**Exercice 1 :** *(Chiffrement DES)*

Soit le message clair: **0123456789ABCDEF** et la clé : **133457799BBCDFF1**. Calculez :

- Appliquer *PI* sur le message clair.

- Déterminer L0 et R0.

- Calculer K1

- Calculer E(R0) ⊕ K1

- Le message crypté après le round 1 (étape 1)

**Solution :**

DES est un crypto-système par blocs qui travaille sur des blocs de 64 bits. Il est constitué de 04 étapes:

*1-****Calcul de la clé***: fabrication de 16 sous- clefs Ki

*2-****Permutation initiale***

*3-****Calcul médian***(16 fois): application d’un algorithme complexe appliqué en fonction de la clef.

*4-****Permutation finale***



* **Calculer K1 :**

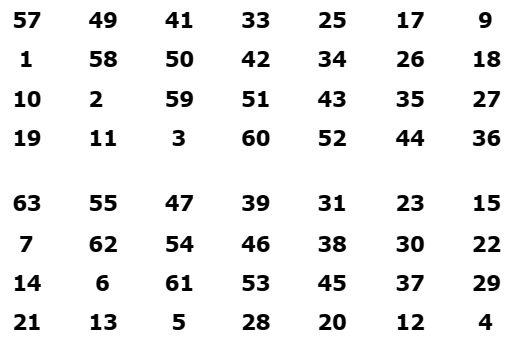
Convertir la clé en binaire :

**133457799BBCDFF1🡪 (64 bits)**

**0001001100110100010101110111100110011011101111001101** **111111110001**

1. **Réduction à 56 bits :**

On enlève les 8 derniers bits (bits en rouge) en effectuant une permutation, basée sur la matrice PC-1 :



**Permutation « PC - 1 »**

On obtient alors une nouvelle clé sur **56 bits** :

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Bits | 8 | 16 | 24 | 32 | 40 | 48 | 56 | 64 |
| K | 00010011 | 00110100 | 01010111 | 01111001 | 10011011 | 10111100 | 11011111 | 11110001 |
| Kp | 1111000 | 0110011 | 0010101 | 0101111 | 0101010 | 1011001 | 1001111 | 0101111 |

1. **Division en sous-clés de 28 bits**

|  |  |  |
| --- | --- | --- |
| Kp | 1111000011001100101010101111 | 0101010101100110011110101111 |

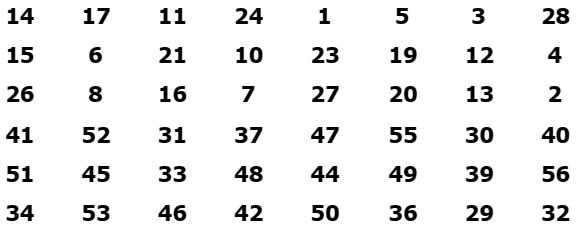
1. **Rotation de la clé**

A chaque itération chaque sous-clé subit un **décalage circulaire à gauche.** On décale de un bit pour les itérations de rang 1, 2, 9 ou 16 et de 2 bits pour les autres itérations. Pour la première itération, nous obtenons donc après 1 décalage à gauche de chacune des deux parties :

|  |  |  |
| --- | --- | --- |
| Kp | 1110000110011001010101011111 | 1010101011001100111101011110 |

1. **Réduction**

Une dernière permutation, dite « **PC-2** », mélange et enlève 8 bits en fonction de la matrice suivante :



**Permutation « PC–2 »**

Nous obtenons donc la clé K1 de **48** **bits** :

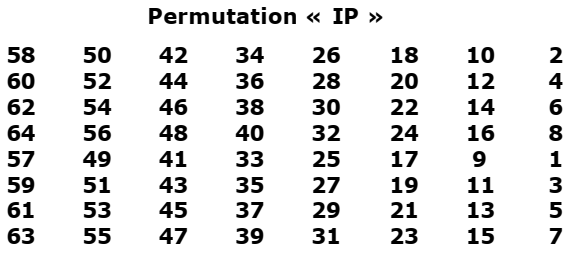
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Bits | 7 | 14 | | 21 | | 28 | 35 | 42 | | 49 | | 56 |
| Kp | 1110000 | 1100110 | | 0101010 | | 1011111 | 1010101 | 0110011 | | 0011110 | | 1011110 |
| K1 | 00011011 | | 00000010 | | 11101111 | | 11111100 | | 01110000 | | 01111010 | |

* **Appliquer PIsur le message clair.**

Le message clair: **0123456789ABCDEF**

Convertir le message en binaire :

000000010010001 010001010110011110001001101010111100110111101111



Appliquer *PI* sur le message clair :

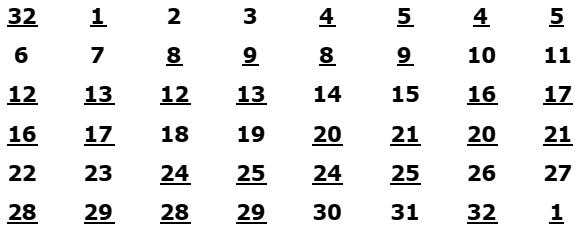
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Bits | 8 | 16 | 24 | 32 | 40 | 48 | 56 | 64 |
| M | 00000001 | 00100011 | 01000101 | 01100111 | 10001001 | 10101011 | 11001101 | 11101111 |
| Mp | 11001100 | 00000000 | 11001100 | 11111111 | 11110000 | 10101010 | 11110000 | 10101010 |

* **Déterminer L0 et R0.**

|  |  |  |  |
| --- | --- | --- | --- |
| L0 | 11001100000000001100110011111111 | R0 | 11110000101010101111000010101010 |

* **Calculer E(R0) ⊕ K1.**

La **Fonction d'Expansion**, « **E** », permutation expansive, utilise une liste de 48 valeurs et travaille sur un bloc de 32 bits. On obtient ainsi un bloc de 48 bits avec répétition de certains de ces bits (valeurs soulignées).



|  |  |
| --- | --- |
| R0 | 11110000101010101111000010101010 |
| E(R0) | 0111 1010 0001 0101 0101 0101 0111 1010 0001 0101 1010 0101 |

Calculer E(R0) ⊕ K1

|  |  |
| --- | --- |
| E(R0) | 011110100001010101010101011110100001010110100101 |
| K1 | 000110110000001011101111111111000111000001111010 |
| E(R0) ⊕K1 | 011000010001011110111010100001100110010111011111 |

* **Le message crypté après le round 1 (étape 1)**

Calculer L1 et R1 :

**L1 = R0** = 11110000101010101111000010101010

Selon le schéma suivant **Ri = Li-1 ⊕ F(Ri-1, Ki)** donc **R1= L0 ⊕ F(R0, K1)**



**Fonction F**:  
1. Expansion  
2. Ajout de la clé  
3. Transformations (S-Box, P-Box)

Nous avons déjà, calculé E(R0) ⊕ K1 donc il nous reste la troisième étape qui est les transformations :

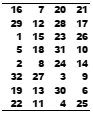
1. Substitution S-Box

L'opération de substitution consiste pour chaque S-box à calculer:  
• **b1b6 = n° de ligne**• **b2b3b4b5 = n° de colonne**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| E(R0)⊕K1 | 011000010001011110111010100001100110010111011111 | | | | | | | |
| E(R0)⊕K1 | 011000 | 010001 | 011110 | 111010 | 100001 | 100110 | 010111 | 011111 |
| N° ligne  N° colonne | 00=0 | 01=1 | 00=0 | 10=2 | 11=3 | 10=2 | 01=1 | 01=1 |
| 1100=12 | 1000=8 | 1111=15 | 1101=13 | 0000=0 | 0011=3 | 1011=11 | 1111=15 |
| Résultat | 0101 | 1100 | 1000 | 0010 | 1011 | 0101 | 1100 | 0010 |

1. Permutation P-Box

L'opération de permutation est réalisée sur le résultat de la substitution des S-box et est basée sur la table suivante:



|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Bits | 4 | 8 | 12 | 16 | 20 | 24 | 28 | 32 |
| S-Box | 0101 | 1100 | 1000 | 0010 | 1011 | 0101 | 1100 | 0010 |
| P-Box | 0010 | 0001 | 0101 | 1010 | 1010 | 0001 | 1001 | 1011 |
| F(R0, K1) | 00100001010110101010000110011011 | | | | | | | |

Le résultat de cette dernière permutation est noté F(R0,K1). Puisque R1= L0 ⊕ F(R0, K1) donc :

|  |  |
| --- | --- |
| L0 | 11001100000000001100110011111111 |
| F(R0, K1) | 00100001010110101010000110011011 |
| R1 | 11101101010110100110110101100100 |

Donc L1 et R1 sont :

|  |  |  |  |
| --- | --- | --- | --- |
| L1 | 11110000101010101111000010101010 | R1 | 11101101010110100110110101100100 |

Le message chiffré après le premier tour est :

1111000010101010111100001010101011101101010110100110110101100100