

## Table of contents

Série 1 : Arithmétique Modulaire . . . . .	1
Notions Clés . . . . .	1
Série 2 : Entropie . . . . .	2
Notions Clés . . . . .	2
Série 3 : Chiffrements Historiques . . . . .	3
Notions Clés . . . . .	3
Série 4 : Chiffrements par Blocs . . . . .	4
Notions Clés . . . . .	4
Série 5 : RSA, Rabin, ElGamal . . . . .	7
Notions Clés . . . . .	7
Série 6 : Fonctions de Hachage et MAC . . . . .	8
Notions Clés . . . . .	8
Série 7 : Authentification et Établissement de Clés . . . . .	9
Notions Clés . . . . .	9
Aide-Mémoire Express . . . . .	12

## Série 1 : Arithmétique Modulaire

### Notions Clés

**Ensembles** :  $\mathbb{Z}_n = \{0, 1, \dots, n-1\}$ ,  $\mathbb{Z}_n^* = \{a \in \mathbb{Z}_n \mid \text{pgcd}(a, n) = 1\}$

**Congruence** :  $a \equiv b \pmod{n} \iff n \mid (a - b)$

**Inversibilité** :  $a$  inversible mod  $n \iff \text{pgcd}(a, n) = 1$

**Théorèmes fondamentaux** :

- **Bézout** :  $ax + by = \text{pgcd}(a, b)$
- **Euler** :  $a^{\Phi(n)} \equiv 1 \pmod{n}$  (si  $\text{pgcd}(a, n) = 1$ )
- **Fermat** :  $a^p \equiv a \pmod{p}$  ( $p$  premier)

**Ordre** :  $\text{ord}_n(a) = \text{plus petit } x > 0 \text{ tel que } a^x \equiv 1 \pmod{n}$

**Générateur** :  $g$  génère  $\mathbb{Z}_n^*$  si  $\text{ord}_n(g) = \Phi(n)$

**Structures** : Groupe  $\rightarrow$  Anneau  $\rightarrow$  Corps (inversibilité croissante)

### Calculs modulaires

**Q** :  $((11 \pmod{7}) \cdot (17 \pmod{7})) \pmod{7}$

**R** :  $4 \cdot 3 = 12 \equiv 5 \pmod{7}$

### Trouver l'ordre

**Q** : Ordre de 2 mod 7 ?

**R** :  $2^1 = 2, 2^2 = 4, 2^3 = 8 \equiv 1 \rightarrow \text{ord}_7(2) = 3$

### Identifier un générateur

**Q** : 3 est-il générateur de  $\mathbb{Z}_7^*$  ?

**R** :  $3^1 = 3, 3^2 = 2, 3^3 = 6, 3^4 = 4, 3^5 = 5, 3^6 = 1 \rightarrow$  génère tous les éléments  $\rightarrow$  **OUI**

---

## Série 2 : Entropie

### Notions Clés

**Entropie** : Mesure l'incertitude/information d'une variable aléatoire

$$H(X) = - \sum_{i=1}^n p_i \log_2(p_i) = \sum_{i=1}^n p_i \log_2 \left( \frac{1}{p_i} \right)$$

**Propriétés** :

- $H(X)$  maximale quand toutes les probabilités sont égales
- $H(X) = 0$  si une seule valeur possible (probabilité = 1)
- Pour  $n$  valeurs équiprobables :  $H(X) = \log_2(n)$

**Entropie jointe** :  $H(X, Y) = - \sum_x \sum_y p(x, y) \log_2(p(x, y))$

**Entropie conditionnelle** :  $H(X|Y) = - \sum_y \sum_x p(y)p(x|y) \log_2(p(x|y))$

**En cryptographie** : On veut  $H(\text{Plaintext}|\text{Ciphertext}) \approx H(\text{Plaintext})$

### Entropie minimale/maximale

**Q** : Variable 256 bits, entropies min/max ?

**R** :

- **Min** :  $H = 0$  (une seule valeur possible,  $p = 1$ )
- **Max** :  $H = 256$  (toutes valeurs équiprobables,  $p = 2^{-256}$ )

### Entropie d'une concaténation

**Q** :  $H(X) = 64$ , on génère une valeur et la concatène à elle-même (512 bits). Entropie ?

**R** :  $H = 64$  (pas de nouvelle information, juste duplication)

### Entropie de mot de passe

**Q** : Mot de passe = date “MM/DD/YYYY” aléatoire (365 jours, années 0000-2025)

**R** :  $365 \times 2026 = 739490$  possibilités  $\rightarrow H = \log_2(739490) \approx 19.5$  bits

### Générateur amélioré

**Q** : Générateur  $G$  :  $P(0) = 0.5 + \delta$ ,  $P(1) = 0.5 - \delta$ . On crée  $A$  : prend 2 bits de  $G$ , garde 01  $\rightarrow$  0 ou 10  $\rightarrow$  1, rejette 00 et 11. Avantage ?

**R** :

- $P_A(0) = P_A(1) = 0.5$  (parfaitement aléatoire !)
- Coût : besoin de  $\frac{2x}{0.5-2\delta^2}$  bits de  $G$  pour  $x$  bits de  $A$

## Série 3 : Chiffrements Historiques

### Notions Clés

#### Chiffre de César :

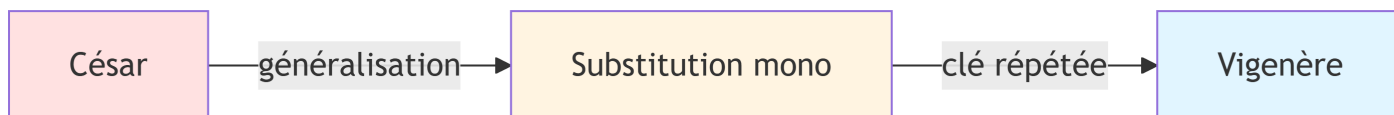
Rotation de  $k$  positions :  $E_k(x) = (x + k) \bmod 26$ ,  $D_k(c) = (c - k) \bmod 26$

#### Substitution monoalphabétique :

Clé = permutation de l'alphabet. Chaque lettre  $\rightarrow$  lettre fixe.

#### Chiffre de Vigenère :

Substitution polyalphabétique :  $C_i = (M_i + K_{i \bmod |K|}) \bmod 26$



#### Cassage :

- **César** : Force brute (25 clés max) ou analyse fréquentielle
- **Mono** : Analyse fréquentielle + structure du langage
- **Vigenère** : Indice de coïncidence + analyse fréquentielle

### Chiffrement César

**Q** : Chiffrer “HELLO” avec  $k = 5$

**R** :  $H \rightarrow M, E \rightarrow J, L \rightarrow Q, L \rightarrow Q, O \rightarrow T \rightarrow$  “MJQQT”

### Chiffrement Vigenère

**Q** : Chiffrer “BONJOUR” avec clé “BAC”

**R** :

- $B+B=C, O+A=O, N+C=P, J+B=K, O+A=O, U+C=W, R+B=S$
- “COPKOWS”

### Cassage Vigenère - Longueur de clé

**Méthode** : Indice de coïncidence

Pour longueur  $L$ , décaler le texte de  $L$  positions et compter les lettres identiques :

$$IC(L) = \frac{\sum_{i=1}^{N-L} [a_i == b_i]}{N - L}$$

Le maximum d'IC indique la longueur de clé (ou un multiple).

### Cassage Vigenère - Trouver la clé

**Méthode** : Analyse fréquentielle par sous-texte

1. Diviser le texte en  $k$  sous-textes (positions  $1, k+1, 2k+1, \dots$ )
2. Pour chaque sous-texte, calculer distance avec fréquences de la langue :

$$\text{Dist}_x = \sqrt{\sum_{i=0}^{25} (F_i - M_{(i+x) \bmod 26})^2}$$

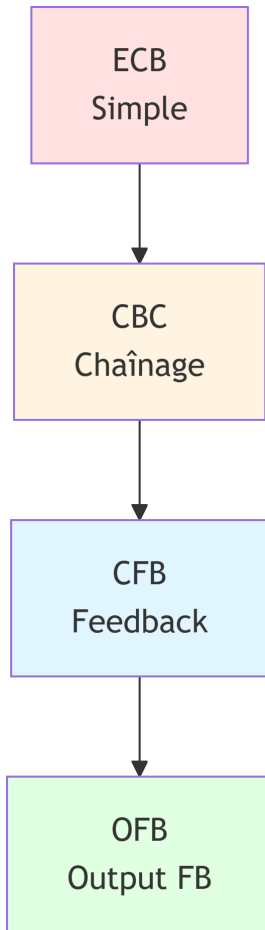
3. Le  $x$  minimisant la distance est la lettre de clé correspondante

---

## Série 4 : Chiffrements par Blocs

### Notions Clés

Modes de chiffrement :



**ECB (Electronic CodeBook) :**

$$C_i = E_K(P_i)$$

Blocs identiques  $\rightarrow$  chiffrés identiques (faible sécurité)

**CBC (Cipher Block Chaining) :**

$$C_i = E_K(P_i \oplus C_{i-1}), \quad C_0 = IV$$

**CFB (Cipher FeedBack) :**

$$C_i = E_K(C_{i-1}) \oplus P_i, \quad C_0 = IV$$

**OFB (Output FeedBack) :**

$$O_i = E_K(O_{i-1}), \quad C_i = O_i \oplus P_i, \quad O_0 = IV$$

**Fonction de chiffrement** : Doit être **inversible** (bijective)

#### Chiffrement linéaire - Danger

**Q** : Chiffrement linéaire  $E_L(k, m_1 \oplus m_2) = E_L(k, m_1) \oplus E_L(k, m_2)$ . Avec 128 textes chiffrés choisis, montrer qu'on peut déchiffrer sans clé.

**R** :

1. Choisir  $c_1, \dots, c_{128}$  où  $c_i$  a seulement le bit  $i$  à 1
2. Tout chiffré  $c$  s'écrit comme XOR de certains  $c_i$
3.  $c = c_{i_1} \oplus \dots \oplus c_{i_n} = E_L(k, m_{i_1} \oplus \dots \oplus m_{i_n})$
4. Donc  $m = m_{i_1} \oplus \dots \oplus m_{i_n}$  (connu !)
5. **Conclusion** : Chiffrement linéaire = très dangereux

#### Fonctions inversibles

**Q** :  $E_i = (B_i \cdot K_i) \bmod 16$  est-elle utilisable ?

**R** : **NON**. Si  $K_i = 2$ , alors  $B_i = 1$  et  $B_i = 9$  donnent tous deux  $E_i = 2 \bmod 16$ . Non-bijective !

#### Chiffrement ECB

**Q** :  $K = (AB)_{16}$ ,  $m = (A741BA)_{16}$ ,  $E_K(B) = B \oplus K$ , chiffrer

**R** :

- $C_1 = A7 \oplus AB = 0C$
- $C_2 = 41 \oplus AB = EA$
- $C_3 = BA \oplus AB = 11$
- **Résultat** :  $(0CEA11)_{16}$

#### Chiffrement CBC

**Q** :  $K = (AB)_{16}$ ,  $IV = (AD)_{16}$ ,  $m = (A741BA)_{16}$ ,  $E_K(B) = B \oplus K$

**R** :

- $C_1 = (A7 \oplus AD) \oplus AB = 0A \oplus AB = A1$
- $C_2 = (41 \oplus A1) \oplus AB = E0 \oplus AB = 4B$
- $C_3 = (BA \oplus 4B) \oplus AB = F1 \oplus AB = 5A$
- **Résultat** :  $(A14B5A)_{16}$

## Série 5 : RSA, Rabin, ElGamal

### Notions Clés

#### RSA :

- **Clés** :  $n = pq$ ,  $e$  avec  $\text{pgcd}(e, \Phi(n)) = 1$ ,  $d = e^{-1} \mod \Phi(n)$
- **Chiffrement** :  $c = m^e \mod n$
- **Déchiffrement** :  $m = c^d \mod n$

**Exponentiation rapide** : Calculer  $a^{42}$  : écrire  $42 = 32 + 8 + 2$  puis  $a^{42} = a^{32} \cdot a^8 \cdot a^2$

#### Rabin :

- **Clés** :  $n = pq$  avec  $p \equiv q \equiv 3 \mod 4$
- **Chiffrement** :  $c = m^2 \mod n$
- **Déchiffrement** : 4 solutions possibles via système de congruences

#### ElGamal :

- **Clés** : Premier  $p$ , générateur  $\alpha$ , clé privée  $a$ , clé publique  $\alpha^a \mod p$
- **Chiffrement** :  $(\lambda, \sigma) = (\alpha^k, m \cdot (\alpha^a)^k) \mod p$
- **Déchiffrement** :  $m = \lambda^{-a} \cdot \sigma \mod p$

#### Générer clés RSA

**Q** :  $p = 11$ ,  $q = 17$ , créer paire de clés RSA

**R** :

1.  $n = 11 \times 17 = 187$
2.  $\Phi(n) = 10 \times 16 = 160$
3. Choisir  $e = 7$  (premier avec 160)
4. Trouver  $d$  :  $7d \equiv 1 \mod 160 \rightarrow d = 23$
5. **Clé publique** :  $(187, 7)$ , **Clé privée** :  $(187, 23)$

#### Chiffrement RSA rapide

**Q** : Chiffrer  $m = 28$  avec  $(n = 247, e = 41)$

**R** : Exponentiation rapide  $28^{41} \mod 247$  :

- $28^1 = 28$ ,  $28^2 = 43$ ,  $28^4 = 120$ ,  $28^8 = 74$ ,  $28^{16} = 42$ ,  $28^{32} = 35$
- $41 = 32 + 8 + 1$  donc  $28^{41} = 35 \cdot 74 \cdot 28 = 149 \mod 247$

#### Casser RSA (petits nombres)

**Q** :  $(n = 247, e = 41)$ , trouver clé privée

**R :**

1. Factoriser :  $247 = 13 \times 19$
2.  $\Phi(n) = 12 \times 18 = 216$
3. Euclide étendu pour  $d = e^{-1} \pmod{216} \rightarrow d = 137$
4. Vérifier :  $41 \times 137 = 5617 = 26 \times 216 + 1 \equiv 1 \pmod{216}$

### Rabin

**Q :**  $n = 253$ , chiffrer  $m = 134$

**R :**  $c = 134^2 = 17956 \equiv 246 \pmod{253}$

Pour déchiffrer (factoriser  $n = 11 \times 23$ ) :

- $m_p = 246^3 \pmod{11} = 9$
- $m_q = 246^6 \pmod{23} = 4$
- 4 solutions dont  $m_4 = 134$

---

## Série 6 : Fonctions de Hachage et MAC

### Notions Clés

#### Propriétés cryptographiques :

1. **Résistance à la préimage :** Difficile de trouver  $x$  tel que  $h(x) = y$
2. **Résistance à la seconde préimage :** Difficile de trouver  $x' \neq x$  avec  $h(x') = h(x)$
3. **Résistance aux collisions :** Difficile de trouver  $x \neq x'$  avec  $h(x) = h(x')$

Collision implique seconde préimage (mais pas préimage)

#### MAC (Message Authentication Code) :

Garantit intégrité ET authenticité. Construit souvent avec CBC :  $MAC = E_K(\dots E_K(E_K(m_1) \oplus m_2) \dots \oplus m_n)$

### Mauvaise fonction de hachage

**Q :**  $h_1(x) = x \pmod{n}$  est-elle sûre ?

**R :** **NON** pour les 3 propriétés :

- Préimage :  $x = y$  donne  $h_1(x) = y$
- Seconde préimage :  $x' = x + n$  donne collision
- Collision : idem seconde préimage



### MAC avec CBC vulnérable

**Q :**  $t_1 = E_K(m_1)$ ,  $t_{i+1} = E_K(m_{i+1} \oplus t_i)$ . Avec  $(m_1 || m_2, t_1 || t_2)$ , falsifier ?

**R :** Message falsifié :  $m' = (m_2 \oplus t_1) || (t_2 \oplus m_1)$

MAC falsifié :  $t' = t_2 || t_1$  (calculable sans clé !)

### MAC = dernier bloc CBC

**Q :** Si  $MAC = c_n$  (dernier bloc CBC), peut-on modifier le message ?

**R : OUI !** On peut modifier tous les blocs  $c_1, \dots, c_{n-1}$  sans changer  $c_n = MAC$ . Le déchiffrement donnera un message différent avec MAC valide !

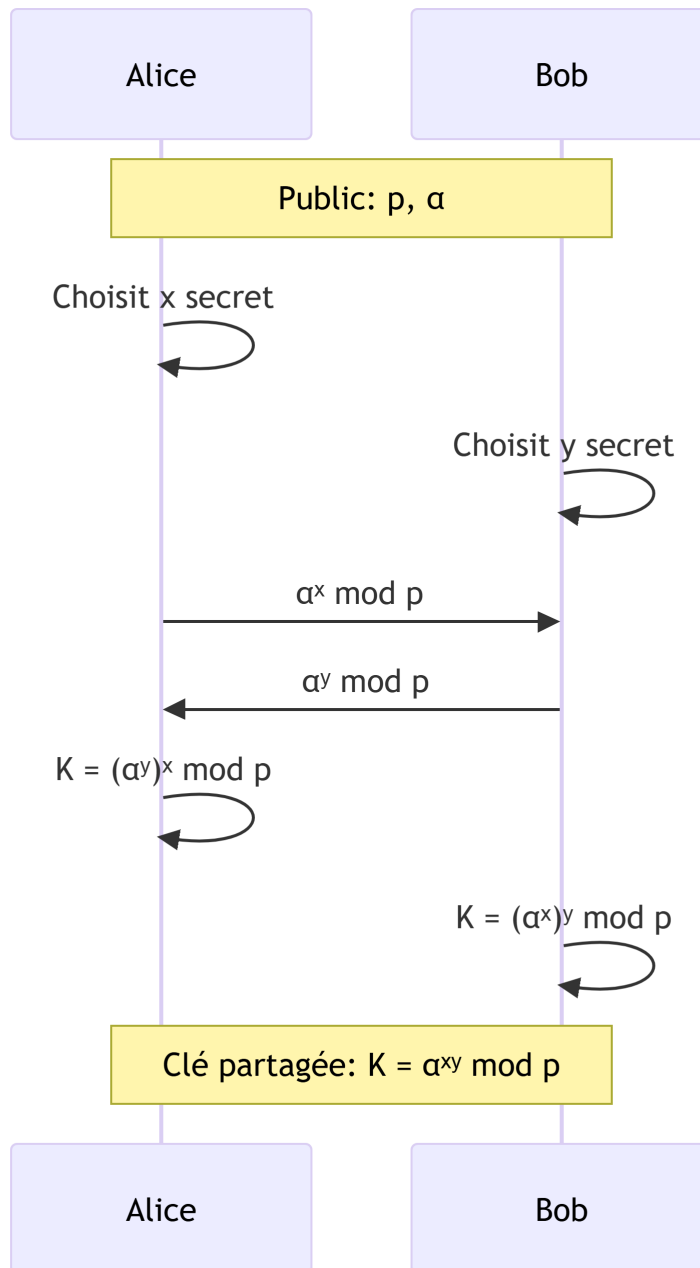
**Solution :** Utiliser deux clés différentes (une pour chiffrement, une pour MAC)

---

## Série 7 : Authentification et Établissement de Clés

### Notions Clés

Diffie-Hellman :



**Attaque Man-In-The-Middle sur DH :** Intercepter et remplacer les échanges

**Propriétés de sécurité :**

- **Authentification implicite :** Seuls A et B peuvent avoir la clé
- **Confirmation de clé :** A et B prouvent qu'ils ont la clé

- **Authentication explicite** : Implicite + Confirmation
- **Perfect Forward Secrecy** : Compromission de clés long-terme ne révèle pas sessions passées
- **Future Secrecy** : Compromission ne révèle pas futures sessions (attaquant passif)

#### Protocole d'authentification faible

**Q** : A envoie  $r_1$  à B, B répond  $(r_2, K_B^{priv}(r_1))$ , A vérifie et envoie  $K_A^{priv}(r_2)$ . Comment C peut usurper A ?

**R** :

1. C envoie  $r_1$  à B
2. B répond  $(r_2, K_B^{priv}(r_1))$
3. C lance protocole avec A, envoie  $r_2$  comme challenge
4. A répond  $(r_3, K_A^{priv}(r_2))$
5. C envoie  $K_A^{priv}(r_2)$  à B → **B authentifie C comme A !**

#### Diffie-Hellman complet

**Q** :  $p = 17$ ,  $\alpha = 3$ , Alice  $x = 7$ , Bob  $y = 11$ . Calculer clé partagée.

**R** :

- Alice calcule et envoie :  $3^7 \bmod 17 = 11$
- Bob calcule et envoie :  $3^{11} \bmod 17 = 7$
- Alice calcule :  $K = 7^7 \bmod 17 = 12$
- Bob calcule :  $K = 11^{11} \bmod 17 = 12$
- **Clé partagée** :  $K = 12$

#### Man-In-The-Middle sur DH

**Q** : Charlie (MitM) avec  $x' = 3$ ,  $y' = 5$ . Comment intercepter ?

**R** :

**Avec Alice :**

- Intercepte  $\alpha^x = 11$ , répond  $\alpha^{y'} = 3^5 = 5$
- $K_{AC} = 5^7 = 10 \bmod 17$

**Avec Bob :**

- Intercepte  $\alpha^y = 7$ , répond  $\alpha^{x'} = 3^3 = 10$
- $K_{BC} = 10^{11} = 3 \bmod 17$

Charlie a 2 clés et contrôle totalement la communication !

### Analyse de protocole

**Q :** A et B partagent  $S$ , échangent  $r_a$  et  $r_b$ , puis  $K = E_S(r_a \oplus r_b)$ . Analyser les propriétés.

**R :**

- **Authentication implicite** (seuls A et B connaissent  $S$ )
- **Confirmation de clé** (pas de preuve de possession)
- **Authentication explicite** (pas de confirmation)
- **Perfect Forward Secrecy** (attaquant avec  $S$  décrypte tout)
- **Future Secrecy** (attaquant passif avec  $S$  calcule futures clés)

---

### Aide-Mémoire Express

**Arithmétique :**  $a^{\Phi(n)} \equiv 1 \pmod n$  | Générateur si  $\text{ord}(g) = \Phi(n)$

**Entropie :**  $H = \log_2(n)$  si équiprobable | Max quand uniforme

**César :**  $E(x) = (x + k) \pmod{26}$  | Cassage : 26 essais

**Vigenère :** IC pour longueur, fréquences pour clé

**Blocs :** ECB simple, CBC chaîné, CFB/OFB feedback | Fonction doit être bijective

**RSA :**  $c = m^e$ ,  $m = c^d$  |  $ed \equiv 1 \pmod{\Phi(n)}$

**Hash :** Préimage < Seconde préimage < Collision

**DH :**  $K = \alpha^{xy} \pmod p$  | Vulnérable au MitM