MOOC de Criptología Matemática. Data Encryption Standard (DES)

Leandro Marín

Módulo I. Sesión 3.
Dificultad Muy Alta

Introducción

- El *Data Encryption Standard* es un sistema de cifrado de bloques que ha sido empleado de forma masiva desde su creación, a principio de los años 70.

Introducción

- El *Data Encryption Standard* es un sistema de cifrado de bloques que ha sido empleado de forma masiva desde su creación, a principio de los años 70.
- Permite cifrar bloques de 64 bits utilizando claves que son a su vez de 64 bits.

Introducción

- El *Data Encryption Standard* es un sistema de cifrado de bloques que ha sido empleado de forma masiva desde su creación, a principio de los años 70.
- Permite cifrar bloques de 64 bits utilizando claves que son a su vez de 64 bits.
- La historia de DES es muy interesante, aunque en este curso no podemos dedicar tiempo a ella, es una lectura interesante.

Introducción

- El *Data Encryption Standard* es un sistema de cifrado de bloques que ha sido empleado de forma masiva desde su creación, a principio de los años 70.
- Permite cifrar bloques de 64 bits utilizando claves que son a su vez de 64 bits.
- La historia de DES es muy interesante, aunque en este curso no podemos dedicar tiempo a ella, es una lectura interesante.
- El estándar está publicado en un documento oficial, este es el primer ejemplo de sistema criptográfico que explicaremos siguiendo el estándar.

Introducción

- El *Data Encryption Standard* es un sistema de cifrado de bloques que ha sido empleado de forma masiva desde su creación, a principio de los años 70.
- Permite cifrar bloques de 64 bits utilizando claves que son a su vez de 64 bits.
- La historia de DES es muy interesante, aunque en este curso no podemos dedicar tiempo a ella, es una lectura interesante.
- El estándar está publicado en un documento oficial, este es el primer ejemplo de sistema criptográfico que explicaremos siguiendo el estándar.
- Todos los detalles del estándar se pueden leer en la publicación `http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf`

Representación de los Bits

- Tal y como hemos comentado en el tema anterior, el orden y la forma de representación de los bits es algo que debe quedar totalmente claro antes de empezar a estudiar un estándar.

Representación de los Bits

- Tal y como hemos comentado en el tema anterior, el orden y la forma de representación de los bits es algo que debe quedar totalmente claro antes de empezar a estudiar un estándar.
- En DES el bit más significativo será el situado más a la izquierda.

Representación de los Bits

- Tal y como hemos comentado en el tema anterior, el orden y la forma de representación de los bits es algo que debe quedar totalmente claro antes de empezar a estudiar un estándar.
- En DES el bit más significativo será el situado más a la izquierda.
- Para referirse a ellos, este bit situado más a la izquierda será numerado como 1, si estamos considerando el bloque completo de 64 bits, el bit más a la derecha será el bit 64.

Representación de los Bits

- Tal y como hemos comentado en el tema anterior, el orden y la forma de representación de los bits es algo que debe quedar totalmente claro antes de empezar a estudiar un estándar.
- En DES el bit más significativo será el situado más a la izquierda.
- Para referirse a ellos, este bit situado más a la izquierda será numerado como 1, si estamos considerando el bloque completo de 64 bits, el bit más a la derecha será el bit 64.
- Todas las notaciones de transformaciones y S-Boxes se dan utilizando esta terminología.

Estructura General del Algoritmo

- El algoritmo debe mezclar los datos que quiere cifrar con la clave.

Estructura General del Algoritmo

- El algoritmo debe mezclar los datos que quiere cifrar con la clave.
- Para hacer eso lo primero que se hace es generar a partir de la clave de 64 bits una sucesión números denominados subclaves, que se denotarán $K_1, K_2, \cdots, K_{16}$, cada una de ellas de 48 bits. El algoritmo que calcula estos valores se llama el planificador de claves.

Estructura General del Algoritmo

- El algoritmo debe mezclar los datos que quiere cifrar con la clave.
- Para hacer eso lo primero que se hace es generar a partir de la clave de 64 bits una sucesión números denominados subclaves, que se denotarán $K_1, K_2, \cdots, K_{16}$, cada una de ellas de 48 bits. El algoritmo que calcula estos valores se llama el planificador de claves.
- Para cifrar se utilizará un proceso de rondas, concretamente 16 rondas Feistel en las que se realizará una operación de mezcla de los datos con la subclave correspondiente.

Estructura General del Algoritmo

- El algoritmo debe mezclar los datos que quiere cifrar con la clave.
- Para hacer eso lo primero que se hace es generar a partir de la clave de 64 bits una sucesión números denominados subclaves, que se denotarán $K_1, K_2, \cdots, K_{16}$, cada una de ellas de 48 bits. El algoritmo que calcula estos valores se llama el planificador de claves.
- Para cifrar se utilizará un proceso de rondas, concretamente 16 rondas Feistel en las que se realizará una operación de mezcla de los datos con la subclave correspondiente.

Estructura General del Algoritmo

- El algoritmo debe mezclar los datos que quiere cifrar con la clave.
- Para hacer eso lo primero que se hace es generar a partir de la clave de 64 bits una sucesión números denominados subclaves, que se denotarán $K_1, K_2, \cdots, K_{16}$, cada una de ellas de 48 bits. El algoritmo que calcula estos valores se llama el planificador de claves.
- Para cifrar se utilizará un proceso de rondas, concretamente 16 rondas Feistel en las que se realizará una operación de mezcla de los datos con la subclave correspondiente.

Estructura General del Algoritmo

- El algoritmo debe mezclar los datos que quiere cifrar con la clave.
- Para hacer eso lo primero que se hace es generar a partir de la clave de 64 bits una sucesión números denominados subclaves, que se denotarán $K_1, K_2, \cdots, K_{16}$, cada una de ellas de 48 bits. El algoritmo que calcula estos valores se llama el planificador de claves.
- Para cifrar se utilizará un proceso de rondas, concretamente 16 rondas Feistel en las que se realizará una operación de mezcla de los datos con la subclave correspondiente.
- El proceso se completa con unas permutaciones iniciales y finales que reordenan los 64 bits de la entrada.

Rondas Feistel

- Una ronda Feistel es un proceso en el cual dado un bloque de información que se supondrá descompuesto en dos partes iguales L_i, R_i, se opera del siguiente modo:

Rondas Feistel

- Una ronda Feistel es un proceso en el cual dado un bloque de información que se supondrá descompuesto en dos partes iguales L_i, R_i, se opera del siguiente modo:
  - Se asigna a L_{i+1} el valor R_i.

Rondas Feistel

- Una ronda Feistel es un proceso en el cual dado un bloque de información que se supondrá descompuesto en dos partes iguales L_i, R_i, se opera del siguiente modo:
  - Se asigna a L_{i+1} el valor R_i.
  - Se asigna a R_{i+1} el valor $L_i \oplus F(R_i, K_i)$ donde $\oplus$ es la operación *o exclusivo* bit a bit y F es una función definida por el algoritmo que se aplica al valor R_i y a la subclave K_i correspondiente.

Rondas Feistel

- Una ronda Feistel es un proceso en el cual dado un bloque de información que se supondrá descompuesto en dos partes iguales L_i, R_i, se opera del siguiente modo:
  - Se asigna a L_{i+1} el valor R_i.
  - Se asigna a R_{i+1} el valor $L_i \oplus F(R_i, K_i)$ donde $\oplus$ es la operación *o exclusivo* bit a bit y F es una función definida por el algoritmo que se aplica al valor R_i y a la subclave K_i correspondiente.
- Una vez completada una ronda Feistel, estamos en condiciones de aplicar una nueva a la salida de la anterior y la nueva subclave de ronda.

Rondas Feistel

- Una ronda Feistel es un proceso en el cual dado un bloque de información que se supondrá descompuesto en dos partes iguales L_i, R_i, se opera del siguiente modo:
  - Se asigna a L_{i+1} el valor R_i.
  - Se asigna a R_{i+1} el valor $L_i \oplus F(R_i, K_i)$ donde $\oplus$ es la operación *o exclusivo* bit a bit y F es una función definida por el algoritmo que se aplica al valor R_i y a la subclave K_i correspondiente.
- Una vez completada una ronda Feistel, estamos en condiciones de aplicar una nueva a la salida de la anterior y la nueva subclave de ronda.
- Esta estructura de rondas Feistel no es única de DES, aunque DES es con gran diferencia el método más importante que las utiliza.

Rondas Feistel

- Una ronda Feistel es un proceso en el cual dado un bloque de información que se supondrá descompuesto en dos partes iguales L_i, R_i, se opera del siguiente modo:
  - Se asigna a L_{i+1} el valor R_i.
  - Se asigna a R_{i+1} el valor $L_i \oplus F(R_i, K_i)$ donde $\oplus$ es la operación *o exclusivo* bit a bit y F es una función definida por el algoritmo que se aplica al valor R_i y a la subclave K_i correspondiente.
- Una vez completada una ronda Feistel, estamos en condiciones de aplicar una nueva a la salida de la anterior y la nueva subclave de ronda.
- Esta estructura de rondas Feistel no es única de DES, aunque DES es con gran diferencia el método más importante que las utiliza.
- Lo que puede ser muy diferente entre unos algoritmos y otros es la función F que mezcla R_i con K_i.

La función F

- La función F recibe como argumentos un valor R_i de 32 bits y una subclave K_i de 48 bits. La salida de F debe tener 32 bits para poder operar con L_i y generar R_{i+1}.

La función F

- La función F recibe como argumentos un valor R_i de 32 bits y una subclave K_i de 48 bits. La salida de F debe tener 32 bits para poder operar con L_i y generar R_{i+1}.
- Lo primero que de debe hacer con R_i es convertirlo en una palabra de 48 bits a partir de sus 32 bits iniciales. Para ello se hace una expansión que se denota por E que consiste en repetir a intervalos regulares algunos de los bits de R_i.

La función F

- La función F recibe como argumentos un valor R_i de 32 bits y una subclave K_i de 48 bits. La salida de F debe tener 32 bits para poder operar con L_i y generar R_{i+1}.
- Lo primero que de debe hacer con R_i es convertirlo en una palabra de 48 bits a partir de sus 32 bits iniciales. Para ello se hace una expansión que se denota por E que consiste en repetir a intervalos regulares algunos de los bits de R_i.
- Una vez calculada esta expansión de 48 bits se opera mediante un *o exclusivo* (que se denota $\oplus$) con la subclave K_i. El resultado será un valor de 48 bits, que se troceará en 8 trozos de 6 bits que numeraremos desde el 1 hasta el 8.

La función F

- La función F recibe como argumentos un valor R_i de 32 bits y una subclave K_i de 48 bits. La salida de F debe tener 32 bits para poder operar con L_i y generar R_{i+1}.
- Lo primero que de debe hacer con R_i es convertirlo en una palabra de 48 bits a partir de sus 32 bits iniciales. Para ello se hace una expansión que se denota por E que consiste en repetir a intervalos regulares algunos de los bits de R_i.
- Una vez calculada esta expansión de 48 bits se opera mediante un *o exclusivo* (que se denota $\oplus$) con la subclave K_i. El resultado será un valor de 48 bits, que se troceará en 8 trozos de 6 bits que numeraremos desde el 1 hasta el 8.
- Para cada uno de esos trozos de 6 bits, dependiendo del valor del primer y el último de esos 6 bits, se elegirá una permutación, conocida como S-box que se aplicará a los 4 bits centrales. En total tenemos 4 permutaciones posibles para cada trozo y 8 trozos, lo que hace 32 permutaciones en total. Todas ellas están listadas en el estándar.

La función F

- La función F recibe como argumentos un valor R_i de 32 bits y una subclave K_i de 48 bits. La salida de F debe tener 32 bits para poder operar con L_i y generar R_{i+1}.
- Lo primero que de debe hacer con R_i es convertirlo en una palabra de 48 bits a partir de sus 32 bits iniciales. Para ello se hace una expansión que se denota por E que consiste en repetir a intervalos regulares algunos de los bits de R_i.
- Una vez calculada esta expansión de 48 bits se opera mediante un *o exclusivo* (que se denota $\oplus$) con la subclave K_i. El resultado será un valor de 48 bits, que se troceará en 8 trozos de 6 bits que numeraremos desde el 1 hasta el 8.
- Para cada uno de esos trozos de 6 bits, dependiendo del valor del primer y el último de esos 6 bits, se elegirá una permutación, conocida como S-box que se aplicará a los 4 bits centrales. En total tenemos 4 permutaciones posibles para cada trozo y 8 trozos, lo que hace 32 permutaciones en total. Todas ellas están listadas en el estándar.
- Las 8 salidas de 4 bits que hemos obtenido al aplicar estas permutaciones se vuelven a juntar en un bloque de 32 bits.

La función F

- La función F recibe como argumentos un valor R_i de 32 bits y una subclave K_i de 48 bits. La salida de F debe tener 32 bits para poder operar con L_i y generar R_{i+1}.
- Lo primero que de debe hacer con R_i es convertirlo en una palabra de 48 bits a partir de sus 32 bits iniciales. Para ello se hace una expansión que se denota por E que consiste en repetir a intervalos regulares algunos de los bits de R_i.
- Una vez calculada esta expansión de 48 bits se opera mediante un *o exclusivo* (que se denota $\oplus$) con la subclave K_i. El resultado será un valor de 48 bits, que se troceará en 8 trozos de 6 bits que numeraremos desde el 1 hasta el 8.
- Para cada uno de esos trozos de 6 bits, dependiendo del valor del primer y el último de esos 6 bits, se elegirá una permutación, conocida como S-box que se aplicará a los 4 bits centrales. En total tenemos 4 permutaciones posibles para cada trozo y 8 trozos, lo que hace 32 permutaciones en total. Todas ellas están listadas en el estándar.
- Las 8 salidas de 4 bits que hemos obtenido al aplicar estas permutaciones se vuelven a juntar en un bloque de 32 bits.
- Estos 32 bits se reordenan siguiendo una permutación P definida también en el estándar.

Las S-boxes

- Estas 32 permutaciones internas que definen la función F han sido estudiadas con mucho detalle.

Las S-boxes

- Estas 32 permutaciones internas que definen la función F han sido estudiadas con mucho detalle.
- Cumplen propiedades importantes que aseguran que el algoritmo es resistente a diversos tipos de ataques.

Las S-boxes

- Estas 32 permutaciones internas que definen la función F han sido estudiadas con mucho detalle.
- Cumplen propiedades importantes que aseguran que el algoritmo es resistente a diversos tipos de ataques.
- Desde el punto de vista del usuario pueden parecer unos números aleatorios, pero no lo son en absoluto.

Las S-boxes

- Estas 32 permutaciones internas que definen la función F han sido estudiadas con mucho detalle.
- Cumplen propiedades importantes que aseguran que el algoritmo es resistente a diversos tipos de ataques.
- Desde el punto de vista del usuario pueden parecer unos números aleatorios, pero no lo son en absoluto.
- Estas permutaciones son las mismas para todas las rondas.

El Planificador de Claves

- Dada la clave K de 64 bits, debemos generar 16 subclaves $K_1, K_2, \cdots, K_{16}$.

El Planificador de Claves

- Dada la clave K de 64 bits, debemos generar 16 subclaves $K_1, K_2, \cdots, K_{16}$.
- Para hacer eso, se definen a partir de los bits de K dos palabras de 28 bits que se llamarán C_0 y D_0. La forma como se corresponden los bits de K en los de C_0 y D_0 está definido mediante una tabla llamada $PC1$ (permuted choice 1) que viene definida en el estándar.

El Planificador de Claves

- Dada la clave K de 64 bits, debemos generar 16 subclaves $K_1, K_2, \cdots, K_{16}$.
- Para hacer eso, se definen a partir de los bits de K dos palabras de 28 bits que se llamarán C_0 y D_0. La forma como se corresponden los bits de K en los de C_0 y D_0 está definido mediante una tabla llamada $PC1$ (permuted choice 1) que viene definida en el estándar.
- Los bits de C_0 y D_0 se rotan a la izquierda una posición. El resultado de esas rotaciones lo llamaremos C_1 y D_1.

El Planificador de Claves

- Dada la clave K de 64 bits, debemos generar 16 subclaves $K_1, K_2, \cdots, K_{16}$.
- Para hacer eso, se definen a partir de los bits de K dos palabras de 28 bits que se llamarán C_0 y D_0. La forma como se corresponden los bits de K en los de C_0 y D_0 está definido mediante una tabla llamada $PC1$ (permuted choice 1) que viene definida en el estándar.
- Los bits de C_0 y D_0 se rotan a la izquierda una posición. El resultado de esas rotaciones lo llamaremos C_1 y D_1.
- Una tabla llamada $PC2$ seleccionará 48 de los bits de C_1 y D_1 para generar K_1.
- Con C_1 y D_1 volveremos a hacer una rotación a la izquierda y generaremos C_2 y D_2. La misma tabla $PC2$ seleccionará los bits de C_2 y D_2 para crear K_2.

El Planificador de Claves

- Dada la clave K de 64 bits, debemos generar 16 subclaves $K_1, K_2, \cdots, K_{16}$.
- Para hacer eso, se definen a partir de los bits de K dos palabras de 28 bits que se llamarán C_0 y D_0. La forma como se corresponden los bits de K en los de C_0 y D_0 está definido mediante una tabla llamada $PC1$ (permuted choice 1) que viene definida en el estándar.
- Los bits de C_0 y D_0 se rotan a la izquierda una posición. El resultado de esas rotaciones lo llamaremos C_1 y D_1.
- Una tabla llamada $PC2$ seleccionará 48 de los bits de C_1 y D_1 para generar K_1.
- Con C_1 y D_1 volveremos a hacer una rotación a la izquierda y generaremos C_2 y D_2. La misma tabla $PC2$ seleccionará los bits de C_2 y D_2 para crear K_2.
- Este proceso de rotaciones y selecciones se repetirá hasta obtener las 16 subclaves. El número de rotaciones a la izquierda no es siempre uno, en algunos casos se rotará dos veces a la izquierda y en otros una sola vez. El número de rotaciones también está definido en el estándar.

Conclusiones

- Como se puede apreciar, el estándar que define DES está lleno de permutaciones y selecciones de bits.

Conclusiones

- Como se puede apreciar, el estándar que define DES está lleno de permutaciones y selecciones de bits.
- Eso impone una gran cantidad de información que debe tenerse en cuenta a la hora de programar el algoritmo.

Conclusiones

- Como se puede apreciar, el estándar que define DES está lleno de permutaciones y selecciones de bits.
- Eso impone una gran cantidad de información que debe tenerse en cuenta a la hora de programar el algoritmo.
- Sin embargo, esa forma de proceder permite implementaciones muy rápidas, especialmente en hardware.

Conclusiones

- Como se puede apreciar, el estándar que define DES está lleno de permutaciones y selecciones de bits.
- Eso impone una gran cantidad de información que debe tenerse en cuenta a la hora de programar el algoritmo.
- Sin embargo, esa forma de proceder permite implementaciones muy rápidas, especialmente en hardware.
- Vamos a ver a continuación los valores de las tablas que hemos ido nombrando.

Conclusiones

- Como se puede apreciar, el estándar que define DES está lleno de permutaciones y selecciones de bits.
- Eso impone una gran cantidad de información que debe tenerse en cuenta a la hora de programar el algoritmo.
- Sin embargo, esa forma de proceder permite implementaciones muy rápidas, especialmente en hardware.
- Vamos a ver a continuación los valores de las tablas que hemos ido nombrando.
- Por supuesto, no es necesario conocerlos a no ser que queramos implementar el algoritmo.

La permutación inicial

```
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 ]
```

La permutación final es la inversa $\texttt{IP}^{-1}$.

Expansión de los 32 bits de R a 48 bits en la función F.

```
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 de salida de la función F.

```
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 ]
```

Permutación para general C_0 y D_0 en el planificador de claves.

```
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 ]
```

Permutación para general la subclave K_i a partir de C_i y D_i en el planificador de claves.

```
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 ]
```

Número de rotaciones que hay que hacer para cada i en los valores C_i y D_i para generar C_{i+1} y D_{i+1}.

```
rot = [1,1,2,2,2,2,2,2,1,2,2,2,2,2,2,1]
```

S-boxes

```
S = [[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],
     [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],
     [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],
     [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],
     [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],
     [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],
     [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],
     [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]]
```

Descifrado DES

- El método DES permite utilizar el mismo algoritmo para cifrar y para descifrar.

Descifrado DES

- El método DES permite utilizar el mismo algoritmo para cifrar y para descifrar.
- Simplemente hay que reordenar las subclaves, y el mismo algorimo nos recupera el mensaje original.

Descifrado DES

- El método DES permite utilizar el mismo algoritmo para cifrar y para descifrar.
- Simplemente hay que reordenar las subclaves, y el mismo algorimo nos recupera el mensaje original.
- Esto facilita enormemente la implementación, puesto que se puede reutilizar casi todo el código.

Triple DES

- Aunque el algoritmo DES ha sido analizado hasta la saciedad sin haber obtenido inseguridades importantes, las claves son cortas, sólo 64 bits de los cuales algunos son de control y realmente sólo se utilizan 56. Esto hace que la capacidad de cálculo de los ordenadores actuales permita romper DES utilizando fuerza bruta.

Triple DES

- Aunque el algoritmo DES ha sido analizado hasta la saciedad sin haber obtenido inseguridades importantes, las claves son cortas, sólo 64 bits de los cuales algunos son de control y realmente sólo se utilizan 56. Esto hace que la capacidad de cálculo de los ordenadores actuales permita romper DES utilizando fuerza bruta.
- Para mejorar el algoritmo se ideó Triple DES, que es aplicar DES tres veces con claves distintas.

Triple DES

- Aunque el algoritmo DES ha sido analizado hasta la saciedad sin haber obtenido inseguridades importantes, las claves son cortas, sólo 64 bits de los cuales algunos son de control y realmente sólo se utilizan 56. Esto hace que la capacidad de cálculo de los ordenadores actuales permita romper DES utilizando fuerza bruta.
- Para mejorar el algoritmo se ideó Triple DES, que es aplicar DES tres veces con claves distintas.
- Eso mejoró enormemente la seguridad, hasta el punto de que se pudo seguir utilizando durante bastantes años, estando hoy en día aun presente en muchos sitios.

Triple DES

- Aunque el algoritmo DES ha sido analizado hasta la saciedad sin haber obtenido inseguridades importantes, las claves son cortas, sólo 64 bits de los cuales algunos son de control y realmente sólo se utilizan 56. Esto hace que la capacidad de cálculo de los ordenadores actuales permita romper DES utilizando fuerza bruta.
- Para mejorar el algoritmo se ideó Triple DES, que es aplicar DES tres veces con claves distintas.
- Eso mejoró enormemente la seguridad, hasta el punto de que se pudo seguir utilizando durante bastantes años, estando hoy en día aun presente en muchos sitios.
- Para aplicar Triple DES se toman tres claves, K_1, K_2 y K_3 y se aplica el cifrado DES con K_1, luego el descifrado DES con K_2 y el cifrado DES con K_3. De esta forma si $K_1 = K_2 = K_3$ hacemos cifrado, descifrado y cifrado, o lo que es lo mismo, un único cifrado. Eso permitió la compatibilidad de los sistemas cuando se pasó de DES a Triple DES, puesto que utilizando tres claves iguales se obtenía el método DES anterior.