UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DISEÑO DE UN ALGORITMO DE CIFRADO DE CLAVE PRIVADA

TESIS

QUE PARA OBTENER EL TÍTULO DE:

INGENIERA EN COMPUTACIÓN

PRESENTA

SANTANA OSORIO ADRIANA

DIRECTOR DE TESIS

ING. ALDO JIMÉNEZ ARTEAGA

Ciudad Universitaria 2012

AGRADECIMIENTOS

A la Universidad Nacional Autónoma de México por brindarme la oportunidad de formar parte de la mejor Universidad en América Latina y formar parte de su historia.

A la Facultad de Ingeniería y profesores por brindarme los conocimientos necesarios en mi formación profesional.

A mi director de tesis Aldo Jiménez Arteaga por su paciencia y tiempo, así mismo por sus apreciados y relevantes aportes, críticas, comentarios y sugerencias durante el desarrollo de este proyecto.

A mis padres por darme la vida y porque cada uno de ellos son un gran ejemplo para mí, porque siempre han sabido guiarme, dándome ejemplos dignos de superación y porque en gran parte gracias a ellos, hoy puedo ver alcanzada mi meta, ya que siempre han creído en mi y estuvieron apoyándome en los momentos más difíciles de mi carrera, y por el orgullo que sienten por mí.

A mis hermanas, que siempre me han apoyado en las decisiones que he tomado, por creer en mí y darme palabras de aliento cuando las he necesitado.

A Josué Joel Monroy Torres por creer en mí y por su apoyo siempre que lo he necesitado, gracias por formar parte de esta etapa tan importante en mi vida y estar junto a mí.

A todos mis amigos, por su constante apoyo y ser parte de mi formación profesional, y por cada una de sus demostraciones de cariño y palabras de aliento cuando lo necesite.

A toda mi familia por su apoyo a lo largo de esta etapa de mi vida.

TABLA DE CONTENIDO

CAPÍTULO 2

DESCRIPCIÓN DEL ALGORITMO CRIPTOGRÁFICO

CAPÍTULO 3

JUSTIFICACIÓN DE DISEÑO DEL ALGORITMO SIMÉTRICO

CAPÍTULO 4

PRUEBAS CRIPTOANÁLISIS DIFERENCIAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

INTRODUCCIÓN

En la actualidad la información es uno de los principales recursos dentro de las empresas así mismo para el usuario final, jugando también un papel importante las redes sociales, correo electrónico y medios por donde viaja la información. Sin embargo, al transferirse por medio de redes o estar almacenada en un equipo o dispositivo electrónico, se encuentra expuesta ante algún intruso, el cual puede acceder a ella y alterar su contenido por citar un ejemplo.

Preocupados por la seguridad de la información, se han buscado mecanismos para mantenerla protegida. Por ejemplo existen los algoritmos de cifrado.

Tomando como base que la información es muy importante en la actualidad y que puede verse alterada en su integridad, confidencialidad y disponibilidad, se plantea diseñar un algoritmo de cifrado, que para fines de este proyecto el algoritmo será simétrico ya que sólo hace uso de una clave; para su diseño se deben considerar aspectos como son sus características, y requerimientos de diseño entre ellos la robustez, la interacción del mensaje con la clave, el uso de cajas-S, si es red Feistel, el número de rondas.

Para la creación del algoritmo es necesario conocer los conceptos primordiales de la seguridad de la información y la Criptografía, como son: qué es un criptosistema, los tipos de algoritmos que existen y sus características, la forma de procesar los datos, qué es una red Feistel, los mecanismos de seguridad, los tipos de ataques, entre otros. Dichos conceptos ayudarán a tener una mayor comprensión sobre el tema.

Antes de iniciar, se lleva a cabo un análisis de los algoritmos simétricos más usados en la actualidad, con la finalidad de tomar como base características de éstos y con ello diseñar el algoritmo. Por otro lado, se sabe que las características básicas para realizar un cifrado simétrico, son la clave que a su vez con ella se realiza una planificación de claves, mientras que también se requiere del mensaje que se desea cifrar, y un número de rondas las cuales están compuestas por operaciones en las que interactúa la clave con el mensaje.

Al hablar de las operaciones que se proponen en el algoritmo, se toma como referencia la ciencia matemática utilizada desde hace años, puesto que para poder realizar un buen diseño de un algoritmo es conveniente basarse en: la teoría de la información, teoría de números, estadística. El no usar dicho conocimiento desemboca en un algoritmo de cifrado clásico cuya seguridad es débil, pues puede ser comprometido con un simple análisis de frecuencias. Partiendo de esta información y con base en el análisis previo de los algoritmos, se presentan las bases matemáticas de estructuras algebraicas, como la suma modular, operación XOR, y los números pseudoaleatorios ya que serán usadas para hacer interactuar el mensaje con la clave.

Una vez que se cuenta con la estructura del algoritmo se verifica que su funcionamiento para cifrar y descifrar información sea correcto; por otro lado cabe mencionar que así como los criptógrafos han buscado métodos o herramientas para proteger la información, los criptoanalistas han diseñado mecanismos para romper con la seguridad de éstos; algunos son el criptoanálisis diferencial, lineal o byclique.

Es sabido que no existe la seguridad al 100%, sin embargo, al diseñar un algoritmo se desea que brinde un mayor porcentaje de seguridad a la información, es por ello que al algoritmo diseñado se le realizarán pruebas de criptoanálisis diferencial con la finalidad de saber si es resistente ante este ataque. La razón de elegir el criptoanálisis diferencial es porque se trata de uno de los ataques con texto en claro escogido con el que son probados los algoritmos de cifrado y es el mínimo indispensable que debe ser superado.

La finalidad de diseñar un algoritmo simétrico además de brindar confidencialidad a la información, es brindar a los alumnos de la Facultad de Ingeniería los conocimientos básicos para diseñar un algoritmo de cifrado, y así mismo conocer dónde se encuentra la robustez de estos. Así mismo, al contar con las bases para el diseño de un algoritmo criptográfico el alumno tiene la oportunidad de darle seguimiento a éste proyecto, o puede diseñar un mejor algoritmo.

OBJETIVO

Objetivo general

- Diseñar un algoritmo simétrico para el cifrado de información.

Objetivos específicos

- Analizar las características de diseño de los algoritmos simétricos de cifrado que existen.
- Estudiar el criptoanálisis diferencial para evaluar la fortaleza del algoritmo diseñado contra este tipo de ataque.

PLANTEAMIENTO DEL PROBLEMA

Con anterioridad la seguridad de la información se realizaba a través de medios físicos, por ejemplo una caja fuerte, en la cual las personas resguardaban objetos o información valiosa para ellos. Con la introducción de las computadoras, y el uso de internet, los usuarios comienzan a almacenar información importante dentro de sus equipos por lo que se considera indispensable el uso de herramientas que brinden protección a la información que se encuentra resguardada dentro de éstas.

Los usuarios informáticos suelen enviar información por la red, en las empresas al realizar trámites, o enviar informes a socios, jefes, entre otros. Esta información corre riesgo de perder su confidencialidad. En este punto es donde la Criptografía entra en escena.

La Criptografía permite ocultar la información, evitando que sea comprendida por cualquier persona que no posea la clave para descifrarla.

JUSTIFICACIÓN

Como se menciona en el planteamiento del problema, el resguardo de información siempre ha sido importante para los seres humanos, con la diferencia que anteriormente no se contaba con los avances tecnológicos que tenemos actualmente, puesto que con la introducción de internet y el uso de dispositivos electrónicos, se tiene mayor riesgo de la información que en ellos se maneja.

Conociendo la importancia de la seguridad de la información, y haciendo uso de la Criptografía, como posible solución al problema se plantea diseñar un algoritmo criptográfico. Esto, con el objetivo de brindar seguridad, por medio del cifrado de la información y evitar que un intruso pueda tener acceso a ésta.

La fortaleza del algoritmo se examinará desde el punto de vista de la matemática utilizada y midiendo su capacidad de resistencia ante ataques de texto en claro elegido (criptoanálisis diferencial[1]). Como se hace mención en la introducción es indispensable para un buen diseño del algoritmo usar como base los algoritmos que ya existen.[2]

En cuanto a la estructura del trabajo, se divide en 5 capítulos: el primero de ellos abarca los antecedentes de la criptografía, definición, clasificación, y conceptos afines de la misma. También, se indica qué es un criptosistema y los elementos que lo integran, los tipos de ataques, criptoanálisis diferencial; con la finalidad de que el lector comprenda mejor el contenido del texto.

El segundo capítulo abarca conceptos matemáticos, que comprenden algebra booleana, operaciones lógicas, números aleatorios y pseudoaleatorios, así mismo algoritmos para generarlos, cajas-S, se define la estructura que toma el algoritmo

[1] http://www.cosic.esat.kuleuven.be/publications/thesis-139.pdf. (visitada el 4 de Junio del 2012)

[2] http://www.cosic.esat.kuleuven.be/publications/thesis-139.pdf. (visitada el 4 de Junio del 2012)

criptográfico simétrico que se diseñó, y su funcionamiento. Finalmente se muestra un ejemplo sencillo, para que el lector tenga una mayor comprensión del mismo.

En el capitulo tres, se justifica cada una de las decisiones tomadas para el diseño del algoritmo, decisiones como son: el tamaño de la clave, generación de cajas-S, elección de operaciones, elección del tamaño de los bloques.

En el cuarto capítulo, se realizan las pruebas que se hacen sobre el algoritmo, tomando como base el criptoanálisis diferencial, se muestra un ejemplo explicando el procedimiento de la prueba, concluyendo si es o no resistente al menos para este tipo de ataque.

Finalmente se muestran las conclusiones a las que se llegaron al realizar el diseño y pruebas del algoritmo, conociendo si realmente se logro el objetivo deseado y proponer mejoras al mismo, por si alguien a futuro desea continuar con el desarrollo del trabajo presentado.

CAPÍTULO 1

SEGURIDAD DE LA INFORMACIÓN

Y

CRIPTOGRAFÍA

SEGURIDAD DE LA INFORMACIÓN Y CRIPTOGRAFÍA

1.1 Seguridad de la información

Hoy en día la información se ha convertido en un activo vital para el éxito y la continuidad en el mercado de cualquier organización. Por tanto el aseguramiento de dicha información y de los sistemas que la procesan es, un objetivo de primer nivel para las organizaciones.

1.1.1 Definición de la seguridad de la información

La seguridad de la información es un proceso de medidas preventivas y reactivas encargado del cuidado de la disponibilidad, confidencialidad e integridad de la información.

De acuerdo con un artículo de la Revista Digital Universitaria, se puede hablar de la Seguridad de la Información como el conjunto de reglas, planes y acciones que permiten asegurar la información manteniendo las propiedades de confidencialidad, integridad y disponibilidad de la misma.

Andrés Cargill, de Soluciones Orión, una de las empresas especialistas en seguridad y tecnologías de la información dice: "La seguridad informática contempla no sólo la protección de la infraestructura y los dispositivos, sino también de la información que éstos almacenan, y la integridad física y moral de los usuarios que la proveen. Para ello, se han creado diversos mecanismos, como la encriptación de datos, la creación de firewalls, detectores de hackers, simuladores de ataques informáticos, por nombrar algunos."[3]

[3] http://tecnoadmin.americaeconomia.com/noticias/seguridad-informatica-confidencialidad-integridad-y-disponibilidad-de-la-informacion. (visitada el 4 de Junio del 2012)

Con la seguridad de la información se busca la preservación de la misma frente a observadores no autorizados. Para ello podemos emplear tanto Criptografía simétrica como asimétrica, estando la primera únicamente indicada en sistemas aislados, ya que si la empleáramos en redes, al tener que transmitir la clave por el canal de comunicación, estaríamos asumiendo un riesgo excesivo.

1.1.2 Importancia de la seguridad de la información

La información es un recurso o activo que, como otros recursos importantes del negocio, es esencial a una organización y a su operación y por consiguiente necesita ser protegido adecuadamente. Esto es especialmente importante en el ambiente comercial cada vez más interconectado. Como resultado de esta ínter conectividad creciente, la información se expone ahora a un número ascendente y a una variedad más amplia de de amenazas y vulnerabilidades.

La información puede ser presentada de muchas formas, por ejemplo, impresa, electrónicamente, transferirse por correo electrónico, ser escrita sobre un papel, hablada, etc., cualquier forma que la información tome, o cualquier medio por donde sea compartida o guardada, siempre debe ser protegida apropiadamente.

En esta parte es donde la seguridad de la información toma un papel importante ya que es la protección de información de una gama amplia de amenazas para asegurar la continuidad comercial, minimizar el riesgo comercial, y aumentar al máximo el retorno en las inversiones y las oportunidades de negocios.

La seguridad de la información se logra llevando a cabo un conjunto conveniente de controles, incluyendo las políticas, los procesos, procedimientos, y funciones del hardware y software.

1.1.3 Servicios de seguridad

Un servicio de seguridad es aquel que mejora la seguridad de un sistema de información y el flujo de la información de una organización.

Los servicios están dirigidos a evitar los ataques de seguridad y para ello se requiere utilizar uno o más mecanismos de seguridad con la finalidad de proveer el servicio requerido, así, los servicios de seguridad son:

a) Confidencialidad: es la capacidad de asegurar que solo las personas autorizadas tienen acceso a la información o recurso en cuestión. La confidencialidad se aplica a todos los datos intercambiados por las personas autorizadas, o bien a segmentos seleccionados de información y recursos dependiendo de los privilegios de la persona.

b) Autenticación: es la identificación ante un sistema, subsistema, red o aplicación, mediante algún mecanismo o combinación de mecanismos; se utiliza para asegurar que las partes involucradas son quiénes dicen ser. Se distinguen dos tipos: de entidad, que asegura la identidad de las entidades participantes en la comunicación, mediante biométrica (huellas dactilares, identificación de iris, etc.), tarjetas de banda magnética, contraseñas (véase la figura 1.1), o procedimientos similares; y de origen de información, que asegura que una unidad de información proviene de cierta entidad, siendo la firma digital el mecanismo más extendido.

Figura 1.1: Autenticación por contraseña

c) **Integridad**: es la capacidad de evitar que los datos sean modificados por usuarios o procesos no autorizados para ello, y con esta finalidad se pueden requerir los servicios de integridad de contenido y de integridad de la secuencia del mensaje. La modificación incluye escritura, cambio, borrado, creación y re actuación de la información procesada.

Para garantizar la integridad de la información, el remitente debe estar siempre autenticado. La combinación de autenticación e integridad da certeza que la información enviada llega a su destinatario exactamente en la misma forma en que se envió.

d) **No repudio**: es la protección que se ofrece para prevenir a los emisores o receptores de negar un mensaje transmitido, de manera que entre los diferentes requerimientos que se pueden presentar están: no repudio de origen, no repudio de envió, no repudio de presentación, no repudio de transporte y no repudio de recepción.

e) **Control de acceso**: se refiere al hecho de que se restringe, discrimina y controla el acceso de los usuarios a la información. El control del acceso verifica si el usuario tiene derecho a acceder al servicio y la información, lo que significa que sólo las personas autenticadas y con los privilegios debidos pueden obtener acceso. Algunos ejemplos de control de acceso puede ser huella digital (véase figura 1.2), reconocimiento de iris, etc.

Figura 1.2: Acceso por huella digital

f) **Disponibilidad**: se refiere al hecho de que los usuarios que necesitan la información y a quiénes va dirigida dicha información siempre tienen acceso a ella. Los métodos para garantizar la disponibilidad incluyen un control físico y técnico de las funciones de los sistemas de datos, así como la protección de los archivos, su correcto almacenamiento y la realización de copias de seguridad.

1.1.4 Mecanismos de seguridad

No existe un único mecanismo que sea capaz de proveer todos los servicios de seguridad antes mencionados, pero la mayoría de ellos hacen uso de técnicas criptográficas basadas en el cifrado de la información, y entre los más destacados se encuentran los siguientes:

a) **Intercambio de autenticación:** corrobora que una entidad, ya sea origen o destino de la información, es la deseada. Por ejemplo, A envía un mensaje cifrado con la clave publica de B, B lo descifra con su clave privada y se lo reenvía a A, demostrando así que es quien pretende ser.

b) **Cifrado:** garantiza que la información no es inteligible para individuos, entidades o procesos no autorizados a través de lo cual proporciona confidencialidad a la información, consiste en transformar un texto en claro mediante un proceso de cifrado en un texto cifrado, gracias a una clave de cifrado.

c) **Integridad de datos:** este mecanismo implica el cifrado de una cadena comprimida de datos a transmitir, llamada generalmente valor de comprobación de integridad. Este mensaje se envía al receptor junto con los datos ordinarios. El receptor repite la compresión y el cifrado posterior de los datos y compara el resultado obtenido con el que le llega, para verificar que los datos no han sido modificados.

d) **Firma digital:** este mecanismo implica el cifrado, por medio de una clave secreta del emisor, de una cadena comprimida de datos que se va a transferir. La firma digital se envía junto con los datos ordinarios, de manera que el mensaje se procesa en el receptor, para verificar su integridad. Así, este mecanismo juega un papel esencial en los servicios de integridad, no repudio e inclusive confidencialidad.

e) **Control de acceso:** esfuerzo para que solo aquellos usuarios autorizados accedan a los recursos del sistema o a la red, como por ejemplo mediante las contraseñas de acceso.

f) **Tráfico de relleno:** consiste en enviar trafico espurio junto con los datos validos para que el atacante no sepa si está enviando información, ni qué cantidad de datos útiles se está transmitiendo.

g) **Control de encaminamiento:** permite enviar determinada información por determinadas zonas consideradas clasificadas. Asimismo, posibilita solicitar otras rutas, en caso que se detecten persistentes violaciones de integridad en una ruta determinada.

h) **Unicidad:** consiste en añadir a los datos un número de secuencia, la fecha y la hora, un numero aleatorio, o alguna combinación de los anteriores, que se incluyen en la firma digital o integridad de datos, de esta forma se evitan amenazas como la reactuación de mensajes.

Para mayor información sobre los servicios y mecanismos de seguridad, el documento ISO 7498-2, consta de una segunda parte en la que se refiere a la arquitectura de seguridad, la cual fue publicada en 1988 en ella se define un servicio de seguridad como el servicio proporcionado por un nivel de un sistema abierto que garantiza la seguridad de los sistemas abiertos o a las transferencias de datos en dichos sistemas.

El documento ISO 7498-2 da una descripción general de los servicios y mecanismos relacionados con la seguridad y sus interrelaciones.[4]

1.2 Criptografía

Para establecer una comunicación de datos entre dos entidades (personas, equipos informáticos, y demás) hacen falta al menos tres elementos básicos: el emisor del mensaje, el receptor del mismo y un medio por el cual se transfieran los datos (véase figura 1.3).

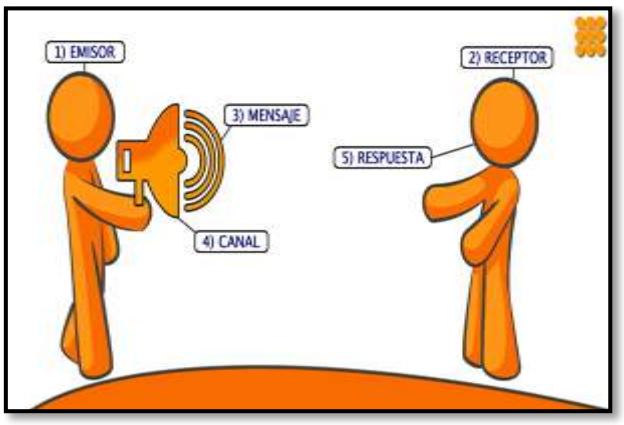

Figura 1.3: Medio de comunicación

En una comunicación normal los datos se envían a través del medio tal como son, sin sufrir modificaciones de ningún tipo, de tal forma que el mensaje que representan puede ser interceptado y leído por cualquier otra entidad que acceda a él durante su viaje por el medio.

Pero hay ocasiones en las que interesa que dicho mensaje solamente pueda ser interpretado correctamente por el emisor del mismo y por el receptor al que va dirigido. En estas ocasiones es necesario implementar algún mecanismo de protección de la información sensible tal que el mensaje viaje seguro desde la fuente al destino, siendo imposible la interceptación por terceros, o que si se produce ésta, el mensaje capturado sea incomprensible para quien tenga acceso al mismo.

[4] http://es.scribd.com/doc/68137528/Arquitectura-ISO-7498-2. (visitada el 14 de Enero del 2012)

Una de las formas de conseguir esto es enviar el mensaje en claro, tal como lo ha redactado el emisor, y protegerlo en el camino mediante sistemas de fuerza; como es el caso de la protección de mensajes mediante personal de seguridad. Otro método posible es el enviar el mensaje por un camino con tanto tráfico de información que resulte muy difícil a las terceras personas detectar que se trata de información confidencial; como es el caso de enviar el mensaje mediante una carta por el sistema estándar de correo.

Desafortunadamente estos métodos de protección de mensajes, al igual que otros análogos, han demostrado su inefectividad a lo largo de los tiempos, por lo que hubo que buscar otro tipo de mecanismos para proteger la información sensible en su camino entre emisor y receptor.

La Criptografía ha demostrado con el tiempo ser una de las mejores técnicas para resolver esta cuestión. Tanto es así que actualmente es el mecanismo más usado en los procesos de protección de datos, como las transacciones bancarias por Internet, el correo electrónico cifrado, etc.

Esto es así porque es tal vez el único medio accesible y fácil de implementar para lograr un acceso controlado a la información en un medio, que por su propia naturaleza es abierto y de acceso libre a la información.

1.2.1 Conceptos

De acuerdo con el Diccionario de la Real Academia, la palabra Criptografía proviene en un sentido etimológico del griego

Kryptós, criptos "ocultar", Graphos, graphé "escritura".

La Criptografía es la ciencia encargada de convertir un texto en claro, en otro llamado criptograma cuyo contenido de información es igual al anterior pero sólo lo pueden entender las personas autorizadas.

La Criptografía es un conglomerado de ciencias y disciplinas, que tratan sobre la protección y ocultamiento frente a observadores no autorizados de la información. Entre las disciplinas que engloba cabe destacar la Teoría de la Información, la Teoría de

Números, Matemática Discreta, Álgebra, Álgebra Lineal, Cálculo, Estadística y más recientemente Geometría Analítica, Teoría de Grafos, y Cálculo Vectorial y la Complejidad Algorítmica.

Criptografía es el estudio de las técnicas matemáticas relativo a los aspectos de seguridad de la información, tales técnicas abarcan: confidencialidad, integridad de los datos, autenticación de la entidad, y autenticación del origen de los datos. Sin embargo la Criptografía no pretende proveer los medios para asegurar la información, sino ofrecer las técnicas para lograr este aseguramiento.

1.2.2 Objetivo de la Criptografía

Aunque el objetivo original de la Criptografía era mantener en secreto un mensaje, en la actualidad no se persigue únicamente la privacidad o confidencialidad de los datos, sino que se busca además garantizar la autenticación de los mismos (el emisor del mensaje es quién dice ser), su integridad (el mensaje no ha sido modificado por entidades no autorizadas) y el no repudio (mediante la firma digital el emisor no puede negar que el envió el mensaje).

1.2.3 Antecedentes de la Criptografía

La Criptografía es tan antigua como la escritura, puesto que desde hace miles de años, los seres humanos se han visto en la necesidad de ocultar información de carácter privado, a fin de mantenerla resguardada o a salvo de intrusos que pudiera hacer mal uso de ella.

A continuación se remontará a siglos pasados haciendo un viaje a través de la historia hasta nuestros días sobre la Criptografía; con esto nos daremos cuenta que han existido diferentes técnicas, métodos, y artilugios utilizados por sacerdotes, emperadores, gobernadores, militares y en general diversas civilizaciones, con la finalidad de mantener su información a salvo.

Remontándonos a los antiguos egipcios encontramos que ellos hicieron uso de métodos criptográficos, y mientras el pueblo utilizaba la lengua demótica, los sacerdotes

usaban la escritura hierática o jeroglífica para comunicarse entre ellos ya que eran incomprensibles para el resto de la gente.

Es increíble que el texto conocido más viejo que contiene uno de los componentes esenciales de la Criptografía (véase figura 1.4), ocurrió hace unos cuatro mil años atrás en el pueblo egipcio de Menet Khufu, donde las inscripciones jeroglíficas en la tumba del noble Khnumhotep II habían sido escritas con un número inusual de símbolos para confundir y oscurecer el significado de las inscripciones.

Figura 1.4: Texto egipcio

En el siglo V a.C. durante la guerra entre Atenas y Esparta, los espartanos, una sociedad guerrera famosa por su severo estilo de vida, desarrollaron un dispositivo criptográfico para enviar y recibir mensajes secretos.

Este dispositivo, un cilindro llamado Scítala (véase figura 1.5), era posesión tanto del emisor como del receptor del mensaje. Para preparar el mensaje, se usa una estrecha franja de pergamino o cuero, el cual se enrolla alrededor de la Scítala y el mensaje se escribe a través de esta. Luego se desenrolla el papel, y es llevado al receptor, la cinta muestra sólo un mensaje que tiene un significado que no es entendible sólo hasta que se vuelva a enrollar alrededor de una Scitala exactamente del mismo diámetro. El código, es un cifrado de Transposición, donde las letras siguen siendo las mismas, pero cambian el orden de su posición. Este método aún es la base de variadas técnicas que se usan hoy en día.

Figura 1.5: La Scítala

En el siglo II a.C. el escritor griego Polybios (203 a.C. – 118 a.C.), inventó un sistema que acabó siendo adoptado muy a menudo como método criptográfico. Colocó las letras del alfabeto en una red cuadrada de 5x5 (véase figura 1.6), el sistema de cifrado consistía en hacer corresponder a cada letra del alfabeto un par de letras que indicaban la fila y la columna (coordenadas de letras o números), en la cual dicha letra estaba ubicada.

	A	B	C	D	E
A	A	B	C	D	E
B	F	G	H	IJ	K
C	L	M	N	O	P
D	Q	R	S	T	U
E	V	W	X	Y	Z

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	IJ	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Figura 1.6: Tabla de cifrado Polybios

Ej.: HOLA = BC CD CA AA = 23 34 31 11

En el siglo I a.C. surge el cifrado César, el cual se considera que fue utilizado por Julio César (101 a.C. – 43 a.C.), aún cuando hay algunos historiadores que indican que éste nunca lo utilizó directamente, pero que fue utilizado por otros emperadores romanos de la época. El cifrado consiste en mover el carácter a representar 3 posiciones adelante dentro del alfabeto a utilizar (véase imagen 1.7).

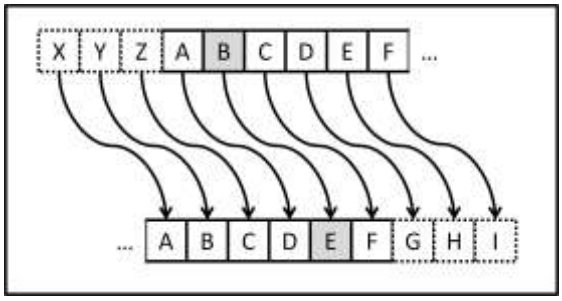

Figura 1.7: Cifrado César

Ej. Hola= krod

En nuestra era:

En la Edad Media tiene un resurgimiento la Criptografía, puesto que es un periodo bástate extenso de la historia europea que va desde la desintegración del imperio romano en el siglo V, pasando por las cruzadas (1095-1270), hasta el siglo XV aproximadamente, donde su uso se vio principalmente impulsado por las intrigas del papado y las ciudades-estado italianas. Por ejemplo, San Bernardino escribía mensajes secretos de tal manera que evitaba la regularidad de los signos sustituyendo letras por varios signos distintos, así, tenía un símbolo para cada consonante, cabe mencionar que usaba tres signos distintos para cada una de las vocales además de incluir símbolos sin valor (véase la figura 1.8).

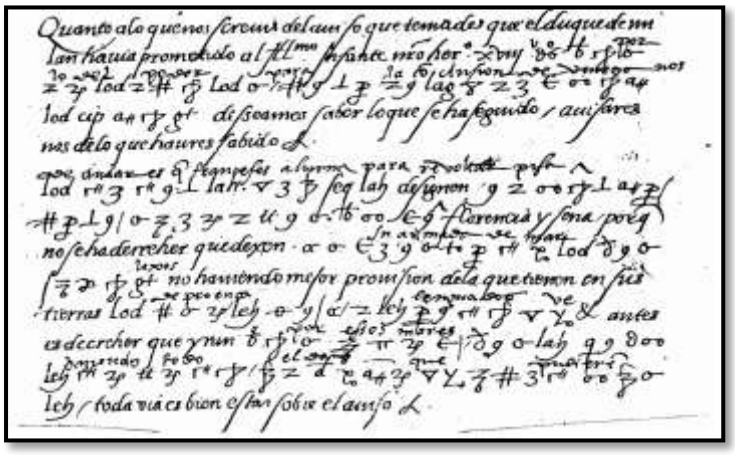

Figura 1.8: Fragmento de misiva cifrada enviada por Carlos V al embajador Lope de Soria en 1523. BRAH, 9/1951

En el Renacimiento, hacia 1466 Leon Batista Alberti (1404 - 1472), considerado el padre de la Criptografía, concibió el primer sistema polialfabético; este mecanismo emplea varios alfabetos, saltando de uno u otro cada tres o cuatro palabras. El emisor y el destinatario deben de ponerse de acuerdo para fijar la posición relativa de dos círculos concéntricos, que determinarán la correspondencia de los signos. Los diferentes alfabetos utilizados eran representados en uno de los discos, mientras que el otro se rellenaba con el abecedario "normal", más los números del 1 al 4. Este disco define 24 posibles sustituciones dependiendo de la posición del disco interior (véase figura 1.9).

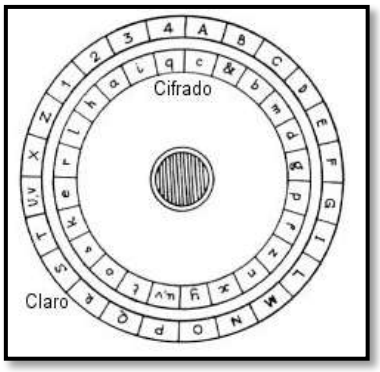

Figura 1.9: Disco de Alberti

Hacia 1535, el disco de Alberti es modificado por Giovanni Battista De la Porta cambiando uno de los alfabetos por una serie de misteriosos símbolos (véase figura 1.10).

Figura 1.10; Disco de Porta

Dentro del periodo renacentista italiano, otros estudios de la criptografía también hicieron contribuciones importantes, como Gerolamo Cardano (1501- 1576) nacido en Pavia, en el campo de la Criptografía aportó en 1950 un sistema basad en una carta o tarjeta con agujeros perforados, de tal manera que el mensaje en claro se obtenía al colocarlo sobre un determinado texto preconcebido, en su momento se conoció como reja de Cardano, nombre que fue modificado y se denomino "mascaras rotativas".

Es así, como en el siglo XVI se generaliza el uso de la Criptografía en los ambientes diplomáticos y para 1586 Blaise de Vigenére (1523 - 1596) diplomático, criptógrafo y químico francés, publica una obra que describe a detalle el cifrado polialfabético desarrollado por Geovanni Battista, motivo por el cual le es otorgado erróneamente a dicho algoritmo el nombre de cifrado de Vigenére.

La rueda de Jefferson (véase figura 1.11), llamada así en honor a su creador Thomas Jefferson (1743 – 1826), es una maquina conformada por una serie de discos que giran libremente alrededor de un mismo eje y llevan impresas las letras del alfabeto escritas en cada discos en diferente orden, de manera que el emisor mueve los discos hasta configurar el mensaje en claro y elige otra línea con la que procesa el mensaje cifrado.

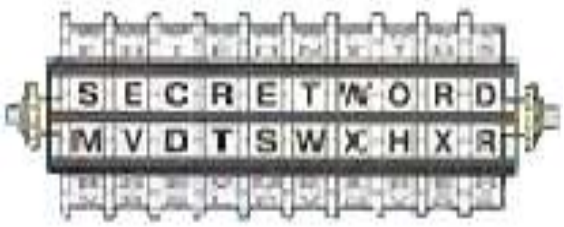

Figura 1.11: Rueda de Jefferson

A mediados del siglo XIX, surge el Disco de Wheatstone (1802 - 1875) el cual realiza una sustitución muy parecida a la ideada por Alberti y cuyo procedimiento de cifrado está basado en el cifrado de Polybios; este dispositivo opera mediante dos discos concéntricos, en el disco exterior se escriben en orden alfabético las 26 letras del alfabeto inglés mas el espacio denotado pro el símbolo +, y en el disco interior se

distribuyen las letras del alfabeto de manera aleatoria, sobre los discos hay dos agujas, y en tanto la mayor se avanza por el disco exterior, la otra se desplaza por el disco interior, de manera que cuando la manecilla mayor ha dado un giro completo la menor hadado una vuelta más una letra siendo esta la correspondiente al cifrado (véase figura 1.12). Wheatstone llamo a su desarrollo "cifrado de Playfair" en honor a su amigo Lord Playfair.

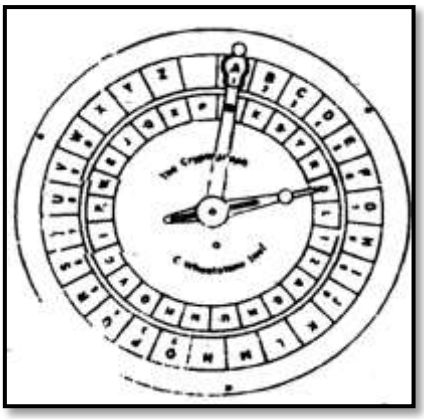

Figura 1.12: Disco de Wheatstone

En 1883, el Dr. Auguste Kerckhoffs (1835 - 1903), lingüística y criptógrafo de origen holandés, publica en la gaceta de Ciencias Militares de París el artículo titulado "La criptografía militar", a través del cual se da a conocer una serie de prácticas y reglas que deben cumplir los sistemas de cifrado y su principio de seguridad, el cual dice:

"La seguridad de un criptosistema se mide suponiendo que el enemigo conoce completamente ambos procesos: cifrado y descifrado".

En este siglo fue tanta la difusión del uso de la criptografía para resguardar secretos que inclusive se encuentra recreado este ambiente en la literatura de ficción en el cuento "The Gold Bug" escrito en 1843 por Edgar Allan Poe (1809 - 1849).

Para el siglo XX la Criptografía juega un papel importante ya que el acontecimiento de la Primera y Segunda guerras mundiales hacen necesaria su utilización en los sistemas de comunicaciones para realizar transmisiones de mensajes secretos, lo que impulso de manera importante el desarrollo tanto de las técnicas como de las maquinas de cifrado.

El método de Vigenére fue llevado hasta su ultima extensión lógica muchos años más tarde, por un criptógrafo americano, el Ingeniero Gilbert Sandford Vernam, quien demostró que para que el cifrado de Vigenére fuera seguro no solamente era necesario que la clave de cifrado fuese más larga que el mensaje, sino que además esta debería ser utilizada una sola vez y que para el procesamiento de los datos era necesario considerarse no solo cada carácter del mensaje sino su representación codificada. Así, Vernam da a conocer el algoritmo desarrollado y publicado en julio de 1919 como la función de Vernam.

También en 1919 se registra la primer patente de una maquina criptográfica, la cual corresponde a una maquina llamada *Enigma* (véase figura 1.13), obra del holandés Alexander Koch y el alemán Arthur Scherbius, este ultimo realizo varias versiones de *Enigma* junto con Richard Ritter, y conjuntamente fundaron en Berlín la compañía Chiffriermaschine Aktien Gersellschaft para llevar a cabo la producción comercial de la maquina. Así, la primera versión comercial fue puesta en venta en 1923 y se llamo *Enigma-A.*

Figura 1.13: Maquina Enigma

Enigma tenía un aspecto exterior de una máquina de escribir muy voluminosa, la cual podía funcionar en dos modos; uno era el modo de operar “normal”, esto es, cuando se oprimía la tecla correspondiente a una A se imprimía una A, en tanto que el otro modo de operación era el de cifrado y operaba de la siguiente manera: se hacía pasar corriente eléctrica a través de un cierto mecanismo, de manera que la letra que se imprimía era ya

el resultado de un complejo sistema de cifrado, en el que había una serie de ruedas colocadas tocándose por sus caras y formando un cilindro.

En el modo de cifrado, al oprimir una tecla la corriente llegaba a la primera rueda. Las ruedas tenían contactos eléctricos delante y detrás, y en su interior estaban conectados los de adelante con los de atrás con base en un patrón previamente establecido; en total, cada rueda contaba con 28 contactos en cada cara, una por cada tecla, así, en el interior de la primera rueda, la letra original era transformada en otra siguiendo el patrón fijado por el cableado al activar el contacto correspondiente en la salida trasera.

Cabe mencionar que la existencia de Enigma y el hecho de que los aliados conocieran sus secretos, fueron durante mucho tiempo, dos de los secretos mejor guardados de la II Guerra Mundial, tal vez porque de esta forma era posible seguir sacándole partido tras la guerra, potenciado su uso en diversos países, que, al instalarla, hacían transparentes sus secretos. Y tras la conclusión de la II Guerra Mundial, la criptografía tiene un desarrollo teórico importante, siendo Claude Elwood Shannon (1916 - 2001) y sus investigaciones sobre teoría de la información, esencial en dicho desarrollo.

Shannon trabajó arduamente en todo aquello referente a la teoría de la información, esto es, la rama de la teoría matemática de la probabilidad y la estadística que estudia la información y todo lo relacionado con ella, de hecho, se trata de la rama de las matemáticas inaugurada por Shannon donde se estudia entre otros aspectos la compresión de datos canales de comunicación y la criptografía.

En 1948 publicó el trabajo más importante de su carrera, desarrolló un método para expresar la información de forma cualitativa. Dicho trabajo es su Teoría Matemática de la Comunicación, el cual ha sido calificado como la carta magna de la era de la información (A Mathematical Theory of Communication, Bell System Technical Journal, Vol. 27, julio y octubre 1948).

Las publicaciones de Shannon en 1949 demostraron cómo se podía analizar la cuantificación de la información mediante métodos estrictamente matemáticos, y desde el estudio del algebra booleana desarrollo la teoría del código binario, esto es la base digital a partir de unidades básicas de información definidas por dos estados conocidos como “0” y “1”, los cuales aparecen como el átomo de la información y la base constructiva del mensaje; de manera que una información compleja es una sucesión de unos y ceros, de hecho, la información así tratada adquiere una dimensión física, cuantificable y mesurable, independientemente del contenido , de los emisores y receptores.

Así, la base matemática de la teoría radica en su cuantificación, en la descripción del concepto técnico de canal, en la codificación y decodificación de las señales, que hace posible medir la verosimilitud de información fraccionada por perdida de bits, distorsión de los mismos, adición de elementos extraños, etc., y hablar con precisión de términos antes vagos, como redundancia, ruido e incluso, expresar el concepto físico de entropía como n proceso continuado de la perdida de información.

Como se ha observado, el mayor desarrollo de la Criptografía se da en el siglo XX, destacándose el mayor auge en las dos guerras mundiales para establecer comunicaciones secretas militares y diplomáticas, creándose fascinantes maquinas de cifrar, que adquirieron gran fama tras su uso en la Segunda Guerra Mundial. Finalizada la contienda, las nuevas tecnologías electrónicas y digitales se adaptaron a las maquinas criptográficas, dándose así, los primeros pasos hacia los sistemas criptográficos más modernos, y mucho más fiables que la sustitución y transposición clásicas.

Un hecho significativo marca la distinción de dos grandes eras en el mundo de la Criptografía:

Los estudios que realizó Claude Shannon en 1948 y que propiciaron las bases para desarrollar posteriormente el sistema criptográfico DES (Data Encryption Standard). Ya que a partir de ese año se considera la criptologia como una ciencia aplicada sobre teoría de la información y desde ese momento deja de ser considerada solo como un arte rodeado de misterios y escepticismo (*la era de la criptografía clásica*), para ser tratada

como una rama mas de las matemáticas y en nuestros días jugar un papel fundamental en la información y las ciencias de la ingeniería (*la era de la criptografía moderna*).

Es a mediados de la década de los 70 cuando se registran dos avances de suma importancia en el mundo de la criptografía moderna:

El primero se refiere precisamente al desarrollo del algoritmo DES por parte de IBM, el cual fue publicado por el NBS (National Bureau of Standards) ahora NIST (National Institute of Standards and Technology) en calidad de borrador en marzo de 1975 y en 1977 como un Estándar de Procesamiento de Información Federal (actualmente el FIPS PUB 46).

DES fue desarrollado por IBM en respuesta a una invitación hecha por el NBS, oficina interesada en promover el desarrollo de sistemas de comunicaciones electrónicas seguras para empresas, bancos y demás organismos financieros principalmente, y se trata del primer algoritmo de cifrado accesible de manera pública y respaldado por la NSA (National Standards Agency), el cual, después de su publicación estimulo el interés público y académico por la Criptografía.

El segundo y tal vez más importante aún, debido a que cambio de manera fundamental la forma en la que los criptosistemas pueden funcionar, se dio en 1976 y se refiere al desarrollo del algoritmo Diffie y Hellman.

Diffie y Hellman quienes describen que los procedimientos de clave pública son teóricamente posibles, a pesar de que se ha intentado demostrar lo contrario. En 1976 Whitfield Diffie y Martin Hellman publican "New Directions in Cryptography" [5] , que introduce un nuevo método de distribución de claves criptográficas para ser usadas en los sistemas de cifrado, pudiendo realizar dicho intercambio de manera segura a través de canales inseguros, lo que era hasta la fecha uno de los problemas fundamentales de la Criptografía. Este mecanismo será conocido como el protocolo Diffie-Hellman de intercambio de claves, sentando así las bases de una criptografía asimétrica.

[5] http://www.cs.jhu.edu/~rubin/courses/sp03/papers/diffie.hellman.pdf. (visitada el 13 de Enero del 2012)

Para 1977, esto es, tan solo un año después de Diffie y Hellman, se da a conocer un nuevo y poderoso algoritmo de cifrado asimétrico, desarrollado en el MIT y nombrado RSA por ser las iniciales de los apellidos de sus creadores, Ronald Rivest, Adi Shamir y Leonard Adleman. Habiendo desarrollado su efectividad y robustez, RSA es el algoritmo criptográfico más ampliamente conocido, difundido y utilizado a la fecha.

En 1982 El físico Richard Feynman diseña el modelo teórico de una computadora cuántica, y 1984 Charles H. Bennett y Gilles Brassard describen la Criptografía cuántica (BB84 protocol).

En 1986 De forma independiente, Neal Koblitz y Victor Miller proponen usar curvas elípticas como modelo de Criptografía de clave pública.

En 1991 Xueija Lai y James Massey desarrollan el algoritmo IDEA en Suiza, que será usado en el software criptográfico PGP.

En 1991 DSA es elegido por el NIST como algoritmo estándar de firma digital. 1991 PGP (Pretty Good Privacy) es diseñado por Phil Zimmermann como un software gratuito y de código libre, con el fin de cifrar e intercambiar archivos con una gran seguridad. Esta es la primera vez que el cifrado híbrido (combinación de Criptografía simétrica y asimétrica) es aplicado a un programa popular para usuarios finales. El objetivo principal era el de cifrar los archivos adjuntos del correo electrónico (que más tarde también fue cubierto por el estándar S/MIME).

Hacia finales del 2001 el algoritmo Rijndael es elegido por el NIST como el sucesor de DES y pasa a denominarse AES (Advanced Encryption Standard).

Después de este recorrido por la historia de la criptografía, puede observarse que hay dos hechos que marcan rotundamente dos puntos de inflexión en el mundo de la criptografía:

- El primero de ellos corresponde al desarrollo de la Teoría Matemática de la Comunicación por Shannon en 1948.

- El segundo es la publicación que hace en 1976 Diffie y Hellman en el que proponen una nueva forma de cifrado.

1.2.4 Criptosistema

Los sistemas capaces de transformar la información para ocultarla y salvaguardarla mediante el cifrado y descifrado de datos se denomina criptosistema o sistemas criptográficos (véase figura 1.14).

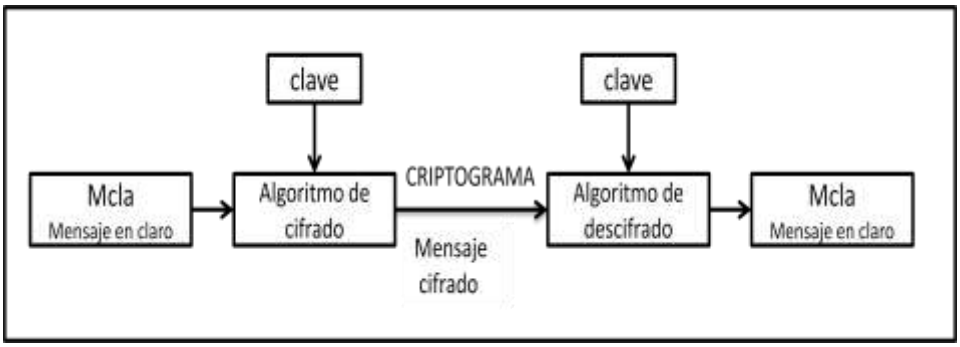

Figura 1.14: Esquema de componentes de un Criptosistema

Se define un criptosistema como una quíntupla (M; C; K; E; D), donde:

M o Mensaje en claro (Mcla): representa el conjunto de todos los mensajes sin cifrar (lo que se denomina texto claro, o plaintext) que pueden ser enviados. Son componentes de un mensaje inteligible (bits, bytes, pixeles, signos, caracteres, etc.) que provienen de un alfabeto previamente establecido.

$$M = \{m_1, m_2, m_3, m_4 \dots m_n\}$$

C representa el conjunto de todos los posibles mensajes cifrados, o criptogramas. Normalmente el alfabeto es el mismo que el utilizado para el mensaje en claro.

$$C = \{c_1, c_2, c_3, c_4 \dots c_n\}$$

K (clave o llave): representa el conjunto de claves que se pueden emplear en el criptosistema. Se supone que es un conjunto altamente aleatorio de caracteres, palabras, bits, bytes, etc. Al menos una de las claves en un criptosistema se guarda en secreto.

$$K = \{k_1, k_2, k_3, k_4 \dots k_n\}$$

E es el conjunto de transformaciones de cifrado o familia de funciones que se aplica a cada elemento de M para obtener un elemento de C. Existe una transformación diferente E_k para cada valor posible de la clave k.

$$E_k : M \rightarrow C \; con \; k \; £ \; K$$

D es el conjunto de transformaciones de descifrado, inverso a E. En este caso, D_k es una aplicación con una clave k (en el espacio de claves K) de C en M. El resultado de aplicar $D_k \;\; a \; C$ es M; por lo tanto, D_k y E_k son inversos. En ocasiones D_k es la operación inversa de E_k, pero también D_k es la misma E_k con el inverso de la clave k.

$$D_k : C \rightarrow M \; con \; k \; £ \; K$$

Todo criptosistema ha de cumplir la siguiente condición:

$$D_k \; (E_k \; (M)) = M$$

Es decir, un mensaje M se cifra empleando la clave k y luego se descifra empleando la misma clave, obteniendo de nuevo el mensaje original M (véase figura 1.15).

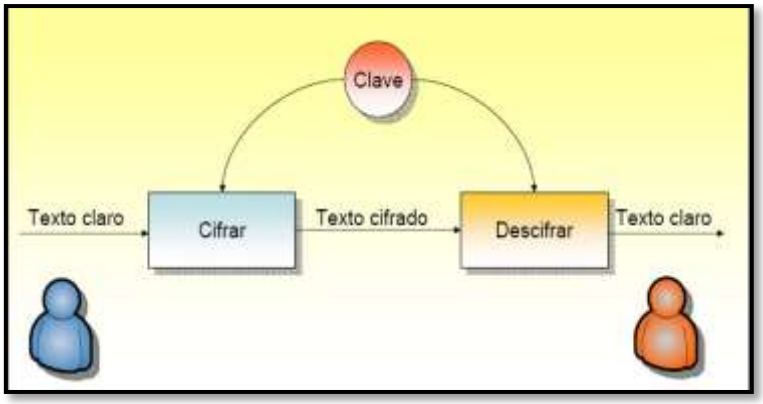

Figura 1.15: Mensaje cifrado y descifrado con misma clave

Existen dos tipos fundamentales de criptosistemas: criptosistemas simétricos o de clave privada, y los criptosistemas asimétricos o de clave pública.

La gran clasificación de los criptosistemas se hace en función de la disponibilidad de la clave de cifrado/descifrado. Existen, por tanto, dos grandes grupos de criptosistemas: los simétricos o clave privada, y los asimétricos o clave pública.

1.3 Criptosistemas simétricos o de clave secreta

La Criptografía simétrica es el sistema de Criptografía más antiguo. Se utiliza desde los tiempos antiguos hasta la actualidad.

La Criptografía simétrica se caracteriza por emplear la misma clave k tanto para cifrar como para descifrar. Presentan el inconveniente de que para ser empleados en comunicaciones la clave k debe estar tanto en el emisor como en el receptor, lo cual lleva a preguntarse cómo transmitir la clave de forma segura.

1.3.1 Características de los algoritmos simétricos

La Criptografía simétrica o de llave secreta es aquélla que utiliza algún método matemático llamado sistema de cifrado para cifrar y descifrar un mensaje utilizando únicamente una llave secreta (véase figura 1.16), se puede observar que la línea punteada es el eje de simetría: lo mismo que hay de un lado existe exactamente igual en el otro, esto ilustra el hecho del porqué se le da el nombre de Criptografía simétrica.

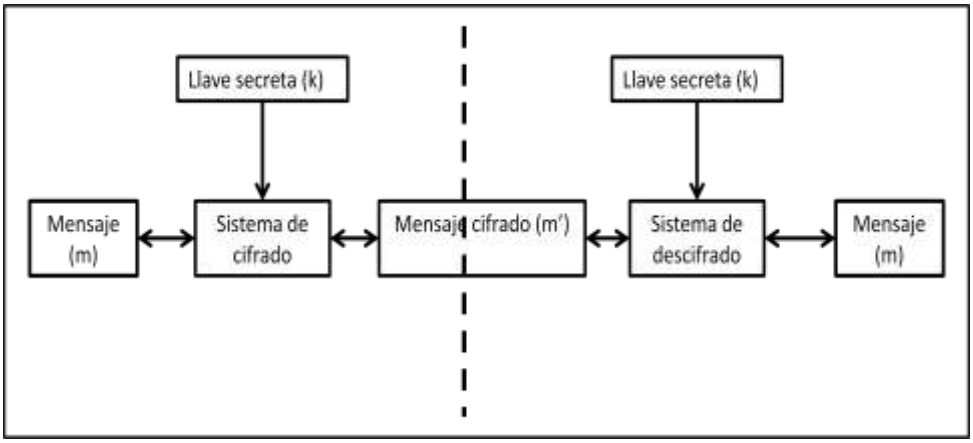

Figura 1.16: Esquema del criptograma simétrico

La llave conocida como secreta se debe de compartir entre las personas que se desea que vean los mensajes.

Con este tipo de Criptografía se garantiza la confidencialidad porque únicamente quien posea la llave secreta será capaz de ver el mensaje.

Para mantener la confidencialidad delante de un criptoanalista, el algoritmo debe cumplir las siguientes condiciones:

- Conocido el criptograma no se puede descifrar el texto ni adivinar la clave.
- Conocido el texto y el criptograma es más caro (en tiempo y/o dinero) encontrar la clave que el valor de la información**.**

Para la segunda condición siempre existe el sistema de "prueba y ensayo" para encontrar la clave, es decir, probar todas las claves posibles hasta encontrar la que descifra el criptograma. La seguridad respecto a este tipo de ataque depende de la longitud de la clave.

1.3.2 Ventajas de los criptosistemas simétricos:

- Se pueden diseñar para obtener velocidades muy elevadas. Las implementaciones en hardware alcanzan velocidades de varios GB por segundo, mientras que las implementaciones de software son capaces de procesar varios MB por segundo.
- Las claves son relativamente cortas y manejables.

- Se pueden emplear para construir otros tipos de primitivas como generadores pseudoaleatorios o funciones hash.
- Se pueden componer y combinar para formar criptosistemas más seguros,
- Son muy populares, sin duda, gracias a la extensa implantación del cifrado de bloques DES.

1.3.3 Desventajas los criptosistemas simétricos:

- La clave ha de ser mantenida en secreto por ambas entidades en una comunicación bipartita.
- En una red grande, con muchos participantes, se ha de mantener una clave distinta para cada pareja de interlocutores posible; en consecuencia, el número de claves resulta, a menudo, muy elevado y exige políticas de gestión de claves complejas.
- Para maximizar la seguridad, se ha de cambiar la clave de forma frecuente, incluso para cada sesión de comunicación distinta. Esto aumenta el problema de la distribución de claves.

1.4 Criptosistemas asimétricos o de clave pública

En la Criptografía de llave pública (véase figura 1.17), no existe simetría en ella a diferencia del cifrado de clave privada, ya que de un lado se cifra o descifra con una llave pública y en el otro lado con una privada. De este hecho es de donde la Criptografía asimétrica debe su nombre.

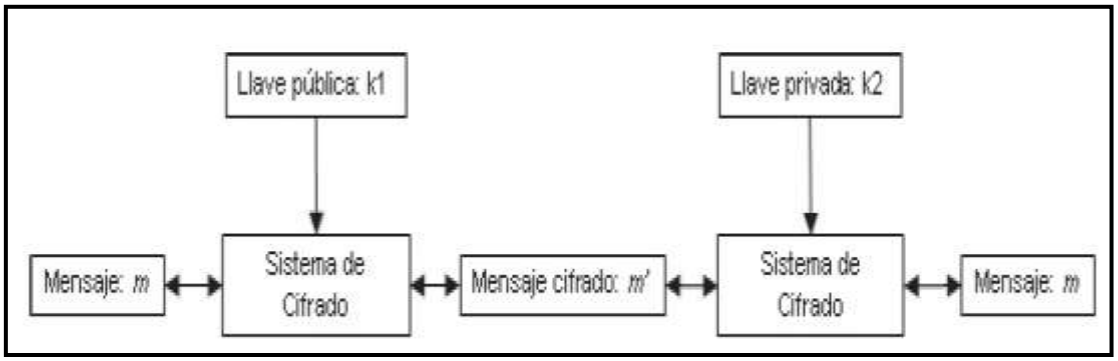

Figura 1.17: Esquema de criptograma asimétrico

1.4.1 Características de los algoritmos asimétricos

Es importante destacar que para este tipo de Criptografía lo que se cifra con una llave se puede descifrar con la otra llave. Esto es de gran ayuda ya que el número de llaves deben poseerse se reduce considerablemente. Si alguien quisiera enviar un mensaje cifrado a n personas, necesitaría saber n llaves públicas una de cada persona. Pero si n personas quiere enviar un mensaje cifrado sólo es necesario que los demás conozcan su llave pública. El problema de la Criptografía asimétrica, es verificar la autenticidad de las llaves públicas.

Los algoritmos asimétricos están basados en funciones matemáticas fáciles de calcular pero muy complicadas para realizar la inversa, por ejemplo, la potencia y el logaritmo discretos.

Las clave privada y pública están relacionadas matemáticamente, pero esta relación debe ser suficientemente compleja para que el criptoanalista no pueda encontrar una clave solamente a partir de la otra. Debido a esto, las claves privadas y públicas no las elige el usuario sino que las calcula un algoritmo y, normalmente, son muy largas 512 bits como mínimo.

Un algoritmo de clave pública debe cumplir:

- Conocido el criptograma no se puede descifrar el texto ni adivinar la clave.
- Conocido el texto y el criptograma es más caro (en tiempo y/o dinero) encontrar la clave que el valor de la información.

En estos sistemas también funciona el criptoanálisis de “prueba y ensayo” y se pueden aplicar las mismas suposiciones que en algoritmos simétricos. Aparte de este método, también hay algoritmos matemáticos para obtener la clave privada desde la pública pero, si el algoritmo es bueno, éstos son más caros que el valor de la información.

El inconveniente de estos sistemas es la dificultad de implementación y la lentitud de proceso. La ventaja es que implementan servicios de autenticación y firma, y además

no tienen problemas con distribución de claves: la clave pública puede ser visible por cualquiera y la privada no se transmite nunca.

1.4.2 Ventajas de los criptosistemas asimétricos

- Solo se ha de mantener secreta la clave privada (aunque se ha de garantizar la autenticidad de las claves públicas).
- La gestión de claves es una red grande con múltiples interlocutores y resulta más sencilla que en el caso de los criptosistemas simétricos. Además el número de claves necesarias es también menor.
- Las claves se pueden mantener durante mucho tiempo, no siendo necesario que sean distintas para cada sesión.
- La mayoría de los criptosistemas asimétricos permiten esquemas eficientes de firma digital.

1.4.3 Desventajas de los criptosistemas asimétricos

- La velocidad de los criptosistemas asimétricos más rápidos esta varios órdenes de magnitud por debajo de los criptosistemas simétricos usuales.
- Las claves suelen tener tamaños muy elevados.
- Los criptosistemas asimétricos basan su seguridad en la supuesta dificultad de un reducido conjunto de problemas teórico-numéricos. El descubrimiento de nuevos algoritmos para tratar dichos problemas puede reducir la seguridad de los criptosistemas asimétricos afectados.
- La historia de los criptosistemas asimétricos es más corta que la de su contrapartida simétrica.

1.5 Procesar datos

La forma en que se lleva a cabo el procesamiento de los datos se vuelve importante para la configuración del algoritmo a utilizar en las transformaciones; así, una clasificación de los sistemas criptográficos con base en la forma de procesar datos es hacerlo de manera serial, o en bloques.

1.5.1 Cifrado por bloques

Los algoritmos de cifrado por bloques emplean bloques de tamaño fijo del texto (véase figura 1.18) en claro y crean un bloque de tamaño fijo de texto cifrado, de igual tamaño que la entrada. Dicho tamaño de bloque debe ser grande, así se evitan ataques de texto cifrado. En tanto la asignación de bloques de entrada a bloques de salida debe ser uno a uno, haciendo así el proceso reversible.

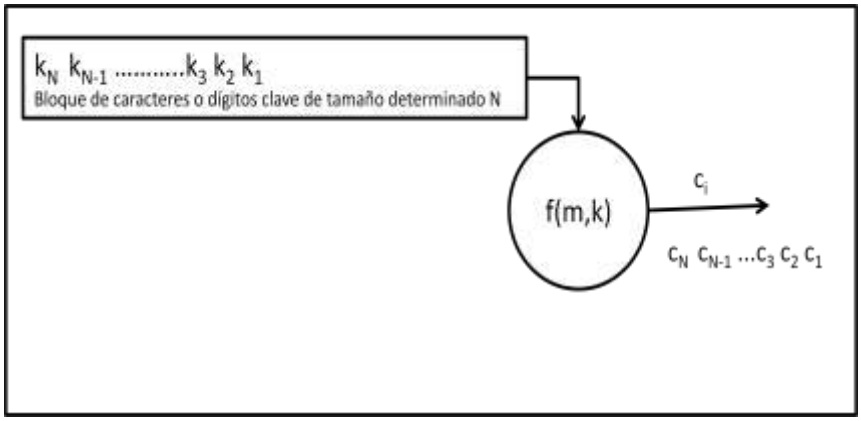

Figura 18: Cifrado en bloques

Para la asignación de bloques, los algoritmos de cifrado simétrico realizan sustituciones y permutaciones en el texto en claro hasta obtener el texto cifrado. La sustitución es el reemplazo de un valor de entrada por otro de los posibles valores de salida.

La permutación es un tipo especial de sustitución en el que los bits de un bloque de entrada son reordenados para producir el bloque cifrado, de este modo se preservan las estadísticas del bloque de entrada (el número de unos y ceros).

Los algoritmos de cifrado por bloques iterativos funcionan aplicando una transformación (función de rotación) sucesivas veces a un bloque de texto en claro. La cantidad de "rotaciones" depende del nivel de seguridad buscado. Se aplica una misma función a los datos usando una subclave obtenida de la clave secreta original dada por el usuario.

Los algoritmos en bloque pueden emplearse en cuatro modos (véase figura 1.19):

- ***Electronic Code Book (ECB).*** Se cifran los bloques de texto por separado.

- ***Cipher Block Chainning (CBC).*** Cada bloque de mensaje en claro se opera mediante una XOR con el bloque cifrado anterior. El primer bloque se opera con un vector de inicialización.

- ***Cipher FeedBack (CFB).*** Se realiza una OR-EXCLUSIVA entre caracteres o bits aislados del texto y las salidas del algoritmo. El algoritmo utiliza como entrada los criptogramas.

- ***Output FeedBack (OFB).*** Igual que el CFB, se realiza una OR-EXCLUSIVA entre caracteres o bits aislados del texto y las salidas del algoritmo. Pero éste utiliza como entradas sus propias salidas, por lo tanto no depende del texto, es un generador de números aleatorios.

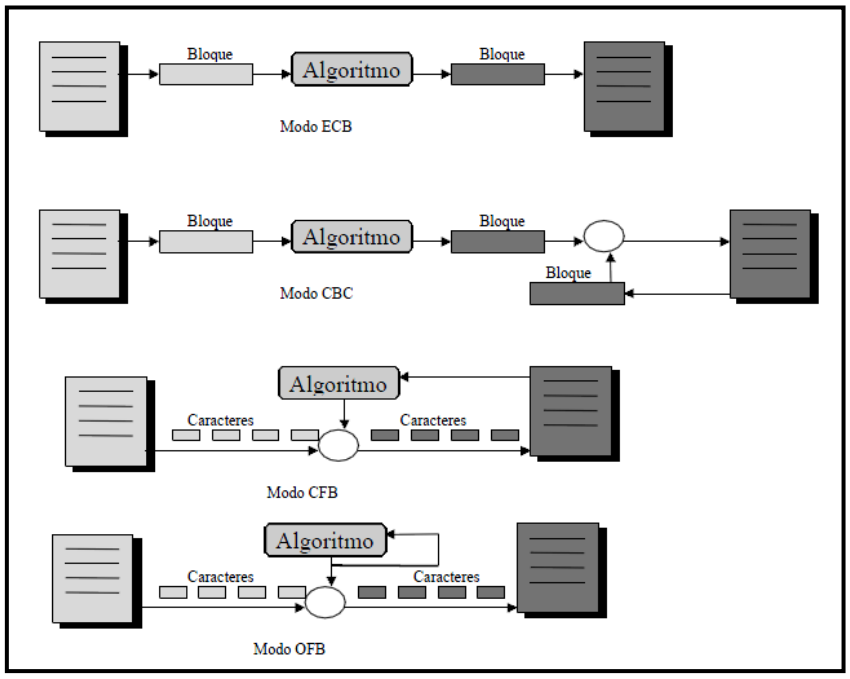

Figura 1.19: Tipos de cifrado por bloques

1.5.2 Cifrado en flujo o serial

Este tipo de cifrado, consiste en cifrar uno a uno los elementos contenidos en el mensaje en claro (véase figura 1.20) y conforme se están generando esos datos a cifrar, se pueden estar generando simultáneamente los caracteres o dígitos que fungirán como clave.

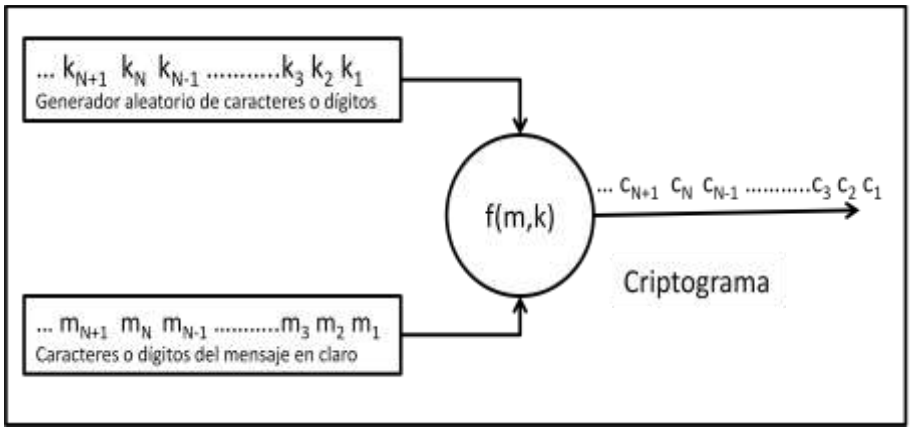

Figura 1.20: Cifrado en flujo o serial

Con base en la figura mostrada se puede prever que para el proceso de descifrado será necesario contar con cada uno de los elementos del criptograma y de manera importante, si se trata de un sistema criptográfico simétrico, se requiere contar con un generador "aleatorio" de caracteres o dígitos que genere exactamente la misma secuencia, ya que de otra forma será imposible recuperar el texto plano, es más, aun si se tratara de un sistema asimétrico sería necesario contar con un generador aleatorio de caracteres o dígitos que genera la secuencia de clave de descifrado correspondiente a la clave de cifrado que se haya usado.

1.6 Red Feistel

La mayoría de los cifrados en bloque, se basan en el diseño de Feistel en el cual un bloque de tamaño N bits se divide en bloques del mismo tamaño. Por ejemplo se divide en dos bloques del mismo tamaño N/2, A y B. A partir de aquí comienza el proceso de cifrado y consiste en aplicar una función unidireccional a un bloque B y a una subclave k1 generada a partir de la llave secreta. Se mezclan el bloque A con el resultado de la

función mediante un XOR. Se permutan los bloques y se repite el proceso n veces. Finalmente se unen los dos bloques en el bloque original (véase figura 1.21).

La principal ventaja de la Red Feistel es: el cifrado y descifrado son idénticos, sólo se cambia el orden de las subclaves.

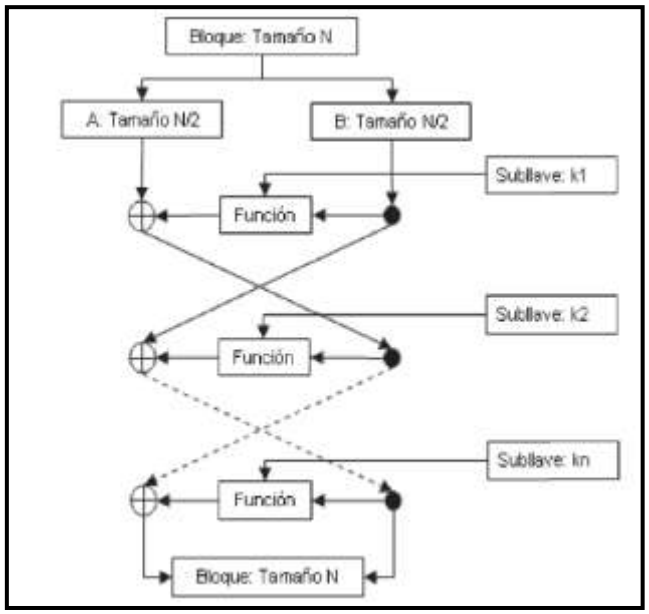

Figura 1.21: Red Feistel

1.7 Criptoanálisis

La Criptografía como ya se sabe es aquélla que se ocupa del diseño de procedimientos para cifrar; es decir, para enmascarar una determinada información de carácter confidencial. El criptoanálisis, por su parte, se ocupa de romper esos procedimientos de cifrado para así recuperar la información original sin necesidad de la clave.

1.7.1 Antecedentes del criptoanálisis

Criptoanálisis clásico

Aunque la expresión criptoanálisis es relativamente reciente, fue acuñada por William F. Friedman en 1920, los métodos para romper códigos y cifrados son mucho más antiguos. La primera explicación conocida del criptoanálisis se debe al sabio árabe del siglo IX, Yusuf Yaqub ibn Ishaq al-Sabbah Al-Kindi, en su Manuscrito (véase figura

1.22) para Descifrar Mensajes Criptográficos. Este tratado incluye una descripción del método de análisis de frecuencias.

Figura 1.20: Manuscrito d Yusuf Yaqub ibn Ishaq al-Sabbah Al-Kindi

El análisis de frecuencias es la herramienta básica para romper los cifrados clásicos. En todas las lenguas conocidas, ciertas letras del alfabeto aparecen más frecuentemente que otras; por ejemplo, en español, las vocales son muy frecuentes, ocupando alrededor del 45% del texto, siendo la E y la A las que aparecen en más ocasiones, mientras que la frecuencia sumada de F, Z, J, X, W y K no alcanza el 2%. Igualmente, se pueden reunir estadísticas de aparición de pares o tríos de letras. El análisis de frecuencias revelará el contenido original si el cifrado utilizado no es capaz de ocultar estas estadísticas. Por ejemplo, en un cifrado de sustitución simple (en el que cada letra es simplemente sustituida por otra), la letra más frecuente en el texto cifrado seria un candidato probable para representar la letra "E".

El análisis de frecuencias se basa tanto en el conocimiento lingüístico como en las estadísticas, pero al volverse cada vez más complicadas los cifrados, la Matemática gana terreno gradualmente en el enfoque predominante en el criptoanálisis. Este cambio fue particularmente evidente durante la Segunda Guerra Mundial, cuando los esfuerzos para romper los códigos del Eje requirieron nuevos niveles de sofisticación matemática. Más

aún, la automatización fue aplicada por primera vez en la Historia al criptoanálisis, bajo la forma de los dispositivos Bomba y Colossus, una de las primeras computadoras.

Criptoanálisis moderno

Aunque la computación fue utilizada con gran éxito durante la Segunda Guerra Mundial, también hizo posibles nuevos métodos criptográficos que eran órdenes de magnitud más complejos que los utilizados hasta la fecha. Tomada como un todo, la Criptografía moderna se ha vuelto mucho más impenetrable al criptoanalista que los métodos de pluma y papel del pasado, y parece que en la actualidad llevan ventaja sobre los métodos del puro criptoanálisis.

Los criptoanálisis exitosos han influido sin lugar a dudas en la Historia. La capacidad de leer los pensamientos, supuestamente secretos, o los planes de otros puede ser una ventaja decisiva, y nunca con mayor razón que en tiempos de guerra. Por ejemplo, durante la Primera Guerra Mundial, el descifrado del Telegrama Zimmermann fue capital para la entrada de los Estados Unidos en la guerra. En la Segunda Guerra Mundial, el criptoanálisis de los códigos alemanes, incluyendo la máquina Enigma y el código Lorenz, ha sido considerado desde un factor que apenas acortó la guerra en algunos meses en Europa, hasta un elemento crucial que determinó el resultado final. Los Estados Unidos también se beneficiaron del criptoanálisis del código japonés PURPLE durante la contienda.

Todos los gobiernos han sido conscientes de los potenciales beneficios del criptoanálisis para la inteligencia militar, tanto en lo puramente bélico como en lo diplomático, y han establecido con frecuencia organizaciones dedicadas en exclusiva al descifrado de códigos de otras naciones, por ejemplo GCHQ y NSA, organizaciones americanas todavía muy activas hoy en día. En 2004, surgió la noticia de que los Estados Unidos habían roto los códigos utilizados por Irán.

Por otro lado, uno de los algoritmo mas mencionados en la literatura que sufrió un ataque por medio del criptoanálisis diferencial es el algoritmo DES, se sospecha sobre puntos débiles ocultos en las cajas-S, pero dichas sospechas fueron descartadas en 1990,

con el descubrimiento independiente y la publicación libre por Eli Biham y Adi Shamir del criptoanálisis diferencial, un método general para romper cifrados de bloque. Las S-cajas de DES eran mucho más resistentes al ataque que si hubiesen sido escogidas al azar, lo que sugería que IBM conocía la técnica allá en los 70. Éste era de hecho, el caso en 1994, Don Coppersmith publicó los criterios de diseño originales para las cajas-S. IBM había descubierto el criptoanálisis diferencial en los 70 y, tras asegurar DES, la NSA les ordenó mantener en secreto la técnica. Coppersmith explica: *"Esto era así porque el criptoanálisis diferencial puede ser una herramienta muy potente, contra muchos esquemas diferentes, y había la preocupación de que aquella información en dominio público podía afectar negativamente a la seguridad nacional"*. Shamir también comentó "*Yo diría, al contrario de lo que algunos creen, que no hay evidencias de influencia alguna en el diseño de DES para que su estructura básica esté debilitada.*"

Las otras críticas eran sobre la longitud de clave considerada demasiado corta, se fundamentaban en el hecho de que la razón dada por la NSA para reducir la longitud de la clave de 64 bits a 56 era que los 8 bits restantes podían servir como bits de paridad, lo que en cierto modo resultaba sospechoso. Es ampliamente aceptado que la decisión de la NSA estaba motivada por la posibilidad de que ellos podrían llevar a cabo un ataque por fuerza bruta contra una clave de 56 bits varios años antes que el resto del mundo.

Pero, DES no solo ha sido punto de ataque del criptoanálisis diferencial, sino también de otro criptoanálisis conocido como lineal.

El criptoanálisis lineal, fue un ataque por fuerza bruta en 1998 el que demostró que DES podría ser atacado en la práctica, el cual necesita 2^{43} textos planos conocidos, y se destacó la necesidad de un algoritmo de repuesto, el cual consistía en 3Des, que era básicamente aplicar el algoritmo DES tres veces consecutivas empleando dos (2TDES, la primera clave en los pasos 1 y 3) o tres (3TDES) claves. 3DES ha sido ampliamente reconocido como seguro por ahora, aunque es bastante lento.

Así como se han generado algoritmos para cifrar información, surgen métodos que buscan romper la seguridad de los mismos, es así como surge otro tipo de ataque llamado criptoanálisis Biclique.

En diciembre de 2010, ponía en duda que cifrar el archivo INSURANCE de Wikileaks con AES256 hubiera sido una buena idea, ahora, ese mismo archivo es casi cuatro veces menos seguro que antes. Al parecer, los investigadores Andrey Bogdanov, de la universidad de Leuven, Dmitry Khovratovich, de la Universidad de Luxemburgo y Christian Rechberger, han descubierto que el algoritmo AES es casi cuatro veces más débil de lo que se pensaba inicialmente.

El método utilizado en esta ocasión, se denomina "Biclique Cryptanalysis of the Full AES"[6], o "Criptoanálisis al AES completo usando grafos bipartitos" y por cierto, parece un método simple y elegante, que solamente necesita 2^8 de memoria para llevarse a cabo y entre 2^{88} datos para AES128 y 2^{40} datos para AES256. Lo que pone de manifiesto de nuevo el posible problema del AES relacionado con el insuficiente número de rondas en su versión de 256 bits y que llevó a algunos criptógrafos a recomendar el uso de AES128, por ser teóricamente más seguro que AES256.

1.7.2 Conceptos de criptoanálisis

Criptoanálisis (del griego kryptós, "escondido" y, "desatar") es el estudio de los métodos para obtener el sentido de una información cifrada, sin acceso a la información secreta requerida para obtener este sentido normalmente.

El criptoanálisis consiste en comprometer la seguridad de un criptosistema. Esto se puede hacer descifrando un mensaje sin conocer la llave, o bien obteniendo a partir de uno o más criptogramas la clave que ha sido empleada en su cifrado. No se considera criptoanálisis el descubrimiento de un algoritmo secreto de cifrado; se ha de suponer por el contrario que los algoritmos siempre son conocidos.

En general el criptoanálisis se suele llevar a cabo estudiando grandes cantidades de pares mensaje-criptograma generados con la misma clave. El mecanismo que se emplee para obtenerlos es indiferente, y puede ser resultado de escuchar un canal de comunicaciones, o de la posibilidad de que el objeto de nuestro ataque responda con un

[6] http://research.microsoft.com/en-us/projects/cryptanalysis/aesbc.pdf. (visitada el 28 de Febrero del 2012)

criptograma cuando le enviemos un mensaje. Obviamente, cuanto mayor sea la cantidad de pares, más probabilidades de éxito tendrá el criptoanálisis.

Uno de los tipos de análisis más interesantes es el de texto claro escogido, que parte de que conocemos una serie de pares de textos claros elegidos y sus criptogramas correspondientes. Esta situación se suele dar cuando hay acceso al dispositivo de cifrado y éste nos permite efectuar operaciones, pero no nos permite leer su clave por ejemplo, las tarjetas de los teléfonos móviles GSM. El número de pares necesarios para obtener la clave desciende entonces significativamente. Cuando el sistema es débil, pueden ser suficientes unos cientos de mensajes para obtener información que permita deducir la clave empleada.

El criptoanálisis abarca muchas técnicas diversas, muchas veces no dependen del conocimiento del algoritmo sino que mediante sistemas de aproximación matemática se puede descubrir el texto en claro o la clave. La dificultad del análisis depende de la información disponible, así el criptoanalista puede tener acceso a:

- Un criptograma
- Un criptograma y su texto en claro.
- Un texto claro elegido y su criptograma.
- Un criptograma elegido y su texto en claro.
- Un texto en claro y su criptograma que están los dos elegidos.

Un par de métodos de criptoanálisis que han dado interesantes resultados son el análisis diferencial y el análisis lineal. El primero de ellos, partiendo de pares de mensajes con diferencias mínimas usualmente de un bit, estudia las variaciones que existen entre los mensajes cifrados correspondientes, tratando de identificar patrones comunes.

El segundo emplea operaciones XOR entre algunos bits del texto claro y algunos bits del texto cifrado, obteniendo usualmente un único bit.

Otro tipo de análisis, esta vez para los algoritmos asimétricos, consistirá en tratar de deducir la llave privada a partir de la pública. Suelen ser técnicas analíticas que

básicamente intentan resolver los problemas de elevado coste computacional en los que se apoyan estos criptosistemas: factorización, logaritmos discretos, etc. Mientras estos problemas genéricos permanezcan sin solución eficiente, podremos seguir confiando en estos algoritmos.

1.7.3 Clasificación de ataques

Existen diversos tipos de ataques en función de la amenaza que supongan para los criptosistemas reales. La máxima de Shannon, el atacante conoce el sistema, establece que el criptoanalista parte con el conocimiento completo del algoritmo, ya que un criptosistema basa su seguridad en el secreto de la clave y no del algoritmo. Además, el criptoanalista, cuenta siempre con la posibilidad de probar todas las claves posibles hasta encontrar la correcta, lo que se denomina *ataque por fuerza bruta.*

Ataque: es la realización de una amenaza y su materialización depende en gran medida del nivel de riesgo presente en el entorno, de manera que para disminuir la posibilidad de una ocurrencia se hace necesario aminorar el riesgo de una amenaza.

Criptoanalista pasivo: es aquel en el que el atacante únicamente monitoriza el canal de comunicación.

Un atacante pasivo solo amenaza la confidencialidad de los datos, pero es extremadamente difícil de detectar.

Criptoanalista activo: es aquel en el que el atacante intenta borrar, añadir o alterar de alguna manera los datos transmitidos por el canal de comunicación.

Un atacante activo amenaza la integridad, la autenticidad y la confidencialidad de los datos, un atacante activo es más fácil de detectar que uno pasivo.

1.7.4 Ataques a criptosistemas

El objetivo de este ataque consiste en recuperar el texto en claro a partir del texto cifrado o, idealmente, deducir la clave de cifrado. Podemos distinguir los siguientes tipos:

Solo texto cifrado: el atacante tiene acceso únicamente a una colección de textos cifrados en el mismo algoritmo y clave. El objetivo es determinar el máximo número posible de textos en claro correspondientes a esos textos cifrados o, idealmente, obtener la clave que se ha usado para cifrarlos.

Texto en claro conocido: el atacante posee un conjunto de textos cifrados con sus correspondientes textos en claro. Se pretende determinar la clave usada para cifrar dichos mensajes con el objetivo de poder descifrar cualquier mensaje cifrado con la misma llave.

Texto en claro elegido: el atacante puede obtener los textos cifrados correspondientes a un conjunto de textos en claro de su elección. Este ataque es mucho más potente que el de texto en claro conocido, ya que permite elegir bits específicos en el texto en claro que, al ser cifrados, ofrezcan mayor información acerca de la clave.

Texto cifrado elegido: al igual que el anterior, solo que se pueden obtener los textos en claro correspondientes a un conjunto de textos cifrados elegidos previamente, esto puede ocurrir, por ejemplo, cuando el atacante tiene acceso a una caja negra sellada que realiza descifrado automático; en este caso, el objetivo es deducir la clave. Este ataque es aplicable, principalmente, a los criptosistemas asimétricos; aunque, a veces, es también eficaz frente a criptosistemas simétricos.

Texto en claro elegido adaptativo: es un caso especial del ataque de texto en claro elegido. El atacante puede, además de elegir el texto en claro, modificar su elección en función de los resultados de los cifrados previos. En el ataque de texto en claro elegido básico, se tiende a usar un texto en claro muy grande que cubra un gran número de posibilidades; en este ataque se eligen textos en claro más reducidos y se modifican progresivamente para dirigir el cifrado a los intereses del atacante.

Texto cifrado elegido adaptativo: equivalente al texto en claro elegido adaptativo per con los textos cifrados; es decir, se elige un conjunto de textos cifrados que se van modificando según progresa el criptoanálisis.

Texto cifrado con claves relacionadas: el atacante puede obtener textos cifrados con claves distintas desconocidas pero que poseen una relación entre ellas que si conoce, este es un ataque poco práctico y complejo.

Puede resultar relativamente difícil realizar ataques prácticos basados en algunos de estos escenarios. No obstante, los ataques de texto en claro conocido o elegido son más comunes de los que se podría pensar en principios: muchos tipos de documentos tienen cabeceras o partes comunes que suponen casos de texto en claro conocido; por ejemplo, el código fuente tiene una serie de palabras clave que se repiten constantemente y en algunos documentos los inicios o finales son siempre los mismos.

1.8 Criptoanálisis diferencial

1.8.1 Conceptos

Criptoanálisis: considerada una técnica criptoanalista de tipo estadístico, que consistente en cifrar parejas de texto en claro escogidas con la condición de que su producto o-exclusivo obedezca a un patrón definido previamente. Los patrones de los correspondientes textos cifrados suministran información con la que conjeturar la clave criptográfica.

El criptoanálisis diferencial es un ataque sobre un criptograma elegido que se basa en lo que se denomina una pareja de criptogramas que está constituido por dos criptogramas cuyos textos en claro tienen una diferencia particular. Los textos en claro no son útiles al ataque y pueden elegirse aleatoriamente siempre que cumplan que su XOR es la deseada. La naturaleza estadística del ataque hace que no siempre tenga éxito, aunque son escasas las ocasiones en que falla.

Una vez fijada una diferencia en el texto en claro, las diferencias en los criptogramas pueden usarse para asignar probabilidades a las claves posibles y extraer la más probable. El método requiere usar muchos pares de textos en claro con una diferencia fija y considera sólo las parejas de criptogramas correspondientes.

Para entender un poco más, se muestra un ejemplo respecto a DES, los datos son manejados en hexadecimal.

Supongamos que se conocen un par de entradas a S1, previas a la XOR con los 6 bits de clave. Supongamos que estas entradas son $S1_E = 1$ y $S1^*{}_E = 35$. Supongamos que el valor de clave que queremos deducir es $S1_k = 23$, con lo que las salidas serán $S1_O = 1$ y $S1^*{}_O = C$, resultando $S1'_O = D$. En primer lugar se determinan todas las entradas a la caja S1 tal que la pareja constituida por cada uno de esas entradas y la XOR de la misma con $S1'_E = 34$, de lugar a un par de salidas cuya XOR sea igual a D. estas entradas para el caso que nos ocupa son: 1C, 28, 06, 32, 16, 10, 24. Los posibles valores de clave serán los que se muestran en la tabla 1.

PAREJAS DE ENTRADA (S1, S1*)	VALOR DE CLAVE
1C, 28	29
28, 1C	1D
06, 32	07
32, 06	33
22, 16	23
16, 22	17
10, 24	11
24, 10	25

Tabla 1.1: Valores posibles de clave para $S1_E = 1$, $S1^*{}_E = 35$, $S1'_O = D$

Para seguir con el análisis, se realiza lo mismo, solo que ahora se toma otra pareja de mensajes, con la misma clave que el primero, se determinan valores y así sucesivamente con varias parejas de texto conocido, de tal forma, que se comparan los posibles valores de las claves, y se va observando que valor prevalece en todas las talas y así determinar el valor de la clave.

Para mayor información del criptoanálisis realizado sobre DES, se puede recurrir al documento Introducción al criptoanálisis diferencial.[7]

[7] http://uam.academia.edu/DavidArroyo/Teaching/25559/Introduccion_al_criptoanalisis_diferencial. (visitada 15 de Enero del 2012)

CAPÍTULO 2

DESCRIPCIÓN DEL ALGORITMO CRIPTOGRÁFICO

DESCRIPCIÓN DEL ALGORITMO CRIPTOGRÁFICO

En este capítulo, se realiza una descripción de cómo se genera el algoritmo, para ello se inicia con un análisis de algoritmos simétricos más usados en la actualidad en donde se comparan sus características, como ya se hizo mención en la introducción dicho análisis es la base para determinar las características que tendrá el algoritmo. Así mismo se muestran conceptos matemáticos con la finalidad de entender su funcionamiento y tener una mayor comprensión de éstos dentro del algoritmo.

En el capítulo 3, se da justificación a cada una de las características que van formando el algoritmo.

2.1 Análisis de algoritmos de cifrado simétrico

Para poder llevar a cabo el diseño del algoritmo de cifrado simétrico, es necesario conocer las características más destacadas de algoritmos de cifrado que existen, para ello se realiza un análisis sobre estos. Se estudiaron las características más relevantes, como son: pertenece a una red Feistel, longitud de la clave, tamaño de los bloques, el número de rondas, que operaciones realiza y si hace uso de las cajas-S (véase tabla 2.1).

CARACTERISTICA	DES	AES	IDEA	TWOFISH	BLOWFISH
LONGITUD DE CLAVE(bits)	56	128,192 o 256	128	Hasta 256	32 a 448
TAMAÑO DE BLOQUE	64	128	64	128	64
RONDAS	16	10,12 O 14	8	16	16
OPERACIONES	$\oplus$, permutación, sustitución	$\oplus$, Desplazamiento, transposición	$\oplus$, $\boxplus$, $\odot$	$\oplus$,$\boxplus$, rotación	$\oplus$,$\boxplus$
CAJAS-S	Si	Si	No	Si	Si
FEISTEL	Si	No	No	Si	Si

Tabla 2.1 Principales características de los algoritmos simétricos de cifrado

Como se observa en la tabla 2.1 la longitud de las claves es variable, DES fue parte fundamental para el diseño de algunos algoritmo, pero al analizar la longitud de la clave se considera un punto débil del algoritmo, por lo tanto la longitud de los demás es variable y mayor a 64 bits, excepto Blowfish que maneja 32 bits como mínimo.

Otro punto a destacar es el tamaño de los bloques, como se observa en la tabla x, el tamaño oscila entre 64 y 128 bits, independientemente de que el tamaño de longitud de clave es variable entre los algoritmo. El numero de rondas también es similar se observa que DES, Twofish y Blowfish manejan el mismo número de rondas, mientras que IDEA usa un número menor de rondas y por otro lado AES es variable dependiendo de la longitud de la clave.

Otra característica importante, son las operaciones que realizan los algoritmos, la XOR es una operación fundamental dentro de los algoritmos de cifrado, ya que permite realizar una modificación de una manera muy sencilla; otras operaciones son desplazamientos, transposición, sustitución, suma modular y multiplicación, siendo esta ultima menos usada debido a que requiere más tiempo de computo comparado con la XOR y suma modular.

En la tabla 2.1 de los algoritmos comparados solo IDEA no hace uso de las cajas-S, mientras que de la red Feistel, solo DES y Blowfish hacen un uso completo de la red, mientras que otros solo tienen parte de su estructura como Twofish que hace uso de la red pero esta se rompe al hacer uso de una rotación de bits.

Del análisis anterior sobre las características más relevantes de los algoritmos se desprenden las siguientes características para el diseño del algoritmo.

- Longitud de clave 256 bits
- Tamaño de bloque de 64 bits
- Numero de rondas se define con el efecto en cascada.
- Cajas-S
- Operaciones: XOR, suma modular, rotación de bytes

Se decide que tenga dichas propiedades porque después de analizar sus características se observa cuales de ellas permiten que sea débil o mas fuerte dicho algoritmo. A lo largo de este capítulo se darán antecedentes matemáticos de las operaciones a realizar, se explicara cómo se diseñan las cajas-S, así mismo se mostrara la

estructura final que tendrá el algoritmo, en el capítulo 3 se dará justificación a cada una de las características que tiene el algoritmo.

2.2 Antecedentes matemáticos

En la Criptografía moderna, la fortaleza de un algoritmo reside en la imposibilidad computacional de encontrar una clave, que a su vez está protegida mediante la robustez de las funciones matemáticas empleadas.

Para comprender más la relación de la Criptografía con la Matemática, en esta sección se da una breve explicación para comprender las operaciones básicas utilizadas en el diseño del algoritmo desarrollado.

2.2.1 Estructuras algebraicas

La operación binaria $(*)$ en un conjunto S es una función $f: S\ x\ S$ que relaciona un par de elementos $a, b\ \in S$ con un tercero $c\ \in S$ llamado resultado. Al par $(S,*)$ se le denominará estructura, si cumple con ciertas propiedades.

Propiedades de las operaciones binarias

Con una operación binaria $(\cdot)$ definida en un conjunto S, se plantean las siguientes propiedades:

Cerradura: si el resultado de aplicar la operación a dos elementos $a, b\ \in S$ está definido en S, entonces la operación es cerrada. Esta propiedad también puede llamar a una operación binaria ley de composición interna.

$$a \cdot b\ \in S$$

Asociación: Si la operación es binaria, entonces no puede operar tres elementos a la vez. Por lo tanto, la operación es asociativa si permite trabajar dos elementos y después ocupar el resultado con el tercer elemento.

$$a \cdot\ b \cdot c = a \cdot (b \cdot c)$$

Existencia del elemento neutro: Si existe un elemento $e \in S$ tal que al operarlo con cualquier otro elemento $a \in S$, y no altera a éste último, entonces se habla de un elemento neutro.

$$a \cdot e = a$$

Existencia de elementos inversos: Esta propiedad se relaciona directamente con el elemento neutro. Si al operar dos números, $a, b \in S$ se obtiene el elemento neutro, entonces los dos elementos son inversos uno del otro.

$$a \cdot \bar{a} = e$$

Conmutativo: Si en la operación binaria no hay orden para trabajar los elementos, se dice que la operación permite la conmutación.

$$a \cdot b = b \cdot a$$

2.2.2 Grupo

Dado un conjunto no vacio G con una operación binaria $(*)$ definida. El sistema $(G,*)$ es un grupo si cumple con las siguientes propiedades de:

- Cerradura
- Asociación
- Elemento neutro
- Elemento inverso

2.2.3 Aritmética modular

La aritmética modular es una parte de la Matemática extremadamente útil en Criptografía, ya que permite realizar cálculos complejos y plantear problemas interesantes, manteniendo siempre una representación numérica compacta y definida, puesto que solo maneja un conjunto finito de números enteros. A continuación se define que es la aritmética modular.

Módulo Sean a y b dos enteros cualesquiera y n un entero positivo. Se dice que a es congruente con b módulo n; se representa de la siguiente forma:

$$a \equiv b \ (mod\ n)$$

La operación "$mod\ n$" significa dividir el resultado de la operación entre n para obtener el residuo X.

La ecuación se lee como a es congruente con b modulo n, lo que indica que $a\ mod\ n = b\ mod\ n$.

Por otro lado también se cumple

$$a = b + kn, \forall\ k \in Z$$

Por ejemplo, $33 \equiv 3\ (mod\ 10), ya\ que\ 33 \equiv 3 + 3 \cdot 10$.

En la aritmética modular se definen las operaciones de suma y producto

$$a + b \equiv c\ (mod\ n) \Leftrightarrow a + b = c + kn \qquad k \in \mathbb{Z}$$

$$ab \equiv c\ (mod\ n) \Leftrightarrow ab = c + kn \qquad k \in \mathbb{Z}$$

Propiedades de la suma:

- Asociativa: $\forall\ a, b, c \in \mathbb{Z}n \quad (a + b) + c \equiv a + (b + c)\ (mod\ n)$
- Conmutativa: $\forall\ a, b \in \mathbb{Z}n \quad a + b \equiv b + a \quad (mod\ n)$
- Elemento Neutro: $\forall\ a \in \mathbb{Z}n \quad \exists\ 0$ tal que $a + 0 \equiv a \quad (mod\ n)$
- Elemento Simétrico: $\forall a \in \mathbb{Z}n \quad \exists\ b$ tal que $a + b \equiv 0 \quad (mod\ n)$

Propiedades del producto:

- Asociativa: $\forall\ a, b, c \in \mathbb{Z}n \quad (a \cdot b) \cdot c \equiv a \cdot (b \cdot c)\ (mod\ n)$
- Conmutativa: $\forall\ a, b \in \mathbb{Z}n \quad a \cdot b \equiv b \cdot a \quad (mod\ n)$
- Elemento Neutro: $\forall\ a \in \mathbb{Z}n \quad \exists\ 1$ tal que $a \cdot 1 \equiv a \quad (mod\ n)$

Propiedad del producto con respecto a la suma:

- Distributiva: $\forall a, b, c \in \mathbb{Z}n \ (a + b) \cdot c \equiv (a \cdot c) + (b \cdot c) \pmod{n}$

La operación suma en este conjunto cumple las propiedades asociativa y conmutativa y posee elementos neutro y simétrico, por lo que el conjunto tendrá estructura de grupo conmutativo. A partir de ahora se llamará grupo finito inducido por *n* a dicho conjunto.

Para fines de diseño del algoritmo se hace uso de la suma modular, más adelante se indica cual es el uso que tiene dentro de la estructura del algoritmo.

2.2.4 Álgebra Booleana

Se ocupa de variables que adoptan dos valores discretos y de operaciones que asumen un significado lógico, los dos valores que pueden adoptar las variables reciben diferentes nombres (verdadero y falso, si y no, etc.), y en términos de bits se asignan los valores 0 y 1.

Dentro de esta lógica se tienen operadores lógicos binarios como son: AND,OR,XOR, pero también hay un operador lógico unario, el cual corresponde a NOT, esto es porque realiza la negación lógica haciendo uso de solo un valor, comparada a los anteriores que utiliza dos valores para poder realizar la operación.

A continuación se darán definiciones para las operaciones que son realizadas dentro del algoritmo de cifrado.

2.2.5 Operación XOR

El operador XOR realiza una comparación bit a bit entre dos expresiones numéricas y establece el bit correspondiente de acuerdo con la tabla de verdad (véase tabla 2.2), donde A y B son las entradas y X es la salida que se obtiene por la operación XOR entre las entras A y B.

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Tabla 2.2: Tabla de verdad XOR

En donde se observa que si los dos bits son iguales el bit correspondiente es 0, pero si son diferentes el bit es 1.

En 1917 Gilbert S. Vernam, nativo de Brooklyn e ingeniero del MIT que trabaja en los laboratorios de la empresa AT&T, diseña un dispositivo criptográfico para comunicaciones telegráficas basado en 32 códigos Baudot de los teletipos desarrollados por su compañía.

El Código Baudot y la transmisión

El código Baudot fue el primer código de caracteres de tamaño fijo (véase figura 2.1). Fue desarrollado por un ingeniero postal francés, Thomas Murray, en 1875 y nombrado después Emile Baudot, un pionero en la impresión telegráfica.

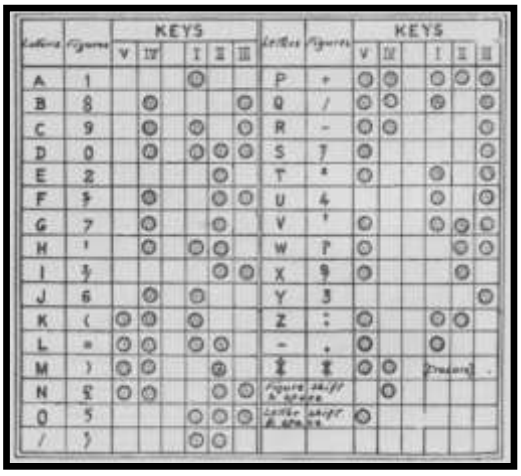

Figura 2.1: Código Baudot

En el código Baudot cada carácter está representado por cinco unidades o pulsos. Dentro del código de 5 bits existen sólo 25 o 32 combinaciones posibles (mark-spaces), lo cual es insuficiente para representar las 26 letras del alfabeto, los 10 dígitos y los diversos signos de puntuación, así como caracteres de control.

Propuesta de Vernam

Crear un dispositivo de tal forma que al transmitir se pueda incluir una llave:

$$mark + mark = space$$
$$mark + space = mark$$
$$space + mark = mark$$
$$space + space = space$$

Y para realizar la recuperación del mensaje será en el sentido contrario. A partir de esta propuesta es como se introduce la operación XOR dentro de los algoritmos, en donde desempeña un papel importante.

Un ejemplo de cómo opera la XOR dentro de la Criptografía es:

Supóngase que deseamos cifrar el mensaje *hola*, el cual se convierte a código binario ya que es el lenguaje que utiliza un equipo de cómputo.

Por lo tanto:

$$M = hola = 01101000\ 01101111\ 01101100\ 01100001$$

Ahora supóngase que nuestra clave es casa:

$$K = casa = 01100011\ 01100001\ 01110011\ 01100001$$

Al realizar la operación XOR se tendría que:

$$M \oplus K = C$$

$M =$ 01101000 01101111 01101100 01100001

$K =$ 01100011 01100001 01110011 01100001

$C = 00001011\ 00001110\ 00011111\ 00000000$

Por tanto C es ahora el mensaje pero cifrado. Esta es la forma en cómo opera la XOR; es una forma sencilla y que logra cifrar rápidamente un mensaje en claro, aunque se requiere de más operaciones para robustecer el cifrado.

2.2.6 Operación NOT

Como ya se mencionó la operación NOT se realiza sobre una sola variable, y el bit correspondiente a la entrada se asigna de acuerdo a la tabla de verdad (véase tabla 2.3).

A	Salida
0	1
1	0

Tabla 2.3: Tabla de verdad NOT

Donde, A es el valor de entrada y la salida es el nuevo valor que se toma después de aplicar dicha operación.

2.2.7 Números aleatorios y pseudoaleatorios

Número aleatorio: es aquél obtenido al azar, es decir, que todo número tenga la misma probabilidad de ser elegido y que la elección de uno no dependa de la elección del otro. Un ejemplo ideal para entender los números aleatorios es el lanzamiento de dados que no estén trucados.

Números pseudoaleatorios: Son números generados por medio de una función y que aparentan ser aleatorios. Estos números pseudoaleatorios se generan a partir de un valor inicial aplicando iterativamente la función. Las secuencias de números pseudoaleatorios no muestran ningún patrón o regularidad aparente desde un punto de vista estadístico, a pesar de haber sido generadas por un algoritmo completamente determinista, en el que las mismas condiciones iniciales producen siempre el mismo resultado.

2.2.8 Generadores de números pseudoaleatorios

Un generador pseudoaleatorios sigue una serie de pasos buscando aleatoriedad de la secuencia generada. Su gran desventaja es la evaluación de la "aleatoriedad" para su uso. Arranca siempre desde un valor inicial llamado semilla. La calidad de ésta es de vital importancia para su uso correcto; una mala semilla empobrece la implementación de los sistemas criptográficos.

Un mismo generador produce secuencias diferentes cuando las semillas de entrada son diferentes. Los valores obtenidos mediante el generador son perfectamente predecibles para quien conoce la semilla inicial, debido al carácter determinista del algoritmo que los genera, y resulta de gran importancia que la semilla inicial de donde arranca toda la secuencia permanezca oculta.

Un generador de números pseudoaleatorios presentan las siguientes características:

- **Periodo.** El tamaño de la secuencia generada, debe ser lo más largo posible.
- **Distribución.** Todos los números que componen la secuencia pseudoaleatoria tienen la misma probabilidad de aparecer.
- **Imprevisibilidad.** No es posible deducir un elemento de la secuencia a partir de los que ya se conocen.
- **Implementación.** La secuencia debe ser generada fácilmente.

Postulados de pseudoaleatoriedad de Golomb

Solomon W. Golomb es un matemático e ingeniero electrónico, profesor de la USC. Propuso tres postulados aplicables a los sistemas de comunicaciones para determinar que tan pseudoaleatoria es una secuencia binaria.

- **GOLOMB 1.** Con base en el periodo de la secuencia considerada, la diferencia entre el número de unos y el de los ceros no excede la unidad.

- **GOLOMB 2**. En la secuencia considerada, la mitad de los grupos debe tener longitud uno, la cuarta parte dos, una octava parte longitud tres, y así sucesivamente. Al mismo tiempo, para cada longitud indicada existe el mismo número de grupos de ceros que de unos.

- **GOLOMB 3.** La autocorrelación $AC(k)$ fuera de fase es constante para todo valor de k.

Algoritmos para generar números pseudoaleatorios

A continuación se presentan diferentes algoritmos determinísticos para generar números pseudoaleatorios, los cuales se clasifican en congruenciales y no congruenciales (véase figura 2.2).

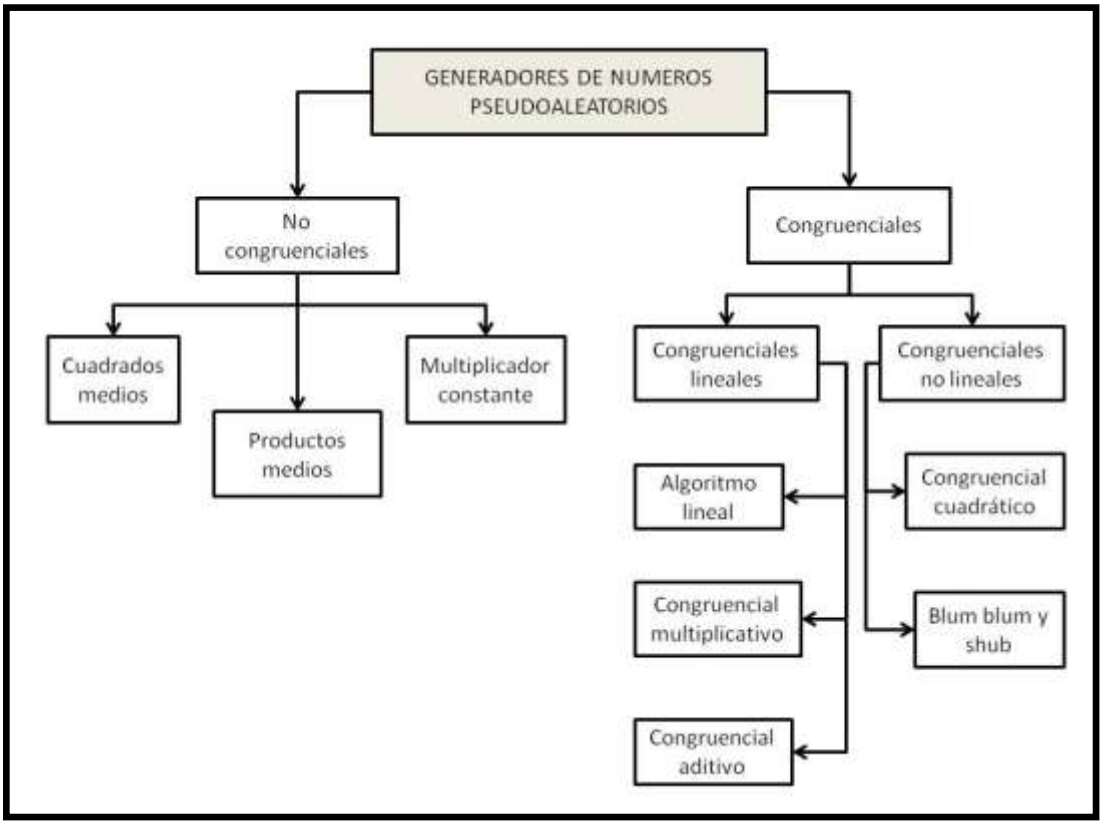

Figura 2.2: Clasificación de generadores de números pseudoaleatorios.

Algoritmo de cuadrados medios

Este algoritmo no congruencial fue propuesto en la década de los cuarenta del siglo XX por Von Neumann y Metrópolis. Requiere un número entero detonador con *D* dígitos, el cual es elevado al cuadrado para seleccionar del resultado los *D* dígitos del centro; el primer número se determina simplemente anteponiendo el "0." a esos dígitos. Para obtener el segundo número se sigue el mismo procedimiento, solo que ahora se eleva al cuadrado los *D* dígitos del centro que se seleccionaron para obtener el primer número.

Este método se repite hasta obtener n números. A continuación se presentan con más detalle los pasos para generar números con el algoritmo de cuadrados medios.

1. Seleccionar la semilla (X_0) con *D* dígitos (*D* > 3).
2. Sea Y_0= resultado de elevar X_0 al cuadrado; sea X_1 = los *D* dígitos del centro y sea r_1 = 0.*D* = dígitos del centro.
3. Sea Y_i= resultado de elevar X_i al cuadrado; sea X_{i+1}= los *D* dígitos del centro y sea r_{i+1}= *0.D* dígitos del centro para toda *i* = 1, 2, 3,..., *n*.
4. Repetir el paso 3 hasta obtener los *n* números r_i deseados.

El algoritmo de cuadrados medios generalmente es incapaz de generar una secuencia de números con período de vida *n* grande. Además, en ocasiones solo es capaz de generar un solo número.

Algoritmo de productos medios

La mecánica de generación de números pseudoaleatorios es similar a la del algoritmo de cuadrados medios. La diferencia entre ambos radica en que el algoritmo de productos medios requiere dos semillas, ambas con *D* dígitos; además, en lugar de elevarlas al cuadrado, las semillas se multiplican y del producto se seleccionan los *D* dígitos del centro, los cuales formarán el primer número pseudoaleatorio r_i = 0.*D*. Después se elimina una semilla y la otra se multiplica por el primer número de *D* dígitos, para luego seleccionar del producto los *D* dígitos que conformarán un segundo número r_i. Entonces se elimina la segunda semilla y se multiplican el primer número de *D* dígitos

por el segundo número de *D* dígitos; del producto se obtiene el tercer número r_i. Siempre se irá eliminando el número más antiguo, y el procedimiento se repetirá hasta generar los *n* números pseudoaleatorios. A continuación se presentan con más detalle los pasos del método.

1. Seleccionar una semilla (X_0) con *D* dígitos.
2. Seleccionar una semilla (X_1) con *D* dígitos.
3. Sea $Y_0 = X_0 * X_1$; sea X_2 = los *D* dígitos del centro y sea r_1 = 0. *D* dígitos del centro.
4. Sea $Y_i = X_i\ X_{i+1}$: sea X_{i+2} = los *D* dígitos del centro y sea r_{i+1}= 0. *D* dígitos del centro para toda *i* = 1, 2, 3,..., n.
5. Repetir el paso 4 hasta obtener los *n* números r_i deseados.

Algoritmo de multiplicador constante

Este algoritmo es similar al algoritmo de productos medios. Los siguientes son los pasos necesarios para generar números pseudoaleatorios.

1. 1. Selecciona una semilla (X_0) con *D* dígitos (*D* > 3).
2. Seleccionar una constante (a) con *D* dígitos (*D* > 3).
3. Sea $Y_0 = a * X_0$; sea X_1 = los *D* dígitos del centro y sea r_1= 0. *D* dígitos del centro.
4. Sea $Y_i = a * X_i$: sea X_{i+1} = los *D* dígitos del centro y sea r_{i+1} = *D* dígitos del centro para toda *i* = 1, 2, 3,..., *n.*
5. Repetir el paso 4 hasta obtener los *n* números r_i deseados.

Algoritmos congruenciales

Dentro de este tipo de algoritmos se tiene dos subclases, los lineales y no lineales.

Algoritmos congruenciales lineales

Algoritmo Lineal

El algoritmo fue propuesto por D. H. Lehmer en 1955. Según Law y Kelton, este algoritmo genera una secuencia de números enteros por medio de la siguiente ecuación recursiva:

$$X_{i+1} = (a\,X_i + c) mod\ (m) \qquad \text{Con } i = 0, 1, 2, 3, \ldots, n$$

Donde X_0 es la semilla, a es la constante multiplicativa, c es una constante aditiva y m es el módulo: $X_0 > 0, a > 0, c > 0\ y\ m > 0$ deben ser números enteros. La operación "$mod\ m$" significa multiplicar X_i por a , sumar c y dividir el resultado entre m para obtener el residuo X_{i+1} Es importante señalar que la ecuación recursiva del algoritmo congruencial lineal genera una secuencia de números enteros y que para obtener números pseudoaleatorios en el intervalo (0, 1) se requiere de la siguiente ecuación:

$$r_i = \frac{X_i}{m-1} \quad \text{Con } i = 0, 1, 2, 3, \ldots, n$$

Algoritmo congruencial multiplicativo

El algoritmo surge del algoritmo lineal cuando $c = 0$. Entonces la ecuación recursiva es:

$$X_{i+1} = (a\,X_i) mod\ (m) \qquad \text{Con } i = 0, 1, 2, 3, \ldots, n$$

En comparación con el algoritmo congruencial lineal, la ventaja del algoritmo multiplicativo es que implica una operación menos a realizar. Los parámetros de arranque de este algoritmo son X_0, $a\ y\ m$, los cuales deben ser enteros y mayores que cero. Para transformar los números X_i en el intervalo (0, 1) se usa la ecuación:

$$r_i = \frac{X_i}{m-1} \qquad \text{Con } i = 0, 1, 2, 3, \ldots,$$

Algoritmo congruencial aditivo

Este algoritmo requiere una secuencia previa de n números aleatorios $X_1, X_2, X_3, X_4, \ldots X_n$ para generar una secuencia de números enteros que empiezan en $X_{n+1}, X_{n+2}, X_{n+3}, X_{n+4}, \ldots$ Su ecuación recursiva es:

$$X_i = (X_{i-1} + X_{i-2})\ mod\ (m) \qquad \text{Con } i = n+1, n+2, n+3, \ldots, N$$

Algoritmos congruenciales no lineales

Dentro de los algoritmos congruenciales no lineales se tiene el algoritmo congruencial cuadrático y el de Blum, Blum, y Shub.

Algoritmo congruencial cuadrático

Este algoritmo tiene la ecuación recursiva:

$$X_{i+1} = (aX_i^2 + bX_i + c) mod(m) \qquad \text{Con } i = 0, 1, 2, 3, \ldots, n$$

En este caso, los números r_i pueden ser generados por la ecuación

$$r_i = \frac{X_i}{m-1}$$

De acuerdo con L'Ecuyer[8] las condiciones que deben cumplir los parámetros *m, a, b y c* para alcanzar un período máximo de $N = m$ son: m debe ser múltiplo de 2^g, donde g debe ser entero, a debe ser un número par, m debe ser un número impar, y $(b-1) mod\ 4 = 1$. De esta manera se logra un período de vida máximo $N = m$.

Blum Blum Shub (BBS)

BBS es un generador de secuencias pseudoaleatorias propuesto por Lenore Blum, Manuel Blum y Michael Shub en 1986. Es quizá el algoritmo que más pruebas de resistencia a superado, con la ventaja adicional de su gran simplicidad, por lo tanto se puede utilizar en Criptografía de clave pública y también como generador de secuencias pseudoaleatorias criptográficamente seguras.

[8] http://virtual.chapingo.mx/fis/aleato.pdf (visitada el 29 de Diciembre del 2011)

Consiste en escoger dos números primos grandes, p y q, que cumplan con la siguiente propiedad:

$$p \equiv 3(mod\ 4) \qquad q \equiv 3(mod\ 4)$$

Sea $n = pq$. Se elige un número x aleatorio primo relativo con n, que será la semilla inicial. El valor de x debe ser mantenido en secreto, al contrario de n que puede ser público. Para calcular los valores s_i de la serie de números pseudoaleatorios se hace de la siguiente forma:

$$s_0 = (x^2)(mod\ n)$$
$$s_{i+1} = (s_i^2)(mod\ n)$$

El algoritmo se utiliza para generar una secuencia binaria pseudoaleatoria, tomando como criterios de selección de los bits los siguientes puntos:

- Se toma el bit menos significativo de s_i
- Se toma el bit más significativo des_i; en este caso es necesario considerar cuantos bits representan al número más grande de la secuencia.
- Se toma el bit de paridad des_i.

Los autores del generador BBS demuestran que el generador es impredecible. Postulan:

1. El conocimiento de n es suficiente para generar las secuencias $x_0, x_1, x_2, \ldots$ a partir de cada valor inicial x_0 (semilla) dado. Es por tanto suficiente para obtener la secuencia de bits $s_0, s_1, s_2, \ldots$
2. Dado n, sus factores p y q son necesarios y suficientes para generar secuencias en dirección reversible. Es decir, a partir de un valor cualquiera de la secuencia x_i podemos obtener todos los anteriores hasta llegar a x_0.

Esta propiedad de la imprevisibilidad es la que en mayor medida ofrece seguridad a nuestro generador cuando se usa para Criptografía, ya que un criptoanalista no puede hacer de modo eficiente una predicción sobre los valores de $s_1, s_2, s_3 \ldots$

2.2.9 Confusión y difusión

La mayoría de los algoritmos simétricos se apoyan en los conceptos de Confusión y Difusión mencionados por Claude Shannon en la Teoría de la Información a finales de los años cuarenta.

Confusión. Trata de ocultar la relación entre el texto claro y el texto cifrado, dicha relación existe y se da a partir de la clave. El mecanismo más simple de confusión es la sustitución, que consiste en cambiar cada ocurrencia de un símbolo en el texto claro por otro.

Difusión. Diluye la redundancia del texto claro repartiéndola a lo largo de todo el texto cifrado. El mecanismo más elemental para llevar a cabo una difusión es la transposición, que consiste en cambiar de sitio elementos individuales del texto claro. Un ejemplo es modificar un bit del mensaje en claro y ver cómo afecta en el cifrado.

Por lo tanto para construir un algoritmo robusto se combinaran la confusión y difusión, esto mediante tablas pequeñas de sustitución, normalmente conocidas como cajas-S.

2.2.10 Cajas S

En Criptografía, una caja-S es un componente básico de los algoritmos de cifrado simétrico. En los algoritmos por bloques son usadas a menudo para oscurecer la relación existente entre texto claro y texto cifrado. En muchos casos las cajas-S son elegidas cuidadosamente para ser resistentes al criptoanálisis.

Una caja-S es una tabla de sustitución de m * n bits, que toma como entrada cadenas de *m* bits y da como salida cadenas de *n* bits. La utilización de las cajas-S es sencilla: se divide el bloque original en trozos de m bits y cada uno de ellos se sustituye por otro de n bits, haciendo uso de la caja-S correspondiente. Normalmente, cuanto más grandes sean las cajas-S, más resistente será el algoritmo resultante.

2.2.11 Rotación de bits

La rotación de bits consiste en que el bit que sale entra en el otro extremo, es decir no se pierden bits. En la figura 2.3 se observa el funcionamiento de la rotación a la izquierda, en donde el bit 7 va al bit 0.

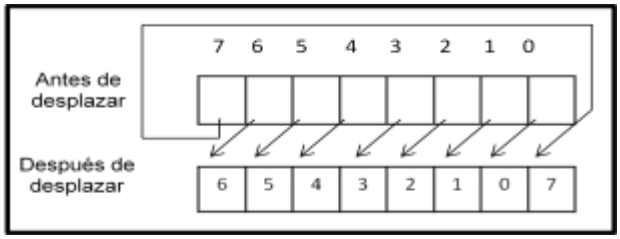

Figura 2.3: Rotación de bits a la izquierda

Para realizar la rotación a la derecha, en lugar de que el bit 7 pase al bit 0, ahora el bit 0 pasara al bit 7, como se muestra en la figura 2.4.

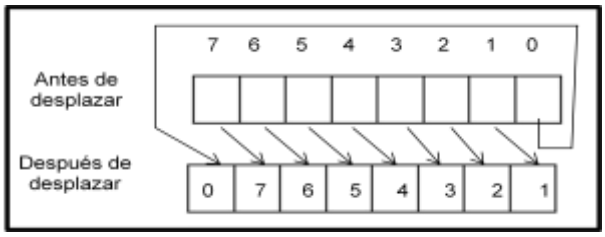

Figura 2.4: Rotación de bits a la derecha

Por ejemplo tenemos 0011 y hacemos una rotación hacia la derecha quedará la información como 1001.

2.3 Descripción del algoritmo criptográfico simétrico

En este apartado y tomando como base los antecedentes matemáticos antes expuestos, a continuación se explica paso a paso como fue desarrollándose el diseño del algoritmo criptográfico.

2.3.1 Estructura

Con los antecedentes del primer capítulo, se sabe que la seguridad de un algoritmo radica en la clave y no tanto en sus operaciones. Por lo tanto, la primera decisión es el tamaño que tendrá la clave; para lo cual se elige una clave de 256 bits, en el siguiente capítulo se justificara la decisión tomada.

Continuando, con la estructura del algoritmo, como se menciono en el capitulo existen dos formas de procesar datos, para diseño del algoritmo se opta por el procesamiento en bloques, por lo tanto se secciona en partes iguales la clave. Quedando así, se sabe que la longitud de la clave es de 256 bits, de los cuales se formaran 4 bloques del mismo tamaño, lo que da paso a la generación de 4 subclaves (véase figura 2.5).

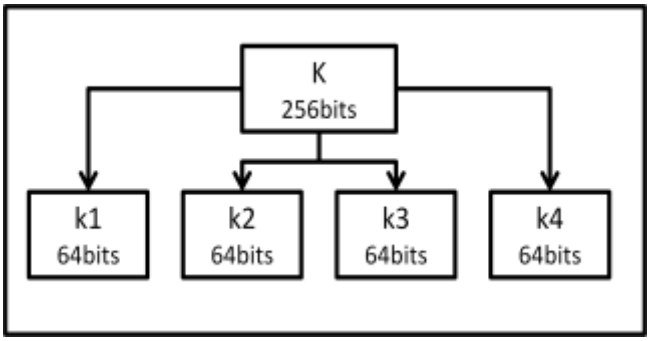

Figura 2.5: Generación de subclaves de misma longitud

Esta misma distribución se aplica para el mensaje en claro (véase figura 2.6).

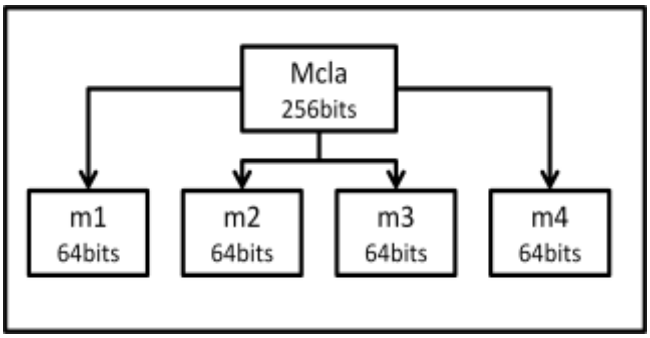

Figura 2.6: Generación de mensajes de tamaño fijo

Por el momento solo se tiene la distribución inicial del mensaje ($Mcla$) y la clave (K), esta distribución sobre el mensaje en claro y la clave, se realizan antes de iniciar la primera ronda. . A continuación se describen las operaciones que se realizaran sobre estos, en cada una de las rondas.

2.3.2 Planificación de claves

A continuación, se explicara las transformaciones que se realizan sobre la clave, se explica cómo se generan las subclaves requeridas para cada una de las rondas. Cabe mencionar que la justificación de cada una de las decisiones tomadas a lo largo del diseño del algoritmo será explicada en el siguiente capítulo.

Primero se utiliza una operación binaria, para transformar la clave antes de que sea seccionada en bloques; esta operación es la negación (NOT). Por lo tanto, las subclaves generadas serán $\overline{k1}$, $\overline{k2}$, $\overline{k3}$, $\overline{k4}$.

Posteriormente a las subclaves obtenidas de los bloques de la K original, se les aplicaran las siguientes transformaciones, con la finalidad de obtener las subclaves que se requieren para interactuar con el mensaje en cada una de las rondas.

***k*1**: Se realiza una rotación de 7 bits a la izquierda de cada byte de $\overline{k1}$.

***k*2** : Se busca en cajas los valores byte a byte de $\overline{k2}$, sustituyendo los valores correspondientes, como se muestra en la tabla 2.4:

BYTES	CAJA-S
0, 4	S3
1, 5	S2
2, 6	S4
3, 7	S1

Tabla 2.4: Tabla que muestra como sustituir bytes en cajas

***k*3**: Se realiza una rotación de 3 bits a la derecha de cada byte de $\overline{k3}$.

***k*4**: Se realiza lo mismo que en el caso de $k2$, solo que ahora sustituyendo los valores de $\overline{k4}$.

Hasta el momento solo se han generado 4 subclaves, pero para el diseño del algoritmo se decide que sean 8 subclaves las que interactúen con el mensaje, por lo tanto, las siguientes cuatro subclaves se generan a partir de las primeras 4.

A continuación se explica cómo se obtienen las subclaves $k5, k6, k7, k8$.

$\boldsymbol{k5}$: Se realiza una rotación de 3 bits a la derecha de cada byte de $k2$.

$\boldsymbol{k6}$: Se realiza una rotación de 7 bits a la izquierda de cada byte de $k4$.

$\boldsymbol{k7}$: Se busca en cajas los valores byte a byte de $k1$ y se sustituyen por los valores de las cajas como se muestra en tabla 2.5:

BYTES	CAJA-S
0, 4	S2
1, 5	S4
2, 6	S1
3, 7	S2

Tabla 2.5: Tabla que muestra como sustituir bytes en cajas

$\boldsymbol{k8}$: Se busca en cajas los valores byte a byte de $k3$ de la misma forma que se hace para $k7$.

Las subclaves generadas son utilizadas en la primera ronda, para cada una de las siguientes rondas se realiza el mismo procedimiento, pero estas se irán obteniendo a partir de las subclaves ya generadas, es decir:

Para ronda 2:

$\boldsymbol{k1}$: Se realiza una rotación de 7 bits a la izquierda de cada byte de $\boldsymbol{k1}$ $de\ la\ ronda$ 1.

$\boldsymbol{k2}$: Se busca en cajas los valores byte a byte de $\boldsymbol{k2}$ de la ronda 1, sustituyéndolos por los valores correspondientes, como se muestra en la tabla 2.3.

$\boldsymbol{k3}$: Se realiza una rotación de 3 bits a la derecha de cada byte de $\boldsymbol{k3}$ de la ronda 1.

$\boldsymbol{k4}$: Se realiza lo mismo que en el caso de $k2$, solo que ahora sustituyendo los valores byte a byte de $k4$ de la ronda 1.

Como se puede observar es el mismo procedimiento solo que haciendo uso de los valores de las subclaves anteriores, de igual forma se generan las subclaves $k5, k6, k7, k8$.

Así sucesivamente se realizan las operaciones para cada ronda, haciendo uso de los valores de las subclaves la ronda anterior.

2.3.3 Mensaje

Como se menciono con anterioridad el $Mcla$ fue seccionado en 4 bloques del mismo tamaño, es decir 64 bits, posteriormente se realiza una rotación de 5 bits a la derecha byte a byte, sobre cada uno de los bloques.

Antes de iniciar la descripción de las operaciones que forman parte de la estructura general del algoritmo, se describirá el proceso para generar las cajas-S.

2.3.4 Diseño de cajas-S

Como se mencionó en los antecedentes, las cajas son parte fundamental de los algoritmos simétricos, a continuación de describe el proceso para generar las cajas que son utilizadas para el algoritmo diseñado.

Algoritmo Blum Blum y Shub

Para fines del proyecto se elige utilizar el algoritmo Blum Blum y Shub para generar la secuencia de números que contendrán las cajas-S, a continuación se explicarán los pasos necesarios para poder crear las cajas.

Generación de números primos

Como se mencionó, el algoritmo Blum Blum y Shub requiere números primos grandes, para evitar que el periodo de repetición sea corto. Por medio de un programa que genera números primos, se eligieron al azar algunos de ellos, de 5 dígitos cada uno, los cuales cumplen con la propiedad $3(mod 4)$.

Por ejemplo se tiene dos números, los cuales serán p y q de acuerdo al algoritmo BBS.

$$p = 19991\,;\ q = 68443$$

Generación de números pseudoaleatorios para cajas S

Una vez elegidos los números primos, se calcula el valor de $N = p \times q$:

$$N = (19991)(68443); \quad N = 1368244013$$

Una vez que se tiene el valor de N, se elije la semilla, que es primo relativo menor que N:

$$s_0 = 563840384$$

Hasta ahora se tienen: p, q, N, s_0, con esto se inicia el cálculo de los números pseudoaleatorios con la siguiente ecuación:

$$s_{i+1} = (s_i^2)(mod\ N)$$

Por ejemplo:

$s_1 = (563840384)^2 \ mod\ 1368244013 \Longrightarrow 1209574881$

$s_2 = (1209574881)^2\ mod\ 1368244013 \Longrightarrow 1110319500$

$s_3 = (1110319500)^2\ mod\ 1368244013 \Longrightarrow 206485380$

$s_4 = (206485380)^2 \ \ mod\ 1368244013 \Longrightarrow 616475981$

$s_5 = (616475981)^2 \ mod\ 1368244013 \Longrightarrow 413684174$

$s_6 = (413684174)^2 \ mod\ 1368244013 \Longrightarrow 925041262$

$s_7 = (925041262)^2 \ mod\ 1368244013 \Longrightarrow 975546110$

$s_8 = (975546110)^2\ mod\ 1368244013 \Longrightarrow 435377244$

Así sucesivamente se obtienen los valores, finalmente se tomara el bit menos significativo, para ir construyendo nuestros números que contendrán las cajas-S.

Entonces, del ejemplo anterior tendríamos

$s_1 = 1209574881 \Longrightarrow \ 1001000000110001010010111100001$

$s_2 = 1110319500 \Longrightarrow \ 1000010001011100010000110001100$

$s_3 = 206485380 \Longrightarrow 11000100111010110111110000100$

$s_4 = 616475981 \Longrightarrow 10010010111110101010110101001101$

$s_5 = 413684174 \Longrightarrow 11000101010000101000111001110$

$s_6 = 925041262 \Longrightarrow 11011100100011000000100110111 0$

$s_7 = 975546110 \Longrightarrow 11101000100101101001101111111 0$

$s_8 = 435377244 \Longrightarrow 11001111100110101010001011100$

Considerando el bit menos significativo como se menciono, la secuencia generada es:

$$s = 10010000 \quad \text{ó} \quad s = 90_{Hex}$$

Como la estructura del algoritmo se divide en 4 bloques, se decide diseñar 4 cajas-S, las cuales se crean a partir de las secuencias de números pseudoaleatorios obtenidas como se mostro en el ejemplo anterior. Las cajas tienen una dimensión de 16 x 16, por lo tanto, tendrá 256 secuencias [0-256].

En el apéndice 2: Cajas-S, se muestran las cajas s1-s4 con sus respectivas inversas generadas con los números pseudoaleatorios obtenidos.

2.4 Estructura final del algoritmo

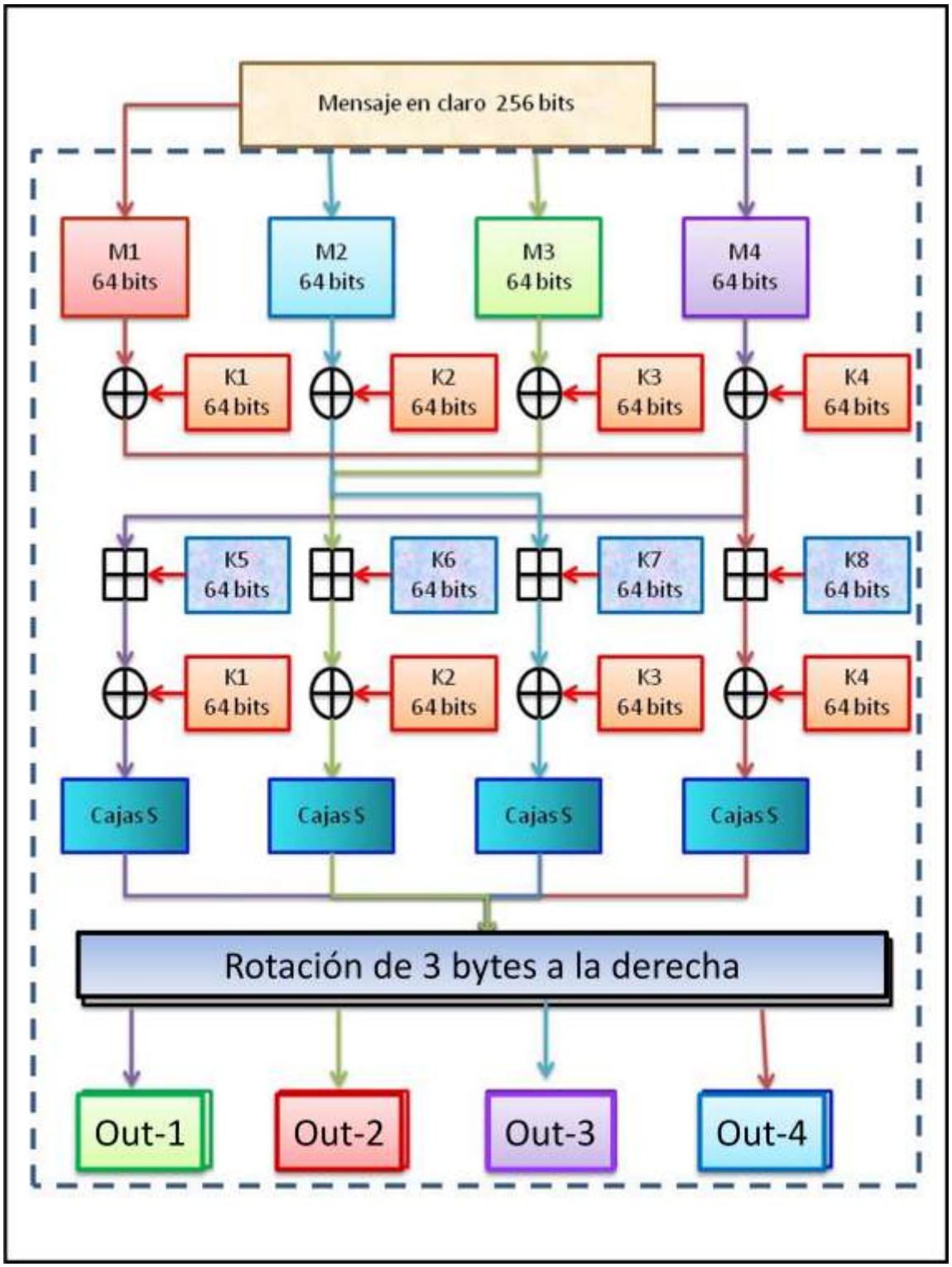

Figura 2.7: Estructura del algoritmo criptográfico simétrico

2.5 Funcionamiento del algoritmo

A continuación se explicara el proceso de la estructura que tiene el algoritmo para cifrar la información.

2.5.1 Cifrado

Se explico con anterioridad que la clave y mensaje en claro, sufren una alteración antes de comenzar a interactuar. También se conoce cómo se obtiene cada subclave, así como la generación de las cajas.

La primera operación que se lleva a cabo es un XOR entre el mensaje en claro con las primeras 4 subclaves, es decir:

$$xor1 = m1 \oplus k1$$
$$xor2 = m2 \oplus k2$$
$$xor3 = m3 \oplus k3$$
$$xor4 = m4 \oplus k4$$

La XOR se realiza sobre cada uno de los bytes que contiene cada bloque, cabe mencionar que las operaciones son llevadas a cabo en sistema hexadecimal.

Posteriormente sigue la suma modular, la cual consiste en sumar los resultados de las XOR obtenidas, con las siguientes 4 subclaves, esta operación se realiza byte a byte; por lo que, se realiza un total de 8 sumas por cada bloque; el modulo utilizado es 256 o bien, 100_{hex}.

Entonces, se tiene que:

$$suma1 = (m1 \oplus k1) \boxplus k8$$
$$suma2 = (m2 \oplus k2) \boxplus k7$$
$$suma3 = (m3 \oplus k3) \boxplus k6$$
$$suma4 = (m4 \oplus k4) \boxplus k5$$

Después se realiza una segunda XOR, pero ahora entre el resultado de las sumas modulares con las primeras 4 subclaves, tal como se muestra:

$$xor5 = ((m1 \oplus k1) \boxplus k8) \oplus k4$$

$$xor6 = ((m2 \oplus k2) \boxplus k7) \oplus k3$$

$$xor7 = ((m3 \oplus k3) \boxplus k6) \oplus k2$$

$$xor8 = ((m4 \oplus k4) \boxplus k5) \oplus k1$$

Una vez que se tienen generados los $XOR_{(5\text{-}8)}$, se hace uso de las cajas-S.

La forma en que se aplicaran las cajas es la siguiente manera (véase tabla 2.6), para cada uno de los bloques:

XOR	Bytes	CAJA-S	BLOQUES
5,6,7,8	0, 4	S1	1,2,3,4
5,6,7,8	1, 5	S2	1,2,3,4
5,6,7,8	2, 6	S3	1,2,3,4
5,6,7,8	3, 7	S4	1,2,3,4

Tabla 2.6: Tabla que indica cómo se manejan los byte de cada XOR dentro de las cajas para generar los bloques de salida

Es decir, se localiza el valor que se tiene en la XOR, dentro de la caja, como se maneja números en hexadecimal, por ejemplo si se tiene A4, el numero 4 indica la fila y la A indica la columna, por tanto, el valor que se encuentre en la casilla será el que sustituya el valor.

Se forman los bloques: $b1, b2, b3, b4$ con los valores obtenidos de las cajas. Para cada bloque, se realiza una rotación de 3 bytes a la derecha, tomando los 4 bloques juntos, y con ellos obtendremos la salida (véase figura 2.7).

Se tiene entonces:

Salida1 (byte)	Bloque (byte)
0	b1[3]
1	b1[4]
2	b1[5]
3	b1[6]
4	b1[7]
5	b2[0]
6	b2[1]
7	b2[2]

Salida2 (byte)	Bloque (byte)
0	b2[3]
1	b2[4]
2	b2[5]
3	b2[6]
4	b2[7]
5	b3[0]
6	b3[1]
7	b3[2]

Salida3 (byte)	Bloque (byte)
0	b3[3]
1	b3[4]
2	b3[5]
3	b3[6]
4	b3[7]
5	b4[0]
6	b4[1]
7	b4[2]

Salida4 (byte)	Bloque (byte)
0	b4[3]
1	b4[4]
2	b4[5]
3	b4[6]
4	b4[7]
5	b1[0]
6	b1[1]
7	b1[2]

Tabla 2.7: Rotación de bytes

Finalmente se hace un intercambio entre salidas para iniciar la siguiente ronda, es decir:

$$M1 = salida2$$

$$M2 = salida4$$

$$M3 = salida1$$

$$M4 = salida3$$

Para iniciar la siguiente ronda, como se menciono anteriormente se generan nuevamente las subclaves, y ahora el mensaje serán las salidas obtenidas y ordenadas como se indica.

2.5.2 Descifrado

Para el descifrado, el procedimiento es similar, lo que cambia son las subclaves, ya que se usan primero las generadas en la última ronda y así sucesivamente hasta llegar a las generadas en la primer ronda.

En operaciones son similares, solo se modifica la suma modular, por una resta modular; también el uso de cajas se ve modificado ya que ahora se hace uso de cajas-S inversas pero el procedimiento es el mismo, y las rotaciones serán en sentido contrario al indicado en el cifrado.

El procedimiento e interacción entre mensaje y subclaves es el mismo, y al final se mezclan las salidas de tal forma que correspondan al lugar original que se tenía es decir:

$$m1 = salida3$$
$$m2 = salida1$$
$$m3 = salida4$$
$$m4 = salida2$$

Finalmente para obtener el Mcla, solo se realiza la rotación de 5 bits a la izquierda byte a byte, ya que es la transformación que se le realizo al inicio antes de interactuar con la clave.

Ejemplo

A continuación se realiza un pequeño ejemplo para comprender mejor el funcionamiento del algoritmo, solo se realizara una ronda tanto para cifrar, como para descifrar. El procedimiento para las rondas siguientes es el mismo, a excepción de las subclaves, como se menciono anteriormente**.**

Procedimiento ronda 1

K= adriana santana osorio algoritmo

Convertimos el código ascii en hexadecimal

K= 61 64 72 69 61 6e 61 20 73 61 6e 74 61 6e 61 20 6f 73 6f 72 69 6f 20 61 6c 67 6f 72 69 74 6d 6f

Se niega la clave y generan las subclaves quedando así:

$\overline{k1}$= 9e 9b 8d 96 9e 91 9e df

$\overline{k2}$= 8c 9e 91 8b 9e 91 9e df

$\overline{k3}$= 90 8c 90 8d 96 90 df 9e

$\overline{k4}$= 93 98 90 8d 96 8b 92 90

Ahora se generan las subclaves $k1 - k8$ que se usaran en la primera ronda.

$k1$: Se realiza la rotación de 7 bits a la izquierda en cada byte, obteniendo:

$$\boldsymbol{k1} = 4f\ cd\ c6\ 4b\ 4f\ c8\ 4f\ ef$$

$k2$: Sustituimos los valores byte a byte de $\overline{k2}$, como se indico en la tabla 2.3.

$$\boldsymbol{k2} = 76\ d4\ 77\ de\ f4\ 8a\ cb\ 7a$$

$k3$: Se realiza una rotación de 3 bits a la derecha de cada byte de $\overline{k3}$.

$$\boldsymbol{k3} = 12\ 91\ 12\ b1\ d2\ 12\ fb\ d3$$

$k4$: Sustituimos valores byte a byte de $\overline{k4}$,como en el caso de $k2$

$$\boldsymbol{k4} = 20\ d0\ f0\ 12\ 6e\ 26\ 2b\ 9a$$

$\boldsymbol{k5}$: Se realiza una rotación de 3 bits a la derecha de cada byte de $k2$.

$$\boldsymbol{k5} = ce\ 9a\ ee\ db\ 9e\ 51\ 79\ 4f$$

$\boldsymbol{k6}$: Se realiza una rotación de 7 bits a la izquierda de cada byte de $k4$.

$$\boldsymbol{k6} = 10\ 68\ 78\ 09\ 37\ 13\ 95\ 4d$$

$\boldsymbol{k7}$: Se busca en cajas como se muestro en la caja 2.4 los valores byte a byte de $k1$.

$$\boldsymbol{k7} = 9d\ d8\ e8\ e4\ 9d\ 2f\ 87\ 7e$$

$\boldsymbol{k8}$: Se busca en cajas los valores byte a byte de $k3$ de la misma forma que se hace para $k7$.

$$\boldsymbol{k8} = f2\ 77\ 02\ ba\ 0e\ 35\ d2\ 6d$$

Ahora realizamos el corrimiento sobre el mensaje para ya iniciar con la interacción entre mensaje y clave.

Mcla= algoritmo criptográfico simétrico

Convertimos el código ascii a sistema hexadecimal:

Mcla= 61 6c 67 6f 72 69 74 6d 6f 20 63 72 69 70 74 6f 67 72 61 66 69 63 6f 20 73 69 6d 65 74 72 69 63

Aplicando rotación de 5 bits a la derecha sobre cada byte tenemos:

$$\boldsymbol{m1} = 0b\ 63\ 3b\ 7b\ 93\ 4b\ a3\ 6b$$
$$\boldsymbol{m2} = 7b\ 01\ 1b\ 93\ 4b\ 83\ a8\ 7b$$
$$\boldsymbol{m3} = 3b\ 93\ 0b\ 33\ 4b\ 1b\ 7b\ 01$$
$$\boldsymbol{m4} = 9b\ 4b\ 6b\ 2b\ a3\ 93\ 4b\ 1b$$

Ahora, iniciamos el procedimiento que indica el algoritmo:

$\boldsymbol{XOR1} = k1 \oplus m1 = 44\ ae\ fd\ 30\ dc\ 83\ ec\ 84$

$$\begin{array}{r} 4f\ cd\ c6\ 4b\ 4f\ \ c8\ 4f\ ef \\ \oplus\ 0b\ 63\ 3b\ 7b\ 93\ 4b\ a3\ 6b \\ \hline \boldsymbol{44\ ae\ fd\ 30\ dc\ 83\ ec\ 84} \end{array}$$

Lo mismo se realiza en las siguientes XOR con los valores correspondientes, obteniendo así:

$\boldsymbol{XOR2} = k2 \oplus m2 = 76\ d4\ 77\ de\ f4\ 8a\ cb\ 7a \oplus 7b\ 01\ 1b\ 93\ 4b\ 83\ a8\ 7b$

$$= 0d\ d5\ 6c\ 4d\ bf\ 09\ 68\ 01$$

$\boldsymbol{XOR3} = k3 \oplus m3 = 12\ 91\ 12\ b1\ d2\ 12\ fb\ d3 \oplus 3b\ 93\ 0b\ 33\ 4b\ 1b\ 7b\ 01$

$$= 29\ 02\ 19\ 82\ 99\ 09\ 80\ d2$$

$$\boldsymbol{XOR4} = k4 \oplus m4 = 20\ d0\ f0\ 12\ 6e\ 26\ 2b\ 9a \oplus 9b\ 4b\ 6b\ 2b\ a3\ 93\ 4b\ 1b$$

$$= bb\ 9b\ 9b\ 39\ cd\ b5\ 60\ 81$$

Ahora calculamos la suma modular:

$$\boldsymbol{SUMA1} = XOR4 \boxplus k5 = \ 89\ 35\ 89\ 14\ 6b\ 06\ d9\ d0$$

$$\begin{array}{r} bb\ 9b\ 9b\ 39\ cd\ b5\ 60\ 81 \\ \boxplus\ ce\ 9a\ ee\ db\ 9e\ 51\ 79\ 4f \\ \hline 89\ 35\ 89\ 14\ 6b\ 06\ d9\ d0 \end{array}$$

Tenemos los primeros bytes $bb\ y\ ce$, para realizar la suma es:

$$bb\ +\ ce\ = 189\ mod\ 256 = \ 89$$

Dicho procedimiento es utilizado para calcular el valor de cada byte en cada uno de los bloques, para obtener las sumas modulares, el resultado de las demás sumas es:

$$\boldsymbol{SUMA2} = XOR3 \boxplus k6 = 29\ 02\ 19\ 82\ 99\ 09\ 80\ d2 \boxplus 10\ 68\ 78\ 09\ 37\ 13\ 95\ 4d$$

$$= 39\ 6a\ 91\ 8b\ d0\ 1c\ 15\ 1f$$

$$\boldsymbol{SUMA3} = XOR2 \boxplus k7 = 0d\ d5\ 6c\ 4d\ bf\ 09\ 68\ 01 \boxplus 9d\ d8\ e8\ e4\ 9d\ 2f\ 87\ 7e$$

$$=0a\ ad\ 54\ 31\ 5c\ 38\ ef\ 7f$$

$$\boldsymbol{SUMA4} = XOR1 \boxplus k8 = 44\ ae\ fd\ 30\ dc\ 83\ ec\ 84 \boxplus f2\ 77\ 02\ ba\ 0e\ 35\ d2\ 6d$$

$$=36\ 25\ ff\ ea\ ea\ b8\ be\ f1$$

Calculando la siguiente XOR:

$$\boldsymbol{XOR5} = SUMA1 \oplus k1 = 89\ 35\ 89\ 14\ 6b\ 06\ d9\ d0 \oplus 4f\ cd\ c6\ 4b\ 4f\ c8\ 4f\ ef$$
$$= c6\ f8\ 4f\ 5f\ 24\ ce\ 96\ 3f$$

$$\boldsymbol{XOR6} = SUMA2 \oplus k2 = 39\ 6a\ 91\ 8b\ d0\ 1c\ 15\ 1f \oplus 76\ d4\ 77\ de\ f4\ 8a\ cb\ 7a$$

$$= 4f\ be\ e6\ 55\ 24\ 96\ de\ 65$$

$$\boldsymbol{XOR7} = SUMA3 \oplus k3 = 0a\ ad\ 54\ 31\ 5c\ 38\ ef\ 7f \oplus 12\ 91\ 12\ b1\ d2\ 12\ fb\ d3$$

$$= b8\ 3c\ 46\ 80\ 8e\ 2a\ 14\ ac$$

$$\boldsymbol{XOR8} = SUMA4 \oplus k4 = 36\ 25\ ff\ ea\ ea\ b8\ be\ f1 \oplus 20\ d0\ f0\ 12\ 6e\ 26\ 2b\ 9a$$

$$= 16\ f5\ 0f\ f8\ 84\ 9e\ 95\ 6b$$

Ahora, generamos los bloques de salida para los usaremos las cajas, como se menciono anteriormente, obteniendo así:

$$\boldsymbol{B1} = 6e\ e1\ 87\ d1\ ec\ 3a\ 6e\ d0$$

$$\boldsymbol{B2} = 28\ 73\ 1e\ 2c\ ec\ c9\ 47\ f9$$

$$\boldsymbol{B3} = b2\ 07\ a4\ 70\ f3\ ab\ 81\ 68$$

$$\boldsymbol{B4} = d0\ 38\ 44\ 93\ cc\ d4\ bd\ c2$$

Finalmente se aplica rotación de 3 bytes a la derecha tomando los cuatro bloques, y obtenemos las salidas finales:

$$\boldsymbol{S1} = d1\ ec\ 3a\ 6e\ d0\ 28\ 73\ 1e$$

$$\boldsymbol{S2} = 2c\ ec\ c9\ 47\ f9\ b2\ 07\ a4$$

$$\boldsymbol{S3} = 70\ f3\ ab\ 81\ 68\ d0\ 38\ 44$$

$$\boldsymbol{S4} = 93\ cc\ d4\ bd\ c2\ 6e\ e1\ 87$$

Por lo tanto el mensaje cifrado es:

$$\boldsymbol{Mcifrado_{Hex}} = d1\ ec\ 3a\ 6e\ d0\ 28\ 73\ 1e\ 2c\ ec\ c9\ 47\ f9\ b2\ 07\ 70\ f3\ ab\ 81\ 68$$

$$d0\ 38\ 44\ \ a4\ 93\ cc\ d4\ bd\ c2\ 6e\ e1\ 87$$

Descifrado

Ahora se inicia el proceso de descifrado para la primer ronda, que se realizo anteriormente, con la finalidad de que se comprenda el proceso y se verifique el funcionamiento del mismo.

Como se ha mencionado el proceso de descifrado es iniciar por las operaciones finales hasta llegar al inicio, y con la diferencia de uso de cajas-S y suma modular, ya que se usan cajas inversas y resta modular respectivamente.

Para generar las claves que se usan es el mismo procedimiento que en cifrado, se hace uso de las calculadas anteriormente para no repetir el proceso.

Se tiene entonces:

$\boldsymbol{k1} = 4f\ cd\ c6\ 4b\ 4f\ c8\ 4f\ ef$

$\boldsymbol{k2} = 76\ d4\ 77\ de\ f4\ 8a\ cb\ 7a$

$\boldsymbol{k3} = 12\ 91\ 12\ b1\ d2\ 12\ fb\ d3$

$\boldsymbol{k4} = 20\ d0\ f0\ 12\ 6e\ 26\ 2b\ 9a$

$\boldsymbol{k5} = ce\ 9a\ ee\ db\ 9e\ 51\ 79\ 4f$

$\boldsymbol{k6} = 10\ 68\ 78\ 09\ 37\ 13\ 95\ 4d$

$\boldsymbol{k7} = 9d\ d8\ e8\ e4\ 9d\ 2f\ 87\ 7e$

$\boldsymbol{k8} = f2\ 77\ 02\ ba\ 0e\ 35\ d2\ 6d$

Ahora el mensaje que se usa es el que se obtuvo del cifrado.

$$\boldsymbol{Mcifrado_{Hex}} = d1\ ec\ 3a\ 6e\ d0\ 28\ 73\ 1e\ 2c\ ec\ c9\ 47\ f9\ b2\ 07\ 70\ f3\ ab\ 81\ 68$$

$$d0\ 38\ 44\ \ a4\ 93\ cc\ d4\ bd\ c2\ 6e\ e1\ 87$$

Como se menciona en la descripción del algoritmo, las salidas son intercambiadas para ser ingresadas en el descifrado, por lo tanto el mensaje a utilizar queda de la siguiente manera.

$\boldsymbol{m1} = 2c\ ec\ c9\ 47\ f9\ b2\ 07\ a4$

$\boldsymbol{m2} = 93\ cc\ d4\ bd\ c2\ 6e\ e1\ 87$

$\boldsymbol{m3} = d1\ ec\ 3a\ 6e\ d0\ 28\ 73\ 1e$

$\boldsymbol{m4} = 70\ f3\ ab\ 81\ 68\ d0\ 38\ 44$

Aplicamos una rotación de 3 bytes en los bloques:

$\boldsymbol{B1}$ =6e e1 87 d1 ec 3a 6e d0

$\boldsymbol{B2} = 28\ 73\ 1e\ 2c\ ec\ c9\ 47\ f9$

$\boldsymbol{B3} = b2\ 07\ a4\ 70\ f3\ ab\ 81\ 68$

$\boldsymbol{B4} = d0\ 38\ 44\ 93\ cc\ d4\ bd\ c2$

Se introducen los valores en las cajas al igual que el cifrado solo que ahora usando las cajas inversas, y obtendremos los valores que corresponden a las XOR5-8

$\boldsymbol{XOR5} = c6\ f8\ 4f\ 5f\ 24\ ce\ 96\ 3f$

$\boldsymbol{XOR6} = 4f\ be\ e6\ 55\ 24\ 96\ de\ 65$

$\boldsymbol{XOR7} = b8\ 3c\ 46\ 80\ 8e\ 2a\ 14\ ac$

$\boldsymbol{XOR8} = 16\ f5\ 0f\ f8\ 84\ 9e\ 95\ 6b$

Se realiza la operación XOR y se obtendrá lo que son los valores de la suma en el cifrado.

$\boldsymbol{suma1} = xor5 \oplus k1 = c6\ f8\ 4f\ 5f\ 24\ ce\ 96\ 3f \oplus 4f\ cd\ c6\ 4b\ 4f\ c8\ 4f\ ef$

$= 89\ 35\ 89\ 14\ 6b\ 06\ d9\ d0$

$\boldsymbol{suma2} = xor6 \oplus k2 = 4f\ be\ e6\ 55\ 24\ 96\ de\ 65 \oplus 76\ d4\ 77\ de\ f4\ 8a\ cb\ 7a$

$= 39\ 6a\ 91\ 8b\ d0\ 1c\ 15\ 1f$

$$\boldsymbol{suma3} = xor7 \oplus k3 = b8\ 3c\ 46\ 80\ 8e\ 2a\ 14\ ac \oplus 12\ 91\ 12\ b1\ d2\ 12\ fb\ d3$$

$$= 0a\ ad\ 54\ 31\ 5c\ 38\ ef\ 7f$$

$$\boldsymbol{suma4} = xor8 \oplus k4 = 16\ f5\ 0f\ f8\ 84\ 9e\ 95\ 6b \oplus 20\ d0\ f0\ 12\ 6e\ 26\ 2b\ 9a$$

$$= 36\ 25\ ff\ ea\ ea\ b8\ be\ f1$$

Se realiza la resta modular para calcular los valores que corresponden a la xor1-4, el procedimiento es el mismo que en la suma solo que ahora se realiza una resta entre bytes.

$$\boldsymbol{XOR1} = suma4 - k8 = 36\ 25\ \text{ff ea ea b8 be f1} - f2\ 77\ 02\ ba\ 0e\ 35\ d2\ 6d$$

$$= 44\ ae\ fd\ 30\ dc\ 83\ ec\ 84$$

$$\boldsymbol{XOR2} = suma3 - k7 = \text{0a ad 54 31 5c 38 ef 7f} - 9d\ d8\ e8\ e4\ 9d\ 2f\ 87\ 7e$$

$$= 0d\ d5\ 6c\ 4d\ bf\ 09\ 68\ 01$$

$$\boldsymbol{XOR3} = suma2 - k6 = 39\ 6a\ 91\ 8b\ d0\ 1c\ 15\ 1f - 10\ 68\ 78\ 09\ 37\ 13\ 95\ 4d$$

$$= 29\ 02\ 19\ 82\ 99\ 09\ 80\ d2$$

$$\textbf{XOR4 =} suma1 - k5 = \ 89\ 35\ 89\ 14\ 6b\ 06\ d9\ d0 - ce\ 9a\ ee\ db\ 9e\ 51\ 79\ 4f$$

$$= bb\ 9b\ 9b\ 39\ cd\ b5\ 60\ 81$$

Ahora, se hace una XOR para obtener el mensaje:

$$\boldsymbol{m1} = xor1 \oplus k1 = 44\ ae\ fd\ 30\ dc\ 83\ ec\ 84 \oplus 4f\ cd\ c6\ 4b\ 4f\ c8\ 4f\ ef$$

$$=0b\ 63\ 3b\ 7b\ 93\ 4b\ a3\ 6b$$

$$\boldsymbol{m2} = xor2 \oplus k2 = 0d\ d5\ 6c\ 4d\ bf\ 09\ 68\ 01 \oplus 76\ d4\ 77\ de\ f4\ 8a\ cb\ 7a$$

$$=7b\ 01\ 1b\ 93\ 4b\ 83\ a8\ 7b$$

$$\boldsymbol{m3} = xor3 \oplus k3 = 29\ 02\ 19\ 82\ 99\ 09\ 80\ d2 \oplus 12\ 91\ 12\ b1\ d2\ 12\ fb\ d3$$

$$= 3b\ 93\ 0b\ 33\ 4b\ 1b\ 7b\ 01$$

$$\boldsymbol{m4} = xor4 \oplus k4 = bb\ 9b\ 9b\ 39\ cd\ b5\ 60\ 81 \oplus 20\ d0\ f0\ 12\ 6e\ 26\ 2b\ 9a$$

$$= 9b\ 4b\ 6b\ 2b\ a3\ 93\ 4b\ 1b$$

Finalmente se realiza una rotación de 5bits a la izquierda, byte a byte.

Mcla= 61 6c 67 6f 72 69 74 6d 6f 20 63 72 69 70 74 6f 67 72 61 66 69 63 6f 20 73 69 6d 65 74 72 69 63

Mcla=algoritmo criptografico simetric

Se puede observar se obtiene el mensaje original, con lo que se comprueba que el algoritmo funciona correctamente.

CAPÍTULO 3

JUSTIFICACIÓN
DEL
ALGORITMO SIMÉTRICO

JUSTIFICACIÓN DE DISEÑO DEL ALGORITMO SIMETRICO

3.1 Justificación del diseño de estructura del algoritmo

En este capítulo se dará justificación a las decisiones tomadas en cada paso de la descripción del algoritmo mencionadas en el capítulo 2, dichas decisiones se tomaron en base al análisis de la tabla 2.1 y bases matemáticas.

3.1.1 Longitud de la clave

La primer pregunta que surge es cuál será la longitud de la clave, ya que es importante para la robustez del algoritmo. Existen algoritmos criptográficos que manejan 64, 128, 256 bits; se realizó un análisis para determinar cuáles eran las diferencias y de esa forma decidir el tamaño.

Se tiene que, con una clave de longitud *n* bits, hay 2^n llaves posible. Este número crece muy rápidamente a medida que aumenta *n.* Por lo tanto se realizó el cálculo de posibles claves, para las longitudes más comunes usadas en algoritmos, obteniendo así:

$2^{64} = 18446744073709551616$

$2^{128} = 3.402823669209384634633746074317 7e^{38}$

$2^{256} = 1.157920892373161954235709850086 9e^{77}$

Como se observa conforme aumenta el número de bits, se tiene un mayor número de combinaciones para encontrar la posible clave, lo que computacionalmente requiere mayor tiempo para conocer la clave correcta.

Por otro lado se toma como referencia la tabla 3.1 y 3.2 que se muestran a continuación:

Longitud de clave (en caracteres)	26 caracteres minúsculas o mayúsculas	36 caracteres mayúsculas o minúscula y dígitos	52 caracteres minúsculas y mayúsculas	96 caracteres (todo)
6	51 minutos	6 horas	2.3 días	3 meses
7	32.3 horas	9 días	4 meses	24 años
8	24 días	10.5 meses	17 años	2288 años
9	21 meses	32.6 años	890 años	219601 años
10	45 años	1.160 años	45840 años	21081705 años

Tabla 3.2: Longitud de claves y tiempos requeridos[9]

Tamaño de clave(bits)	**Numero de claves alternativas**	**Tiempo necesario a 1 cifrado/μs**	**Tiempo necesario a 10^6 cifrado/μs**
32	$2^{32} = 4.3 \times 10^9$	$2^{31}\mu s = 35.8\ min$	2.15 milisegundos
56	$2^{56} = 7.2 \times 10^{16}$	$2^{55}\mu s = 1.142\ años$	10.01 horas
128	$2^{128} = 3.4 \times 10^{38}$	$21^{27}\mu s = 5.4 \times 10^{24}\ años$	$5.4 \times 10^{18}\ años$
168	$2^{168} = 3.7 \times 10^{50}$	$21^{67}\mu s = 5.9 \times 10^{36} años$	$5.9 \times 10^{30}\ años$

Tabla 3.2 Tiempo medio para la búsqueda exhaustiva de la clave[10]

Después de realizar los cálculos y ver el tiempo que se tomaría de acuerdo a la longitud, se decide por una clave de 256 bits o 32 caracteres en código ASCII, adicionalmente que la clave cuente con una combinación de mayúsculas, minúsculas y dígitos lo cual la hace más segura y por tanto requiere de mucho tiempo para poder deducir cual es la clave correcta.

[9] López Barrientos Ma. Jaquelina, Criptografía (Universidad Nacional Autónoma de México, 2009, P.123)
[10] Fundamentos de seguridad en redes. Aplicación y estándares. http://books.google.com.mx (visitada el 2 de Mayo del 2012)

3.1.2 Forma de procesar los datos:

¿Cómo se procesará la información? Como se mencionó en el primer capítulo se habla de dos formas para procesar datos, que son por flujo y bloques, ésta última utilizada para el diseño del algoritmo.

La decisión fue tomada a partir de saber que el cifrado por flujos se realiza bit a bit, lo que lo hace más vulnerable a un ataque, por ejemplo, mediante la frecuencia de caracteres en el o bien por previsibilidad de los primeros bits de cifrado como en RC4; sin embargo, al realizar el cifrado por bloques de misma longitud es más seguro, ya que se realiza un combinación de caracteres dentro del bloque, y permite hacer uso de cajas-S para tener una mayor confusión en el y al utilizar alguno de los modos de cifrado (CBC, OFB, etc.) permite mayor difusión del mensaje.

¿Cuál será la longitud de cada bloque? Los bloques deben ser del mismo tamaño, por lo que se decide realizar 4 bloques que fueran de 64 bits cada uno, considerando que es un tamaño regular, y eficiente para realizar las operaciones que requiere el algoritmo.

3.1.3 Planificación de subclaves

¿Por qué trabajar con bytes y no con bits? Esto es porque se trabajará en el sistema hexadecimal, ya que nos permite que las operaciones sean más sencillas de llevarse a cabo, y si fuera a nivel binario, sería más lento el proceso al realizar las operaciones entre cada uno de los bits.

¿Por qué se realiza una negación a la clave original? El motivo es para que desde un inicio antes de la interacción entre la clave y el mensaje, ésta ya cambiado, y con ello ir dando robustez a la clave.

¿Por qué generar las subclave a partir de la clave? Como se menciono en el capitulo anterior, las primeras 4 subclaves se obtienen de la clave original, mientras que las siguientes son derivadas de las 4 subclaves iniciales. Lo cual brinda mayor seguridad ya que son diseñadas con operaciones diferentes, por tanto, si el atacante logra capturar

una subclave no pone en riesgo descubrir la clave original ya que en cada ronda estas son modificadas, y todo esto en conjunto hace a las subclaves independientes.

¿Por qué se realizan rotaciones de 3 y 7 bits? Para realizar las rotaciones se buscan números que no sean potencias de 2, ya que la mayoría de los sistemas siempre son basados en potencias 2 y sería muy fácil la deducción por esta razón se pueden rotar palabras o bytes completos. En cambio realizar rotaciones con potencias que no sea 2, genera que se realice un más de trabajo para conocer el número de rotaciones que se le realizan al bloque y saber el sentido en que la realiza. Por lo tanto se toman la rotación de 7 y 3 bits de cada uno de los bytes que forman cada bloque, como se observa ninguno es potencia de 2.

¿Por qué hacer uso de las 4 cajas para generar una subclave? Esto es con la finalidad de enmascarar lo más posible los valores reales de la clave y del mensaje, es decir, al hacer uso de las 4 cajas se brinda mayor seguridad, puesto que no sólo se basa en obtener los valores de una sola caja, se deben conocer las 4 cajas, además de que dos bytes de cada bloque van a caja-S1, dos a caja-S2, así sucesivamente.

Si las cajas-S no fueran públicas, o más aun que cada usuario generara sus propias cajas, esto permitiría hacer más robusta la protección que se le da a la clave, puesto que se le agregan valores que no corresponden a la misma, lo cual provocaría que la deducción de las subclaves sea una tarea difícil.

¿Por qué modificar las subclaves en cada ronda? Conociendo las subclaves de la primer ronda se procede a generar las de la siguiente ronda, con la finalidad de que si el atacante deduce una, se le complique deducir las demás, ya que son independientes.

3.1.4 Mensaje

¿Cuál es la longitud que tendrá el mensaje? Tomando como base la longitud de la clave, el mensaje será partido de la misma forma para que pueda interactuar en el proceso de cifrado.

¿Por qué alterar el mensaje antes de iniciar la primera ronda? Al igual que la clave sufre una alteración antes de iniciar las rondas, con el mensaje pasa lo mismo, se realiza una rotación sobre cada uno de los bloques. Como se mencionó en el capítulo anterior se realiza la rotación de 5 bits a la derecha de cada uno de los bytes, con la finalidad de que desde un inicio entre al proceso de cifrado un mensaje completamente diferente al $Mcla$ original.

3.1.5 Justificación del diseño de las cajas-S

¿Cuál es la dimensión que tendrá cada una de las cajas? Tomando que son 256 bits se decide que sea la cantidad de números que contendrá la caja (0-256), de tal forma que la dimensión será 16 x 16, dichos valores estarán en hexadecimal.

¿Por qué elegir BBS para generar números pseudoaleatorios? El algoritmo Blum Blum y Shub es el más recomendado para utilizar en la Criptografía debido a que es muy resistente desde el punto de vista de la seguridad[11], lo que se relaciona con la calidad del generador en cuanto a la complejidad computacional de la factorización de enteros, esto es mediante una correcta elección de los números primos que el BBS requiere, y se hace uso del bit menos significativo de cada valor obtenido.

Si la factorización de enteros es difícil entonces BBS con grandes valores de N ($módulo\ N$) tendrán un resultado libre de todo patrón no aleatorio que puede ser descubierto mediante una cantidad razonable de cálculos. Esto hace que el método sea tan seguro como otras tecnologías de cifrado asociadas al problema de factorización.

BBS es un algoritmo seguro por ello es utilizado para generar los números necesarios, entonces, lo que se requiere son números primos (en este caso se eligieron números de 5 dígitos). El motivo es porque entre mayor sea el número primo, los valores tendrán menos probabilidad de que se repitan frecuentemente.

Al contar con los números primos y llevar a cabo el algoritmo para generarlos, se van obteniendo una serie de números, de los cuales se decide utilizar el bit menos

[11] http://oa.upm.es/193/1/09200317.pdf (visitada el 14 de Marzo del 2012)

significativo con los cuales se van a formar los valores finales, a continuación se muestra en negritas el bit menos significativo.

$x = 805769636 \rightarrow \ 11000000000111000100011010001\mathbf{0}$

$x = 856030346 \rightarrow \ 11001100000101111111001000101\mathbf{0}$

$x = 468454322 \rightarrow \ 11011111011000000101110110\mathbf{0}$

$x = 659684356 \rightarrow \ 10011101010001111111000000010\mathbf{0}$

Al tomar el bit más significativo, se tiene que rellenar con 0 para que el tamaño de todos los valores sea de la misma longitud, de tal forma que los números finales que se generan a partir de los valores obtenidos serían muy repetitivos, lo cual provocaría que se desprecien varios valores ya que se buscan sean 256 valores diferentes tomando como base la longitud de la clave.

$x = 805769636 \rightarrow \ \mathbf{1}10000000001110001000110100100$

$x = 856030346 \rightarrow \mathbf{1}11001100000101111111001000101\mathbf{0}$

$x = 468454322 \rightarrow \mathbf{1}11011111011000000101110110\mathbf{0}$

$x = 659684356 \rightarrow \mathbf{1}00111010100011111111000000010\mathbf{0}$

Para formar un valor que será introducido a la caja, se necesita el cálculo de 8 valores de x, del cual al extraer el bit menos significativo se transformara a hexadecimal, finalmente se introducen a la caja de forma horizontal.

¿Por qué generar cajas inversas? Esto es para poder hacer reversible el proceso, es decir, que el cifrado se puede realizar como una red Feistel, si al momento de generar los números para las cajas S se hace una intersección entre la caja S y la S inversa; es decir, cuando se genera por ejemplo A1 con 45 entonces en la misma caja S se rellena la celda destinada a 45 con A1.

Entonces si en la caja s1 en la casilla A1 se tiene 45, para la inversa se tendrá en la casilla 45 el valor de A1 (véase figura 3.1), de tal forma que regresa al valor original.

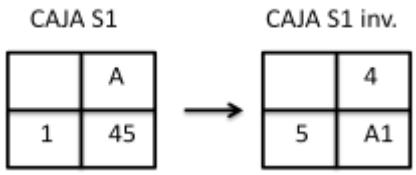

Figura 3.1: Caja s1 e inversa

3.1.6 Justificación de operaciones

¿Por qué utilizar la operación XOR? La XOR es una operación sencilla pero permite enmascarar fácilmente los valores reales, se decide llevar a cabo la XOR en los 4 bloques, al inicio para difuminar el mensaje con la subclave correspondiente de manera directa, es decir, la k1 con m1 y así sucesivamente.

Además, de que la XOR es una operación que utiliza muy pocos recursos de un equipo de cómputo, lo que permitirá que al ser implementado en algún lenguaje el proceso de cifrado y descifrado no sea tan demandante.

¿Por qué implementar la suma modular? Se decide realizar una operación más, pero buscando que no genere alguna asociación con la XOR, por tal motivo se decide implementar una suma modular.

Dentro de la suma modular, el módulo es en base a la longitud de la clave, es decir 256, pero trabajando en sistema hexadecimal corresponde al $mod100$. Dicha operación se realiza byte a byte, esto es por facilidad y para que cada byte sea afectado del mismo modo y así evitar procesos demandantes al calcular un mod 2^{256}.

Por otro lado se tiene el uso de las cajas-S, en donde al igual que las subclaves, los bytes no sólo son introducidos en una caja, sino que se hace uso de las 4, con la finalidad de enmascarar los valores de las subclaves.

¿Por qué realizar una rotación después del uso de las cajas-S? Al igual que en las subclaves se usa para difuminar los datos. La diferencia es que en esta rotación se toma el conjunto de los 4 bloques, para trabajarlo como uno solo con la finalidad de lograr el efecto en cascada, que es básicamente, observar que tanto van cambiando los

bytes con el simple hecho de modificar uno solo, de tal forma que se obtenga el mayor porcentaje de bytes alterados entre un mensaje y otro.

Por ejemplo, se tiene un $Mcla_1$= hola, k= adiós y para observar el efecto en cascada se usa el $Mcla_2$= Hola, es decir, solo se modifica un byte para el $Mcla_2$, en el apéndice efecto en cascada, se muestra cómo se van modificando los bytes en cada ronda con solo cambiar un byte en el segundo mensaje en claro.

El efecto en cascada, justamente sirve para decidir cuál será el número de rondas que tendrá el algoritmo, como se muestra en el *apéndice 1: efecto en cascada*, se observa que en la ronda 16 los bytes son diferentes totalmente, por lo tanto se considera realizar el doble de rondas para difuminar mas los datos, para establecer mayor seguridad.

¿Por qué combinar las salidas que se convertirán en entradas? Al final del algoritmo se obtienen 4 salidas las cuales pasan a ser lo que era el mensaje de entrada en la primera ronda, solo que se combinan es decir:

$$m1 = salida3$$

$$m2 = salida1$$

$$m3 = salida4$$

$$m4 = salida2$$

Con la finalidad de difuminar más la información que se tiene, y con ello hacer que el algoritmo sea más robusto, ante algún ataque de tipo estadístico al evaluar la distribución de bits iniciales y finales del mensaje.

3.2 Funcionamiento del algoritmo

A continuación se mostrará un ejemplo del algoritmo con el procedimiento obtenido del programa realizado en Dev-C++ con la finalidad de comprobar el funcionamiento correcto del mismo, y automatizar la ejecución de las 32 rondas.

3.2.1 Proceso de cifrado

Se ejecuta el programa, al iniciar pide al usuario que introduzca la clave (véase figura 3.2).

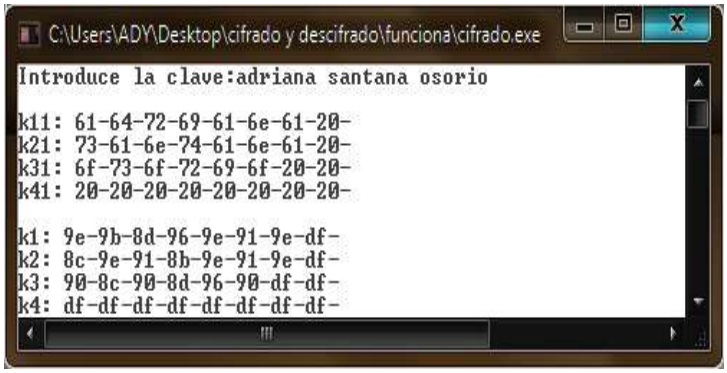

Figura 3.2: Programa pide la clave al usuario

El programa al inicio muestra las subclaves $k11 - k41$ partiendo de la clave original (véase figura), posteriormente se observan las subclaves $k1 - k4$ negadas (véase figura 3.2), las cuales son la base para generar las subclaves $k1 - k8$. Los espacios en blanco también son considerados en la generación de las subclaves.

Una vez que se ingresa la clave, pide al usuario ingrese el mensaje (véase figura 3.3).

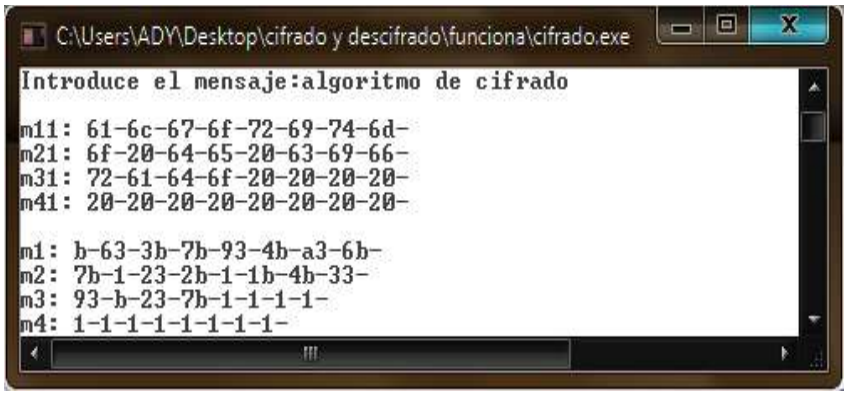

Figura 3.3 Programa pide ingresar mensaje

El programa muestra el mensaje original en hexadecimal($m11 - m41$), después realiza la rotación de bytes, de acuerdo a lo establecido en el diseño para generar m1-m4 (véase figura 3.3), dichos valores se usara para iniciar el proceso de cifrado.

Posteriormente se genera las subclaves k1-k8 que se usan para cifrar (véase figura 3.4).

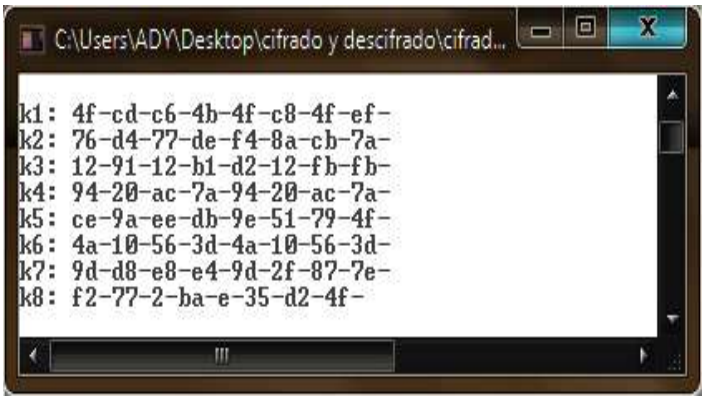

Figura 3.4 Se muestran las subclaves k1-k8

RONDA 1

Al tener la clave y mensaje, el programa inicia el procedimiento para cifrar la información, en la figura 3.5 se muestra el procedimiento que realiza para la primer ronda así mismo se muestran los valores de salida, obtenidos en dicha ronda, el procedimiento es igual para las 32 rondas.

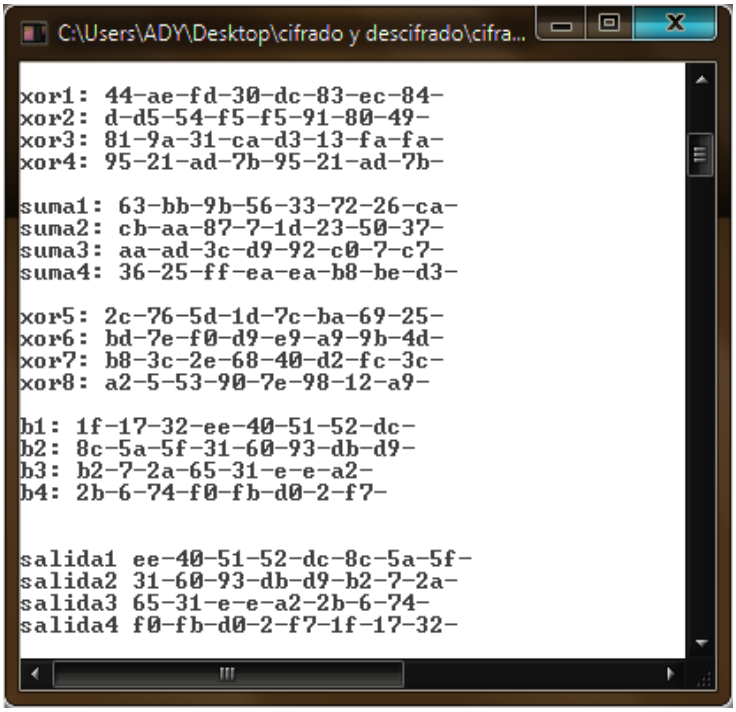

Figura 3.5 Ronda 1 de cifrado

RONDA 2

Los valores de salida que se obtienen en la ronda 1, serán ahora las entradas para la ronda 2, como se puede observar en la figura 3.6 las subclaves son diferentes a las utilizadas en la primer ronda (véase figura 3.4). El procedimiento es el mismo en cuanto a operaciones.

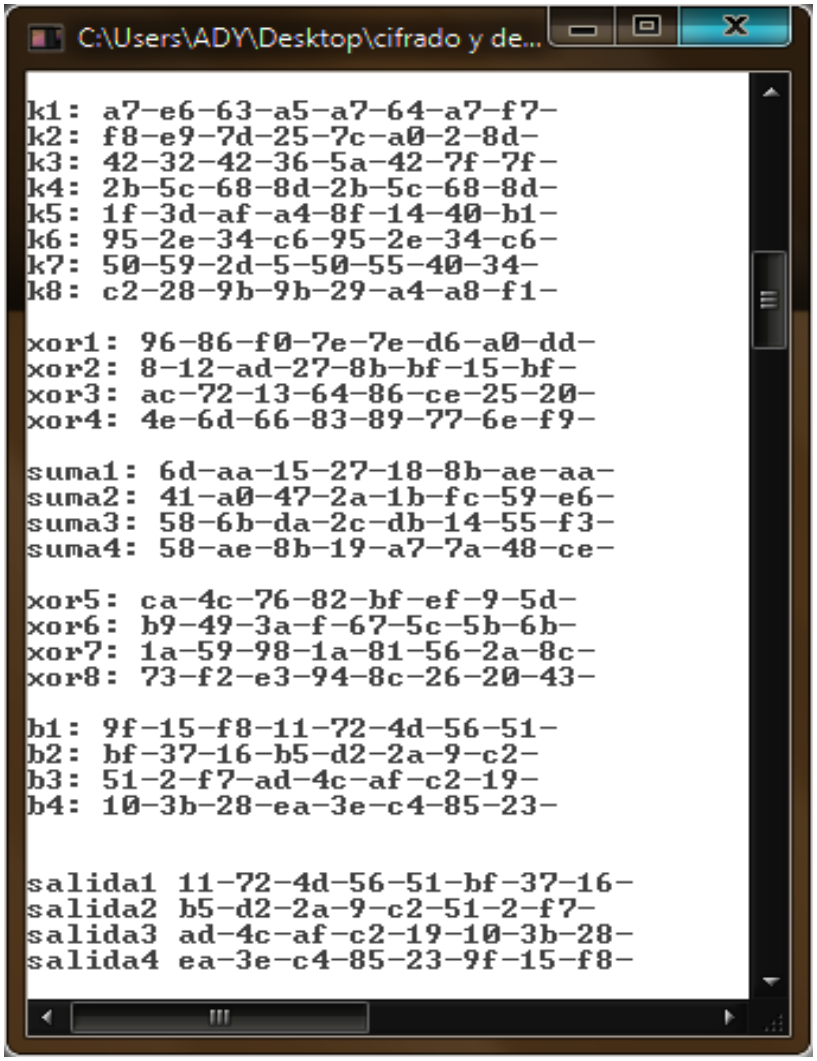

Figura 3.6 Ronda 2 de cifrado

RONDA 3

Como se muestra (véase figura 3.7), el procedimiento es el mismo, lo único que se va modificando son las subclaves, y de la misma forma la salida de la ronda anterior es la entrad para esta ronda.

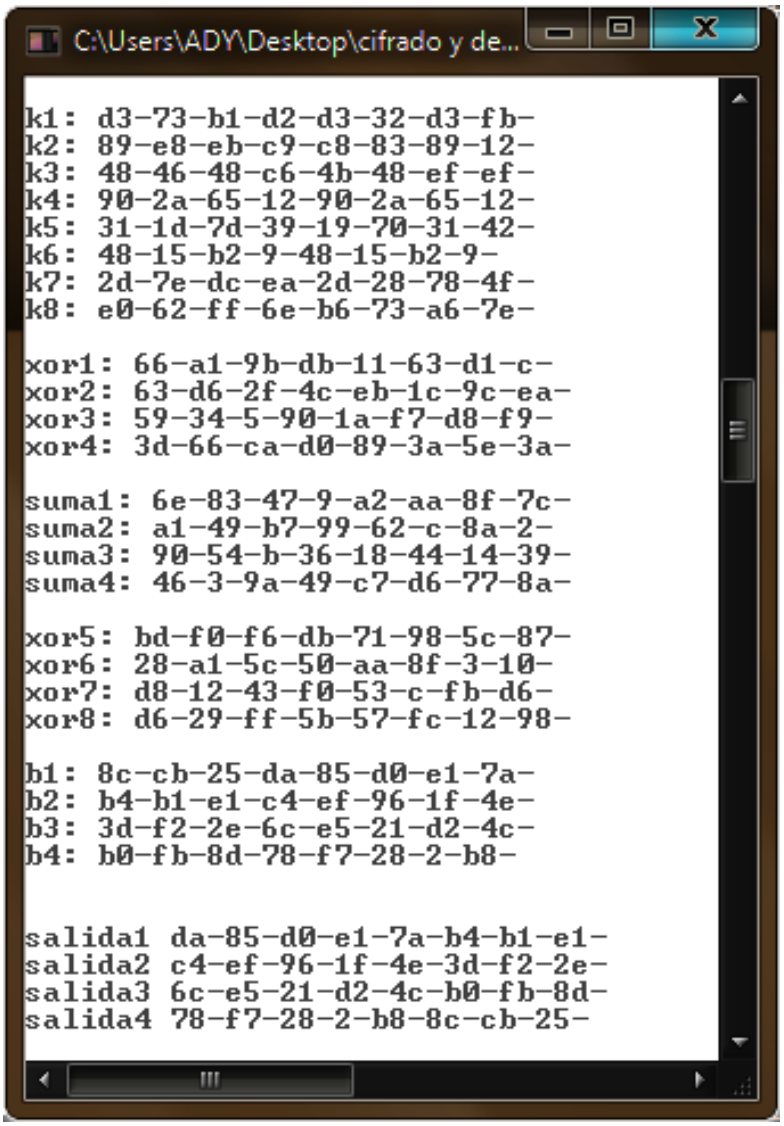

Figura 3.7 Ronda 3 de cifrado

El procedimiento para cada una de las rondas es el mismo, por lo tanto solo se muestran las pantallas de las 3 primeras rondas y las 3 últimas, con la finalidad de comparar con los valores obtenidos en las mismas rondas del descifrado.

RONDA 30 (véase figura 3.8)

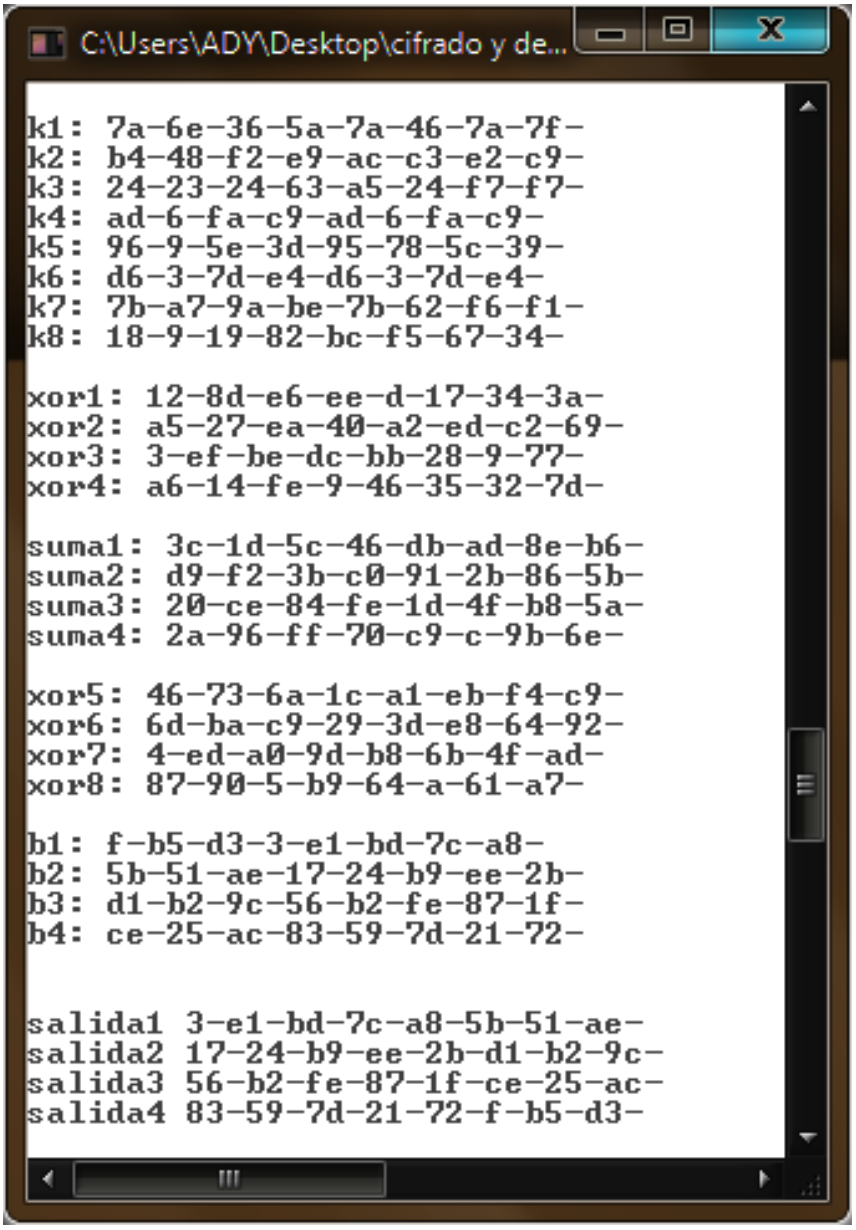

Figura 3.8 Ronda 30 de cifrado

RONDA 31 (véase figura 3.9)

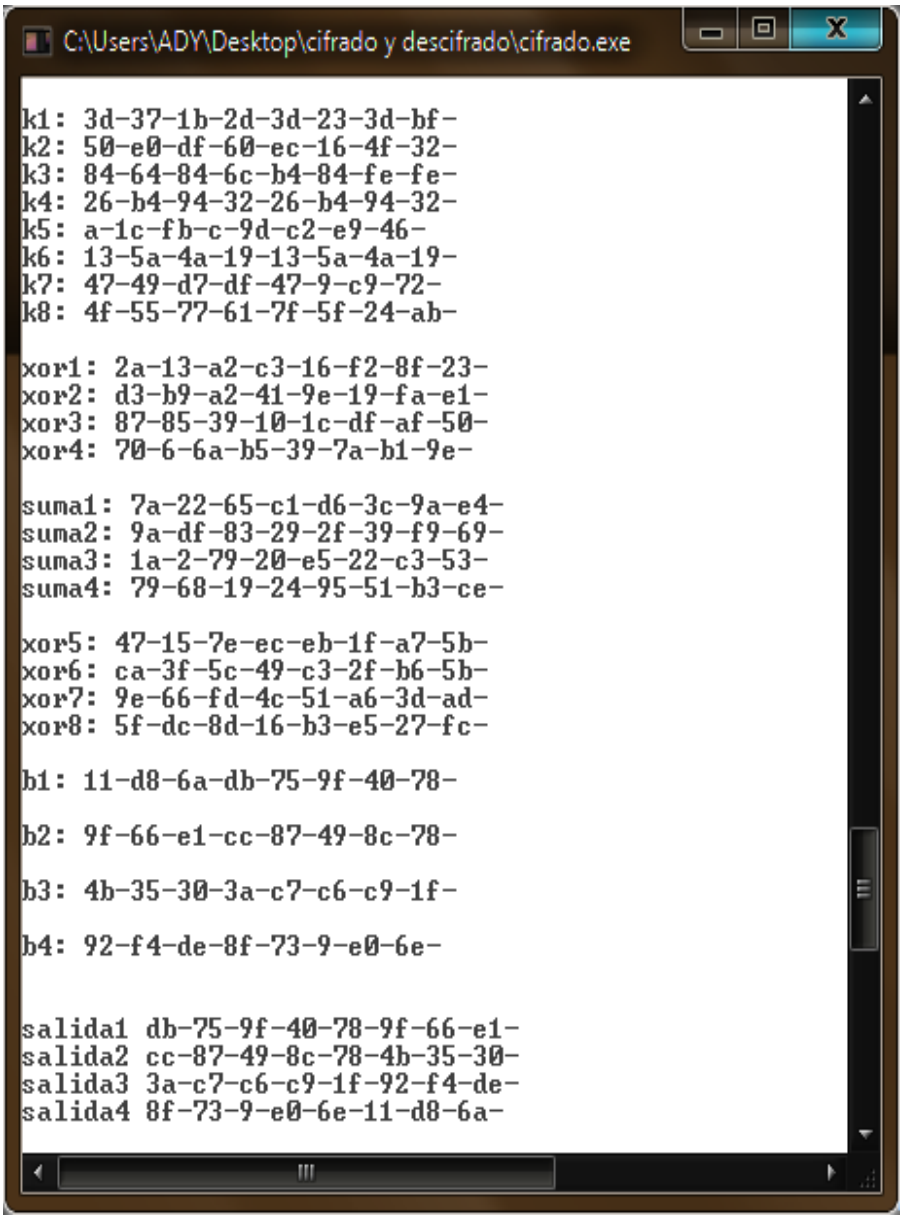

Figura 3.9 Ronda 31 de cifrado

RONDA 32:

Finalmente se tiene la última ronda (véase figura 3.10) en la cual se muestra el mensaje cifrado.

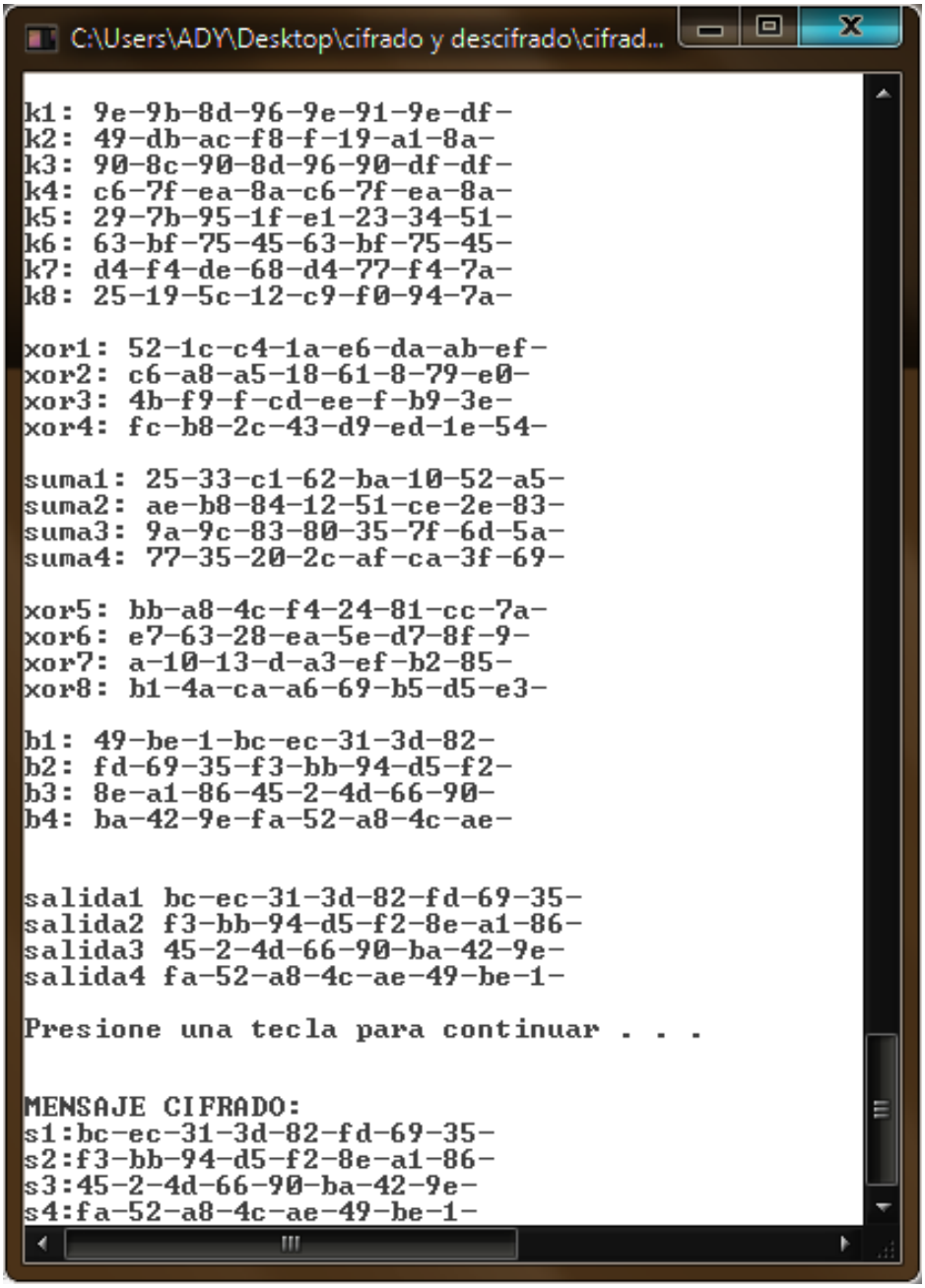

Figura 3.10 Ronda 32 de cifrado

Descifrado

A continuación se muestra el proceso de descifrado que se realiza por medio del programa. En la figura 3.11 el programa muestra cual es el mensaje que se desea descifrar, posteriormente pide la clave al usuario, como se sabe es la misma que se usa para cifrar.

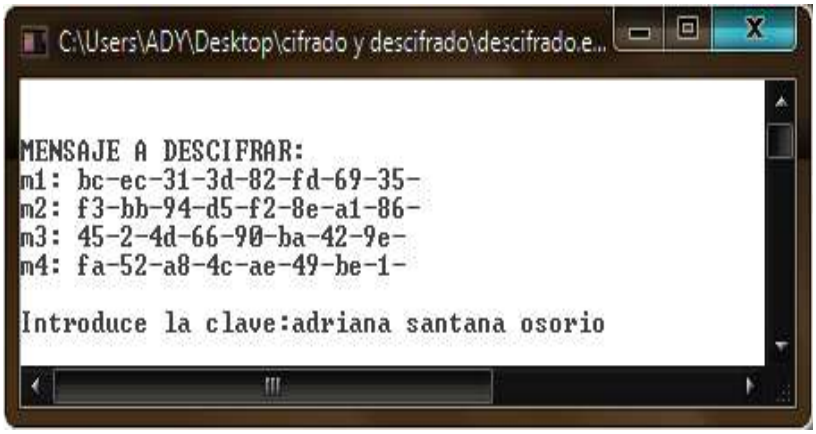

Figura 3.11 Pide la clave para el descifrado

En la figura 3.12 se muestran las subclaves para iniciar el descifrado, comparando con las que se muestran en la figura 3.10 se observa que son las mismas, es decir, las subclaves utilizadas en la última ronda del cifrado son las usadas en la primer ronda del descifrado.

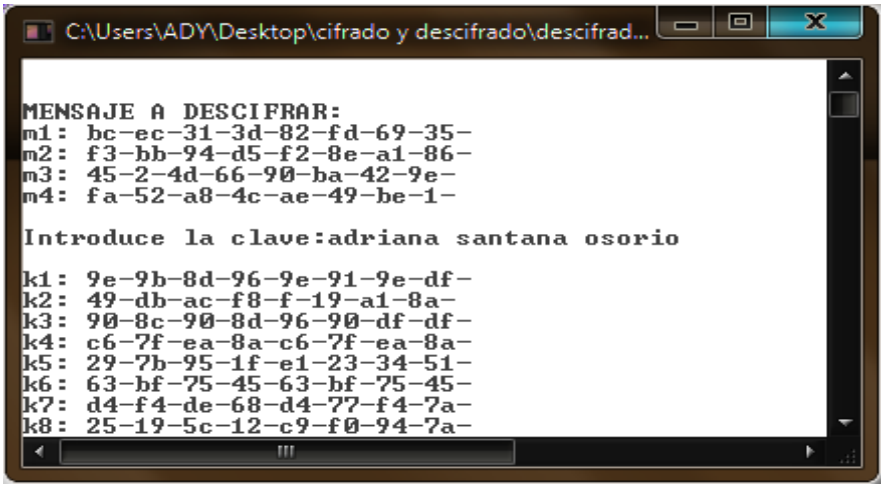

Figura 3.12 Subclaves iniciales para el descifrado

RONDA 1:

En la figura 3.13 se muestra el procedimiento de descifrado que corresponde a la ronda 1, SE OBSERVA que se obtienen los valores del bloque al que se le realiza una rotación en sentido inverso al cifrado, el procedimiento de la XOR es el mismo, y en el caso de la suma se realiza el inverso, la resta modular, posteriormente se realiza otro XOR, para obtener valores de salida que corresponden a la entrada de la siguiente ronda.

Pueden compararse los valores de la figura 3.10 con la 3.13, cifrado y descifrado respectivamente, donde los valores corresponden al proceso inverso.

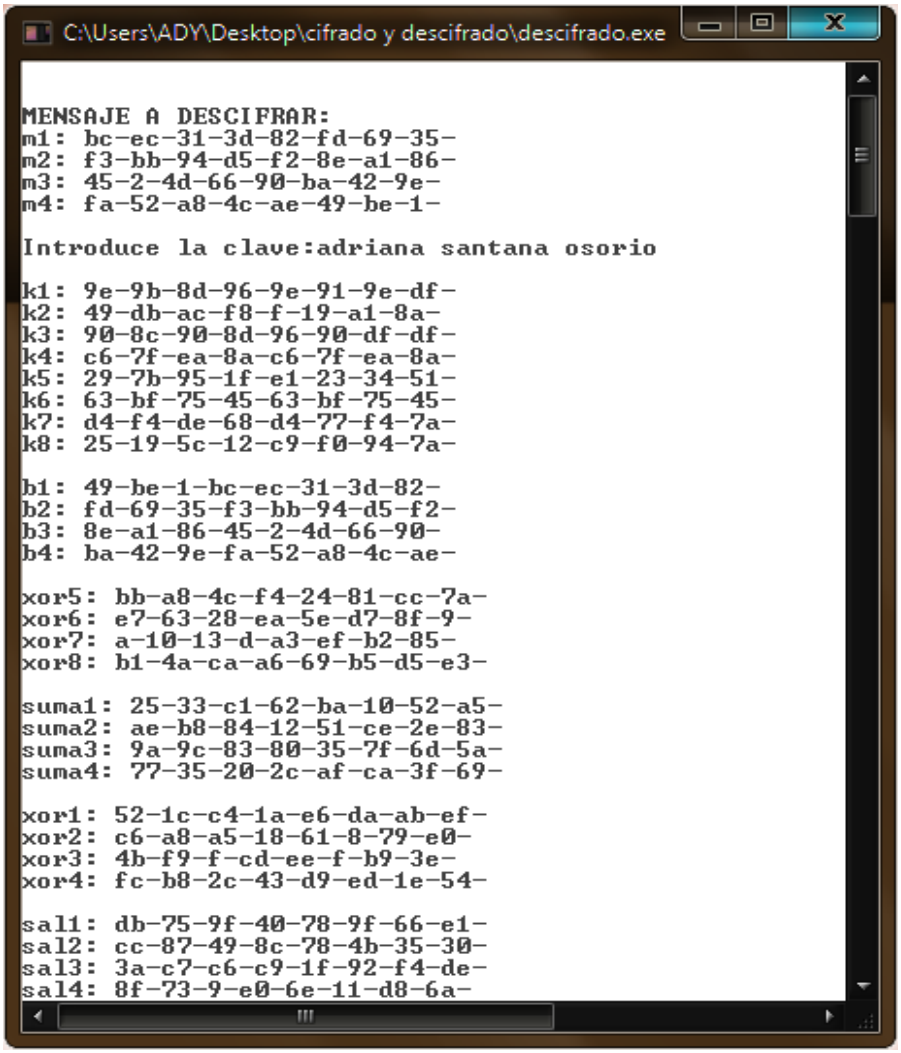

Figura 3.13 Ronda 1 de descifrado

RONDA 2

El procedimiento es el mismo que para la primer ronda (véase figura 3.14), ahora pueden compararse los valores con los de la ronda 31 del cifrado (véase figura 3.9).

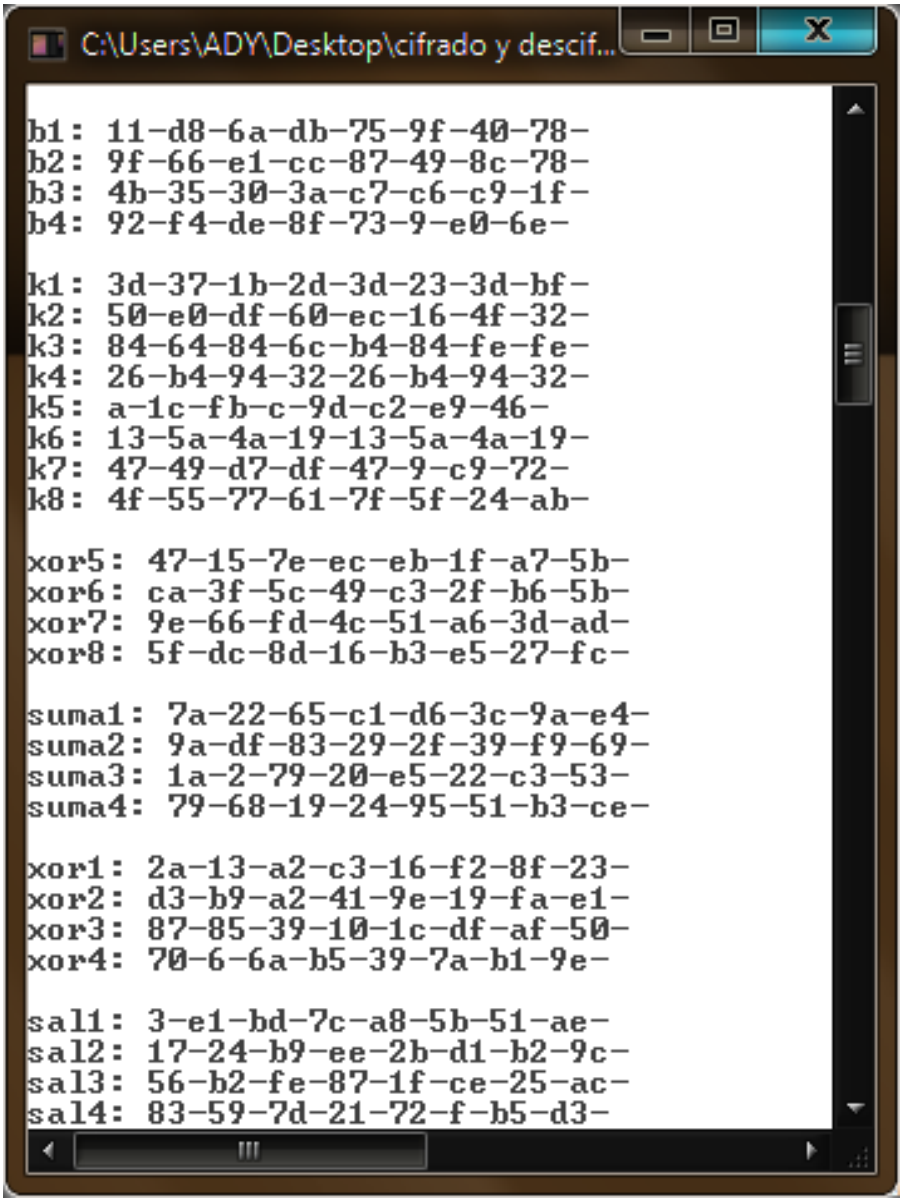

Figura 3.14 Ronda 2 de descifrado

RONDA 3 DESCIFRADO

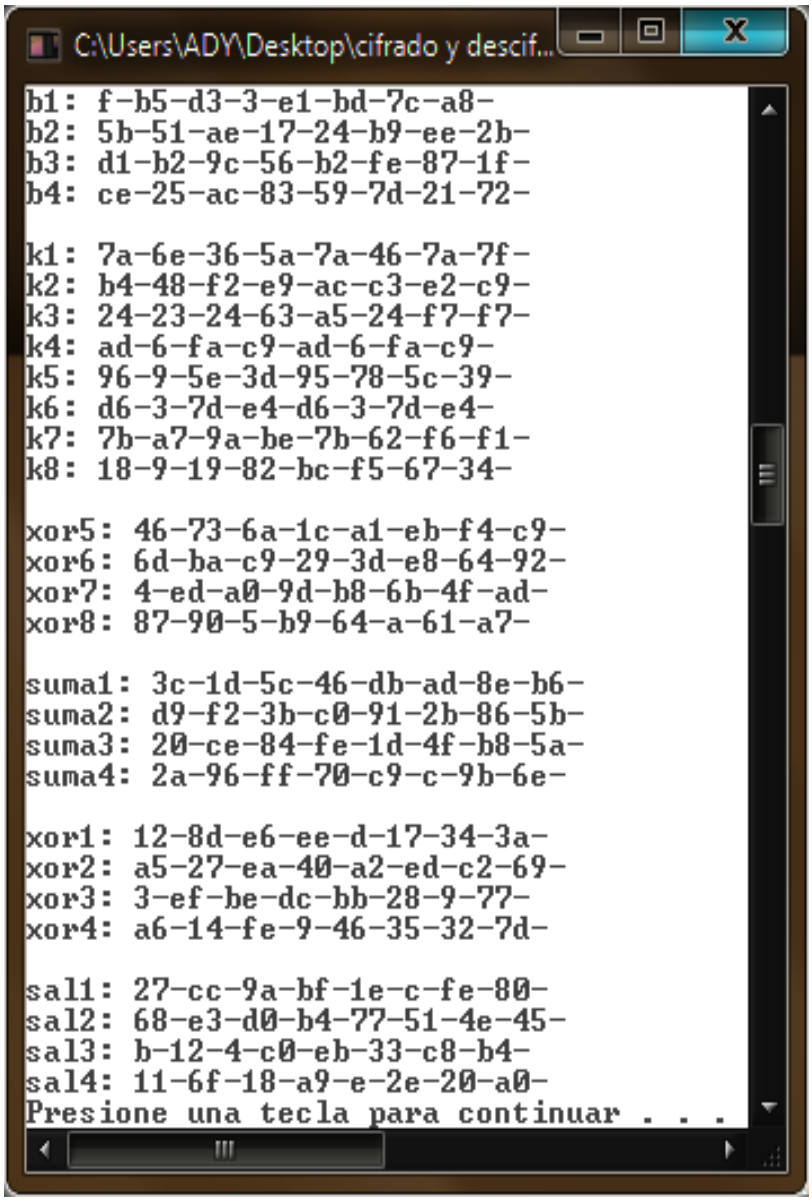

Figura 3.15 Ronda 3 de descifrado

RONDA 30

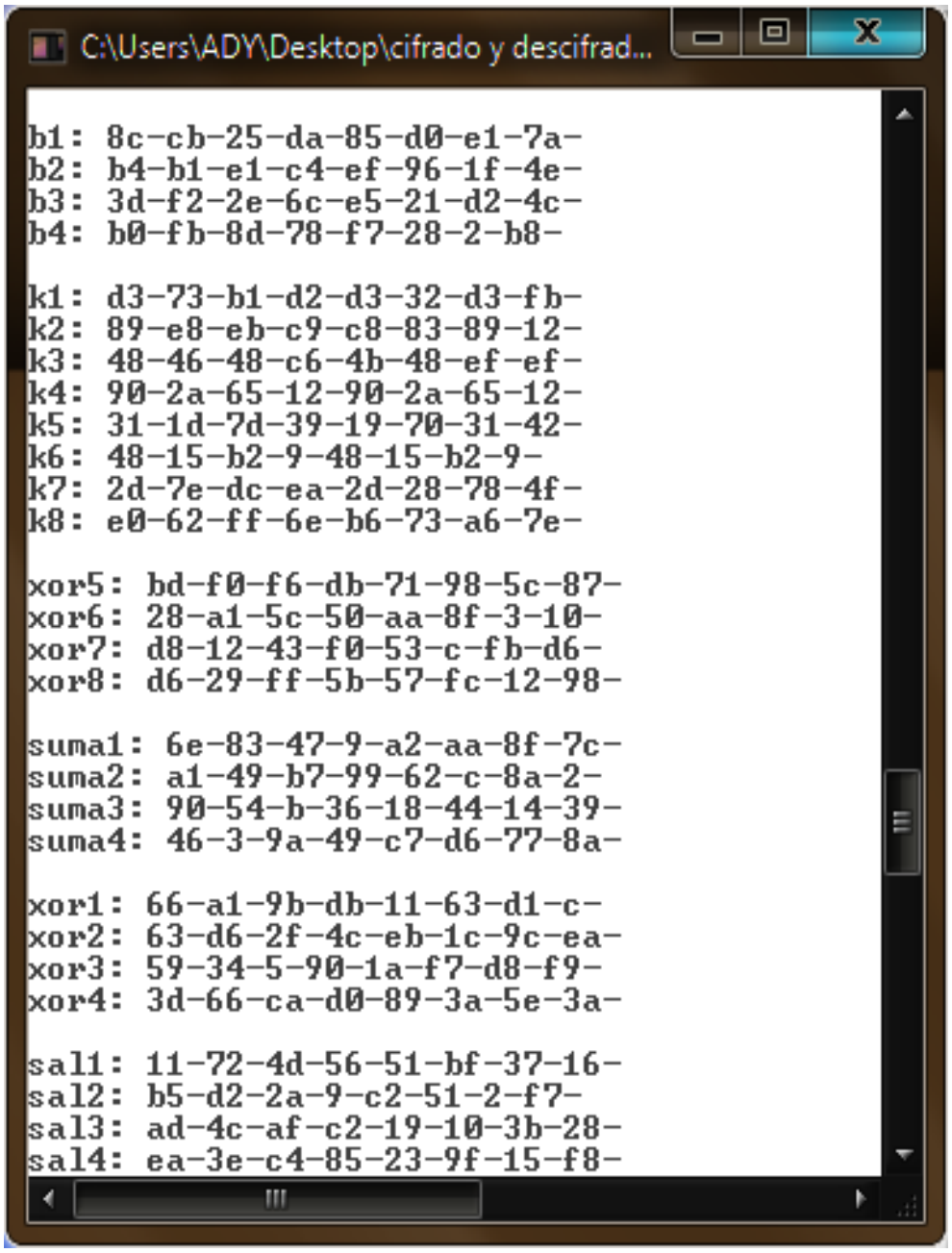

Figura 3.16 Ronda 30 de descifrado

RONDA 31

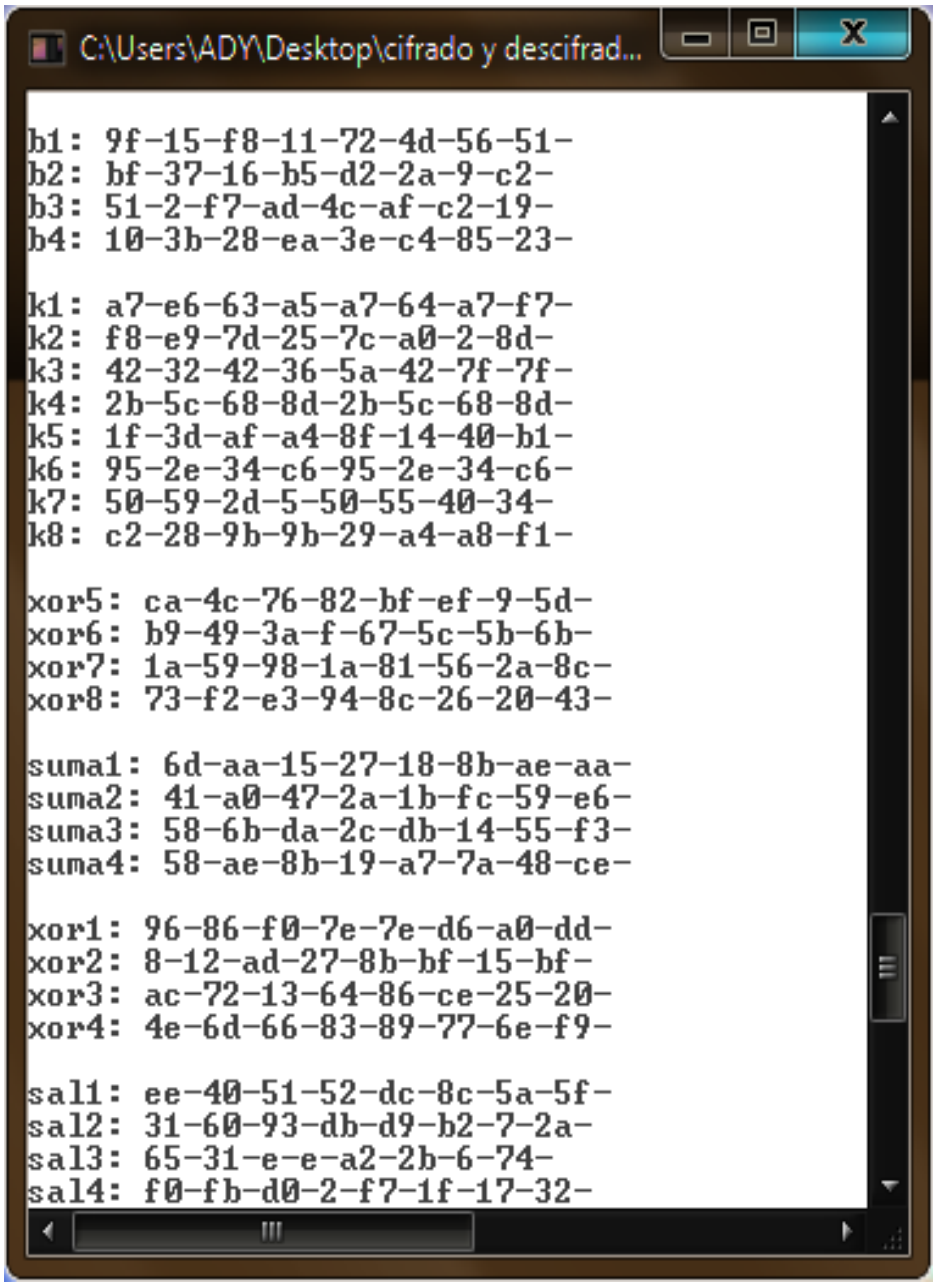

Figura 3.17 Ronda 31 de descifrado

RONDA 32

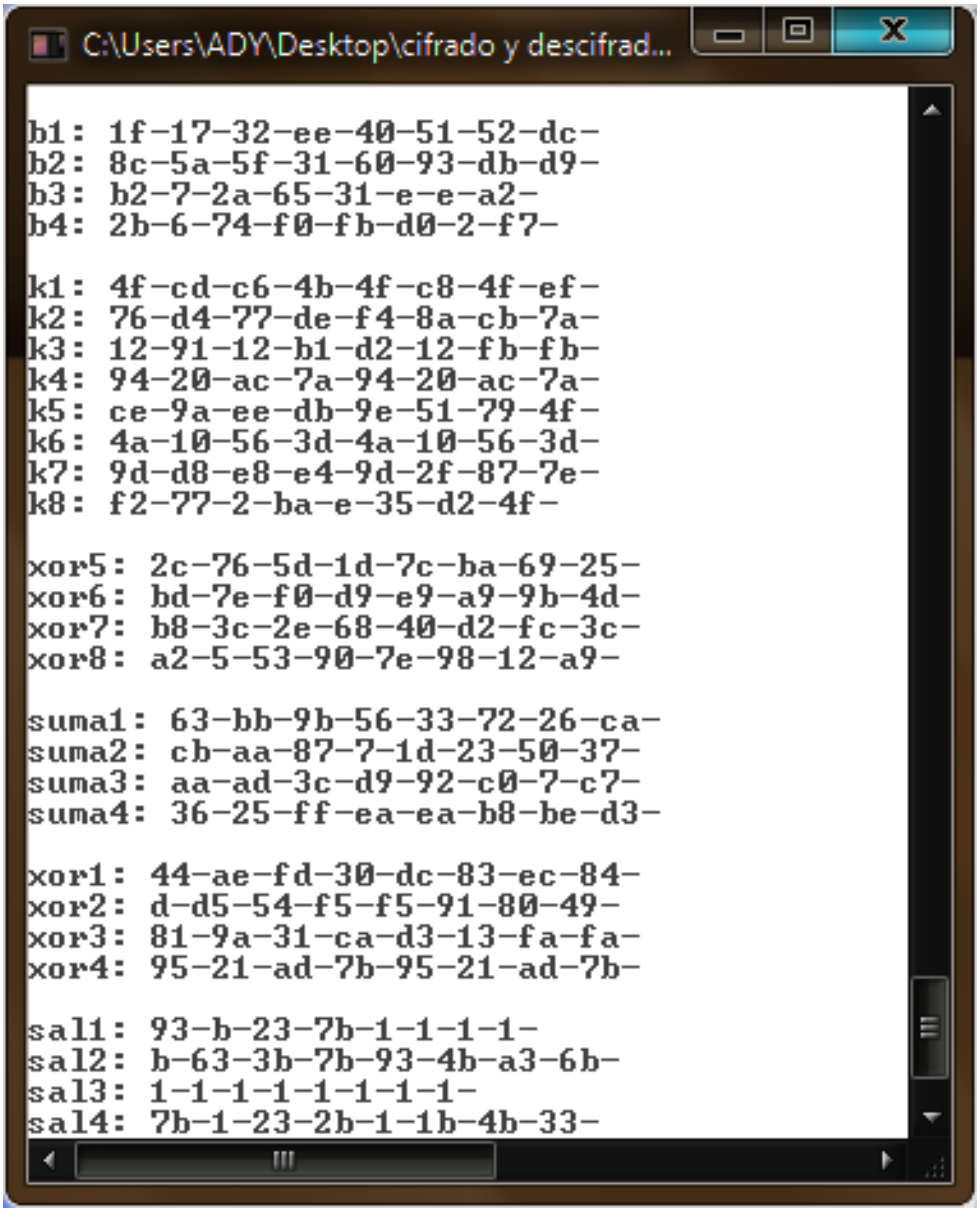

Figura 3.18 Ronda 32 de descifrado

MENSAJE DESCIFRADO

En la figura 3.19 se observa el mensaje en claro que se obtiene con el proceso de descifrado, después de realizar la rotación de bytes a los valores obtenidos en la salida de la ronda 32 (véase figura 3.18) puede compararse con los valores de la figura 3.3, puede verse que corresponden al mensaje que el usuario introdujo para cifrar.

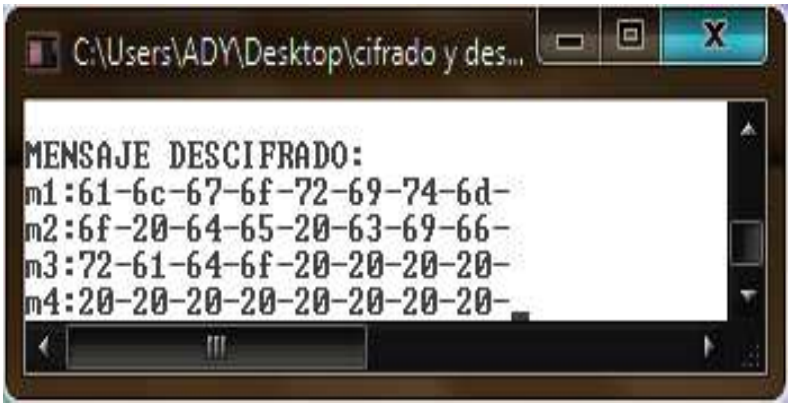

Figura 3.19 Mensaje en claro después del descifrado

Por lo tanto queda demostrado que el algoritmo cifra y descifra correctamente, al ser probado manualmente como se realizo en el capítulo anterior, y ahora por medio de un programa en Dev-C++.

CAPÍTULO 4

PRUEBAS
DE
CRIPTOANÁLISIS DIFERENCIAL

PRUEBAS CRIPTOANÁLISIS DIFERENCIAL

4.1 Criptoanálisis diferencial

Una vez que se tiene diseñado y funcionando correctamente el algoritmo, que como se mencionó en su diseño se busca la seguridad y robustez del mismo, se decide realizar una prueba con un tipo de criptoanálisis, en este caso el diferencial, porque es uno de los más usuales y más sencillo de llevar a cabo.

Se sabe que existen más tipos de criptoanálisis, como el lineal sólo que el objetivo es demostrar si es resistente contra el criptoanálisis diferencial.

4.2 Pruebas de criptoanálisis

Para realizar las pruebas de criptoanálisis se requiere de una serie de mensajes en claro ($Mcla$), donde cada uno será cifrado con la misma clave. Los datos necesarios son la XOR de entrada y la XOR de salida, es decir, la XOR1 y XOR8, correspondientes con el algoritmo simétrico (véase figura 1), cabe mencionarse que en esta parte es donde se rompe la asociación entre las operaciones XOR y la suma modular, ya que las diferencias tanto de entrada como de salida se concentran en la entrada y salida de la suma modular.

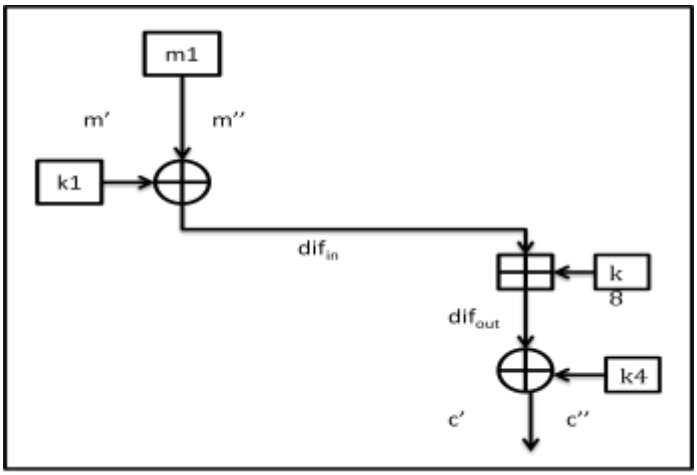

Figura 4.21: Diagrama de bloque 1 dentro del algoritmo

La XOR1 se utiliza ya que es la primera interacción entre el mensaje con la clave, mientras que la XOR8 es aquella que sigue el recorrido que realiza el primer bloque de datos.

De acuerdo a la figura 4.1 se tiene que:

$m' = Mcla1,\ \ m'' = Mcla2$	$c'\ y\ c''\ son\ los\ valores\ despues\ de\ \ XOR8$
$(m'\oplus k1)\oplus(m''\oplus k1) =$	$(c'\oplus k8)\oplus(c''\oplus k8) =$
$(m'\oplus k1)\oplus(k1\oplus m'') =$	$(c'\oplus k8)\oplus(k8\oplus c'') =$
$m'\oplus(k1\oplus k1)\oplus m'') =$	$c'\oplus(k8\oplus k8)\oplus c'' =$
$m'\oplus(0x0)\oplus m'' =$	$c'\oplus(0x0)c'' =$
$m'\oplus m'' = dif_{in}$	$c'\oplus c'' = dif_{out}$

Con esto se permite obtener las diferencias de entrada y salida, que nos permitirán realizar los cálculos necesarios para determinar los posibles valores de la subclave que corresponda a la ronda y bloque adecuado.

El procedimiento que se utiliza para generar los posibles valores que pueden pertenecer a la claves es: se toman dos mensajes, con una misma clave, se realiza el cifrado por medio del algoritmo implementado en Dev-C++, y se obtienen los valores de las XOR mencionadas anteriormente.

$Mcla_1 = \ análisis, \quad Mcla_2 = diferencial, \quad k = abcdefgh$

Los resultados obtenidos de las XOR de entrada para ambos mensajes son los siguientes:

$xor_{m1} = 44\ bd\ 45\ ae\ 06\ 57\ 07\ 50$

$xor_{m2} = 6c\ 85\ 7d\ e6\ de\ e7\ 3f\ d0$

Con estos datos se calcula la diferencia de entrada (dif_{in}) como se mostró anteriormente; para ello se toma el primer byte de ambos XOR y se realiza la XOR como se muestra.

$dif_{in} = m1\oplus m2 = 44 \oplus 6c = 28$

Ahora se procede a obtener la diferencia de salida (dif_{out}), se toma el primer byte de la XOR8 de ambos mensajes, y se tiene:

$c1 = a7\ 11\ bb\ 87\ 61\ eb\ 75\ e5$

$c2 = cf\ d9\ e3\ 4f\ 59\ 7b\ bd\ 65$

$dif_{out} = c1 \oplus c2 = a7 \oplus cf = 68$

El procedimiento para obtener los valores de diferencia de entrada y salida para los primeros 5 bytes son los siguientes (véase tabla 4.1), los cuales fueron calculados de la misma forma sólo que respetando el byte correspondiente.

PAREJA 1:

$k = abcdefgh, \quad Mcla_1 = analisis, \quad Mcla_2 = diferencial$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	44	6c	28
1	bd	85	38
2	45	7d	38
3	ae	e6	48
4	06	de	d8

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	a7	cf	68
1	11	d9	c8
2	bb	e3	58
3	87	4f	c8
4	61	59	38

Tabla 4.1: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 1 de M_{cla}

Se realiza el mismo procedimiento para otras 9 parejas de texto, con la finalidad de realizar un mayor filtro, e ir descartando valores.

A continuación se muestran las parejas de texto utilizadas y los valores correspondientes a las diferencias de entrada y salida para cada pareja (véase tablas 4.2-4.32).

PAREJA 2:

$k = abcdefgh, \quad Mcla_1 = adriana, \quad Mcla_2 = 1234567$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	44	c6	82
1	ed	5f	b2
2	dd	d7	0a
3	08	6c	ea
4	46	e4	a2

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	a7	21	86
1	41	f3	b2
2	03	05	06
3	af	c1	6e
4	a1	47	e6

Tabla 4.2: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 2 de M_{cla}

PAREJA 3:

$k = abcdefgh, \quad Mcla_1 = criptografia, \quad Mcla_2 = ataque$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	54	44	10
1	5d	6d	30
2	05	45	40
3	4e	46	08
4	ee	e6	08

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	d7	a7	70
1	f1	c1	30
2	7b	bb	C0
3	e7	ef	08
4	49	41	08

Tabla 4.3: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 3 de M_{cla}

PAREJA 4:

$k = abcdefgh, \quad Mcla_1 = criptografia, \quad Mcla_2 = seguridad$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	54	d4	80
1	5d	e5	B8
2	05	75	70
3	4e	66	28
4	ee	de	30

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	d7	57	80
1	f1	79	88
2	7b	eb	90
3	e7	cf	28
4	49	59	10

Tabla 4.4: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 4 de M_{cla}

PAREJA 5:

$k = abcdefgh, \quad Mcla_1 = h0l4, \quad Mcla_2 = 4d105$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	0c	ee	e2
1	4f	ed	a2
2	2d	c7	ea
3	6c	4c	20
4	4c	e4	a8

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	6f	49	26
1	e3	41	a2
2	53	35	66
3	c1	e1	20
4	af	47	e8

Tabla 4.5: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 5 de M_{cla}

PAREJA 6:

$k = abcdefgh, \quad Mcla_1 = clave, \quad Mcla_2 = cifrado$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	54	54	00
1	ad	85	28
2	45	7d	38
3	7e	5e	20
4	66	46	20

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	d7	d7	00
1	01	d9	d8
2	bb	e3	58
3	b7	d7	60
4	c1	a1	60

Tabla 4.6: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 6 de M_{cla}

PAREJA 7:

$k = abcdefgh, \quad Mcla_1 = adriana\ santana, \quad Mcla_2 = aldo\ jimenez$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	44	44	00
1	ed	ad	40
2	dd	6d	b0
3	86	b6	30
4	46	4c	0a

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	a7	a7	00
1	41	01	40
2	03	93	90
3	af	7f	d0
4	a1	af	0e

Tabla 4.7: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 7 de M_{cla}

PAREJA 8:

$k = abcdefgh, \quad Mcla_1 = ataque, \quad Mcla_2 = pasivo$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}	byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	44	cc	88	**0**	a7	2f	88
1	6d	c5	a8	**1**	c1	19	d8
2	45	d5	90	**2**	bb	0b	b0
3	46	86	c0	**3**	ef	af	40
4	e6	fe	18	**4**	41	79	38

Tabla 4.8: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 8 de M_{cla}

PAREJA 9:

$k = abcdefgh, Mcla_1 = criptoanalisis\ lineal, Mcla_2 = algoritmo\ simetrico$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}	byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	54	44	10	**0**	d7	a7	70
1	5d	ad	f0	**1**	f1	01	f0
2	05	75	70	**2**	7b	eb	90
3	4e	b6	f8	**3**	e7	7f	98
4	ee	de	30	**4**	49	59	10

Tabla 4.9: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 9 de M_{cla}

PAREJA 10:

$k = abcdefgh, \quad Mcla_1 = criptosistema, \quad Mcla_2 = blum\ blum\ y\ shub$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}	byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	54	5c	08	**0**	d7	df	08
1	5d	ad	f0	**1**	f1	01	f0
2	05	e5	e0	**2**	7b	1b	60
3	4e	a6	e8	**3**	e7	8f	68
4	ee	4c	a2	**4**	49	af	e6

Tabla 4.10: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 10 de M_{cla}

PAREJA 11:

$k = abcdefgh, \quad Mcla_1 = definicion, \quad Mcla_2 = redes\ inalambricas$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	6c	04	68
1	e5	bd	58
2	7d	75	8
3	86	e6	60
4	3e	3e	00

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	cf	5f	a8
1	79	79	68
2	e3	93	08
3	af	4f	e0
4	b9	51	00

Tabla 4.11: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 11 de M_{cla}

PAREJA 12:

$k = abcdefgh, \quad Mcla_1 = la\ comunicación, \quad Mcla_2 = EL\ TERMINO$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	2c	65	49
1	c5	ac	69
2	4f	4f	00
3	d6	6f	b9
4	36	67	51

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	8f	c0	4f
1	19	00	19
2	8d	8d	00
3	5f	c4	9b
4	b1	c2	73

Tabla 4.12: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 12 de M_{cla}

PAREJA 13:

$k = abcdefgh, \quad Mcla_1 = c1fr4d0123, \quad Mcla_2 = AdR14n4$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	54	45	11
1	47	ed	aa
2	7d	dc	a1
3	5e	44	1a
4	ec	ec	00

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	d7	a0	77
1	9b	41	da
2	e3	02	e1
3	d7	e9	3e
4	4f	4f	00

Tabla 4.13: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 13 de M_{cla}

PAREJA 14:

$k = abcdefgh, \quad Mcla_1 = 135pruebas, \quad Mcla_2 = InG3n13R14$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	c6	05	c3
1	57	bd	ea
2	e7	74	93
3	4e	54	1a
4	de	3e	e0

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	21	60	41
1	eb	11	fa
2	15	ea	ff
3	e7	d9	3e
4	59	b9	e0

Tabla 4.14: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 14 de M_{cla}

PAREJA 15:

$k = abcdefgh, \quad Mcla_1 = procesos, \quad Mcla_2 = transferir$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	cc	ec	20
1	5d	5d	0
2	35	45	70
3	d6	be	68
4	66	d6	b0

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	2f	4f	60
1	f1	f1	0
2	ab	bb	10
3	5f	77	28
4	c1	51	90

Tabla 4.15: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 15 de M_{cla}

PAREJA 16:

$k = abcdefgh, \quad Mcla_1 = tutorial\ de\ voz\ ip, \quad Mcla_2 = antigüedad$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	ec	44	a8
1	65	bd	d8
2	ed	ed	00
3	b6	86	30
4	de	76	a8

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	4f	a7	e8
1	f9	11	e8
2	13	13	00
3	7f	af	d0
4	59	f1	a8

Tabla 4.16: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 16 de M_{cla}

PAREJA 17:

$k = abcdefgh, \quad Mcla_1 = ANTIVIRUS, \quad Mcla_2 = politicas$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}	byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	45	cc	89	**0**	a0	2f	8f
1	bc	b5	09	**1**	10	09	19
2	ec	2d	c1	**2**	12	53	41
3	87	86	01	**3**	ac	af	03
4	ff	ee	11	**4**	7a	49	33

Tabla 4.17: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 17 de M_{cla}

PAREJA 18:

$k = abcdefgh, Mcla_1 = cryptography, \qquad Mcla_2 = computer$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}	byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	54	54	00	**0**	d7	d7	00
1	5d	b5	e8	**1**	f1	09	f8
2	85	25	a0	**2**	fb	5b	a0
3	4e	4e	00	**3**	e7	e7	00
4	ee	e6	08	**4**	49	41	08

Tabla 4.18: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 18 de M_{cla}

PAREJA 19:

$k = abcdefgh, \quad Mcla_1 = malware1.2, \quad Mcla_2 = proyecto$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}	byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	24	cc	e8	**0**	87	2f	a8
1	c5	5d	98	**1**	19	f1	e8
2	2d	35	18	**2**	53	ab	f8
3	76	06	70	**3**	bf	2f	90
4	46	66	20	**4**	a1	c1	60

Tabla 4.19: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 19 de M_{cla}

PAREJA 20:

$k = abcdefgh, \quad Mcla_1 = correcciones, \quad Mcla_2 = diferencias$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	54	6c	38
1	b5	85	30
2	dd	7d	a0
3	5e	e6	b8
4	66	de	b8

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	d7	cf	18
1	09	d9	d0
2	03	e3	e0
3	d7	4f	98
4	c1	59	98

Tabla 4.20: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 20 de M_{cla}

PAREJA 21:

$k = abcdefgh, \quad Mcla_1 = resguardar, \quad Mcla_2 = politicas$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	dc	cc	10
1	e5	b5	50
2	d5	2d	f8
3	f6	86	70
4	e6	ee	08

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	5f	2f	70
1	79	9	70
2	b	53	58
3	3f	af	90
4	41	49	08

Tabla 4.21: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 21 de M_{cla}

PAREJA 22:

$k = abcdefgh, \quad Mcla_1 = transferir, \quad Mcla_2 = introduccion$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	ec	04	e8
1	5d	bd	e0
2	45	ed	a8
3	be	5e	e0
4	d6	36	e0

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	4f	67	28
1	f1	11	e0
2	bb	13	a8
3	77	d7	a0
4	51	b1	e0

Tabla 4.22: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 22 de M_{cla}

PAREJA 23:

$k = abcdefgh, \quad Mcla_1 = caracteristica, \quad Mcla_2 = red\ feistel$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	54	dc	88
1	c5	e5	20
2	dd	6d	b0
3	c6	cc	0a
4	56	7e	28

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	d7	5f	88
1	19	79	60
2	3	93	90
3	6f	61	0e
4	d1	f9	28

Tabla 4.23: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 23 de M_{cla}

PAREJA 24:

$k = abcdefgh, \quad Mcla_1 = propuesta, \quad Mcla_2 = confusion$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	cc	54	98
1	5d	b5	e8
2	35	3d	8
3	4e	fe	b0
4	e6	e6	00

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	2f	d7	f8
1	f1	09	f8
2	ab	a3	8
3	e7	37	d0
4	41	41	00

Tabla 4.24: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 24 de M_{cla}

PAREJA 25:

$k = abcdefgh, \quad Mcla_1 = texto\ claro, \quad Mcla_2 = difusion$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	ec	6c	80
1	e5	85	60
2	8d	7d	f0
3	6e	66	08
4	36	d6	e0

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	4f	cf	80
1	79	d9	a0
2	f3	e3	10
3	c7	cf	08
4	b1	51	e0

Tabla 4.25: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 25 de M_{cla}

PAREJA 26:

$k = abcdefgh, \quad Mcla_1 = transformacion, \quad Mcla_2 = cajas - s$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}	byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	ec	54	b8	**0**	4f	d7	98
1	5d	c5	98	**1**	f1	19	e8
2	45	1d	58	**2**	bb	43	f8
3	be	c6	78	**3**	77	6f	18
4	d6	d6	00	**4**	51	51	00

Tabla 4.26: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 26 de M_{cla}

PAREJA 27:

$k = abcdefgh, Mcla_1 = bloques \quad , Mcla_2 = flujos$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}	byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	5c	7c	20	**0**	df	ff	20
1	ad	ad	00	**1**	01	01	00
2	35	e5	d0	**2**	ab	1b	b0
3	46	9e	d8	**3**	ef	97	78
4	e6	36	d0	**4**	41	b1	f0

Tabla 4.27: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 27 de M_{cla}

PAREJA 28:

$k = abcdefgh, \quad Mcla_1 = funcionamiento, \quad Mcla_2 = reordenamiento$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}	byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	7c	dc	a0	**0**	ff	5f	98
1	65	e5	80	**1**	f9	79	80
2	3d	35	08	**2**	a3	ab	08
3	d6	5e	88	**3**	5f	d7	88
4	06	6e	68	**4**	61	c9	a8

Tabla 4.28: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 28 de M_{cla}

PAREJA 29:

$k = abcdefgh, \quad Mcla_1 = linealidad, \quad Mcla_2 = distribucion$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	2c	6c	40
1	85	85	00
2	3d	d5	e8
3	e6	6e	88
4	46	de	98

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	8f	cf	40
1	d9	d9	00
2	a3	0b	a8
3	4f	c7	88
4	a1	59	f8

Tabla 4.29: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 29 de M_{cla}

PAREJA 30:

$k = abcdefgh, \quad Mcla_1 = busqueda, \quad Mcla_2 = fuerza\ bruta$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	5c	7c	20
1	65	65	00
2	d5	65	b0
3	46	5e	18
4	e6	9e	78

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	df	ff	20
1	f9	f9	00
2	0b	9d	96
3	ef	d7	38
4	41	19	58

Tabla 4.30: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 30 de M_{cla}

PAREJA 31:

$k = abcdefgh, \quad Mcla_1 = 123456789, \quad Mcla_2 = 1a2b3c4d5e$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	c6	c6	00
1	5f	c5	9a
2	d7	df	08
3	6c	de	b2
4	e4	d4	30

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	21	21	00
1	f3	19	ea
2	5	1d	18
3	c1	57	96
4	47	57	10

Tabla 4.31: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 31 de M_{cla}

PAREJA 32:

$k = abcdefgh, \quad Mcla_1 = combinaciones, \quad Mcla_2 = procedimientos$

byte	$xor1_{m1}$	$xor1_{m2}$	dif_{in}
0	54	cc	98
1	b5	5d	60
2	25	35	10
3	de	d6	08
4	06	66	60

byte	$xor8_{m1}$	$xor8_{m2}$	dif_{out}
0	d7	2f	f8
1	09	f1	f8
2	5b	ab	f0
3	57	5f	08
4	61	c1	a0

Tabla 4.32: Valores correspondientes a las dif_{in} y dif_{out} de la pareja 32 de M_{cla}

Una vez que se cuenta con las diferencias de entrada y salida para 32 parejas de texto elegidas aleatoriamente, se analizan las ocurrencias que hay en las diferencias (véase tablas 4.33 – 3.37), con la finalidad de realizar un cálculo de posibles valores de la clave.

BYTE 0				
parejas	**ocurrencias**		**parejas**	**ocurrencias**
28,68	1		20,60	1
82,86	1		a8,e8	1
10,70	**3**		89,8f	1
80,80	2		e8,a8	1
e2,26	1		38,18	1
00,00	**4**		e8,28	1
88,88	2		98,f8	2
08,08	1		b8,98	1
68,a8	1		20,20	2
49,4f	1		a0,98	1
11,77	1		40,40	1
c3,41	1			

Tabla 4.33 Ocurrencia de parejas de dif_{in} y dif_{out} para el byte 0

BYTE 1				
parejas	**ocurrencias**		**parejas**	**ocurrencias**
38,c8	1		00,00	**4**
b2,b2	1		d8,e8	1
30,30	1		09,19	1
b8,88	1		e8,f8	**2**
a2,a2	1		98,e8	**2**
28,d8	1		30,d0	1
40,40	1		50,70	1
a8,d8	1		e0,e0	1
f0,f0	**2**		20,60	1
58,68	1		60,a0	1
69,19	1		80,80	1
aa,da	1		9a,ea	1
ea,fa	1		60,f8	1

Tabla 4.34 Ocurrencia de parejas de dif_{in} y dif_{out} para el byte 1

BYTE 2				
parejas	**ocurrencias**		**parejas**	**ocurrencias**
38,58	2		c1,41	1
0a,06	1		a0,a0	1
40,c0	1		18,f8	1
70,90	2		a0,e0	1
ea,66	1		f8,58	1
b0,90	2		a8,a8	1
90,b0	1		f0,10	1
e0,60	1		58,f8	1
08,08	**3**		d0,b0	1
00,00	2		e8,a8	1
a1,e1	1		b0,96	1
93,ff	1		08,18	1
70,10	1		10,f0	1

Tabla 4.35 ocurrencia de parejas de dif_{in} y dif_{out} para el byte 2

BYTE 3				
parejas	**ocurrencias**		**parejas**	**ocurrencias**
48,c8	1		68,28	1
ea,6e	1		01,03	1
08,08	**3**		00,00	1
28,28	1		70,90	2
20,20	1		b8,98	1
20,60	1		e0,a0	1
30,d0	2		0a,0e	1
c0,40	1		d0,b0	1
f8,98	1		78,18	1
e8,68	1		d8,78	1
60,e0	1		88,88	2
b9,9b	1		18,38	1
1a,3e	2		b2,96	1

Tabla 4.36 Ocurrencia de parejas de dif_{in} y dif_{out} para el byte 3

BYTE 4				
parejas	**ocurrencias**		**parejas**	**ocurrencias**
d8,38	1		b0,90	1
a2,e6	2		a8,a8	1
08,08	**3**		11,33	1
30,10	**3**		b8,98	1
a8,e8	1		28,28	1
20,60	2		d0,f0	1
0a,0e	1		68,a8	1
18,38	1		98,f8	1
00,00	**4**		78,58	1
51,73	1		60,a0	1
e0,e0	**3**			

Tabla 4.37 Ocurrencia de parejas de dif_{in} y dif_{out} para el byte 4

Como se muestra en las tablas (véase tablas 4.33–4.37) hay ciertas parejas que tienen mayor ocurrencia (marcadas en negritas), dichas parejas son tomadas para inducir los posibles valores de la clave ya que presentan un patrón repetitivo; para el caso del byte 1 y 4 se muestran varias parejas (véase tablas 4.34, 4.37), esto es, porque en el caso de las parejas (00,00) los posibles valores son los 256. Por lo tanto, no se tendría una reducción de valores lo cual no sería útil, de tal forma se eligen las parejas que tienen mayor ocurrencia y se calculan los posibles valores de la clave para los cuales posteriormente se realiza un filtro, es decir, se extraen los valores que concuerden para cada una de dichas parejas.

El proceso para encontrar las posibles parejas consiste en buscar los valores tales que su diferencia sea la dif_{in} , posteriormente de dichos valores se buscan las parejas que den la dif_{out} , y se aplicará la búsqueda exhaustiva pero con una reducción del espacio de posibles claves.

En la tabla 4.38 se muestran las parejas asignadas a cada byte, las cuales fueron seleccionadas por la ocurrencia obtenida Se procede a calcular los posibles valores que pertenezcan a la clave, dichos valores son calculados por medio de un programa realizado en Dev C++ con la finalidad de agilizar la búsqueda, los valores obtenidos se muestran en el apéndice 2 pruebas de criptoanálisis.

BYTE	**PAREJAS**		
0	10,70		
1	f0,f0	e8,f8	98,e8
2	08,08		
3	08,08		
4	08,08	30,10	e0,e0

Tabla 4.38 Parejas utilizadas en cada byte para calcular posibles valores de clave

4.3 Resultado de pruebas

En la tabla 4.39 se muestra el número total de posibles valores que corresponde a cada uno de los bytes analizados.

BYTE	P1
0	142
1	60
2	240
3	240
4	115

Tabla 4.39 Cantidad de valores posibles de clave para cada byte

En la tabla 4.39 se observa que la cantidad de posibles valores varia, entre cada uno de los bytes aunque el análisis haya sido el mismo.

Analizando estos resultados, se tiene:

La muestra fue 32 parejas de texto, sin embargo no se logra obtener algún dato concreto que pueda indicarnos que se encontró la clave, por lo tanto se requiere de un mayor número de muestra de mensajes cifrados bajo la misma clave, con la finalidad de encontrar una pareja que tenga una mayor ocurrencia, es decir, que el cambio sea más marcado sobre las demás parejas.

Por otro lado supongamos que pretendemos determinar *k* bits de una cierta clave. Para ello, será preciso efectuar 2^k pruebas y llevar a cabo un registro de las salidas XOR. Este conteo arroja un número medio de pruebas el cual es obtenido mediante la fórmula:

$$\#\ \text{medio de prubas} = \frac{\mathrm{m} \cdot \alpha \cdot \beta}{2^{\mathrm{k}}}$$

De donde:

Para encontrar un byte de la clave se requieren $2^k = 2^8$ combinaciones posibles.
$m = \#$ parejas analizadas

$\alpha = \#$ medio de diferencias por parejas analizadas

$\beta =$ # de parejas ocurrente entre total de parejas analizadas

A continuación, para calcular el número medio de pruebas, se hará uso de las ocurrencias obtenidas anteriormente de las diferencias de entrada y salidas. Se tiene entonces que el número de ocurrencias son 4, 3 y 2 (véase tablas 4.33-4.37).

CÁLCULO PARA 4 OCURRENCIAS:

$m = 32\ parejas\ analizadas$

$\alpha = \frac{4}{32} = 0.125 \quad , \quad \beta = \frac{4}{32} = 0.125$

$\#\ medio\ de\ pruebas = \frac{32(0.125)(0.125)}{256} = \frac{0.5}{256} = 0.001953125$

CÁLCULO PARA 3 OCURRENCIAS:

$m = 32\ parejas\ analizadas$

$\alpha = \frac{3}{32} = 0.09375 \quad , \quad \beta = \frac{3}{32} = 0.09375$

$\#\ medio\ de\ pruebas = \frac{32(0.09375)(0.09375)}{256} = \frac{0.28125}{256} = 0.0010986$

CÁLCULO PARA 2 OCURRENCIAS:

$m = 32\ parejas\ analizadas$

$\alpha = \frac{2}{32} = 0.0625 \quad , \quad \beta = \frac{2}{32} = 0.0625$

$\#\ medio\ de\ pruebas = \frac{32(0.0625)(0.0625)}{256} = \frac{0.125}{256} = 0.00048828125$

Como se muestra, se tiene el cálculo para 3 valores distintos de ocurrencias, para ver un mayor cambio se procede a tomar como base que para 32 parejas de texto analizado se tiene que el mayor número de ocurrencias es 4. Por lo tanto para 256 parejas de texto por medio de interpolación se tendrían 32 ocurrencias:

CÁLCULO PARA 32 OCURRENCIAS:

$m = 256\ parejas\ analizadas$

$$\alpha = \frac{32}{256} = 0.125 \quad , \quad \beta = \frac{32}{256} = 0.125$$

$$\#\ medio\ de\ pruebas = \frac{256(0.125)(0.125)}{256} = \frac{4}{256} = 0.015625$$

En la tabla 4.40 se muestra el porcentaje que arroja cada uno de los cálculos de las diferentes ocurrencias que se presentan, se observa que a mayor número de ocurrencias tenemos una mayor probabilidad de encontrar el byte que pertenezca a la clave, sin embargo eso implica que se requiera de un mayor número de parejas de textos, que hacen al proceso más lento.

# ocurrencias	porcentaje
2	0.048828125 %
3	0.10986 %
4	0.1953125 %
32	1.5625 %

Tabla 4.40 Porcentaje obtenido a partir del número de ocurrencias

Ahora, se calcula el número de posibles mensajes que se requieren para encontrar un byte de la clave:

Para ello se hace uso de la probabilidad, ya que se necesita tener al menos el 50% de seguridad de que al realizar los cálculos encontraremos un byte de la clave, corriendo así

mismo el riesgo de tener la misma posibilidad de no encontrarlo, ya que no sería alentador tener una probabilidad de éxito de un 20% contra un 80%.

Dentro de la probabilidad se tiene la siguiente fórmula:

$$q = 1 - p$$

$p =$ es la probabilidad de éxito de un evento

$q =$ la probabilidad de fracaso del evento

Entonces si deseamos conocer la probabilidad de encontrar al menos un byte de la clave se puede plantear la siguiente ecuación, la cual es igualada a 0.5 ya que en la probabilidad el fracaso es representado con 0 y el éxito con 1, y al esperar l menos el 50% de éxito el valor que corresponde es 0.5.

$$0.5 = \frac{m}{2^k}$$

m = valor de posibles mensajes

Para encontrar un byte de la clave se requieren $2^k = 2^8$ combinaciones posibles

Despejando la variable m se tiene $\rightarrow \quad 2^k \cdot 0.5 = m$

$$m = 2^8 \cdot 0.5 \qquad \rightarrow \quad m = 128$$

Por lo tanto, existe la probabilidad de que con 128 parejas de texto se encuentre un byte de la clave. Ahora si el mismo cálculo se realiza para los 256 bits tenemos que:

$$m = 2^{256} \cdot 0.5 \qquad \rightarrow \quad m = 5.78960446186580977117854925043 44e^{76}$$

Como se observa el valor de posibles mensajes incrementa demasiado, lo cual requiere de un mayor tiempo, para poder deducir los valores correctos.

Con base a estos resultados obtenidos, durante el criptoanálisis diferencial, se puede concluir que el algoritmo no es fácilmente vulnerable, ya que se requiere de un número muy grande de posibles parejas de texto.

Por otro lado, tomando en cuenta que para 256 existen $5.7689e^{76}$ posibles combinaciones de clave, y tomando en cuenta que el cálculo solo es considerado para una ronda, al considerar las 32 rondas se eleva el número de combinaciones. Sin embargo, si las cajas fueran privadas o dependientes de la clave, se tendrían que considerar más bytes extras, es decir, por cada caja se tienen 256 bytes; entonces al ser 4 se tiene 1024 bytes extras. Entonces se tiene un total de 1056 bytes que el atacante debe obtener.

TIEMPO EN PROCESADORES

Se realizo también la prueba de tiempo que tarda en ejecutase el algoritmo en diferentes equipos con características diferentes, (véase tabla 4.41).

CARACTERISTICAS	EQUIPO 1	EQUIPO 2	EQUIPO 3	EQUIPO 4	EQUIPO 5
PROCESADOR	Intel Core i7 1.60GHz	Genuine Intel T2300, 1.66GHz	Intel Core 2 duo, 2,00 GHz	Intel ATOM N270 1.60 GHz	AMD Athlon II Dual-Core 2.10GHz
RAM	4.00GB	1.00 GB	2.00 GB	1.00 GB	2.00GB
SISTEMA OPERATIVO	Windows 7 Home Basic 64 bits	XP Profesional	Windows 7 Ultímate 32 bits	Windows XP Home	Windows 7 Starter 32 bits

Tabla 4.41 Características de equipos

En la tabla 4.42 se muestra el tiempo que tarda en ejecutarse el programa en cada uno de los equipos, puede verse claramente que varían, sin embargo cabe mencionarse que también influye la velocidad con que el usuario introduce la información que se le pide.

EQUIPO	TIEMPO DE CIFRADO (diez milésimas de seg.)	TIEMPO DE DESCIFRADO (diez milésimas de seg.)
1	0.00005	$5.26315e^{-5}$
2	$3.22580e^{-5}$	$2.12765e^{-5}$
3	0.0001	0.0001
4	0.00003125	$1.07526e^{-5}$
5	$4.16666e^{-5}$	$4.16666e^{-5}$

Tabla 4.42 Tiempos en procesadores

CONCLUSIONES

CONCLUSIONES

Hoy en día la información viaja por diversos medios de comunicación, lo cual pone en riesgo su integridad y confidencialidad al poder ser interceptada por algún intruso, que haga mal uso de la misma. Por esa razón se planteó diseñar un algoritmo de cifrado con la finalidad de conocer el funcionamiento de uno de los métodos que ayuda a proteger la información.

Al iniciar la planeación del algoritmo se presentan varios problemas acerca de cómo se estructuraría, los cuales se mencionan a continuación:

- ¿Cuál será la longitud de la clave y como generar las subclaves?
- ¿Cuáles serán las operaciones que realizará?
- Como diseñar las cajas-S
- Definir el número de rondas que tendrá el algoritmo

Como se mostró a lo largo de los capítulos, a cada una de las preguntas aquí expuestas se les fue dando solución. Sin embargo, para poder resolverlas se presentaron varias dificultades ya que fue necesario realizar un estudio a fondo sobre las bases matemáticas y electrónicas en las que se fundamentan los algoritmos de cifrado. Como se menciona en el capítulo 2 en la tabla 2.1, haciendo una comparativa del algoritmo diseñado con los analizados, el algoritmo propuesto presenta las siguientes características: longitud de clave (mayor a DES y dentro de los valores de AES), se propone una función no lineal basada en grupos algebraicos XOR y suma modular combinadas, el efecto en cascada influye para establecer 32 rondas de cifrado.

Por otro lado, un aspecto importante es el tiempo que tomará en llevarse a cabo al ser implementado en software o hardware, es aquí donde entra la decisión de qué operaciones se realizan. Por ejemplo el uso de la XOR, es una operación sencilla pero que genera un gran cambio al combinar bytes, del mismo modo la implementación de la suma modular y la rotación, logran causar la difusión los bytes, sin necesidad de utilizar grandes recursos de cómputo. Cabe mencionarse que podría haberse implementado una

operación de multiplicación, sin embargo este tipo de operaciones requieren de mayores recursos y tiempo de procesamiento lo cual haría lento el proceso de cifrado.

Durante el análisis de diseño del algoritmo la distribución equitativa de los bits siempre es fundamental para lograr la difusión. Por ejemplo, las cajas S construidas a partir de números pseudoaleatorios son una herramienta fácil de generar y permiten una distribución adecuada de los bits a lo largo de la red de cifrado hasta el criptograma. Al tomar la decisión de formar las cajas S, fue complicado ya que se cuenta con números aleatorios y pseudoaleatorios, de los cuales se tenía que elegir el mejor, y posteriormente estudiar los métodos que existen para generar dichos números y de esa forma seleccionar el más adecuado a las necesidades que se planteaban, es decir el uso de números pseudoaleatorios, ya que en el campo de la Criptografía brindan mayor seguridad.

Otro aspecto que ayuda a difuminar los bytes en cada una de las rondas es el efecto en cascada, con la rotación de bytes que se aplica al finalizar haciendo uso de los 4 bloques se logra el efecto en cascada, cabe mencionarse que el proceso de cambio en cada ronda no fue el esperado: lo deseable es que el cambio se notara en menos de 16 rondas y de esa forma el algoritmo tendría un menor número de rondas lo cual implica un menor tiempo de procesamiento. En el capítulo 4 pueden observarse los tiempos que se utilizan tanto para el proceso de cifrado como el descifrado en diferentes procesadores.

Una vez que se diseñó el algoritmo se realizaron las pruebas necesarias para conocer si es robusto. Se concluye que no es débil ante un criptoanálisis de tipo diferencial simple, ya que se necesita una enorme cantidad de parejas de texto para llevarlo a cabo. Debe recalcarse que el algoritmo diseñado puede ser débil ante un criptoanálisis lineal, o ante el nuevo criptoanálisis por bycliques que se ha probado contra AES, o por una combinación del criptoanálisis lineal y diferencial o diferencial modificado. Es importante tomar en cuenta que para que un algoritmo sea considerado realmente robusto debe ser probado por todos los tipos de criptoanálisis más usados

contra los algoritmos, de esta forma se tendrá mayor seguridad al hacer uso de dicho algoritmo.

El algoritmo puede sufrir modificaciones que robustezcan su seguridad, entre las cuales pueden ser:

- Declarar las cajas como privadas o dependientes de la clave, pues el atacante requerirá criptoanalizar 1056 bytes.
- Brindar al usuario la oportunidad de generar sus propias cajas-S por medio de números pseudoaleatorios de su elección

Ante estas consideraciones cabe mencionar la siguiente desventaja:

- Al ser privadas, el problema que se presenta es el medio por el cual serán enviadas al receptor, sin que algún intruso pueda acceder a ellas.
- Al diseñar el usuario sus cajas, podría presentarse el problema de que les tomaría mayor tiempo para realizar el cifrado, al elegir números primos grandes que no repitan valores en un ciclo corto.

Es complicado analizar todas las cuestiones que ayuden a robustecer el algoritmo. Como se menciona existen distintos tipos de criptoanálisis, así que deben analizarse cuáles serían los puntos débiles del algoritmo, ante los diversos tipos de ataque. Sin embargo, una vez diseñado y probado puede observarse en dónde podrían hacerse mejoras dentro del algoritmo, así mismo la metodología para llevar a cabo el diseño de un algoritmo ya sería más sencilla, porque ya se conocen los puntos del algoritmo que pueden fortalecerse.

Al iniciar el proyecto se plantearon objetivos, los cuales considero fueron cumplidos, ya que por un lado se tiene el algoritmo de cifrado de clave privada funcionando ; y por otra parte, estudiar el criptoanálisis diferencial para probar la fortaleza del algoritmo ha complementado satisfactoriamente el diseño del algoritmo.

Por otro lado, desde una perspectiva personal, considero fundamental mantener informados y educados a los usuarios, sobre la importancia de la seguridad de la

información, puesto que la mayoría de los usuarios consideran que su información es valiosa y privada pero no están bien informados que existen políticas, metodologías y herramientas que les ayudan a protegerla, o si saben de éstas no conocen su funcionamiento.

Finalmente, el desarrollo del presente trabajo, brindó la oportunidad de aplicar los conocimientos que se obtuvieron durante la educación académica, y a partir de ellos tener la capacidad de diseñar un algoritmo de cifrado desde cero, y no sólo conformarse con llevar a cabo la implementación de alguno existente. Por lo tanto se da por cumplido el objetivo central que llevó a realizar este proyecto de trabajo.

APÉNDICES

- **Efecto en Cascada**
- **Cajas S**
- **Pruebas de Criptoanálisis**

APÉNDICE A

Efecto en Cascada

En este apartado se muestra el efecto en cascada, al cual se hace referencia en el capítulo 3, donde se observa que los bytes de modifican completamente en la ronda 16. Por tal motivo se consideran 32 rondas con la finalidad de que los bytes estén completamente modificados.

La columna uno hace referencia a los mensajes en claro, los cuales son cifrados bajo la misma clave. Los mensajes y claves utilizadas para el ejemplo son:

$Mcla_1$= hola

$Mcla_2$= Hola

K=adiós

La segunda columna indica el número de ronda, y la tercera columna indica el mensaje cifrado, donde se va mostrando en color rojo los bytes que van modificándose en cada una de las rondas tomando como base el cifrado del $Mcla_1$= hola y $Mcla_2$= Hola.

EFECTO EN CASCADA

Mcla	RONDA	CIFRADO
hola	**1**	90-70-d2-26-dc-e6-ab-0e-d9-e6-ab-0e-d9-f5-95-9c-04-b7-ab-a0-89-7c-53-89-ca-34-c2-ca-7f-ad-8e-f1
Hola	**1**	90-70-d2-26-dc-e6-ab-0e-d9-e6-ab-0e-d9-**8f**-95-9c-04-b7-ab-a0-89-**c5**-53-89-ca-34-c2-ca-7f-**3c**-8e-f1
hola	**2**	5e-00-2a-93-a4-5f-12-b3-2b-17-f3-9d-00-a9-a6-81-95-3a-0d-0a-a3-03-5c-e4-81-3a-db-68-41-85-6c-38
Hola	**2**	5e-00-**2d**-93-a4-5f-12-b3-2b-17-f3-9d-00-**ad**-a6-81-95-3a-**8f**-0a-a3-**84**-5c-e4-81-3a-**d6**-68-41-**7d**-6c-38
hola	**3**	8c-16-ec-6f-d5-52-99-02-1c-ca-4d-9e-7d-02-2b-a5-05-55-23-38-2a-62-30-9b-47-3d-bf-cc-f3-97-6e-cc
Hola	**3**	8c-16-**79**-6f-d5-52-99-**a7**-1c-ca-4d-9e-7d-**d9**-2b-**a3**-05-55-**18**-38-2a-**f6**-30-9b-47-3d-**60**-cc-f3-**45**-6e-**c1**
hola	**4**	ca-3c-16-29-19-87-fe-7c-ee-8e-49-03-6c-4b-a3-cd-a6-c1-fe-16-88-b8-d7-32-40-e8-3b-8d-58-ed-7b-85
Hola	**4**	ca-3c-**49**-29-19-87-fe-**f5**-ee-8e-49-03-**2c-22**-a3-**42**-a6-c1-**a8**-16-**74-28**-d7-32-40-e8-**84**-8d-**d4-3f**-7b-**bc**
hola	**5**	04-a1-2b-a2-6f-cc-be-3c-09-a8-92-ca- e1- c1-6d-2b-90-5f-be-d3-98-cd- c9-37-9f-f4-22-16-3f-b5-8d-a7
Hola	**5**	04-**55-10**-a2-6f-cc-be-**73**-09-a8-92-ca-**1c-64**-6d-**27**-90-**81-df**-d3-**d0-28**-c9-37-9f-**a7-89**-16-**7f-29**-8d-**e0**
hola	**6**	af-5e-52-e4-fe-34-bf-de-44- cd-08-6a- fa - 31-c5-73-21-c9-39-98-26 -fc- 40-de-1d-bc-60-af-50-d5-40-d0
Hola	**6**	af-**90-21**-e4-fe-34-**f4-14**-44-cd-08-6a-**b6-d9-79-9a**-21-**97-44**-98-**c7-20**-40-de-1d-**cc-ae**-af-**47-45-90-b0**
hola	**7**	fb-09-35-07-ea-a9-cc-11-45-c6-8c-f7-69-b6-56-71-7f-b2-8e-17-b1-7d-40-4f-e6-26-67-44-d9-29-f0-30
Hola	**7**	fb-**cd-41**-07-ea-a9-**2b-9a**-45-c6-8c-**2e-1b-d7-a-b8**-7f-**8b-ac-d3-89-71**-40-4f-e6-**e9-a0-42-f3-fb-9-85**

Tabla A1: Efecto en casada ronda 1 a 7 rondas

EFECTO EN CASCADA

Mcla	RONDA	CIFRADO
hola	**8**	34-e7-35-06-02-ca-7d-57-97-f2-e6-b7-af -50-af- a6- 4c- 61- f1-3c -f4- b2-71-3c-86-dd-2b-d3-34-60-77-91
Hola	**8**	**8f-6d-10-0**6-02-ca-**9e-2a**-97-f2-e6-**10-e6-e7-ee-35-c3-95-49-3f-21-86**-71-3c-**fe-14-ba-6f- 57-32-04-6b**
hola	**9**	21-95-04-c4-14-1d-b3-39-16-07-c7-ed-62-92-45-8d-e4-d7-a1-2f- c9- f3-a3-c8-ea-94-88-95-7c-43-b2-41
Hola	**9**	**17-10-50**-c4-14-**3b-f9-b1**-16-07-c7-**7f-54-68-8d-51-39-ec-67-c1-d8-ab**-a3-c8-eb-**24-f8-cb-94-c3-2c-c2**
hola	**10**	73-60-de-0a-9d-a5-b7-db-03-8a-31-c2-77-e7-8d-09-31-35-11- 93-39- 6b-f9-85-0e-4a- 7c-dc-88- 07-3f-69
Hola	**10**	**3c-44-ae**-0a-9d-**ee-77-96**-03-8a-**c3-5c-13-1b-93-3e-d2-42-d3-58-33-a8**-f9-85-**37-7a-cc-ae-94-87-85-65**
hola	**11**	b7-68-92-e9-46-0c-e9-6a-a8-52-e3-bc-00-b2-bd -ff- 57-c6- de- e1-a7-1f- 87-27-fb-4e- 37-4f- d2-b9-63-10
Hola	**11**	**1a-80-c4**-e9-46-**6f-b9-23**-a8-52-**ce-e6-eb-bb-32-89-e2-47-ee-4f-9b-f0**-87-**b3-0e-26-d8-ea-92-56-92-17**
hola	**12**	56-c2-ea-f0-7f- c4- 5b- f0- 4b-16-cd-5f- 0c- c8- 6b-9c-d7-4c-e5- d4-59- f1- 63-d8-e1-09-d6-27-d8-ad-cb-bb
Hola	**12**	**02-ec-0e**-f0-**01-94-50-db**-4b-16-**41-9c-b5-4f-99-eb-0e-6c-19-89-a5-0a**-63-**9d-9c-88-d3-bb-ba-a1-2c-a0**
hola	**13**	90-0a-c0-29-bb-34-06-46-47-14-13-3b-47-d9-11-e1-6d-40- c4- fe- 00- 0d-14-db-6e-53- 07-a3-16-fe-90-8a
Hola	**13**	**c4-35-9b**-29-**62-02-80-44**-47-0**3-c8-ce-42-cf-88-59-b5-66-e7-5d-b9-97**-14-**44-c9-5d-98-5a-25-48-b0-ae**

Tabla A2: Efecto en cascada ronda 8 a 13 rondas

EFECTO EN CASCADA

Mcla	RONDA	CIFRADO
hola	**14**	86-c9-68-f6-75-7a-fb-2a-4a-ec-69-7c-8f-71-61-36-5f-c6-f9-80-49-2b-1b-6a-20-54-b5-95-1c- b7- ce-db
Hola	**14**	**0-3e-2b-**f6-**4c-77-4a-be**-4a-**48-f6-23-4e-90-87-46-78-27-d2-d1-d2-1-d0-a3-1f-f-b7-c1-39-ee-a4-06**
hola	**15**	08-68-05-b0-3f-cb-6b-6e-5e-c6-b8- ac- 1f- cd-c1-46-63-1a- b3- e4-54-eb-88-05-35- 6f- fe- b4-ce-f2-3b-47
Hola	**15**	**36-37-1d-95-bf-75-f6-4c**-5e-**58-62-50-99-c0-dc-52-56-a3-32-ae-ea-a3-82-29-49-42-7d-e1-3b-a3-c8-29**
hola	**16**	3b-13-c1-4c-6e-ef- 0e-9a-b0-73-9c-98-66-14-41-7f- 42-3b-c8-b2-83-77-a1- 85-69-9d-8b-75-b3-98-7c-76
Hola	**16**	**ef-49-60-96-98-b2-c3-df-67-81-b1-fe-d5-dd-30-00-aa-cf-6d-38-09-75-35-1c-ae-cc-91-99-68-ae-69-0b**

Tabla A3: Efecto en cascada ronda 14 a 16 rondas

APÉNDICE B

Cajas-S

En este apartado se muestran las cajas-S utilizadas para el cifrado, así mismo se muestra cada una con su inversa la cual es usada en el proceso de descifrado.

El uso adecuado de las cajas es: el primer digito del byte hace referencia a la columna, mientras que el segundo indica la fila, siendo así la intersección de estos el nuevo valor. Ejemplo:

El usuario tiene el byte *3E*, el cual tiene que remplazarlo con el byte de la caja-S1, se tiene entonces que el 3 indica la columna y la E indica la fila, por tanto, el nuevo valor es *29*.

Para obtener el inverso es el mismo procedimiento, pero ahora usando la caja-S1 inversa, entonces, el valor es *29* de donde el 9 indica la fila y el 2 la columna, teniendo así de nuevo el valor *3E*.

CAJA S1

S1	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	90	F2	E3	93	31	F0	F8	E8	57	9A	5A	2F	88	E6	96	0
1	42	B6	A2	EE	54	C7	D5	85	4C	D8	E1	BA	6C	4A	45	4
2	9D	5D	1	8A	63	3A	67	D9	FF	38	2B	A6	39	EA	2D	A0
3	CA	17	A4	81	15	E5	82	10	9C	3B	2	73	87	6D	83	1B
4	D1	8	EC	B1	D4	84	59	1D	CC	78	89	41	B	76	B3	56
5	AA	C0	C9	A3	FE	CB	C3	4D	74	A9	5	36	D7	2C	97	EB
6	95	D0	CF	9B	F	E0	C6	33	D6	68	C1	62	6E	B0	D	99
7	E2	7B	26	27	11	F7	D2	A8	CE	B9	AE	C	A5	6	FD	34
8	20	C2	B4	DA	5E	86	9E	30	FC	80	DC	B2	70	3D	22	44
9	2A	B7	1C	71	C4	35	52	7D	46	50	8B	BF	32	DB	60	53
A	8E	51	21	FA	98	BE	19	8D	43	47	EF	F6	9F	66	94	D3
B	77	BD	C5	7C	E4	A1	65	37	DE	E7	9	49	4E	E9	75	4F
C	F4	79	1F	8F	CD	AC	61	40	3E	DD	3	C8	A7	23	AD	3C
D	F5	F9	DF	24	B8	69	5B	A	12	ED	64	8C	13	1E	48	6A
E	E	7	18	29	1A	BB	91	FB	F3	4B	14	16	6F	25	AF	AB
F	6B	58	55	B5	28	92	5C	F1	2E	3F	5F	72	BC	7A	7E	7F

Tabla B1: Caja S1

CAJA S1-INVRESA

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

Tabla B2: Caja S1 inversa

CAJA S2

S2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	40	A1	5C	F5	68	24	C	A2	B	25	83	6A	58	F	DB	CB
1	97	F6	59	ED	F7	DA	12	2C	31	8A	B1	3E	D5	AE	C0	27
2	45	F2	A6	E7	C2	6C	BB	63	6D	B3	D1	41	76	E	67	3B
3	13	CD	1A	1D	5F	C8	69	B5	EB	64	1B	54	16	2D	C3	1E
4	1F	7A	18	5D	89	AA	A4	FF	4F	77	70	7F	82	E9	C1	39
5	6	D8	E5	6F	8	B8	92	A3	C7	EC	BC	A8	85	84	9	38
6	B4	19	C4	FC	CE	AF	35	17	6B	C9	C6	D7	7E	E2	9B	8C
7	A7	33	F8	A	6E	CA	D3	4E	56	9A	50	98	F0	94	11	D9
8	79	F9	BF	AD	E0	62	88	44	3	D0	BE	81	E4	14	B9	E1
9	B7	DE	FB	8D	37	2	75	48	E3	EE	93	55	5	91	E8	65
A	7D	10	AB	0	42	29	CF	7B	A0	2F	2B	51	D2	7C	E6	DF
B	BA	72	1	80	B6	C5	FE	FA	26	F3	36	4	61	32	BD	EF
C	21	43	60	7	15	2A	4A	D	A5	CC	8B	78	23	F4	3C	28
D	8F	FD	EA	47	46	86	8E	22	4C	F1	DC	71	87	2E	B2	AC
E	74	34	99	B0	9E	90	D6	5A	A9	D4	DD	73	3A	5E	3F	57
F	9C	9F	49	66	9D	1C	95	4B	96	30	52	5B	3D	20	4D	53

Tabla B3: Caja S2

CAJA S2-INVERSA

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

Tabla B4: Caja S2 inversa

CAJA S3

S3	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	FD	43	85	2C	8	49	9D	BF	14	5C	9C	DA	36	EA	64	5F
1	4	A7	5E	12	BA	58	21	EB	8E	73	6D	DC	7B	CE	B8	39
2	B4	2	B6	A	9B	F3	68	5	6C	15	37	66	4F	57	E3	41
3	1F	86	69	B	2E	74	2D	BE	62	20	D9	E7	7E	78	28	AB
4	5B	81	19	38	60	C4	EE	91	77	2B	B0	50	3C	B1	C1	7C
5	AC	8A	80	4E	D1	F0	1A	C0	5A	BD	83	C7	31	4C	11	29
6	45	55	C6	9A	A4	B9	CC	F8	FC	6E	FA	8C	E8	A9	1E	25
7	D0	B5	E0	27	53	A3	FE	88	54	AA	40	AF	48	CF	B2	67
8	99	92	35	82	FF	AD	18	9F	A2	F7	A0	BB	5D	10	F2	89
9	56	BC	D6	ED	E2	59	52	F5	B3	3	D4	B7	AE	E6	EF	E9
A	33	1C	C2	16	C5	6B	D3	F6	1B	17	E5	C3	9E	4D	1D	6
B	63	D7	90	F9	2F	9	46	42	F1	DB	3E	4A	13	DD	FB	D2
C	97	98	A5	D	01	E1	CA	C8	76	3A	EC	07	3D	3B	0F	0E
D	34	C	75	C9	D8	32	22	7D	DE	79	26	4B	3F	00	65	30
E	DF	6F	2A	8F	7A	7F	23	6A	51	F4	CB	72	CD	47	70	24
F	44	61	71	84	87	8B	95	A8	D5	E4	A1	93	96	94	A6	8D

Tabla B5: Caja S3

CAJA S3-INVERSA

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

Tabla B6: Caja S3 inversa

CAJA S4

S4	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	9F	4E	79	20	FF	C4	8	DD	70	F0	9A	E3	29	FE	B7	6C
1	61	26	E5	5D	99	85	C0	63	BD	77	8E	AF	A9	41	10	5C
2	89	35	84	28	A4	F1	9B	15	11	2B	B1	46	9E	34	4F	DF
3	30	2D	9	36	23	B9	C8	7E	BF	4D	6D	D7	D5	E4	AE	43
4	64	24	F5	60	71	C7	55	BE	5F	EA	37	67	FD	FB	EF	BC
5	B2	FC	DC	16	4B	2C	F9	EC	90	D4	9D	42	4A	A	E9	87
6	88	8F	21	6A	62	0E	25	6F	14	2A	FA	1D	C	4C	59	B4
7	69	81	8D	49	7F	C5	80	7D	7A	1C	72	22	CD	E7	B	12
8	5	92	98	E6	73	4	65	D	27	B8	0F	B3	2F	9C	F8	93
9	F2	86	17	75	CC	D6	54	5E	8C	07	F7	83	A8	31	13	CA
A	6	AD	C1	3D	AA	3B	7B	82	95	2E	18	00	CE	50	F3	94
B	AB	57	D2	C9	7C	78	C2	F6	BB	F4	6B	01	2	DA	DE	74
C	A6	3	44	A2	3A	8A	E0	5B	19	33	68	40	3F	C3	DB	6E
D	45	EE	66	1B	D9	51	8B	EB	E2	56	1F	1A	D8	A5	52	5A
E	CF	E1	1E	ED	76	38	A7	A0	3E	CB	E8	D3	91	47	32	3C
F	B5	39	C6	D0	A1	D1	48	BA	53	58	97	96	A3	AC	B0	B6

Tabla B7: Caja S4

CAJA S4-INVERSA

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

Tabla B8: Caja S4 inversa

APÉNDICE C

Pruebas de Criptoanálisis

En este apartado se muestran los valores que se obtienen respecto a las parejas con mayor ocurrencia para cada uno de los bytes mencionados en el capítulo 4 (véase tablas I-IX), los cuales como se mencionó se obtienen de verificar que valores dan la diferencia de entrada, y posteriormente esos valores dan la diferencia de salida.

Se observan también (véase tablas X y XI), valores que se obtiene al realizar un filtro para los byte 1 y 4, que cuentan con 3 parejas de igual ocurrencia, dicho filtro se obtuvo al seleccionar los valores que coincidían en dichas parejas.

BYTE 0: valores de pareja (10,70)

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

Tabla C1:Byte 0 pareja 10,70

BYTE 1: valores de pareja (f0, f0)

0		5		A		F		84		89		8E		FE		7B		F9		76		F4		71
1		6		B		80		85		8A		8F		7D		FB		78		F6		73		F1
2		7		C		81		86		8B		7F		FD		7A		F8		75		F3		
3		8		D		82		87		8C		FF		7C		FA		77		F5		72		
4		9		E		83		88		8D		7E		FC		79		F7		74		F2		

Tabla C2: Byte 1 pareja f0,f0

8		C		88		8C		7		5		3		1		7A		**7E**		FA		FE		76		74		72
9		D		89		8D		87		85		83		81		7B		**7F**		FB		FF		F6		F4		F2
A		E		8A		8E		6		4		2		78		7C		**F8**		FC		77		75		73		71
B		F		8B		8F		86		84		82		79		7D		F9		FD		F7		F5		F3		F1

Tabla C3:Byte 1 pareja e8,f8

28		2E		3C		AA		B8		BE		26		A5		23		A2		8		E		1C		8A		98
29		2F		3D		AB		B9		BF		36		B5		33		B2		9		F		1D		8B		99
2A		38		3E		AC		BA		27		A6		24		A3		21		A		18		1E		8C		9A
2B		39		3F		AD		BB		37		B6		34		B3		31		B		19		1F		8D		9B
2C		3A		A8		AE		BC		A7		25		A4		22		A1		C		1A		88		8E		
2D		3B		A9		AF		BD		B7		35		B4		32		B1		D		1B		89		8F		

Tabla C4: Byte 1 pareja 98,e8

BYTE 2, 3, 4: valores de pareja (08, 08)

Los bytes 2,3 y 4 coinciden al tener la misma ocurrencia de parejas 08,08, por lo tanto se generan los mismos posibles valores. La tabla V es para los 3 bytes.

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

Tabla C5: Byte 2 pareja 8,8

BYTE 4: valores de pareja (30, 10)

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

Tabla C6: Byte 4 pareja 30,10

BYTE 4: valores de pareja (e0, e0)

0	A	14	1E	88	92	9C	7C	77	72	6D	68	63
1	B	15	1F	89	93	9D	FC	F7	F2	ED	E8	E3
2	C	16	80	8A	94	9E	7B	76	71	6C	67	62
3	D	17	81	8B	95	9F	FB	F6	F1	EC	E7	E2
4	E	18	82	8C	96	7F	7A	75	70	6B	66	61
5	F	19	83	8D	97	FF	FA	F5	F0	EB	E6	E1
6	10	1A	84	8E	98	7E	79	74	6F	6A	65	
7	11	1B	85	8F	99	FE	F9	F4	EF	EA	E5	
8	12	1C	86	90	9A	7D	78	73	6E	69	64	
9	13	1D	87	91	9B	FD	F8	F3	EE	E9	E4	

Tabla III Byte 4 pareja e0, e0

FILTRO DE VALORES BYTE 1

8		D		8A		8F		5		83		78		7D		FA		FF		75		F3
9		E		8B		7		85		2		79		7E		FB		77		F5		72
A		F		8C		87		4		82		7A		7F		FC		F7		74		F2
B		88		8D		6		84		1		7B		F8		FD		76		F4		71
C		89		8E		86		3		81		7C		F9		FE		F6		73		F1

Tabla C8: Valores después de filtro de byte 1

FILTRO DE VALORES BYTE 4

1		B		14		1E		89		92		9C		FD		77		72		EE		69		E4
2		C		15		1F		8A		93		9D		7C		F7		F2		6D		E9		63
3		D		16		81		8B		94		9E		FC		76		71		ED		67		E3
4		E		17		82		8C		95		9F		7B		F6		F1		6C		E7		62
5		F		19		83		8D		96		7F		FB		F5		70		EC		66		E2
6		10		1A		84		8E		97		FF		7A		74		F0		6B		E6		61
7		11		1B		85		8F		99		7E		FA		F4		6F		EB		65		E1
9		12		1C		86		90		9A		FE		79		73		EF		6A		E5		
A		13		1D		87		91		9B		7D		F9		F3		6E		EA		64		

Tabla C9: Valores después de filtro de byte 4

GLOSARIO

-A-

AES: Advanced Encryption Standard también conocido como Rijndael, es un esquema de cifrado por bloques, desarrollado por dos belgas, Joan Daemen y Vincent Rijmen

Amenaza: Es un evento o acción latente que puede producir un daño sobre los elementos de un sistema. En el caso de la Seguridad de la Información, los elementos son la información.

Ataque: Es la puesta en acción de una amenaza.

Ataque fuerza bruta: Una técnica de ataque que trata de recuperar una clave probando todas las combinaciones posibles hasta encontrar aquélla que permite el acceso.

-B-

BBS: Blum Blum Shub es un generador de números pseudoaleatorio propuesto por Lenore Blum, Manuel Blum y Michael Shub en 1986.

Bit: Unidad mínima de almacenamiento de información cuyo valor puede ser representada con los símbolos 0 ó 1, o bien verdadero o falso.

Byte: Conjunto de 8 bits el cual suele representar un valor asignado a un carácter.

-C-

C/C++: C es un lenguaje de programación creado en 1969, por Ken Thompson y Dennis M. Ritchie en los laboratorios de Bell, como evolución del anterior lenguaje B. Está orientado a la implementación de sistemas operativos; ya que, dispone de estructuras de alto nivel pero, a su vez, de instrucciones de bajo nivel. C++ es diseñado a mediados de los años 80, por Bjame Stroustrup, como extensión del lenguaje C. se le

considera un lenguaje hibrido; abarca tres paradigmas de la programación: estructurada, genérica y orientada a objetos.

Cifrado: Es un proceso que permite ocultar un mensaje o de un archivo mediante la mezcla de un mensaje entendible con un elemento secreto llamado clave.

Codificar: es la acción de transformar un contenido a un código.

Criptoanálisis: Conjunto de procedimientos, técnicas y programas para romper los mecanismos de cifrado.

-D-

Decodificar: Es el proceso por el cual se convierten símbolos en información entendible por el receptor.

DES: (Data Encryption Standard) es un algoritmo de cifrado simétrico escogido como un estándar en los Estados Unidos en 1976,

Descifrar: Proceso por el cual se transforma un mensaje cifrado en su original de texto simple.

DSA: (Digital Signature Algorithm, en español Algoritmo de Firma digital) es un estándar del Gobierno Federal de los Estados Unidos de América o FIPS para firmas digitales.

-F-

FIPS: Federal Information Processing Standards (en español Estándares Federales de Procesamiento de la Información) son estándares desarrollados por el gobierno de los Estados Unidos para la utilización por parte de todas las agencias del gobierno no militares y por los contratistas del gobierno.

-I-

Información: Es un conjunto organizado de datos, que constituye un mensaje sobre un cierto fenómeno o conocimiento.

Intruso: Se aplica a la persona que se ha introducido en una propiedad, lugar, asunto o actividad sin derecho o autorización.

-L-

Llave: En cifrado y firmas digitales, es un valor utilizado en combinación con un algoritmo para cifrar o descifrar información, también llamado clave.

Llave pública: Es el valor de la clave o llave que el usuario da a conocer al público y cualquiera puede utilizarla para cifrar o verificar un mensaje.

Llave privada: El usuario mantiene la llave privada secreta y la utiliza para firmar y descifrar los mensajes.

-M-

Mensaje: Es la información que el emisor envía al receptor a través de un canal determinado o medio de comunicación (como el habla, la escritura, etc.)

-N-

NITS: Instituto Nacional de Normas y Tecnología (en inglés, *National Institute of Standards and Technology*) es una agencia de la Administración de Tecnología del Departamento de Comercio de los Estados Unidos. La misión de este instituto es promover la innovación y la competencia industrial en Estados Unidos mediante avances en metrología, normas y tecnología de forma que mejoren la estabilidad económica y la calidad de vida.

-S-

Seguridad de la información: Son todas aquellas medidas preventivas y reactivas del hombre, de las organizaciones y de los sistemas tecnológicos que permitan resguardar y proteger la información buscando mantener la confidencialidad, la disponibilidad e Integridad de la misma.

Seguridad informática: Es el área de la informática que se enfoca en la protección de la infraestructura computacional y todo lo relacionado con ésta (incluyendo la información contenida). Para ello existen una serie de estándares, protocolos, métodos, reglas, herramientas y leyes concebidas para minimizar los posibles riesgos a la infraestructura o a la información.

-V-

Vulnerabilidad: Es una falla o debilidad en el diseño, implementación u operación de un sistema que puede llevar a que sea explotado para violar las políticas de seguridad por parte de un intruso.

REFERENCIAS

BIBLIOGRAFÍA

- Amparo, F. Técnicas Criptografícas. México: Alfaomega.
- Antón, H. (2003). Introducción al álgebra lineal. México: Limusa S.A. de C.V.
- Ariel, M. (2009). Criptografía: técnicas de desarrollo para profesionales.
- Eldon, W. (1971). Álgebra booleana y sus aplicaciones. México: México Continental.
- Jaquelina, L. (2009). Criptografía. México: Universidad Nacional Autónoma de México, Facultad de Ingeniería.
- Spiegel, M. (1922). Teoría y problemas de probabilidad y estadística. México: McGraw-Hill.
- Víctor, M. (1997). Álgebra elemental Matemáticas activas y sencillas. JIT Press.

ENLACES WEB

- *Algebra universal para la Ciencia de la Computación: Aplicación a la Criptografía*

http://es.scribd.com/doc/19300362/49/Cifrado-por-bloque-y-%EF%AC%82ujo (visitada 16 de Mayo 2012)

- *Aritmética modular*

http://personal.telefonica.terra.es/web/jms32/Cifra/CodSecretos/Cap06/Cap0606.html#LaAritmeticaModular (visitada 16 de Mayo 2012)

- *Biclique Cryptanalysis of the Full AES*

http://research.microsoft.com/en-us/projects/cryptanalysis/aesbc.pdf (visitada 28 de Febrero 2012)

- *Cifrado por bloques*

http://www.alegsa.com.ar/Dic/cifrado%20por%20bloques.php (visitada 20 de Febrero 2012)

- *Clasificación de los criptosistemas*

http://mmc.geofisica.unam.mx/LuCAS/Manuales-LuCAS/doc-unixsec/unixsec-html/node307.html (visitada 28 de Febrero 2012)

- *Criptografía*

http://computacion.cs.cinvestav.mx/~jjangel/chiapas/criptografia.pdf (visitada 16 de Mayo 2012)

- *Criptografía clásica*

http://www.hezkuntza.ejgv.euskadi.net/r43573/es/contenidos/informacion/dia6_sigma/es_sigma/adjuntos/sigma_24/9_Criptografia_clasica.pdf (visitada 30 de Diciembre 2011)

- *Criptografía de clave privada*

http://profesores.sanvalero.net/~w0505/Seguridad/ssi_v110___doc2.pdf (visitada 30 de Diciembre 2011)

- *Criptografía moderna*

http://alarcos.inf-cr.uclm.es/doc/psi/tema4.pdf (visitada 16 de Mayo 2012)

- *Criptografía y Seguridad en Computadores*

http://es.scribd.com/doc/61286876/55/Generador-Blum-Blum-Shub (visitada 16 de Mayo 2012)

- *Criptología*

http://www.tierradelazaro.com/public/libros/cripto.pdf (visitada 29 de Noviembre 2011)

- *Criptosistema de Vernam*

http://cryptomex.org/CursoCriptoTec/vernam.html (visitada 29 de Enero 2012)

- *CrypTool: Practical Introduction to Cryptography and Cryptanalysis*

https://www.cryptool.org/images/ct1/presentations/CrypToolPresentation-en.pdf (visitada 29 de Enero 2012)

- *Epistemowikia: Criptografía*

http://campusvirtual.unex.es/cala/epistemowikia/index.php?title=Criptografia#Algoritmo_DES (visitada 20 de Enero 2012)

- *Estadística para todos: números aleatorios*

http://www.estadisticaparatodos.es/taller/aleatorios/aleatorios.html (visitada 29 de Enero 2012)

- *Introducción a la Criptografía*

http://web.usal.es/~hernando/segi2008/2IntroCripto.pdf (visitada 29 de Enero 2012)

- *Introducción al Criptoanálisis diferencial*

http://www.slideshare.net/darg0001/fundamentos-del-criptoanlisis-diferencial (visitada 16 de Mayo 2012)

- *Instituto para la Seguridad en Internet: El azar en la Criptografía*

http://www.instisec.com/publico/verarticulo.asp?id=26 (visitada 29 de Enero 2012)

- *Kriptópolis: AES, casi cuatro veces menos seguro que antes*

http://www.kriptopolis.org/node/8523 (visitada 16 de Mayo 2012)

- *Números aleatorios y pseudoaleatorios*

https://belenus.unirioja.es/~secarcam/criptografia/criptoanalisis/numeros_aleatorios.html (visitada 16 de Mayo 2012)

- *Números pseudoaleatorios*

http://es.scribd.com/doc/2557289/Numeros-Pseudoaleatorios (visitada 16 de Mayo 2012)

- *Seguridad de la información*

http://www.segu-info.com.ar/criptologia/simetricos.htm (visitada 16 de Febrero 2012)

- *Seguridad de la Información para empresas*

http://www.cibernarium.tamk.fi/seguridad_2/maaritelma_index_2.htm# (visitada 16 de Febrero 2012)

- *Seguridad en sistemas de información*

http://ccia.ei.uvigo.es/docencia/SSI/0910/apuntes/Tema2.parte1.pdf (visitada 16 de Febrero 2012)

- *Seguridad informática*

http://www.firmadigital.gob.mx/Seguridad.pdf (visitada 16 de Febrero 2012)

- *Synthesis of Cellular Automata with Aditive Code 683 in a Feistel Network*

http://www.deasis.es/articles/Synthesis%20of%20Cellular%20Automata%20with%20Aditive%20Code%20683%20in%20a%20Feistel%20Network.pdf (visitada 20 de Enero 2012)

- *UNAM Criptografía: Breve historia de la Criptografía*

http://unamcriptografia.wordpress.com/2011/10/06/breve-historia-de-la-criptografia/ (visitada 29 de Noviembre 2011)

- *Un sistema generador de números pseudoaleatorios*

http://virtual.chapingo.mx/fis/aleato.pdf (visitada 29 de Diciembre 2011)

- *Web pericial: Generación de Números Pseudoaleatorios usados por Sistemas Criptográficos*

http://www.periciascaligraficas.com/v2.0/resultados.php?contenidosID=111 (visitada 29 de Diciembre 2011)

- *WifiTeCH: Criptografía*

http://wifitech.wordpress.com/criptografia/ (visitada 16 de Mayo 2012)

- *WIKIPEDIA: tamaño de la clave*

http://en.wikipedia.org.es.mk.gd/wiki/Key_size (visitada 16 de Mayo 2012)