DISEÑO DE UN ALGORITMO DE CIFRADO DES TRENZADO (DES BRAIDS) PARA BLOQUES DE 64 BITS

CRISTIAN DAIR DUARTE USECHE

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

DISEÑO DE UN ALGORITMO DE CIFRADO DES TRENZADO (DES BRAIDS) PARA BLOQUES DE 64 BITS

CRISTIAN DAIR DUARTE USECHE

Trabajo para optar por el título de Ingeniero Electrónico

Director
Ivan Dario Ladino Vega

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

Nota de aceptación

_____

_____

_____

_____

_____

Firma

Nombre:

Presidente del jurado

_____

Firma

Nombre:

Jurado

_____

Firma

Nombre:

Jurado

Bogotá D.C., Diciembre 7 de 2017

CONTENIDO

Pág.

LISTA DE FIGURAS

RESUMEN

El siguiente documento pretende extender específicamente la implementación de un sistema de transmisión de información, donde se garantice un nivel de seguridad eficiente y que no sea "fácil" de violentar por agentes externos. Actualmente garantizar un buen nivel de seguridad, se vuelve una tarea ardua y extensa, al seguir basados en los tipos de esquemas criptográficos usados hasta ahora, ya que la raíz de funcionamiento para estos modelos matemáticos son los métodos abelianos (conmutativos); por otro lado al hacer uso de grupos no abelianos, como en este caso, suelen ser, los grupos trenzados, el nivel de seguridad puede ser mayor debido a su complejidad computacional con respecto a los sistemas de criptografía.

Para este caso el algoritmo desarrollado es denominado DES trenzado (DES Braids) y el cual requiere un manejo de datos de 64 bits, fundamentado en el proceso de criptografía de clave privada y grupo de trenza. Con esto se introducirán ciertos conceptos que permitirán llegar a un mayor entendimiento de la funcionalidad de este tipo de cifrado; partiendo esencialmente de el sistema DES convencional, pero en el cual las salidas utilizadas en cada ronda se van cruzando entre sí.

KEYWORD: Cifrado, Descifrado, Grupo de trenzas, Algoritmos.

INTRODUCCIÓN

Actualmente con el desarrollo tecnológico que se experimenta día a día, se puede ver el gran intercambio de información que se lleva a cabo por medio de internet, correos electrónicos, teléfonos celulares, tarjetas inteligentes, entre otros, resultando ser algo rutinario. De tal forma que al momento de proteger información y datos importantes, se requiere que el tiempo de ejecución sea más rápido que el tiempo en que pueda ser atacado el sistema de seguridad que brinda la red, es por eso que al hacer uso de sistemas criptográficos, la duración de estos procesos serán reducidos de forma acertada, por lo que cada avance en tecnología exige que se desarrollen mas sistemas que garanticen este tipo de seguridad y eficiencia, debido a que los sistemas de cómputo, tienen un avance año tras año en la capacidad de almacenamiento y velocidad de funcionamiento.

En 1973 la Oficina Nacional de Estándares (National Bureau of Standards: NBS) de los Estados Unidos publicó una solicitud para recibir propuestas de algoritmos de cifrado para proteger información durante su transmisión y almacenamiento [2].El sistema de cifrado DES (Data Encryption Standard), desarrollado por un equipo de la corporación IBM alrededor de 1974, respondió a este llamado y fue adoptado como estándar en 1977. El sistema DES fue uno de los primeros en desarrollarse comercialmente cuya estructura fue completamente publicada. Después de ser sometido a intensos escrutinios por una amplia comunidad de investigadores y sobrevivir por varios años se hizo evidente la necesidad de mejorarlo e incluso reemplazarlo.

El aumentar la seguridad de los sistemas de información, permite generar una mayor complejidad en los algoritmos criptográficos, de tal forma que se deben poner en práctica diferentes medios matemáticos de los manejados tradicionalmente en la criptografía moderna [11]; principalmente el hacer uso del algebra abstracta y la combinación de grupos permite visualizar que muchos de estos no ponen en práctica la propiedad conmutativa, la cual hace parte esencial de las matemáticas, y de esta manera los diferencia radicalmente de los grupos utilizados en la criptografía actual.

1. OBJETIVOS

1.1. OBJETIVO GENERAL

Generar la implementación de un algoritmo que brinde mayor seguridad en el proceso de cifrado de textos en claro de 64 bits, al llevarse a cabo cruces trenzados con n posibilidades en su desarrollo, teniendo como referencia los sistemas computacionales de protección de información manejados hasta el momento.

1.2. OBJETIVOS ESPECÍFICOS

- Diseñar el proceso de cifrado por derecha, cifrado por izquierda y la forma en que los habilitadores realizan este tipo de cruces, dando una idea más clara acerca del DES trenzado (DES braids).
- Implementar el proceso de diseño de tal forma que se pueda comparar la eficiencia de este algoritmo respecto al DES convencional, ya que la clave aumentara su capacidad de 48 bits a 64 bits de manera que brinde una mayor protección en el momento de transmisión de datos, al ser violentados por ataques basados en criptoanálisis.
- Validar el nivel de seguridad en el proceso de trenzado al utilizar este cifrado en textos de 64 bits, permitiendo así la combinación de sus habilitadores y claves o llaves de cifrado.

2. MARCO TEÓRICO

2.1. CRIPTOGRAFÍA

La criptografía representa un conjunto de técnicas orientadas a la solución de los problemas generados al enviar información a través de un canal inseguro. Mientras que el criptoanálisis es el conjunto de métodos que intentan debilitar alguna de las metas de seguridad que se persigue en la criptografía.

El cifrado es el conjunto de operaciones (criptográficas) usadas para hacer que la información que se envía sea incomprensible para cualquiera excepto para el receptor. Este proceso toma un fragmento de información conocido como texto plano y lo transforma en un texto cifrado o criptograma usando un fragmento de información adicional llamado llave para cifrar. El descifrado es el proceso inverso al cifrado. El receptor recupera el mensaje (texto plano) mediante este proceso a partir del texto cifrado y el uso de la llave para descifrar. El cifrado y descifrado forman, en conjunto, un esquema de cifrado o criptosistema [3], Ver Figura 2.1.

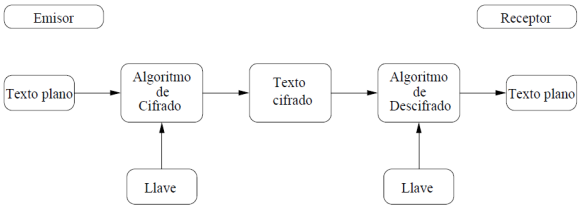

Figura 2.1 : Estructura de cifrado.[12].

Un criptosistema general [9] 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 quintupla (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.

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, existe una clave $K^{'}$, y una funcion g_k inversa de f_k, de tal forma que para cada mensaje plano m: g_k (f_k(m)).

Subyacente a las funciones de cifrado y cifrado se encuentra el denominado grupo plataforma, es decir un grupo (en el contexto de las matemáticas [14]) 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 [16]) son por lo general transformaciones de naturaleza lineal; sin 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.

2.1.1. Criptografía de clave privada

Los cifrados simétricos emplean la misma clave para cifrar y descifrar mensajes. El emisor, encargado de enviar el mensaje aplica un algoritmo simétrico para esconder la información usando la clave que es del conocimiento del receptor, cuando el mensaje llega a su destinatario, este haciendo uso de la clave acordada con el emisor, aplica el algoritmo para descifrar los datos transmitidos. Los principales problemas que surgen el este tipo de criptografía, es la distribución de la clave al momento de mantener un intercambio de información con más de dos destinatarios.

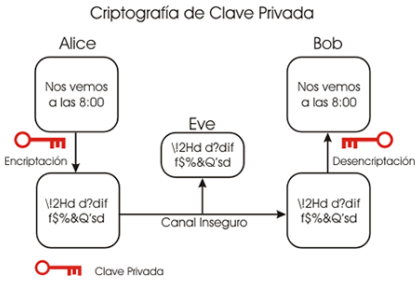

Figura 2.2 : Estructura clave privada.[15].

En la Figura 2.2, se puede observar la función que cumple la clave privada o simétrica. Donde el emisor en este caso Alice le envía un mensaje a Bob el cual cumple con el perfil de receptor: llevándose a cabo un proceso de encriptación para poder enviar el mensaje por el canal inseguro en el cual un agente externo puede interceptarlo en el ejemplo es Eve pero al no conocer la clave privada no podrá acceder al texto transmitido. Por lo cual el receptor podrá obtener el texto cifrado y haciendo uso de la clave privada y el proceso de desencriptación podrá conocer a totalidad la información.

2.1.2. Criptografía de clave pública

Los algoritmos de clave asimétrica emplean dos claves separadas, pública y privada, como los nombres lo indican, la clave publica 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.

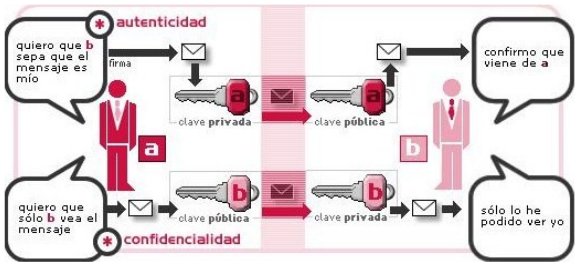

Figura 2.3 : Estructura clave pública.[13]

Como se ve en la Figura 2.3, el remitente usa la clave pública del destinatario para cifrar el mensaje, una vez cifrado, sólo la clave privada del destinatario podrá descifrar este mensaje, ya que es el único que la conoce. Por tanto se logra la confidencialidad del envío del mensaje, nadie salvo el destinatario puede descifrarlo. Si el propietario del par de claves usa su clave privada para cifrar el mensaje, cualquiera puede descifrarlo utilizando su clave pública. En este caso se consigue por tanto la identificación y autentificación del remitente, ya que se sabe que sólo pudo haber sido él quien empleó su clave privada (salvo que alguien se la hubiese podido robar). Esta idea es el fundamento de la firma electrónica.

2.1.3. 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, porque 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 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 función F, que consiste basicamente en el proceso de cifrado; paso seguido, el resultado es operado con el lado L mediante una operación or exclusiva. El proceso continua en forma iterada hasta completar un determinado numero de rondas, por ejemplo en el algoritmo DES se emplean 16 rondas. El proceso de la celda de Feistel se muestra en la Figura 2.4 para una iteración y expresa la propiedad que tienen este tipo de algoritmos para ser reversibles.

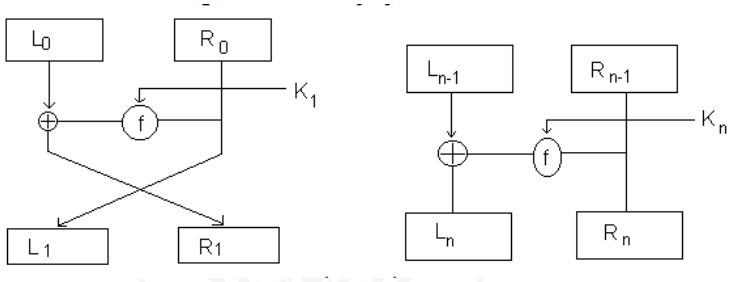

Figura 2.4 : Estructura red feistel.[8].

2.1.4. Sistea de cifrado DES (Data Encryption Standard)

El DES es un sistema de cifrado por bloque operando sobre bloques de información de 64 bits de longitud, de tal forma que produce un bloque de texto cifrado de igual tamaño. Comúnmente la clave se expresa como un bloque de 64 bits de los cuales cada octavo bit se usa para paridad y finalmente se eliminan quedando un bloque de 56 bits de longitud. El sistema DES inicia con una permutación sobre un bloque de texto plano de 64 bits. El bloque permutado se divide en bloques de 32 bits cada una para aplicar 16 iteraciones (rondas) de una función F de tipo Feistel que hace una combinación con la clave original. Cuando se termina la iteración numero 16, las mitades se unen de nuevo para formar un nuevo bloque y aplicar así la permutación final y terminando el algoritmo. Los bits de la clave original, forman 16 nuevas claves (subclaves) de 48 bits cada una y las cuales son usadas por cada iteración. En la función Feistel la mitad derecha del bloque de información se expande a un bloque de 48 bits mediante una función de expansión. Luego se combina con la subclave correspondiente mediante la operación binaria Xor. El bloque resultante pasa por 8 cajas de sustitución (tablas s) produciendo un bloque de 32 bits y se maneja según la tabla de permutación. La salida de esta función es combinada con la parte izquierda mediante un Xor

para formar la nueva mitad derecha del siguiente bloque, mientras q la última mitad derecha pasa a ser la nueva mitad izquierda. Todo este proceso forma una sola ronda del sistema DES, repitiéndose así 16 veces para cumplir con el proceso del algoritmo. Ver Figura 2.5.

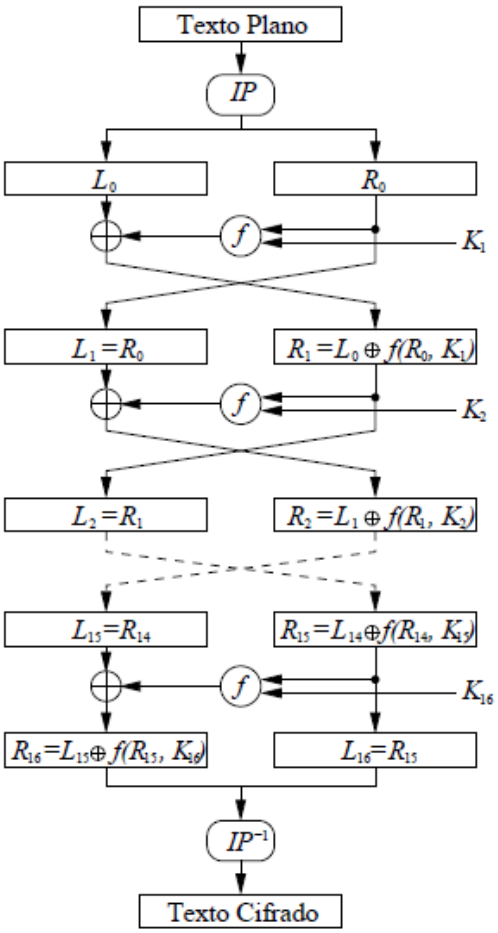

Figura 2.5 : Estructura DES convencional.[12].

- PERMUTACIÓN PC1 Y PC2.

Luego de que la clave de 64 bits se ha reducido a 56 bits, se realiza la permutación según la tabla PC1, ver Tabla 1. Este tipo de tabla se puede conocer como una permutación de compresión ya que permuta el orden de los bits y determina un subconjunto de los mismos. Por medio de esta clave, son generadas 16 subclaves de 48 bits de longitud que se usaran en las 16 rondas del DES. De igual forma en la formación de las subclaves es utilizada otra permutación de compresión la cual se conoce como PC2, la cual toma como entrada 56 bits y devuelve un subconjunto de 48 bits, ver Tabla 2

Tabla 1: Permutación PC1.

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

Tabla 2: Permutación PC2.

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

- GENERACIÓN DE LAS SUBCLAVES.

Para generar las subclaves, principalmente, se deben eliminar los 8 bits de paridad de la clave original de 64 bits. Los 56 bits restantes se permutan de la forma que dice la tabla PC1. De esta forma el bloque resultante se divide en dos partes de 28 bits cada una, llamadas de forma tradicional como C0 Y D0. Posteriormente, los bits realizan un corrimiento hacia la izquierda cierto número de posiciones según la tabla de la Tabla 3. A modo de ejemplo, podemos ver si un bloque de 28 bits realiza un movimiento en dos posiciones hacia la izquierda, esto significa que los bits de la posición tercera y cuarta, ahora ocupan la primera y segunda posición respectivamente. Los bits posteriores realizan un recorrido hasta la posición 26, mientras que los bits 1 y 2, ocuparan la posición 27 y 28 respectivamente. Produciendo así, 16 bloques Cj Y 16 bloques Dj de 18 bits con j=1,. . . ., 16. Después, se realiza la concatenación de los bloques Cj y Dj para formar un solo bloque CjDj, del cual son extraídos 48 bits según la permutación PC2 para cada j=1,. . . ., 16. De tal forma que los bloques resultantes son las llamadas subclaves.

Tabla 3: Corrimientos por ronda

Ronda	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Corrimiento	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

- PERMUTACIÓN IP E IP-1.

La permutación inicial IP se lleva a cabo antes de la ronda 1, mientras que la permutación final IP-1 se realiza después de la ronda 16. La tabla IP transpone al texto plano, ver Tabla 4 mientras que la tabla IP-1 regresa el orden de los bits a la configuración inicial, ver Tabla 5. Como se han manejado todas las tablas en este proceso, son leídas de izquierda a derecha y de arriba hacia abajo. De esta forma el bit 58 ubicado en la tabla IP se mueve a la primera posición, el bit 50 se mueve a la segunda y así sucesivamente.

Tabla 4: Permutación IP.

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Tabla 5: Permutación IP-1

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	4	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

- PERMUTACIÓN E.

El objetivo de esta tabla es expandir la mitad derecha de un bloque de información de 32 bits a 48 bits. Esto es realizado duplicando y reordenando la mitad de los bits de la mitad derecha, ver Tabla 6. Este tipo de tabla se conoce como tabla de expansión y tiene dos propósitos. Principalmente es igualar la longitud de la mitad derecha con la longitud de la clave para llevar a cabo el o-exclusivo. El siguiente es proporcionar finalmente un bloque más largo que pueda ser comprimido cuando se apliquen las cajas de sustitución.

Tabla 6: Permutación E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

- PERMUTACIÓN P.

Esta tabla de permutación es aplicada en la última etapa del cálculo de la función F, siendo aplicada a un bloque de 32 bits y los reordena de acuerdo a la Tabla 7.

Tabla 7: Permutación P

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

- CAJAS DE SUSTITUCIÓN DEL DES.

Las tablas de sustitución o tablas S, son 8 tablas de diferentes valores, reconocidas como S1, S2,. . . ., S8, que realizan la sustitución de un bloque de 6 bits por uno de 4 bits. Un bloque de 48 bits se divide en un sub bloque de 6 bits. Luego, cada sub bloque se opera con su caja

correspondiente. Cada caja S tiene 4 filas y 16 columnas, en cada fila hay una permutación de todos los posibles valores de 4 bits, lo cual significa que es una permutación de los números del 0 al 15. La entrada de 6 bits se distribuye de la siguiente forma: principalmente se elije la fila de acuerdo al valor binario formado por la concatenación del primero y sexto bit, mientras que la columna está dada por el valor binario de 4 bits que quedan en medio. como se ve en las siguientes Tablas:

Tabla 8: S1

S1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Tabla 9: S2

S2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

Tabla 10: S3

S3	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

Tabla 11: S4

S4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

Tabla 12: S5

S5	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
1	14	11	2	12	4	7	13	1 5	0	15	10	3	9	8	6	
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

Tabla 13: S6

S6	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

Tabla 14: S7

S7	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

Tabla 15: S8

S8	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

2.1.5. Problema de cifrar por bloques

En general los problemas 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 [5]. 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.

2.2. 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 .[7]. 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 seis hebras, donde las líneas rotas sobre las hebras significa que pasan por debajo de la hebra cuya curva es continua.

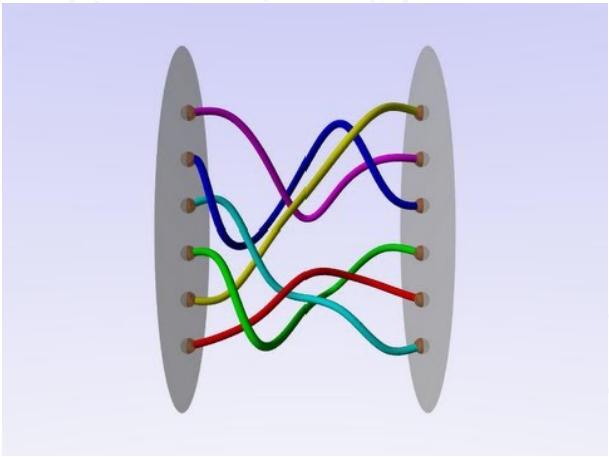

Figura 2.6 : Grupos de trenzas B_6.[6].

Definicion 3.1: El grupo Artin B_n [4] es el grupo generado por $n-1$ generadores σ_1, σ_2,...,σ_{n-1} y las relaciones de trenza:

$$\sigma_i\sigma_j = \sigma_j\sigma_i \forall i,j = 1,2,\ldots,n-1 \; con \mid i-j \mid \leq 2 \tag{1}$$

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

$$B_n = \sigma_1,\ldots,\sigma_{n-1} | (\sigma_i\ \sigma_j = \sigma_j\ \sigma_i\ \forall \mid i-j \mid \leq 2), (\ \sigma_i\ \sigma_{i+1}\ \sigma_i = \sigma_{i+1}\ \sigma_i\ \sigma_{i+1}\ \forall\ \mathrm{i} > \mathrm{n}) \tag{2}$$

El grupo B_1=1 por definición corresponde al grupo trivial. El grupoB_2 es el grupo sin relaciones generado por un solo σ_1, es decir un grupo cíclico infinito [16].

2.2.1. Operaciones Geométricas

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 Figura 2.7 se describe visualmente cada propiedad para denominar a las trenzas como un grupo.

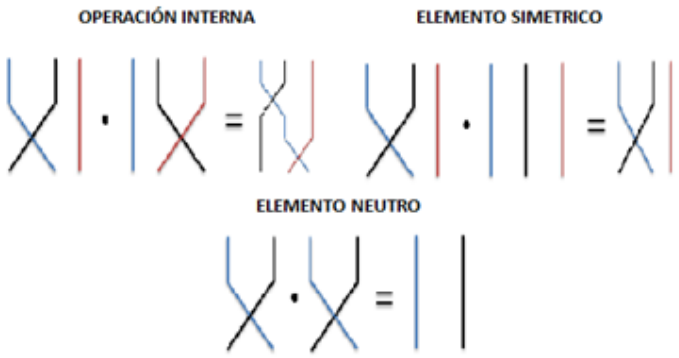

Figura 2.7 : Tipos de operaciones con trenzas.

2.3. RELACION RED FEISTEL Y GRUPOS DE TRENZAS

Al boservar 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].

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. Básicamente el primero se realizará la operación de la función 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 2.8, se plantea gráficamente la explicación de estos dos conceptos.

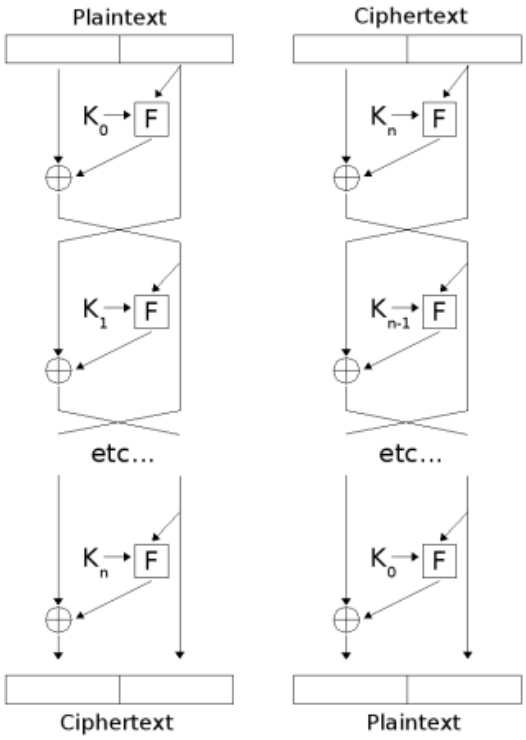

Figura 2.8 : Esquema de cruces red Feistel. [10].

Partiendo de figura anterior desde el punto de vista geométrico, se puede observar dos trenzas similares a las de la figura 2.7 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.

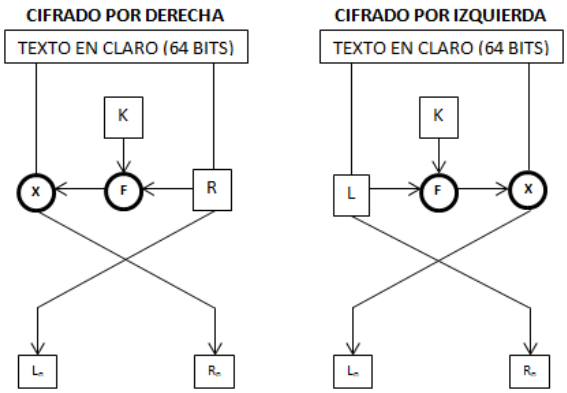

Figura 2.9 : Esquema de cifrado derecha e izqyuierda.

(los habilitadores figura 2.10), que corresponden a las dieciséis 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.

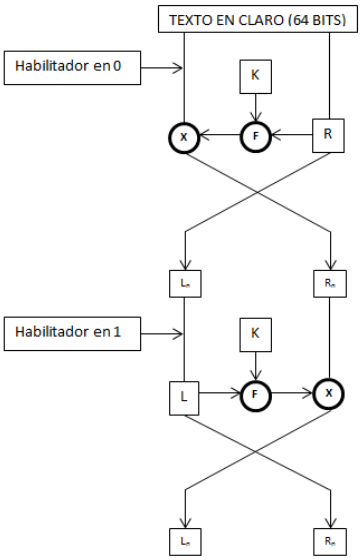

Figura 2.10 : Cifrado y descifrado para una ronda DES Braids.

3. IMPLEMENTACIÓN DES TRENZADO

La aplicación de este tipo de cifrado fue realizada por medio del software de MATLAB y Simulink, cumpliendo paso a paso con las siguientes etapas:

- Generar el pseudocódigo de la función F y las tablas de permutación para su debida aplicación en este diseño.
- Implementación de el generador de claves o llaves y habilitadores para ser utilizados en cada ronda del proceso de cifrado y descifrado.
- Realizar pruebas del algoritmo para un pequeño grupo de iteraciones de tal forma que el resultado al descifrar sea igual al de los datos utilizados al inicio del proceso.

Inicialmente para el desarrollo del pseudocódigo de la función F se requiere definir las variables que cumplirían con este proceso para cada permutación; respectivamente la primer variable es R0 y esta depende de la distribución que se lleva a cabo en el mensaje de entrada entre las cuales están (L0 y R0), y hacen referencia mediante L (para los 32 bits de la izquierda) y R (para los 32 bits de más a la derecha). De igual forma la otra variable es K, y esta se refiere a la subclave usada en cada una de las rondas del algoritmo. Posteriormente se realiza la aplicación de la tabla de permutación E, Tablas 6, a la variable R0; donde se obtiene un bloque de 48 bits. Este resultado se combina con la subclave K, por medio de una XOR. Después el nuevo bloque se divide en 8 sub-bloques de 6 bits de longitud y a estos se les aplica las cajas de sustitución S, Tablas 8 a la Tabla 15, y con los cuales se recupera de nuevo un bloque de 32 bits como se tenía originalmente y este será la salida de la función F. (Ver Anexo 6.1).

Para el desarrollo de las subclaves, se lleva a cabo el uso de la tabla de permutación PC-1, Tabla 1, de esta forma se crean 16 bloques Ci, Di y los cuales se forman a partir de la tabla de rotaciones circulares de derecha a izquierda que se verá en la Tabla 3. De esta forma cada par de bloques entran en una función que es la encargada de realizar el desplazamiento el número de veces que indica la tabla mencionada anteriormente por cada iteración y al resultado que se obtiene en cada ronda se le aplica la permutación PC-2. Tabla 2 (Ver Anexo 6.2).

De esta forma y por medio de Simulink se generan los bloques correspondientes a cada una de las funciones mencionadas anteriormente, de manera que con las cajas resultantes se pueda

llevar a cabo la prueba para una sola ronda en la cual se realza el cifrado por derecha y el cual está definido por medio de los habilitadores generados para cada ronda de tal manera que si el habilitador es contrario al establecido como derecho, se realizara un cifrado por izquierda y de igual forma en el caso contrario. Teniendo ya esto establecido, se maneja el bloque correspondiente al mensaje de entrada y se realiza la conversión de caracteres a tipo hexadecimal para obtener los 64 bits correspondientes para el desarrollo del algoritmo y así poder aplicar el cifrado; con el resultado obtenido a la salida de esta ronda, se aplica el bloque de descifrado por lo que se tendría nuevamente el texto de entrada con sus 64 bits respectivamente.

El diagrama de bloques de la Figura 3.1, expone con mayor claridad el proceso de cifrado y descifrado que se lleva a cabo para el caso en el que solo se realza una ronda, en el desarrollo del DES Braids. De igual forma se realiza la comparación entre los datos de entrada(los cuales corresponden al texto inicial) y los datos obtenidos al realizar el descifrado, notándose así que tienen el mismo resultado y lo cual se da a entender con el vector de ceros que se encuentra en la parte izquierda de este proceso.

Partiendo principalmente de estos resultados se lleva a cabo la implementación del sistema de cifrado DES Braids trenzado.

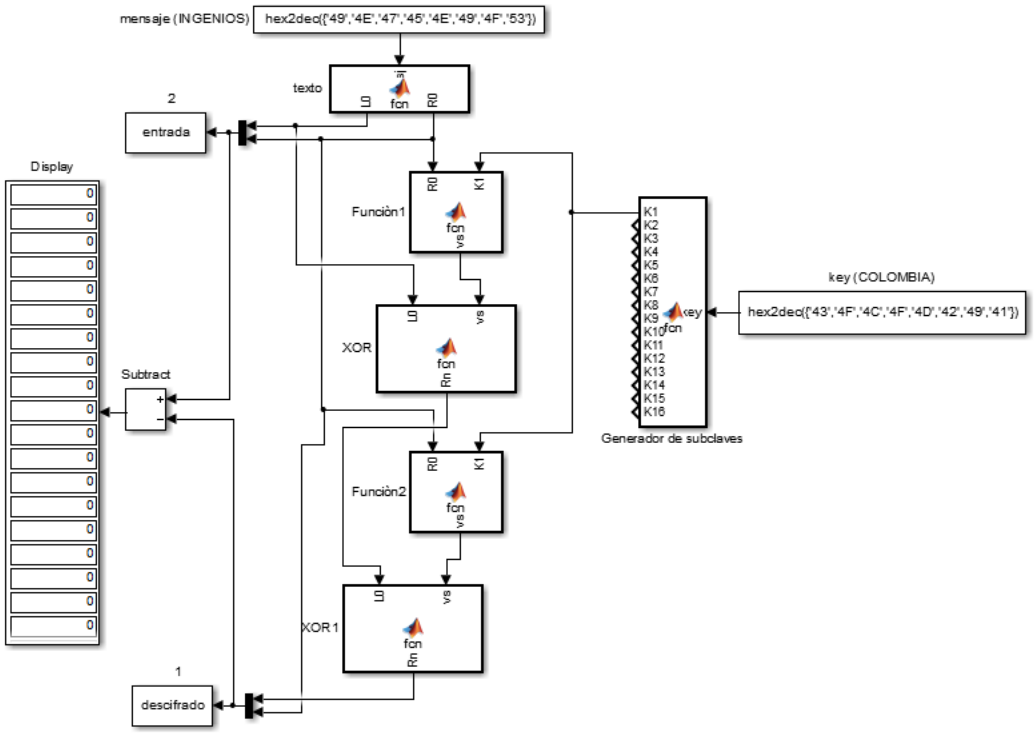

Figura 3.1 : Cifrado y descifrado para una ronda DES Braids (SIMULINK).

Para el cifrado simétrico de textos en claro de 64 bits con sus respectivas 16 iteraciones, serán tenidos en cuenta los habilitadores en cada ronda, ya que según el valor que tengan aleatoriamente el modo de cifrado puede variar de forma izquierda o derecha. En este caso los que tienen el valor binario cero "0" realizan el cifrado por la derecha (R) y los que toman el valor uno "1" realizan el proceso por la parte izquierda (parte L). Debido a que este algoritmo solo utiliza dos secciones o hebras de 32 bits para el trenzado, solo se llevan a cabo este par de cruces como se muestra a continuación:

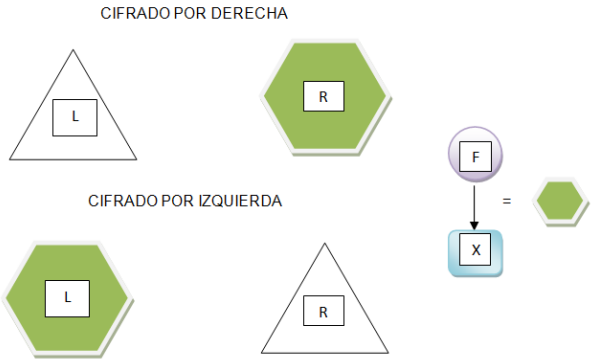

Figura 3.2 : Tipos de cifrado..

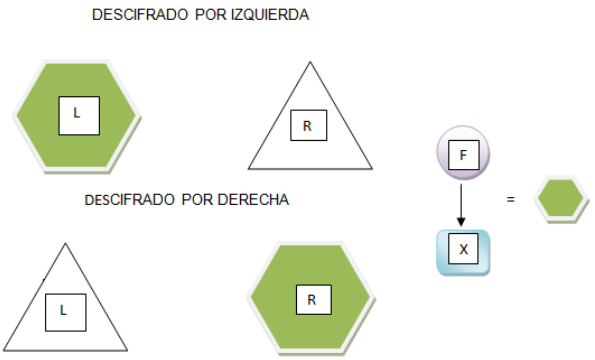

Figura 3.3 : Tipos de descifrado.

En la Figura 3.2 Y 3.3 se pueden notar los tipos de cifrado que pueden ser llevados a cabo por medio de este par de hebras, y también los descifrados, que secuencialmente serian el inverso o complemento en cada caso posible.

Para la implementación de los habilitadores que permitirán la variación que corresponda en cada ronda, se diseño un bloque auxiliar, en el cual se genero un pseudocódigo que da como resultado 16 bits que serán adicionados al tamaño de la clave general, por lo que se obtendrá un valor neto de 64 bits (teniendo claro que la clave utilizada en el DES convencional son 48) y de esta forma poder aplicar cada uno de estos por ronda en el proceso del sistema DES Braids. (Ver Anexo 6.3).

3.1. DESARROLLO E IMPLEMENTACIÓN

Teniendo en cuenta lo mencionado anteriormente, los habilitadores cumplen con la tarea de controlar el sentido de los cruces y de esta misma forma orientar a la función F y a la XOR los grupos de datos correspondientes a la orden dada por el valor de cada habilitador.

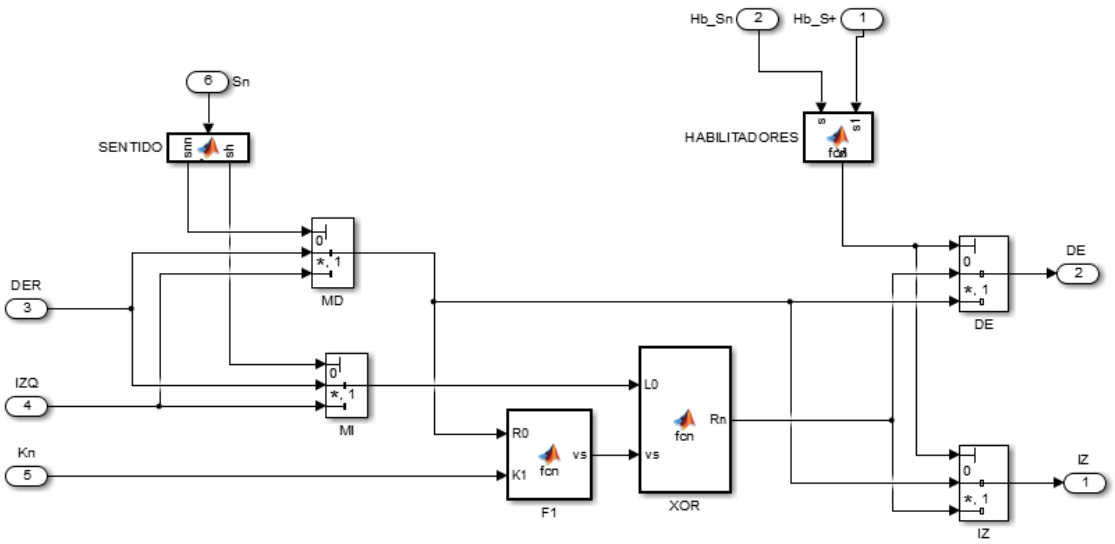

Figura 3.4 : Esquema general en Simulink.

El habilitador correspondiente a la ronda n entra a la función (SENTIDOS en la Figura 3.4), de tal forma que dará la dirección correcta de acuerdo al implementación realizada por el diseñador; las salidas de esta función se conectan a dos multiplexores, a los cuales entra un valor contrario al otro (1 o 0) y de esta manera se habilitan los vectores (Ln y Rn), enlazándose posteriormente con la función F (F1) y la operación XOR. La salida de esta operación, al igual que los datos que provienen del multiplexor (MD), serán las entradas que se dirigen a los dos multiplexores (DE e IZ) respectivamente y los cuales habilitaran el orden de las salidas de acuerdo al valor dado por la función (HABILITADORES), la cual es la encargada de realizar esta comparación entre el habilitador de la ronda n y n+1 para así asegurar el sentido que deben tomar estas para la continuidad del algoritmo. Del esquema desarrollado en la Figura 3.4 se genera un bloque que integra todo este proceso dejándole como entradas los habilitadores de la ronda actual y la siguiente, la subclave determinada para cada etapa y los vectores del texto en claro o cifrado.

4. RESULTADOS

Realizando este desarrollo por medio de MATLAB (Simulink), se puede comprobar el correcto funcionamiento de la implementación dada como DES Braids. De tal forma que al llevar a cabo esta prueba con un pequeño número de rondas de cifrado y descifrado, el resultado final será igual a los datos de entrada correspondientes al texto en claro utilizado para este proceso. Por lo que al realizar más o el total de las 16 rondas el resultado es el mismo.

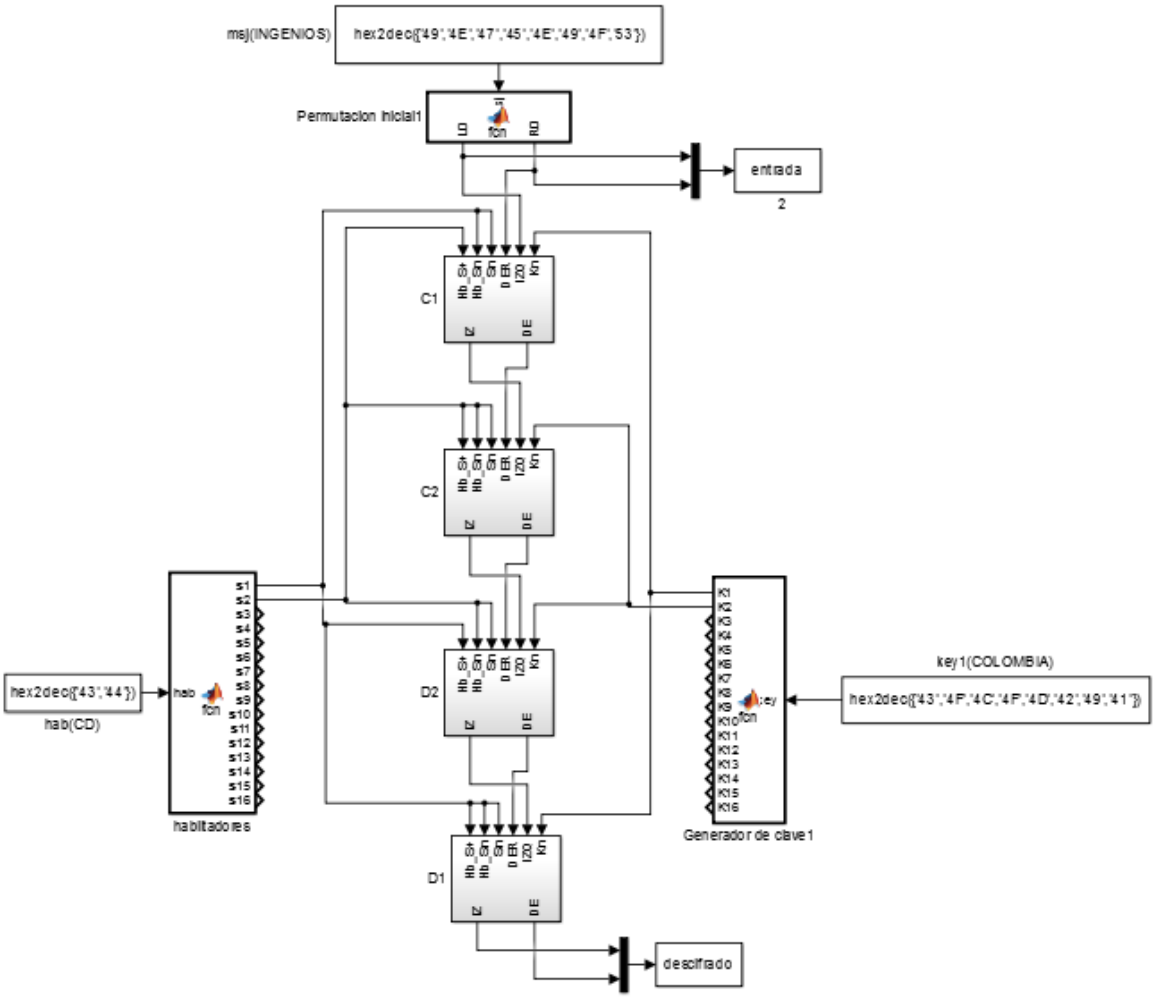

Figura 4.1 : DES Braids para 2 rondas.

Los bloques que cumplen con el proceso de cifrado (C1 y C2 en la Figura 4.1), cumplen tareas

diferentes con respecto a los bloques de descifrado (D2 y D1); es así que este último grupo de bloques se identificaría como el inverso o complemento del cifrado. De esta forma el bloque de descifrado utiliza los habilitadores de manera opuesta a los manejados en la parte de cifrado, por ejemplo, si el habilitador de cifrado en la ronda 1 es "1" la correspondiente al descifrado de esta ronda es "0" y así sucesivamente con cada una de las rondas implementadas en este tipo de diseño. Del mismo modo este proceso se desarrolla en forma descendente y por el contrario el cifrado se realiza en forma ascendente durante sus 16 etapas como el DES convencional.

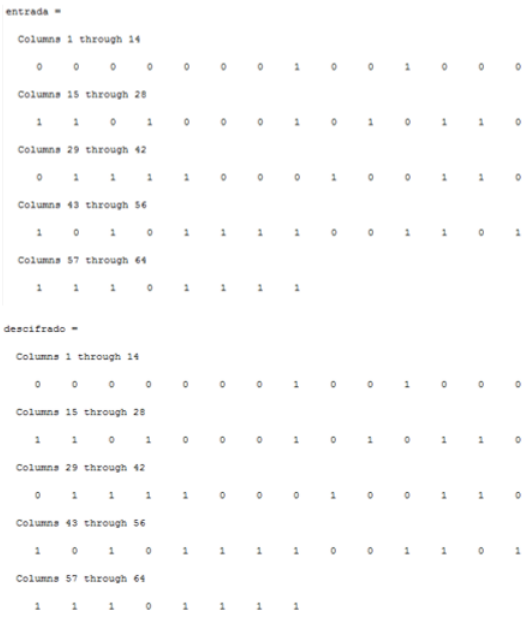

Figura 4.2 : Resultados de simulación.

Figura 4.3 : Diferencia en los datos obtenidos.

En la Figura 4.2 se pueden notar los vectores correspondientes a la parte de entrada o texto en claro y a los de la parte de descifrado en sistema binario, por lo que como se menciono anteriormente la diferencia entre estos da como resultados vector de ceros, Figura 4.3, reflejando así la igualdad entre ellos y cumpliendo con los direccionamientos establecidos por cada habilitador.

5. CONCLUSIONES

En este trabajo se ha presentado la implementación de un sistema de cifrado, diferente al DES convencional, en el cual se hace uso de grupos matemáticos no abelianos como lo son los grupos de trenzas, permitiendo así llevar a cabo nuevas técnicas de criptografía como fundamento para brindar una mayor seguridad en la transmisión de datos, al mismo tiempo que se experimenta el desarrollo tecnológico en la actualidad.

Para hacer más complejos los algoritmos de cifrado no es necesario aumentar la cantidad de bits utilizados en la clave, sino que por el contrario esto puede ser desarrollado al tener en cuenta sistemas con estructuras matemáticas más complejas de las que se han llegado a utilizar y aplicar hasta el momento.

De igual forma al hacer uso de este algoritmo, se pueden representar otros sistemas de cifrado conocidos, con el fin de reconocer características y funciones que puedan servir como guía en el diseño de nuevos sistemas de seguridad informática.

Basados en la implementación desarrollada por medio de MATLAB, se puede concluir:

En el momento de aumentar el tamaño de la clave utilizada para este sistema con, los 16 bits correspondientes a los habilitadores se cumple con el objetivo de llevar a cabo la complejidad de este tipo criptografía haciendo asi menos eficiente los ataques de agentes externos por medo de criptoanálisis.

Debido a que a cada ronda requiere de un habilitador, por medio de los grupos de trenzas se pueden llevar a cabo 16 tipos de combinaciones correspondientes a los modos de cifrado derecha e izquierda, que por el contrario del DES convencional solo puede llevar a cabo este proceso por medio de un solo tipo de cifrado (cifrado por derecha) y lo cual lo hace más vulnerable a perder la información.

6. ANEXOS

A continuación se encuentran los pseudocódigos desarrollados en la implementación del sistema de cifrado DES Braids (DES trenzado). De tal forma que se llevan a cabo los usos de las tablas de permutación necesarias para este tipo de diseño.

6.1. FUNCIÓN F

Para este tipo de función se manejaron como entradas la variable (Ro) y la cual corresponde a los datos del mensaje identificados como los 32 bits de la parte derecha y la otra variable (K1), como la subclave a utilizar en esta ronda. Es así como se aplica la tabla de permutación E y de igual forma se realiza la operación correspondiente a la XOR con la subclave mencionada anteriormente y el resultado final pasa a trabes de las cajas de permutación S.

```
function vs = fcn(R0,K1)
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;
```

TABLAS S

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];

Vectores organizados de las tablas 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
```

Proceso final resultante de realizar la concatenación de dichos vectores.

```
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. SUBCLAVES

El desarrollo de esta función, se lleva a cabo por medio de la clave de entrada determinada por el diseñador, de tal forma que, al cumplir con el debido proceso de las tablas de permutación y los corrimientos específicos en cada ronda, se pueden tener a la salida las 16 subclaves diferentes para el desarrollo del cifrado.

```
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';
```

Tabla de permutacion PC-1.

```
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)';
```

Proceso de corrimientos por ronda y tabla de permutaciòn PC-2.

```
K1F=[ic dc];

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
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(14);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
```

6.3. HABILITADORES

Esta función tiene como entrada los caracteres determinados para generar los valores de datos aleatorios según corresponda para cada tipo de cifrado que se desee desarrollar, de tal forma que en este bloque se realiza la conversión para que no se exceda ni hagan falta la cantidad de habilitadores requeridos para la efectividad del algoritmo.

```
function [s1,s2,s3,s4,s5,s6,s7,s8,s9,s10,s11,s12,s13,s14,s15,s16] = fcn(hab)

h=dec2bin(hab,8); h=reshape(h',1,16); h=h-'0';

h=[h(1);h(2);h(3);h(4);h(5);h(6);h(7);h(8);h(9);h(10);h(11);h(12);h(13);h(14) h(15);h(16)]';

s1=h(1);
s2=h(2);
s3=h(3);
s4=h(4);
s5=h(5);
s6=h(6);
s7=h(7);
s8=h(8);
s9=h(9);
s10=h(10);
s11=h(11);
s12=h(12);
s13=h(13);
s14=h(14);
s15=h(15);
s16=h(16);


end
```

REFERENCES

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

[2] Schneier B. *Applied Cryptography Second Edition: Protocols, algorithms, and source code in C.*

[3] J. Buchmann. *Introduction to cryptography.* Springer-Verlag, New York, 2001.

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

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

[6] Eduardo Ruíz Duarte. *Algebraic geometry logic, GRUPOS DE TRENZAS.* http://b3ck.blogspot.com.co/2014/07/grupos-de-trenzas.html, 2014.

[7] Artin E. *Annals of Mathematics.* 1947.

[8] Fernando Espinosa. *seguridad en redes.* http://djferpersempre.blogspot.com.co/2013/03/cifrado-feistel.html, 2013.

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

[10] Cutberto Hernández Mendoza Lourdes Acuayte Alcántara. *cifrado feistel.* http://cifradofeistel.blogspot.com.co/, 2012.

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

[12] Paulo Sergio García Méndez. *descripcion polinomial de los sistemas de cifrado DES y AES.* http://mat.izt.uam.mx/mcmai/documentos/tesis/Gen.05-O/Garcia-MPS-Tesis.pdf, 2011.

[13] WinGuard Pro. *criptografia de llave publica.* https://seguridadinformatica6851.wikispaces.com/CRIPTOGRAFIA+DE+LLAVE+PUBLICA, 2005.

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

[15] textos cientificos. *algoritmos de calve simetrica.* https://www.textoscientificos.com/criptografia/privada, 2005.

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