UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE INGENIERÍA, ARQUITECTURA Y DISEÑO ENSENADA

PROGRAMA DE POSGRADO

MAESTRÍA Y DOCTORADO EN CIENCIAS E INGENIERÍA

DISEÑO DE UN ALGORITMO DE CIFRADO CAÓTICO Y SU IMPLEMENTACIÓN EN MICROCONTROLADOR PARA APLICACIONES EMBEBIDAS

TESIS

que para cubrir parcialmente los requisitos necesarios para obtener el grado de

DOCTOR EN CIENCIAS

presenta:

MIGUEL ÁNGEL MURILLO ESCOBAR

Ensenada, Baja California, México, Mayo de 2015.

UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

FACULTAD DE INGENIERÍA, ARQUITECTURA Y DISEÑO ENSENADA

DISEÑO DE UN ALGORITMO DE CIFRADO CAÓTICO Y SU IMPLEMENTACIÓN EN MICROCONTROLADOR PARA APLICACIONES EMBEBIDAS

TESIS

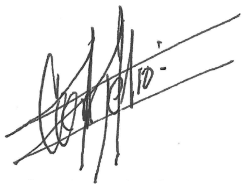
Que para obtener el grado de Doctor en Ciencias presenta:

MIGUEL ÁNGEL MURILLO ESCOBAR

Aprobada por el siguiente comité:

Dr. César Cruz Hernández
Director del comité

Dra. Rosa Martha López Gutiérrez
Miembro del comité



Dr. Cornelio Posadas Castillo
Miembro del comité

Dra. Liliana Cardoza Avendaño
Miembro del comité

Dr. Martín Velasco Villa
Miembro del comité

Ensenada, Baja California, México. Mayo de 2015

RESUMEN de la tesis de **Miguel Ángel Murillo Escobar**, presentada como requerimiento parcial para obtener el grado de DOCTOR EN CIENCIAS en ELÉCTRICA con orientación en CONTROL, del programa de Maestría y Doctorado en Ciencias e Ingeniería de la Universidad Autónoma de Baja California. Ensenada, Baja California, México. Mayo de 2015.

DISEÑO DE UN ALGORITMO DE CIFRADO CAÓTICO Y SU IMPLEMENTACIÓN EN MICROCONTROLADOR PARA APLICACIONES EMBEBIDAS

Resumen aprobado por:

Dr. César Cruz Hernández
Director de tesis

En este trabajo de tesis doctoral, se diseñó un algoritmo criptográfico basado en caos y se implementa en un microcontrolador de 32 bits para la transmisión y almacenamiento de datos de forma segura en aplicaciones embebidas.

Primeramente, se analiza la implementación de dos sistemas caóticos: Lorenz y logístico. Se opta por utilizar el mapa logístico en el algoritmo criptográfico ya que es un sistema discreto por naturaleza, tiene mejor desempeño y requiere menos recursos de implementación comparado con el sistema tridimencional de Lorenz; además, se verifica la existencia de caos mediante el estudio de exponentes de Lyapunov. Después, se diseña el algoritmo criptográfico basado en una ronda de la arquitectura de confusión y difusión. Finalmente, el algoritmo criptográfico propuesto se aplica en tres diferentes casos para probar su versatilidad: *1) Imagen a color RGB* (a nivel software), *2) Texto alfanumérico* y *3) Plantilla de huella dactilar*.

Se presentan varios análisis estadísticos de seguridad para validar el algoritmo criptográfico propuesto como error de desencriptado, espacio de claves, sensibilidad a la clave, sensibilidad a la entrada, histogramas, correlación, entropía de la información, frecuencia flotante, N-gramas, autocorrelación, pruebas estadísticas de aleatoriedad con FIPS-140-2 del NIST. Por otra parte, se presenta un análisis a nivel hardware como memoria utilizada, puertos de comunicación, frecuencia del sistema y tiempo de cifrado, para validar el uso del algoritmo propuesto en sistemas embebidos en base a microcontroladores para aplicaciones en tiempo real. Algunas aplicaciones que pueden ser de interés son la milicia, medicina, telecomunicaciones, biometría, informática, financiera, sismología, comercio electrónico, información personal, entre otros.

Palabras clave: cifrado caótico, análisis de seguridad, microcontrolador, aplicaciones embebidas.

Abstract of the thesis presented by **Miguel Ángel Murillo Escobar**, as a partial requirement to obtain the DOCTOR IN SCIENCE degree in ELECTRIC with orientation in CONTROL, of the program of Master and Doctorate in Science and Engineering of the Autonomous University of Baja California. Ensenada, Baja California, Mexico. May 2015.

DESIGN OF A CHAOTIC CIPHER ALGORITHM AND ITS IMPLEMENTATION IN MICROCONTROLLER FOR EMBEDDED APPLICATIONS

Abstract approved by:

Dr. César Cruz Hernández
Thesis director

In this PhD thesis, a cryptographic algorithm based on chaos is designed and implemented in a 32-bit microcontroller for secure data transmission and storage in embedded applications.

Firstly, the implementation of two chaotic systems are analyzed: Lorenz system and logistic map. We chose logistic map due it is a discrete system by nature, it has better performance, and it requires less implementation resources than Lorenz system; in addition, chaos existence is verified with maximum Lyapunov exponent. Then, the cipher algorithm is designed by using just one round of the confusion and diffusion architecture. Finally, the proposed encryption algorithm is applied in three different cases to prove its versatility: *1: Color image* (at software level), *2: Alphanumeric text* y *3: Fingerprint template.*

Several security analysis such as decryption error, key space, secret key sensitivity, plain input sensitivity, histograms, correlation, information entropy, floating frequency, N-grams, autocorrelation, random statistic, test with FIPS-140-2, encryption time, and hardware analysis as memory used, communication ports, frequency system, and cipher time, verify and validate the proposed cipher algorithm to be used in real-time embedded systems.

Some applications of interest are military, medicine, telecommunications, biometric, computation, industry, electronic payment, personal information, into others.

Keywords: chaotic cipher, security analysis, microcontroller, embeeded applications.

A mi familia

Agradecimientos

A Dios, por todas sus bendiciones y permitirme vivir con salud, dicha y felicidad.

A mi familia, mis padres Miguel y Sharo, mis hermanos Yahaira, Angelina, Araceli y Daniel, a mis sobrinos Milton, David, Sofía y Ximena por ser mi motor y mi alegría de cada día. En especial a mi esposa Maribel por todo su apoyo, cariño y amor incondicional y también a mis suegros Roberto y Clara y mi cuñado Adrián. Para mi hermano Daniel, que va siguiendo mis pasos y que le deseo lo mejor del mundo.

A la Universidad Autónoma de Baja California (UABC), por brindarme un espacio integro donde realizarme profesionalmente y seguir en el camino de la superación. En especial a la Facultad de Ingeniería, Arquitectura y Diseño Ensenada (FIAD), a mis profesores del doctorado, directivos y personal administrativo.

Al Consejo Nacional de Ciencia y Tecnología (CONACyT), por financiar mis estudios doctorales; también, por el apoyo económico brindado a través del Proyecto de Grupos de Investigación en Ciencia Básica, Referencia 166654.

Al Dr. César Cruz Hernández, por permitirme ser su discípulo y guiarme por el camino del conocimiento, por todo el apoyo brindado y por sus consejos. Gracias por sus sabias enseñanzas y por convertirme en una persona preparada íntegramente para el mundo de la investigación.

A la Dra. Rosa Martha López Gutiérrez, por todo su cariño, apoyo incondicional y consejos. Por estar al pendiente de mi durante mis estudios doctorales y ser parte de mi preparación profesional.

A mi comité de tesis, Dra. Liliana Cardoza Avendaño, Dr. Martín Velasco Villa, Dra. Rosa Martha López Gutiérrez y Dr. Cornelio Posadas Castillo, por ser parte de mi equipo de formación profesional, por sus comentarios, consejos y correcciones en el desarrollo de la tesis. Al Dr. Enrique Efrén García Guerrero por su apoyo al inicio de mis estudios doctorales.

Al coordinador del posgrado, Dr. Juan de Dios López Sánchez por sus asesorías administrativas.

A mis amigos, Salomón, Alma, Alondra, Salma, Gaby, Michelle, Eduardo, Fausto, Adrián, Aram, Elen, Oscar, Luis L, Kuotaro, Rogelio, Luis R. y Yazmín.

Ensenada, B.C., México.
Mayo de 2015

Miguel Ángel Murillo Escobar

Tabla de Contenido

Lista de Figuras

Lista de Tablas

Capítulo 1

Introducción

En la actualidad, miles de kilobytes de datos privados se transmiten diariamente a través de medios de comunicaciones inseguros (como internet, redes de computadoras, sistemas de comunicaciones, etc.) en distintas áreas que son susceptibles a robo de información, principalmente para fines de estafas, fraudes o bélicos (como en comercio electrónico, informática, sistemas biométricos, telemedicina, sismología, información personal, sistemas embebidos, entre otros), por lo que surge la necesidad de proteger la información en su almacenamiento y transmisión.

En los últimos años, los sistemas *no lineales* y especialmente los *sistemas caóticos* han generado mucho interés en muchas áreas científicas, por ejemplo en física, matemáticas, economía, biología, meteorología, ciencias sociales, medicina y astronomía, ya que muchos de sus sistemas se pueden describir de forma caótica. Otra aplicación muy interesante del caos, es en las comunicaciones seguras y la *criptografía*. El objetivo principal de la criptografía es garantizar la confidencialidad de la información almacenada o transmitida, es decir, garantizar que sólo el personal autorizado tenga acceso a dicha información.

Por otra parte, el criptoanálisis es la ciencia que se ocupa de corromper un sistema criptográfico y determinar el mensaje original a partir del mensaje cifrado o la clave de cifrado. Para ello, se utilizan análisis matemáticos y estadísticos que se conocen como ataques criptoanalíticos, los cuales varían según el método criptográfico implementado. Por tanto, un sistema criptográfico debe resistir los distintos ataques criptoanalíticos conocidos en la actualidad para que se considere seguro criptográficamente (Sec. 3.3). Además de la seguridad criptográfica, el sistema criptográfico debe ser eficiente para cifrar información a alta velocidad y que cada cifrado genere características similares de seguridad.

Desde 1930, máquinas mecánicas y electromecánicas de cifrado (basadas en criptografía clásica) se hicieron presente en la Segunda Guerra Mundial con el objetivo de establecer comunicaciones seguras a nivel militar, como la máquina electromecánica de rotores llamada Enigma utilizada por alemanes. Mientras que a nivel civil, comienzan en las décadas de los 70, donde se desarrolla, estandariza e implementa el famoso algo-

ritmo criptográfico *DES* (del inglés, *Data Encryption Standard*) para aplicaciones de comunicación electrónica segura para empresas financieras. Actualmente este algoritmo de cifrado es considerado inseguro para muchas aplicaciones debido a que utiliza una clave de cifrado de 56 bits, debido a que no resiste al criptoanálisis de un ataque de búsqueda exhaustiva donde se prueba cada posible clave para determinar el mensaje claro [1].

La metodología de los algoritmos criptográficos actuales (criptografía convencional) se basan en propiedades algebraicas y numéricas, como estructura de Feistel del *TDES* (del inglés, *Triple Data Encryption Standard*) y arquitectura de permutación y difusión del *AES* (del inglés, *Advanced Encryption Standard*). Frente a estos paradigmas, el *caos* pertenece al campo de los sistemas dinámicos *no lineales*, que manifiesta comportamientos "pseudoaleatorios" determinísticos, por lo que se clasifica en la criptografía caótica. Actualmente, se reconoce que los sistemas caóticos (representados matemáticamente por ecuaciones diferenciales o en diferencias no lineales) poseen muchas propiedades criptográficas como alta sensibilidad a condiciones iniciales y a parámetros de control, mezcla de datos, entro otros (Sec. 4.5), lo cual, hace que el caos sea muy interesante y efectivo para el cifrado de información.

A partir del trabajo de Pecora y Carroll en 1990, donde se reportó la sincronización de dos sistemas caóticos [2, 3], en los últimos años, se reportaron en la literatura varias técnicas de cifrado caótico *analógico* con base en la sincronización de caos, como *enmascaramiento caótico*, *conmutación caótica*, *modulación caótica*, entre otros, ver por ejemplo [4–24], donde la idea fundamental consiste en utilizar un sistema caótico en régimen caótico para generar una secuencia de banda ancha pseudoaletoria y la combina con el mensaje claro para producir una señal de aspecto incomprensible que se transmite a través de un canal inseguro. Mediante la sincronización, el sistema receptor produce la misma señal pseudoaleatoria para restarla con la señal que recibe del canal inseguro y recuperar el mensaje claro (ver figura 1.1).

Estos sistemas criptográficos son implementados mediante circuitos analógicos, tanto en el transmisor como en el receptor, empleando resistencias, condensadores, bobinas y amplificadores operacionales básicamente (ver figura 1.2). Los valores de las resistencias y condensadores determinan los valores de los parámetros de control de los sistemas caóticos y representan la *clave secreta* de cifrado. En el mercado, estos componentes se consiguen con una exactitud de 0.1 % como máximo, lo que ocasiona un pequeño desajuste entre los valores de los parámetros del transmisor y el receptor (además, los componentes presentan desgaste y el valor nominal varía con el tiempo). Debido a la alta sensibilidad que presentan los sistemas caóticos a los parámetros de control, este factor influye de forma negativa en dos aspectos de suma importancia [25]:

1. *Recuperación del mensaje claro en el receptor*. Por el desajuste natural de los componentes electrónicos, la sicronización caótica entre transmisor y receptor presenta un error que incrementa con el tiempo, es decir, $x \approxeq x^R$ en figura 1.1. Por este motivo, la recuperación del mensaje claro en el receptor es inexacta, es

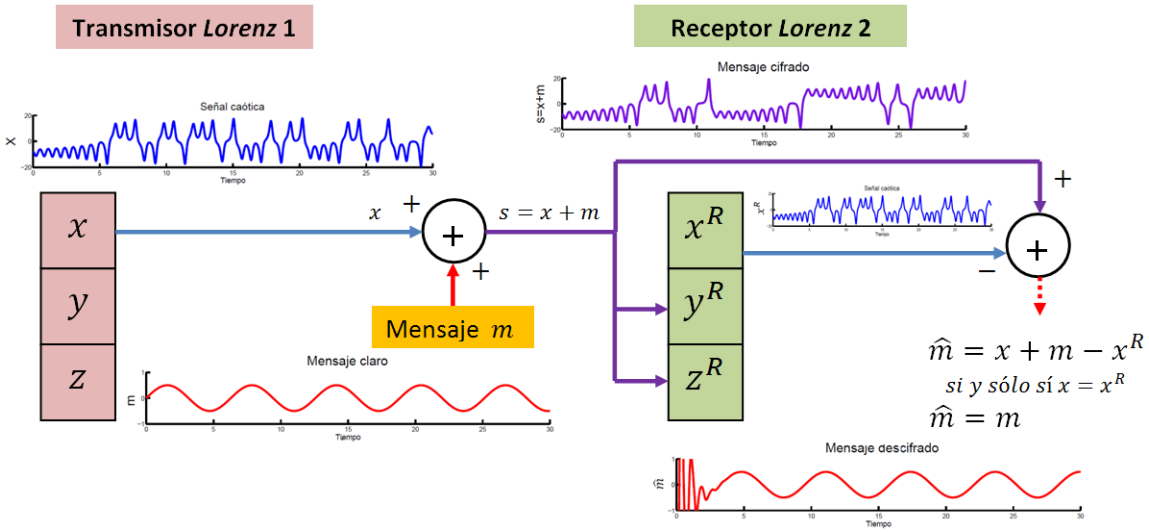


Figura 1.1: Esquema de cifrado *analógico* unidireccional empleando *enmascaramiento caótico* con dos sistemas de Lorenz [7].

decir, $\widehat{m} \approx m$. Este problema influye de forma negativa en aplicaciones donde se requiere que el mensaje descifrado sea exactamente el mismo que el mensaje claro, por ejemplo en la milicia, medicina, telecomunicaciones, biometría, sismología, entre otros.

2. *Seguridad criptográfica*. Un gran número de resultados criptoanalíticos a sistemas criptográficos basados en caos *analógico* se reportaron en la literatura y se mostró que no son suficientemente seguros desde el punto de vista *criptográfico*, ya que presentan problemas como baja sensibilidad a la clave secreta (tal vez el problema más serio [26]), espacio de claves reducido, fácil estimación de parámetros, fácil estimación de la señal portadora del sistema caótico maestro en técnicas de enmascaramiento caótico, extracción del texto claro de forma directa mediante filtrado, análisis de potencia, análisis de periodo corto, entre otros [27–30]. Aunque en la literatura se presentaron ciertas contramedidas, los sistemas criptográficos basados en caos *analógico* son considerados poco seguros en términos criptográficos [27], por lo que se requiere de mayor atención en la seguridad del cifrado en este tipo de sistemas de comunicación.

Algunos esquemas de cifrado analógico presentados son razonablemente veloces para cifrar información y simples de implementar, sin embargo, la mayor parte de los reportes no incluyen un análisis de seguridad para verificar el nivel de confidencialidad de la información cifrada. En general, se consideró al sistema criptográfico "seguro" por el hecho de utilizar un sistema caótico para cifrar información y de acuerdo con Kocarev [31], el uso de caos *analógico* en un sistema criptográfico no es garantía de seguridad. Para evaluar la seguridad de un sistema criptográfico, todos los ataques criptoanalíticos conocidos deben ser probados sobre el esquema de cifrado [27]. Cabe mencionar que la seguridad no es el único requerimiento para un esquema de cifrado práctico, también

se deben considerar otras propiedades como velocidad de cifrado y robustez al ruido externo.

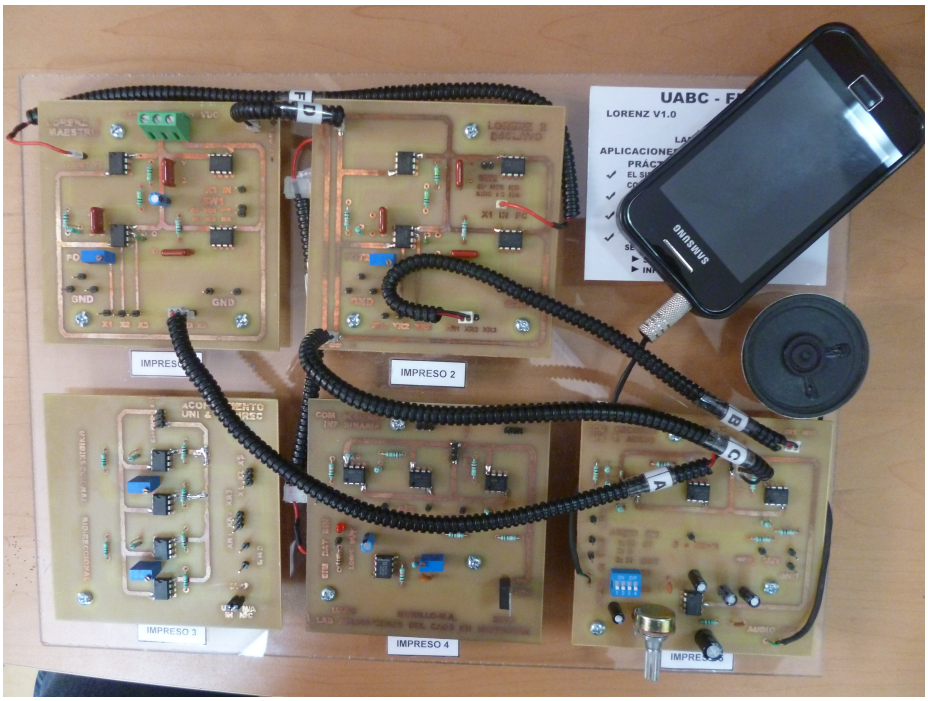


Figura 1.2: Sistema criptográfico basado en caos *analógico* implementado en circuitería electrónica para el cifrado de audio analógico.

Por otra parte, la aplicación de cifrado caótico *digital* se ha convertido en un campo emergente con altas expectativas en seguridad, flexibilidad, aplicabilidad y eficiencia criptográfica. Cifrado caótico *digital* no requiere el proceso de sincronización entre transmisor y receptor, por tanto, se evitan problema de seguridad que se presentan en sistemas criptográficos basados en caos *analógico*. Un sistema caótico es implementado en forma *digital* mediante soluciones numéricas si el sistema caótico esta representado por ecuaciones diferenciales (tiempo continuo) no lineales como Lorenz, Chua, Chen, Lu, CNN, Genesio-Tesi y Rössler, o el sistema caotico es implementado directamente si esta representado por ecuaciones en diferencias (tiempo discreto) no lineales como logístico 1D, logístico 2D, Hénon y Arnold-cat en una computadora, microcontrolador, FPGA, ASIC u otro sistema digital. Los valores de condiciones iniciales y parámetros de control de los sistemas caóticos (varia la cantidad según el sistema caótico que se utilice y si este es de una, dos o más dimensiones) constituyen la clave secreta (magnitudes digitales), de tal forma que *la dinámica caótica generada en el transmisor y en el receptor son idénticas*. Por tanto, los sistemas criptográficos basados en caos *digital* pueden ser más seguros y prometedores que los sistemas criptográficos basados en caos *analógico* por lo siguiente:

1. *Recuperación del mensaje claro en el receptor*. En un sistema de cifrado caótico digital se garantiza que $x = x^R$ (ver figura 1.1). Por tanto, el mensaje descifrado corresponde de forma exacta al mensaje claro, es decir, $\widehat{m} = m$ en la figura 1.1.

2. *Seguridad criptográfica*. La precisión digital de magnitudes digitales puede ser de 10^{-15}. Considerando esto y el diseño de un algoritmo criptogáfico que genere información cifrada con excelentes propiedades estadísticas de pseudoaleatoriedad, la criptografia basada en caos digital puede presentar mayor seguridad ante diversos ataques criptoanalíticos conocidos en la actualidad [32–51].

3. *Flexibilidad*. El algoritmo criptográfico se puede implementar en cualquier sistema digital, por ejemplo en microcontrolador, FPGA, ASIC, computadora, entre otros.

4. *Aplicabilidad*. El cifrado de datos aplica para cualquier tipo de información digital y también, para información analógica (con el uso de un convertidor analógico a digital.)

5. *Eficiencia criptográfica*. El cifrado de información digital es altamente eficiente al considerar costo *vs* velocidad de cifrado, ya que es posible cifrar datos a alta velocidad a un bajo costo.

En los últimos años, en la literatura se reportaron distintas técnicas de cifrado *digital*, por ejemplo, cifrado de flujo basado en PRNG caóticos (generación de números pseudoaleatorios) [32–34], cifrado caótico basado en la ergodicidad (seccionar en n partes la secuencia caótica) [35, 36], cifrado en bloque con cajas-S dinámicas caóticas (referente a la S-box del algoritmo AES) [37–39], entre otros. Algunos algoritmos de cifrado caótico con base a programación en MatLab se reportan en la literatura, para texto en [40], imágenes en [41–49] y datos biométricos en [50, 51]. Sin embargo, en su mayoría presentan serios problemas de seguridad y han sido vulnerados, ver por ejemplo [52–62].

El uso de caos *digital* tampoco es garantía de seguridad. La seguridad de un algoritmo de cifrado basado en caos *digital* consiste de un espacio de claves suficientemente grande para resistir un ataque exhaustivo, en el diseño de un algoritmo de cifrado que genere texto cifrado con excelentes propiedades estadísticas, que resista ante los ataques criptoanalíticos conocidos y que el algoritmo sea factible para aplicaciones en tiempo real [63].

1.1. Motivación

El crecimiento tecnológico de las últimas décadas ha traído múltiples beneficios a la humanidad, principalmente el desarrollo de nuevos sistemas de comunicaciones entre individuos y desarrollo de nuevos sistemas digitales embebidos para la recolección, control y transmisión de datos. Sin embargo, cualquier información transmitida a través de un canal inseguro es comprometida y puede ser robada por intrusos, lo que genera un problema de seguridad.

La *teoría de caos* se estableció desde 1970 en varias disciplinas científicas, como física, matemáticas, economía, biología, astronomía, ingeniería y química. Los sistemas

caóticos tienen muchas propiedades interesantes como ergodicidad, alta sensibilidad a condiciones iniciales, alta sensibilidad a parámetros de control, mezcla de datos, no linealidad, atractores extraños, dinámicas pseudoaleatorias, etc., que en su mayoría, están relacionadas con los requerimientos de Shannon para los procesos de permutación y difusión, para construir un sistema criptográfico seguro [64, 65]. Debido a la estrecha relación entre caos y criptografía [66–68], existe un gran interés en construir esquemas de comunicación segura con el uso de caos, para proteger información confidencial y evitar robo o acceso ilegal a la información en su transmisión y en su almacenamiento, en áreas como la milicia, telemedicina, biometría, informática, industria, información personal, entre otros.

De acuerdo con el principio de Kerckoff [69], la seguridad del sistema criptográfico debe recaer sobre la clave secreta y no sobre el proceso de cifrado, ya que este se considera de dominio público (criptografía moderna). Esto no significa desatender el proceso de cifrado, al contrario, este debe ser un proceso complejo. Por tanto, la clave de cifrado es de suma importancia, la cual, debe estar claramente definida y debe tener las combinaciones suficientes para resistir un ataque exhaustivo considerando el poder computacional actual.

De acuerdo con lo mencionado al inicio de este capítulo, los sistemas criptográficos basados en caos *analógico* tienen serios problemas de seguridad en el sentido criptográfico, ya que presentan desventajas como *problemas para la recuperación del mensaje claro de forma exacta*, un *espacio de claves reducido*, *baja sensibilidad a la clave secreta*, *fácil estimación del parámetro de control*, *fácil estimación de la señal portadora*, entre otros. Estos inconvenientes ocasionan que los sistemas criptográficos basado en caos *analógico* no sean criptográficamente confiables para su uso en aplicaciones prácticas.

Los problemas de seguridad que presenta la criptografía basada en caos *analógico* se ven grandemente reducidos con el empleo de la tecnología digital, por lo que, la criptografía basada en caos *digital* es un área de investigación emergente de los últimos años, con altas expectativas en eficiencia y seguridad criptográfica. Es de mucho interés público y científico, el diseñar e implementar nuevos sistemas criptográficos no convencionales, principalmente con el uso de sistemas caóticos, que sean seguros contra los ataques criptoanalíticos más poderosos reportados en la literatura y que en los últimos años han quebrantado un sin número de algoritmos de cifrado caótico digital, y que además sean eficientes para aplicaciones embebidas, principalmente para implementaciones biométricas con el uso de microcontroladores, donde los datos biométricos personales son susceptibles a robo de identidad en esquemas presentados recientemente en la literatura (ver sección 9.1).

Muchos de los primeros sistemas criptográficos basado en caos analógico presentados en la literatura no eran conscientes de los conceptos y estándares *criptográficos*, por lo que en su mayoría resultaron criptográficamente inseguros e ineficientes. Actualmente, muchos sistemas criptográficos basado en caos digital que fueron propuestos en la literatura, tuvieron en cuenta muchos principios criptográficos, sin embargo, no han

sido suficientes y siguen siendo inseguros e ineficientes, principalmente por problemas en el diseño del algoritmo criptográfico y el mal uso del caos. Por tanto, existe una gran necesidad de continuar en la investigación, desarrollo y construcción de nuevos sistemas criptográficos caóticos que sean altamente seguros y eficientes para aplicaciones en tiempo real.

1.2. Objetivos y alcances de la tesis

Debido al gran interés que ha surgido en la comunidad científica sobre la aplicación de algoritmos criptográficos no convencionales, en particular el cifrado caóticos, en la realización de este trabajo de tesis doctoral se planteó alcanzar el siguiente *objetivo general*:

Diseñar e implementar un algoritmo de cifrado caótico digital altamente seguro y eficiente, para su aplicación en sistemas embebidos.

Que para cumplir con el objetivo general, se plantea alcanzar los siguientes *objetivos particulares*:

1. Determinar el sistema caótico a utilizar para ser implementado en microcontrolador, con base a su desempeño y recursos físicos: sistema de Lorenz o mapa logístico.

2. Diseñar un algoritmo criptográfico con base en: clave simétrica, arquitectura de confusión y difusión y caos digital.

3. Describir los detalles del algoritmo criptográfico con base a los requerimientos básicos para sistemas criptográficos basados en caos.

4. Simular en MatLab el algoritmo de cifrado propuesto para la protección de imagen a color RGB, mostrar su flexibilidad para el cifrado de distintas imágenes y realizar su respectivo análisis de seguridad.

5. Implementar el algoritmo criptográfico en un sistema embebido, con base a un microcontrolador de 32 bits para el cifrado de texto alfanumérico y para el cifrado de datos biométricos (plantilla dactilar).

6. Evaluar la seguridad del cifrado caótico a nivel software (lógico) en cada implementación embebida, con análisis estadísticos, diferenciales y de desempeño, para determinar su potencial uso en aplicaciones embebidas.

El alcance de la presente tesis doctoral aplica de forma teórica y práctica. Se diseña un algoritmo criptográfico caótico con altas características de seguridad y eficiencia, lo cual, se verifica con un amplio análisis de seguridad; por otra parte, el algoritmo se implementa en tecnología digital de microcontrolador para aplicaciones embebidas. El

algoritmo criptográfico propuesto es utilizable para la transmisión y almacenamiento de información de forma segura.

Los datos cifrados son recolectados del microcontrolador mediante memoria USB para realizar los distintos análisis de seguridad, con base a simulaciones en MatLab.

Las contribuciones de este trabajo de investigación con relación a los objetivos mencionados, se pueden consultar en **Productos derivados de este trabajo doctoral** (Sec. 10.3).

1.3. Organización del manuscrito

Este trabajo de tesis doctoral esta compuesto por diez capítulos. En los siguientes párrafos se describe de manera breve el contenido en cada uno de ellos.

- **Capítulo 1:** se presenta la introducción de este trabajo de investigación, la motivación y los objetivos a alcanzar.
- **Capítulo 2:** en este capítulo, se describe al caos desde una perspectiva matemática. También, se presentan las propiedades de un sistema caótico. En particular, se describen dos sistemas caóticos: sistema de Lorenz y mapa logístico. En cada caso, se verifica el comportamiento caótico determinando el máximo exponente de Lyapunov, con base a simulaciones en MatLab.
- **Capítulo 3:** se presenta breve historia e introducción a la criptografía. Se describen los componentes de un sistema criptográfico y muestra su clasificación. Los datos cifrados deben cumplir con cierto grado de aleatoriedad de acuerdo con el estándar FIPS 140-2, los cuales, se menciona en este capítulo. Se muestran los dos tipos de cifrado caótico propuestos en la literatura: analógico y digital. Además, se describen los requerimientos y reglas básicas que un sistema criptográfico basado en caos digital debe cumplir.
- **Capítulo 4:** se proporciona una introducción a los sistemas embebidos, sus componentes y sus aplicaciones. Se presenta el microcontrolador utilizado en este trabajo de investigación. Los problemas de seguridad en sistemas embebidos complejos se presentan para motivar el uso de criptografía caótica. También, se presenta una revisión bibliográfica sobre la implementación de sistemas caóticos y criptográficos en microcontroladores.
- **Capítulo 5:** se reporta la implementación en microcontrolador del sistema de Lorenz y mapa logístico con programación en lenguaje C. Además, la existencia de caos se verifica con un estudio de exponentes de Lyapunov. También, se presenta la eficiencia y los recursos de implementación en cada caso.
- **Capítulo 6:** se presenta el algoritmo de cifrado propuesto y las consideraciones matemáticas para generar un cifrado con excelentes características de aleatorie-

dad para que pueda resistir los distintos ataques criptoanalíticos y para que sea eficiente.

- **Capítulo 7:** en este capítulo, se muestra la implementación en MatLab del algoritmo de cifrado propuesto, para *imágenes a color RGB*. Un análisis de seguridad mediante estadísticas valida la calidad del cifrado y una comparación con otro algoritmo lo ratifica.

- **Capítulo 8:** se presenta la implementación en microcontrolador del cifrado propuesto, para *texto alfanumérico*. La calidad del cifrado se valida con una serie de análisis de seguridad a nivel lógico (estadístico). Además, se presenta su robustez ante ataques poderosos reportados en la literatura, el desempeño y los recursos de implementación para validar su aplicación en sistemas embebidos.

- **Capítulo 9:** se presenta la implementación en microcontrolador del cifrado propuesto, para *plantilla de huella dactilar* con aplicación a sistemas embebidos de control de accesos físico o lógico. Un análisis completo de seguridad validan el cifrado caótico propuesto y en este caso, se muestra la calidad criptográfica con un análisis de aleatoriedad. Finalmente, la implementación digital se valida con un análisis de recursos físicos y eficiencia de cifrado.

- **Capítulo 10:** se mencionan las conclusiones de este trabajo doctoral de manera general, las publicaciones que generó este trabajo de investigación doctoral, las contribuciones más sobresalientes y algunas vertientes para trabajos futuros.

Capítulo 2

Caos

En este capítulo, se describe al *caos* desde una perspectiva intuitiva, matemática y gráfica, se presentan las características intrínsecas de un sistema caótico y las propiedades más importantes que posee para su uso en el cifrado. En particular, se describe el sistema caótico tridimensional de Lorenz (en tiempo continuo) y el mapa logístico unidimensional (en tiempo discreto). También, se calcula el máximo exponente de Lyapunov con base a simulaciones en MatLab para verificar la existencia de *caos*. Para mayores detalles sobre el tópico, los lectores interesados puedes consultar, por ejemplo [70, 71].

2.1. Introducción

Históricamente, el matemático francés H. Poincaré mostró los primeros descubrimientos de dinámicas caóticas con su trabajo de investigación en 1890, con el problema de los tres cuerpos celestes experimentando atracción gravitacional mutua (es decir, una estrella y dos planetas que giran desordenadamente entre ellos de forma circular, ver figura 2.1). Poincaré mostró que las dinámicas muy complicadas (ahora dinámica caótica) que generaba este sistema era posible, en el sentido de que una pequeña perturbación en el estado inicial de la posición del un cuerpo, podría llevar eventualmente a trayectorias radicalmente distintas. Posteriormente, en los años de 1920 al 1980, se desarrolló trabajo matemático sobre la dinámica caótica y la Teoría de caos por G. Birkhoff, M. Cartwright, J. Littlewood, S. Smale, A. Lyapunov, A. Kolmogorov, V. Arnold, E. Lorenz, R. May, M. Feigenbaum, entre otros matemáticos.

A pesar de los avances realizados sobre la dinámica caótica en aquellos tiempos, muy recientemente se apreció que muchos de los sistemas físicos reales se comportan de forma caótica. La principal razón fue la dificultad para diseminar los avances matemáticos a otras áreas de investigación y otra razón fue por que los teoremas mostrados no eran suficientes para convencer a los científicos de otra áreas, de que este tipo de comportamiento podría ser importante en sus sistemas bajo estudio. Actualmente, la situación ha cambiado drásticamente debido a que los sistemas dinámicos son analizados con el uso de computadoras. Avances en esta área, muestran que la dinámica caótica esta pre-

sente en una gran variedad de sistemas, por ejemplo en fluidos, plasmas, dispositivos de estado sólido, circuitos electrónicos, láseres, dispositivos mecánicos, biología, química, acústica y mecánica celestial [70].

Intuitivamente, las personas relacionamos el *caos* con adjetivos como desorden, confusión, desconcierto, enredo o desorganización. En matemáticas y otra ciencias, el *caos* se puede describir como un comportamiento impredecible de sistemas dinámicos *no lineales* que son *determinísticos*, presentan extrema sensibilidad a condiciones iniciales y a parámetros de control, presentan una dinámica aperiódica (nunca se repite exactamente) y generan un atractor extraño con dimensión fractal (estructura irregular que ser repite a diferentes escalas).

La no linealidad se puede mostrar de forma intuitiva al caracterizar el comportamiento de un sistema en términos de estímulo y respuesta: si a un sistema se le da una "patada" y observamos cierta respuesta a esa patada, después nosotros nos podemos preguntar que pasa si le damos una patada al sistema el doble de fuerte. Si la respuesta es doblemente mayor, el comportamiento del sistema se dice que es lineal (al menos para el rango de "patadas"). Si la respuesta no resulta en el doble de grande (que sea menor o que sea mayor), se dice que el comportamiento del sistema es no lineal [71]. Por ejemplo, la segunda ley de Newton donde dice que la fuerza de una partícula con masa m es igual a la masa m por su aceleración, en este caso, la fuerza es proporcional a la aceleración de la partícula con una masa fija m, por lo cual, es un sistema lineal. En la figura 2.2(a), se muestra un ejemplo de la segunda ley de Newton, donde se tiene una masa fija (automóvil), el cual, es empujado sobre una recta horizontal con la fuerza de un hombre para generar una aceleración de 10 km/h; mientra que, si a esta misma masa se le aplica la fuerza de dos hombres, la aceleración del automóvil sera 20 km/h (el doble).

Por otra parte, un pequeño cambio en el parámetro de entrada en un sistema no lineal resulta en cambios dramáticos en el comportamiento del sistema tanto cualitativa como cuantitativamente. Para ciertos sistemas no lineales, el comportamiento puede ser punto fijo (estable) para cierto valor, para otro valor muy pero muy similar, el comportamiento puede ser ciclo limite (periódico) y para otro valor muy cercano, puede presentar comportamiento caótico. Un sistema se dice que es *determinista*, si su evolución con respecto al tiempo se puede determinar completamente a partir de las condiciones iniciales y parámetros que describen dicho sistema.

Es importante mencionar que determinismo no es igual a predictibilidad. La evolución temporal de las ecuaciones diferenciales son deterministas desde el punto de vista formal, ya que las condiciones iniciales y parámetros de control determinan el futuro de su evolución temporal. En el caso de los sistemas caóticos, estos presentan alta sensibilidad a condiciones iniciales y parámetros de control y su evolución temporal es "predecible" a muy corto plazo (ver figura 2.6). En meteorología (sistema complejo que se comporta de forma caótica), la predicción del clima es fiable (precisa) a sólo cuatro días con los modelos computacionales actuales, sin embargo, a diez días, la "predicción"

es aproximada y muchas veces cambia de forma drástica (por la sensibilidad a condiciones iniciales).

El meteorólogo y matemático estadounidense Edward N. Lorenz es pionero en la *Teoría de caos*. En 1963 desarrolló un sistema de ecuaciones diferenciales no lineales (sistema dinámico) para la predicción del clima, sin embargo, en uno de sus análisis descubrió que el sistema generaba resultados muy diferentes si se variaban ligeramente las condiciones iniciales y además, al generar la gráfica de fase producía un atractor que no era punto fijo ni ciclo límite, el atractor tenía forma *extraña* (ver figura 2.4).

Caos ha generado mucho interés en áreas científicas, como en ingeniería, física, matemáticas, economía, biología, meteorología, ciencias sociales, entre otras. Sin embargo, recientemente se propuso la aplicación de *caos* en las comunicaciones seguras para brindar confidencialidad a información privada debido a que posee propiedades criptográficas muy interesantes [2].

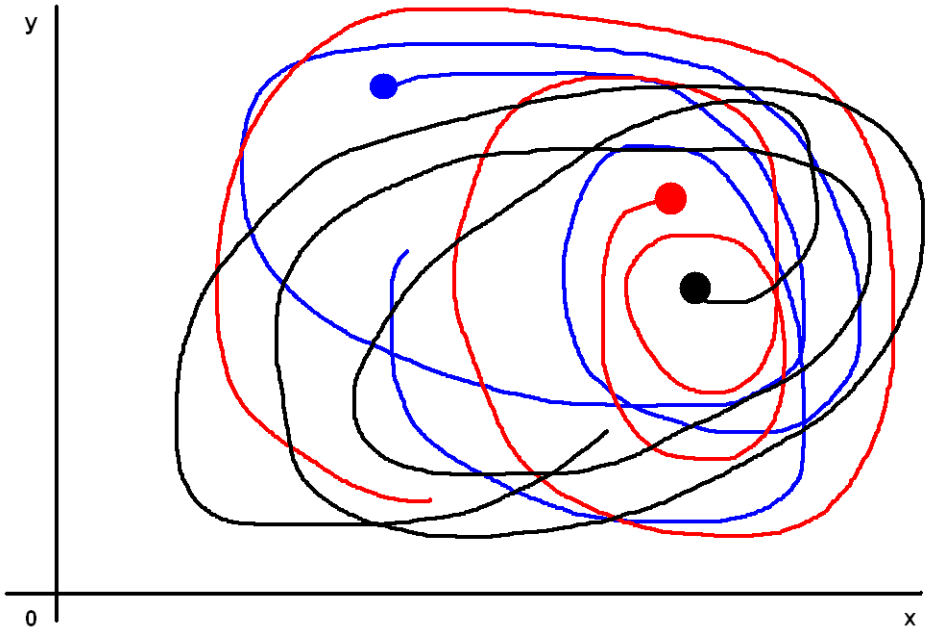


Figura 2.1: Ejemplo de la dinámica del problema de los tres cuerpos.

2.2. Sistema caótico y sus propiedades

En matemáticas y física, los sistemas dinámicos se pueden clasificar en estables, inestables o caóticos. Un sistema estable es aquel que genera fuerzas de atracción y lo mantiene confinado en un atractor de punto fijo u órbita periódica, mientras que, un sistema inestable generan fuerzas de repulsión que expulsa a la trayectoria fuera del atractor. En este sentido, un sistema caótico presenta ambas características, es decir,

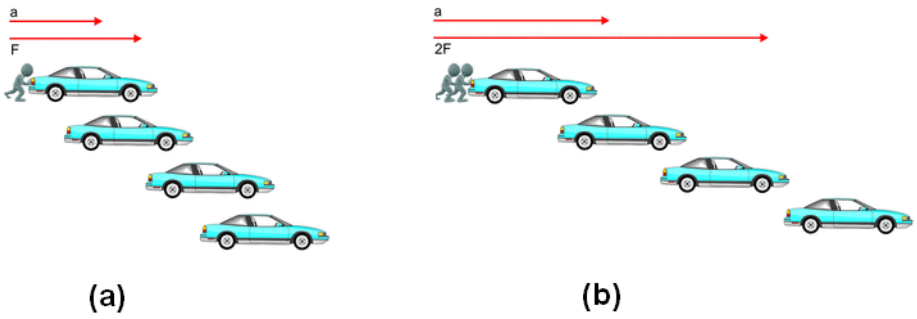


Figura 2.2: Ejemplo de un sistema lineal con la segunda ley de Newton.

existe un atractor que atrae la trayectoria del sistema, pero a la vez, hay otras fuerzas que lo alejan de este y permanece confinado en una zona del atractor para trazar una trayectoria extraña.

Un sistema caótico se puede describir por un conjunto de ecuaciones diferenciales o en diferencias no lineales, que generan secuencias caóticas que son deterministas, es decir, el valor futuro depende del valor actual y que además, presentan las siguientes propiedades:

- *No linealidad.* Son sistemas de ecuaciones diferenciales (tiempo continuo) o en diferencias (tiempo discreto) no lineales, que no cumple con el principio de superposición.
- *Sensibilidad exponencial a condiciones iniciales y parámetros de control.* La dinámica o trayectoria del sistema caótico se verá altamente modificada si se varía ligeramente una condición inicial o parámetro de control.
- *Mezcla de datos.* Un pequeño rango de condiciones iniciales cubre la mayor parte del espectro caótico.
- *Ergodicidad.* La trayectoria caótica se mantiene confinada en un espacio conocido como atractor extraño con respecto al tiempo cubriendo en su totalidad su espacio para cualquier entrada de condición inicial o parámetro de control.
- *Exponente de Lyapunov positivo.* Un sistema de dimensión N posee N exponentes de Lyapunov; si uno de ellos es positivo, el sistema es caótico; si dos o más son positivos, el sistema es hipercaótico.
- *Atractor extraño con dimensión fractal.* La gráfica de fase del sistema genera lo que se conoce como atractor, que puede ser punto fijo (sistema estable), ciclo límite (sistema periódico) o atractor extraño (sistema caótico).

Muchas de las propiedades mencionadas arriba están estrechamente relacionadas con propiedades criptográficas (Sec. 3.5), por lo que los sistemas caóticos se utilizan para la protección de datos privados en sistemas de comunicación inseguros.

2.3. Sistema caótico 3D de Lorenz

El sistema de Lorenz se conoce ampliamente en la literatura de teoría de caos y su comportamiento dinámico, está descrito por las siguientes ecuaciones diferenciales no lineales [72]:

$$\frac{dx}{dt} = \sigma(y - x), \tag{2.1a}$$

$$\frac{dy}{dt} = \rho x - y - xz, \tag{2.1b}$$

$$\frac{dz}{dt} = xy - \beta z \tag{2.1c}$$

donde x, y y z son los estados del sistema, x_0, y_0 y z_0 son las condiciones iniciales, σ, ρ y β son los parámetros de control y t es el tiempo. Con base en programación en MatLab, en la figura 2.3(a)-(c) se muestran los comportamientos temporales de los tres estados caóticos del sistema de Lorenz, a partir de las condiciones iniciales $x_0 = -8$, $y_0 = -8$ y $z_0 = 24$ y valores de parámetros de control $\sigma = 10$, $\rho = 8/3$ y $\beta = 28$. Mientras que la figura 2.4 muestra el atractor extraño generado por del sistema de Lorenz proyectado en el plano xyz.

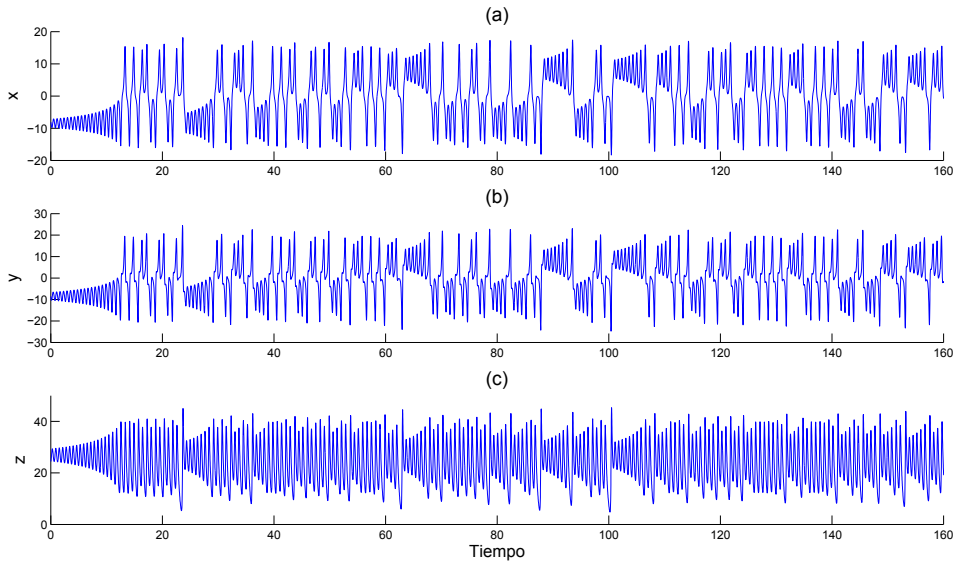


Figura 2.3: Comportamientos temporales del sistema caótico de Lorenz: a) estado x, b) estado y y c) estado z.

2.3.1. Máximo exponente de Lyapunov

La sensibilidad a condiciones iniciales de un sistema dinámico puede medirse a través del máximo exponente de Lyapunov, el cual, determina si dos trayectorias que inician extremadamente cercanas divergen con el tiempo [73]. El sistema de Lorenz es tridimensional (tres estados) por lo que tiene tres exponentes de Lyapunov (λ_1, λ_2 y λ_3)

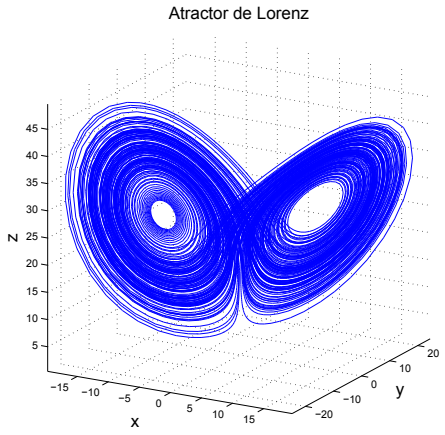


Figura 2.4: Atractor extraño generado por el sistema caótico de Lorenz ployectado en el plano xyz.

por cada estado del sistema caótico y si existe al menos un exponente de Lyapunov positivo, la secuencia generada es caótica.

El método que se implementa para calcular el máximo exponente de Lyapunov se llama *separación de órbita* [74]. El proceso consta de dos secuencias del sistema de Lorenz resuelto por RK4 (Lorenz "discreto", Sec. 5.2) S y S^* que inician con la misma condición inicial $S_0 = S_0^*$, hasta este punto ambas trayectorias son idénticas; después, S_0^* se perturba por una pequeña constante ϵ y se vuelve a determinar S^*. La distancia entre las dos soluciones numéricas se calcula y almacena. Posteriormente, el valor de S^* se cambia por las ecuaciones (2.2)(a)-(c), de tal forma que esté separado sólo ϵ de S_0, pero que S^* se mantenga en su dirección original:

$$S_k^* = S_k + (\frac{\epsilon}{D_k})C_k, \tag{2.2a}$$

$$D_k = \sqrt{(C_k)^2}, \tag{2.2b}$$

$$C_k = S_{a_k} - S_{b_k} \tag{2.2c}$$

donde $1 \leq k \leq T$ es el número de iteraciones, D_k es la separación de órbita entre S y S^* y C_k es sólo una resta entre la órbita S y S^*.

Hasta este punto, únicamente se tiene la solución numérica y el proceso de normalización, por lo que para determinar el máximo exponente de Lyapunov, es necesario hacer otros cálculos: en cada iteración se calcula el logaritmo natural de la separación relativa con ec. (2.3) y se suma el valor de cada iteración,

$$Sep = log(\frac{D_k}{\epsilon}) \tag{2.3}$$

donde Sep es la suma de los logaritmos en cada iteración. Al terminar las iteraciones

T, el mayor exponente de Lyapunov de la órbita S se calcula con la expresión

$$\lambda_1 = \frac{(Sep/T)}{\epsilon} \tag{2.4}$$

donde λ_1 es el *mayor exponente de Lyapunov*.

Con base en simulaciones en MatLab, se determina el máximo exponente de Lyapunov para los siguientes valores iniciales: $x_0 = -8$, $y_0 = -8$ y $z_0 = 24$ y valores de los parámetros $\sigma = 10$, $\rho = 8/3$, $\beta = 28$, $\epsilon = 1 \times 10^{-9}$, $T = 10,000$, con lo cual, se obtiene el mayor exponente de Lyapunov $\lambda_1 = 0.8$, lo que verifica la existencia de caos en la secuencia generadas en MatLab (fig. 2.3).

2.4. Mapa caótico 1D logístico

En 1976, Robert May estudió un modelo matemático no lineal en tiempo discreto, con el cual, explicó la dinámica poblacional de especies animales [75]. En su modelo, consideró la población proporcional entre 0 y 1, donde 0 representa cero individuos y 1 el máximo número de individuos que pueden existir; para la estimación de la población en un instante de tiempo, consideró la población en un instante de tiempo previo multiplicado por una constante (que depende del clima, alimento, ambiente, etc.) y esto a su vez multiplicado por 1 menos la población en un instante previo (a mayor población, esta crece con más dificultad). May encontró en su modelo soluciones punto fijo, periódicas y caóticas según el valor del parámetro. El hecho de que el modelo generaba dinámicas complejas deterministas para ciertos valores de la constante, generó gran interés en la comunidad científica y fue uno de los modelos matemáticos no lineales que fue base para estudios en la teoría de caos.

El mapa *logístico* unidimensional es conocido como el sistema no lineal *más simple* que existe y que exhibe claramente la ruta al caos, está descrito por la siguiente ecuación en diferencias [75]:

$$x_{n+1} = ax_n(x_n - 1) \tag{2.5}$$

donde $x_n \in (0, 1)$ es el estado del mapa discreto, x_0 es la condición inicial con valores entre $0 < x_0 < 1$ y a es el parámetro de control con $3.57 < a < 4$ para que el mapa genere secuencias caóticas.

En la figura 2.5 se muestra el comportamiento temporal del mapa logístico (2.5) a partir de la condición inicial $x_0 = 0.8$ y el valor del parámetro $a = 3.9$; se observa que la trayectoria es caótica con respecto al tiempo discreto.

2.4.1. Máximo exponente de Lyapunov

La secuencia caótica del mapa logístico se verifica con el exponente de Lyapunov; se tiene un exponente de Lyapunov ya que el mapa logístico es unidimensional (un estado). De forma similar a la vista en 2.3.1, se generan dos secuencias caóticas del mapa logístico

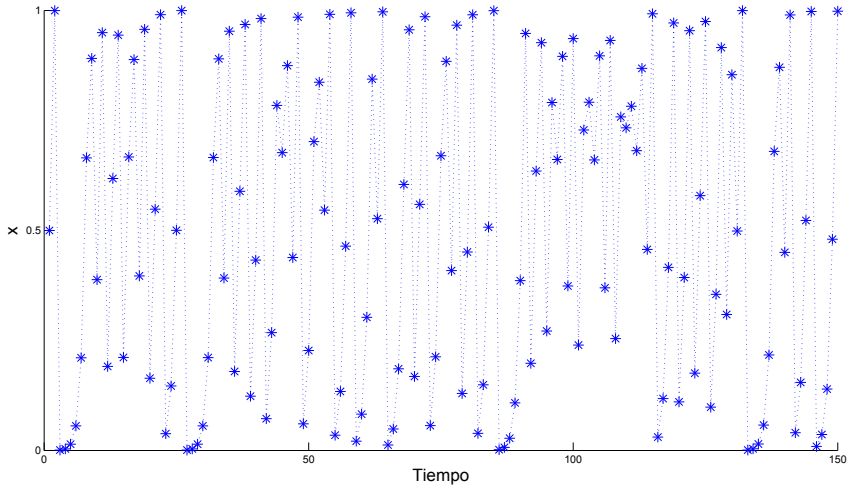


Figura 2.5: Comportamiento temporal del mapa logístico.

con el mismo valor del parámetro de control pero con condiciones iniciales muy cercanas.

El exponente de Lyapunov se determina con la siguiente expresión

$$\lambda = \frac{1}{T}\ ln\ |\frac{f^n(x_n - \delta_0) - f^n(x_n)}{\delta_0}| \tag{2.6}$$

donde λ es el exponente de Lyapunov, x_0 es una condición inicial, $x_0' = x_0 + \delta_0$ es otra condición inicial extremadamente cercana y T es el número de iteraciones.

Para el estudio, se utiliza como condición inicial $x_0 = 0.234567898765432$, una perturbación de $\delta_0 = 1 \times 10^{-13}$, número de iteraciones $T = 1,000$ y parámetro de control $a = 3.9$, para obtener el valor del exponente de Lyapunov $\lambda = 0.499$, para verificar que la secuencia es caótica.

En la figura 2.6(a), se muestran las primeras 100 iteraciones de la trayectoria de ambas secuencias caóticas que inician muy cercanas pero al pasar el tiempo, estas divergen; en la figura 2.6(b) se muestra el error entre dos trayectorias caóticas generadas por las condiciones iniciales $x_0 = 0.234567898765\mathbf{4}32$ y $x_0' = 0.234567898765\mathbf{5}32$.

2.5. Conclusiones

Las características y propiedades de sistemas caóticos fueron presentados en este capítulo, particularmente se presentó el sistema de Lorenz y el mapa logístico, por ser dos sistemas no lineales pioneros en la teoría de caos. También, se presentó el cálculo de los exponentes de Lyapunov en cada caso, para verificar la existencia de caos mediante simulaciones en MatLab. Las secuencia generada por el sistema caótico tiene características de pseudoalatoriedad, por lo que una de sus aplicaciones es la criptografía.

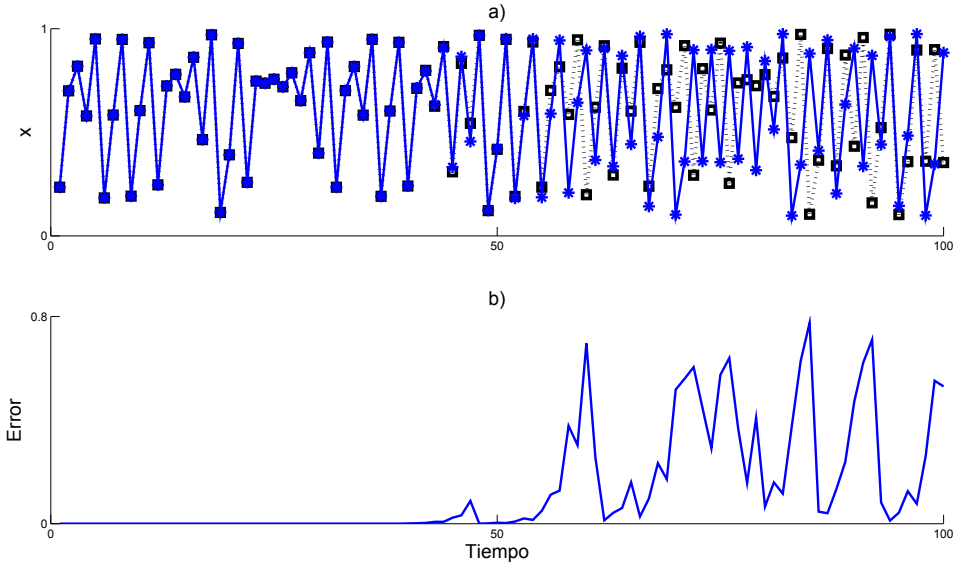


Figura 2.6: Sensibilidad del mapa logístico a condiciones iniciales: a) comportamiento temporal y b) gráfica del error.

Capítulo 3

Criptografía

En este capítulo, se presenta una breve historia de la criptografía, así como su definición y clasificación; también se muestran los componentes de un sistema criptográfico mediante un esquema y además, se mencionan las tres formas de vulnerarlo. Los datos cifrados deben cumplir con cierto grado de aleatoriedad de acuerdo con el estándar FIPS 140-2, los cuales, se menciona en este capítulo. Recientemente, los sistemas caóticos han sido utilizados en la protección de datos ya que poseen propiedades criptográficas interesantes. Desde hace dos décadas, en la literatura se reportan sistemas criptográficos basados en caos analógico y digital. Los sistemas criptográficos basados en caos digital son mas prometedores y seguros desde el punto de vista criptográfico y en este capítulo se describe porque. Además, se describen los requerimientos y reglas básicas que un sistema criptográfico basado en caos digital debe cumplir. Finalmente, se presentan algunas conclusiones del capítulo.

3.1. Introducción

A lo largo del tiempo, el hombre ha tenido la necesidad de transmitir información de manera secreta ya sea por motivos militares, diplomáticos, comerciales o personales, es por ello que nace la *criptografía*.

De manera histórica, desde el año 1500 a.C., métodos y artilugios para esconder información fueron creados por el hombre y clasificados ahora en criptografía clásica, algunos ejemplos son: las *tablillas de arcilla* utilizadas por comerciantes asirios; el instrumento de los griegos llamado *escítala* utilizada en el siglo V a.C. (ver figura 3.1), que consistía en una tela que se enrollaba en un cilindro de madera y que utilizaba la técnica de cambio de lugar; los escribanos hebreos utilizaban el alfabeto al revés conocido como *código espejo*; el *cifrado de Julio César* que consistía en sustitución de un símbolo por su tercer símbolo consecutivo, utilizado en el siglo I a.C.; *disco de cobre de cifrado* que básicamente consistía en el cifrado de Julio César (1466); *cifrado Vigenère* (1595); *cifrador de discos de Jefferson* (1790); *cifrador de Wheatstone* (1867); *cifrador de Bazeries* (1890) que combina el cambio de lugar y la sustitución de letras.

La criptografía clásica termina aquí y nace la **criptografía moderna** con máquinas mecánicas y electromecánicas, como la *Enigma* en 1918 construida por los alemanes para cifrar y descifrar mensajes mediante un método rotatorio electromecánico, utilizada generalmente en unidades de combate en la Segunda Guerra Mundial (ver figura 3.2); *máquina Purple* en 1940 construida por los japoneses similar a la Enigma; *máquina Sigaba* en 1940 desarrollada por los estadounidenses, similar a la Enigma; *máquina de Lorenz* en 1940 construida por los alemanes en comunicaciones militares de alto nivel durante la Segunda Guerra Mundial (no tiene que ver con el sistema de Lorenz, ya que éste fue construido en 1963). En 1949, Claude Shannon considerado padre de la criptografía matemática, publicó una serie de trabajos relacionados con la teoría de la información y comunicaciones que establecieron una base sólida teórica para la criptografía y el criptoanálisis, por lo que la criptografía pasó de ser un arte a ser reconocida como una ciencia [64, 65].

A partir de entonces, la invención de la computadora y el desarrollo de las matemáticas, permitieron que los nuevos algoritmos de cifrado fueran cada día más complejos, como el *DES* (del inglés, *Data Encryption Algorithm*), importante por ser el primer algoritmo de cifrado estándar en los años 70's y con ello el nacimiento de la criptografía simétrica, *AES* (del inglés, *Advanced Encryption Standard*), *RSA* (del inglés, *Rivest, Shamir y Adleman*), *IDEA* (del inglés, *International Data Encryption Algorithm*), entre otros.

Desde un punto de vista teórico, la *criptología* es la ciencia que trata las escrituras ocultas comprendida por las siguientes ramas [76]:

- **Criptografía**: ciencia que estudia las técnicas y métodos para transformar la información (texto claro) de manera que sea ilegible (texto cifrado) a personas no autorizadas, esta es uno de los servicios de la criptografía conocido como *confidencialidad*. También, la criptografía soporta otros servicios de seguridad como:
  - *Integridad:* la información no se altera durante su transmisión o almacenamiento.
  - *Autentificación:* verifica la identidad de quien envió la información.
  - *Vinculación:* garantiza que el emisor envió la información.
- **Criptoanálisis**: ciencia que se ocupa de analizar el algoritmo y texto cifrado, para encontrar el texto claro o la clave secreta; de manera que criptoanálisis y criptográfia son ciencias complementarias pero contrarias.
- **Esteganografía**: ciencia que estudia métodos y técnicas que permitan ocultar mensajes dentro de otros, conocidos como portadores, de tal manera que no se perciba su existencia y pase inadvertido.

.

La criptografía aplica en dos ámbitos de la seguridad informática: transmisión y almacenamiento de datos. En este trabajo de tesis doctoral, el algoritmo criptográfico propuesto aplica para ambos casos, en la transmisión de información confidencial a través de canales inseguros como internet y el almacenamiento de datos personales previamente cifrados en base de datos.

Figura 3.1: *Escítala*, instrumento criptográfico utilizada por los griegos en el siglo V a.C.

Figura 3.2: *Enigma*, máquina criptográfica utilizada por los alemanes en 1940.

3.2. Sistema criptográfico y su clasificación

Un *sistema criptográfico* consiste de un algoritmo usualmente conocido y que depende de un parámetro llamado clave secreta para cifrar (codificar) y descifrar (decodificar, proceso inverso del cifrado) información entre dos entidades (emisor y receptor) que están ubicados remotamente. En la figura 3.3 se muestra el esquema a bloques de un sistema criptográfico donde el canal de comunicación se considera inseguro. Un sistema criptográfico se compone con los siguientes cinco elementos [77]:

- m: representa el texto claro.
- c: representa el texto cifrado.
- K: representa la clave secreta.
- E: representa la función de cifrado.
- D: representa la función de descifrado.

En todo sistema criptográfico, se debe cumplir la siguiente condición

$$D_K(E_K(m)) = m \tag{3.1}$$

es decir, si un mensaje m se cifra con una función E y una clave K y después se descifra con la misma clave K, se obtiene el mensaje original m.

La clasificación de los sistemas criptográficos, según [77] es como sigue:

1. **Distribución de la clave secreta:**
    - *Simétricos o de clave privada:* Utilizan una clave K para cifrar y descifrar. La desventaja de estos sistemas de clave simétrica es que la transmisión de clave se debe gestionar de manera segura.
    - *Asimétricos o de clave pública:* Para evitar el problema de intercambio de clave secreta en un sistema simétrico, en este caso se utilizan dos claves, una pública K_p y otra privada K_q. La clave pública se utiliza para cifrar y la clave privada para descifrar. La ventaja es la seguridad en el manejo y tamaño de claves ya que no es necesario que dos personas se pongan de acuerdo en que clave utilizar, lo único que tiene que hacer el emisor es conseguir la clave pública del receptor. Sin embargo, se requiere mayor tiempo de cálculo, mayor espacio de memoria y claves mayores de 1,024 bits para considerarse seguro, por lo que se traduce en mayor costo y tiempo de cálculo, comparándola con criptografía simétrica. Algunos ejemplos de algoritmos criptográficos de clave asimétrica son: Rivest, Shamir y Adleman (RSA), Algoritmo de Firma Digital (DSA), Diffie-Hellman, criptografía de curva elíptica, Merkle-Hellman y Goldwasser-Micali.

- *Criptografía híbrida:* método criptográfico que se basa en la combinación de la criptografía simétrica y de la criptografía asimétrica; se utiliza el cifrado de clave pública para compartir una clave para el cifrado simétrico. La clave usada es diferente para cada sesión.

2. **Operaciones realizadas en el algoritmo:**

   Confusión y difusión son dos conceptos relacionados con la teoría de la información y las comunicaciones seguras, que aplican sólo para cifrado simétrico (el cifrado asimétrico se basa en operaciones matemáticas). Claude Shannon fue participe de ciertos postulados de los sistemas criptográficos para hacerlos más resistentes ante ataques estadísticos y uno de estos postulados, indica que un algoritmo criptográfico debe contar con procesos de confusión y difusión. Los algoritmos simétricos modernos como 3DES o AES implementan operaciones de confusión y difusión.

   - *Confusión:* Consiste en permutar o **cambiar de posición** cada elemento del texto claro (bit o bytes) de manera desordenada con respecto a la clave secreta y generar elementos cifrados. Objetivo: dificultar el descubrimiento de la clave con análisis estadístico.
   - *Difusión:* Consiste en **cambiar el valor** a cada elemento del texto claro de manera desordenada con respecto a la clave secreta, para transformarlo en otro elemento del mismo alfabeto y generar elementos cifrados. Objetivo: Ocultar cualquier relación estadística entre texto claro y texto cifrado.

3. **Estructura del algoritmo de cifrado:**

   - *Cifrado de flujo:* Cifran el mensaje bit a bit (o byte a byte) hasta terminarlo.
   - *Cifrado de bloque:* Cifran el mensaje en bloques de k bits simultáneamente hasta terminarlo. Algunos ejemplos son el DES, 3DES, RC5, AES, Blowfish e IDEA.

4. **Método para construir el algoritmo criptográfico:**

   - *Convencionales o tradicionales:* Las herramientas que se utiliza en la criptografía convencional para diseñar los algoritmos, son teoría de números, álgebra, curvas elípticas, etc. como ejemplo mencionamos los algoritmos criptográficos 3DES, AES e IDEA.
   - *No convencionales:* En este caso, se utilizan herramientas matemáticas en estado de investigación como la criptografía cuántica, la criptografía caótica y criptografía con ADN.

El algoritmo propuesto en este trabajo doctoral, recae en la clasificación no convencional y de criptografía simétrica, ya que utiliza caos y la misma clave para cifrar y descifrar. Además, una de las ventajas en la implementación digital es que requiere menos consumo de memoria, presenta mayor velocidad de cifrado y tiene un manejo práctico de clave secreta, en comparación con el método asimétrico.

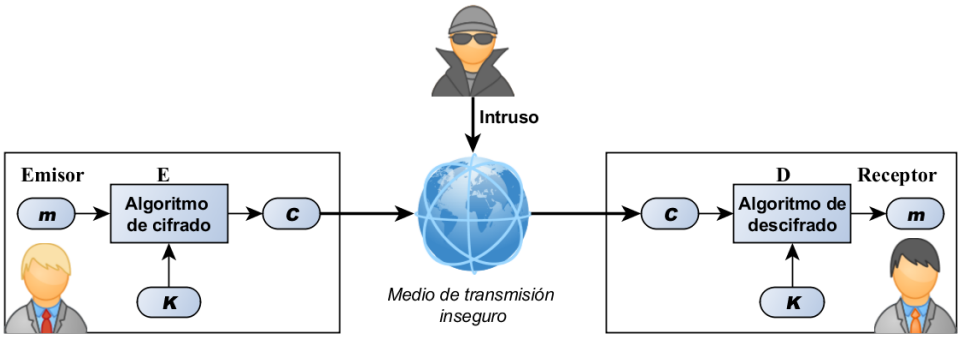


Figura 3.3: Esquema de un sistema criptográfico simétrico.

3.3. Seguridad de un sistema criptográfico

La seguridad de un sistema criptográfico debe recaer en la clave secreta y no sobre el algoritmo de cifrado, esto se conoce como principio de Keckhoff [69]. Por tanto, el algoritmo de cifrado se considera de conocimiento público.

Un sistema criptográfico se considera *vulnerado* si un criptoanalista encuentra la forma de determinar la clave secreta y en consecuencia el texto claro. Básicamente, existen tres formas de vulnerar un sistema criptográfico [78]:

1. **Ataques teóricos (lógicos):** Aplicar teoría de la información y criptoanálisis para quebrantar el algoritmo. La seguridad se evalúa mediante análisis basados en métodos matemáticos, donde se muestra que el mensaje cifrado y clave secreta no pueden ser revelados al implementar ataques conocidos (criptoanálisis actual).

2. **Ataques físicos:** Vulnerabilidades del sistema criptográfico, que se pueden aprovechar para quebrantar el algoritmo mediante ataques físicos, por ejemplo la información de tiempo, el consumo de energía, fugas electromagnéticas o incluso sonido, pueden proporcionar fuentes adicionales de información, que puede aprovecharse para romper el sistema criptográfico.

3. **Ataques humanos:** Presionar mediante sobornos u otros medios a personas que poseen información privilegiada.

Algunos ataques de tipo físico considerados poderosos son:

- *Ataque de sincronización:* Ataques basados en medir la cantidad de tiempo que diversos cálculos llevan a cabo.

- *Ataque de monitoreo de energía:* Ataques que hacen uso de diferentes consumo de energía en el hardware durante el cálculo.

- *Ataques electromagnéticos:* Ataques basados en la radiación electromagnética filtrada que pueden proporcionar directamente textos claros y otra información. Con un análisis de potencia se puede determinar la clave.

- *Análisis de sonido:* Ataques que se aprovechan de sonido producida durante un cálculo.
- *Remanencia de datos:* Los datos sensibles se leen después de haber sido supuestamente borrados.

Algunos ataques de tipo teórico considerados poderosos son:

- *Ataque exhaustivo o fuerza bruta:* Todas las posibles claves son utilizadas para descifrar un mensaje.
- *Sólo texto cifrado:* En este ataque el criptoanalista conoce el algoritmo y texto cifrado. Prácticamente es un ataque exhaustivo, por lo que se requiere que el espacio de claves sea suficientemente grande para resistir un ataque exhaustivo.
- *Texto claro conocido:* Es un ataque diferencial poderoso donde el criptoanalista conoce el texto claro de un texto cifrado y utiliza esta información para intentar determinar la clave secreta y así, descifrar otros criptogramas. Si el algoritmo cae ante este ataque, se considera no seguro.
- *Texto claro elegido:* Es otro ataque diferencial poderoso en donde el criptoanalista elije su propio texto claro para cifrar, posteriormente hace una ligera modificación (un bit) al texto claro y se vuelve a cifrar para determinar una relación entre la entrada y salida para determinar la clave secreta y descifrar otros criptogramas.

La seguridad del algoritmo criptográfico propuesto en esta tesis doctoral, es evaluada mediante análisis teóricos para resistir distintos métodos de ataques criptoanalíticos conocidos reportados en la literatura, como análisis de frecuencias, entropía de la información, histogramas, criptoanálisis diferencial, criptoanálisis estadístico, ataque por fuerza bruta, entre otros.

3.4. Pruebas estadísticas de aleatoriedad

La ref. [79] reporta un conjunto de pruebas estadísticas, utilizadas para determinar si una secuencia binaria s es puramente aleatoria, que en criptografía determina si el criptograma posee buenas propiedades estadísticas de aleatoriedad. Sin embargo, estas pruebas no garantizan un generador de bit aleatorio ya que son pruebas en base a probabilidades. Sea $s = s_0, s_1, s_2, ..., s_{n-1}$ una secuencia binaria de longitud n, las pruebas de Menezes *et al.* reportados en [79], se describen brevemente a continuación:

1. **Prueba monobit (de frecuencia):** El propósito de esta prueba consiste en determinar los números de 0's y 1's en s, estos deben ser aproximadamente los mismos en una secuencia aleatoria. Sea n_0 y n_1 los números de 0's y 1's en la

secuencia binaria s, respectivamente. La prueba estadística consiste en calcular X_1,

$$X_1 = \frac{(n_0 - n_1)^2}{n} \tag{3.2}$$

con distribución de probabilidad de Chi-cuadrada χ^2 con grado de libertad 1 si $n \geq 10$. En la práctica, se recomienda que la longitud de n sea mucho mayor, por ejemplo $n \geq 10,000$.

2. **Prueba poker:** Sea m un número positivo de manera que $(n/m) = 5 \times (2^m)$. La secuencia binaria s se divide en k secciones no traslapadas de longitud m y sea n_i el número de concurrencias de la i^{ma} posibilidad de longitud m, $1 \leq i \leq 2^m$. La prueba *poker* determina si el número de las secuencias de longitud m aparece el mismo número de veces en s, como se esperaría en una secuencia aleatoria. La prueba estadística utilizada se expresa como

$$X_2 = \frac{2^m}{k} \left(\sum_{i=1}^{s^m} n_i^2 \right) - k \tag{3.3}$$

la cual, sigue una distribución de probabilidad de Chi-cuadrada χ^2 con $2^m - 1$ grados de libertad. Cabe notar, que la prueba *poker* es una generalización de la prueba *monobit*, pero en este caso se utiliza $m \geq 1$.

3. **Prueba runs:** El propósito de la prueba de *runs* es determinar los números de *runs* de ceros y unos, de varias longitudes en una secuencia s. El número esperado de *gaps* o *blocks* de longitud i en una secuencia aleatoria de longitud n es $e_i = (n-i+3)/2^{i+2}$. Sea B_i y G_i el número de *blocks* y *gaps*, respectivamente, de longitud i, para $1 \leq i \leq k$. La prueba estadística se realiza mediante la expresión

$$X_3 = \sum_{i=1}^{k} \frac{(B_i - e_i)}{e_i} + \sum_{i=1}^{k} \frac{(G_i - e_i)}{e_i} \tag{3.4}$$

con aproximadamente una distribución de probabilidad de Chi-cuadrada χ^2 con $2k - 2$ grados de libertad.

4. **Prueba serial:** El propósito de esta prueba de aleatoriedad es determinar los números de ocurrencias de 00, 01, 10 y 11 como subsecuencias de s, que en una secuencia aleatoria serian aproximadamente los mismos. Sea n_0, n_1 los números de 0's y 1's en s, respectivamente y sean n_{00}, n_{01}, n_{10} y n_{11} los números de ocurrencias de 00, 01, 10 y 11, respectivamente. Notar que $n_{00} + n_{01} + n_{10} + n_{11} = (n - 1)$ ya que se permite que las secuencias se traslapen. La expresión para realizar esta prueba es mediante

$$X_4 = \frac{4}{n-1} \left(n_{00}^2 + n_{01}^2 + n_{10}^2 + n_{11}^2 \right) - \frac{2}{n} \left(n_0^2 + n_1^2 \right) + 1 \tag{3.5}$$

con aproximadamente una distribución de probabilidad de Chi-cuadrada χ^2 con 2 grados de libertad si $n \geq 21$.

El estándar FIPS 140-2 del *NIST* (del inglés, *National Institute of Standards and Technology*) de Estados Unidos, especifica cuatro pruebas estadísticas de aleatoriedad para validar, mencionadas en su mayoría en los párrafos anteriores [80, 81]. Una cadena de bits de $n = 20,000$ bits de longitud esta sujeta a cada una de las siguientes pruebas y si una falla, la secuencia no pasa la prueba:

1. *Prueba monobit:* El número n_1 de 1's en s debe satisfacer $9725 \leq n_1 \leq 10275$.

2. *Prueba poker:* la estadística X_2 definida por la expresión (3.3) se calcula para $m = 4$ y 5000 segmentos de 4 bits continuos. La prueba pasa si $2.16 \leq X_2 \leq 46.17$.

3. *Prueba runs:* Los números B_i y G_i de *blocks* y *gaps*, respectivamente, de longitud i en s se calculan para cada i, $1 \leq i \leq 6$. Se pasa la prueba si el número de B_i y G_i, entra en el rango de la tabla 3.1.

4. *Prueba long runs:* La longitud del mayor *run* en la secuencia de 20,000 bits (para 0's y 1's) debe ser menor de 26.

Las pruebas de aleatoriedad mencionadas arriba, son utilizadas para validar el alto "desorden" que genera el cifrado caótico propuesto en esta tesis doctoral. En la implementación en microcontrolador de cifrado de plantilla dactilar descrito en el capítulo 9, se muestra que mediante la prueba FIPS 140-2, el algoritmo de cifrado supera las pruebas de aleatoriedad de monobit, poker, runs y serial. Esto indica que el algoritmo de cifrado propuesto en esta tesis doctoral genera un criptograma con excelentes características de aleatoriedad y un criptoanálisis estadístico no tendría éxito.

Longitud	Intervalo
1	$2315 \leq X_3 \leq 2685$
2	$1114 \leq X_3 \leq 1386$
3	$527 \leq X_3 \leq 723$
4	$240 \leq X_3 \leq 384$
5	$103 \leq X_3 \leq 209$
6	$103 \leq X_3 \leq 209$

Tabla 3.1: Intervalos para *prueba runs* de acuerdo con FIPS 140-2.

3.5. Cifrado caótico

El **cifrado** de información consiste en transformar un mensaje claro en un mensaje ilegible mediante un algoritmo y una clave secreta, de tal manera que sólo la persona autorizada pueda decodificar y entender. Como se mencionó, en la criptografía moderna se han construido algoritmos de cifrado convencionales como el 3DES, AES, IDEA, etc., que utilizan operaciones algebraicas y un canal de comunicación digital (común) bidireccional entre emisor y receptor (internet). Sin embargo, en la actualidad se están investigando algoritmos no convencionales como:

- **Criptografía cuántica**: Se basa en el principio de incertidumbre de Heisenberg, esto es, que al observar un sistema cuántico éste se perturba a si mismo, e impide que el observador conozca su estado exacto antes de la observación. Por tanto, si un sistema cuántico se utiliza para transferir información, alguien que quiera espiar la comunicación, o incluso el receptor previsto, podría verse impedido de obtener toda la información enviada por el emisor. La criptografía cuántica hace uso de dos canales de comunicación entre los dos participantes. Un canal cuántico, el cual, tiene una sola dirección y que generalmente consiste en una fibra óptica. El otro es un canal convencional, público y bidireccional. Para la transmisión de información, los datos binarios son codificados mediante fotones.

- **Criptografía ADN**: Las características del ADN como habilidad para almacenar mucha información, paralelismo y poco consumo de potencia, hacen del ADN utilizable en la seguridad criptográfica. Operaciones de ADN basadas en suma, complemento, eliminación e inserción, son utilizadas actualmente para el cifrado de datos.

- **Criptografía caótica**: Se basa en ecuaciones no lineales diferenciales o en diferencias, las cuales, generan secuencias desordenadas o caóticas, pero que son deterministas y que presentan sensibilidad a condiciones iniciales. No existe una fórmula simple que defina a un sistema caótico en cualquier punto dado, lo que se califica como un problema muy difícil de resolver, lo que es una ventaja para su aplicación en la criptografía, eliminando las desventajas fundamentales de la criptografía convencional. Las secuencias que produce un sistema caótico son utilizadas como referencia para transformar un texto claro a un texto cifrado mediante un algoritmo y una clave secreta que esta relacionada con las condiciones iniciales.

Los sistemas caóticos tiene ciertas propiedades interesantes que sirven para generar operaciones de confusión y difusión eficientes, que son requeridas por un sistema criptográfico, estas propiedades de los sistemas caóticos están relacionadas con propiedades criptográficas de la siguiente forma [78]:

- **Ergodicidad *vs* confusión**: La densidad de los datos caóticos (distribución) ayudan a obtener una confusión optimizada, es decir, una secuencia caótica puede generar valores para cambiar de posición la mayoría del texto claro.

- **Sensibilidad a condiciones iniciales y parámetros de control *vs* difusión**: Un pequeño cambio en la entrada puede causar un gran cambio en la salida, ya que los datos caóticos divergen con pequeños cambios en las condiciones iniciales y parámetros de control, por lo que la difusión se ve ampliamente alterada.

- **Mezcla de datos *vs* difusión**: Un pequeño rango de condiciones iniciales y de parámetros de control en un sistema caótico, pueden generar muchísimas posibilidades de cifrado.

- **Dinámica determinista *vs* pseudoaleatoriedad**: La dinámica caótica determinística genera un comportamiento tipo pseudoaleatorio.

- **Complejidad de la estructura *vs* complejidad del algoritmo**: La no linealidad de los sistemas caóticos los hacen complicados de resolver, lo que ayuda ante ataques al algoritmo criptográfico.

Actualmente, en la literatura se reportan dos clases de implementaciones de criptográfia basada en caos [27, 78]:

- **Analógica**: A partir del trabajo de Pecora y Carroll en 1990 donde se reportó la sincronización de dos sistemas caóticos [2], en los últimos veinte años, en la literatura se reportaron muchos métodos de cifrado caótico analógico que se basan en la técnica de sincronización de caos como enmascaramiento caótico, conmutación caótica, modulación caótica, entre otros, ver por ejemplo [4–24], donde se considera como clave secreta el valor de los parámetros representados por magnitudes fijas mediante resistencias electrónicas. Sin embargo, la mayor parte de los reportes no incluyen un análisis de seguridad para verificar el nivel de confidencialidad de la información cifrada. En general, consideró al sistema criptográfico "seguro" por utilizar un sistema caótico para cifrar. Sin embargo, el uso de caos en un sistema criptográfico no es garantía de seguridad [31]. Para evaluar la seguridad de un sistema criptográfico, todos los ataques criptoanalíticos conocidos deben ser probados sobre el esquema de cifrado [27]. Un gran número de resultados criptoanalíticos a sistemas criptográficos basados en caos analógico se reportaron en la literatura y se mostró que no son suficientemente seguros desde el punto de vista criptográfico, ya que presentan problemas como baja sensibilidad a la clave secreta (tal vez el problema más serio) [26], espacio de claves secretas reducido, fácil estimación de parámetros, fácil estimación de la señal portadora del sistema caótico maestro en enmascaramiento caótico, extracción del texto claro de forma directa mediante filtrado, análisis de potencia, análisis de periodo corto, entre otros. Aunque se ha presentado ciertas contramedidas en la literatura, los sistemas criptográficos basados en caos analógico se consideran poco seguros criptográficamente [27], por lo que se se requiere de mayor atención en la seguridad del cifrado en este tipo de sistemas de comunicación.

- **Digital**: En este caso, no se requiere el proceso de sincronización implementada en los sistemas criptográficos basados en caos analógico, por lo que se evitan los problemas de seguridad que estos presentan. Por otra parte, los sistemas caóticos son implementados en su forma digital mediante soluciones numéricas (para el caso de sistemas caóticos en tiempo continuo, por ejemplo Lorenz, Chua, Chen, etc.) o directamente (si estos son de naturaleza discreta, por ejemplo, logístico, Hénon, etc.) en sistemas digitales, como en una computadora, microcontrolador, FPGA, ASIC, etc. En este caso, los valores de condiciones iniciales y parámetros de control de los sistemas caóticos constituyen la clave secreta (magnitudes digitales), de tal forma que *la dinámica caótica generada en el transmisor y en el receptor son idénticas*. En este sentido, los sistemas criptográficos basados en caos *digital* son más seguros y prometedores que los sistemas criptográficos basados en caos *analógico*. En los últimos años, la aplicación de cifrado caótico *digital* se ha convertido en un campo emergente con altas expectativas en seguridad, flexibilidad y

eficiencia criptográfica. En la literatura se reportaron distintos métodos de cifrado digital como cifrado de flujo basado en PRNG caóticos (generación de números pseudoaleatorios) [32–34], cifrado caótico basado en la ergodicidad (seccionar en n partes la dinámica caótica) [35, 36], cifrado en bloque con cajas-S dinámicas caóticas (referente a la S-box del algoritmo AES) [37–39], entre otras. Algunos algoritmos de cifrado caótico con base a programación en MatLab se reportan en la literatura, para texto en [40], imágenes en [41–49] y datos biométricos en [50, 51]. Sin embargo, en su mayoría presentan serios problemas de seguridad y han sido vulnerados, ver por ejemplo [52–62].

3.6. Requerimientos básicos de un cifrado caótico digital

Alvarez y Li en [78], presentaron una seria de reglas que un sistema criptografico basado en caos digital debe incluir. Los puntos que deben describirse cuidadosamente son los siguientes:

1. Se debe describir que sistema caótico utiliza el cifrado.
2. La degradación digital sebe ser evaluada, en caso de que se discretize un sistema continuo.
3. El sistema criptográfico debe ser fácil de implementar con base a costos aceptables y buena velocidad de cifrado.
4. La clave secreta debe ser claramente definida.
5. El espacio de claves debe ser especificada sólo para generar secuencias caóticas.
6. El efecto avalancha debe producirse para cualquier clave secreta: alta sensibilidad a la clave secreta.
7. Información parcial de la clave secreta, no debe revelar información parcial del texto claro, tampoco de la parte de la clave desconocida.
8. El proceso para generar secuencias caóticas a partir de la clave secreta debe estar claramente definido.
9. El cifrado debe tener alta sensibilidad al texto claro.
10. EL cifrado debe generar un texto cifrado con distribución de probabilidad uniforme.

Por otra parte, el sistema criptográfico debe resistir los siguientes ataques criptoanalíticos (de tipo lógico), con base al principio de Kerckhoffs; es decir, se conoce todo sobre el sistema criptográfico, excepto la clave secreta:

1. *Ataques diferenciales*. Son ataques del tipo solo texto claro elegido y conocido, donde se debe mostrar alta sensibilidad del sistema criptográfico a la clave secreta y al texto claro, para que el sistema criptográfico pueda resistirlos.

2. *Ataques estadísticos*. Son ataques de histogramas y correlación, donde se debe mostrar mediante análisis de correlación e histogramas, la uniformidad del texto cifrado, para resistir estos ataques.

3. *Ataque exhaustivo*. Son ataques donde se tratan todas las posibles combinaciones de claves, por lo que, la clave debe contener más de 2^{100} opciones [78].

En el capítulo 6, se define al algoritmo de cifrado caótico propuesto en este trabajo de tesis doctoral, considerando los requerimientos y reglas básicas de un sistema criptográfico basado en caos digital, con base a Alvarez y Li [78].

3.7. Conclusiones

La cripografía se encarga de cifrar información de tal manera que sólo personas autorizadas puedan decodificar y entender. Métodos de cifrado no convencionales como la criptografía caótica están actualmente en investigación, donde se utilizan las propiedades que presentan los sistemas caóticos para diseñar algoritmo criptográficos. En la literatura se reportan sistemas criptográficos basados en caos analógico con serios problemas de seguridad y también se reportan sistemas criptográficos basados en caos digital con perspectiva de mayor seguridad criptográficamente. Sin embargo, en la mayoría de las propuestas se omiten análisis de seguridad para validar su uso en sistemas de comunicaciones seguras

Capítulo 4

Sistemas embebidos

En este capítulo, se describe brevemente a los *sistemas embebidos* y se mencionan sus principales aplicaciones en gran parte de los sistemas electrónicos que nos rodean. Generalmente se basan en una unidad central llamada microcontrolador, la cual, se encarga de la gestión de las distintas tareas del sistema embebido mediante su programación en software y comunicaciones entre sus periféricos de entrada/salida. Aplicaciones recientes de estos sistemas son implementados en ambientes inseguros con almacenamiento y transmisión de datos a través de internet. Debido a lo cual, la información confidencial debe ser asegurada mediante métodos criptográficos para evitar robo de datos privados, alteración de umbrales, robo de identidad biométrica, entre otros. Finalmente, se mencionan algunas implementaciones en microcontroladores de algoritmos criptográficos que utilizan caos reportados recientemente en la literatura.

4.1. Introducción

Un *sistema embebido* (o *empotrado*) es un sistema de computación software-hardware, que realiza tareas específicas generalmente en tiempo real, con aplicaciones en sistemas de control de accesos, cajeros automáticos, máquinas expendedoras, lavadoras, microondas, televisiones, máquinas de fax, teléfonos, robótica, medición y control de variables físicas como temperatura, presión, humedad, etc., frenado de un automóvil, señales de tránsito, control de tráfico aéreo, equipos médicos (medicina), medición y control de procesos vía internet, sistemas industriales, entre muchas otras aplicaciones (ver figura 4.1). Los sistemas embebidos utilizan un procesador relativamente pequeño, memoria limitada y consume poca energía.

IBM (por sus siglas en ingles, *International Business Machines*) fue la primera compañía en elaborar los primeros equipos embebidos en 1980. Los sistemas embebidos se pueden programar con distintos lenguajes como ensamblador, C/C++, JAVA, entre otros. El factor costo *vs* efectividad *vs* potencia es la principal ventaja de los sistemas embebidos comparados con sistemas de computación completos como una computadora personal.

Figura 4.1: Algunas aplicaciones de sistemas embebidos.

Los componentes de un sistema embebido son básicamente una unidad central de procesamiento (como microcontrolador, procesador digital de señales (DSP), arreglos de compuertas lógicas programables (FPGA), circuitos integrados de aplicación específica (ASIC), entre otros), memoria (como Flash, memoria de acceso aleatorio (RAM), memoria de sólo lectura (ROM), etc.) y módulos de entrada y salida de datos (como teclado, pantalla , LED's, motor, relevadores, sensores, internet, etc.).

Es importante resaltar que el uso de sistemas embebidos en aplicaciones complejas, implica el desafío de brindar seguridad para proteger la información contenida en el sistema embebido y también la información que se va a transmitir mediante internet. De manera que se requiere incluir funciones criptográficas en los sistemas embebidos.

4.2. Microcontrolador

Un *microcontrolador* es un circuito integrado que incluye en su interior las tres unidades funcionales de una computadora: procesador, memoria y periféricos de entrada/salida (ver figura 4.2). Se trata de un computador completo en un sólo circuito integrado, el cual, es programable mediante *software* con un conjunto de instrucciones que se ejecuta de manera secuencial para realizar tareas específicas, desde las más sencillas hasta las más complejas.

En 1971, se presentó el primer microcontrolador de 4 bits por la compañía *Intel*.

Desde entonces, se han desarrollado en distintas arquitecturas de 4, 8, 16 ó 32 bits diseñados para reducir el costo económico y el consumo de energía de un sistema en particular. Actualmente, en el mercado existe una gran gama de microcontroladores para cada aplicación específica como en la aeronáutica, automotriz, industrial, etc. donde empresas como *Atmel*, *Freescale*, *Intel*, *Microship* y *Texas Instruments* son fabricantes de los principales microcontroladores en el mercado.

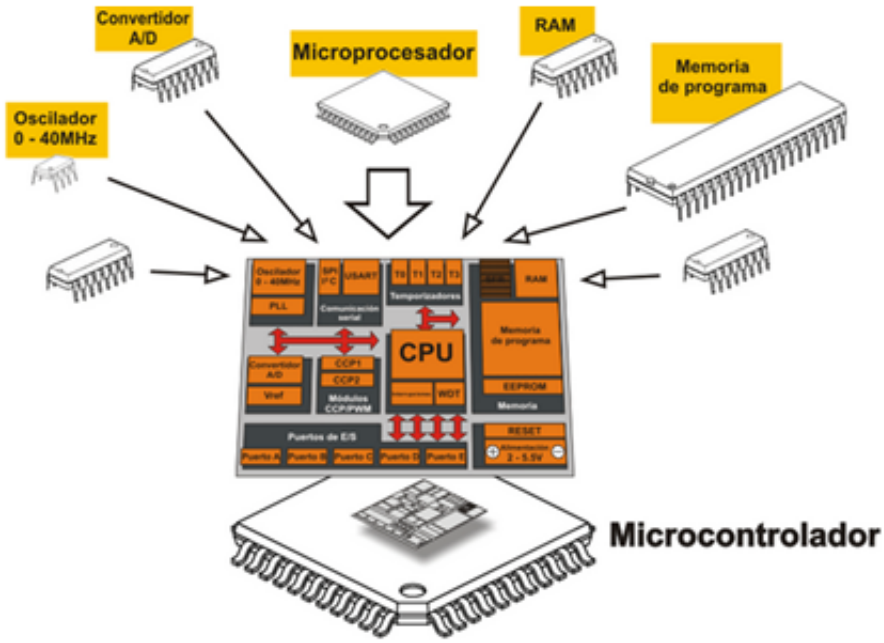


Figura 4.2: Estructura de un microcontrolador.

En las aplicaciones experimentales realizadas en este trabajo doctoral, se utiliza el microcontrolador de 32 bits MCF52259 de *Frescale* integrado un una tarjeta M52259DEMOKIT (ver figura 4.3). La familia de microcontroladores MCF5225X consiste de dispositivos altamente integrados con comunicación en chip de USB, Ethernet, CAN y funciones de cifrado. Están basados en un procesador de 32 bits ColdFire con una velocidad hasta de 80 MHz, memoria flash de 512 MB y 64 KB de SRAM.

Además, posee interfaz externa que proporciona flexibilidad para agregar memoria adicional. Éstas y otras características hacen del microcontrolador MCF5225X ideal para aplicaciones en control industrial, redes de internet industriales, sector salud, seguridad, que requieren una amplio rango de conectividad y alto desempeño. Se utiliza programación en lenguaje C en el software CodeWarrior 7.1 de Freescale con soluciones de software MQX 3.5 para alta integración en comunicación USB y Ethernet en tiempo real [82].

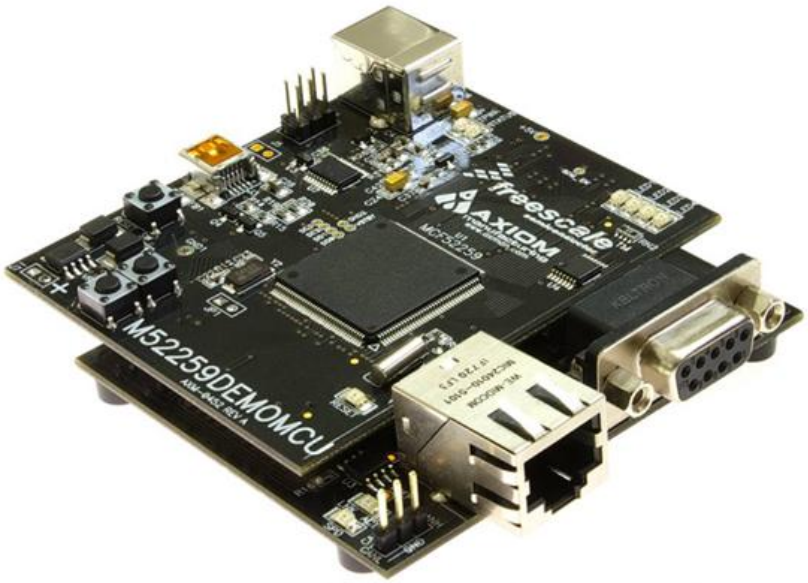


Figura 4.3: Microcontrolador de 32 bits M52259DEMOKIT utilizado en las aplicaciones experimentales en este trabajo de tesis.

4.3. Seguridad

Actualmente, muchos sistemas embebidos se pueden conectar a la red internet para mejorar sus características y desempeño, lo que permite que desde electrodomésticos hasta grandes equipos industriales puedan ser monitoriados y controlados de manera remota a través de conexiones a internet, como ejemplo cambiar los niveles de aire acondicionado de una casa, controlar luces y electrodomésticos al acceder a un sitio web, monitoreo remoto del clima, medición y control de variables físicas, control remoto de motores eléctricos, etc.

Sin embargo, estas ventajas digitales de los sistemas embebidos representan una nueva escala de riesgos, ya que no sólo la integridad de la información se ve comprometida, sino también los diferentes actores se pueden ver afectados, como ejemplo, cambiar la temperatura dramáticamente de un recinto puede ser peligroso para niños o personas mayores, un intruso podría monitorear el estado de encendido y apagado de electrodomésticos y determinar cuando no este nadie en casa para robarla; por lo que los aspectos de seguridad de estos sistemas embebidos quedan en segundo plano. Se tienen diferentes técnicas para enfrentar este problema de seguridad, como el uso de claves, contraseñas, NIP's, etc. como barrera de primer nivel. Sin embargo, estas técnicas presenta algunos problemas, como fáciles de adivinar, difíciles de recordar, transferencia de claves a terceros, etc.

Otra técnica de protección de datos son los protocolos de seguridad de internet como SSL, SSH, Kerberos, entre otros. Sin embargo, muchos sistemas embebidos no tienen suficiente capacidad para el procesamiento requerido por estos protocolos, ni suficiente recursos de memoria para implementarlos. Existen ataques de tipo lógico y físico por

lo que se tienen que diseñar sistemas embebidos que resistan estos tipos de ataques, algunos de ellos se describieron en la sección 3.3.

Es notorio que el cifrado de datos no es suficiente, sin embargo es un paso más en la integridad de los datos en su almacenamiento y transmisión en sistemas embebidos. El estándar del NIST FIPS 140-2, presenta los requerimientos de seguridad que un sistema criptográfico debe tener tanto física como lógicamente en cuatro distintos niveles. Cabe notar que en este trabajo de tesis doctoral, la seguridad del sistema es a nivel lógico (software).

4.4. Implementación de caos en microcontrolador

En los últimos años, el mapa logístico y el sistema de Lorenz se implementaron en microcontroladores, ver por ejemplo [83] y [84], los autores reportan la simple implementación del mapa logístico en microcontrolador; mientras que en [85] se reporta la implementación en microcontrolador de dos mapas logísticos combinados para obtener un *generador de bits caótico* y la pseudoaleatoriedad de las secuencias son verificadas con las pruebas estadísticas FIPS-140-2 del *NIST* (del inglés, *National Institute of Standards and Technology*); en el mismo año 2013, [86] se propuso la implementación experimental del sistema caótico de Lorenz aproximado por Euler para demostrar que este tipo de osciladores (tiempo continuo) pueden ser utilizados en microcontroladores.

Sin embargo, la implementación en microcontroladores de algoritmos criptográficos basados en caos es relativamente escasa. En [87] se realizó una implementación en un microcontrolador Atmel AVR de un algoritmo de cifrado basado en caos, donde se utiliza el mapa generalizado de Hénon y la arquitectura de confusión y difusión para cifrar texto alfanumérico e imágenes, sin embargo, los autores no presentan analisis de seguridad básicos como precisión de clave secreta, espacio de clave secreta, y entre otros, sólo se presenta un histograma, por lo que el desempeño y seguridad de esquema propuesto no es verificado.

Recientemente, en [88] se presentó un algoritmo de cifrado basado en el sistema caótico de Chua, la aleatoriedad de la secuencia caótica se verifica con FIPS-140-2, es decir, con análisis de poker, monobit, runs y long run; además, el algoritmo de cifrado de texto se implementa en un microcontrolador Atmel AVR, aunque el sistema puede encriptar texto ASCII en tiempo real y a bajo costo, los autores omiten el análisis de seguridad del sistema criptográfico para probar la efectividad y la seguridad del esquema que se presenta.

4.5. Conclusiones

En la actualidad, la humanidad utiliza sistemas embebidos aun sin saberlo, están en todas partes para hacer más fácil la vida de las personas. Son sistemas de computo que realizan básicamente una tarea específica como la de controlar el lavado de ropa en

una lavadora moderna, controlar las señales de tránsito como semáforos, hasta tareas más complejas como controlar la temperatura de una habitación a través de internet, apagar y encender electrodomésticos mediante una página web o aplicaciones donde se requiere almacenar o transmitir información confidencial por medios inseguros como medir y controlar variables físicas en la industria y en la telemedicina.

El crecimiento tecnológico ha traído consigo múltiples ventajas para el hombre en el tema de las comunicaciones, como la transmisión de información de forma instantánea y almacenamiento de datos en la nube; sin embargo, esto conlleva a que nuestros datos sean vulnerables y puedan ser robados para fines fraudalentos, maliciosos, bélicos, personales, entre otros, por lo que es de vital importancia que se mantenga la integridad de dichos datos para evitar este tipo de actos ilegales, por lo que se requiere considerar algún tipo de seguridad en el diseño integral de sistemas embebidos que utilicen almacenamiento de datos privados y transmisión de información confidencial.

En particular, para hacer frente a estos problemas de seguridad, frecuentemente se recurre a la criptografía. En este trabajo de tesis doctoral, se aplica la criptografía caótica en sistemas embebidos, lo cual será mostrado en capítulos posteriores.

Capítulo 5

Análisis de implementación de sistemas caóticos en microcontrolador

En este capítulo, se presenta el análisis de la implementación en microcontrolador (programación C) de dos sistemas caóticos: sistema de Lorenz (tiempo continuo) y mapa logístico (tiempo discreto). El principal objetivo de este capítulo, es determinar cuál de los dos sistemas caóticos utilizar en el algoritmo criptográfico propuesto en esta tesis, con base a su eficiencia, seguridad y recursos de implementación. Además, en cada caso se verifica la existencia de caos con exponentes de Lyapunov.

5.1. Introducción

Un aspecto importante en las implementaciones de sistemas caóticos en microcontroladores, es verificar la existencia de caos. Además, analizar su eficiencia y determinar los recursos necesarios para llevar a cabo su implementación es muy importante para mostrar su potencial uso en criptografía. Estos últimos aspectos, son considerados en este capítulo para validar la implementación en microcontrolador del algoritmo criptográfico propuesto basado en caos.

Cabe notar, que en las implementaciones de caos en microcontroladores reportadas en la literatura, por ejemplo en [83–86], no presentan ningún análisis de los que se presentan en las siguientes secciones como verificación de caos y análisis de implementación.

5.2. Sistema 3D de Lorenz

El sistema de Lorenz representado por un conjunto de ecuaciones diferenciales (tiempo continuo) no lineales de tres estados, tiene que ser resuelto por un método numérico de ecuaciones diferenciales ("discretización") para implementarse en tecnología digital. Para ello, se tienen varios métodos, siendo los más populares *Euler* y *Runge-Kutta de*

cuarto orden (RK4). El método de discretización de *Euler*, es fácil de entender y programar. Aunque *Euler* tiene mejor tiempo de respuesta que RK4, su precisión es baja y puede llevar al sistema caótico a un punto fijo, debido a la degradación del caos en la implementación digital [89]. Por otra parte, RK4 es uno de los métodos más utilizado para resolver ecuaciones diferenciales ya que es más preciso que *Euler*, al tener un error de truncación de $O(H^4)$ (utiliza cuatro funciones para estimar el valor futuro, mientras que *Euler* utiliza una) [90]. Una desventaja de RK4 es el sacrificio de tiempo de computo por precisión. En esta tesis doctoral se utiliza el método de discretización RK4, que se basa en un problema de valor inicial de la siguiente forma

$$y'(x) = f(x, y), \quad y(x_0) = y_0 \tag{5.1}$$

donde y' es la primera derivada de la variable y, es decir, corresponde a una ecuación diferencial de primer orden con la condición inicial $y(x_0) = y_0$ (por ejemplo, el sistema tridimensional de Lorenz). Entonces el método RK4 de discretización para este problema está dado por la siguiente expresión

$$y_{i+1} = y_i + (h/6)(k_1 + 2k_2 + 2k_3 + k_4), \tag{5.2}$$

con

$$k_1 = f(x_i, y_i), \tag{5.3a}$$
$$k_2 = f(x_i + h/2, y_i + (h/2)k_1), \tag{5.3b}$$
$$k_3 = f(x_i + h/2, y_i + (h/2)k_2), \tag{5.3c}$$
$$k_4 = f(x_i + h, y_i + hk_4), \tag{5.3d}$$

donde y_{i+1} es el valor estimado a partir de y_i más el producto de un intervalo h por una pendiente estimada. Por tanto, el sistema de Lorenz (2.1), aproximado por RK4 queda de la siguiente manera

$$\begin{cases} x_{i+1} = x_i + (h/6)(k_1^x + 2k_2^x + 2k_3^x + k_4^x), \\ y_{i+1} = y_i + (h/6)(k_1^y + 2k_2^y + 2k_3^y + k_4^y), \\ z_{i+1} = z_i + (h/6)(k_1^z + 2k_2^z + 2k_3^z + k_4^z) \end{cases} \tag{5.4}$$

con

$$k_1^x = \sigma(y(n) - x(n)), \tag{5.5a}$$
$$k_1^y = \rho x(n) - y(n) - x(n)z(n), \tag{5.5b}$$
$$k_1^z = x(n)y(n) - \beta z(n) \tag{5.5c}$$

y

$$k_2^x = \sigma(y(n) - x(n)) + (h/2)k_1^x, \tag{5.6a}$$
$$k_2^y = (\rho x(n) - y(n) - x(n)z(n)) + (h/2)k_1^y, \tag{5.6b}$$
$$k_2^z = (x(n)y(n) - \beta z(n)) + (h/2)k_1^z \tag{5.6c}$$

y

$$k_3^x = \sigma(y(n) - x(n)) + (h/2)k_2^x, \tag{5.7a}$$
$$k_3^y = (\rho x(n) - y(n) - x(n)z(n)) + (h/2)k_2^y, \tag{5.7b}$$
$$k_3^z = (x(n)y(n) - \beta z(n)) + (h/2)k_2^z \tag{5.7c}$$

y

$$k_4^x = \sigma(y(n) - x(n)) + (h)k_3^x, \tag{5.8a}$$
$$k_4^y = (\rho x(n) - y(n) - x(n)z(n)) + (h)k_3^y, \tag{5.8b}$$
$$k_4^z = (x(n)y(n) - \beta z(n)) + (h)k_3^z \tag{5.8c}$$

donde x, y y z son los tres estados del sistema de Lorenz (2.1), h es el tamaño de los pasos y σ, ρ, β son los parámetros de control.

Las ecuaciones (5.4) representan la aproximación discreta del sistema tridimensional de Lorenz (2.1) por el método RK-4, es decir, la representación del sistema de Lorenz en ecuaciones no lineales en diferencias, las cuales, pueden implementarse en sistemas digitales. Estas ecuaciones son implementadas en microcontrolador mediante programación optimizada en lenguaje C, utiliza arquitectura de precisión finita tipo punto flotante para las operaciones aritméticas con una precisión tipo doble (64 bits), lo que permite tener una precisión de 10^{-15} decimales (estándar IEEE 754 para operaciones aritméticas de punto flotante) para las condiciones iniciales x_0, y_0, z_0 y para los parámetros σ, ρ, β; esta precisión ayuda para generar dinámicas caóticas sensibles a cambios sumamente pequeños en sus condiciones iniciales y aportar un espacio de claves mucho mayor para aplicaciones criptográficas.

En la figura 5.1 se muestra la lectura de los primeros 20 datos generados en el microcontrolador, a partir de las condiciones iniciales $x_0 = -8.12345678987654$, $y_0 = -8.56789876543212$ y $z_0 = 24.4321234567899$, con los parámetros de control $\sigma = 10$, $\rho = 28$, $\beta = 2.666666667$ y con el tamaño de pasos $h = 0.007$.

5.2.1. Existencia de caos

Las secuencias caoticas (x, y, z) generadas por el sistema de Lorenz ec. (5.4) implementado en microcontrolador se analizan con base en los exponentes de Lyapunov para verificar la existencia de caos y así, garantizar que las secuencias que se utilizaran como referencia para encriptar información, correspondan a secuencias caóticas, ya que esto es de vital importancia para generar un criptograma (información cifrada) con excelentes características criptográficas.

Para realizar el análisis respectivo con base en la sección 2.2.1, se extraen 10,000 datos discretos del microcontrolador con memoria USB, generados por la aproximación discreta del sistema de Lorenz (5.4) y se consideran dos casos:

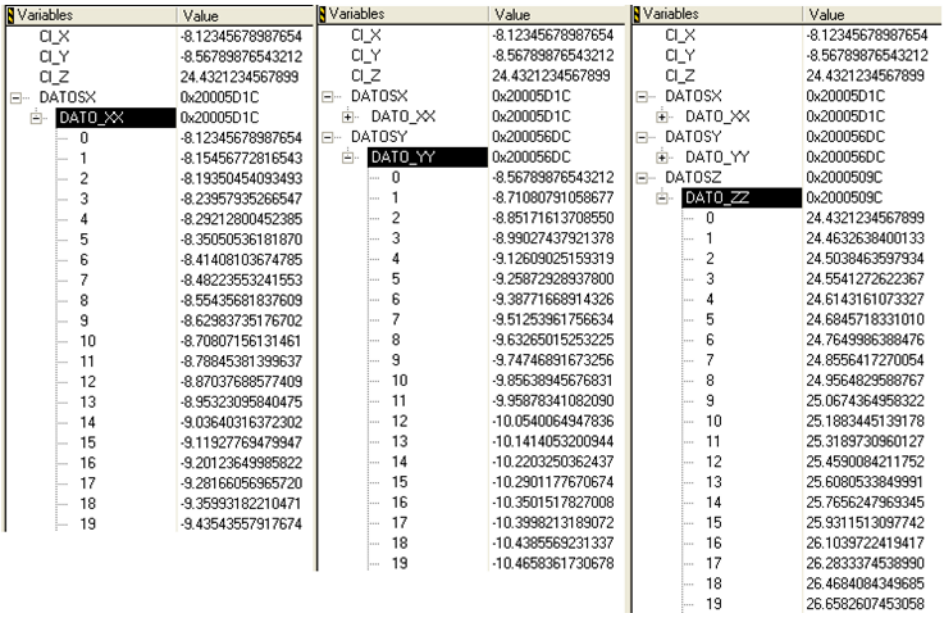


Figura 5.1: Datos generados del sistema de Lorenz discreto (5.4) implementado en microcontrolador con una precisión de 10^{-15}.

1. *Secuencia estable:* con base en las condiciones iniciales y parámetros de control de la tabla 5.1, se generan 10,000 datos y se extraen del microcontrolador (ver figura 5.2(a)); en la figura 5.3 se muestran las gráficas temporales de cada estado y sus gráficas de fase, donde se observa que el sistema es estable con respecto al tiempo. En la tabla 5.2, se muestra el máximo exponente de Lyapunov, el cual resulta ser negativo.

2. *Secuencia caótica:* con base en las condiciones iniciales y parámetros de control de la tabla 5.1, se generan 10,000 datos y se extrae del microcontrolador (figura. 5.2(b)); en la figura 5.4 se muestran las gráficas temporales de cada estado y sus gráficas de fase, donde se observa que el sistema es caótico con respecto al tiempo. En la tabla 5.2, se muestra el máximo exponente de Lyapunov, el cual resulta ser positivo, garantizando la existencia de caos.

La determinación del máximo exponente de Lyapunov muestra que las secuencias generadas (x, y, z) del sistema discretizado de Lorenz implementado en microcontrolador son caóticas, una de las aportaciones importantes de este trabajo doctoral. Además, con base en este trabajo se pueden implementar otros sistemas caóticos o hipercaóticos de tiempo continuo como Chen, Lu, Chua, Rössler, Genesio–Tesi, CNN, entre otros.

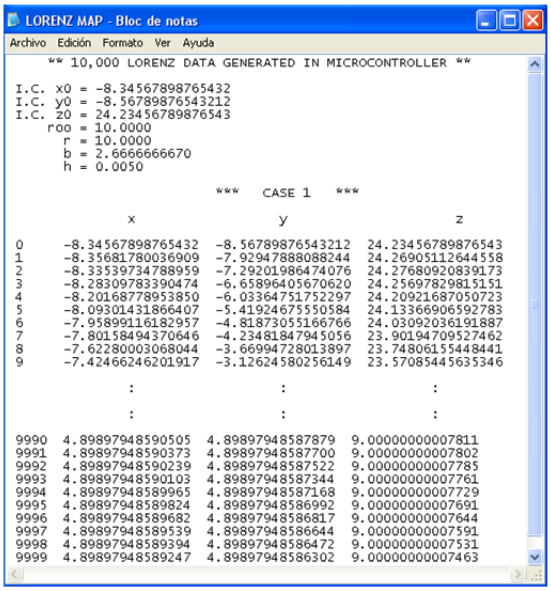
5.2.2. Eficiencia y recursos de implementación

Un buen sistema criptográfico embebido no solo debe ser seguro, sino también debe tener buena combinación de desempeño y costo. En esta sección, se determina la velocidad para generar secuencias caóticas y los recursos de hardware que se requieren para su implementación. En cuanto a la velocidad, se calcula el tiempo que se requiere para

	Estable	Caos
x_0	-8.34567898765432	-8.34567898765432
y_0	-8.56789876543212	-8.56789876543212
z_0	24.23456789876543	24.23456789876543
σ	10	10
ρ	24.23456789876543	2.666666667
β	**10**	**28**
h	0.005	0.005
T	10000	10000
ϵ	10^{-9}	10^{-9}
D_k	$\sqrt{(C_{x_k})^2 + (C_{y_k})^2 + (C_{z_k})^2}$	
C_k	$C_{x_k} = x_k - x_k^*, C_{y_k} = y_k - y_k^*, C_{z_k} = z_k - z_k^*$	

Tabla 5.1: Valores para el análisis del máximo exponente de Lyapunov del sistema de Lorenz (5.4) implementado en microcontrolador.

a)



b)

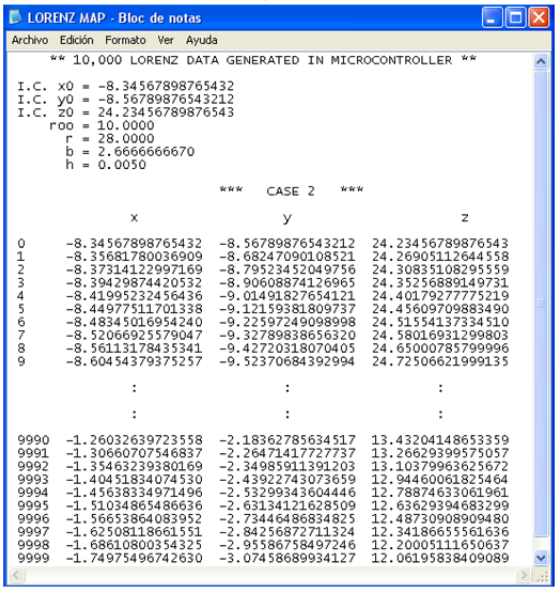


Figura 5.2: Extracción de 10,000 datos de la implementación en microcontrolador del sistema de Lorenz (5.4): a) secuencia estable y b) secuencia caótica.

	Máximo exponente de Lyapunov
Estable	-0.4940
Caos	0.9328

Tabla 5.2: Máximo exponente de Lyapunov para las secuencias de la aproximación discreta de Lorenz (5.4) generadas en microcontrolador.

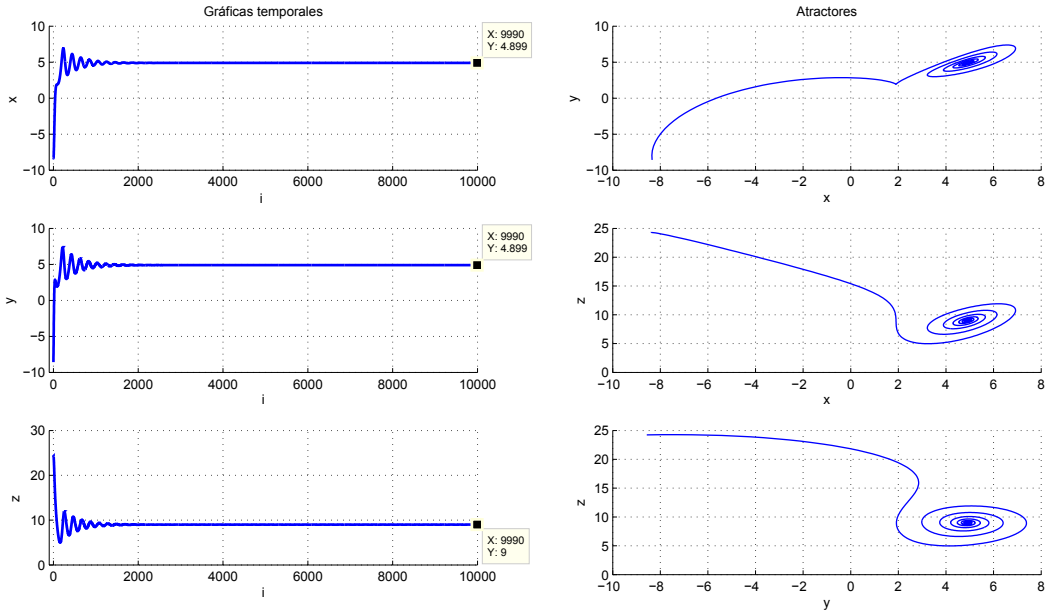


Figura 5.3: Gráficas temporales y atractores de punto fijo, de la aproximación discreta de Lorenz (5.4) generada en microcontrolador.

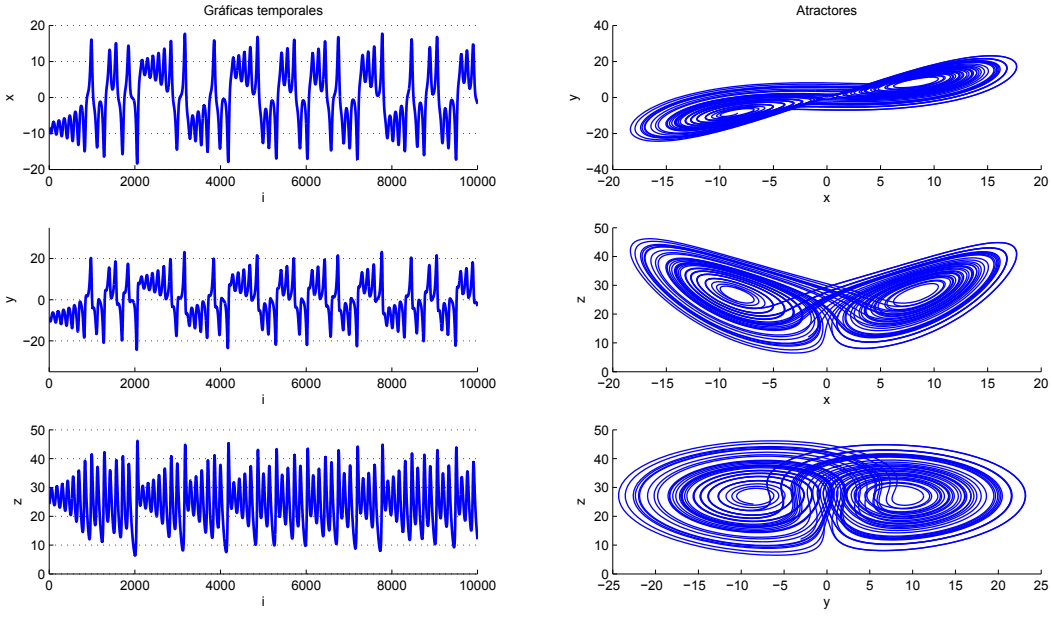


Figura 5.4: Gráficas temporales y atractores caóticos de la aproximación discreta de Lorenz (5.4) generada en microcontrolador.

generar 1,000 datos caóticos (por cada estado) con la frecuencia máxima de 80 MHz y se determinan los ciclos de reloj necesarios (con registro *programm counter* del microcontrolador). En la tabla 5.3 se muestra la eficiencia y recursos de la implementación del sistema de Lorenz (5.4) en microcontrolador. Básicamente, se requiere programación optimizada en lenguaje C para generar 1,000 datos caóticos en 1.2 segundos, 110 KB de memoria y bajo consumo de energía con 118 mA, por lo que el sistema de Lorenz discretizado (5.4) es una buena opción para ser utilizado en sistemas embebidos con aplicaciones criptográficas.

	Lorenz	logístico
Memoria Flash (KBs)	110/512 (21.6 %)	103/512 (20 %)
Periféricos I/O	Comunicación USB 2.0	
Complemento	Freescale MQX 3.5 (MQX + USB host)	
Frecuencia Máxima (MHz)	80/80 (100 %)	
Calculo de 1000 datos (seg)	1.2	0.5
Voltaje (Vdc)	5	
Corriente (mA)	118	
Costo (USD)	115	
Dimensiones (in)	2×2	

Tabla 5.3: Eficiencia y recursos de implementación de Lorenz discreto (5.4) y logístico en microcontrolador.

5.3. Mapa 1D logístico

El mapa logístico es representado por una ecuación en diferencias (ver ec. (2.5)) y posee un estado, por lo que su implementación digital es directa y sencilla, es decir, en este caso no se requiere discretizar el sistema con algún método numérico como Euler o RK4, ya que el mapa logístico es discreto por naturaleza. Tomando en cuenta las mismas características de programación que en el caso del sistema de Lorenz discreto (5.4), se calculan 10,000 iteraciones con la condición inicial $x_0 = 0.567898765432123$ y parámetro de control $a = 3.999999999999999$. En la figura 5.5(a)-(b) se muestran los primeros y últimos 20 datos caóticos calculados en microcontrolador.

5.3.1. Existencia de caos

La secuencia caótica del mapa logístico en microcontrolador es verificada mediante el cálculo de exponentes de Lyapunov, con base a la sección 2.3.1. El proceso consta en utilizar dos mapas logísticos independientes para generar dos trayectorias caóticas con condiciones iniciales muy cercanas y el parámetro de control se mantiene fijo; estos vectores son extraídos del microcontrolador con memoria USB para ser analizados en el entorno de MatLab.

a)

Variables	Value
CONDICION_INICIAL	0.567898765432123
DATOS	0x2000512C
DATO_MAPA	0x2000512C
0	0.567898765432123
1	0.981559030611174
2	0.0724036001473055
3	0.268645275332058
4	0.785899965495284
5	0.673044838919182
6	0.880221934893737
7	0.421725120902652
8	0.975492173209182
9	0.0956287728672378
10	0.345935642668576
11	0.905056695200221
12	0.343716294693900
13	0.902301613823184
14	0.352613646061046
15	0.913109050690325
16	0.317363648950955
17	0.866575853101959
18	0.462488575690283
19	0.994371572185025

b)

Variables	Value
80	0.0274932492828897
81	0.106949482107034
82	0.382045161536286
83	0.944346624331996
84	0.210224309779040
85	0.664120197427864
86	0.892258243184956
87	0.384533882613808
88	0.946670302943032
89	0.201942561875118
90	0.644647054313729
91	0.916308918713444
92	0.306747536798573
93	0.850613941866324
94	0.508279455075833
95	0.999725802494589
96	0.0010964892845566
97	0.0043811479832219
98	0.0174478141022843
99	0.0685735515413457

Figura 5.5: Datos generados por el mapa logístico en microcontrolador: a) primeros 20 de 10,000 y b) últimos 20 de 10,000.

En el análisis se utiliza la condición inicial $x_0 = 0.456789876543212$, una condición inicial muy cercana de $x_0 + \delta_0 = 0.456789876543212$ con $\delta_0 = 0.000005$, el parámetro de control $a = 3.999999999999999$ y con un número de iteraciones de $T = 10,000$. En la figura 5.6 se muestran los primeros y últimos 20 datos extraídos del microcontrolador visualizados en archivo *.tex., mientra que en la figura 5.7 se muestra la trayectoria de ambas secuencias caóticas (para las primeras 100 iteraciones) y se observa que estas divergen con el tiempo, mostrando sensibilidad a condiciones iniciales.

Con la ecuación (2.6), se determina el máximo exponente de Lyapunov para las dos trayectorias de la figura 5.7 y como resultado se tiene que $\lambda = 0.692738 \approx ln2$, de manera que la secuencia generada es caótica, incluso para cualquier condición inicial. Una de las aportaciones de este trabajo doctoral fue verificar que las secuencias generadas por el mapa logístico implementado en microcontrolador, genere secuencias caóticas para que puedan utilizase en el cifrado de información.

5.3.2. Eficiencia y recursos de implementación

Naturalmente, el mapa logístico tiene ventajas en eficiencia y recursos sobre el sistema de Lorenz, por ser un sistema unidimensional y con mínimas operaciones aritméticas. La velocidad de cálculo del mapa logístico se basa en iterar 1,000 veces a la máxima frecuencia del microcontrolador 80 Megahertz y determinar cuantos ciclos de reloj se requieren (mediante registro *programm counter*). En la tabla 5.3 se presentan los recursos necesarios para la implementación del mapa logístico y la velocidad de cálculo;

SECUANCIAS_CAOTICAS - Bloc de notas

Archivo Edición Formato Ver Ayuda

** SECUENCIAS CAOTICAS DEL MAPA LOGISTICO EXTRAIDAS DEL MICROCONTROLADOR **
PARA DETERMINAR EL MAXIMO EXPONENTE DE LYAPUNOV CON 10,000 DATOS

CONDICION INICIAL x0 = 0.456789876543212
CONDICION INICIAL x0 + d0 = 0.456794876543212
PARAMETRO DE CONTROL A = 3.999999999999999
DELTA d0 = 0.000005000000000

n	f^n(x_n)	f^n(x_n + d0)
0	0.456789876543212	0.456794876543212
1	0.992531540923396	0.992533269228335
2	0.029650724782499	0.029643914812995
3	0.115086237209486	0.115060612510219
4	0.407365580857791	0.407286671835970
5	0.965675457560742	0.965616955122995
6	0.132585472902374	0.132803404407963
7	0.460026261110512	0.460666640742473
8	0.993608400796780	0.993811547398073
9	0.025402986651382	0.024600622625284
:	:	
9990	0.212069993657785	0.541172919440106
9991	0.668385245791089	0.993219162819114
9992	0.886585635999499	0.026939429712050
9993	0.402206184155452	0.104854787355358
9994	0.961745478330251	0.375441043696082
9995	0.147164452966272	0.937940265617914
9996	0.502028306997640	0.232833295002042
9997	0.999983543882893	0.714487806962136
9998	0.000065823385212	0.815979922658293
9999	0.000263276209977	0.600626753907436

Figura 5.6: Datos del mapa logístico extraídos del microcontrolador para determinar el máximo exponente de Lyapunov.

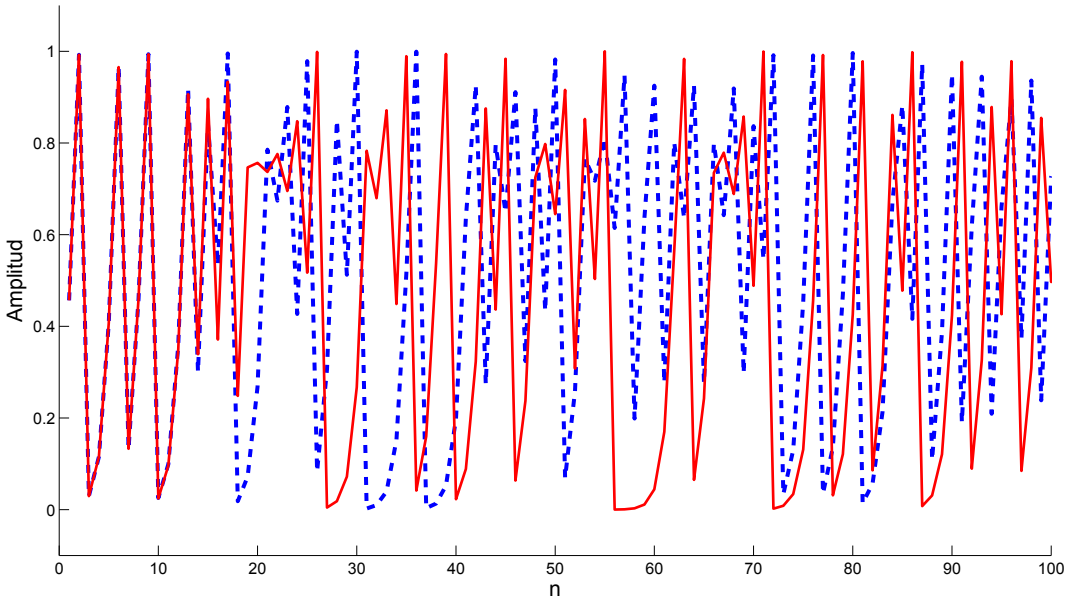


Figura 5.7: Trayectorias de dos secuencias caóticas del mapa logístico generadas en microcontrolador para determinar el máximo exponente de Lyapunov.

se muestran los resultados esperados en comparación con el sistema discreto de Lorenz (5.4), ya que se requiere de 7 kilobytes menos de memoria y se tiene menor tiempo de cálculo con 0.5 segundos para la generación de 1,000 datos.

5.4. Conclusiones

Los sistemas caóticos de tiempo continuo requieren discretizarse por algún método numérico para que puedan implementarse en tecnología digital, en este caso en microcontroladores. Este tipo de sistemas requieren de mayor tiempo de cálculo y de mayor espacio en memoria en sus implementaciones, ya que son caracterizados en forma discreta por un conjunto de ecuaciones en diferencias con múltiples operaciones aritméticas, ya que generalmente son sistemas de tres estados.

Por otra parte, los mapas caóticos pueden ser implementados directamente en tecnología digital, ya que son de tiempo discreto naturalmente y de dimensión menor. En este capítulo, el sistema discretizado de Lorenz y el mapa logístico fueron implementados en microcontrolador, se verificó la existencia de caos mediante el cálculo del máximo exponente de Lyapunov y también, se analizó la eficiencia y los recursos de implementación. La velocidad de procesamiento de datos caóticos está directamente relacionada con el tiempo de cifrado y los recursos de implementación lo están con los costos de producción, es por ello que se utiliza el mapa logístico en el algoritmo criptográfico propuesto en esta tesis doctoral por ofrecer mejor combinación de velocidad *vs* recursos de hardware.

Capítulo 6

Algoritmo de cifrado caótico propuesto

En este capítulo, se presentan los detalles del algoritmo de cifrado propuesto en esta tesis doctoral. Se basa en la arquitectura ampliamente utilizada en la criptografía, conocida como confusión y difusión. Se propone una clave secreta simétrica de 128 bits representada por 32 caracteres hexadecimales para generar secuencias caóticas de dos mapas logísticos y cifrar los datos de manera secuencial (flujo). Finalmente, se describen algunas características de seguridad que presenta el algoritmo de cifrado.

6.1. Introducción

Los riesgos potenciales que presenta el almacenamiento y transmisión de información confidencial en medios inseguros es cada día mayor, por lo que se requiere del cifrado de información. Además de la seguridad que requiere el cifrado para hacerlo confiable, también debe poseer velocidad para cifrar datos y requerir poco espacio de memoria para que pueda utilizarse en sistemas digitales en tiempo real, principalmente en sistemas embebidos, en los cuales, se tienen fuentes limitadas de procesamiento y almacenamiento.

El algoritmo de cifrado propuesto en este trabajo doctoral, se basa en las siguientes características criptográficas [77]:

- *Cifrado simétrico*. El algoritmo utiliza la misma clave secreta para cifrar y descifrar.
- *Arquitectura de confusión y difusión*. El algoritmo utiliza procesos para cambiar de posición y cambiar de valor a cada elemento claro en una sola operación.
- *Cifrado a flujo*. El algoritmo cifra cada elemento del texto claro, uno a la vez hasta terminarlo. En este caso, se utilizan elementos que son representados por 8 bits (1 byte).

- *Cifrado no convencional*. El algoritmo utiliza **secuencias caóticas del mapa logístico** (Sec. 2.3), que son determinadas por la clave secreta para generar secuencias pseudoaleatorias para realizar el proceso de confusión y difusión.

El uso de mapas caóticos unidimensionales como el mapa logístico, tiene cierta desventaja si se utilizan para fines criptográficos. Algunas de las *desventajas* son [91]:

- Rangos caóticos discontinuos.
- Distribución de datos no uniforme.
- Espacio de claves pequeño.
- Periodicidad en rangos caóticos.

Sin embargo, los sistemas caóticos unidimensionales tienen poderosas *ventajas* como [92]:

- Simple estructura.
- Fácil de implementar en sistemas digitales.
- Poco consumo de memoria y recursos físicos.
- Generación de datos a alta velocidad.

Por lo que, en el cifrado propuesto en esta tesis doctoral, se utiliza el mapa logístico unidimensional (2.5) con algunas consideraciones para evitar las desventajas mencionadas.

En la figura 6.1 se muestra el **diagrama de bloques del proceso de cifrado**, el cual, procede de la siguiente manera: primero, se itera el mapa logístico 2 con base en la clave secreta, después se calcula el valor de Z que tiene relación con el texto claro y secuencias caóticas del mapa logístico 2, se continua con la iteración del mapa logístico 1 con base en la clave secreta y el valor de Z para realizar los procesos de confusión y difusión sobre el texto claro, y finalmente se agrega el valor de Z al criptograma para que el usuario autorizado pueda descifrar correctamente la información.

El proceso de descifrado consiste en invertir el proceso de cifrado. En la figura 6.2 se muestra el **diagrama de bloques del proceso de descifrado**, el cual, procede como sigue: primero, el valor de Z se extrae de un elemento del criptograma (Z no se calcula como en el caso de cifrado ni se iteran datos caóticos del mapa logístico 2), después 5000 datos caóticos son calculados del mapa logístico 1 con el uso de la clave secreta y el valor de Z, posteriormente, se realizan los procesos de confusión y difusión inversos para recuperar el texto claro P.

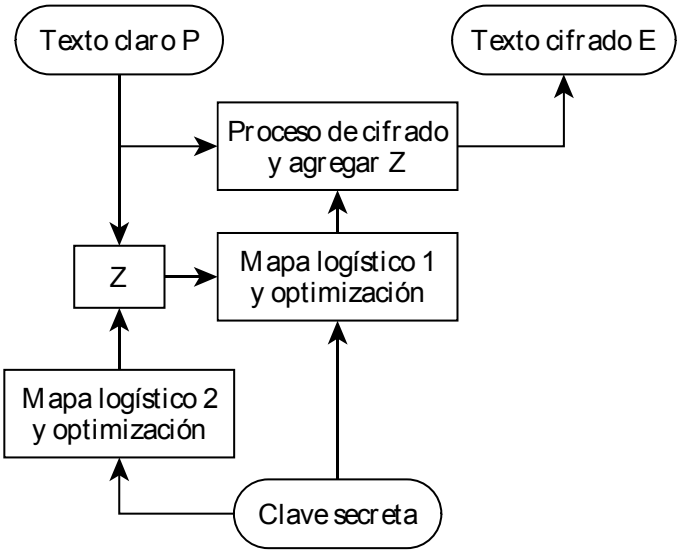


Figura 6.1: Diagrama a bloques del proceso de cifrado caótico del algoritmo criptográfico propuesto.

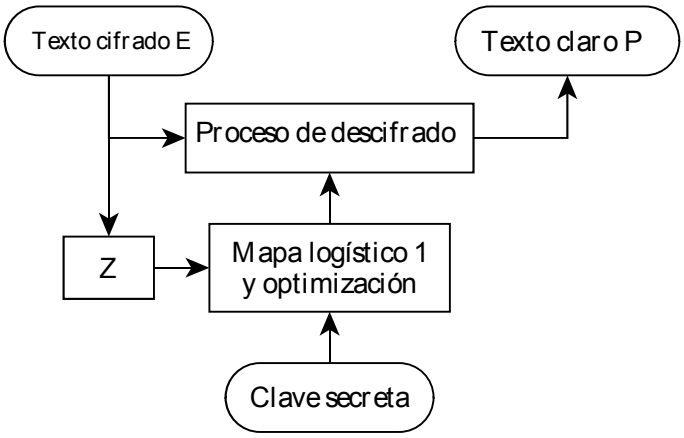


Figura 6.2: Diagrama a bloques del proceso de descifrado caótico del algoritmo criptográfico propuesto.

6.2. Definición de la clave secreta

La clave secreta esta definida como una secuencia de 128 bits, caracterizada por 32 caracteres hexadecimales $K \in (0-9, A-F)$; la condición inicial y el valor del parámetro de control de dos mapas logísticos se determinan de manera indirecta con la clave secreta de 128 bits, por lo que se elimina el problema de poco espacio de claves que presentan los sistemas unidimensionales, al utilizar la técnica propuesta (ver tabla 6.1).

Además, las ventanas periódicas que se presentan en rangos caóticos del mapa logístico son eliminadas, al considerar un rango del parámetro de control entre $(3.999, 4)$, lo que evita claves débiles. Se considera una precisión decimal de 10^{-15} para evitar la degradación caótica y periodicidad en su implementación digital. Todas las combinaciones de la clave secreta genera secuencias caóticas, las cuales, son verificadas determinando el valor del máximo exponente de Lyapunov (ver figura 6.3), donde se utiliza una condición inicial arbitraria x_0 fija, con una $\delta_0 = 1 \times 10^{-13}$ y 1,000 iteraciones en cada prueba y el que se varía es el parámetro de control entre $(a \in 3.999 - 4)$ con incrementos de 5×10^{-8} para generar 20,000 pruebas.

Clave secreta	Parámetro de control			Condición inicial
32 dígitos Hex	$H_1, H_2, \ldots, H_{32}$ donde $H \in [0-9, A-F]$			
Cálculos	$A = \frac{(H_1, H_2, \ldots, H_8)_{10}}{2^{32}+1}$	$B = \frac{(H_9, H_{10}, \ldots, H_{16})_{10}}{2^{32}+1}$	$C = \frac{(H_{17}, H_{18}, \ldots, H_{24})_{10}}{2^{32}+1}$	$D = \frac{(H_{25}, H_{26}, \ldots, H_{32})_{10}}{2^{32}+1}$
Logístico 1	$a_1 = 3.999 + [((A + B + Z)\ mod\ 1) * 0.001]$			$x_{1_0} = (C + D + Z)\ mod\ 1$
Logístico 2	$a_2 = 3.999 + [((A + B)\ mod\ 1) * 0.001]$			$x_{2_0} = (C + D)\ mod\ 1$
Rango	$3.999 < a_{1,2} < 4$			$0 < x_{1_0, 2_0} < 1$
Precisión	10^{-15}			
	donde $(a\ mod\ b) = (a - b) \times (a/b)$ con $b \neq 0$			

Tabla 6.1: Clave secreta propuesta.

6.3. Optimización de secuencia caótica

Se conoce que el mapa logístico genera muchos valores caóticos muy cercanos a 0 y a 1 como se aprecia en la figura 6.4(a), lo que genera una distribución *no uniforme* [91]. Esta mala distribución, afecta la seguridad del cifrado en los procesos de confusión y difusión, debido a que muchos elementos del texto cifrado no se verán modificados en gran proporción en comparación con el texto claro, lo cual, generaría un criptograma E con propiedades estadísticas inadecuadas y sería susceptible ante ataques de criptoanálisis. Por tanto, no se es recomendable utilizar las secuencias de valores caóticos del mapa logístico directamente.

Para superar este problema, en el algoritmo criptográfico propuesto en esta tesis **se propone un proceso de optimización de secuencias caóticas** mediante una simple operación, la cual, consiste en eliminar los primeros 3 dígitos después del punto de los valores caóticos generados por le mapa logístico; este proceso transforma todos aquellos valores muy cercanos a 0 (ejemplo 0.001321... a 0.321...) y muy cercanos a

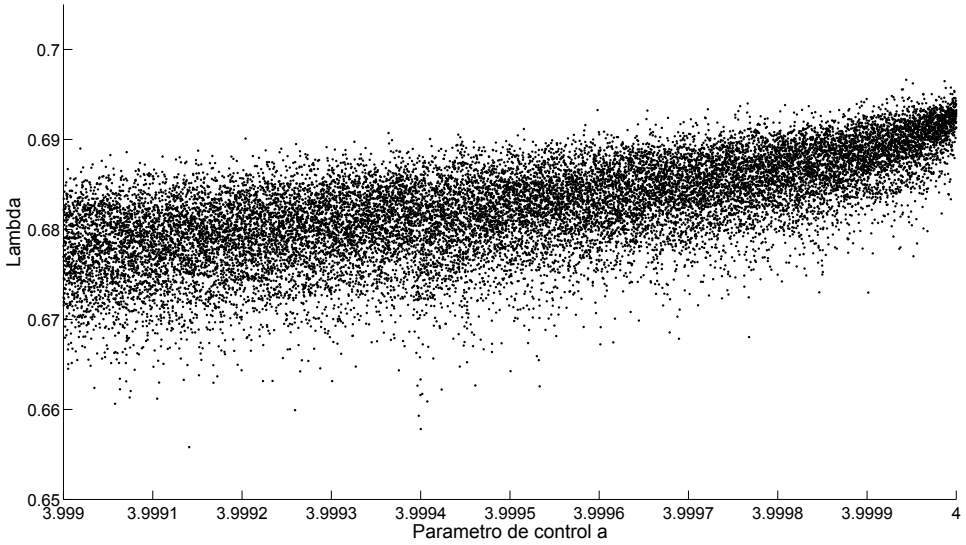


Figura 6.3: Máximo exponente de Lyapunov del mapa logístico para todo el espacio de claves.

1 (ejemplo 0.999231... a 0.231...), a valores totalmente diferentes. Con este proceso se genera una mejor distribución de datos en la secuencia caótica, como se observa en la figura 6.4(b).

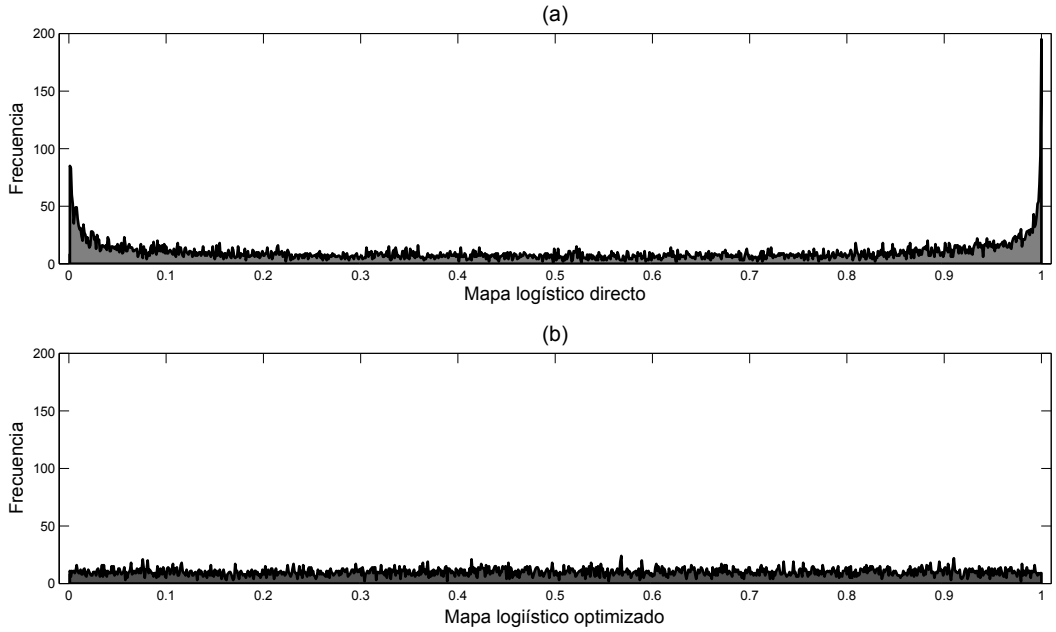


Figura 6.4: Distribución de 10,000 valores del mapa logístico caótico con una separación de 0.001: (a) Datos del mapa logístico directo y (b) datos del mapa logístico "optimizado" (utilizados en el proceso de cifrado).

6.4. Cálculo de Z

El valor de Z es importante para incrementar la sensibilidad a pequeños cambios del texto claro P y a la clave secreta a nivel de bit; al utilizar el valor de Z, el proceso de cifrado es robusto ante ataques diferenciales como ataque de texto claro conocido y ataque de texto claro elegido descritos en la sección 4.3. Para determinar el valor de Z, todos los elementos de texto claro P se suman con la secuencia de datos caóticos del mapa logístico 2. Primero, el mapa logístico 2 es iterado I_2 veces con el parámetro de control a_2 y condición inicial x_{2_0} tomados de la tabla 6.1 para generar una secuencia caótica de datos $x^{L2} = x_1^{L2}, x_2^{L2}, x_3^{L2}, \ldots, x_{I_2}^{L2}$ con $x^{L2} \in (0,1)$ y una precisión decimal de 10^{-15}. Posteriormente, la secuencia caótica x^{L2} es "optimizada" con la siguiente expresión

$$x_i^{L2} = \left(x_i^{L2} * 1000\right) \quad (\text{mód } 1), \quad para\ i = 1, 2, 3, \ldots, I_2 \tag{6.1}$$

donde I_2 es el número de iteraciones del mapa logístico 2, que depende de cada aplicación (entre 1000 y 2000) y *mód* es la operación de módulo. Después, todos los elementos del texto claro se suman con x^{L2} como sigue

$$S = \left\{S + [P_i * x_{I_2+1-i}^{L2}] + x_{I_2+1-i}^{L2}\right\} \quad (\text{mód } 1), \quad para\ i = 1, 2, 3, \ldots, I_2 \tag{6.2}$$

donde P_i representa el elemento i de texto claro, S es una variable inicializada en cero, y x^{L2} corresponde a la secuencia caótica. **El valor de Z se genera del resultado de** S según sea la clase de texto claro como imagen, texto alfanumérico, plantilla dactilar, etc. (por ejemplo ver Sec. 7.2, Sec. 8.2 y Sec. 9.3).

6.5. Cifrado

El mapa logístico 1 se itera $I_1 = 5,000$ veces con valores a_1 y x_{1_0} tomados de la tabla 6.1, para generar la segunda secuencia caótica de datos $x^{L1} = x_1^{L1}, x_2^{L1}, x_3^{L1}, \ldots, x_{I_1}^{L1}$ con $x^{L1} \in (0,1)$ y una precisión decimal de 10^{-15}. Posteriormente, la secuencia x^{L1} es "optimizada" con la siguiente expresión

$$x_i^{L1} = \left(x_i^{L1} * 1000\right) \quad (\text{mód } 1), \quad para\ i = 1, 2, 3, \ldots, I_1 \tag{6.3}$$

donde I_1 es el número de iteraciones para el mapa logístico 1. Cabe mencionar, que sólo los últimos datos requeridos para el cifrado (depende la aplicación) son optimizados para reducir tiempo de cálculo innecesario.

De la secuencia caótica x^{L1} se determinan subsecuencias para los procesos de confusión y difusión. Por ejemplo, para el caso de cifrado de imagen a color RGB se requieren dos subsecuencias para el proceso de confusión (una para las filas y otra para las columnas de la imagen) y una secuencia de difusión.

Una subsecuencia para el proceso de confusión se determina de la siguiente manera

$$CF_i = \ round\left[x_{I_1-\ell+i}^{L1} * (\ell - 1)\right] + 1, \quad para\ i = 1, 2, 3, \ldots, \ell \tag{6.4}$$

donde ℓ es la longitud requerida y $CF \in [1, \ell]$ es el vector pseudoaleatorio para realizar el proceso de confusión (Nota: se toman los últimos datos de la secuencia caótica x^{L1} para incrementar sensibilidad a la clave secreta y texto claro). En un proceso de confusión eficiente, todos los elementos del texto claro se deben permutar entre si mismos; sin embargo, la ec. (6.4) genera valores para reposicionamiento repetido. Por tanto, los valores repetidos de CF son cambiados mediante programación como sigue

$$G_h = [K_h], \quad con\ h \ll \ell \tag{6.5}$$

donde K_h es el valor que no esta en CF de menor a mayor. El vector de valores repetidos G se divide en dos secciones y cada valor se asigna a CF de manera alternada donde un valor repetido aparece. Cuando este proceso termina, se tiene un vector para confusión con todas las posibles posiciones (confusión optimizada).

Una subsecuencia para difusión se determina de x^{L1} de la misma longitud ℓ. Como se mencionó, una desventaja del mapa logístico es que genera muchos valores cercanos a 0 y a 1, lo que resultaría en un proceso de difusión ineficiente. Una solución a este problema es eliminar los primeros tres decimales de los datos caóticos, proceso que se realiza sólo para una longitud determinada por ℓ y generar un proceso de difusión optimizado. La subsecuencia para difusión se calcula como

$$DF_i = \left(x^{L1}_{I_1-\ell+i} + Z\right) \quad (\text{mód } 1), \quad para\ i = 1, 2, 3, \ldots, \ell \tag{6.6}$$

donde $DF_i \in (0, 1)$ es el vector pseudoaleatorio para el proceso de difusión con longitud ℓ (Nota: se toman los últimos datos de la secuencia x^{L1} por motivos de seguridad). Para cada aplicación en específico, el vector para difusión se determina a partir de DF según se requiera.

El proceso de cifrado se calcula con la siguiente expresión

$$E_i = P(CF_i) + DF_i, \quad para\ i = 1, 2, 3, \ldots, \ell \tag{6.7}$$

donde P es el texto claro y E_i el el texto cifrado (criptograma).

El valor de Z debe ser incluido en el criptograma para que el usuario autorizado pueda descifrar correctamente, ya que no se puede calcular directamente de E. Depende la aplicación, el valor de Z es simplemente incorporado al final del criptograma en el caso de texto o escondido en un pixel de una imagen.

6.6. Descifrado

El proceso de descifrado consiste en invertir todos y cada uno de los pasos desarrollados en el cifrado. Se debe utilizar exactamente la misma clave de 128 bits, ya que si un bit cambia, no se podrá recuperar el mensaje claro.

Primero, el valor de Z debe ser recuperado. Después, el mapa logístico 1 es iterado 5,000 veces con la clave secreta y el valor de Z. Posteriormente, se calculan las subsecuencias CF y DF para confusión y difusión, respectivamente. Finalmente, el descifrado se realiza con la siguiente expresión

$$D_i(CF_i) = E_i - DF_i, \quad para\ i = 1, 2, 3, \ldots, \ell \tag{6.8}$$

donde E_i el texto cifrado (criptograma) y D_i es el mensaje recuperado.

6.7. Características de seguridad y eficiencia

El algoritmo de cifrado caótico de información propuesto en este trabajo de investigación doctoral, posee ciertas características de seguridad que aportan robustez ante los ataques criptoanalíticos más poderosos reportados en la literatura (ataque de texto claro elegido/conocido), que han vulnerado un sin número de esquemas criptográficos basados en caos (principalmente para imágenes), ver por ejemplo [52–62]. Estas características se discuten a continuación:

1. **Inicialización de secuencias caóticas**. La condición inicial y parámetro de control de dos mapas logísticos, se determinan de manera indirecta a partir de una clave de 128 bits. Con este proceso, se elimina la desventaja de poco espacio de claves de mapas caóticos unidimensionales.

2. **Optimización de secuencias caóticas**. Los datos caóticos del mapa logístico son modificados mediante una simple operación para generar una mejor distribución, lo que genera mejores procesos de confusión y difusión y por tanto, un criptograma con mejores propiedades estadísticas.

3. **Cálculo del** Z. Al considerar las características del texto claro para realizar el proceso de cifrado, se incremente por mucho la sensibilidad a pequeños cambios (nivel de bit) en el texto claro y a la clave secreta. Además, se utilizan datos caóticos de un segundo mapa logístico para evitar ataques de sólo texto claro elegido (seleccionar un texto claro en ceros y cancelar el valor de Z en el cifrado). Por tanto, este proceso ayuda a dar robustez al algoritmo criptográfico ante los ataques más poderosos que han quebrantado múltiples algoritmos criptográficos basados en caos.

4. **Confusión y difusión optimizados**. Los vectores para cifrado se calculan de tal forma que la confusión es 100 % aplicada en todo el texto claro. Mientras, el proceso de confusión se aplica sobre el texto claro con datos caóticos que presentan una distribución uniforme, lo que genera un cifrado con excelentes características estadísticas.

5. **Selección de datos caóticos**. Los vectores de confusión y difusión se determinan con los últimos valores caóticos del mapa logístico de 5,000 iteraciones. El

motivo es porque dos trayectorias caóticas que inician muy cercanas divergen con el tiempo (se vuelven completamente diferentes sin correlación entre ellas) y esto se traduce a una super sensibilidad a la entrada de los mapas logísticos, lo que genera trayectorias completamente distintas al utilizar claves secretas con un bit de diferencia.

6. **Eficiencia de cifrado**. Uso del mapa logístico unidimensional. Este sistema caótico posee ventajas de implementación tanto en velocidad de generación de datos como poco espacio de memoria requerido. Además, los procesos de confusión y difusión son aplicados al texto claro en un mismo proceso, es decir, se evita realizar operaciones de cifrado independientes. De esta manera, se aumenta la eficiencia de procesamiento con reducciones en tiempo de cifrado.

7. **Robustez ante ataques**. Los distintos análisis de seguridad presentados en cada implementación de este trabajo (capítulos 7, 8 y 9) muestran y validan la seguridad estadística del cifrado caótico propuesto.

6.8. Conclusiones

Un buen algoritmo criptográfico basado en caos, debe cumplir con varios aspectos de seguridad para que pueda ser implementado en aplicaciones prácticas, tanto en el método de cifrado como en la selección del generador de secuencias caóticas. En este capítulo, se describió el algoritmo criptográfico basado en caos propuesto en esta tesis doctoral, donde se mencionan las características y aspectos de seguridad que posee para mostrar la eficacia en la protección de información.

Capítulo 7

Cifrado caótico de imagen a color RGB en MatLab

En este capítulo, se presenta una aplicación del algoritmo de cifrado caótico propuesto, para proteger imágenes a color RGB (también aplica para imágenes a escala de grises). El algoritmo es adaptado para generar dos vectores pseudoaleatorios para el proceso de confusión (confusión en filas y columnas) y un vector para el proceso de difusión. El algoritmo se implementa a nivel de software y programado en MatLab, donde se utilizan diferentes imágenes de distintos tamaños para realizar un análisis de seguridad a nivel estadístico. Los resultados muestras que el cifrado propuesto es robusto ante los ataques más comunes reportados en la literatura y tiene un tiempo de cifrado rápido, por lo que puede implementase en aplicaciones en tiempo real.

7.1. Introducción

En las últimas décadas, los sistemas de comunicaciones han cambiado, debido al crecimiento tecnológico y nuevas redes de comunicación. Actualmente, miles de KB de información confidencial se transmiten por canales de comunicación inseguros como internet. Sin embargo, esta información confidencial puede ser interceptada por personas no autorizadas. La imagen digital se utiliza a gran escala en internet; si la imagen tiene información clasificada, necesita ser cifrada antes de ser transmitida o almacenada. Un esquema de comunicaciones seguras consiste en que el emisor cifra la imagen clara para generar una imagen cifrada mediante un algoritmo y sólo el receptor autorizado podrá descifrar con la clave secreta correcta.

Los principales métodos para cifrado de imagen son confusión (confusión de pixeles), difusión (cambiar el valor del pixel) o ambos confusión y difusión. Algunas aplicaciones donde se requiere cifrado seguro de imágenes son en telemedicina, en la milicia, en video conferencias, en sistemas biométricos, imagen personal, entre otros.

Por otra parte, muchas técnicas de cifrado de imágenes se reportan en la literatura tanto para imágenes a escala de grises como a color. Algunas técnicas son metodologías

de compresión, criptografía ADN, dominio transformado y conceptos matemáticos pero en su mayoría tienen problemas de seguridad, ver por ejemplo [93]. Los algoritmos criptográficos convencionales como *3DES*, *AES*, *IDEA*, etc., son excelentes algoritmos para cifrado de texto, pero no son útiles para cifrado de imágenes porque poseen alta correlación y capacidad de datos, lo que hace un cifrado lento e inseguro, ver por ejemplo [94–97].

Cifrado de imagen basado en caos ha probado ser superior debido a propiedades del caos como ergodicidad, alta sensibilidad a condiciones iniciales y parámetros de control, pseudoaleatoriedad, mezcla de datos, etc., todas útiles para diseñar algoritmos de cifrado seguros y rápidos [43, 48, 98, 99]. Desde 1998, la arquitectura de confusión y difusión ha sido implementada de acuerdo al esquema de Fridrich [67] y hasta estos días, muchos algoritmos de cifrado de imagen (gris y color) han sido reportados en la literatura [42, 43, 46, 48, 67, 95, 98–105]; sin embargo, tods tienen problemas de seguridad, ver por ejemplo [52–60, 62, 93].

En 2012, en [46] se propuso un algoritmo de cifrado de imagen a color basado en el mapa logístico caótico, donde la innovación es cifrar los componentes RGB en una operación, por lo que la correlación entre los componentes se reduce; sin embargo, presenta las siguientes desventajas: el espacio de claves esta definido sobre órbitas periódicas del mapa caótico por lo que el espacio de claves se ve reducido y genera un cifrado débil, los datos del mapa logístico son utilizados directamente por lo que las propiedades estadísticas del cifrado se ven afectadas, los autores no presentan análisis de seguridad importantes como tiempo de cifrado y entropía. Además, en [62] se criptoanalizó el algoritmo de Wang y encontró que los procesos de confusión y difusión pueden ser quebrantados en secciones separadas con la estrategia de divide y vencerás, al utilizar dos imágenes en un ataque de texto claro elegido.

En 2010, en [106] se propuso un algoritmo de cifrado de imagen a color, basado en el mapa estándar y mapa logístico con tiempos de cifrado rápido y excelentes resultados de correlación basado en su trabajo previo en [42]; pero ambos algoritmos fueron quebrantados con ataques de sólo texto claro conocido/elegido en [58] y [60], respectivamente. Además, el algoritmo de Patidar no presenta sensibilidad a imagen clara y las órbitas caóticas son muy débiles.

En [101], los autores presentaron un esquema de cifrado de imagen a color donde se utiliza una nueva composición de dos sistemas caóticos unidimensionales con buenas características estadísticas; algunas ventajas es que el espacio de claves se puede expandir y el tiempo de cifrado es rápido, pero el algoritmo fue quebrantado con un ataque de imagen clara elegida, el espacio de claves genera secuencias no caóticas, no tiene sensibilidad a la imagen clara y la secuencia caótica tiene malas propiedades estadísticas según [57].

En este contexto, otros algoritmos de cifrado de imagen a escala de grises se reportaron en [67, 95, 100, 102–105] fueron criptoanalizados y quebrantados en [52–56, 59, 93],

respectivamente. Básicamente, todos fueron quebrantado con ataques de sólo texto claro conocido/elegido. El principal problema de estos algoritmos de cifrado, es que utilizan la misma secuencia caótica cuando diferentes imágenes claras con cifradas; por tanto, la secuencia pseudoaleatoria requiere ser calculada de la clave secreta y de características de imagen clara para reducir el riesgo de ataque de imagen clara conocida/elegida.

Avances recientes se reportan en la literatura para superar este tipo de ataques, donde la secuencia pseudoaleatoria para el cifrado de imagen es determinada de la clave secreta, de la imagen clara o de ambos, mediante distintas técnicas [107–112].

En [112], los autores propusieron un algoritmo de cifrado novedoso, con base en la inserción de pixeles aleatorio a la primera columna de la imagen original antes del proceso de cifrado, de manera que los detalles de la imagen clara cambian, por lo que esta técnica puede tener problemas en aplicaciones como imagen de satélite, telemedicina, militar, biométrica, etc., donde un pixel tiene información importante; además, la velocidad de cifrado es lento para imágenes a color y los autores no presentan análisis de sensibilidad a la imagen clara, la cual, es muy importante para probar el potencial del algoritmo ante ataques de texto claro elegido y conocido.

Recientemente, otra técnica denominada difusión dependiente es utilizada en algoritmos de cifrado de imágenes a escala de grises [107–111], donde los valores del pixel cifrado influencia en la siguiente operación de confusión y difusión o la secuencia pseudoaleatoria para el proceso de cifrado se determina con la clave secreta y la imagen clara para evitar un ataque de texto claro conocido/elegido; sin embargo, algunos algoritmos presentan un espacio de claves reducido, alto tiempo de cifrado y cifran únicamente imágenes cuadradas.

El algoritmo de cifrado propuesto en este trabajo doctoral, esta basado en las características totales de la imagen clara y secuencias pseudoaleatorias optimizadas generadas por el mapa logístico. El esquema es robusto ante ataques de imagen clara conocida y elegida con una ronda de confusión y difusión. Una clave secreta de 128 bits se utiliza para calcular la condición inicial y parámetro de control del mapa logístico de una manera indirecta para evitar el bajo espacio de claves que presenta un mapa unidimensional. Además, se presenta un análisis de seguridad para verificar la robustez del proceso de cifrado contra los ataques más conocidos reportados en la literatura, ver por ejemplo [52–62].

7.1.1. Características de imagen a color RGB

Una imagen difiere de texto en capacidad de información (tiene muchos más KB de datos de texto) y está representada por pixeles que están altamente correlacionados entre ellos (los colores entre pixeles adyacentes son similares), por lo que algoritmos criptográficos convencionales como *3DES*, *IDEA*, *RSA*, etc. no pueden utilizarse en este caso.

Una imagen a escala de grises esta compuesta por una matriz de M columnas por N filas, donde cada elemento (pixel) esta definido por 8 bits para representar una escala de grises de $0-255$, donde 0 es el nivel más claro. La representación matemática se muestra en la figura 7.1, donde cada elemento de la matriz representa el valor de un pixel.

Por otra parte, una imagen a color RGB por sus tres componentes rojo, verde y azul (del inglés, **R**ed, **G**reen, **B**lue), es similarmente definida como una imagen a grises, pero en este caso la imagen RGB esta representada por 3 matrices, una por cada componente que es representado en una escala de $0-255$, donde 0 es el nivel más claro de rojo, verde y azul. El color de un pixel en una imagen a color es la combinación de los tres colores RGB, por tanto, existen $256 \times 256 \times 256$ posibles combinaciones. La representación matemática es $I_R(M,N,1)$, $I_G(M,N,2)$ y $I_B(M,N,3)$ por lo que el componente rojo esta representado por la matriz 1, el componente verde por la matriz 2 y el componente azul por el componente 3 como se observa en la figura 7.2 (el color en las matrices RGB se muestra gris pero es la intensidad del rojo, verde y azul).

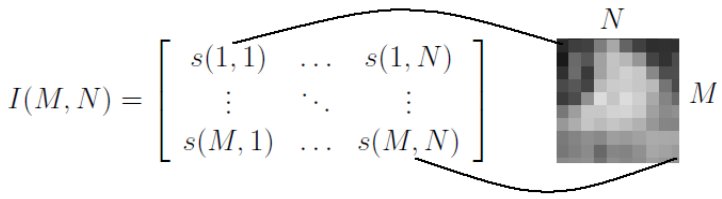


Figura 7.1: Representación matricial de una imagen a escala de grises.

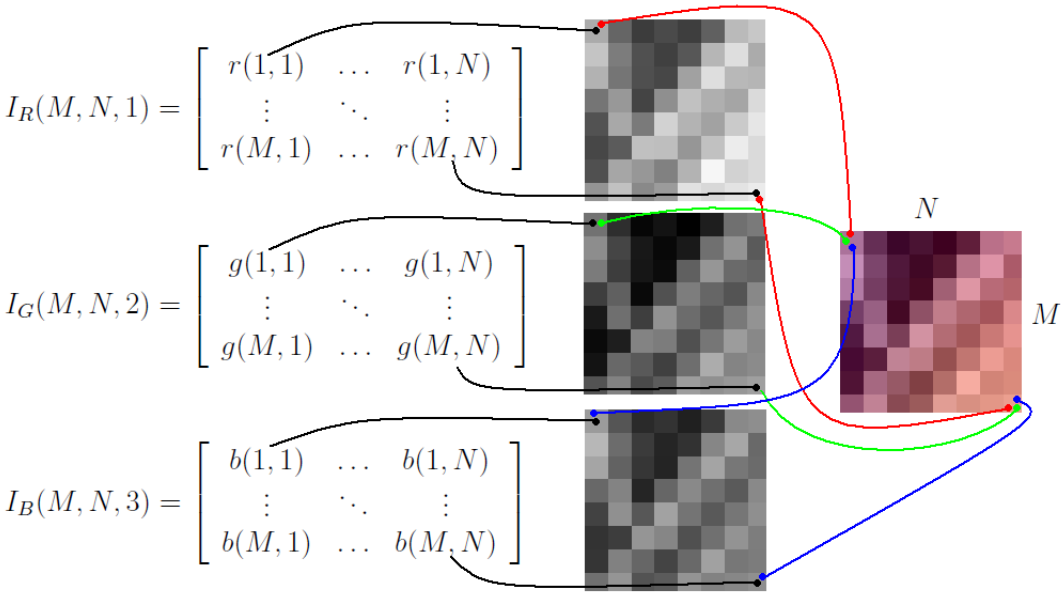


Figura 7.2: Representación matricial de una imagen a color RGB.

7.2. Cifrado

Considerar una imagen clara P de $M \times N \times 3$ pixeles, donde M son los renglones, N son las columnas y 3 representa cada uno de los componentes; cada componente RGB tiene una dimensión de $M \times N$ con valores entre $0 - 255$ (8 bits). La figura 7.3 muestra el diagrama a bloques del proceso de cifrado.

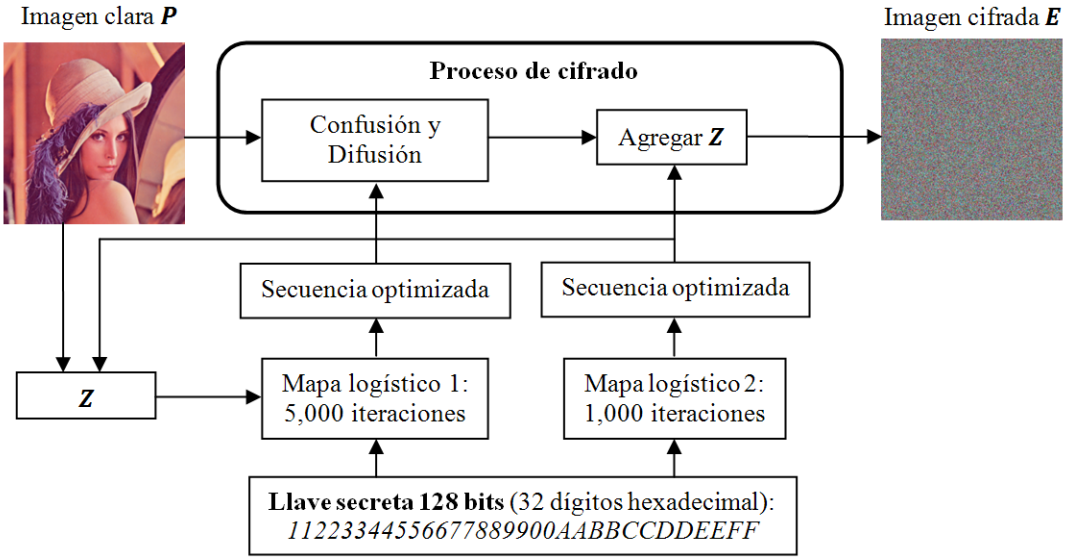


Figura 7.3: Diagrama a bloques del proceso de cifrado de imagen a color RGB.

Primero, el valor de Z se determina de la siguiente manera:

La imagen clara se transforma de $P \in [0, 255]$ a $P \in (0, 1)$ con una precisión de 10^{-15} y se realiza la siguiente operación

$$S = S + P(i, j, k), \tag{7.1}$$

para $i = 1, 2, 3, \ldots, M$; $j = 1, 2, 3, \ldots, N$; $k = 1, 2, 3$ y S es una constante inicializada en cero. S se amplifica 1,000 veces para incrementar la sensibilidad a la imagen clara. Se genera x^{L2} de $I_2 = 1000$ iteraciones; después, los últimos 50 valores caóticos de la secuencia x^{L2} son sumados como sigue

$$F = F + x^{L2}_{(I_2 - t)}, \quad para\ t = 0, 1, 2, 3, \ldots, 49 \tag{7.2}$$

donde F es una constante inicializada en cero. El siguiente cálculo es

$$V_1 = [(S \times 1000) + F] \quad (\text{mód } 1), \tag{7.3}$$

donde $V_1 \in (0, 1)$ con precisión decimal de 10^{-15} y *mód* es la operación de módulo. Es muy importante usar un valor de V_1 que sea proporcional a $1 - 254$, por tanto se calcula lo siguiente

$$V_2 = 1 + round\ (V_1 \times 253), \tag{7.4}$$

donde $V_2 \in [1, 254]$ y *round* es la operación de redondeo al valor más cercano. Finalmente, el valor de Z con precisión de 10^{-15} está determinado por

$$Z = V_2/255. \tag{7.5}$$

Segundo, el proceso de cifrado consiste de las siguientes operaciones:

Se genera la secuencia caótica x^{L1} de $I_2 = 5,000$ iteraciones y se procede a calcular las siguientes dos subsecuencias, la primera de longitud M se determina como sigue

$$RE_m = round\ \left[(x^{L1}_{(I_2-M+m)}) * (M-1)\right] + 1, \quad para\ m = 1, 2, 3, \ldots, M \tag{7.6}$$

donde $RE \in [1, M]$ es un vector con longitud M. La segunda subsecuencia de longitud N es

$$CO_n = round\ \left[(x^{L1}_{(I_2-N+n)}) * (N-1)\right] + 1, \quad para\ n = 1, 2, \ldots, N \tag{7.7}$$

donde $CO \in [1, N]$ es un vector con longitud N. Se considera que RE_m y CO_n son dos vectores optimizados, que contienen todas las posiciones de forma pseudoaleatoria (Sec. 6.5).

Una tercera subsecuencia de 5,000 valores se calcula para el proceso de difusión optimizado como sigue

$$M_g = \left\{\left[x^{L1}_g * 1000\right] + Z\right\} \quad (\text{mód } 1), \quad para\ g = 1, 2, 3, \ldots, 5000 \tag{7.8}$$

donde $M \in (0, 1)$ con precisión de 10^{-15}. Finalmente, la imagen clara se transforma de $P \in [0, 255]$ a $P \in (0, 1)$ con una precisión de 10^{-15} y los procesos de confusión y difusión (cifrado) se calcula con la siguiente expresión

$$E(i, j, k) = [P(RE_i, CO_j, k) + (M_g)] \quad (\text{mód } 1), \tag{7.9}$$

para $i = 1, 2, 3, \ldots, M$; $j = 1, 2, 3, \ldots, N$; $k = 1, 2, 3$, $g = 1, 2, 3, \ldots, M \times N$ (mód $5,000$), E es la imagen cifrada y P es la imagen clara. La imagen cifrada se transforma de $E \in (0, 1)$ a $E \in [0, 255]$ con tamaño $M \times N \times 3$.

Tercer paso, se agrega el valor de Z al criptograma:

Z se esconde en un pixel de manera pseudoaleatoria como sigue:

$$I = round\ \left\{\left[x^{L2}_{(R-10)} \times (M-1)\right] + 1\right\}, \tag{7.10}$$

$$J = round\ \left\{\left[x^{L2}_{(R-100)} \times (N-1)\right] + 1\right\}, \tag{7.11}$$

$$K = round\ \left\{\left[x^{L2}_{(R-200)} \times 2\right] + 1\right\}, \tag{7.12}$$

donde $I \in [1, M]$, $J \in [1, N]$ y $K \in [1, 3]$. De aquí, Z es incluida en la imagen cifrada como

$$E(I, J, K) = V_2. \tag{7.13}$$

El **proceso de descifrado** consiste en invertir los pasos realizados anteriormente en el cifrado, con la misma clave secreta, se recupera el valor de Z y se calcula lo siguiente para recuperar la imagen clara

$$D(RE_i, CO_j, k) = [E(i, j, k) - (M_g)] \quad (\text{mód } 1), \tag{7.14}$$

donde $i = 1, 2, 3, \ldots, M$; $j = 1, 2, 3, \ldots, N$; $k = 1, 2, 3$, $g = 1, 2, 3, \ldots, M \times N$ (mód 5000), $D \in (0, 1)$ es la imagen clara recuperada y $E \in (0, 1)$ es la imagen cifrada. La imagen descifrada se transforma de $D \in (0, 1)$ a $D \in [0, 255]$.

7.3. Análisis de seguridad

La implementación del algoritmo de cifrado se realiza en la plataforma de MatLab V7.6 (R2008a) en una computadora laptop con procesador AMD Turion 2.0 GHz, 3.18 GB de RAM y sistema operativo Windows XP 32 bits. Se utiliza representación aritmética de punto flotante (estándar IEEE 754) tipo *double* (64) bits para los datos caóticos y proceso de cifrado, por lo que se tiene una precisión de 10^{-15} decimales.

Una imagen digital tiene un amplio espectro y un algoritmo de cifrado debe ser capaz de cifrar cualquier imagen RGB con resultados similares en seguridad y desempeño. El algoritmo de cifrado de imagen RGB propuesto en este trabajo doctoral (ver Apéndice A) es aplicado a distintas imágenes para verificar la universalidad del cifrado. La figura 7.4(a)-(d) muestra cuatro imágenes claras RGB con tamaño de 512×512 con sus histogramas y las correspondientes imágenes cifradas; otras dos imágenes claras RGB se muestran en la figura 7.5(a)-(d) con sus tres RGB componentes y sus correspondientes criptogramas. En el cifrado, se utilizó como clave secreta "1234567890ABCDEF1234567890ABCDEF". Como resultado, los histogramas uniformes y componentes RGB irreconocibles de las imágenes cifradas confirmas las capacidades del algoritmo para cifrar cualquier imagen a color RGB. Además, el algoritmo puede cifrar imágenes a escala de grises.

Ahora, la robustez del algoritmo de cifrado de imagen se verifica con el siguiente análisis de seguridad: error de descifrado, espacio de claves secretas, sensibilidad a la clave secreta, sensibilidad a la imagen clara, histogramas, correlación, ataques clásicos, entropía de la información y tiempo de cifrado.

7.3.1. Error de descifrado

En aplicaciones de comunicaciones seguras como en telemedicina, milicia, biometría, etc., la imagen descifrada debe ser igual a la imagen original. En [112], los autores presentan un algoritmo con inserción de pixeles aleatorios en imagen original antes de ser cifrada; por tanto, la imagen descifrada tiene un error entre la imagen original y la descifrada hasta un 0.2 % del total de los pixeles (tabla 7.1).

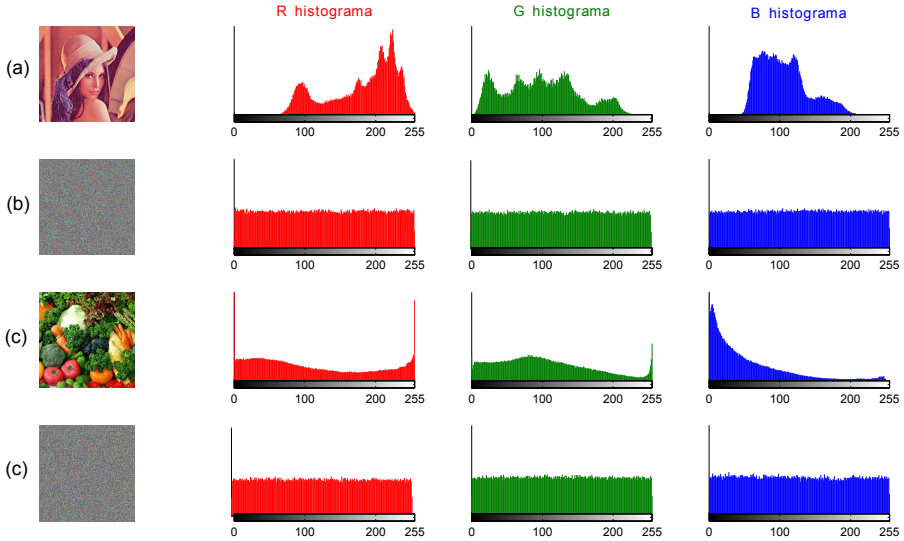


Figura 7.4: Cifrado de imagen a color: (a) Lena clara y sus histogramas RGB, (b) Lena cifrada y sus histogramas RGB, (c) vegetales claro y sus tres histogramas RGB y (d) vegetales cifrado y sus tres histogramas RGB.

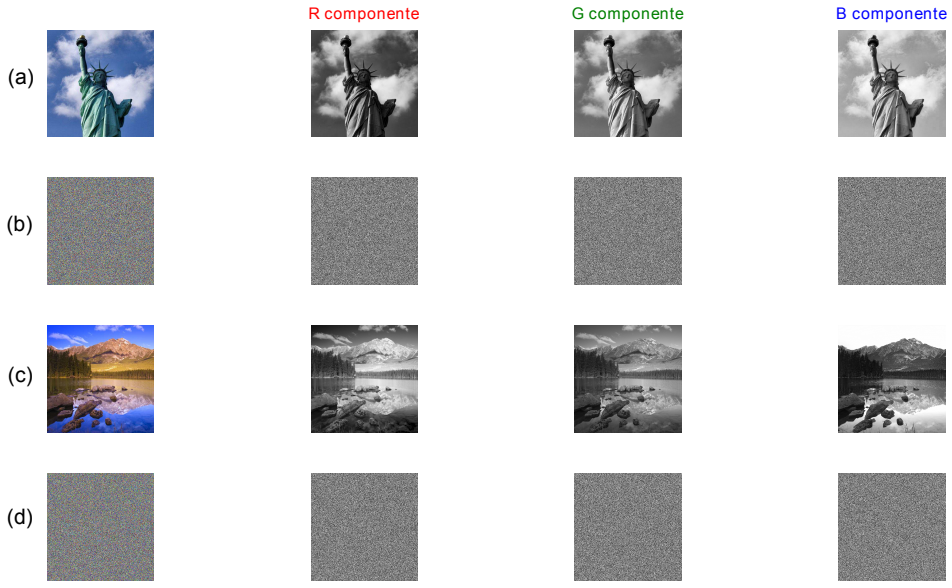


Figura 7.5: Cifrado de imagen a color: (a) estatua de la libertad clara y sus tres componentes RGB, (b) estatua de la libertad cifrada y sus tres componentes RGB, (c) paisaje claro y sus tres componentes RGB y (d) paisaje cifrado y sus tres componentes RGB.

Por otra parte, el algoritmo de cifrado caótico propuesto puede recuperar la imagen original con un error muy bajo (ver tabla 7.1); por tanto, la imagen descifrada puede considerarse similar a la imagen original. El error de descifrado se determina mediante la expresión

$$DErr(\,\%) = \frac{\sum_{i=1,j=1}^{i=M,j=N} Q(i,j)}{M \times N} \times 100 \tag{7.15}$$

donde $DErr$ es el error de descifrado, M son los renglones, N son las columnas y

$$Q(i,j) = \begin{cases} 0 & \text{if } PI(i,j) = DI(i,j) \\ 1 & \text{if } PI(i,j) \neq DI(i,j) \end{cases} \tag{7.16}$$

donde PI es la imagen clara y DI es la imagen descifrada.

512×512 imagen RGB	Componente	Algoritmo propuesto DErr (%)	Ref. [112] DErr (%)
Lena	R	0	0.1945
	G	0	0.1941
	B	0.0003	0.1953
Estatua de la libertad	R	0	0.1953
	G	0.0019	0.1934
	B	0.0007	0.1941

Tabla 7.1: Error de descifrado de imagen RGB para dos imágenes.

7.3.2. Espacio de clave secreta

Todo sistema criptográfico es susceptible a un ataque exhaustivo (ataque de *fuerza bruta*), donde cada posible clave secreta se utiliza para descifrar un criptograma. Si el espacio de claves es pequeño, es decir, menor a 2^{56} posibilidades, el sistema criptogáfico no es seguro ante un ataque exhaustivo o de fuerza bruta. "Para proporcionar seguridad suficiente contra un ataque exhaustivo, el espacio de claves debe ser mayor a 2^{100}" sugerencia 15 reportada en la referencia [78]. Además, cada clave secreta se debe considerar fuerte, es decir, que genere secuencias caóticas y no periódicas. La clave secreta propuesta consiste de 32 dígitos hexadecimales (128 bits) y todas se consideran fuertes (Sec. 6.2), por tanto, el algoritmo propuesto en esta tesis utiliza un espacio de claves de 2^{128} y puede resistir un ataque exhaustivo.

7.3.3. Sensibilidad a clave secreta

En este análisis, la sensibilidad a la clave secreta se prueba y verifica. Un buen sistema criptográfico debe ser sensible a pequeños cambios en la clave secreta tanto para cifrado y descifrado. En el proceso de descifrado, si la misma imagen clara se cifra dos veces con dos claves parecidas, las imágenes cifradas deben ser completamente diferentes entre ellas (verificada con análisis de correlación Sec. 7.3.6).

Para verificar la sensibilidad a la clave secreta, la imagen clara *"vegetales"* se cifra con 3 claves similares y los criptogramas son comparados entre ellos mediante análisis de correlación; los resultados se muestran en la tabla 7.2, donde una correlación de 0 indica que ambas imágenes son totalmente diferentes.

Por otra parte, en el proceso de descifrado sólo la clave correcta puede recuperar la imagen original, es decir, no se podrá recuperar la imagen original si se utiliza una clave similar, incluso mantiene las mismas características de seguridad. La figura 7.6(a) muestra la imagen descifrada con la clave correcta; las figuras 7.6(b) y (c) muestran dos imagenes descifradas con claves muy similares (1 bit diferente).

Esta prueba fue determinada con:

- Clave secreta 1 como *1234567890ABCDEF1234567890ABCDEF*
- Clave secreta 2 como *1234567**9**90ABCDEF1234567890ABCDEF*
- Clave secreta 3 como *1234567890ABCDEF1234567**9**90ABCDEF*

Clave secreta	Componente	Correlación
clave 1 *vs* clave 2	R	-0.1281
	G	0.0683
	B	0.0529
clave 2 *vs* clave 3	R	-0.0361
	G	0.0856
	B	-0.0155

Tabla 7.2: Sensibilidad a clave secreta en proceso de cifrado de imagen RGB.

7.3.4. Sensibilidad a imagen clara

Un buen sistema criptográfico debe ser sensible con respecto a la imagen clara, es decir, pequeños cambios (un bit) en la imagen clara genera un gran cambio en la imagen cifrada; si el algoritmo tiene esta propiedad, el cifrado puede resistir un ataque diferencial, el cual, básicamente es un ataque de texto claro conocido. Dos medidas son utilizadas para determinar la sensibilidad a imagen clara: $NPCR$ (del inglés, *Net Pixel Change Rate*), tasa de cambio de pixel neto y $UACI$ (del inglés, *Unified Avarage Changing Intensity*), promedio unificado de cambio de intensidad.

$NPCR$ mide cuantos pixeles son diferentes entre dos imágenes cifradas E_1 y E_2 con la misma clave secreta; es representado en porcentajes, donde 100 % indica que ambas imágenes son totalmente diferentes (cada pixel comparado a la par tienen valor diferente). $NPCR$ a nivel de componente se calcula como sigue

$$NPCR = \frac{\sum_{i=1,j=1}^{i=M,j=N} W(i,j)}{M \times N} \times 100 \tag{7.17}$$

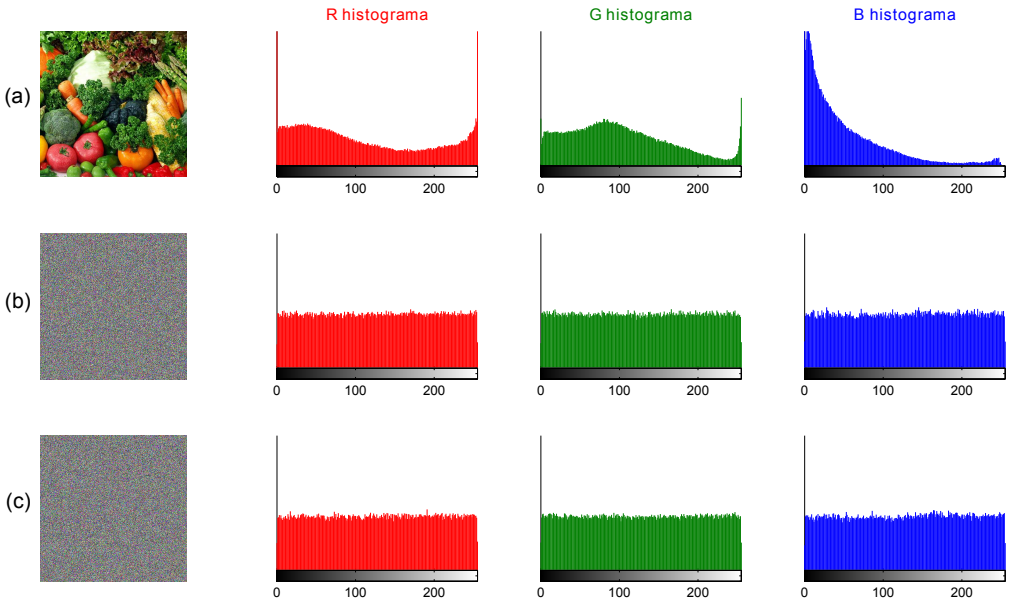


Figura 7.6: Sensibilidad a clave secreta en proceso de descifrado: (a) imagen descifrada con la clave secreta correcta 1 y sus histogramas RGB, (b) imagen descifrada con incorrecta clave secreta 2 y sus histogramas RGB y (c) imagen descifrada con incorrecta clave secreta 3 y sus histogramas RGB.

donde M son los renglones, N son las columnas y

$$W(i,j) = \begin{cases} 0 & \text{if } E_1(i,j) = E_2(i,j) \\ 1 & \text{if } E_1(i,j) \neq E_2(i,j) \end{cases} \tag{7.18}$$

$UACI$ es el promedio de intensidad diferente entre dos imágenes cifradas E_1 y E_2 con la misma clave secreta, es decir, que tan diferentes son diferentes en magnitud. $UACI$ a nivel componente se calcula

$$UACI = \frac{100}{M \times N \times 255} \sum_{i=1,j=1}^{i=M,j=N} |E_1(i,j) - E_2(i,j)|, \tag{7.19}$$

donde M son los renglones, N son las columnas, $E_1(i,j)$ y $E_2(i,j)$ son los valores de cada imagen cifrada.

$NPCR$ y $UACI$ son calculados con los siguientes pasos: primero, la imagen clara de *"Lena"* es cifrada E_1 con la clave secreta 1 (figura. 7.7(a)-(b)); después, el valor de un pixel en $P(1,20,1) = 222$ se cambia a $P(1,20,1) = 223$ (un bit diferente en componente R) y *"Lena"* con un pequeño cambio es cifrada E_2 con la misma clave secreta 1 (figura. 7.7(c)-(d)). La tabla 7.3 muestra los resultados para ambas medidas; el valor de $NPCR$ es cercano a 100 %, es decir, ambas imágenes cifradas E_1 y E_2 son muy diferentes. El valor de $UACI$ es del 33 % entre E_1 and E_2, es decir, un 33 % diferentes en magnitud. Por tanto, el algoritmo propuesto es altamente sensible a la imagen clara por

lo que es robusto ante ataques diferenciales como ataque de sólo imagen clara conocida. En la tabla 7.4 se muestra una comparación con otros algoritmos criptográficos recientes presentados en la literatura, los cuales, tienen valores similares a los reportados en esta tesis doctoral.

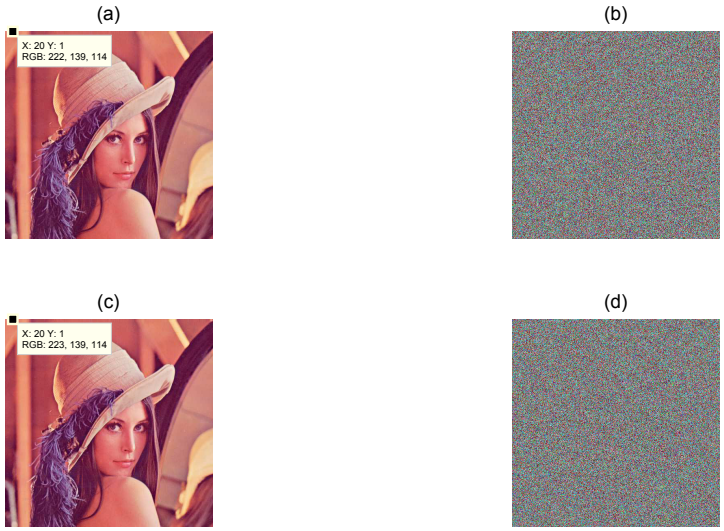


Figura 7.7: Análisis diferencial de dos imágenes similares: (a) imagen clara Lena, (b) imagen cifrada Lena, (c) imagen clara Lena con un pequeño cambio y (d) Lena con un pequeño cambio es cifrada con la misma clave secreta.

Prueba	R	G	B
$NPCR(\%)$	99.63	99.60	99.61
$UACI(\%)$	33.31	33.34	33.43

Tabla 7.3: Resultados de $NPCR$ y $UACI$ con el algoritmo propuesto.

Prueba	Propuesto	Ref. [112]	Ref. [107]
$NPCR(\%)$	99.61	No presenta	99.60
$UACI(\%)$	33.36	No presenta	33.40

Tabla 7.4: Comparación de análisis diferencial con otros algoritmos reportados en la literatura.

7.3.5. Histogramas

El histograma muestra la intensidad de los colores de la imagen, mediante una gráfica y representa información estadística. Un sistema criptográfico es susceptible a un ataque de histogramas y puede resistir, si la imagen cifrada tiene un histograma uniforme (información impredecible). El algoritmo propuesto genera un histograma

uniforme a nivel componente (excelente proceso de difusión) que se puede observar en la figura 7.4(b) y (d). Otros ejemplos se muestran en la figura 7.6(b) y (c) en sensibilidad a clave secreta. Por tanto, el algoritmo criptográfico propuesto en esta tesis se considera robusto contra ataques de histogramas.

7.3.6. Análisis de correlación

En esta sección, el desempeño del proceso de confusión y difusión propuesto es probado y verificado. Se sabe que los pixeles adyacentes de una imagen están altamente correlacionados, esto significa que el valor de un pixel y el valor de su pixel vecino (horizontal, vertical o diagonal) son muy parecidos. La correlación de una imagen se puede graficar y también se puede medir entre -1 y 1, donde 0 significa correlación nula. Un criptoanalista puede utilizar esta información en un ataque estadístico para encontrar la clave secreta y recuperar la imagen clara. Por tanto, la imagen cifrada debe tener correlación nula.

La figura 7.8(a)-(c) muestra la distribución de la correlación RGB horizontal de 5,000 pixels de la imagen de *Lena* seleccionados aleatoriamente, donde cada valor de un pixel y el valor de su pixel adyacente horizontal es similar (alta correlación). Por otra parte, las figuras 7.8(d)-(f) muestran la correlación de la imagen cifrada, donde se muestra que el valor entre el pixel y su adyacente son muy distintos (baja correlación).

De manera teórica, la correlación se determina como sigue

$$Cr = \frac{N \times \sum_{i=0}^{N}(x_i \times y_i) - \sum_{i=0}^{N} x_i \times \sum_{i=0}^{N} y_i}{\sqrt{\left(N \times \sum_{i=0}^{N}(x_i)^2 - \left(\sum_{i=0}^{n} x_i\right)^2\right) \times \left(N \times \sum_{i=0}^{N}(y_i)^2 - \left(\sum_{i=0}^{n} y_i\right)^2\right)}} \tag{7.20}$$

donde x y y son valores de dos pixeles adyacentes a nivel componente y N es el número de pares de pixeles. El valor de correlación es $Cr \in (-1, 1)$ donde 0 significa nula correlación y 1 significa alta correlación. La tabla 7.5 se muestra el coeficiente de correlación horizontal de 5,000 pares de pixeles seleccionados aleatoriamente; la correlación de la imagen clara es cercano a 1 (alta correlación), mientras que la correlación de la imagen cifrada es cercano a 0 (baja correlación). Por tanto, el algoritmo propuesto puede resistir un ataque estadístico de correlación.

7.3.7. Ataque de sólo texto claro elegido y conocido

Muchos algoritmos de cifrado basado en caos con excelentes características estadísticas fueron quebrantados con un ataque de sólo texto claro conocido/elegido. En el algoritmo propuesto se consideran los siguientes puntos para prevenir este tipo de ataque poderoso:

1. Los procesos de confusión y difusión son realizados en una sóla etapa de acuerdo con la ecuación (7.9). Por tanto, el algoritmo propuesto en esta tesis puede resistir un ataque divide y vencerás [60].

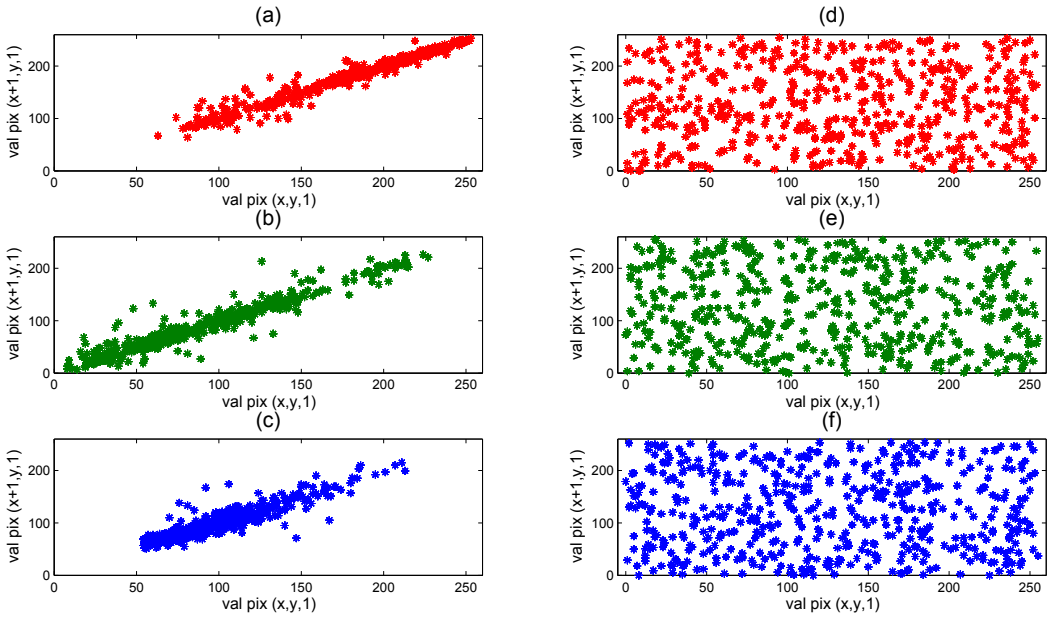


Figura 7.8: Distribución de correlación horizontal de imagen clara y de imagen cifrada de Lena: (a) componente R de imagen clara, (b) componente G de imagen clara, (c) componente B de imagen clara, (d) componente R de imagen cifrada, (e) componente G de imagen cifrada y (f) componente B de imagen cifrada.

512×512 Imagen RGB	Componente	Imagen clara	Imagen cifrada
Lena	R	0.9777	0.0135
	G	0.9604	-0.0835
	B	0.9101	-0.0170
Vegetales	R	0.9256	0.0306
	G	0.9113	0.0326
	B	0.9142	0.0287
Estatua de la libertad	R	0.9823	-0.0652
	G	0.9778	0.0641
	B	0.9822	0.0551
Paisaje	R	0.9715	0.0508
	G	0.9253	0.0517
	B	0.9647	0.0251

Tabla 7.5: Coeficiente de correlación horizontal.

2. La secuencia caótica para cifrado es determinada de la clave secreta y de las características totales de la imagen clara de acuerdo con la tabla 6.1.

3. Se utiliza una mejor distribución del mapa logístico en el proceso de cifrado de acuerdo con la ecuación (7.8); la figura 6.4(b) muestra esta distribución.

4. El proceso de cifrado es realizado bajo *mod* 1 con una precisión decimal de 10^{-15} y la imagen cifrada es transformada y redondeada a $E \in [0, 255]$ en el último paso de cifrado.

En un ataque de imagen clara elegida, el criptoanalista tiene acceso al sistema criptografico y puede elegir una imagen especial para cifrar y buscar una relación entre la entrada y salida, para determinar la clave secreta: primero, el criptoanalista elige una imagen negra con todos los pixeles fijos en cero (figura 7.9(a)); después, el criptoanalista cifra la imagen negra (figura 7.9(b)) y el resultado es la posible secuencia caótica utilizada en el cifrado (clave secreta). Posteriormente, el criptoanalista puede implementar un ataque de imagen clara conocida (figura 7.9(c)) con la posible clave secreta pero no tiene éxito (figura 7.9(d)). De la tabla 6.1 el valor de S de la figura 7.9(a) es 0, mientras que *Lena* fue cifrada con un valor de $S \neq 0$. Por tanto, se tienen diferentes valores de Z y diferentes secuencias caóticas para cada imagen cifrada.

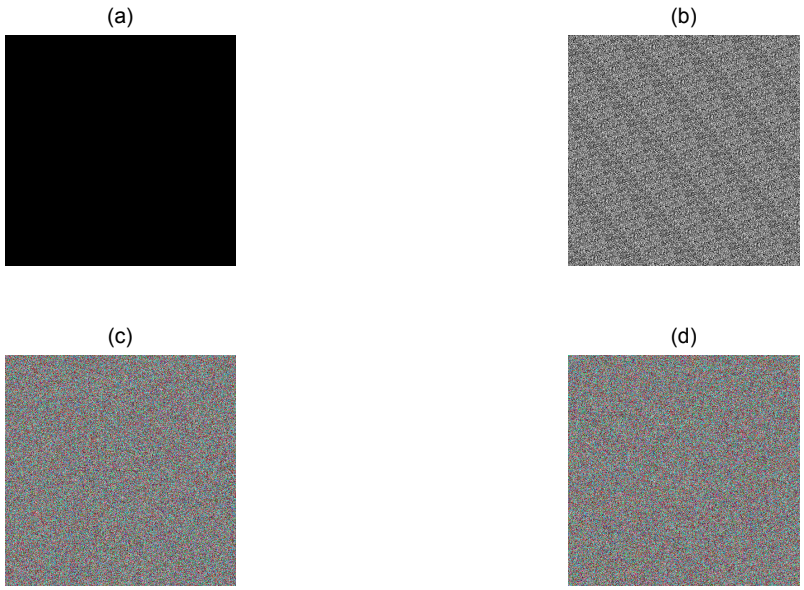


Figura 7.9: Ataque de sólo imagen clara conocida y elegida: (a) imagen clara elegida, (b) imagen negra cifrada, (c) imagen de *Lena* cifrada y (d) imagen de *Lena* descifrada con la posible clave secreta.

7.3.8. Entropía de la información

La *entropía* determina que tan impredecible es un mensaje, es decir, mide cuanto desorden genera el algoritmo de cifrado. Si el proceso de cifrado es bueno, este genera alto desorden en la imagen cifrada; por tanto, mayor será la entropía. Caso contrario, si el proceso de cifrado no es suficientemente aleatorio, el algoritmo criptográfico puede estar sujeto a un exitoso ataque de entropía, porque el criptograma es predecible.

En esta sección, el desempeño del cifrado propuesto en la etapa de difusión es probado y verificado. La entropía $H(m)$ de un mensaje m puede calcularse como sigue

$$H(m) = \sum_{i=0}^{2^N-1} p(m_i)\ log_2(1/p(m_i)), \tag{7.21}$$

donde N es el número de bits que representan la unidad básica del mensaje m, 2^N son todas las combinaciones de la unidad básica, $p(m_i)$ representa una probabilidad de m_i, $\log_2$ es el logaritmo base 2 y la entropía esta expresada en bits, donde la máxima entropía es N. Si un mensaje m es cifrado con 2^N posibles valores, la entropía debería ser idealmente $H(m) = N$, si m es puramente aleatorio.

Una imagen RGB esta representado con 8 bits como unidad básica, es decir $N = 8$ a nivel componente. De manera que, la máxima entropía por componente es de 8. La tabla 7.6 muestra los resultados de la entropía y una comparación con otros algoritmos recientes reportados en la literatura. La entropía de la imagen cifrada es cercana a 8, por tanto el proceso de difusión genera alto desorden para resistir un ataque de entropía.

Imagen RGB	Componente	Algoritmo propuesto	Ref. [112]	Ref. [107]
256×256	R	7.9949	7.9976	NP
	G	7.9953	NP	NP
	B	7.9942	NP	NP
512×512	R	7.9974	7.9993	7.9993
	G	7.9975	NP	NP
	B	7.9969	NP	NP
1024×1024	R	7.9978	7.9998	NP
	G	7.9976	NP	NP
	B	7.9976	NP	NP

Tabla 7.6: Resultados de entropía y su comparación con otros algoritmos reportados en la literatura, donde NP significa que no presento.

7.3.9. Tiempo de cifrado

Un buen algoritmo de cifrado debe ser robusto ante ataques pero además requiere ser rápido para aplicaciones de tiempo real en telemedicina, milicia, imagen personal,

sistemas biométricos, industria, etc. El tiempo de cifrado se determina mediante los comandos *tic* y *toc* de MatLab. La tabla 7.7 y tabla 7.8 muestran los resultados de tiempo de cifrado del algoritmo propuesto con imágenes RGB y grises de distinto tamaño y los resultados se comparan con otros algoritmos criptográficos recientes reportados en la literatura. El tiempo de descifrado es similar al cifrado. La velocidad de cifrado para una imagen RGB de 256×256 (1.572 MB) con el algoritmo propuesto alcanza los 24 MB/seg. Por tanto, es ámpliamente superior a los otros algoritmos.

Imagen (pixel)	Algoritmo propuesto	Ref. [112]	Ref. [106]
256×256	0.0657	0.1789	0.1933
512×512	0.2432	0.6639	0.7579
1024×1024	1.1205	3.1416	2.9293

Tabla 7.7: Tiempo de cifrado en segundos para imágenes a color RGB.

Imagen (pixel)	Algoritmo propuesto	Ref. [108]	Ref. [110]	Ref.[98]
125×125	0.0128	0.0980	NP	NP
256×256	0.0373	NP	0.0320	NP
512×512	0.1198	NP	NP	0.1244
1024×1024	0.4735	NP	NP	NP

Tabla 7.8: Tiempo de cifrado en segundos para imágenes a escala de grises, donde NP significa que no presento.

7.4. Comparación con otro algoritmo de la literatura

En esta sección, el algoritmo propuesto se compara en términos de seguridad, desempeño y eficiencia con un algoritmo criptográfico basado en codificación ADN y mapa logístico. Recientemente, en [113] se propuso un algoritmo de cifrado basado en codificación ADN combinado con mapa logístico; los autores utilizan operaciones algebraicas ADN, codificación binaria ADN y datos caóticos del mapa logístico en el proceso de confusión y difusión. A pesar que el algoritmo criptográfico utiliza conceptos de criptografía ADN y teoría de caos para incrementar la seguridad, el algoritmo fue quebrantado con un ataque de sólo imagen clara conocida porque la codificación ADN y secuencias caóticas son dependientes únicamente de la clave secreta. Además, el cifrado presenta baja sensibilidad a la imagen clara y a la clave secreta [114].

La tabla 7.9 muestra la comparación entre el algoritmo criptográfico propuesto y el algoritmo de la referencia [113]. Con base en la comparación de los resultados, el algoritmo criptográfico propuesto en esta tesis es altamente seguro, presenta gran desempeño y puede resistir un ataque de imagen clara conocida y elegida. Algunos puntos concuerdan, como histogramas, correlación y entropía que están relacionados con un buen

proceso de confusión-difusión. Sin embargo, un sistema criptográfico para imágenes debe incluir los siguientes puntos para considerarse eficiente y seguro [78]:

1. El espacio de la clave secreta debe ser claramente definida sobre 2^{100} todas las combinaciones consideradas fuertes.
2. Ser altamente sensible a la clave secreta.
3. Ser altamente sensible a la imagen clara.
4. Las secuencias caóticas utilizadas en el proceso de cifrado, deben ser dependientes de la clave secreta y de la imagen clara, para resistir un ataque de imagen clara conocida y elegida.
5. Un buen proceso de confusión y difusión: histograma uniforme, correlación cercana a 0 y entropía cercana a 8.
6. Alta velocidad de cifrado para aplicaciones en tiempo real.

	Algoritmo Propuesto	Ref. [113]
Seguridad		
Espacio de clave	2^{128} todas fuertes	2^{166} con claves equivalentes
Sensibilidad a la clave	Si	No
Sensibilidad a la imagen clara	Si	No
Características de imagen clara para cifrado	Si	No
Datos caóticos optimizados	Si	No
Histograma uniforme	Si	Si
Coeficiente de correlación	0.0135	0.0059
Entropía de componente R	7.997	7.989
Desempeño		
Velocidad de cifrado	24 MB/seg	21 MB/seg

Tabla 7.9: Comparación del algoritmo de cifrado de imagen RGB propuesto en la tesis *vs* algoritmo reciente reportado en literatura.

7.5. Conclusiones

El algoritmo criptográfico propuesto en esta tesis doctoral cumplió con los requerimientos de seguridad que se requieren para ser implementado en la protección de imágenes a color RGB. Además, se mostró que el algoritmo de cifrado propuesto genera un criptograma con excelente propiedades estadísticas y puede resistir los ataques más poderosos reportados en la literatura.

Capítulo 8

Cifrado caótico de texto alfanumérico en microcontrolador

Actualmente, muchos sistemas embebidos están conectados a internet para transmitir información privada o simplemente para almacenamiento de datos. Por ejemplo, desde un sistema embebido que esta instalado en una casa habitación para controlar de forma remota aparatos electrodomésticos o la temperatura del aire acondicionado mediante una pagina web, hasta aplicaciones industriales donde se miden y controlan variables físicas y procesos de manufactura o estaciones meteorológicas que se ubican de forma aislada y remota para trasmitir via internet los detalles del clima. Toda esta información sensible requiere ser cifrada antes de su transmisión y almacenamiento para aportar seguridad a la comunicación y en la base de datos, respectivamente.

En este capítulo, se describe una implementación en microcontrolador de 32 bits del algoritmo de cifrado simétrico propuesto, para proteger texto en sistemas embebidos, se verifica la seguridad y eficiencia mediante un amplio análisis de seguridad como espacio de claves, sensibilidad a la clave y texto claro, frecuencia flotante, histogramas, N-gramas, autocorrelación, entropía y ataques clásicos. También, se realiza un análisis de implementación y desempeño como detalles de programación, memoria requerida, frecuencia de operación, costos de implemetación y tiempo de cifrado. Todos los análisis, muestran y verifican que el algoritmo propuesto en este trabajo doctoral, puede implementase en sistemas embebidos para cifrado de texto con alta seguridad lógica en aplicaciones de tiempo real.

8.1. Introducción

El problema de seguridad en sistemas de comunicaciones inseguros, se discutió ampliamente en los capítulos anteriores, se mostró que se requiere de un cifrado para mantener la integridad de la información en su almacenamiento y transmisión. Actualmente, *Triple Data Encryption Algorithm* (*TDEA*) y *Advanced Encryption Standard* (*AES*) son estándares de cifrado simétrico adoptados por los EUA, de los cuales, los datos son cifrados en bloques y la misma clave secreta se utiliza para cifrar y descifrar.

Están basados en una red de feistel y en la arquitectura de confusión y difusión, respectivamente [115, 116]. Sin embargo, *TDEA* es más lento que *AES* para cifrar datos, pero *AES* tiene desventajas en el número de rondas de confusión y difusión, es un algoritmo altamente estructurado que podría ser quebrantado en los siguientes años [117]. *Rivest, Shamir and Adleman* (*RSA*) es un algoritmo de cifrado asimétrico con ventajas de seguridad pero requiere de mayor tiempo y memoria que cualquier otro algoritmo simétrico.

Recientemente, criptografía ADN (ácido desoxirribonucleico) ha sido propuesta para cifrado de texto, donde las cuatro bases de ADN son caracterizados por datos binarios, operaciones de complemento ADN se utiliza en proceso de cifrado y secuencias ADN son usadas como claves secretas [118–123]. Sin embargo, los algoritmos basados en ADN no son efectivos debido a la influencia de factores biológicos [93]. Por otra parte, las características de caos estan relacionadas con propiedades criptográficas, por lo que se han presentado algoritmos criptográficos basados en caos para texto [40] y principalmente para imágenes (Sec. 7.1).

Existen muchas áreas como la milicia, industria, comercio electrónico, sistemas biométricos, telemedicina, datos personales, etc., en las cuales, se requiere mantener la integridad de la información mediante cifrado para evitar problemas de seguridad cuando la información se transmite a través de canales inseguros o en su almacenamiento en sistemas embebidos [124]. En los últimos años, se realizaron algunos avances de implementaciones en microcontrolador de sistemas caóticos y pocas implementaciones de algoritmos de cifrado caótico (Sec. 4.6). Sin embargo, estas implementaciones recientes no presentan un análisis detallado sobre la seguridad del cifrado ni muestran la existencia de caos, por lo que la seguridad y desempeño no se verifica en su totalidad.

En este capítulo, se presenta una implementación digital en un microcontrolador de 32 bits del algoritmo de cifrado propuesto en este trabajo de tesis, con el propósito de proteger datos en forma de texto alfanumérico. Un análisis de seguridad completo muestra la efectividad de la implementación y la seguridad que brinda, por lo que puede utilizarse para proteger la integridad de datos en sistemas embebidos en tiempo real. En las siguientes secciones, se muestran detalles del cifrado, después la implementación digital y el análisis de seguridad para resistir ataques de tipo lógicos (teóricos).

8.2. Cifrado

Se considera un texto claro P con longitud ℓ basado en código imprimible ASCII: minúsculas, mayúsculas, espacio, números y signos de puntuación, con un total de 95 caracteres. Cada código tiene un valor decimal entre 32-126 de acuerdo a la tabla ASCII. La figura 8.1 muestra el diagrama a bloques del proceso de cifrado de texto, donde se utiliza como clave secreta *11223344556677889900AABBCCDDEEFF*.

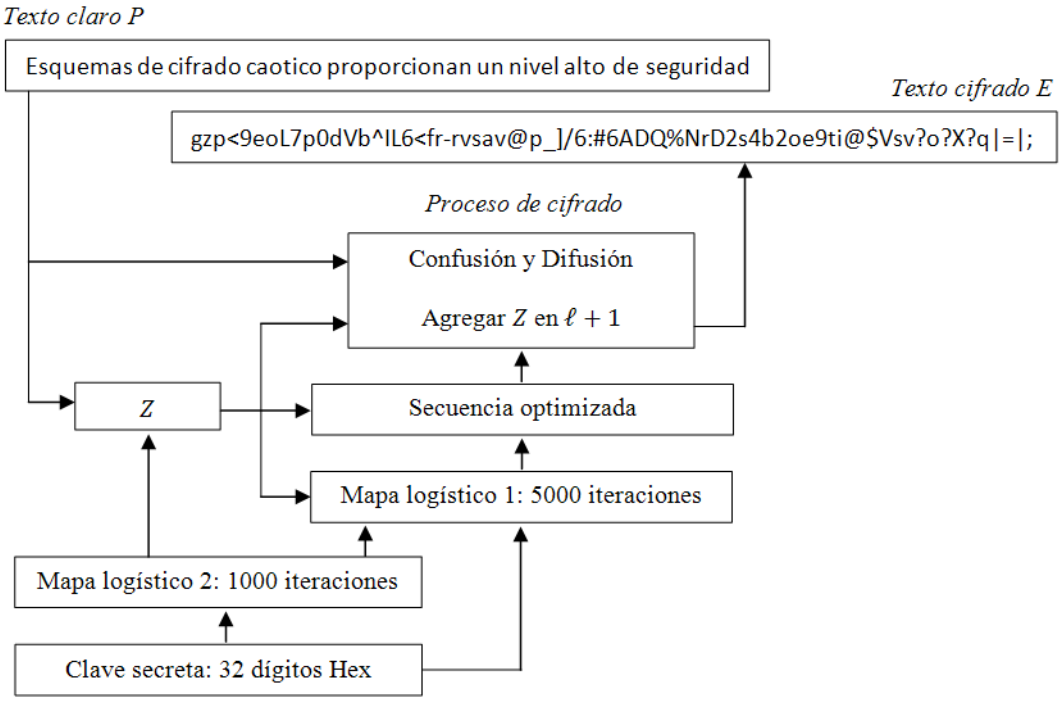


Figura 8.1: Diagrama a bloques del proceso de cifrado caótico de texto.

Primero, el valor de Z se determina de la siguiente manera:

Todos los caracteres del texto claro se suman con datos caóticos del mapa logístico para incrementar la seguridad ante ataques diferenciales. El mapa logístico 2 es iterado $I_2 = \ell + 100$ veces con valores de a_2 y x_{2_0} tomados de la tabla 6.1 para generar la secuencia de datos caóticos x^{L2} con $x^{L2} \in (0, 1)$ y una precisión decimal 10^{-15}. Los valores del texto claro se suman de acuerdo a la siguiente expresión

$$S = \left[S + \left(P_i * x^{L2}_{I_2+1-i}\right)\right] \quad (\text{mód } 1), \quad para\ i = 1, 2, 3, \ldots, \ell \tag{8.1}$$

donde P_i representa el texto claro en valor decimal, S es una variable inicializada en cero y *mód* es la operación de módulo. Z debe ser un valor proporcional a 32-126 para evitar su cancelación ante un ataque diferencial, donde se utiliza un texto claro fijo en 32 (espacio); esto se resuelve de la siguiente forma

$$V = \ round(S * 94) + 32, \tag{8.2}$$

donde *round* es la operación de redondeo al valor más cercano. Finalmente, Z con una precisión de 10^{-15} se determina por

$$Z = V/127. \tag{8.3}$$

Segundo, el proceso de cifrado consiste de las siguientes operaciones:

El mapa logístico 1 se itera $I_1 = 5,000$ con valores de a_1 y x_{1_0} tomados de la tabla 6.1 para generar la secuencia caótica de datos x^{L1} con $x^{L1} \in (0, 1)$ y con una precisión de

10^{-15}. De esta secuencia caótica se determinan dos subsecuencias, una para el proceso de confusión y otra para difusión como sigue

$$PERM_i = \ round\left[x^{L1}_{I_1-\ell+i} * (\ell - 1)\right] + 1, \quad para\ i = 1, 2, 3, \ldots, \ell \tag{8.4}$$

donde $PERM_i \in [1, \ell]$ es un vector con longitud ℓ. Se considera que el vector $PERM_i$ esta optimizado y contiene todas las posiciones de forma pseudoaleatoria (Sec. 6.5).

La segunda subsecuencia para el proceso de difusión es

$$M_i = \left[\left(x^{L1}_{I_1-\ell+i} * 100\right) + Z\right] \quad (\text{mód } 1) \ \ para\ i = 1, 2, 3, \ldots, \ell \tag{8.5}$$

donde $M_i \in (0, 1)$ es un vector de longitud ℓ. Después, M_i se transforma como

$$Y_i = \ round\left(M_i * 94\right), \quad para\ i = 1, 2, 3, \ldots, \ell \tag{8.6}$$

donde $Y_i \in [1, 95]$ es un vector de longitud ℓ. Finalmente, el proceso de cifrado se calcula con

$$E_i = \{[P(PERM_i) - 32] + Y_i \ mod\ 95\} + 32, \quad para\ i = 1, 2, 3, \ldots, \ell \tag{8.7}$$

donde $E_i \in [32, 126]$ es el texto cifrado y $P_i \in [32, 126]$ es el texto claro.

Tercer paso

El valor de Z se agrega al criptograma de la siguiente forma

$$E_{\ell+1} = V, \tag{8.8}$$

donde $V_i \in [32, 126]$.

El **proceso de descifrado** consiste básicamente de invertir los pasos del cifrado. Primero, Z se recupera del criptograma como sigue

$$Z = E_{\ell+1}/127. \tag{8.9}$$

El valor de Z puede ser conocido por intrusos, pero **recuperar el mensaje original sin la clave correcta es casi imposible**. Después, con la clave secreta de 128 bits y Z, 5000 datos caóticos del mapa logístico 1 para determinar $PERM_i$, M_i y Y_i como en proceso de cifrado. En la figura 8.2 se muestra el esquema a bloques del proceso de descifrado (notar que no se requiere el mapa logístico 2, por lo que se ahorra tiempo de cálculo). Se utiliza como clave secreta *11223344556677889900AABBCCDDEEFF*. El proceso de descifrado se calcula con la siguiente expresión

$$D(PERM_i) = \{[(E(i) - 32) - Y(i)] \ mod\ 95\} + 32, \quad para\ i = 1, 2, 3, \ldots, \ell \tag{8.10}$$

donde $D \in [32, 126]$ es el texto claro recuperado.

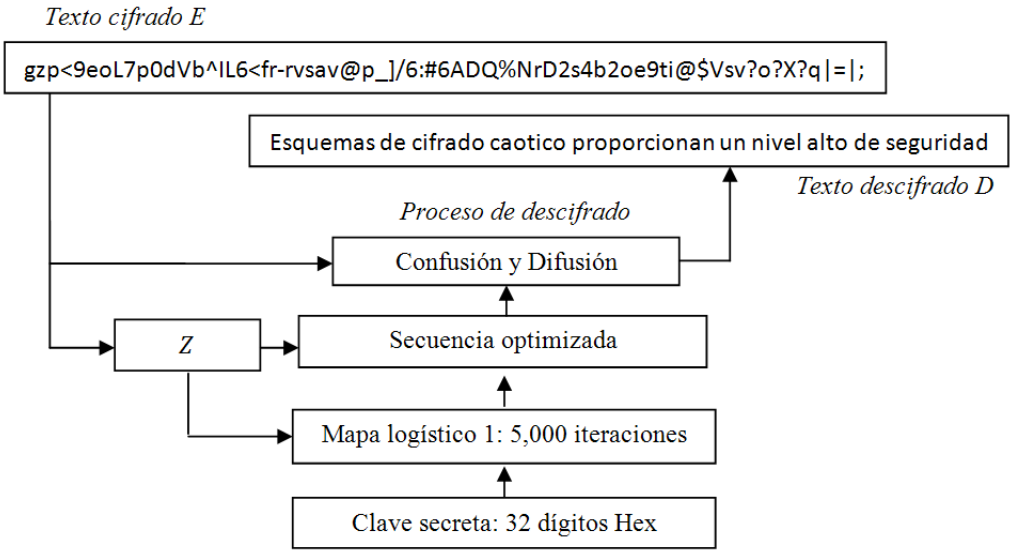


Figura 8.2: Diagrama a bloques del proceso de descifrado de texto.

8.3. Análisis de seguridad

El algoritmo de cifrado se implementa en un microcontrolador de 32 bits con procesador COLDFIRE de Freescale (M52259DEMOKIT) que tiene una frecuencia de operación hasta 80 MHz [125]. Se utiliza el software CodeWarrior para Colfire V7.1 para realizar la programación en lenguaje C y también, se usan librerías de MQX 3.5 para almacenar datos en memoria USB. El programa del algoritmo de cifrado es almacenado en memoria FLASH del microcontrolador. Para los análisis de seguridad se utiliza la plataforma de MatLab V7.6 (R2008a) en una computadora laptop con procesador AMD Turion 2.0 GHz, 3.18 GB de RAM y sistemas operativo Windows XP 32 bits; se utiliza representación punto flotante con precisión doble (64 bits) y una precisión de 10^{-15}. La clave secreta de 32 dígitos hexadecimales y el texto claro son introducidos en el microcontrolador por programación (de manera práctica); sin embargo, la clave y el texto claro pueden ser introducidos con un teclado o una memoria USB, comunicación RS-232, conexión inalámbrica, etc.

Los primeros 945 caracteres de este capítulo se utiliza como texto claro (no se consideran los acentos) y *1122334455667788990OAABBCCDDEEFF* como clave secreta. Una vez que el texto es cifrado por el microcontrolador, éste se extrae en formato *.txt con memoria USB para realizar los distintos análisis de seguridad lógicos en MatLab. La figura 8.3 muestra la clave secreta, el texto claro y el correspondiente texto cifrado extraído del microcontrolador.

8.3.1. Espacio de clave secreta

De acuerdo con la Sec. 7.3.2, el espacio de claves es de 2^{128} posibilidades, por lo que puede resistir un ataque exhaustivo o de fuerza bruta.

Figura 8.3: Texto claro y texto cifrado extraído del microcontrolador con memoria USB.

8.3.2. Sensibilidad a clave secreta

Alvarez y Li [78], mencionan que un algoritmo de cifrado debe ser altamente sensible a la clave secreta, incluso a nivel de bit. Esta propiedad aplica para el proceso de cifrado y descifrado; en el proceso de cifrado, si el mismo texto claro es cifrado dos veces con dos claves secretas similares (un bit de diferencia), el texto cifrado debe ser muy diferente entre ellos y sin correlación; mientras que, en el proceso de descifrado, únicamente la clave secreta correcta puede recuperar el mensaje original. En esta sección, la sensibilidad de la clave secreta en el proceso de cifrado y descifrado, se prueba y verifica con tres claves secretas similares (ver tabla 8.1).

Para la prueba, se utilizan los primeros 50 caracteres de este capítulo: la sensibilidad de la clave secreta en el proceso de cifrado se muestra en la figura 8.4, en el cual, cada clave secreta utilizada, es decir CLAVE 1, CLAVE 2 Y CLAVE 3 generan diferentes criptogramas; mientras que en la figura 8.5 muestra que sólo la clave secreta correcta (CLAVE 1) recupera el mensaje original. Por tanto, el algoritmo de cifrado caótico de esta tesis, para cifrado de texto es altamente sensible a la clave secreta.

No. clave	Clave secreta
CLAVE 1	11223344556677889900AABBCCDDEEFF
CLAVE 2	1122334**5**556677889900AABBCCDDEEFF
CLAVE 3	11223344556677889900AAB**C**CCDDEEFF

Tabla 8.1: Claves secretas utilizadas para análisis de sensibilidad a la clave en cifrado de texto alfanumérico.

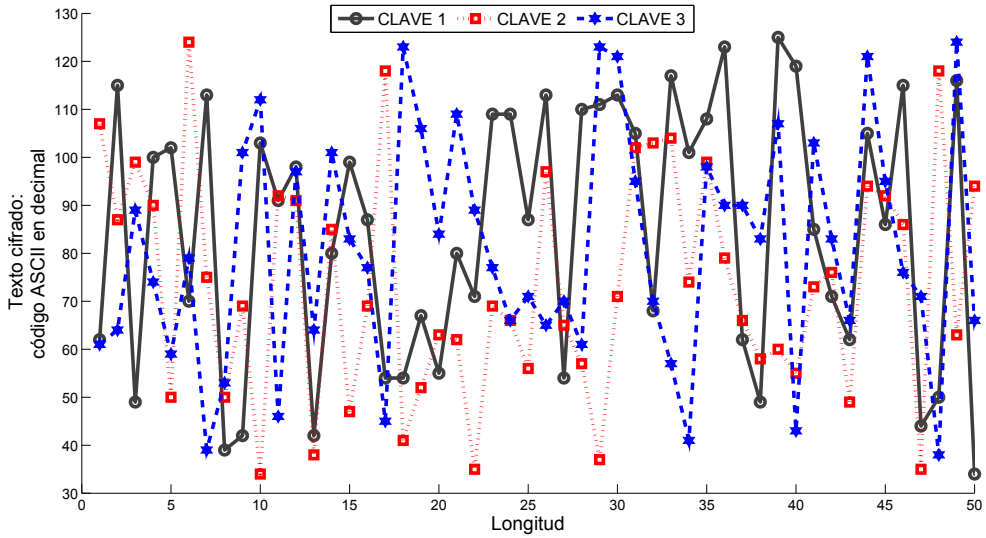


Figura 8.4: Sensibilidad a la clave secreta en proceso de cifrado de texto.

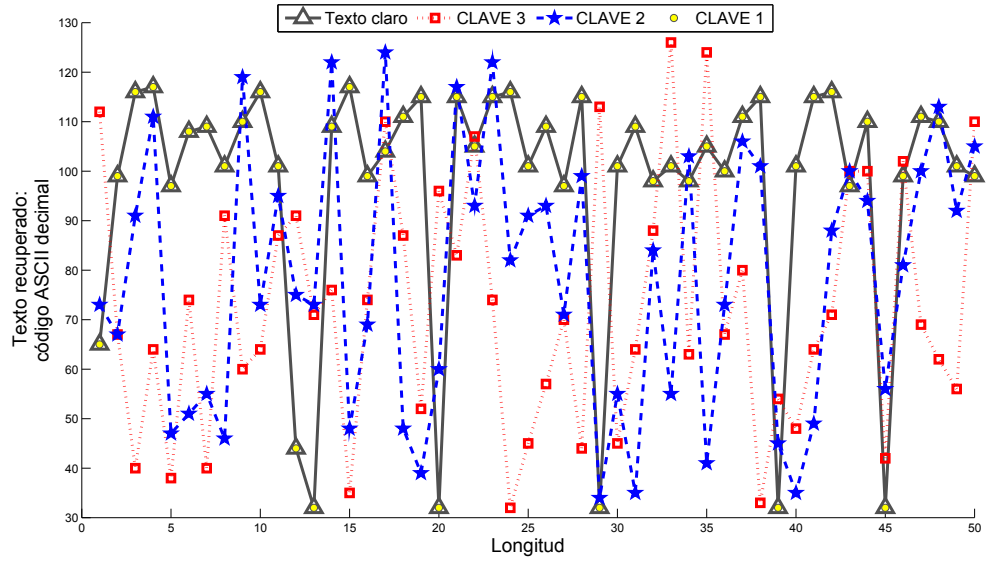


Figura 8.5: Sensibilidad a la clave secreta en proceso de descifrado de texto

8.3.3. Sensibilidad a texto claro

De acuerdo con la Sec. 7.3.4, para resistir un ataque diferencial, se requiere que el algoritmo criptográfico presente sensibilidad a pequeños cambios en el texto claro y se utilizan dos conceptos para determinarlo: NPCR y UACI. NPCR se calcula con la siguiente expresión

$$NPCR = \frac{\sum_{i=1}^{\ell} W(i)}{\ell} \times 100 \tag{8.11}$$

con

$$W(i) = \begin{cases} 0 & \text{if } E_1(i) = E_2(i) \\ 1 & \text{if } E_1(i) \neq E_2(i) \end{cases} \tag{8.12}$$

y el valor de UACI se determina con

$$UACI = \frac{100}{\ell \times 95} \sum_{i=1}^{\ell} |E_1(i) - E_2(i)| \tag{8.13}$$

donde ℓ es la longitud del texto, E_1 and E_2 son los dos criptogramas. El proceso para determinar los valores es como sigue: primero, el texto claro se cifra con la CLAVE 1 para generar E_1; después, el primer símbolo del texto claro se cambia de **A** a **B**, y el proceso de cifrado se repite con la misma CLAVE 1 para generar E_2. La tabla 8.2 muestra los resultados de NPCR y UACI con E_1 and E_2 en tres distintas pruebas. Por tanto, el esquema propuesto es robusto ante ataques diferenciales, ya que el 99 % de los símbolos son diferentes con una diferencia de magnitud en promedio del 33 %.

Prueba	Prueba 1	Prueba 2	Prueba 3
NPCR(%)	99.15	98.94	99.36
UACI(%)	31.53	34.15	33.56

Tabla 8.2: Resultados de análisis diferencial NPCR y UACI para cifrado de texto.

8.3.4. Frecuencia flotante

La *frecuencia flotante* determina el número de símbolos diferentes en una sección (ventana) del mensaje. Esta información puede utilizase para determinar si el texto cifrado es cifrado con propiedades de *uniformidad*, es decir, si el texto cifrado se divide en varias secciones, cada sección debe tener todos los posibles símbolos de forma uniforme. Para este análisis, se utiliza una ventana de 95 elementos que son el total de elementos que acepta el sistema criptográfico: inicialmente, se seleccionan los primeros 95 elementos del texto y después, la ventana se recorre una posición a la derecha y la prueba se repite a estos otros 95 elementos; así sucesivamente hasta terminar todo el mensaje. La figura 8.6(a) y (b) muestra la frecuencia flotante del texto claro y del texto cifrado, respectivamente; la frecuencia flotante del texto cifrado es uniforme, por tanto, el cifrado propuesto no tiene secciones débiles. Además, el texto cifrado tiene hasta 60 % de todos los posibles elementos contra 23.5 % del texto claro.

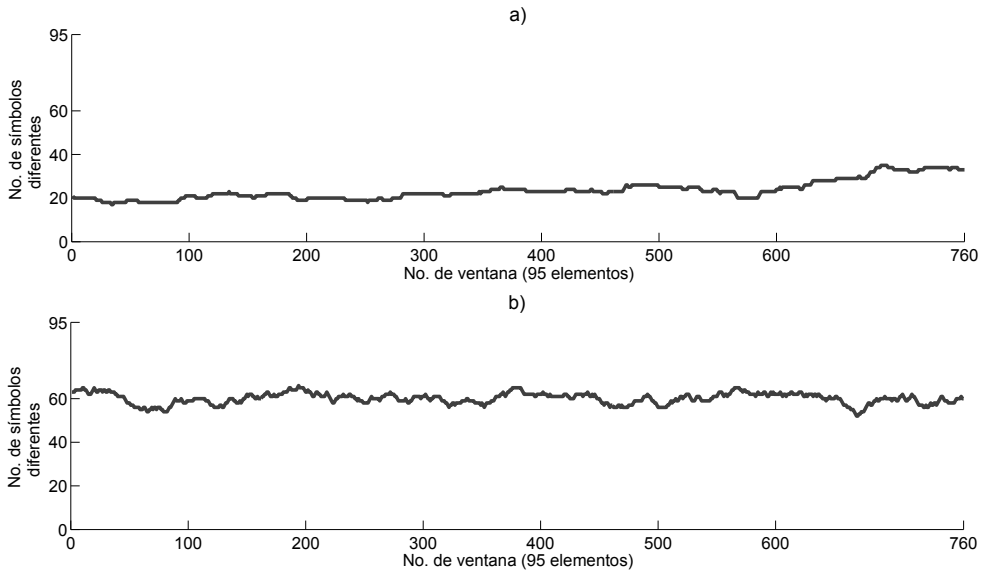


Figura 8.6: Análisis de frecuencia flotante de cifrado de texto: a) texto claro y b) texto cifrado.

8.3.5. Histogramas

De acuerdo con la Sec. 7.3.5, el histograma del texto cifrado debe ser uniforme para resistir un ataque estadístico en el lenguaje que fue cifrado originalmente. El histograma del texto claro se muestra en la figura 8.7(a) y se puede apreciar la característica estadística de la fuente; sin embargo, el histograma del texto cifrado es uniforme, como se aprecia en la figura 8.7(b). Por tanto, el algoritmo criptográfico propuesto es robusto ante un ataque de histograma y ataque de frecuencia de letras.

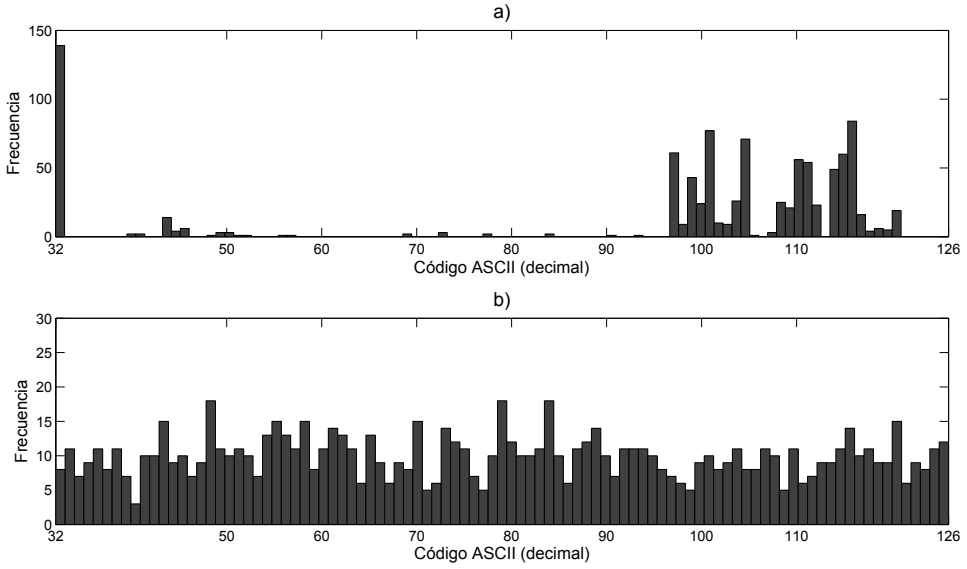


Figura 8.7: Histogramas de texto: a) texto claro y b) texto cifrado.

8.3.6. N-gramas

El N-grama es una subsecuencia continua de N símbolos de un mensaje y es utilizado para predecir el siguiente símbolo después de observar el símbolo N-1 mediante métodos de probabilidad. Para este análisis, se utilizan bigramas (N=2) y trigramas (N=3) para determinar la frecuencia de 2 y de 3 símbolos continuos en el texto claro y texto cifrado. La tabla 8.3 muestra los primeros diez bigramas y trigramas, del texto claro y del texto cifrado. Como resultado, el texto cifrado tiene frecuencias muy bajas de bigramas y trigramas comparado con el texto claro que tiene altas frecuencias de bigramas y trigramas. Por tanto, el proceso de cifrado se considera robusto porque no genera altas frecuencias de bigramas y trigramas. Esta prueba fue desarrollada mediante el software CrypTool (notar que se incluye el espacio).

	Texto claro				Texto cifrado			
No.	Bigrama	Freq.	Trigrama	Freq.	Bigrama	Freq.	Trigram a	Freq.
1	a	32	de	13	9	2	}—8	2
2	e	24	de	11	![	2	¡	1
3	s	20	ion	9	%a	2	%A	1
4	de	17	se	8	&-	2	*W	1
5	d	16	cio	8	&E	2	,2	1
6	es	16	os	8	) %	2	3{	1
7	on	16	s d	8	.2	2	93	1
8	te	15	aci	7	/3	2	95	1
9	a	14	est	7	/H	2	;k	1
10	e	14	nte	7	2A	2	¡V	1

Tabla 8.3: Análisis de bigramas y trigramas de texto claro y cifrado.

8.3.7. Autocorrelación

En procesamiento de señales, la autocorrelación se define como la correlación de una señal consigo misma desplazada k posiciones y determina si la señal posee patrones repetitivos, periodicidad o alguna dependencia. Se utiliza este contexto para verificar si el proceso de cifrado genera patrones repetitivos o periodicidad. La autocorrelación se determina con

$$AC(k) = \frac{A - D}{T}, \tag{8.14}$$

donde $AC \in [1 - 1]$ es la autocorrelación del mensaje desplazado k posiciones, A es el número de elementos que concuerdan entre el mensaje original y el mensaje desplazado, D es el número de elementos que no concuerdan y T es la longitud del mensaje. Altos valores positivos de AC significa que muchos bits son idénticos y altos valores negativos de AC significa que muchos bits son opuestos; mientras que un valor de AC nulo significa que se tiene los mismos números de 1's y 0's (deseable).

En este análisis, se calcula la autocorrelación a nivel bit y se utilizan 7 bits para representar cada símbolo de acuerdo con la tabla ASCII. Primero, la autocorrelación del texto claro se calcula como sigue: los 945 caracteres se transforma a un renglón de 6615 bits: *1001001110*...; después, se determina la autocorrelación del mensaje con un valor de hasta $k = 500$ hacia la derecha. La figura 8.8(a) y (b), muestran la autocorrelación del texto claro y del texto cifrado, respectivamente. La autocorrelación del texto claro tiene patrones repetitivos con con altos valores positivos y altos valores negativos de AC, mientras que el texto cifrado tiene una autocorrelación cercana a 0, es decir, el proceso de cifrado genera números pseudoaleatorios uniformemente.

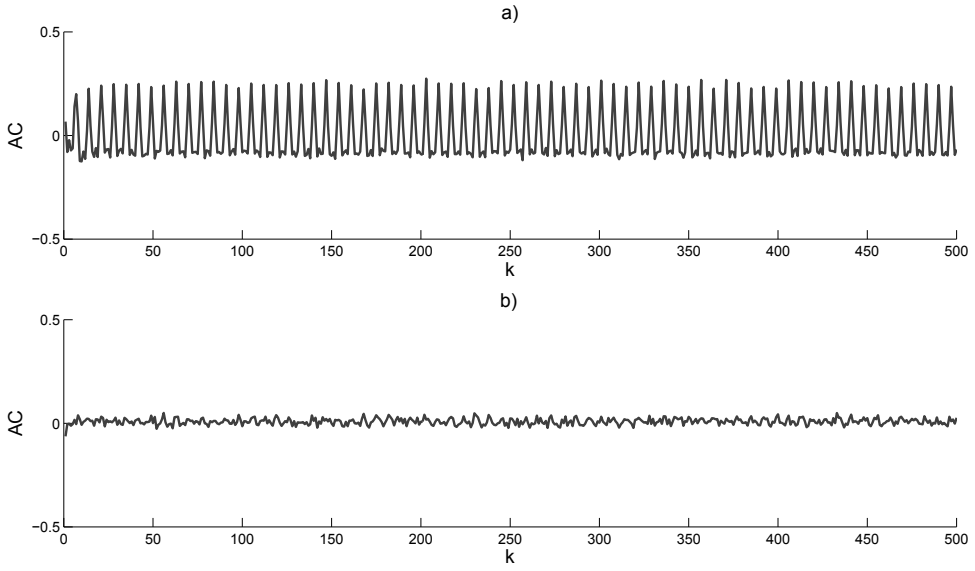


Figura 8.8: Autocorrelación de texto alfanumérico: a) texto claro y b) texto cifrado.

8.3.8. Ataque de sólo texto claro elegido y conocido

Las consideraciones que se tomaron para diseñar el algoritmo de cifrado caótico propuesto y presentadas en la Sec. 7.3.7, como procesos de confusión y difusión en una sola operación, secuencias caóticas determinadas tanto de la clave secreta como del texto claro y una mejor distribución del mapa logístico, todas ellas reducen el éxito de los siguientes ataques:

1. **Ataque de sólo texto claro elegido**

   El adversario (o intruso) elige un trozo de texto claro; en este caso, el adversario puede elegir un texto que consiste de caracteres específicos para estudiar el comportamiento del algoritmo de cifrado y encontrar la clave secreta.

2. **Ataque de sólo texto claro conocido**

El adversario conoce al menos un trozo de texto claro y su correspondiente texto cifrado, si el algoritmo de cifrado utiliza operaciones de XOR, el texto claro XOR con el texto cifrado podría revelar la clave secreta utilizada.

Debido a que las secuencias caóticas para los procesos de confusión y de difusión, se determinan de la clave secreta y de las características totales del texto claro, cada vez que un texto claro es cifrado, diferentes secuencias caóticas son generadas, incluso si se utiliza la misma clave secreta. De tal forma que, el cifrado es robusto ante estos poderosos ataques que han quebrantado múltiples sistemas criptográficos basados en caos para imágenes.

8.3.9. Entropía de la información

De acuerdo con la Sec. 7.3.8, la entropía determina que tan impredecible es cierto mensaje. A mayor entropía, más impredecible es un mensaje. El cifrado de texto utiliza 95 símbolos diferentes con una entropía máxima de $H = 6.56$. La entropía del texto claro es de $H(P) = 4.35$, mientras que el texto cifrado tiene $H(E) = 6.52$. Por tanto, el cifrado propuesto genera todos los símbolos con aproximadamente la misma probabilidad para generar alto desorden en el texto cifrado.

8.3.10. Recursos de implementación y desempeño

En cualquier implementación digital es importante determinar los recursos de implementación empleados, como memoria requerida del programa, arquitectura del microcontrolador, módulos de comunicación, etc., que estan relacionados con el costo de la aplicación. La tabla 8.4 muestra los recursos utilizados en la implementación en microcontrolador del algoritmo de cifrado propuesto en la tesis y se muestra que el software del cifrado requiere de poco espacio de memoria, requiere de bajo voltaje, es portátil, de bajo costo y de dimensiones pequeñas.

	Usada/Total
Memoria Flash (KB)	115/512 (22.5 %)
Puertos comunicación	USB 2.0
Complemento 1	Freescale MQX 3.5 (MQX + USB host)
Frecuencia (MHz)	80/80 (100 %)
Voltaje (Vdc)	5
Corriente (mA)	118
Costo M52259DEMOKIT (USD)	115
Dimensiones M52259DEMOKIT (in)	2×2

Tabla 8.4: Recursos de la implementación de cifrado de texto en microcontrolador.

El desempeño basado en programación en lenguaje C y una frecuencia de 80 MHz del microcontrolador de 32 bits, el *cifrado* de 945 elementos requiere de 0.04 segundos; por tanto, la implementación puede utilizarse en sistemas embebidos en tiempo real

para proteger datos sensibles. El tiempo de *descifrado* es similar al tiempo de *cifrado*, incluso un poco menos ya que no se requiere de 1,000 iteraciones del mapa logístico 2.

8.4. Conclusiones

En este capítulo, se presentó la implementación en microcontrolador de 32 bits de un algoritmo de cifrado de texto alfanumérico. El algoritmo requiere de sólo una ronda de confusión y difusión, una clave secreta de 128 bits y dos mapas logísticos con secuencias caóticas optimizadas. Una alta seguridad (lógica) del algoritmo de cifrado fue probada mediante simulaciones en MatLab con datos extraídos directamente del microcontrolador (texto claro y texto cifrado). Los recursos de implementación son bajos con un buen tiempo de cifrado, por lo que el esquema propuesto puede ser aplicado en sistemas embebidos en áreas como la militar, biométrica, industrial, climáticas, comercio electrónico, telemedicina, datos personales, entre otros.

Capítulo 9

Cifrado caótico de plantilla de huella dactilar en microcontrolador

Desde mucho tiempo atrás, la huella dactilar ha sido utilizada para identificar o autentificar a una persona, debido a propiedades como singularidad, inalterabilidad, universalidad, accesibilidad y aceptabilidad que posee como identificador biométrico. Hoy en día, los sistemas de autentificación se usan en control de accesos físicos en bancos, oficinas, fábricas, etc. y en accesos lógicos como comercio electrónico, celulares, computadoras, etc. Sin embargo, los identificadores biométricos no se encuentran protegidos en estos sistemas de control de acceso y de identificación y se corre el riesgo de robo de identidad. Una solución es el cifrado de plantilla biométrica para proporcionar seguridad a la identidad del usuario cuando se utiliza un sistema de autentificación en sistemas embebidos.

En este capítulo, se presenta un sistema de autentificación basado en la huella dactilar y su cifrado caótico para incrementar la seguridad del sistema embebido y evitar robo de identidad; el proceso de cifrado se basa en el algoritmo criptográfico propuesto en este trabajo doctoral, con mínimas adecuaciones. Se reporta un análisis completo de seguridad a nivel lógico como espacio de la clave, sensibilidad a la clave, sensibilidad a la plantilla biométrica, correlación, entropía y análisis de aleatoriedad con FIPS-140-2. Además, se presenta los recursos de implementación y el desempeño del cifrado propuesto. Basado en los resultados, el sistema de autentificación embebido propuesto es seguro, efectivo y de bajo costo para ser implementado en sistemas de control de accesos.

9.1. Introducción

Una persona tiene características fisiológicas y conductuales únicas que la distingue de cualquier otra persona en el mundo, estas características se conocen como *identificadores biométricos*, por ejemplo las características del rostro, iris, palma de la mano, huella dactilar, geometría de la mano, voz, forma de teclear, entre otros, han sido utilizados para identificar personas porque son únicos e inalterables y además, se pueden cuantificar. Debido a estas ventajas, actualmente la identificación de personas de forma

biométrica se esta haciendo popular, comparado con técnicas de identificación tradicional como tarjetas de identificación (ID), claves, número de identificación personal (NIP), tarjeta inteligente, etc. En 1926, EUA empezó a utilizar la huella dactilar para identificar criminales con métodos manuales; desde entonces, métodos automáticos para identificación por medio de huella dactilar son desarrollados con mejor precisión. Actualmente, la huella dactilar es la característica biométrica más utilizada para identificar personas ya que es práctica, es la más estudiada, es la más desarrollada y sus costos de implementación son bajos. Otros identificadores biométricos comúnmente utilizados en estos días son la palma de la mano, iris, rostro y voz, ver por ejemplo [48, 126].

El proceso de *identificación* se basa en dos etapas: *enrolamiento* y *verificación*. En la etapa de *enrolamiento*, una o más muestras se toman del identificador biométrico mediante un método (por ejemplo extracción de minucias) para generar una plantilla (usualmente de varios KB) que se almacena local o remotamente en una base de datos para futuras comparaciones. La etapa de *verificación*, una o más muestras son tomadas del identificador biométrico para generar una plantilla que se compara con una plantilla previamente almacenada en la base de datos para verificar a un usuario. Además, un sistema biométrico tiene dos objetivos: identificación y autentificación. La identificación biométrica se utiliza para identificar a una persona, por ejemplo en el forense, a un criminal, a un paciente, etc., donde la plantilla se compara con todas las registradas en la base de datos; mientras que, autentificación biométrica implementa comparaciones de plantilla uno a uno, por ejemplo en sistemas de acceso seguro a nivel físico y lógico, como en oficinas, bancos, industria, hospitales, universidades, centros de investigación, comercio electrónico, computadoras, celulares, entre otros, ver por ejemplo [127].

Sin embargo, muchos sistemas de identificación biométrica comerciales, tienen problemas de seguridad como robo de identidad mediante ataque a plantillas almacenadas en la base de datos o ataques en las lineas de comunicación ya que no cifran la información biométrica de los usuarios [128]. Recientemente, *biométria cancelable* y *baúl difuso* (Fuzzy vault) fueron propuestos para proteger la información biométrica mediante el uso de la criptografía. El método de biométrico cancelable consiste en distorsionar los datos con una función de distorsión de un sentido (one-way) [129, 130], pero las principales desventajas es que la seguridad del sistema es difícil de verificar debido que no se tienen fundamentos matemáticos [131]. Mientras que el método de *baúl difuso* es un esquema criptográfico que une la plantilla biométrica con una clave aleatoria uniforme, para construir una plantilla al azar que se utiliza en vez de la plantilla original. Sin embargo, este método presenta muchas vulnerabilidades cuando el *baúl difuso* se implementa en sistemas de autentificación [132].

Esteganografía, marca de agua y criptografía son otros métodos utilizados para resolver el problema de seguridad; esteganografía es una técnica, el la cual, el mensaje se oculta en otro mensaje para que pase desapercibido, por ejemplo ocultar texto en una imagen clara. La técnica de marca de agua se utiliza para protección de derechos de autor.

Por otra parte, criptografía es una técnica ámpliamente utilizada en esquemas de comunicación segura y almacenamiento de datos privados, donde básicamente los datos claros son convertidos en datos cifrados por un algoritmo y una clave secreta. En capítulos anteriores 7 y 8, se mostró que los sistemas caóticos son muy recomendados para utilizarse en la criptografía.

Los sistemas de autentificación biométrica se implementan usualmente en sistemas embebidos. En la literatura, se reportan algunos esquemas de autentificación basados en identificadores biométricos: en [133] se presentó un método de autentificación por huella dactilar que es rápido y preciso; en el cual, la imagen de la huella se captura y se aplica cancelación de ruido con un filtro pasa baja. Después, el proceso de verificación se lleva a cabo con detección de desplazamiento y características circulares de la huella y finalmente, el método del autor se implementa en un microcontrolador de 8 bits; aunque los resultados experimentales muestran el alto desempeño y eficiencia del método propuesto, los datos biométricos no son protegidos de ninguna forma, por lo que el templete del usuario esta expuesto y corre riesgo de robo de identidad. En [134] se propuso un sistema novedoso de autentificación biométrico, donde la huella dactilar y la voz se utilizan en etapas separadas; cada etapa se implementa en un microcontrolador para verificar la identidad del usuario, pero los datos biométricos no se protegen de ninguna manera. En [135] se propuso un sistema de autentificación basado en puntos singulares, la comparación de huella utiliza pocas características en el proceso de verificación, por tanto, la tasa de reconocimiento es mayor que el método por minucias. Además, el esquema se implementa en tecnología digital FPGA (*Arreglo de Compuertas Lógicas Programables*) con excelentes resultados como alta velocidad de reconocimiento. Otro esquema de autentificación implementado en FPGA fue presentado en [136], donde la propiedad intelectual se protege con la huella dactilar. Sin embargo, el cifrado de datos biométricos no es considerado en ninguna implementación mencionada anteriormente. También, hay otras imeplementaciones donde se utiliza combinación de microcontrolador y FPGA, para sistemas de autentificación con huella dactilar reportados en [137, 138]; también los datos biométricos no son protegidos y la identidad del usuario sigue expuesta [139].

En este capítulo, se presenta un novedoso sistema de autentificación por huella dactilar, implementado en un microcontrolador de 32 bits, con protección de plantilla dactilar con su cifrado mediante sistemas caóticos. Se utiliza el algoritmo de cifrado propuesto en este trabajo doctoral (Sec. 6) para proteger la identidad del usuario. En la etapa de *enrolamiento*, la plantilla del usuario se lee por el módulo MAHD que funciona como esclavo y los datos se envían al microcontrolador vía RS-232; una vez que la plantilla esta en el microcontrolador, esta se cifra y almacena en memoria Flash. En el proceso de *autentificación*, el usuario solicita su plantilla cifrada al microcontrolador y la plantilla original se recupera mediante el descifrado con la misma clave secreta de 32 caracteres hexadecimales. Después, el microcontrolador envía la plantilla descifrada al módulo lector de huella y se lleva a cabo el proceso de comparación de plantillas con la plantilla actual del usuario para autentificarlo. Se realiza un análisis muy completo de seguridad del cifrado como en el capítulo 8, pero en este caso, se incluye un

importante análisis de aleatoriedad con FIPS-140-2 para verificar la calidad del cifrado y determinar si es una fuente meramente aleatoria. También, se presentan los recursos de implementación para validar su uso en sistemas embebidos, así como el desempeño donde se muestra que puede ser utilizado en aplicaciones en tiempo real.

9.2. Sistema de autentificación propuesto

El proceso de autentificación biométrica tradicional se basa en la lectura del identificador biométrico mediante un sensor, extracción de características del biométrico, procesamiento de plantillas biométricas y comparación con plantillas previamente almacenadas en la base de datos. Inicialmente, en el proceso de *enrolamiento* se determina un biométrico físico o de comportamiento como iris, rostro, huella dactilar, palma de la mano, voz, firma, etc., para utilizarse como identificador; después, un sensor realiza la adquisición de datos y un algoritmo los procesa para extraer características biométricas (plantillas); finalmente, la plantilla se almacena en una memoria local o remota para futuras comparaciones. Una vez que el *enrolamiento* se ha concluido con éxito, se puede realizar el proceso de *verificación* para autentificar al usuario con los siguientes pasos: leer el biométrico, extraer características y comparar *uno a uno* con la plantilla previamente almacenada (ver figura 9.1) [140].

En la figura 9.2, se muestra el esquema de protección de plantilla biométrica implementada en este trabajo. Antes del almacenamiento de plantillas en el proceso de *enrolamiento*, la plantilla se cifra mediante el algoritmo de cifrado caótico propuesto en este trabajo doctoral (ver capítulo 6); una vez que se ha cifrado la información, ésta se puede transmitir por un canal inseguro y almacenar local o remotamente. En el proceso de autentificación, la plantilla se descifra y compara con una plantilla reciente del mismo biométrico del usuario para validar su identificación.

La implementación del sistema embebido propuesto está basado en tres módulos de hardware: microcontrolador de 32 bits M52259, módulo de autentificación de huella dactilar (MAHD) Futronic FS83 [141] y dispositivos de interfaz humana como LCD y teclado (ver figura 9.3). El proceso de enrolamiento puede registrar un nuevo usuario autorizado en el sistema embebido con una huella dactilar (ejemplo, la huella dactilar del dedo índice derecho); primero, MAHD utiliza el método de extracción de minucias para generar una plantilla del usuario (2,072 bytes de tres muestras) y la envía al microcontrolador vía RS-232; después, la plantilla se cifra mediante el algoritmo basado en caos y finalmente, la plantilla cifrada se almacena en la memoria flash del microcontrolador, ver figura 9.4. Una vez que el usuario autorizado fue registrado en el sistema, este puede acceder al área restringida física o lógicamente mediante el proceso de autentificación: primero, el usuario autorizado solicita su plantilla cifrada al sistema embebido; después, el microcontrolador envía la plantilla descifrada al MAHD y finalmente, el proceso de autentificación se lleva a cabo en el MAHD. Si las plantillas son muy similares (lo determina el algoritmo de MAHD), el sistema permite el acceso del usuario al área restringida, ver figura 9.5.

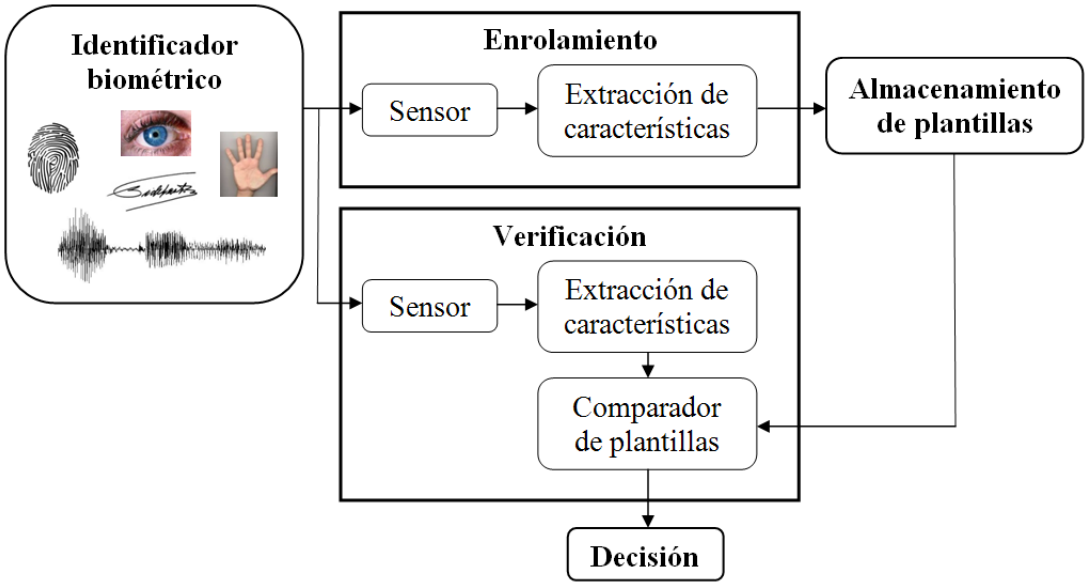


Figura 9.1: Esquema de autentificación biométrica tradicional [140].

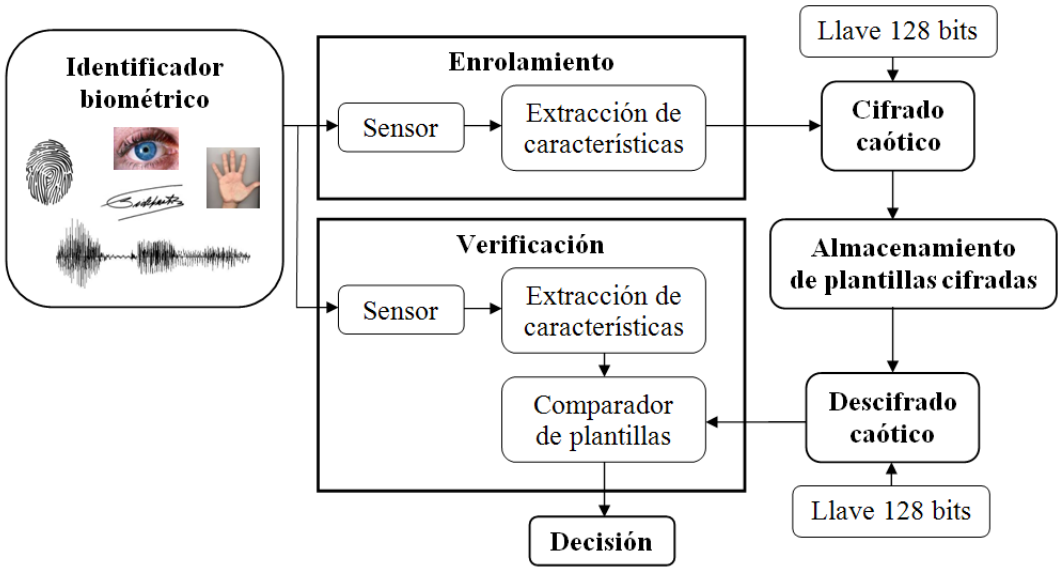


Figura 9.2: Esquema de autentificación biométrica con protección de plantilla empleando caos propuesto en esta tesis.

El escaner del MAHD esta basado en LEDs infrarrojos y un procesador con algoritmo de reconocimiento de huella dactilar basado en el método de extracción de minucias con una tasa de falsa aceptación de 10^6 (una en un millón) y taza de falso rechazo de 10^2 (una de cada cien). Además, MAHD realiza el proceso de comparación y otros. La unidad central del sistema es el microcontrolador, el cual, es responsable de realizar tareas como: leer la plantilla dactilar del usuario, cifrar la plantilla, descifrar la plantilla, almacenar plantilla en memoria flash, solicitar el proceso de comparación para autentificar a un usuario, leer el teclado, enviar datos a una pantalla LCD y controlar señales de salida para el control de acceso físico o lógico. La interfaz humana esta basada en una pantalla LCD, teclado, botones de presionar, entre otros que permiten desempeñar los procesos de enrolamiento y autentificación.

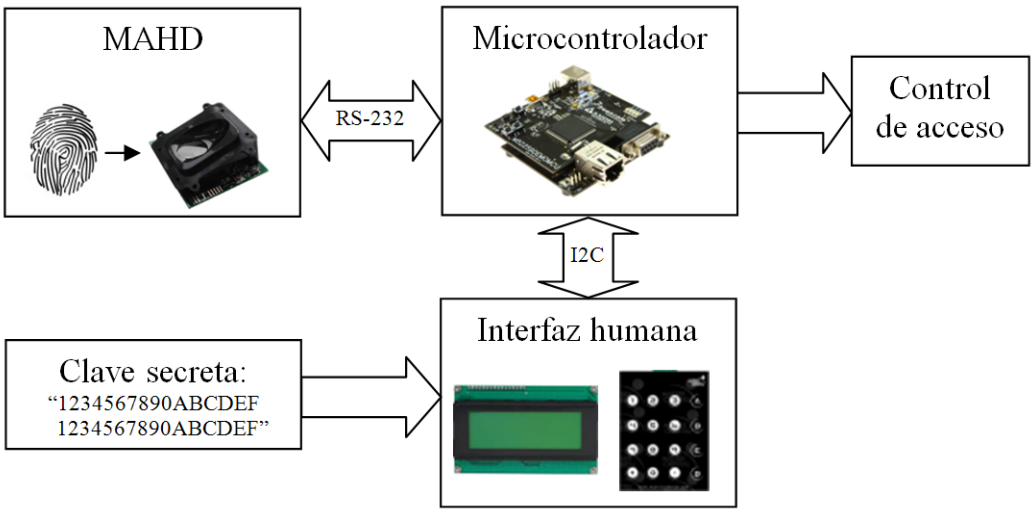


Figura 9.3: Esquema del sistema embebido de autentificación por huella dactilar seguro y propuesto en esta tesis.

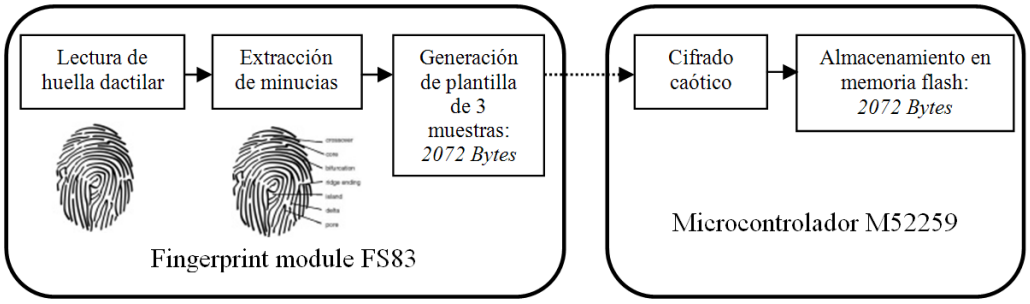


Figura 9.4: Esquema del proceso de enrolamiento del sistema embebido propuesto en esta tesis.

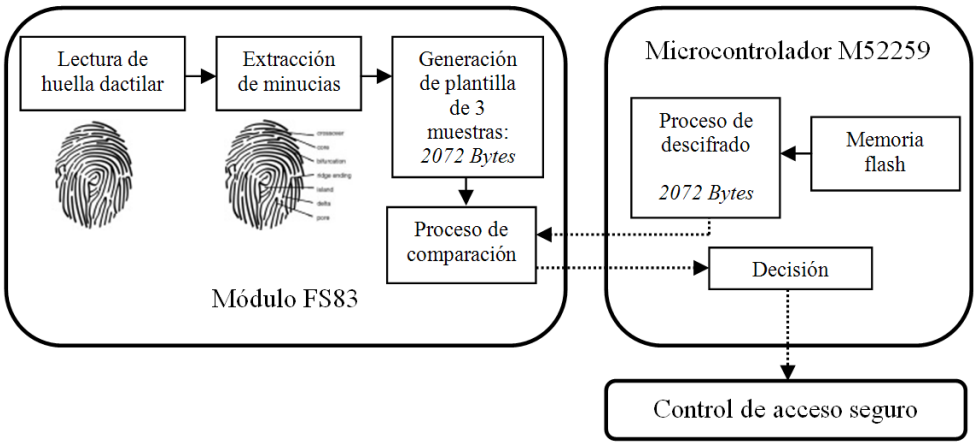


Figura 9.5: Esquema del proceso de autentificación del sistema embebido propuesto en esta tesis.

9.3. Cifrado

El proceso de cifrado es similar al presentado en el capítulo anterior, ver sección 8.2. Se asume un texto claro $P \in [0, 255]$ de longitud $\ell = 2072$ basado en datos leídos del módulo MAHD que están representados por 8 bits (0-255 en decimal). La figura 9.6 muestra el diagrama a bloques del proceso de cifrado. Se tienen 2,072 bytes del módulo FS83 como plantilla clara cada vez que se registra un nuevo usuario al sistema de autentificación o solicita que se autentifique su identidad.

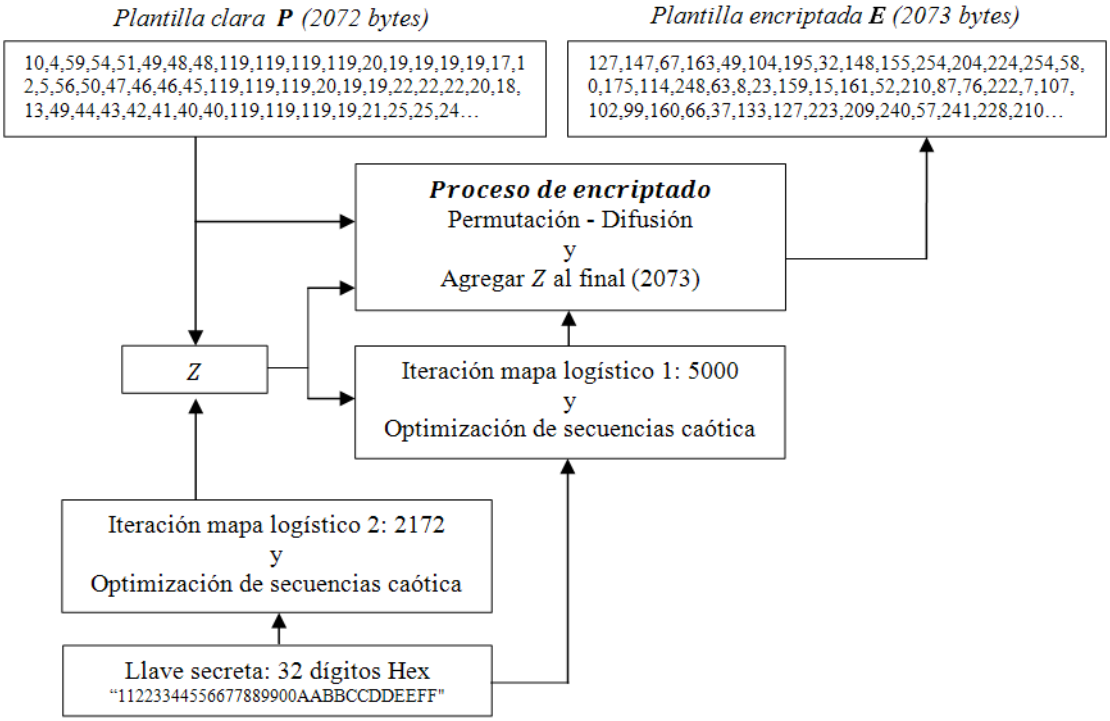


Figura 9.6: Esquema de cifrado de plantilla de huella dactilar.

Primero, el valor de Z se determina de la siguiente manera:

El mapa logístico 2 se itera $I_2 = 2,172$ con valores de a_2 y x_{2_0} tomados de la tabla 6.1 para generar la secuencia caótica de datos x^{L2} con $x^{L2} \in (0,1)$ y precisión 10^{-15}. x^{L2} se optimiza de la siguiente manera

$$x_I^{L2} = \left\{ \left[x_{I_2}^{L2} \right] * 1000 \right\} \quad (\text{mód } 1), \quad para\ I_2 = 1, 2, 3, \ldots, 2172 \tag{9.1}$$

Después, todos los elementos de la plantilla clara se suman con la secuencia caótica x^{L2} como sigue

$$S = \left\{ S + [P_i * x_{2173-i}^{L2}] + x_{2173-i}^{L2} \right\} \quad (\text{mód } 1), \quad para\ i = 1, 2, 3, \ldots, 2072 \tag{9.2}$$

donde P_i representa el elemento i de la plantilla clara, S es una variable inicializada en cero y *mód* es la operación de módulo. El valor de Z debe ser un valor entero entre 1-255 para evitar la cancelación de Z, por tanto se calcula lo siguiente

$$V = \ round(S * 254) + 1, \tag{9.3}$$

donde $V \in [1, 255]$ y *round* es la operación de redondeo al valor más cercano. Finalmente, el valor de Z está dado por

$$Z = V/256, \tag{9.4}$$

donde $V \in (0,1)$ con precisión de 10^{-15}.

Segundo, el proceso de cifrado consiste de las siguientes operaciones:

El mapa logístico 1 se itera $I_1 = 5,000$ con valores de a_1 y x_{1_0} tomados de la tabla 6.1 para generar la secuencia caótica de datos x^{L1} con $x^{L1} \in (0,1)$ y una precisión de 10^{-15}. De esta secuencia caótica se determinan dos subsecuencias, una para el proceso de confusión y otra para difusión como sigue

$$Q_i = \ round\left(x_{2928+i}^{L1} * 2071 \right) + 1, \quad para\ i = 1, 2, 3, \ldots, 2072 \tag{9.5}$$

donde $Q_i \in [1, 2072]$ es el vector de confusión pseudoaleatorio. Se considera que el vector Q_i esta optimizado y contiene todas las posiciones de forma pseudoaleatoria, ver sección 6.5.

La segunda subsecuencia para el proceso de difusión se determina como sigue

$$F_i = \left\{ \left(x_{2928+i}^{L1} * 1000 \right) + Z \right\} \quad (\text{mód } 1), \quad para\ i = 1, 2, 3, \ldots, 2,072 \tag{9.6}$$

donde $F_i \in (0,1)$ es un vector de longitud de 2072 elementos con precisión de 10^{-15}. Después, F_i se transforma de $(0,1)$ a $[0, 255]$ como sigue

$$Y_i = \ round\left(M_i * 255 \right), \quad para\ i = 1, 2, 3, \ldots, 2072 \tag{9.7}$$

donde $Y_i \in [0, 255]$ es un vector de longitud 2072. Finalmente, el proceso de cifrado se calcula con

$$E_i = (P(Q_i) + Y_i) \ \ mod\ 256, \quad donde\ i = 1, 2, 3, \ldots, 2072 \tag{9.8}$$

donde $E_i \in [0, 255]$ es la plantilla cifrada y P es la plantilla clara.

Tercer paso:

El valor de Z se agrega al criptograma en la posición 2,073 como sigue

$$E_{2073} = V. \tag{9.9}$$

El **proceso de descifrado** consiste básicamente de invertir los pasos del cifrado y se considera que el criptograma no fue modificado durante su transmisión y almacenamiento. Primero, Z se recupera del criptograma como sigue

$$Z = E_{2073}/256. \tag{9.10}$$

Después, x^{L1} se genera con la misma clave secreta utilizada en el cifrado; posteriormente, se determinan Q_i y Y_i de la misma forma que el proceso de cifrado. Finalmente, la plantilla descifrada se calcula con la siguiente expresión

$$D(Q_i) = (E_i - Y_i) \ \ mod\ 256, \quad para\ i = 1, 2, 3, \ldots, 2072 \tag{9.11}$$

donde Q_i es el vector de confusión, E_i es el criptograma, Y_i es el vector de difusión y D es la plantilla recuperada.

9.4. Análisis de seguridad

Los detalles de programación e implementación en microcontrolador se presentaron en la sección 8.3. Para los análisis de seguridad, en este caso, también se extrae del microcontrolador la plantilla clara y su respectiva plantilla cifrada, para realizar los distintos análisis de seguridad en MatLab. En las siguientes secciones, se presentan los diferentes análisis de seguridad a los que se somete el cifrado de plantilla dactilar con el algoritmo propuesto implementado en microcontrolador para un sistema embebido de autentificación biométrica por huella dactilar.

9.4.1. Espacio de clave secreta

De acuerdo con la sección 7.3.2, el espacio de claves es de 2^{128} posibilidades, por lo que puede resistir un ataque exhaustivo.

9.4.2. Sensibilidad a clave secreta

En los capítulos 7 y 8, se mencionó la necesidad de que el algoritmo de cifrado responda de manera drástica a pequeños cambios en la clave secreta. En esta sección, la sensibilidad a la clave secreta en el proceso de cifrado y descifrado, se prueba y verifica, con tres claves secretas similares, las cuales, se pueden consultar en la tabla 9.1. Para realizar este análisis, se propone una plantilla clara de 80 elementos fijos en 150. La figura 9.7 muestra la sensibilidad de la clave secreta al proceso de cifrado con el uso de las claves de la tabla 9.1; se aprecia que cada criptograma tiene su propia ruta, por tanto el algoritmo de cifrado es altamente sensible a la clave secreta. En la figura 9.8 se muestra la sensibilidad a la clave secreta en el proceso de descifrado donde sólo la clave correcta puede recuperar el mensaje original de 50 elementos fijos en 100.

No. clave	Clave secreta
Clave 1	11223344556677889900AABBCCDDEEFF
Clave 2	1122334**5**556677889900AABBCCDDEEFF
Clave 3	11223344556677889900AAB**C**CCDDEEFF

Tabla 9.1: Claves secretas utilizadas para análisis de sensibilidad a la clave en cifrado de plantilla de huella dactilar.

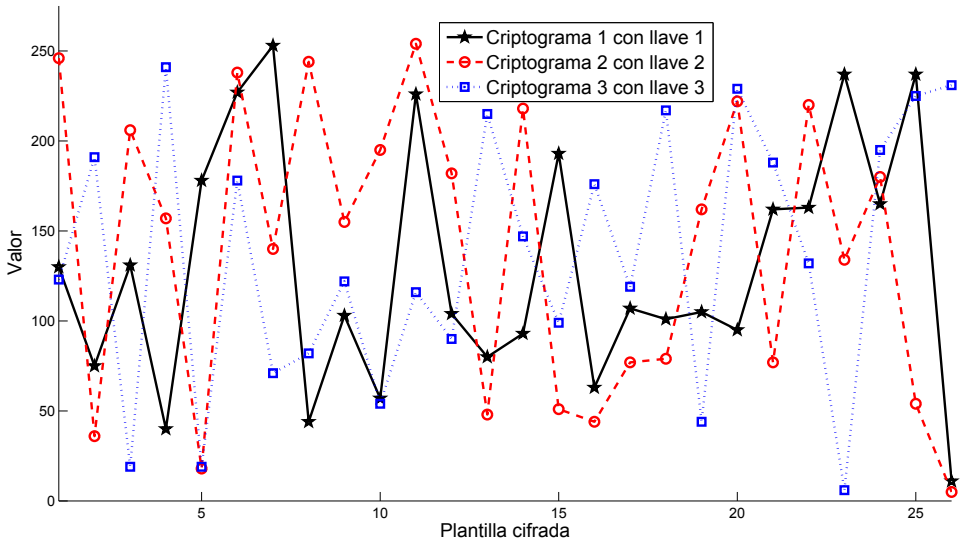


Figura 9.7: Sensibilidad a la clave secreta en proceso de cifrado de plantilla.

9.4.3. Sensibilidad a texto claro

En los capítulos 7 y 8, se menciona la necesidad de que el algoritmo de cifrado responda de manera drástica a pequeños cambios en la plantilla clara. Se utilizan dos

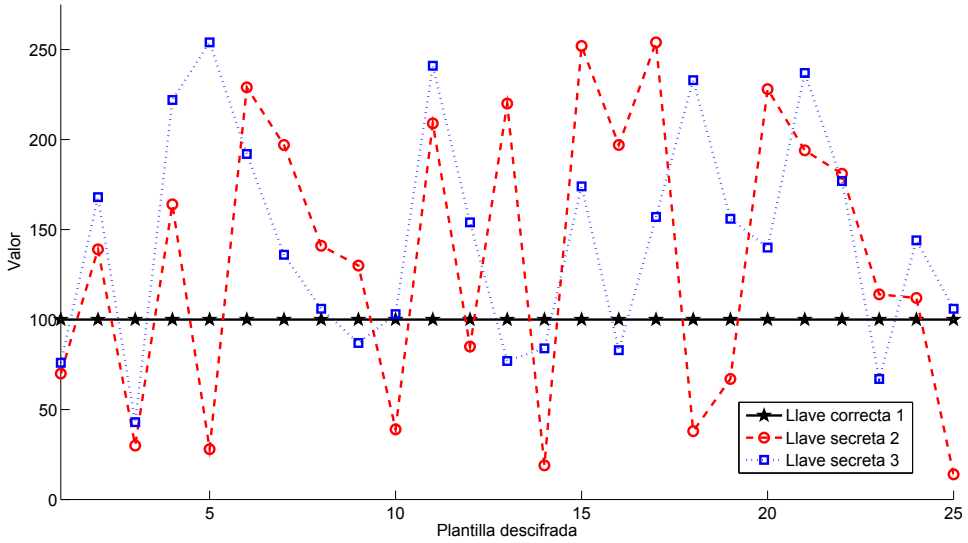


Figura 9.8: Sensibilidad a la clave secreta en proceso de descifrado de plantilla.

medidas para determinar esta sensibilidad: *NPCR* y *UACI* (Sec. 8.3.3). El valor de *NPCR* se determina con

$$NPCR = \frac{\sum_{i=1}^{i=2073} W(i)}{2073} \times 100 \tag{9.12}$$

donde

$$W(i) = \begin{cases} 0 & \text{if } E_1(i) = E_2(i) \\ 1 & \text{if } E_1(i) \neq E_2(i) \end{cases} \tag{9.13}$$

y el valor de *UACI* se determina con

$$UACI = \frac{100}{2073} \sum_{i=1}^{i=2073} |E_1(i) - E_2(i)| \tag{9.14}$$

donde ℓ es la longitud del texto, E_1 y E_2 son los dos criptogramas.

Para este análisis, se utilizan plantillas claras con un bit de diferencia entre ellos y se cifran con la misma clave secreta. Se tiene un valor de $NPCR = 99.5\,\%$ y $UACI = 88.4\,\%$ en promedio de 800 criptogramas; este resultado muestra la sensibilidad a la plantilla clara, ya que casi el 100 % de los elementos son diferentes comparando dos criptogramas E_1 y E_2 con una magnitud en promedio de 88 %. Además, los resultados son uniformes de acuerdo con la figura 9.9; el pico negativo que se muestra en el criptograma 288 se debe a que existe una pequeña probabilidad de obtener el mismo valor de Z, pero esto no es un problema de seguridad, ya que para descifrar correctamente se requiere de la clave secreta de 128 bits.

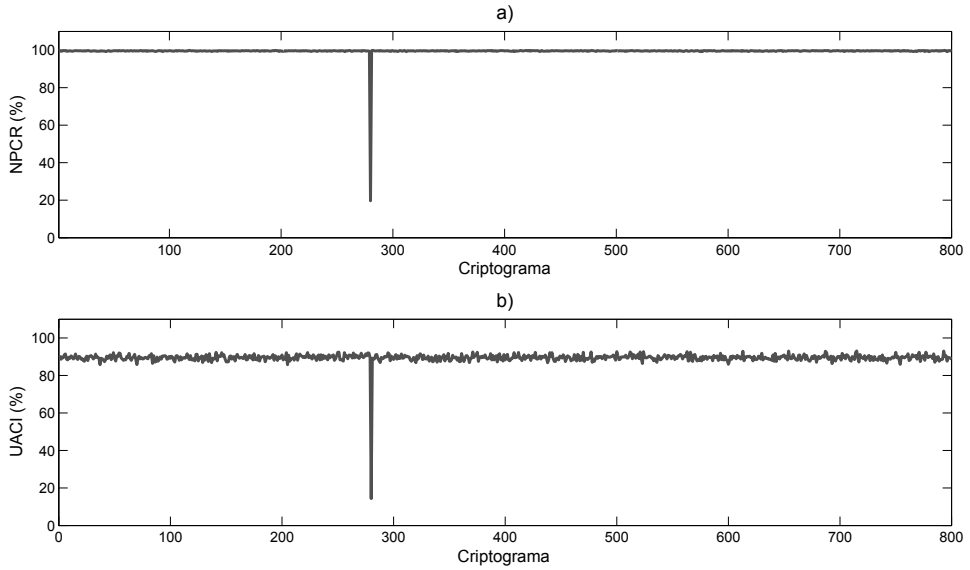


Figura 9.9: Resultados diferenciales de 800 criptogramas de huella dactilar: a) resultados de NPCR y b) resultados de UACI.

9.4.4. Frecuencia flotante

La *frecuencia flotante* determina cuantos símbolos son diferentes en una sección (ventana) del mensaje. Esta información puede utilizarse para determinar si la plantilla cifrada tiene propiedades de uniformidad, es decir, si la plantilla cifrada se divide en varias secciones, cada sección debe tener todos los posibles símbolos de forma uniforme. Para este análisis, se utiliza una ventana de 256 elementos que son el total de elementos que acepta el sistema criptográfico: inicialmente, se seleccionan los primeros 256 elementos del texto y después, la ventana se recorre una posición a la derecha y la prueba se repite a estos otros 256 elementos; así sucesivamente hasta terminar todo el mensaje. La figura 9.10(a) y (b) muestra la frecuencia flotante de la plantilla clara y de la plantilla cifrada, respectivamente; la frecuencia flotante de la plantilla cifrada es uniforme, por tanto, el cifrado propuesto no tiene secciones débiles. Además, el texto cifrado tiene hasta 62 % de todos los posibles elementos contra 24 % del texto claro.

9.4.5. Histogramas

De acuerdo con la sección 7.3.5, el *histograma* de la plantilla cifrada debe ser uniforme para resistir un ataque estadístico. El histograma de la plantilla clara se muestra en la figura 9.11(a) y se puede apreciar la característica estadística de la fuente; sin embargo, el histograma del texto cifrado es uniforme, como se aprecia en la figura 9.11(b). Por tanto, el esquema propuesto es robusto ante un ataque de histograma y ataque de frecuencia de letras.

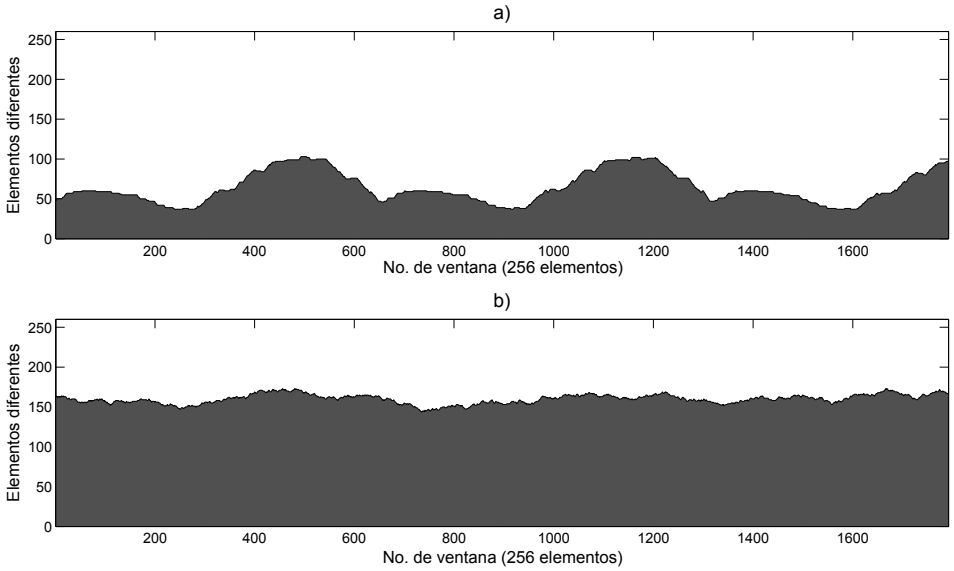


Figura 9.10: Análisis de frecuencia flotante de plantilla de huella dactilar: a) plantilla clara y b) plantilla cifrada.

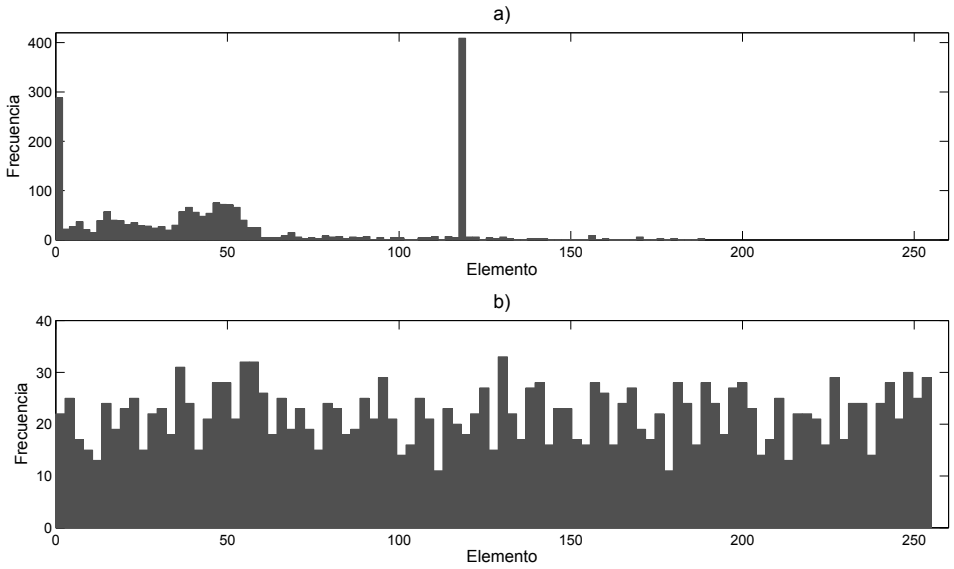


Figura 9.11: Histogramas de la plantilla: a) plantilla clara y b) plantilla cifrado.

9.4.6. Autocorrelación

De acuerdo con la sección 8.3.7, con la *autocorrelación* se puede determinar si la plantilla cifrada tiene patrones repetitivos, periodicidad o alguna dependencia. Esta se calcula con

$$AC(k) = \frac{A - D}{T}, \tag{9.15}$$

donde $AC \in [1 - 1]$ es la autocorrelación del mensaje desplazado k posiciones, A es el número de elementos que concuerdan entre el mensaje original y el mensaje desplazado, D es el número de elementos que no concuerdan y T es la longitud del mensaje. Grandes valores positivos de AC significa que muchos bits son idénticos y grandes valores negativos de AC significa que muchos bits son opuestos; mientras que un valor de AC nulo significa que se tiene el mismo número de 1's y 0's, situación deseable.

En este análisis, se calcula la autocorrelación a nivel bit y se utilizan 8 bits para representar cada elemento de la plantilla. Primero, la autocorrelación de la plantilla clara se calcula como sigue: los 2,072 elementos se transforman a un renglón de 16,576 bits: *1001001110*...; después, se determina la autocorrelación del mensaje con un valor de hasta $k = 500$ hacia la derecha. La figura 9.12(a) y (b), muestran la autocorrelación de la plantilla clara y de la plantilla cifrada, respectivamente. La autocorrelación de la plantilla clara tiene patrones repetitivos con altos valores positivos y altos valores negativos de AC; mientras que la plantilla cifrada tiene una autocorrelación cercana a 0, es decir, el proceso de cifrado genera números pseudoaleatorios uniformemente.

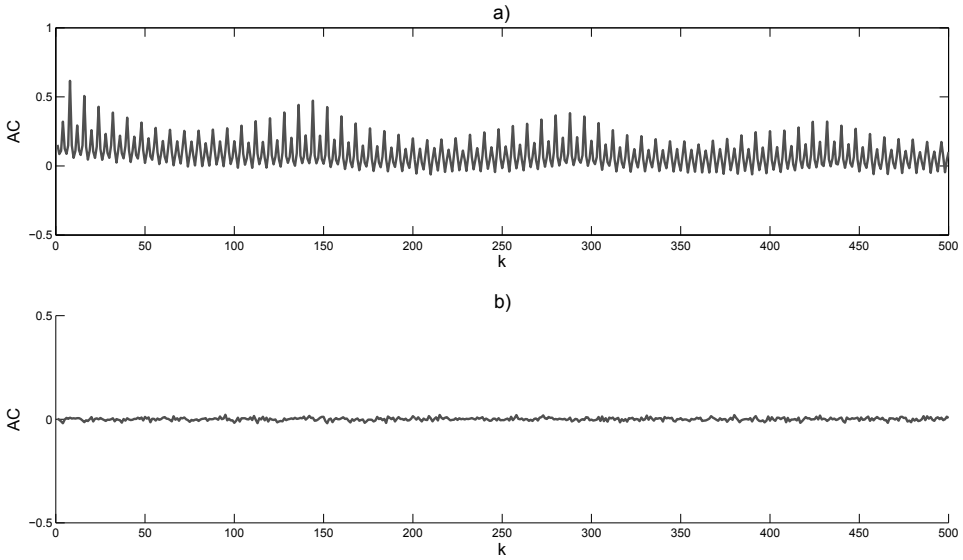


Figura 9.12: Autocorrelación de plantilla dactilar: a) plantilla clara y b) plantilla cifrada.

9.4.7. Ataque de sólo texto claro elegido y conocido

El criptoanalista puede atacar un sistema criptográfico para encontrar la clave secreta bajo el principio de Kerckhoffs: el algoritmo de cifrado se conoce (es de dominio público), en cambio la clave es secreta [69]. El ataque de plantilla clara elegida y ataque de plantilla clara conocida, se basan en la suposición de que una clave secreta se utiliza para generar varios criptogramas, por lo que el criptoanalista puede implementar este tipo de ataques sobre el algoritmo de cifrado, para determinar la clave secreta que se ha utilizado durante un tiempo y con ello descifrar otros criptogramas que tiene a su alcance. Un ataque de plantilla clara elegida y conocida es muy poderoso y puede quebrantar el sistema criptográfico, si no se hace una consideración importante en el proceso de cifrado caótico: las secuencias caóticas utilizadas para el cifrado deben ser diferentes para cada plantilla clara considerando el uso de la misma clave secreta.

En un esquema de cifrado donde se utiliza la arquitectura de confusión y difusión, si se elige una plantilla clara para cancelar el proceso de difusión (por ejemplo, una plantilla clara definida en ceros), el criptograma puede representar la secuencia caótica que se utilizó para cifrar (clave secreta) y puede ser utilizada por el criptoanalista para descifrar otros criptogramas que fueron previamente cifrados con esta clave secreta. El el proceso de cifrado de este trabajo, las secuencias caóticas para confusión y difusión, son determinadas de la clave secreta, de la plantilla clara y de datos caóticos del mapa logístico 2. Por tanto, en cada cifrado se generan secuencias caóticas diferentes, incluso con el uso de la misma clave secreta y con una plantilla clara definida en ceros; este proceso evita un ataque de plantilla clara elegida y conocida exitoso. En la figura 9.13 y figura 9.14, se muestra la sensibilidad de los vectores de confusión y difusión, respectivamente; se tienen tres plantillas claras de 25 elementos cada una que varían en un bit entre ellas (se considera que una esta definida en ceros) y son cifradas con la misma clave secreta. Notar que el vector de confusión tiene un rango de 0-25 ya que son las posiciones que se pueden representar, mientras que el vector de difusión se mantiene entre 0-255.

9.4.8. Entropía de la información

De acuerdo con la sección 7.3.8, la *entropía* determina qué tan impredecible es cierto mensaje. A mayor entropía, más impredecible es un mensaje. El cifrado de plantilla utiliza 256 símbolos diferentes con una entropía máxima de $H = 8$. La entropía de la plantilla clara es de $H(P) = 5.47$, mientras que la plantilla cifrada tiene $H(E) = 7.91$. Por tanto, el cifrado propuesto genera todos los símbolos con aproximadamente la misma probabilidad para generar gran desorden en la plantilla cifrada.

9.4.9. Análisis de aleatoriedad FIPS-140-2

El análisis de aleatoriedad esta basado en las siguientes pruebas de acuerdo con el estándar FIPS-140-2: *monobit*, *poker*, *run* y *serial* [79]. Los resultados de estas prue-

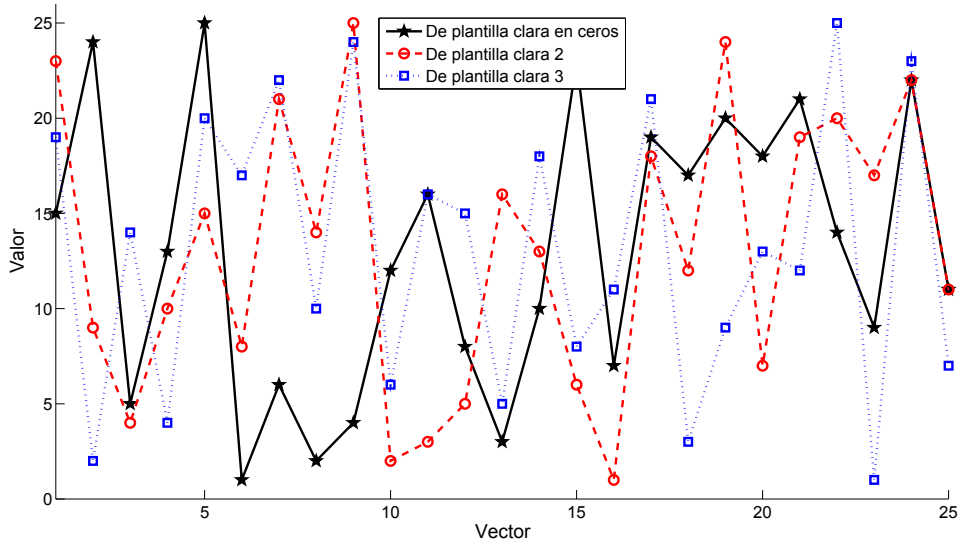


Figura 9.13: Sensibilidad del vector de confusión a clave secreta y plantilla clara.

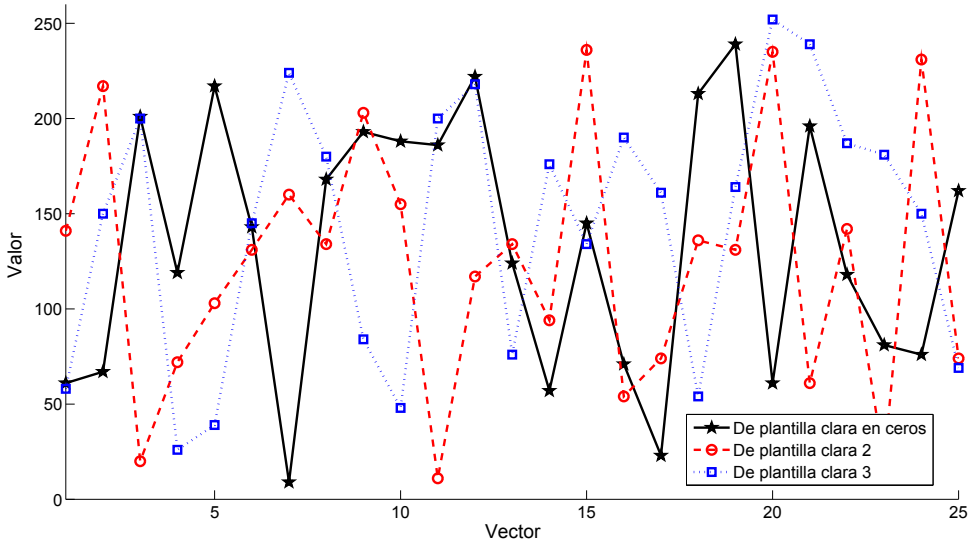


Figura 9.14: Sensibilidad del vector de difusión a clave secreta y plantilla clara.

bas están basados en la distribución Chi-cuadrada χ^2 con un nivel de significancia de $\alpha = 0.05$ y un grado de libertad variable. Además, cada prueba esta basada en 100 criptogramas de la misma plantilla clara pero de cien claves secretas seleccionadas aleatoriamente.

El criptograma debe tener el mismo número de 0's y 1's idealmente. La prueba de *monobit* determina cuantos 0's y 1's tiene un criptograma. La prueba *monobit* se calcula con la siguiente expresión

$$Mt = (zeros - ones)^2 / (2073 * 8) \tag{9.16}$$

donde Mt es la prueba de *monobit* del templete cifrado, *zeros* es el número de 0's en el criptograma y *ones* es el número de 1's en el criptograma. Se considera una distribución de chi-cuadrada χ^2 de un grado de libertad y el rango critico para Mt debe ser menor a 3.814. La figura 9.15(a) muestra el resultado de *monobit* de 100 criptogramas y el 96 % pasa la prueba. La prueba *poker* verifica si una secuencia de longitud m y todas sus combinaciones se reproduce con la misma probabilidad; esta prueba se determina como sigue

$$Pt = \left(\frac{2^m}{k}\right)\left(\sum_{i=1}^{i=c} sum(i)\right) - k, \tag{9.17}$$

donde Pt es la prueba *poker*, $sum(i)$ es la concurrencia de la secuencia i en el criptograma, $m = 3$ es la longitud de la secuencia, $c = 2^3$ son todas las posibles combinaciones de $m \in (000, 001, ...111)$ y $k = (2073 * 8)/3$ son los trigramas no traslapados del criptograma; de acuerdo con la tabla de χ^2 y el grado de libertad definido por $2^m - 1 = 7$, el umbral de Pt máximo es 14.07. En la figura 9.15(b), cien pruebas de *poker* se muestran para cada criptograma y el 97 % la pasa con éxito. En la prueba *run*, una corrida de j 1's 0111...10 (número de 1's continuos) es conocido como *block* y una corrida de j 0's 1000...01 es llamado *gap*; este análisis prueba si el número de corridas binarias de 1's aparece con la misma probabilidad en el criptograma. La siguiente expresión se utiliza para calcular las corridas para $j = 1, 2, 3$ de 1's y de 0's

$$Rt = \sum_{j=1}^{3}\left[\frac{(B_j - E_j)^2}{E_j}\right] + \left[\frac{(G_j - E_j)^2}{E_j}\right] \tag{9.18}$$

donde Rt es la prueba de *run*, $E_j = (N - j + 3)/2^{(j+2)}$ es el número esperado de corridas de longitud $j = 1, 2, 3$ en una secuencia de $N = 8$-bit, B_j es el número de *blocks* de longitud j en el criptograma y G_j es el número de *gaps* de longitud j en el criptograma; con grado de libertad determinado por $(2 * j - 2) = 4$, el límite es de 9.488 de acuerdo con la tabla χ^2. En la figura 9.15(c), se muestran los resultados de la prueba de *run* de cien criptogramas y el 97 % pasa el análisis. La prueba *serial* revisa si un bit es dependiente de su bit predecesor, calculado mediante

$$St = \left(\frac{4}{L}\right)\left((N_{00})^2 + (N_{01})^2 + (N_{10})^2 + (N_{11})^2\right) - \left(\frac{2}{L}\right)\left((N_0)^2 + (N_1)^2\right) + 1, \tag{9.19}$$

donde St es la prueba *serial*, $L = (2073 * 8) - 1$ es la longitud, N_0 y N_1 son los números de 0's y 1's en el criptograma y $N_{00,01,10,11}$ son los números importantes de la subsecuencia con traslape; la distribución χ^2 con dos grados de libertad es de 5.991 y la figura 9.15(d) muestra el resultado de St de cien criptogramas, donde el 94 % pasa con éxito la prueba de aleatoriedad.

Las pruebas anteriores aportan información importante sobre la calidad del cifrado en términos de aleatoriedad y se puede considerar que el proceso de cifrado genera criptogramas con características de pseudoaleatoriedad excelentes. También, el 95 % de las claves secretas se pueden considerar fuertes y el otro 5 % se considera de mediana fuerza pero no débiles ya que pueden generar buenas propiedades pseudoaleatorias.

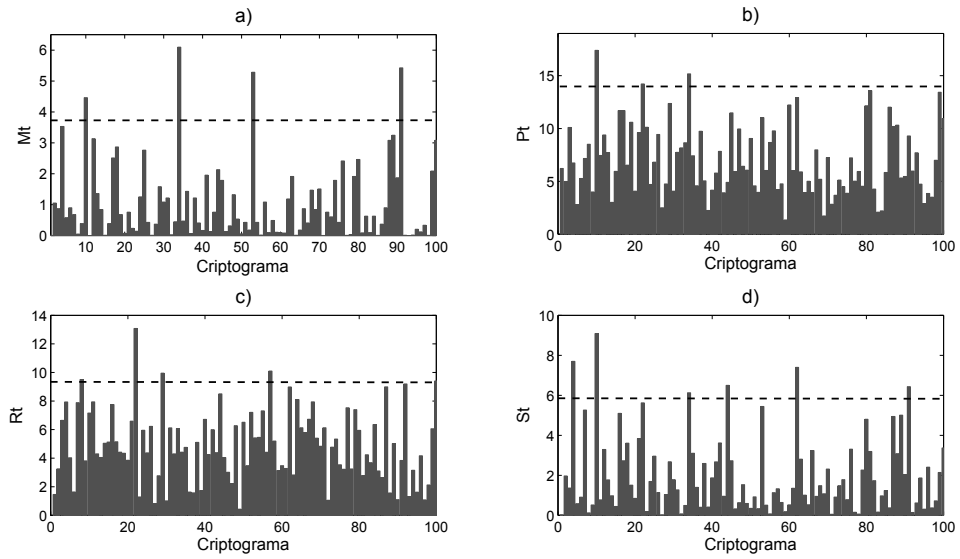


Figura 9.15: Análisis de aleatoriedad FIPS-140-2 de cien criptogramas: a) prueba *monobit*, b) prueba *poker*, c) prueba *run* y d) prueba *serial*.

9.4.10. Recursos de implementación y desempeño

La implementación esta basada en un microcontrolador de 32 bits con procesador COLDFIRE de Freescale; el algoritmo de cifrado se programa con lenguaje C y mediante el software CodeWarrior de Freescale se programa la memoria flash del microcontrolador. La frecuencia de operación es de 80 MHz, se utiliza RS-232 para la comunicación con el módulo FS83 y comunicación I2C para el teclado y pantalla LCD. También, se utiliza la memoria flash y salidas digitales para control de acceso.

El algoritmo de cifrado y configuración del microcontrolador requiere de 116 KB (22 %) de un total de 512 KB; los otros 400 KB son utilizados para almacenar plantillas cifradas de los usuarios autorizados. Por tanto, el sistema embebido de autentificación puede registrar hasta 370 usuarios con la memoria flash del microcontrolador. Además,

se puede utilizar una memoria externa para dar mayor capacidad.

El sistema de autentificación propuesto posee la precisión del módulo FS83 con una tasa de falsa aceptación de 10^6 (uno en un millón acepta un falso usuario) y una tasa de falso rechazo de 10^2 (una de cada cien rechaza a un usuario autorizado). El tiempo de registro de un nuevo usuario requiere de 2.2 segundos, lo que incluye lectura de huella dactilar, cifrado en microcontrolador y almacenamiento en memoria flash. El proceso de autentificación requiere de 3.1 segundos que incluye leer la huella dactilar, descifrar el criptograma del usuario y proceso de comparación. El costo de implementación es menor a $250 usd. Por tanto, el esquema propuesto puede ser implementado en aplicaciones en tiempo real.

9.5. Conclusiones

En este capítulo, un sistema embebido de autentificación basado en huella dactilar y su cifrado caótico fue presentado. El sistema embebido esta basado en un microcontrolador de 32 bits que funciona como unidad central y un módulo FS83 como esclavo (que lee la huella dactilar y realiza proceso de comparación). El sistema propuesto cifra la plantilla de huella dactilar mediante el algoritmo propuesto en este trabajo doctoral; el cifrado se considera robusto con excelentes características pseudoaleatorias de acuerdo con los análisis de seguridad presentados. Los recursos de implementación son mínimos y a bajo costo para generar un rápido tiempo de cifrado y de autentificación. Basado en las limitaciones de la memoria flash del microcontrolador, el sistema puede registrar hasta 370 usuarios en 400 KB de memoria. Debido a la alta seguridad, precisión, bajo costo y velocidad del sistema embebido propuesto en esta tesis, puede ser implementado en aplicaciones en tiempo real en comercio electrónico, celulares, oficinas, bancos, industria, hospitales, entre muchas otras áreas donde se requiere un sistema de autentificación seguro para el control de acceso.

Capítulo 10

Conclusiones generales

En este trabajo de tesis doctoral, se diseñó e implementó un algoritmo criptográfico basado en caos, que se caracteriza por ser seguro y eficiente para aplicaciones en sistemas embebidos (y no embebidos) para brindar confidencialidad a la información cuando se almacena en base de datos o se transmite a través de un canal inseguro y con ello resolver un problema de seguridad que se presenta en los sistemas de comunicación actuales.

Primeramente, se analizaron e implementaron dos sistemas caóticos (sistema de Lorenz y mapa logístico) en la tecnología digital del microcontrolador, para determinar qué sistema ofrecía la mejor combinación de desempeño *vs* seguridad para ser utilizado en el cifrado. Se optó por el mapa logístico que presenta algunas desventajas en su aplicación en la criptografía pero se hacen algunas consideraciones para evitarlas. Además, la existencia de caos fue verificada con la determinación del máximo exponente de Lyapunov.

Después, se diseñó un algoritmo criptográfico que tiene las siguientes propiedades: clave secreta simétrica, cifrado de flujo, una ronda de la arquitectura de confusión-difusión y secuencias caóticas del mapa logístico (cifrado no convencional) con distribución mejorada. Las características de seguridad y eficiencia que presenta son: clave secreta de 32 dígitos hexadecimales (128 bits) para determinar las secuencias caóticas de manera indirecta (resiste un ataque exhaustivo aún con el uso de un mapa caótico unidimensional), distribución mejorada de las secuencias caóticas con una simple operación (criptograma con mejores propiedades estadísticas), se consideran las características de la información clara para determinar las secuencias pseudoaleatorias de cifrado (incrementa la sensibilidad a texto claro y clave secreta), procesos de confusión y difusión optimizados (el 100 % de los elementos de texto claro son permutados y la difusión se realiza con datos optimizados del mapa logístico), selección de los últimos datos caóticos para el cifrado (incrementa la sensibilidad a la clave secreta) y eficiencia de cifrado (proceso de confusión y difusión en una operación).

Finalmente, el algoritmo criptográfico propuesto en la tesis doctoral se aplicó en tres diferentes casos para probar su versatilidad:

- *Imagen a color RGB*: como primera fase, se implementa en MatLab a nivel software con programación en lenguaje C para cifrar imágenes a color RGB (también cifra imágenes a escala de grises, sin embargo estas no se reportan en este trabajo doctoral por ser de menor interés).
- *Texto alfanumérico* y *Plantilla de huella dactilar*: implementados en un microcontrolador de 32 bits con programación en lenguaje C para aplicaciones embebidas donde se cifra texto alfanumérico correspondiente a la tabla de código ASCII y para cifrar plantillad biométricas de huella dactilar.

La seguridad del cifrado se verificó (a nivel lógico basada en simulaciones en MatLab) con distintos análisis de seguridad como error de desencriptado, espacio de claves, sensibilidad a la clave, sensibilidad al texto claro, histogramas, correlación, entropía de la información, frecuencia flotante, N-gramas, autocorrelación, pruebas estadísticas de aleatoriedad FIPS-140-2 del NIST. Por otra parte, la eficiencia de la implementación del cifrado en microcontrolador se validó con análisis de recursos *vs* desempeño como memoria utilizada, frecuencia del sistema, dimensiones, costos y tiempo de cifrado.

Los resultados muestran que el algoritmo criptográfico propuesto en la tesis, puede ser implementado en sistemas embebidos en aplicaciones como la milicia, medicina, biometría, informática, financiera, sanidad, pagos electrónicos, información personal, telemedicina, entre otros, donde se requiera almacenamiento y/o transmisión de información de manera segura.

10.1. Principales contribuciones de este trabajo doctoral

La siguiente lista muestra a manera de resumen, las principales contribuciones de este trabajo de investigación doctoral al cifrado de información empleando caos:

1. Se implementó el sistema de Lorenz 3D y el mapa logístico en microcontrolador de 32 bits con una precisión de 10^{-15}.
2. Se analizó la eficiencia *vs* seguridad del sistema caótico de Lorenz y logístico, en microcontrolador para su aplicación en criptografía.
3. Se verificó la existencia de la dinámica caótica generada en microcontrolador para 10,000 iteraciones en ambos sistemas caóticos mediante el máximo exponente de Lyapunov.
4. Se diseñó un algoritmo criptográfico basado en caos, que utiliza una ronda de confusión y difusión y características de la información clara para generar información cifrada, con excelentes propiedades estadísticas de pseudoaleatoriedad.

5. Se propuso un método para determinar la condición inicial y parámetro de control del mapa logístico, a partir de una clave de 128 bits y de las características totales del texto claro.

6. Se propuso un método sencillo para optimizar la distribución de datos del mapa logístico y con ello generar un mejor cifrado.

7. Algoritmo criptográfico diseñado es resistente ante un ataque de sólo texto claro elegido y conocido, con el cual, muchos algoritmos recientes basados en caos han sido quebrantados.

8. Se cifró imagen a color RGB, texto alfanumérico y plantillas dactilares.

9. Se presentaron dos esquemas de cifrado en sistema embebido: texto alfanumérico y plantillas de huella dactilar para sistemas de acceso seguro.

10. En cada caso, se analizó la calidad del cifrado mediante un amplio análisis de seguridad a nivel teórico o lógico.

10.2. Trabajo futuro

Como trabajo futuro que se deriva de esta investigación, se contempla encaminar las actividades en las siguientes direcciones:

1. Realizar análisis de seguridad a nivel físico de los sistemas embebidos presentados es este trabajo, como análisis de la información de tiempo de cálculos, el monitoreo de consumo de energía, fugas electromagnéticas, análisis de sonido o remanencia de datos, que puede proporcionar una fuente adicional de información que puede ser explotada para romper el sistema.

2. Implementar en microcontrolador y analizar otros mapas caóticos unidimensionales o bidimensionales principalmente en tiempo discreto, para su aplicación en criptografía: por ejemplo mapa círculo, mapa de Bernoulli, mapa Gaussiano, mapa de Duffing, mapa exponencial, mapa de Hénon, mapa de Ikeda, mapa de Lozi, mapa estándar, mapa de tent, ecuación Duffing y oscilador Van der Pol.

3. Discretizar sistemas hipercaóticos e implementarlos en microcontrolador para determinar la combinación de eficiencia *vs* seguridad para su aplicación en criptografía.

4. Utilizar criptografía ADN y caótica para diseñar un algoritmo criptográfico e implementarlo en sistemas embebidos.

5. Aplicación del cifrado propuesto en sistemas embebidos con transmisión de datos a través de internet: meteorología remota, control de acceso físico y lógico, monitoreo de sismos y maremotos, entre otros.

6. Implementación de algoritmo de cifrado en FPGA para proteger imágenes digitales y video, para aplicación en videoconferencia y televisión de paga.

7. Validar el algoritmo criptográfico propuesto con el estándar FIPS-140-2: requerimientos de seguridad para sistemas criptográficos con nivel de seguridad 3 donde se requiere seguridad física.

10.3. Productos derivados de este trabajo doctoral

Los estudios doctorales consistieron de cuatro años, durante los cuales, se realizaron trabajos de investigación relacionados con la criptografía caótica y su implementación en sistemas embebidos. Estos trabajos están publicados o sometidos en revistas indexadas en SCI (del ingles, *Science Citation Index*), congresos nacionales e internacionales y revistas de divulgación. Estos productos se listan a continuación:

I) Revistas indexadas (SCI)

1. **Murillo-Escobar M.A.**, Cruz-Hernández C., Abundiz-Pérez F., López-Gutiérrez R.M., y Acosta Del Campo O.R. (2015) A RGB image encryption algorithm based on total plain image characteristics and chaos. *Signal Processing*, **109**: 119–131. Factor de Impacto: 2.238

2. **Murillo-Escobar M.A.**, Cruz-Hernández C., Abundiz-Pérez F. y López-Gutiérrez R.M. (2014) A robust embedded biometric authentication system based on fingerprint and chaotic encryption. *Expert Systems with Applications*, sometido. Factor de Impacto: 1.965

3. **Murillo-Escobar M.A.**, Cruz-Hernández C., Abundiz-Pérez F. y López-Gutiérrez R.M. (2015) Implementation of an improved chaotic encryption algorithm for real-time embedded systems by using a 32-bit microcontroller. *Microprocessors and Microsystems*, sometido. Factor de Impacto: 0.598

II) Artículos en extenso en congresos nacionales e internacionales

1. **Murillo-Escobar M.A.**, Cruz-Hernández C., Abundiz-Pérez F. y López-Gutiérrez R.M. (2014) Cifrado caótico de plantilla de huella dactilar en sistemas biométricos. *XVI Congreso Latinoamericano de Control Automático CLCA 2014, Cancún, Quintana Roo*, pp. 18-23.

2. **Murillo-Escobar M.A.**, Abundiz-Pérez F., Cruz-Hernández C. y López-Gutiérrez R.M. (2014) A novel symmetric text encryption algorithm based on logistic map. *2014 Proceedings of the International Conference on Communications, Signal Processing and Computers, Interlaken, Suiza*, pp. 49-53. ISBN: 978-1-61804-215-6

III) Artículos de divulgación

1. **Murillo-Escobar M.A.**, López-Gutiérrez R.M., Cruz-Hernández C. y Abundiz-Pérez F. (2013). Encriptado caotico de audio utilizando el sistema de Lorenz. *XX Jornadas de Ingeniería Arquitectura y Diseño, FIAD-UABC*, pp. 97-100. ISBN: 978-0-615-93971-1

2. **Murillo-Escobar M.A.**, López-Gutiérrez R.M., Cruz-Hernández C. y Abundiz-Pérez F. (2013). Implementación del sistema caótico de Lorenz en tecnología digital FPGA. *XX Jornadas de Ingeniería Arquitectura y Diseño, FIAD-UABC*, pp.101-104. ISBN: 978-0-615-93971-1

3. **Murillo-Escobar M.A.**, López-Gutiérrez R.M., Cruz-Hernández C. y Abundiz-Pérez F. (2013). Sincronización de nueve luciérnagas electrónicas. *XX Jornadas de Ingeniería Arquitectura y Diseño, FIAD-UABC*, pp. 105-108. ISBN: 978-0-615-93971-1

IV) Seminarios

1. **Murillo-Escobar M.A.**, Cruz-Hernández C., Abundiz-Pérez F. y López-Gutiérrez R.M. Cifrado caótico de plantilla de huella dactilar en sistemas biométricos. Seminarios del cuerpo académico *Sistemas complejos y sus aplicaciones*, Platica impartida en FIAD-UABC en octubre de 2014.

2. **Murillo-Escobar M.A.** Encriptado simétrico para imágenes a color basado en la arquitectura de permutación–difusión y el mapa logístico caótico. Platica impartida en Física Aplicada, CICESE, noviembre de 2013.

3. **Murillo-Escobar M.A.** Encriptado de imagen a color usando caos. Seminarios del cuerpo académico *Sistemas complejos y sus aplicaciones*, Platica impartida en FIAD-UABC en mayo de 2012.

10.3.1. Trabajos en colaboración

I) Revistas indexadas (SCI)

1. Acosta-Del Campo O.R., Cruz-Hernandez C., López-Gutiérrez R.M., Arellano-Delgado A. y **Murillo-Escobar M.A.** (2015) Complex Networks Synchronization of Rosslers Oscillators. *Mathematical Problems in Engineering*, Sometido.

2. Acosta-Del Campo O.R., Cardoza-Avendano L., López-Gutiérrez R.M., Cruz-Hernandez C., Abundiz-Perez F., **Murillo-Escobar M.A.** (2015) Network synchronization of chaotic Nd:YAG lasers. *Chaos, Solitons and Fractals*, Sometido.

3. Abundiz-Pérez F., Cruz-Hernández C., **Murillo-Escobar M.A.** y López-Gutiérrez R.M. (2015) Hyperchaotic encryption of fingerprint image in biometric system. *ETRI Journal*, sometido.

II) Artículos en extenso en congresos nacionales e internacionales

1. Abundiz-Pérez F., Cruz-Hernández C., **Murillo-Escobar M.A.** y López-Gutiérrez R.M. (2014) Fingerprint image encryption based on Rössler map. *2014 Proceedings of the International Conference on Communications, Signal Processing and Computers, Interlaken, Suiza*, pp. 193-197.

2. Abundiz-Pérez F., Cruz-Hernández C., **Murillo-Escobar M.A.**, López-Gutiérrez R.M. Michel-Macarty J.A. y Cervantes-De Avila H. (2014) Encriptado de imágenes utilizando caos y secuencia de ADN. *XVI Congreso Latinoamericano de Control Automático CLCA 2014, Cancún, Quintana Roo*, pp. 12-17.

III) Artículos de divulgación

1. Cruz-Hernández C., López-Gutiérrez R.M., Abundiz-Pérez F. y **Murillo-Escobar M.A.** (2013). Sistema de acceso seguro basado en huella dactilar y encriptado caótico con mapa logístico. *XX Jornadas de Ingeniería Arquitectura y Diseño, FIAD-UABC*, pp. 109-112.

Actualmente, se cuenta con material para someter en congresos o revista de divulgación, los cuales se mencionan a continuación:

IV) Trabajos en proceso de sometimiento

1. A novel dynamic model to synchronize fireflies based on coupling matrix. *Para congreso.*

2. Estudio de redes complejas: Una Teoría en su infancia. *Para artículo de divulgación.*

Parte de las actividades de formación en los estudios doctorales, fue participar en la revisión de trabajos que fueron sometidos en revistas indexadas y congresos, estos se listan a continuación:

V) Arbitraje de artículos para revistas indexadas y congresos

1. *Optics & Laser Technology.* Original Research (SCI). Trabajo revisado en 2012.

2. *Mathematical Problem in Engineering.* Original Research (SCI). Trajo revisado en 2013.

3. *Nonlinear Dynamics.* Original Research (SCI). Trabajo revisado en 2014.

4. *Nonlinear Dynamics.* Original Research (SCI). Trabajo revisado en 2014.

5. *XVI Congreso Latinoamericano de Control Automático CLCA 2014.* Manuscrito para congreso revisado en 2014.

6. *XVI Congreso Latinoamericano de Control Automático CLCA 2014*. Manuscrito para congreso revisado en 2014.

7. *XVI Congreso Latinoamericano de Control Automático CLCA 2014*. Manuscrito para congreso revisado en 2014.

Bibliografía

[1] Electronic Frontier Foundation (1998). Cracking DES: Secrets of Encryption Research, Wiretap Politics & Chip Design. *O'Reilly Media*, 1ra Ed., pp. 226.

[2] Pecora L. M. y Carroll T. L. (1990). Synchronization in chaotic systems. *Physical Review Letters*, **64**(8): 821-824.

[3] Pecora L. M. y Carroll T. L. (1991). Synchronizing chaotic systems. *IEEE Transactions in Circuits Systems*, **38**: 454-456.

[4] Cuomo K. M., Oppenheim A. V. y Isabelle S. H. (1992). Spread spectrum modulation and signal masking using synchronized chaotic system. *MIT Res. Lab. Electron.*, TR570.

[5] Cuomo K. M. y Oppenheim A. V. (1992). Synchronized chaotic circuits and systems for communications. *MIT Res. Lab. Electron.*, TR575.

[6] Cuomo K. M. y Oppenheim A. V. (1993). Circuit implementation of synchronized chaos with applications to communications. *Fis. Rev. Lett.*, **71**(1): 65-68.

[7] Cuomo K. M., Oppenheim A. V. y Strogatz S. H. (1993). Synchronization of Lorenz-Based Chaotic circuits with applications to communications. *IEE transactions on circuits and systems II: Analog and digital signal Processing*, **40**(10): 623-633.

[8] Milanović V. y Zaghloul M. E. (1996). Improved masking algorithm for chaotic Communication systems. *Electronic Letters*, **32**(1): 11-12.

[9] Gonzales O. A., Han G., Pineda de Gyvez J. y Sanchez-Sinencio E. (2000). Lorenz-based chaotic cryptosystem: a monolithic implementation. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, **47**(8): 1243-1247.

[10] Cruz-Hernández C. (2004). Synchronization of time-delay Chua's oscillator with application to secure communication. *Nonlinear Dynamics and Systems Theory*, **4**(1): 1-13.

[11] Posadas-Castillo C., Cruz-Heránndez C. y Núñez-Pérez R. F. (2004). Experimental realization of binary signals transmission based on syncrhonized Lorenz circuits. *Journal of Applied Research and Technology*, **2**(2): 127-137.

[12] López-Mancilla D. y Cruz-Hernández C. (2004). An analysis of robustness on the synchronization of chaotic systems under nonvanishing perturbations using sliding modes. *WSEAS Transactions on Mathematics*, **3**(2): 364-369.

[13] López-Mancilla D. y Cruz-Hernández C. (2005). A note on chaos-based communication schemes. *Revista Mexicana de Física*, **51**(3): 265-269.

[14] Cruz Hernández, C., López-Mancilla D., García-Gradilla V. J., Serrano-Guerrero H. y Núñez-Pérez R. F. (2005). Experimental realization of binary signals transmission using chaos. *Journal of Circuit, Systems and Computer*, **14**(3): 453-468.

[15] López-Mancilla D., Cruz-Hernández C. y Posadas-Castillo C. (2005). A modified chaos-based communication scheme using hamiltonian forms and observer. *Journal of Physics: Conference Series*, **23**(23): 267-275.

[16] Cruz-Hernández C. y Romero-Haros N. R. (2006). Communicating via synchronized time-delay Chuas circuits. *Communications in Nonlinear Science and Numerical Simulation*, **13**(3): 645-659.

[17] Posadas-Castillo C., López-Gutiérrez R. M. y Cruz-Hernández C. (2007). Synchronization of chaotic solid-state Nd:YAG lasers: Application to secure communication. *Communications in Nonlinear Science and Numerical Simulation*, **13**(8): 1655-1667.

[18] Aguilar-Bustos A. Y., Cruz-Hernández C., López-Gutiérrez R. M. y Posadas-Castillo C. (2008) Synchronization of different hyperchaotic maps for encryption. *Nonlinear Dynamics and Systems Theory*, **8**(3): 221-236.

[19] López-Gutiérrez R. M., Rodríguez-Orozco E., Cruz-Hernández C., Inzunza-González E., Posadas-Castillo C., García-Guerrero E. E. y Cardoza-Avendaño L. (2009). Secret communications using synchronized sixth-order Chuas circuits. *World Academy of Science, Engineering and Technology*, **54**(30): 608-612.

[20] López-Gutiérrez R. M., Posadas-Castillo C., López-Mancilla D. y Cruz-Hernandez C. (2009). Communicating via robust synchronization of chaotic lasers. *Chaos, Solitons and Fractalsy*, **42**(1): 277-285.

[21] Gámez-Guzmán L. M., Cruz-Hernández C., López-Gutiérrez R. M. y García-Guerrero E. E. (2009). Synchronization of Chuas circuits with multi-scroll atractors: application to communication. *Communications in Nonlinear Science and Numerical Simulation*, **14**(6): 2765-2775.

[22] Serrano-Guerrero H., Cruz-Hernández C., López-Gutiérrez R. M., Posadas-Castillo C. e Inzunza-González E. (2010). Chaotic synchronization in star coupled networks of three dimensional cellular neural networks and its application in communications. *International Journal of Nonlinear Sciences and Numerical Simulation*, **11**(8): 571-580.

[23] Cardoza-Avendaño L., López-Gutiérrez R. M., Cruz-Hernández C., Spirine V., Chávez-Pérez R. y Arellano-Delgado A. (2012). Encrypted audio transmission via synchronized chaotic Nd:YAG lasers. *Revista Mexicana de Física*, **58**(6): 472-480.

[24] Arellano-Delgado A., López-Gutiérrez R. M., Cruz-Hernández C., Posadas-Castillo C., Cardoza-Avendaño L. y Serrano-Guerrero H. (2012). Experimental network synchronization via plastic optical fiber. *Optical Fiber Technology*, **19**(12): 93-108.

[25] Orúe A. B., García-Mart´ínez M. J., Pastor G., Montoya F., Sánchez Ávila C. (2012). Criptoanálisis de un criptosistema de dos canales basado en una función no lineal caótica. *XII Reunión Española sobre Criptología y Seguridad de la Información (RECSI2012) San Sebastián, España*, 119–124.

[26] Wang X., Zhan M., Lai C.-H. y Hu G. (2004). Error function attack of chaos synchronization based encryption schemes. *Chaos*, **14**(1): 128-137.

[27] Li S., Alvarez G., Li Z. y Halang W. A. (2007). Analog chaos-based secure comunications and cryptanalysis: a brief survey. *PhysCon*, arXiv:0710.5455v1.

[28] Zhang Y., Tao C. y Jiang J. J. (2006). Theoretical and experimental studies of parameter estimation based on chaos feedback synchronization. *Chaos*, **16**(4): 1-8.

[29] Qi G., Antonie van Wyk M., Jacobus van Wyk B. y Chen G. (2009). A new hiperchaotic system and its circuit implementation. *Chaos, Solitions & Fractals*, **40**(5): 2544-2549.

[30] Pang S. y Liu Y. (2011). A new hyperchaotic system from the Lu system and its control. *Journal of Computational and Applied Mathematics*, **235**(8): 2775-2789.

[31] Kocarev L. (2001). Chaos-based cryptography: A brief overview. *IEEE Circuits and Systems Magazine*, **1**(2): 6-21.

[32] Yoon J. W. y Kim H. (2010). An image encryption scheme with a pseudorandom permutation based on chaotic maps. *Communications in Nonlinear Science and Numerical Simulation*, **15**(12): 3998–4006.

[33] Ahadpour S., Sadra Y., Arasteh Fard Z. (2012). A novel chaotic encryption scheme based on pseudorandom bit padding. *International Journal of Computer Science Issues*, **9**(1): 449-456.

[34] Francois M., Grosges T., Barchiesi D. y Erra R. (2013). A new pseudo-random number generator based on two chaotic maps. *INFORMATICA*, **24**(2): 181–197.

[35] Baptista, M. S. (1998). Cryptography with chaos. *Physics Letters A*, **240**(1): 50-54.

[36] Wang X.-Y. y Teng L. (2012). A one-time one-key encryption algorithm based on the ergodicity of chaos. *Chinese Physics B*, **21**(2): 20504-20508.

[37] Asim M. y Jeoti V. (2007). Hybrid chaotic image encryption scheme based on S-box and ciphertext feedback. *International Conference on Intelligent and Advanced Systems, Kuala Lumpur, Malasia*, 25-28 noviembre de 2007, 736-741.

[38] Hussaina I., Azama N. A. y Shaha T. (2014). Stego optical encryption based on chaotic S-box transformation. *Optics & Laser Technology*, **61**(6): 50–56.

[39] Zhang X., Mao Y. y Zhao Z. (2014). An efficient chaotic image encryption based on alternate circular S-boxes. *Nonlinear Dynamics*, **78**(1): 359-369.

[40] Mishra M. y Mankar V. H. (2012). Message embedded cipher using 2-D chaotic map. *International Journal of Chaos, Control, Modelling and Simulation*, **1**(1): 13-24.

[41] Pareek N. K., Patidar V. y Sud K. K. (2006). Image encryption using chaotic logistic map. *Image and Vision Computing*, **24**(9): 926-934.

[42] Patidar V., Pareek N. K. y Sud K. K. (2009). A new substitution-diffusion based image cipher using chaotic standard and logistic maps. *Commun Nonlinear Sci Numer Simulat*, **14**(7): 3056-3075.

[43] Hongjun L. y Xingyuan W. (2010). Color image encryption based on one-time keys and robust chaotic maps. *Computers and Mathematics with Applications*, **59**(10): 3320-3327.

[44] Cui D. (2010). A novel fingerprint encryption algorithm based on chaotic system and fractional fourier transform. *IEEE International Conference on Machine Vision and Human-machine Interface*, 168-171.

[45] Chen D. y Chang Y. (2011). A novel image encryption algorithm based on logistic maps. *Advances in Information Sciences and Service Sciences*, **3**(7): 364-372.

[46] Wang X., Teng L. y Qin X. (2012). A novel colour image encryption algorithm based on chaos. *Signal Processing*, **92**(4): 1101-1108.

[47] Liu R. (2012). Chaos-based fingerprint images encryption using symmetric cryptography. *9th International Conference on Fuzzy Systems and Knowledge Discovery*, pp. 2153-2156.

[48] Inzunza-González E. y Cruz-Hernández C. (2013). Double hyperchaotic encryption for security in biometric systems. *Nonlinear Dynamics and Systems Theory*, **13**(1): 55-68.

[49] Naeem E. A., Abd Elnaby M. M., Soliman N. F., Abbas A. M., Faragallah O. S., Semary N., Hadhoud M. M., Alshebeili S. A. y Abd El-Samie F. E. (2014). Efficient implementation of chaotic image encryption in transform domains. *The Journal of Systems and Software*, **97**(7): 118–127.

[50] Khurram Khan M., Zhang J. y Tian L. (2007). Chaotic secure content-based hidden transmission of biometric templates. *Chaos, Solitons and Fractals*, **32**(5): 1749-1759.

[51] Xiaomin W., TaiHua Xu y Wenfang Z. (2011). Chaos-based biometrics template protection and secure authentication. *State of the Art in Biometrics*, Dr. Jucheng Yang (Ed.), pp. 293-314.

[52] Wang K., Pei W., Zou L., Song A. y He Z. (2005). On the security of 3D Cat map based symmetric image encryption scheme. *Physics Letters A*, **343**(6): 432-439.

[53] Wang Y., Liao X., Xiang T., Wong K.-W. y Yang D. (2007). Cryptanalysis and improvement on a block cryptosystem based on iteration a chaotic map. *Physics Letters A*, **363**(4): 277-281.

[54] Balaji N. y Nagaraj N. (2008). Cryptanalysis of a chaotic image encryption algorithm. *National Conference on Nonlinear Systems and Dynamics*, arXiv:0801.0276, pp. 1-14.

[55] Cokal C. y Solak E. (2009). Cryptanalysis of a chaos-based image encryption algorithm. *Physics Letters A*, **373**(15): 1357-1360.

[56] Alvarez G. y Li S. (2009). Cryptanalyzing a nonlinear chaotic algorithm (NCA) for image encryption. *Commun Nonlinear Sci Numer Simulat*, **14**(11): 3743-3749.

[57] Li C., Chen G. y Halang W. A. (2009). Cryptanalysis of an image encryption scheme based on a compound chaotic sequence. *Image and Vision Computing*, **27**(8): 1035-1039.

[58] Rhouma R., Solak E. y Belghith S. (2010). Cryptanalysis of a new substitution-diffusion based image cipher. *Commun Nonlinear Sci Numer Simulat*, **15**(7): 1887-1892.

[59] Solak E., Cokal C., Yildiz O. T. y Biyikiglu T. (2010). Cryptoanalysis of Fridrich's chaotic image encryption. *International Journal of Bifurcation and Chaos*, **20**(5): 1405-1413.

[60] Li C., Li S. y Lo K.-T. (2011). Breaking a modified substitution-diffusion image cipher based on chaotic standard and logistic maps. *Commun Nonlinear Sci Numer Simulat*, **16**(2): 837-843.

[61] Zhang Y., Li C., Li Q., Zhang D. y Shum S. (2011). Breaking a chaotic image encryption algorithm based on perceptron model. *Nonlinear Dynamics*, **69**(3): 1091-1096.

[62] Li C., Zhang Y., Ou R. y Wong K.-W. (2012). Breaking a novel colour image encryption algorithm based on chaos. *Nonlinear Dynamics*, **70**(4): 2383-2388.

[63] Arroyo D., Alvarez G. y Li S. (2009). Some Hints for the Design of Digital Chaos-Based Cryptosystems: Lessons Learned from Cryptanalysis. *Conference on Analysis and Control of Chaotic Systems*, pp. 171-175.

[64] Shannon, C. E. (1949). A mathematical theory of communication. *The Bell System Technical Journal*, **27**: 379–423.

[65] Shannon, C. E. (1949). Communication theory of secrecy systems. *The Bell System Technical Journal*, **28**: 656-715.

[66] Brown R. y Chua L. O. (1996). Clarifying chaos: examples and counterexamples. *International Journal of Bifurcation and Chaos*, **6**(2): 219–249.

[67] Fridrich J. (1998). Symmetric ciphers based on two-dimensional chaotic maps. *Int. J. Bifurcation Chaos*, **8**(6): 1259.

[68] Alvarez G., Montoya F., Pastor G. y Romera M. (1999). Chaotic cryptosystems. *International Carnahan Conference on Security Technology, Madrid, España*, 5-7 de octubre de 1999, 332-338.

[69] Petitcolas F.A.P. (2011). Kerckhoffs' Principle. *Encyclopedia of Cryptography and Security*, 2da Ed., pp. 675.

[70] Ott E. (1993). Chaos in dynamical systems. *Cambridge University Press*, 1ra Edición, pp. 385.

[71] Hilborn R. C. (2005). Chaos and nonlinear dynamics: An introduction for scientists and engineering. *OXFORD University Press*, 2da Edición, pp. 672.

[72] Lorenz E. N. (1963). Deterministic nonperiodic flow. *Journal of the atmospheric science*, **20**(2): 130-142.

[73] Wolf A. (1986). Quantifying chaos with Lyapunov exponents. *Chaos*, Princeton University Press, cap. 13, pp.273-289.

[74] Sprott J. C. (2003). Lyapunov exponents. *Chaos and time-series analysis*, Oxford University Press, cap. 5.

[75] May R. M. (1976). Simple mathematical models with very complicated dynamics. *Nature*, **261**(5560): 459-467.

[76] Granados, G. (2006). Introducción a la criptografía. *Revista Digital Universitaria*, **7**(7): 1-17.

[77] Kotulsk Z. y Szczepañski J. (1997). Discrete chaotic cryptography. *Proc. NEEDS 1997*, 1–11.

[78] Alvarez G. y Li S. (2006). Some basic cryptographic requirements for chaos-based cryptosystems. *International Journal of Bifurcation and Chaos*, **16**(8): 2129-2151.

[79] Menezes A., van Oorschot P. y Vanstone S. (1996). Pseudorandom Bits and Sequences. *Handbook of Applied Cryptography*, CRC Press, cap. 5, pp. 169-190.

[80] FIPS 140-1 U.S. Department of Commerce / National Institute of Standards and Technology (1994). Security requirements for cryptographic modules FIPS 140-1. *Federal Information Processing Standards Publication*, FIPS 140-1.

[81] FIPS 140-2 U.S. Department of Commerce / National Institute of Standards and Technology (2001). Security requirements for cryptographic modules FIPS 140-2. *Federal Information Processing Standards Publication*, FIPS 140-2.

[82] MCF5225X Freescale Semiconductor (2008). MCF5225x Family. *Freescale Semiconductor*, KRN3MCF5225XFS/REV 2.

[83] Aboul-Seoud A. K., El-Badawy E. A., Mokhtar A., El-Marsy W., El-Barbry M. y El-Din Sayed Hafez A. (2011). A simple 8 bit digital microcontroller implementation for chaotic sequence generation. *28th National Radio Science Conference, National Telecomunication Institute, Cairo, Egipto*, 26-28 abril de 2011, 1-9.

[84] Serna J. D. y Joshi A. (2011). Visualizing the logistic map with a microcontroller. *Physics Education*, **47**(6): 1-6.

[85] Volos C. K. (2013). Chaotic random bit generator realized with a microcontroller. *Journal of Computations & Modelling*, **3**(4): 115-136.

[86] Chiu R., Mora-González M. y López-Mancilla D. (2013). Implementation of a chaotic oscillator into a simple microcontroller. *IERI Procedia*, **4**, 247-252.

[87] Stanciu M. y Datcu O. (2012). Atmel AVR microcontroller implementation of a new enciphering algorithm based on a chaotic generalized Hénon map. *9th International Conference on Communications, Bucharest, Rumania*, 21-23 junio de 2012, 319-322.

[88] Andreatos A. S. y Volos C. K. (2014). Secure text encryption based on hardware chaotic noise generator. *2nd International Conference on Cryptography and Its Applications in the Armed Forces, Atenas, Grecia*, 2 de abril del 2014.

[89] Zidan M. A., Radwan A. G. y Salama K. N. (2011). The effect of numerical techniques on differential equation based chaotic generators. *IEEE International Conference on Microelectronics, Hammamet, Túnez*, 19-22 diciembre de 2011, 1-4.

[90] Yang W. Y., Cao W., Chung T. y Morris J. (2005). Aplied numerical methods using Matlab. *Wiley-Interscience*, 509 páginas.

[91] Arroyo D., Alvarez G. y Fernandez V. (2008). On the inadequacy of the logistic map for cryptographic applications. *X Reunión Española sobre Criptología y Seguridad de la Información*, 77-82.

[92] Cristian-Iulian R. y Vasile-Gabriel I. (2017). Aspects regarding chaotic maps hardware implementations. *Revue Roumaine Des Sciences Techniques*, **52**(2): 219-227.

[93] Zhou S, Zhang Q., Wei X. y Zhou C. (2012). A summarization of image encryption. *IETE Technical Review*, **27**(6): 503-510.

[94] Dang P. P. y Chau P. M. (2000). Image encryption for secure internet multimedia applications. *International Journal of Computer Science & Engineering*, **46**(3): 396-403.

[95] Chen G., Mao Y. y Chui C. K. (2004). A symmetric image encryption scheme based on 3D chaotic cat maps. *International Journal of Computer Science & Engineering*, **21**(3): 749-761.

[96] Zeghid M., Machhout M., Khriji L., Baganne A. y Tourki R. (2007). A modified AES based algorithm for image encryption. *International Journal of Computer, Information, Systems and Control Engineering*, **1**(3): 726-731.

[97] Muhaya F. B., Usama M. y Khan M. K. (2009). Modified AES using chaotic key generator for satellite imagery encryption. *Emerging Intelligent Computing Technology and Applications*, **5754**: 1014-1024.

[98] Wong K.-W., Kwok S.-H. y Law W.-S. (2008). A fast image encryption scheme based on chaotic standard map. *Physics Letters A*, **372**(15): 2645–2652.

[99] Usamaa M., Khana M. K., Alghathbar K. y Lee C. (2010). Chaos-based secure satellite imagery cryptosystem. *Computers and Mathematics with Applications*, **60**(2): 326-337.

[100] Wang X., Yang L., Liu R. y Kadir A. (2010). A chaotic image encryption algorithm based on perceptron model. *Nonlinear Dynamics*, **62**(3): 615-621.

[101] Tong X. y Cui M. (2008). Image encryption with compound chaotic sequence cipher shifting dynamically. *Image and Vision Computing*, **26**(6): 843-850.

[102] Feng Y., Li L. y Huang F. (2006). A symmetric image encryption approach based on line maps. *1st International Symposium on Systems and Control in Aerospace and Astronautics, Harbin, República Popular China*, 19-21 junio de 2006, 1367.

[103] Gao H., Zhang Y., Liang S. y Li D. (2006). A new chaotic algorithm for image encryption. *Chaos, Solitons & Fractals*, **29**: 393-399.

[104] Xiang T., Liao X., Tang G., Chen Y. y Wong K.-W. (2006). A novel block cryptosystem based on iterating a chaotic map. *Physics Letters A*, **349**(1): 109-115.

[105] Guan Z.-H., Huang F. y Guan W. (2005). Chaos-based image encryption algorithm. *Physics Letters A*, **346**(1): 153-157.

[106] Patidar V., Pareek N. K., Purohit G. y Sud K. K. (2010). Modified substitution-diffusion image cipher using chaotic standard and logistic maps. *Commun Nonlinear Sci Numer Simulat*, **15**(10): 2755-2765.

[107] Zhang W., Wong K.-W., Yu H. y Zhu Z.-L. (2013). An image encryption scheme using reverse 2-dimensional chaotic map and dependent diffusion. *Commun Nonlinear Sci Numer Simulat*, **18**(8): 2066-2080.

[108] Bakhshandeh A. y Eslami Z. (2013). An authenticated image encryption scheme based on chaotic maps and memory cellular automata. *Optics and Lasers in Engineering*, **51**(6): 665-673.

[109] Fu C., Hou S., Zhou W., Liu W.-Q. y Wang D.-L. (2013). A chaos-based image encryption scheme with a plaintext related diffusion. *9th Int. Conf. on Information, Communications and Signal Processing, Tainan, República de China*, 10-13 de diciembre de 2013,1-5.

[110] Zhu C. (2012). A novel image encryption scheme based on improved hyperchaotic sequences. *Optics Communications*, **285**(1): 29-37.

[111] Wang X. y Yu C. (2009). Cryptanalysis and improvement on a cryptosystem based on a chaotic map. *Computers and Mathematics with Applications*, **57**(3): 476-482.

[112] Zhou Y., Bao L. y Philip Chen C. L. (2014). A new 1D chaotic system for image encryption. *Signal Processing*, **97**(1): 172-182.

[113] Liu L., Zhang Q. y Wei X. (2012). A RGB image encryption algorithm based on DNA encoding and chaos map. *Computers and Electrical Engineering*, **38**(5): 1240-12481.

[114] Liu Y. (2014). Cryptanalyzing a rgb image encryption algorithm based on dna encoding and chaos map. *Optics and Laser Technology*, **60**: 111-115.

[115] FIPS PUB 46-3 (1999). Data encryption standard (DES). *National Institute of Standards and Technology*, http://csrc.nist.gov/publications/fips/archive/fips46-3/fips46-3.pdf

[116] FIPS PUB 197 (2001). Advanced encryption standard (AES). *National Institute of Standards and Technology*, http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[117] Ferguson N., Schroeppel R. y Whiting D. (2001). A simple algebraic representation of Rijndael. *Lecture Notes in Computer Science*, **2259**: 103-111.

[118] Adleman L. M. (1994) Molecular computation of solution to combinatorial problems. *Science*, **266**(5187): 1021-2024.

[119] Guozhen X., Mingxin L., Lei Q. y Xuejia L. (2006) New field of cryptography: DNA cryptography. *Chinese Science Bulletin*, **51**(12): 1413-1420.

[120] Bordoa M. y Tornea O. (2010) DNA secret writing techniques. *8th International Conference on Communications*, pp. 451-456.

[121] XueJia L., MingXin L., Lei Q., JunSong H. y XiWen F. (2010) Asymmetric encryption and signature method with DNA technology. *Science China Information Sciences*, **53**(3): 506-514.

[122] Abbasy M.R., Manaf A.A. y Shahidan M.A. (2011) Data Hiding Method Based on DNA Basic Characteristics. *Communications in Computer and Information Science*, **194**: 53-62.

[123] Liu H., Lin D. y Kadir A. (2013) A novel data hiding method based on deoxyribonucleic acid coding. *Computers and Electrical Engineering*, **39**(4): 1164-1173.

[124] Fournaris A.P. y Sklavos N. (2014) Secure embedded system hardware design: a flexible security and trust enhanced approach. *Computers and Electrical Engineering*, **40**(1): 121–133.

[125] MCF52259RM. MCF52259 ColdFire integrated microcontroller reference manual. *MCF52259RM* 2009;Rev. 2 8/2009.

[126] Jain A.K. and Uludag U. (2003). Hiding biometric data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**(11): 1494-1498.

[127] Soutar C., Roberge D., Stoianov A., Gilroy R. y Vijaya Kumar B.V.K. (1999). Biometric encryption. *ICSA Guide to Cryptography*, McGraw-Hill.

[128] Ratha N.K., Connell J.H. y Bolle R.M. (2001). An analysis of minutiae matching strength. *Lecture Notes in Computer Science*, **2091**: 223-228.

[129] Prasad M.V.N.K. y Kumar C.S. (2014). Fingerprint template protection using multiline neighboring relation. *Expert Systems with Applications*, **41**(14): 6114–6122.

[130] Jin Z., Beng Jin Teoh A., Song Ong T. y Tee C. (2012). Fingerprint template protection with minutiae-based bit-string for security and privacy preserving. *Expert Systems with Applications*, **39**(6): 6157–6167.

[131] Kevenaar T., Korte U., Merkle J., Niesing M, Ihmor H., Busch C. y Zhou X. (2010). A reference framework for the privacy assessment of keyless biometric template protection systems. *In Proceedings of BIOSIG, Darmstadt, Alemania*, 9-10 de septiembre de 2010, 45-56.

[132] Nandakumar K., Nagar A. y Jain A.K. (2007). Hardening fingerprint fuzzy vault using password. *Lecture Notes in Computer Science* **4642**: 927-937.

[133] Chen C.-H. y Dai J.-H. (2005). An embedded fingerprint authentication system with reduced hardware resources requirement. *Proceedings of the Ninth International Symposium on Consumer Electronics*, 14-16 de junio de 2005, 145-150.

[134] Nayak D.R. (2008). A novel architecture for embedded biometric authentication system. *European Symposium on Computer Modeling and Simulation, Liverpool, Amsterdam, Inglaterra*, 8-10 de septiembre de 2008, 567-572.

[135] Militello C., Conti V., Vitabile S. y Sorbello F. (2008). A novel embedded fingerprints authentication system based on singularity points. *International Conference on Complex, Intelligent and Software Intensive Systems, Barcelona, España*, 4-7 de marzo de 2008, 72-78.

[136] Nie T., Li Y., Zhou L. y Toyonaga M. (2013). A multilevel fingerprinting method for FPGA IP protection. *IEEE International Symposium on Circuits and Systems, Beijing, China*, 19-23 de mayo de 2013, 1789-1792.

[137] Fons M., Fons F. y Cantó E. (2007). Embedded security: new trends in personal recognition systems. *Research in Microelectronics and Electronics Conference, Bordeaux, Francia*, 2-5 de julio de 2007, 89-92.

[138] Wang Y., Li D., Isshiki T. y Kunieda H. (2004). A novel SOC architecture embedded bit serial FPGA. *IEEE Conference on Circuits and Systems*, **1**: 133-136.

[139] Roberts C. (2007). Biometric attack vectors and defences. *Computers and Security*, **26**(1): 14-25.

[140] Luis-García R. de, Alberola-López C., Aghzout O. y Ruiz-Alzola J. (2003). Biometric identification systems. *Signal Processing*, **83**(12): 2539-2557.

[141] Futronic. Futronic FS83C PIV serial Fingerprint Authentication Module(sFAM). *Futronic Technology Company Limited*, http://www.futronic-tech.com/download/FS83C/brochure.pdf.

Apéndice A

Programa MatLab: cifrado de imagen a color RGB

En esta apéndice, se presenta el software de cifrado propuesto en este trabajo doctoral aplicado a imágenes a color RGB. Se realizo con lenguaje C en MatLab. Los software de cifrado de texto y de plantilla dactilar que se implementaron en microcontrolador son similares al que se presenta en este apéndice.

```
clear all;  % Limpia variables.
format long; % Variables con 15 decimales de precision.
tic % Mide tiempo de proceso.

%% Clave secreta:
CLAVE = '1234567890ABCDEF1234567890ABCDEF';
TAMANO_CLAVE = size(CLAVE,2);

%% Leer las matrices RGB de imagen clara:
P = imread('lena_png_512.png');
[FIL COL MAT]=size(P)

%% Manejo de clave secreta Tabla 6.1
% Separar en 4 secciones:
SA1 = CLAVE(1:8);
SA2 = CLAVE(9:16);
SX1 = CLAVE(17:24);
SX2 = CLAVE(25:32);

% Convertir de hexadecimal a decimal
dec_SA1 = hex2dec(SA1);
dec_SA2 = hex2dec(SA2);
dec_SX1 = hex2dec(SX1);
dec_SX2 = hex2dec(SX2);

% Determinar el valor a sumar en las C.I. y P.C.:
A = dec_SA1 / (4294967296 + 1);
B = dec_SA2 / (4294967296 + 1);
C = dec_SX1 / (4294967296 + 1);
D = dec_SX2 / (4294967296 + 1);

%% Iteracion del mapa logistico 2:
PC_L2 = mod(A + B,1) * 0.001;
x2 = mod(C + D,1);

a2 = 3.999 + PC_L2; % Parametro de control de logistico 2
LOG_2(1) = x2;      % Condicion inicial de logistico 2

R=1000;     % Iteraciones de mapa logistico 2
for n=1:R
```

```
41      LOG_2(n+1)=a2*LOG_2(n)*(1-LOG_2(n));
42  end
43
44  LOG_2=mod(LOG_2*1000,1); % Optimizacion de secuencias caoticas.
45
46  %% Suma de pixeles de imagen clara con 50 datos caoticos de logistico 2:
47  a_sumar_32k=im2double(P); %Transformar de 0-255 a 0-1
48
49  if(MAT == 1)     % Para imagen a escala de grises
50      sum_vec_32k=0;
51      for k=1:FIL
52      sum_vec_32k =sum_vec_32k + a_sumar_32k(k,:,1);
53      end
54  else
55      sum_vec_32k=0;
56      for k=1:FIL
57      sum_vec_32k =sum_vec_32k + a_sumar_32k(k,:,1) + a_sumar_32k(k,:,2) +
58      a_sumar_32k(k,:,3);
59      end
60  end
61
62  S=0;
63  for j=1:size(sum_vec_32k,2)
64     S=S+sum_vec_32k(j);
65  end
66
67  % Agregar los ultimos 50 datos de mapa logistico 2:
68  F=0;
69  for n=1:50
70      F = F + LOG_2(1001-n);
71  end
72
73  v1 = mod((S*1000) + F,1);    % Valor entre 0-1.
74  v2 = 1 + round(v1 * 253);    % Valor entre 1-254.
75  Z = v2 / 255;                % Valor de Z.
76
77  %% Iteracion del mapa log\'istico 1:
78  PC_L1 = mod(A + B + Z,1) * 0.001;
79  x1 = mod(C + D + Z,1);
80
81  a1 = 3.999 + PC_L1; % Parametro de control de logistico 1
82  LOG_1(1) = x1;      % Condicion inicial de logistico 1
83
84  T=5000;     % Iteraciones de mapa logistico 2
85  for n=1:T
86      LOG_1(n+1)=a1*LOG_1(n)*(1-LOG_1(n)); % mapa logistico
87  end
88
89  %% Determinar local para almacenar Z en criptograma:
90  if(MAT == 1) % Para imagen a escala de grises.
91      I = round((LOG_2(R-10) * (FIL-1))+1);
92      J = round((LOG_2(R-100)* (COL-1))+1);
93      K = 1;
94  else         % Para imagen a color.
95      I = round((LOG_2(R-10) * (FIL-1))+1);
96      J = round((LOG_2(R-100)* (COL-1))+1);
97      K = round((LOG_2(R-200) * (MAT-1))+1);
98  end
99
100 %% Sub-secuencia para confusion de filas:
101 % Seleccion de los ultimos datos caoticos.
102 RE(1:FIL)=round((LOG_1((T+1)-FIL:T).*(FIL-1))+1);
103
104 % Limpia de variables
105 valor_r_32k = 0;
106 vect_r_32k = 0;
107 num_no_est_32k = 0;
108
109 % Busca posiciones de repetidos y los pone en un vector
110 bus_32k(1)=RE(1);
111 s=1;
112 for b=2:FIL
```

```
113      en=1;
114      bus_32k(b)=RE(b);
115      for j=1:b-1
116         if(bus_32k(b) == bus_32k(j) && en == 1)
117              valor_r_32k(s)=bus_32k(j);
118              vect_r_32k(s)=b;
119              s=s+1;
120              en=0;
121         end
122      end
123  end
124
125  % Buscar numeros que no esten en RE
126  tam_v_32k=size(valor_r_32k,2);
127  preg_num=1;
128  saltar=0;
129  no=1;
130  for v=1:FIL
131      for vv=1:tam_v_32k
132            if(saltar == 0 && preg_num == valor_r_32k(vv))
133                saltar=1;
134                %entra aqui si esta el numero (no se puede usar)
135             end
136      end
137      % Pregunta si esta en RE
138      for vvv=1:FIL
139          if(preg_num == RE(vvv))
140              saltar=1;  % si esta entra.
141          end
142      end
143      % Si no esta en repetidos y en RE, entra
144      if(saltar == 0)
145          num_no_est_32k(no)=preg_num;  % Guarda n\'umero.
146          no=no+1;
147      end
148      saltar=0;
149      preg_num=preg_num+1;
150  end
151
152  tam_vect_32k=size(vect_r_32k,2);
153
154  % Se actualiza RE (tiene todos los espacios)
155  en_dos=round(tam_vect_32k/2);
156  cambio=1;
157  S=1;
158  for s=1:tam_vect_32k
159      if(cambio == 1)
160       RE(vect_r_32k(s))=num_no_est_32k(S);
161       cambio=0;
162      else
163       RE(vect_r_32k(s))=num_no_est_32k(S+en_dos);
164       S=S+1;
165       cambio=1;
166      end
167  end
168
169  %% Sub-secuencia para confusion de columnas:
170  CO = round((LOG_1((T+1)-COL-FIL:T-FIL).*(COL-1))+1);
171
172  % Lipmia de variables
173  valor_re_32k = 0;
174  vect_re_32k = 0;
175  num_no_esta_32k = 0;
176
177  % Busca posiciones de repetidos y los pone en un vector
178  busq_32k(1)=CO(1);
179  s=1;
180  for b=2:COL
181      en=1;
182      busq_32k(b)=CO(b);
183      for j=1:b-1
184         if(busq_32k(b) == busq_32k(j) && en == 1)
```

```
185             valor_re_32k(s)=busq_32k(j);
186             vect_re_32k(s)=b;
187             s=s+1;
188             en=0;
189         end
190     end
191 end
192
193  % Buscar numeros que no esten en CO
194 tam_ve_32k=size(valor_re_32k,2);
195 preg_nume=1;
196 salta=0;
197 no=1;
198 for v=1:COL
199     for vv=1:tam_ve_32k
200           if(salta == 0 && preg_nume == valor_re_32k(vv))
201               salta=1;
202               %entra aqui si esta el numero (no se puede usar)
203            end
204     end
205     % Pregunta si esta en CO
206     for vvv=1:COL
207         if(preg_nume == CO(vvv))
208             salta=1;  % si esta entra.
209         end
210     end
211     % Si no esta en repetidos y en CO, entra
212     if(salta == 0)
213         num_no_esta_32k(no)=preg_nume;  % Guarda numero.
214         no=no+1;
215     end
216     salta=0;
217     preg_nume=preg_nume+1;
218 end
219
220 tam_vecto_32k=size(vect_re_32k,2);
221
222  % Se actualiza CO (tiene todos los espacios)
223 en_dos=round(tam_vecto_32k/2);
224 cambio=1;
225 S=1;
226 for s=1:tam_vecto_32k
227     if(cambio == 1)
228      CO(vect_re_32k(s))=num_no_esta_32k(S);
229      cambio=0;
230     else
231      CO(vect_re_32k(s))=num_no_esta_32k(S+en_dos);
232      S=S+1;
233      cambio=1;
234     end
235 end
236
237  %% Sub-secuencia para difusion:
238 M = mod((LOG_1((T+1)-5000:T).*(1000)) + Z,1);
239
240  % Construir una matriz de M x N
241 inc=1;
242 for k=1:FIL
243     for kk=1:COL
244         MATRIZ_M(k,kk)=M(inc);
245         inc=inc+1;
246         if (inc == 5001)
247             inc=1;
248         end
249     end
250 end
251
252  %% Proceso de cifrado: una ronda de confusion-difusion
253 P_DOUBLE=im2double(P);  % Se transforma P de 0-255 a 0-1
254
255 if(MAT == 1)  % Para imagenes a escala de grises.
256     E_DOUBLE(:,:,1)=P_DOUBLE(RE(:),CO(:),1) + MATRIZ_M(:,:);
```

```
257 else        % Para imagen a color.
258     E_DOUBLE(:,:,1)=P_DOUBLE(RE(:),CO(:),1) + MATRIZ_M(:,:);
259     E_DOUBLE(:,:,2)=P_DOUBLE(RE(:),CO(:),2) + MATRIZ_M(:,:);
260     E_DOUBLE(:,:,3)=P_DOUBLE(RE(:),CO(:),3) + MATRIZ_M(:,:);
261 end
262
263  %% Agregar valor de Z en imagen cifrada
264 E_DOUBLE(I,J,K)= Z;  % Agrega Z
265 E_DOUBLE = mod(E_DOUBLE,1);  % Valores entre 0 - 1
266 E = im2uint8(E_DOUBLE);  % Transformar a 0 - 255
267
268 toc % Termina de medir tiempo de calculo.
269
270  %% Guardar imagen cifrada y desplegar histogramas.
271 imwrite(E, 'E.png'); %Guarda imagen cifrada
272  % Muestra imagen clara y cifrada
273 figure;subplot(1,2,1);imshow(P)
274 title('Imagen clara')
275 subplot(1,2,2);imshow(E)
276 title('Imagen cifrada')
277
278  % Histogramas
279 figure;
280 if(MAT == 1)  % Para imagenes a escala de grises.
281     matriz_roj=E(:,:,1);
282     subplot(2,1,1);imshow(matriz_roj)
283     title('Imagen a ROJO')
284     subplot(2,1,2);imhist(matriz_roj)
285     title('Histograma Imagen ROJO')
286 else     % Para imagen a color.
287     matriz_roj=E(:,:,1);
288     matriz_verd=E(:,:,2);
289     matriz_azu=E(:,:,3);
290
291     subplot(3,2,1);imshow(matriz_roj)
292     title('Imagen a ROJO')
293     subplot(3,2,3);imshow(matriz_verd)
294     title('Imagen a VERDE')
295     subplot(3,2,5);imshow(matriz_azu)
296     title('Imagen a AZUL')
297      % Graficar Histograma
298     subplot(3,2,2);imhist(matriz_roj)
299     title('Histograma Imagen ROJO')
300     subplot(3,2,4);imhist(matriz_verd)
301     title('Histograma Imagen VERDE')
302     subplot(3,2,6);imhist(matriz_azu)
303     title('Histograma Imagen AZUL')
304 end
```