



# La cryptographie Moderne

# Introduction

- ❑ Objectifs de la cryptologie
  - Confidentialité
  - Authenticité
  - Intégrité
  
- ❑ Ne couvre qu'une partie du problème de la sécurité informatique qui compte également :
  - *Disponibilité*

# Cryptographie moderne

- ❑ Introduction aux protocoles
  - Les protocoles régissent notre vie quotidienne
  - Lorsque l'on achète dans un magasin
  - Lorsque l'on téléphone
  - Lorsque l'on prend le bus
  - etc.
- ❑ Un protocole est une série d'étapes ordonnées
- ❑ Il organise la résolution de problèmes entre au moins 2 participants

## Suite

### ❑ Caractéristiques d'un protocole:

- Tous les participants doivent le connaître
- Chaque participant doit accepter le protocole
- Non ambiguïté du protocole
  - les étapes sont claires
- Un protocole doit être complet
  - toutes les situations du problème à résoudre sont prises en compte

### ❑ Protocole cryptographique

- Il ne doit pas être possible d'en apprendre plus que ce que le protocole prévoit



## ❑ Protocole arbitré

- S'appuie sur un tiers de confiance
- Exemple de la vie courante : achat d'une maison
  - Deux participants : l'acheteur et le vendeur
  - Ils ne se connaissent pas et ne se font pas confiance
    - la maison appartient-elle au vendeur, le chèque de l'acheteur est-il provisionné ?
  - Tous les deux se tournent vers un notaire en qui chacun a confiance

## ❑ Exemple informatique : certificat électronique

- Le tiers de confiance est l'autorité de certification qui garantit le lien entre le certificat numérique et l'entité physique



## ❑ Protocole à discipline intrinsèque

- Pas de tiers de confiance
- La tricherie est mise en évidence par le protocole lui-même

# Cryptosystème à clé secrète

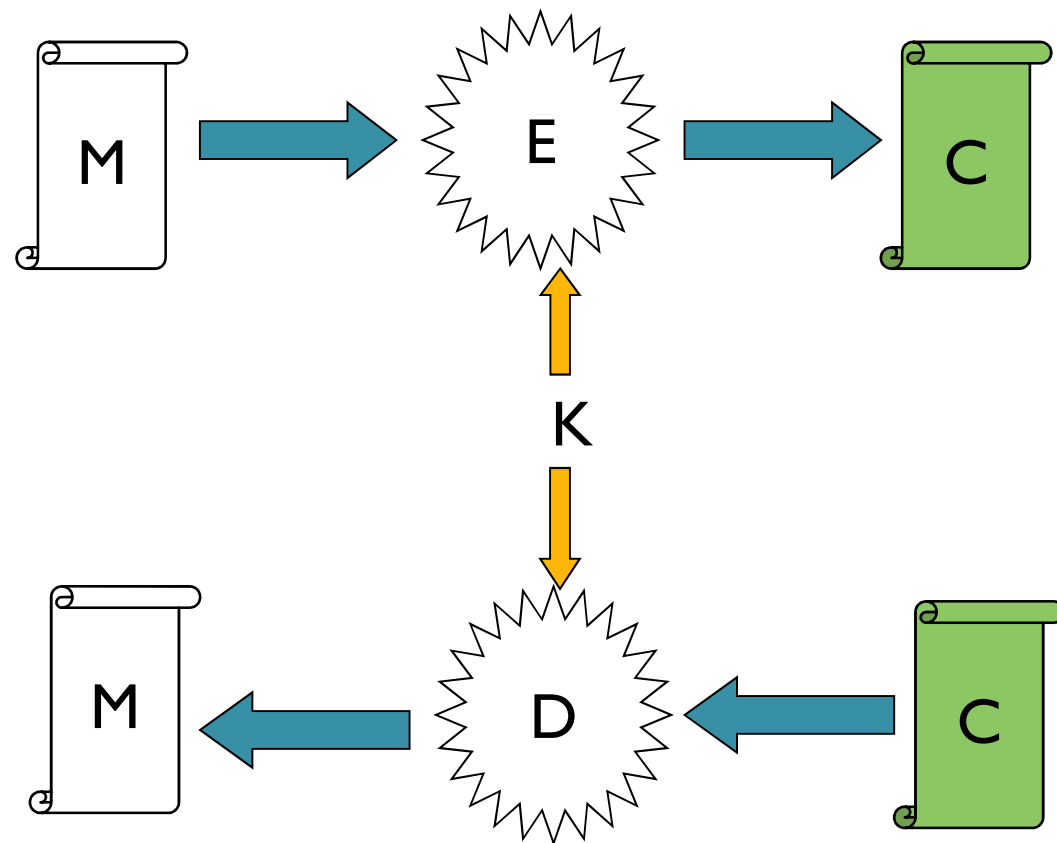
□ = cryptographie symétrique

□ Le protocole


1. Alice et Bob choisissent un cryptosystème C
2. Alice et Bob choisissent une clé K
3. Alice chiffre son texte clair avec le cryptosystème C et avec la clé K
4. Alice envoie le texte chiffré
5. Bob déchiffre le texte chiffré avec le cryptosystème C et la clé K

## Suite

- ❑ Clé K secrète partagée entre Alice et Bob





- 
- ❑ Les étapes 1 et 4 peuvent être publiques
    - La sécurité ne repose pas sur l'algorithme
    - On ne peut pas garantir la sécurité du réseau de transmission
  - ❑ La sécurité repose sur l'étape 2
    - Les clés doivent être distribuées secrètement
    - Le nombre de clés à distribuer croît avec le carré du nombre d'utilisateurs
    - La longueur des clés doit être bien choisie
    - La clé doit être secrète avant, pendant et après



## ❑ Chiffre de Vernam (1917)

- L'algorithme de chiffrement et de déchiffrement est un simple XOR
- La longueur de la clé est égale à celle du message
- Une clé n'est utilisée qu'une seule fois
- La clé doit être la plus aléatoire possible
  - C'est le cryptosystème le plus sûr !
  - Mais il n'est pas très pratique...

### ❑ Exemple :

- Soit M le message à chiffrer (ici, 24 bits)

$$M = 010011100101001001010100$$

- On génère spécialement une clé aléatoire K (de 24 bits)

$$K = 110101010011100101010010$$

- $C = M \oplus K$

$$C = 100110110110101100000110$$

- $M = C \oplus K$



## ❑ Chiffre de Vernam

- C'est l'algorithme à clé secrète le plus simple et le plus sûr
- Deux inconvénients :
  - La taille des clés
  - Le générateur peut-il être vraiment aléatoire ?
- En pratique, on est obligé de prendre des clés plus courtes, avec des algorithmes plus compliqués



## ❑ Sécurité calculatoire


- Une attaque exhaustive est-elle possible ?
- Dépend du temps et des moyens disponibles
- Exemple : la longueur de la clé est 128 soit  $2^{128}$  clés différentes
- Supposons que l'on puisse tester  $2^{40}$  clés par seconde
  - Il faut  $2^{88}$  secondes pour les tester toutes
  - Soit plus que l'âge de l'univers ( $2^{61}$  secondes)
  - D'un point de vue calculatoire, cela semble sûr
  - Mais pour combien de temps ?



# DES

# Historique

- ❑ 1973 : appel d'offre du NIST pour un cryptosystème standard
  - Sécurité élevée
  - Spécification complète et compréhensible
  - Sécurité liée à la clé
  - Disponible à tous les utilisateurs
  - Réalisation électronique simple et rentable
  - Efficacité
  - Validable

- 
- ❑ Peu de réponses à l'appel d'offre
  - ❑ Nouvel appel d'offre en 1974 :
    - Un candidat sérieux : IBM
    - L'algorithme candidat : LUCIFER
    - Publication de l'algorithme en 1975
    - Evaluation et modification de l'algo par la NSA
    - Clé de 128 bits réduite à 56 bits
    - Modification des S-tables





## ❑ En 1977

- Elaboration de « Data Encryption Standard » **DES**
- Réévaluation tous les 5 ans
- La dernière version date de 1999
- Quelques variantes : 3DES, GDES
- Il est utilisé pour chiffrer les mots de passe dans le système Unix

# Principe général de l'algorithme

- ❑ Chiffrement par bloc de 64 bits
  - Clé de 64 bits
    - 56 bits utilisés
    - 1 bit sur 8 peut être utilisé comme contrôle de parité
- ❑ Structure de l'algorithme : Réseau de Feistel en 16 rondes



## ❑ Réseau de Feistel

- Le bloc initial est décomposé en 2 parties
- Algorithme en plusieurs rondes (ou tours)
- Lors de chaque ronde
  - Utilisation d'une sous-clé particulière
  - Un sous-bloc est combiné à une sous-clé par une fonction  $f$
  - Le sous-bloc transformé est combiné par XOR au sous-bloc non transformé
  - Les deux sous-blocs sont échangés pour la ronde suivante



## □ Principe

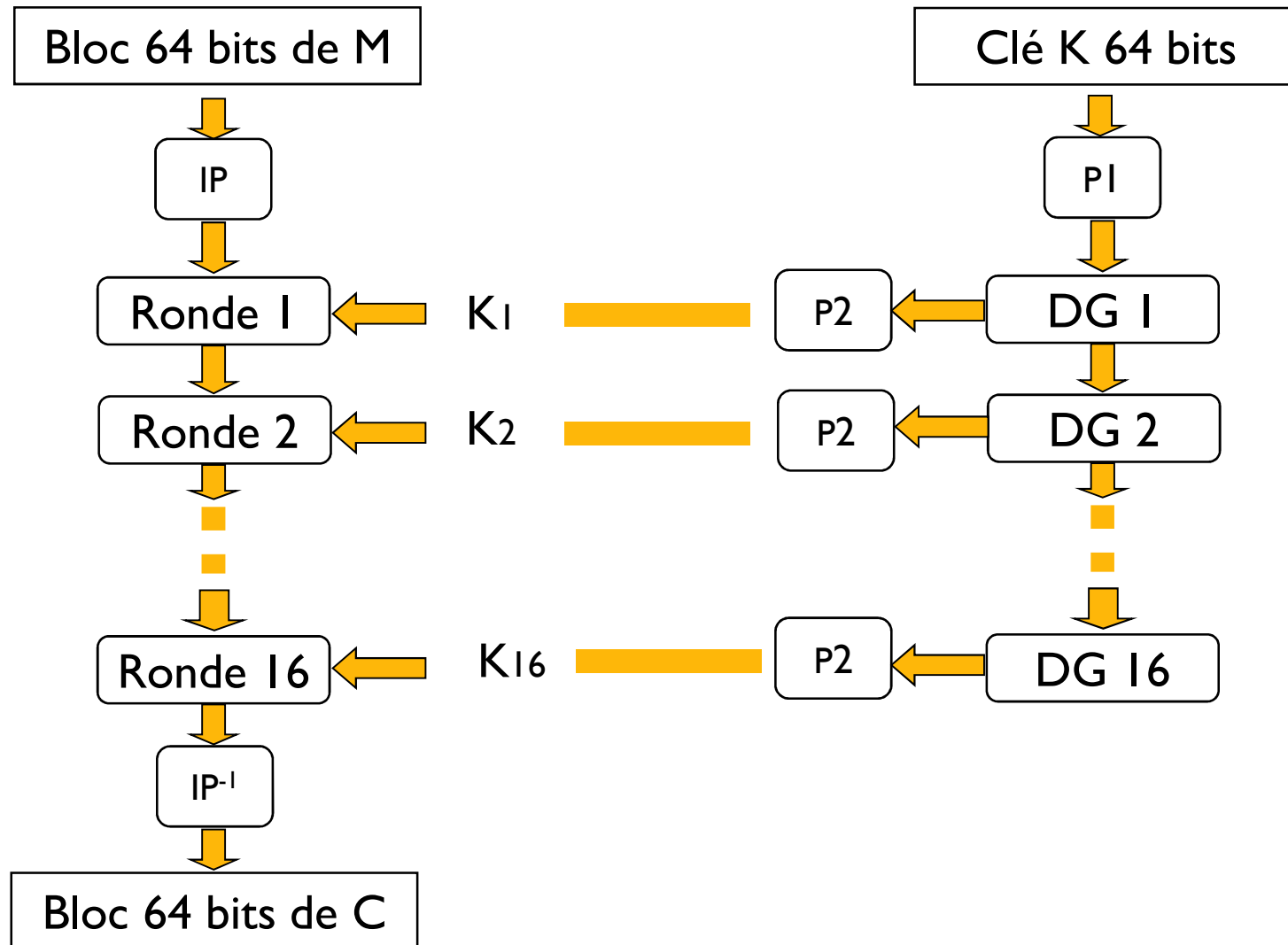
- DES est basé sur un ensemble de transformations:  
transpositions, substitutions et opérations non-linéaires
- Chiffrement de blocs de 64 bits grâce à une clé de 56 bits
- Processus de chiffrement: Permutations  $IP$  et  $IP^{-1}$ , 16 rondes utilisant 16 clés secondaires  $K_1$  à  $K_{16}$  de 48 bits issues à partir de la clé  $K$ .
- Processus de génération des clés secondaires:  
Permutations  $P_1$  et  $P_2$ , Décalages circulaires

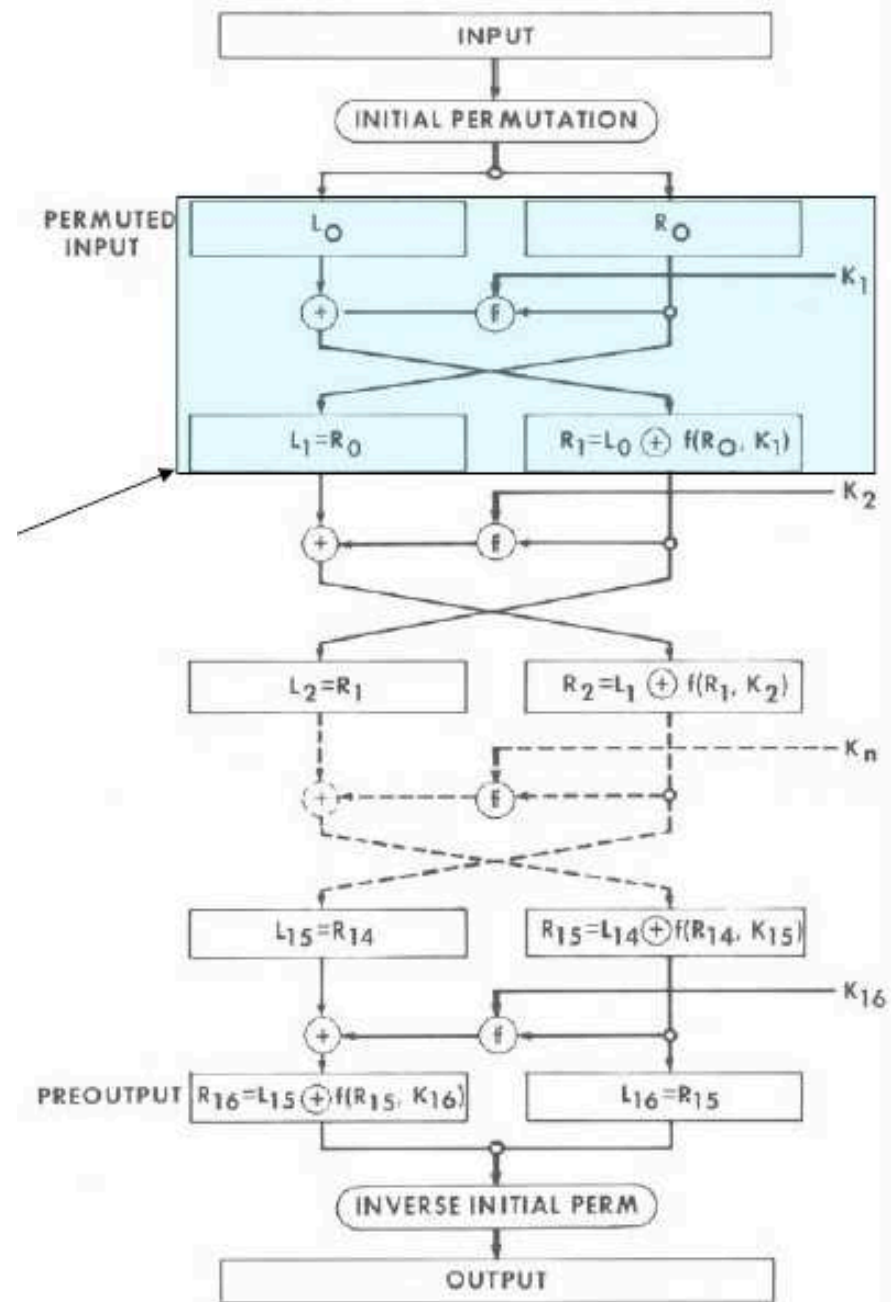


## ❑ Propriétés d'un réseau de Feistel

- La symétrie : le chiffrement et le déchiffrement utilisent le même algorithme (ou presque)
- La simplicité : n'utilise que des opérateurs simples (opérateurs logiques, substitutions et permutations de bits)
- La modification d'un seul bit en entrée produit une sortie totalement différente.
  - Chaque bit du texte chiffré dépend de tous les bits du texte clair
- Principe utilisé dans d'autres algorithmes symétriques

# Schéma général du DES





# Fonctionnement de DES

- ❑ DES fait subir à  $M$  une permutation  $IP \longrightarrow M'$ 
  - $M' = IP(M)$
- ❑  $M'$  est divisé en deux mots de 32 bits
  - $L_0$ : partie gauche
  - $R_0$ : partie droite
- ❑ DES exécute 16 itérations de la fonction  $f$ 
  - Ces itérations combinent: substitutions et transpositions
- ❑ Les parties gauche et droite sont modifiées comme suit:
  - $L_i = R_{i-1}$
  - $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$



- ❑ Pour  $1 \leq i \leq 16$  et  $f$  une fonction qui prend en entrée les 32 bits de la partie droite et les 48 bits de la clé et fournit une sortie de 32 bits
- ❑  $f$  utilise huit permutations appelées S-boxes qui associent à 6 bits d'entrée 4 bits de sortie
- ❑ Chaque  $K_i$  contient un sous-ensemble différent des 56 bits de la clé originale.
- ❑ Enfin le pré-cryptogramme  $C' = (R_{16}, L_{16})$  subit une permutation inverse de la permutation initiale  $IP^{-1}$  et donne le cryptogramme final

## Fonction f et S-boxes

### □ $f(R_{i-1}, K_i)$

- Dans un premier temps,  $R_{i-1}$  est étendu en un bloc de 48 bits en utilisant la table E (Expansion)
  - $R_{i-1} = r_1 r_2 \dots r_{32}$  est transformé en  $r_{32} r_1 r_2 \dots r_{32} r_1$  en répétant certains bits
- Opération  $\oplus$  entre  $E(R_{i-1})$  et  $K_i$
- On scinde ensuite le résultat de cette opération en huit blocs de 6 bits  $B_1, B_2, \dots, B_8$ 
  - $E(R_{i-1}) \oplus K_i = B_1 B_2 \dots B_8$
- Chaque  $B_j$ , pour  $1 \leq i \leq 8$  est ensuite utilisé comme l'entrée d'une fonction de substitution (S-box),  $S_j$  qui envoie un bloc de 4 bits en sortie  $S_j(B_j)$ .

## Calcul de $S_j(B_j)$

- Chaque S-box associe à un bloc de 6 bits

$$B_j = b_1 b_2 b_3 b_4 b_5 b_6$$

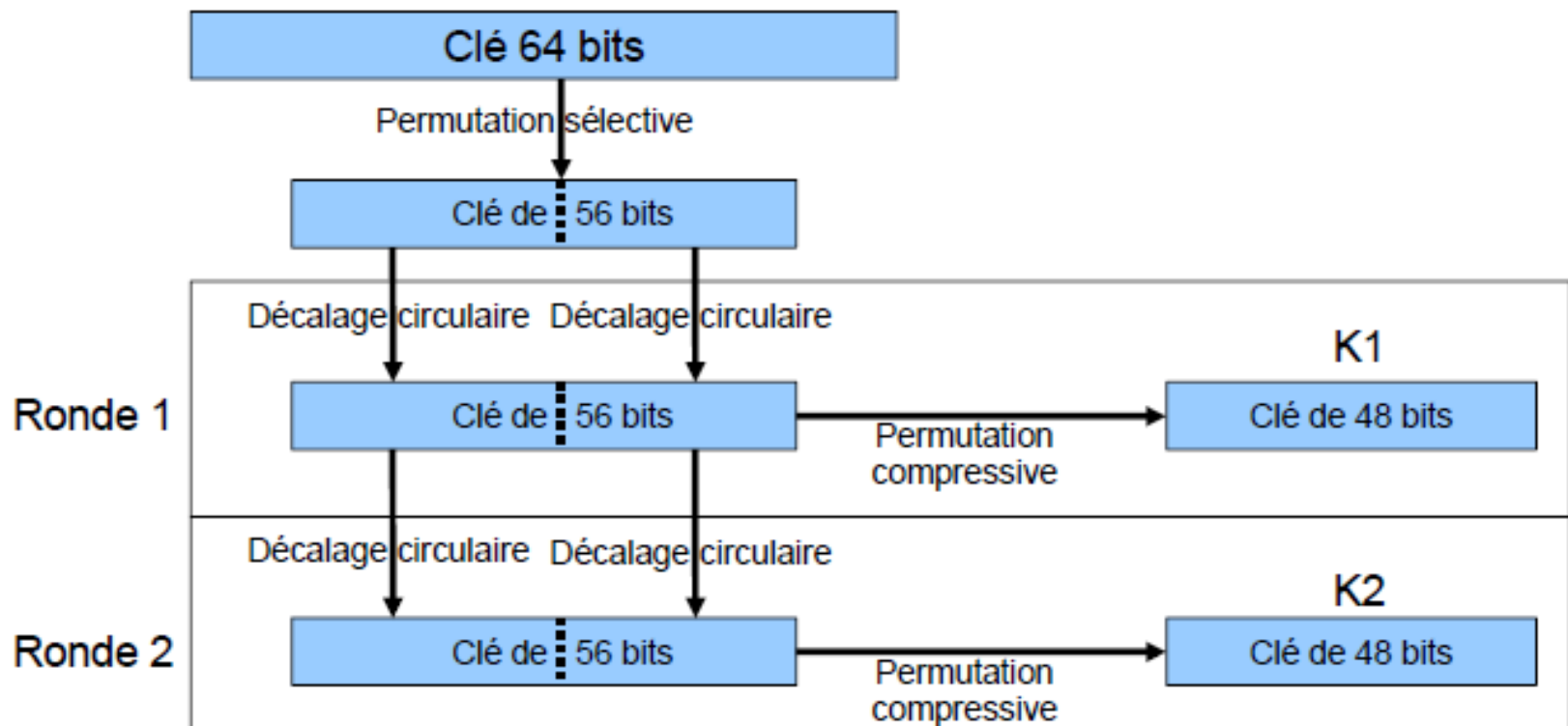
- Un bloc de 4 bits est créé

- L'entier représenté par  $b_1 b_6$  sélectionne une ligne
- L'entier représenté par  $b_2 b_3 b_4 b_5$  sélectionne une colonne

- La valeur de  $S_j(B_j)$  est la représentation de l'entier inscrit dans la S-Box à cette position

# Génération de clés

## Génération des 16 sous-clés de 48 bits



# Permutation sélective

## ❑ Permutation sélective (P1)

- Le bit 1 de la clé résultat correspond au bit 57 de la clé initiale, le bit 2 correspond au bit 49, etc.
- Extrait une clé de 56 bits à partir des 64 bits de départ

```
unsigned int    permutation_selective[56] = {  
    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  
};
```

# Décalage circulaire gauche

## □ Décalage circulaire des demi-clés de 28 bits

- Lors de la 1<sup>ère</sup> ronde, les bits sont décalés d'une position vers la gauche

➤ Ex :  $b_1b_2b_3\dots b_{28}$    $b_2b_3\dots b_{28}b_1$

- Le même décalage est appliqué sur la demi-clé gauche et la demi-clé droite
- Ils sont décalés d'une position lors de la 2<sup>e</sup> ronde, de 2 positions lors de la 3<sup>e</sup> ronde, etc.

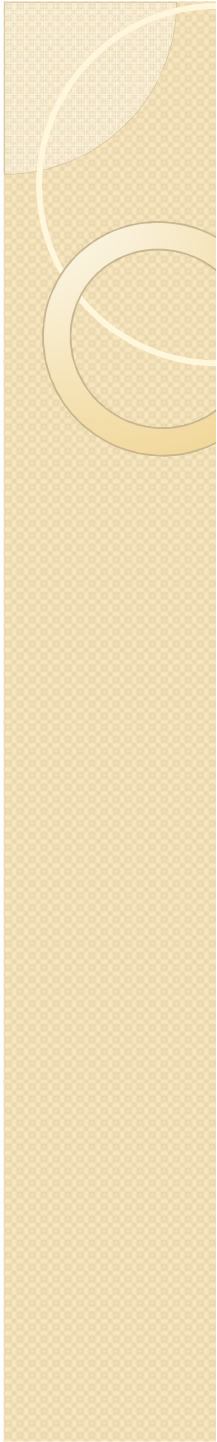
```
unsigned int    decalage_cle[16] = {  
    1, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1  
};
```

# Permutation compressive

## ❑ Permutation compressive

- Choisit 48 bits parmi 56 bits de la clé décalée en les permutant
- Le bit 1 de la clé compressée correspond au bit 14 de la clé décalée, etc.

```
unsigned int    permutation_compressive[48] = {  
    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  
};
```

- 
- ❑ Ces transformations sur les clés ont pour but:
    - De fournir des sous-clés de 48 bits à la fonction  $f$
    - D'utiliser tous les bits de la clé initiale (hormis les 8 ignorés dès le départ) à peu près le même nombre de fois



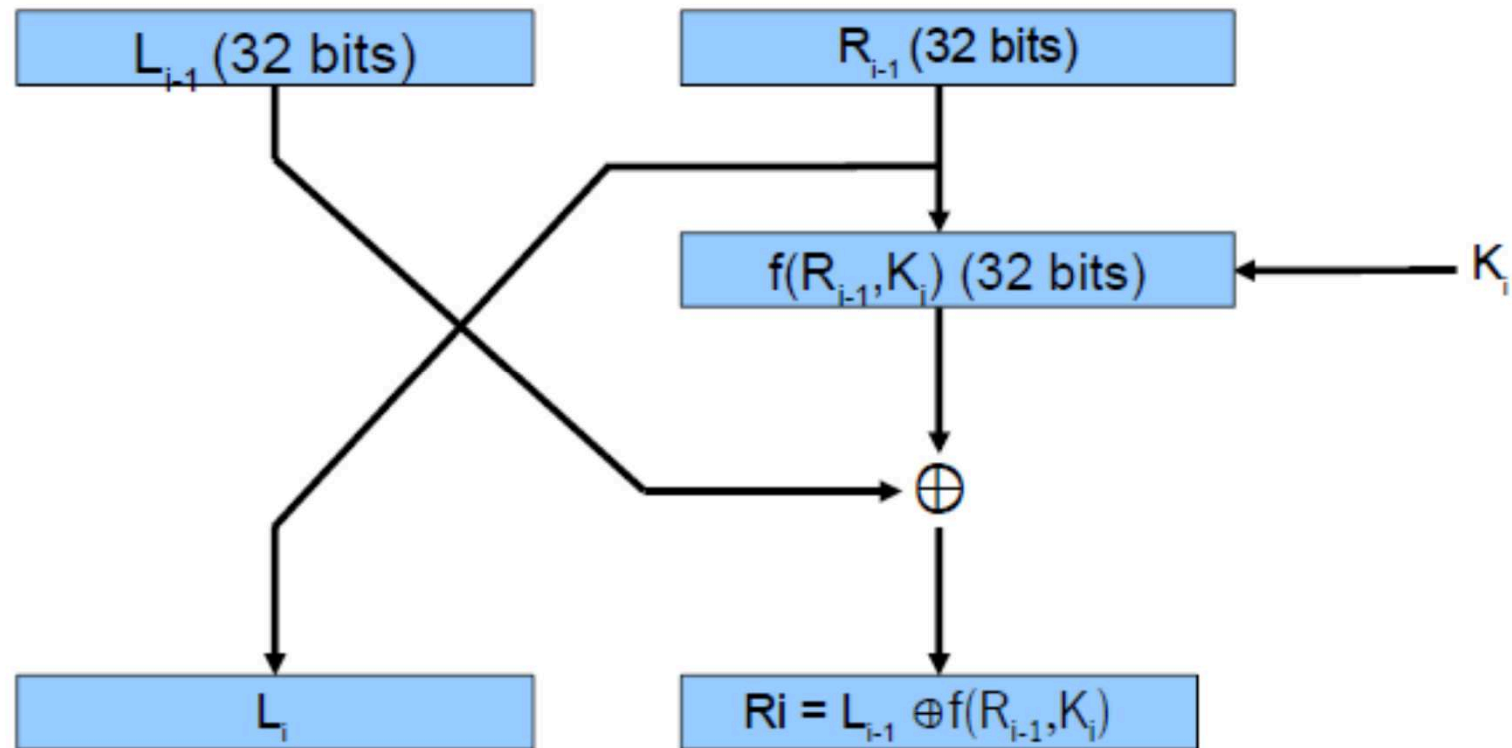
## Permutation initiale et finale (IP et IP<sup>-1</sup>)

- ❑ Les blocs de données subissent des permutations au début et à la fin du chiffrement.

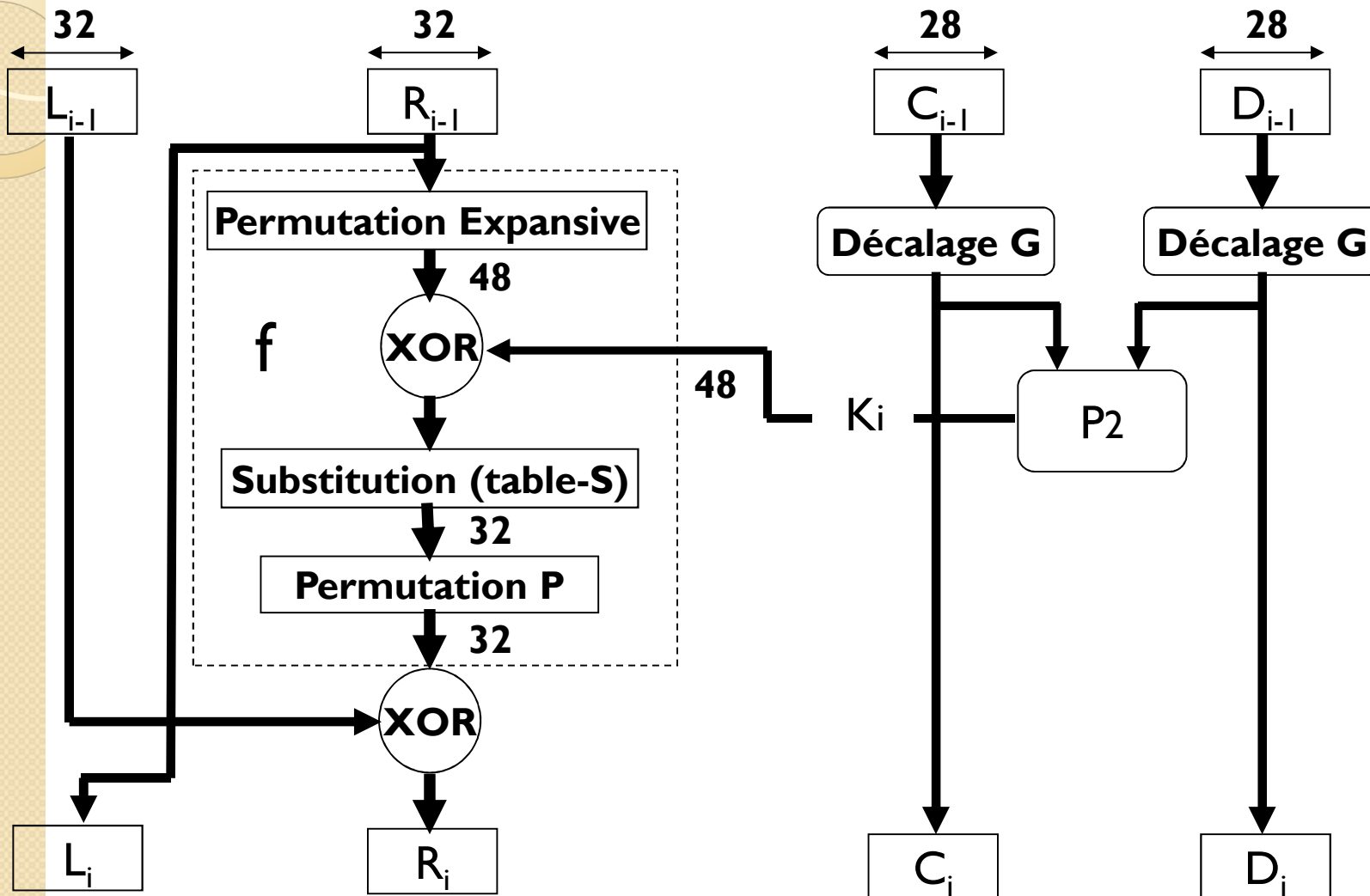
```
unsigned int    permutation_initiale[64] = {
    58, 50, 42, 34, 26, 18, 10, 2,  60, 52, 44, 36, 28, 20, 12, 4,
    62, 54, 46, 38, 30, 22, 14, 6,  64, 56, 48, 40, 32, 24, 16, 8,
    57, 49, 41, 33, 25, 17, 9,  1,  59, 51, 43, 35, 27, 19, 11, 3,
    61, 53, 45, 37, 29, 21, 13, 5,  63, 55, 47, 39,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

unsigned int    permutation_finale[64] = {
    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, 44, 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
};
```

# Principe d'une ronde

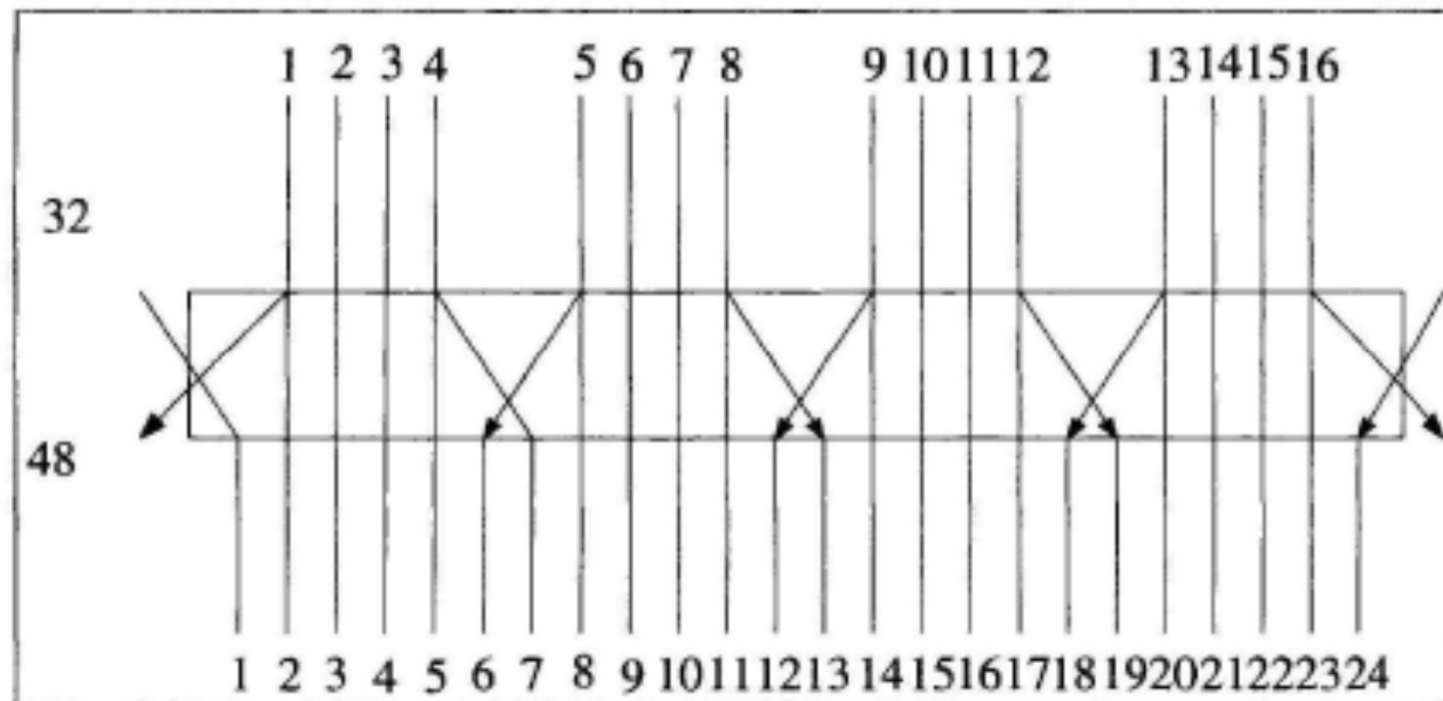


# Description d'une ronde DES



## Permutation expansive

- ❑ Duplique 16 bits parmi les 32 bits d'entrée
  - Obtenir autant de bits (48) que dans la clé  $K_i$



## Suite

- ❑ Pour chaque groupe de 4 bits le 1<sup>er</sup> et le 4<sup>e</sup> sont dupliqués

```
unsigned int permutation_expansive[48] = {  
    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  
};
```

# Substitution par tables-S

## □ 8 tables-S

### ■ Chaque table :

- 4 lignes x 16 colonnes
- Les éléments sont des valeurs sur 4 bits (<16)

```
unsigned int table_s[8][4][16] = {  
    {  
        1ère table {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}  
    },
```

## Substitution par table-S

- ❑ La permutation expansive fournit un mot de 48 bits (8 fois 6 bits)
- ❑ La substitution remplace ces 48 bits par 32 nouveaux bits
- ❑ Soit un groupe de 6 bits :  $b_1b_2b_3b_4b_5b_6$ 
  - $b_1b_6$  : nombre binaire compris en 0 et 3, correspond à la ligne
  - $b_2b_3b_4b_5$  : nombre binaire compris entre 0 et 15, correspond à la colonne

## Suite

### □ Exemple :

- $b_1b_2b_3b_4b_5b_6 = 100111_2$

- $b_1b_6 = 11_2 = 3_{10}$

- $b_2b_3b_4b_5 = 0011_2 = 3_{10}$

- La sortie est  $2_{10} = 0010_2$



# Permutation-P

- ❑ Permutation pure pour mélanger encore un peu...

```
unsigned int    permutation_pure[32] = {  
    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  
};
```

# Chiffrement et déchiffrement

## ❑ Chiffrement

- Les clés sont utilisées dans l'ordre où elles sont générées  $K_1, K_2, \dots, K_{16}$

## ❑ Déchiffrement

- Le même algorithme est utilisé, mais les clés sont utilisées en sens inverse (la ronde 1 utilise la clé  $K_{16}$ , la ronde 2 utilise la clé  $K_{15}$ , la ronde  $i$  utilise la clé  $K_{16-i+1}$ )

## Suite

- ❑ Le déchiffrement est effectué par le même algorithme en inversant l'ordre des clés:
  - Utiliser  $K_{16}$  pour la première itération
  - Utiliser  $K_1$  pour la dernière itération
  - La permutation finale est l'inverse de la permutation initiale

$$\begin{aligned} R_{i-1} &= L_i \\ L_{i-1} &= R_i \oplus f(L_i, K_i) \end{aligned}$$

# Caractéristiques de DES

## ❑ Résistance du DES

### ■ Recherche exhaustive:

- Nombre total d'essais  $2^{56} = 7,2 * 10^{16}$  clés
- Durée d'un essai = 1  $\mu$ seconde
- Temps nécessaire =  $7,2 * 10^{10}$  secondes
- Avec une machine parallèle à un million d'éléments,
- Temps nécessaire =  $7,2 * 10^4$  secondes (20 heures)
- Coût : 20 millions de dollars
- Clé de 56 bits trop courte
- Clé de 128 bits : DES virtuellement incassable

## Suite

### ❑ Résistance de DES

- 1998 : message décrypté en 56 heures
- 1999 : le temps tombe à 22 heures

## Résistance du DES

### ● Sensibilité aux erreurs

1 bit erroné dans C

1 bit erroné dans K

M	1000	0000	0000	0000
K	0030	0000	0000	0000
C	958E	6E62	7A05	557B
C	858E	6E62	7A05	557B
K	0030	0000	0000	0000
M	8D48	93C2	966C	C211
C	958E	6E62	7A05	557B
K	0010	0000	0000	0000
M	6D4B	9453	7672	5395

## Autres algorithmes

Algorithme	Taille clé	Rondes	Opérations
DES	56	16	XOR, Tables S fixes
Triple DES	112 ou 128	48	XOR, Tables S fixes
IDEA	128	8	XOR, Additions, Multiplications
Blowfish	Jusqu'à 448	16	XOR, Additions, Tables S variables
RC5	Jusqu'à 2048	Jusqu'à 255	XOR, Soustractions, Additions, Rotations



- ❑ Invention de la cryptanalyse différentielle par Biham et Shamir

➡ DES commençait à ne pas offrir de garantie de sécurité suffisante

- ❑ NIST (National Institute of Standards and Technologie) lance un appel d'offres pour remplacer DES
  - Un algorithme de chiffrement capable de protéger la confidentialité des informations pour le gouvernement américain et le secteur privé
  - AES est un algo de chiffrement à clé secrète par blocs dont la taille est 128 bits.