DISEÑO E IMPLEMENTACIÓN DE UN CIFRADO DES EXTENDIDO TRENZADO.

DIEGO ALEJANDRO TÉLLEZ BERNAL

FUNDACIÓN UNIVERSITARIA LOS LIBERTADORES
FACULTAD DE INGENIERÍA Y CIENCIAS BÁSICAS
INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C
2017

DISEÑO E IMPLEMENTACIÓN DE UN CIFRADO DES EXTENDIDO TRENZADO.

DIEGO ALEJANDRO TÉLLEZ BERNAL

Trabajo para optar por el título de ingeniero electrónico

Director
Ivan Dario Ladino Vega

FUNDACIÓN UNIVERSITARIA LOS LIBERTADORES
FACULTAD DE INGENIERÍA Y CIENCIAS BÁSICAS
INGENIERÍA ELECTRÓNICA
BOGOTÁ D.C
2017

Nota de aceptación

_____

_____

_____

_____

_____

Firma

Nombre:

Presidente del jurado

_____

Firma

Nombre:

Jurado

_____

Firma

Nombre:

Jurado

Bogotá D.C, Septiembre 21 de 2017

CONTENIDO

Pág.

LISTA DE FIGURAS

Pág.

RESUMEN

Con el acelerado desarrollo de la computación no se podrá garantizar en el futuro próximo el mismo nivel de seguridad en las redes de comunicaciones con los esquemas criptográficos actuales, debido a que el modelo matemático en el que se basa su funcionamiento se encuentra estructurado a partir de grupos abelianos; si se abordan grupos no abelianos (no conmutativos) como los grupos de trenza se puede elevar exponencialmente la complejidad computacional de los algoritmos de criptografía.

De hay que el problema que se abordó, fue el de obtener una primera aproximación a un esquema criptográfico a partir del uso de grupos de trenza, de tal forma que la complejidad del algoritmo se incremente con respecto a los algoritmos tradicionales sobre grupos abelianos. El algoritmo desarrollado consiste en un esquema DES donde las salidas de cada ronda se mezclan empleando trenzas.

Aunque los esquemas de hoy en día determinan una complejidad de tal escala, que para que un atacante pueda romper la seguridad de una red requiere unos tiempos muy superiores al de la validez de la información, es de esperar, que en un futuro próximo con el vertiginoso avance de los procesadores y de las estrategias de criptoanálisis dichos tiempos se vayan reduciendo, por tanto, es fundamental que se sigan desarrollando nuevos algoritmos que permitan seguir garantizando el mismo nivel de seguridad en el futuro.

Los sistemas computacionales crecen en sus capacidades de computo y almacenamiento año tras a año, es decir que las velocidades de computo en términos del número de millones de instrucciones por segundo (MIPS) y el número de núcleos dentro de los procesadores de las computadoras alcanzarán valores que permitan reducir los tiempos de ejecución de tareas computacionales de alta complejidad [11] tales como: el ataque por fuerza bruta a los sistemas de seguridad de las redes de comunicaciones y, la ejecución de algoritmos de criptoanálisis. Tal reducción de los tiempos de ejecución puede llevar la seguridad informática a un estado donde no se pueda garantizar el mismo nivel de seguridad con que se cuenta hoy en día [11].

Aumentar la seguridad de los sistemas informáticos plantea el reto de elevar el nivel de complejidad de los esquemas criptográficos [11], ello se traduce en la necesidad de abordar conceptos matemáticos distintos a los tradicionalmente empleados en la criptografía moderna; ello significa que la alternativa se encuentra en el álgebra abstracta y en particular en la teoría combinatoria de grupos, área de las matemáticas dedicada al análisis de grupos libres [10], ya que en relación a dichos grupos se tienen varias clases que no tiene la propiedad conmutativa, lo que los diferencia radicalmente de los grupos tradicionalmente empleados en la criptografía actual.

OBJETIVOS

OBJETIVO GENERAL

Implementar un algoritmo que permita cifrar textos en claro de 128 bits, que dada su estructura trenzada, existirán n posibilidades de formas de cifrado aumentando la complejidad del algoritmo exponencialmente con respecto a los algoritmos de cifrado simetricos implementados en la actualidad.

OBJETIVOS ESPECÍFICOS

- Introducir y definir conceptos de cifrado por derecha, cifrado por izquierda y habilitadores, que darán un mejor entendimiento del algoritmo DES extendido trenzado.
- Aumentar la capacidad de la clave de 48 bits a 112 bits con el fín de garantizar mayor seguridad frente a ataques por medio de criptoanálisis.
- Cifrar tramas de texto en claro de 128 bits, con n posibilidades de cruces acorde a la conformación de la clave y los habilitadores.

1. CRIPTOGRAFÍA

Un criptosistema general [2] es una colección indexada de criptosistemas básicos, indexados por un conjunto de claves denominado espacio de claves. Formalmente se define como:

Definición 1.1.1. Un criptosistema corresponde a una tupla (P , C , K , E , D) donde:

P = el conjunto de unidades de mensajes de texto planos, se denomina el espacio de texto-plano.

C = el conjunto de unidades de texto-cifrado, se le llama espacio de texto-cifrado

K = un conjunto de indices llamado el espacio de claves. Los elementos se denominan claves.

E = un conjunto de indices llamado el espacio de claves. Los elementos , indexado por el espacio de claves y. Se le denomina el conjunto de funciones de encripción.

D = un conjunto de funciones indexado también por el espacio de claves. Se le denomina el conjunto de funciones de encripción.

La propiedad básica de un criptosistema radica en que, para cada $k \in K$, existe una clave K' y una función g_k inversa de f_k, de tal forma que para cada mensaje plano $m : g_k\left(f_k\left(m\right)\right)$.

Subyacente a las funciones de cifrado y descifrado se encuentra el denominado grupo plataforma, es decir un grupo (en el contexto de las matemáticas [4]) sobre el cual se realizan todas las operaciones relacionadas con las funciones de cifrado, descifrado y la generación de claves. En algunos casos los grupos plataforma son de naturaleza abeliana, lo que se traduce en que las operaciones sobre el grupo son simples ya que las representaciones (en el sentido matemático [6]) son por lo general transformaciones de naturaleza lineal; sim embargo no son operaciones tan simples como las que se tendrían si la plataforma fuera un campo (cuerpo en matemáticas) debido a que los grupos tienen menos propiedades y operaciones que los campos.

1.1. CRIPTOGRAFIA DE CLAVE PRIVADA

El sistema de clave privada o simétrica utiliza una clave para cifrado y descifrado. Tanto el emisor y receptor comparten la misma clave en secreto para intercambiar información por medio de un algoritmo de cifrado. Si es de conocimiento la clave, el algoritmo será más vulnerable de este modo el mensaje ininteligible quedará al descubierto aplicando la clave sobre el algoritmo de cifrado.

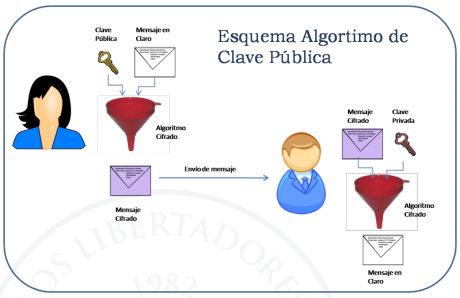

Figura 1.1 : Esquema de criptografía simétrica [12].

En la imagen anterior se puede denotar la funcionalidad de la clave privada. Alice el agente emisor enviará un mensaje a Bob agente receptor, por medio de una encriptación se envia el mensaje por un canal inseguro en el que Eve puede interceptarlo, pero si no es de su conocimiento la clave privada no podrá acceder al texto claro que se transmiste. El receptor recibirá el texto cifrado, por medio de la clave privada y la tecnica de encriptación podrá conocer la información.

1.2. CRIPTOGRAFIA DE CLAVE PÚBLICA

Los algoritmos de clave asimétrica emplean dos claves separadas, pública y privada, como los nombres lo indican, la clave pública es de conocimiento en el medio de comunicación, mientras que la clave privada solo la conoce el emisor que envía el mensaje. Esta pareja de claves se generan solo una vez por medio del método criptográfico empleado, por lo cual se asegura la imposibilidad de que dos personas tengan una misma clave.

Como se describe en la imagen 2, el emisor envía un mensaje cifrado, por medio de la clave pública compartida, cuando la información llega al destinatario, este haciendo uso de la clave privada y el metodo de cifrado tendrá acceso al texto en claro. Las tecnicas de cifrado para la clave pública que se emplean tienen la característica de ser algoritmos complejos por el

tamaño de bits de las claves públicas y privadas que se generan.

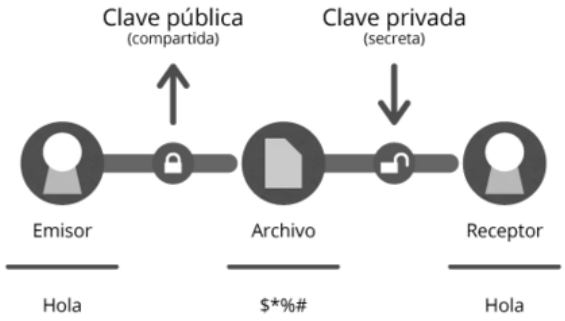

Figura 1.2 : Esquema de criptografía simétrica [8].

1.3. RED FEISTEL

Esta red fue diseñada por Horst Feistel, denominado el padre del cifrado de bloques, ya que los principales algoritmos implementados emplean celdas de Feistel. La principal característica de esta red es su simplicidad, por que para cifrar y descifrar un texto en claro no hay necesidad de modificar su estructura, simplemente se invierte el orden en que se aplica las subclaves que componen la clave.

La estructura consta de una entrada dividida en dos campos de igual longitud para el texto plano (en la figura 1.3, L y R) y de una entrada para la clave distribuida en una serie de subclaves, La parte derecha del texto (R) y una subclave (ko) son aplicadas a una funcion F, que consiste basicamente en el porceso de cifrado; paso seguido, el resultado es operado con el lado L mediante una operación or exclusiva. El porceso continua en forma iterada hasta completar un detreminado numero de rondas, por ejemplo en el algoritmo DES se emplean 16 rondas. El proceso de la celda de Feistel se muestra en la siguiente imagen para una iteración[1].

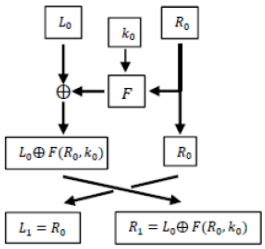

Figura 1.3 : Esquema de una red Feistel [1].

1.4. SISTEMA DE CIFRADO DES (DATA ENCRYPTION STANDARD)

El DES es un sistema criptográfico basado en red Feistel,[1] que opera sobre bloques de información de 64 bits. Comúnmente la clave se expresa en bloques de 64 bits de los cuales cada octavo bit se utiliza para paridad. El sistema DES comienza con una permutación sobre el texto en claro, posteriormente se divide en dos secciones denominadas derecha e izquierda de 32 bits, cada una para aplicar 16 rondas de una función f de tipo Feistel junto con la generación de 16 subclaves. Para la última ronda se concatenan los bloques de derecha e izquierda y aplicando sobre este una permutación final. Para el descifrado, se realiza el mismo procedimiento, la única variación es el orden de las 16 claves, para cifrado es en orden ascendente y para este caso de descifrado es de orden descendente. En la siguiente imagen se describe la estructura del cifrado DES y en [13] se detalla con profundidad, la generación de las subclaves, y las operaciones matemáticas.

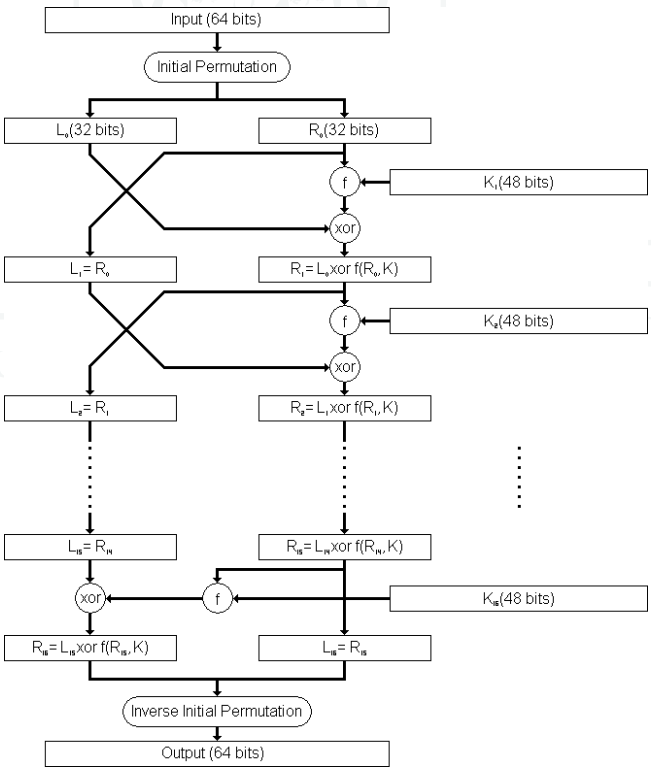

Figura 1.4 : Estructura cifrado DES convencional [7].

1.5. VULNERABILIDADES DE CIFRADO POR BLOQUES

En general la vulnerabilidad del cifrado DES radica en primera instancia en que es de tipo lineal, adicionalmente las operaciones de cifrado consisten básicamente en or exclusivas, ello lleva a que los ataques a este algoritmo son tipo diferencial y se enfocan en el proceso de generación de claves [3]. En el caso de esta propuesta la estrategia desarrollada consistió en trenzar las salidas y entradas entre las diferentes rondas del algoritmo DES, ello determina que el trenzado o no trenzado entre rondas consecutivas se traduce en un bit adicional para la clave total.

2. GRUPO DE TRENZAS

El matemático alemán Adolf Hurwitz en un artículo publicado en 1891 y dedicado a las coberturas ramificadas de superficies mencionó por primera vez en forma indirecta el concepto de grupo de trenzas, sin embargo, fue Emil Artin en 1920 quien introdujo formalmente la teoría de las trenzas al abordar un problema relacionado con objetos topológicos que modelaban el entrelazado de cuerdas en un espacio Euclidiano. Artin mostró que las trenzas con un número fijo de cuerdas conforman un grupo [5]. El hecho de que las trenzas son objetos topológicos y que además sean grupos determina que su estudio sea desarrollado tanto por algebristas como por topólogos, por tanto las trenzas están relacionadas con otros abstractos de la matemática como las Categorías, los Nudos y los Grupos Cuánticos. En la grafica mostrada a continuación se puede apreciar una trenza de cuatro hebras, donde las líneas rotas sobre las hebras significa que pasan por debajo de la hebra cuya curva es continua.

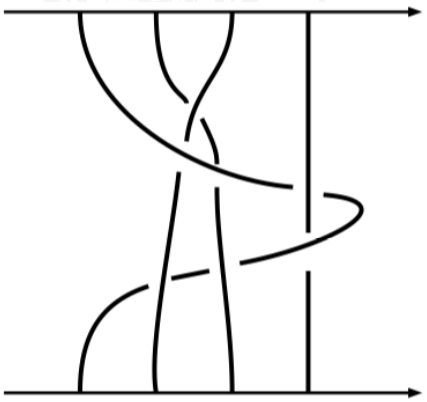

Figura 2.1 : Grupos de trenza B_4 [5].

Definicion 3.1: El grupo Artin B_n [9] es el grupo generado por $n-1$ generadores $\sigma_1, \sigma_2, ..., \sigma_{n-1}$ y las relaciones de trenza:

$$\sigma_i\sigma_j = \sigma_j\sigma_i \forall i, j = 1, 2, ..., n-1 con \mid i-j \mid \geq 2 \tag{1}$$

$$\sigma_i\sigma_{i+1}\sigma_i = \sigma_{i+1}\sigma_i\sigma_{i+1} \forall i = 1, 2, ..., n-1 \tag{2}$$

Por lo tanto el grupo de trenza para $i, j \in \{1, 2, ..., n-1\}$ tiene la presentación:

$$B_n = \langle \sigma_1, ..., \sigma_{n-1} \mid (\sigma_i \sigma_j = \sigma_j \sigma_i \forall \mid i - j \mid \geq 2), (\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1} \forall i < n) \rangle \qquad (3)$$

El grupo $B_1 = 1$ por definición corresponde al grupo trivial. El grupo B_2 es el grupo sin relaciones generado por un solo $\sigma 1$, es decir un grupo cíclico infinito [6].

2.1. OPERACIONES DESDE EL PUNTO DE VISTA GEOMÉTRICO

La operación para el grupo de trenzas es la concatenación, es decir que la unión de dos o más trenza como resultado dará otra trenza. El elemento neutro para las trenzas se denomina identidad, ya que es un conjunto de hilos paralelos que en ningún instante se entrecruzan, al concatenarla con otro conjunto de trenzas el resultado será el mismo, ahora el elemento simétrico es la inversa de la de la trenza que al unirlas se obtendrá la trenza identidad. En la siguiente figura se describe visualmente cada propiedad para denominar a las trenzas como un grupo.

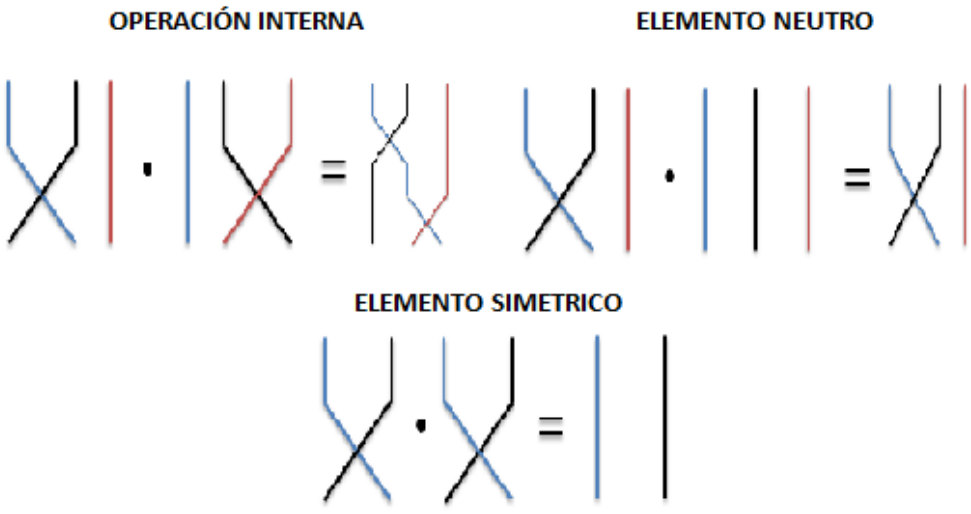

Figura 2.2 : Operaciones de los grupos de trenza.

2.2. RELACIÓN ENTRE GRUPOS DE TRENZAS Y RED FEISTEL

Al observar la red Feistel se puede distinguir que corresponde a una estructura trenzada de dos líneas, cada cruce es una iteración donde se producen las operaciones de la funcion f y la operación XOR [1].

Figura 2.3 :Red Feistel y grupos de trenza [1].

Sucede lo mismo para un algoritmo de cifrado DES. Teniendo esto claro, se introduce dos conceptos, que son cifrado por derecha y cifrado por izquierda. Basicamente el primero se realizará la operación de la funcion f para la parte derecha del texto en claro, este resultado se operará con la parte izquierda dentro de una operación XOR. Caso contrario ocurre con el cifrado por izquierda, la función f se realiza entre la clave y la parte izquierda del texto en claro, el resultado y la parte derecha se aplican a la operación XOR. En la figura 2.4, se plantea gráficamente la explicación de estos dos conceptos.

Partiendo de figura anterior desde el punto de vista geométrico, se puede observar dos trenzas similares a las de la figura 2.2 empleadas para la explicación del elemento neutro. Es decir que si empleamos una sola ronda de cifrado, por ejemplo por derecha, entonces el proceso de descifrado se debe realizar con la operación por izquierda empleando la misma subclave. Ahora si el proceso de cifrado involucra varias rondas, entonces la primera ronda de descifrado corresponde a la inversa de la última ronda de cifrado, es decir si la ultima consiste en un cifrado por derecha entonces la primera ronda de descifrado es por izquierda y asi sucesivamente hasta llegar a ultima ronda de descrifrado, en la cual se realiza proceso inverso de la primera ronda de cifrado.

Este es el principio basico del cifrado simétrico DES con trenzas. La generación de claves, es idéntica como se realiza en el cifrado DES convencional, al igual que las operaciones que se realizan en la funcion F, sin embargo se tiene un conjunto de 16 bits adicionales de clave

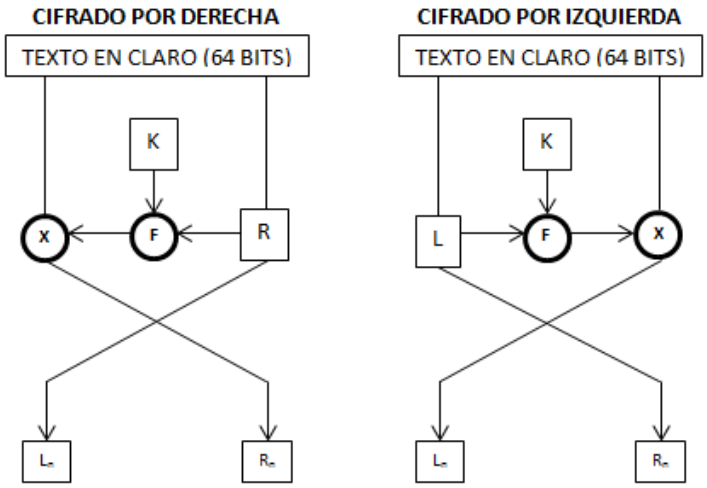

Figura 2.4 :Esquema cifrado por derecha e izquierda.

(los habilitadores figura 2.5), que corresponden a las dieciseis escogencias al azar de cifrado por izquierda o por derecha. Al igual que en el cifrado DES convencional, en el proceso de descifrado el conjunto de subclaves se aplica en forma inversa a la del cifrado.

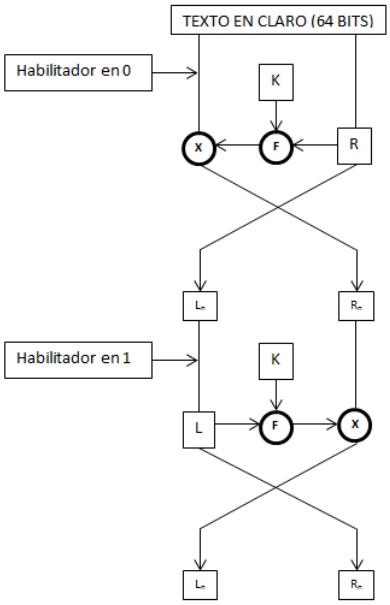

Figura 2.5 : Cifrado y Descifrado de una ronda DES trenzado.

3. IMPLEMENTACIÓN

A continuación se detallará la descripción del algoritmo a través del software Matlab y la herramienta Simulink.

El desarrollo del algoritmo de DES extendido trenzado se divide en las siguientes etapas.

- Pseudocódigos de función F, generador de subclaves y habilitadores.
- Creación de bloques en Simulink para cada operación basados en pseudocódigos.
- Pruebas del algoritmo para dos iteraciones de cifrado y descifrado.

El primer paso se diseñan los pseudocódigos de la función F de un sistema DES convencional. Se definen las variables de entrada, para este caso la subclave y una sección del texto cifrado que se denominan k y R_o respectivamente. Se realizan la operación XOR entre la subclave y el resultado de la permutación E, posteriormente se aplican las tablas de sustición S (Ver anexos 6.1).

La generación de subclaves se denotan como primera instancia por una tabla de permutación PC-1, seguido se crean 16 bloques $C_i y Di$ cada par de bloques entran en una función que desplazará un valor númerico determinado de bits por cada iteración, al resultado de cada bloque se aplica la tabla de permutación PC-2 (Ver anexos 6.2).

Por medio de Simulink se generan bloques de las funciones anteriormente mencionadas. Se realiza la prueba de funcionamiento para un cifrado DES simple, por una ronda, se toma un texto en claro de 64 bits hexadecimal, se aplica un cifrado por derecha para una sola ronda, obteniendo a la salida un bloque de texto cifrado. A continuación se aplica sobre este bloque el descifrado, por lo tanto se tendría a la salida el mensaje de entrada de 64 bits.

El diagrama de bloques de la figura 3.1 corresponde a la figura 2.5 desde otro punto de vista, donde se muestra descriptivamente el proceso de cifrado y descifrado por una ronda. Comparando la entrada y la salida (in, out en figura 3.1), la sustracción entre estos dos arreglos, se obtiene un resultado de un vector de ceros , dando a entender que existe igualdad entre el texto en claro y el texto cifrado.

Este es el principio básico que se tomará como base para el diseño del DES extendido trenzado.

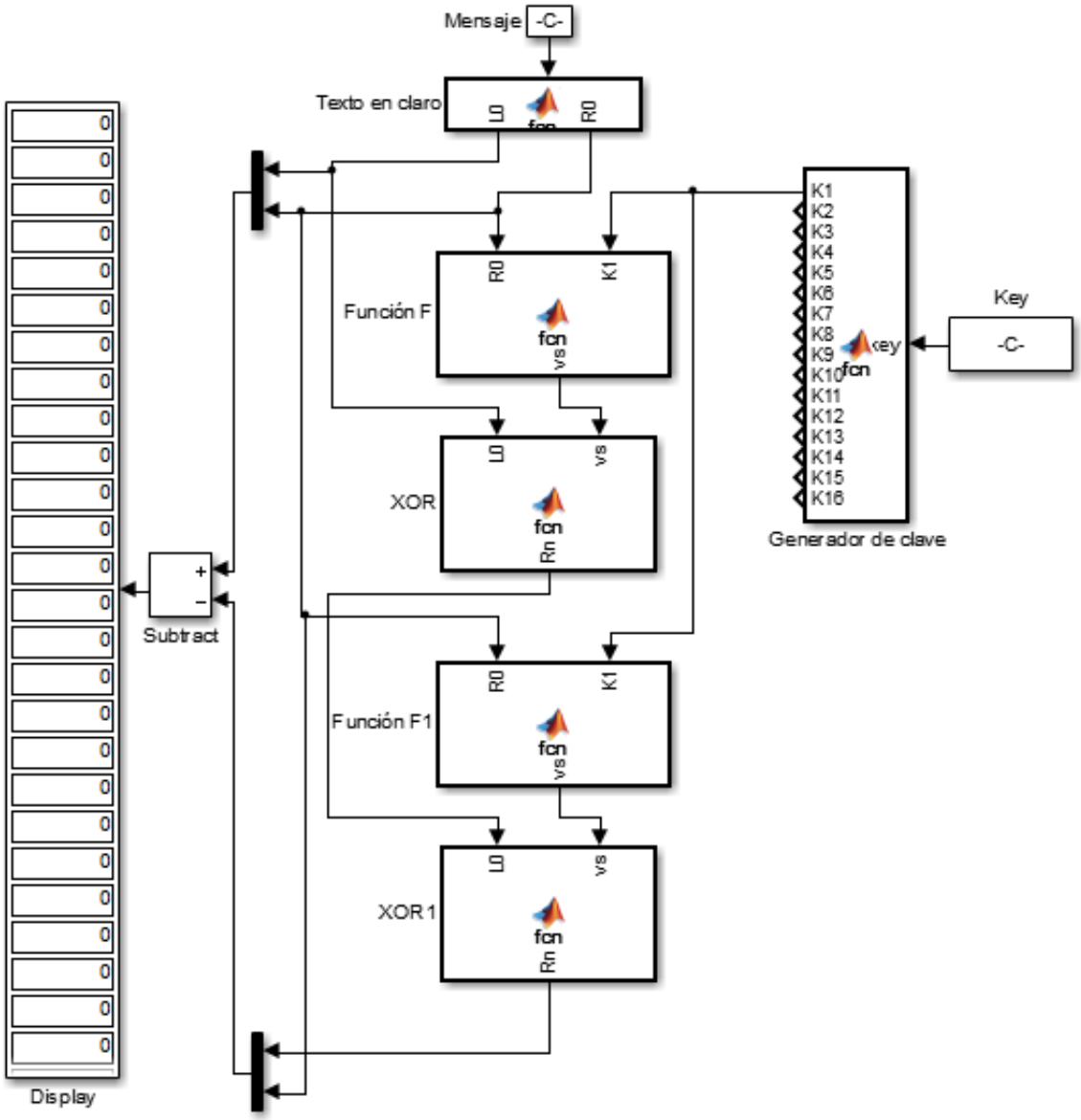

Figura 3.1 : Cifrado y descifrado DES en Simulink.

Para cifrar textos en claro de 128 bits, se divirá este en cuatro secciones de igual longitud (32 bits), se denominará cada sección como una hebra. Se insertarán funciones de Feistel en las hebras, de ahí se tendrán tres posibilidades de que cada hebra cruce con las tres restantes. En terminos matemáticos, en el sistema DES hay posibilidad de un cruce entre las dos hebras 2^1, obteniendo dos combinaciones, cifrar por derecha o izquierda. Para el caso del DES extendido trenzado, el número de posibilidades de cruce aumenta considerablemente obteniendo 2^4 posibles combinaciones por ronda.

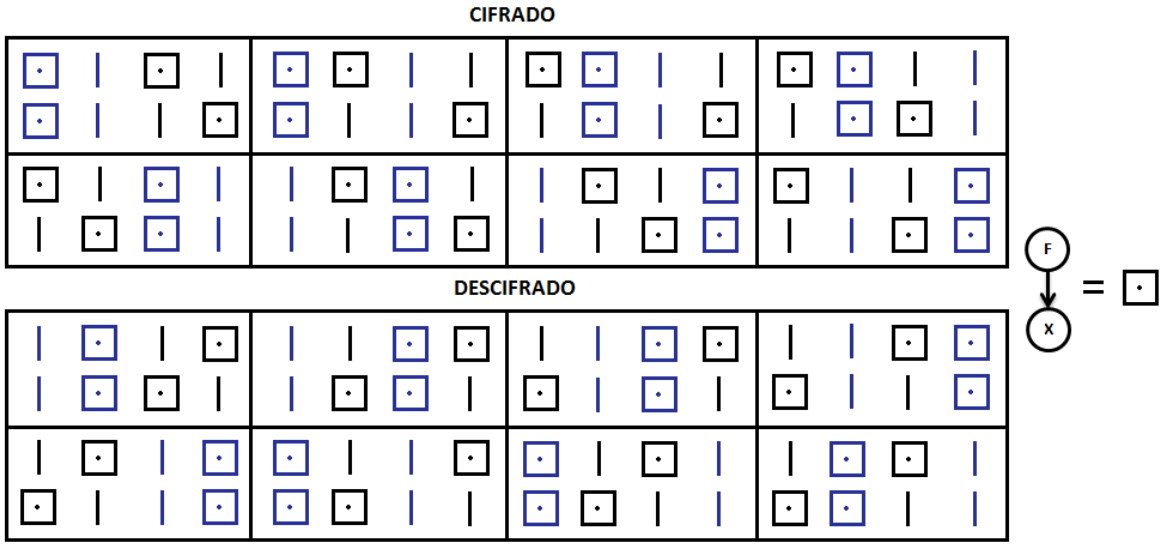

Figura 3.2 : Número de combinaciones de cifrado y descifrado para EDES trenzado.

En la figura 3.2 se describen todas las posibilidades de cruces para cifrado y la respectiva inversa de cada combinación que hace referencia al descifrado. Las figuras que corresponden al mismo color indican que entre ellas se realiza la trenza.

La generación de los valores de habilitadores se basan en la figura 3.2, las combinaciones tendrán una codificación de 4 bits, que controlarán el sentido de la trenza.

Combinaciones (Cifrado)															
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Combinaciones (Descifrado)															
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1111	1110	1101	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001	0000

Tabla 3.1 : Codificación de combinaciones EDES Braids.

A partir de la tabla 1, se puede denotar la codificación que se utilizó en el diseño del algoritmo,

es importante aclarar, que esta codificación es determinada por el diseñador. También para descifrar el texto cifrado, se debe tener en cuenta que la codificación de cada combinación es la inversa que la implementada en el cifrado.

3.1. IMPLEMENTACIÓN EN MATLAB (SIMULINK)

Como se mencionó anteriormente, los habilitadores tendrán la función de controlar el sentido de la trenza, y determinará la dirección de cada hebra hacia la función F, XOR o salida.

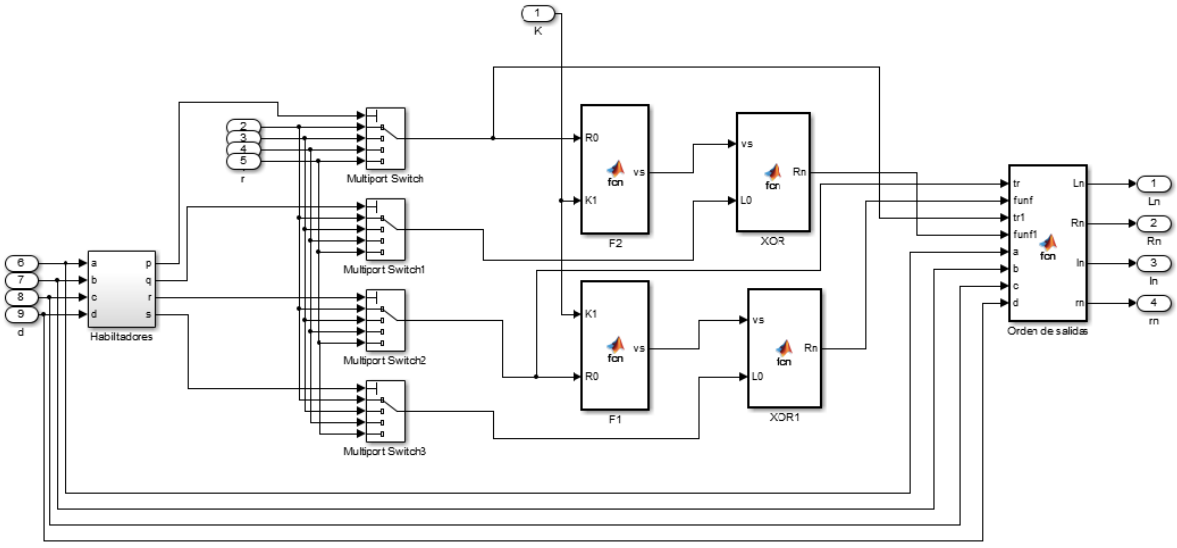

Figura 3.3 : Esquema general en simulink

Los habilitadores pasarán por una función (Habilitadores en figura 3.3) la cual direccionará cada vector en el sentido correcto de acuerdo a la codificación que se implemente, las salidas de esta función conectarán a 4 multiplexores, dependiendo de el valor númerico (1,2,3 y 4), habilitarán los vectores (Ln, Rn, ln, rn) que enlazarán con la función f (F1 ó F2) y la operación XOR (XOR ó XOR1). Las salidas de las operaciones XOR, junto con los enlaces de las señales provenientes de los multiplexores Multiport Switch y Multiport Switch3 respectivamente, se dirigen como entradas a la funcion "Orden de salidas", que redireccionan las señales de acuerdo a los habilitadores de entrada. Del esquema de la figura 3.3 se integrará en un bloque que tendrá como entradas cada habilitador, la subclave de determinada ronda, y los vectores de texto en claro o cifrado.

4. RESULTADOS

Para comprobar el correcto funcionamiento del algoritmo, se realiza la simulación en Matlab (Simulink), para dos rondas de cifrado y dos rondas de descifrado. Se espera obtener a la salida el texto en claro que se determina en la entrada.

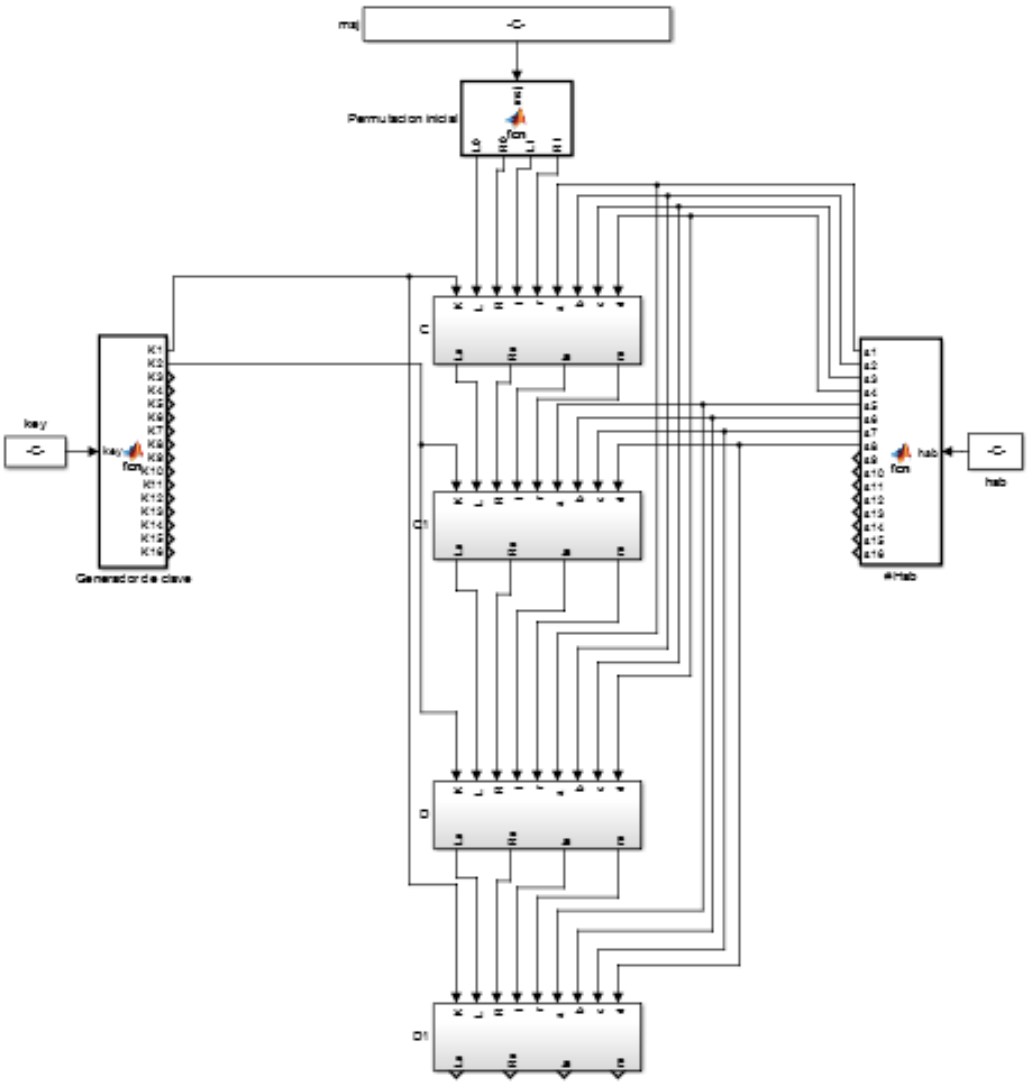

Figura 3.4 : EDES Braids, simulación dos rondas.

Los sistemas de cifrado (C y C1 en figura 3.4) y sistemas de descifrado (D y D1) son diferentes, ya que como se menciona anteriormente, la codificación que se implemente en descifrado será la inversa a la de cifrado, también cambiará el sentido de los habilitadores es decir, que a la entrada de la funcion (a,b,c,d) "Habilitadores" (Figura 3.3) pasarán por una compuerta de negación NOT (a',b',c',d'), ello garantiza descifrar el texto por rondas, la clave enlazará en cada bloque de descifrado en sentido contrario al cifrado.

De la siguiente figura, se comparan los resultados que se obtienen a la salida del descifrado con respecto al texto en claro en bits.

```
int =

  Columns 1 through 24

    0     0     0     0     0     0     0     1     0     0     1     0     0     0     1     1     0     1     0     0     0     1     0     1

  Columns 25 through 48

    0     1     1     0     0     1     1     1     1     0     0     0     1     0     0     1     1     0     1     0     1     0     1     1

  Columns 49 through 72

    1     1     0     0     1     1     0     1     1     1     1     0     1     1     1     1     0     0     1     1     0     1     1     0

  Columns 73 through 96

    0     0     1     0     0     0     0     1     1     1     1     0     0     0     0     1     0     1     1     1     1     1     0     1

  Columns 97 through 120

    1     0     1     0     1     0     0     0     1     0     1     1     1     0     1     0     0     0     0     1     0     0     1     0

  Columns 121 through 128

    1     1     1     0     1     0     1     1

out =

  Columns 1 through 24

    0     0     0     0     0     0     0     1     0     0     1     0     0     0     1     1     0     1     0     0     0     1     0     1

  Columns 25 through 48

    0     1     1     0     0     1     1     1     1     0     0     0     1     0     0     1     1     0     1     0     1     0     1     1

  Columns 49 through 72

    1     1     0     0     1     1     0     1     1     1     1     0     1     1     1     1     0     0     1     1     0     1     1     0

  Columns 73 through 96

    0     0     1     0     0     0     0     1     1     1     1     0     0     0     0     1     0     1     1     1     1     1     0     1

  Columns 97 through 120

    1     0     1     0     1     0     0     0     1     0     1     1     1     0     1     0     0     0     0     1     0     0     1     0

  Columns 121 through 128

    1     1     1     0     1     0     1     1
```

Figura 3.5 : Resultados de simulación

El texto en claro (int en figura 3.5) y texto descifrado (out en figura 3.5) en sistema binario, corresponden al mismo valor númerico bit a bit,los habilitadores están controlando el sentido de las trenzas y además cada salida de ronda están siendo direccionadas correctamente en cada salida (Ln, Rn, ln y rn). Si se realizan 16 rondas o iteraciones como generalmente sucede en el cifrado DES convencional, por cada ronda se tendrán 4 habilitadores adicionales, en total 64 valores númericos binarios, dando a entender que se adicionarán 64 bits a cada subclave que consta de 48 bits. 112 bits en total por generación de las subclaves.

5. CONCLUSIONES

En el presente documento se estableció un algoritmo de cifrado de grupos no abelianos. A diferencia del DES convencional, el algoritmo planteado ofrece una mayor seguridad frente a practicas de criptoanálisis de la actualidad, ya que existirá una alta exigencia a nivel computacional para romper el cifrado.

Las estructuras matemáticas abstractas como los grupos de trenza, permiten concebir nuevas estrategias criptográficas con miras a un futuro incierto por el advenimiento de la computación cuántica. Complejizar los algoritmos de cifrado no necesariamente lleva a aumentar el numero de bits de clave, se puede conseguir a tráves de abordar estructuras matemáticas mas complejas. Abordar nuevas soluciones a problemas generales de la ingeniería significa en muchos casos adentrase en el área de las matemáticas abstractas.

De acuerdo a la implementación que se realizó en Matlab, se puede concluir:

- Se extiende en número de bits en la generación de cada subclave de 48 bits a 112 bits (64 bits para habilitadores), aumentando significativamente la complejidad del algoritmo a ataques por medio de criptoanálisis de fuerza bruta.
- Por medio de los grupos de trenza, en cada ronda de cifrado existirán 16 posibilidades de cifrar el texto, a diferencia en el DES convencional que solo existe una posibilidad de cifrar (Cifrado por derecha).
- Se tendrán entradas de texto en claro de 128 bits, el doble que un cifrado DES convencional.

6. ANEXOS

% Se genera la función Feistel por medio de entradas de $K_n y Rn$ haciendo uso de la tabla de permutación E y la operación XOR entre el resultado de la permutación y la subclave.

6.1. FUNCIÓN F

function vs $= fcn(K1, R0)$
ER=$[R0(32); R0(1); R0(2); R0(3); R0(4); R0(5); R0(4); R0(5); R0(6); R0(7); R0(8);$
$R0(9); R0(8); R0(9); R0(10); R0(11); R0(12); R0(13); R0(12); R0(13); R0(14); R0(15);$
$R0(16); R0(17); R0(16); R0(17); R0(18); R0(19); R0(20); R0(21); R0(20); R0(21); R0(22);$
$R0(23); R0(24); R0(25); R0(24); R0(25); R0(26); R0(27); R0(28); R0(29); R0(28); R0(29);$
$R0(30); R0(31); R0(32); R0(1)]'$;
suma=$zeros(1, 48)$;
for $i = 1 : 48$
XOR=$ER(i) + K1(i)$;
if $XOR == 0 || XOR == 2$
suma(i)=0;
else suma(i)=1;
end
end
EX=suma;
% Permutación tablas S, convertir bloques de 6 bits a 4 bits.
S1=[14 4 13 1 2 15 11 8 3 10 6 12 5 9 0 7; 0 15 7 4 14 2 13 1 10 6 12 11 9 5 3 8;
4 1 14 8 13 6 2 11 15 12 9 7 3 10 5 0; 15 12 8 2 4 9 1 7 5 11 3 14 10 0 6 13];
S2=[15 1 8 14 6 11 3 4 9 7 2 13 12 0 5 10; 3 13 4 7 15 2 8 14 12 0 1 10 6 9 11 5;
0 14 7 11 10 4 13 1 5 8 12 6 9 3 2 15; 13 8 10 1 3 15 4 2 11 6 7 12 0 5 14 9];
S3=[10 0 9 14 6 3 15 5 1 13 12 7 11 4 2 8; 13 7 0 9 3 4 6 10 2 8 5 14 12 11 15 1;
13 6 4 9 8 15 3 0 11 1 2 12 5 10 14 7; 1 10 13 0 6 9 8 7 4 15 14 3 11 5 2 12];

S4=[7 13 14 3 0 6 9 10 1 2 8 5 11 12 4 15; 13 8 11 5 6 15 0 3 4 7 2 12 1 10 14 9; 10 6 9 0 12 11 7 13 15 1 3 14 5 2 8 4; 3 15 0 6 10 1 13 8 9 4 5 11 12 7 2 14];
S5=[2 12 4 1 7 10 11 6 8 5 3 15 13 0 14 9; 14 11 2 12 4 7 13 1 5 0 15 10 3 9 8 6; 4 2 1 11 10 13 7 8 15 9 12 5 6 3 0 14; 11 8 12 7 1 14 2 13 6 15 0 9 10 4 5 3];
S6=[12 1 10 15 9 2 6 8 0 13 3 4 14 7 5 11; 10 15 4 2 7 12 9 5 6 1 13 14 0 11 3 8; 9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6; 4 3 2 12 9 5 15 10 11 14 1 7 6 0 8 13];
S7=[4 11 2 14 15 0 8 13 3 12 9 7 5 10 6 1; 13 0 11 7 4 9 1 10 14 3 5 12 2 15 8 6; 1 4 11 13 12 3 7 14 10 15 6 8 0 5 9 2; 6 11 13 8 1 4 10 7 9 5 0 15 14 2 3 12];
S8=[13 2 8 4 6 15 11 1 10 9 3 14 5 0 12 7; 1 15 13 8 10 3 7 4 12 5 6 11 0 14 9 2; 7 11 4 1 9 12 14 2 0 6 10 13 15 3 5 8; 2 1 14 7 4 10 8 13 15 12 9 0 3 5 6 11];
% **Se organizan los vectores resultantes de las tablas de sustitución S.**
V1=zeros(1,4);
V2=zeros(1,4);
V3=zeros(1,4);
V4=zeros(1,4);
V5=zeros(1,4);
V6=zeros(1,4);
V7=zeros(1,4);
V8=zeros(1,4);
for i=1:48
if i==6
S1Ci=bi2de([EX(6) EX(1)]);
S1Cj=bi2de([EX(5) EX(4) EX(3) EX(2)]);
V1=dec2bin(S1(S1Ci+1,S1Cj+1),4);
V1=V1-'0';
end
if i==12
S2Ci=bi2de([EX(12) EX(7)]);
S2Cj=bi2de([EX(11) EX(10) EX(9) EX(8)]);
V2=dec2bin(S2(S2Ci+1,S2Cj+1),4);
V2=V2-'0';
end
if i==18
S3Ci=bi2de([EX(18) EX(13)]);
S3Cj=bi2de([EX(17) EX(16) EX(15) EX(14)]);
V3=dec2bin(S3(S3Ci+1,S3Cj+1),4);
V3=V3-'0';
end

if i==24
S4Ci=bi2de([EX(24) EX(19)]);
S4Cj=bi2de([EX(23) EX(22) EX(21) EX(20)]);
V4=dec2bin(S4(S4Ci+1,S4Cj+1),4);
V4=V4-'0';
end if i==30
S5Ci=bi2de([EX(30) EX(25)]);
S5Cj=bi2de([EX(29) EX(28) EX(27) EX(26)]);
V5=dec2bin(S5(S5Ci+1,S5Cj+1),4);
V5=V5-'0';
end
if i==36
S6Ci=bi2de([EX(36) EX(31)]);
S6Cj=bi2de([EX(35) EX(34) EX(33) EX(32)]);
V6=dec2bin(S6(S6Ci+1,S6Cj+1),4);
V6=V6-'0';
end
if i==42
S7Ci=bi2de([EX(42) EX(37)]);
S7Cj=bi2de([EX(41) EX(40) EX(39) EX(38)]);
V7=dec2bin(S7(S7Ci+1,S7Cj+1),4);
V7=V7-'0';
end
if i==48
S8Ci=bi2de([EX(48) EX(43)]);
S8Cj=bi2de([EX(47) EX(46) EX(45) EX(44)]);
V8=dec2bin(S8(S8Ci+1,S8Cj+1),4);
V8=V8-'0';
end
end

if length(V1)¡4
V1=[zeros(1,4-length(V1)) V1];
end
if length(V2)¡4
V2=[zeros(1,4-length(V2)) V2];
end

if length(V3)¡4
V3=[zeros(1,4-length(V3)) V3];
end
if length(V4)¡4
V4=[zeros(1,4-length(V4)) V4];
end
if length(V5)¡4
V5=[zeros(1,4-length(V5)) V5];
end
if length(V6)¡4
V6=[zeros(1,4-length(V6)) V6];
end
if length(V7)¡4
V7=[zeros(1,4-length(V7)) V7];
end
if length(V8)¡4
V8=[zeros(1,4-length(V8)) V8];
end

% **La salida se obtendrá la concatenación de las cajas de sustitución.**
vsi=[V1 V2 V3 V4 V5 V6 V7 V8];
vs=zeros(1,32);
for i=1:32
vs(i)=vsi(i);
end
end

6.2. GENERACIÓN DE SUBCLAVES

% **Se generan las 16 subclaves a partir de la clave principal determinada como variable de entrada.**
function [K1,K2,K3,K4,K5,K6,K7,K8,K9,K10,K11,K12,K13,K14,K15,K16] = fcn(key)
c=dec2bin(key,8);
c=reshape(c',1,64);

c=c-'0';

c=[c(57);c(49);c(41);c(33);c(25);c(17);c(9);c(1);c(58);c(50);c(42);c(34);c(26);c(18)
c(10);c(2);c(59);c(51);c(43);c(35);c(27);c(19);c(11);c(3);c(60);c(52);c(44);c(36)
c(63);c(55);c(47);c(39);c(31);c(23);c(15);c(7);c(62);c(54);c(46);c(38);c(30);c(22)
c(14);c(6);c(61);c(53);c(45);c(37);c(29);c(21);c(13);c(5);c(28);c(20);c(12);c(4)]';

ic0=c(1:1:28);
dc0=c(29:1:56);

ic=circshift(ic0',-1)';
dc=circshift(dc0',-1)';

K1F=[ic dc];
% **Se realizan los corrimientos de acuerdo a cada número de iteración.**
K1=[K1F(14);K1F(17);K1F(11);K1F(24);K1F(1);K1F(5); K1F(3);K1F(28);K1F(15);K1F(6);
K1F(21);K1F(10); K1F(23);K1F(19);K1F(12);K1F(4);K1F(26);K1F(8); K1F(16);K1F(7);
K1F(27);K1F(20);K1F(13);K1F(2); K1F(41);K1F(52);K1F(31);K1F(37);K1F(47);K1F(55);
K1F(30);K1F(40);K1F(51);K1F(45);K1F(33);K1F(48); K1F(44);K1F(49);K1F(39);K1F(56);
K1F(34);K1F(53); K1F(46);K1F(42);K1F(50);K1F(36);K1F(29);K1F(32)]';

ic1=circshift(ic',-1)';
dc1=circshift(dc',-1)';
K2F=[ic1 dc1];

K2=[K2F(14);K2F(17);K2F(11);K2F(24);K2F(1);K2F(5); K2F(3);K2F(28);K2F(15);K2F(6);
K2F(21);K2F(10); K2F(23);K2F(19);K2F(12);K2F(4);K2F(26);K2F(8); K2F(16);K2F(7);
K2F(27);K2F(20);K2F(13);K2F(2); K2F(41);K2F(52);K2F(31);K2F(37);K2F(47);K2F(55);
K2F(30);K2F(40);K2F(51);K2F(45);K2F(33);K2F(48); K2F(44);K2F(49);K2F(39);K2F(56);
K2F(34);K2F(53); K2F(46);K2F(42);K2F(50);K2F(36);K2F(29);K2F(32);]';

ic2=circshift(ic1',-2)';
dc2=circshift(dc1',-2)';

K3F=[ic2 dc2];

K3=[K3F(14);K3F(17);K3F(11);K3F(24);K3F(1);K3F(5); K3F(3);K3F(28);K3F(15);K3F(6);
K3F(21);K3F(10); K3F(23);K3F(19);K3F(12);K3F(4);K3F(26);K3F(8); K3F(16);K3F(7);
K3F(27);K3F(20);K3F(13);K3F(2); K3F(41);K3F(52);K3F(31);K3F(37);K3F(47);K3F(55);
K3F(30);K3F(40);K3F(51);K3F(45);K3F(33);K3F(48); K3F(44);K3F(49);K3F(39);K3F(56);
K3F(34);K3F(53); K3F(46);K3F(42);K3F(50);K3F(36);K3F(29);K3F(32);]';

ic3=circshift(ic2',-2)';
dc3=circshift(dc2',-2)';
K4F=[ic3 dc3];

K4=[K4F(14);K4F(17);K4F(11);K4F(24);K4F(1);K4F(5); K4F(3);K4F(28);K4F(15);K4F(6);
K4F(21);K4F(10); K4F(23);K4F(19);K4F(12);K4F(4);K4F(26);K4F(8); K4F(16);K4F(7);
K4F(27);K4F(20);K4F(13);K4F(2); K4F(41);K4F(52);K4F(31);K4F(37);K4F(47);K4F(55);
K4F(30);K4F(40);K4F(51);K4F(45);K4F(33);K4F(48); K4F(44);K4F(49);K4F(39);K4F(56);
K4F(34);K4F(53); K4F(46);K4F(42);K4F(50);K4F(36);K4F(29);K4F(32);]';

ic4=circshift(ic3',-2)';
dc4=circshift(dc3',-2)';
K5F=[ic4 dc4];

K5=[K5F(1);K5F(17);K5F(11);K5F(24);K5F(1);K5F(5); K5F(3);K5F(28);K5F(15);K5F(6);
K5F(21);K5F(10); K5F(23);K5F(19);K5F(12);K5F(4);K5F(26);K5F(8); K5F(16);K5F(7);
K5F(27);K5F(20);K5F(13);K5F(2); K5F(41);K5F(52);K5F(31);K5F(37);K5F(47);K5F(55);
K5F(30);K5F(40);K5F(51);K5F(45);K5F(33);K5F(48); K5F(44);K5F(49);K5F(39);K5F(56);
K5F(34);K5F(53); K5F(46);K5F(42);K5F(50);K5F(36);K5F(29);K5F(32);]';

ic5=circshift(ic4',-2)';
dc5=circshift(dc4',-2)';
K6F=[ic5 dc5];

K6=[K6F(14);K6F(17);K6F(11);K6F(24);K6F(1);K6F(5); K6F(3);K6F(28);K6F(15);K6F(6);

K6F(21);K6F(10); K6F(23);K6F(19);K6F(12);K6F(4);K6F(26);K6F(8); K6F(16);K6F(7);
K6F(27);K6F(20);K6F(13);K6F(2); K6F(41);K6F(52);K6F(31);K6F(37);K6F(47);K6F(55);
K6F(30);K6F(40);K6F(51);K6F(45);K6F(33);K6F(48); K6F(44);K6F(49);K6F(39);K6F(56);
K6F(34);K6F(53); K6F(46);K6F(42);K6F(50);K6F(36);K6F(29);K6F(32);]';

ic6=circshift(ic5',-2)';
dc6=circshift(dc5',-2)';
K7F=[ic6 dc6];

K7=[K7F(14);K7F(17);K7F(11);K7F(24);K7F(1);K7F(5); K7F(3);K7F(28);K7F(15);K7F(6);
K7F(21);K7F(10); K7F(23);K7F(19);K7F(12);K7F(4);K7F(26);K7F(8); K7F(16);K7F(7);
K7F(27);K7F(20);K7F(13);K7F(2); K7F(41);K7F(52);K7F(31);K7F(37);K7F(47);K7F(55);
K7F(30);K7F(40);K7F(51);K7F(45);K7F(33);K7F(48); K7F(44);K7F(49);K7F(39);K7F(56);
K7F(34);K7F(53); K7F(46);K7F(42);K7F(50);K7F(36);K7F(29);K7F(32);]';

ic7=circshift(ic6',-2)';
dc7=circshift(dc6',-2)';
K8F=[ic7 dc7];

K8=[K8F(14);K8F(17);K8F(11);K8F(24);K8F(1);K8F(5); K8F(3);K8F(28);K8F(15);K8F(6);
K8F(21);K8F(10); K8F(23);K8F(19);K8F(12);K8F(4);K8F(26);K8F(8); K8F(16);K8F(7);
K8F(27);K8F(20);K8F(13);K8F(2); K8F(41);K8F(52);K8F(31);K8F(37);K8F(47);K8F(55);
K8F(30);K8F(40);K8F(51);K8F(45);K8F(33);K8F(48); K8F(44);K8F(49);K8F(39);K8F(56);
K8F(34);K8F(53); K8F(46);K8F(42);K8F(50);K8F(36);K8F(29);K8F(32);]';

ic8=circshift(ic7',-1)';
dc8=circshift(dc7',-1)';
K9F=[ic8 dc8];

K9=[K9F(14);K9F(17);K9F(11);K9F(24);K9F(1);K9F(5); K9F(3);K9F(28);K9F(15);K9F(6);
K9F(21);K9F(10); K9F(23);K9F(19);K9F(12);K9F(4);K9F(26);K9F(8); K9F(16);K9F(7);
K9F(27);K9F(20);K9F(13);K9F(2); K9F(41);K9F(52);K9F(31);K9F(37);K9F(47);K9F(55);
K9F(30);K9F(40);K9F(51);K9F(45);K9F(33);K9F(48); K9F(44);K9F(49);K9F(39);K9F(56);

K9F(34);K9F(53); K9F(46);K9F(42);K9F(50);K9F(36);K9F(29);K9F(32);]';

ic9=circshift(ic8',-2)';
dc9=circshift(dc8',-2)';
K10F=[ic9 dc9];

K10=[K10F(14);K10F(17);K10F(11);K10F(24);K10F(1);K10F(5); K10F(3);K10F(28);K10F(15);
K10F(6);K10F(21);K10F(10); K10F(23);K10F(19);K10F(12);K10F(4);K10F(26);
K10F(8); K10F(16);K10F(7);K10F(27);K10F(20);K10F(13);K10F(2); K10F(41);K10F(52);
K10F(31);K10F(37);K10F(47);K10F(55); K10F(30);K10F(40);K10F(51);K10F(45);K10F(33);
K10F(48); K10F(44);K10F(49);K10F(39);K10F(56);K10F(34);K10F(53); K10F(46);K10F(42);
K10F(50);K10F(36);K10F(29);K10F(32);]';

ic10=circshift(ic9',-2)';
dc10=circshift(dc9',-2)';
K11F=[ic10 dc10];

K11=[K11F(14);K11F(17);K11F(11);K11F(24);K11F(1);K11F(5); K11F(3);K11F(28);K11F(15);
K11F(6);K11F(21);K11F(10); K11F(23);K11F(19);K11F(12);K11F(4);K11F(26);K11F(8);
K11F(16);K11F(7);K11F(27);K11F(20);K11F(13);K11F(2); K11F(41);K11F(52);K11F(31);
K11F(37);K11F(47);K11F(55); K11F(30);K11F(40);K11F(51);K11F(45);K11F(33);K11F(48);
K11F(44);K11F(49);K11F(39);K11F(56);K11F(34);K11F(53); K11F(46);K11F(42);K11F(50);
K11F(36);K11F(29);K11F(32);]';

ic11=circshift(ic10',-2)';
dc11=circshift(dc10',-2)';
K12F=[ic11 dc11];

K12=[K12F(14);K12F(17);K12F(11);K12F(24);K12F(1);K12F(5); K12F(3);K12F(28);K12F(15);
K12F(6);K12F(21);K12F(10); K12F(23);K12F(19);K12F(12);K12F(4);K12F(26);K12F(8);
K12F(16);K12F(7);K12F(27);K12F(20);K12F(13);K12F(2); K12F(41);K12F(52);K12F(31);
K12F(37);K12F(47);K12F(55); K12F(30);K12F(40);K12F(51);K12F(45);K12F(33);K12F(48);
K12F(44);K12F(49);K12F(39);K12F(56);K12F(34);K12F(53); K12F(46);K12F(42);K12F(50);

K12F(36);K12F(29);K12F(32);]';

ic12=circshift(ic11',-2)';
dc12=circshift(dc11',-2)';
K13F=[ic12 dc12];

K13=[K13F(14);K13F(17);K13F(11);K13F(24);K13F(1);K13F(5); K13F(3);K13F(28);K13F(15);
K13F(6);K13F(21);K13F(10); K13F(23);K13F(19);K13F(12);K13F(4);K13F(26);K13F(8);
K13F(16);K13F(7);K13F(27);K13F(20);K13F(13);K13F(2); K13F(41);K13F(52);K13F(31);
K13F(37);K13F(47);K13F(55); K13F(30);K13F(40);K13F(51);K13F(45);K13F(33);K13F(48);
K13F(44);K13F(49);K13F(39);K13F(56);K13F(34);K13F(53); K13F(46);K13F(42);K13F(50);
K13F(36);K13F(29);K13F(32);]';

ic13=circshift(ic12',-2)';
dc13=circshift(dc12',-2)';
K14F=[ic13 dc13];

K14=[K14F(14);K14F(17);K14F(11);K14F(24);K14F(1);K14F(5); K14F(3);K14F(28);K14F(15);
K14F(6);K14F(21);K14F(10); K14F(23);K14F(19);K14F(12);K14F(4);K14F(26);K14F(8);
K14F(16);K14F(7);K14F(27);K14F(20);K14F(13);K14F(2); K14F(41);K14F(52);K14F(31);
K14F(37);K14F(47);K14F(55); K14F(30);K14F(40);K14F(51);K14F(45);K14F(33);K14F(48);
K14F(44);K14F(49);K14F(39);K14F(56);K14F(34);K14F(53); K14F(46);K14F(42);K14F(50);
K14F(36);K14F(29);K14F(32);]';

ic14=circshift(ic13',-2)';
dc14=circshift(dc13',-2)';
K15F=[ic14 dc14];

K15=[K15F(14);K15F(17);K15F(11);K15F(24);K15F(1);K15F(5); K15F(3);K15F(28);K15F(15);
K15F(6);K15F(21);K15F(10); K15F(23);K15F(19);K15F(12);K15F(4);K15F(26);K15F(8);
K15F(16);K15F(7);K15F(27);K15F(20);K15F(13);K15F(2); K15F(41);K15F(52);K15F(31);
K15F(37);K15F(47);K15F(55); K15F(30);K15F(40);K15F(51);K15F(45);K15F(33);K15F(48);
K15F(44);K15F(49);K15F(39);K15F(56);K15F(34);K15F(53); K15F(46);K15F(42);K15F(50);

K15F(36);K15F(29);K15F(32);]';

ic15=circshift(ic14',-1)';
dc15=circshift(dc14',-1)';
K16F=[ic15 dc15];

K16=[K16F(14);K16F(17);K16F(11);K16F(24);K16F(1);K16F(5); K16F(3);K16F(28);K16F(15);
K16F(6);K16F(21);K16F(10); K16F(23);K16F(19);K16F(12);K16F(4);K16F(26);K16F(8);
K16F(16);K16F(7);K16F(27);K16F(20);K16F(13);K16F(2); K16F(41);K16F(52);K16F(31);
K16F(37);K16F(47);K16F(55); K16F(30);K16F(40);K16F(51);K16F(45);K16F(33);K16F(48);
K16F(44);K16F(49);K16F(39);K16F(56);K16F(34);K16F(53); K16F(46);K16F(42);K16F(50);
K16F(36);K16F(29);K16F(32);]';

end

REFERENCES

[1] y Goldfeld D. Anshel, I. *The Artin-Feistel Symmetric Cipher.* 1(12), 2012.

[2] Kreuzer M. Rosenberger G. Baumslag G., Fine B. *A course in Mathematical Cryptography.* De Gruyter, 2010.

[3] Don. Coppersmith. *The data encryption standard (DES) and its strength against attacks.* IBM Journal of Research and Development, 1994.

[4] Foote R. Dummit D. *Abstract Algebra.* Wiley, New York, 2004.

[5] Artin E. *Theory of Braids.* Annals of Mathematics, 1947.

[6] Harris J. Fulton W. *Representation Theory.* Springer Verlag, Berlin, Heidelberg, New York, 1999.

[7] Orlin Grabbe. Des (how works?). urlhttp://manansingh.github.io/Cryptolab-Offline/c13-des-block.html.

[8] Juan Carlos Ríos Jiménez. Cifrado asimétrico. urlhttp://jcriosj5im7.blogspot.com.co/2016/10/cifrado-asimetrico.html.

[9] Turaev V. Kassel C. *Braid Groups, Graduate Texts in Mathematics.* Springer, New York, 2008.

[10] Schupp P.E. Lyndon R.C. *Combinatorial Group Theory.* Springer Verlag, Berlin, Heidelberg, New York, 1977.

[11] Aushakov A. Myashnikov A., Shpilrain V. *Non-Commutative Cryptography and Complexity of Group-Theoric Problems.* American Mathematical Society, 2011.

[12] Oscar Nogales. Certificados ssl i: Conceptos básicos. urlhttp://www.elblogtic.com/certificados-ssl-conceptos-basicos/.

[13] B. Schneier. *Applied Cryptography.* Wiley, India, 1996.