Sécurité Informatique

3éme licence informatique

Université de Jijel

F.BOUDJERIDA

Chapitre

Chiffrement moderne

❑Critères de construction d'un bon chiffrement

- Rappel: trouver Dk' inverse de Ek est **un problème difficile (NP-complet).**
- **Principe de Kerckhoffs (1883):**
  La difficulté ne doit pas dépendre du secret des algorithmes mais du secret des clés
- Un chiffrement doit être **stable**(on ne peut le changer que très rarement).
- Réalisation **simple et rapide** du chiffrement et du déchiffrement (pour atteindre des débits élevés).

❑Critères de construction d'un bon chiffrement

■ Une clé doit être choisie de **façon aléatoire** dans un espace de clés qui doit être très **grand**.=> Une clé doit être **imprévisible**.

■Un chiffrement dépend de clés qui doivent être **changées fréquemment et donc facilement =>** éviter un **encombrement important des clés** (pour le transport et le confort des utilisateurs).

■Les messages en clair doivent comporter **le moins de redondances possibles:**

Soit par **compression** préalable
Soit par **extension** en ajoutant des informations constituant du bruit

❑le point de vue de Shannon:

confusion et diffusion

■**Idée de confusion** La relation entre, d'une part le texte clair et la clé de chiffrement, et d'autre part le texte chiffré doit être aussi difficile que possible à établir.

- Le calcul de l'inverse est de complexité élevée.
- Principe présenté souvent comme réalisé par les substitutions.

■**Idée de diffusion** Une modification même mineure du texte en clair doit se traduire par une modification très importante du texte chiffré.

- Chaque symbole en clair doit être « diffusé » dans tout le texte chiffré (chaque symbole chiffré doit dépendre de beaucoup de symboles en clair).
- Principe présenté souvent comme réalisé par les permutations.

Chiffrement symétrique

Alice

Milieu non sécurisé

Bob

Utilisation de la même clé de chiffrement par les deux interlocuteurs

Chiffrement symétrique (à clé privée)

Les méthodes de chiffrement Ek et de déchiffrement Dk' sont liées du point de vue du secret des clés k et k'.

- La connaissance de k entraine celle de k' et réciproquement
- Pratiquement k = k'.
- La clé étant partagée entre l'émetteur et le destinataire, le secret de la clé doit être très bien gardé.
- Il faut pour N communicants potentiels N(N-1)/2 clés secrètes.

Chiffrement symétrique

2 grandes familles: blocs et flots

Chiffrement par blocs: les messages sont découpés en blocs

DES: blocs de 64 bits, clés de 56 bits

IDEA: blocs de 64 bits, clés de 128 bits

AES: blocs de 128 bits, clés de 128, 256 bits

...

Chiffrement par flots: les données sont traitées en flux

Pseudo-Vernam : bit par bit

RC4 : chiffrement octet par octet

...

Chiffrement par bloc

- Une des primitives (« briques ») les plus largement utilisées en cryptographie
- La longueur *n* des blocs et la taille *l* des clés sont deux caractéristiques des systémes de chiffrement par blocs.
- Le message clair *m* à chiffrer est découpé en blocs de *n* bits (typiquement 64, 128 ou 256 bits). $m = m1m2 \ldots mk$.
- Si la longueur du message n'est pas un multiple de la longueur d'un bloc, on le complète : c'est le bourrage ou padding en anglais. Plusieurs techniques de bourrage existent.

Chiffrement par bloc

Une technique de bourrage

- Une façon de bourrer (RFC2040) consiste à compléter le dernier bloc par autant d'octets que nécessaire, chaque octet ayant pour valeur le nombre d'octets ajoutés.
- Par exemple, s'il manque trois octets au message $m = o_1 o_2 o_3 o_4 o_5$ pour obtenir un bloc de huit octets, on ajoute trois octets égaux à 3

o_1	o_2	o_3	o_4	o_5	03	03	03

- S'il se trouve que la taille de la donnée à chiffrer est un multiple de la taille d'un bloc, on ajoute un bloc entier dont chaque octet a pour valeur la taille en octet d'un bloc.
- Par exemple, pour des blocs de huit octets, on ajoute le bloc

08	08	08	08	08	08	08	08

Chiffrement par bloc

Avant 1975: chiffrements artisanaux (Vigenère, Hill)

1975–2000: le DES (Data Encryption Standard), mais aussi FEAL, IDEA, RC5,...

2000–... AES (Advanced Encryption Standard), RC6, CAMELLIA,...

Construction de chiffrement par bloc

- Algorithmes itératifs : une fonction de tour est itérée t fois à fin de propager la diffusion et la confusion sur tout le bloc de chiffrement
- Génération de clés de tour (ou sous-clés) à partir de la clé secrète K
- Utilisation d'opérations simples et efficaces (+, XOR, *, tableaux)

Construction de chiffrement par bloc

Principe général

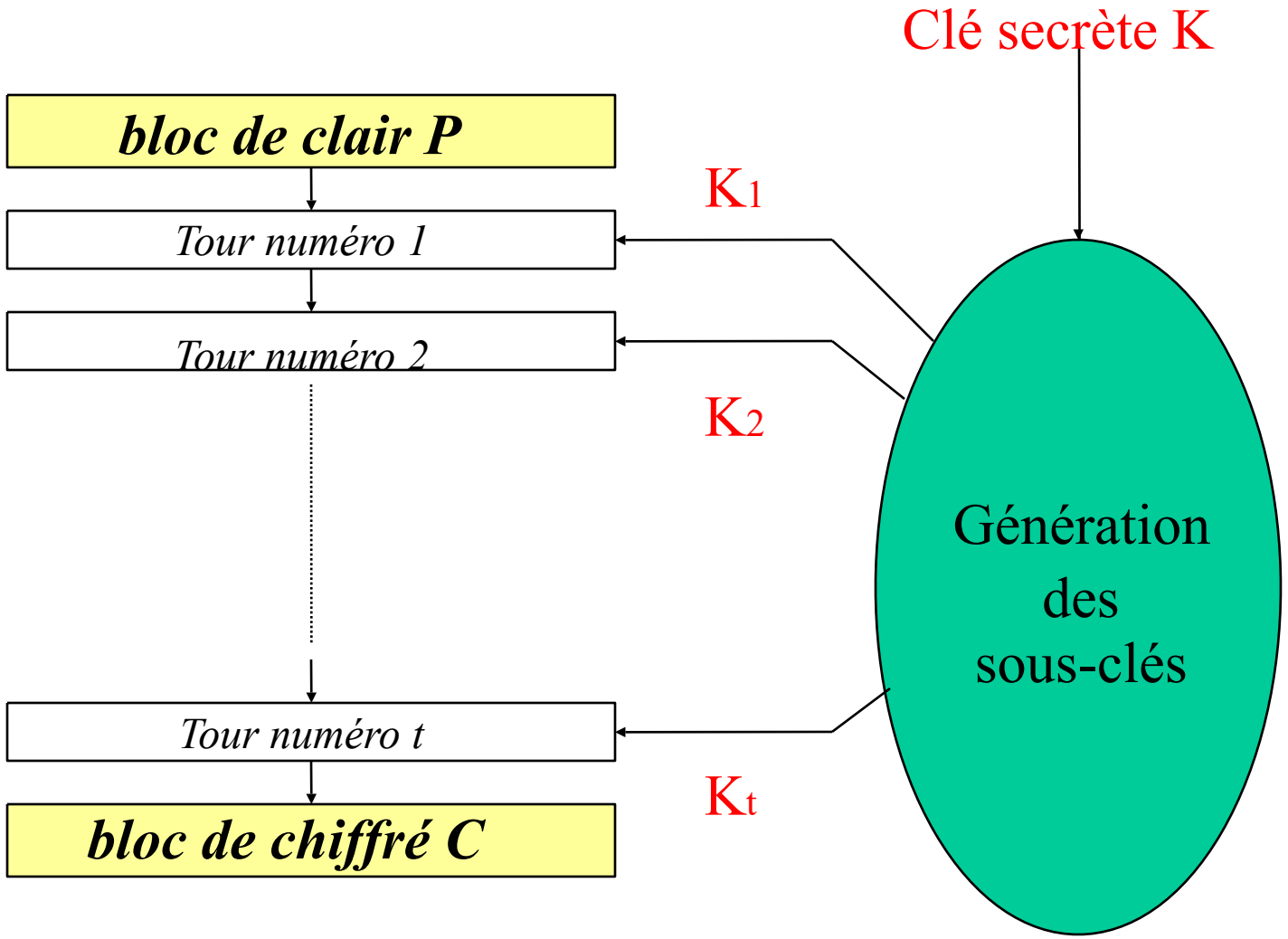
Chiffrement produit = empilement de tours, utilisant chacun une sous-clé

si E est une certaine fonction et M le message clair, l'idée est de calculer successivement

C1 := E(M);
C2 := E(C1);
: : :
Cr := E(Cr-1) :

- Construction iterative.
- Les sous-clés (ou clés de tours) sont dérivées de la clé K
- La fonction de tour est destinée à être optimisée.

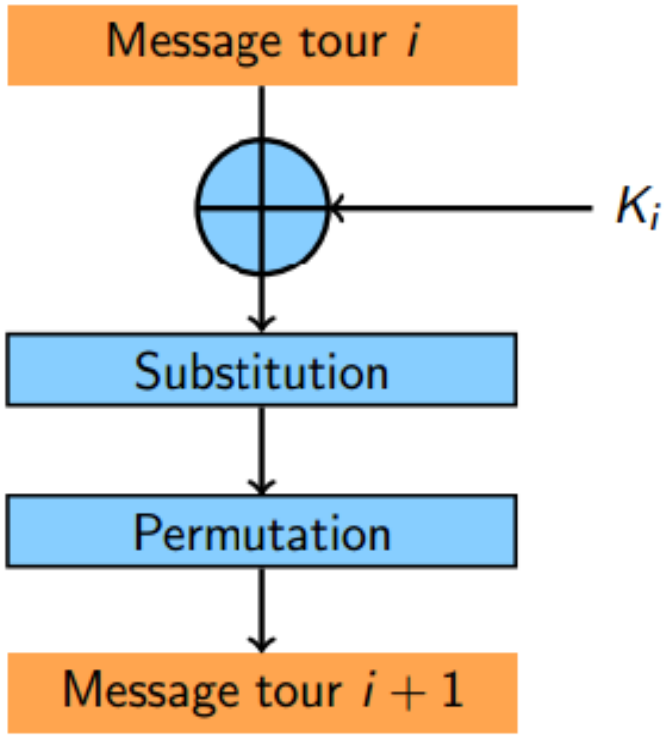
Construction de chiffrement par bloc



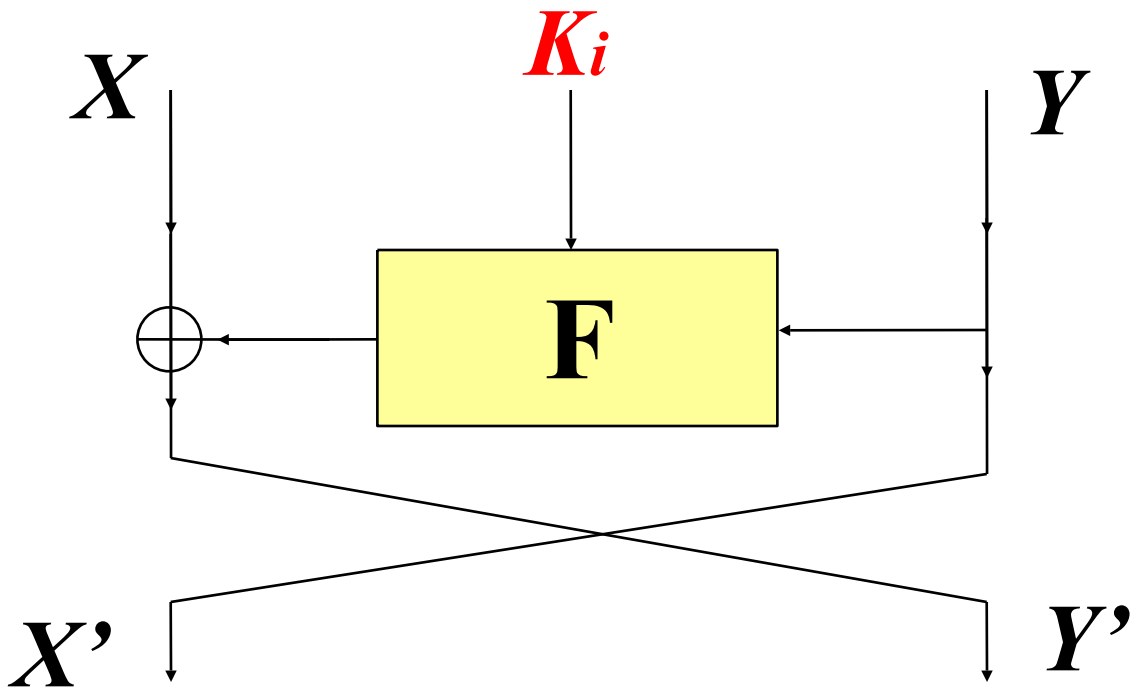
Construction de chiffrement par bloc

Décomposition d'un tour:

- Intervention de la sous-clé
- Couche de substitution
- Couche de permutation



Schémas de Feistel

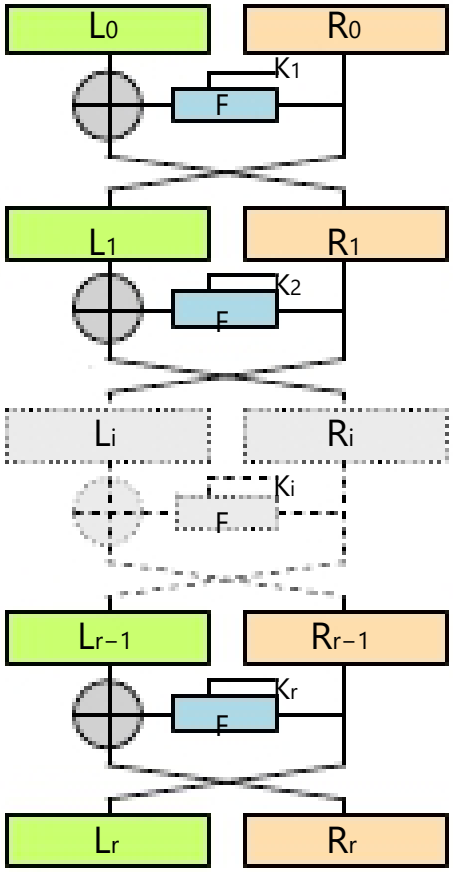


Fonction de tour inversible même si F ne l'est pas !!

Schémas de Feistel

Empilement de schéma de Feistel

Un tour isolé donne une bijection simpliste, la complexité croît de façon exponentielle avec le nombre de tours.



Les modes de chiffrement

- Que ce soit pour DES ou des cryptosystèmes symétriques plus récents comme IDEA ou AES ou pour des cryptosystèmes asymétriques comme RSA ou ElGamal les clés sont de longueur fixée. Les messages eux peuvent avoir une longueur arbitraire. Pour adapter la taille du message à celle de la clef on décompose le message par blocs de taille fixe correspondant aux tailles des clés que l'on chiffre ensuite un à un et que l'on envoie successivement.
- Pour cela quatre modes essentiels de chiffrement par blocs sont possibles: ECB, CBC, CFB et OFB.

Les modes de chiffrement

- **Objectifs des modes opératoires**

  - Ils ne concernent que le chiffrement par bloc.
  - Ils doivent masquer les blocs clairs identiques.
  - Deux blocs identiques chiffrés avec la même clé ne donnent pas les mêmes chiffrés si le reste du message diffère.

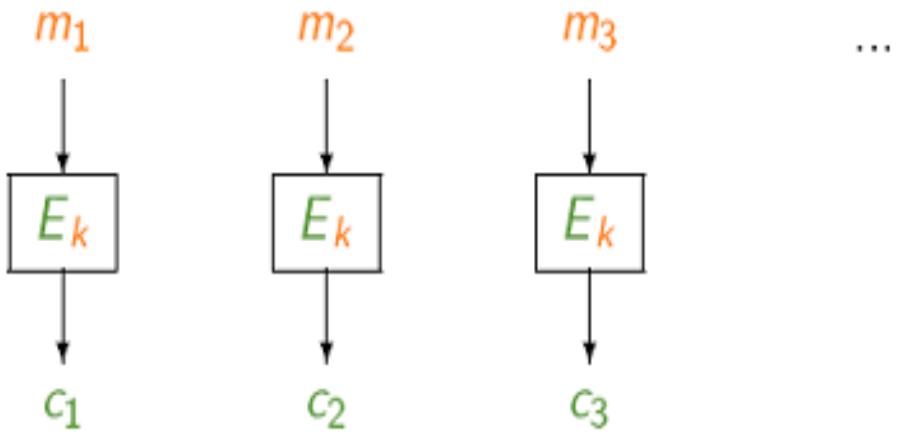
Mode ECB (Electronic Code Book)

Chiffrement : chaque bloc clair *mi* est chiffré indépendamment et donne un bloc chiffré *ci* = *Ek*(*mi*)

Déchiffrement : chaque chiffré est déchiffré indépendamment pour donner le clair correspondant *mi* = *Dk*(*ci*) .

Conséquence :

- deux blocs clairs identiques donnent toujours le même bloc chiffré pour une clé *k* fixée.
- Aucune sécurité, pas d'utilisation



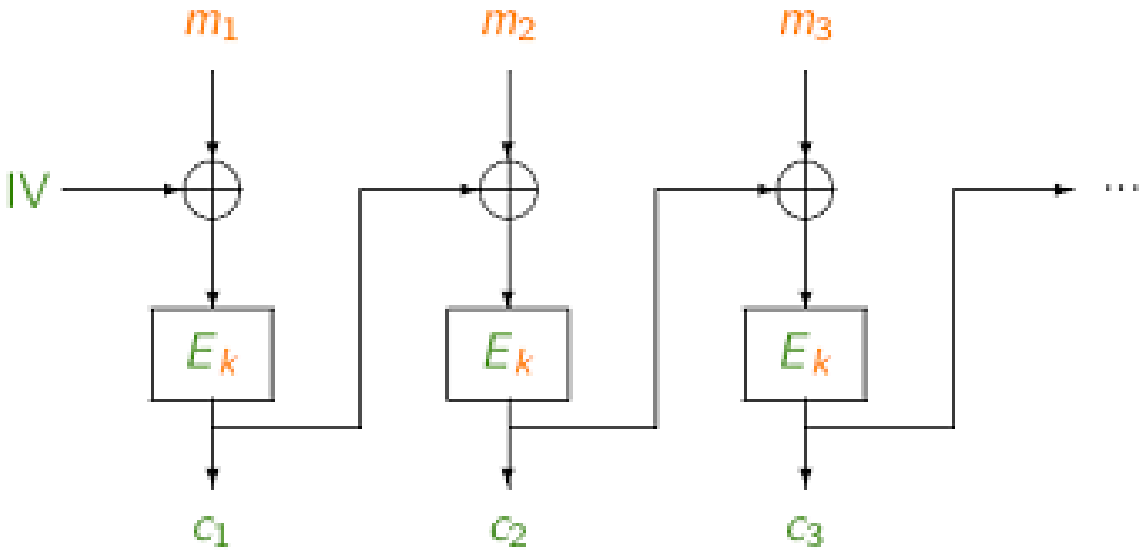
Mode CBC (Cipher Bloc Chaining)

Chiffrement : un vecteur d'initialisation *IV* est généré aléatoirement.
$ci = Ek(mi \oplus ci - 1)$. Le vecteur *IV* est transmis avec les blocs chiffrés.

Déchiffrement : $mi = Dk(ci) \oplus ci - 1$.,

Conséquence :

- deux blocs clairs identiques chiffrés différemment.
- Mode le plus utilisé



Fonction $\oplus$ dans $\mathbb{Z}_{26}$

$\oplus$ *est la fonction* XOR binaire appliqué sur les représentations entières, alors :

$$\text{OXR}_{26}(a, b) = (a \oplus b) \bmod 26$$

où : $a, b \in \mathbb{Z}_{26}$ donc $0 \leq a, b \leq 25$

$\oplus$ est l'opérateur XOR bit à bit.

Exemple de calcul

Prenons $a = 24$ et $b = 12$:

En binaire (sur 5 bits, car $24_{10} = 11000_2$

$$a = 11000$$
$$b = 01100$$

XOR bit à bit :$11000 \oplus 01100 = 10100$ (binaire)

Conversion en décimal :

$$10100_{\,2} = 20$$

Modulo 26 :

$$20 \bmod 26 = 20$$

Donc : $\text{OXR}_{26}(24, 12) = 20$

Fonction $\oplus$ dans $\mathbb{Z}_{26}$

Propriétés principales

- **Commutativité** $A \oplus B = B \oplus A$

L'ordre des opérandes n'affecte pas le résultat.

- **Associativité** $(A \oplus B) \oplus C = A \oplus (B \oplus C)$

Cela permet d'appliquer l'opération à plusieurs bits sans tenir compte de l'ordre des parenthèses.

- **Élément neutre** $A \oplus 0 = A$

Le bit 0 n'affecte pas le résultat.

- **Auto-inverse** $A \oplus A = 0$

Tout bit XOR lui-même donne toujours 0.

- **Utilisation en annulant un bit** $(A \oplus B) \oplus B = A$

Cette propriété est très utilisée en cryptographie et en codage, car elle permet de récupérer un bit original après un XOR avec une clé ou un masque.

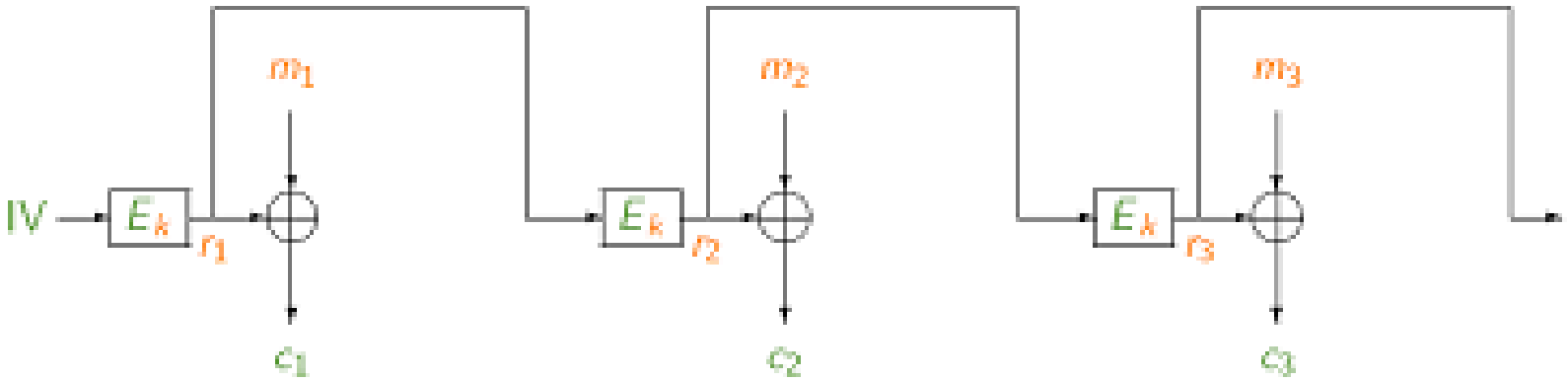
Mode OFB (Output FeedBack)

Chiffrement : un vecteur d'initialisation *IV* est généré aléatoirement.
ci = *ri* ⊕ *mi* , o`u *r*0 = *IV* et pour *i* ≥ 1 , *ri* = *Ek*(*ri* − 1). Le vecteur *IV* est transmis avec les blocs chiffrés.

Déchiffrement : *mi* = *ci* ⊕ *ri* ; *ri* = *Ek*(*ri* − 1)

Conséquence :

- deux blocs clairs identiques chiffrés différemment.
- Remarque : c'est du chiffrement à flut.
- Totalement symétrique
- Moins de câblage
- Utilisé dans les satellites



Mode CFB (Cipher FeedBack)

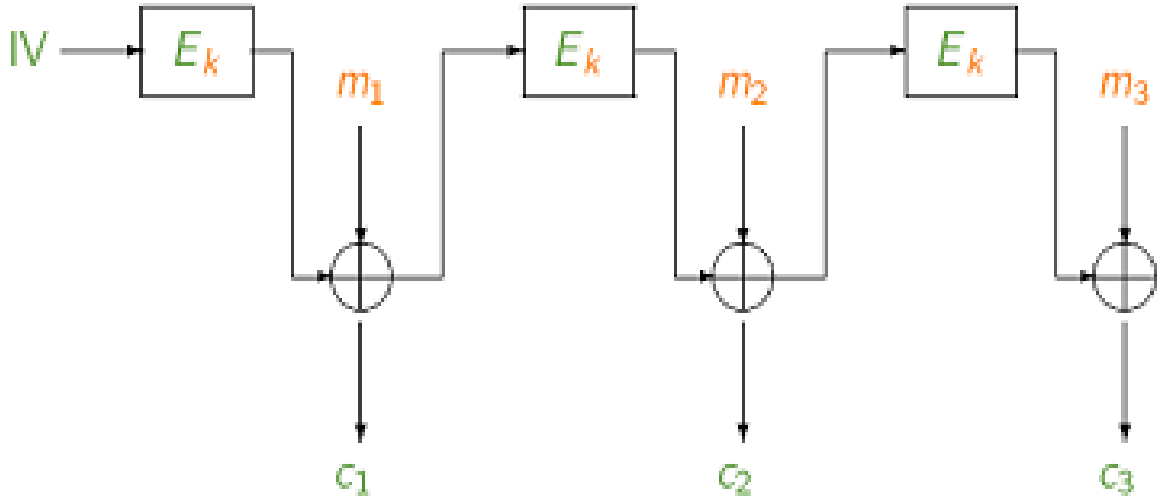
Chiffrement : un vecteur d'initialisation *IV* est généré aléatoirement. Le vecteur *IV* est transmis avec les blocs chiffrés

$$Ci = Ek(Ci - 1) \oplus mi$$

Déchiffrement : $$mi = Ci \oplus Ek(Ci - 1)$$

Conséquence :

- deux blocs clairs identiques chiffrés différemment.
- Moins sûr, parfois plus rapide
- Utilisé dans les réseaux



DES: Data Encryption Standard

- En 1970, le NBS (*National Board of Standards*) lance un appel d'offre au *Federal Register* pour définir un algorithme de cryptage ayant un niveau de sécurité élevé qui ne dépend pas de la confidentialité de l'algorithme mais seulement des clés secrètes, qui fait intervenir des clés secrètes pas trop grandes, rapide, robuste, bon marché, facile à implémenter.

Le DES a été approuvé en 1978 par le NBS

DES: Data Encryption Standard

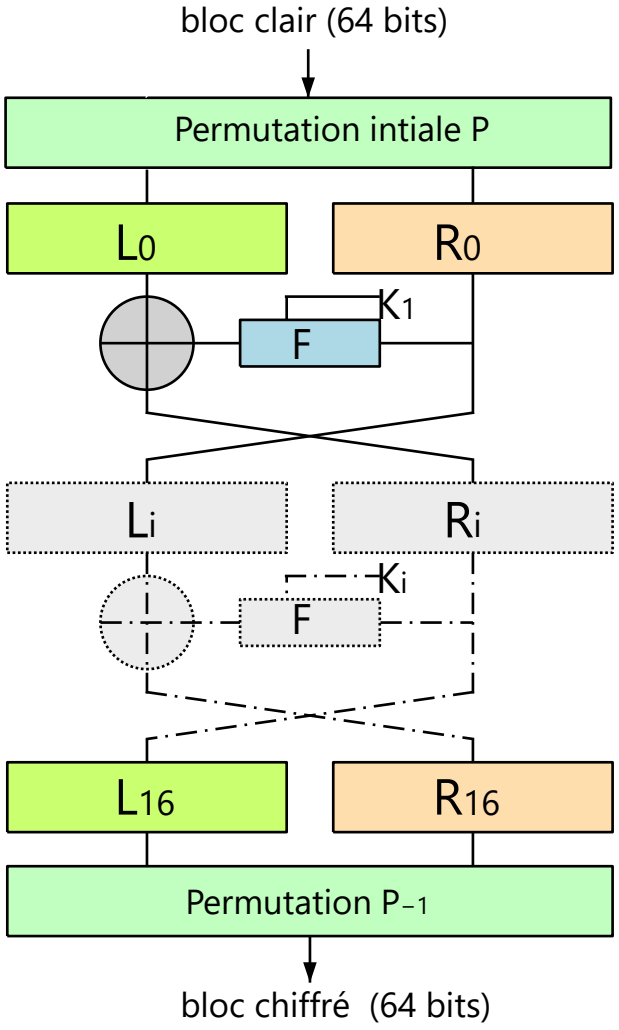
- Algorithme de chiffrement par blocs
- Utilise une clé de 56 bits et des blocs de 64 bits
- Feistel 16 tours avec des sous-clés de 48 bits

DES: Data Encryption Standard

- Le texte est découpé en blocs de *64* bits
- Les blocs sont permutés
- Ils sont coupés en deux: droite et gauche
- On effectue *16* fois un cycle de permutations et de substitutions faisant intervenir la clé secrète
- On regroupe les parties gauche et droite puis on effectue les permutations inverses.

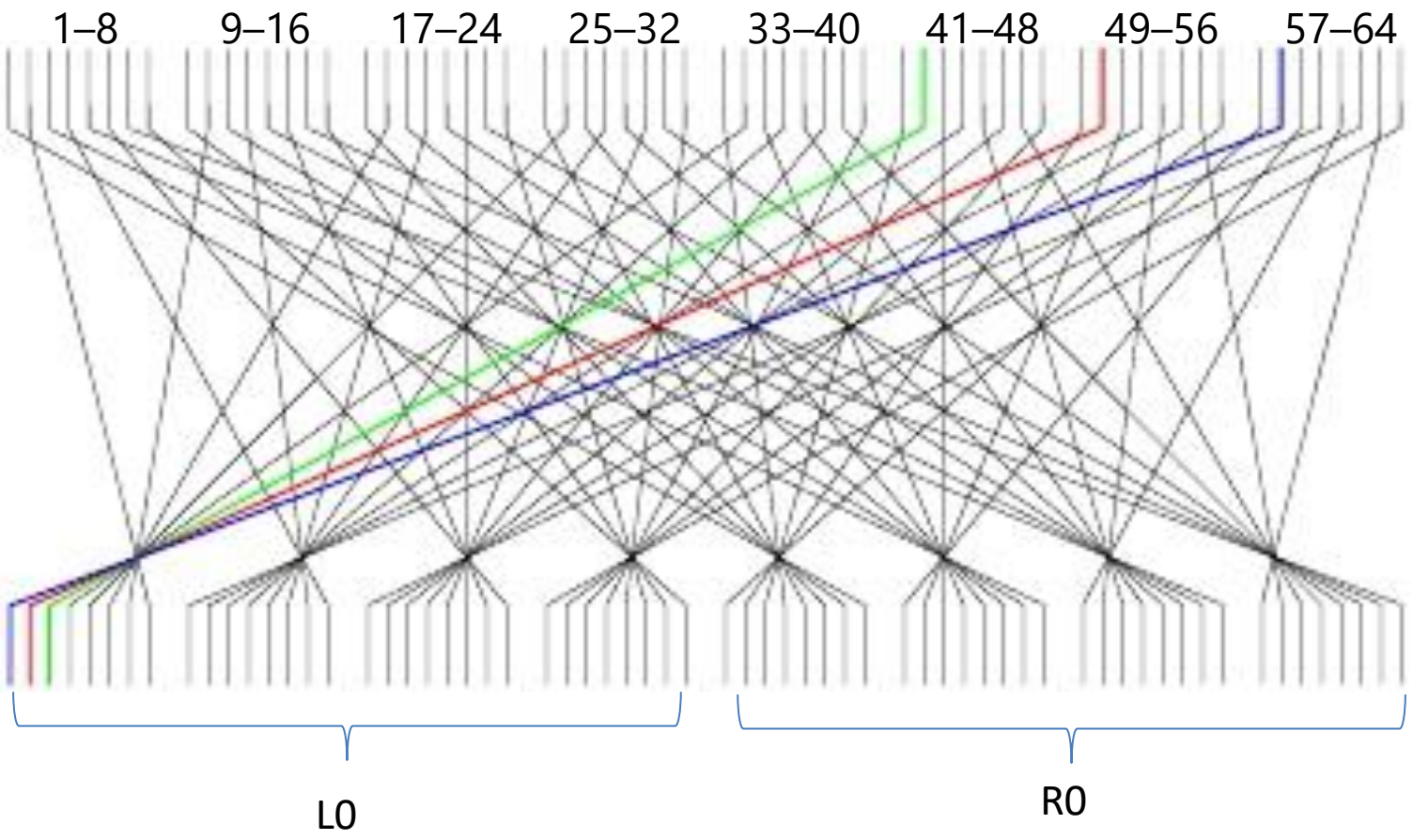
DES: Data Encryption Standard

La complexité du cryptage repose dans le choix de la fonction f. La fonction f
f(Ri−1,Ki) = P(S(E(Ri−1) Xor Ki))



DES: (Permutation Initiale)

bloc clair (64 bits



DES: (Permutation Initiale)

L0

R0

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

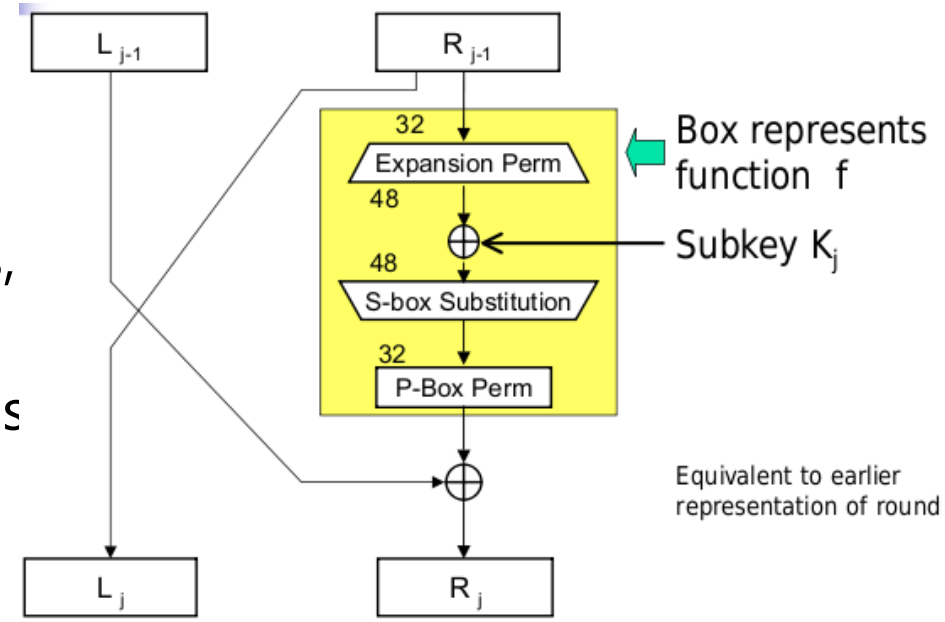
Le bit numéro 21 de la sortie
…
provient du bit numéro 30 de l'entrée

Une fois la permutation initiale réalisée, le bloc de 64 bits est scindé en deux blocs de 32 bits, notés respectivement *G* et *D* (pour gauche et droite). On note *G*0 et *D*0 l'état initial de ces deux blocs

DES: (Fonction de tour)

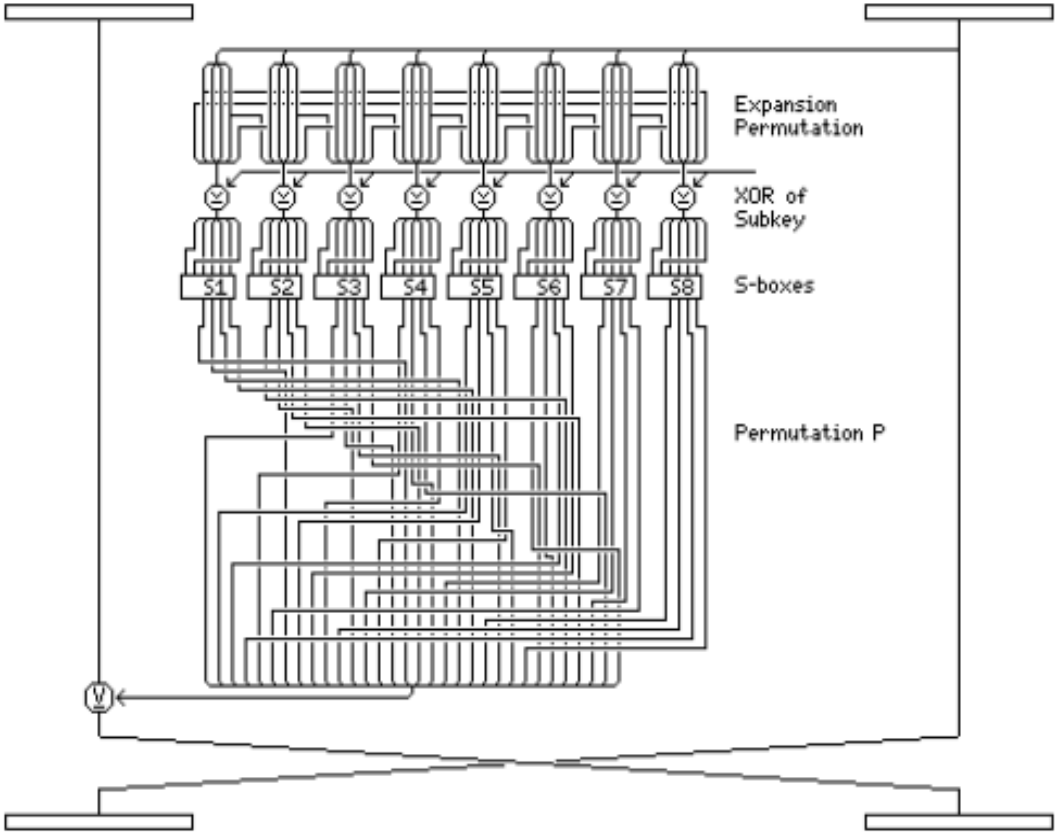
Dans le DES, la fonction f est une fonction de 32 bits vers 32 bits, constituée:

- d'une expansion de message de 32 vers 48 bits
- d'un XOR avec 48 bits dérivés de la clé
- de la concatenation de 8 sous-fonctions de 6 vers 4 bits, appelées boîtes S
- d'une permutation des 32 bits s



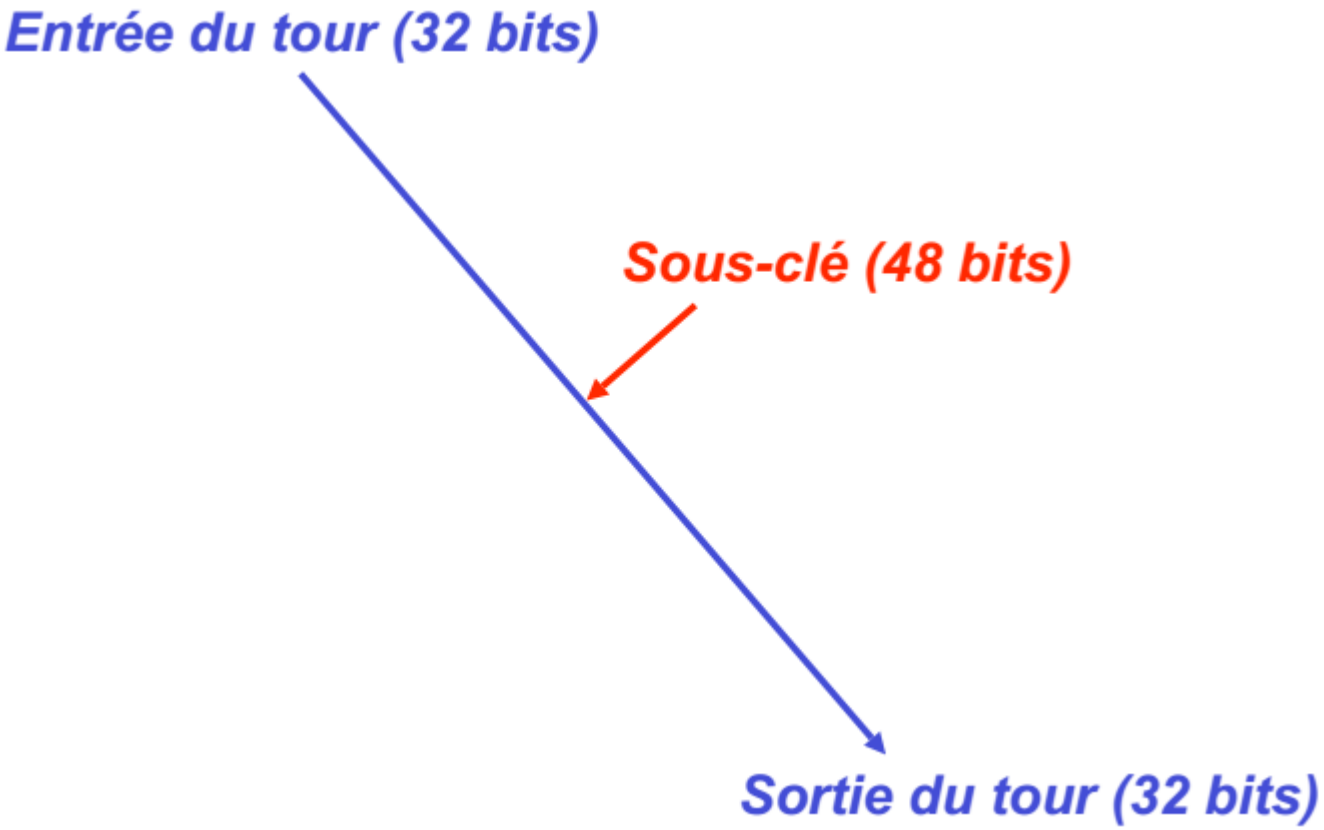
f(Ri−1,Ki) = P(S(E(Ri−1) Xor Ki))

DES: (Fonction de tour)



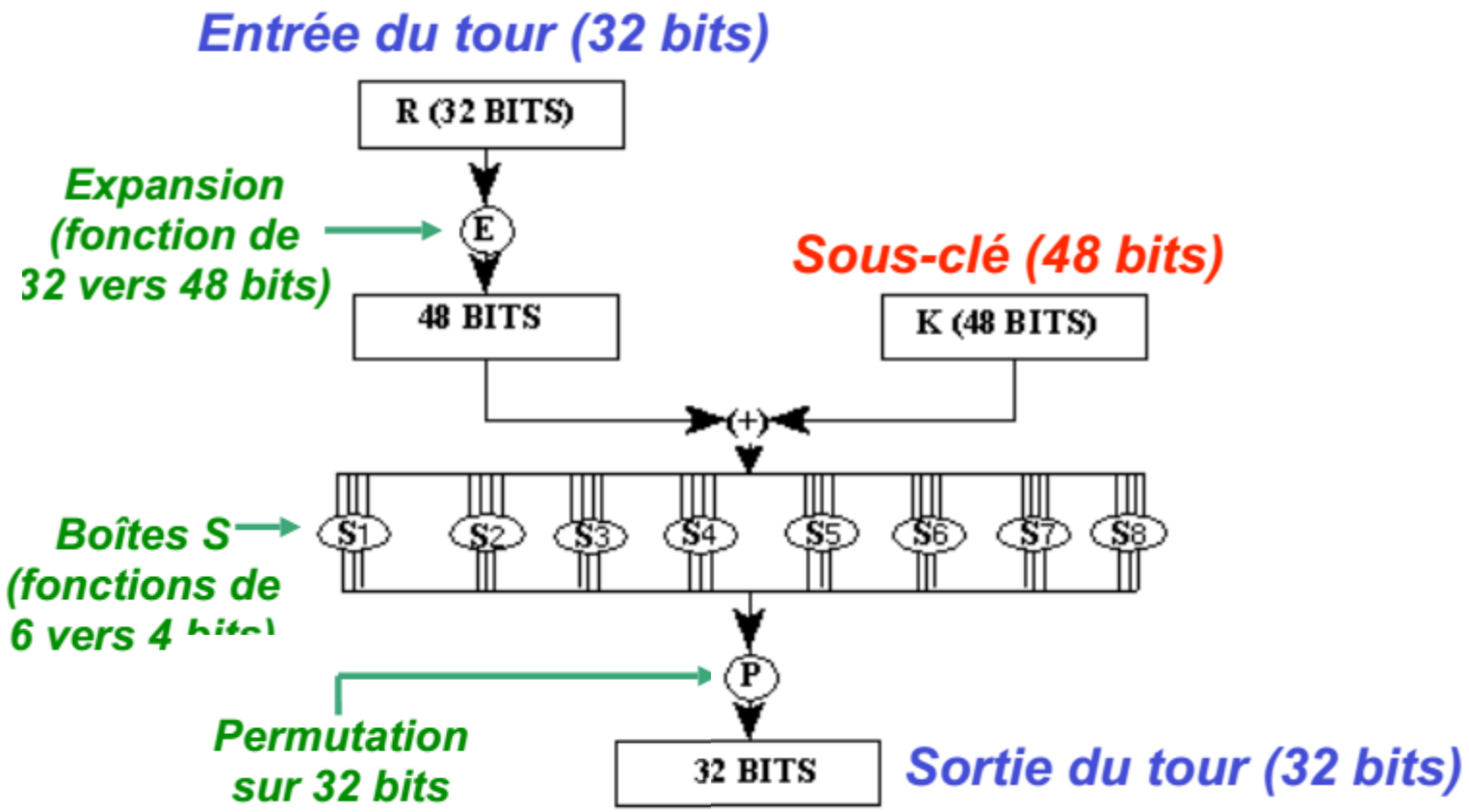
f(Ri−1,Ki) = P(S(E(Ri−1) Xor Ki))

DES: (Fonction de tour)



f(Ri−1,Ki) = P(S(E(Ri−1) Xor Ki))

DES: (Fonction de tour)



f(Ri−1,Ki) = P(S(E(Ri−1) Xor Ki))

DES: (Fonction de tour)

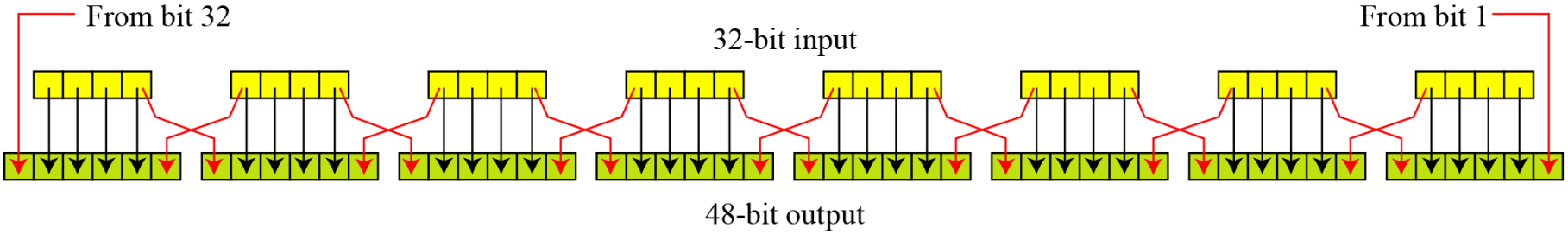
QUELQUES REMARQUES

- La fonction f n'a pas besoin d'être inversible (elle ne l'est pas)
- Expansion: certains bits sont dupliqués
- L'ajout de la sous-clé (un simple"xor") se fait sur 48 bits
- Les boîtes S assurent la non-linéarité (6 bits → 4 bits, 8 fois)
- La permutation P contribue à la diffusion (en plus du "croisement"dans le Feistel)

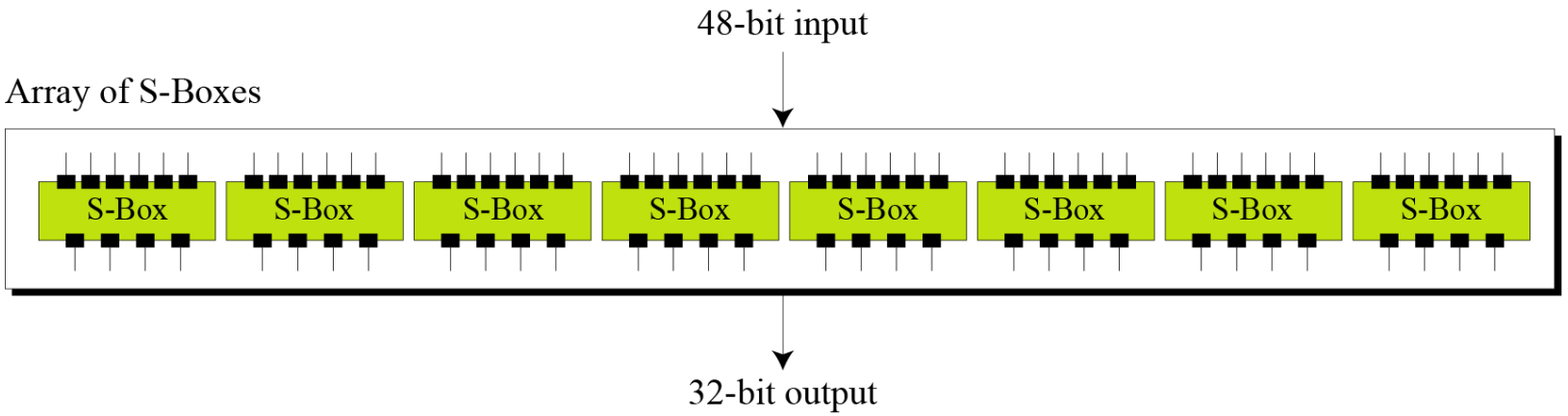
Expansion (32 → 48)

Les 32 bits de R0 entrent dans une table de sélection de bits, ils sont mélangés et répétés. On obtient 48 bits

32	01	02	03	04	05
04	05	06	07	08	09
08	09	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	31	31	32	01



DES: (Boîtes S)


48-bit input
Array of S-Boxes
S-Box
S-Box
S-Box
S-Box
S-Box
S-Box
S-Box
S-Box
32-bit output

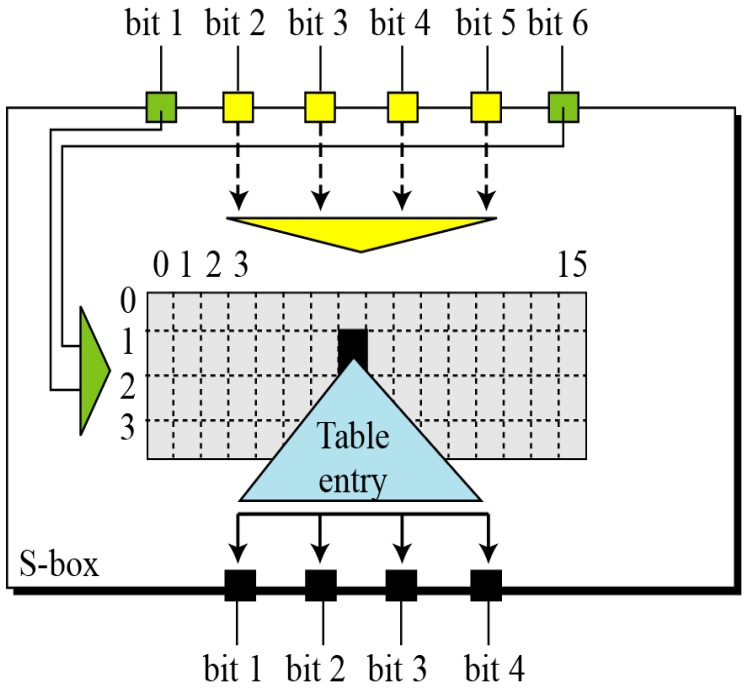
DES: (Boîtes S)

Le résultat précédent est décomposé en 8 blocs de 6 bits chacun (b1, b2, b3,b4, b5, b6)

– Cette décomposition permet de calculer une position dans une table de sélection **S** à 8 blocs de 16 colonnes et 4 lignes

• Les premiers et derniers bits de chaque *D0i* détermine (en binaire) la ligne de la fonction de sélection, les autres bits (respectivement 2, 3, 4 et 5) déterminent la colonne.

• La sélection de la ligne se faisant sur deux bits, il y a 4 possibilités (0,1,2,3). La sélection de la colonne se faisant sur 4 bits, il y a 16 possibilités (0 à 15).



– On substitue au bloc de 8 bits le bloc de 4 bits trouvé dans la table => total de 32 bits pour les 8 blocs

DES: (Boîtes S)

$S_1(101110) = (11)_{10} = (1011)$

10 → S_1	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
S_2	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
S_3	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
S_4	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

DES: (Boîtes S)

Boîte S: tableau 64 entrées, à valeurs dans 16

$S_1(110101) = 1001 = 9$ 1010

	S_1	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
11 →		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
	S_2	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
	S_3	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
	S_4	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

DES: (Boîtes S)

S_5	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

S_6	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

S_7	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

S_8	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

DES: Caractéristiques (Boîtes S)

- Propriétés cryptanalytiques intéressantes
- Non linéaire (substitution différente de César) : pas d'attaque simple
- Spécialement conçues pour contrer la cryptanalyse différentielle
- Même de très petites modifications des S -box peuvent affaiblir considérablement le chiffrement A la base de controverse autour de DES
- secret entourant la génération de $\{Si\}1 \leq i \leq 8$ et de P

DES: (OU Exclusif et itération)

- L'ensemble de ces résultats en sortie de *P* est soumis à un OU Exclusif avec le *I*0 de départ (comme indiqué sur le premier schéma) pour donner *R*1 *,* tandis que le *R*0 initial donne *L*1 *.*
- L'ensemble des étapes précédentes (rondes) est réitéré 16 fois.

DES: Permutation initiale inverse

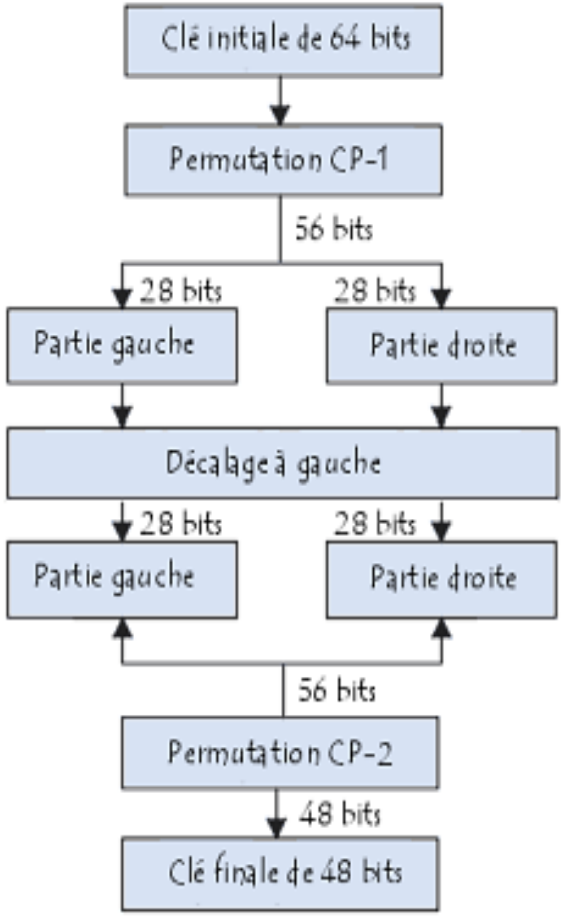
A la fin des itérations, les deux blocs *G*16 et *D*16 sont recollés, puis soumis à la permutation initiale inverse

PI^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	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

DES: Génération des clés

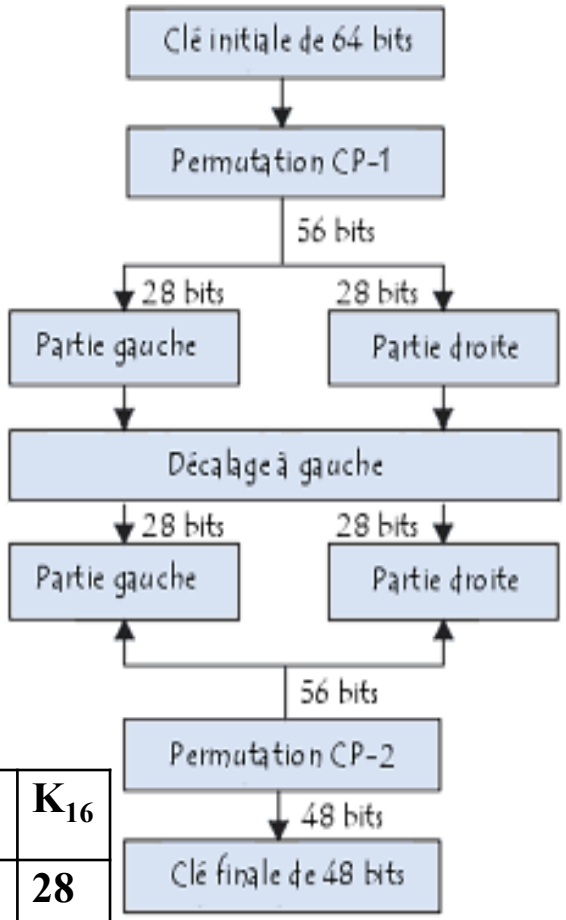
Etant donné que l'algorithme du DES présenté ci-dessus est public, toute la sécurité repose sur la complexité des clés de chiffrement.

L'algorithme ci-après montre comment obtenir à partir d'une clé de 64 bits (composé de 64 caractères alphanumériques quelconques) 8 clés diversifiées de 48 bits chacune servant dans l'algorithme du DES



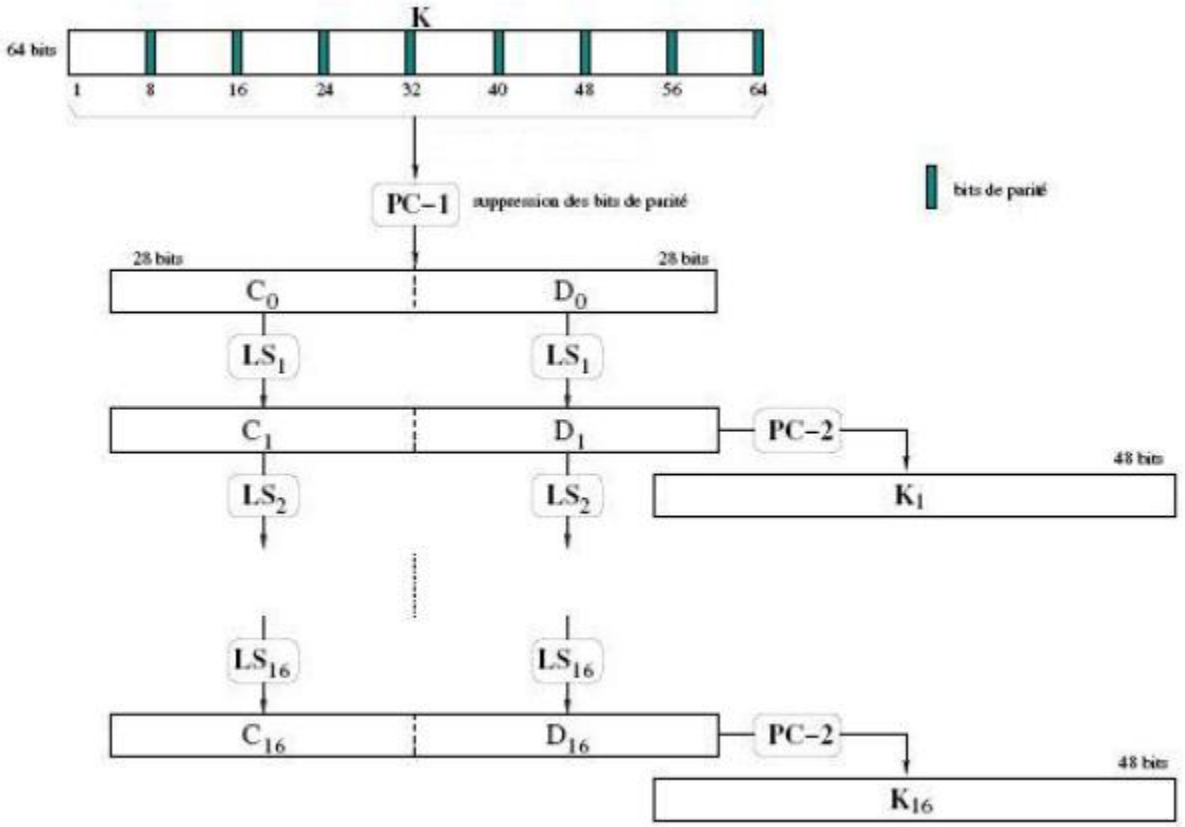
DES: Génération des clés

1. La 1ère étape consiste à éliminer les bits de parité pour ainsi obtenir une clé de 56 bits.
2. Appliquer une fonction de permutation initiale notée CP-1 sur l'ensemble des bits.
3. Séparer le bloc ainsi obtenu en deux sous blocs notés G_0 et D_0 chacun de 28 bits
4. Appliquer un décalage à gauche (rotation à gauche) sur G_0 et D_0.
5. Rassemblement des deux sous blocs pour construire un seul bloc de 56 bits.
6. Soumettre le bloc à une permutation finale notée CP-2 pour obtenir la 1ère sous-clé K_1.
7. Pour les autres sous-clés réitérer les étapes de 2 à 5 n fois (n est donné ci-dessous):



K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8	K_9	K_{10}	K_{11}	K_{12}	K_{13}	K_{14}	K_{15}	K_{16}
1	2	4	6	8	10	12	14	15	17	19	21	23	25	27	28

DES: Génération des clés



avec PC-1 et PC-2 des permutations (tables) et LSi une permutation circulaire de 2 bits vers la gauche (ou 1 bit si i = 1; 2; 9; 16)

Table PC-1

57	49	41	33	25	17	09	01
58	50	42	34	26	18	10	02
59	51	43	35	27	19	11	03
60	52	44	36	63	55	47	39
31	23	15	07	62	54	46	38
30	22	14	06	61	53	45	37
29	21	13	05	28	20	12	04

DES: Génération des clés

Cette matrice peut en fait s'ecrire sous la forme de deux matrice Gi et Di (pour gauche et droite) composees chacune de 28 bits :

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

- On note G0 et D0 le resultat de cette premiere permutation
- Ces deux blocs subissent ensuite une rotation a gauche, de telles facons que les bits en seconde position prennent la premiere position, ceux en troisieme position la seconde, ...
- Les bits en premiere position passent en derniere position.

DES: Génération des clés

Les 2 blocs de 28 bits sont ensuite regroupes en un bloc de 56 bits. Celui-ci passe par une permutation, notee PC-2; fournissant en sortie un bloc de 48 bits, representant la cle Ki .

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

Les bits 9, 18, 25, 35, 38, 43, 54 sont abandonnés : c.est pour cette raisons que la permutation est compressive.

DES: Commentaires

Ces choix peuvent paraître arbitraires mais :

- Toutes les briques sont très simples à coder et efficaces en hardware
- Les boîtes S apportent la non-linéarité
- Expansion et permutation garantissent une diffusion rapide

Test de robustesse du DES

- « casser » le cryptage DES
- 1997, ordinateur développé spécialement, le « DES cracker » + un réseau mondial d'env. 100 000 PC
- 245 milliards de clés par seconde
- DES forcé en 22 heures 15 minutes

DES: **Améliorations**

- **Faiblesse du DES :** la longueur de clé limitée à 56 bits.
- **Solution possible :** construire un sur-chiffrement à partir du DES en appliquant plusieurs fois de suite le DES.
- **Solution 1 : Le double DES (le 2DES)**

  2DESk1k2 (M) = DESk2 (DESk1 (M))

  Difficulté 2DES : pas vraiment meilleur que le DES.
- **Solution 2 :** Le triple DES (le 3DES ou TDES ici en mode EDE)

  3DESk1k2k3 (M) = DESk3 (DESk2 -1 (DESk1 (M)))

  Avantage : meilleure sécurité grâce à la clé de 168 bits**.**

  Inconvénient : 3DES prend trois fois plus de temps que DES.
- **Conclusion :** Quand les performances le permettent 3DES est recommandé.

DES: **Sécurité**

- **Le chiffre le plus controversé et le plus attaqué.**
- **Nombreuses attaques imaginées à propos du DES**

  A texte choisi : Pré-calcul exhaustif.

  Cryptanalyse linéaire.

  Cryptanalyse différentielle (le DES a peut-être été conçu pour parer ces attaques).
- **En 1996 bilan des moyens d'attaque.**
- **Finalement le DES est considéré comme un bon chiffre.**
- **Mais : clé 56 bits => possibilité des attaques force brute**

  Avec 7,5 millions d'euros on cassait le DES 56 bits en 6 minutes.
- **L'attaque du DES est à la portée de plus en plus de monde.**

DES: **Conclusion**

DES a été pendant 25 ans le principal chiffre à clés secrètes.

- **1976 : DES est certifié (standard officiel du gouvernement américain).**
- **1981 : DES devient un standard pour le secteur privé;**
- **1983 : DES est à nouveau certifié comme standard;**
- **1987 : La NSA américaine ne veut plus certifier le DES, et propose à** la place une série d'algorithmes de remplacement, tenus secrets
- **1988 : Suite à de longues discussions, le DES est finalement recertifié** pour cinq ans, avec la promesse que ce sera pour la dernière fois;
- **1993 : Aucune alternative n'étant disponible, DES est recertifié pour** cinq ans.

DES: **Conclusion**

Problème posé des le départ: la taille de la clé : 56 bits.

- En 25 ans les **méthodes de cryptanalyse ont fait des progrès.**
- Une **recherche exhaustive peut être faite à condition d'en mettre le** prix => **ne plus utiliser le DES sauf pour un chiffrement contre des attaques à petits moyens.**
- **Lancement d'un concours en 1997 pour le successeur du DES.**