implementation of elliptic curve cryptography over binary fields". K. Ko¸c and C. Paar (Eds.): CHES 2000, LNCS 1965, pp. 1–24, 2000. Springer-Verlag Berlin Heidelberg 2000. Recuperado de: 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).

Iavich, M., Kuchukhidze, T., & Gagnidze, A. (2024). "Post-quantum Digital Signature Using Verkle Trees And Lattices". Scientific and Practical Cyber Security Journal (SPCSJ), 8(3): 35-52. ISSN: 2587-4667. Scientific Cyber Security Association (SCSA).Tbilisi, Georgia. Recuperado de: https://journal.scsa.ge/issues-archive/, (23/03/2025).

Ishchukova, E., Maro, E., & Pristalov, P. (2020). Algebraic Analysis of a Simplified Encryption Algorithm GOST R 34.12-2015. Computation, 8(2), 51. https://doi.org/10.3390/computation8020051. Recuperado de: https://www.mdpi.com/2079-3197/8/2/51, (06/06/2024).

Iyengar, S. S., Kannan, R., & Ganapathi, S. (*2021a*). "Inteligencia artificial y criptografía: Tendencias y desafíos". IEEE Transactions on Emerging Topics in Computing, 9(2), 833-844.

Iyengar, S. S., Kannan, R., & Ganapathi, S. (*2021b*). "Análisis de patrones en datos cifrados utilizando inteligencia artificial". Pattern Recognition Letters, 146, 168-175.

Javidi, B., & Horner, J.L. (1994). "Optical Pattern Recognition for Validation and Security Verification". Optical Engineering, 33, 1752-1756. DOI: https://doi.org/10.1117/12.170736.

Jiménez, M., Flores, O., & González, M.G. (2015). "Sistema para codificar información implementando varias órbitas caóticas". Ingeniería. Investigación y Tecnología, vol. XVI, núm. 3, julio-septiembre, pp. 335-343. ISSN 1405-7743 FI-UNAM / ISSN: 1405-7743. Universidad Nacional Autónoma de México. Distrito Federal, México. Recuperado de: http://www.redalyc.org/articulo.oa?id=40440683002, (09/11/2024).

Kalsi, S., Kaur, H., & Chang, V. (*2018*). "DNA Cryptography and Deep Learning using Genetic Algorithm with NW algorithm for Key Generation". Convergence of Deep Machine Learning and Nature Inspired Computing Paradigms for Medical Informatics. Image & Signal Processing; In Journal of Medical Systems, volume 42, Article number: 17. DOI: https://doi.org/10.1007/s10916-017-0851.

Katz, J., & Lindell, Y. (*2019*). "Introduction to modern cryptography". CRC Press.

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

Kuncheva, L. I., & Jain, L. C. (1999). "Nearest Neighbor Classifier: Simultaneous editing and feature selection". Pattern Recognition Letters, 20, 1149-1156.

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

Linfei, C., and Daomu, Z. (2005). "Optical Image Encryption Based On Fractional Wavelet Transform". Opt. Comm. Vol. 254, p.p. 361-367. Recuperado de: https://ui.adsabs.harvard.edu/abs/2005OptCo.254..361C/abstract, (06/06/2024).

Liu, X., & Wang, X. (2021). A Secure Data Encryption Scheme Based on Chaotic Map and DNA Computing. Journal of Cybersecurity, 1(1), 1-12.

Liu, X., & Wang, X. (*2022*). "Quantum cryptanalysis of lattice-based cryptographic protocols". Physical Review X, 12(2), 021004.

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

Mamman, J., Onoja, O., & Blessing, E. (2024). "Mitigating The Impact Of Phishing Attacks On The E-learning Infrastructure". Scientific and Practical Cyber Security Journal (SPCSJ), 8(3): 21-34. ISSN: 2587-4667. Scientific Cyber Security Association (SCSA). Tbilisi, Georgia. Recuperado de: https://journal.scsa.ge/issues-archive/, (23/03/2025).

Matthews, R.A.J. (*1993*). "The use of genetic algorithms in cryptanalysis". Cryptologia, 17(4), 187-201.

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

Microsoft (2024). "Descarga de software". Microsoft. Recuperado de: https://www.microsoft.com/es-mx/software-download, (18/11/2024).

Mohammed, W.A.Y., Fattah, S., Saeed, K.M.O., Ibrahim, A.O., & Eltahier, S. (2025). "Enhancing the RC4 Algorithm by Eliminating the Initiative Vector (IV) Transmission". Engineering, Technology & Applied Science Research (e-ISSN: 1792-8036), 15(1), february, 20242–20248. DOI: https://doi.org/10.48084/etasr.9208. Recuperado de: https://etasr.com/index.php/ETASR/article/view/9208, (01/06/2025).

Montgomery, P.L. (1987). "Speeding up the Pollard rho method". Mathematics of Computation, 48(177), 453-456.

Muhammed, R. K., Aziz, R.R., Hassan, A.A., Aladdin, A.M., Saydah, S.J., Rashid, T. A., & Hassan, B.A. (2024). "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. DOI: https://doi.org/10.24017/science.2024.1.5, URL: https://doi.org/10.48550/arXiv.2407.16274. Disponible en: (https://arxiv.org/abs/2407.16274, 01/06/2025).

Nagaraj, S., & Srinadth, V. (*2015*). "Data Encryption and Authetication Using Public Key Approach". International Conference on Computer, Communication and Convergence (ICCC 2015).

NIST. (2001). "FIPS PUB 197: Advanced Encryption Standard (AES)". U.S. Department of Commerce.

NIST. (2013). "Special Publication 800-56A Revision 2: Recommended methods for key establishment using public key cryptography". NIST Special Publication 800-56A. Recuperado de: *https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://nvlpubs.nist.gov/nistpubs/specialpubl*

ications/nist.sp.800-56ar2.pdf&ved=2ahUKEwio1eCz3KGMAxWgJ0QIHYc4BXQQFnoECBsQAQ&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).

Oppliger, R. (2005). "Contemporary cryptography", (1ra. ed.), Artech House Computer Security Library, Boston/London (ISBN: 1-58053-642-5, 978-8189265038), 510p. Recuperado de: 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%25 20Oppliger.pdf&ved=2ahUKEwiT1tm1xNWMAxUAmO4BHWYLNa0QFnoECBoQAQ&usg=AOvVaw3GvxLI6QzJhvEXqcl9efPz, (13/04/2025).

Pisarchik, A.N., & Flores-Carmona, N.J. (*2006*). "Computer Algorithms For Direct Encryption And Decryption Of Digital Images For Secure Communication". Proceeding of the 6th WSEAS International Conference On Applied Computer Science (Canary Islands, Spain), pp. 29-34.

Pisarchik, A.N., & Zanin, M. (2008). "Imagen Encryption Witch Chaotically Coupled Chaotic Maps". Elsevier Physica, abril [en línea], D 237. Recuperado de: www.elsevier.com/locate/physd.

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

Pydroid3 versión 7.4_arm64 (2024). [IDE for Python 3. Lenguaje de programación y compilador]. Google Play Store. Recuperado de: https://play.google.com/store/apps/details?id=ru.iiec.pydroid3&hl=en&pli=1, (18/11/2024).

PyPI (2024). "Pycryptodome 3.21.0. Python Software Foundation". Recuperado de: https://pypi.org/project/pycryptodome/, (16/11/2025).

Python.org (2024). "The Python Network". Python.org. Recuperado de: https://www.python.org/downloads/, (18/11/2024).

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

Rajan, B., & Saumitr, P.A. (2006). "Novel Compression And Encryption Scheme Using Variable Model Arithmetic Coding And Coupled Chaotic System". IEEE Transactions on circuits and system- I, april, vol. 53 (No. 4). Recuperado de: https://www.researchgate.net/publication/3451216_A_novel_compression_and_encryption_scheme_using_variable_model_arithmetic_coding_and_coupled_chaotic_system, (06/06/2024).

Rambe, B.M., Nababan, E.B., & Nasution, M.K. (2024). "Performance Analysis Of The Combination Of Blum Blum Shub And RC5 Algorithm In Message Security". JITE (Journal of Informatics and Telecommunication. Engineering, e-ISSN: 2549-6255), 7(2), january issue, 409-423. DOI: 10.31289/jite.v7i2.10937. Recuperado de: https://ojs.uma.ac.id/index.php/jite/article/view/10937, (06/01/2025).

Rangel, E. (2002). "Vecinos Envolventes para Variantes de la Regla del Vecino más Cercano". MSc Thesis [Tesis de Maestría en Ciencias]. División de Estudios de Posgrado e Investigación, Instituto Tecnológico de Toluca, Metepec, Estado de México, México, 2002.

Rangel, E. (2022). "La Regla De Los k Vecinos Más Cercanos (k-NN) Basada En Distancia De Manhattan (City-Block) Para Mejorar La Clasificación De Patrones". En Quinto (V) Congreso Nacional De Investigación En Ciencia E Innovación De Tecnologías Productivas. Tecnológico Nacional De México, Instituto Tecnológico de Ciudad Altamirano, Estado De Guerrero, México. Noviembre, 2022. Recuperado de: https://erangel.coolpage.biz/pappers/edgarrangel2022.pdf, (22/11/2022).

Rangel, E. (2024). "Documento Entregable Que Corresponde Al Objetivo y Actividad #1". [Informe no publicado]. Primer Informe Parcial, Proyecto: (19329): La Regla Del Vecino Más Cercano Como Alternativa

Para Inyectar Ruido A Mensajes Encriptados Por El Algoritmo: Noised Random Pseudo-Hexadecimal (Clave: 19329.24-P). Tecnológico Nacional de México. Instituto Tecnológico de Ciudad Altamirano. Departamento de Sistemas y Computación (01/08/2024).

Rangel, E., & García, O. (2002). "Nearest Centroid Neighbour, An Alternative in the Speech Recognition for the Execution from a Mobile Robot Simulator (Applied to the Imbalanced Training Sample Problem)". In CIICC'02, the 9th International Congress On Computer Science Research. October 23-25, 2002, Instituto Tecnológico de Puebla, México. ISBN: ISBN-970-18-8526-0. Recuperado de: https://erangel.coolpage.biz/pappers/p2002.jpg, (25/10/2002).

Rangel, E., & Rangel, K.U. (2024a). "Novel Random Encryption Methods Based On Mutation Strategies Of Artificial Intelligence". SPCSJ, Scientific and Practical Cyber Security Journal (Edited & Published by SCSA, Scientific Cyber Security Association in Tbilisi, Georgia). ISSN: 2587-4667, vol. 8, no. 3, pp. 84-91, september issue 2024. Recuperado de: https://journal.scsa.ge/issue/september-2024/, (https://journal.scsa.ge/papers/novel-random-encryption-methods-based-on-mutation-strategies-of-artificial-intelligence/, 05/11/2024).

Rangel, E., & Rangel, K.U.(2024b). "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 (Edited & Published by IBERAMIA. Iberoamerican Society of Artificial Intelligence: Sociedad Iberoamericana de Inteligencia Artificial in Valencia, España). ISSN: 3020-7444, vol. 1, no. 2, pp. 1–15, december issue 2024. Recuperado de: https://inteletica.iberamia.org/index.php/journal/article/view/16, (16/12/2024).

Rangel E., & Rangel K.U. (en revisión, 2024). "Novel Pseudo-Hexadecimal Encryption Strategies For Camouflaging Ciphertext Based On Nearest Neighbor With Artificial Intelligence". Manuscrito en revisión. IJCOPI, International Journal of Combinatorial Optimization Problems and Informatics (Edited & Published by Editorial Académica Dragón Azteca, S. de R.L..C.V., México). ISSN: 2007-1558, issue 2024. URI: https://ijcopi.org/ojs/authorDashboard/submission/529, (15/10/2024).

Rangel, E., & Rangel, K.U. (2025a). "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. Recuperado de: https://inteletica.iberamia.org/index.php/journal/issue/view/4, (https://inteletica.iberamia.org/index.php/journal/article/view/45, 03/09/2025).

Rangel, E., & Rangel, K.U. (en revisión, 2025). "A Case Study of the TwoFish Encryption Algorithm Based On Random Noisy Injection with Artificial Intelligence". Manuscrito en revisión #1253. JCERP, Journal of Cybersecurity Education, Research and Practice [Ed. Bepress, Elsevier. Part of the Information Security Commons, Management Information Systems Commons, and the Technology and Innovation Commons. Edited & Published by DigitalCommons@ Kennesaw GA: Kennesaw State University, Coles College of Business Center for Security Information in Georgia, United States (USA). ISSN: 2472-2707]. Website: https://digitalcommons.kennesaw.edu/jcerp, (22/07/2025).

Rangel, E., Rangel, K.U., Bernal, C.A., González, L., Rodríguez, C.A., Ascencio, L.J., & Reynoso, R.I. (en revisión, 2025a). "A Case Study of the Salsa20 Encryption Algorithm Using Random Noisy Injection Enhanced by Artificial Intelligence. Manuscrito en revisión. Innovation and Software Journal: Innovación y Software. Revista de la Facultad de Ingenierías y Arquitectura de ULASALLE (Editado por: La Salle Universidad, Arequipa, Perú. ISSN: 2708-0927, ISSN-e: 2708-0935), in press. URI: https://revistas.ulasalle.edu.pe/innosoft/issue/archive, (30/09/2025).

Rangel, E., Rangel, K.U., & González, L. (2024). "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. Recuperado de: https://sabes.edu.mx/revista-electronica/27/# , (https://sabes.edu.mx/revista-electronica/27/docs/4_Cifrado%20de_Datos_Dinamico.pdf, 17/12/2024).

Rangel, E., Rangel, K.U., & González, L. (2025a). "Dynamic Encryption Methods Based On Noisy Injection And Camouflaging Ciphertext Strategies With Artificial Intelligence". SPCSJ, Scientific and Practical Cyber Security Journal (Edited & Published by SCSA, Scientific Cyber Security Association in Tbilisi, Georgia). ISSN: 2587-4667, vol. 9, no. 1, 82-104, march issue 2025. Recuperado de: https://journal.scsa.ge/issue/march-2025/, (23/04/2025).

Rangel, E., Rangel, K.U., & González, L. (2025b). "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. Recuperado de: https://sabes.edu.mx/revista-electronica/28/# , (https://sabes.edu.mx/revista-electronica/28/docs/2_inyeccion_de_ruido.pdf, 01/08/2025).

Rangel, E., Rangel, K.U., González, L., Bernal, C.A., Ascencio, L.J., Reynoso, R.I., & Rodríguez, C.A. (2025c). "Novel Dynamic CAST-128 Encryption Scheme And Comparison With Five Random Noisy Injection Strategies With Artificial Intelligence". SPCSJ, Scientific and Practical Cyber Security Journal (Edited & Published by SCSA, Scientific Cyber Security Association in Tbilisi, Georgia). ISSN: 2587-4667, vol. 9, no. 4, pp. 32-49, september issue 2025. Recuperado de: https://journal.scsa.ge/issue/september-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).

Rangel, E., Rangel, K.U., González, L., Bernal, C.A., Reynoso, R.I., Rodríguez, C.A., & Ascencio, L.J. (en revisión, 2025b). "ChaCha20 Encryption Algorithm Security Enhancement through Artificial Intelligence-Based Random Noisy Injection: A Case Study". Manuscrito en revisión. CULCYT. Cultura Científica y Tecnológica. Revista de investigación en ingeniería e innovación tecnológica (Editado por Revistas Electrónicas, Universidad de Ciudad Juárez, Chihuahua, México. ISSN: 2007-0411). URI: https://revistas.ulasalle.edu.pe/innosoft/authorDashboard/submission/317, (https://erevistas.uacj.mx/ojs/index.php/culcyt, 30/10/2025).

Rangel, E., Rangel, K.U., González, L., Ortiz, A., & Rodríguez, C.A. (2025d). "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 & Published by SCSA, Scientific Cyber Security Association in Tbilisi, Georgia). ISSN: 2587-4667, vol. 9, no. 3, pp. 59-77, june issue 2025. Recuperado de: 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).

Rangel, E., Rangel, K.U., Medrano, J., Bernal, C.A., & González, L. (2023). "Algoritmo Genético Para Cifrado De Datos, Basado En Un Nuevo Concepto Pseudo-Hexadecimal Con Inteligencia Artificial". En Sexto (VI) Congreso Nacional De Investigación En Ciencia E Innovación De Tecnologías Productivas. Tecnológico Nacional De México, Instituto Tecnológico de Ciudad Altamirano, Estado De Guerrero, México. Noviembre, 2023. Reserva de Derechos al Uso Exclusivo No. 04-2023-091910490900-102. ISSN: En trámite. Año 1, No. 2, pp. 1-35, issue marzo-mayo, 2024. Recuperado de: https://www.cdaltamirano.tecnm.mx/index.php/17-vi-congreso-nacional-de-investigacion-en-ciencia-e-innovacion-de-tecnologias-productivas/140-tecnm-40, (11/12/2023).

Rangel, E., Rangel, K.U., Rangel, C.M. (en revisión, 2025c). "Comparação de Dois Esquemas de Criptografia Assimétrica Ruidosa Aleatória (Comparison of Two Random Noisy Asymmetric Encryption Schemes)". Manuscrito en revisión. Revista Ventana Informática (Editado por la Facultad de Ciencias e Ingeniería - Universidad de Manizales, Caldas, Colombia. e-ISSN: 2665-4857). URI: https://doi.org/10.5753/rbie.yyyy.id, https://revistasum.umanizales.edu.co/ojs/index.php/ ventanainformatica/, (30/10/2025).

Rangel, E., Rangel, K.U., Rangel, C.M., Bernal, C.A., Rodríguez, C.A., & González, L. (en revisión, 2025d). "A Novel IDEA-128 Enhancement Using Noisy Injection And Artificial Intelligence For Increasing Ciphertext Security". Manuscrito en revisión. SPCSJ, Scientific and Practical Cyber Security Journal (Edited & Published by SCSA, Scientific Cyber Security Association in Tbilisi, Georgia). ISSN: 2587-4667, vol. 9, december issue 2025. URI: https://journal.scsa.ge/issues-archive/ (30/10/2025).

Reddaiah, B. (2016). "A Study on Pairing Functions for Cryptography". IJCA (0975-8887), Vol. 149, No. 10, September, pp.4-7.

Reddaiah, B. (2019). "A Study on Genetic Algorithms for Cryptography". International Journal of Computer Applications (0975 – 8887). Volume 177 - No. 28, December. Department of Computer Applications. Yogi Vemana University Kadapa, A.P, India. Recuperado de: https:// www.researchgate.net/publication/338012809_A_Study_on_Genetic_Algorithms_for_Cryptography.

Rivest, R. L., Shamir, A., & Adleman, L. (1978). "A method for obtaining digital signatures and public-key cryptosystems". Communications of the ACM, 21(2), 120-126. Recuperado de: https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://people.csail.mit.edu/rivest/Rsapaper.pdf&ved=2ahUKEwjg2fiM2qGMAxXbhu4BHe9iCSsQFnoECDUQAQ&usg=AOvVaw1FtI2T7P32pKAb6jnZSqxi, (23/03/2025).

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

Rueda, A.S., & Lasprilla, M. (*2002*). "Encriptación Por Conjugación De Fase En Un BSO Utilizando Señales Ópticas De Baja Potencia". Rev. Col. Fís., Vol. 34, No.2, P.P.636-640.

Rueda, J.E., Romero, A.L., and Castro, L.M. (2005). "Criptografía Digital Basada En Tecnología Óptica". Bistua: Revista de la Facultad de Ciencias Básicas, vol. 3, núm. 2, julio, pp. 19-25. ISSN 0120 - 4211. Universidad de Pamplona, Colombia. Recuperado de: http://www.redalyc.org/articulo.oa?id=90330203, (09/11/2024).

Sami Sulaiman, A., & M. Hammood, M. (2025). "Enhancing AES Security based on Camellia Key Schedule". Samarra Journal of Pure and Applied Science (e-ISSN: 2789-6838), 7(1), march issue, pp. 299–309. DOI: https://doi.org/10.54153/sjpas.2025.v7i1.1020. Recuperado de: https://sjpas.com/index.php/sjpas/article/view/1020/400, (30/03/2025).

Sandoval, M., & Santamaría, S.E. (2024). "Evaluación Comparativa entre WPA2 y WPA3 en Redes Inalámbricas". Trabajo Final de Grado. Repositorio: Unidades Tecnológicas de Santander. URI: http://repositorio.uts.edu.co:8080/xmlui/handle/123456789/17095, (01/06/2025).

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

Schneier, B. (2000). "Secrets and lies: Digital security in a networked world". Wiley.

Sengupta, S., & Ghosh, S. (2023). "Quantum Computing Encryption Threats: Why RSA and AES Are at Risk". Journal of Cryptographic Research.

Shannon, C. E. (1948). "A mathematical theory of communication". Bell System Technical Journal, vol. 27, no. 3, pp. 379-423, July issie. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

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

Stinson, D.R., & Paterson, M.B. (2019). "Cryptography: Theory and Practice", (4ta ed.), Chapman and Hall Book/CRC Press (Taylor & Francis Group, ISBN: 978-1-1381-9701-5). Recuperado de: *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_CryptographyTheoryAndPractice-*

CRC%2520Press%2520%25282019%2529.pdf&ve d=2ahUKEwi8_euUtdWMAxWzh-4BHShGAaQQFno ECCUQAQ&usg=AOvVaw1DCBGkDbaZMeHuyR45xZZC, (13/04/2025).

Stubblefield, A., Ioannidis, J., and Rubin, AD. (2002). "Using the Fluhrer, Mantin, and Shamir attack to break WEP". Lecture Notes in Computer Science. Reporte Técnico, AT&T Labs, 2001. NDSS, 2002. Recuperado de: https://scholar.google.com.mx/scholar?hl=es&as_sdt=0%2C5&as_vis=1&q=Using+ the+Fluhrer%2C+Mantin%2C+and+Shamir+attack+&btnG=#d=gs_qabs&t=1750281807768&u=%23p%3DV 2c50dkwmJkJ, (01/06/2025).

Susilo, W., Tonien, J., & Yang, G. (2021). "Divide and capture: An improved cryptanalysis of the encryption standard algorithm RSA", Computer Standards & Interfaces (ISSN: 0920-5489), vol. 74 (1): 103470, february, DOI: https://doi.org/10.1016/j.csi.2020.103470. Recuperado de: https://www.sciencedirect.com/science/article/pii/S0920548920303561, (06/06/2024).

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

Unicode Consortium (2022). "The Unicode Standard, Version 14.0". Unicode Consortium. Recuperado de: https://www.unicode.org/consortium/consort.html, (11/11/2024).

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

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

Wang, S., Cui, T., Wang, M. (2016).."Improved Differential Cryptanalysis of CAST-128 and CAST-256". In Information Security and Cryptology. Inscrypt. Lecture Notes in Computer Science [Chen, K., Lin, D., Yung, M. (eds), Springer, Cham, ISBN: 978-3-319-54705-3], vol 10143, march, 2017. DOI: https://doi.org/10.1007/978-3-319-54705-3_2. (https://link.springer.com/chapter/10.1007/978-3-319-54705-3_2, 01/07/2025).

Wang, X., & Chen, X. (*2022*). "Optical image encryption based on compressive sensing and double random phase encoding". Journal of Optics, 24(2), 025401.