CAPÍTULO II

El objetivo de este capítulo es describir brevemente algunos criptosistemas simétricos que serán de utilidad más adelante. Se dice que un criptosistema es simétrico si con la clave que se cifra es la misma con la que se descifra. Nosotros no haremos una exposición de algunos criptosistemas clásicos, como por ejemplo: Vigenere, Hill, Permutación etc. [1]. Se inicia nuestra discusión con el primer sistema simétrico de cifrado, que fue la norma internacional de cifrado de datos, de acuerdo con la NIST. El nombre de este algoritmo de encriptación es: “Data Encryption Standard” ─ DES. Este criptosistema lo propuso la Corporación IBM en 1974 [2]. De hecho, este sistema de encriptación fue inspirado por el algoritmo de cifrado de nombre “LUCIFER”.

Por otro parte, se menciona que todo criptosistema consta de cinco elementos, a saber: el primero es el conjunto de los textos planos, cuyos elementos son todos los mensajes que pueden ser cifrados. Por ejemplo, este conjunto pudiera estar formado por todos los mensajes que se escriben en español en una cadena de 19 símbolos. Este conjunto se denota como 𝒫.

El segundo es el conjunto de textos cifrados, esto es, el conjunto cuyos elementos son los textos encriptados. Este conjunto se denota como . En este trabajo, se utiliza como alfabeto para los textos planos al código ASCII y para los textos encriptados el sistema hexadecimal. También, es necesario precisar que la simbología para representar cadenas (concatenación de ceros y unos de longitud finita) se utilizan letras mayúsculas como:

El tercer elemento es el conjunto de claves o de llaves para cifrar los mensajes. En la Criptografía moderna, las llaves de los criptosistemas simétricos son cadenas de ceros o unos de longitud finita, por ejemplo, 64 bits. Este conjunto se denota como . En todo sistema de cifrado simétrico el número de elementos de este conjunto es muy importante, porque al menos los criptosistemas propuestos deben resistir el ataque de fuerza bruta.

El cuarto y quinto elemento son dos funciones uno a uno. La primera de estas funciones cifra la información, y se denota como ; en este caso es el mensaje o texto claro, y es el texto cifrado. La segunda función descifra el mensaje ; lo cual se denota como: .

**II.1**. **DES.**

A continuación se inicia con una descripción de alto nivel (a grandes rasgos) del criptosistema DES, y posteriormente se da una explicación un poco más detallada. Además, se señala que el algoritmo de encriptación consta a su vez de 2 algoritmos; uno de ellos realiza el proceso de cifrado y el otro genera las llaves ronda. Por otro lado, si el lector desea profundizar aún más en el tema, se recomienda leer la norma FIPS PUB 46-3. El algoritmo de encriptación procede de acuerdo con los siguientes pasos:

1. Dada una cadena de texto claro, la cual es de 64 bits; se calcula otra aplicando una permutación inicial fija “”. Esto último, se escribe de la siguiente manera:. La cadena a su vez, se divide en 2 subcadenas de 32 cada una las cuales se denotan como: ; lo que significa que son los primeros 32 bits, y son los 32 bits restantes.
2. Se aplican 16 iteraciones o rondas y en cada una de ellas se calculan de acuerdo con las siguientes expresiones:

(2.1)

(2.2) Para 1, 2, , 16

La permutación fija se muestra en la tabla 2.1.1.

Tabla 2.1.1. Permutación inicial .

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 |

La función, se describirá más abajo y las cadenas son las llaves ronda de 48 bits. Estas cadenas se obtienen de acuerdo con un algoritmo; de hecho, éste se presenta en la siguiente sección.

1. Después de la ronda 16 se invierten los bloques, esto es, se ordenan como:, y posteriormente se aplica la permutación inversa a la cadena para obtener el texto cifrado . La tabla se presenta en la Tabla 2.2.

Como fue señalado en el inciso b la función f tiene 2 argumentos; i.e. y ; donde ambas son cadenas de bits de diferente longitud, esto es, es de 32 bits y es de 48 bits. Por otro lado, da como resultado una cadena de 32 bits. Ahora bien, todo el proceso que realiza la función f, se lleva a cabo mediante los siguientes pasos:

1. El primer argumento es expandido, dando como resultado una cadena de 48 bits. Este aumento se ejecuta de acuerdo con una la función de expansión “E” que se presenta en la Tabla 2.1.3.

Tabla 2.1.2. Permutación inversa de .

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 40 | 8 | 48 | 16 | 56 | 24 | 64 | 32 |
| 39 | 7 | 47 | 15 | 55 | 23 | 63 | 31 |
| 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 2.1.3. Función de expansión

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | 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 |

1. Se calcula E y el resultado se observa como la concatenación de 8 subcadenas, de 6 bits cada una, las cuales se denotan como: .
2. En este paso se introduce un nuevo elemento que se denomina “cajas de substitución”. De hecho, se tienen 8 cajas de substitución, las cuales son usadas en cada una de las subcadenas , con 1. El proceso de substituir a una cadena de entrada , por otra de salida se denota como ; la mecánica de substitución es como sigue: digamos que la subcadena bits; entonces, los bits nos indican el número de renglón de la caja, y los bits nos señalan la columna.

El resultado es una cadena de 4 bits, esto es, . Tanto los renglones como las columnas corresponden a una tabla de 4 x 16. A continuación se muestra solamente la primera tabla de substitución, o sea.

Tabla 2.1.4. Primera tabla de substitución

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 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 |

1. Queda claro del inciso anterior, que la cadena de salida está dada por 8 bloques de 4 bits cada una, o sea, 32 bits en total. Posteriormente, a la cadena de salida se aplica la permutación “P”, lo cual se escribe como P(). Ahora bien, el resultado P() es lo que se define como . La permutación “P” se muestra en la Tabla 2.1.5.

Tabla 2.1.5. La permutación P

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | 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 |

Finalmente, es conveniente ilustrar los pasos de manera gráfica. La figura 2.1 muestra cómo se realizan estos pasos.

Resumiendo, para obtener los bloques se aplica de la permutación IP, con lo cual se inicia el procedimiento de cifrado. Además, que después de la ronda 16 se invierten los bloques izquierdo y derecho, esto es, se tiene la siguiente cadena:. También, que el paso final del proceso de cifrado emplea la permutación PI-1.

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
|  |  |

Figura 2.1.1. La *i*-ésima ronda del proceso de cifrado.

Ahora bien, queda pendiente describir el algoritmo para la generación del programa de llaves o las llaves ronda; en otras palabras, cómo se obtienen las , que intervienen en cada una de las 16 rondas, con 1, 2, , 16.

Se parte de una cadena de 64 bits, la cual es la llave y se denota como . A la llave se le aplica la permutación PC-1, esta permutación elimina 8 bits. De acuerdo a la norma, las posiciones de los bits que se eliminan son: 8, 16, 24, 32, 40, 48, 56 y 64. A estos bits se les denomina “bits de paridad”. PC-1 se presenta en la Tabla 2.1.6.

Ahora bien, para obtener las 16 llaves ronda se llevan a cabo los siguientes pasos:

1. Después de aplicar la permutación PC-1 a la llave , el resultado se escribe como: PC-1 donde son los primeros 28 bits y son los 28 bits restantes.
2. Para se evalúan las siguientes expresiones:

. Representa un corrimiento de una o dos posiciones a la izquierda dependiendo del valor de . La norma establece que se corre una posición cuando ó 16 y se corren dos posiciones a la izquierda de otra manera.

Tabla. 2.1.6. La permutación PC-1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  | P C - 1 |  |  |  |
| 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 |

La permutación PC-2 elimina otros 8 bits, quedando una cadena de solamente 48 bits. La Tabla 2.1.7 muestra la permutación PC-2, y para ilustrar el proceso de cómo se generan las llaves del programa que intervienen en cada ronda, se presenta la Figura 2.1.2.

Tabla 2.1.7. La permutación PC-2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | PC | - 2 |  |  |
| 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 |

Con relación a la complejidad del criptosistema DES podemos decir que es de 256; lo anterior significa que ante un ataque de fuerza bruta se tiene que probar a lo sumo 256 llaves.

PC-1

PC-2

PC-2

Figura 2.1.2. Generación del programa de llaves.

**II.2.1 Triple DES**.

Continuaremos con la presentación del algoritmo Triple DES, que, como su nombre lo indica es una variante del algoritmo DES. De hecho, son 3 ciclos DES.

El algoritmo de cifrado de Triple DES se divide en 2 partes; la primera consiste en el encriptado de la información, y la segunda, la generación de las llaves ronda; por otra parte, los textos plano y cifrado son de 64 bits, de igual forma que en DES.

Con relación al proceso de cifrado de la información mencionaremos de manera general que el procedimiento de encriptación tiene 3 pasos, a saber: en el primero se cifra el texto plano de acuerdo con el algoritmo DES y una llave de 64 bits . En el segundo paso, se descifra el texto encriptado en el primer ciclo conforme a la llave . Ahora bien, como el resultado no es el texto plano inicial, sino otra cadena de 64 bits.

En el tercer paso, se toma como texto plano al texto descifrado del ciclo anterior y se realiza el procedimiento de encriptado. El resultado de este ciclo de cifrado es finalmente el texto encriptado. Este procedimiento se lleva a cabo con una llave .

Algunos comentarios al respecto.- Si denotamos un paso o ciclo de cifrado como: y al ciclo de descifrado como ; donde es el texto plano y es el texto encriptado. Entonces, Triple DES procede de la siguiente forma:

Por otro lado, hay dos variantes en la encriptación de mensajes, a saber:

a) Cuando y

b) Cuando las son todas diferentes, para =1, 2, 3.

En el primer caso se tiene una complejidad de 2112 para solucionar el problema; y en el segundo esquema se tiene una complejidad de 2168.

Es importante mencionar que el algoritmo Triple DES está vigente; esto es, hay empresas o instituciones que aún lo usan.

**II.2.2 Triple-DES-96.**

Los autores descubrieron una variante para el algoritmo Triple DES. Esta variante tiene la ventaja de ser más rápida y tener las mismas complejidades para resolver el algoritmo, claro está, para ataques de fuerza bruta. El nombre de este algoritmo es Triple-DES-96 [3].

Ahora bien, se inicia la discusión dando respuesta a la siguiente pregunta: ¿cómo se define el algoritmo Triple-DES-96? Para lo cual es suficiente con la descripción de un ciclo; esto es, mostrar el algoritmo DES-96. Digamos que la relación entre DES y Triple-DES, es la misma que hay entre DES-96 y Triple- DES-96. En este orden de ideas, denotemos al ciclo de cifrado DES-96 como sigue: , donde es el texto plano de 96 bits y es la llave de 64 bits. A continuación se hará una descripción de la ronda , donde = 1,2..., 16. La función de DES-96 es diferente de la función de DES. La función realiza los siguientes pasos:

1.- La cadena de entrada a la -ésima ronda, con = 1, 2, 16, tiene una longitud de 96 bits. Esta cadena de entrada se divide en 2 sub-bloques, ; donde cada una de ellas es de 48 bits de longitud. Se ejecuta la operación x-or ⊕ = Input. La cadena es la -ésima llave del programa de llaves que también tiene una longitud de 48 bits.

2.- La cadena Input de 48 bits, es la entrada a las cajas (8 en total) en donde se sigue la regla de substitución de DES. La salida de las cajas es una cadena de 32 bits, llamémosla .

3.- La salida de las cajas es una cadena de 32 bits, o sea , y debiera aplicase la permutación P a para que dé como resultado una cadena de 32 bits de longitud [4]; sin embargo, se requiere una cadena de 48 bits para realizar la operación x-or con la cadena izquierda anterior.

Por esta razón, en esta investigación se propone emplear a la permutación E que usa la norma [5]. Resumiendo, en este trabajo se elimina a la permutación P y en su lugar se utiliza a la permutación E; esto es, E. Entonces, el resultado de la función es E().

Además, de acuerdo con la norma internacional el proceso de encriptación de Triple DES inicia con una permutación fija sobre un bloque de 64 bits, sin embargo, en este libro se propone aplicar una permutación variable en la entrada de la tercera ronda del primer ciclo y su inversa al inicio de la décimo quinta ronda del tercer ciclo del proceso de cifrado.

El término de permutación variable, significa que es desconocida para un posible opositor y no como el caso de Triple DES, que es fija y conocida.

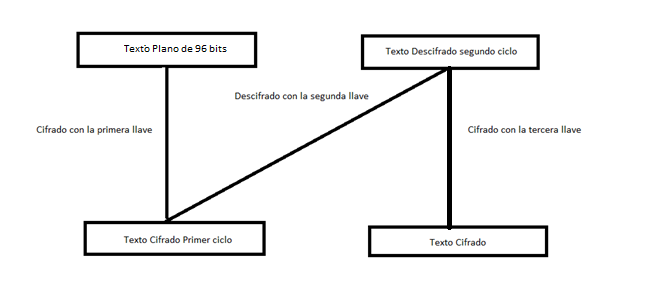


Figura 2.2.2.1 Diagrama de cifrado del criptosistema Triple-DES-96

Las ventajas que tiene el criptosistema propuesto con relación a Triple DES son las siguientes: primera, el criptosistema Triple-DES-96 cifra bloques de 96 bits y segunda; el criptosistema propuesto elimina una permutación del criptosistema Triple-DES.

Teniendo en cuenta los 2 aspectos anteriores, da como resultado una mayor rapidez de cifrado cuando se utiliza Triple-DES-96 que cuando se emplea Triple-DES, de hecho, más del doble.

También, la complejidad se ve beneficiada debido a que si se desea descifrar un texto plano con una permutación diferente a la usada en el cifrado, manteniendo fijas las llaves de 64 bits, dos o tres de ellas según sea el caso, no se obtendrá el texto plano original [6].

En este libro no se presentará una prueba formal del aumento de la complejidad ante un ataque de fuerza bruta. En la figura 2.2.2.1 se presenta un gráfico que sintetiza la manera de encriptar de Triple-DES-96. En este punto se considera conveniente presentar un ejemplo de una permutación de 96 elementos, digamos que es el equivalente a la permutación . Claro está, de acuerdo con las consideraciones hechas anteriormente. Por otro lado, no olvidemos que en este caso hay 96! posibles permutaciones. En las tablas 2.2.2.1 y 2.2.2.2 se ilustran dos casos particulares de y .

Tabla 2.2.2.1 Permutación de 96 elementos.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 86 | 74 | 62 | 50 | 38 | 26 | 14 | 2 |
| 88 | 76 | 64 | 52 | 40 | 28 | 16 | 4 |
| 90 | 78 | 66 | 54 | 42 | 30 | 18 | 6 |
| 92 | 80 | 68 | 56 | 44 | 32 | 20 | 8 |
| 94 | 82 | 70 | 58 | 46 | 34 | 22 | 10 |
| 96 | 84 | 72 | 60 | 48 | 36 | 24 | 12 |
| 85 | 73 | 61 | 49 | 37 | 25 | 13 | 1 |
| 87 | 75 | 63 | 51 | 39 | 27 | 15 | 3 |
| 89 | 77 | 65 | 53 | 41 | 29 | 17 | 5 |
| 91 | 79 | 67 | 55 | 43 | 31 | 19 | 7 |
| 93 | 81 | 69 | 57 | 45 | 33 | 21 | 9 |
| 95 | 83 | 71 | 59 | 47 | 35 | 23 | 11 |

**II.3**. **AES.**

El algoritmo que se presenta a continuación es: “Advanced Encryption Standard”, popularmente conocido como AES. La descripción detallada está en la norma internacional FIPS PUB 197 [7].

**II.3.1**. **Preliminares.**

Antes de iniciar nuestra presentación de AES, es conveniente describir las operaciones modulares con polinomios [8]. Muy probablemente, hemos trabajado la operación modular usando los números enteros, afortunadamente, hay una gran similitud entre las operaciones modulares con polinomios y los números enteros.

En la aritmética modular con los números enteros utilizamos a los primos para garantizar la existencia de inversos multiplicativos; de misma forma, en las operaciones modulares con polinomios se emplean los polinomios irreductibles. El significado de polinomio irreductible es similar al de número primo; esto es, un polinomio irreductible es aquel que solamente es divisible entre uno y el mismo.

Tabla 2.2.2.2. Permutación

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 56 | 8 | 64 | 16 | 72 | 24 | 80 | 32 |
| 88 | 40 | 96 | 48 | 55 | 7 | 63 | 15 |
| 71 | 23 | 79 | 31 | 87 | 39 | 95 | 47 |
| 54 | 6 | 62 | 14 | 70 | 22 | 78 | 30 |
| 86 | 38 | 94 | 46 | 53 | 5 | 61 | 13 |
| 69 | 21 | 77 | 29 | 85 | 37 | 93 | 45 |
| 52 | 4 | 60 | 12 | 68 | 20 | 76 | 28 |
| 84 | 36 | 92 | 44 | 51 | 3 | 59 | 11 |
| 67 | 19 | 75 | 27 | 83 | 35 | 91 | 43 |
| 50 | 2 | 58 | 10 | 66 | 18 | 74 | 26 |
| 82 | 34 | 90 | 42 | 49 | 1 | 57 | 9 |
| 65 | 17 | 73 | 25 | 81 | 33 | 89 | 41 |

Cuando hablamos de la operación de división entre polinomios, nos referimos a la misma operación que nos enseñaron en el curso de Álgebra elemental.

Por otro lado, el polinomio irreductible que usa AES como módulo es el siguiente: . Se aclara que los coeficientes de la variable con son: cero o uno. Lo cual quiere decir que los valores que toman los coeficientes de son elementos de . En este orden de ideas, el residuo puede ser un polinomio de grado 7 como máximo y como mínimo el valor 0. Entonces, se tienen 256 posibilidades; esto es, un Byte.

Ahora bien, se inicia definiendo la operación para dos polinomios; y . En este punto se recuerda que: -1 mod. 2 ≡ 1.

Si consideramos como resultado de la suma de los dos polinomios anteriores a: ; entonces, los coeficientes para .

En el caso de la multiplicación de dos polinomios y ; módulo = , se lleva a cabo de la siguiente manera: se realiza la multiplicación de por y el producto se divide entredonde el residuo de la división es el resultado de la operación modular.

Consideramos que es conveniente mostrar un ejemplo.

Ejemplo 2.3.1.1. Dados y , calcular el producto módulo . Entonces es simple verificar lo siguiente:

Además, mod. , puede escribirse como:

+ . Claro está, para escribir la expresión anterior se debe llevar a cabo la división. También, puede observarse que el residuo es;.

Ahora bien, hay una forma más sencilla de realizar la operación modular; i. e., solamente con corrimientos a la izquierda y la operación x-or.

Supongamos que se tiene los mismos dos polinomios citados en el ejemplo, entonces, cada uno de ellos puede expresarse como una cadena de 8 bits, o sea, un Byte. Esto es, es equivalente a 01001011. Como puede observase, cuando aparece la variable elevada a una potencia se escribe uno y en caso contrario es cero. Digamos que hay una correspondencia uno a uno entre polinomios y las cadenas de 8 bits.

Entonces, es equivalente a 10100100 y el producto puede escribirse como sigue:

(2.3.1.1) 10100100 (01000000 00001000 00000010 00000001)

Obsérvese que la cadena 00000001 es la identidad, y por lo tanto, el producto con 10100100 es:

(2.3.1.2) 10100100 00000001 = 10100100.

Ahora bien, el producto 10100100 00000010 es equivalente a multiplicar por , lo que a su vez es similar a realizar un corrimiento a la izquierda de una posición en la cadena 10100100; esto es, 01001000. El símbolo “{}” significa que se llegó a la posición ocho, contando a partir de la posición cero. También, es simple ver que cuando se realiza el corrimiento a la izquierda se agrega un cero a la derecha, de tal manera que después del símbolo aparece una cadena de ocho bits.

Por otro lado, si se llega a la posición ocho se debe realizar la operación x-or con la cadena 00011011 = . A continuación, se ilustra esta operación:

(2.3.1.3) 01001000 00011011 = 01010011

Como se observa, en la expresión anterior se elimina el bit de la posición ocho. Asimismo, si escribimos el resultado en hexadecimal la expresión es como sigue: {5 3}.

Entonces, las multiplicaciones de la expresión 2.3.1.1 que restan son las siguientes:

(2.3.1.4) 10100100 00001000 = 01010111 y

(2.3.1.5) 10100100 01000000 = 10001110

De aquí, si tomamos en cuenta los resultados de las expresiones que van de 2.3.1.2 a 2.3.1.5 se tiene lo siguiente:

10100100 01010011 01010111 10001110 = 00101110

Se puede expresar el producto en el sistema hexadecimal usando la notación de la norma, a continuación se escribe el resultado:

{a 4} ● {4 b} = {2 e}.

Sintetizando, la equivalencia que hay entre cadenas de ocho bits y polinomios nos dice que el primer bit, contando de derecha a izquierda, es la potencia cero y el octavo bit es la potencia 7. En este orden de ideas, la posición que tenga un bit con valor uno indica la potencia de la variable, y la potencia de la variable señala el número de corrimientos.

Entonces, el procedimiento tiene los siguientes pasos:

a) Uno de los factores se expresa como la suma, de cadenas que tienen un solo bit diferente de cero.

b) Dependiendo de la posición del bit diferente de cero es el número de corrimientos; sin embargo, cuando se llega a la posición ocho se debe aplicar la operación x-or con la cadena = 00011011. Si aún faltan corrimientos por hacer, se continúan con ellos después del resultado de la operación x-or; claramente, si se vuelve a llegar a la posición ocho nuevamente se debe hacer la operación x-or con .

c) Cuando se tienen los productos de cada uno de los sumandos, se ejecuta la operación x-or de todos los resultados parciales, para obtener el resultado final.

Una vez que nosotros sabemos multiplicar de la manera que se explicó anteriormente, podremos realizar la operación “MixColumns”.

**II.3.2**. **Descripción de alto nivel.**

Continuaremos nuestra explicación, mencionando que el algoritmo de cifrado AES se subdivide a su vez en dos partes, a saber: La primera de ellas, nos indica la forma en que se mezcla la información, y la segunda, nos describe la forma en que se generan las llaves ronda.

Por otro lado, se señala que hay diferentes algoritmos de cifrado. En este sentido, decimos que hay tres tipos de cifrado, a saber: el que tiene 10 rondas, el de 12 y el de 14 rondas.

De forma general se afirma que los tres algoritmos, comienzan con una cadena de 128 bits de texto plano, esto es, 16 Bytes. Asimismo, se menciona que los tres algoritmos son muy parecidos. Aquí, se describe solamente el algoritmo de 10 rondas; si el lector desea saber cómo son los otros dos se recomienda que consulte la norma FIPS 197.

En todos los casos se inicia con una cadena de entrada, o texto plano, de 128 bits. Esta cadena se presenta en forma matricial; esto es:

2.3.2.1

Cada uno de los es una cadena de ocho bits; donde representan los primeros ocho bits de la cadena de entrada de 128 bits y son los últimos ocho bits de la misma cadena.

Como se observa en la expresión 2.3.2.1, los Bytes de la cadena de entrada se escriben de arriba hacia abajo y de izquierda a derecha. De hecho, cada columna es de 32 bits, o sea, una palabra.

Un estado intermedio en el proceso de cifrado se representa de la siguiente forma:

2.3.2.2

El estado final o texto cifrado se muestra de manera matricial también. A continuación, se presenta en forma matricial un bloque de 128 bits encriptado:

2.3.2.3

En el proceso de encriptación se realizan los siguientes procesos:

1. La operación x-or o también denominada como: AddRoudKey
2. SubBytes
3. ShiftRows
4. MixColumns

La operación del primer punto se lleva a cabo de la siguiente forma: digamos que se tiene un mensaje con bits de longitud; de aquí podemos tener los siguientes dos subcasos:

1. El número es múltiplo de 128
2. El número no es múltiplo de 128

Con relación al primer inciso; simplemente se divide el mensaje en bloques de 128 bits. En este sentido, podemos decir que para cada uno de los bloques de 128 bits se realiza el mismo proceso de cifrado, entonces, sin perder generalidad se toma el bloque de los primeros 128 bits, el cual denotaremos como .

En caso de estar en el segundo inciso se define el siguiente procedimiento: se agrega el número de ceros necesarios a la derecha del mensaje ; de tal manera que el número de bits de la cadena resultante sea múltiplo de 128. Posteriormente, se procede de la misma forma que en el inciso i; esto es, se toma el bloque de los primeros 128 bits y se denota como .

Entonces, la primera operación es ; donde es la primera llave del programa de llaves. Más adelante se presentará la manera de generar las 11 llaves a partir de una llave de 128 bits. Como se mencionó anteriormente, en este trabajo se describe solamente el algoritmo de cifrado de 10 rondas.

En realidad, la razón por la que se generan 11 llaves de 128 bits, es porque en cada ronda se inicia con un x-or utilizando la llave , con y en la última ronda se aplican dos llaves, a saber: al inicio de ésta y al final de la ronda.

Queda claro que la operación x-or es bit a bit, además, el resultado es 0 si los bits tienen el mismo valor, y 1 si el valor de ellos es diferente.

Continuando con el proceso de cifrado, el inciso b corresponde a la operación de substitución.

Para realizar esta operación es necesaria una tabla (s-box de 8 x 8). Esta tabla es una permutación de un arreglo de 256 símbolos hexadecimales.

De hecho, estos símbolos van desde el número 00 hasta el número ff. La tabla opera como una función biyectiva; donde un elemento del dominio está definido como Input = y la imagen es el conjunto de elementos que cumplen con: Output = [9].

La tabla 2.3.2.1 nos muestra esta permutación de 256 elementos.

Ahora bien, la mecánica es la siguiente: dada una cadena de entrada (); los primeros 4 bits nos indican el número del renglón, y los cuatro bits restantes determinan el número de columna. El elemento que está en el cruce del renglón con la columna es la salida o el Output.

Claramente puede observase, que en la Tabla 2.3.2.1 todos los números corresponden a dos símbolos hexadecimales, o sea, 8 bits.

De aquí, lo anterior puede representarse de manera simbólica como sigue:

() : → ()

La rutina ShiftRows realiza el siguiente procedimiento:

Tabla. 2.3.2.1. Tabla de substitución del criptosistema AES-FIPS 197.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
| 0 | 63 | 7c | 77 | 7b | f2 | 6b | 6f | c5 | 30 | 01 | 67 | 2b | fe | d7 | ab | 76 |
| 1 | ca | 82 | c9 | 7d | fa | 59 | 47 | f0 | ad | d4 | a2 | af | 9c | a4 | 72 | c0 |
| 2 | b7 | fd | 93 | 26 | 36 | 3f | f7 | cc | 34 | a5 | e5 | f1 | 71 | d8 | 31 | 15 |
| 3 | 04 | c7 | 23 | c3 | 18 | 96 | 05 | 9a | 07 | 12 | 80 | e2 | eb | 27 | b2 | 75 |
| 4 | 09 | 83 | 2c | 1a | 1b | 6e | 5a | a0 | 52 | 3b | d6 | b3 | 29 | e3 | 2f | 84 |
| 5 | 53 | d1 | 00 | ed | 20 | fc | b1 | 5b | 6a | cb | be | 39 | 4a | 4c | 58 | cf |
| 6 | d0 | ef | aa | fb | 43 | 4d | 33 | 85 | 45 | f9 | 02 | 7f | 50 | 3c | 9f | a8 |
| 7 | 51 | a3 | 40 | 8f | 92 | 9d | 38 | f5 | bc | b6 | da | 21 | 10 | ff | f3 | d2 |
| 8 | cd | 0c | 13 | ec | 5f | 97 | 44 | 17 | c4 | a7 | 7e | 3d | 64 | 5d | 19 | 73 |
| 9 | 60 | 81 | 4f | dc | 22 | 2a | 90 | 88 | 46 | ee | b8 | 14 | de | 5e | 0b | db |
| a | e0 | 32 | 3a | 0a | 49 | 06 | 24 | 5c | c2 | d3 | ac | 62 | 91 | 95 | e4 | 79 |
| b | e7 | c8 | 37 | 6d | 8d | d5 | 4e | a9 | 6c | 56 | f4 | ea | 65 | 7a | ae | 08 |
| c | ba | 78 | 25 | 2e | 1c | a6 | b4 | c6 | e8 | dd | 74 | 1f | 4b | bd | 8b | 8a |
| d | 70 | 3e | b5 | 66 | 48 | 03 | f6 | 0e | 61 | 35 | 57 | b9 | 86 | c1 | 1d | 9e |
| e | e1 | f8 | 98 | 11 | 69 | d9 | 8e | 94 | 9b | 1e | 87 | e9 | ce | 55 | 28 | df |
| f | 8c | a1 | 89 | 0d | bf | e6 | 42 | 68 | 41 | 99 | 2d | 0f | b0 | 54 | bb | 16 |

En la expresión 2.3.2.4 se muestra la manera de hacer corrimientos de Bytes a la izquierda, renglón por renglón. Suponga que estamos en un paso intermedio de cifrado, entonces:

Inicial Final

2.3.2.4

Como puede observase, de la expresión 2.3.1.4 se desprende que el primer renglón no sufre ninguna transformación; el segundo renglón tiene un desplazamiento circular de un Byte a la izquierda. El tercer y cuarto renglón tiene un desplazamiento circular de dos y tres Bytes a la izquierda respectivamente.

La cuarta rutina MixColumns utiliza las columnas como polinomios de grado tres. Lo anterior implica que los coeficientes son Bytes. En este punto es conveniente ilustrar este concepto. Digamos que se tiene una matriz de un paso intermedio como la mostrada en la expresión 2.3.2.2.

Entonces, el polinomio de la primera columna es: { + + + {, queda claro que, de la misma forma se pueden escribir polinomios de grado tres para las demás columnas.

Por otro lado, el módulo que se emplea en esta parte es: {01 + {01}, también, es importante señalar que este polinomio de grado cuatro no es irreductible. Sin embargo, hay polinomios de grado menor que cumplen con que el máximo común divisor de ellos con {01 + {01} es uno.

El algoritmo de encriptación AES utiliza al siguiente polinomio: {03 + + + {02, el cual tiene inverso módulo {01 + {01}. El inverso de este polinomio es: {0b + + + {0e.

La mecánica para llevar a cabo la operación MixColumns es la siguiente:

2.3.2.5 con

El valor se obtiene realizado la operación de multiplicación definida en la subsección **II.3.1**. Esto es:

= ({02} ● {}) ({03} ● {}) {} {}. Se hacen un par de aclaraciones, a saber: El número {01} es la identidad, por lo tanto {01} ● {} = {} y {01} ● {} = {}.

El segundo punto corresponde al cálculo de , y . Queda claro que los cálculos de estos valores se realizan de la misma forma como se llevó a cabo para .

Por otro lado, para conocer los resultados de las multiplicaciones {02} ● {} y {03} ● {} para cualquier valor de {}, es necesario construir una tabla de 256 multiplicaciones en total. Si procedemos de esta forma se gana tiempo. Los autores consideran conveniente presentar un ejemplo para ilustrar este punto.

d4

bf

5d

30

El ejemplo consistirá en ejecutar la multiplicación del primer renglón de la matriz de la izquierda por la matriz columna.

Se deja al lector las multiplicaciones restantes.

Entonces, esta multiplicación se expresa como sigue: ({02} ● {d4}) ({03} ● {bf}) {5d} {30}. Sin embargo, de acuerdo a lo visto en la subsección II.3.1 el factor de la derecha se separa en sumas x-or de cadenas que tienen solamente un uno.

En este orden de ideas, el producto representado arriba es más conveniente realizarlo poniendo el factor {02} del lado derecho de {d4}.

Además, es adecuado expresar los factores de manera binaria.

Dicho lo anterior, los diferentes sumandos se escriben de la siguiente manera:

{1101 0100} ● {0000 0010} mod. = {1011 0011}

{1011 1111} ● ({0000 0010} {0000 0001}) mod. = {1101 1010}

{0101 1101} ● {0000 0001} mod. = {0101 1101}

{0011 0000} ● {0000 0001} mod. = {0011 0000}

Entonces, llevando a cabo la suma (x-or) de los cuatro resultados anteriores, esto es:

{1011 0011} {1101 1010} {0101 1101} {0011 0000} = {0000 0100}

Ahora bien, si lo escribimos de manera resumida la expresión es la siguiente:

({02} ● {d4}) ({03} ● {bf}) {5d} {30} = {04}.

**II.3.3**. **Algoritmo de cifrado.**

A continuación mostraremos en seudocódigo el algoritmo para cifrar bloques de 128 bits con llaves de 128 bits en 10 rondas. Se señala que dicho algoritmo se obtuvo de la norma FIPS 197.

Inicio.

Input es la cadena de entrada de 128 bits denominada texto plano.

AddRoundKey

Input . // Más adelante se explica cómo se obtiene el programa de llaves.

= 1.

For = 1 hasta 9

SubBytes

Shif**t**Rows

MixColumns

AddRoundKey

End For

= 10

SubBytes

ShiftRows

AddRoundKey

Out. // Cadena de salida de 128 bits.

End

Si el lector desea profundizar un poco más sobre el tema puede recurrir a la norma FIPS 197. También, se recomienda que el estudiante programe el algoritmo en el lenguaje de su preferencia. Puede tomar como guía el anexo de la norma citada anteriormente, que corresponda a la llave de 128 bits.

Asimismo, es conveniente mencionar que hay otros dos tamaños de llaves, a saber: de 192 y 256 bits. Sin embargo, los algoritmos de encriptación son similares.

En cuanto al algoritmo de descifrado se señala que la s-box inversa de la Tabla se puede ver en la norma FIPS 197; así también, la matriz inversa de la expresión 2.4.2.5 puede consultase en la citada norma.

**II.3.4**. **Algoritmo del programa de llaves.**

Es esta parte se muestra la forma de generar el programa de llaves o llaves ronda. Se aclara que solamente se ilustrará el caso particular de una llave de 128 bits. Por otro lado, se menciona que en la elaboración del programa de llaves utilizando cadenas de 192 o 256 bits el proceso es similar.

Como fue señalado anteriormente cada palabra es una columna de la matriz de estados, por lo que una palabra es una cadena de 32 bits. Se trabaja con una cadena, , de 128 bits; o sea, cuatro palabras de 32 bits. De hecho, se toma a como la primera llave ronda.

Antes de presentar el algoritmo que genera las llaves, se hacen algunas observaciones:

Como se sabe se desarrollan 11 llaves de 128 bits cada una; sin embargo, cada una de ellas se divide en cuatro palabras de 32 bits, lo que significa que se generarán 44 palabras para obtener las 11 llaves.

La notación que se utiliza para denominar la palabra es: “Word” o también w[]. Por otro lado, la notación de cada Byte en una palabra es: “key[]” donde = 0, 1, 2, 3.

A continuación, se describen las instrucciones que intervienen en el algoritmo de la generación de llaves:

SubWord. Esta instrucción significa que cada Byte de una palabra es substituido por otro Byte utilizando la tabla de substitución 2.3.2.1. La regla de substitución es la misma que se describió antes.

RotWord. Este procedimiento consiste en llevar a cabo un corrimiento circular a la izquierda de un Byte, en una palabra. Mostraremos un pequeño ejemplo para ilustrar el punto:

Suponga que se tiene la siguiente palabra: 09 cf 4f 3c, observe que cada símbolo es una representación numérica en el sistema hexadecimal. Entonces, un corrimiento circular a la izquierda de un Byte nos da el siguiente resultado: cf 4f 3c 09.

Rcon. Es una constante que toma diferentes valores, dependiendo del número de iteración, en la tabla 2.4.4.1 se muestran estos valores.

Tabla 2.4.4.1. Valores de las constantes Rcon.

|  |  |
| --- | --- |
| Iteraciones | Valores |
| 4 – 7 | 01 00 00 00 |
| 8 -11 | 02 00 00 00 |
| 12 – 15 | 04 00 00 00 |
| 16 -19 | 08 00 00 00 |
| 20 – 23 | 10 00 00 00 |
| 24 – 27 | 20 00 00 00 |
| 28 – 31 | 40 00 00 00 |
| 32 – 35 | 80 00 00 00 |
| 36 – 39 | 1b 00 00 00 |
| 40 - 43 | 36 00 00 00 |

También, el algoritmo emplea dos variables, las cuales se escriben como: “temp” y “word”.

Una vez hechas todas las aclaraciones anteriores, en este punto estamos preparados para escribir el algoritmo.

Inicio.

temp, word

0

while ( < 4 )

w[] = word ( key [ 4 ], key [4 ], key [4 ], key [4 + 3] )

+ 1

end while

= 4

while ( < 44 )

temp w [ – 1]

if ( mod. 4 0 )

temp = SubWord ( RotWord ( temp ) ) Rcon [ ]

end if

w [ ] w [ – 4 ] temp

+ 1

end while

end.

Con relación a la generación del programa de llaves cuando la llave es de 192 o 256 bits, podemos decir que el cálculo de las llaves ronda es similar al de 128 bits.

**II.3.5**. **Modos de operación.**

En esta parte se verán solo dos modos de operación, a saber: ECB y CBC.

**II.3.5.1**. **Modo ECB.**

Este modo de operación simplemente aplica un ciclo AES a cada bloque de 128 bits. A favor de este procedimiento se puede decir que es paralelizable, lo cual hace que sea rápido el cifrado.

En contra de él, está el hecho de que no es bueno para cifrar imágenes, ya que, en caso de figuras de dos colores, como por ejemplo blanco y negro, el cifrado dejaría siluetas que pueden dar información.

**II.3.5.2**. **Modo CBC.**

En el modo CBC se parte de una cadena inicial de 128 bits. Esta cadena inicial puede elegirse de manera aleatoria. Entonces, el primer bloque de texto plano de 128 bits, decimos , se le aplica la operación x-or con ; esto es, . De aquí en adelante, para realizar el procedimiento de cifrado se inicia con la operación x-or del texto cifrado anterior, y el bloque de texto plano a cifrar . En otras palabras, se realiza el cálculo antes de iniciar el ciclo de encriptación.

Lo anterior se escribe de la siguiente manera: . Donde es la clave con la que se generan las llaves ronda y el subíndice cumple con; 1.

De acuerdo con lo anterior el procedimiento de descifrado es el siguiente: con 1.

El procedimiento anterior puede servirnos como técnica de autenticación, ya que, si hay un cambio en alguno de los valores de la cadena inicial , el receptor del mensaje no podrá descifrar el mensaje. Más adelante se verá cómo puede enviarse la cadena al receptor por medio de un criptosistema asimétrico. Se aclara, que el remitente elige al azar una diferente para el envío de un nuevo mensaje.

En este sentido podemos decir que la cadena nos dice quién envió el mensaje.

La única debilidad de este procedimiento es que la cadena es de longitud muy corta, solamente de 128 bits. Existen otros procedimientos de autenticación (firma digital) cuyas cadenas llegan a ser de 1024 bits, con objeto de darle mayor seguridad al procedimiento.

**II.4. Substitution Permutation Network.**

En esta parte definiremos un esquema general para criptosistemas simétricos construidos a partir de tres cosas, a saber:

1. La definición de un algoritmo iterativo para obtener un programa de llaves, llamadas llaves ronda, las cuales son aplicadas en cada paso de un procedimiento.
2. La construcción de una permutación que denotamos como . En realidad esta permutación es una caja de substitución de elementos. En el caso de Triple DES las cajas de substitución son permutaciones de 16 elementos, y cuando es AES la caja de substitución es de 256 elementos. Queda claro, que nosotros podemos proponer una caja de substitución diferente a la de AES con propiedades de no-linealidad similares.
3. La elaboración de una permutación la cual consiste en alterar el orden de los elementos del bloque a ser cifrado. Por ejemplo, en el caso de AES se hablaría de una permutación de 128 elementos. Sin embargo, se pueden construir permutaciones en arreglos mucho más grandes, de cientos de miles de elementos. Más adelante se verá una técnica para construir permutaciones sobre arreglos muy grandes—cientos de miles.
4. La definición de una función que describe un proceso iterativo llamado rondas. En este procedimiento hay rondas. Para un caso particular de AES son 10 rondas. Digamos que en general esta función aplica de manera iterativa; Permutaciones, Substituciones y operaciones x-or.

Un criptosistema simétrico que se ajusta a la descripción anterior se le llama “Substitution – Permutation – Network”. La importancia de este tipo de esquema es que da al especialista en Criptografía la libertad de desarrollar un criptosistema simétrico particular. Un ejemplo de una caja de 256 elementos y que puede ser usada como caja de substitución en AES se muestra en la tabla 2.4.1 [10].

Tabla 2.4.1. Caja de substitución con 256 elementos.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 3d | 19 | 25 | 45 | 0a | 1d | 0f | f1 | b5 | cf | ff | b9 | 54 | 36 | 07 | d8 |
| 52 | cd | 20 | 8e | 2d | d6 | 41 | 37 | 55 | 04 | ca | 62 | 7c | 50 | 4b | 46 |
| 97 | e7 | b3 | 64 | 63 | 24 | f5 | b2 | 17 | 5c | 03 | bb | ad | 7a | 3e | 77 |
| 0b | 67 | 8c | bd | f7 | 51 | a9 | 7e | 79 | 70 | 27 | 87 | 40 | e6 | b6 | ac |
| c7 | b0 | 73 | 1e | 5f | 05 | 9d | b1 | fb | 2b | e8 | ae | 30 | 42 | fe | 48 |
| e9 | 4e | 5d | 0c | c0 | 9b | 6e | 44 | 14 | 9a | c9 | 7b | 2f | 78 | c3 | 29 |
| 84 | c8 | 85 | 6c | d2 | 08 | f8 | b4 | 26 | 7d | de | 49 | 3f | 1f | 4d | 74 |
| d4 | a6 | 86 | 83 | 90 | 21 | 32 | ef | 3a | 02 | 13 | bf | 92 | e2 | 89 | c4 |
| 93 | fc | d0 | a4 | 98 | 72 | 39 | 82 | fd | 3b | da | a3 | 6d | 80 | 06 | 16 |
| ec | 91 | a2 | e0 | 0e | 76 | 1c | 3c | 8b | db | af | 7f | a7 | e5 | 18 | f4 |
| 31 | 2c | 88 | df | e1 | 4a | a0 | a8 | a5 | 2a | 5b | 69 | 96 | 75 | fa | 9e |
| ed | 1a | bc | 9f | 8a | 33 | ab | 43 | 53 | e4 | 38 | 4c | e3 | 68 | d9 | 47 |
| 8f | 6b | dc | 34 | 9c | 15 | 66 | 99 | 8d | 0d | 95 | 81 | 6a | 01 | 57 | 56 |
| d7 | 10 | c6 | cc | cb | 23 | 94 | ba | b7 | 6f | 12 | ee | f2 | 59 | 00 | 4f |
| 09 | 5a | eb | 1b | 35 | ce | 65 | 71 | 28 | 61 | f3 | f0 | c2 | d3 | a1 | 58 |
| dd | 22 | b8 | d1 | 5e | 2e | 11 | aa | be | c5 | d5 | ea | f6 | f9 | 60 | c1 |

Más adelante se verá que la tabla anterior se puede generar de acuerdo a un algoritmo particular, y además, utilizando el Caos [11].

Así también, es posible construir una permutación particular del tamaño de la imagen, alterando la posición que tienen el arreglo de los pixeles en la imagen. De hecho, esta permutación puede ser variable, por ejemplo, se puede hacer depender de la primera llave ronda .

De igual forma, la generación de las llaves ronda puede ser desarrollada de forma particular. Por ejemplo, pueden utilizarse los números transcendentes, como , o también a la Curva Elíptica y el Caos.

Ejercicios

1.- Utilizando el patrón de la tabla 2.2.2.1, construya una permutación de un arreglo de 96 elementos, 1 – 96. Además, genere su inversa.

2.- Construya una tabla de expansión diferente de E.

3.- Construya una tabla de substitución siguiendo los patrones de las tablas .

4.- Encuentre la primera llave del programa de llaves,, del criptosistema DES, cuando la llave = a10f4708c621be73.

5.- Realice la operación {7 c} ● {0 3} mod. . Donde .

6.- Proponga un polinomio irreductible de grado ocho. Considerando que los coeficientes pueden ser 0 o 1.

7.- Dados los valores ac, 3f, 5a y bd; lleve a cabo la operación de substitución usando las cajas 2.3.2.1 y 2.4.1. Compare resultados.

8.- Proponga un sistema de cifrado “Substitution Permutation Network”.

9.- Encuentre la segunda llave ronda del criptosistema AES, , cuando la llave de 128 bits es la siguiente: 491f7a65de283be0bfe8462ca40ce63b.

10.- Escriba un programa de cómputo para el algoritmo AES.

11.- Utilizando el algoritmo AES, escriba un programa de cómputo para el modo CBC de cifrado.

Bibliografía

[1] Douglas R. Stinson. “CRYPTOGRAPHY: Theory and practice”, Chapman & Hall/ CRC Press, 2006.

[2] FIPS PUB 46-3. Federal Information Processing Standards Publication, 1999.

[3] V.M. Silva-García, R. Flores-Carapia, C. Rentería-Márquez, B. Luna-Benoso,”Triple-DES-96 Cryptographic System”, Vol. 8 No 19, pp. 925-934, Hikari LTD, 2013.

[4] Silva-García, V. M., Flores-Carapia, R., & Rentería-Márquez C. Triple-DES block of 96 bits: An application to colour image encryption. *Applied Mathematical Sciences*, Hikari, Vol. 7, (2013), no. 23, pp. 1143–1155.

[5] Barker W, Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, NIST Special Publication, (2008), 800-67.

[6] Víctor Manuel Silva-García, Rolando Flores-Carapia, Carlos Rentería- Márquez, Benjamín Luna-Benoso, Cesar Antonio Jiménez-Vázquez, Cipher image damage and decisions in real time, Journal of Electronic Imaging, SPIE, doi: 10.1117/1.JEI.24.1.013012, 2015.

[7] FIPS PUB 197. Federal Information Processing Standards Publications, 2001.

[8] Koblitz M., “A Course in Number Theory and Cryptography”, Springer-Verlag, New York, 1987.

[9] Gallian J., “Contemporary abstract algebra”, seventh edition, Brooks/Cole, 2011.

[10] Silva-García V.M., Flores-Carapia R., Rentería-Márquez C., Luna-Benoso B., Aldape-Pérez M., Substitution box generation using Chaos: An image encryption application, Applied Mathematics and Computation, Elsevier, 332, pp. 123-135, 2018.

[11] Víctor Manuel Silva García, Marlon David González Ramírez, Rolando Flores Carapia, Eduardo Vega-Alvarado, Eduardo Rodríguez Escobar, A Novel Method for Image Encryption Based on Chaos and Transcendental Numbers, IEEE Access, Vol. 7, pp. 163729-163739, 2019.