DECLARACION DE AUTORIA Y ORIGINALIDAD DE LA TESIS

Yo, Jesús Agustín Aboytes González, estudiante del Posgrado en Ciencias Aplicadas de la Facultad de Ciencias de la Universidad Autónoma de San Luis Potosí, como autor/(a) de la tesis "Diseño e implementación de una caja de sustitución al sistema de cifrado CSAC", declaro que la tesis es una obra original, inédita, auténtica, personal, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales vigentes que protegen los derechos de autor y de propiedad intelectual e industrial. Las ideas, doctrinas, resultados y conclusiones a los que he llegado son de mi absoluta responsabilidad.

Resumen

Este trabajo se dedico al diseño, validación y acoplamiento de una caja de sustitución o S-box, al sistema de cifrado CSAC. Debido a que al efectuar nuevas pruebas contra criptoanalisis a dicho sistema, se detecto una vulnerabilidad en la seguridad y para agregar mas fortaleza al sistema CSAC, se busco una solución que no afectara en gran medida al rendimiento del sistema CSAC al momento de procesar la información que es la S-box. Por lo que se proponen dos nuevas configuraciones del sistema CSAC y una nueva aplicación a la S-box, dentro de un sistema VMEI.

Abstract

This work was dedicated to the design, validation and coupling of a substitution box or S-box, to the CSAC encryption system. Due to the fact that new tests against cryptanalysis were performed on this system, a security vulnerability was detected and to add more strength to the CSAC system, a solution was sought that would not affect to a great extent the performance of the CSAC system when processing the information, which is the S-box. Therefore, two new configurations of the CSAC system and a new application to the S-box, within a VMEI system, are proposed.

Agradecimientos

Especialmente a mi madre, mi esposa y mi hija, por la confianza y el apoyo para poder realizar mis estudios de posgrado.

A mis asesores, el Dr. José Salomé Murguía Ibarra y a la Dra. Marcela Mejía Carlos por su paciencia y consejos.

Al CONACYT por el apoyo económico otorgado para la realización de este trabajo.

A todos mis profesores que me impartieron clases y aconsejaron a lo largo de mi vida académica, ya que con su profesionalismo, ayudaron al desarrollo de mi formación académica y profesional.

A mi familia, mi suegra, cuñado y amigos, por sus buenos consejos y por el apoyo incondicional, ya que ustedes son parte importante de esto.

Al IICO tanto a su planta académica como y administrativa, por todo el apoyo brindado en mis estudios.

Al centro de información del IICO por sus servicios y labor de fomentar la investigación.

Por Al proyecto de CONACYT de Ciencia Básica CB2017-2018-A1-S-45697, denominado .Estudio y análisis de señales fisiológicas e imágenes cifradas".

A la compañía INTEL por la donación de la tarjeta DE2-115 Intel a través del "FPGA University Program".

Dedicatoria

En este trabajo quedan plasmados años de estudio e investigación. Y dicho esfuerzo se lo dedico a cada una de las personas que me motivo a seguir en los momentos de mas cansancio. A mi madre, a mi esposa y a mi hija. A aquellos que ya no están conmigo, mi padre y mis abuelos. A mi nueva familia que sin importar nada siempre me apoyaron en este sueño de convertirme en Doctor, Doña Yola y Juancho. Y finalmente al resto de mi familia por su apoyo y consejos incondicionales.

Por ultimo quiero dedicar este proyecto a mis asesores la Dra. Marcela y el Dr. Salomé, que día a día me inspiraron con su conocimiento, dedicación y comprensión. A mis compañeros y amigos de toda la vida, que gracias a sus ideas, cosecharon ese amor que siento por la ciencia.

"El día que dejemos de soñar, la entropía del universo habrá terminado..."

J.A. Aboytes-Gonzaléz.

Índice general

Índice de figuras

Índice de tablas

1

Introducción

Desde su origen, la humanidad ha tenido la necesidad de comunicarse para compartir información, pero a la par también ha surgido la necesidad de ocultar cierto tipo de información, ya que de ser divulgada podría suponer un riesgo tanto para el individuo que la comparte como para la población a la que pertenece. Es por esto que la humanidad en su afán de ocultar información ha desarrollado una serie de herramientas tanto mecánicas como matemáticas que con el paso del tiempo se han hecho más complejas. A esta nueva área del conocimiento se le denominó **Criptografía** que literalmente tomando sus etimologías griegas significa *el arte de escribir algo que no se puede entender* [1].

A través de los años las técnicas para ocultar la información han ido evolucionando hasta volverse muy complejas, utilizando transformaciones matemáticas cada vez más abstractas [2], hasta el punto de necesitar computadoras más eficientes para poder procesar la información en un menor tiempo. Un ejemplo de esto se dio durante la Segunda Guerra Mundial, cuando Alan Turing usando las primeras computadoras fue capaz de romper la codificación de *Enigma* una maquina usada por el ejército Nazi para ocultar su información y cuya configuración era modificada antes de que los aliados fueran capaces de analizar los suficientes datos como para romper su codificación, con lo que se acorto la guerra y se salvaron millones de vidas [3].

En la actualidad muchas de las herramientas matemáticas que son utilizadas para ocultar la información se enfocan en procesos matemáticos que se usan para modelar fenómenos de orden caótico [4, 5], ya que al ser procesos demasiado sensibles pueden ocultar la información de una forma más abstracta sin perder la capacidad de poder recuperar la información que se ocultó. A las herramientas que se encargan de ocultar la información, se les denomina sistemas de cifrado o criptosistemas, mientras que al proceso de ocultar la información se le conoce comúnmente en la lengua española como encriptación, debido a que en inglés este proceso se denomina como *encryption*. Sin embargo, la palabra encriptación no está definida por la Real Academia Española (RAE), por lo que el termino correcto es ?cifrado?.

Hoy en día existen infinidad de sistemas de cifrado, cada uno con características únicas que los hacen buenos o malos para cifrar cierto tipo de datos, por esta razón han ideado varias pruebas para medir la calidad y desempeño de los sistemas de cifrado. Y es así como los países en su afán de crear un mundo globalizado, han buscado un estándar para cifrar la información y se han encargado de seleccionar un criptosistema cuyo cifrado de datos sea robusto ante cualquier tipo de información, de esta forma el sistema AES (Advanced Encryption Standar) [6] se ha convertido en el sistema de cifrado oficial del ?mundo?.

A la par que los sistemas de cifrado se han fortalecido, las técnicas que intentan romper o vulnerabilizar la seguridad de los sistemas de cifrado se han sofisticado más. El éxito para romper la codificación de un criptosistema se debe en gran medida al análisis de grandes volúmenes de datos. Al conjunto de todas las técnicas que se emplean para analizar este tipo de datos se les denomina como *Criptoanálisis*. Por tal motivo se deben de generar sistemas de cifrado que puedan ser actualizados sin sacrificar seguridad o velocidad, con el propósito de corregir vulnerabilidades que no se tenían en cuenta al momento de ser diseñados. Existen una gran cantidad de ataques y/o pruebas para evaluar la eficiencia de los sistemas de cifrado, así como de algunos de sus componentes. Por ejemplo, se cuenta con el histograma, la correlación cruzada y adyacente, entropía, sensibilidad a la llave, así como el NPCR (tasa o razón de cambio de píxeles) que mide y la UACI (intensidad de cambio promedio unificado), ambas pruebas

usadas extensamente para verificar la fortaleza del cifrado contra criptoanálisis diferencial, se aplica a imágenes o texto plano que por lo general solo varían en un bit [38] Asimismo, se cuenta con pruebas estadísticas bien establecidas para evaluar la aleatoriedad de los generadores de llaves, como lo son las de la NIST [7, 8], o bien para las S-boxes o cajas de sustitución, que son componentes no lineales de los sistemas de cifrado que consisten en tablas cuya tarea es permutar los datos de entrada con los de salida restando claridad al texto plano [32], tal como no-linealidad, independencia de bits, efecto avalanche, entre otros [9]. Con base al gran número de pruebas y/o ataques, no es fácil tener una evaluación completa de los sistemas de cifrado o de sus componentes, además de que muchos sistemas son diseñados para cifrar cierto de tipo de información. Por ejemplo, el modo ECB del sistema AES que tenia una vulnerabilidad al cifrar imágenes debido a la alta correlación que existe en ese tipo de información, así como de la vulnerabilidad que presenta ante ataques criptográficos diferenciales[10, 23].

En este trabajo nos concentraremos en mejorar el sistema CSAC (Cifrado por Sincronización de Autómatas Celulares) [12, 23, 20, 22] para detectar posibles vulnerabilidades al aplicar ataques de criptoanálisis. El sistema CSAC usa autómatas celulares como kernel para cifrar la información, debido a que son sistemas dinámicos discretos no lineales de carácter caótico, lo cual ha resultado ser útil para varios sistemas de cifrado. Al aplicar ataques de criptoanálisis al sistema CSAC, tal como el UACI o el Chosen Plain Image Attack [23], los resultados obtenidos han sido satisfactorios, así como en las pruebas contempladas en [23]. Con la finalidad de contemplar más pruebas para evaluar el sistema de cifrado CSAC, se considero el Criterio Estricto de Avalancha (SAC por sus siglas en inglés) postulado en [15]. Esta prueba se usa para cuantificar probabilísticamente la diferencia que existe entre dos bloques de texto cifrado cuando proceden de dos bloques de texto claro con la diferencia de un solo bit. Los resultados obtenidos de la aplicación del SAC al sistema CSAC no son tan favorables como el valor que la literatura marca como aceptable, es por esto que se buscó una opción que previniera esta vulnerabilidad y a la vez no afectara la velocidad de procesamiento ni la seguridad del sistema CSAC. La opción por la que se optó fue la de usar una caja de sustitución o S-box por sus siglas

en inglés (Substitution box) e insertarla en el sistema de cifrado para repetir las pruebas. Esta vez el resultado fue favorable para la prueba SAC, por lo que ahora una cuestión importante es diseñar e implementar una S-box que conserve la esencia del sistema CSAC y que esté al nivel de las que se usan en el sistema AES u otros sistemas de cifrado modernos.

La estructura del documento de tesis es el siguiente. En el Capítulo 2 se presenta el marco teórico que auxilia este trabajo, desde los preliminares del sistema de cifrado CSAC hasta las herramientas matemáticas para la generación de la S-box, como lo son los campos finitos de Galois. En el Capítulo 3 se presenta la implementación y diseño de una S-box basada en la evolución de autómatas celulares y complementada con una transformación no lineal sobre los campos finitos de Galois, así como el análisis de seguridad contra ataques de criptoanálisis. Mientras que en el Capítulo 4 se presenta el uso de la S-box para la reestructuración del sistema CSAC y su respectiva evaluación al aplicar pruebas de seguridad, así como una aplicación de la S-box. Finalmente, en el Capítulo 5 se describen los resultados obtenidos en este trabajo y se bosquejan algunas líneas de trabajo futuro.

2

Marco Teórico

En este Capítulo se definen y se describen algunos conceptos importantes, tales como los autómatas celulares, el fenómeno de sincronización de los autómatas celulares así como la estructura matricial de las componentes que conforman el sistema de cifrado CSAC (Encryption by Synchronization of Cellular Automaton). De igual modo se describen algunos conceptos básicos requeridos para la generación de la S-box propuesta en este trabajo.

2.1 Autómatas Celulares

Los Autómatas Celulares (AC) fueron propuestos originalmente por John Von Neumann y Stanislaw Ulam en la década de los 40 [16] y fueron definidos en [17] como sistemas dinámicos en los cuales el tiempo y el espacio son representados de forma discreta, lo cual permite que sean modelados gráficamente utilizando un arreglo de celdas que pueden tomar un número finito de estados. Una de sus aplicaciones principales es modelar el comportamiento complejo de algunos sistemas naturales, en donde se involucran iteraciones locales. Como, por ejemplo, el juego de la vida [18], en donde un par de condiciones iniciales (elementos de una población) evolucionan en el tiempo siguiendo un conjunto de reglas, mostrando el auge o declive

de una población. Para representar el comportamiento de estos sistemas dinámicos discretos, se hace uso de una serie de simples reglas locales de evolución con el motivo de no hacer uso de herramientas matemáticas más complejas tales como un sistema de ecuaciones diferenciales.

Las reglas locales de evolución de los AC fueron postuladas por Stephen Wolfram en 1984 [19], en donde se representa un Autómata Celular Básico (ACB) como un AC de dimensión uno y con vecindario de radio 1. Para calcular el estado siguiente se toman en cuenta los estados de las celdas laterales junto con el que tiene la celda que se quiere calcular un tiempo antes, como se muestra en la Figura 2.1. Al ser un ACB los valores o estados de sus celdas serán de orden booleano, por lo que sólo se podrán tomar los valores $\mathbb{Z}_2 = 0, 1$. En el juego de la vida esto significaría que las celdas que contengan el valor de uno serán elementos vivos, mientras que los de valor de cero indicarán elementos muertos.

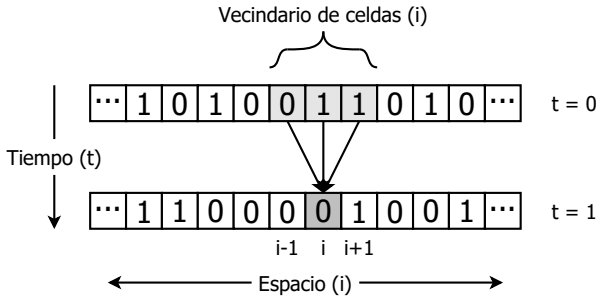

Figura 2.1: Autómata Celular Básico.

El vecindario del ACB está compuesto por tres celdas y el máximo posible de combinaciones que se podrán generar son 8, lo que daría como resultado $2^8 = 256$ posibles reglas de evolución, el ejemplo más recurrente de estas reglas de evolución es la regla 90, que realiza una operación XOR sobre las celdas laterales del estado que se quiere calcular. En la Tabla 2.2 se explica cómo se constituyen las reglas 0, 90 y 165 siendo ésta última la regla complementaria de la regla 90.

	Estados del Vecindario del ACB							
Regla Local	111	110	101	100	011	010	001	000
Regla 0	0	0	0	0	0	0	0	0
Regla 90	0	1	0	1	1	0	1	0
Regla 155	1	0	1	0	0	1	0	1

Figura 2.2: Ejemplo de las distintas reglas de evolución.

Si se analiza esta tabla, se puede verificar que para la regla 0, sólo se necesita poner ceros en el siguiente estado del ACB. Si se observa, los resultados de la regla 90, sólo se toman en cuenta los valores de las celdas laterales, realizando una operación XOR para obtener el valor de la celda en el siguiente tiempo, lo que se traduce en la ecuación $x_i^{t+1} = x_{i+1}^t$ xor x_{i-1}^t, en donde x representa la celda que se quiere calcular, i la localidad de la celda y t el paso en el tiempo. La regla 165 es la operación XNOR con las mismas celdas laterales. Haciendo una analogía de las reglas locales con el mundo digital, estas serían el equivalente al algoritmo a seguir por un programa.

Gracias a la dinámica de los AC, estos pueden usarse en el campo de la criptografía. Y debido a que la mayoría de la información que se maneja es digital, se pueden utilizar los ACB, los cuales consumen muy pocos recursos de cómputo y permiten tener mayores velocidades de procesamiento y menor almacenaje.

2.2 Sistema de cifrado CSAC

El sistema CSAC es un sistema de cifrado basado en la sincronización y evolución de los autómatas celulares. Su nombre proviene de las siglas de Cifrado por Sincronización de Autómatas Celulares y es un sistema de cifrado simétrico en el cual, la misma llave de cifrado

se emplea para cifrar o descifrar la información. Este sistema ha ido evolucionando en su forma de operación agregando o quitando funciones [20, 12, 23, 22] con el propósito de hacerse mas robusto contra los ataques de criptoanálisis sin sacrificar en gran medida la velocidad de operación.

El sistema CSAC puede cifrar o descifrar la información, y cada uno de estas fases cuenta con tres módulos. Por ejemplo, en la etapa de cifrar el primer módulo es el que realiza el pre-procesamiento de la información que vamos a cifrar. El siguiente módulo realiza la generación de llaves, mientras que el último módulo realiza el cifrado de la información anterior. La etapa de descifrar tiene los módulos similares a la etapa anterior, pero realiza la operación inversa, ver Figura 2.3. A la configuracion del sistema CSAC que cuenta con los tres módulos se le denominara como **Versión 2** mientras que a la configuracion que carece del módulo de pre procesamiento se le llamara **Versión 1**. En particular, como se verá en el resto de esta sección los módulos del sistema se diseñan acorde a las familias de permutaciones Ψ, Φ y la función h, que se obtienen gracias a la unidad básica de cifrado que a continuación se describe.

2.2.1 Sincronización de autómatas celulares

Dos sistemas dinámicos acoplados se sincronizan si después de un largo periodo de tiempo, sus comportamientos consiguen estar arbitrariamente cerca. Para el caso de los autómatas celulares existe la sincronización como resultado de un acoplamiento no trivial [20], y ocurre cuando un conjunto determinado de coordenadas acopladas es copiada de uno de los sistemas, autómata celular manejador, al sistema de respuesta que será el autómata celular de respuesta. Lo anterior indica que, para cada paso de tiempo, ambos sistemas evolucionan de acuerdo a la misma regla, hasta que las coordenadas acopladas del autómata celular manejador son copiadas a las correspondientes coordenadas del autómata celular de respuesta. En la Figura 2.4 se muestra un ejemplo que ilustra el acoplamiento unidireccional en autómatas celulares.

En la Figura 2.4 la secuencia de acoplamiento es

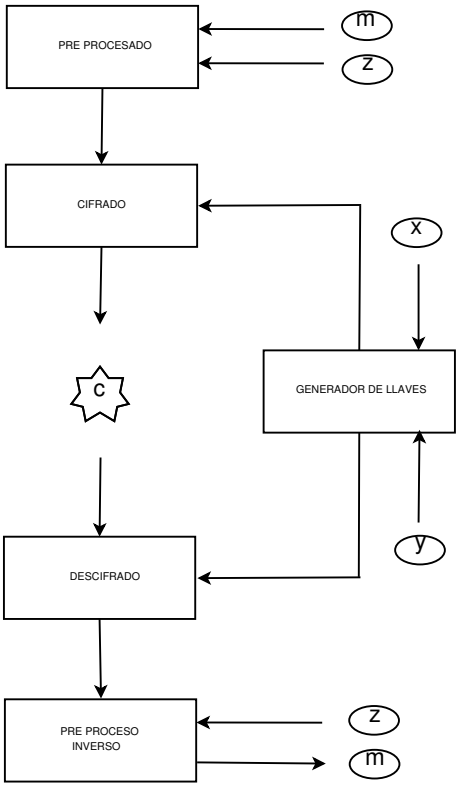

Figura 2.3: Diagrama a bloques que ilustra la funcionalidad del sistema CSAC.

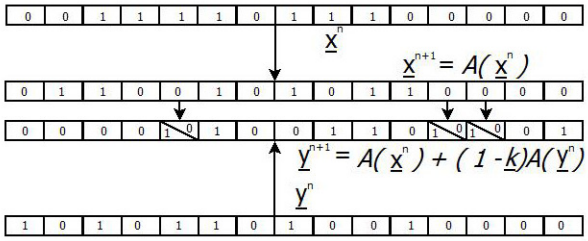

Figura 2.4: Ilustración de la sincronización de autómatas celulares.

$$\underline{\kappa} = (0,0,0,0,1,0,0,0,0,0,0,1,1,0,0),$$

y las coordenadas acopladas son κ_4, κ_{11} y κ_{12}. Los estados en el tiempo t son $\underline{x}^t$ y $\underline{y}^t$

$$\underline{x}^t = (0,0,1,1,1,1,0,1,1,1,0,0,0,0,0),$$

$$\underline{y}^t = (1,0,1,0,1,1,0,1,0,0,1,0,0,0,0),$$

por lo que el estado del autómata manejador en el tiempo $t+1$ es

$$\underline{x}^{t+1} = \mathscr{A}(\underline{x}^t) = (0,1,1,0,0,1,0,1,0,1,1,0,0,0,0).$$

El estado del autómata respuesta en el tiempo $t+1$ es ilustrado de la siguiente forma:

$$\mathscr{A}(\underline{y}^t) = (0,0,0,0,1,1,0,0,1,1,0,1,0,0,1)$$

$$\underline{\kappa} = (0,0,0,0,1,0,0,0,0,0,0,1,1,0,0)$$

$$1 - \underline{\kappa} = (1,1,1,1,0,1,1,1,1,1,1,0,0,1,1)$$

$$\kappa\mathscr{A}(\underline{x}^t) = (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

$$(1 - \underline{\kappa})\mathscr{A}(\underline{y}^t) = (0,0,0,0,0,1,0,0,1,1,0,0,0,0,1)$$

$$(1 - \underline{\kappa})\mathscr{A}(\underline{y}^t) + \kappa\mathscr{A}(\underline{x}^t) = (0,0,0,0,0,1,0,0,1,1,0,0,0,0,1)$$

Por lo tanto un par acoplado $(\mathscr{A},\underline{\kappa})$ sincroniza cuando la diferencia $\underline{z}^t = \underline{y}^t$ - $\underline{x}^t$ de los vectores de estado $\underline{x}^t$, $\underline{y}^t$ (correspondientes al autómata manejador y al autómata de respuesta, respectivamente) se iguala al vector nulo $\underline{0} = (...,0,0,0,...)$ después de un cierto número de pasos en el tiempo t [21].

2.2.2 Unidad básica de cifrado

Debido al fenómeno de sincronización de los autómatas celulares, la unidad básica de cifrado (UBC) se encarga de implementar los principales elementos del sistema CSAC. Básicamente la UBC se puede definir como un patrón cuadrado de $N \times N$ de palabras binarias de longitud N que evolucionan en el tiempo, es decir, $(x_1^0,x_1^1,...,x_1^{N-1}),...,(x_N^0,x_N^1,...,x_N^{N-1})$. Las principales

palabras binarias son las que rodean la UBC: (a) en el lado izquierdo se encuentran las palabras $\mathbf{x} = (x_0^0, x_0^1, ..., x_0^{N-1})$ y $\mathbf{y} = (x_1^0, x_1^1, ..., x_1^{N-1}, x_1^N)$, las cuales conforman las semillas iniciales del generador de números aleatorios; (b) en el lado derecho se encuentra la secuencia cifrada $\mathbf{c} = (x_{N+1}^0, x_{N+1}^1, ..., x_{N+1}^{N-1})$, (c) en la parte superior se encuentra la palabra aleatoria resultante $\mathbf{t} = (x_2^0, x_3^0, ..., x_{N+1}^0)$; (d) finalmente en la parte inferior se tiene el texto plano $\mathbf{m} = (x_1^N, x_2^N, ..., x_N^N)$, como se muestra en la Figura 2.5

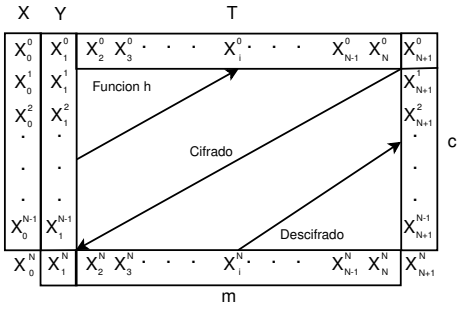

Figura 2.5: Unidad Básica de Cifrado.

Para obtener estas palabras se evoluciona en el tiempo de acuerdo a la regla 90, desde $t = 0$ hasta $t = N = 2^k - 1$, donde $k = 1, 2, 3...$ y los índices horizontales $i \neq 0$ e $i \neq N+1$, que van desde x_0^t hasta x_{N+1}^t, para todo t. De manera general, para calcular las palabras o secuencias binarias previas, de cada fase (cifrado/descifrado), es necesario que en la UBC el autómata celular este iterando en el tiempo hacia adelante o hacia atrás, dando lugar a las permutaciones del sistema.

2.3 Enfoque matricial del sistema CSAC

En el trabajo [12] se presentó una nueva forma de implementar el sistema CSAC y donde se implementaron en forma matricial las familias de permutaciones y el generador de números aleatorios (PRNG por sus siglas en inglés), dando al sistema una mayor flexibilidad para su implementación y mejora en seguridad.

Para generar el texto cifrado se usa la permutación Ψ que como se observa en la Fig. **??** el

modulo de cifrado esta en función de las matrices $\mathbf{P}_N$ y $\mathbf{Q}_N$, que interactuan con los vectores del texto pre procesado ($\hat{m}$) y la llave de cifrado (x) y dicha operación la podemos ver en la siguiente ecuación

$$\mathbf{c} = \Psi_x(m) = [(\mathbf{P}_N \times \mathbf{x}) + (\mathbf{Q}_N \times \mathbf{m})] \qquad \text{mód} \quad 2, \tag{2.1}$$

en donde $\mathbf{m}$, $\mathbf{c}$ y $\mathbf{x}$ tienen dimensiones $N \times 1$, y las dos matrices $\mathbf{P}_N$ y $\mathbf{Q}_N$, son matrices cuadradas de orden N, con $N = 2^n - 1$, para $n = 1,2,3,....$ La matriz triangular superior $\mathbf{P}_N$ se genera inicialmente a partir del vector $\mathbf{p} = [p_1, p_2, ..., p_N]$, que constituye la primera fila, y los componentes con el índice posición $j = (2^n + 1) - 2^{i+1}, i = 0,1,2,...,(n-1)$, tienen el valor de 1, y los demás son 0. Las $(N-1)$ filas se generan aplicando un desplazamiento a la derecha de una posición de la fila anterior con un cero como su primer valor. De la misma manera, la matriz triangular inferior $\mathbf{Q}_N$ puede ser generada inicialmente a partir del vector $\mathbf{a} = [a_1, 0, ..., 0]$, donde la componente a_1 tiene el valor de 1, y N es el número de bits. Por lo que, $\mathbf{a}$ es un vector con N componentes, y constituye la primera fila de la matriz $\mathbf{Q}_N$. Las filas $(N-1)$ son generadas aplicando la regla 90 de los CA a la fila anterior, fijando valores de frontera de cero a los lados izquierdo y derecho. Para $N = 7$ las matrices $\mathbf{Q}_N$ y $\mathbf{P}_N$ tienen las siguientes formas

$$\mathbf{Q}_7 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}; \quad \mathbf{P}_7 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2.2}$$

Para la otra familia de permutaciones Φ, se observa en la Fig. **??** que en el módulo de descifrado las matrices $\mathbf{R}_N$ y $\mathbf{T}_N$, son las que procesan a los vectores que contienen el texto cifrado ($\mathbf{c}$) y a la

llave de cifrado (**x**). La implementación de la matriz de permutación inversa tiene una estructura similar a la de 2.1, que es,

$$\mathbf{m} = \Phi_x(c) = [(\mathbf{R}_N \times \mathbf{x}) + (\mathbf{T}_N \times \mathbf{c})] \qquad \text{mód} \quad 2, \tag{2.3}$$

donde las secuencias y matrices tienen las mismas dimensiones que las que se usan en el proceso de cifrado. $\mathbf{R}_N = [-\mathbf{Q}_N^{-1}\mathbf{P}_N] \qquad \text{mód} \quad 2$, y $\mathbf{T}_N$ es la matriz inversa de $\mathbf{Q}_N$. Si tomamos a $N = 7$ tendremos que $\mathbf{R}_N$ y $\mathbf{T}_N$ se ven de la siguiente manera

$$\mathbf{R}_7 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \qquad \mathbf{T}_7 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}. \tag{2.4}$$

En los trabajos [12, 23], se introdujeron dos enfoques matriciales similares al del trabajo [22] para calcular las secuencias pseudoaleatorias que generan llaves de cifrado de N bits. En los tres casos se usa la matriz $\mathbf{H}_N$ de orden $(2N-1)$. Si hacemos que $N = 7$ entonces esta matriz se vería así

$$\mathbf{H}_7 = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}. \tag{2.5}$$

para generar esta matriz se usan dos vectores filas con la siguiente forma, $\mathbf{v} = [v_1, 0, ..., v_{N+2}, ..., 0]$ y $\mathbf{w} = [0, w_2, 0, ..., w_{N+1}, 0, w_{N+3}, ..., 0]$, donde las componentes $v_1, v_{N+2}, w_2, w_{N+1}$ y w_{N+3} tienes el valor de 1, y N es el número de bits. Los vectores **v** y **w** constituyen las dos primeras filas de la matriz $\mathbf{H}_N$. Y las $(N-2)$ filas son generadas al aplicar la suma módulo 2 de las dos filas anteriores y con elementos de la fila anterior corridos una posición a la derecha.

Básicamente, la matriz $\mathbf{H}_N$ calcula la secuencia pseudo aleatoria de la llave, donde la parte baja de la matriz $\mathbf{H}_N$ calcula la retroalimentación de las semillas iniciales $[x, y]$. Por lo tanto, una vez que el número N de bits de las secuencias es definido, podemos generar secuencias pseudo aleatorias de N bits con ayuda de la matriz $\mathbf{H}_N$

$$\mathbf{U}_{K+1} = \mathbf{H}_N U_K, \quad K = 1, 2, ..., \tag{2.6}$$

donde $U_K = [xy]^T$ corresponde a las primeras entradas de las semillas iniciales del PRNG, y U_{K+1} es compuesta por las siguientes entradas del PRNG; note que U_{K+1} está formado por la llave pseudo aleatoria que se genera y la secuencia de retroalimentación.

2.4 Descripción de operación del sistema CSAC

En esta Sección se describe la manera operacional del sistema CSAC. Se debe tener en cuenta que el sistema CSAC considera cinco módulos: (1) etapa de pre proceso, (2) generador de llaves, (3) módulo de cifrado, (4) módulo de descifrado, y (5) pre proceso inverso. En la Figura 2.6 se muestra el sistema en forma de bloques. Dentro de los módulos se utilizan las matrices antes mencionadas para realizar las distintas operaciones a la información que ingresa [23], es decir, los bloques de la Figura 2.6 están acorde al enfoque matricial propuesto en la Referencia [12]. A continuación, se describe la funcionalidad del sistema CSAC en términos de los módulos.

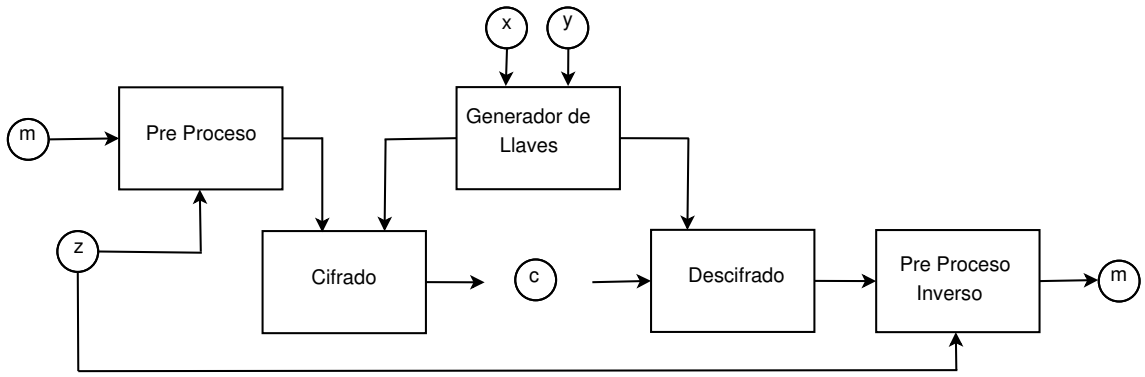

Figura 2.6: Estructura del sistema CSAC en términos de módulos.

2.4.1 Pre procesamiento

Este módulo fue diseñado para remover la correlación de la información a cifrar, lo cual evita que la información cifrada por el sistema CSAC no se comprometa bajo ciertas técnicas de criptoanálisis [23]. El funcionamiento de este módulo es muy similar al del módulo del

generador de llaves, como se verá más adelante en la Sección 2.4.2. Básicamente se utilizan las matrices $\mathbf{H}_{N_T}$, y a diferencia del módulo generador de llaves, las entradas de este módulo son el bloque de información $\mathbf{m}$ que se necesita cifrar el cual es de longitud N, mientras que la segunda secuencia Z_k la cual es de carácter aleatorio de longitud $(N+1)$. Al multiplicar $\mathbf{H}_{N_T}$ por el vector $[\mathbf{m}, Z_k]^t$, se obtiene el nuevo vector $\hat{\mathbf{m}}$ de longitud N. Para el próximo dato a pre procesar, se requiere nuevamente otro bloque de información y una secuencia aleatoria Z_{k+1}, el cual se obtiene al retroalimentar $\hat{\mathbf{m}}$ en la parte de los bits menos significativos y como el MSB se considera el LSB de $\mathbf{Z}_k$. El proceso se realiza de manera sucesiva y la estructura general del bloque se puede observar en la Figura 2.7.

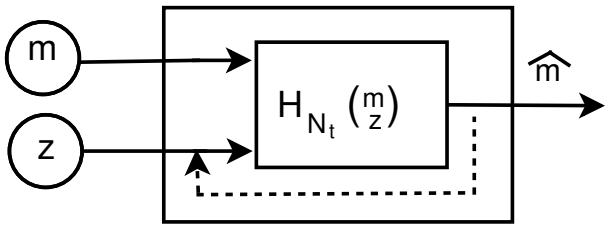

Figura 2.7: Bloque de pre procesamiento previo al cifrado.

2.4.2 Generador de Llaves

Sabemos que la función h genera cadenas de pseudo-números o secuencias de carácter aleatorio. Dentro del módulo del generador se considera la matriz $\mathbf{H}_N$, ver Ecuación (2.5). Tal módulo va a requerir dos semillas, $[X_k, Y_k]$, cuyas longitudes son N y $N+1$, respectivamente. Ambas llaves se concatenan en un nuevo vector U_k, con longitud de $(2N+1)$ elementos, coincidiendo con las dimensiones de $\mathbf{H}_N$. De esta forma se puede realizar la multiplicación entre ambos elementos obteniendo un nuevo vector $\mathbf{U}_{k+1}$ de mismas dimensiones. Los primeros N bits corresponden al número pseudo-aleatorio $\mathbf{X}_{k+1}$ siendo la llave generada del sistema; los siguientes $(N+1)$ bits corresponden al vector $\mathbf{Y}_{k+1}$, resultando la secuencia de la semilla $\mathbf{X}_k$ y en la posición del bit más significativo (MSB) está el bit menos significativo (LSB) de $\mathbf{Y}_k$. Estos nuevos elementos

(X_{k+1} y Y_{k+1}) serán las nuevas semillas $\mathbf{X}_k$ y $\mathbf{Y}_k$ para calcular la siguiente llave. En la Figura 2.8 se ilustra este procedimiento para $N = 3$.

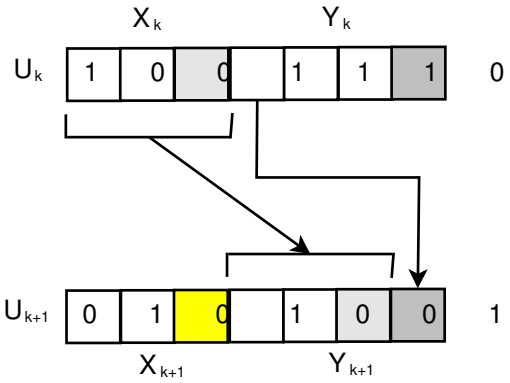

Figura 2.8: Retroalimentación de U_{k+1}.

Cabe mencionar que la retroalimentación del bit para la secuencia $\mathbf{Y}_{k+1}$ se hizo con el propósito de completar la longitud de $(N+1)$, puesto que la longitud de $\mathbf{X}_k$ es (N) y se necesita un bit para tener la longitud apropiada de $\mathbf{Y}_{k+1}$. Obviamente lo anterior se puede realizar de diferentes formas.

Debido a que cada llave $\mathbf{X}_{k+1}$ se necesita para cifrar o descifrar cada bloque de información, entonces la operación antes descrita en este módulo se repetirá hasta que todos los bloques de la información hayan sido cifrados o descifrados. La Figura 2.9 muestra el esquema iterativo del módulo generador de llaves.

2.4.3 Cifrado de la información

Este módulo tiene como secuencias de entrada el resultado del módulo de pre procesado del texto plano, $\hat{\mathbf{m}}$, y del módulo generador de llaves, $\mathbf{x}$. En el interior del módulo se realiza la multiplicación de la matriz $\mathbf{P}_N$ por $\mathbf{x}$ y $\mathbf{Q}_N$ por $\hat{\mathbf{m}}$, cuya secuencia resultante $\mathbf{c}$ se obtiene al aplicar la operación de módulo 2 a la suma de los productos matriciales anteriores. En la Figura 2.10 se observa dicho módulo.

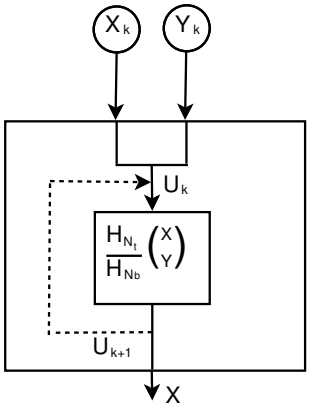

Figura 2.9: Ilustración de la iteración para generar las llaves del sistema CSAC.

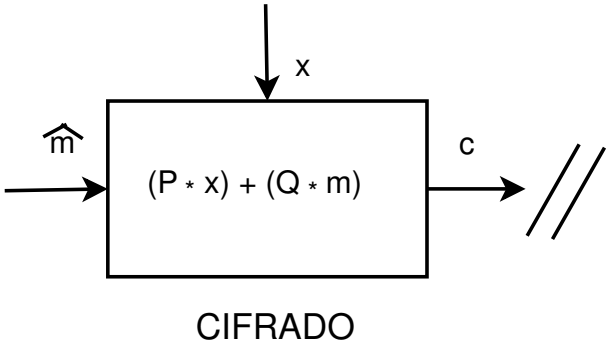

Figura 2.10: Módulo descriptivo del cifrado.

2.4.4 Descifrado de la información

Esta parte opera de manera similar al módulo anterior, pero en sentido inverso. Este módulo recibe como entradas la información cifrada **c** y la llave o secuencia aleatoria **x**. Ahora las matrices que se ocupan son $\mathbf{R}_N$ y $\mathbf{T}_N$, las cuales multiplican a las secuencias anteriores, respectivamente. La salida de este módulo es el bloque de texto $\hat{\mathbf{m}}$, el cual necesita ser tratado por el módulo de pre proceso inverso. En la Figura 2.11 se muestra su estructura.

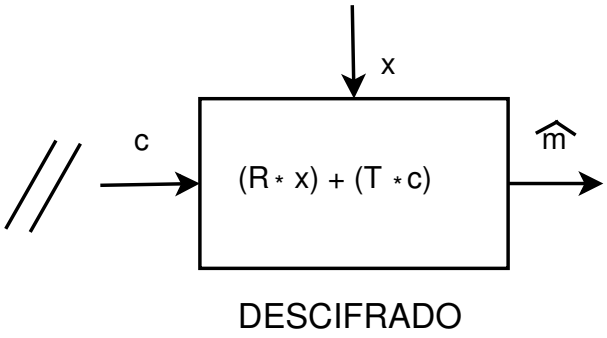

Figura 2.11: Módulo descriptivo de descifrado.

2.4.5 Pre procesado inverso

Básicamente este módulo devuelve el bloque de información $\hat{m}$ que se procesó en la etapa de cifrado a su forma original de texto plano **m**. Como secuencias de entrada se considera la secuencia $\mathbf{Z}_k$ inicial y el bloque de datos descifrado $\hat{\mathbf{m}}$. La matriz que se requiere $\mathbf{M}_N$, con dimensiones $N \times (2N+1)$, la cual se conforma por las matrices $\mathbf{Q}_N$ y $\mathbf{D}_N$ (matriz de banda de dimensiones $N \times (N+1)$), es decir,

$$\mathbf{M}_N = [\mathbf{Q}_N \mid \mathbf{D}_N]. \tag{2.7}$$

La matriz $\mathbf{D}_N$ tiene valores de uno en la primer superdiagonal y las subdiagonales k_a, con $k_a = 2^a - 1$, para $a = 0, 1, 2, ..., (n-1)$. Por ejemplo, si $N = 7$ entonces $\mathbf{D}_N$ queda como:

$$\mathbf{D}_7 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}. \tag{2.8}$$

Y por lo tanto la matriz $\mathbf{M}_N$ queda:

$$\mathbf{M}_7 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}. \tag{2.9}$$

Así al multiplicar el vector $[\hat{\mathbf{m}}, \mathbf{Z}_k]$ por la matriz $\mathbf{M}_N$, se obtiene el bloque de texto plano $\mathbf{m}$ y para el siguiente bloque de datos se retroalimenta $\hat{\mathbf{m}}$ en la siguiente secuencia $\mathbf{Z}_k$. En la Figura 2.12 se muestra el diagrama de bloques del módulo.

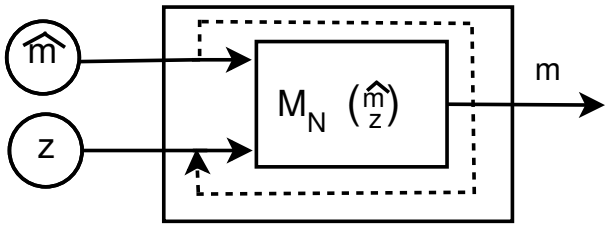

Figura 2.12: Bloque descriptivo del pre procesado inverso.

2.4.6 Operacion del CSAC

A continuacion describimos la forma en que funciona el sistema CSAC.

1.- El texto plano denominado como m, las dimensiones de este bloque puede ser del tamaño que sea si se usa el enfoque propuesto en [23] y este bloque entra al módulo de pre proceso para poder quitar correlación a la información, a este nuevo bloque de datos se le llamara $\hat{m}$.

2.- Al mismo tiempo que el bloque m es procesado, el módulo de generación de llaves crea una llave k a partir de las semillas x y y.

3.- La semilla x pasa a ser la nueva semilla y y la llave k será la nueva semilla x. Las dimensiones de las llaves x y k son de N, mientras que las de y son $N+1$. Por lo que al retroalimentar la semilla x a la entrada y sus dimensiones difieren en un bit, la solucion es pasar la semilla x entera y colocar el bit menos significativo en la posición del bit mas significativo, así se tendrán los $N+1$ bits que se necesitan.

4.- Posteriormente la llave k y el bloque de datos $\hat{m}$ son pasados al módulo de cifrado, obteniendo el bloque de datos cifrado **c** a la salida del sistema CSAC.

2.5 Campos de Galois

Los campos finitos o campos de Galois, son espacios que contienen un número finito de elementos, el total de elementos se establece desde su definición.

Sea $GF(p^n)$ la definición de un campo de Galois, la p indica el carácter de los elementos del campo y el número de elementos vendrá dado por p^n, en la siguiente ecuación 2.10 se muestra esta definición

$$\begin{aligned} GF(p^n) =& (0,1,2,...,p-1) \ \cup \\ & (p,p+1,p+2,...,p+(p-1)) \ \cup \\ & (p^2,p^2+1,p^2+2,...,p^2+(p-1)) \ \cup \ ... \ \cup \\ & (p^n+p,p^n+p+1,p^n+p+2,...,p^n+p+(p-1)). \end{aligned} \tag{2.10}$$

Un par de ejemplos prácticos se muestran a continuación

$$GF(8) = (0,1,2,3,4,5,6,7) \tag{2.11}$$

$$\begin{aligned} GF(2^3) =& (0,1) \\ & (2,2+1) \\ & (2^2,2^2+1) \\ & (2^2+2,2^2+2+1) \end{aligned} \tag{2.12}$$

$$GF(2^3) = (0,1,2,3,4,5,6,7)$$

En la ecuación 2.11, podemos ver qué $p = 8$ y $n = 1$, por lo que $GF(8)$ solo tiene 8 elementos, al igual que 2.12

Sobre estos campos finitos solo 4 operaciones aritméticas están definidas, la suma, resta, multiplicación y división. Aunque las reglas que se aplican para estas operaciones son un tanto distintas de las que se acostumbran, por ejemplo, para dividir dos elementos pertenecientes a un espacio de Galois, se tiene que obtener el inverso multiplicativo del divisor y después hacer una multiplicación del inverso multiplicativo con el dividendo y obtener el módulo p.

Los campos de Galois son usados en criptografía principalmente para desordenar la información antes de ser cifrada. Una de las operaciones mas utilizadas, es la obtencion del inverso multiplicativo, la cual hace uso de un polinomio irreducible de orden $p^n + 1$ [24]. Un ejemplo del uso de los campos de Galois con fines criptográficos es la generación de las cajas de sustitución de los sistemas de cifrado DES [9] y AES [6], que están enteramente construidas del uso de inversos multiplicativos.

2.6 S-boxes

En la década de los 80, Webster y Tavares [15] introdujeron los términos de confusión y difusión de la información y desarrollaron una herramienta para conseguir que los sistemas de cifrado de esa época obtuvieran esas propiedades sin hacer cálculos más complicados.

Por lo que postularon las primeras cajas de sustitución. Una caja de sustitucion (S-box) es una herramienta criptográfica que consiste de una entrada y una salida de datos. Su forma de trabajar es sustituir el dato de entrada por otro, cabe recalcar que el dato de entrada y salida tienen muy poca o nula correlación entre ellos. Para lograr esto se necesita que la S-box posea una alta no linealidad, para generar las cajas de sustitución se utiliza un proceso no lineal [25, 26] que genera una cadena de números aleatorios que tiene una alta no linealidad.

Los sistemas de cifrado como el AES o el DES, generan sus cajas de sustitución haciendo uso de los campos de Galois. Ya que al obtener el inverso multiplicativo de una S-box se garantiza una alta no linealidad. Y esta alta no linealidad permite introducir cadenas de información con un bit de diferencia y obtener a la salida de la S-box dos cadenas de bits completamente diferentes.

A continuación, en la Figura 2.13 se muestra la S-box del sistema AES.

Las S-box dentro de un sistema de cifrado son consideradas como los únicos elementos no lineales al cifrar la información, los criptosistemas realizan siempre un procedimiento mecánico que muchas veces puede ser roto analizando los datos de entrada y salida. Sin embargo, al incluir la S-box, esta restará correlación a los datos de entrada de manera que las salidas diferirán

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
	4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
	A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
	B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
	C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
	E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Figura 2.13: S-box del sistema de cifrado AES.

demasiado de las entradas, este es el principal motivo de incorporarlas a los criptosistemas además de que al solo hacer una sustitución de datos al criptosistema no le tomará demasiado poder de cómputo efectuar esta acción.

3

Diseño de la S-box y Evaluación

Cuando se propuso el sistema CSAC y verificar su seguridad, quedo claro que era un sistema de cifrado robusto. Sin embargo, día a día surgen nuevas ideas y técnicas enfocadas en como vulnerar la seguridad de los sistemas de cifrado. Y debido a esto es recomendado que cualquier sistema de cifrado sea evaluado contra nuevas técnicas de criptoanálisis cada cierto tiempo. Es así que el sistema CSAC fue puesto a prueba contra una nueva serie de ataques y técnicas de análisis de datos, por lo que se detecto una vulnerabilidad contra la prueba denominada como Criterio Estricto Avalanche (SAC) y que no había sido considerada, cuya función es analizar los datos de entrada con los datos de salida en busca de similitudes. Cuando, dos bloques de información de entrada difieren en tan solo un bit [15]. Así que una solución para solventar esta vulnerabilidad sin afectar el tiempo de procesamiento fue el uso de una S-box para restar correlación y añadir la propiedad de difusión a las entradas del sistema CSAC.

Una de las interrogantes que surgió en el proceso fue, ¿cuál S-box es la indicada? Para responder esta interrogante existen varias opciones. Una de estas opciones es usar la S-box del sistema AES [6], ver Fig. 2.13, y posteriormente probar con otras S-box como la Gray Box [27], o las creadas por Khan [28] y Farwa [29], para comparar resultados. Al usar cualquier S-box en el sistema CSAC se pudo observar que tal sistema de cifrado presentó un buen desempeño ante

los ataques de criptoanálisis. Por lo tanto, el verdadero nuevo reto es crear una S-box propia que esté al mismo nivel de seguridad que la del sistema AES.

3.1 Propuesta de S-box

En ese sentido, para diseñar e implementar la S-box se toma en cuenta que es una componente no lineal de un sistema de cifrado [30, 31, 32].Por lo que se consideró que la S-box tenía que estar basada en alguna fenómeno matemático o natural de naturaleza no lineal, y tomando en cuenta la naturaleza del sistema CSAC, se usaron los autómatas celulares como primera opción. En la parte izquierda de la Figura 3.1, se muestra un patrón parcial de tiempo-espacio generado por la evolución de la regla local 90, la cual se aplicó a la primer fila, donde en la cuarta posición se tiene un valor de 1 y 0 en el resto de las celdas.

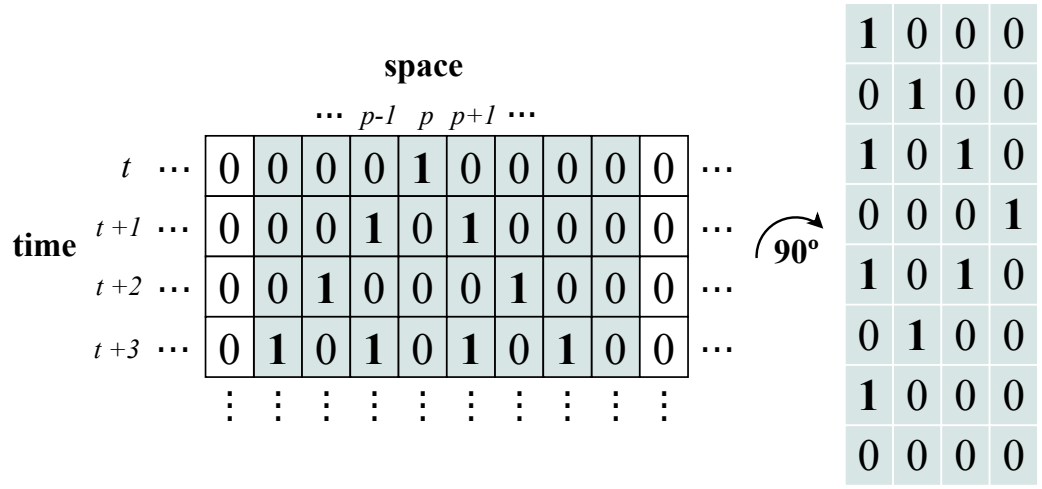

Figura 3.1: *Izquierda:* Una configuración típica de la regla 90 de los AC que considera una condición inicial con valor central de 1, y el resto de las celdas con valor 0. *Derecha:* Rotación de 90° en dirección de las manecillas del reloj de las celdas de fondo gris de la configuración mostrada en la izquierda.

En ese sentido, se consideró generar cadenas de bits aleatorios basándonos en la función

H, la cual rige la dinámica del generador de números aleatorios (bloque para generar llaves de cifrado) del sistema CSAC. Sin embargo, al considerar la estructura completa de H, los resultados no alcanzaban los estándares de seguridad que marcaba la S-box del sistema AES. Por tanto, se optó tomar en cuenta las propiedades de la función H procediendo a considerar el uso de una semilla inicial de 8 bits, y al aplicarle de manera iterativa la regla local 90 de AC tres ocasiones en el tiempo, los patrones obtenidos, incluido el patrón original, se almacenaron en los renglones de una matriz de dimensiones 4×8. De esta forma el patrón original quedaba posicionado en la primera columna y el último patrón ocupaba la última fila. Posteriormente la matriz era girada 90° hacia la derecha generando la matriz $\boldsymbol{K_L}$ de dimensiones 8×4, , ver parte derecha de la Figura 3.1. De forma algebraica se tiene

$$\boldsymbol{K_L} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{3.1}$$

Por otra parte, una matriz $\boldsymbol{K_R}$ se obtenía al multiplicar por la izquierda a $\boldsymbol{K_L}$ por una matriz de permutación fija $\boldsymbol{P}$, esto es,

$$K_{\boldsymbol{R}} = \underbrace{\begin{pmatrix} 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \\ 1&0&0&0&0&0&0&0 \end{pmatrix}}_{\boldsymbol{P}} \underbrace{\begin{pmatrix} 1&0&0&0 \\ 0&1&0&0 \\ 1&0&1&0 \\ 0&0&0&1 \\ 1&0&1&0 \\ 0&1&0&0 \\ 1&0&0&0 \\ 0&0&0&0 \end{pmatrix}}_{\boldsymbol{K}_L} = \begin{pmatrix} 0&1&0&0 \\ 1&0&1&0 \\ 0&0&0&1 \\ 1&0&1&0 \\ 0&1&0&0 \\ 1&0&0&0 \\ 0&0&0&0 \\ 1&0&0&0 \end{pmatrix}. \tag{3.2}$$

por tanto, y tomando en cuenta la forma de implementación de los elementos matriciales del sistema CSAC, la matriz resultante que consideramos $\boldsymbol{K}$ se conforma en términos de las matrices $\boldsymbol{K_L}$ y $\boldsymbol{K_R}$ de la siguiente forma

$$\boldsymbol{K} = \left(\begin{array}{c|c} K_{\boldsymbol{L}} & K_{\boldsymbol{R}} \end{array} \right) = \left(\begin{array}{cccc|cccc} 1&0&0&0&0&1&0&0 \\ 0&1&0&0&1&0&1&0 \\ 1&0&1&0&0&0&0&1 \\ 0&0&0&1&1&0&1&0 \\ 1&0&1&0&0&1&0&0 \\ 0&1&0&0&1&0&0&0 \\ 1&0&0&0&0&0&0&0 \\ \underbrace{0&0&0&0}_{K_{\boldsymbol{L}}} & \underbrace{1&0&0&0}_{K_{\boldsymbol{R}}} \end{array} \right). \tag{3.3}$$

Por tanto, la matriz $\boldsymbol{K}$ será parte fundamental en nuestro proceso para generar nuestra S-box. Cabe mencionar que la condición inicial $[0,0,0,1,0,0,0,0]$ para implementar la matriz $\boldsymbol{K}$ se consideró debido a que este vector ha resultado ser uno de los principales pivotes para la creación de los elementos matriciales del sistema de cifrado CSAC, los cuales han revelado características multifractales, ver por ejemplo [12].

Con la matriz $\boldsymbol{K}$ generada, se procedió a buscar alguna estrategia para representar una cadena de números aleatorios en el rango $[0,255]$, el cual es en el que operan las cadenas de bytes (8 bits). Lo anterior nos condujo a ver la forma de representar cada byte como un elemento de un campo finito, tal como lo es el campo de Galois denotado como $GF(2^8)$ de orden 256, el cual corresponde a la representación binaria de números del 0 al 255, y las operaciones a realizar son módulo 2. Dicho campo es frecuentemente utilizado para las llaves de cifrado o descifrado, tal como en los sistemas DES o AES. Los elementos del campo $GF(2^8)$ se pueden representar como el conjunto de todos los polinomios de grado a lo más 7, cuyos coeficientes son elementos de $GF(2)$, los enteros 0 y 1. En este caso la suma de polinomios corresponde a la operación usual de la suma entre ellos, donde los coeficientes en los cálculos se llevan a cabo en $GF(2)$. Por ejemplo, la suma de $[(x^4+x^3+1)+(x^3+x^2+1)] \mod 2 = [x^4+2x^3+x^2+2] \mod 2 = x^4+x^2$, donde los coeficientes de los términos de x^3 y x^0 módulo 2 son cero. Al considerar los coeficientes como secuencias de bits se tendría $11001+1101 = 10100$, lo cual resulta fácil de implementar en diferentes lenguajes de programación, debido a que corresponde a la aplicación de operación XOR de las secuencias de bits. Mientras que la multiplicación entre polinomios se realiza usando el módulo de algún polinomio irreducible $I(x)$ en el campo $GF(2^8)$, lo cual lo hace más difícil de implementar.

En este trabajo consideramos al polinomio irreducible $I(x) = x^8+x^6+x^5+x^3+1$, y se propuso la siguiente transformación lineal fraccional $f : GF(2^8) \to GF(2^8)$

$$f(x) = \begin{cases} (\boldsymbol{K} \times \mathbf{x}')^{-1}, & \text{if } x \neq 0, \\ 0, & \text{if } x = 0. \end{cases} \tag{3.4}$$

la cual toma ventaja de la no-linealidad de dicha función para considerarla como una función de sustitución de bytes, $\boldsymbol{K}$ es la matriz (3.3), mientras que $\mathbf{x}'$ es la transpuesta de la representación binaria del elemento $\mathbf{x}$. En los cálculos el bit menos significativo (LSB, por sus siglas en inglés) de cualquier representación binaria es bit que se encuentra en la parte más hacia la derecha.

Tomar en cuenta que el producto $(\boldsymbol{K} \times \mathbf{x}')$ en (3.4) se lleva a cabo módulo 2, y el inverso multiplicativo se calcula en el campo $GF(2^8)$, ver Referencia [33] para más detalles. En la Tabla 3.1 se ilustran algunos resultados obtenidos al usar (3.4), mientras que en la Tabla 3.2 se listan todos los elementos resultantes de la S-box propuesta en el formato de representación convencional.

TABLA 3.1: Imágenes de $f(x)$, donde le bit menos significativo (LSB) de cualquier representación binaria corresponde al bit que se encuentra más hacia la derecha. Los subíndices b y d se refieren a las representaciones binaria y decimal, respectivamente.

$x \in GF(2^8)$	x en forma binaria	$(\boldsymbol{K} \times x')$	$f(x)$
0	0 0 0 0 0 0 0 0	$(0\,0\,0\,0\,0\,0\,0\,0)_b = 0_d$	$(0\,0\,0\,0\,0\,0\,0\,0)_b = 0_d$
1	0 0 0 0 0 0 0 1	$(0\,1\,0\,1\,0\,1\,0\,1)_b = 85_d$	$(1\,1\,1\,0\,0\,1\,0\,0)_b = 228_d$
2	0 0 0 0 0 0 1 0	$(0\,0\,1\,0\,0\,0\,1\,0)_b = 34_d$	$(1\,1\,0\,1\,0\,0\,1\,1)_b = 211_d$
⋮	⋮	⋮	⋮
16	0 0 0 1 0 0 0 0	$(1\,0\,1\,0\,1\,0\,1\,0)_b = 170$	$(0\,1\,1\,1\,0\,0\,1\,0)_b = 114_d$
17	0 0 0 1 0 0 0 1	$(1\,1\,1\,1\,1\,1\,1\,1)_b = 255_d$	$(0\,1\,0\,1\,1\,1\,0\,0)_b = 92_d$
⋮	⋮	⋮	⋮
254	1 1 1 1 1 1 1 0	$(1\,0\,0\,0\,1\,0\,1\,1)_b = 139_d$	$(1\,1\,1\,1\,1\,1\,0\,0)_b = 252_d$
255	1 1 1 1 1 1 1 1	$(1\,1\,0\,1\,1\,1\,1\,0)_b = 222_d$	$(1\,0\,0\,0\,0\,1\,0\,1)_b = 133_d$

La matriz resultante será la forma final de nuestra S-box, que al evaluarla se muestra que posee propiedades criptográficas eficientes muy comparables con diferentes S-boxes, como lo es la S-box del AES.

3.2 Análisis de la S-box

Para validar las propiedades criptográficas de la S-box, se hace mediante la evaluación de varias pruebas ya establecidas en [34], como lo son el Criterio Estricto de Avalancha (SAC), Criterio

TABLA 3.2: Elementos de la S-box propuesta en forma de matriz con dimensiones 16×16.

	0	1	2	3	4	5	6	7	8	9	*A*	*B*	*D*	*D*	*E*	*F*
0	0	228	211	223	18	141	12	111	45	62	129	126	30	91	93	21
1	114	92	218	68	214	117	64	13	16	188	78	246	144	155	115	234
2	207	221	69	150	72	176	105	170	209	174	97	100	201	160	103	168
3	29	219	146	75	249	113	65	3	112	74	241	134	210	130	222	26
4	36	197	9	226	28	204	14	39	180	152	81	235	82	225	104	59
5	88	229	242	143	2	67	102	123	122	172	173	119	247	58	161	184
6	24	128	61	6	56	4	8	7	106	83	202	110	120	22	165	179
7	33	85	86	131	37	183	94	167	48	66	227	163	138	77	215	136
8	90	32	53	148	162	51	220	47	54	159	151	195	27	248	233	99
9	76	178	157	181	31	137	87	231	35	199	118	25	251	205	182	95
A	107	156	200	101	1	244	166	132	187	5	254	19	40	255	79	127
B	142	193	140	55	149	63	239	50	240	125	171	116	206	213	42	189
C	60	73	41	109	208	43	15	121	108	70	10	250	20	23	185	98
D	11	145	196	34	80	245	153	243	253	175	191	238	203	224	124	17
E	186	230	57	52	164	135	44	154	71	236	38	232	216	147	177	192
F	217	237	46	169	158	84	198	190	212	96	89	49	194	139	252	133

de Independencia de Bits (BIC), la Probabilidad de Aproximación Lineal (LP), la Probabilidad de Aproximación Diferencial (DP) y la última, y no menos importante, la No Linealidad (NL). Cada una de estas pruebas son importantes para garantizar que la S-box es un elemento fuerte contra ataques de criptoanálisis, por lo que a continuación se describirán.

3.2.1 Criterio Estricto de Avalancha (SAC)

La primera de las pruebas se denomina Criterio Estricto de Avalancha (SAC) y estudia la relación que pudiera existir entre la entradas y salidas de las S-boxes. La prueba del SAC fue propuesta en 1985 por Webster y Tavares en [15], argumentando que algunos criptosistemas de

la época mostraban una correlación entre sus entradas y salidas, ya que si se procesaban dos palabras binarias cuya única diferencia fuera de un bit, los datos de salida también mostrarían el mismo patrón, dicha vulnerabilidad es capaz de romper la codificación del sistema de cifrado si se hace de forma meticulosa.

En la Tabla 3.3, se muestran los resultados obtenidos al aplicar la SAC a la S-box que se propone, junto con los resultados obtenidos para las S-box del AES [6], de Farwa [29] y de Khan [28]. Como se puede observar, los resultados obtenidos son muy parecidos a los que se lograron con la S-box del sistema AES. Se puede deducir que los resultados obtenidos son buenos, ya que tienden al valor de 0.5, que en teoría es el mejor resultado que se puede obtener en esta prueba. Estos resultados indican que al menos la mitad de los bits de la palabra que se procesa por la S-box son permutados, por lo que se puede decir que la S-box propuesta consigue pasar esta prueba.

3.2.2 Criterio de Independencia de Bits (BIC)

El criterio de la independencia de bits, introducida por Webster y Tavares [15], analiza el comportamiento de los patrones de los bits en la salida de la S-box y los efectos de estos cambios en las subsecuentes rondas del cifrado [29]. Este criterio se analiza al modificar un simple bit de entrada del texto plano y los vectores binarios de salida son analizados para ver la independencia. El valor numérico de esta prueba es mostrado en la Tabla 3.3, en donde podemos ver que el valor promedio obtenido por la S-box propuesta es comparable con los resultados de las otras S-boxes, pero presenta un mejor desempeño que la S-box presentada en [28].

3.2.3 No Linealidad (NL)

La no linealidad (NL) es una propiedad que se define como el número de bits que deben ser alterados en la tabla de verdad de una función Booleana para aproximarse a la función afín más

cercana [29]. El límite superior de la no linealidad es $N = 2^{n-1} - 2^{\frac{n}{2}-1}$ cuando las S-boxes son representadas en el campo de Galois $GF(2^n)$ [27]. Para $GF(2^8)$, el valor máximo teórico de N es 120. A pesar de que muchas S-boxes no son por completo funciones no lineales, al calcular la NL de las S-boxes que se consideran en este trabajo, los resultados son cercanos al valor del límite superior teórico. La cuarta columna de la Tabla 3.3 ilustra los resultados de este análisis obteniendo un valor promedio de NL de 112. Asimismo, la S-box de Khan [28] presenta el valor más pequeño de NL, lo cual indica que su comportamiento es más lineal que el de las otras S-boxes.

3.2.4 Probabilidad de Aproximación Lineal (LP)

La probabilidad de aproximación lineal es usada para determinar el máximo valor de desequilibrio del evento entre los bits de entrada y salida, y matemáticamente esta prueba se define como

$$LP = \max_{M_x, M_y \neq 0} \left| \left(\frac{\#\{x | x \cdot M_x = S(x) \cdot M_y\}}{2^n} \right) - \frac{1}{2} \right| \tag{3.5}$$

donde M_x y M_y son las máscaras de entrada y salida, respectivamente, x denota el conjunto de todas las posibles entradas, $\#\{\cdot\}$ es la cardinalidad del conjunto $\{\cdot\}$, mientras que 2^n es el número de todos sus elementos [29]. Los resultados de esta prueba se muestran en la quinta columna de la Tabla 3.3, donde podemos observar que todos los resultados obtenidos por la mayoría de las S-boxes que se comparan en este trabajo son similares a los que se obtiene con la S-box del sistema AES, considerada un estándar de seguridad. Lo anterior nos permite considerar que la S-box propuesta también presenta un buen desempeño.

3.2.5 Probabilidad de Aproximación Diferencial (DP)

Este análisis mide como un cambio se presenta en respuesta a cualquier modificación en la entrada. En circunstancias ideales, se espera que la S-box debe mostrar una uniformidad diferencial. Para asegurar esto, una entrada diferencial Δx debe ser mapeada solamente a una salida diferencial Δy. La Probabilidad de Aproximación Lineal (DP) se define como

$$DP(\Delta x \to \Delta y) = \left[\frac{\#\{x \in X \mid S(x) \oplus S(x \oplus \Delta x) = \Delta y\}}{2^n} \right] \tag{3.6}$$

donde X es el conjunto de todos los valores de entrada posibles y 2^n indica el número de estos elementos [29]. En caso de que la uniformidad diferencial sea más pequeña, entonces la S-box es más fuerte. En la última columna de la Tabla 3.3 se muestran los resultados obtenidos y se observa que todas las S-boxes presentan el mismo valor, por lo que se puede decir que los resultados son comparables con los de las otras S-boxes.

En la siguiente tabla se muestran los resultados numéricos obtenidos al aplicar las pruebas de criptoanálisis a la S-box propuesta.

TABLA 3.3: Resultados Numéricos del Criterio Estricto de Avalanche (SAC), Criterio de Independencia de Bits (BIC), No Linealidad (NL), Probabilidad de Aproximación Lineal (LP), y Probabilidad de Aproximación Diferencial (DP) para la S-box propuesta y otras consideradas.

S-box	SAC	BIC	NL	LP	DP
Propuesta	**0.4998**	**112**	**112**	**0.0625**	**0.015625**
AES [6]	0.4999	112	112	0.0625	0.015625
Farwa [29]	0.5016	112	112	0.0625	0.015625
Khan [28]	0.4864	103.7	103.7	0.1563	0.017188

Como conclusión podemos decir que la S-box generada en este trabajo cuenta con una buena

seguridad contra ataques de criptoanálisis, ahora se le aplicarán pruebas estadísticas para verificar las fortalezas de la S-box al procesar imágenes.

3.3 Análisis estadístico de la S-box para imágenes

Para evaluar estadísticamente la S-box se usa el Generalized Majority Logistic Criterion o GMLC por sus siglas en inglés propuesto por Hussain en [35]. El GMLC es un conjunto de pruebas usadas para evaluar imágenes que fueron procesadas por una S-box. Las pruebas que conforman dicho criterio son la entropía, energía, correlación, contraste y homogeneidad. En ese sentido, con la finalidad de realizar tal análisis estadístico se consideraron un total de seis imágenes en escala de grises de dimensiones de 512×512 píxeles para procesarlas con la S-box propuesta y las S-boxes de los trabajos considerados en la Sección 3.2, ver Tabla 3.3. Las imágenes planas de prueba se muestran en la columna (a) de la Fig. 3.3 , mientras que en las columnas (b)-(e) de la Fig. 3.3 se muestran las imágenes procesadas por la S-box propuesta y las diferentes S-boxes consideradas en este trabajo. Podemos observar niveles aceptables de confusión en la forma visual de los datos, pero no podemos lograr una completa e ininteligible forma. Lo anterior se debe a que la aplicación de la S-box es solo una parte de las diferentes etapas que conforman a un sistema de cifrado. En caso de que se combinaran con el resto de las etapas, entonces podemos lograr una forma mejor e ininteligible [29]. Para evaluar el desempeño de las S-boxes se consideran las siguientes pruebas estadísticas: entropía, energía, correlación, contraste y homogeneidad, las cuales se describen de forma general a continuación.

La primera prueba es la entropía con la finalidad de medir la aleatoriedad de las imágenes. Esta prueba proporciona información sobre la textura de una imagen y devuelve un valor escalar H. Si H se aproxima al valor ideal de 8, este caso ocurre cuando todos los valores del histograma de la imagen están igualmente distribuidos por lo que menor será el éxito que tengan los atacantes para decodificar las imágenes cifradas. La entropía de una variable aleatoria X se calcula mediante la siguiente fórmula

$$H = -\sum_{j=0}^{N-1} p(x_j) \log_b p(x_j). \tag{3.7}$$

en donde la variable aleatoria X toma n resultados, es decir, $x_0, x_1, x_2, ..., x_n$, $p(x_j)$ es la función de masa de probabilidad en x_j y b es la base del logaritmo utilizado. En la tercer columna de la Tabla 3.4 se muestran los resultados del cálculo de H aplicado a las imágenes de prueba y considerando las diferentes S-boxes. Podemos observar que los valores de entropía, para todos las S-boxes, están cercanos al valor ideal, lo cual implica una alta resistencia a los ataques de entropía. En particular, los resultados obtenidos con la S-box propuesta muestran en general un gran desempeño con respecto a las otras S-boxes.

Para la prueba de energía, la cual cuantifica la energía en una imagen, se utiliza la matriz de co-ocurrencias de niveles de grises (GLCM) [34]. Para cuantificar esta prueba se usa la ecuación $\sum_{i,j} p(i,j)^2$, en donde $p(i,j)$ corresponde al valor de la matriz de co-ocurrencia en las coordenadas (i,j). Pequeños valores de energía indican una buen funcionamiento de la S-box. Los resultados de esta prueba para las imágenes procesadas se muestran en la cuarta columna de la Tabla 3.4, donde se observan valores bajos de esta medida para todos las S-boxes. Por lo tanto, las imágenes procesadas presentan características de uniformidad.

Para mostrar que la imagen cifrada es independiente de la imagen plana, calculamos el coeficiente de correlación entre ambas imágenes. Si este coeficiente es cercano a 0 sugiere que no existe una correlación lineal o que hay una correlación lineal baja [29]. La correlación está dada de la siguiente manera:

$$r = \frac{Cov(X,Y)}{S.D(X) \times S.D(Y)} \tag{3.8}$$

donde $Cov(X,Y)$ es la covarianza de las variables aleatorias X y Y, $S.D(X)$ y $S.D(Y)$ son las desviaciones estándar de las variables aleatorias X e Y, respectivamente.

En la quinta columna de la Tabla 3.4 se ilustra la correlación entre las imágenes de prueba contra sus versiones procesadas por la S-boxes es cercana a 0, por lo que no hay correlación entre los píxeles cuando las S-boxes se utilizan para procesar las diferentes imágenes de prueba. Esto nos permite tener buenas propiedades de cifrado como confusión y difusión, entre otras.

Asimismo, la prueba de contraste se utiliza para determinar la difusión en una imagen. Los valores similares de píxeles en la observación resulta en un contraste bajo, mientras que un valor de contraste alto significa un aumento de la aleatoriedad, en [34] se propone esta prueba para el análisis de imágenes procesadas por s-boxes y se nota que para imágenes planas los valores de contraste son pequeños en comparación con los que se obtienen de una imagen procesada por una s-box fuerte contra ataques de criptoanálisis. Para calcular este coeficiente se usa la siguiente ecuación

$$C = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} |i-j|^2 p(i,j) \tag{3.9}$$

en donde $p(i,j)$ es el valor del pixel e (i,j) indican las coordenadas de la imagen. Para este análisis, los resultados numéricos de los S-boxes muestran altos niveles de contraste, ver sexta columna de la Tabla 3.4. De este modo, las S-boxes revelan un fuerte rendimiento de cifrado en las imágenes de prueba.

Por último, el análisis de homogeneidad se utiliza para determinar las características que muestran los elementos de la matriz GLCM con respecto a su diagonal. Si se obtiene un valor alto, entonces los valores grandes están en la diagonal principal. Para calcular esta prueba se considera la siguiente ecuación

$$H = \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} \frac{p(i,j)}{1+|i-j|} \tag{3.10}$$

en donde $p(i,j)$ es el valor del pixel y (i,j) denotan la coordenada a analizar. Los resultados

de esta prueba estadística se muestran en última columna de la Tabla 3.4, donde se muestran valores aceptables, y la S-box presenta un buen desempeño comparado con el resto de las S-boxes.

Con estos resultados podemos decir que la S-box propuesta tiene un buen desempeño contra algunos ataques de criptoanálisis.

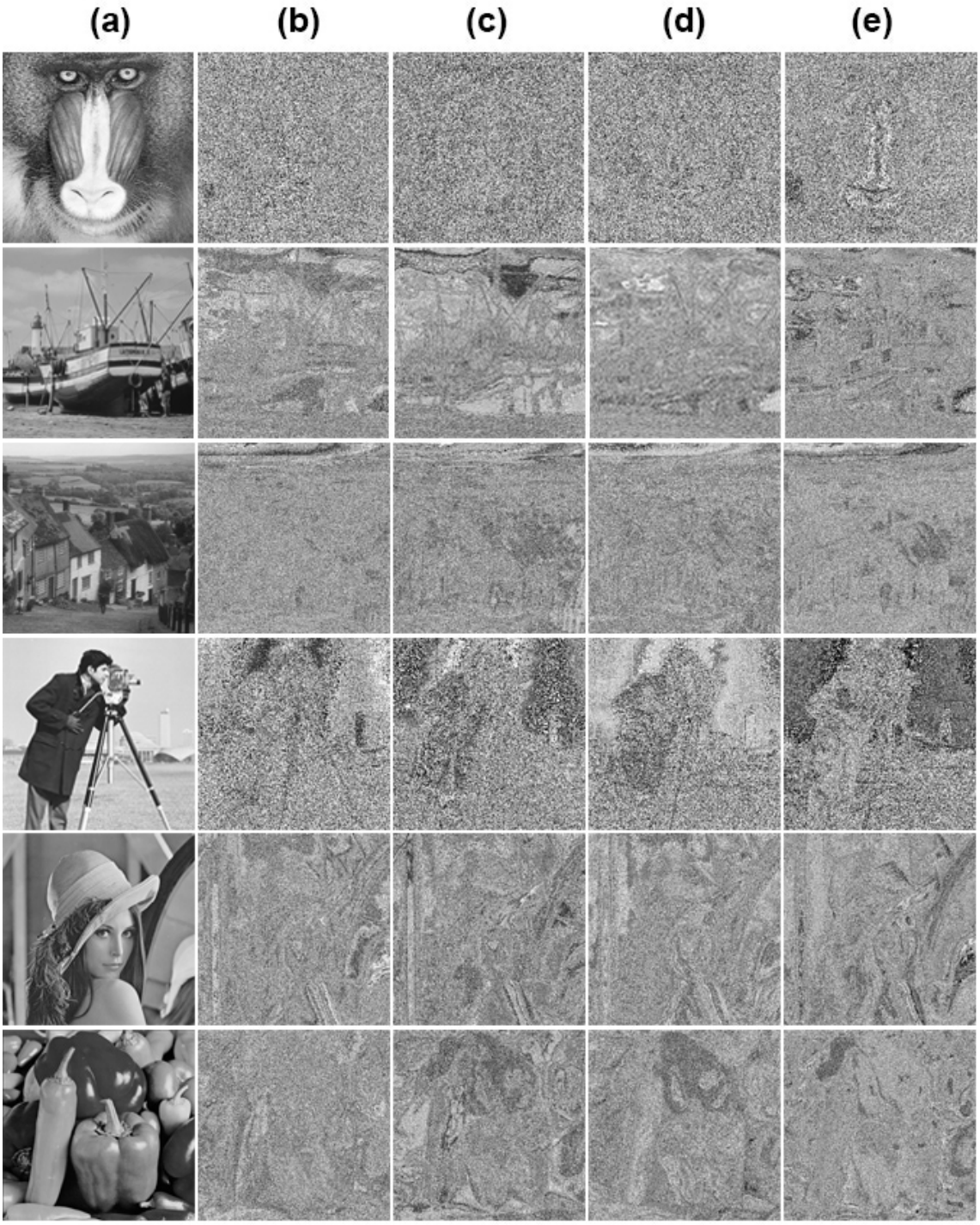

Figura 3.2: Imágenes de prueba en escala de grises: baboon, boats, hills, cameraman, Lena y peppers, en la columna a), en las columnas b), c), d) y e) se exhiben las versiones procesadas por las S-box propuesta, la del sistema AES, la de Farwa y Khan, respectivamente.

TABLA 3.4: Valores estadísticos del análisis de entropía, correlación, energía, contraste, y homogeneidad aplicado a varias imágenes en escala de grises procesadas por las S-boxes consideradas.

Imagen de Prueba	S-box	Entropía	Energía	Correlación	Contraste	Homogeneidad
Baboon	Propuesta	7.9488	0.0161	0.0124	9.8323	0.4053
	AES	7.9539	0.0161	0.0135	10.4028	0.4013
	Farwa	7.9180	0.0160	0.0071	10.5875	0.3992
	Khan	7.9504	0.0162	0.0297	9.7738	0.4078
Boats	Propuesta	7.9390	0.0174	0.0939	9.7575	0.4472
	AES	7.9536	0.0176	0.1350	9.2532	0.4586
	Farwa	7.9293	0.0178	0.0898	9.4207	0.4473
	Khan	7.9312	0.0186	0.0740	9.8044	0.4514
Hill	Propuesta	7.9527	0.0170	0.0622	8.7278	0.4372
	AES	7.9391	0.0174	0.0627	10.2369	0.4359
	Farwa	7.9516	0.0165	0.0666	10.3084	0.4293
	Khan	7.9328	0.0177	0.0767	8.5779	0.4445
Cameraman	Propuesta	7.9302	0.0190	0.1004	8.7964	0.4588
	AES	7.9164	0.0190	0.0776	10.4722	0.4407
	Farwa	7.8946	0.0240	0.1272	7.7911	0.4867
	Khan	7.8775	0.0220	0.1361	8.8423	0.4704
Lena	Propuesta	7.9487	0.0174	0.0727	9.0372	0.4463
	AES	7.9444	0.0174	0.0711	10.4978	0.4398
	Farwa	7.9452	0.0169	0.0643	10.0932	0.4337
	Khan	7.9559	0.0176	0.1006	8.3532	0.4558
Peppers	Propuesta	7.9506	0.0170	0.0473	9.5510	0.4333
	AES	7.9458	0.0170	0.0766	10.1587	0.4328
	Farwa	7.9567	0.0166	0.0598	9.8999	0.4258
	Khan	7.9585	0.0172	0.0634	8.6984	0.4418

4

Mejora del Sistema CSAC en términos de la S-box

En este Capítulo se presentan algunas modificaciones, en función de la S-box propuesta, a la estructura del sistema CSAC, así como sus respectivas evaluaciones. Con base a los resultados contra los ataques de criptoanálisis, se sugiere considerar como la versión más robusta, a la modificación del sistema CSAC que presentó un mejor desempeño. Además, se describe de manera general la contribución del sistema de cifrado CSAC y la S-box propuesta a un sistema de imágenes cifradas visualmente significativas denominado VMEI por sus siglas en inglés.

4.1 Versión Mejorada del Sistema CSAC (Versión 3)

Con la finalidad de encontrar y corregir vulnerabilidades no detectadas en la última versión del sistema CSAC, se aplicaron algunas pruebas de criptoanálisis que no se habían considerado anteriormente. En ese sentido se empezó a estudiar la prueba denominada como criterio estricto de avalancha o SAC por sus siglas en inglés (Strict Avalanche Criterion), la cual se describe de forma general más adelante. La seguridad de las dos primeras versiones del sistema CSAC, que son las presentadas en los trabajos [22] (**Versión 1**) y [23] (**Versión 2**), ante tal prueba resultaron no ser tan robustas, debido a que los resultados obtenidos del SAC [15] fueron cercanos a cero,

por lo que un ataque criptográfico centrado en esta debilidad sería capaz de romper el cifrado de tales versiones, ver Tabla 4.2.1. Con la finalidad de corregir la falta de seguridad se propone integrar una S-box al sistema de cifrado. En particular, la adherencia de una S-box auxiliará en "remover" correlación a la información y debido a que es solo una permutación de datos, resultará un proceso más rápido. Cabe mencionar que la S-box es la única componente de carácter no lineal dentro de la estructura del sistema CSAC.

En la Figura 4.1 se ilustra un diagrama de bloques de una versión con la S-box y la forma en que trabaja el sistema para cifrar información, la cual denominaremos como **versión 3**. El texto plano $\mathbf{m}$ de dimensiones $1 \times n$ es la entrada a la S-box, resultando el bloque de texto de mismas dimensiones $\mathbf{m_s}$, el cual será entrada al bloque de pre proceso junto con la condición inicial $\mathbf{z}$ de dimensiones $1 \times (n+1)$, al ser preprocesada la información. La salida del pre procesado se denomina $\hat{\mathbf{m}}$ y será la señal a la que se le aplique la familia de permutación junto con la llave de cifrado k de tamaño $1 \times n$; la llave $\mathbf{k}$ es el resultado proveniente de la salida del bloque encargado de generar llaves para cifrar la información, donde las entradas de dicho bloque se conforman de dos condiciones iniciales $\mathbf{x}$ y $\mathbf{y}$ de tamaño $1 \times n$ y $1 \times (n+1)$, respectivamente. Al final, se tiene la señal cifrada, la cual se representa por $\mathbf{c}$ y su tamaño es $1 \times n$. El proceso para descifrar es similar, pero en orden inverso.

La última modificación que consideramos, la cual denominaremos como **versión 4**, elimina el módulo de preproceso y usa solo la S-box para restar correlación al texto plano, todo esto con el propósito de hacer más rápido el proceso de cifrado. En la Figura 4.2 se ilustra un diagrama de bloques que muestra la configuración de esta versión. En tal diagrama podemos ver la forma en que opera esta configuración al tener como entrada a la S-box el texto plano e inmediatamente el texto sustituido es enviado al bloque de la familia de cifrado junto con la llave de cifrado, de igual forma que la configuración de la versión 3.

En el diagrama de la figura Fig. 4.2 podemos ver que el módulo de pre proceso ya no está y en su lugar solo se encuentra la S-box. La forma en que trabaja esta configuración es pasando

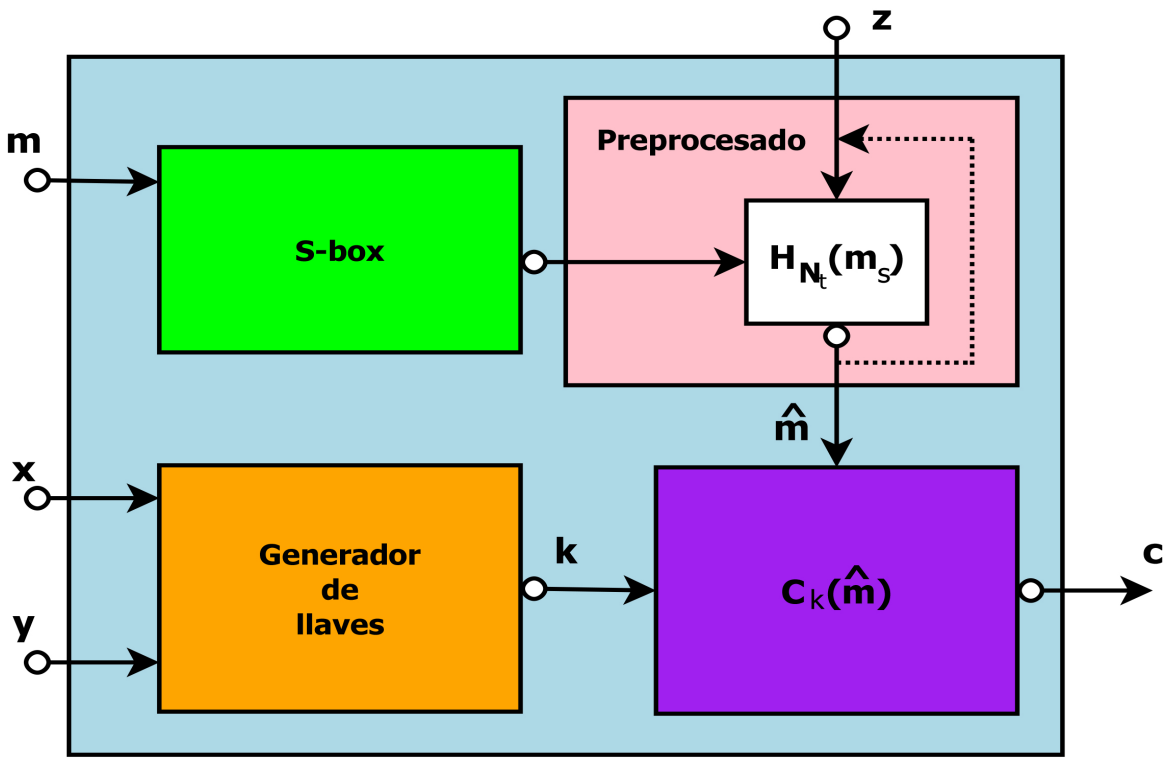

Figura 4.1: Diagrama de bloques de la versión mejorada (versión 3) del sistema CSAC.

el texto plano (m) por la S-box e inmediatamente después de que permutan los datos, el texto sustituido (m_S) es enviado al bloque de cifrado junto con la llave de cifrado K, de igual forma que la configuración anterior (4.1), el resultado de esta operación nos entregara el texto cifrado (c).

4.2 Evaluación del sistema CSAC modificado

Para evaluar la seguridad criptográfica de las versiones modificadas del sistema CSAC, se aplicarán una serie de pruebas al cifrado de imágenes resultantes de las versiones del sistema CSAC.

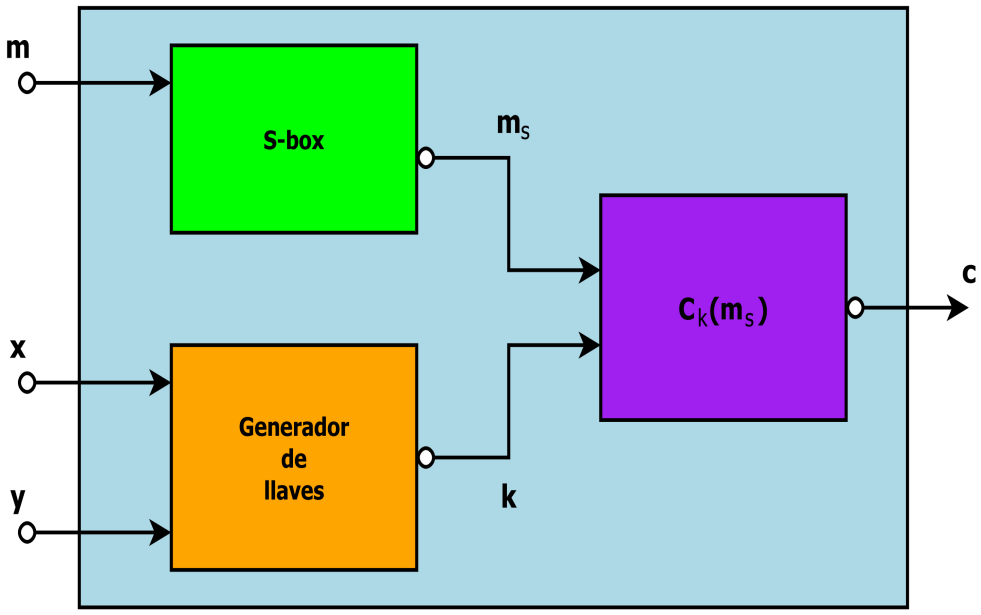

Figura 4.2: Diagrama de bloques de la versión compacta (versión 4) del sistema CSAC.

Las pruebas que conforman la evaluación se describirán a continuación.

4.2.1 Criterio Estricto de Avalancha (SAC)

La prueba denominada como criterio estricto de avalancha o SAC por sus siglas en inglés (Strict Avalanche Criterion), consiste en cifrar dos bloques de texto plano con la diferencia de un solo bit entre ellos, y analizar que tanto cambiaban las señales de salida. Tal prueba se aplica de forma recurrente a un conjunto de palabras binarias y los resultados se analizan estadísticamente. Si el resultado promedio se aproxima a 0.5 se considera un éxito, mientras que los resultados que tienden a 1 o 0 son considerados como fracaso. Como se mencionó anteriormente, la seguridad del sistema CSAC ante tal prueba resultó no ser tan robusta, debido a que los resultados obtenidos del SAC fueron cercanos a cero, por lo que un ataque criptográfico centrado en esta debilidad sería capaz de romper el cifrado del sistema CSAC. En la Tabla 4.1 se presentan los resultados obtenidos después de aplicar el SAC a las cuatro versiones del CSAC,

así como del sistema AES. Se puede observar que los resultados obtenidos de las versiones 3 y 4 del sistema CSAC presentan un buen desempeño comparable con los resultados obtenidos por el sistema AES, los cuales se aproximan al valor de 0.5, mientras que las versiones que no cuentan con S-box en su estructura, obtienen resultados cercanos a cero, indicando vulnerabilidad ante tal prueba.

TABLA 4.1: Resultados numéricos obtenidos después de aplicar el SAC a las cuatro versiones del sistema CSAC y del sistema AES.

Sistema de Cifrado	**Valor SAC**
CSAC Versión 1	0.23 - 0.35
CSAC Versión 2	0.15 - 0.23
CSAC Versión 3	**0.49 - 0.52**
CSAC Versión 4	0.46 - 0.48
AES	0.49 - 0.51

4.2.2 Distribución del Histograma

Otra prueba común en los sistemas de cifrado es calcular el histograma de un conjunto de datos cifrados y observar que los valores del texto cifrado estén distribuidos uniformemente y así detectar rápidamente si algún valor tiene una mayor incidencia que otros. Para esto se hace una gráfica de todos datos cifrados según su intensidad e incidencia. En la Fig. 4.3 se muestra la imagen de baboon en escala de grises y los histogramas de sus versiones 3 y 4 cifradas.

Se observa que los histogramas están distribuidos uniformemente, por lo que los resultados en esta prueba, son satisfactorios para las imágenes cifradas por las versiones 3 y 4 del sistema CSAC.

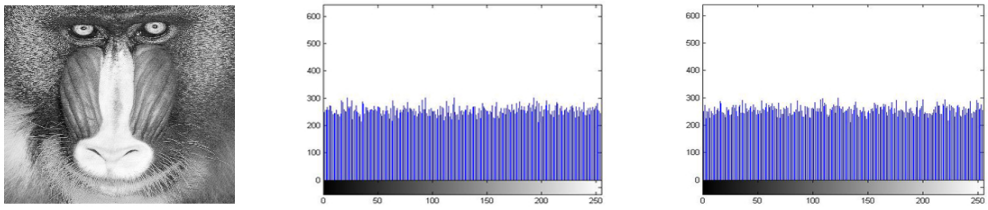

Figura 4.3: a) Imagen de Baboon en escala de grises, b) Histograma de la imagen Baboon cifrada por la versión mejorada del sistema CSAC. c) Histograma de la imagen Baboon cifrada por la versión compacta del sistema CSAC.

4.2.3 Correlación entre Texto Plano y Texto Cifrado

Esta prueba nos muestra si el texto plano y el texto cifrado son realmente independientes, al medir la correlación que existe entre cada bit de ambos bloques de texto. Si el resultado de la prueba tiende a cero, entonces se puede decir que no existe correlación o semejanza entre los bloques de texto. En la Tabla 4.2.3 se muestran los coeficientes de correlación obtenidos entre el texto plano que corresponde a una imagen y sus versiones cifradas con las diferentes variantes del sistema CSAC y AES. Como se observa los coeficientes en todos los casos son muy pequeños, por lo que se concluye, que no existe correlación alguna y concluimos que todos los sistemas de cifrado tienen buenos resultados.

4.2.4 Correlación Adyacente del Texto Cifrado

Cuando se observa una imagen, podemos notar que debido a su naturaleza tienen una alta correlación entre sus píxeles, ya sean en dirección horizontal, vertical o diagonal. Es por esta razón que es importante medir la correlación entre sus píxeles adyacentes en el texto cifrado, para garantizar que no exista en la imagen cifrada. La correlación entre una pareja de píxeles adyacentes ya sea horizontales, verticales o diagonales, siempre debe ser muy pequeña para decir que es un buen cifrado de imágenes. Ya que para valores menores a ± 0.3 significan que

TABLA 4.2: Resultados numéricos de la correlación entre texto plano y cifrado obtenidos por todas las versiones del CSAC y AES.

Sistema de Cifrado	Correlación
CSAC Versión 1	0.009
CSAC Versión 2	0.007
CSAC Versión 3	0.0012
CSAC Versión 4	0.00164
AES	-0.0015

no existe relación entre las imágenes [23].

En la Tabla 4.2 se muestran los resultados obtenidos por cada una de las versiones 3 y 4 del sistema CSAC al aplicarse a la imagen en escala de grises de Lena. Y se observa que la correlación adyacente entre píxeles es muy baja en ambas versiones por lo que podemos decir que no existe correlacion adyacente en ninguna direccion, por lo que se trata de un cifrado bueno. Mientras que en la Figura 4.4 se muestra la imagen de Lena (a) que es el texto plano y sus respectivos cifrados por la versión mejorada y compacta del sistema CSAC (d, g), en la columna central se presentan los histogramas de cada imagen respectivamente (b, e, h) y finalmente en la ultima columna se muestran las gráficas de correlación adyacente diagonal para cada imagen de la primer columna (c, f, i).

Como se observa en las imágenes de la ultima columna, podemos concluir que las versiones mejoradas del sistema CSAC hacen un buen cifrado al eliminar la correlación adyacente a la información de entrada. Las gráficas de la ultima columna muestran que la correlación adyacente esta dispersa por lo que no hay patrón.

TABLA 4.3: Resultados numéricos de la correlación adyacente entre los píxeles del texto plano y el texto cifrado obtenidos por todas las versiones del CSAC y AES.

Sistema de Cifrado	CAD	CAH	CAV
CSAC Versión 1	**0.0159**	**0.0316**	**-0.0038**
CSAC Versión 2	0.0345	-0.0094	-0.0097
CSAC Versión 3	-0.00281	0.0037	0.0068
CSAC Versión 4	0.0049	0.004	0.0059
AES	-0.0015	0.0273	0.0017

4.2.5 NPCR y UACI

Para medir la fortaleza del cifrado de imágenes contra ataques de criptoanálisis diferencial se usan dos parámetros, comúnmente conocidos como NPCR (Number of Pixels Chage Rate) y UACI (Unified Averaged Changing Intensity). Estos parámetros se encargan de medir los cambios en dos imágenes cifradas con la misma llave y que proceden de imágenes planas con un solo bit de diferencia en alguno de sus píxeles. Para describir esta prueba se asume que las imágenes $P1$ y $P2$ son dos imágenes planas y que la única diferencia entre ellas es un bit en su pixel inicial, por lo demás son iguales. Al cifrar ambas imágenes con la misma llave se obtienen las imagenes cifradas $C1$ y $C2$ respectivamente, en donde $C2$ es la versión cifrada de $P2$ que es la imagen con el pixel modificado. Entonces para medir los parámetros de NPCR y UACI, usamos las Ecuaciones 4.1 y 4.2 respectivamente

$$NPCR(C1,C2) = \sum_{i=0}^{N}\sum_{j=0}^{N}\frac{D(i,j)}{T} \times 100 \tag{4.1}$$

$$UACI(C1,C2) = \sum_{i=0}^{N}\sum_{j=0}^{N}\frac{||C1(i,j) - C2(i,j)||}{F \times T} \tag{4.2}$$

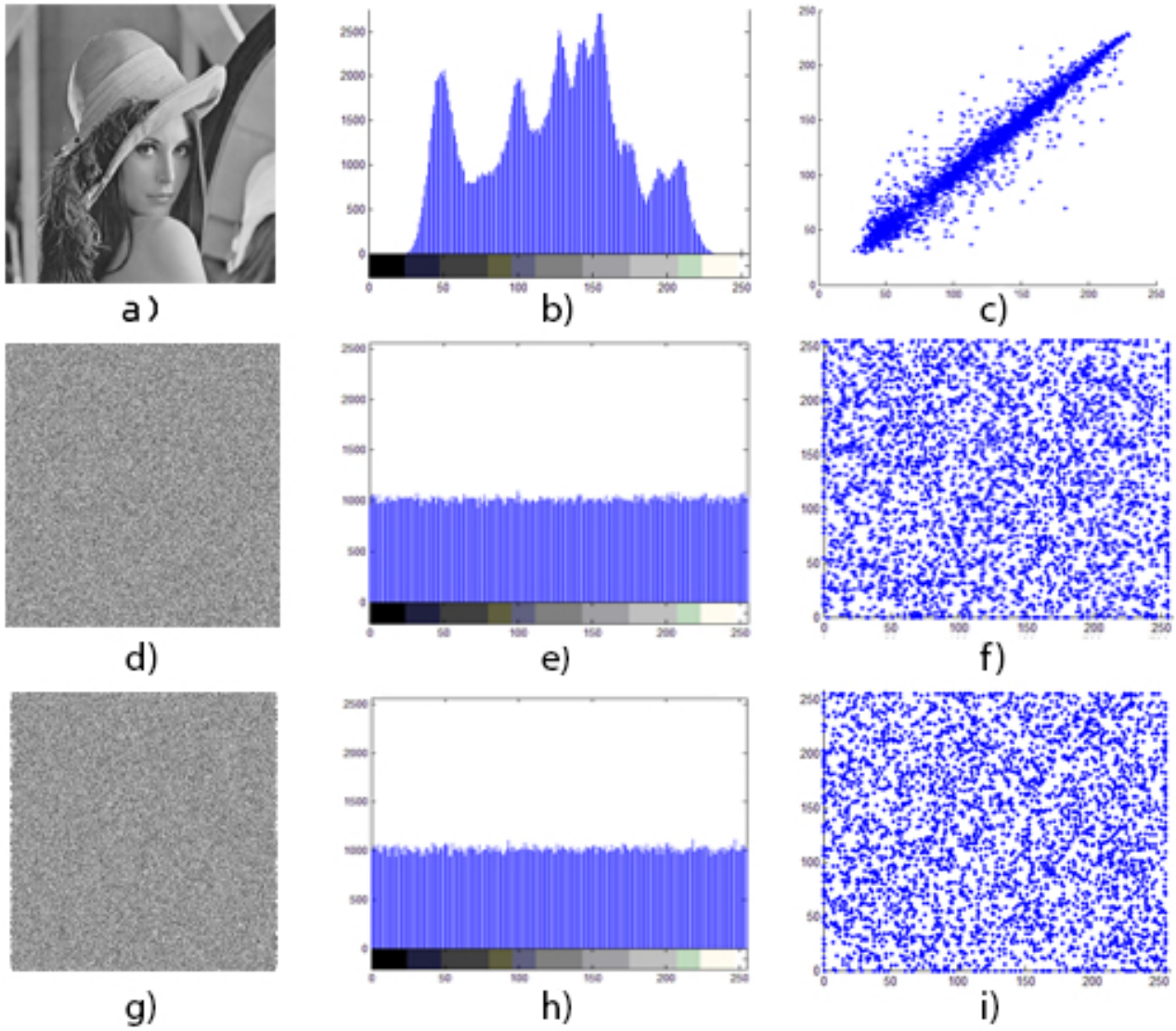

Figura 4.4: Imagen de prueba Lena (a) y sus versiónes cifradas por el sistema CSAC mejorado (versión 3) (d) y compacto (versión 4) (g), junto a sus histogramas (b, e y h, respectivamente) y sus gráficas de correlación adyacente diagonal (c, f, i)

en donde $D(i,j) = 0$ si $C1(i,j) = C2(i,j)$, y $D = 1$ en caso contrario, T nos indica el número de píxeles de cada imagen y F nos dice el máximo valor que puede tomar un pixel, para las imágenes de escala de grises este valor es 256, tomando en cuenta que los valores van desde 0

TABLA 4.4: Resultados numéricos obtenidos en las pruebas de UACI y NPCR, por los criptosistemas propuestos en este trabajo (versión mejorada del sistema CSAC y la versión compacta del mismo).

Sistema de Cifrado	UACI	NPCR
Versión 3	0.3359	0.9959
Versión 4	0.3351	0.9958

hasta 255, los índices i y j denotan la numeración del renglón y columna, respectivamente de los píxeles de la imagen. Entonces Teóricamente para imágenes en escala de grises los valores máximos que se pueden obtener son 99.609 % para el porcentaje NPCR y 33.464 para el UACI. Los resultados de estas pruebas al variar el primer pixel y el bit menos significativo de la imagen y aplicarlas a las diferentes variantes del sistema CSAC están reportadas en la Tabla 4.4 y en la Fig. 4.5 se ilustra un ejemplo de dos imágenes planas idénticas a excepción de un bit en su primer pixel.

Como se puede observar, los resultados numéricos nos indican que las modificaciones propuestas en este trabajo cumplen su función de hacer un buen cifrado.

4.2.6 Chosen-Plain Image Attack

Esta técnica más que ser una prueba es un ataque de criptoanálisis diferencial en la cual los atacantes tienen la libertad de escoger una imagen (por lo general una imagen en color negro, ya que el negro en forma binaria se representa con el número cero) y cifrarla bajo las mismas condiciones de otra imagen que corresponde al texto plano. Algunos sistemas muestran vulnerabilidades a este tipo de ataques como lo fue el sistema CSAC en su primer versión que al aplicar el ataque mostraba la información correspondiente al texto plano, es de allí que nace la segunda mejora que se mencionaba en el capítulo 3, que corrige esta vulnerabilidad y posteriormente los sistemas propuestos en este trabajo también deben ser capaces de resistir el

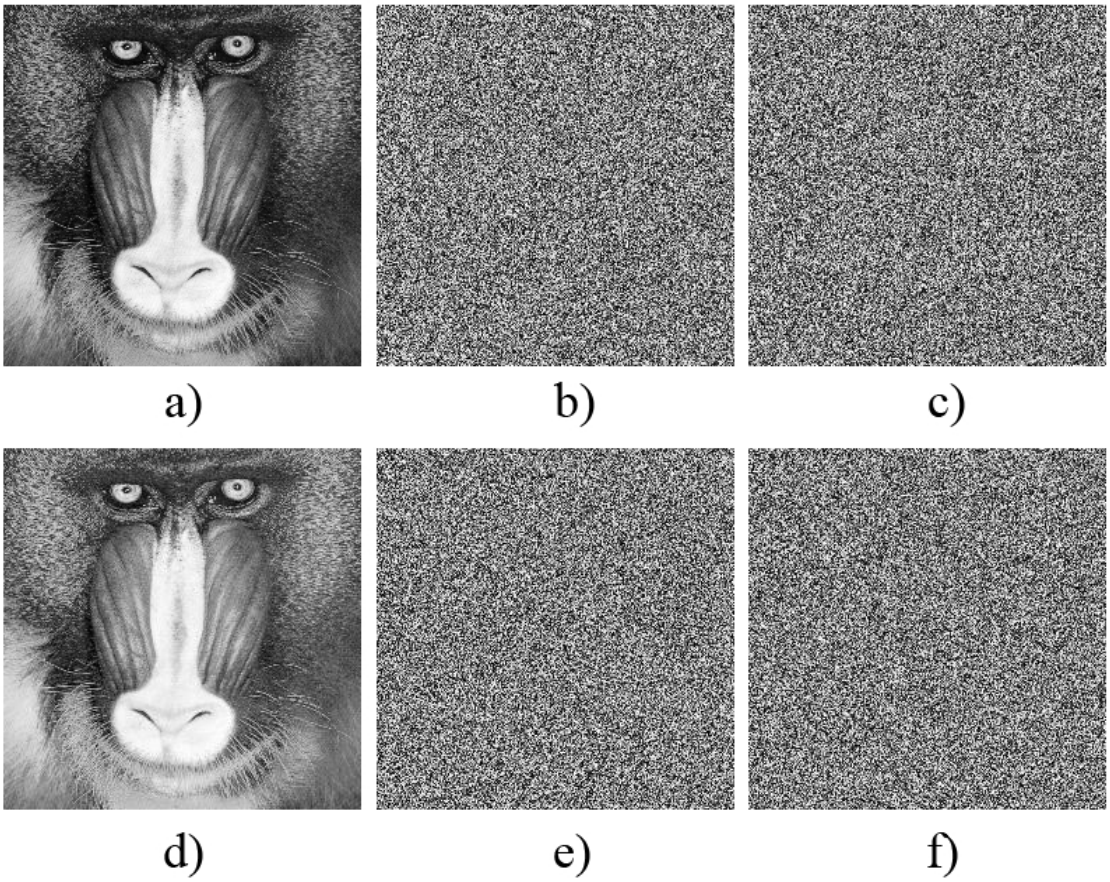

Figura 4.5: Imagen de prueba Babbon (a) y otra imagen de Babbon modificada con un solo pixel de diferencia (d), junto a sus versiones cifradas por los criptosistemas CSAC mejorado (b y c, para la imagen de babbon original) y CSAC compacto (e y f, para la imagen de babbon modificada)

ataque, por lo que se efectuara este ataque sobre las dos mejoras propuestas y los resultados se muestran en las imagen Fig. 4.6 y 4.7.

Como podemos ver, los resultados son satisfactorios, ya que al aplicar el ataque de ambas versiones, no se obtuvo información. La vulnerabilidad de los sistemas de cifrado ante esta prueba se observa en la Ec. 4.3, que muestra que si se cifran dos imágenes con la misma llave y después de aplica una operación XOR entre las dos imágenes resultantes del cifrado.

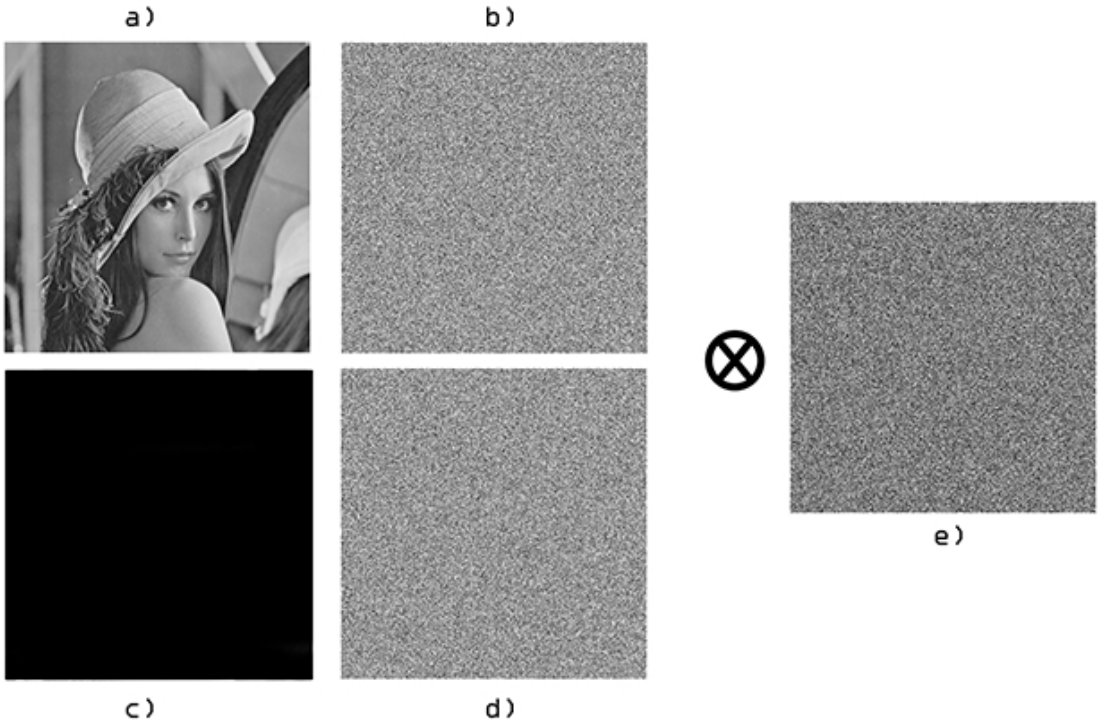

Figura 4.6: Prueba del Chosen Plain Image Attack, a) imagen de prueba Lena, b) versión cifrada de Lena por el sistema CSAC mejorado, c) imagen Plana en color negro, d) versión cifrada de la imagen plana, e) resultado de la prueba chosen plain image attack

El resultado sera el texto plano, siempre y cuando, una de las imágenes que se cifren sea una imagen plana de ceros o totalmente negra, esta imagen es seleccionada por el atacante por lo que de ahí se le da el nombre de la prueba.

$$
\begin{aligned}
&C_P = I_P \oplus K \\
&C_N = I_N \oplus K = 0 \oplus K \\
&I_R = C_P \oplus C_N = (I_P \oplus K) \oplus (0 \oplus K) \\
&= I_P \oplus K \oplus K = I_P \\
&\therefore I_R = I_P
\end{aligned}
\tag{4.3}
$$

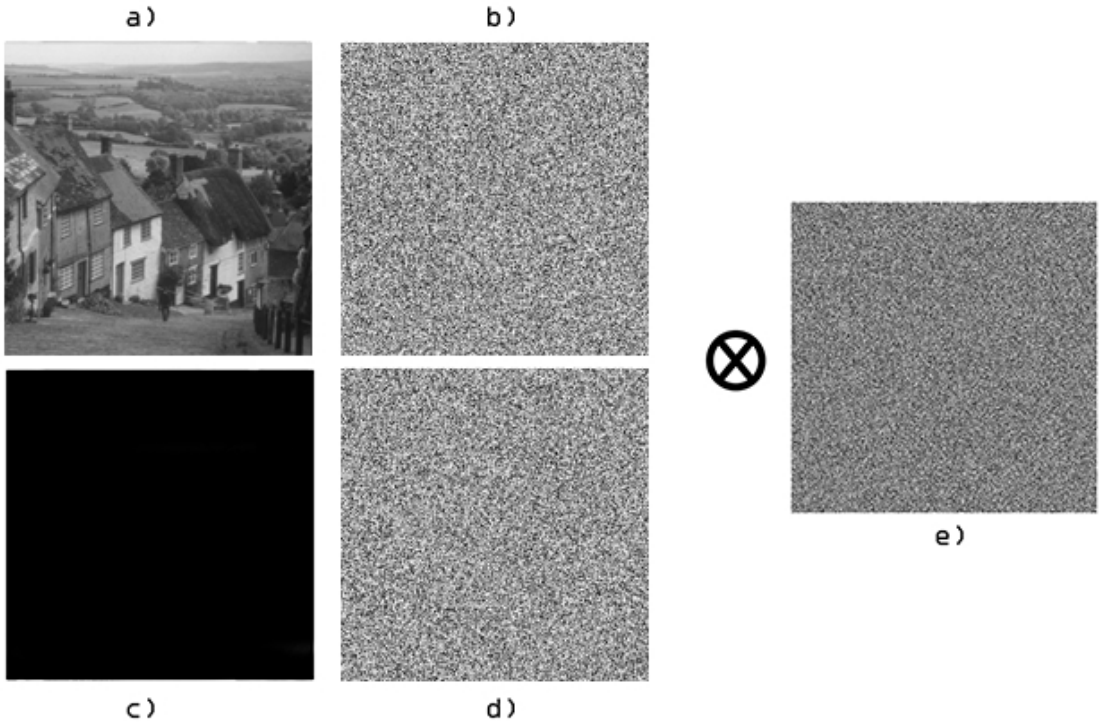

Figura 4.7: Prueba del Chosen Plain Image Attack, a) imagen de prueba Hills, b) versión cifrada de Hills por el sistema CSAC compacto, c) imagen Plana en color negro, d) versión cifrada de la imagen plana, e) resultado de la prueba chosen plain image attack

Sea I_P el texto plano y I_N la imagen que el atacante seleccionó en este caso una imagen en negro, que corresponde a una matriz de ceros y K la llave de cifrado. Entonces ambas imágenes se cifran con la misma llave y se obtienen las imágenes C_P y C_N. Al momento de cifrar la información, los sistemas de cifrado por lo general emplean una operación XOR para procesar la información, es de esta característica de la que se aprovecha el chosen plain image attack. Otro punto clave es que cuando se hace la operación XOR de dos valores iguales el resultado será cero. Por lo que la operación descrita da como resultado el texto plano.

Al observar los valores obtenidos en las pruebas por las modificaciones propuestas, podemos decir que los resultados son similares a los que se obtuvieron con la versión del sistema CSAC

[23] a excepción de la prueba SAC que mejoro su desempeño, con lo que se asegura un nivel de seguridad alto en el cifrado de imágenes y texto plano.

Sin embargo, la S-box no solo se usó como un complemento en un sistema de cifrado. Si no que también se le dio una aplicación para realizar un sistema VMEI, en subsección siguiente es explicado cómo se realizó la aplicación.

4.2.7 Aplicaciones en sistema VMEIS

De acuerdo con los resultados obtenidos en [32] se puede afirmar que la S-box propuesta en este trabajo resulta ser fuerte contra ataques de criptoanálisis, haciéndola una componente muy valiosa en los sistemas de cifrado en donde sea implementada. Sin embargo, por su comportamiento no lineal puede ser utilizada para otras aplicaciones que necesiten de componentes susceptibles a pequeños cambios para alterar completamente las salidas. Tales aplicaciones pueden ser el de restar correlación a imágenes, con esta idea se hace un aporte al trabajo presentado en [36] que consiste en un sistema denominado como Esquema de Imágenes Cifradas Visualmente Significativas o VMEIS por sus siglas en inglés, estos sistemas propuestos originalmente por Bao y Zhou [37], tratan de ocultar una imagen cifrada previamente, dentro de otra imagen sin cifrar con la intención engañar a los atacantes al hacerlos pensar de que se trata solo de una imagen sin importancia.

En la Figura 4.8 se observa cómo está constituido el sistema VMEIS presentado en [37]

mientras que en la Figura 4.9 se presenta la propuesta del trabajo [36] en donde se puede observar que se incorpora la S-box propuesta al sistema. La intensión de incorporar la S-box al sistema VMEIS es la de procesar la imagen de referencia (I_R) para restarle correlación y agregarle la propiedad de difusión, el siguiente proceso es introducir las columnas de la imagen a la S-box y este proceso se lleva acabo dos veces. Posteriormente se repite, pero ahora para las filas de la imagen. Debido a que las dimensiones de la S-box son de 16×16 por lo que el total

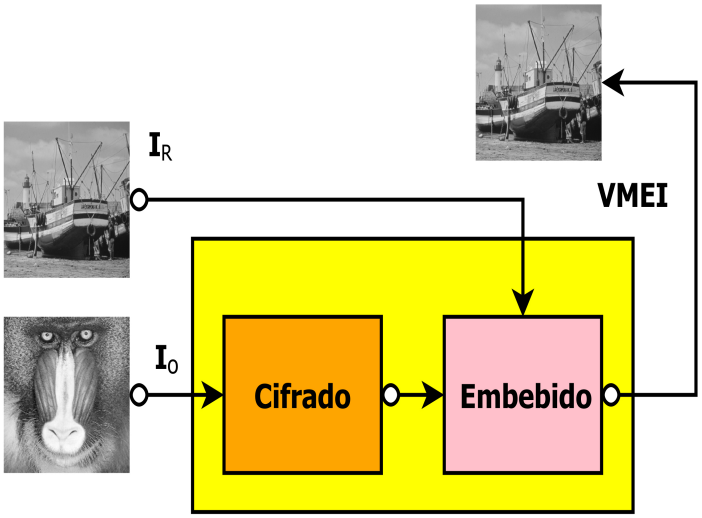

Figura 4.8: Diagrama de bloques del sistema VMEIS propuesto por Bao [37]

de coordenadas que puede procesar son 256, así que para imágenes que superen los 256×256 píxeles se necesitara dividir la imagen en secciones de 256×256 píxeles hasta completar el tamaño completo de la imagen.

A continuación, se hace una breve explicación de cómo funciona cada uno de los elementos del sistema VMEIS propuesto en [36].

El primer paso es tomar la imagen de referencia (I_R) y aplicarle una función threshold definido en la Ecuación 4.4. Este proceso se usa para delimitar los valores de los píxeles, ya que el siguiente paso es aplicar una transformada entera de Haar y algunos valores de pixel hacen que la transformada de Haar arroje valores fuera del rango de las imágenes (0 a 255),

$$f(n) = \begin{cases} \alpha, & \text{si } p(x,y) < \alpha \\ (255-\alpha), & \text{si } p(x,y) > (255-\alpha) \\ p(x,y), & \text{para cualquier otro resultado} \end{cases} \tag{4.4}$$

en la ecuacion 4.4 la $p(x,y)$ representa el valor del pixel y α es un peso que delimitara el umbral de acción.

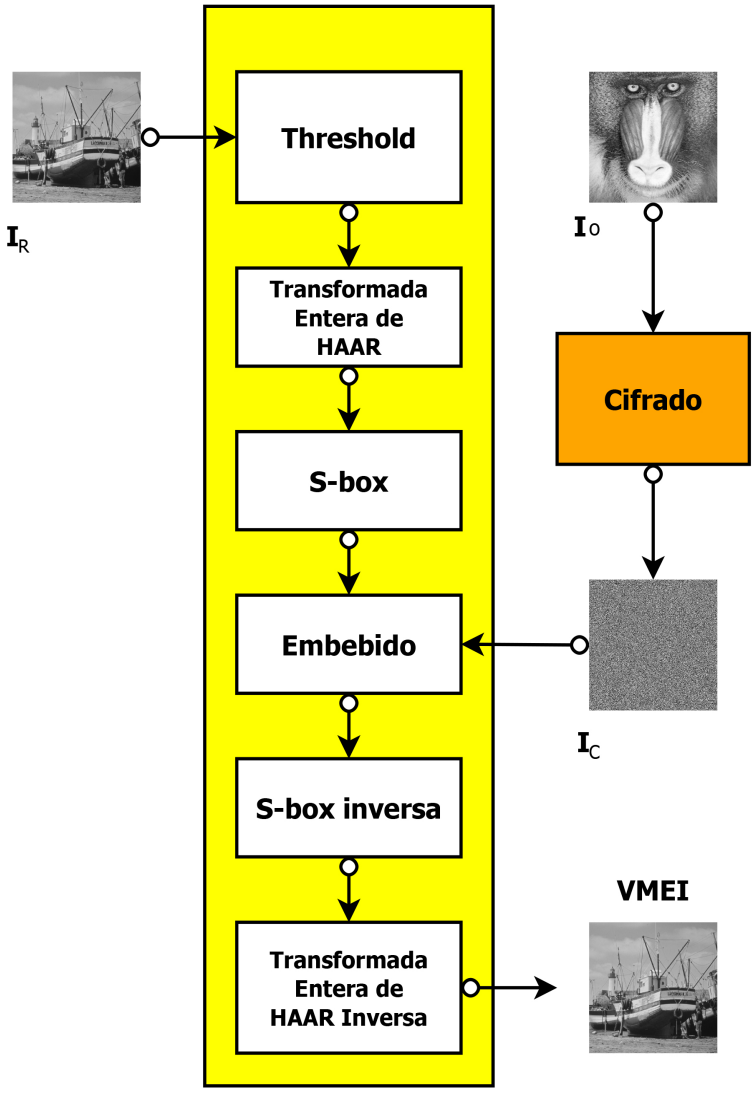

Figura 4.9: Diagrama de bloques del sistema VMEIS propuesto en [36]

El siguiente paso es aplicar la transformada wavelet entera (IWT), que es una variante especial de la transformada wavelet discreta (DWT), que se utiliza principalmente para el procesamiento y análisis de imágenes. Básicamente, la IWT transforma los valores enteros de los píxeles de una imagen en coeficientes wavelet enteros, que están en el mismo rango dinámico que la señal original sin pérdidas de información. Para poder calcular estos coeficientes se usan las siguientes

ecuaciones:

$$l(n) = \lfloor \frac{x[2n] + x[2n+1]}{2} \rfloor, \text{n } = 0, 1, ..., \frac{N}{2} - 1 \tag{4.5}$$

$$h(n) = x[2n] - x[2n+1], \text{n } = 0, 1, ..., \frac{N}{2} - 1 \tag{4.6}$$

una vez que se tiene la imagen de promedios y detalles que es el resultado de aplicar la IWT se procede a procesar columnas y filas con la S-box propuesta en [32] para generar una imagen con un patrón difuso.

A la par de que este proceso se lleva acabo, la imagen que se va a esconder (I_O) es cifrada por algún sistema de cifrado en el caso de [36] se usó el criptosistema CSAC [23, 12, 22] y la imagen cifrada es denominada como I_C. El paso siguiente es embeber las dos imágenes que se tienen en este punto, para poder hacer esto la imagen de referencia I_R debe tener el doble de las dimensiones de la imagen que se va a ocultar I_O. Para el proceso de embebido se aprovechan las 4 regiones de coeficientes de la imagen procesada por la S-box (Fig. 4.10), ya que cada región tiene las mismas dimensiones que I_O. El proceso para incrustar la imagen es muy sencillo, se toma un pixel de la imagen I_C y se pasa a su forma binaria $[b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0]$ siendo b_0 el bit menos significativo y después b_7, b_6 reemplazan a los bits menos significativos de la imagen procesada por la S-box en la banda HH, respetando la posicion que tenia en I_C. Y así sucesivamente con los bits b_5, b_4, b_3, b_2, b_1, b_0 en las bandas HL, LH, HH, respectivamente.

Una vez que ambas imágenes fueron embebidas se procede a aplicar el procesamiento que se hizo con la S-box pero ahora con su versión inversa y después se aplica la inversa de la transformada entera de Haar para recuperar I_O con la informacion oculta de I_C. A esta nueva imagen se le llama $VMEI$ que es la salida del sistema.

En este trabajo solo se muestra una pequeña parte de los resultados y el trabajo hecho para [36]. En la Fig 4.11 se muestra la imagen I_O que corresponde a un plano que se quiere ocultar dentro de la imagen de lena (I_R1), I_C corresponde a la versión cifrada de I_O y finalmente las imágenes

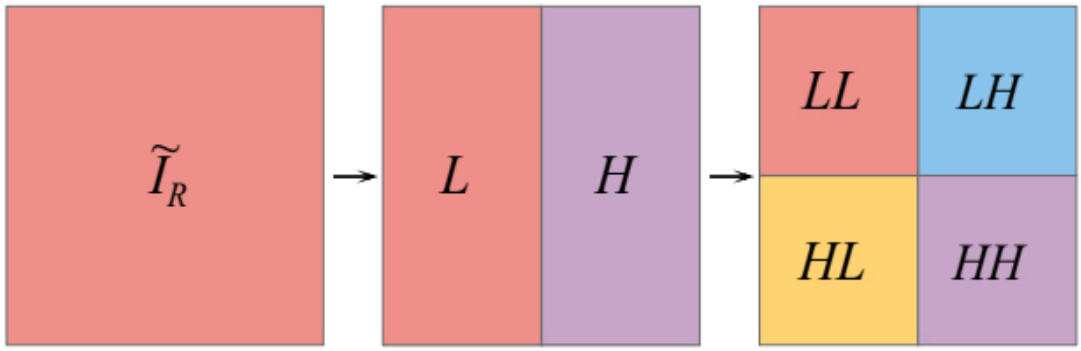

Figura 4.10: Ejemplo de comportamiento de la transformada Wavelet en sus niveles HH, HL, LH y LL.

$VMEI_1$ y $VMEI_2$, corresponden a las imágenes embebidas sin y con procesado de la S-box, respectivamente.

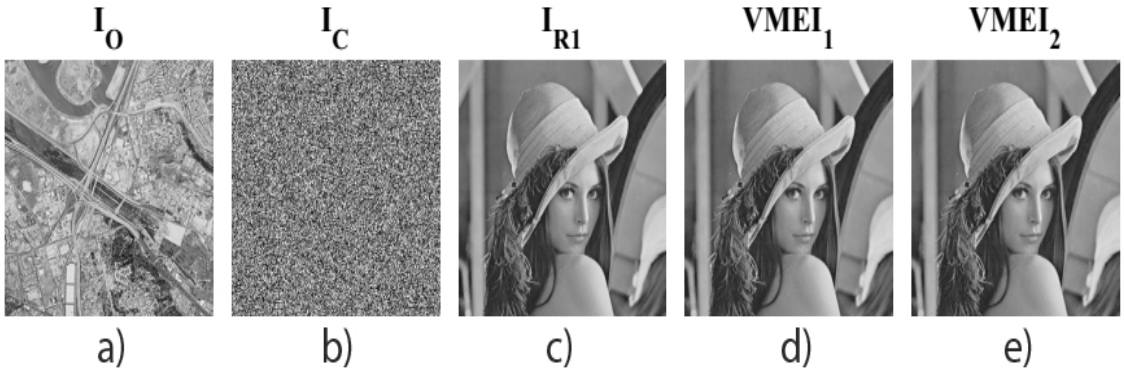

Figura 4.11: a) Imagen a esconder (I_O), b) versión cifrada por el sistema CSAC de I_O, c) la imagen de referencia lena (I_R), d) imagen embebida sin usar S-box y e) imagen embebida con usando la S-box

En la Fig. 4.12 se observan las imágenes I_R1, $VMEI_1$ y $VMEI_2$ junto a sus histogramas correspondientes. Y podemos observar que no se alteran en gran medida, por lo que si se usa o no la S-box no altera en gran proporción el resultado y hace que un atacante tenga mas dificultad de encontrar e interpretar la imagen oculta.

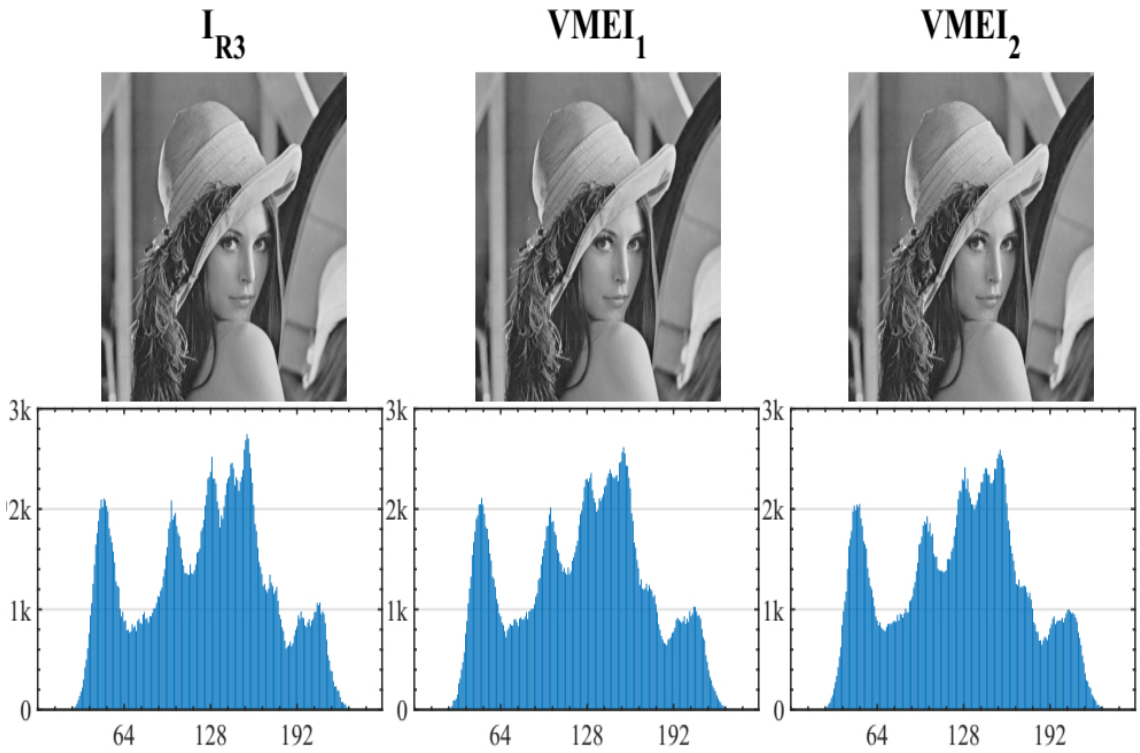

Figura 4.12: Las imágenes mostradas en la primer columna corresponden a la imagen original (imagen de prueba Lena) y su histograma, mientras que en la segunda columna se muestra la imagen de prueba embebida usando la s-box junto a su histograma y finalmente en la ultima columna se observa la imagen de Lena embebida sin usar la s-box y su histograma correspondiente.

Para obtener más información de las pruebas realizadas al sistema VMEIS y sus resultados, consultar la Referencia [36]}

5

Conclusiones

Al momento que se postula un sistema de cifrado nuevo, los diseñadores se enfocan en ciertos problemas poniendo atención en detalles que lleven a lograr una seguridad alta contra ataques de los que pudiera ser víctima el nuevo sistema de cifrado. Sin embargo, la evolución de las técnicas que permiten romper la codificación de los sistemas, se vuelve más sofisticada día a día, es por esto que los diseñadores de los sistemas de cifrado aplican nuevas pruebas para tratar de localizar vulnerabilidades que no se tenían en cuenta en el momento de diseñar sus sistemas de cifrado.

En este trabajo se presentó como al aplicar una prueba de criptoanálisis a un sistema de cifrado ya establecido (CSAC) y con una alta seguridad, se vio vulnerable ante una técnica que no se tenía contemplada al momento del diseñar el sistema de cifrado. Por lo que se tuvieron que tomar medidas para actualizar y reforzar el sistema cifrado contra este tipo de amenazas. Para lograrlo se planteó el uso el de una S-box que usando la misma arquitectura del sistema de cifrado CSAC, que son los autómatas celulares.

Al evaluar la caja de sustitución se obtuvieron resultados comparables con los que presentan por otras S-boxes de sistemas de cifrado robustos como lo son el AES o el DES. Y sin afectar a las otras pruebas ya realizadas en versiones anteriores del sistema CSAC. El diseño se basó

en una transformación no lineal sobre los campos finitos de Galois. Y según los datos que se presentaron en el capítulo 3 podemos constatar que la transformación propuesta presenta un buen desempeño para generar la componente no lineal de un sistema de cifrado e incluso en algunas pruebas con un desempeño superior al de las S-Box más fuertes.

También actualmente se está desarrollando una línea de trabajo en la cual se estudian las propiedades de la transformación propuesta para generar la S-box propuesta, puesto que al variar las condiciones iniciales es posible generar una serie de S-boxes que cumplen con el criterio de biyectividad y que, a su vez, al observar su fortaleza se comprueba que también tienen un buen desempeño contra ataques criptográficos.

A la par también se presentó que la S-box propuesta, podía ser implementada dentro de la estructura de un sistema VMEI obteniendo buenos resultados de ocultamiento de imágenes. Otra línea de trabajo futuro es el uso de la S-box propuesta en forma de mapeo de imágenes con el propósito de restar correlación a imágenes sin hacer uso de muchas iteraciones para así, generar nuevas alternativas a los mapeos de orden caótico que en la actualidad se usan, como el Arnold Cat Map o el Bakers Map.

Bibliografía

[1] Goldreich, Oded.: Foundations of Cryptography: Volume 2, Basic Applications. Vol. 2. Cambridge university press, 2004.

[2] Ibrahim A. Al-Kadi (April 1992), "The origins of cryptology: The Arab contributions?, Cryptologia 16 (2): 97:126

[3] Bowen, Jonathan P. (2019). "The Impact of Alan Turing: Formal Methods and Beyond". In Bowen, Jonathan P.; Liu, Zhiming; Zhang, Zili (eds.). Engineering Trustworthy Software Systems. SETSS 2018. Lecture Notes in Computer Science. 11430. Cham: Springer. pp. 202-235.

[4] Çokal, C., Solak, E.: Cryptanalysis of a chaos-based image encryption algorithm. Physics Letters A 373(15), 1357?1360 (2009) ISSN 0375-9601

[5] Fridrich, J.: Symmetric ciphers based on two-dimensional chaotic maps. International Journal of Bifurcation and Chaos 8(6), 1259?1284 (1998)

[6] Daemen, J., Rijmen, V.: The Design of Rijndael. Springer-Verlag New York, Inc. Secaucus, NJ, USA (2002)

[7] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray, S. Vo, A statistical test suite for random and pseudorandom number generators for cryptographic applications, National Institute of Standards and Technology (NIST), special publication 800-22, August 2008

[8] J. S. Murguía, G. Flores-Eraña, M. Mejía Carlos and H. C. Rosu, Int. J. Mod. Phys. C 23, 1250078 (2012)

[9] Biham E, Shamir A, Differential cryptanalysis of DES-like cryptosystems. J Cryptol 4(1):3-72, 1991

[10] Diffie, W.; Hellman, M. , New directions in cryptography, 22, IEEE transactions on Information Theory, pp. 644?654, (1976)

[11] Ramírez-Torres M.T., Murguía J.S., Mejía-Carlos M.. Image encryption with an improved cryptosystem based on a matrix approach. International Journal of Modern Physics C 2014;25(10):1450054.

[12] Murguía J.S., Flores-Eraña G., Mejía-Carlos M., Rosu H.C.. Matrix approach of an encryption system based on cellular automata and its numerical implementation. International Journal of Modern Physics C 2012; 23(11):1250078.

[13] J. Urías, G. Salazar and E. Ugalde, Synchronization of Cellular Automaton Pairs, Chaos, 8, 814-818, 1998.

[14] Ramirez-Torres, M. T., Murguia, J. S., and Mejia-Carlos, M. (2014, December). Fpga implementation of a reconfigurable image encryption system. In 2014 International Conference on ReConFigurable Computing and FPGAs (ReConFig14) (pp. 1-4). IEEE.

[15] Webster, A. F.; Tavares, Stafford E. (1985). «On the design of S-boxes». Advances in Cryptology - Crypto '85 218. New York, NY: Springer-Verlag New York, Inc. pp. 523-534

[16] J. von Neumann, The Theory of Self-Reproducing Automata, A.W. Burks (ed.), University of Illinois Press, Urbana, IL, 1966.

[17] M. Tomassini and M. Perrenoud, Cryptography with cellular automata, Appl. Soft Computing, vol. 1, pp. 151-160, 2001.

[18] Adamatzky,A.,2010.Game of Life Cellular Automata.Springer,London,p.xix,579p

[19] Wolfram, S.: Cryptography with cellular automata. In: Advances in Cryptology: Crypto:85, pp. 429-432. Lecture

[20] J. Urías, G. Salazar and E. Ugalde, "Synchronization of Cellular Automaton Pairs", Chaos, 8, 814-818, 1998.

[21] Mario Alberto Almazán Montelongo, Tesis de Maestria: "Encriptación Implementada en un FPGA de información comprimida con la transformada ondeleta". 2007.

[22] Marcela Mejía Carlos, Tesis de Doctorado "Encriptación por Sincronizacion en Automatas Celulares". 2001.

[23] M. T. Ramírez, J.S. Murguía and M. Mejía Carlos, Int. J. Mod. Phys. C25, 1450054, 2014.

[24] Benvenuto C.J., Galois field in cryptography. University of Washington, Seattle, 2012

[25] Szaban, M., Seredynski, F.: Dynamic cellular automatabased S-boxes. In: International Conference on Computer Aided Systems Theory, pp. 184-191 (2011)

[26] Lambi´c, D.: A novel method of S-box design based on discrete chaotic map. Nonlinear Dyn. 87, 2407-2413 (2017)

[27] Trant M. T., Bui D. K., Doung, A. D.: Gray S-box for advanced encryption standard. In: IEEE International Conference on Computational Intelligence and Security (CIS:08), pp. 253?258 (2008)

[28] Khan, M., Asghar, Z.: A novel construction of substitution box for image encryption applications with Gingerbreadman chaotic map and S8 permutation. Neural Comput. and Appl. (2016). https://doi.org/10.1007/s00521-016-2511-5

[29] Farwa, S., Shah, T., Idrees, L.: A highly nonlinear S-box based on a fractional linear transformation. Springerplus 5(1), 1658 (2016). https://doi.org/10.1186/ s40064-016-3298-7

[30] Kocarev, L.: Chaos-based cryptography: a brief overview. IEEE Circuits Syst. Mag. 1, 6?21 (2001)

[31] Seredynski, F., Bouvry, P., Zomaya, A.Y.: Cellular automata computations and secret key cryptography. Parallel Comput. 30, 753?766 (2004)

[32] Aboytes-González, J.A., Murguía, J.S., Mejía-Carlos, M. et al. Nonlinear Dyn (2018) 94: 2003. https://doi.org/10.1007/s11071-018-4471-z

[33] Paar, C., Pelzl, J.: Understanding Cryptography: A Textbook for Students and Practitioners. Springer-Verlag Berlin Heidelberg (2010)

[34] Hussain, I., Shah, T., Asif-Gondal, M., Ahmad-Khan, W., Mahmood, H.: A group theoretic approach to construct cryptographically strong substitution boxes. Neural Comput. Appl. 23, 97-104 (2013)

[35] Hussain, I., Shah, T., Asif-Gondal, M., Mahmood, H.: Generalized majority logic criterion to analyze the statistical strength of S-boxes. Z. Naturforsch. A **67a**, 282-288 (2012)

[36] J. O. Armijo-Correa, J. S. Murguía, M. Mejía-Carlos, V. E. Arce-Guevara and J. A. Aboytes-González. An improved visually meaningful encrypted image scheme. Opt. Laser Technol. (2020)

[37] L. Bao, Y. Zhou, Image encryption: Generating visually meaningful encrypted images, Information Sciences 324. 197-207. (2015)

[38] Wu, Y., Noonan, J. P., Agaian, S. 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.(2011)