DESDE 2013
https://repository.uaeh.edu.mx/revistas/index.php/icbi/issue/archive
Pädi Boletín Científico de Ciencias Básicas e Ingenierías del ICBI
ISSN: 2007-6363

Publicación Semestral Pädi Vol. 11 No. Especial 2 (2023) 142–151

Encriptación de imágenes digitales utilizando programación genética
Image encoding using genetic programming

Ariel Cavazos-Amador [a], Ismael Rojas-Montes [a], Eddie Clemente [b], Abraham Flores-Vergara [b,*]

[a] *Maestría en Ciencias en Ingeniería Mecatrónica, TecNM/IT Ensenada, Blvd. Tecnológico 150, Ex-ejido Chapultepec, 22780, Ensenada, Baja California, México.*
[b] *Departamento de Sistemas Computacionales, TecNM/IT Ensenada, Blvd. Tecnológico 150, Ex-ejido Chapultepec, 22780, Ensenada, Baja California, México.*

Resumen

La encriptación de imágenes digitales es un problema de seguridad informática, en particular cuando estas imágenes se desean transmitir de forma segura. En este trabajo se propone una nueva función que opera como generador de números pseudoaleatorios, especializada para la criptografía en imágenes digitales. Esta función es resultado de un proceso de optimización llevado a cabo por programación genética. La función obtenida es capaz de generar criptogramas con un índice de entropía de 7.999, superando y alcanzando el desempeño de generadores de números pseudoaleatorios implementando funciones caóticas. Los criptogramas se validan contra ataques diferenciales bajo los algoritmos NPCR y UACI, obteniendo resultados sobresalientes.

Palabras Clave: Criptografía de imágenes, Programación genética, PRNG.

Abstract

The encryption of digital images is a problem of computer security, particularly when these images are to be transmitted securely. In this work, a new function is proposed that operates as a pseudorandom number generator, specialized for cryptography in digital images. This function is the result of an optimization process carried out by genetic programming. The obtained function is capable of generating cryptograms with an entropy index of 7.999, surpassing and reaching the performance of pseudorandom number generators implementing chaotic functions. The cryptograms are validated against differential attacks under the NPCR and UACI algorithms, obtaining outstanding results.

Keywords: Image encryption, Genetic programming, PRNG.

1. Introducción

El informe de Cisco Annual Internet Report, reporta que para el 2023 estarán conectados a internet cerca de 29,300 millones de dispositivos en todo el mundo alcanzando un tráfico global de 278 exabytes (Cisco, 2022; Aman *et al.*, 2020). Los dispositivos conectados a internet, utilizan servicios y aplicaciones que transmiten y almacenan información de tipo confidencial, por lo que se requieren de técnicas de protección de información que garanticen la segura transmisión y almacenamiento de dicha información por los canales públicos de comunicación.

Actualmente, se utilizan técnicas que implementan algoritmos criptográficos como Advanced Encryption Standard (AES) (Pousa, 2011), Rivest-Shamir-Adleman (RSA) (Balbás Gutiérrez, 2019), Elliptic Curve Cryptography (ECC) (Koblitz *et al.*, 2000), entre otros (Vargas y Mnedez, 2015). Sin embargo, con el avance y desarrollo de la tecnología computacional, dichas técnicas caen en cierto grado de vulnerabilidad, comprometiendo la seguridad de la información confidencial (Murillo-Escobar *et al.*, 2019; Balbás Gutiérrez, 2019). Por lo anterior, es necesario proponer nuevas técnicas de protección confidencial que garanticen la protección y resguardo seguro de dicha información para su transmisión y/o almacenamiento. En la literatura, se encuentran documentadas investigaciones que proponen métodos y técnicas modernas para el encriptamiento de información confidencial. De las tecnologías documentadas se puede mencionar a la criptografía caótica (Tutueva *et al.*, 2020; Teh *et al.*, 2020; Lambić, 2020), la criptografía ADN (Borda y Tornea, 2010), la criptografía cuántica (Lema Andrango, 2022; Sáez de Buruaga Brouns, 2022) y la criptografía implementando técnicas de cómputo evolutivo (Hernandez *et al.*, 2004; Kösemen *et al.*, 2018; Lamenca-Martinez *et al.*, 2006), entre otras. Dichas

*Autor para correspondencia: aflores@ite.edu.mx
Correo electrónico: al16760481@ite.edu.mx (Ariel Cavazos-Amador), alm17280668@ite.edu.mx (Ismael Rojas-Montes), eclemente@ite.edu.mx (Eddie Clemente), aflores@ite.edu.mx (Abraham Flores-Vergara).

Historial del manuscrito: recibido el 25/03/2023, última versión-revisada recibida el 19/06/2023, aceptado el 22/06/2023, publicado el 11/09/2023. **DOI:** https://doi.org/10.29057/icbi.v11iEspecial2.10814

investigaciones, compiten por posicionarse dentro de las mejores técnicas de encriptamiento de información confidencial optimizando factores como la sensibilidad de datos, velocidad, eficiencia, complejidad de implementación y recurso computacional.

El encriptado de imágenes, también llamado cifrado de imágenes, es una técnica utilizada para garantizar la protección, privacidad y confidencialidad en la información visual de una imagen. Lo anterior, con el propósito de obtener una imagen ininteligible para terceros no autorizados y realizar de manera segura su transmisión y/o almacenamiento. El cifrado de imágenes se clasifica principalmente en dos tipos, el cifrado en flujo y el cifrado en bloques, ambos dependientes de un elemento de cifrado llamado *clave* (Shannon, 1949). En el cifrado en flujo, la clave se utiliza para generar un flujo pseudoaleatorio o serie cifrante que, combinada con los datos a encriptar, producen un flujo o *serie cifrada* llamada *criptograma*. En el cifrado en bloques, los datos a encriptar se dividen en bloques discretos de tamaño fijo y cada bloque se cifra de manera independiente utilizando una clave única. La ventaja principal del cifrado en flujo, es que produce métodos rápidos y eficientes, ya que el cifrado se realiza en forma continua y en tiempo real; sin embrago, puede presentar vulnerabilidad debido a errores en la generación del flujo pseudoaleatorio. El cifrado en bloques produce métodos muy seguros, sin embargo, presenta reducción en la rapidez y eficiencia debido a que los datos a encriptar se tienen que dividir en bloques y encriptarse uno por uno.

Con respecto a la clave de cifrado, los métodos criptográficos se clasifican en cifrado simétrico y cifrado asimétrico. El cifrado simétrico, establece una clave que se utiliza tanto para el encriptado como para el desencriptado de la imagen. En el cifrado asimétrico, se utiliza una clave para el encriptado de la imagen y otra para el desencriptado de la misma (Pousa, 2011).

El presente trabajo de investigación, propone un método de encriptamiento utilizando una versión del cifrado en flujo con clave simétrica y las técnicas de diseño mediante programación genética para generar series pseudoaleatorias cifrantes en el encriptamiento de imágenes en escala de grises e imágenes a color. Lo anterior con el propósito de optimizar recursos tales como rapidez, eficiencia, complejidad de implementación y recurso computacional.

Primeramente, se describe una versión del método de cifrado en flujo con clave simétrica en el encriptado de datos de tipo discreto, ver sección 2. En seguida se describe la implementación de la programación genética para proponer un generador de series pseudoaleatorias (PRNG) dedicado al encriptamiento de imágenes, ver sección 3. En la sección 4, se presentan los resultados obtenidos en su implementación en imágenes a color RGB considerando diferentes tamaños de imagen, su encriptado y desencriptado y la validación de los criptogramas mediante pruebas de análisis de seguridad criptográfica tales como Entropía, NPCR (Number of Changing Pixel Rate) y UACI (Unified Averaged Changed Intensity). Finalmente las conclusiones se presentan en la sección 5.

2. Método de cifrado en flujo con clave simétrica para datos de tipo discreto

El método de cifrado en flujo, es uno de los métodos que se utilizan para el encriptamiento de información confidencial y su particular característica es el encriptamiento de flujos de datos continuos en tiempo real. El cifrado en flujo utiliza para el encriptado de la información, una clave de cifrado que sirve para generar una serie pseudoaleatoria cifrante mediante un generador de series pseudoaleatorias (PRNG). La serie cifrante y los datos a encriptar, se combinan mediante una operación de encriptamiento para producir la imagen encriptada o criptograma.

El método de cifrado propuesto es una versión de la técnica de cifrado en flujo para implementarse en datos de tipo discreto, en este caso, imágenes a color RGB. Una imagen RGB, se considera como una composición de una matriz tridimensional con n número de pixeles, cada pixel representa el color o la intensidad de la luz en una ubicación específica de la imagen. Utilizando una codificación de 8 bits por pixel, cada pixel puede tomar un valor en un rango de valores entre 0 y 255. El PRNG propuesto genera una serie cifrante de n números de elementos discretos mediante una clave para ser combinada con el flujo de valores discretos de la imagen a encriptar.

La figura 1 muestra el método de encriptado propuesto, en donde $SC = \{sc_1, sc_2, ..., sc_n\}$ representa la serie cifrante discreta obtenida mediante el PRNG propuesto, siendo n el número de elementos, es decir, el equivalente al total de pixeles de la imagen a encriptar. $D = \{d_1, d_2, d_3, ..., d_n\}$ representa el conjunto de valores discretos que componen la imagen siendo n el número de elementos o pixeles de la imagen y está determinado por n = largo $\times$ ancho $\times$ número$_{capas}$. La operación de encriptamiento involucra al operador XOR en el encriptado de la imagen para obtener el flujo del criptograma $C = \{c_1, c_2, c_3, ..., c_n\}$. Lo anterior produce que se pueda utilizar el mismo método para el proceso de encriptado y desencriptado, es decir, en el encriptado $\text{XOR}(SC, D) = C$, mientras que en el desencriptado $\text{XOR}(SC, C) = D$.

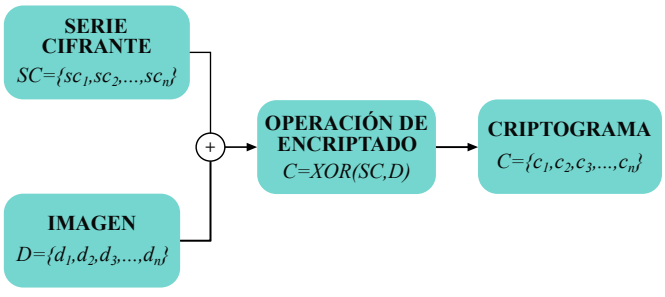


Figura 1: El criptograma C de una imagen se obtiene al realizar una operación XOR entre cada elemento de una serie cifrante, SC, y el valor de cada pixel de una imagen dada, D.

2.1. *Método propuesto para generar serie cifrante*

El método propuesto para la obtención de la serie cifrante se muestra en la figura 2. El generador de serie cifrante, genera n valores discretos pseudoaleatorios en un rango de valores de 0 a 255 o de codificación de 8 bits, correspondiente al mismo rango de valores que puede tomar cada pixel de la imagen

a encriptar. Para la obtención de los valores pseudoaleatorios de la serie cifrante discreta, se utiliza un PRNG que en conjunto con alguna clave x_0 produce una serie pseudoaleatoria $S = \{s_1, s_2, s_3, ..., s_n\}$ de n elementos, los cuales pasan por un procedimiento de conversión o normalización para finalmente obtener los valores discretos de la serie cifrante.

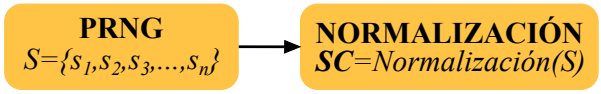


Figura 2: Para obtener la serie cifrante, SC, es necesario aplicar un proceso de normalización a una serie de números pseudoaleatorios, S.

El procedimiento de conversión o normalización, produce una serie de valores discretos, $SC = \{sc_1, sc_2, ..., sc_n\}$, en el rango de 0 a 255. Esto se logra multiplicando cada valor de la serie pseudoaleatoria $S = \{s_1, s_2, s_3, ..., s_n\}$ por el factor 1×10^9 y obtener el residuo del cociente entre $(s_i \times 10^9)$ y 255, con $i = 1$ hasta $i = n$, ver (1). Es decir:

$$sc_i \equiv (s_i \times 10^9) \text{ mód } 255 \quad (1)$$

3. Generación de criptogramas basados en programación genética

Para lograr un criptograma aceptable, se propone optimizar un PRNG específico para esta tarea. Así, el generador de la serie pseudoaleatoria, se obtiene con base a la aplicación de programación genética.

La programación genética es un algoritmo evolutivo que a partir de su estructura y representación es capaz de proponer soluciones para problemas de modelado y reconocimiento de patrones. Los modelos obtenidos por programación genética son analíticos, lo cual permite su análisis e interpretación. El algoritmo sigue una analogía con la teoría sintética de la evolución natural o Neodarwinismo. En este sentido, dado un problema de optimización, cada solución candidata es vista como un individuo; y un conjunto de soluciones candidatas forman una población. De esta forma, la adaptación del individuo al medio, se representa con un índice de bondad asignado a cada solución candidata, dicho índice mide la calidad de la resolución del problema y se denomina aptitud. Con lo cual, el algoritmo busca de forma iterativa (generaciones) la mejor solución, que maximice el desempeño con respecto a su aptitud. En el caso de este problema, se busca la mejor función PRNG que ofrezca el índice más alto de entropía en el criptograma de una imagen digital.

En la programación genética, una posible solución se representa como un árbol sintáctico, donde los nodos del árbol representan funciones u operadores y los nodos hojas o terminales, representan los operandos o variables de una función matemática o computacional, ver Figura 3. Así, la programación genética busca optimizar funciones que cumplan un propósito definido por un criterio.

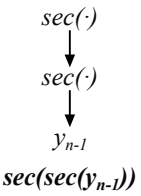


Figura 3: Las funciones en programación genética se representan como árboles sintácticos, y estas se interpretan recorriendo la estructura en *inorder*.

El proceso de la evolución artificial se resume en los siguientes pasos:

1. La población inicial se construye con un conjunto de soluciones candidatas creadas aleatoriamente. En este problema, la población es un conjunto de posibles funciones PRNG.

2. Cada posible solución es evaluada con respecto a una función objetivo o de "aptitud", la cual asigna un índice que refleja su desempeño bajo algún criterio dado. En este caso, se propone una función que indique la calidad de encriptación de la imagen.

3. Se seleccionan las soluciones que actuarán como padres de nuevas soluciones. Esto se realiza aplicando un proceso probabilístico, el cual pondera la probabilidad de selección a aquellas soluciones que tengan un mejor desempeño o aptitud y que estructuralmente son más sencillas. En el presente trabajo se aplica el proceso de selección de tipo torneo "lexictour".

4. Una vez que los padres se han seleccionado, estos se recombinan para crear una descendencia. La recombinación se realiza intercambiando ramas de los árboles sintácticos de los respectivos padres.

5. Los hijos pueden sufrir un cambio aleatorio, llamado mutación, lo que pudiera ayudar a salir de un óptimo local y extender la búsqueda en otros nichos.

6. Una vez que la descendencia ha sido creada. Se seleccionan los individuos que conformarán la nueva población o siguiente generación. En este caso, se ha optado por el método de remplazo generacional con elitismo. Es decir, la descendencia reemplazará toda la generación anterior y solo se conservará el individuo con el valor más alto de aptitud de la población anterior.

7. El proceso se repite a partir del paso 2 hasta cumplir con un número definido de generaciones, ver Figura 4.

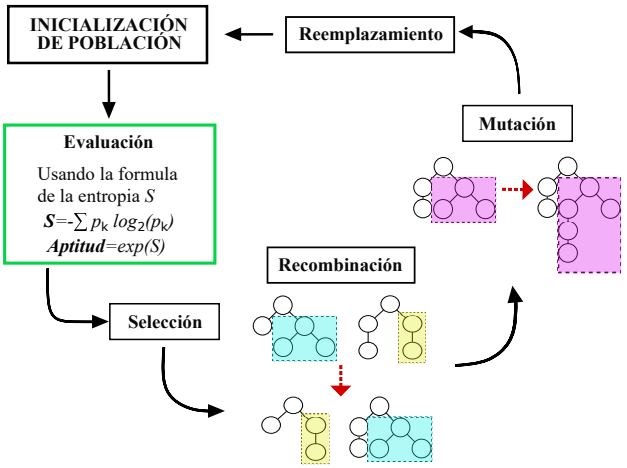


Figura 4: Diagrama general del proceso evolutivo aplicado para la generación de un criptograma, maximizando su entropía.

3.1. *Espacio de búsqueda*

Una parte clave en el empleo de la programación genética es la definición del espacio de búsqueda, este consiste en todas las posibles soluciones candidatas del problema. Dado que una función es representada por sus funciones y terminales en el árbol sintáctico, entonces una elección adecuada de estas permitirá aproximarse a la mejor solución. En este caso las tablas 1 y 2 muestran las funciones y terminales elegidas para este problema. Las funciones contemplan operadores aritméticos, trigonométricos, hiperbólicos, exponenciales, logarítmicos, y funciones especiales como valor absoluto, parte real, signum, máximo y mínimo de un par de argumentos. Mientras las terminales son variables que pueden depender del valor actual, denotado como n, o del valor anterior, $n-1$, o bien valores constantes. Con lo cual es posible construir una función de la forma $y_{n+1} = f(y_n, y_{n-1})$.

Tabla 1: Funciones utilizadas para la generación de los árboles sintácticos.

$+, -, *, /, (\cdot)^2, (\cdot)^3, (\cdot)^{(\cdot)}, \sqrt{\cdot}$, sin(·) , cos(·) , tan(·), sec(·), csc(·), cot(·), asin(·), acos(·), atanr(·), sinh(·), sat(·, ·, ·), cosh(·), tanh(·), sech(·), csch(·), asinh(·), acosh(·), atanh(·), asech(·), acsch(·), acoth(·), $e^{(\cdot)}$, ln(·), Re(·), $\|\cdot\|$, $|\cdot|$, sign(·), máx(·), mín(·)

Tabla 2: Terminales utilizadas para la generación de los árboles sintácticos.

Término	Descripción
y_n	Dependencia del valor actual
y_{n-1}	Dependencia del valor anterior
0.5, 1, 2	Constantes

3.2. *Función de aptitud*

La función de aptitud asigna un valor que representa la calidad de una posible solución. En este caso, una medida adecuada es la entropía. La entropía de la información, permite medir la aleatoriedad que presentan los criptogramas como resultado de implementar un método criptográfico a la imagen original. Para calcular la entropía, S, de una imagen en tonos de gris se realiza mediante (2), donde p_i es la probabilidad de que un píxel tome cierto nivel de gris.

$$S(p) = -\sum_{i=1}^{256} p_i \log_2(p_i) \tag{2}$$

Para una fuente puramente aleatoria que está emitiendo 2^N símbolos con la misma probabilidad, después de evaluar (2) se obtiene una entropía $S(p) = N$. La información que corresponde a los elementos del criptograma son valores numéricos de 8 bits. Por lo tanto, el valor numérico de entropía de la información de la total aleatoriedad de los valores de los elementos del criptograma para este caso es de 8 bits/símbolo. De esta forma, para este trabajo se propone una función de aptitud que contemple la entropía del criptograma, generado por la aplicación de este posible PRNG, ver (3).

$$f_{aptitud} = exp(S(p)) \tag{3}$$

La función de aptitud se calcula como una función exponencial de la entropía, esto produce que las soluciones que se acerquen a una alta aleatoriedad, tengan una diferencia significativa en el índice de bondad dado por la función de aptitud. Lo cual, es benéfico para los procesos de selección en el ciclo evolutivo.

En este trabajo, se utilizó la librería GPLAB escrita en Matlab®, para la implementación del algoritmo de programación genética. En general no existe un método para obtener los mejores parámetros de un algoritmo evolutivo, sin embargo, existen algunas guías, como darle un mayor peso a la probabilidad de cruza. Así, los parámetros utilizados son propuestos después de una etapa de sintonización (prueba y error), estos parámetros se resumen en la tabla 3. El tiempo promedio de ejecución del algoritmo evolutivo, considerando los parámetros propuestos, fue de 47 horas y 29 minutos. Este promedio comprende la ejecución de 30 corridas independientes del algoritmo evolutivo. Las ejecuciones se realizaron utilizando una estación de trabajo Precision 7910 con procesador Dual Intel Xeon Processor E5-2 687W v3 y 64GB en RAM.

Tabla 3: Parámetros utilizados en el proceso evolutivo.

Parámetro	Valor
Número de generaciones	100
Tamaño de población	200
Probabilidad de cruza	80 %
Probabilidad de mutación	20 %
Máxima profundidad del árbol	12
Selección	Lexicográfico
Elitismo	Mantener el mejor

El comportamiento de la evolución con respecto a la aptitud se ilustra en la figura 5. Nótese que la aptitud alcanza su nivel más alto alrededor de la generación 50.

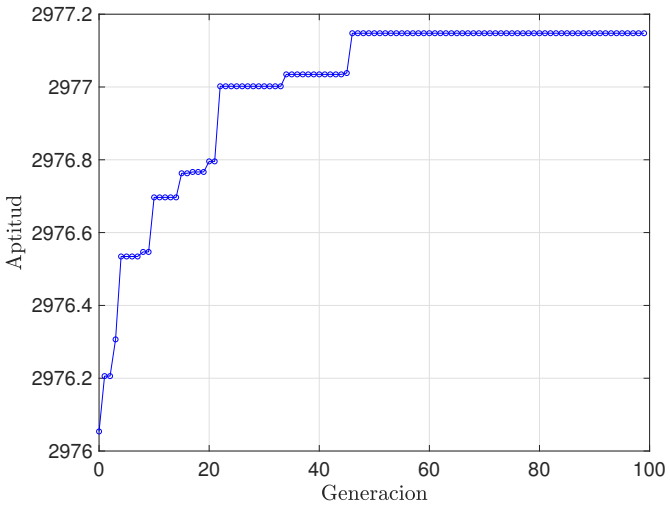


Figura 5: Comportamiento de la aptitud de las soluciones a través de la evolución en 100 generaciones.

El mejor resultado obtenido después del proceso evolutivo está descrito por el modelo matemático (4), donde la función Re{·} obtiene el valor real de su argumento. Este resultado fue entrenado y probado sobre números reales con una precisión de 64 bit.

$$y_{n+1} = \sec(y_n) - \operatorname{csch}(\sec(\sec(\sec(y_n)))) + y_n - \operatorname{atan}(\operatorname{Re}(\sec(\sin(y_n))))) \quad (4)$$

Durante el proceso evolutivo solo se consideró una sola imagen en tonos de grises de 320×320 píxeles de resolución, ver figura 6a, en la cual se evaluaron las diferentes soluciones propuestas por medio de (3). La figura 6e, muestra el histograma de la figura original, mientras la figura 6b muestra el criptograma obtenido aplicando (4), obtenida por la programación genética, y su respectivo histograma, ver Figura 6f.

A manera de comparación, se consideraron dos generadores de series pseudoaleatorias caóticas. La primera es dada por la ecuación logística o mapa logístico (M-L), $y_{n+1} = ry_n(1 - y_n)$, la cual dependiendo del parámetro r y de su valor inicial y_0, esta puede tener un comportamiento caótico. En este caso se tomaron $r = 3{,}7$ y $y_0 = 0{,}6$. Las figuras 6c y 6g muestran el criptograma resultante y su correspondiente histograma al aplicar la función del mapa logístico para producir la serie cifrante. El segundo PRNG considera el mapa Tinkerbell, el cual está definido por un sistema dinámico discreto de dos ecuaciones: $x_{n+1} = x_n^2 - y_n^2 + ax_n + by_n$; y $y_{n+1} = 2x_ny_n + cx_n + dy_n$. Para tener un comportamiento caótico se consideraron los parámetros siguientes $a = 0{,}9$, $b = -0{,}6013$, $c = 2{,}0$, $d = 0{,}5$, $x_0 = -0{,}72$, $y_0 = -0{,}68$. Las figuras 6d y 6h muestran el criptograma y su correspondiente histograma, considerando la serie cifrante, generada por la función caótica Tinkerbell.

Nótese que el criptograma resultante del PRNG-GP propuesto no muestra rastros perceptibles de la imagen original y tiene un comportamiento similar a los generados por las funciones caóticas de mapa logístico y Tinkerbell. De igual forma, el correspondiente histograma muestra una tendencia hacia la uniformidad, característica deseada ya que refleja una adecuada codificación, en este sentido los histogramas correspondientes a los criptogramas generados por el mapa logístico y Tinkerbell muestran esta tendencia.

Si bien los criptogramas e histogramas nos muestran un resultado cualitativo, es necesario obtener una comparación cuantitativa. Lo anterior se logra calculando el índice de aptitud y entropía del criptograma resultante de la aplicación de la función PRNG-GP propuesta, y de las funciones de mapa logístico y Tinkerbell. La tabla 4 muestra estos resultados, note que la función PRNG-GP muestra un mejor desempeño a la función del mapa logístico y un valor comparable con Tinkerbell, aún cuando Tinkerbell es un sistema dinámico definido por dos funciones discretas.

Tabla 4: Comparación en la evaluación de PRNG-GP contra PRNG utilizando funciones caóticas.

Generador	Entropía	Aptitud
Mapa logístico	7.99719	2972.6
Tinkerbell	7.99747	2973.4
PRNG-GP	7.99745	2973.3

La figura 7 muestra las gráficas de las series pseudoaleatorias generadas por el PRNG-GP (Figura 7a), la función mapa logístico (Figura 7b) y la función Tinkerbell (Figura 7c). En estas figuras se muestran sus diferentes dominios y rangos de valores, las cuales después del proceso de normalización, descrito en la sección 2.1, se obtienen sus respectivas series cifrantes. Se puede observar que la trayectoria de los elementos obtenidos a partir de la PRNG-GP (Figura 7d), presenta un comportamiento pseudoaleatorio idóneo para la generación de series potenciales para ser utilizadas como serie cifrante y que además es comparable con series cifrantes generadas por el mapa logístico (Figura 7e) y Tinkerbell (Figura 7f).

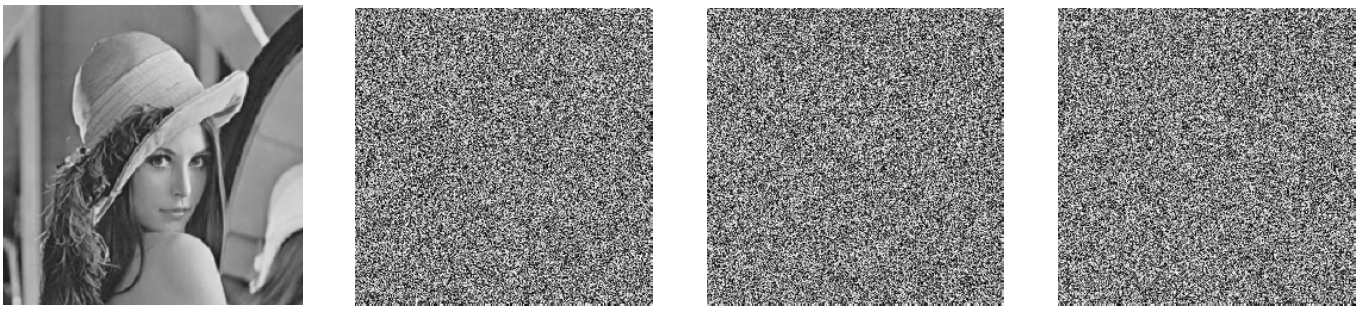
4. Resultados de la implementación del método criptográfico en imágenes a color RGB

4.1. Encriptado de imágenes RGB

En las figuras 8-11 se muestran los resultados de implementar el método criptográfico propuesto en imágenes RGB con resoluciones de 114x171 píxeles (Figura 8), 228x342 píxeles (Figura 9), 912 x 1368 píxeles (Figura 10) y 3648 x 5472 píxeles (Figura 11). Se observa que el resultado del método de encriptado produce imágenes totalmente ininteligibles y los histogramas correspondientes presentan una distribución de valores con tendencia a la uniformidad.

4.2. Desencriptado del criptograma

En la figura 12, se muestran los resultados de implementar el mismo procedimiento del método criptográfico de manera inversa, lo anterior con el propósito de recuperar la imagen original. Se observa que se obtienen imágenes idénticas a las respectivas imágenes originales, es decir, el procedimiento de desencriptado recupera el total de la información de la imagen original. Lo anterior, produce que se pueda utilizar el mismo sistema criptográfico en el proceso de encriptado como en el proceso de desencriptado.

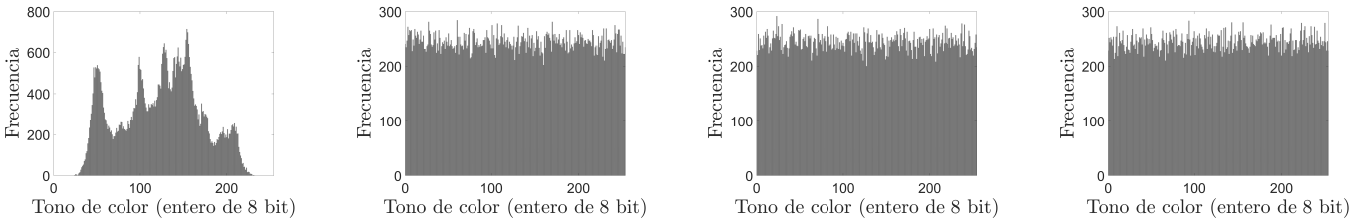


(a) Imagen sin encriptar en escala de grises.

(b) Criptograma obtenido al aplicar PRNG-GP.

(c) Criptograma obtenido al aplicar mapa logístico.

(d) Criptograma obtenido al aplicar Tinkerbell.



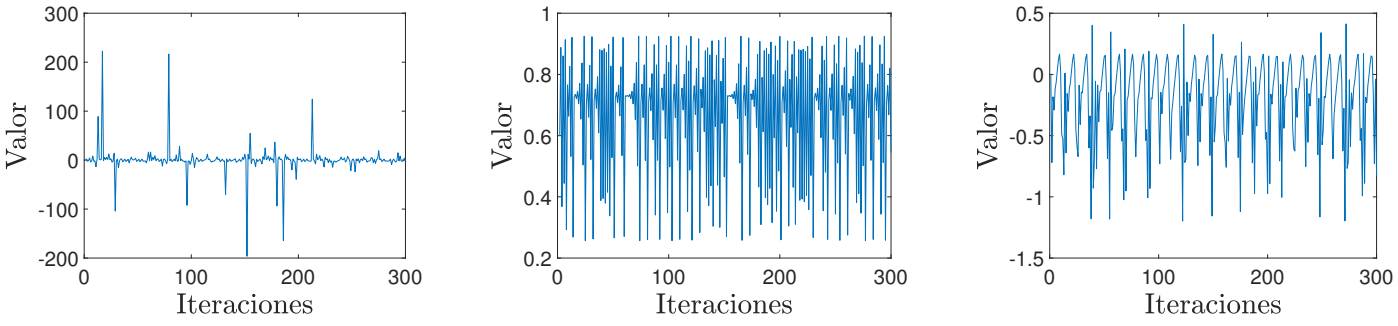
(e) Histograma de la imagen en escala de grises.

(f) Histograma del criptograma obtenido al aplicar PRNG-GP.

(g) Histograma del criptograma obtenido al aplicar M-L.

(h) Histograma del criptograma obtenido al aplicar Tinkerbell.

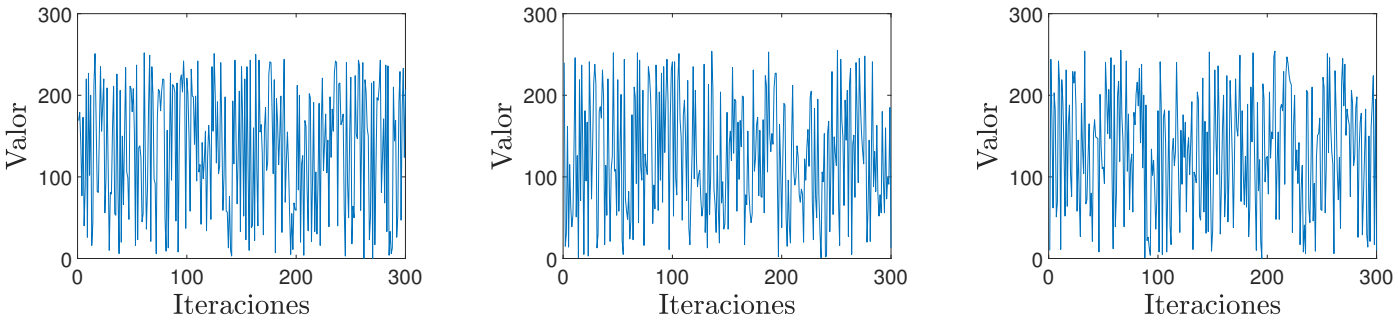
Figura 6: Comparación de tres generadores pseudoaleatorios diferentes para encriptar una imagen. El generador PRNG-GP corresponde a nuestra propuesta, mientras mapa logístico (M-L) y Tinkerbell se refiere a utilizar un PRNGP basado en la función caótica de mapa logístico y Tinkerbell respectivamente.



(a) Serie pseudoaleatoria PRNG-GP.

(b) Serie pseudoaleatoria M-L.

(c) Serie pseudoaleatoria Tinkerbell.



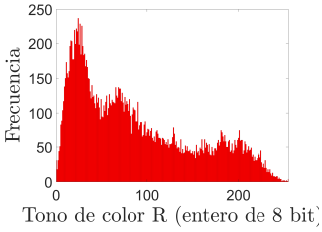
(d) Serie cifrantes PRNG-GP.

(e) Serie cifrante M-L.

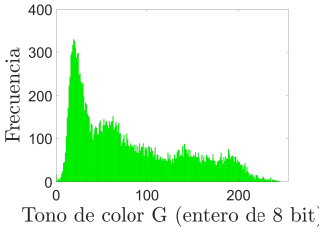
(f) Serie cifrante Tinkerbell.

Figura 7: Comparación de las series pseudoaleatorias y cifrantes.

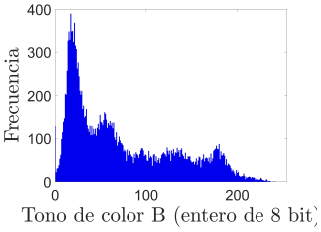
(a) Imagen original.



(b) Histograma de la banda roja.

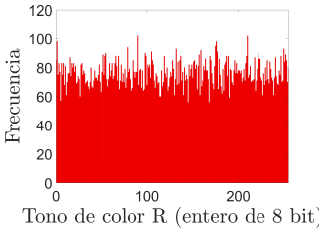


(c) Histograma de la banda verde.

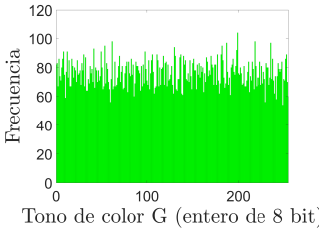


(d) Histograma de la banda azul.

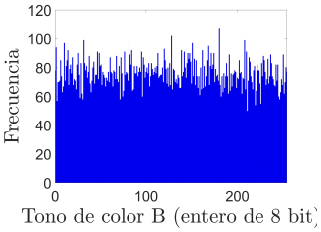
(e) Criptograma.



(f) Histograma de la banda roja.

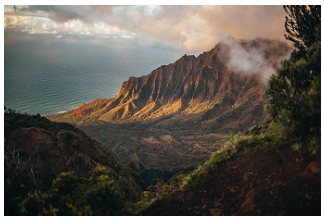


(g) Histograma de la banda verde.

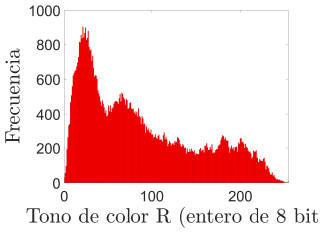


(h) Histograma de la banda azul.

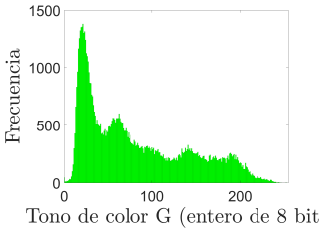
Figura 8: Resolución de 114 × 171



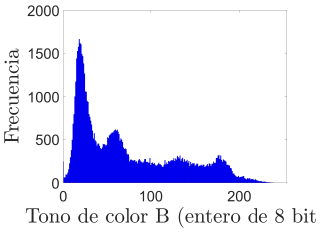
(a) Imagen original.



(b) Histograma de la banda roja.

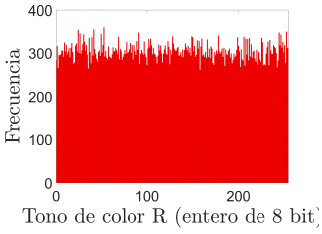


(c) Histograma de la banda verde.

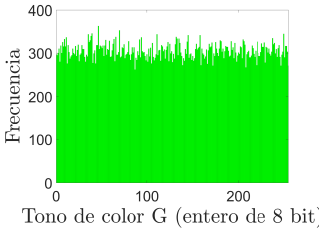


(d) Histograma de la banda azul.

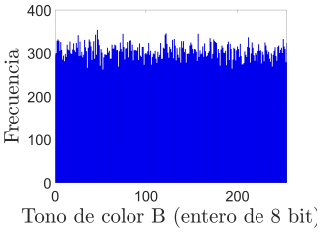
(e) Criptograma.



(f) Histograma de la banda roja.

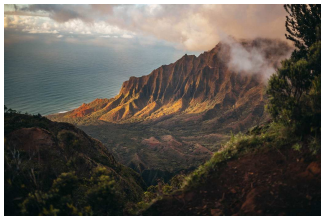


(g) Histograma de la banda verde.

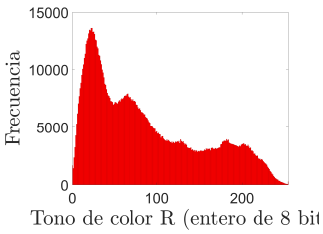


(h) Histograma de la banda azul.

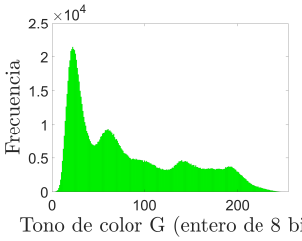
Figura 9: Resolución de 228 × 342



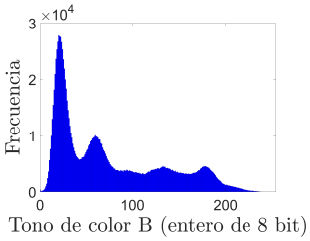
(a) Imagen original.



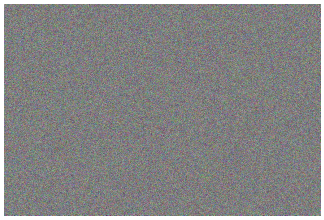
(b) Histograma de la banda roja.



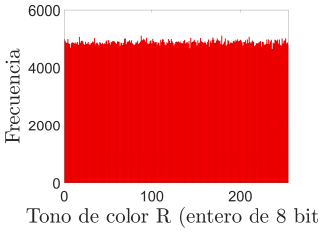
(c) Histograma de la banda verde.



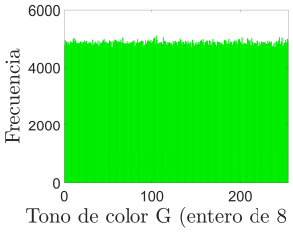
(d) Histograma de la banda azul.



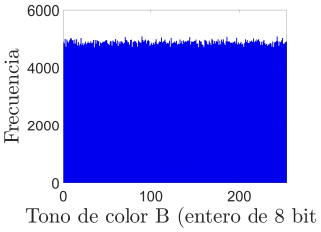
(e) Criptograma.



(f) Histograma de la banda roja.

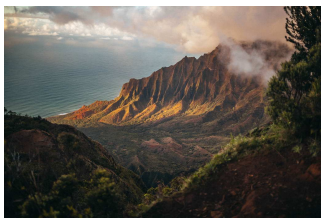


(g) Histograma de la banda verde.

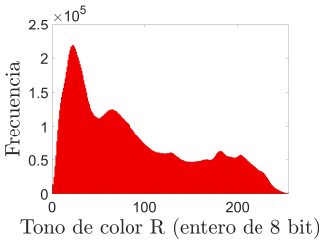


(h) Histograma de la banda azul.

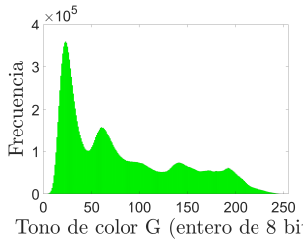
Figura 10: Resolución de 912 × 1368



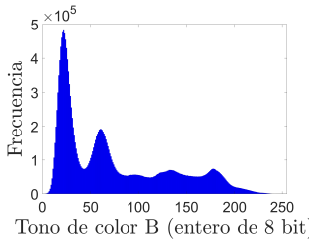
(a) Imagen original.



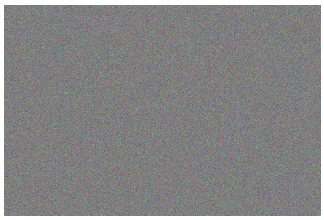
(b) Histograma de la banda roja.



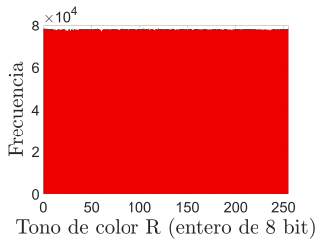
(c) Histograma de la banda verde.



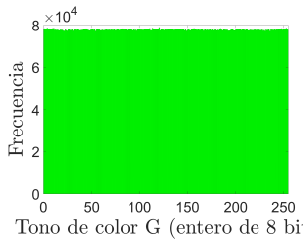
(d) Histograma de la banda azul.



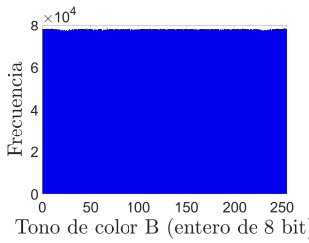
(e) Criptograma.



(f) Histograma de la banda roja.



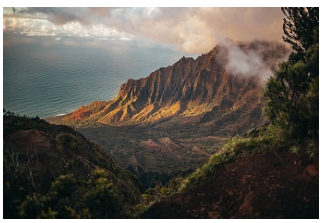
(g) Histograma de la banda verde.



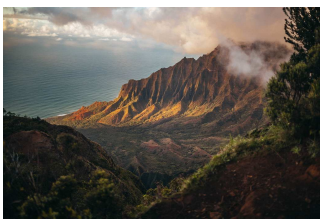
(h) Histograma de la banda azul.

Figura 11: Resolución de 3648 × 5472

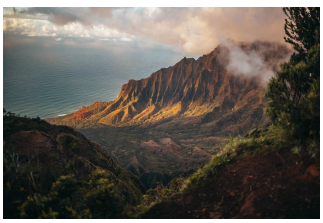
(a) Imagen recuperada 170p.



(b) Imagen Recuperada 350p.



(c) Imagen recuperada 1k.



(d) Imagen recuperada 4k.

Figura 12: Recuperación de las imágenes a través de la clave generada.

4.3. *Validación de criptogramas mediante entropía de la información*

La tabla 5 muestra los resultados de calcular la entropía de la información a los criptogramas obtenidos, se observan valores muy próximos al de la total aleatoriedad.

Tabla 5: Entropía de los criptogramas obtenidos

Criptograma	R	G	B	Total
114x171	7.9909	7.9900	7.9888	7.9968
228x342	7.9973	7.9976	7.9977	7.9992
912x1368	7.9998	7.9998	7.9998	7.9999
3648x5472	7.9999	7.9999	7.9999	7.9999

4.4. *Validación de criptogramas contra ataques diferenciales NPCR y UACI*

Las pruebas NPCR y UACI están diseñadas para probar la cantidad de píxeles cambiantes y la cantidad de intensidad modificada promedio entre dos criptogramas, lo anterior para determinar si el método de encriptamiento resiste criptoanálisis de ataque diferencial. En términos estadísticos, N_0,05, N_0,01 y N_0,001 son los valores críticos para la prueba NPCR en el encriptamiento de imágenes con un $\alpha = 0{,}05$, $\alpha = 0{,}01$ y $\alpha = 0{,}001$ respectivamente (Wu *et al.*, 2011). Considerando los criptogramas C_1 y C_2, si los valores obtenidos al realizar la prueba NPCR son menores a N_α, entonces se rechaza la hipótesis nula fallando la prueba y C_1 y C_2 se dice que no son criptogramas pseudoaleatorios y que el método no resiste ataque diferencial. La tabla 6 muestra los resultados de la prueba en diferentes resoluciones de imagen, se observa que el método implementado en distintas resoluciones acepta la hipótesis nula y los criptogramas son pseudoaleatorios con un nivel de significancia α resistiendo criptoanálisis de ataque diferencial.

En la tabla 7, se muestran los resultados de la prueba UACI con niveles de significancia U_α compuesto por U_α^- y U_α^+ considerando los valores críticos correspondientes (Wu *et al.*, 2011). Si los valores obtenidos están fuera del intervalo (U_α^-, U_α^+), entonces, se considera que C_1 y C_2 fallan la prueba y se dice que los criptogramas C_1 y C_2 no son pseudoaleatorios y que el método no resiste criptoanálisis con ataque diferencial. Se observa que los criptogramas C_1 y C_2 pasan la prueba y el método de cifrado resiste criptoanálisis de ataque diferencial.

Para las pruebas, se consideraron dos criptogramas C_1 y C_2 obtenidos al encriptar la imagen original con series cifrantes distintas, ambas generadas con el PRNG propuesto. Las imágenes originales son: lena.jpg con resolución de 250×250 píxeles, mandril.jpg con resolución de 512×512 píxeles y house.jpg con resolución de 1024 × 1024 píxeles.

Tabla 6: Tabla de resultados de la prueba NPCR.

NPCR - C1 y C2 - 256 × 256 píxeles			
Valores críticos N_α	R 99.6121 %	G 99.6156 %	B 99.6142 %
$\alpha = 0{,}05$ 99.5693 %	Pasó	Pasó	Pasó
$\alpha = 0{,}01$ 99.5893	Pasó	Pasó	Pasó
$\alpha = 0{,}001$ 99.5994 %	Pasó	Pasó	Pasó
NPCR - C1 y C2 - 512 × 512 píxeles			
Valores críticos N_α %	R 99.6063 %	G 99.6063 %	B 99.6215 %
$\alpha = 0{,}05$ 99.5527 %	Pasó	Pasó	Pasó
$\alpha = 0{,}01$ 99.5810 %	Pasó	Pasó	Pasó
$\alpha = 0{,}001$ 99.5952 %	Pasó	Pasó	Pasó
NPCR - C1 y C2 - 1024 × 1024 píxeles			
Valores críticos N_α %	R 99.6121 %	G 99.6156 %	B 99.6142 %
$\alpha = 0{,}05$ 99.5994	Pasó	Pasó	Pasó
$\alpha = 0{,}01$ 99.5952	Pasó	Pasó	Pasó
$\alpha = 0{,}001$ 99.5906	Pasó	Pasó	Pasó

Tabla 7: Tabla de resultados de la prueba UACI.

UACI - C1 y C2 - 256 × 256 píxeles			
Valores críticos (U_α^-, U_α^+) %	R 33.4330 %	G 33.5111 %	B 33.4780 %
$\alpha = 0{,}05$ (33.2824,33.6447)	Pasó	Pasó	Pasó
$\alpha = 0{,}01$ (33.2255,33.7016)	Pasó	Pasó	Pasó
$\alpha = 0{,}001$ (33.1594,33.7677)	Pasó	Pasó	Pasó
UACI - C1 y C2 - 512 × 512 píxeles			
Valores críticos (U_α^-, U_α^+) %	R 33.4930 %	G 33.5332 %	B 33.5183 %
$\alpha = 0{,}05$ (33.3730,33.5541)	Pasó	Pasó	Pasó
$\alpha = 0{,}01$ (33.3445,33.5826)	Pasó	Pasó	Pasó
$\alpha = 0{,}001$ (33.3115,33.6156)	Pasó	Pasó	Pasó
UACI - C1 y C2 - 1024 × 1024 píxeles			
Valores críticos (U_α^-, U_α^+) %	R 33.4765 %	G 33.4421 %	B 33.4973 %
$\alpha = 0{,}05$ (33.4183,33.5088)	Pasó	Pasó	Pasó
$\alpha = 0{,}01$ (33.4040,33.5231)	Pasó	Pasó	Pasó
$\alpha = 0{,}001$ (33.3875,33.5396)	Pasó	Pasó	Pasó

5. Conclusiones

Con el método propuesto para generar un PRNG óptimo para la encriptación de imágenes, se generaron series pseudoaleatorias para ser utilizadas como series cifrantes en métodos criptográficos modernos. El método criptográfico propuesto, produce criptogramas de imágenes a color totalmente ininteligibles y con una entropía de la información muy próxima a la de la total aleatoriedad, siendo el resultado de entropía mayor a medida que incrementa el tamaño de la imagen. Se comprobó, mediante las pruebas NPCR y UACI, que el método propuesto de encriptamiento de imágenes a color, resiste criptoanálisis mediante ataque diferencial de manera satisfactoria. Finalmente, se concluye que las series cifrantes generadas con el PRNG basado en programación genética propuesto e implementado en el método para encriptado y desencriptado de imágenes a color, puede ser utilizado en sistemas criptográficos modernos.

Referencias

Aman, A. H. M., Yadegaridehkordi, E., Attarbashi, Z. S., Hassan, R., y Park, Y.-J. (2020). A survey on trend and classification of internet of things reviews. *Ieee Access*, 8:111763–111782.

Balbás Gutiérrez, D. (2019). *Ataques al criptosistema RSA*. Tesis doctoral, Universidad de Cantabria.

Borda, M. y Tornea, O. (2010). Dna secret writing techniques. pp. 451–456.

Cisco (2022). Cisco annual internet report. Technical report.

Hernandez, J. C., Seznec, A., e Isasi, P. (2004). On the design of state-of-the-art pseudorandom number generators by means of genetic programming. En *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753)*, volumen 2, pp. 1510–1516. IEEE.

Koblitz, N., Menezes, A., y Vanstone, S. (2000). The state of elliptic curve cryptography. *Designs, codes and cryptography*, 19:173–193.

Kösemen, C., Dalkiliç, G., y Aydin, Ö. (2018). Genetic programming-based pseudorandom number generator for wireless identification and sensing platform. *Turkish Journal of Electrical Engineering and Computer Sciences*, 26(5):2500–2511.

Lambić, D. (2020). A new discrete-space chaotic map based on the multiplication of integer numbers and its application in s-box design. *Nonlinear Dynamics*, 100(1):699–711.

Lamenca-Martinez, C., Hernandez-Castro, J. C., Estevez-Tapiador, J. M., y Ribagorda, A. (2006). Lamar: A new pseudorandom number generator evolved by means of genetic programming. En *Parallel Problem Solving from Nature-PPSN IX: 9th International Conference, Reykjavik, Iceland, September 9-13, 2006, Proceedings*, pp. 850–859. Springer.

Lema Andrango, A. R. (2022). Estudio introductoria a la criptografía cuántica. B.S. thesis, Quito: EPN, 2022.

Murillo-Escobar, M. A., Meranza-Castillón, M. O., López-Gutiérrez, R. M., y Cruz-Hernández, C. (2019). Suggested integral analysis for chaos-based image cryptosystems. *Entropy*, 21(8):815.

Pousa, A. (2011). *Algoritmo de cifrado simétrico AES*. Tesis doctoral, Universidad Nacional de La Plata.

Sáez de Buruaga Brouns, J. (2022). *Integración de Tecnologías de Distribución Cuántica de Claves (QKD) en Protocolos Criptográficos Clásicos*. Tesis doctoral, Telecomunicacion.

Shannon, C. E. (1949). Communication theory of secrecy systems. *Bell. Sys Tech. J.*, 28:656–715.

Teh, J. S., Alawida, M., y Sii, Y. C. (2020). Implementation and practical problems of chaos-based cryptography revisited. *Journal of Information Security and Applications*, 50:102421.

Tutueva, A. V., Nepomuceno, E. G., Karimov, A. I., Andreev, V. S., y Butusov, D. N. (2020). Adaptive chaotic maps and their application to pseudo-random numbers generation. *Chaos, Solitons & Fractals*, 133:109615.

Vargas, Y. T. M. y Mnedez, H. A. M. (2015). Comparación de algoritmos basados en la criptografía simétrica des, aes y 3des. *Mundo Fesc*, 5(9):14–21.

Wu, Y., Noonan, J. P., y Agaian, S. (2011). Npcr and uaci randomness tests for image encryption. *Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT)*, 1(2):31–38.