Cryptographie

Cours 2

Chiffrement Symétrique

Jérémy Briffaut
STI 2A

Plan

I. Histoire, définition et objectifs de la cryptographie
   - Concepts et algorithmes de permutation et de substitution

II. Chiffrement Symétrique
   - DES, 3DES, AES, IDEA

III. Chiffrement Asymétrique
   - RSA, ElGamal

IV. Signature, Hachage et Scellement

V. Echange de clés
   - Algorithme Deffie-Hellman

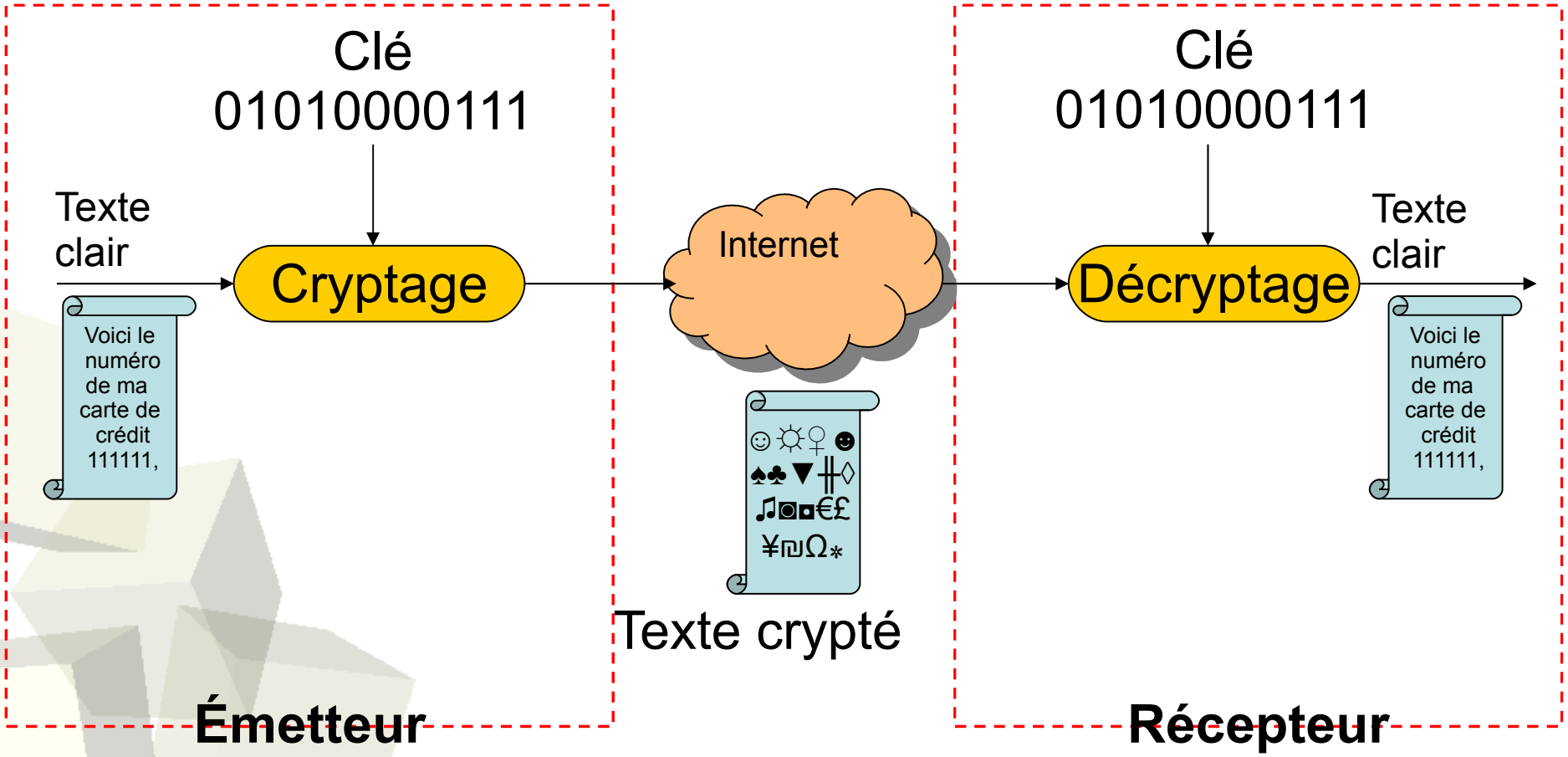
VI. Hachage : MD5, SHA-1, SHA-2

VII. Code d'Authentification & MAC

II. Confidentialité et algorithmes de chiffrement

- Différents types d'algorithmes :
  - Algorithmes symétriques ou à clef secrète
    - → Plus rapides donc préférés pour le chiffrement de données
  - Algorithmes asymétriques ou à clef publique
    - → Échange de clefs secrètes
    - → Signature

- Chiffrement symétrique – Principe
  - Clef de *chiffrement* = clef de *déchiffrement*
    - ➔ elle doit rester secrète



Informatisation de ...

- Alphabet binaire : $\mathcal{A} = \{0,1\}$.
- Espace des messages (en clair) :

  $\mathcal{M} = \{m=(m_1 \ldots m_n) \mid \forall\ i,\ m_i \in \mathcal{A}\ et\ m\ a\ un\ sens\}$.
- Espace des (messages) chiffrés :

  $\mathcal{C} = \{c=(c_1 \ldots c_n) \mid \forall\ i,\ c_i \in \mathcal{A}\}$.
- Espace des clés :

  $\mathcal{K} = \{k=(k_1 \ldots k_n) \mid \forall\ i,\ k_i \in \mathcal{A}\}$.

… la cryptographie symétrique

- Fonctions de chiffrement :

$$E : \mathcal{M} \times \mathcal{K} \rightarrow \mathcal{C}.$$
$$E(..,k) = E[k](..) = E_k(..).$$

- Fonctions de déchiffrement :

$$D : \mathcal{C} \times \mathcal{K} \rightarrow \mathcal{M}$$
$$D(..,k) = D[k](..) = D_k(..)$$
$$\forall\, m \in \mathcal{M},\ \forall\, k \in \mathcal{K},\ D_k(\, E_k(m)\,) = m$$

Fonctions de base

- Transposition sur alphabet binaire :
  - Confusion sur la syntaxe du message.
- Substitution sur alphabets binaires :
  - Confusion sur l'alphabet du message.
- Combinaison transposition/substitution :
  - Diffusion de la confusion syntaxique et alphabétique sur l'ensemble du message.

Outils informatiques

- Transposition :
  - Permutation.
- Substitution :
  - Tableaux indexés.
  - Opérations binaires : XOR, ...

Algorithmes de chiffrement symétriques

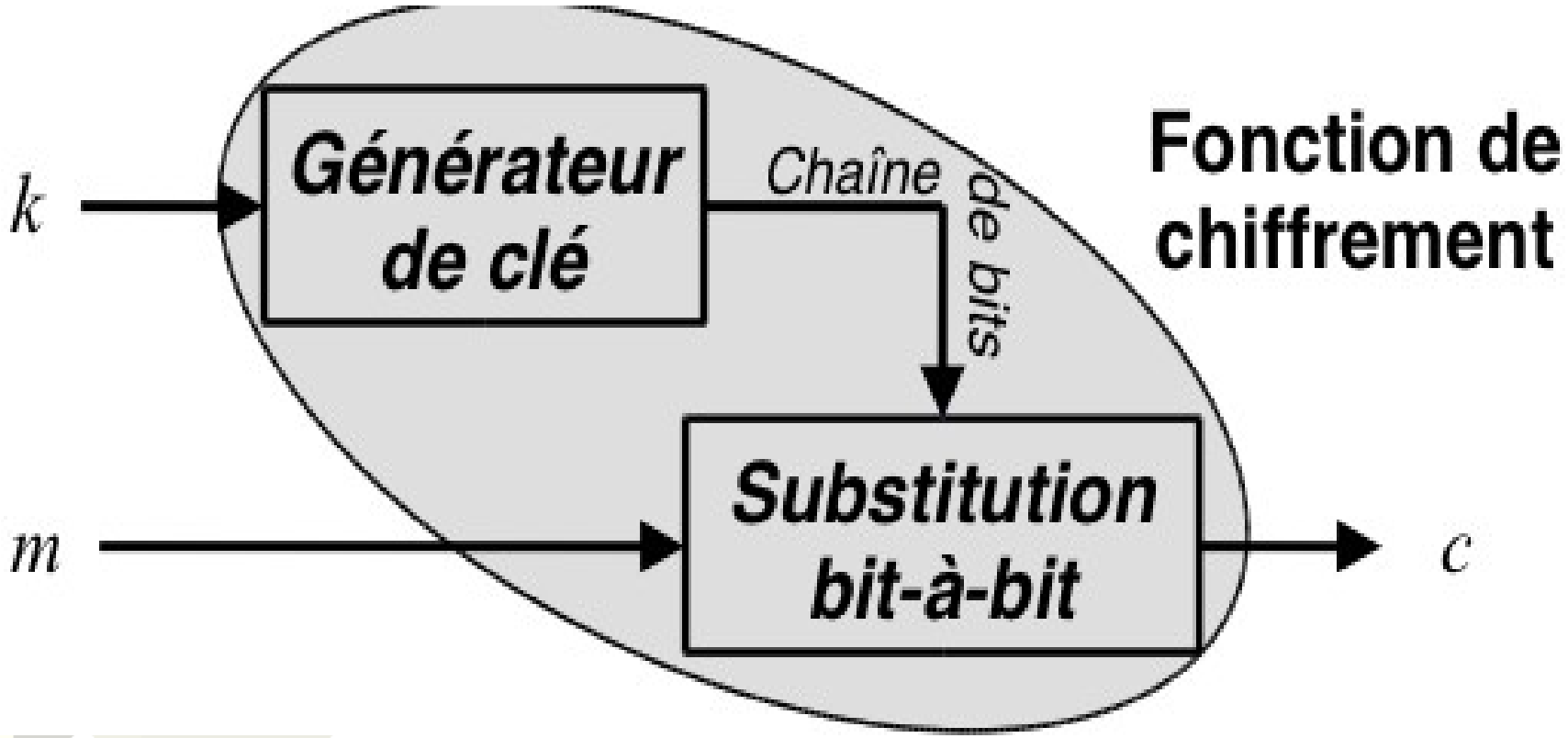
- Deux familles d'algoritmes symétriques
  - ***Le chiffrement par flux***
    - Opère sur un flux continu de données
    - Mode adapté pour la communication en temps réel
    - Implémenté en général sur des supports hardwares
  - ***Le chiffrement par blocs***
    - Opère sur des blocs de données de taille fixe
    - Implémentation logicielle en générale
    - On distingue dans ce type d'algoritme deux modes :
      - Le mode **ECB** (Electronic Code Book)
      traitement d'un bloc en code brouillé indépendamment des autres blocs. Inconvénient : un même bloc produit le même code (entête IP)
      - Le mode **CBC** (Cipher Block Chaining)
      utilisation des blocks précédemment envoyés pour coder le bloc courant.

- **Algorithmes**
  - ***Chiffrement par blocs (block cipher)***
    - Opèrent sur le texte en clair par blocs (généralement, 64 bits)
    - **Blowfish** longueur variable (32 à 448 bits) pour microprocesseurs puissants
    - **CAST**(Carlisle Adams Statford Tavares)
    - **DES** (Data Encryption Standard)
      1977, 64 bits de clé donc 56 utiles (controle de parité), chiffre de FEISTEL à 16 rondes. Variante le triple DES 112 bits.
    - **IDEA** (International Data Encryption Algorithm)
      année 1991, itération 128 bits et 8 rondes
    - **AES** (Advance Encryption Standard) remplace le triple DES 128, 192,256 bits plus rapide que IDEA et triple DES.
  - ***Chiffrement par flux***
    - Agissent sur un bit à la fois
    - **RC4** (Rivest Cypher) (pb de synchronisation).

Chiffrement par flux

Chiffrement par flux (Stream cipher)

- Chiffrement *à la One-Time-Pad* :
  - Taille du message *m* = *n*.
  - *k* permet de générer *k'* de taille *n*.
  - *c* = *m* XOR *k'* et *m* = *c* XOR *k'*.



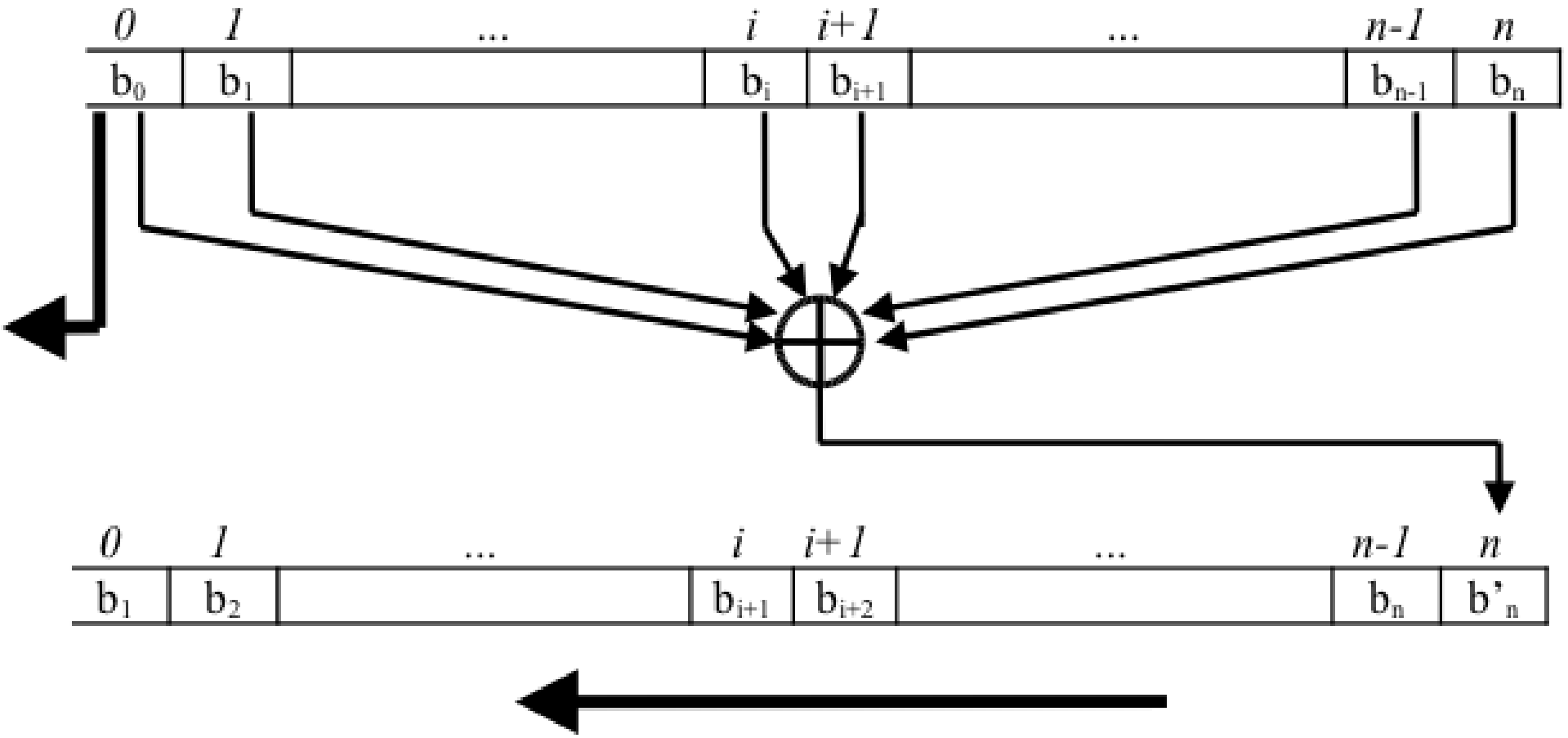
- Fonction de substitution :
  - Connue et (généralement) trivial.

- Fonction de génération de clé :
  - Fonction *pseudo-aléatoire* : impossible *à l'échelle humaine* à prédire.

La sécurité repose sur le générateur de clé

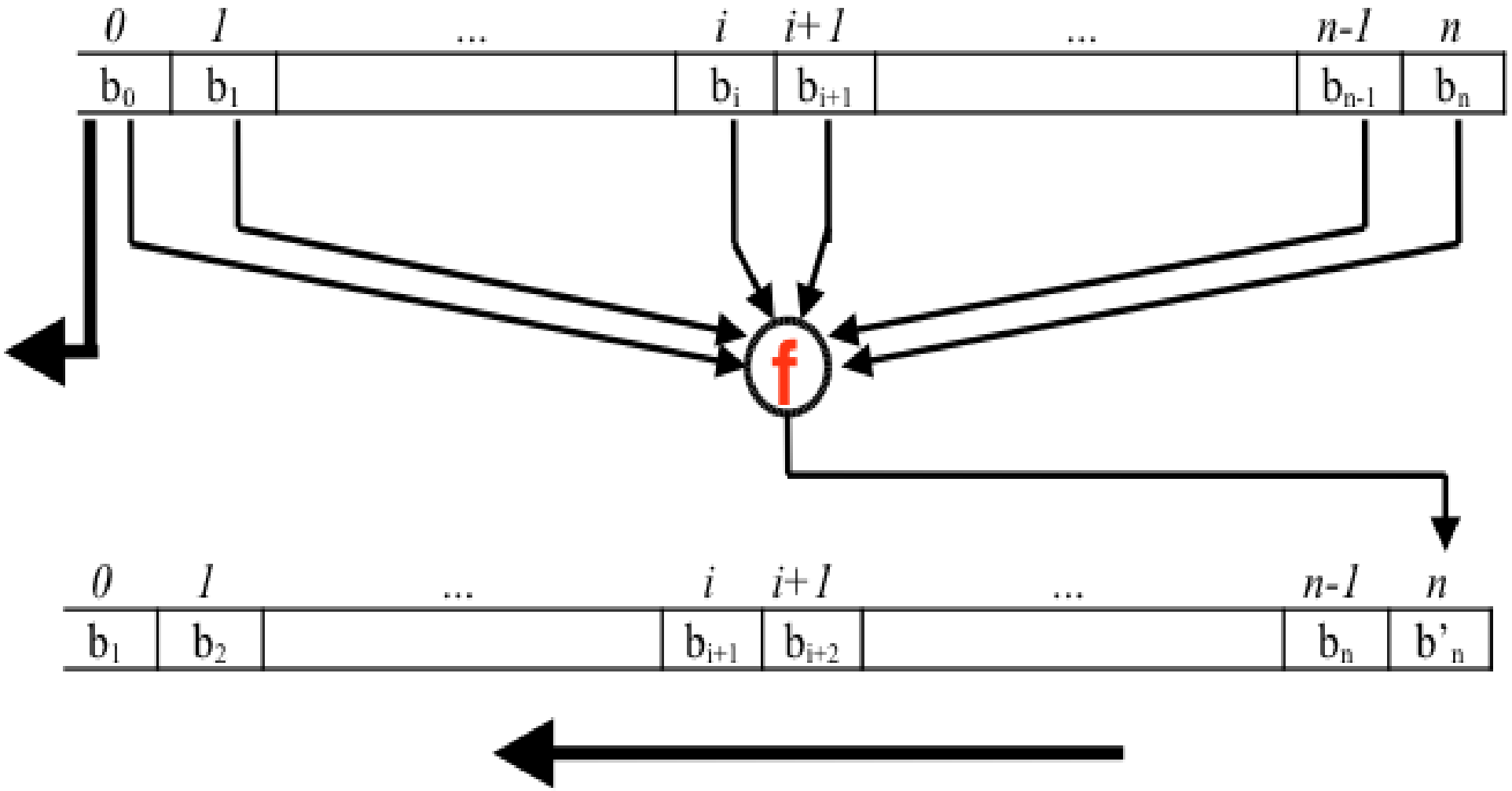
Exemple de générateur de clé

Registres linéaires de décalage (*Linear Feedback Shift Registers)*



Exemple de générateur de clé

Registres non-linéaires de décalage *(Non-Linear Feedback Shift Registers)*



Propriétés des registres de décalage

- Utilisation des registres de décalage :
  - Registres linéaires : initialisation avec la clé *K*.
  - Registres non-linéaires : *K* est un paramètre de la fonction *f*.
- Implémentation facile et rapide en hardware.

Sécurité des registres de décalage

- Périodicité des registres :
  - Si *n* registres, la période est au mieux de 2_n-1
- Conséquence sur le niveau de sécurité :
  - *Bon* niveau de sécurité.
  - Généralement, la fonction *f* est secrète.

Algorithme RC4

Rivest Cipher ou Ron Code

- Générateur de clé :
  - Permutation sur un *tableau de substitution* (*S-box*).
  - Permutation fonction de la clé de chiffrement.
- Fonction de substitution :
  - XOR entre la clé générée et le message en clair.

Algorithme secret propriété de RSA Data Security, Inc.

Algorithme RC4

Secret jusqu'en 1994 ...

- Initialisation :

S ::= [0] [1] [2] [3] [4] ... [255]

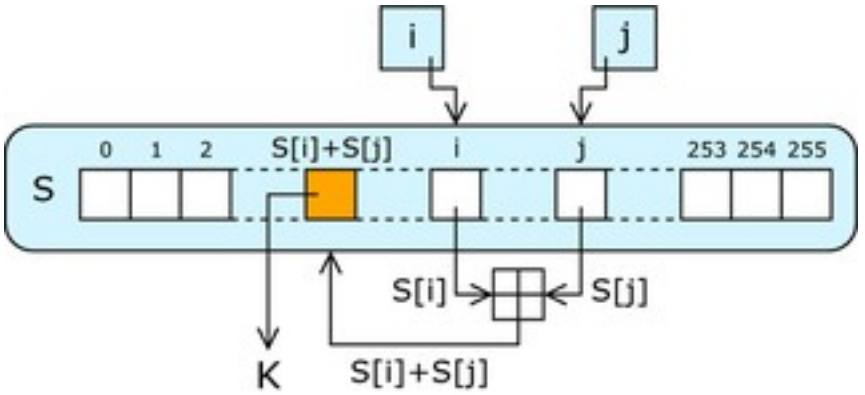
```
pour i de 0 à 255
    S[i] := i
finpour
j := 0
pour i de 0 à 255
     j := (j + S[i] + clé[i mod longueur_clé]) mod 256
    échanger(S[i], S[j])
Finpour
```

A la fin S pourrait être :

[184] [106] [163] [64] [70] ... [201]

Algorithme RC4

Secret jusqu'en 1994 ...



- Génération :

i := 0
j := 0
tant_que générer une sortie:
i := (i + 1) mod 256
j := (j + S[i]) mod 256
échanger(S[i], S[j])
octet_chiffrement = S[(S[i] + S[j]) mod 256]
result_chiffré = octet_chiffrement XOR octet_message
fintant_que

Analyse de RC4

- Table de substitution :
  - L'initialisation des **Si** dépend de la clé.
  - A chaque instant, les **Si** forment une permutation des nombres entre 0 et 255.
- Chiffrement / déchiffrement :
  - Les 8 bits générés sont utilisés pour chiffrer/déchiffrer (OU exclusif) les 8 bits suivants du message.

Analyse de RC4

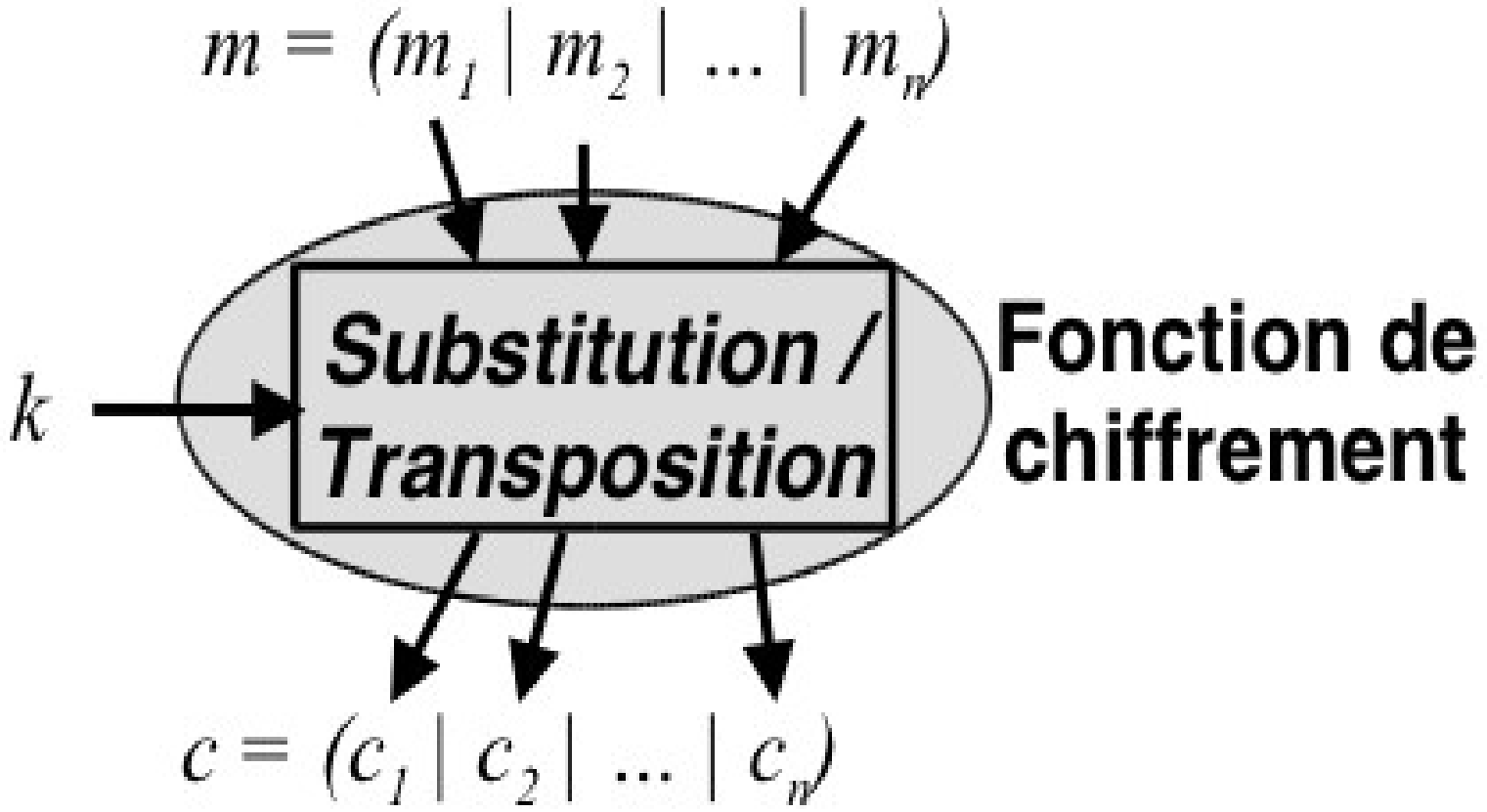
- Implémentation :
  - Code facile à retenir.
  - Très bonne performance (10 fois plus rapide que DES).
  - Lotus Notes, Apple Computer, Oracle Secure SQL ...

- Sécurité :
  - Utilise des clés de taille variable, jusqu'à 256 bits.
  - La table de substitution de RC4 possède
    $(256! \times 256_2) \approx 2_{1700}$ états
  - Bonne confiance.

Chiffrement par bloc

Block cipher

- Principe de base :
  - Message *m* = (*m*1 | … | *mn*), *n blocs* de *x* bits.
  - Transposition et substitution *bloc par bloc* .



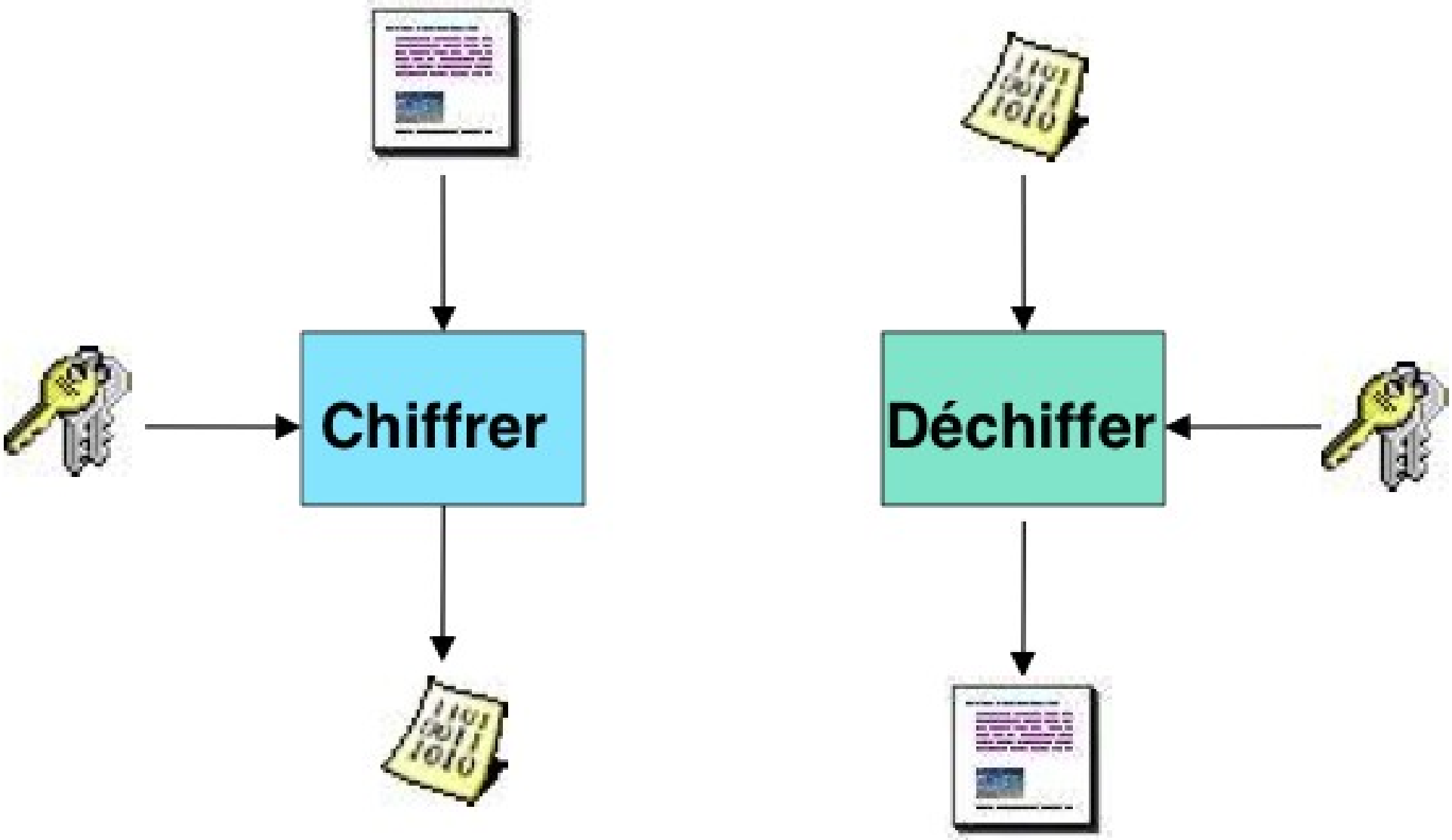
Sécurité

- Substitution polyalphabétique.
  - Utilisation de tableaux de substitution ou *S-boxes*.
- Transposition :
  - Agit sur le message, la clé et les *S-boxes*.

La sécurité repose sur la combinaison substitution/transposition.

Modes de chiffrement

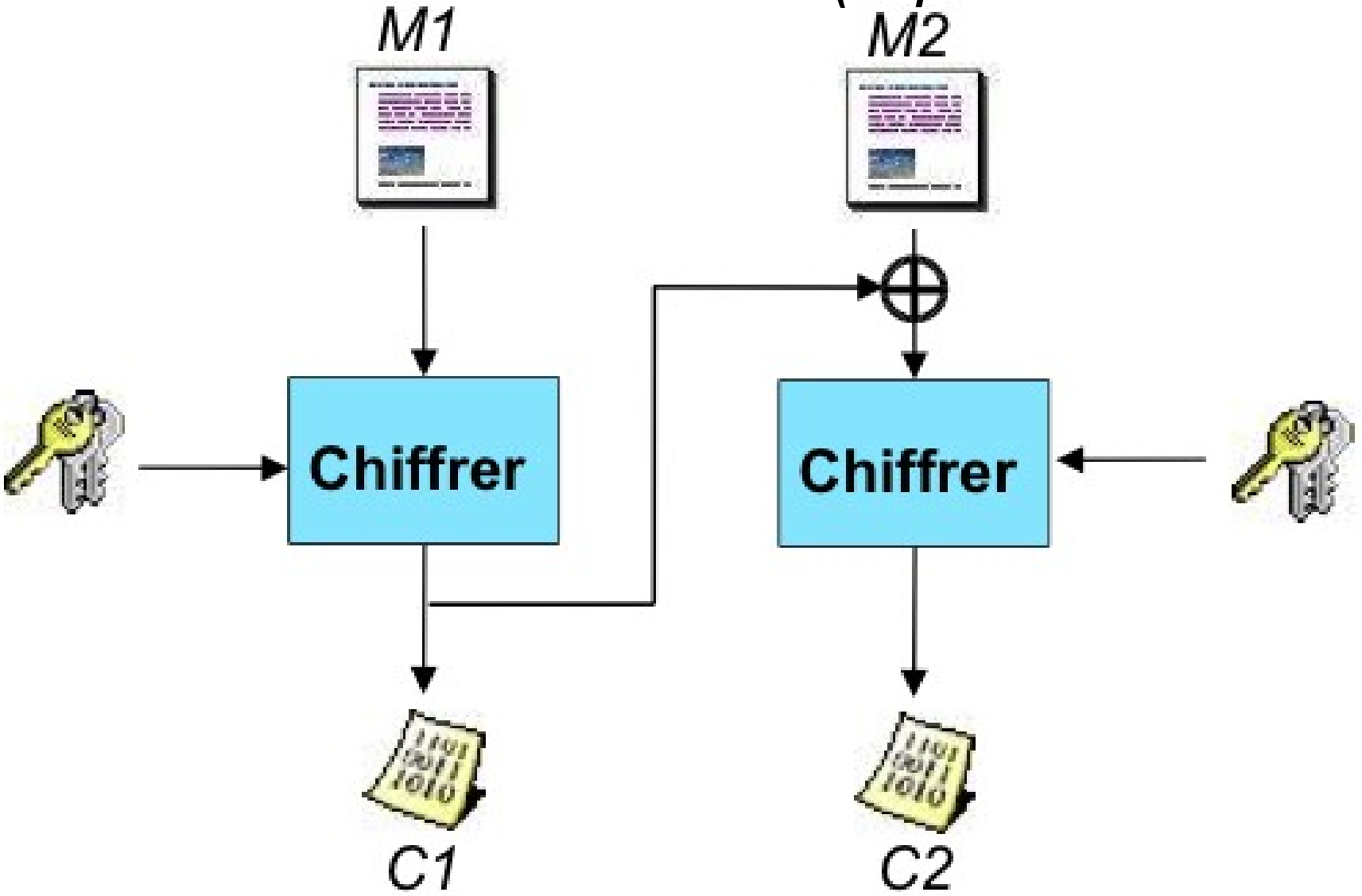
Mode ECB (Electronic CodeBook)



Modes de chiffrement

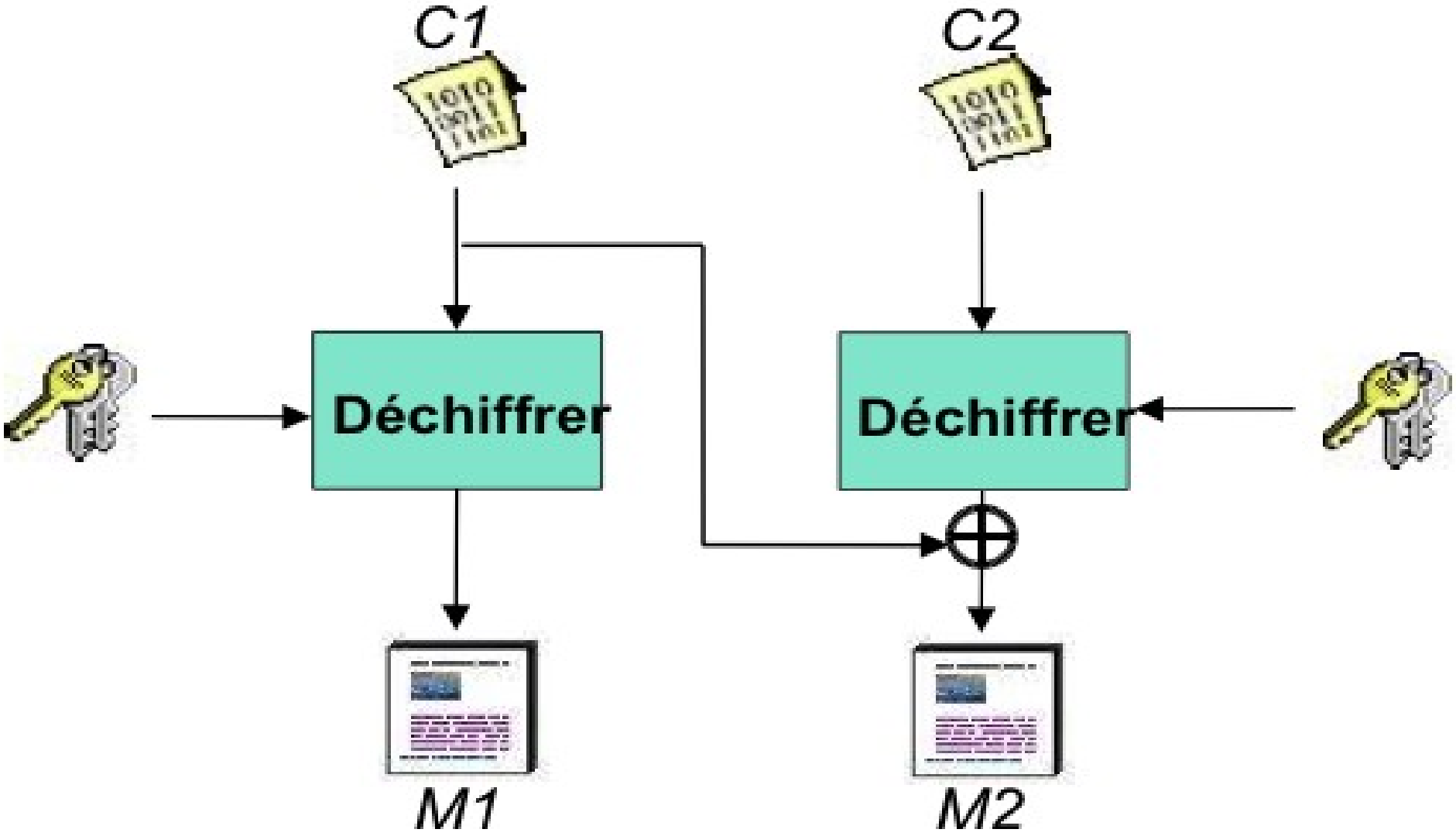
Chiffrement en mode CBC :

Mode CBC (Cipher Block Chaining)



Déchiffrement en mode CBC.

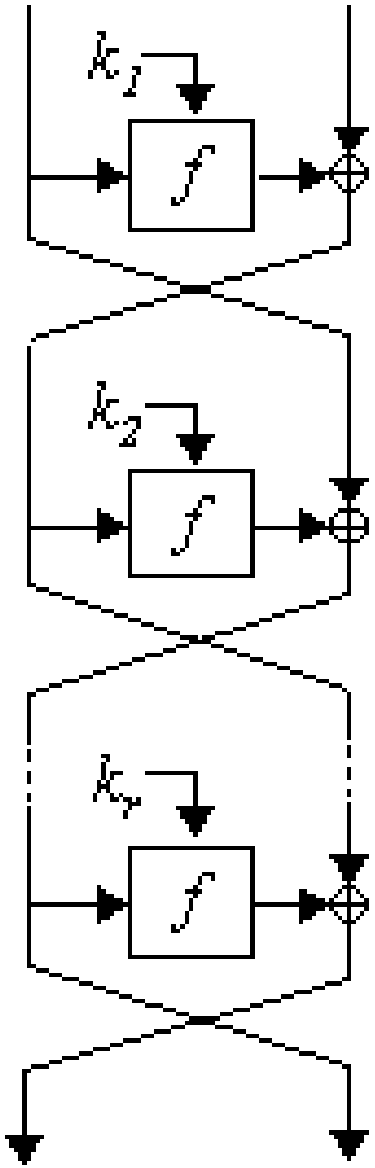
Mode CBC (Cipher Block Chaining)



Modes de chiffrement

Propriétés

- Mode ECB :
  - Chiffrement *statique* et pas de propagation d'erreur.
- Mode CBC :
  - Chiffrement *chaîné* et propagation d'erreur limitée à 1 bloc.



Le chiffre de FEISTEL, blocs avec itérations

On chiffre les blocs par un processus comportant plusieurs rondes.
Dans chaque ronde, la même transformation est appliquée au bloc, en utilisant une sous-clef dérivée de la clef de chiffrement.

Exemple la famille des chiffres de Feistel :

- Un bloc de texte en clair est découpé en deux ;
- la transformation de ronde est appliquée à une des deux moitiés, et le résultat est combiné avec l'autre moitié par **ou exclusif**.
- Les deux moitiés sont alors inversées pour l'application de la ronde suivante.

Un avantage de ce type d'algorithmes est que chiffrement et déchiffrement sont structurellement identiques.

Algorithme DES

Data Encryption Standard

- Historique :
  - Milieu des années 70.
  - 1er algorithme de chiffrement pour l'industrie.
  - Standard américain FIPS 46-2.

- Principes de base :
  - Produit de substitutions/transpositions.
  - Chiffrement à la Feistel : itération de la fonction de chiffrement.

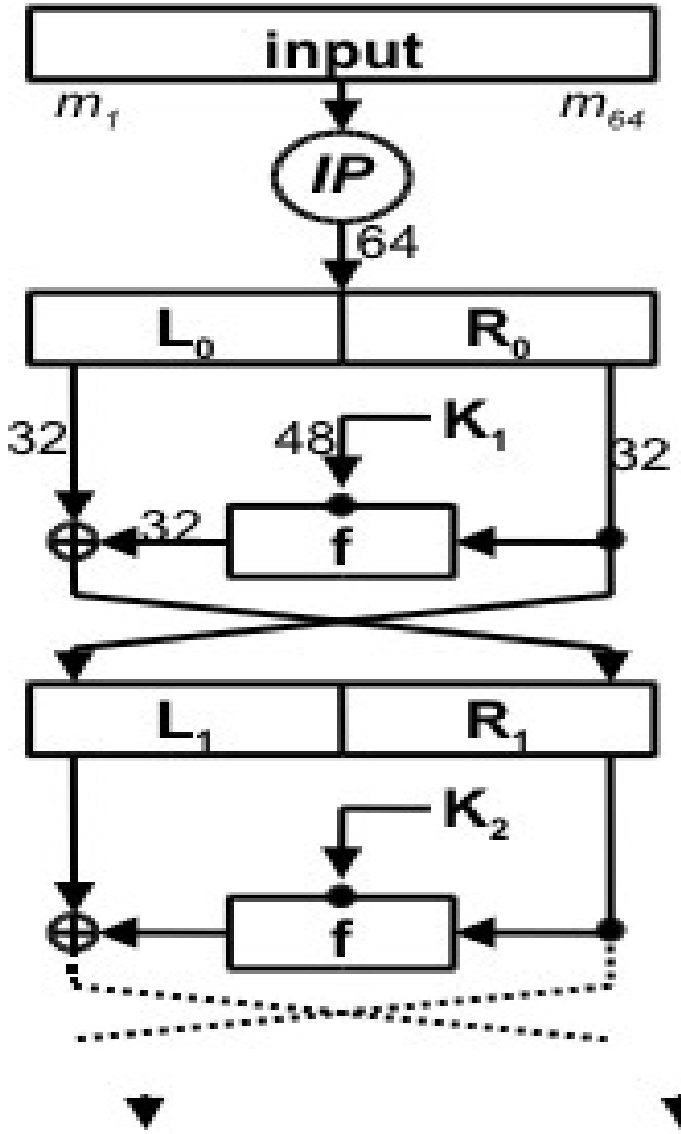
Algorithme DES

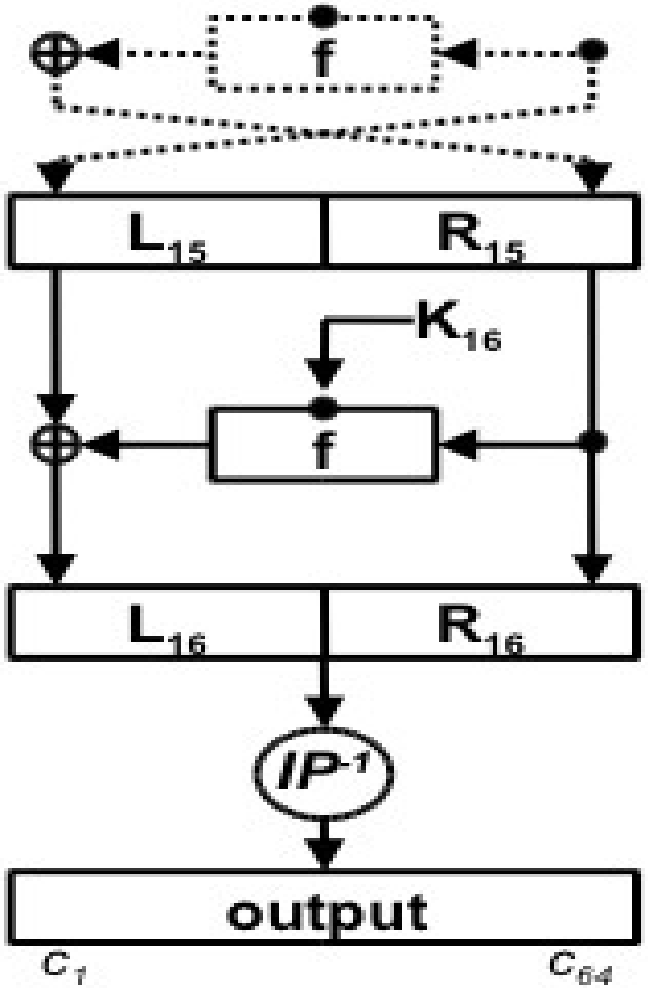
Informations techniques

- Taille des blocs : 64 bits.
- Taille de la clé : 56 bits.
- Structure globale :
  - Permutation initiale.
  - Fonction itérée : expansion, substitution, permutation
  - Nombre d'itérations : 16.
  - *Key schedule* : 16 sous-clés de 48 bits.

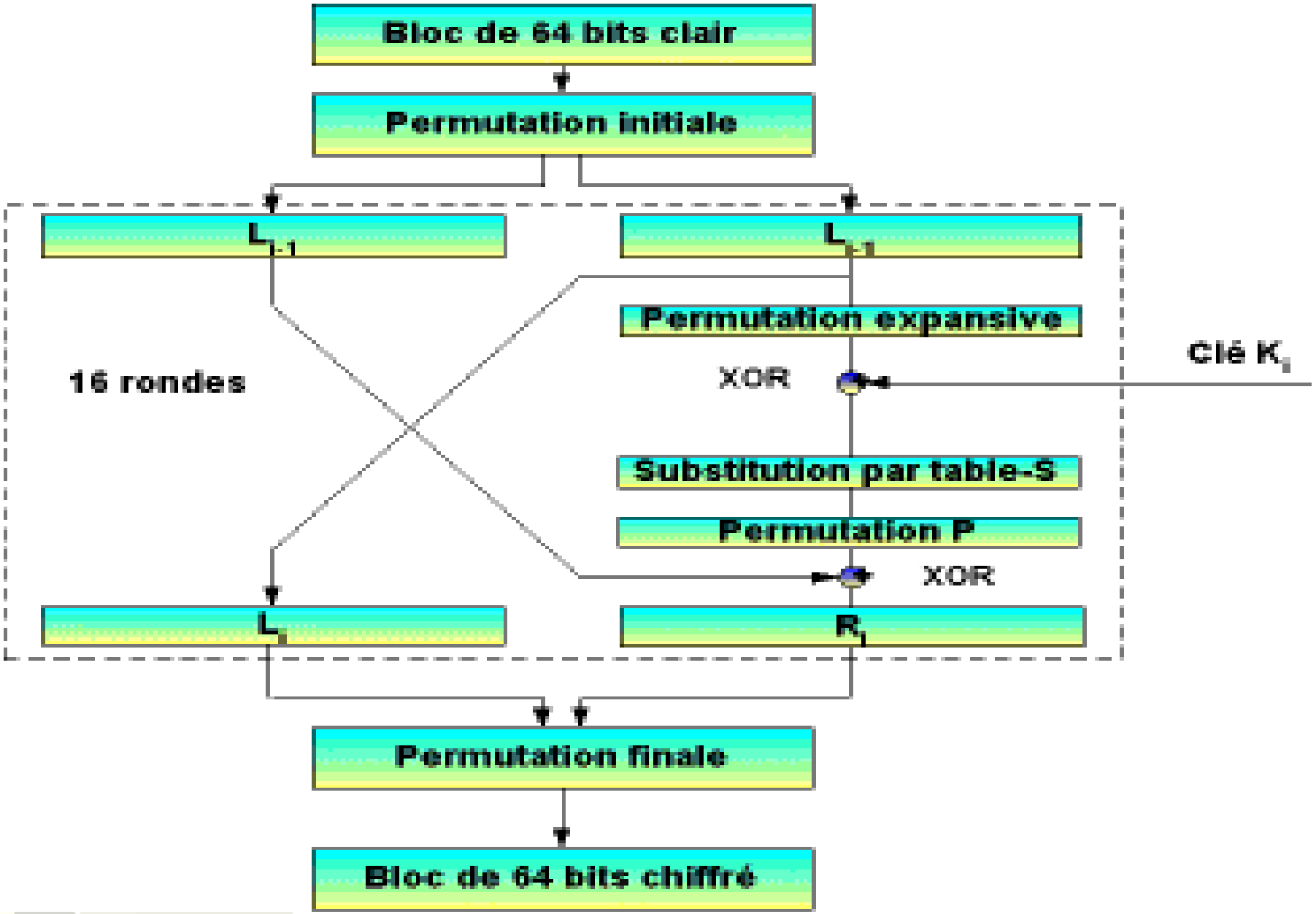
Algorithme DES

Algorithme global



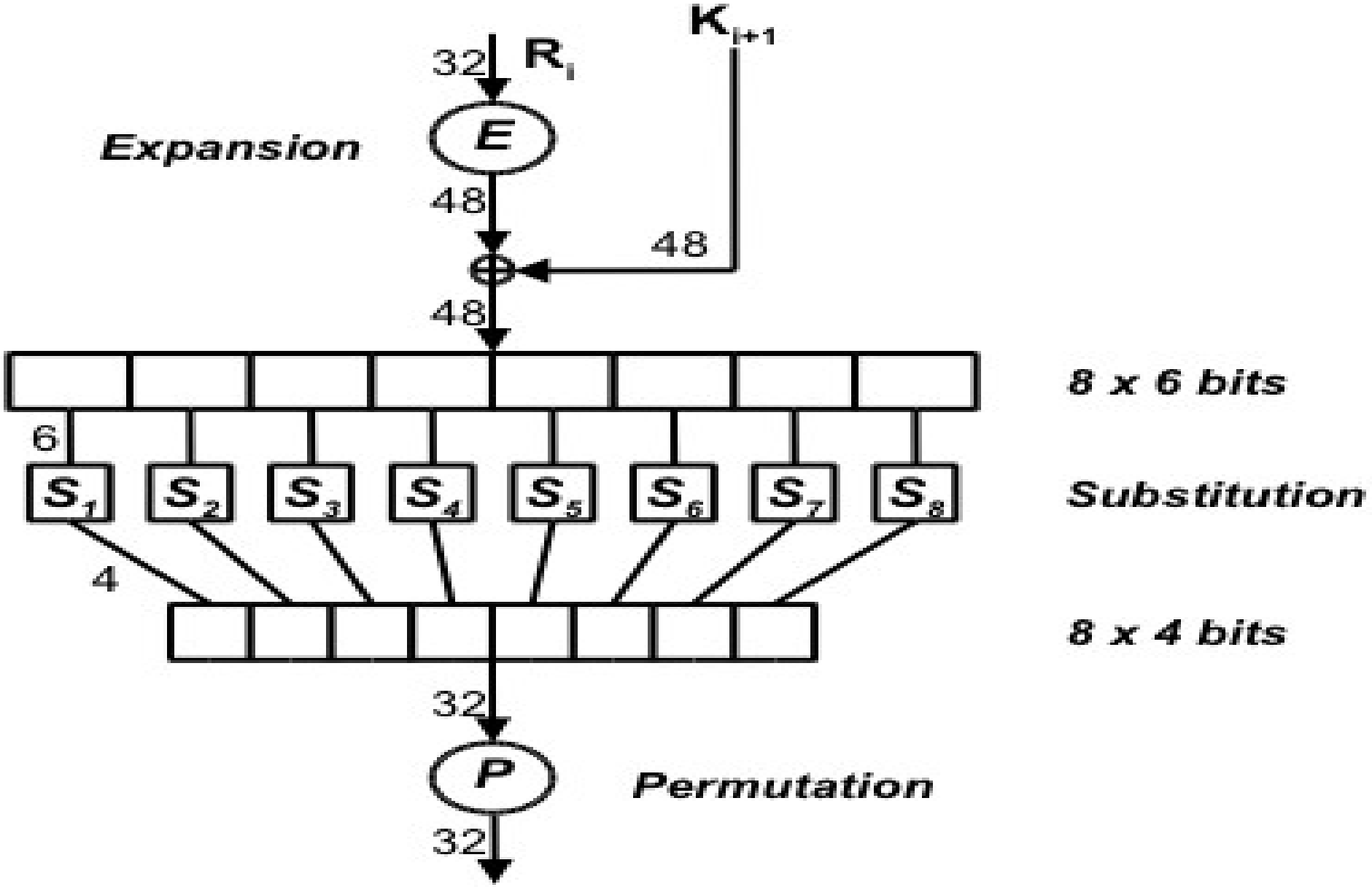


Algorithme DES



Algorithme DES

Fonction itérative



Permutation initiale

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

- 64 bits en entrée, 64 bits en sortie :
  - Le bit n°58 se retrouve à la position 1, le bit n°50 à la position 2...
- Fonction (trivialement) inversible.

Algorithme DES

Expansion de la fonction itérative

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

- 32 bits en entrée, 48 bits en sortie :
  - Les 6 premiers bits de sortie sont les bits (32,1,2,3,4,5), les 6 bits suivants sont (4,5,6,7,8,9) ...
- Fonction inversible :
  - Il n'existe qu'un bloc d'entrée pour un bloc de sortie donnée.

Algorithme DES

Tableaux de

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

- 8 tableaux de 4 lignes x 16 colonnes :
  - Chaque cellule est un nombre de 4 bits (de 0 à 15).
- Chaque tableau définit une substitution :
  - La S-Box *Si* permet de subtituer les 6 bits ($b_1b_2b_3b_4b_5b_6$) par les bits $Si[b_1b_6][b_2b_3b_4b_5]$.
  - Exemple : *S1* substitue *(011010)* en *S1[00][1101] = 9 (1001)*.
- Fontion inversible comme toute substitution.

Algorithme DES

Permutation de la fonction itérative

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

- 32 bits en entrée, 32 bits en sortie :
  - Le bit n°16 se retrouve à la position 1, le bit n°7 à la position 2 ...
- Fonction (trivialement) inversible.

Permutation finale

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

- 64 bits en entrée, 64 bits en sortie :
  - Permutation inverse de la permutation initiale.

- Le bloc utilisé en entrée est $L_{16}R_{16}$:
  - Le même algorithme peut être utilisé pour chiffrer et déchiffrer.

Génération des sous-clés (Key Schedule)

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

- Clé de 56 bits stockée sur 64 bits :
  - 1 bit de parité sur chaque octect : bit n°8, n°16 ...
- Permutation initiale de la clé :
  - N'apporte rien *a priori* du point de vue de la sécurité.

Génération des sous-clés (Key Schedule)

Itération	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Décalage	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

- Clé de 56 bits vue comme deux clés de 28 bits.
- Sur chaque sous-clé de 28 bits :
  - Décalage à gauche de une ou deux positions, en fonction de l'itération.

Algorithme DES

Génération des sous-clés (Key Schedule)

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

- Permutation compressive ou choix permuté :
  - 56 bits en entrée, les 48 bits de sous-clé en sortie.
  - Le bit n°14 se retrouve en position 1, alors que les bits n° 9, 18, 22, 25, 35, 38, 43, 54 ne sont pas utilisés.
- Décalage / permutation compressive :
  - Chaque bit est utilisé dans environ 14 des 16 sous-clés.

Algorithme de déchiffrement

- Permutation initiale / permutation finale :
  - Inverse l'une de l'autre.
  - La permutation finale est réalisée sur $L_{16}R_{16}$ et non $R_{16}L_{16}$.

- Fonction itérative :
  - Permutation expansive : une entrée unique pour une sortie donnée.
  - Tableaux de substitution : construits pour garantir que l'inverse de la fonction itérative soit elle-même.

- Conséquence :
  - Algorithmes de déchiffrement et de chiffrement identique.
  - Les sous-clés *Ki* sont utilisées dans le sens inverse.

Analyse du DES

- Confusion / Diffusion au sein d'un bloc :
  - Après seulement 5 itérations, chaque bit du chiffré dépend de chaque bit du message en clair et de chaque bit de la clé.
- Propriété de *complémentarité* :

  $y = DES_k(x) \Rightarrow y = DES_k(x)$
- DES n'est pas un groupe :

  $\forall$ k1,k2, $\exists$ k3 | $\forall$ *x*, $DES_{k1}(DES_{k2}(x)) = DES_{k3}(x)$
- Bonnes performances :
  - Puces dédiées au DES : en 1995, le 6868 de VLSI permettait de chiffrer 64 Mo par seconde.
- 64 clés faibles (i.e., ne générant pas 16 sous-clés différentes).

Confusion et Diffusion ?

- Confusion *totale* :
  - Chiffrement de « aaaaaaaaaaaaaaaaaaaaa » :

    f99a 4388 c6a8 57db 1a0c c4d4
    ad1a 89f7 119d 9d91 7827 94b5

- Diffusion *totale* en mode CBC :
  - Chiffrement de « eaaaaaaaaaaaaaaaaaaaa » :

    a290 3816 10d3 97e7 aa2a 25f3
    c3e0 a3cf 9438 f2b2 dbb8 f3da

Sécurité du DES

Méthode d'attaque	Texte connu	Texte choisi	Complexité de stockage	Complexité de calcul
Précalcul exhaustif		1	2^{56}	1 tableau
Recherche exhaustive	1			2^{55}
Cryptanalyse linéaire	2^{43}		*Pour les textes*	2^{43}
	2^{38}		*Pour les textes*	2^{50}
Cryptanalyse différentielle		2^{47}	*Pour les textes*	2^{47}
	2^{55}		*Pour les textes*	2^{55}

Attaques possibles sur le DES et complexité.

Sécurité du DES

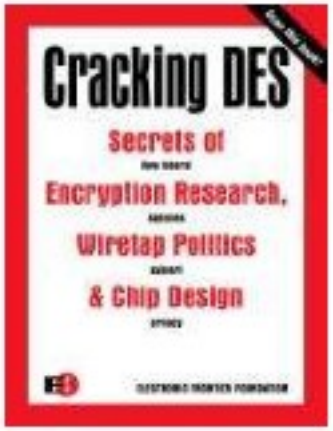
Type d'attaquant	Budget	Outil	Clé de 40 bits	Clé de 56 bits
Simple hacker	Négligeable	Soft	1 semaine	Impossible
	300 €	Circuit prédiffusé	5 heures	38 ans
PME	7500 €	Circuit prédiffusé	*12 minutes*	18 mois
Grande entreprise	225 k€	Circuit prédiffusé	*24 secondes*	19 jours
	225 k€	ASIC	*0.18 seconde*	3 heures
Multinationale	7,5 M€	ASIC	*5 msec.*	6 minutes
Etat	225 M€	ASIC	0.2 msec.	12 secondes

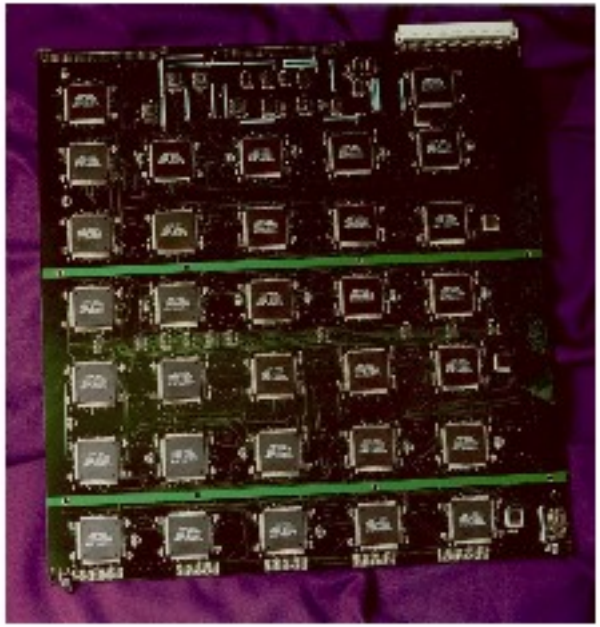
Coût / performance d'une recherche exhaustive (1996).

Sécurité du DES

DES Cracker (1998).

`http://www.eff.org/descracker.html`



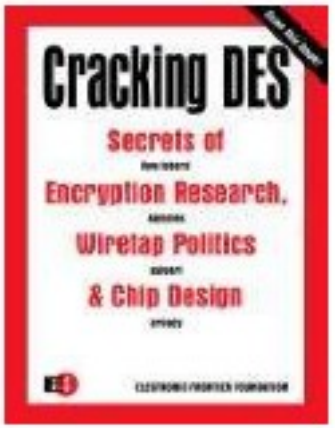


Sécurité du DES

DES Cracker (1998).

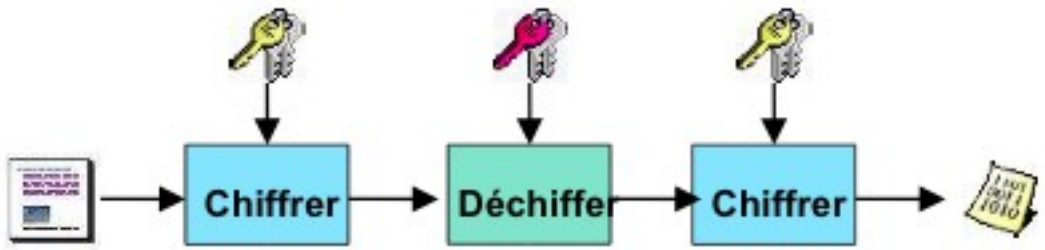
`http://www.eff.org/descracker.html`

- Coût du DES Cracker : environ 250000 $.
- Performance : 56 heures.



L'après DES ... Triple DES

- Principe :



C = DES[K1](DES-1[K2](DES[K1](M)))

- Avantages :
  - Algorithme bien connu.
  - Compatibilité ascendante.

Sécurité du Triple DES

- Performance :
  - Médiocre pour un chiffrement 112 bits.

- Niveau de sécurité :
  - Equivalent à l'utilisation d'une clé de 80 bits (estimation).
  - Suffisant aujourd'hui, mais demain ...

Advanced Encryption Standard

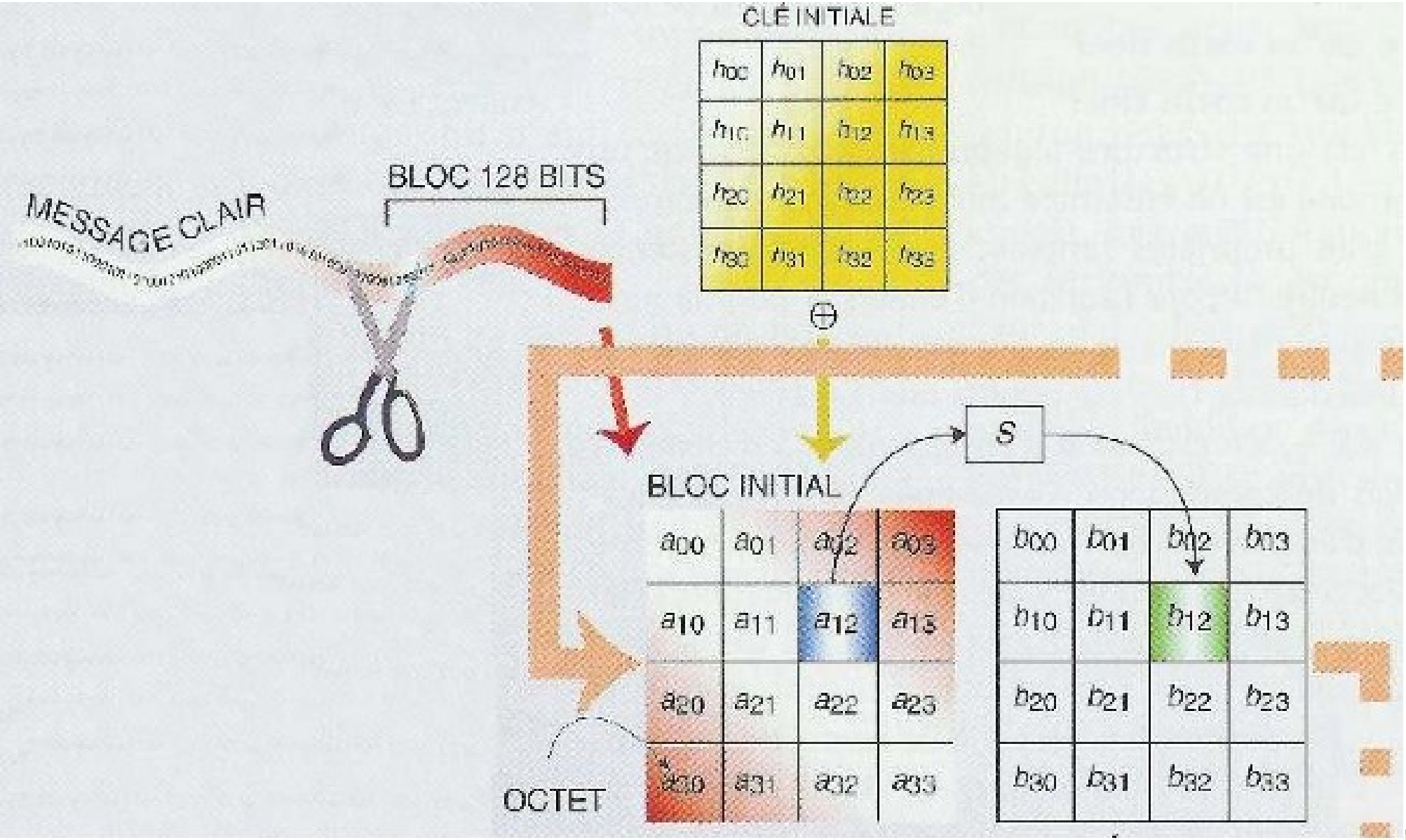
- Processus de sélection :
  - 1/1997 : Appel à candidature.
  - 8/1998 : Fin des candidatures.
  - 8/1998 - 8/2000 : Analyse des candidats.
  - 8/2000 : Sélection de l'AES.

Rjindael

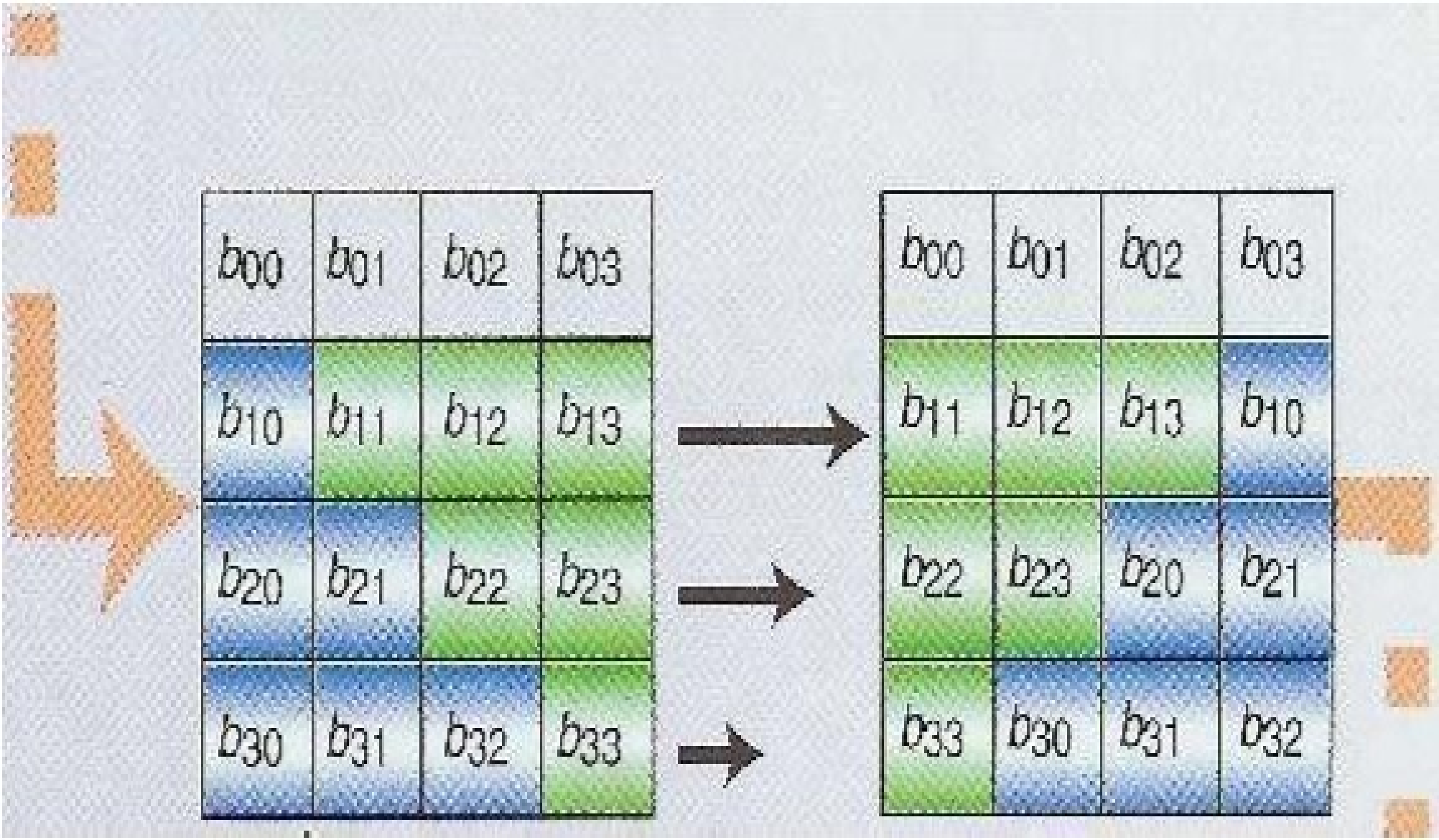
Informations techniques

- Taille des blocs : variable (de 128 à 256 bits).
- Taille de la clé : variable (de 128 à 256 bits).
- Structure globale :
  - Le bloc à chiffrer est disposé dans un tableau.
  - Fonction itérée : permutation des cellules, décalage des lignes, permutation des colonnes, substitution (XOR).
  - Nombre d'itérations : 10.
  - *Key schedule* : 10 sous-clés.

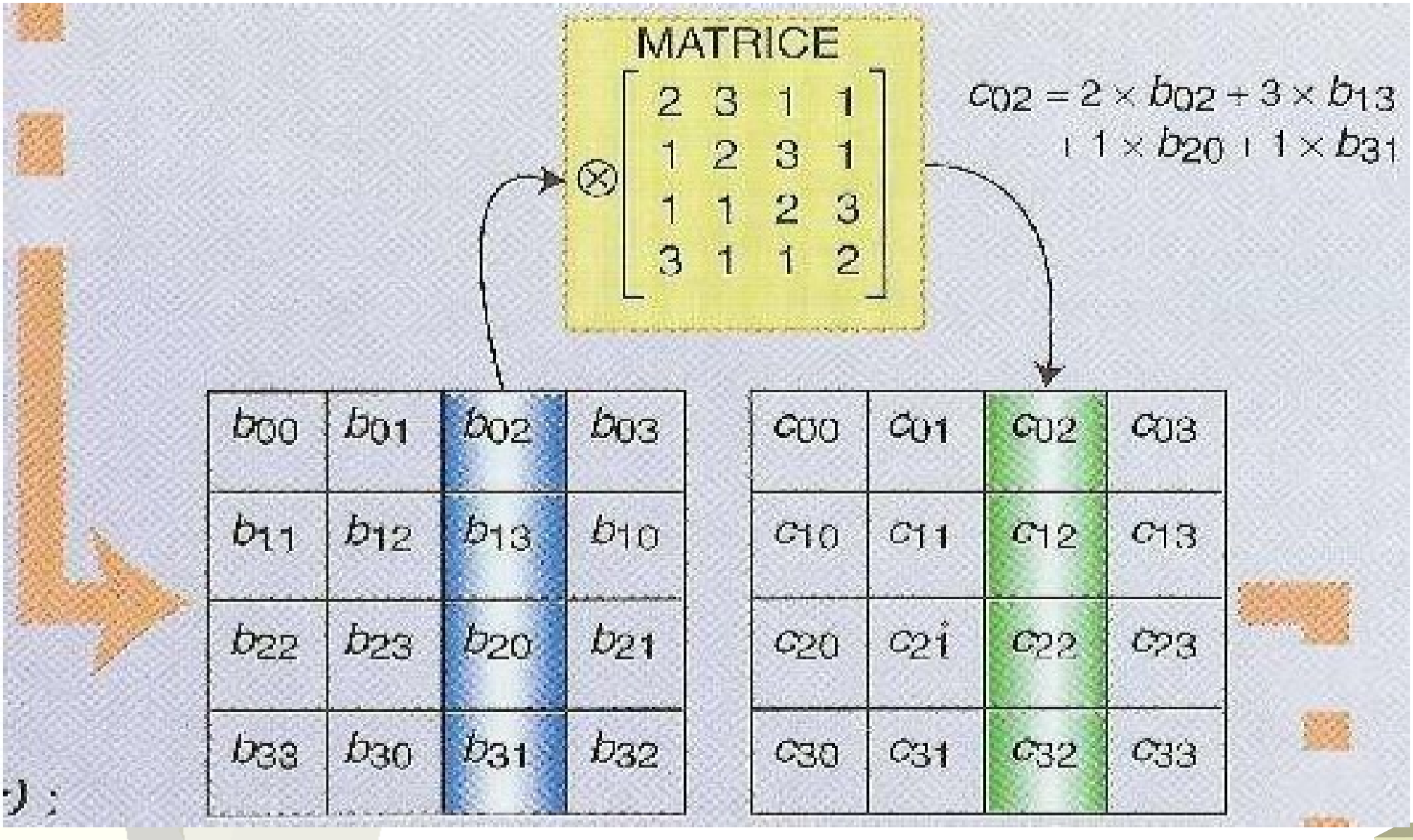
Etape 1 : Transformation non-linéaire



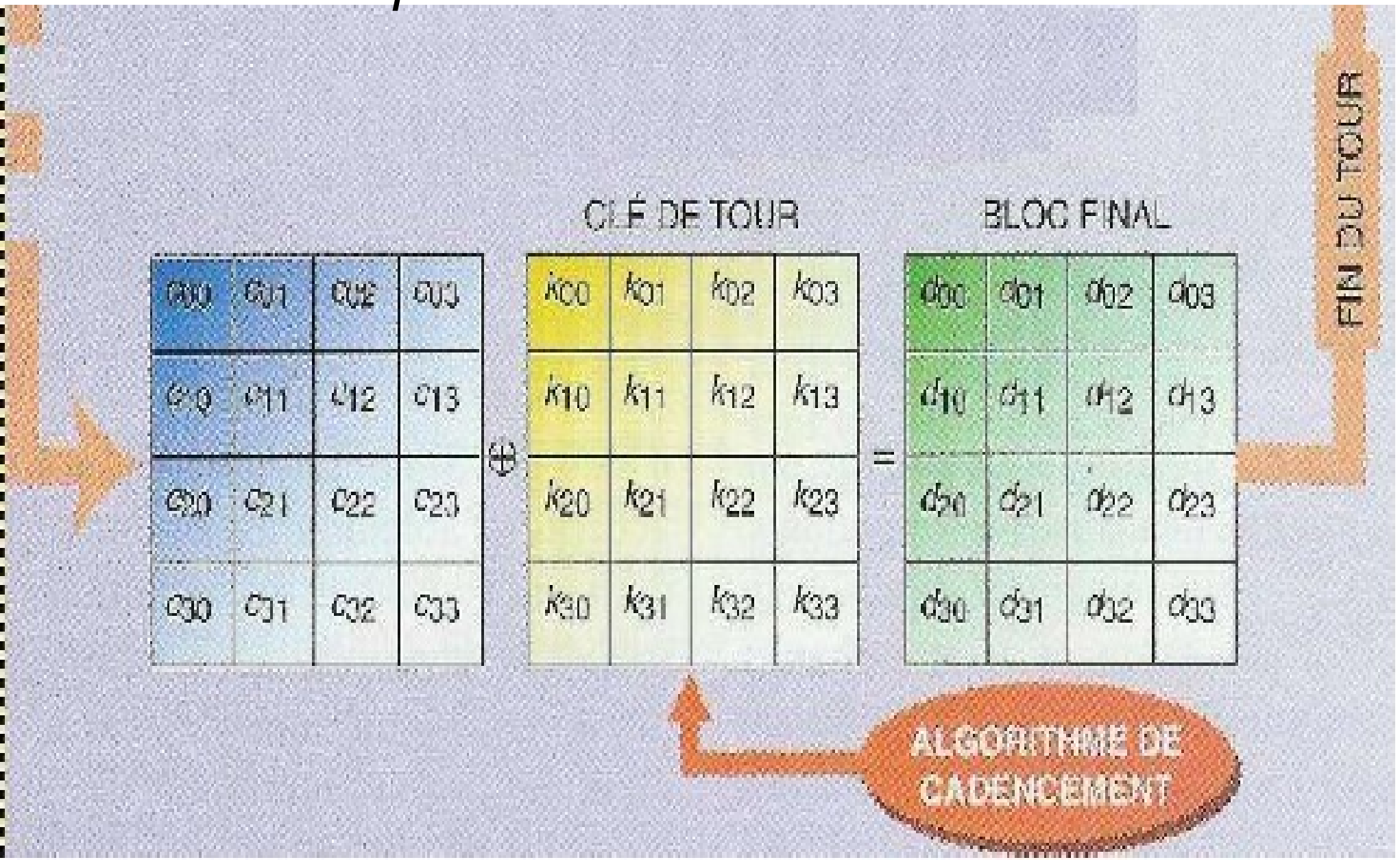
Etape 2 : Décalage de lignes



Etape 3 : Brouillage de colonnes

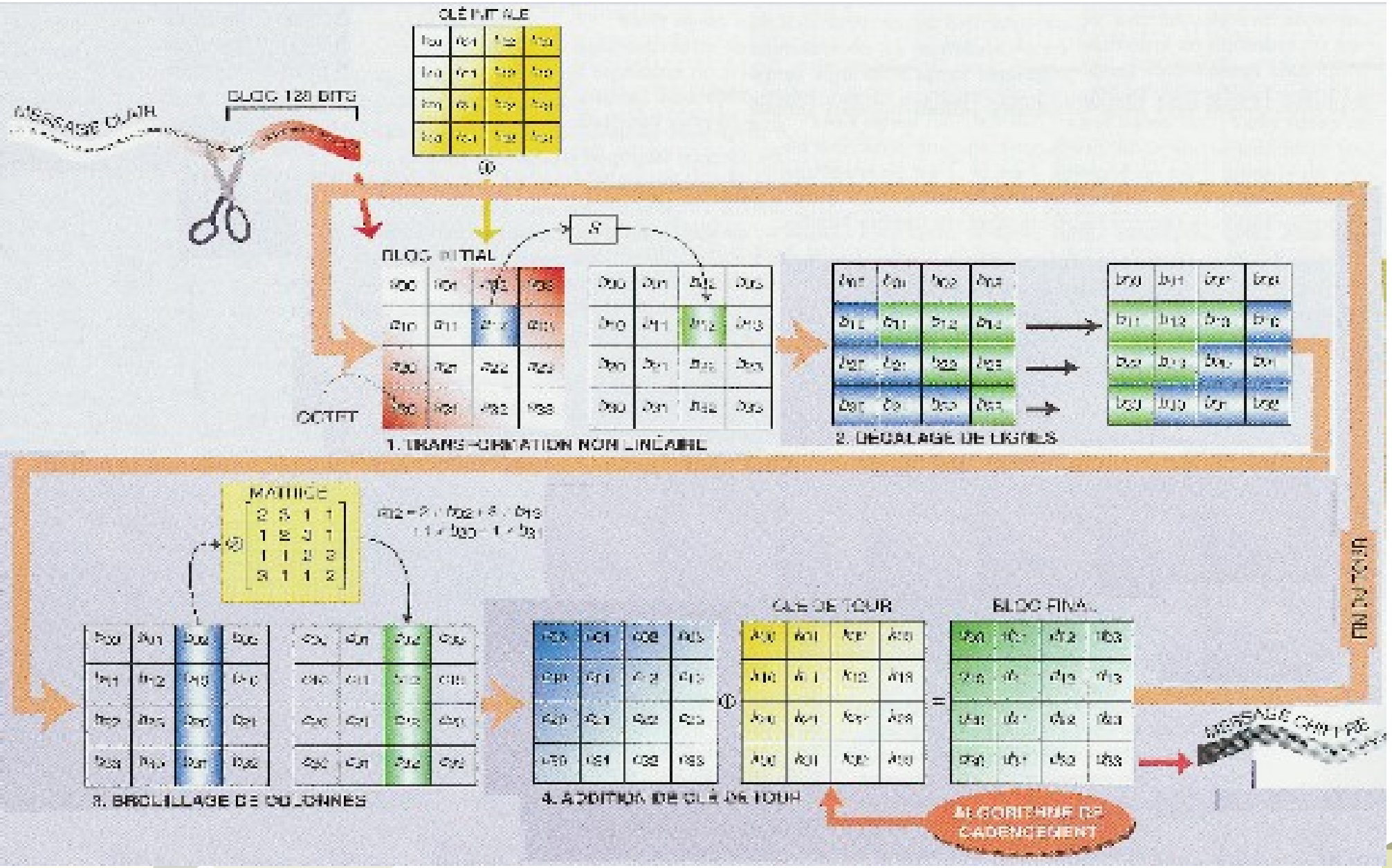


Etape 4 : Addition de clé de tour



Algorithme AES

Vue globale.



Analyse de l'AES

- Performance :
  - Très bonne (condition de sélection).
- Niveau de sécurité :
  - Aucune attaque significative connue à ce jour ...
  - ... mais seulement 5 ans de recherche !

Confusion et Diffusion ?

- Confusion *totale* :
  - Chiffrement de « aaaaaaaaaaaaaaaaaaaaa » :

    c9ab 593d c7cc c32f da1a 9c19 26a2 cad2
    bfb4 ff83 5b93 9474 d4e6 4cd1 45f2 0f4e

- Diffusion *totale* en mode CBC :
  - Chiffrement de « eaaaaaaaaaaaaaaaaaaaa » :

    e78b 6ee4 223f 13f5 4cfd 1206 b99f dbee
    329f b7eb bfef 88ce bca1 09fd 2764 2a32

Source : http://www.schneier.com/blowfish-speed.html

Bruce Schneier

Speed Comparisons of Block Ciphers on a Pentium

Algorithm	Clock cycles per round	# of rounds	# of clock cycles per byte encrypted	Notes
Blowfish	9	16	18	Free, unpatented
Khufu/Khafre	5	32	20	Patented by Xerox
RC5	12	16	23	Patented by RSA Data Security
DES	18	16	45	56-bit key
IDEA	50	8	50	patented by Ascom-Systec
Triple-DES	18	48	108	

Note sur le chiffrement symétrique

Chiffrement :
ECB
vs
CBC

Pour chi rerde longs messages

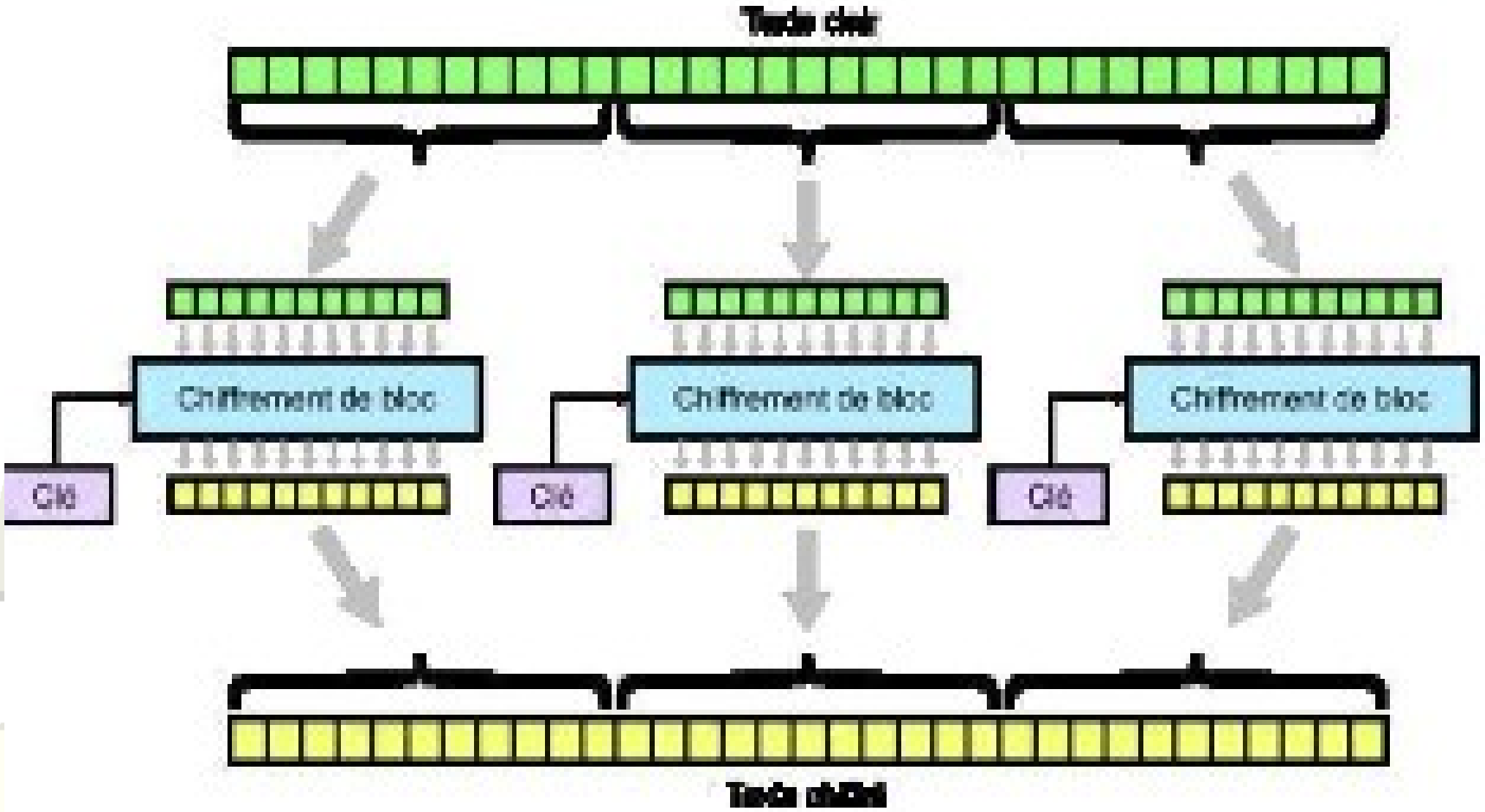
On sait chi rerdes petits messages (blocs)

Chi rementpar bloc

Comment assembler les blocs ?

Deux modes classiques
- Electronic codebook (ECB)
- Cipher Block Chaining (CBC)

Electronic codebook (ECB)



Inconvénients

Malléable :

A partir de $\{m1 \cdot m2 \cdot \cdot \cdot mp\}$

$$k = \{ m1 \}k \cdot \{ m2 \}k \cdot \cdot \cdot \{ mp \}k ,$$

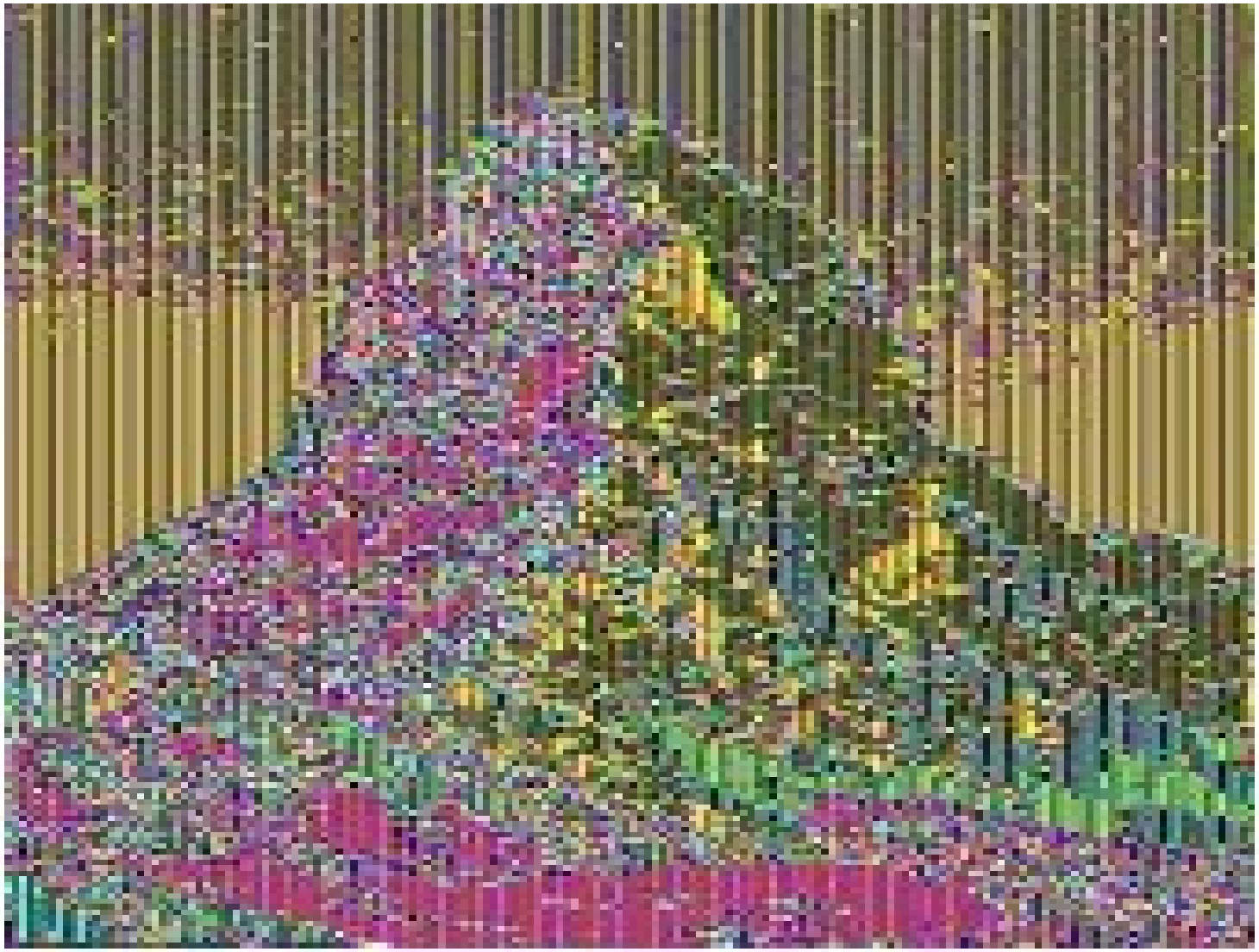
On peut calculer $\{m2 \cdot m1 \cdot \cdot \cdot mp \}k$

On peut tester l'´egalité de certains blocs :

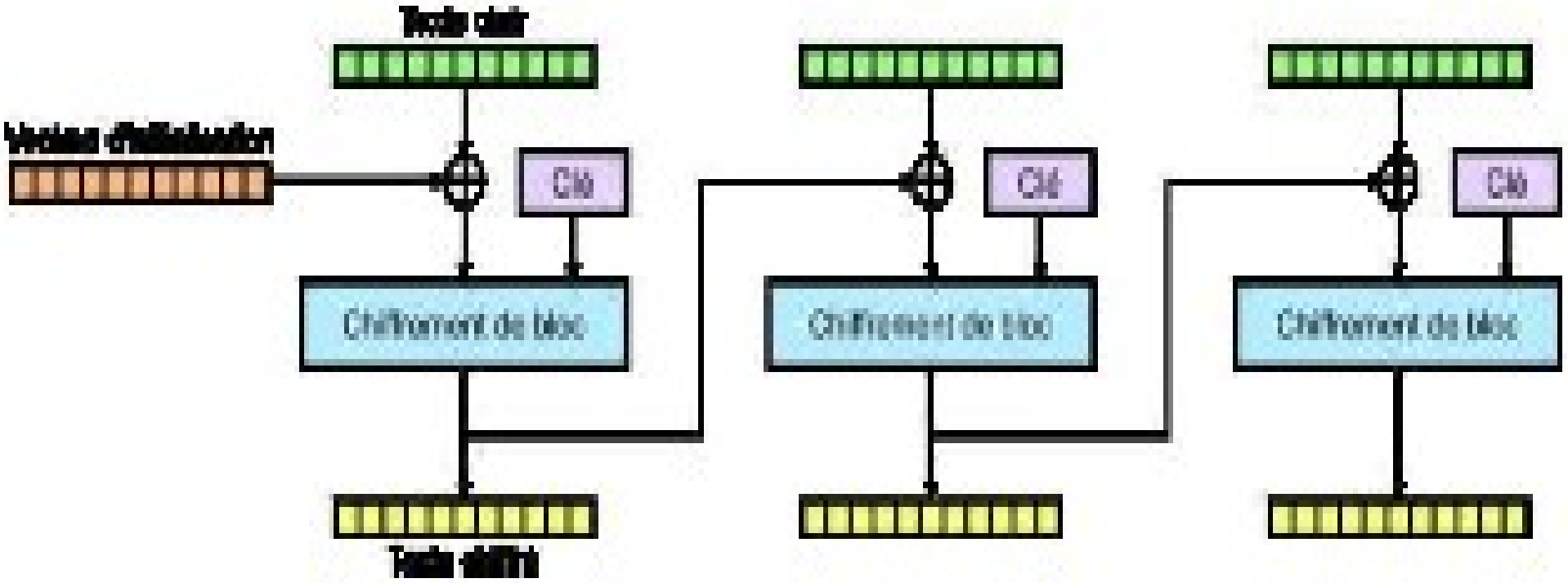
$$\{ m1 \}k = \{ m2 \}k \qquad m1 = m2$$

Exemple en images
Le Cervin

Exemple en images
Le Cervin chi réavec ECB



Cipher Block Chaining (CBC)



Exemple en images
Le Cervin chi réavec CBC