



# Information, Calcul et Communication

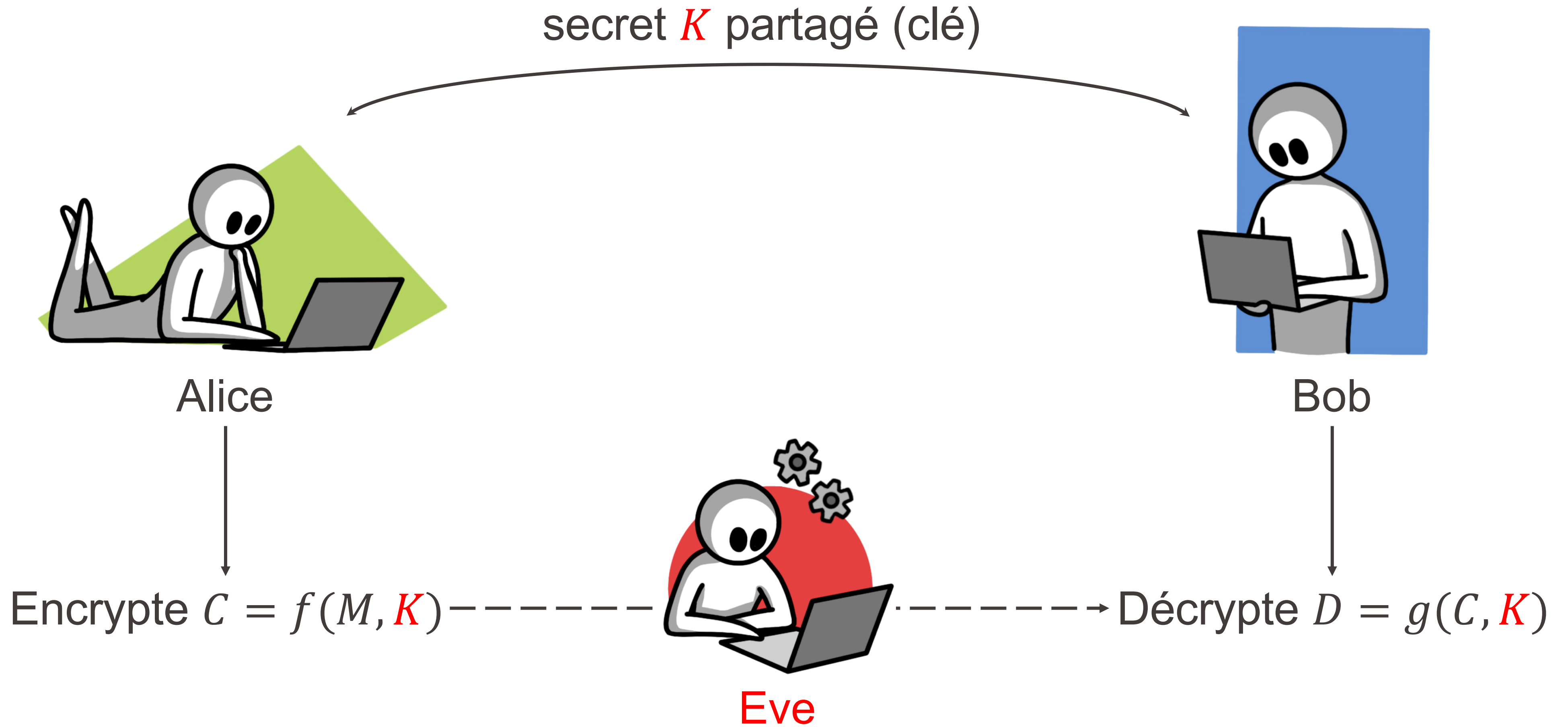
Cryptographie à  
clé secrète

Olivier Lévêque

# EPFL Scénario



# EPFL Scénario



Eve intercepte  $C$  mais ne sait pas trop quoi en faire sans la clé  $K$ ...

# Clé à usage unique («*one-time pad*»)

- Voici une recette 100% sûre pour encrypter un message  $M$  de  $n$  bits :

Pour cela, il faut supposer que la clé secrète  $K$  est composée également de  $n$  bits **et** générée de la façon suivante :

Chacun des bits  $K_i$  est tiré uniformément au hasard

$$\text{(i.e. } P(K_i = 1) = P(K_i = 0) = 1/2)$$

et tous les tirages sont effectués indépendamment,  
et aussi indépendamment du message  $M$ .

- Alice envoie alors  $C = M \oplus K$  (XOR bit par bit : pas de retenue ici)
  - Exemple : Si  $M = 01101101$  et  $K = 11101100$ , alors  $C = 10000001$

# Clé à usage unique («*one-time pad*»)

- Pour décrypter le message, Bob effectue l'opération :

$$D = C \oplus K = (M \oplus K) \oplus K = M \oplus \underbrace{(K \oplus K)}_{= \text{séquence de 0 !}} = M$$

- Si Eve intercepte  $C$ , elle ne peut rien faire car :

$$P(C_i = 0) = P(M_i \oplus K_i = 0) = P(K_i = M_i) = \frac{1}{2} \quad \forall M_i$$

$$P(C_i = 1) = P(M_i \oplus K_i = 1) = P(K_i \neq M_i) = \frac{1}{2} \quad \forall M_i$$

- Pour Eve, le message  $C$  est une séquence de bits tirés uniformément au hasard ! → Le système est sûr à 100% !

# Défauts de ce système

- Pour envoyer un message de longueur  $n$ , il faut générer une clé de même longueur, qu'il faut trouver le moyen de partager secrètement avant de communiquer...
- Si la clé n'est pas générée parfaitement aléatoirement, alors le secret n'est plus assuré à 100% (imaginez par exemple que  $K = 00000000$ , dans le pire des cas...)
- Gare à ne pas réutiliser la même clé  $K$  pour envoyer deux messages  $M_1$  et  $M_2$  à la suite avec ce système !

A partir de  $C_1 = M_1 \oplus K$  et  $C_2 = M_2 \oplus K$ , Eve peut effectuer :

$$C_1 \oplus C_2 = (M_1 \oplus K) \oplus (M_2 \oplus K) = (M_1 \oplus M_2) \oplus (K \oplus K) = M_1 \oplus M_2$$

→ plus aucune garantie de sécurité !

# Un essai pour réutiliser la clé $K$ :

## Data Encryption Standard (*DES*), 1976

**Principe (et seulement le principe, pas tout le système) :**

On suppose que le message  $M$  et la clé  $K$  sont l'un et l'autre de longueur  $2n$  bits et on décompose ceux-ci en :

$$M = (\underbrace{M_a}_{n \text{ bits}}, \underbrace{M_b}_{n \text{ bits}}), \quad K = (\underbrace{K_a}_{n \text{ bits}}, \underbrace{K_b}_{n \text{ bits}}) \quad (\text{NB: en pratique } n = 32 \text{ bits})$$

Soit  $f: \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$  une fonction **non-linéaire**.

$$K, M \mapsto f(K, M)$$

**Note :** Si on utilisait  $f(K, M) = K$ , le système décrit à la page suivante ne serait rien d'autre que le «one-time pad» décrit avant.

# Data Encryption Standard

- Alice calcule *successivement* :

$$C_a = M_a \oplus f(K_a, M_b) \quad \text{puis} \quad C_b = M_b \oplus f(K_b, C_a)$$

et envoie  $C = (C_a, C_b)$ .

- Comment Bob peut-il déchiffrer ce message ? Tout simplement, en fait !  
Il refait **les mêmes opérations dans l'ordre inverse** :

$$D_b = C_b \oplus f(K_b, C_a) \quad \text{puis} \quad D_a = C_a \oplus f(K_a, D_b)$$

- Ainsi, Bob retrouve  $M$ , car :

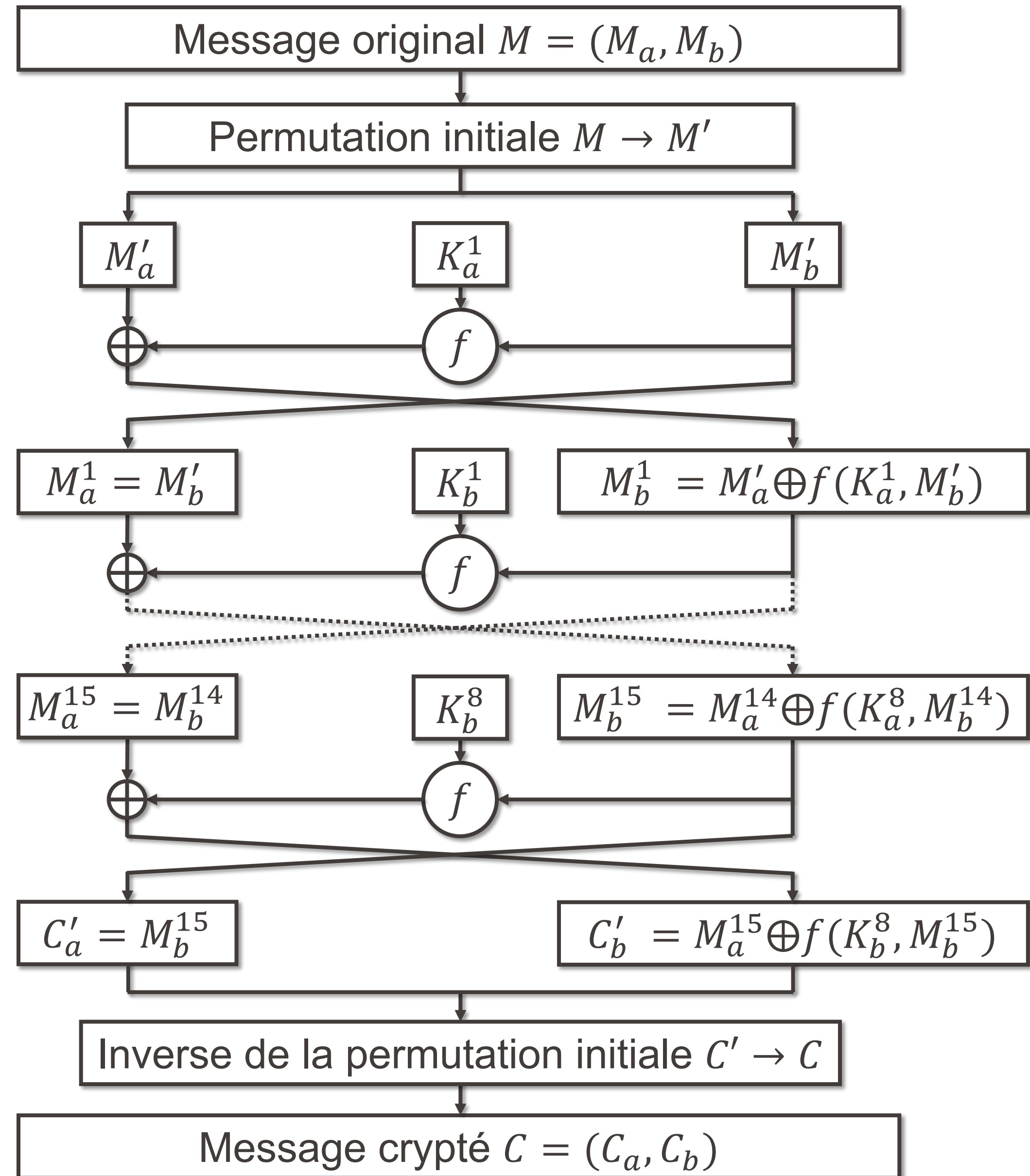
$$\begin{cases} D_b = C_b \oplus f(K_b, C_a) = M_b \oplus f(K_b, C_a) \oplus f(K_b, C_a) = M_b \\ D_a = C_a \oplus f(K_a, \underbrace{D_b}_{= M_b}) = M_a \oplus f(K_a, M_b) \oplus f(K_a, M_b) = M_a \end{cases}$$

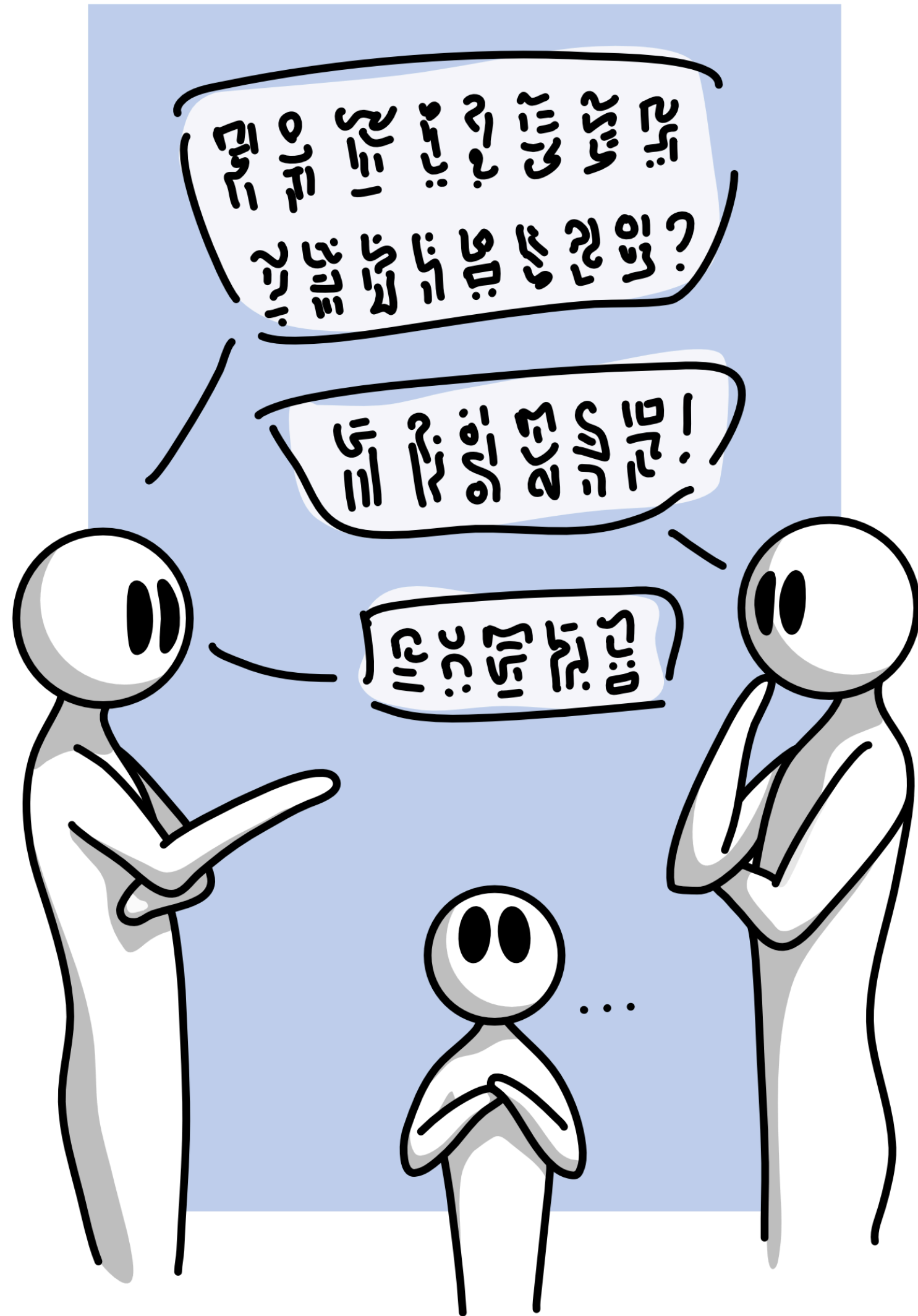


- Le système DES applique le principe ci-dessus **8 fois de suite** (avec une clé  $K$  8 fois plus longue) pour chiffrer un message, avec en plus une permutation du message à l'entrée et à la sortie (voir illustration page suivante).
- Ainsi, Alice peut chiffrer plusieurs messages  $M_1, M_2, \dots$  avec la même clé  $K$  et espérer que même si Eve intercepte  $C_1, C_2, \dots$  elle ne sera pas capable de retrouver les messages  $M_1, M_2, \dots$  sans la clé  $K$ .
- Pourtant, ce système de chiffrement a été « cassé » pour la première fois en 1999 et remplacé depuis par le système AES (Advanced Encryption Standard).

# Systeme DES complet

(Notez que le chiffrement et le déchiffrement s'effectuent à nouveau avec les mêmes opérations !)





# Information, Calcul et Communication

Cryptographie à clé publique

Olivier Lévêque

Alice et Bob désirent échanger des informations de manière confidentielle (et/ou envoyer des messages authentifiés) **sans** disposer d'une clé secrète  $K$  échangée au préalable.

==== (Comment) est-ce possible ? =====

- **Réponse 1**

Ça n'est pas possible !

- **Réponse 2**

C'est possible en pratique grâce aux **opérations difficilement inversibles** ou « **opérations à sens unique** ».

# Protocole: version simplifiée

**Exemple** : Supposons qu'il soit facile de multiplier mais difficile de diviser !

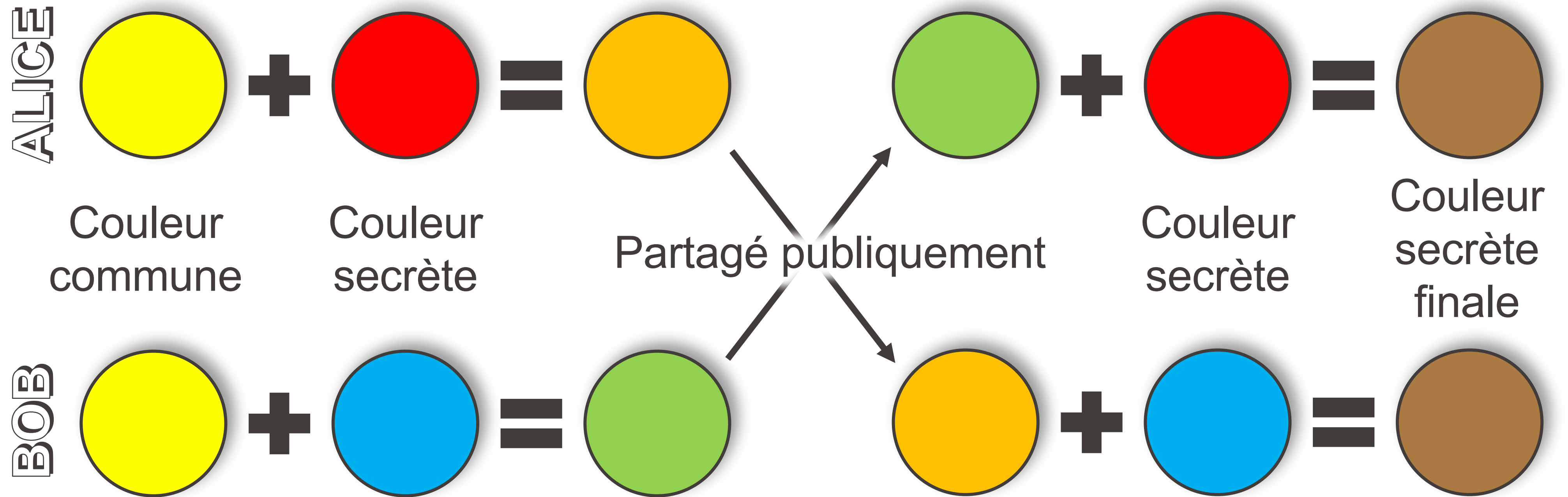
- Alice choisit un nombre  $N_1$   
Bob choisit un nombre  $N_2$
- Ils se mettent d'accord sur un 3<sup>ème</sup> nombre  $M$  (public)
- Alice calcule  $N_3 = N_1 \cdot M$  de son côté et envoie  $N_3$  à Bob  
Bob calcule  $N_4 = N_2 \cdot M$  de son côté et envoie  $N_4$  à Alice
- Alice calcule ensuite  $N_1 \cdot N_4 = N_1 \cdot N_2 \cdot M$   
Et Bob calcule  $N_2 \cdot N_3 = N_2 \cdot N_1 \cdot M$  ← égaux = secret partagé

Alice et Bob sont ainsi tombés d'accord sur un même nombre.

- Si Eve intercepte  $N_3$  ou  $N_4$  (ou même les deux nombres) et qu'elle ne sait pas diviser, elle ne peut pas retrouver le secret.

# La même chose avec des couleurs

- Les couleurs sont facilement mélangeables, mais difficilement séparables



# EPFL Arithmétique modulaire

- Bien sûr, diviser est une opération facile en réalité :

Eve peut donc effectuer  $N_3/M$  et retrouver  $N_1$ , de même que  $N_2$ .  
Par conséquent elle peut calculer le « secret »  $N_1 \cdot N_2 \cdot M$ .

---

---

Il faut donc trouver autre chose !

---

---

- Soit  $P$  un **grand** nombre premier. Sur l'ensemble  $\{0,1, \dots, P - 1\}$ , on définit l'addition, la multiplication et l'exponentiation **modulo  $P$**  :  
(toutes des opérations faciles à exécuter)

$$N_1 + N_2 \pmod{P}$$

$$N_1 \cdot N_2 \pmod{P}$$

$$N_1^{N_2} \pmod{P}$$

**Exemples :**  $4 + 3 \pmod{5} = 2$   
(avec  $P = 5$ )

$$4 \cdot 3 \pmod{5} = 2$$

$$4^3 \pmod{5} = 4$$

- Il se trouve que l'opération  $N_1^{N_2} \pmod{P} = N_3$  est difficile à **inverser** (i.e., si on nous donne  $P, N_1$  et  $N_3$ , il est difficile de retrouver  $N_2$ ).

Voilà donc l'**opération à sens unique** que nous allons utiliser.


## Remarques

- La difficulté de l'opération d'inversion dépend des nombres  $P$  et  $N_1$  choisis (c'est un long chapitre : celui du **logarithme discret**).
- Il existe par contre des algorithmes efficaces pour trouver de grands nombres premiers et donc réaliser concrètement ce qui va suivre !



# Protocole d'échange de clé de Diffie-Hellman

Utilisons cette opération à sens unique pour l'échange d'une clé secrète entre Alice et Bob :

- Alice et Bob choisissent d'abord ensemble un grand nombre premier  $P$  à  $n$  chiffres, ainsi qu'un autre nombre  $Q$  entre 1 et  $P - 1$ .  
( $P$  et  $Q$  sont donc publics)
- Alice choisit un nombre  $N_1$  entre 1 et  $P - 1$   
Bob choisit un nombre  $N_2$  entre 1 et  $P - 1$   secrètement
- Alice effectue  $N_3 = Q^{N_1} \pmod{P}$  et publie  $N_3$   
Bob effectue  $N_4 = Q^{N_2} \pmod{P}$  et publie  $N_4$
- Alice effectue ensuite  $N_4^{N_1} \pmod{P} = (Q^{N_2})^{N_1} \pmod{P} = Q^{N_1 \cdot N_2} \pmod{P} = K$   
Bob effectue quant à lui  $N_3^{N_2} \pmod{P} = (Q^{N_1})^{N_2} \pmod{P} = Q^{N_1 \cdot N_2} \pmod{P} = K$

- Alice et Bob ont ainsi trouvé une clé secrète commune  $K$  (= un grand nombre = une longue suite de bits).
- Notez également qu'Alice ne connaît toujours pas  $N_2$ , ni Bob ne connaît  $N_1$ .
- A partir de  $P, Q, N_3$  et  $N_4$ , il est difficile pour Eve de retrouver  $N_1$  ou  $N_2$ , car il faudrait inverser

$$N_3 = Q^{N_1}(\text{mod } P) \text{ ou } N_4 = Q^{N_2}(\text{mod } P)$$

et donc Eve n'a pas non plus accès au secret  $K$ .