

HAI709I - Fondements cryptographiques pour la sécurité

Cours 4 - Chiffrement par blocs

Rocco Mora

30 Septembre, 2025

Université de Montpellier – Faculté des Sciences
M1 informatique, parcours Algo, IASD, Imagine

Chiffrements par blocs [Block ciphers]

Chiffrement par blocs :

- comme une **permutation pseudo-aléatoire forte** F : permutation pseudo-aléatoire où le distingueur a également accès à l'**oracle de la fonction inverse** F_k^{-1} , en plus de F_k
- Par contre, les chiffrements par blocs peuvent gérer **clés de longueur arbitraire**.

Les chiffrements par flot chiffrent une unité (p. ex. bits ou octets) à la fois

vs

Les chiffrements par blocs fonctionnent sur des blocs de données de taille fixe

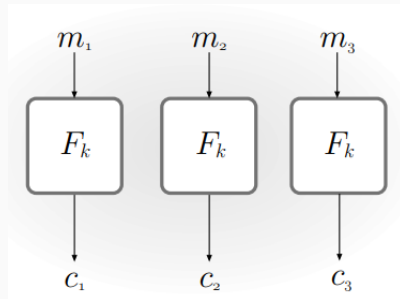
Modes opératoires [ modes of operations] : ceux-ci décrivent comment **appliquer de manière répétée l'opération sur un seul bloc d'un chiffrement** pour transformer en toute sécurité des quantités de données supérieures à un bloc

Mode Dictionnaire de codes [🇬🇧 Electronic Code Book (ECB) mode]

Dictionnaire de codes (ECB)

Application directe naïve du chiffrement par blocs à chaque bloc de texte en clair

$$c := F_k(m_1), F_k(m_2), \dots, F_k(m_\ell).$$

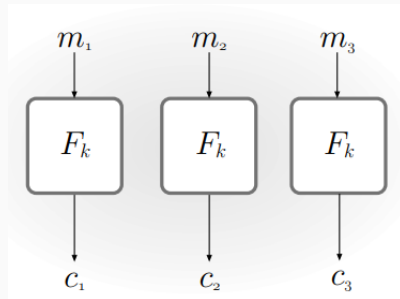


Mode Dictionnaire de codes [🇬🇧 Electronic Code Book (ECB) mode]

Dictionnaire de codes (ECB)

Application directe naïve du chiffrement par blocs à chaque bloc de texte en clair

$$c := F_k(m_1), F_k(m_2), \dots, F_k(m_\ell).$$



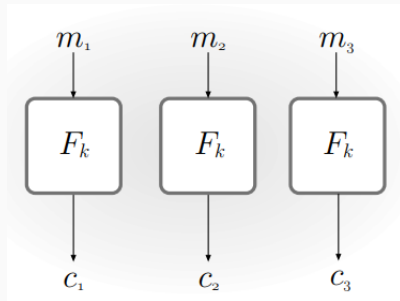
- L'ECB n'est pas CPA-sûr car il est déterministe
- L'ECB n'est même pas EAV-sûr. Pourquoi ?

Mode Dictionnaire de codes [🇬🇧 Electronic Code Book (ECB) mode]

Dictionnaire de codes (ECB)

Application directe naïve du chiffrement par blocs à chaque bloc de texte en clair

$$c := F_k(m_1), F_k(m_2), \dots, F_k(m_\ell).$$



- L'ECB n'est pas CPA-sûr car il est déterministe
- L'ECB n'est même pas EAV-sûr. Pourquoi ?

Réponse : Un **bloc répété** dans le texte en clair donne un bloc répété dans le texte chiffré

Chiffrement d'une image avec ECB



Image originale à gauche, image chiffrée avec ECB au centre, chiffrement sûr à droite.

Le mode ECB ne doit **jamaïs** être utilisé !

Mode Enchaînement des blocs [🇬🇧 Cipher Block Chaining (CBC) mode]

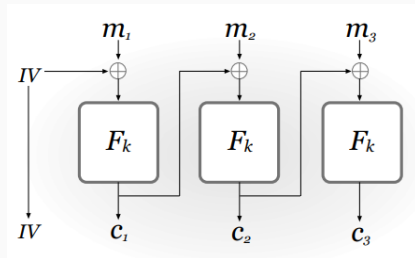
Enchaînement des blocs (CBC)

- Un IV uniforme est choisi comme premier texte chiffré :

$$c_0 := IV.$$

- Les textes chiffrés sont générés en appliquant le chiffrement par blocs au XOR du bloc de texte en clair actuel et du bloc de texte chiffré précédent.

$$c_i := F_k(c_{i-1} \oplus m_i).$$



- + Si F est une permutation pseudo-aléatoire, alors le mode CBC est CPA-sûr
- Le chiffrement doit être effectué séquentiellement
→ ce n'est pas la meilleure option si le parallélisme est disponible

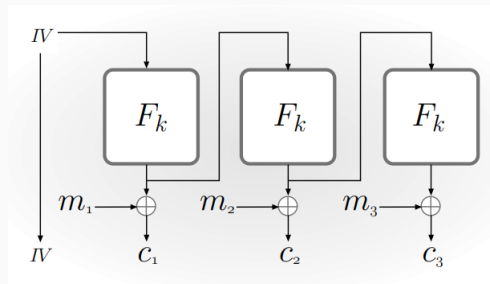
Mode de rétroaction de sortie [🇬🇧 Output Feedback (OFB) mode]

Rétroaction de sortie (OFB)

Construit comme un chiffrement de flux

- $y_0 := IV$
- $y_i := F_k(y_{i-1})$
- $c_i := y_i \oplus m_i$

IV est inclus dans le texte chiffré (pour permettre le déchiffrement).



- + F n'a pas besoin d'être une permutation
- + Le flux peut être tronqué → texte en clair de n'importe quelle longueur
- + CPA-sûr si F est une fonction pseudo-aléatoire
- + Le flux pseudo-aléatoire peut être précalculé (toujours séquentiel)

Mode Compteur [🇬🇧 Counter (CTR) mode]

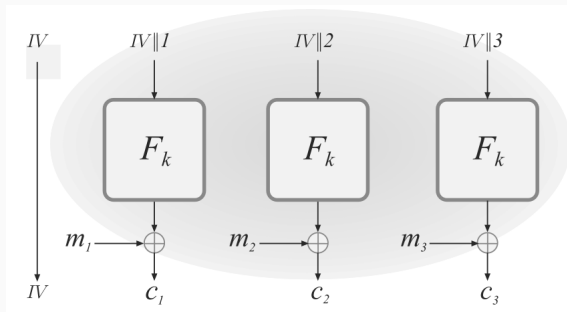
Compteur (CTR)

Construit comme un chiff. de flux

- Choisissez $IV \in \{0, 1\}^{3n/4}$
- $y_i := F_k(IV \parallel \langle i \rangle)$, avec le **compteur** i encodé sous forme de chaîne de $(n/4)$ bits
- $c_i := y_i \oplus m_i$

IV est inclus dans le texte chiffré.

- + F n'a pas besoin d'être une permutation
- + Texte en clair de n'importe quelle longueur
- + CPA-sûr si F est une fonction pseudo-aléatoire
- + Enc et Dec peuvent être **entièrement parallélisés**



Choix intéressant en pratique !

Vers une instanciation pratique des chiffrements par blocs

Le **paradigme de confusion-diffusion** [Shannon, 1949]

1. **Idée** : construire une permutation F à l'apparence aléatoire avec une grande longueur de bloc à partir de **nombreuses permutations $\{f_i\}$ à l'apparence aléatoire et de petite longueur de bloc.**

Exemple : F doit avoir une longueur de bloc de 128 bits.

- La clé k pour F spécifie 16 perm.s $f_1, \dots, f_{16}$ de longueur 8 bits ($128 = 16 \cdot 8$).
- Étant donné $x \in \{0, 1\}^{128}$, avec $x = x_1 \dots x_{16}$ et définissons

$$F_k(x) = f_1(x_1) || \dots || f_{16}(x_{16}).$$

On dit que les **fonctions de tour/ronde** [ round functions] $\{f_i\}$ introduisent une **confusion** dans F .

2. **Problème** : modifier un bit de l'entrée n'affecte qu'un seul octet de la sortie.

Solution : après l'étape de confusion, les bits de la sortie sont permutés, c.-à-d. qu'ils sont **mélangés** → les changements locaux sont **diffusés** dans tout le bloc

Ronde = étape de confusion + étape de diffusion

Réseaux de substitution-permutation [🇬🇧 Substitution-permutation networks]

Les réseaux de substitution-permutation (SPN) sont une implémentation du paradigme confusion-diffusion, où

- la fonction de substitution (c'est-à-dire la permutation) S est fixe et appelée **S-box**
- la clé k définit la fonction $f(x) = S(k \oplus x)$

De nombreux tours sont **répétés** : l'entrée est la sortie du tour précédent

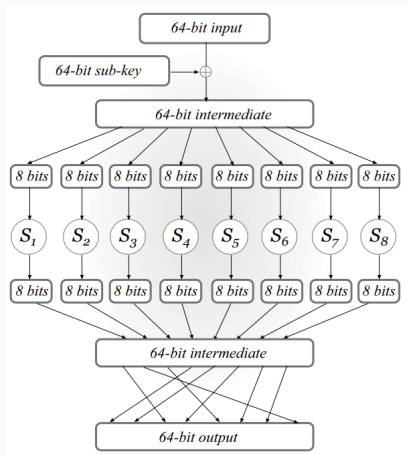
Par conséquent, un **tour** comprend :

1. **Mélange de clés** : [🇬🇧 Key-mixing] Définissez $x := x \oplus k$, où k est la **sous-clé/clé de tour** [🇬🇧 round key] actuelle
2. **Substitution** : Définir $x := S_1(x_1) || \dots || S_j(x_j)$
3. **Permutation de mélange** : [🇬🇧 Mixing permutation] Permuter les bits de x pour obtenir la sortie du tour

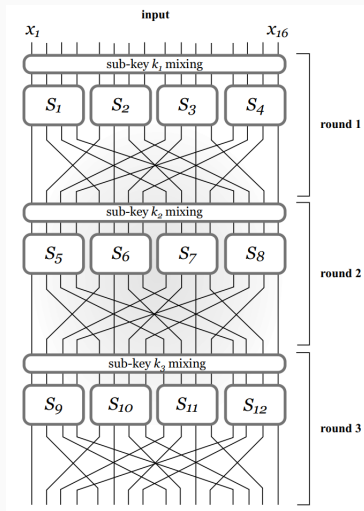
⚠ Selon le principe de Kerchoff, l'S-box et la permutation de mélange sont **publiques**

Si un SPN n'a pas suffisamment de tours, il peut être facilement cassé

Un tour pour une longueur de bloc de 64 bits et des S-box de 8 bits



SPN avec 3 tours pour une longueur de bloc de 16 bits et des S-box de 4 bits



Clés :

- le chiffrement par blocs dispose d'une **clé maîtresse/principale** [🇬🇧 master key]
- les **clés de tour** sont dérivées de la clé principale selon une **preparation de clés** [🇬🇧 key schedule], par exemple en utilisant différents sous-ensembles de la clé principale comme clés de tour
- étant donné une clé, le SPN peut être **inversé**, quel que soit le nombre de tours, en commençant par le dernier tour et en remontant :
 - La permutation de mélange peut être inversée car il s'agit simplement d'un réordonnancement des bits
 - la *S*-box est une permutation
 - Le XOR de la sous-clé donne l'entrée d'origine

Effet avalanche : afin d'induire de grands changements dans la sortie à partir de petits changements dans l'entrée, nous avons besoin :

1. la **S-box** est telle que la modification d'un bit dans l'entrée modifie toujours au moins 2 bits dans la sortie de la *S*-box
2. Les bits sortis par une *S*-box doivent affecter l'entrée de plusieurs *S*-box après la **permutation de mélange**

Réseaux de Feistel [🇬🇧 Feistel network]

Réseaux de Feistel : contrairement aux S-boxes dans les SPN, ils peuvent utiliser des **composants non inversibles** f_i pour construire une fonction inversible.

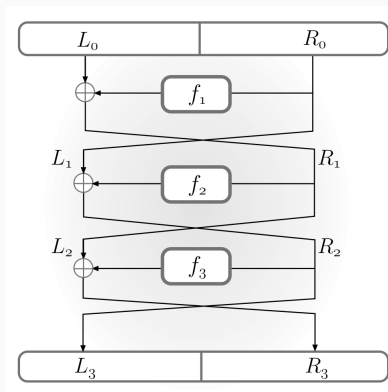
Plusieurs tours, au tour i :

- l'entrée de ℓ bits du tour est divisée en 2 moitiés (gauche/droite) L_{i-1}, R_{i-1}
- la sortie (L_i, R_i) est
$$L_i := R_{i-1}, \quad \text{et} \quad R_i := L_{i-1} \oplus f_i(R_{i-1}).$$

Un réseau de Feistel peut toujours être **inversé**

- $L_{i-1} := R_i \oplus f_i(R_{i-1})$ (**pas besoin d'inverser f_i**),

Exemple : Réseau de Feistel à 3 tours



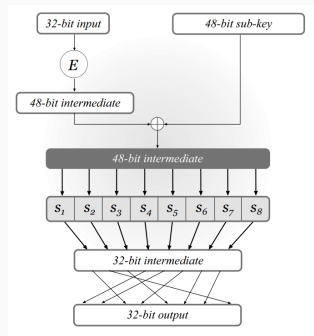
Les réseaux de Feistel sont également vulnérables si peu de tours.

Data Encryption Standard (DES) :

- développé dans les 1970s par IBM et adopté comme **standard** par USA
- très bon **effet avalanche**
- peu sûr aujourd'hui, car la **clé est trop courte**

Description générale :

- Réseau de Feistel à 16 tours, même fonction de tour f
- longueur de bloc de 64 bits et longueur de clé de 56 bits
- f prend une sous-clé de 48 bits (en tant que sous-ensemble permuté de la clé principale de 56 bits) et une entrée de 32 bits
- f est essentiellement un SPN, mais les “S-boxes” ne sont pas inversibles
- Une fonction E étend l'entrée à 48 bits avant le XOR et les S-boxes mappent 6 bits en 4 bits.



DES en tant que boîte noire

Résumé du DES : + très bonne conception – clé courte

Idée : Utiliser DES comme boîte noire pour construire un nouveau chiffrement par blocs

Double chiffrement : étant donné un chiffrement par blocs F avec une longueur de clé de n bits et une longueur de bloc de ℓ bits, construire un chiffrement par blocs F' avec une longueur de clé de $2n$ bits comme suit :

$$F'_{k_1, k_2}(x) \stackrel{\text{def}}{=} F_{k_2}(F_{k_1}(x))$$

Supposons que la meilleure attaque contre F soit la recherche exhaustive. Quelle est la sécurité de F' ? Une attaque triviale nécessite 2^{2n} . Toutefois, **l'attaque par rencontre au milieu** [🇬🇧 Meet-in-the-middle (MITM) attack] ne nécessite qu'environ $\approx 2^n$:

1. Pour chaque $k_1 \in \{0, 1\}^n$, calculez $z := F_{k_1}(x)$ et stockez (z, k_1) dans une liste L .
2. Pour chaque $k_2 \in \{0, 1\}^n$, calculez $z := F_{k_2}^{-1}(y)$ et stockez (z, k_2) dans une liste L' .
3. (k_1, k_2) est une solution si $(z, k_1) \in L$ et $(z, k_2) \in L'$ pour un certain z .

Triple chiffrement :

- Variante 1 (3 clés) : $F''_{k_1, k_2, k_3}(x) = F_{k_3}(F_{k_2}^{-1}(F_{k_1}(x)))$
- Variante 2 (2 clés) : $F''_{k_1, k_2}(x) = F_{k_1}(F_{k_2}^{-1}(F_{k_1}(x)))$

⚠ Pourquoi $F_{k_2}^{-1}$? Pour des raisons de compatibilité ascendante : si $k_1 = k_2 = k_3$, alors $F_{k_1} = F''_{k_2}$.

3DES, **standardisé** en 1999, est basé sur 3 invocations de DES

Sécurité :

- Variante 2 : la longueur de la clé est $2n$, une attaque nécessite un temps 2^{2n} , mais meilleure attaque par texte clair connu avec de nombreuses paires d'entrée/sortie
→ n'est plus recommandée
- Variante 1 : la longueur de la clé est $3n$, mais MITM nécessite un temps 2^{2n}
→ toujours utilisée aujourd'hui, mais en voie de disparition car relativement lente

Advanced Encryption Standard (AES) :

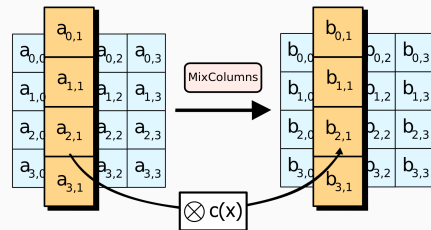
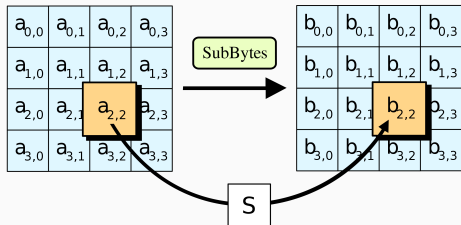
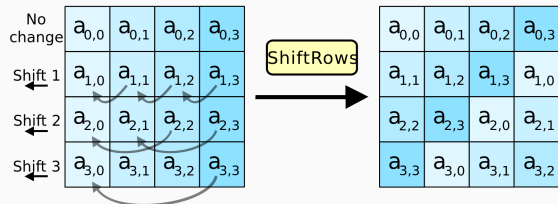
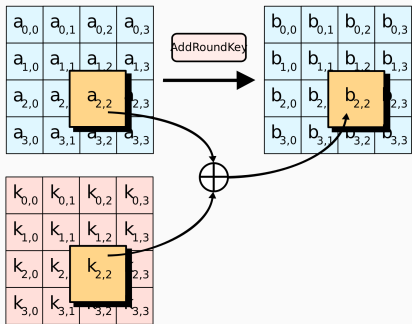
- Concours lancé en 1997 par le NIST (National Institute of Standards and Technology américain) pour **remplacer DES**
- 15 candidats soumis du monde entier
- De nombreux chercheurs ont tenté de les cryptanalyser
- **Rijndael** chiffrement par blocs [Rijmen, Daemen, 1998] sélectionné comme gagnant
 - essentiellement un réseau de substitution-permutation
 - **3 variantes** (AES-128, AES-192, AES-256) en fonction de la longueur de la clé
 - La longueur de la clé affecte le **nombre de tours** (10, 12, 14 respectivement) et preparation de clés, mais pas la structure de haut niveau
 - **Aucune faiblesse significative**

Un tour d'AES

Blocs de 128 bits.

1. `AddRoundKey` : une sous-clé de 128 bits est dérivée de la clé principale et considérée comme un tableau de 4×4 octets, puis soumise à une opération XOR avec l'état
 2. `SubBytes` : une *S*-box fixe est appliquée à chaque octet
 3. `ShiftRows` : les octets de chaque ligne sont décalés cycliquement de 0, 1, 2, 3 vers la gauche pour les 1ère, 2ème, 3ème et 4ème lignes respectivement
 4. `MixColumns` : transformation linéaire inversible appliquée à 4 octets dans chaque colonne
- Étapes 3+4 = permutation du tour SPN
 - dans le tour final, `MixColumns` est remplacé par `AddRoundKey`.

([Lien](#) vers une animation de Rijndael)



Cryptanalyse des chiffrements par blocs

Cryptanalyse différentielle : Tabuler des différences dans l'entrée qui conduisent à des différences spécifiques dans la sortie avec une proba "élevée".

- Introduite par Biham et Shamir dans les années 1980
- Peut être utilisée pour lancer une **attaque par récupération de clé sur les SPN**
- Utilisée pour attaquer DES en 1993, nécessite 2^{47} paires (texte clair, texte chiffré) avec **texte clair choisi** (pas *réaliste/pratique*)
- Beaucoup plus dévastatrice sur d'autres chiffrements par blocs

Cryptanalyse linéaire : Considérer les relations linéaires entre l'entrée, la sortie et la clé qui se vérifient avec une proba élevée

- Introduite par Matsui dans les années 1990
- Elle ne nécessite que des **textes en clair connus** au lieu de textes en clair choisis
- L'attaque contre DES nécessite 2^{43} paires (texte en clair, texte chiffré)

Les chiffrements par blocs modernes sont conçus de manière à résister à ces attaques