

1. Introduction

1.1. Communications sécurisées

- Confidentialité des données transmises : importance
 - Contexte personnel 
 - **Protection de la vie privée** : les données personnelles (médicales, bancaires, communications privées) doivent être gardées confidentielles pour protéger le droit des individus à la vie privée
 - **Protection contre le vol d'identité** : la confidentialité aide à empêcher l'accès non-autorisé au données personnelles, pour réduire les risques de vol d'identité et de fraude
 - **Sécurité personnelle** : elle peut être compromise si des informations sensibles (adresse personnelle, emploi du temps) tombent entre de mauvaises mains

1.1. Communications sécurisées

- Confidentialité des données transmises : importance
 - Contexte professionnel 
 - **Avantage compétitif** : stratégies d'entreprises et propriétés intellectuelles sont critiques pour les compagnies. Les dévoiler pourrait causer des pertes ou permettre leur exploitation par les concurrents
 - **Confiance des clients** : les entreprises qui traitent des données clients doivent maintenir leur confidentialité. Toute fuite serait dommageable pour la réputation et pourrait avoir des répercussions légales
 - **Obligation légale** : pour certaines industries
 - [https://fr.wikipedia.org/wiki/Règlement général sur la protection des données](https://fr.wikipedia.org/wiki/R%C3%A8glement_g%C3%A9n%C3%A9ral_sur_la_protection_des_donn%C3%A9es)

1.1. Communications sécurisées

- Confidentialité des données transmises : importance
 - Contexte gouvernemental 
 - **Sécurité nationale** : la confidentialité des opérations gouvernementales, stratégies de défense et informations classées est vitale pour la sécurité nationale
 - **Relations diplomatiques** : une communication confidentielle entre les gouvernements garantie des négociations et une diplomatie sans interférence
 - **Enquêtes policières et judiciaires** : garder certaines informations confidentielles facilite les enquêtes menées et protège l'identité des informateurs et témoins

1.1. Communications sécurisées

- Risques associés aux communications
 - Interception 
 - **Écoute clandestine** : des individus ou entités non-autorisés peuvent intercepter les communications (e-mails, appels téléphoniques, messages) pour récolter des informations sensibles
 - **Fuite de données** : des pirates peuvent infiltrer des systèmes ou réseaux pour voler des données, en accédant à des informations confidentielles enregistrées
 - Falsification 
 - **Altération de données** : des agents malveillants peuvent modifier l'information durant sa transmission, générant des information fausses ou fallacieuses
 - **Intégrité des données** : elle peut être attaquée, afin d'affecter le processus de prise de décision

1.1. Communications sécurisées

- Risques associés aux communications
 - Imitation 
 - **Spoofing** : falsifier l'identité d'un expéditeur ou d'un destinataire, pour accéder de manière non-autorisée à des informations ou un système
 - **Hameçonnage (phishing)** : faux e-mails ou faux sites web piégeant les individus en les poussant à divulguer des informations sensibles, comme leur identifiants et mots de passes
 - Déni de service 
 - **Attaques DDoS** (attaque par déni de service distribuée) : inonder un système ou un réseau avec un trafic excessif pour rendre des services disponibles aux utilisateurs légitimes
 - **Coupure de service** : toute action interrompant la disponibilité normale d'un service, empêchant l'accès aux informations et systèmes critiques

1.1. Communications sécurisées

- Risques associés aux communications
 - Menaces internes 
 - Menaces de cybersécurité qui proviennent d'**utilisateurs autorisés**, c'est-à-dire d'employés, de fournisseurs ou de partenaires commerciaux, qui abusent, intentionnellement ou non, de leur accès légitime, ou dont les comptes sont détournés par des cybercriminels
 - Menaces physiques 
 - **Le vol ou la perte de matériel** (portables, clés USB) contenant des informations sensibles peut conduire à des accès non-autorisés
 - **Surveillance physique** : l'observation direct des canaux de communication physique, contournant ainsi les mesures de sécurité numérique

- **Définition**

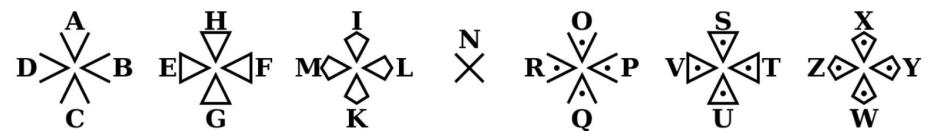
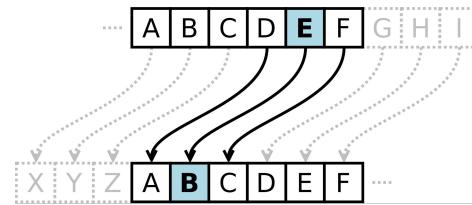
- Discipline de la cryptologie s'attachant à protéger les communications et les données en les convertissant dans un format illisible, sauf pour ceux possédant la clef de déchiffrement
- Cela implique donc des techniques pour écrire l'information de telle manière que seules les parties autorisées peuvent y accéder et la comprendre
- Notes :
 - La **cryptologie** englobe
 - Cryptographie, l'écriture secrète
 - Elle-même se subdivise en chiffrement et encodage (voir ci-après)
 - Cryptanalyse, l'analyse de cette dernière
 - La cryptographie se distingue de la **stéganographie** qui fait passer inaperçu un message dans un autre message

(p. ex. message caché dans une image, avec OutGuess)

1.2. Bases de la cryptographie

- Rôles de la cryptographie, dont trois propriétés des données qu'elle vise à garantir
 - **Confidentialité** : encoder les données sorte que, même si elles sont interceptées, elles seront inintelligibles pour les utilisateurs non-autorisés
 - **Intégrité** : maintenir l'intégrité des données en détectant toute altération non-autorisée (voir fonctions de hachage et *checksums*)
 - **Authentification** : les signatures numériques et protocoles d'authentification utilisent des techniques cryptographiques pour garantir que les messages échangés proviennent de sources légitimes
 - **Non-réputation** : les signatures numériques permettent de s'assurer qu'un contrat ne peut être remis en cause par l'une des parties
 - **Transactions sécurisées** : e-commerce, banques en ligne
 - **Protection de la vie privée** : protéger les données et communications personnelles des accès non-autorisés

- Antiquité
 - **Chiffrement par substitution monoalphabétique**
 - Substituer dans un message chacune des lettres de l'alphabet par une autre
 - Chiffrement par décalage (César, ROT13), chiffre des Templiers



- **Chiffrement par transposition (ou permutation)**
 - Changer l'ordre des lettres (anagrammes)
 - Scytale



1.3. Évolution des techniques

- Renaissance
 - **Chiffrement par substitution polyalphabétique** : pour une lettre donnée, son chiffrement ne sera pas toujours le même selon l'endroit du texte
 - **Chiffre d'Alberti** (1404 - 1472)
 - Les substitutions sont définies par le positionnement relatif de deux disques concentriques
 - Ce positionnement peut changer selon l'endroit du texte
 - « dans mon message, j'écrirai un D majuscule et, à partir de ce point, k ne signifiera plus B mais D »



- Renaissance

- **Chiffrement par substitution polyalphabétique**

- **Chiffre de Trithémius** (1462 - 1516)

- On chiffre la première lettre du message clair avec la première ligne, la deuxième lettre avec la deuxième ligne, etc.
 - Cela revient à une suite de décalages de César
 - La première lettre n'est pas décalée, la deuxième est décalée d'un cran dans l'alphabet, la troisième de deux crans, etc.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	
C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	
F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	
G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X

1.3. Évolution des techniques

- Renaissance

- Chiffrement par substitution polyalphabétique

- Chiffre de Vigenère (1523 - 1596)

- Introduit la notion de clé

Lettre en clair	
Lettre de la clé	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
A	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
B	B C D E F G H I J K L M N O P Q R S T U V W X Y Z A
C	C D E F G H I J K L M N O P Q R S T U V W X Y Z A B
D	D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
E	E F G H I J K L M N O P Q R S T U V W X Y Z A B C D
F	F G H I J K L M N O P Q R S T U V W X Y Z A B C D E
G	G H I J K L M N O P Q R S T U V W X Y Z A B C D E F
H	H I J K L M N O P Q R S T U V W X Y Z A B C D E F G
I	I J K L M N O P Q R S T U V W X Y Z A B C D E F G H
J	J K L M N O P Q R S T U V W X Y Z A B C D E F G H I
K	K L M N O P Q R S T U V W X Y Z A B C D E F G H I J
L	L M N O P Q R S T U V W X Y Z A B C D E F G H I J K
M	M N O P Q R S T U V W X Y Z A B C D E F G H I J K L
N	N O P Q R S T U V W X Y Z A B C D E F G H I J K L M
O	O P Q R S T U V W X Y Z A B C D E F G H I J K L M N
P	P Q R S T U V W X Y Z A B C D E F G H I J K L M N O
Q	Q R S T U V W X Y Z A B C D E F G H I J K L M N O P
R	R S T U V W X Y Z A B C D E F G H I J K L M N O P Q
S	S T U V W X Y Z A B C D E F G H I J K L M N O P Q R
T	T U V W X Y Z A B C D E F G H I J K L M N O P Q R S
U	U V W X Y Z A B C D E F G H I J K L M N O P Q R S T
V	V W X Y Z A B C D E F G H I J K L M N O P Q R S T U
W	W X Y Z A B C D E F G H I J K L M N O P Q R S T U V
X	X Y Z A B C D E F G H I J K L M N O P Q R S T U V W
Y	Y Z A B C D E F G H I J K L M N O P Q R S T U V W X
Z	Z A B C D E F G H I J K L M N O P Q R S T U V W X Y

Texte en clair : j'adore écouter la radio toute la journée
Clé répétée : M USIQU EMUSIQU EM USIQU EMUSI QU EMUSIQU
^ ^ ^
| ||Colonne 0, ligne I : on obtient la lettre W.
| |Colonne D, ligne S : on obtient la lettre V.
| Colonne A, ligne U : on obtient la lettre U.
Colonne J, ligne M : on obtient la lettre V.

Texte chiffré : V'UVWHY IOIMBUL PM LSLYI XAOLM BU NAOJVUY
Clé répétée : M USIQU EMUSIQU EM USIQU EMUSI QU EMUSIQU
^ ^ ^
| ||Ligne I, on cherche W : on trouve la colonne 0.
| |Ligne S, on cherche V : on trouve la colonne D.
| Ligne U, on cherche U : on trouve la colonne A.
Ligne M, on cherche V : on trouve la colonne J.

Arithmétique modulaire

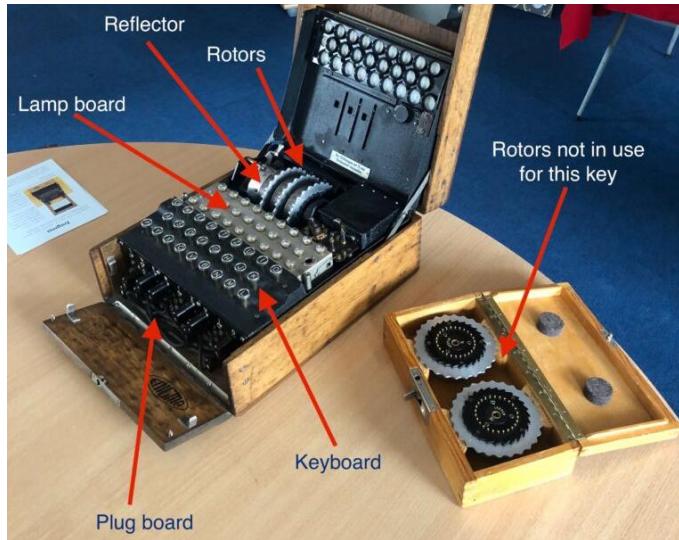
$$\text{Chiffré}[i] = (\text{Texte}[i] + \text{Clés}[i]) \bmod 26$$

$$\text{Texte}[i] = (\text{Chiffré}[i] - \text{Clés}[i]) \bmod 26$$

1.3. Évolution des techniques

- Seconde guerre mondiale
 - Enigma

- Famille de machines de chiffrement électromécaniques
- Version commerciale brevetée par Arthur Scherbius en 1918
- Plus tard, version militaire utilisée par l'armée allemande



Lorsqu'une touche du clavier est pressée :

- L'ampoule correspondant à la lettre chiffrée s'allume
- Le système de rotors tourne d'un cran
 - Presser la même lettre deux fois consécutives donnera deux lettres chiffrées différentes (chiffrement polyalphanumérique)

1.3. Évolution des techniques

- Seconde guerre mondiale
 - Enigma : design et combinaisons
 - 3 rotors crantés, sur 3 emplacements (droite, milieu, gauche) à choisir parmi 5 rotors différents
 - $5 \times 4 \times 3 = 60$
 - Chaque rotors possède 26 crans
 - $26 \times 26 \times 26 = 17\,576$
 - Tableau de connexions avec 10 câbles : permet de définir des permutations pour 10 paires de lettres (version militaire seulement)
 - $$\frac{26!}{6! \times 10! \times 2^{10}} = 150\,738\,274\,937\,250$$
 - Total = 158 962 555 217 826 360 000 configurations possibles

1.3. Évolution des techniques

- Seconde guerre mondiale
 - Enigma : **fonction de chiffrement involutive** (*self-reciprocal cipher*)
 - Les processus de chiffrement et déchiffrement sont les mêmes
 - On déchiffre en tapant le message chiffré sur une machine Enigma avec la même configuration que celle ayant servi au chiffrement
 - Propriété garantie par l'utilisation d'un **réflecteur** à la suite du dernier rotor
 - Facilite l'utilisation
 - Mais empêche de substituer une lettre à elle-même dans le texte chiffré
→ Vulnérabilité à une « attaque à texte clair connu »
 - Conception de la « bombe » par Alan Turing pour trouver la configuration d'Enigma utilisée, c.-à-d. trouver la clé
 - Note : les Alliés finir par capturer des exemplaires de ces machines...

1.3. Évolution des techniques

- Cryptologie moderne
 - **1940s - Avènement de l'informatique** : (classique → moderne) les ordinateurs révolutionnent la cryptographie, permettant des calculs plus rapides et des algorithmes plus complexes
 - **1970s - Développements mathématiques** : concepts comme la théorie des nombres, les structures algébriques et la théorie de la complexité
 - **1976 - Cryptographie asymétrique** : introduction généralement attribuée à Whitfield Diffie et à Martin Hellman
 - **1977 - Algorithme RSA** : développement par Rivest, Shamir et Adleman, aujourd'hui fondamental pour le chiffrement et les signatures numériques
 - **1980s - Fonctions de hachage cryptographiques** : leur émergence offre des moyens de vérifier l'authenticité et l'intégrité des données
 - **1990s - Cryptanalyse différentielle et linéaire** : développement de techniques capables de casser des algorithmes de chiffrement symétriques
 - **2001 - Advanced Encryption Standard (AES)** : adoption, en remplacement de Data Encryption Standard (DES), du fait de sa robustesse et de son efficience

- Informatique quantique et cryptographie post-quantique 

 - **Défis posés** : le développement des ordinateurs quantiques constitue une menace pour les méthodes de chiffrement actuelles, car ils pourraient résoudre, de manière efficiente, les problèmes mathématiques sur lesquels ces algorithmes reposent
 - RSA : décomposition en facteurs premiers
 - (EC)DH : problème du logarithme discret
 - **Solutions explorées** : développement d'algorithmes à résistance quantique basés sur des problèmes d'optimisation sur les réseaux euclidiens
 - Problème des vecteurs courts (SVP)
 - Problème de l'apprentissage avec erreur (LWE)

- Blockchain et crypto-monnaies 
 - **Défis posés** : résoudre les problèmes d'extensibilité (*scalability*), d'anonymat et de sécurité, sans compromettre la décentralisation
 - **Solutions explorées** : nouveaux mécanismes de consensus et techniques comme la ***zero-knowledge proof***
- Chiffrement homomorphe 
 - Algorithme permettant d'effectuer certaines opérations mathématiques sur des données chiffrées sans avoir à les déchiffrer d'abord
 - Permet de confier des calculs à un agent externe, sans que les données ni les résultats soient accessibles à cet agent
 - **Défis posés** : améliorer l'efficience
 - **Solutions explorées** : concernent principalement le *cloud computing*

- **Chiffrement** (*encryption*) : processus de conversion d'un message en clair, m (données intelligibles) en un message chiffré c (données inintelligibles), en utilisant un algorithme E et une clé k
 - $E(k_e, m) = c$
 - Déterministe ou probabiliste
- **Déchiffrement** (*decryption*) : processus réciproque du chiffrement, transformant c en m en utilisant une clé appropriée
 - $D(k_d, c) = m$
 - ! Toujours déterministe
- **(E, D)** : cryptosystème
- **Primitive cryptographique** : opération fondamentale cryptographique, de bas niveau et bien établi (*building blocks*), sur la base de laquelle est bâti tout système de sécurité informatique, algorithme ou protocole
 - Exemples : Chiffrement de flux, par bloc, fonction de hachage, MAC, PRNG, RSA, DH

- **Cryptanalyse** : ensemble des techniques d'analyse des systèmes cryptographiques, ayant pour but de trouver des vulnérabilités et d'être capable de déchiffrer sans connaître la clé. Le processus par lequel on tente de comprendre un message en particulier est appelé une **attaque**.
- **Adversaire** : entité cherchant à casser la sécurité d'un système cryptographique
 - Cryptanalyste : adversaire théorique (individu ou algorithme)
 - Attaquant : quelqu'un essayant activement d'exploiter des failles du système cryptographique
- **Alice et Bob** : pour les « personne A » et « personne B » qui cherchent à communiquer
 - D'autres noms sont utilisés, comme Eve (*eavesdropper*), Mallory (*malicious attacker*), Oscar (*opponent*) ou Trudy (*intruder*)

- **Chiffre (cipher) vs code** : le chiffrement d'un message s'opère au niveau des caractères de celui-ci, caractères individuels ou bien petits groupes de caractères.
Un code, lui, encrypte un message par remplacement des mots ou des phrases. Il opère donc au niveau de la signification du message.
 - Un message peut être encodé avant chiffrement pour compliquer la cryptanalyse
 - **Nécessité d'un livre-code** ou d'une table de correspondance
 - Exemples :
« Les sanglots longs des violons de l'automne... » = « Sabotez les voies ferroviaires ! »

6706	reichlich
13850	finanziell
12224	unterstützung
6929	und
14991	einverständnis

Extrait du décodage du télégramme diplomatique Zimmermann (1917)

- **Cryptographie symétrique** (ou à clé secrète) : la même clé est utilisée pour le chiffrement et le déchiffrement
 - Problématique de la distribution des clés, qui doit être sécurisée
 - Chiffrement par flot (ou de flux, ou continu) : opère sur un flux continu de données (*i.e.* pas de découpage). Exemples : A5/1, RC4
 - Chiffrement par bloc : opère sur des données découpées en blocs de taille généralement fixe. Exemples : DES, AES
 - Code d'authentification de message (MAC) : code accompagnant des données dans le but d'assurer l'intégrité de ces dernières

- **Cryptographie asymétrique** (ou à clé publique) : système cryptographique utilisant des paires de clés, une publique pour le chiffrement, une privée pour le déchiffrement
 - Chiffrement asymétrique (RSA, DH, ...)
 - Schéma de signature numérique : mécanisme permettant d'authentifier l'auteur d'un document électronique et d'en garantir la non-répudiation
- **Cryptographie hybride** : combine
 - un chiffrement asymétrique (plus sécurisé) pour l'échange d'une clé symétrique,
 - et un chiffrement symétrique (plus rapide) pour les données transmises.
 - souvent couplées avec un mécanisme d'authentification
(p. ex. TLS, *Transport Layer Security*)



2. Chiffrements symétriques

- **Chiffre de Vernam** (1917) : XOR entre une clé secrète k et le message clair m pour produire le chiffré c

$$m \oplus k = c$$

Chiffrement

$$\begin{array}{r} 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ m \\ \oplus \\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ k \\ \hline =\ 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ c \end{array}$$

Déchiffrement

$$\begin{array}{r} 1\ 1\ 0\ 1\ 0\ 1\ 1\ 0\ c \\ \oplus \\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ k \\ \hline =\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ m \end{array}$$

On peut également retrouver k à partir de m et c : $m \oplus c = k$

2.1. Masque jetable

- **Vulnérabilité** : utiliser deux fois la même clé k pour deux messages clairs m_1 et m_2 permet, à quelqu'un qui aurait intercepté m_2 , c_1 et c_2 , de déchiffrer m_1 sans la clé k

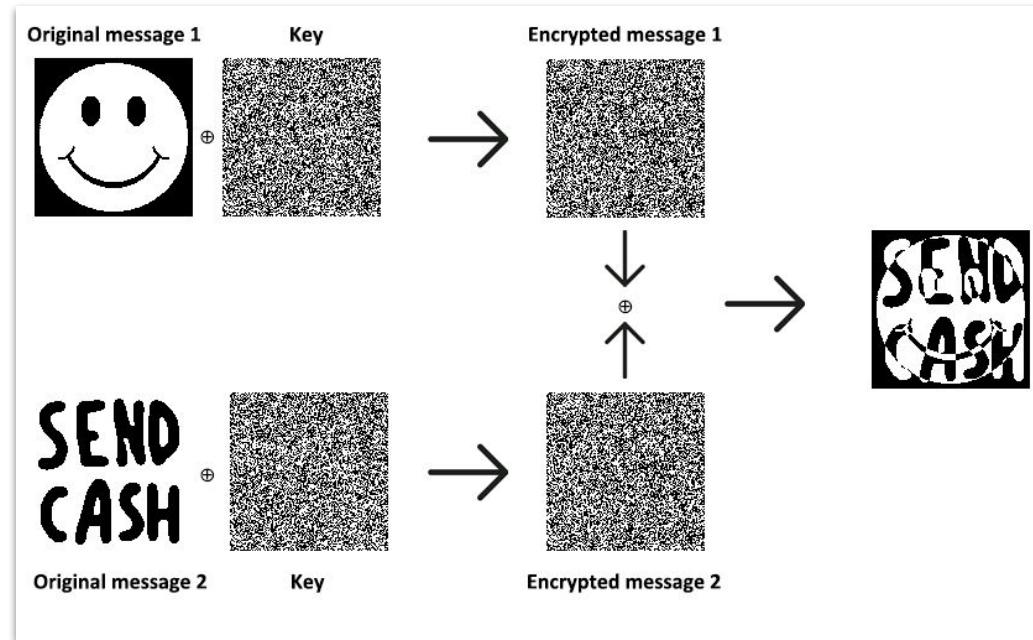
$$m_1 \oplus k = c_1$$

$$m_2 \oplus k = c_2$$

$$m_1 \oplus c_1 = m_2 \oplus c_2$$

$$m_1 = (m_2 \oplus c_2) \oplus c_1$$

Ci-contre : par commutativité, on a aussi $m_1 \oplus m_2 = c_1 \oplus c_2$
Cela crée une vulnérabilité en interceptant uniquement c_1 et c_2 (analyse fréquentielle, analyse par *crib*)



- La sécurité du chiffrement repose donc sur une **utilisation unique de la clé** (ou masque) → Masque jetable ou ***one-time pad (OTP)***
- La fonction **XOR** nécessite que la clé soit de la **même longueur que le message**
Très peu pratique (→ chiffréments de flux), surtout si la clé doit être gardée secrète

- Si la clé secrète k (à usage unique et de même longueur que m) **est parfaitement aléatoire**, alors OTP permet un « secret parfait » (*perfect secrecy*), concept introduit par Claude Shannon (1949)
OTP est le seul algorithme offrant une sécurité parfaite

Aucune information* sur le message clair m ou la clé k ne peut être extraite** du chiffré c , même en disposant d'une puissance de calcul illimitée
→ OTP résiste aux attaques par force brute avec un ordinateur quantique

- Clé parfaitement aléatoire : à chaque position de la séquence, les valeurs 0 et 1 sont équiprobables
→ Distribution uniforme (discrète) : $k \sim U(0^l, 1^l)$, où l est la longueur de la clé

*Information telle que définie par Claude Shannon

**La longueur de m est cependant connue et égale à celle de c

- Exemple pour $k \in K = \{0, 1\}^3$

$$\begin{array}{rclclclcl} \oplus & 111 & c & \oplus & 111 & c & \oplus & 111 & c \\ \oplus & 000 & k_1 & = & 001 & k_2 & = & 011 & k_3 \\ = & 111 & m_1 & = & 110 & m_2 & = & 100 & m_3 \\ & & & & & & & & \\ \oplus & 111 & c & \oplus & 111 & c & \oplus & 111 & c \\ \oplus & 110 & k_5 & = & 100 & k_6 & = & 101 & k_7 \\ = & 001 & m_5 & = & 011 & m_6 & = & 010 & m_7 \end{array}$$

$$\begin{array}{rclclclcl} \oplus & 111 & c & \oplus & 111 & c & \oplus & 111 & c \\ \oplus & 111 & k_4 & = & 000 & k_4 & = & 000 & m_4 \\ = & 000 & m_4 & & & & & & \\ & & & & & & & & \\ \oplus & 111 & c & \oplus & 111 & c & \oplus & 111 & c \\ \oplus & 010 & k_8 & = & 101 & k_8 & = & 101 & m_8 \\ = & 101 & m_8 & & & & & & \end{array}$$

Avec une clé **k inconnue de l'adversaire** et générée de manière parfaitement aléatoire, le message chiffré c « 111 » pourrait résulter du chiffrement de tous les messages clairs possibles (m_1 à m_8) de façon équiprobable (1 chance sur 8).

Propriété : un **XOR** entre n'importe quelle séquence et une séquence aléatoire produit une séquence également aléatoire.

2.1. Masque jetable

$$\Pr[E(k, m_0) = c] = \Pr[k \oplus m_0 = c] \quad (1)$$

$$= \frac{|k \in \{0, 1\}^m : k \oplus m_0 = c|}{\{0, 1\}^m} \quad (2)$$

$$= \frac{1}{2^m} \quad (3)$$

$$\Pr[E(k, m_1) = c] = \Pr[k \oplus m_1 = c] \quad (4)$$

$$= \frac{|k \in \{0, 1\}^m : k \oplus m_1 = c|}{\{0, 1\}^m} \quad (5)$$

$$= \frac{1}{2^m} \quad (6)$$

donc

$$\boxed{\Pr[E(k, m_0) = c] = \Pr[E(k, m_1) = c] \quad \forall m_0, m_1 \in M, \text{ avec } |m_0| = |m_1| \\ \forall c \in C}$$

avec $M = K = \{0, 1\}^m$

- **Convention d'écriture**
 - M : espace des messages en clair
 - $M = \{0, 1\}^l$
 - $|M| = 2^l$
 - C : espace des messages chiffrés
 - K : espace des clés
- La sécurité parfaite d'OTP requiert que $|K| \geq |M|$

- **Contenu en information** (ou auto-information)

- Défini par Claude Shannon, 1948

Soit une variable discrète X qui prend une valeur x

(p. ex. la valeur donnée à une position sur une séquence numérique)

Le contenu en information $I_X(x)$ de cet évènement est :

$$I_X(x) := -\log [p_X(x)] = \log \left(\frac{1}{p_X(x)} \right)$$

avec $p_X(x) \neq 0$ et $I_X(x) \in [0, +\infty]$

En utilisant un logarithme de base 2, l'unité de $I_X(x)$ est le shannon (Sh) ou bit ;
le hartley (Hart) pour la base 10 ; le *natural unit of information* (nat) pour la base e

- **Entropie de Shannon**

- Elle correspond au contenu en information moyen   sur toutes les positions de la s  quence. Elle s'  crit $H(X)$.

$$\begin{aligned} H(X) &= \sum_x -p_X(x) \log p_X(x) \\ &= \sum_x p_X(x) I_X(x) \\ &\stackrel{\text{def}}{=} E[I_X(X)], \end{aligned}$$

avec $H(X) \in [0, +\infty]$

- Pour une s  quence de longueur donn  e, **l'entropie de Shannon est maximum** si les diff  rentes valeurs possibles    chaque position sont equiprobables, c.-  -d. si la s  quence est parfaitement al  atoire (loi uniforme ; **maximum d'incertitude**)
Note : maximum mais pas ∞ , car s  quence de longueur finie

- **Entropie de Shannon**

- Exemple : entropie de la séquence de valeurs décimales « 31100427 »

$$p(0) = 2/8 = 0,25$$

$$p(1) = 2/8 = 0,25$$

$$p(2) = 1/8 = 0,125$$

$$p(3) = 1/8 = 0,125$$

$$p(4) = 1/8 = 0,125$$

$$p(5) = 0/8 = 0$$

$$p(6) = 0/8 = 0$$

$$p(7) = 1/8 = 0,125$$

$$p(8) = 0/8 = 0$$

$$p(9) = 0/8 = 0$$

$$\begin{aligned} H(X) &= \\ -0,25 \log_2(0,25) &+ \\ -0,25 \log_2(0,25) &+ \\ -0,125 \log_2(0,125) &+ \\ -0,125 \log_2(0,125) &+ \\ -0,125 \log_2(0,125) &+ \\ -0,125 \log_2(0,125) &+ \\ = 2,125 \text{ bits} \end{aligned}$$

• Entropie conditionnelle

- Pour deux séquences X et Y :

$$H(Y|X) = - \sum_{x \in \mathcal{X}, y \in \mathcal{Y}} p(x, y) \log \frac{p(x, y)}{p(x)}$$

- $H(Y|X) \leq H(Y)$

$H(Y|X) = H(Y) \Leftrightarrow Y$ et X indépendantes

- Pour un message M et son chiffré C , on définit ainsi :

- Sécurité parfaite : $H(M|C) = H(M)$

- Connaître C n'apporte aucune information sur M (sauf sa longueur)
- La probabilité qu'un adversaire* casse le chiffrement est nulle

- Sécurité inconditionnelle : $H(M|C) \leq H(M)$

- La probabilité qu'un adversaire* casse le chiffrement est négligeable

* disposant d'une puissance de calcul illimitée

- OTP requiert une clé parfaitement aléatoire
 - Peut être obtenue grâce à un générateur de nombres aléatoires physique

True random number generator (TRNG)

Basé sur la variation aléatoire de propriétés physiques : p. ex. *timestamp*



Le mur de lampes à lave de Cloudflare

- **Limitation** : produisent un nombre limité de bits aléatoires par seconde

- **Générateur de nombres pseudo-aléatoires**

Pseudorandom number generator, PRNG

Algorithme qui génère une séquence de nombres qui apparaît aléatoire, mais qui est déterminée par une valeur initiale appelée « graine » (seed)

⚠ Sortie **déterministe** → **même graine = même sortie**

- Exemple : **générateur congruentiel linéaire**

$$X_{n+1} = (a \cdot X_n + c) \bmod m,$$

où X_n est le nombre pseudo-aléatoire courant,
et a, c et m sont des constantes, $m > 0$

Sans nécessairement être aléatoire, la graine doit être choisie de sorte à ne pas être prédite

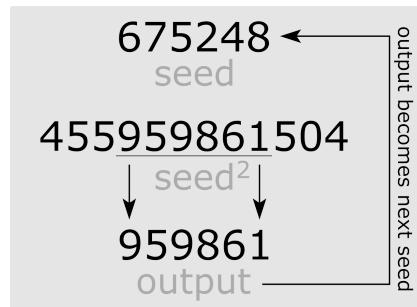
La valeur de X_0 est celle de la graine, à partir de laquelle la formule se répète de façon récurrente, pour générer une séquence de nombres.

Limitation : un mauvais choix des constantes peut générer une suite périodique

P. ex., pour le générateur d'UNIX, $a = 1103515245$, $c = 12345$ et $m = 2^{31}$

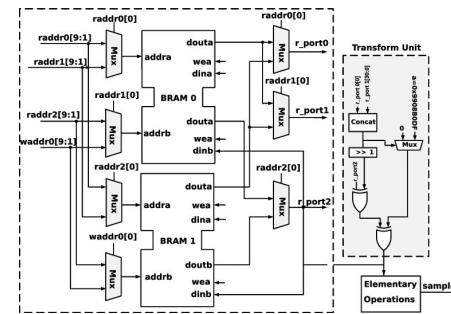
- **Générateur de nombres pseudo-aléatoires**

- 💀 Une périodicité dans la clé pseudo-aléatoire compromet la sécurité, de la même manière qu'une réutilisation de la même clef
- Autres exemples :



Méthode du carré médian

Pour une graine de $2n$ chiffres, la période maximum est de 10^n (un peu plus, si nombre impair de chiffres).



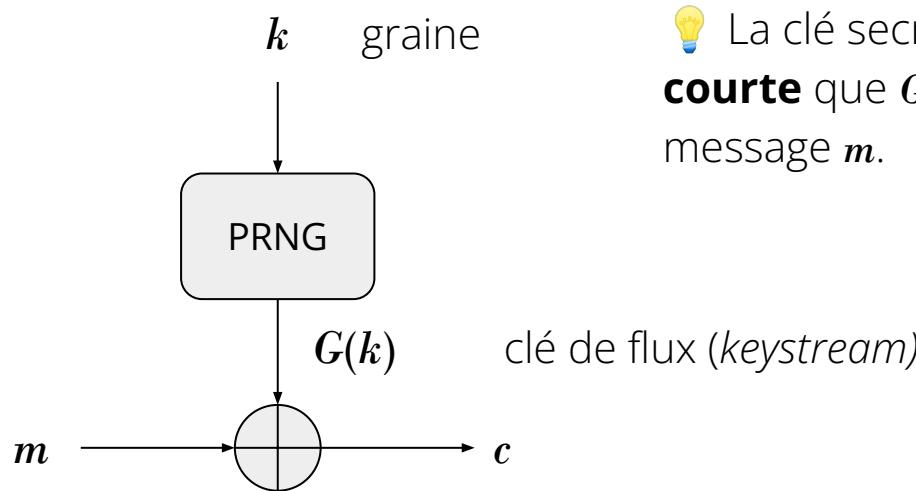
Mersenne Twister (MT19937)

Sa période maximum est de $2^{19937} - 1$. Mais insuffisant pour la cryptographie, car il existe des algorithmes qui permettent d'en prédire le comportement.



`import random`

- Architecture générale



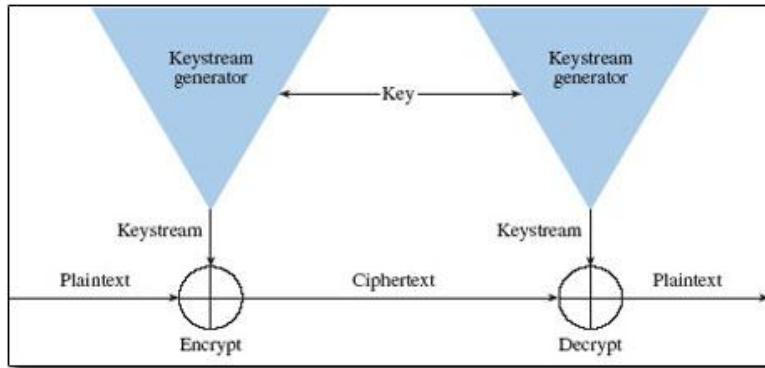
💡 La clé secrète échangée est k . Elle est **beaucoup plus courte** que $G(k)$ qui est de la même longueur que le message m .

$$E(k, m) = m \oplus G(k)$$

$$D(k, c) = c \oplus G(k)$$

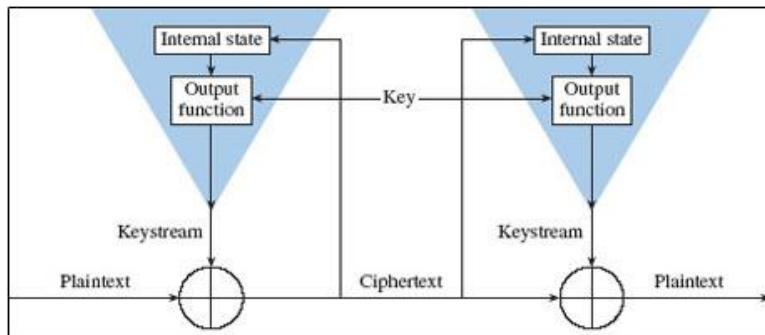
On peut donc considérer un chiffrement continu comme un **XOR** combiné à un PRNG. Ce dernier doit impérativement être **sécurisé pour la cryptographie**.

- Deux catégories



Synchrone

Calcule le flux de clé en fonction de la clé secrète k , mais indépendamment du texte clair m et du chiffré c .



Asynchrone (ou *self-synchronizing*)

Calcule chaque bit du flux de clé comme une fonction de k et des n précédents bits de c . Avantage : le récepteur se synchronise avec le générateur après réception de n bits de c , ce qui facilite la correction d'erreurs. Désavantage : 1 bit d'erreur à la transmission entraînera n bits d'erreurs pour le récepteur (propagation d'erreur).

- **Générateur de nombres pseudo-aléatoires sécurisé**

Cryptographically secure PRNG (CSPRNG)

- **RC4 (Rivest Cipher 4)**, 1987

- Utilisé depuis les premières versions des protocoles SSL/TLS, WEP, WPA
 - Génération d'un **flux d'octets** pseudo-aléatoires par modifications itératives (permutations) de l'état interne (vecteur de 256 octets), dépendantes de la clé secrète
 -  Beaucoup de vulnérabilités, dont corrélations entre graine et flux

- **Salsa20** et **ChaCha**, 2008

- Protocoles TLS, IPsec, SSH et OpenVPN

- **Trivium**, A5/2 (2G), E0 (Bluetooth)

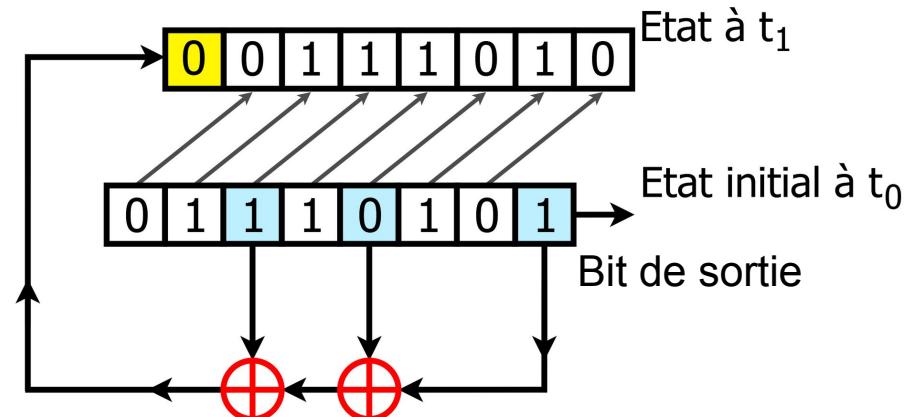
- Basés sur des **registres à décalage à rétroaction linéaire (LFSR)**...

- **Registre à décalage à rétroaction linéaire**

Linear feedback shift register (LFSR)

- Registre à décalage de type SIPO (*Serial In - Parallel Out*) dans lequel un ou plusieurs « étages » du registre subissent une transformation pour être réinjectés en entrée de celui-ci

- Il est dit de longueur r lorsqu'il est composé de r éléments, appelés « étages » ou « cellules », le contenu de l'ensemble de ces éléments à un instant t est **l'état du LFSR** à ce moment.



- À chaque top d'horloge, le contenu d'un étage est transféré au suivant et le premier est rempli (**bit de rétroaction**) par le résultat d'une **fonction linéaire** qui prend en compte **l'état d'un ou de plusieurs étages**.

- **Registre à décalage à rétroaction linéaire**

Linear feedback shift register (LFSR)

- Seulement utilisé comme **élément** de chiffrement de flux, car rétro-ingénierie à partir de la sortie
 - Combinaisons non-linéaires de LFSRs (p. ex. dans le chiffrement **Grain**)
- Convient peu à une implémentation logicielle (beaucoup de boucles)
→ Très efficace en **implémentation matérielle**
- *maximum length sequence* (MLS) : suite périodique qui explore toutes les valeurs pouvant être produites par un LFSR.
S'il comporte r bascules (*flip-flops*), $2^r - 1$ valeurs sont parcourues, l'état « tout à zéro » étant exclu, car il ne génère jamais de changement

- **Preuve de sécurité**

Démonstration qu'un schéma cryptographique respecte des propriétés de sécurités spécifiques, comme la confidentialité, l'intégrité et l'authenticité.

Il existe différents types de preuves cryptographiques, dont :

- Théorie de l'information
- Argument hybride 
- Preuve par jeu
- Réduction polynomiale
- *Zero knowledge proof*

• Indistinguabilité calculatoire

- Deux distributions de probabilités sont calculatoirement indistinguables, s'il n'existe pas d'algorithme **efficace** qui puisse les discerner de manière **significative**.
 - A : algorithme probabiliste terminant en temps polynomial (proportionnel à la longueur de la séquence en entrée x)
 - Aussi appelé distinguiseur (*distinguisher*) ou test statistique
 - $A(x) = 1$: la sortie (booléen) de A vaut 1 (ou **Vrai**), s'il détecte que l'entrée x a été échantillonnée à partir d'**une distribution plutôt que l'autre**
 - Deux distributions D_n et E_n sont calculatoirement indistinguables si l'**avantage** $\delta(n)$ de l'adversaire est négligeable en fonction du « paramètre de sécurité » n :

$$\delta(n) = \left| \Pr_{x \leftarrow D_n} [A(x) = 1] - \Pr_{x \leftarrow E_n} [A(x) = 1] \right|$$

- Si la distribution est générée par un PRNG, n sera la longueur de la graine ; Dans RSA, n sera la valeur du module

• Indistinguabilité calculatoire

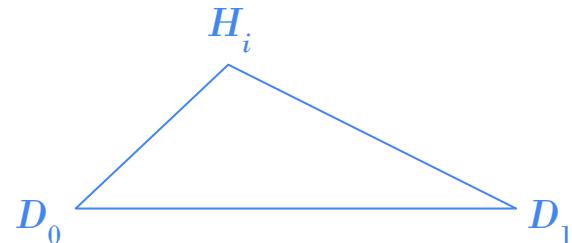
-  La question posée par le calcul de l'avantage est : est-ce que A se comporte différemment selon qu'on lui donne une entrée échantillonnée à partir de D_n ou bien de E_n ? Si oui, cette différence est-elle négligeable (notation : $D_n \approx E_n$), c.-à-d. est que $\delta(n) \rightarrow 0$ quand $n \rightarrow \infty$?
- **Fonction négligeable** : une fonction $f(n)$ est considérée comme négligeable si, \forall polynôme positif $\text{poly}(n)$, \exists une valeur suffisamment grande de n tq $f(n) < \frac{1}{\text{poly}(n)}$ c.-à-d. si $f(n) \rightarrow 0$ plus vite que l'inverse de n'importe quel polynôme
- Indistinguabilité calculatoire est donc une relaxation de l'indistinguabilité statistique prenant en compte le fait que la puissance de calcul des algorithmes est limitée (notions de complexité)

- **Argument hybride**

Pour montrer l'indistinguabilité calculatoire de deux distributions D_0 et D_1 , on définit une séquence de distributions hybrides tq $D_0 := H_0, H_1, \dots, H_n =: D_1$ qui permet de transformer graduellement la distribution D_0 en la distribution D_1 . L'adversaire \mathcal{A} ne pourra lire que les n premières hybrides (entier au plus polynomial).

Si $H_i \approx H_{i+1}, \forall 0 \leq i < n$, alors $D_0 \approx D_1$

[Note](#) : conséquence de l'inégalité triangulaire



Dans un preuve de sécurité d'un chiffrement par flot, D_0 serait la sortie du PRNG et D_1 une séquence binaire parfaitement aléatoire.

- **Argument hybride**

La manière de définir les hybrides dépendra du PRNG dont on souhaite prouver la sécurité cryptographique.

À titre d'exemple, pour un générateur et sa graine $G(k)$:

- $H_0 : G(k)$
- $H_1 : G(k) \parallel G(G(k))$
- $H_2 : G(k) \parallel G(G(k)) \parallel G(G(G(k)))$
- ...
- $H_n : G(k) \parallel G(G(k)) \parallel \dots \parallel G^{(n)}(k)$
- $H_{n+1} : U \parallel U \parallel \dots \parallel U$

avec U représentant une séquence parfaitement aléatoire (uniforme)

- **Exemple** : soit $G : \{0, 1\}^n \rightarrow \{0, 1\}^l$ un PRNG sécurisé (avec $l = f(n)$, fonction polynomiale) ; soit un hybride $G'(k) = G(k) \| G(k)$ et soit la distribution uniforme $U(0^{2l}, 1^{2l})$
Question : la sortie de $G'(k)$ est-elle calculatoirement indistinguabile de U ?

Réponse : On peut définir un distingueur A , qui détecte que la séquence $x \leftarrow G'(k)$ (et renvoie donc Vrai), en deux étapes :

1. Coupe x en 2 segments de même longueur tq $x_1 \| x_2 = x$
2. $A(x) = \text{Vrai}$ si $x_1 = x_2$

Calcul de l'avantage de A :

$$\delta(n) = |P_{x \leftarrow G'(k)}[A(x) = \text{Vrai}] - P_{x \leftarrow U}[A(x) = \text{Vrai}]|$$

Par la définition même de A , on a : $P_{x \leftarrow G'(k)}[A(x) = \text{Vrai}] = 1$

$$P_{x \leftarrow U}[A(x) = \text{Vrai}] = ?$$

- **Intuition** avec les plus petites valeurs possibles : 2 bits

Un générateur aléatoire peut générer $x \in \{00, 01, 10, 11\}$

Mais $G'(k)$ ne peut générer que $x \in \{00, 11\}$

$$\mathbf{P}_{x \leftarrow U} [A(x) = \text{Vrai}] = 2/4 = \frac{1}{2}$$

$$\delta(n) = | \mathbf{P}_{x \leftarrow G'(k)} [A(x) = \text{Vrai}] - \mathbf{P}_{x \leftarrow U} [A(x) = \text{Vrai}] | = 1 - \frac{1}{2} = \frac{1}{2}$$

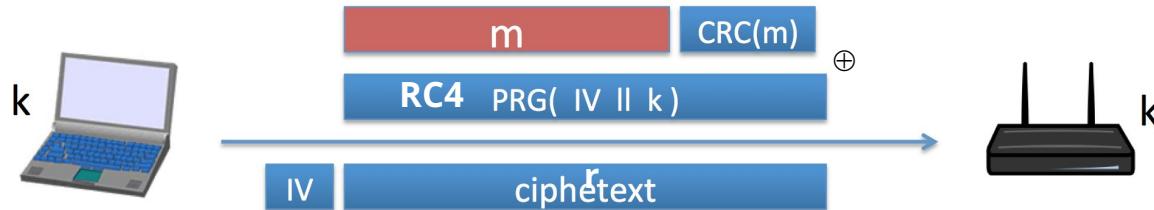
- Pour toute longueur l de x

$$\mathbf{P}_{x \leftarrow U} [A(x) = \text{Vrai}] = 2^l / 2^{2l} = 2^{-l}$$

$$\delta(n) = | 1 - 2^{-l} | = | 1 - 2^{-f(n)} | > \frac{1}{\text{poly}(n)} \quad \text{non-négligeable ; } G'(k) \text{ n'est pas sécurisé}$$

- **Limitations**

- **802.11b WEP:**



La même clé k était réutilisée ; seul le IV changeait à chaque message, mais était trop court (24 bits) → Utiliser WPA2 ou WPA3 pour sécurisé les réseaux WiFi

- **Chiffrement de disque :**

- Des erreurs dans le chiffré peuvent empêcher le déchiffrement
 - Beaucoup de données : la périodicité de la clé de flux ou sa réutilisation cause l'apparition de motifs répétés dans le chiffré, détectables en cryptanalyse
 - ↳ Utiliser un **chiffrement par bloc**

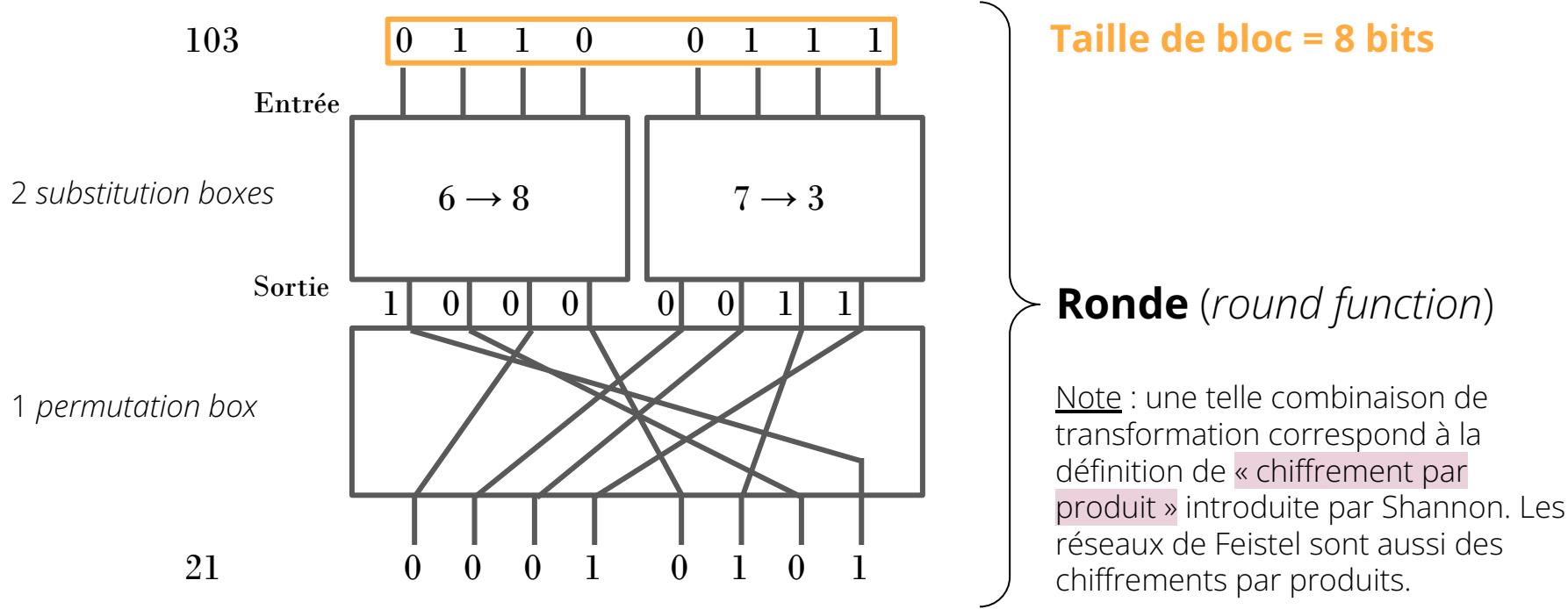
- **Chiffrement par bloc**

Un chiffrement par bloc découpe le message clair m en blocs d'une certaine taille (p. ex. 128 bits) et produit en sortie le message chiffré c , constitué d'autant de blocs chiffrés, chacun de cette même taille.

 Dans un chiffrement continu, le caractère $c[i]$ résulte du chiffrement du caractère $m[i]$. Cela équivaut à une **taille de bloc de 1**.

- **Réseau de substitution-permutation (SP)**

- Structure pour construire des chiffrements par bloc (Shannon, 1949)



- Réseau de substitution-permutation (SP)

- **S-Box (substitution box)**

Comme leur nom l'indique, elles sont utilisées pour effectuer des opérations de substitutions au sein de chaque bloc durant le processus de chiffrement.

Elles sont implémentées comme des tables de correspondances (*Lookup Table*, LUT) dans lesquelles sont définies les règles de substitutions.

Ces tables peuvent être fixes ou bien générées dynamiquement à partir de la clé.

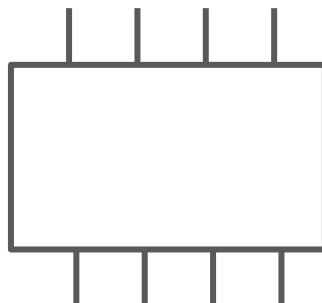


Illustration d'une S-Box : 4 bits en entrée et sortie.

La table de substitution comporte donc $2^4 = 16$ règles, pour des valeurs décimales de 0 à 15 : $0 \rightarrow 6 ; 1 \rightarrow 13 ; 2 \rightarrow 1 ; \dots$

La taille des S-Boxes AES est de 8 bits.

Dans DES, elle est de **6 bits en entrée, 4 bits en sortie**.

Note : les substitutions réciproques doivent être évitées.

- Réseau de substitution-permutation (SP)

- **S-Box (*substitution box*)**

Exemple d'une S-Box 3×3 non-linéaire :

Entrée	000	001	010	011	100	101	110	111
Sortie	011	101	111	100	000	010	001	110

Table de correspondance souvent simplifiée sur 2 dimensions :

	00	01	10	11
0	011	101	111	100
1	000	010	001	110

- Réseau de substitution-permutation (SP)

- S-Box (*substitution box*)

Exemple d'une S-Box 3×3 linéaire :

	00	01	10	11
0	000	011	111	100
1	110	101	001	010

Équations linéaires :

$$\begin{aligned}
 c_1 &= x_1 \oplus x_2 & = 1 \cdot x_1 \oplus 1 \cdot x_2 \oplus 0 \cdot x_3 \\
 c_2 &= x_1 \oplus x_2 \oplus x_3 & = 1 \cdot x_1 \oplus 1 \cdot x_2 \oplus 1 \cdot x_3 \\
 c_3 &= x_2 \oplus x_3 & = 0 \cdot x_1 \oplus 1 \cdot x_2 \oplus 1 \cdot x_3
 \end{aligned}$$

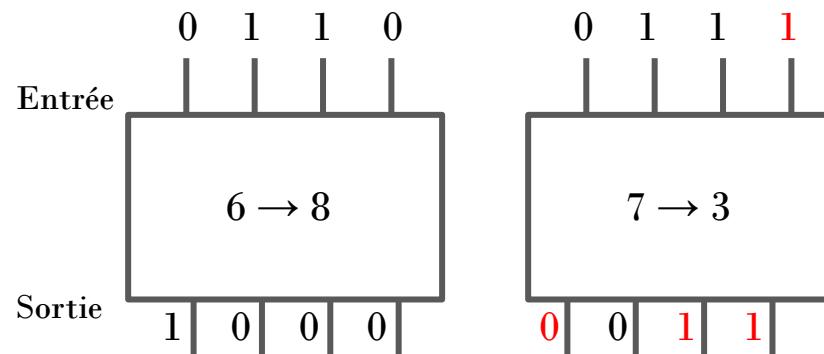
- Réseau de substitution-permutation (SP)

- **S-Box (substitution box)**

En pratique, les S-boxes sont des fonctions booléennes vectorielles **non-linéaires**.

Cette **non-linéarité** des S-boxes introduit ce que Shannon définit comme de la « **confusion** » dans la relation entre le chiffré et la clé secrète : pour un attaquant essayant de trouver la clé, le calcul réciproque à partir du chiffré est rendu difficile.

Une S-box est également source de « **diffusion** » : si l'on change un seul bit du texte clair (entrée), alors une grande partie du texte chiffré (sortie) change.



- **Réseau de substitution-permutation (SP)**

 Pour une bonne diffusion, il faudrait une grande S-box (p. ex. 128 bits). Mais cela serait beaucoup trop coûteux à implémenter. Ainsi, les S-box utilisées sont courtes et n'apportent quasiment pas de diffusion, que de la confusion.

- **P-Box (permutation box)**

Utilisées pour permuter (réarranger) les bits des données d'entrée, de manière systématique, selon un motif prédéfini. Mathématiquement, les opérations des P-boxes sont **linéaires**.

C'est la combinaison d'une P-Box **avec plusieurs S-Boxes courtes** qui introduit de la diffusion. Cette diffusion aide à ce que la structure statistique du texte clair (les *patterns*) ne soit pas préservée dans le chiffré.

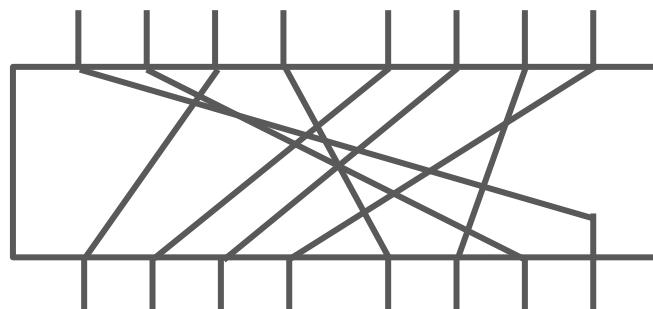
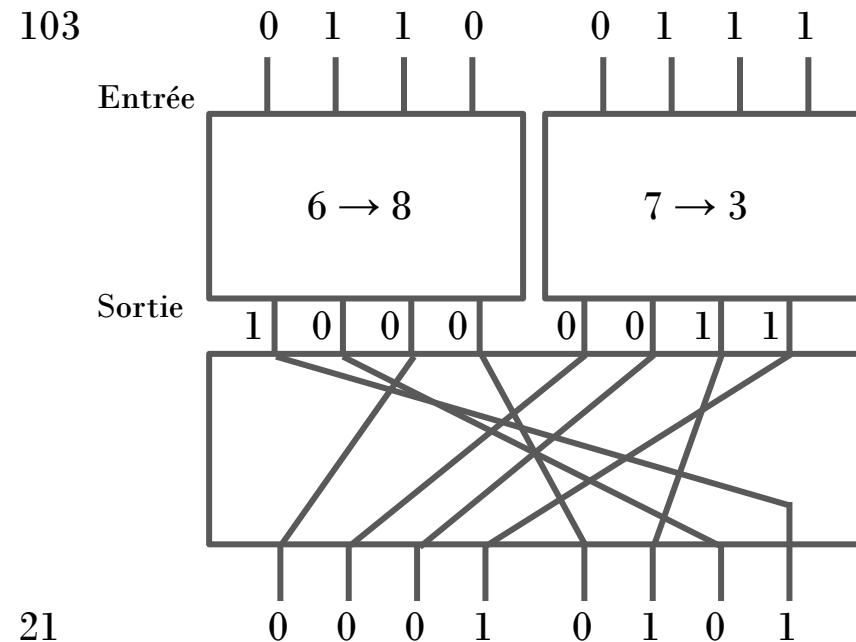
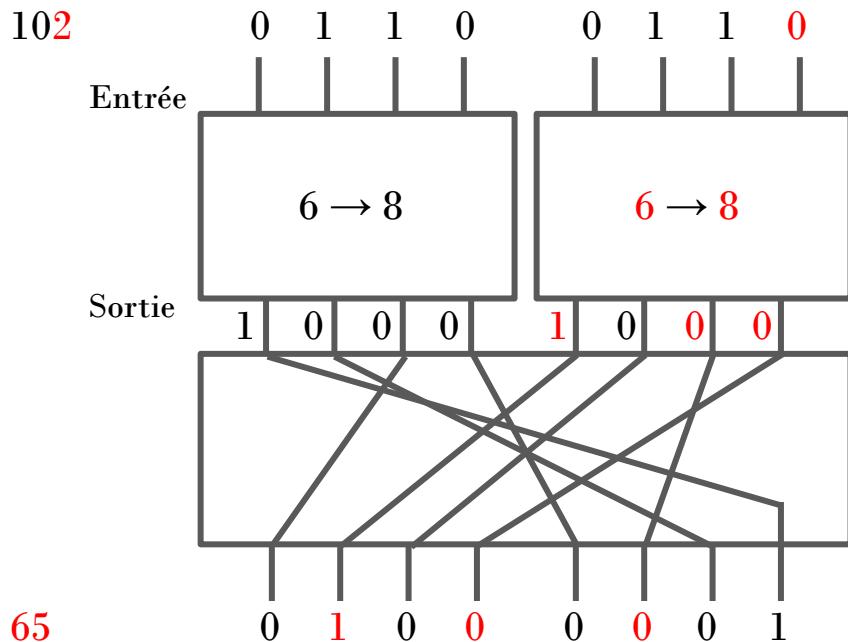


Illustration d'une P-Box droite : 8 bits I/O

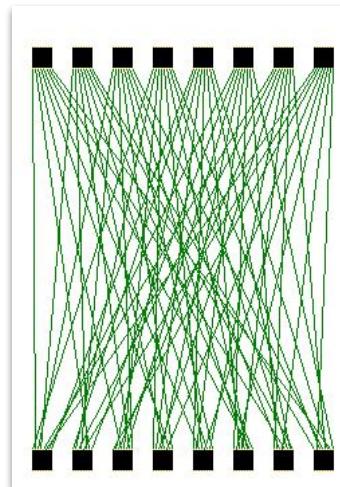
- **Diffusion**

- Exemple avec une seule ronde et sans ajout de clé



- **Remarque**

- La diffusion est également appelée « effet avalanche »
- Lorsque la diffusion est dite **complète** (propriété des fonctions booléennes)
 - On obtient un effet avalanche strict : la modification d'un seul bit en entrée affecte tous les bits en sortie avec une probabilité de 50%
 - Chaque bit en sortie est dépendant de tous les bits en entrée



- **Réseau de substitution-permutation (SP)**

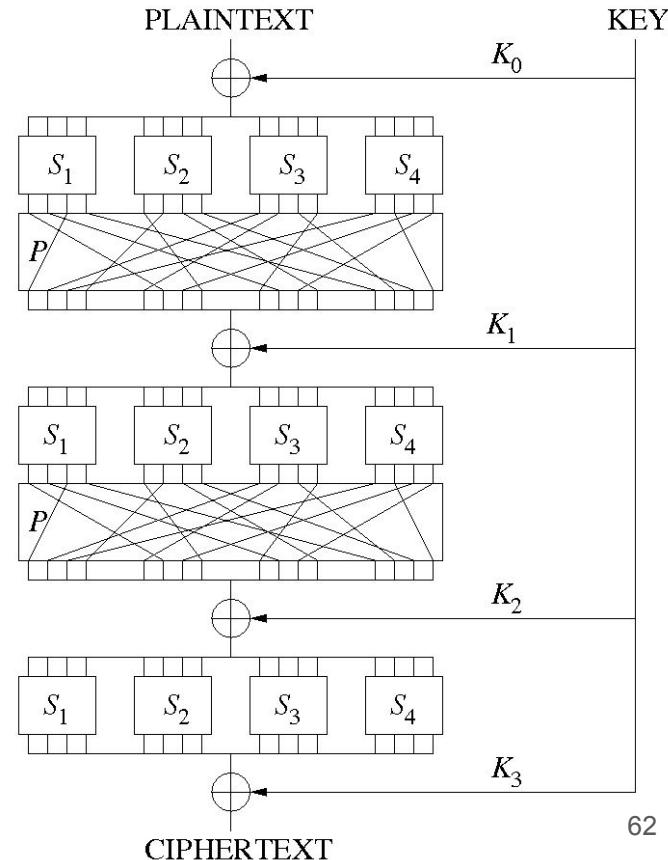
Plusieurs rondes doivent être effectués : leur nombre est un compromis en vitesse et sécurité

On introduit l'utilisation d'un clé secrète afin que, sans elle, **on ne puisse plus inverser la fonction** combinant S-Box et P-Box (« boîtes blanches »).

Préparation des clés (key schedule) : on crée des sous-clés à partir de la clé principale (« expansion » et division en clés de ronde).

Chaque ronde se termine donc par un **XOR** avec la sous-clé (*key mixing*). Ci-contre : 3 rondes.

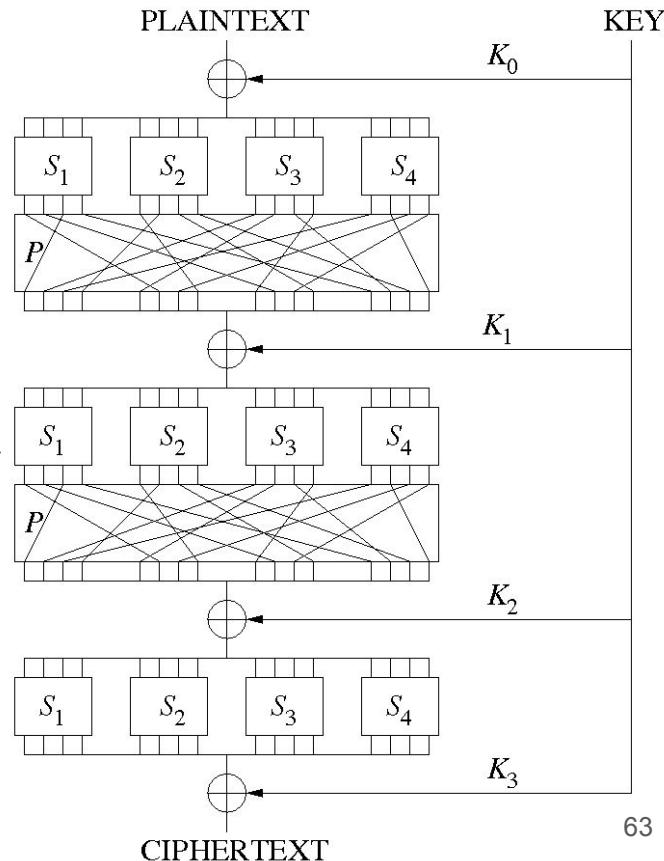
Déchiffrement : réciproques des S-boxes et P-boxes, avec utilisation des clés de rondes en ordre inversé.



- **Réseau de substitution-permutation (SP)**

Remarques :

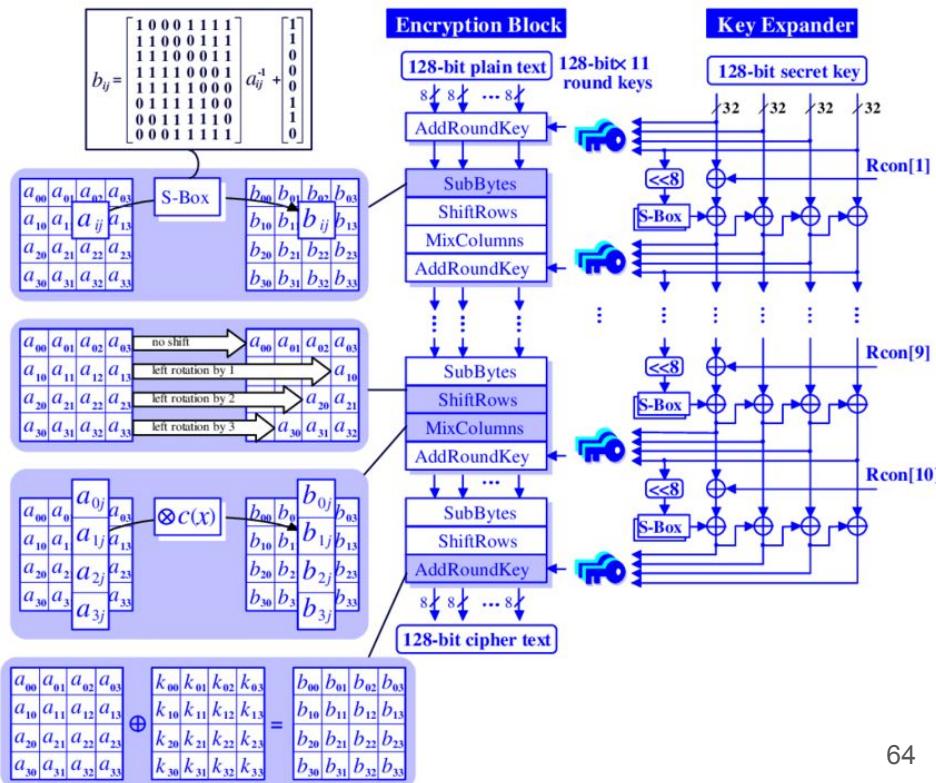
- La première ronde commence également par un **XOR** avec une sous-clé K_0
- Comme la dernière ronde produit le chiffré, elle n'a pas de P-Box : si celle-ci est publiée (c.-à-d. son schéma est rendu publique), alors un attaquant peut simplement « dé-permuter » les bits du chiffré. Elle n'a donc aucune valeur cryptographique. Le retrait de cette dernière P-box permet de diminuer les coûts de calculs et d'implémentation



Rijndael

Réseau SP de Rijmen et Daemen, 1998

- Bloc de données arrangé en grille
P. ex. $4 \times 4 = 16$ octets = 128 bits
- **Substitutions** (S-Box) :
Table de correspondance → rapide
 - ⚠ Pas de point fixe
 - Pas de point fixe opposé (p. ex. 1010 \Leftrightarrow 0101)
- **Permutations** :
 - Décalages des rangs
 - Mélange des colonnes
 - Multiplication par une matrice
→ Transformations linéaires
- **Addition de la clé de ronde**
- Dernière ronde : permutation inutile



- **Rijndael**

- Algorithme choisi par le NIST pour l'*Advanced Encryption Standard* (AES) pour remplacer (Triple) DES : devait être aussi sécurisé, mais beaucoup plus rapide
- L'AES fixe :
 - la longueur de bloc à 128 bits
 - la longueur des clés à 128, 192, ou 256 bits
 - et le nombre de rondes associé : 10, 12 et 14, respectivement
- Chaque étape peut être effectuée dans le **sens inverse** → **déchiffrement**
- <https://formaestudio.com/rijndaelinspector/archivos/Rijndael Animation v4 eng-html5.html>
- **Arithmétique modulaire**

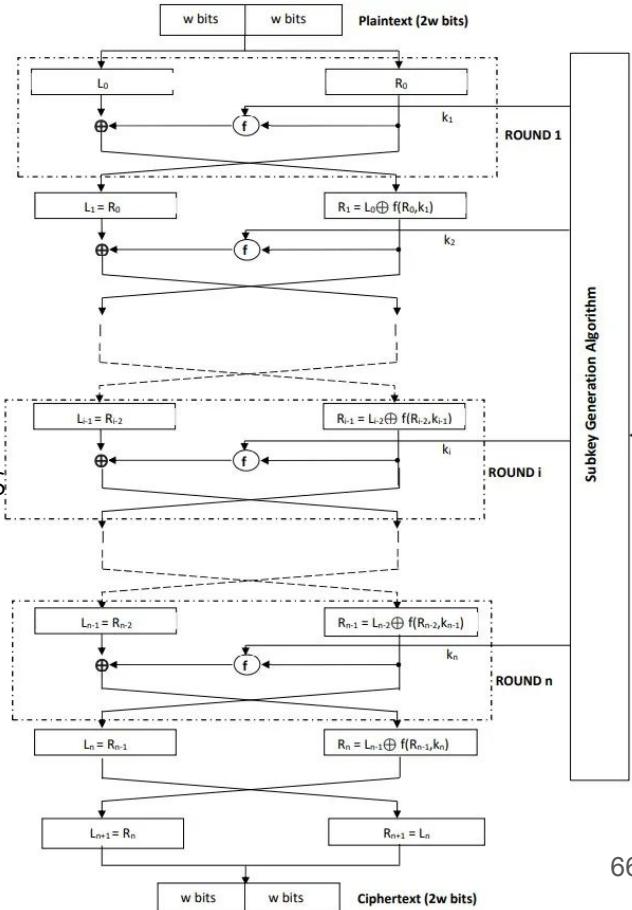
Toutes les opérations (addition, soustraction, multiplication, inversion) se font dans un « corps fini » (ou corps de Galois) de 2^8 éléments, soit 1 octet, les valeurs allant de 0000 0000 à 1111 1111. Les 256 éléments sont donc représentés par un polynôme irréductible de coefficients binaires $m(x)=x^8 + x^4 + x^3 + x + 1$ et les opérations se font modulo ce polynôme

2.6. Chiffrement par bloc

• Réseau de Feistel

Horst Feistel et Don Coppersmith, 1973

- Structure (comme réseaux SP) pour construire des chiffrements par bloc (p. ex. DES, Twofish)
- **Étapes du chiffrement :**
 - Le bloc en entrée est divisé en 2 parties (de tailles égales ou inégales)
 - Le chiffrement consiste en une série de rondes
 - La moitié droite du bloc est transformée par une fonction de ronde f , qui dépend d'une sous-clé dérivée de la clé principale
 - La sortie de f est ensuite XORée avec la moitié droite du bloc
 - **Après la dernière ronde** de (dé)chiffrement,
→ Permutation des deux moitiés



- Réseau de Feistel

- Le déchiffrement est essentiellement **identique au chiffrement**, la seule différence étant l'utilisation des sous-clés dans l'ordre inverse

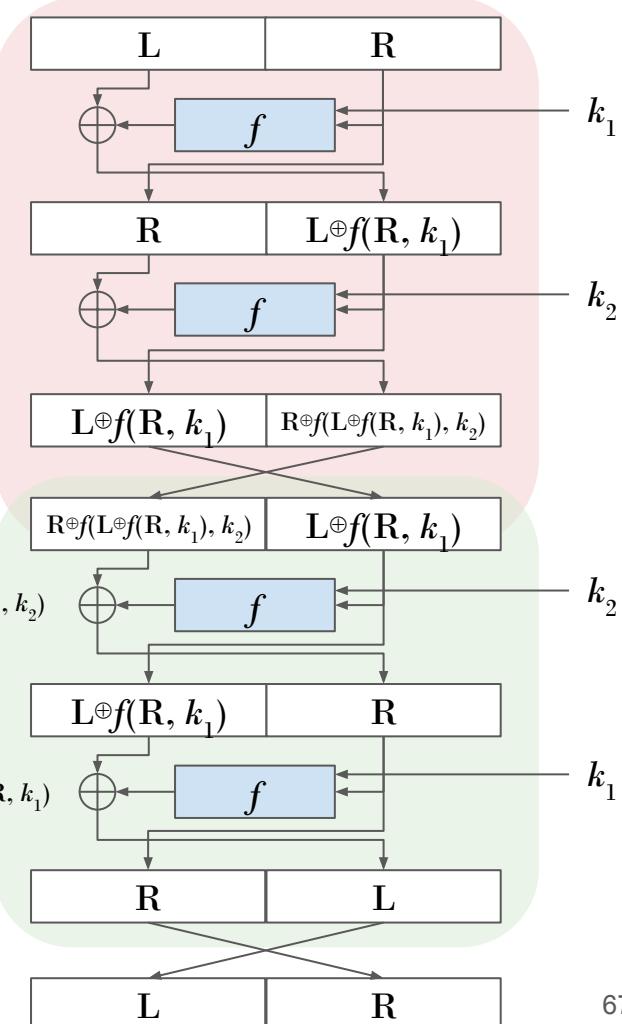
Ci-contre : exemple avec 2 rondes

- Fonctionne
 - quelque soit f
 - pour tout nombre de rondes

 $R \oplus f(L \oplus f(R, k_1), k_2)$
 $\oplus f(L \oplus f(R, k_1), k_2)$
 $L \oplus f(R, k_1) \oplus f(R, k_1)$
 $\oplus f(R, k_1)$

R	L
---	---

L	R
---	---



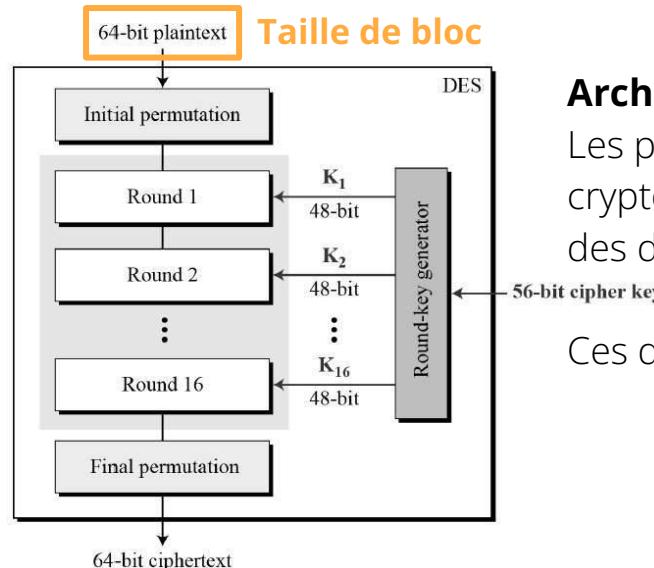
- Réseau de Feistel

- DES (*Data Encryption Standard*), 1975

Clé de 56 bits : peut être trouvée par force brute en 2^{55}

opérations **en moyenne** → Difficile à faire, jusque dans les années 90...

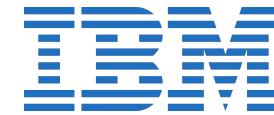
- Remplacé par Triple DES → Sécurisé, mais 3 fois plus lent !



Architecture générale du DES

Les permutations initiales et finales n'ont aucun intérêt cryptographique : elles ne sont là que pour le (dé)chargement des données sur le matériel 8 bits des années 70.

Ces deux fonctions sont mutuellement réciproques.

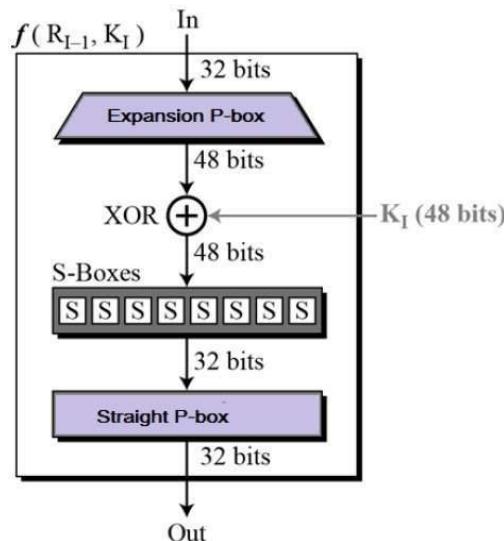


- Réseau de Feistel

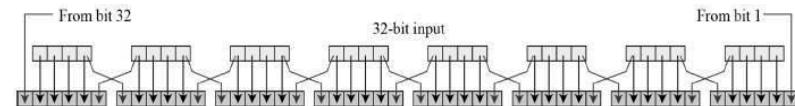
- DES (*Data Encryption Standard*), 1975

- Fonction de ronde f

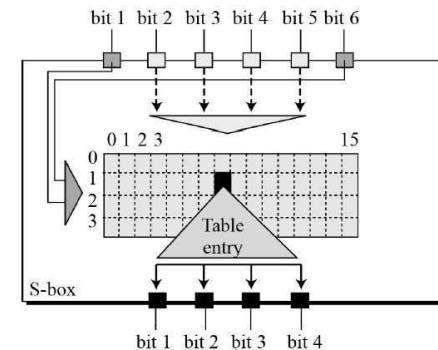
Combine une sous-clé de 48 bits avec les 32 bits de poids faibles (LSB) pour produire une sortie de 32 bits



Expansion P-Box (E-Box)



S-Box (6/4 bits)

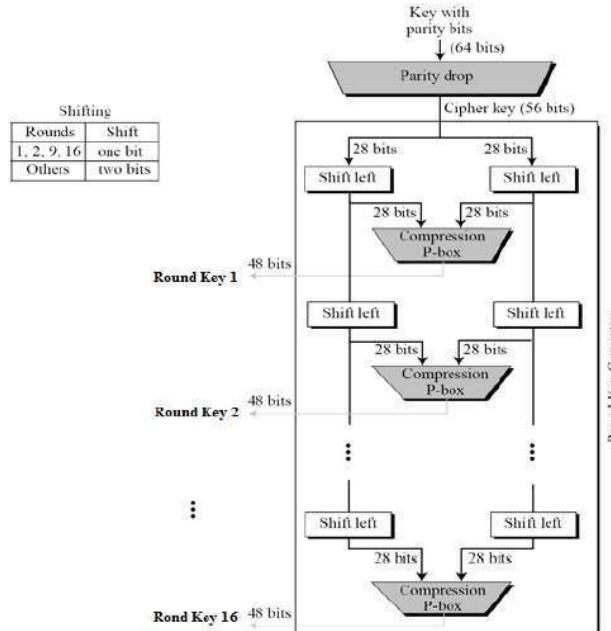


- Réseau de Feistel

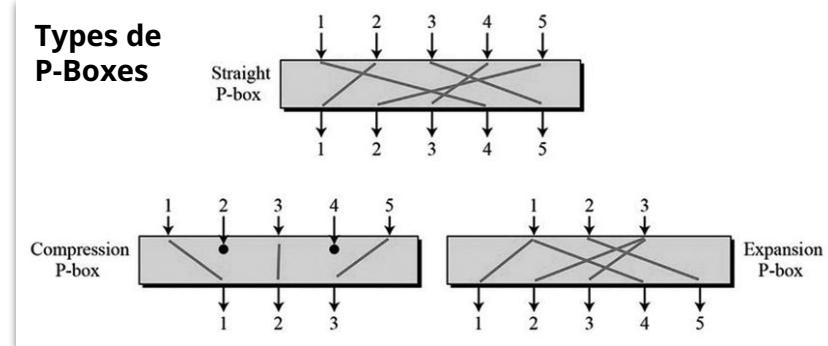
- DES (*Data Encryption Standard*), 1975

- **Key schedule** (préparation des clés de rondes)

Création de 16 clés de 48 bits, à partir de la clé principale de 56 bits



8 bits de parité, ajoutés aux 56 bits de données, pour assurer l'intégrité de la clé durant sa transmission (code correcteur)



- **Réseau de Feistel**

- DES-X (Rivest, 1984)

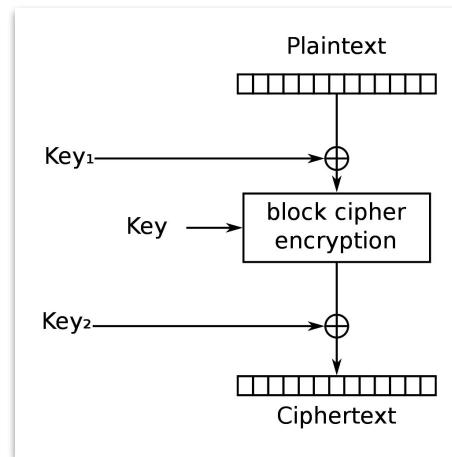
But : augmenter la complexité des attaques par force brute

 La force brute est, à ce jour, le seul type d'attaque connu contre DES

Utilisation d'une clé k de 184 bits, comprenant deux clés de 64 bits, k_1 et k_2 :

$$\text{DES-X}(m) = k_2 \oplus \text{DES}_k(m \oplus k_1)$$

Note : $184 = 56 + 2 \times 64$



Processus de *key whitening*

- **Mode de fonctionnement** (*mode of operation*)

Un mode d'opération décrit comment appliquer de manière répétée l'opération monobloc d'un chiffrement pour transformer en toute sécurité des quantités de données supérieures à un bloc

Afin d'utiliser les chiffrements par bloc dans diverses applications, **cinq** modes d'opération ont été définis par le NIST (SP 800-38A)

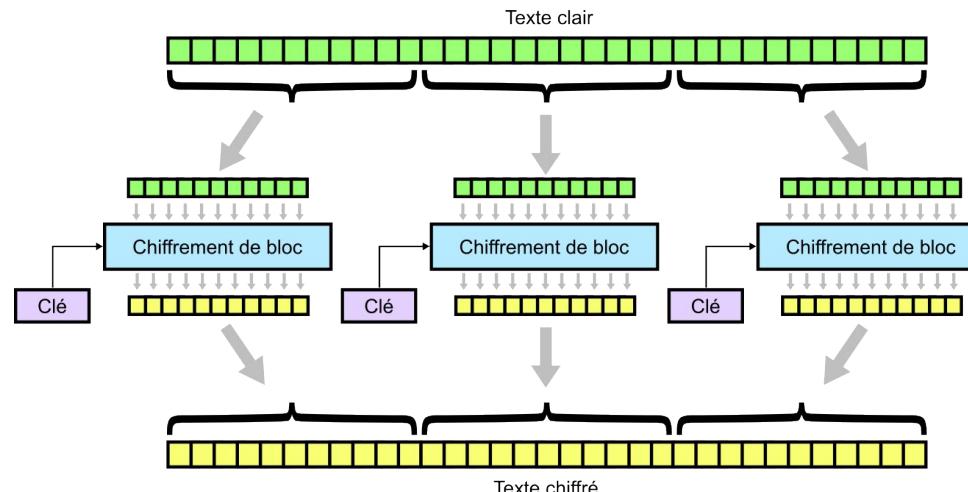
NIST SP 800-38A *Recommendation for Block Cipher Modes of Operation: Methods and Techniques* (déc. 2001)

- ECB
- CBC
- CTR
- CFB
- OFB

- **ECB** (*Electronic Codebook Block*, dictionnaire de codes)

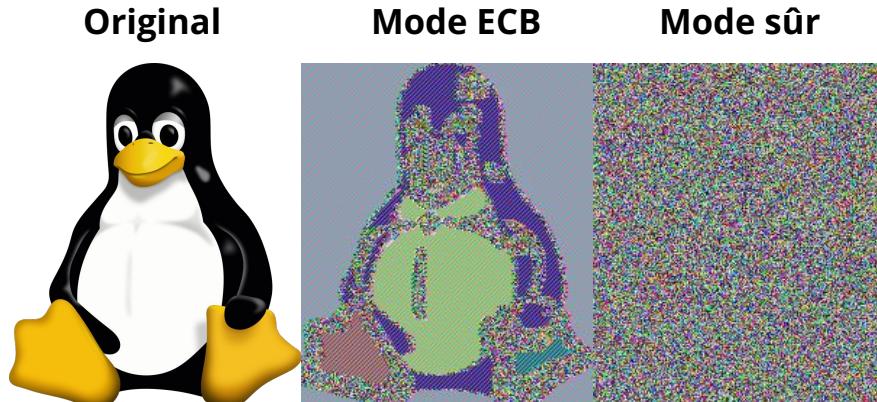
Approche la plus naïve

- Couper le message en blocs et les chiffrer un à la fois
- Si message trop court → *padding*, afin que sa longueur soit un multiple de la taille de bloc
- Déchiffrement : fonction réciproque



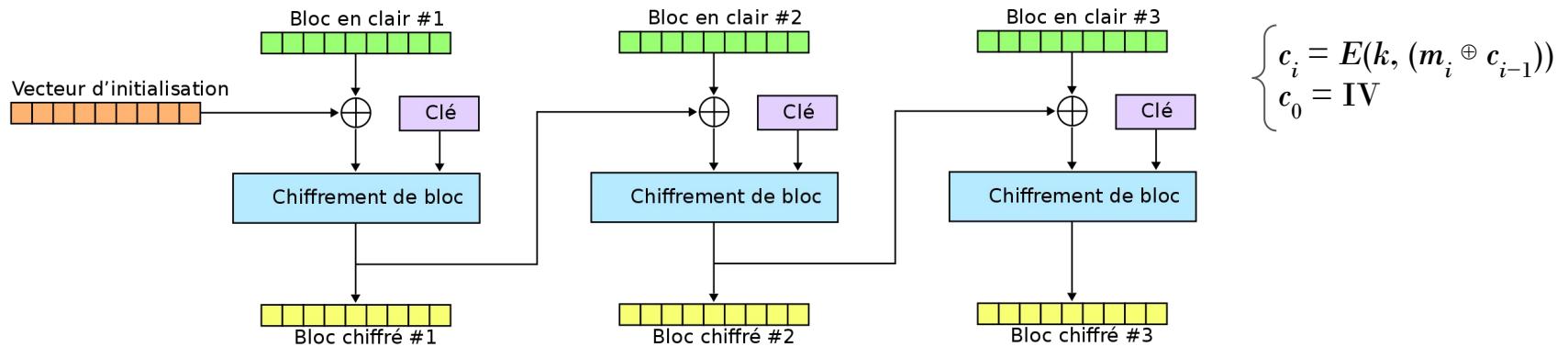
2.7. Modes de fonctionnement

- **ECB** (*Electronic Codebook Block*, dictionnaire de codes)
 - ✓ Facile à implémenter
 - ✓ Rapide et parallélisable
 - ✓ Permet un déchiffrement direct d'une zone quelconque du texte chiffré (*random read access*)
 - ⚠ Non-sécurisé : si deux blocs clairs sont identiques, les deux blocs chiffrés correspondants le seront aussi (même clé k)
- Exemples : requête HTTP répétée à l'identique,  [zoom](#)



Si des parties de l'image (données de pixels) sont identiques, leur chiffrements le sont aussi
→ Motifs (*patterns*) visibles

- **CBC** (*Cipher Block Chaining*, enchaînement de blocs)



La sortie d'un bloc est reliée à l'entrée du bloc suivant : permet de masquer le i -ème bloc du texte clair par le chiffrement du bloc $i-1$ du texte chiffré

→ **2 blocs clairs identiques donnent 2 blocs chiffrés différents**

Premier bloc : entrée du **XOR**ée avec un bloc aléatoire « imaginaire », appelé **vecteur d'initialisation (IV)**

L'IV n'a pas besoin d'être secret, mais doit être aléatoire (cas du CBC) et **unique** : un nouvel IV pour chaque message (stockage dans une mémoire permanente)

→ Si deux blocs #1 sont les mêmes, leurs chiffrés seront différents

2.7. Modes de fonctionnement

- **CBC** (*Cipher Block Chaining*, enchaînement de blocs)

Le mode CBC est désormais peu utilisé



Séquentiel

↳ Perte du parallélisme d'ECB : on ne peut plus **chiffrer** le bloc $i+1$ avant le bloc i



Padding nécessaire, car **XOR** requiert un bloc clair complet

↳ Vulnérabilité à *padding attack* (sauf si *ciphertext stealing*, CTS)



Enchaînement de blocs et **XOR** → propagation d'erreur

■ Altération du bloc chiffré i → Problème avec les blocs clairs i et $i+1$

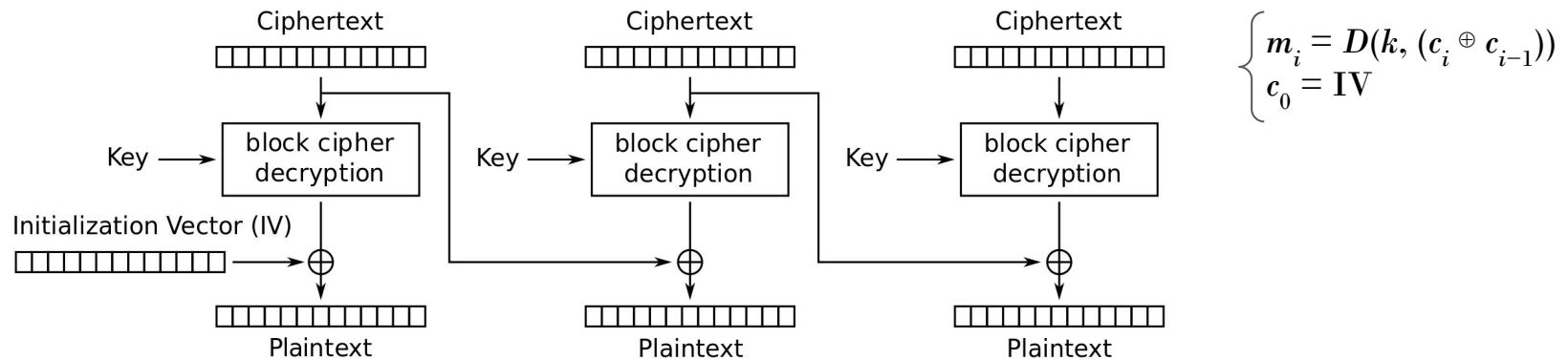
↳ Perte de 2 blocs !!

- Erreur de transmission réseau
- *bit flipping attack*

2.7. Modes de fonctionnement

- **CBC** (*Cipher Block Chaining*, enchaînement de blocs)

Le déchiffrement suit processus exactement inverse : le chiffré #1 est déchiffré, puis **XOR** avec **l'IV, qui est transmis avec le message**, pour obtenir le clair #1



💡 Le déchiffrement peut être parallélisé : déchiffer avec un IV incorrect entraînera une corruption du premier bloc de texte clair, mais les blocs clairs suivants seront corrects

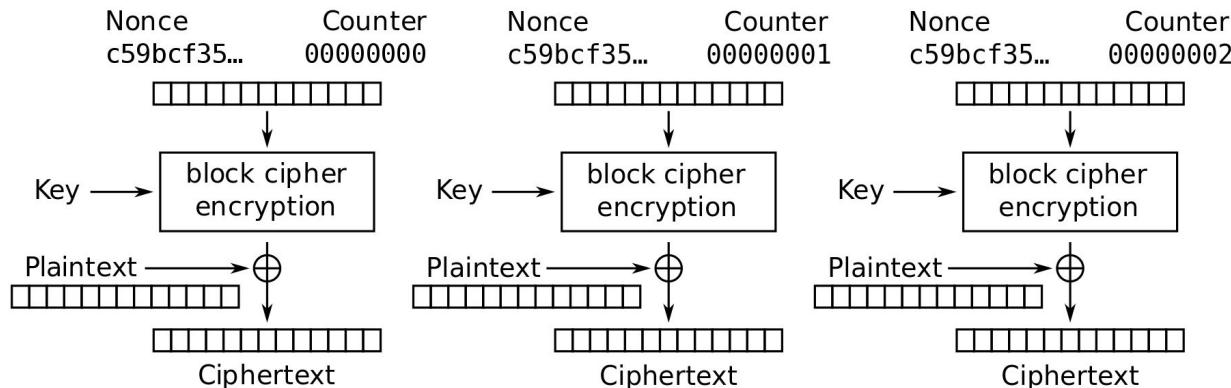
2.7. Modes de fonctionnement

- **CTR** (*CounTeR*), Diffie et Hellman, 1979

Nonce (*number used once*) : nombre arbitraire, **unique à chaque communication** et qui n'a pas besoin d'être secret, ni aléatoire

Il est **concaténé à un compteur** pour ensuite être chiffré par bloc à l'aide d'une clé

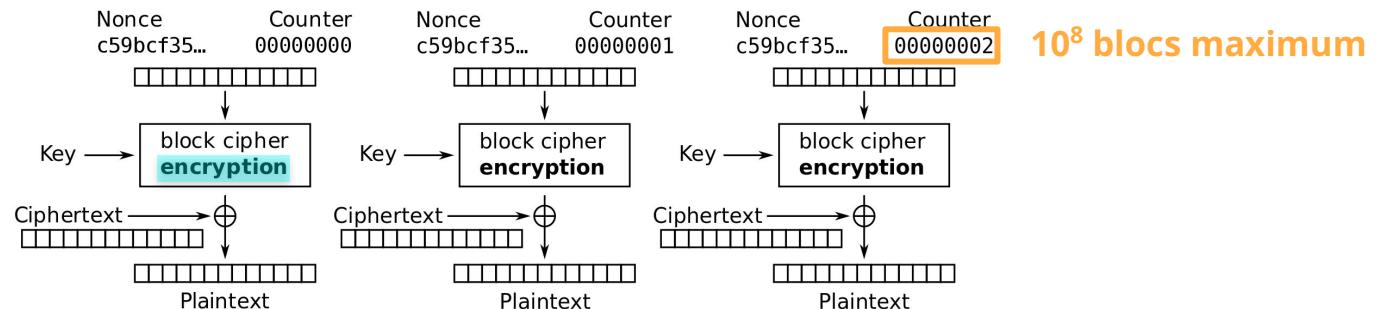
⚠ Terminologie : le *nonce*, concaténé ou non, est parfois appelé vecteur d'initialisation



Le bloc chiffré aléatoire produit en sortie est **XORé** avec le bloc clair, comme une clé de flux.
→ Le mode CTR transforme un chiffrement par bloc en chiffrement de flux

- **CTR (CounTeR)**

- ✓ Si même bloc clair → blocs chiffrés différents, car clés de flux différentes
- ✓ *Random read access*
- ✓ Parallélisation possible
- ✓ Plus besoin de *padding*, car transformation en chiffrement de flux
- ⚠ La taille du compteur limite la taille du message (*i.e.* le nombre de blocs) qu'on peut envoyer
- ⚠ **XOR** : inversion d'un bit dans le chiffré → inversion dans le clair
- ↳ Requiert un MAC (*Galois CTR mode, GCM* → permet confidentialité **et intégrité**)
- ✓ Le **déchiffrement** utilise le même processus que le chiffrement car on génère la même clé de flux
- ↳ Diminue les coûts d'implémentation (p. ex. on n'implémente pas la fonction de déchiffrement d'AES)

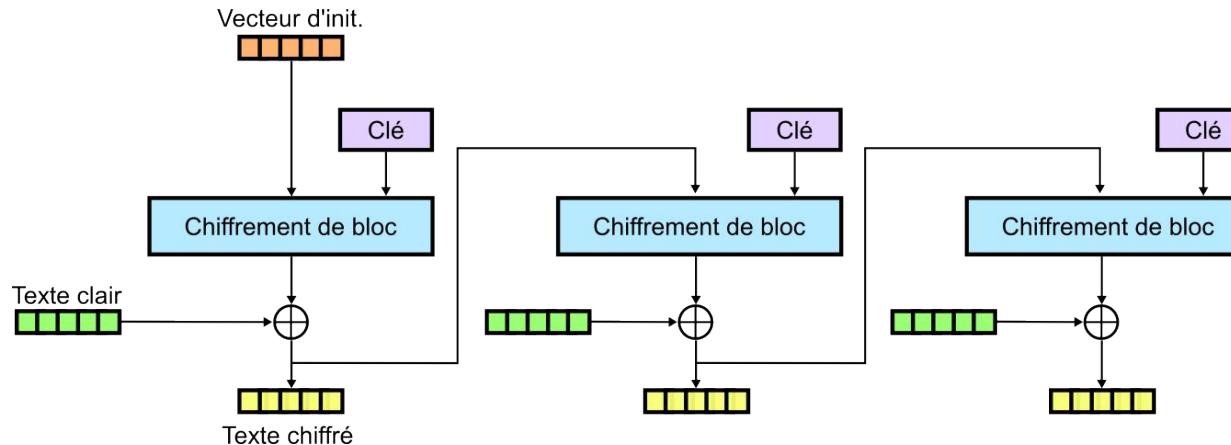


2.7. Modes de fonctionnement

- **CFB** (*Cipher Feedback Block*, chiffrement à rétroaction)

Comme CBC : la sortie d'un bloc est reliée à l'entrée du bloc suivant, ce qui permet de masquer le i -ème bloc du texte clair par le chiffrement du bloc $i-1$ du texte chiffré

Comme CTR : transformation en un chiffrement de flux (asynchrone) → pas besoin de *padding*

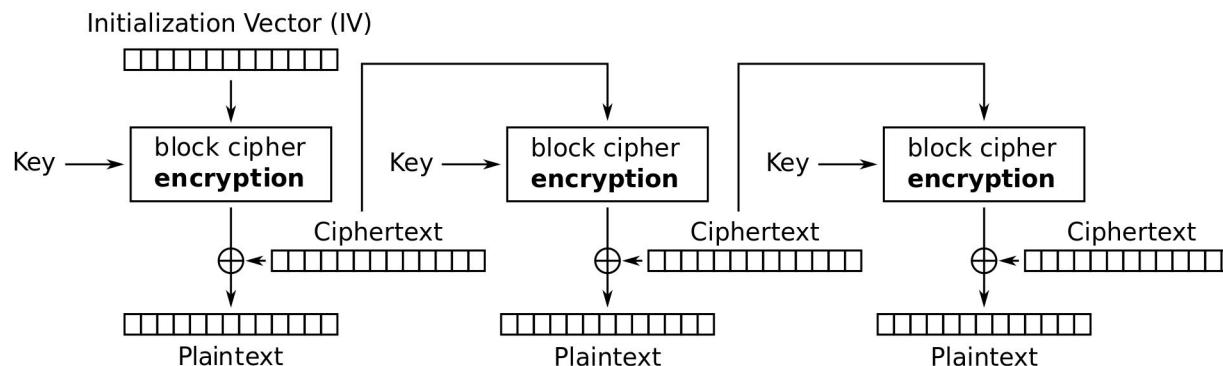


2.7. Modes de fonctionnement

- **CFB** (*Cipher Feedback Block*, chiffrement à rétroaction)

Comme CBC : seul le déchiffrement peut être parallélisé

Comme CTR : utilisation de la même fonction pour chiffrage et déchiffrement



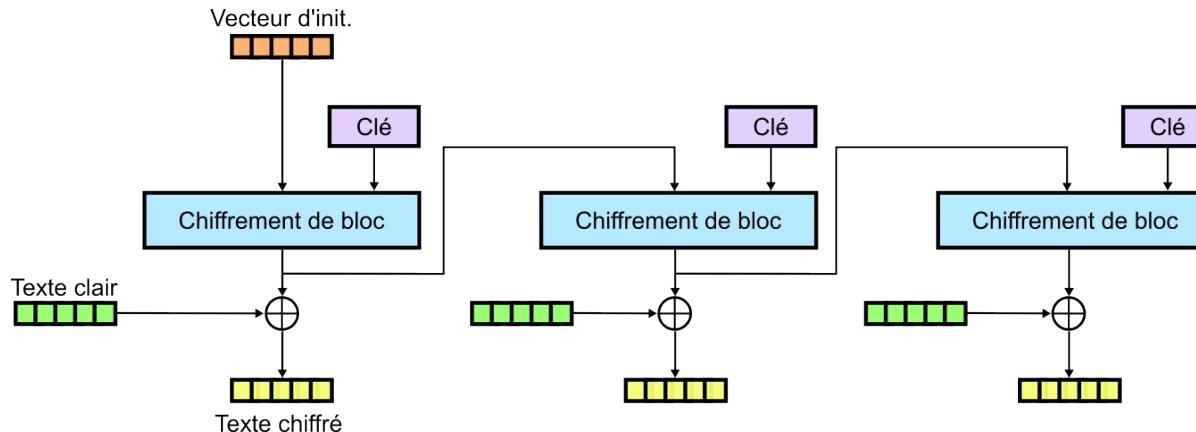
2.7. Modes de fonctionnement

- **OFB** (*Output Feedback*, chiffrement à rétroaction de sortie)

Le flux de clé est obtenu en chiffrant le précédent flux de clé

↳ Plus rapide que CFB, car permet un pré-calcul de toutes les clés de flux

↳ Plus fragile que CFB, car un attaquant n'a besoin de connaître que l'IV d'un message chiffré et le clair d'un autre message chiffré



2.7. Modes de fonctionnement

- **En résumé**

	ECB	CBC	CTR	CFB	OFB
Chiffrement parallélisable	Oui	Non	Oui	Non	Précalcul des keystreams
Déchiffrement parallélisable	Oui	Oui	Oui	Oui	Précalcul des keystreams
<i>Random read access</i>	Oui	Oui	Oui	Oui	Non

3. Sécurité des chiffrements

- **Modèles d'attaques cryptographiques**

- Ils représentent différents scénarios dans lesquels la sécurité du cryptosystème étudié est menacée par un adversaire
 - Adversaire « réaliste » : dispose des meilleurs algorithmes et capacités de calculs actuellement connus
 - À partir des données auxquelles l'adversaire a accès, différents types d'attaques sont possibles
 - Varient en force et en vraisemblance
 - La résistance du chiffrement caractérise son niveau de sécurité
-  **Preuve de sécurité par jeu** (*security game*)
p. ex. IND-CPA, IND-CCA1, ...

- **Attaque sur texte chiffré seul** (*ciphertext-only attack*, COA)

Dans ce scénario, le cryptanalyste a seulement accès à une collection de textes chiffrés, pas aux textes en clair

- Le plus vraisemblable en cryptanalyse
- Le plus faible, car manque d'information
- Chiffrements modernes très résistants aux COA
 - En général, ne peut réussir que si le cryptanalyste dispose d'informations sur le texte en clair : *langue utilisée*, formatage, etc.
- Cas particulier : **attaque par force brute**, qui recherche exhaustivement la clé de déchiffrement parmi toutes les possibilités
 - Tous les chiffrements y sont vulnérables, sauf OTP
 - En combien de temps (tentatives) ?
 - ↳ Sert de comparaison avec les autres attaques (*faster-than-brute-force attacks*)
 - *Attaque par dictionnaire* : s'appuie sur une liste de mots clairs vraisemblables

- **Attaque à texte clair connu** (*known-plaintext attack, KPA*)

Dans ce scénario, le cryptanalyste dispose d'un jeu de textes chiffrés et des textes clairs correspondants, pour une clé donnée. Ces paires aident à comprendre l'algorithme de chiffrement, voire à retrouver la clé.

- **Analyse par crib**

- Les *cribs* sont généralement des parties de texte en clair que l'on sait, **ou bien que l'on suspecte** (mots ou expressions récurrents), être présentes dans le chiffré
 - Exemple : « Cordialement » à la fin des mails
 - Les codes sources chiffrés sont vulnérables, car pleins de mots récurrents
- Cryptanalyse d'Enigma, avant que les Alliés ne capturent un exemplaire de la machine

Enigma : permutation sans point fixe

Une permutation f est dite sans point fixe, s'il n'existe aucun x tq $f(x) = x$

i.e. impossibilité de substituer une lettre à elle-même

→ Vulnérabilité à une analyse par *crib*

1	<table border="1"><tr><td>H</td><td>Q</td><td>E</td><td>Y</td><td>S</td><td>A</td><td>W</td><td>Q</td><td>S</td><td>T</td><td>N</td><td>T</td><td>L</td><td>G</td><td>K</td><td>P</td><td>E</td><td>S</td><td>R</td><td>E</td><td>V</td><td>L</td></tr> <tr><td>S</td><td>E</td><td>C</td><td>R</td><td>E</td><td>T</td><td>M</td><td>E</td><td>S</td><td>S</td><td>S</td><td>A</td><td>G</td><td>E</td><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L	S	E	C	R	E	T	M	E	S	S	S	A	G	E	...								
H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L																									
S	E	C	R	E	T	M	E	S	S	S	A	G	E	...																																
2	<table border="1"><tr><td>H</td><td>Q</td><td>E</td><td>Y</td><td>S</td><td>A</td><td>W</td><td>Q</td><td>S</td><td>T</td><td>N</td><td>T</td><td>L</td><td>G</td><td>K</td><td>P</td><td>E</td><td>S</td><td>R</td><td>E</td><td>V</td><td>L</td></tr> <tr><td>...</td><td>S</td><td>E</td><td>C</td><td>R</td><td>E</td><td>T</td><td>M</td><td>E</td><td>S</td><td>S</td><td>A</td><td>G</td><td>E</td><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L	...	S	E	C	R	E	T	M	E	S	S	A	G	E	...								
H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L																									
...	S	E	C	R	E	T	M	E	S	S	A	G	E	...																																
3	<table border="1"><tr><td>H</td><td>Q</td><td>E</td><td>Y</td><td>S</td><td>A</td><td>W</td><td>Q</td><td>S</td><td>T</td><td>N</td><td>T</td><td>L</td><td>G</td><td>K</td><td>P</td><td>E</td><td>S</td><td>R</td><td>E</td><td>V</td><td>L</td></tr> <tr><td>...</td><td>S</td><td>E</td><td>C</td><td>R</td><td>E</td><td>T</td><td>M</td><td>E</td><td>S</td><td>S</td><td>A</td><td>G</td><td>E</td><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L	...	S	E	C	R	E	T	M	E	S	S	A	G	E	...								
H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L																									
...	S	E	C	R	E	T	M	E	S	S	A	G	E	...																																
4	<table border="1"><tr><td>H</td><td>Q</td><td>E</td><td>Y</td><td>S</td><td>A</td><td>W</td><td>Q</td><td>S</td><td>T</td><td>N</td><td>T</td><td>L</td><td>G</td><td>K</td><td>P</td><td>E</td><td>S</td><td>R</td><td>E</td><td>V</td><td>L</td></tr> <tr><td>...</td><td>S</td><td>E</td><td>C</td><td>R</td><td>E</td><td>T</td><td>M</td><td>E</td><td>S</td><td>S</td><td>A</td><td>G</td><td>E</td><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L	...	S	E	C	R	E	T	M	E	S	S	A	G	E	...								
H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L																									
...	S	E	C	R	E	T	M	E	S	S	A	G	E	...																																
5	<table border="1"><tr><td>H</td><td>Q</td><td>E</td><td>Y</td><td>S</td><td>A</td><td>W</td><td>Q</td><td>S</td><td>T</td><td>N</td><td>T</td><td>L</td><td>G</td><td>K</td><td>P</td><td>E</td><td>S</td><td>R</td><td>E</td><td>V</td><td>L</td></tr> <tr><td>...</td><td>S</td><td>E</td><td>C</td><td>R</td><td>E</td><td>T</td><td>M</td><td>E</td><td>S</td><td>S</td><td>A</td><td>G</td><td>E</td><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L	...	S	E	C	R	E	T	M	E	S	S	A	G	E	...								
H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L																									
...	S	E	C	R	E	T	M	E	S	S	A	G	E	...																																
6	<table border="1"><tr><td>H</td><td>Q</td><td>E</td><td>Y</td><td>S</td><td>A</td><td>W</td><td>Q</td><td>S</td><td>T</td><td>N</td><td>T</td><td>L</td><td>G</td><td>K</td><td>P</td><td>E</td><td>S</td><td>R</td><td>E</td><td>V</td><td>L</td></tr> <tr><td>...</td><td>S</td><td>E</td><td>C</td><td>R</td><td>E</td><td>T</td><td>M</td><td>E</td><td>S</td><td>S</td><td>A</td><td>G</td><td>E</td><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L	...	S	E	C	R	E	T	M	E	S	S	A	G	E	...								
H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L																									
...	S	E	C	R	E	T	M	E	S	S	A	G	E	...																																
7	<table border="1"><tr><td>H</td><td>Q</td><td>E</td><td>Y</td><td>S</td><td>A</td><td>W</td><td>Q</td><td>S</td><td>T</td><td>N</td><td>T</td><td>L</td><td>G</td><td>K</td><td>P</td><td>E</td><td>S</td><td>R</td><td>E</td><td>V</td><td>L</td></tr> <tr><td>...</td><td>S</td><td>E</td><td>C</td><td>R</td><td>E</td><td>T</td><td>M</td><td>E</td><td>S</td><td>S</td><td>A</td><td>G</td><td>E</td><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L	...	S	E	C	R	E	T	M	E	S	S	A	G	E	...								
H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L																									
...	S	E	C	R	E	T	M	E	S	S	A	G	E	...																																
8	<table border="1"><tr><td>H</td><td>Q</td><td>E</td><td>Y</td><td>S</td><td>A</td><td>W</td><td>Q</td><td>S</td><td>T</td><td>N</td><td>T</td><td>L</td><td>G</td><td>K</td><td>P</td><td>E</td><td>S</td><td>R</td><td>E</td><td>V</td><td>L</td></tr> <tr><td>...</td><td>S</td><td>E</td><td>C</td><td>R</td><td>E</td><td>T</td><td>M</td><td>E</td><td>S</td><td>S</td><td>A</td><td>G</td><td>E</td><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L	...	S	E	C	R	E	T	M	E	S	S	A	G	E	...								
H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L																									
...	S	E	C	R	E	T	M	E	S	S	A	G	E	...																																
9	<table border="1"><tr><td>H</td><td>Q</td><td>E</td><td>Y</td><td>S</td><td>A</td><td>W</td><td>Q</td><td>S</td><td>T</td><td>N</td><td>T</td><td>L</td><td>G</td><td>K</td><td>P</td><td>E</td><td>S</td><td>R</td><td>E</td><td>V</td><td>L</td></tr> <tr><td>...</td><td>S</td><td>E</td><td>C</td><td>R</td><td>E</td><td>T</td><td>M</td><td>E</td><td>S</td><td>S</td><td>A</td><td>G</td><td>E</td><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L	...	S	E	C	R	E	T	M	E	S	S	A	G	E	...								
H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L																									
...	S	E	C	R	E	T	M	E	S	S	A	G	E	...																																
10	<table border="1"><tr><td>H</td><td>Q</td><td>E</td><td>Y</td><td>S</td><td>A</td><td>W</td><td>Q</td><td>S</td><td>T</td><td>N</td><td>T</td><td>L</td><td>G</td><td>K</td><td>P</td><td>E</td><td>S</td><td>R</td><td>E</td><td>V</td><td>L</td></tr> <tr><td>...</td><td>S</td><td>E</td><td>C</td><td>R</td><td>E</td><td>T</td><td>M</td><td>E</td><td>S</td><td>S</td><td>A</td><td>G</td><td>E</td><td>...</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table>	H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L	...	S	E	C	R	E	T	M	E	S	S	A	G	E	...								
H	Q	E	Y	S	A	W	Q	S	T	N	T	L	G	K	P	E	S	R	E	V	L																									
...	S	E	C	R	E	T	M	E	S	S	A	G	E	...																																

- **Attaque à texte clair choisi** (*chosen-plaintext attack, CPA*)

Dans ce scénario, le cryptanalyste peut choisir arbitrairement un jeu de textes clairs et obtenir de l'**oracle de chiffrement** les chiffrés correspondants

- **Oracle** : entité informatique qui fournit des informations sur demande : chiffrement, déchiffrement, valeur de hachage, fuite d'information (cf. attaque par canal auxiliaire).
 - Les détails de son fonctionnement ne sont pas censés être connus de l'adversaire, à l'exception de l'algorithme de chiffrement qu'il simule. La clé de chiffrement demeure secrète.
 - Cela le définit comme une **boîte grise**



Preuve de sécurité par jeu : l'oracle y sera représenté par le *challenger*
→ s'oppose à l'adversaire et interagit avec lui

- **Attaque à texte clair choisi** (*chosen-plaintext attack, CPA*)

- Dans les schémas de **chiffrements asymétriques**, la clé utilisée pour chiffrer est distribuée publiquement, ce qui permet les tentatives de CPA
 - Ils doivent donc être résistants aux CPA → **sécurité sémantique** (sémantique *adj.* : qui concerne le sens, la signification)

- Chiffrements de flux résistants : Trivium, Salsa20, Grain-128, etc.

- **Attaque à texte clair choisi adaptative**

(*adaptive chosen-plaintext attack, CPA2*)

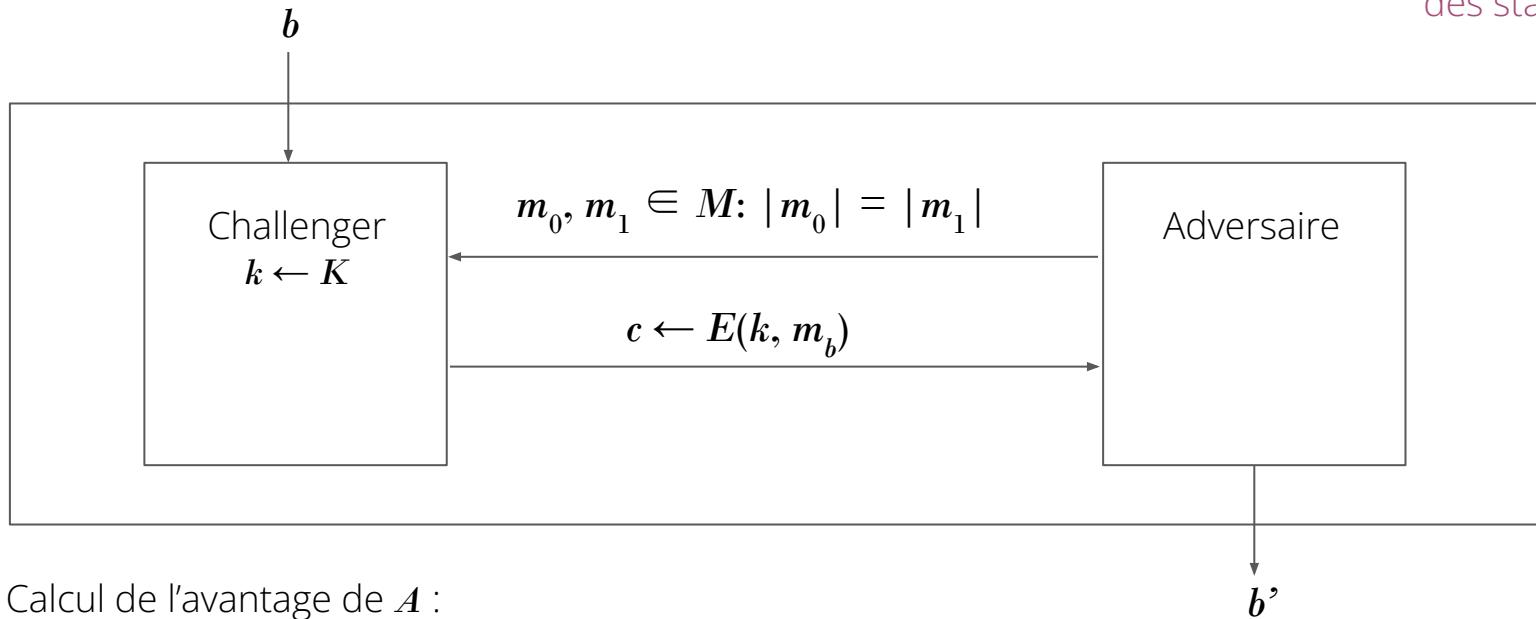
Ici, le cryptanalyste n'est pas obligé de choisir tous les textes clairs dès le début : il peut adapter son choix de texte clair en fonction des chiffrés renvoyés par l'oracle

3.3. CPA

Soit une expérience $\text{EXP}(b)$, avec $b \leftarrow U(0, 1)$ au cours de laquelle :

- l'adversaire A envoie deux messages m_0 et m_1 au *challenger*,
- le *challenger* renvoie le chiffrement c du message m_b à l'adversaire,
- ce dernier doit deviner qui de m_0 ou m_1 a été chiffré : prédiction $b' \in \{0,1\}$.

Répétitions de la procédure, pour calculer des statistiques



Calcul de l'avantage de A :

$$\delta(n) = |\Pr_{c \leftarrow E(k, m_0)}[A(c) = 1] - \Pr_{c \leftarrow E(k, m_1)}[A(c) = 1]|$$

Négligeable si $\delta(n) < \frac{1}{\text{poly}(n)}$ → « IND-CPA », E sémantiquement sécurisé

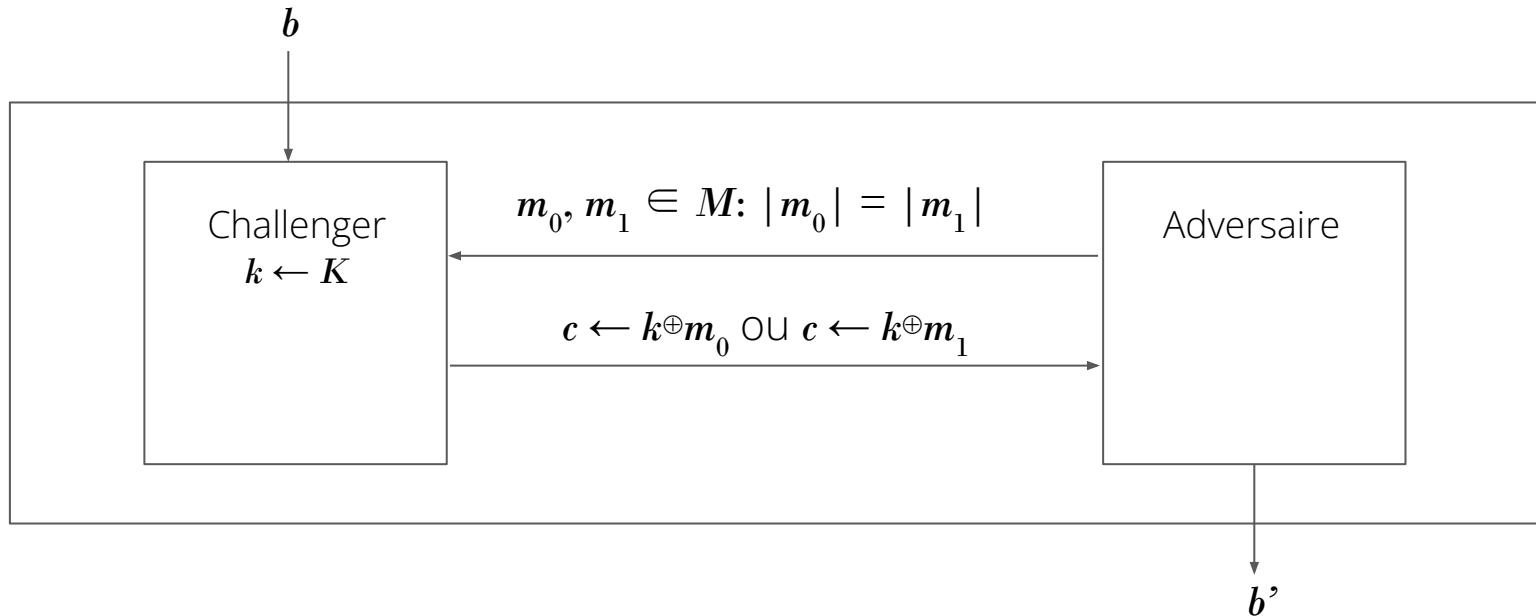
- **Attaque à texte clair choisi** (*chosen-plaintext attack*, CPA)

- **Sécurité sémantique (IND-CPA)** : pour un adversaire réaliste A (disposant des meilleurs algorithmes et capacités de calculs actuellement connus), deux messages chiffrés c_1 et c_2 , qui correspondent à des messages originaux différents m_1 et m_2 , sont calculatoirement indistinguables

Note : aussi appelée *ciphertext indistinguishability*

- Pour A , le chiffré c ne révèle rien sur la sémantique de m .
- $P_{c \leftarrow E(k, \square_o)}[A(c) = 1] = 1 - P_{c \leftarrow E(k, \square_i)}[A(c) = 1]$
 $\delta(n) = | 2 \cdot P_{c \leftarrow E(k, \square_o)}[A(c) = 1] - 1 |$

OTP est sémantiquement sécurisé



$$\delta(n) = |P[A(k \oplus m_0) = 1] - P[A(k \oplus m_1) = 1]| = |\frac{1}{2} - \frac{1}{2}| = 0, \text{ IND-CPA}$$

Rappel : un **XOR** entre une clef aléatoire et n'importe quel message produit une séquence également aléatoire

- **Attaque à texte chiffré choisi** (*chosen-ciphertext attack, CCA*)

Dans ce scénario, le cryptanalyste peut choisir arbitrairement un **nombre limité** de textes chiffrés et obtenir de l'oracle de déchiffrement les textes clairs correspondants

- **Attaque à texte chiffré choisi adaptative**

(*adaptive chosen-ciphertext attack, CCA2*)

Ici, le cryptanalyste n'est pas obligé de choisir tous les textes chiffrés dès le début : il peut adapter son choix de texte chiffré en fonction des textes clairs renvoyés par l'oracle

Sous ce modèle d'attaque, l'indistinguabilité est équivalente à la **non-malléabilité** ($\text{IND-CCA2} \Leftrightarrow \text{NM-CCA2}$)

- **Malléabilité**

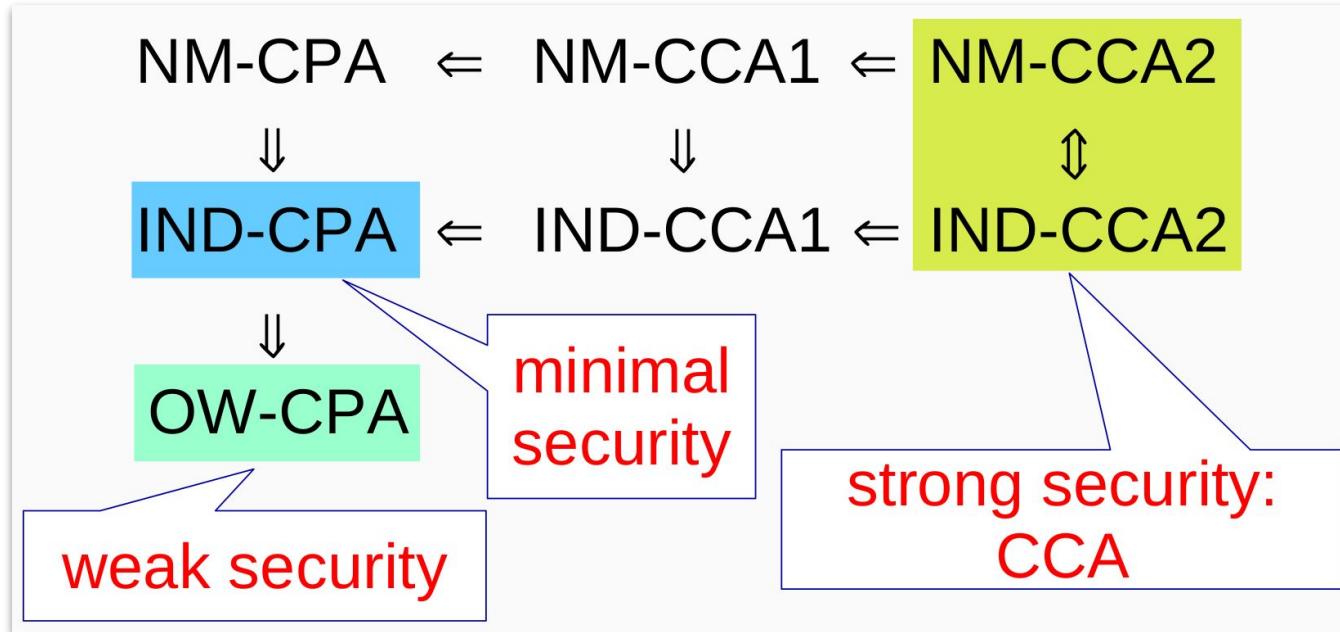
Un cryptosystème est dit malléable s'il est possible de transformer un chiffré d'un message m en un chiffré pour un message $f(m)$ pour une fonction f connue sans connaître le message originel m ni obtenir d'information sur lui.

- Peut permettre à un attaquant de **modifier le contenu** des messages
- Permet d'exécuter des calculs sur des données chiffrés sans les connaître :
 - ↳ Chiffrement homomorphe

- Chiffrement de flux

Avec un second message t , un adversaire peut construire un chiffrement de $m \oplus t$, tq $t \oplus E(m) = t \oplus m \oplus G(k) = E(t \oplus m)$

- **Solutions** : attacher au message un MAC, signature ou ZKP...



À voir dans le chapitre des fonctions de hachages et signatures numériques :
 OW-CPA, *One-Way Chosen-Plaintext Attack*
 EUF-NMA, *Existential Unforgeability under No-Message Attacks*

- **Principe de Kerckhoffs :**

- Axiome guidant la conception des cryptosystèmes (1883)
« La sécurité d'un cryptosystème ne doit reposer que sur le secret de la clé. »
- Autrement dit, on suppose que l'adversaire connaît tous les détails de l'algorithme, exceptée la clé de chiffrement
 - Expression connue sous le nom de la **maxime de Shannon** (1949)
- S'oppose à la **sécurité par l'obscurité**
 - Mesure insuffisante

4. Chiffrements asymétriques

- **Nombres premiers entre eux**

On dit que deux entiers a et b sont premiers entre eux (ou a est (co)premier avec b), si leur plus grand commun diviseur (pgcd) est égal à 1 :

$$\text{pgcd}(a, b) = 1$$

En d'autres termes, s'ils n'ont aucun diviseur autre que 1 et -1 en commun. De manière équivalente, ils sont premiers entre eux s'ils n'ont aucun facteur premier en commun.

- **Théorème de Bézout**

Pour deux entiers relatifs a et b , il existe deux entiers relatifs x et y tq :

$$ax + by = \text{pgcd}(a, b)$$

→ a et b sont copremiersssi l'équation $ax + by = 1$ admet des solutions

- **Indicatrice d'Euler**

Fonction arithmétique de la théorie des nombres, qui à tout entier naturel n non nul associe le nombre d'entiers compris entre 1 et n (inclus) et copremiers avec n .

Exemple pour $n = 8$:

- Les entiers positifs ≤ 8 sont $\{1, 2, 3, 4, 5, 6, 7, 8\}$
- Parmis eux, ceux qui sont copremiers avec 8 sont $\{1, 3, 7, 5\}$
- Donc $\varphi(8) = 4$

- Si n est premier, $\varphi(n) = n - 1$, car $\text{pgcd}(a, n) = 1$
- Si n est le produit de deux nombres premiers distincts p et q : $\varphi(n) = (p - 1) \times (q - 1)$

Explications : il y a $p - 1$ multiples de p qui sont $\leq n$ et $q - 1$ multiples de q qui sont $\leq n$

Donc le nombre total d'entiers qui sont $\leq n$ et premiers avec n :

$$n - (p - 1) - (q - 1) = n - p - q + 1 = (p \times q) - (p + q) + 1 = (p - 1) \times (q - 1)$$

- **Algorithme d'Euclide**

Permet de déterminer si deux nombres entiers sont premiers entre eux

1. Soit deux entiers a et b , avec $a \geq b$
2. On divise a par b et on garde le reste r
3. On remplace a par b et b par r
4. On répète le processus, jusqu'à ce que b soit égal à 0

Le **pgcd** des deux entiers initiaux est la valeur non-nulle de b quand le processus se termine

Exemple

$$\text{pgcd}(252, 105) = ?$$

$$252 \% 105 = 42$$

$$105 \% 42 = 21$$

$$42 \% 21 = 0$$

$$\rightarrow \text{pgcd}(252, 105) = 21$$

Rappel : l'opérateur modulo % donne le reste d'une division **euclidienne**

- **Algorithme d'Euclide étendu**

Permet de calculer le $\text{pgcd}(a, b)$ — comme l'algorithme d'Euclide — ainsi que les entiers x et y du théorème de Bézout tq :

$$ax + by = \text{pgcd}(a, b)$$

Pseudo-code :

```
function extended_gcd(a, b):
    if b == 0:
        return a, 1, 0
    else:
        gcd, x, y = extended_gcd(b, a%b)
        return gcd, y, x - (a//b) * y
```

Algorithme d'Euclide

Pourrait être sur une seule colonne avec l'opérateur %

Exemple : $a = 240, b = 46$

index i	quotient q_{i-1}	Remainder r_i	s_i	t_i
0		240		0
1		46		1
2	$240 \div 46 = 5$	$240 - 5 \times 46 = 10$	$1 - 5 \times 0 = 1$	$0 - 5 \times 1 = -5$
3	$46 \div 10 = 4$	$46 - 4 \times 10 = 6$	$0 - 4 \times 1 = -4$	$1 - 4 \times -5 = 21$
4	$10 \div 6 = 1$	$10 - 1 \times 6 = 4$	$1 - 1 \times -4 = 5$	$-5 - 1 \times 21 = -26$
5	$6 \div 4 = 1$	$6 - 1 \times 4 = 2$	$-4 - 1 \times 5 = -9$	$21 - 1 \times -26 = 47$
6	$4 \div 2 = 2$	$4 - 2 \times 2 = 0$	$5 - 2 \times -9 = 23$	$-26 - 2 \times 47 = -120$

$$240 \times -9 + 46 \times 47 = 2$$

• Arithmétique modulaire

Ensemble de méthodes permettant la résolution de problèmes sur les nombres entiers.

Ces méthodes dérivent de l'étude du reste obtenu par une division euclidienne.

Exemple : sur un cadran d'horloge, 21h00 se lit également « 09h00 du soir ».

9 n'est pas égal à 21, mais sur un cadran de 12 entiers, on dit que 9 et 21 sont « congrus modulo 12 » et cela s'écrit : $9 \equiv 21 \pmod{12}$

Cette relation d'équivalence s'applique également aux entiers relatifs : $-1 \equiv 11 \pmod{12}$

Propriété de la congruence sur les entiers a et $b \in \mathbb{Z}$:

$a \equiv b \pmod{n}$, avec $n \in \mathbb{N}^*$

→ n divise $a - b$

→ $a = b + kn$, avec $k \in \mathbb{Z}$

→ les divisions euclidiennes de a et b par n ont le même reste

Remarques : il n'y a pas de congruence modulo 0, mais la relation d'égalité ;

le module correspond à la base de numération;

un **XOR** est une addition modulo 2 sans retenue

- **Inverse modulaire**

u est l'inverse modulaire de a si $au \equiv 1 \pmod{n}$

On peut alors écrire $u \equiv a^{-1} \pmod{n}$

Condition

u existe ssi $\text{pgcd}(a, n) = 1$

Unicité

$$11 \times 2 = 22 \equiv 1 \pmod{7}$$

$$11 \times 9 = 99 \equiv 1 \pmod{7}$$

Mais 2 est l'unique inverse modulaire de 11 modulo 7, car $9 \equiv 2 \pmod{7}$

- **Calcul de l'inverse modulaire**

Si a et b sont copremiers, alors $\text{pgdc}(a, b) = 1$

Il existe un inverse modulaire

- de a modulo b
- de b modulo a

Par le théorème de Bézout, on a : $ax + by = \text{pgdc}(a, b) = 1$

$ax = 1 + (-yb)$ x est l'inverse modulaire de a modulo b

$by = 1 + (-xa)$ y est l'inverse modulaire de b modulo a

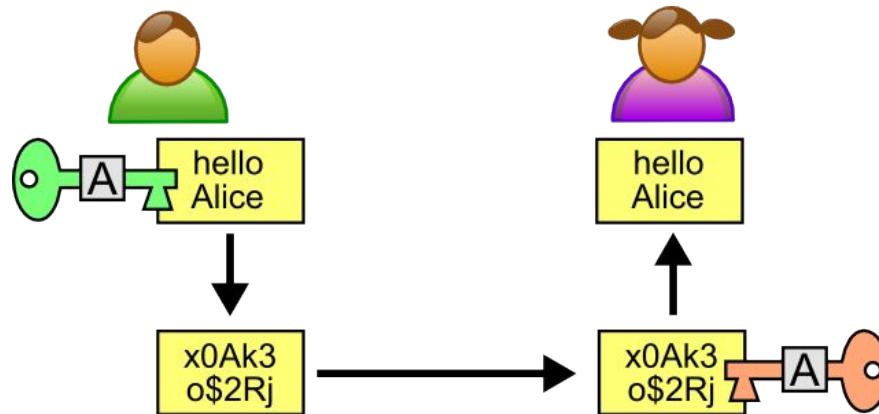
Rappel : $au \equiv 1 \pmod{b} \rightarrow au = 1 + kb$

Ainsi, l'algorithme d'Euclide étendu permettra de calculer l'inverse modulaire

- ### Motivation

En cryptographie symétrique, la même clé est utilisée pour le chiffrement et le déchiffrement (→ symétrie). Cette clé doit donc être **transmise tout en étant maintenue secrète** (→ aussi appelée « cryptographie à clé secrète »), ce qui peut être très contraignant...

La cryptographie asymétrique fonctionne avec **deux clés** : une clé publique, partagée avec tous les expéditeurs pour le chiffrement des messages et une clé privée que seul le destinataire possède pour déchiffrer les messages qui lui sont adressés.



Bob utilise la **clé publique d'Alice** pour chiffrer les messages qu'il lui envoie.

Alice utilise sa **clé privée** pour les déchiffrer.

On parle aussi de « cryptographie à clé publique ».

- **Théorème d'Euler-Fermat**

Pour tout module n et pour tout entier a copremier de n :

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

- **Chiffrement RSA** (Rivest, Shamir, Adleman, 1977)

Soit n le produit de deux nombres premiers distincts p et $q \rightarrow \varphi(n) = (p - 1) \times (q - 1)$

Soit l'exposant e tq $\text{pgcd}(e, \varphi(n)) = 1$

Soit le chiffré C du message clair M , produit avec la clé publique (e, n) :

$$C \equiv M^e \pmod{n}$$

Soit une clé privée (d, n) définie comme l'**inverse modulaire de e modulo $\varphi(n)$** :

$$ed \equiv 1 \pmod{\varphi(n)}$$

On a donc le déchiffrement :

$$C^d = (M^e)^d = M^{ed} = M^{1 + k\varphi(n)} = M \times (M^{\varphi(n)})^k = M \times 1^k \equiv M \pmod{n}$$

- **Preuve de sécurité**

Un attaquant qui intercepterait C aurait besoin de générer la clé secrète d pour le déchiffrer. Il dispose également des valeurs publiques (e, n) . Il lui manque $\varphi(n)$.

La sécurité de RSA repose sur le fait que calculer $\varphi(n)$ seulement à partir de n nécessite de décomposer n en produit de facteurs premiers, ce qui est un problème très difficile à résoudre.

- **Remarque**

À la place de $\varphi(n)$, il est possible d'utiliser l'indicatrice de Carmichael $\lambda(n)$. Cela permet de générer des clés privées plus courtes et d'accélérer les calculs, mais peut créer des vulnérabilités à prendre en considération...

- **Choix des valeurs**

Mathématiquement, e pourrait être n'importe quel entier relatif, tq $\text{pgcd}(e, \varphi(n)) = 1$

Pour la vitesse des calculs, on choisit $1 < e < \varphi(n)$

Une valeur souvent utilisée pour e est 65537 qui vaut 1000000000000001 en base 2.
Cela facilite les calculs.

Le module n est choisi comme le produit de **deux grands nombres premiers**.

Par exemple, la taille de la clé n peut être de 2048 bits.

- De même que la décomposition en facteurs premiers, il existe d'autres problèmes mathématiques dont la difficulté garantit la sécurité de certaines méthodes cryptographiques
- **Problème du logarithme discret**

Le logarithme réel donne l'exposant x dans l'exponentiation de a par x :

$$a^x = b$$

Le logarithme discret est son analogue dans un groupe G .

Il donne l'exposant x dans l'exponentiation de g par x (modulo n) :

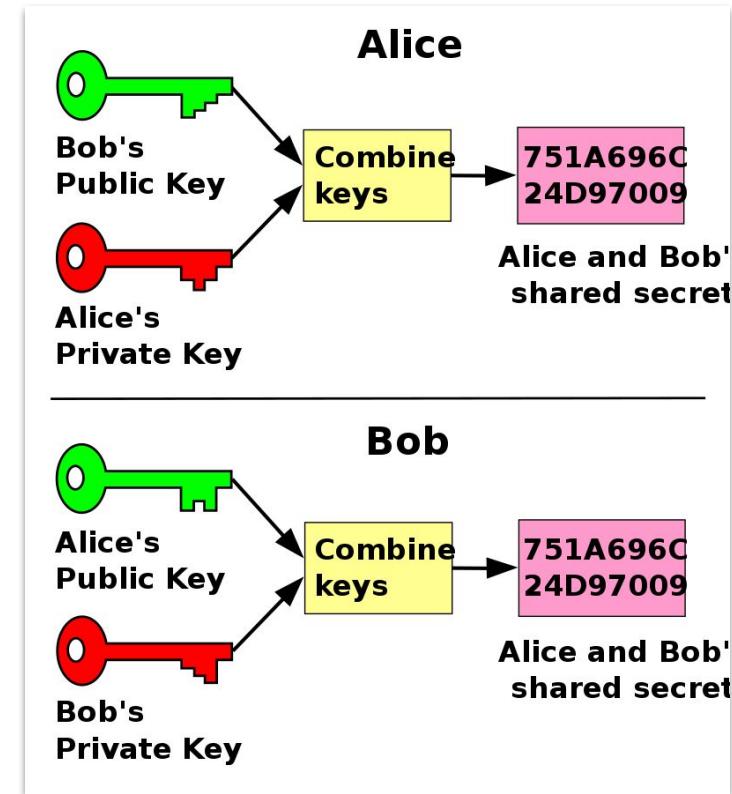
$$g^x \equiv h \pmod{n}$$

Cette exponentiation est facile à calculer, mais le calcul réciproque (logarithme discret) peut être très difficile (pour certaines valeurs de g et de n). **Cette difficulté à calculer g garantit la sécurité du protocole d'échange de clé Diffie-Hellman (DH, 1976).**

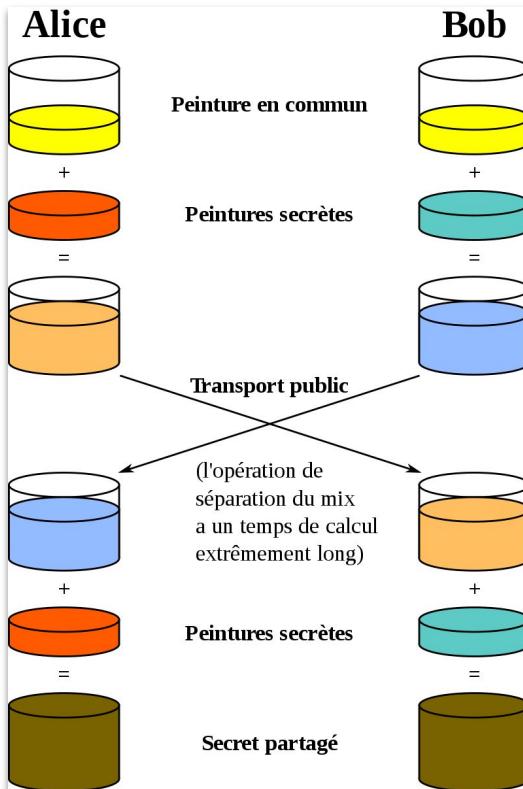
- **Principe général**

Il s'agit d'**établir une clé secrète** commune, que seuls Alice et Bob connaissent

Elle peut être utilisée ensuite pour un **chiffrement symétrique** des données à transmettre



- Calculs**



Variables publiques : le module n et le « générateur » g (échangées au début du *handshake* ou bien définies dans les standards)

Variables privées :

- Alice choisit $a \in [1, n]$
- Bob choisit $b \in [1, n]$

Étapes :

1. Alice calcule $g^a \pmod{n}$ et le transmet à Bob
2. Bob calcule $g^b \pmod{n}$ et le transmet à Alice
3. Alice calcule $(g^b)^a \pmod{n} = g^{ab} \pmod{n}$
4. Bob calcule $(g^a)^b \pmod{n} = g^{ab} \pmod{n}$

Même si Eve (*eavesdropper*) intercepte $g^a \pmod{n}$ ou $g^b \pmod{n}$, elle ne pourra pas extraire les couleurs privées a et b (même en connaissant g et n), et ne pourra donc pas deviner $g^{ab} \pmod{n}$, la couleur partagée.

• Intuition

Ci-contre : un cadran avec $n = 11$ valeurs

L'ensemble des entiers de 0 à 10 muni de la multiplication forme le groupe multiplicatif des entiers **modulo 11**, noté $(\mathbb{Z}/11\mathbb{Z})^\times$

C'est un « groupe », car

- la loi de composition (multiplication) est associative
- la multiplication est interne : multiplier n'importe quelle paire d'élément modulo 11 résulte en un autre élément du groupe
- il existe un élément neutre : multiplier un nombre par 1 le laisse inchangé
- chaque élément non-nul admet un élément symétrique (ici, un inverse) :

$$1 \times 1 \equiv 1 \pmod{11}$$

$$6 \times 2 \equiv 1 \pmod{11}$$

$$2 \times 6 \equiv 1 \pmod{11}$$

$$7 \times 8 \equiv 1 \pmod{11}$$

$$3 \times 4 \equiv 1 \pmod{11}$$

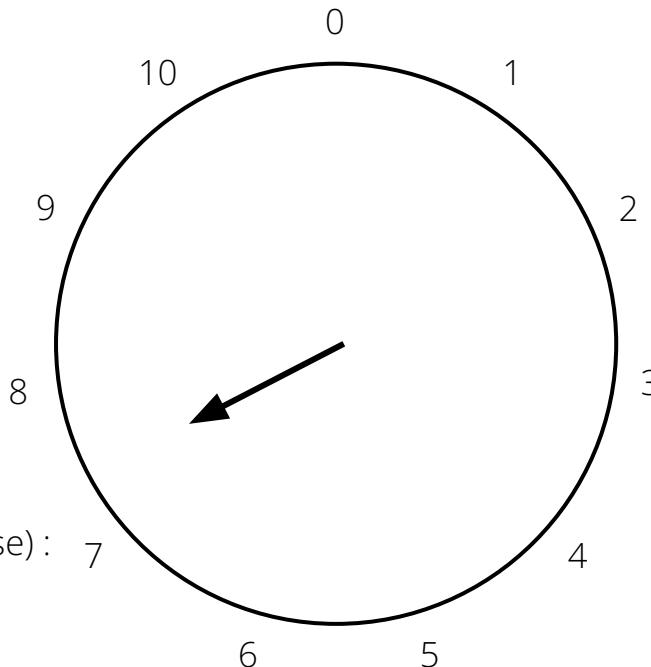
$$8 \times 7 \equiv 1 \pmod{11}$$

$$4 \times 3 \equiv 1 \pmod{11}$$

$$9 \times 5 \equiv 1 \pmod{11}$$

$$5 \times 9 \equiv 1 \pmod{11}$$

$$10 \times 10 \equiv 1 \pmod{11}$$



L'ordre du groupe est son nombre d'éléments $|(\mathbb{Z}/11\mathbb{Z})^\times|$.

- **Intuition**

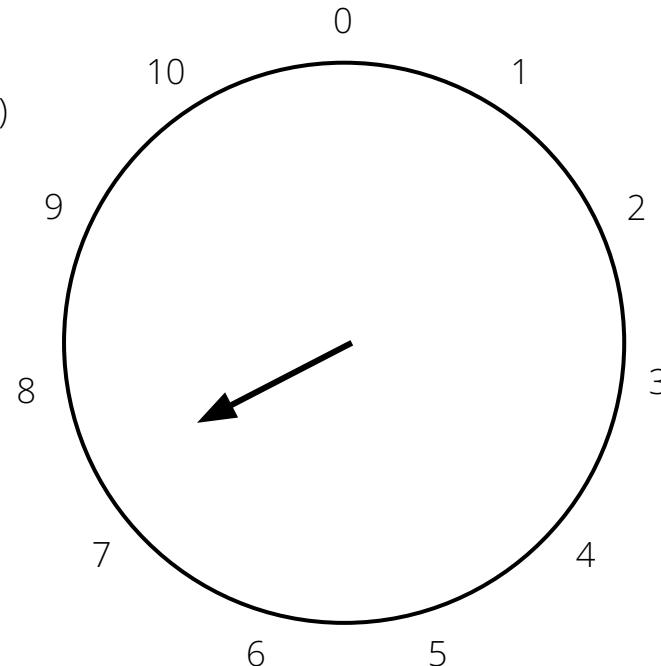
Un groupe est « cyclique » quand il contient un élément g (générateur) à partir duquel tous les autres éléments peuvent être obtenus par application répétée de la loi de composition (ici, la multiplication). Autrement dit, tout élément du groupe peut s'écrire g^k , avec k entier

$$\begin{array}{ll} 2^1 \equiv 2 \pmod{11} & 2^6 \equiv 9 \pmod{11} \\ 2^2 \equiv 4 \pmod{11} & 2^7 \equiv 7 \pmod{11} \\ 2^3 \equiv 8 \pmod{11} & 2^8 \equiv 3 \pmod{11} \\ 2^4 \equiv 5 \pmod{11} & 2^9 \equiv 6 \pmod{11} \\ 2^5 \equiv 10 \pmod{11} & 2^{10} \equiv 1 \pmod{11} \end{array}$$

Note : testez avec 5...

Le nombre de générateurs est égal à $\varphi(n)$, qui vaut ici 10.

Dans un groupe non-cyclique, il est possible de calculer le logarithme discret plus vite que par force brute (algorithme rho de Pollard, algorithme $p - 1$ de Pollard)



- **Intuition**

Question :

Soit $g^k \equiv 8 \pmod{n}$, avec $g = 2$ et $n = 12$.

Quelle est la valeur de k ?

« L'aiguille pointe 8h00 et le point de départ était 2h00. »

Réponse :

$k = 5$

$$2^5 \pmod{12} = 32 \% 12 = 8$$

On comprend que l'ordre du groupe (la « taille du cadran ») doit être le plus grand possible, afin qu'il ne soit pas envisageable d'inverser l'exponentiation (calcul du logarithme discret) par force brute.

- **Choix des valeurs**

La difficulté du calcul du logarithme discret repose sur (par ordre d'importance) :

1. un module n premier, pour garantir que le groupe soit cyclique, et très grand, pour se protéger des attaques par force brute

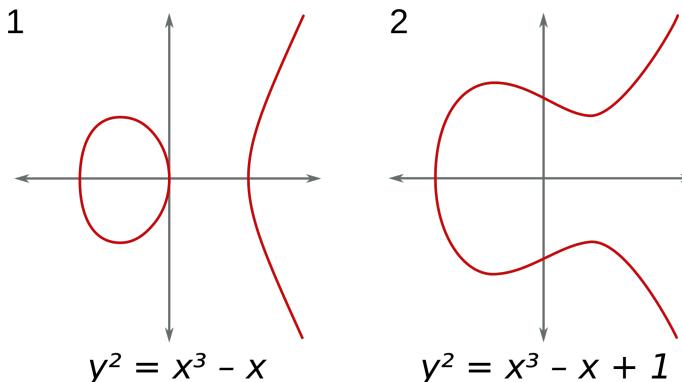
Les tailles des n utilisés varient entre 2048 et 4096 bits

2. un générateur g qui doit être une racine primitive modulo n , autrement dit un générateur du groupe $\mathbb{Z}/n\mathbb{Z}$

On utilise souvent $g = 2$ ou $g = 3$

(pour plus de sécurité, des valeurs de 5, 7, 17 ou 19 sont aussi utilisées)

- Le problème du logarithme discret est encore plus difficile à résoudre sur des courbes elliptiques que dans un corps fini
 - On s'autorise donc à utiliser des clés plus courtes (256 ou 384 bits)
 - Cela **accélère le protocole** (moins de calculs)
 - HTTPS, SSL et TLS
- Équation de Weierstrass : $y^2 \equiv x^3 + ax + b \pmod{n}$
- **Deux caractéristiques notables** d'une courbe elliptique :
 - Symétrique par rapport à l'axe des abscisses
 - Intersection avec une droite en 3 points maximum



4.5. Courbes elliptiques

- Addition de deux points distincts $P + Q = R$

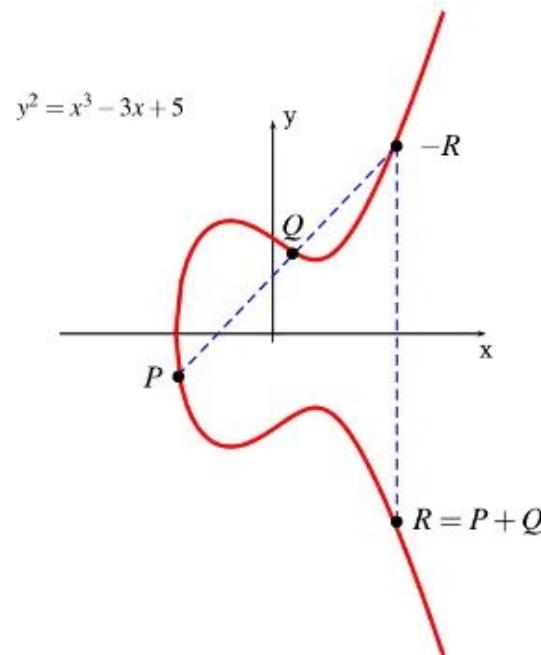
$$\lambda = \frac{y_q - y_p}{x_q - x_p}$$

$$x_r = \lambda^2 - x_p - x_q$$

$$y_r = \lambda(x_p - x_r) - y_p$$

- Addition d'un point P avec lui-même
 - Implique la tangente à la courbe (dérivée en P)

$$\lambda = \frac{3x_p^2 + a}{2y_p}$$

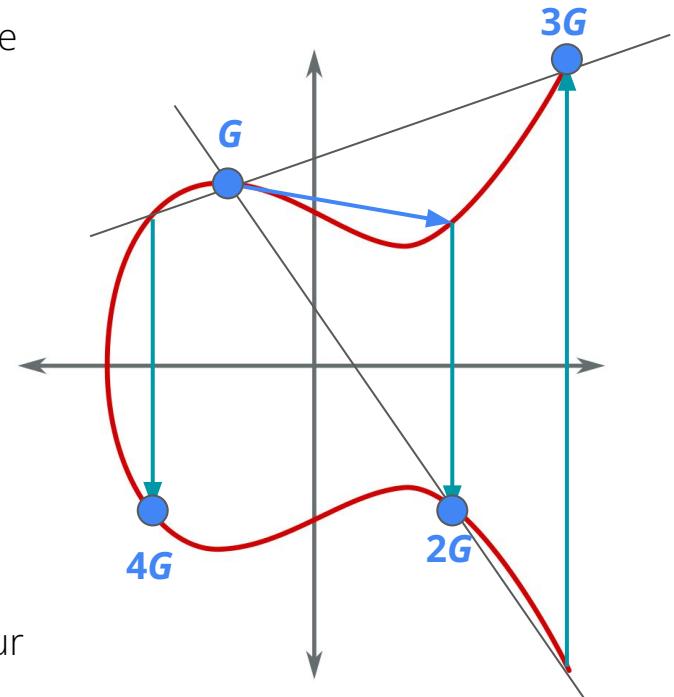


NB : tous les calculs se font modulo n , ordre de la courbe, *i.e.* le nombre de points sur la courbe

- **Multiplication par un scalaire sur une courbe elliptique :**

opération consistant à additionner un point à lui-même sur une courbe elliptique, de manière répétée

- Le **générateur G** est un point de départ sur la courbe
 - La **tangente en G** coupe la courbe en $-2G$
 - Par symétrie axiale on obtient $2G$ ($= G+G$)
 - La droite ($G+2G$) coupe la courbe en $-3G$
 - Etc.
- Problème du logarithme discret : pour un point $P = aG$ sur la courbe, quelle est la valeur du scalaire a ?
- Pour définir la taille des clés, on définit une valeur maximum sur l'axe des x



Comptage des points sur une courbe elliptique

- Plusieurs méthodes :
 - Exhaustive (ou force brute)
 - Algorithme de Schoof
 - Algorithme de Schoof-Elkies-Atkin (SEA)

x	x^2	$x^3 + x + 3$	y	Points
0	0	3	-	
1	1	5	-	
2	4	6	-	
3	2	5	-	
4	2	1	1, 6	(4, 1) (4, 6)
5	4	0	0	(5, 0)
6	1	1	1, 6	(6, 1) (6, 6)

Exemple de l'approche exhaustive pour

$$y^2 \equiv x^3 + x + 3 \pmod{7}$$

Les six points faisant partie de cette courbe sont :
 $(4,1), (4,6), (5,0), (6,1), (6,6)$ et \mathbf{O} , point imaginaire à l'infini.

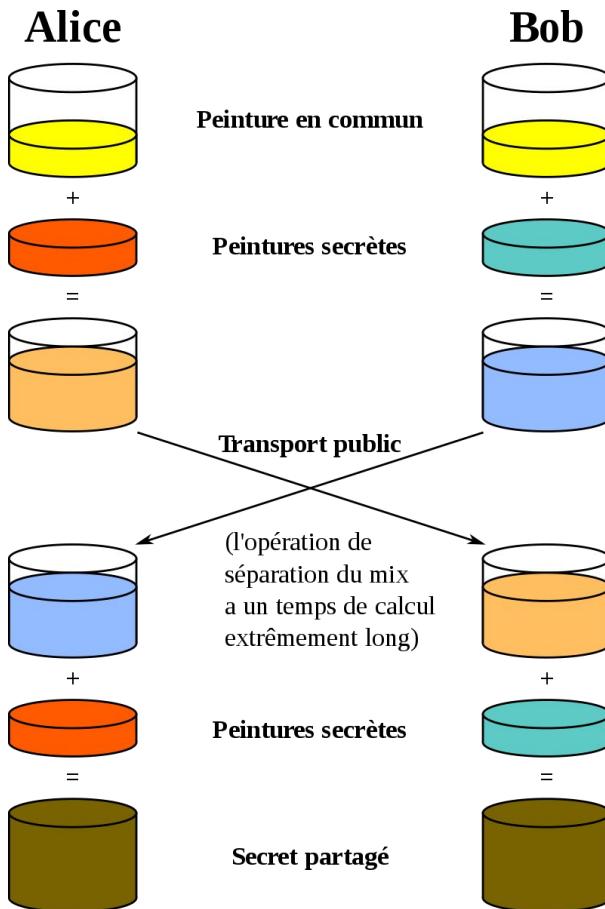
Pour trouver les éléments générateurs de cet ensemble de points, on vérifie si la multiplication de chaque point par un scalaire permet de retrouver tous les autres points.

Trouver les éléments générateurs

- Soit $(4, 1)$ le point de départ
 - Addition avec lui même :
 - $\lambda = (3 \times 4^2 + 1) / (2 \times 1) = 49/2 \equiv 0 \pmod{7}$
 - $x = -4 - 4 = -8 \equiv 6 \pmod{7}$
 - $y = 0 - 1 \equiv 6 \pmod{7}$
 - Le point $(6, 6)$ est généré
 - Addition de $(4, 1)$ et $(6, 6)$:
 - $\lambda = (6 - 1) / (6 - 4) = 5/2$
 - $\lambda^2 = 25/4 \equiv 4/4 \pmod{7} = 1$
 - $x = 1 - 4 - 6 = -9 \equiv 5 \pmod{7}$
 - $y = 5/2 \times (4 - 5) - 1 = -7/2 \equiv 0 \pmod{7}$
 - Le point $(5, 0)$ est généré
 - Addition de $(4, 1)$ et $(5, 0)$:
 - ...
 - Les points $(6, 1)$, $(4, 6)$ et \mathbf{O} sont ensuite générés
 - L'élément $(4, 1)$ est donc générateur (ordre 6)

- Pour $(4, 6)$ on obtient $(4, 6)$ puis $(6, 1)$ puis $(5, 0)$ puis $(6, 6)$ puis $(4, 1)$ puis \mathbf{O} puis $(4, 6)$ etc.
- Pour $(5, 0)$ il est d'ordre 2.
- Pour $(6, 1)$, on obtient $(6, 1)$ puis $(6, 6)$, il est donc d'ordre 3.
- Pour $(6, 6)$ on obtient $(6, 6)$ puis $(6, 1)$ puis \mathbf{O} , il est donc d'ordre 3.

4.5. Courbes elliptiques



Variables publiques :

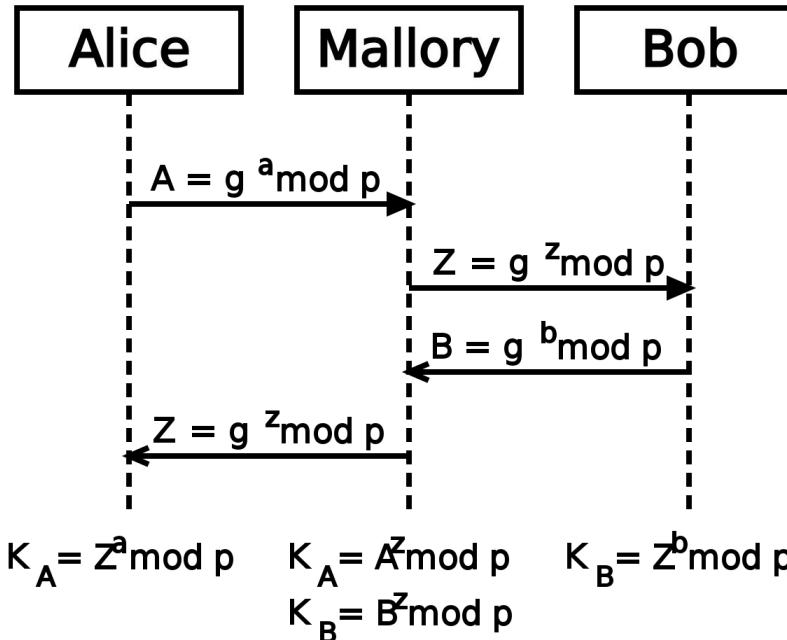
- une **courbe elliptique** spécifique
- un générateur **G**

Variables privées (grand nombres) :

- Alice choisit **$a \in [1, n]$** (n est l'ordre de la courbe)
- Bob choisit **$b \in [1, n]$**

Étapes :

1. Alice calcule **aG** et le transmet à Bob
2. Bob calcule **bG** et le transmet à Alice
3. Alice calcule **abG**
4. Bob calcule **abG**



- Les protocoles DH et ECDH sont sensibles à « l'attaque de l'homme du milieu »
 - C'est pourquoi, ils sont très souvent utilisés avec RSA pour authentifier l'origine des messages reçus.