

INTRODUCTION À LA SÉCURITÉ INFORMATIQUE SÉANCE 5

Marc-Olivier Killijian

UQAM, CNRS

INF4471 A21

ALGO D'EUCLIDE

- Autre exemple : $a=9945$ $b=3003$
 - Calculer $\text{PGCD}(a,b)$
 - Calculer u,v tq $a.u+b.v = \text{PGCD}(a,b)$

ALGO D'EUCLIDE

- Autre exemple : $a=9945$ $b=3003$
- Calculer $\text{PGCD}(a,b)$
- Calculer u,v tq $a.u+b.v = \text{PGCD}(a,b)$
- Solution : $\text{PGCD}=39$ $u=-16$ $v=53$

ZIP(HISTOIRE)& VOCABULAIRE

PRÉ-ZIP

- Cryptologie = **Cryptographie** + **Cryptanalyse**
 - « **faire** des codes secrets »
vs. « **casser** des codes secrets »
- **Chiffrer** pas ~~rypter~~ ou ~~en~~crypter : $c = ENC(m, K_{ENC})$
 - Objectif : sans K_{DEC} **impossible** de retrouver m à partir de c
- **Déchiffrer** pas ~~décrypter~~: $m = DEC(c, K_{DEC})$
 - **Décrypter** : trouver m sans K_{DEC} (cryptanalyse)
- Chiffrement **symétrique** : $K_{ENC} = K_{DEC}$
- Chiffrement **asymétrique** : $K_{ENC} \neq K_{DEC}$

ZIP

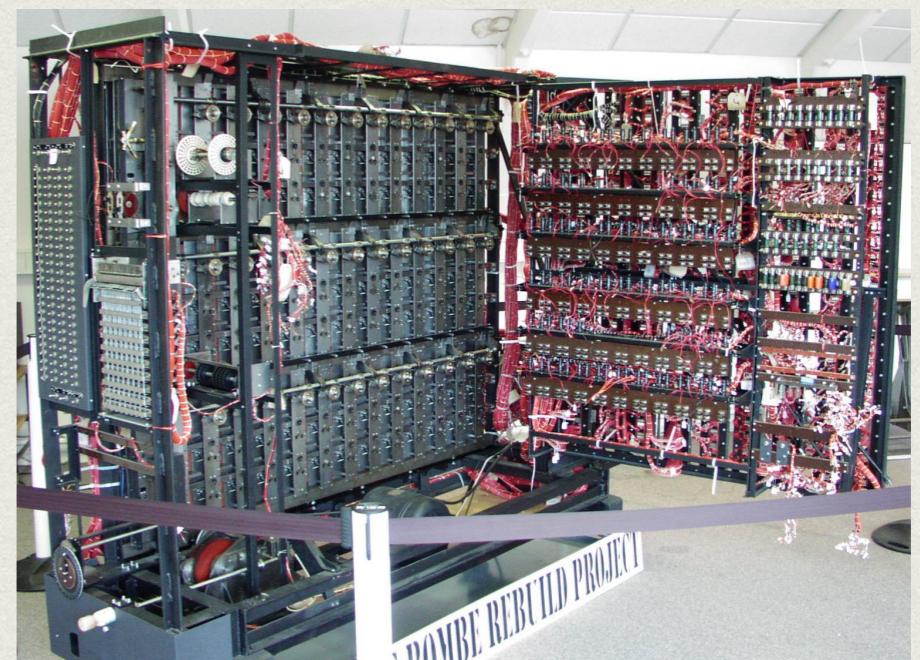
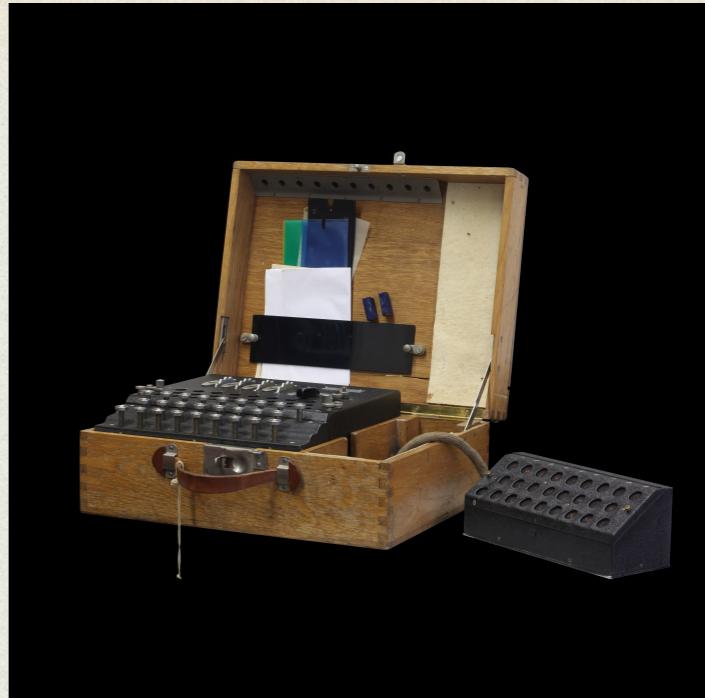
- **Sténographie** : cacher les messages ou l'existence de messages
 - ex. : Tablettes de cires, encre sympathique, etc.
 - moderne : message chiffré caché dans une vidéo (**tatouage**)
- Chiffrement par **décalage** : scytale, chiffre de césar, Vigénère, etc.
- Chiffrement par **substitution** : mono- ou poly-alphabétique
 - analyse **fréquentielle** -> invention de la **cryptanalyse**



Image : Wikimedia

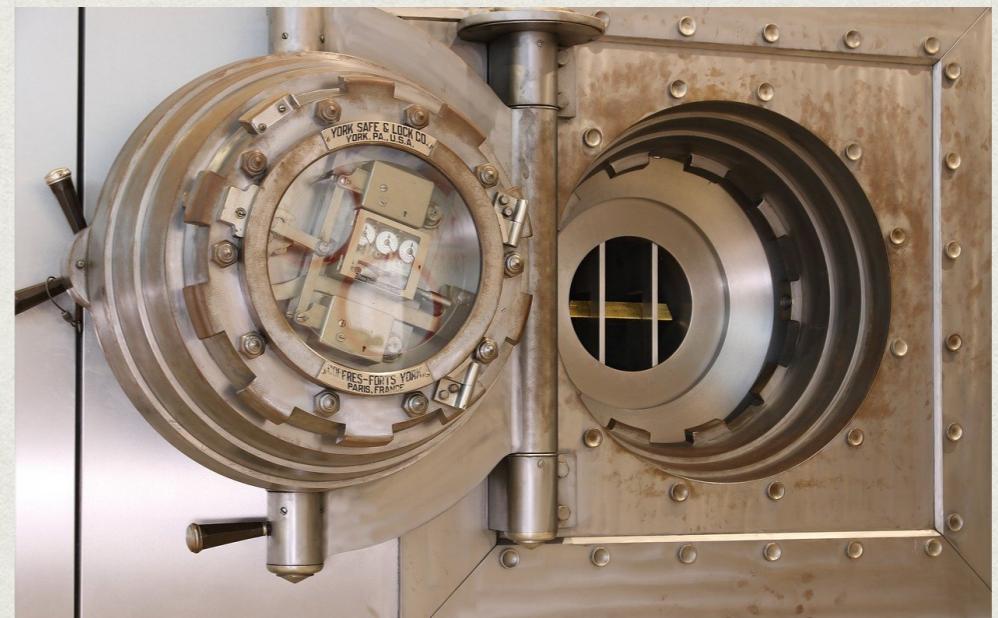
POST-ZIP

- **Enigma** : machine électromagnétique de chiffrement **symétrique**
 - Allemagne, Pologne, Angleterre, France
- Marian **Rejewski**, Alan **Turing**
- La **Bombe**



PRINCIPES DE KERCKHOFFS

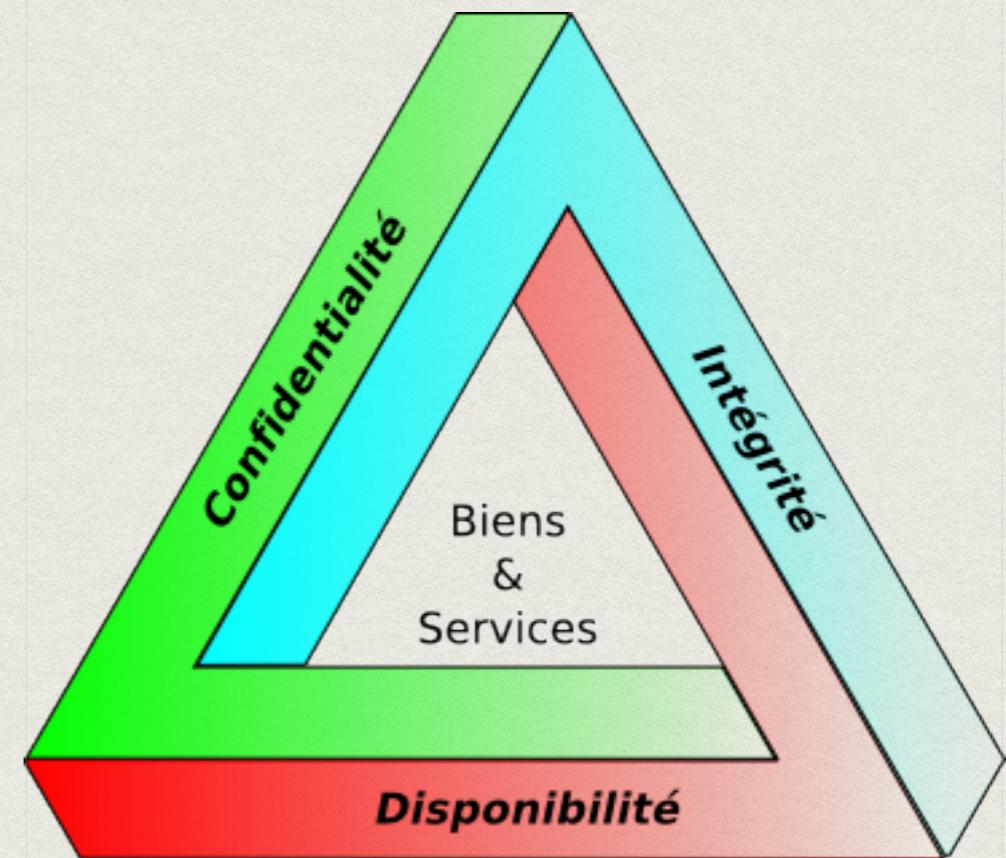
- Dans *La Cryptographie militaire*, 1883
 1. « La sécurité d'un système ne doit pas être fondée sur son caractère secret »
 2. « Seule une donnée de petite taille (clé) doit assurer la sécurité »
- La confidentialité ne repose pas sur le *secret de l'algorithme* mais sur celui de la clé
- Code ouvert qui peut être analysé par des experts



CRYPTOGRAPHIE: INTRODUCTION

OBJECTIFS DE LA CRYPTOGRAPHIE

- Confidentialité
- Intégrité
- Authenticité



ADVERSAIRES (FAIBLE À FORT)

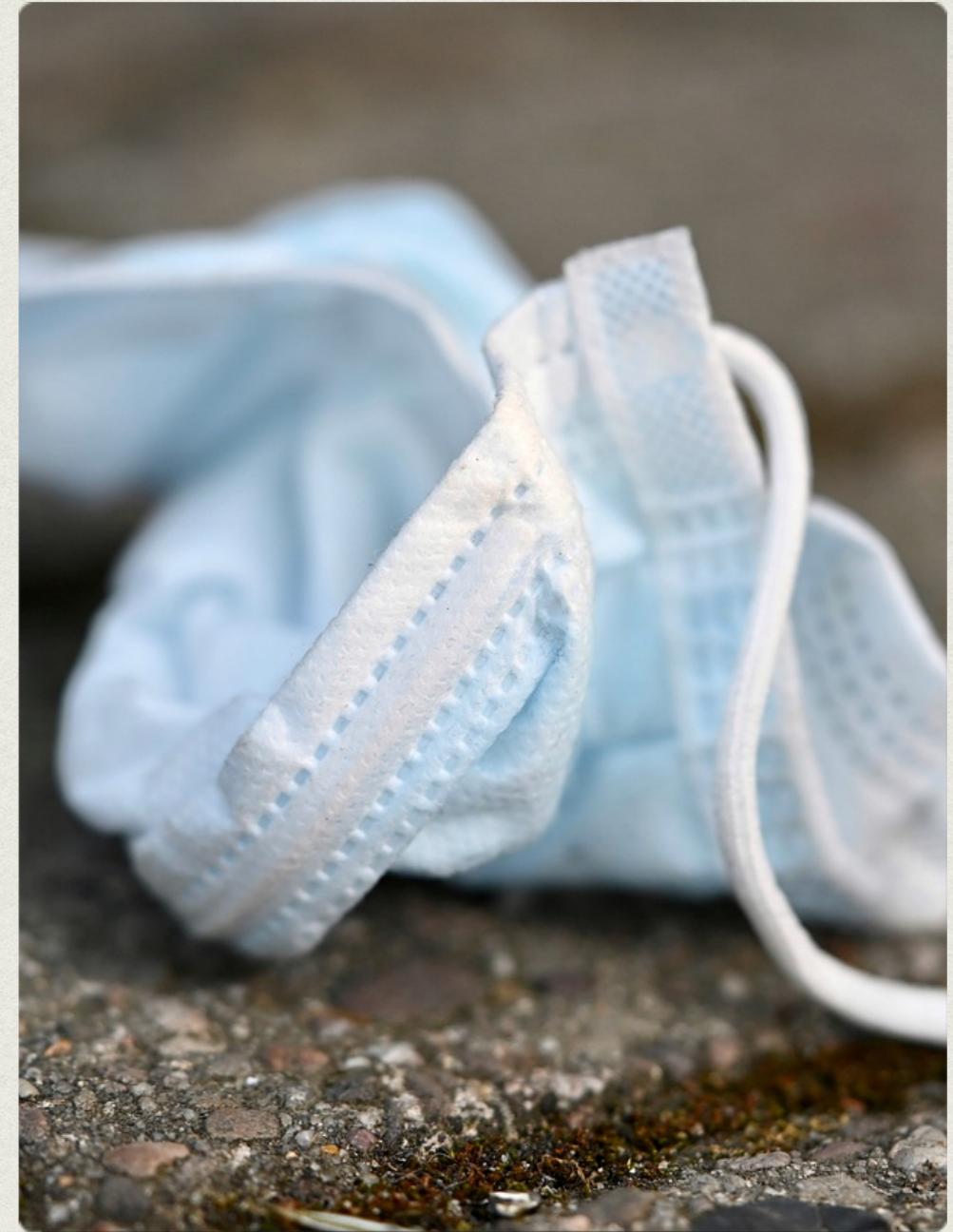
- Attaque à **texte chiffré** : un ou plusieurs chiffrés
- Attaque à texte **clair connu** : un ou plusieurs couples <clair; chiffré>
- Attaque à texte **clair choisi** (oracle de chiffrement) : une ou plusieurs paires <clair; chiffré> dont clair choisi
- Attaque à texte **chiffré choisi** (oracle de déchiffrement) : une ou plusieurs paires <clair; chiffré> dont chiffré choisi

DIFFÉRENTS TYPES DE SÉCURITÉ

- **Sécurité calculatoire** : impossibilité pour l'adversaire de briser une hypothèse cryptographique en temps raisonnable
 - factorisation du produit de grands nombres premiers, logarithme discret
 - recherche exhaustive possible si puissance de calcul illimitée
- **Sécurité au sens de la théorie de l'information** : impossibilité pour l'adversaire d'exploiter les observations qu'il fait
 - le chiffré n'apporte aucune information pour trouver le clair
 $p(M = m \mid C = c) = p(M = m)$
 - meilleure stratégie : deviner de façon aléatoire

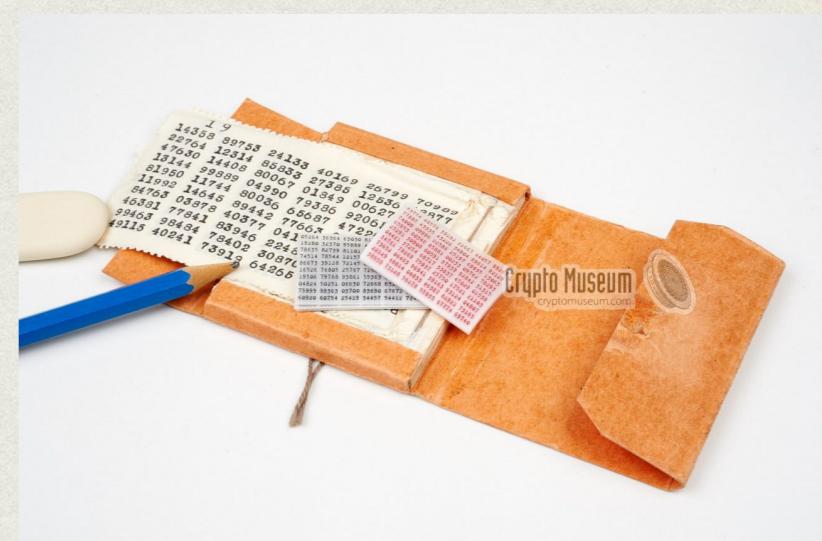
MASQUES JETABLES ONE-TIME PAD

- Sécurité inconditionnelle !
 - Chiffrement Sym. Vernam 1917
 - Preuve Shannon 1949
- Exemple pour $(m, k \in \{0,1\}^n)$
 - Chiffrement $m \oplus k = c$
 - Déchiffrement $k \oplus c = m$



CLÉ À USAGE UNIQUE

- Sécurité inconditionnelle !
 - ssi les clés sont à usage unique !
$$c_1 = m_1 \oplus k \text{ et } c_2 = m_2 \oplus k$$
$$c_1 \oplus c_2 = (m_1 \oplus k \oplus m_2 \oplus k) = m_1 \oplus m_2$$
$$c_1 \oplus c_2 \oplus m_1 = m_1 \oplus m_2 \oplus m_1 = m_2$$
 - ssi les clés sont distribués de façon sécuritaire
 - ssi les clés sont parfaitement aléatoires
- et il faut de nombreuses clés
- **Théorème** (Shannon) : pour la transmission d'un message privé de n bits, il est nécessaire et suffisant de partager une clé aléatoire de n bits



Images : Crypto Museum

CRYPTOGRAPHIE MODERNE

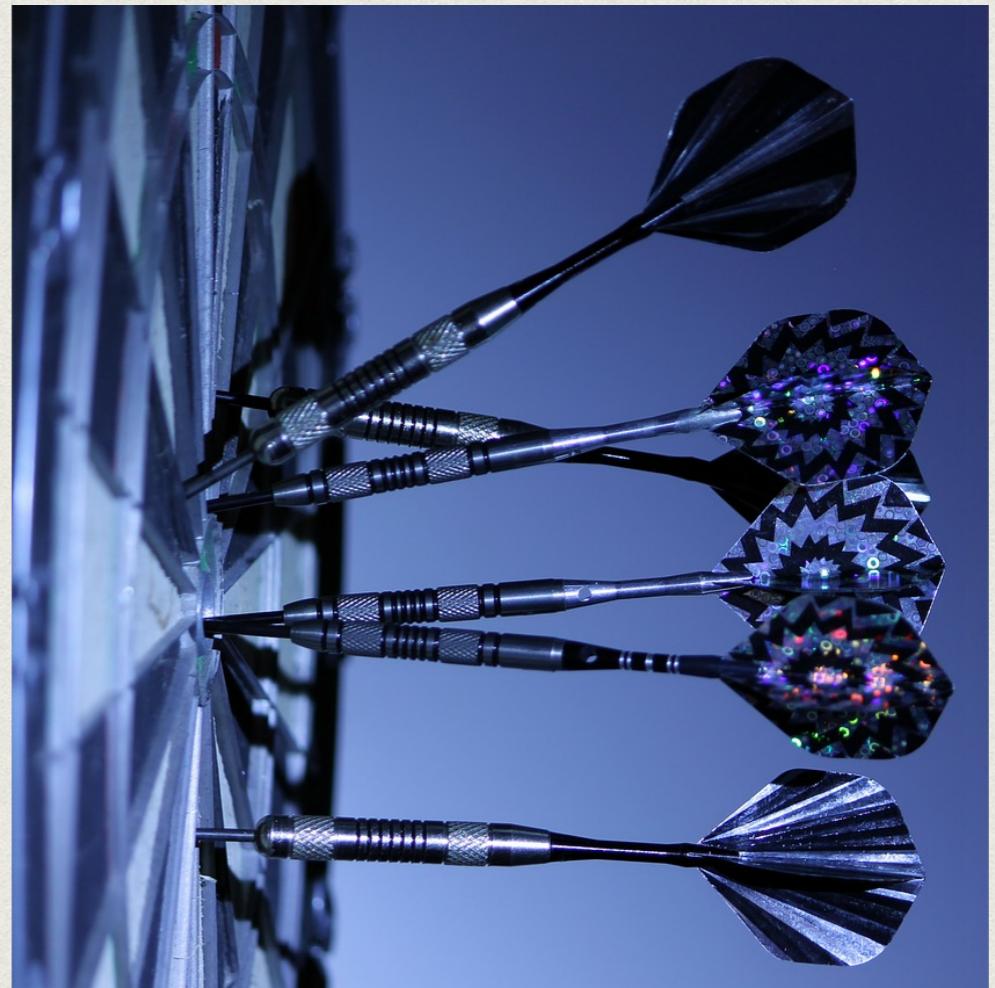
CRYPTO IDÉALE

- Longueur de clé **indépendante** de la taille des messages
- **Ré-utilisation** de la clé
- ▶ **Impossible** d'avoir une sécurité inconditionnelle (*Shannon*)



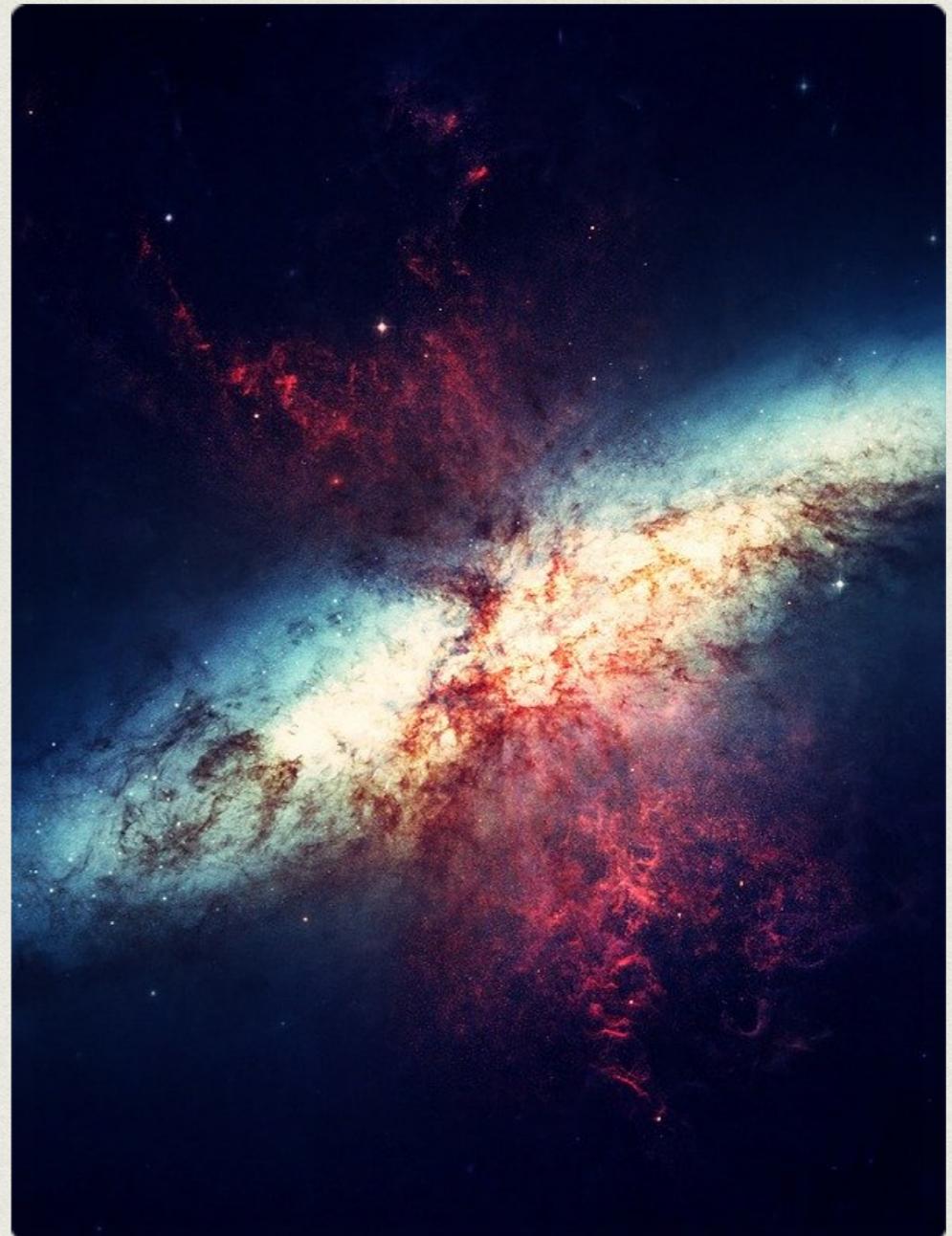
SÉCURITÉ PLUS RÉALISTE

- Sécurité plus **faible** : résister à tout sauf recherche (quasi-) **exhaustive**
 - déchiffrer avec **toutes les clés possibles**
 - conserver les messages **corrects**
 - trouver le **message initial**
- **Sécurité calculatoire** : attaqué **que** par recherche exhaustive
 - difficulté dépend de la **taille de clé**



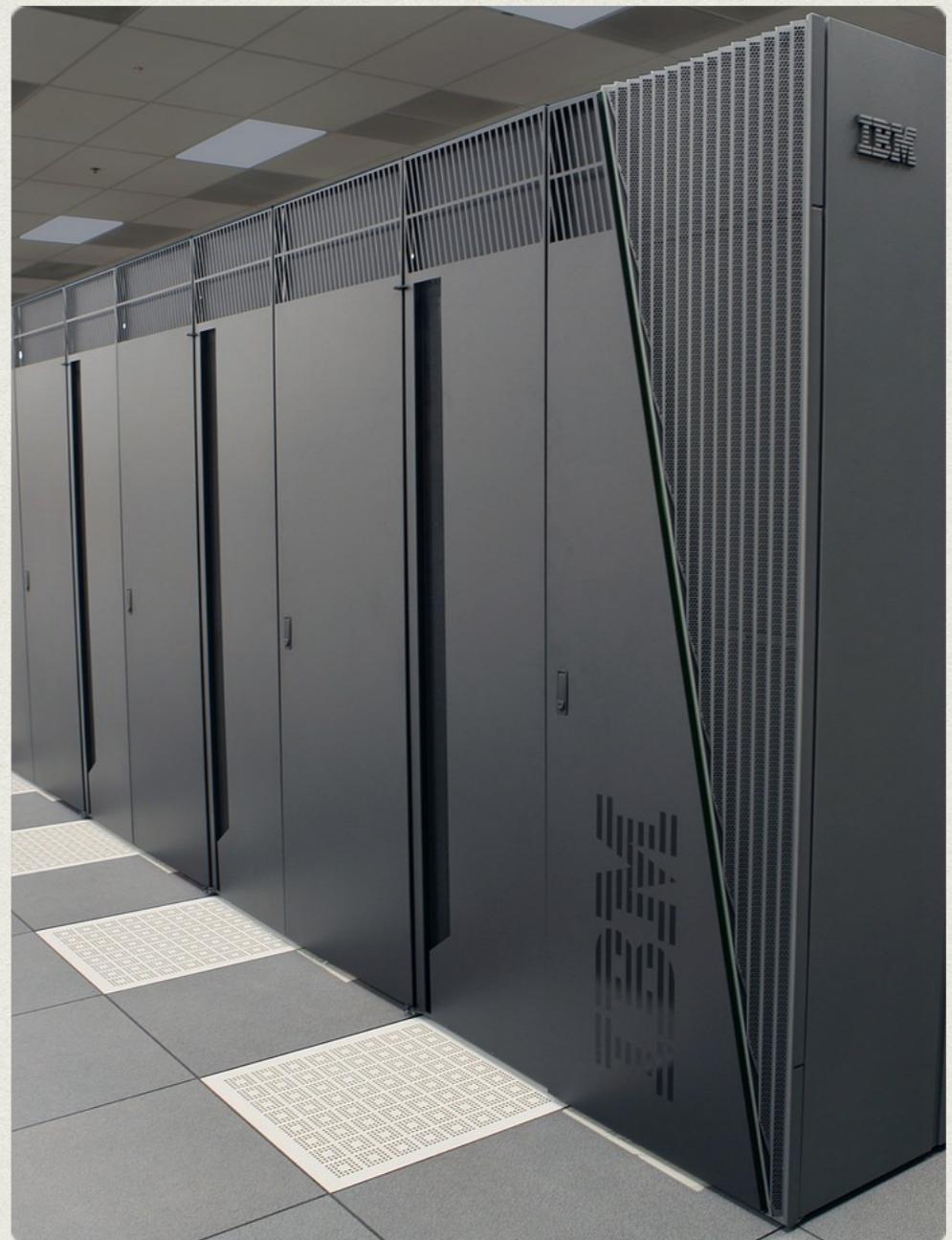
TAILLES DE CLÉS

- Si DEC(c, k) prend 1ns
- Recherche exhaustive avec des clés de longueur :
 - 16 bits : $2^{16} * 10^{-12}$ s = 66 µs
 - 32 bits : $2^{32} * 10^{-12}$ s = 430 ms
 - 56 bits : $2^{56} * 10^{-12}$ s = 20h
 - 64 bits : $2^{64} * 10^{-12}$ s = 214j
 - 128 bits : $2^{128} * 10^{-12}$ s =
10.790.283.070.806.000.000 années
(Note: l'univers a 15.000.000.000 ans)
- 64 est faisable mais 128 ne l'est absolument pas !



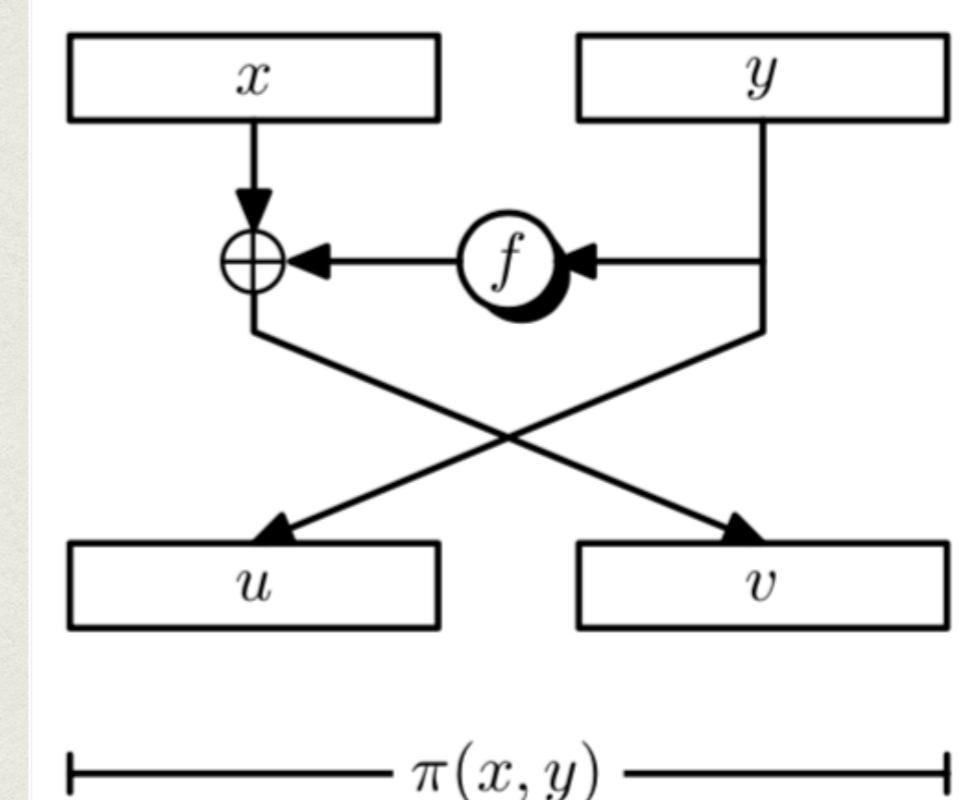
DATA ENCRYPTION STANDARD (DES)

- Développé par IBM en 1975 et adopté comme standard NIST (avec des clés de 56 bits...)
- Chiffrement par blocs de données de 64 bits
- 16 itérations d'une ronde simple : permutation de Feistel
- Expansion de la clé



RONDE DE FEISTEL

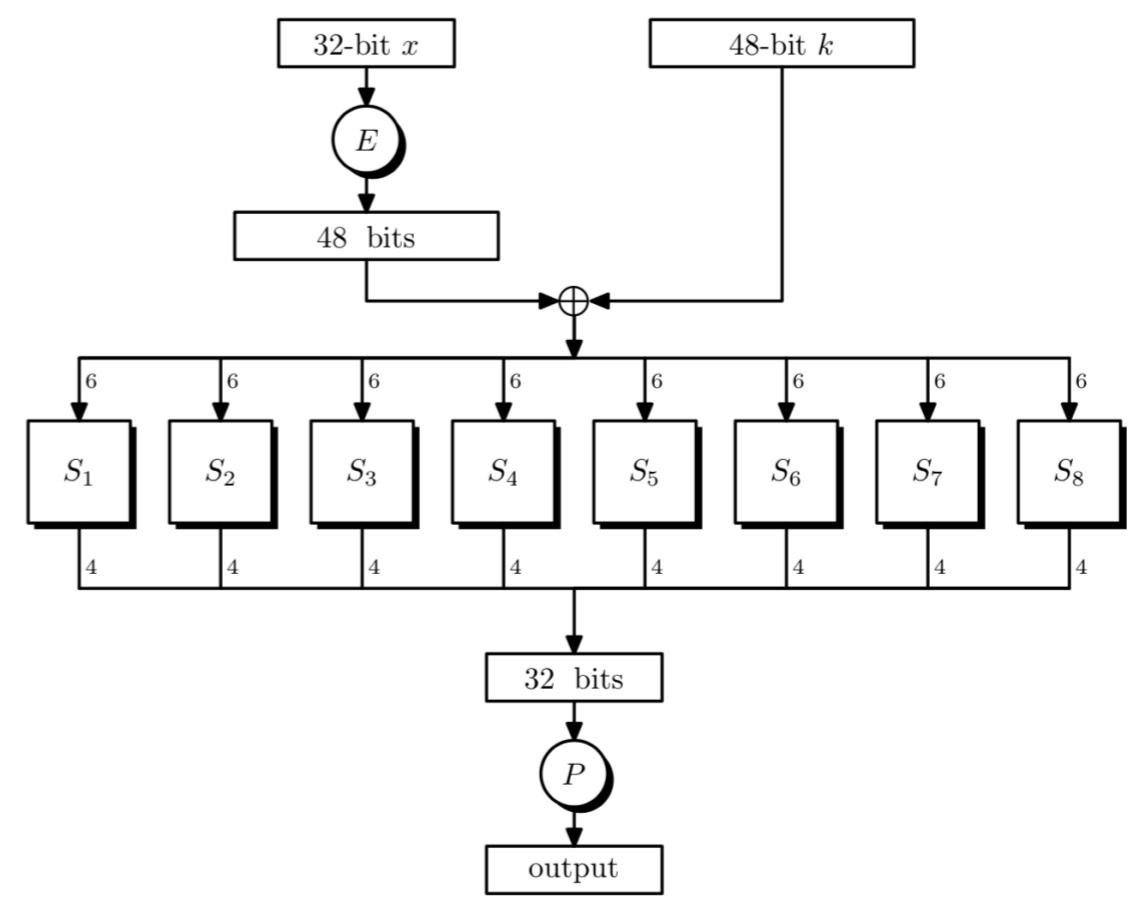
- Permutation aléatoire π à partir d'une fonction arbitraire f
 $\pi(x, y) = (y, x \oplus f(y))$
- Inverse très similaire
 $\pi^{-1}(u, v) = (v \oplus f(u), u)$
-> même hardware



FONCTION D'UNE RONDE DES

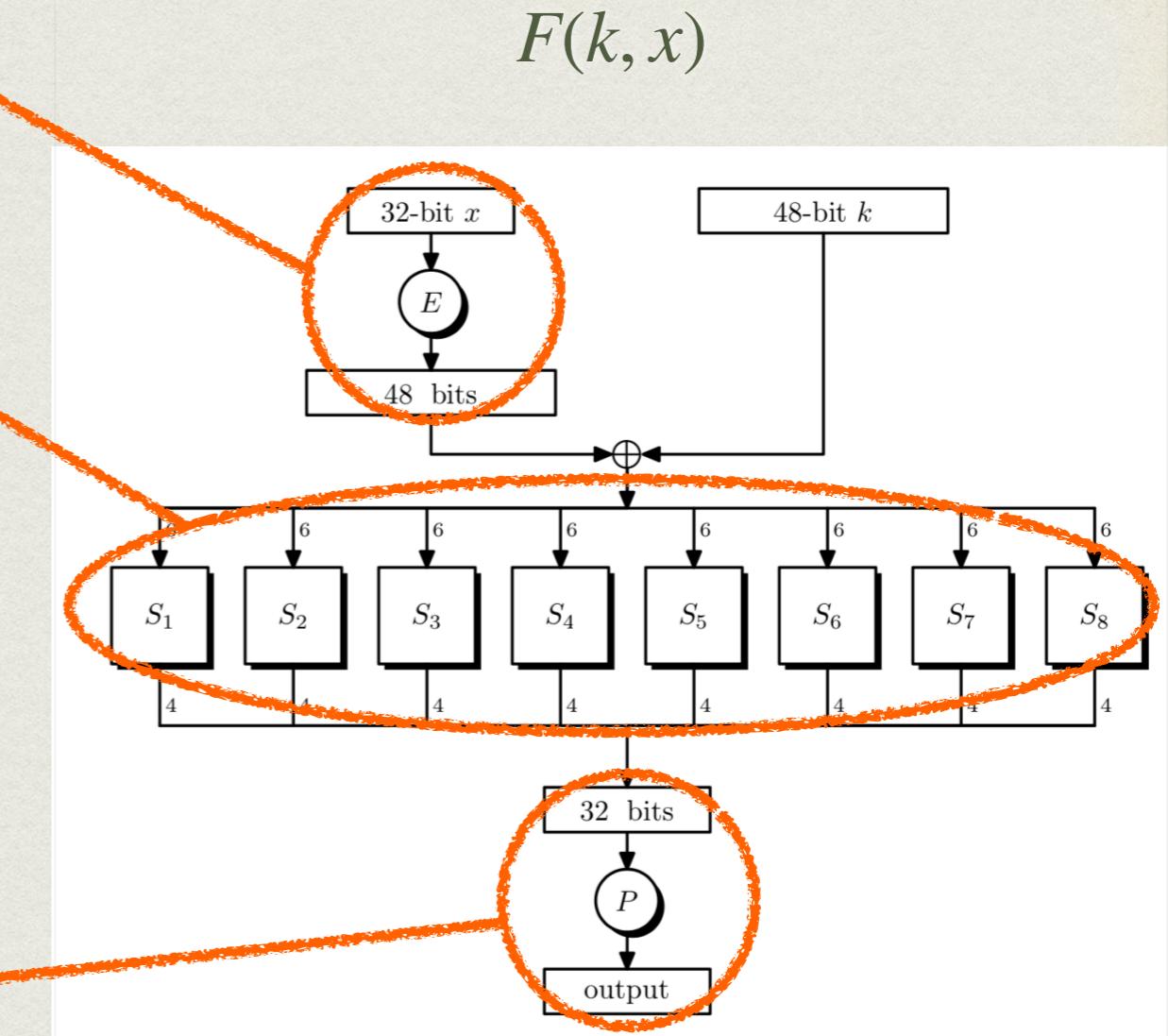
- DES : 16 rondes Feistel
 $f(x) = F(k_i, x)$

$F(k, x)$



FONCTION D'UNE RONDE DES

- Expansion à 48bits par réarrangement/replication
- 8 S-boites 6 \Rightarrow 4bits par Lookup-tables
- DES : 16 rondes Feistel
- $f(x) = F(k_i, x)$
- Permutation 32 \Rightarrow 32



DES COMPLET

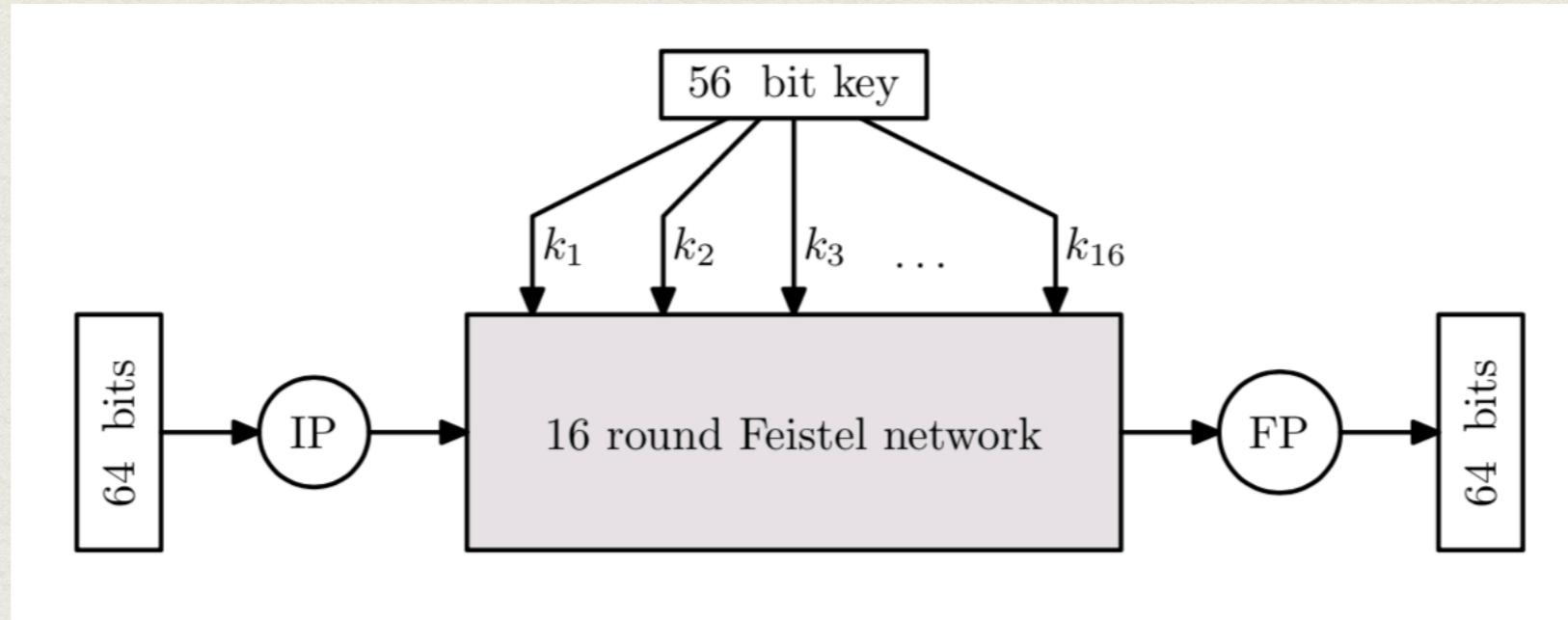


Image : Applied Crypto - D Boneh, V Shoup

- 16 rondes Feistel + permutations initiale et finale
 - XOR + permutations + lookup-tables
 - choix des S-boites critique (dans le standard)
 - Expansion de la clé 56b \implies 16x48 bits : 16 sous-ensembles des 56b

ATTAQUES CONTRE DES

- Cryptanalyse différentielle (Biham-Shamir 91) : faire varier 2^{47} clairs et observer les chiffrés
- Cryptanalyse linéaire (Gilbert-Matsui 93) : 2^{43} couples clair/chiffré arbitraires
- Canaux cachés : par exemple analyse de la consommation électrique
- Recherche exhaustive 56b ... pas si difficile
 - Challenge DES : 3 couples clair/chiffrés retrouver la clé pour 10000\$
jan97: 96j solution distribuée
jan98: 41j solution distribuée
juil98: 56h machine dédiée DeepCrack
jan99: 22h DeepCrack + distribué
- 2007 Machine dédiée COPACOBANA 120 FPGA : 2^{56} clés testées en 12.8j
- 56bits c'est trop peu ! Choisi à dessein ?

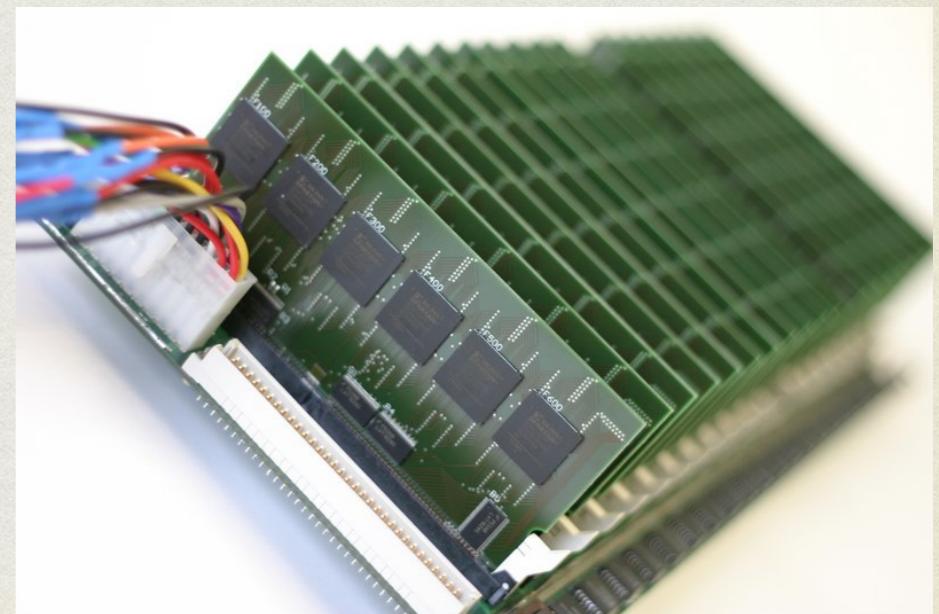


Image : copacobana.org

PATCHES

- 2DES ... non attaque de type rencontre au milieu
- $3DES_{k_1, k_2}(m) = DES_{k_1}(DES_{k_2}^{-1}(DES_{k_1}(m)))$
 - 2 chiffrements et un déchiffrement intermédiaire
 - 112 bits de clé (2x56b)
 - encore populaire dans le monde bancaire
 - mais lent (3x DES) et taille de bloc de 64 bits ...

ADVANCED ENCRYPTION STANDARD (AES)

- 2001 nouveau standard NIST pour remplacer DES
- Schéma de **chiffrement par blocs** similaire à DES basé sur *Rijndael* de Daemen et Rijmen
- Blocs de **128 bits**
- Clés de **128, 192 et 256 bits**
- Simple, efficace en mémoire et en calcul et **réversible**

AES

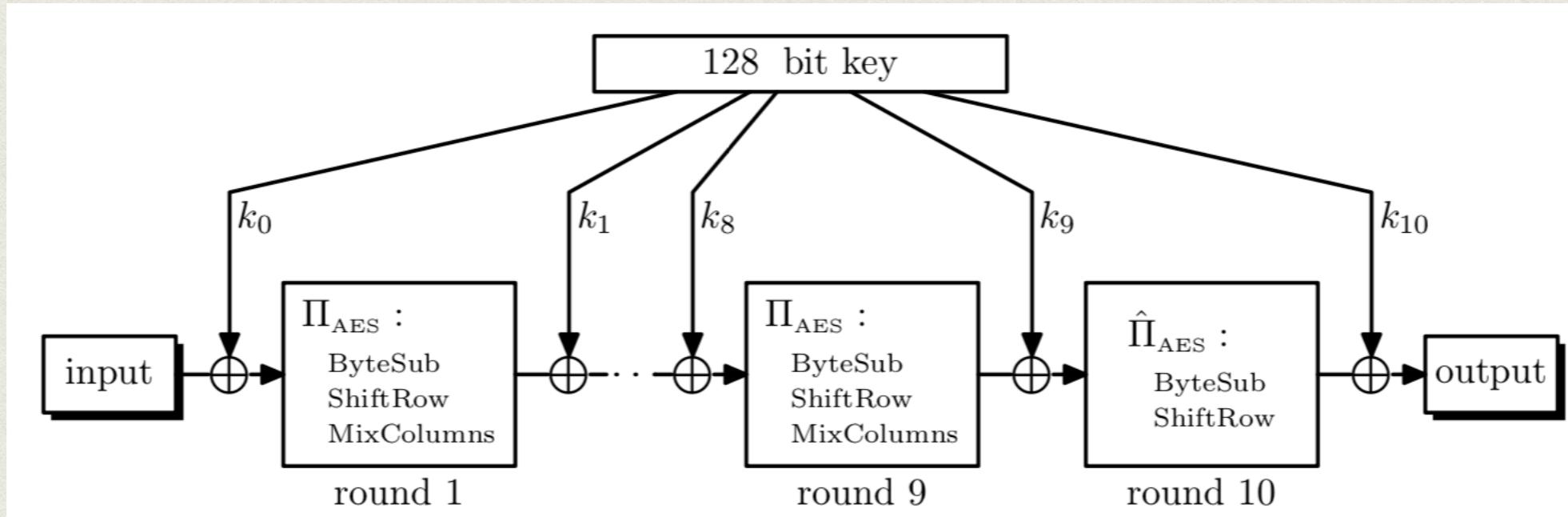
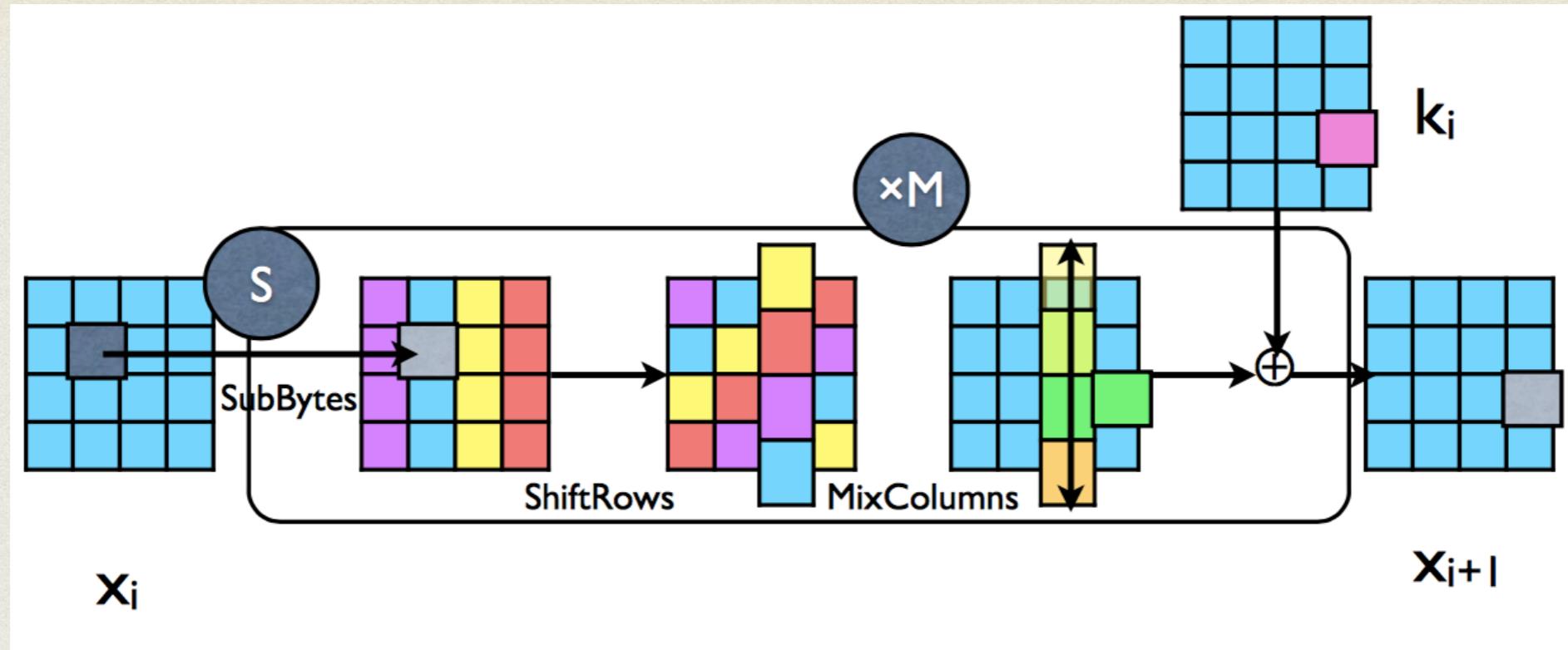


Image : Applied Crypto - D.Boneh,V.Shoup

- Le nombre de rondes dépend de la taille de la clé
 - 128bits : 10 rondes ; 192bits : 12 rondes; 256bits : 14 rondes
- La dernière ronde est différente
- La **clé initiale est étendue** pour donner #rondes clés internes (à partir de rotations, SubBytes et XOR)

RONDE AES



- Bloc de 128bits vus comme une matrice de 4x4 octets
- SubBytes: permutation octet à octet
- ShiftRows: rotation des lignes 2 (1 octet), 3 (2) et 4 (3)
- MixColumns: multiplication par une matrice fixe

MODES DES SCHÉMAS DE CHIFFREMENT PAR BLOC



- Comment chiffrer un **message de taille arbitraire** avec un chiffrement par bloc ?
 - Mode de fonctionnement : comment décomposer les messages ? *padding* ? (comment compléter un bloc non plein?)
 - ECB : Electronic Codebook Mode
 - CBC : Cipher Block Chaining Mode
 - CTR : Counter Mode

ELECTRONIC CODEBOOK (ECB)

- On découpe le clair m en blocs de 64 (ou 128) bits $m = (m_0, m_1, \dots, m_{n-1})$
- On chiffre chaque m_i et on traite indépendamment $c = (ENC(m_0, k), \dots, ENC(m_{n-1}, k))$
- Parallélisable mais déterministe (si $m_i = m_j$ alors $c_i = c_j$)

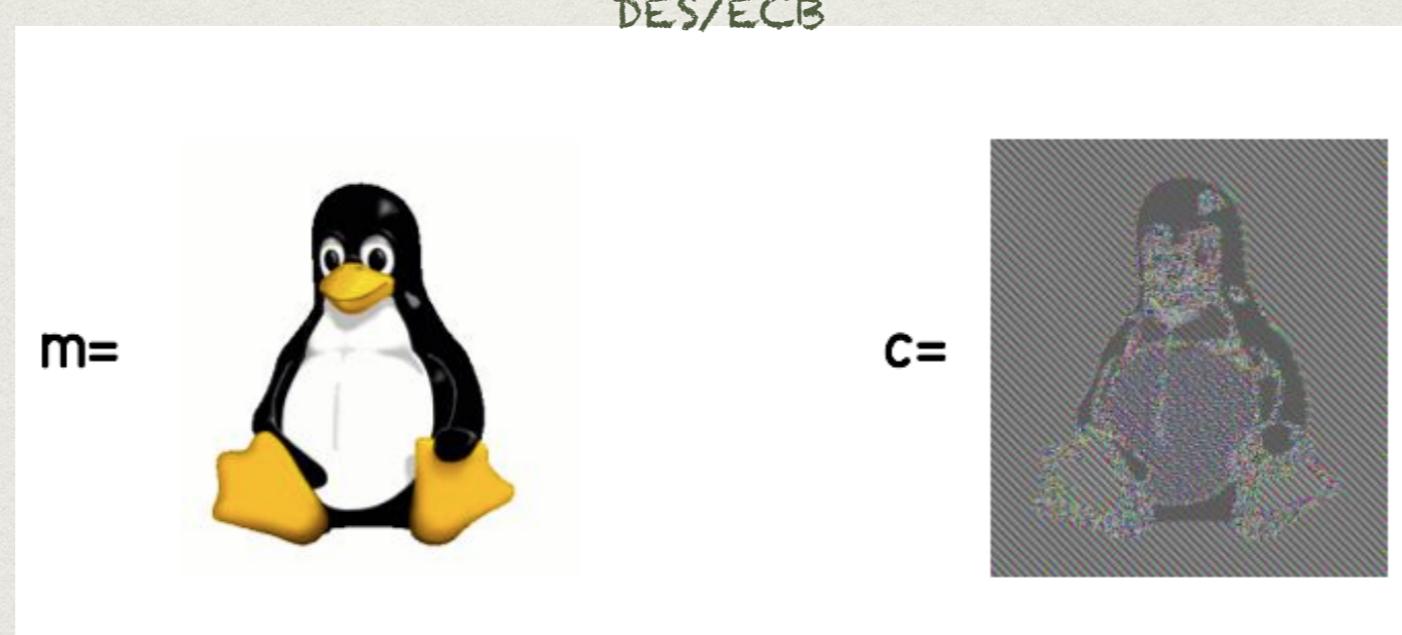
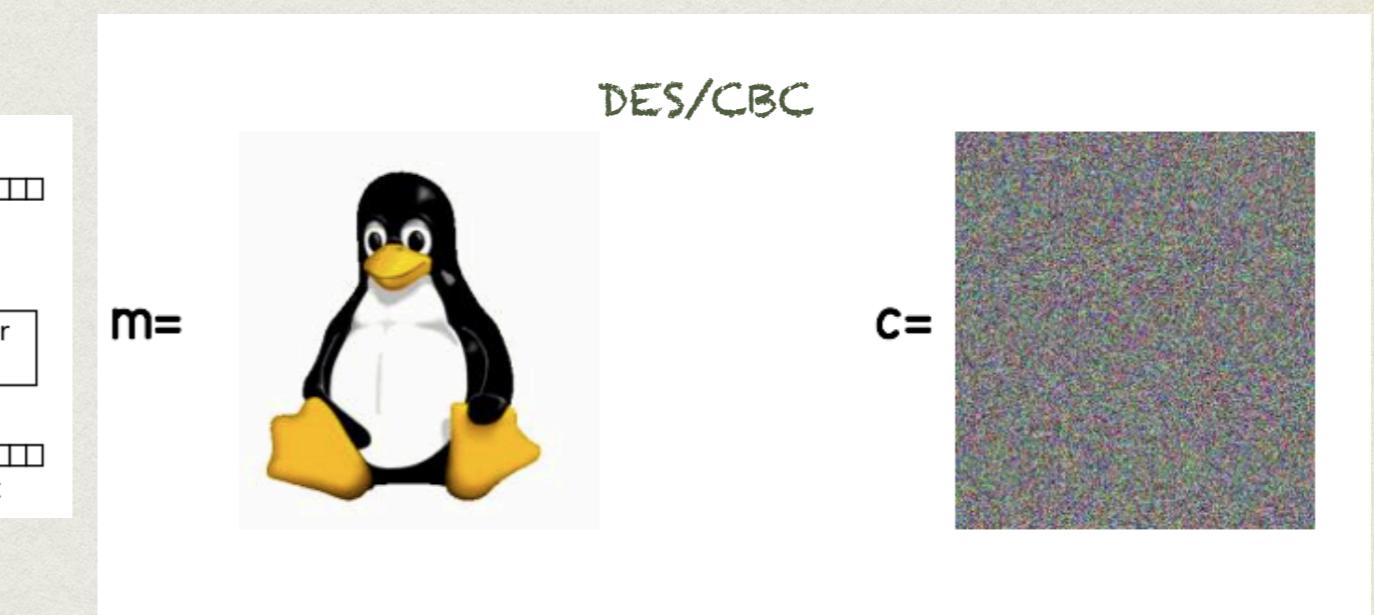
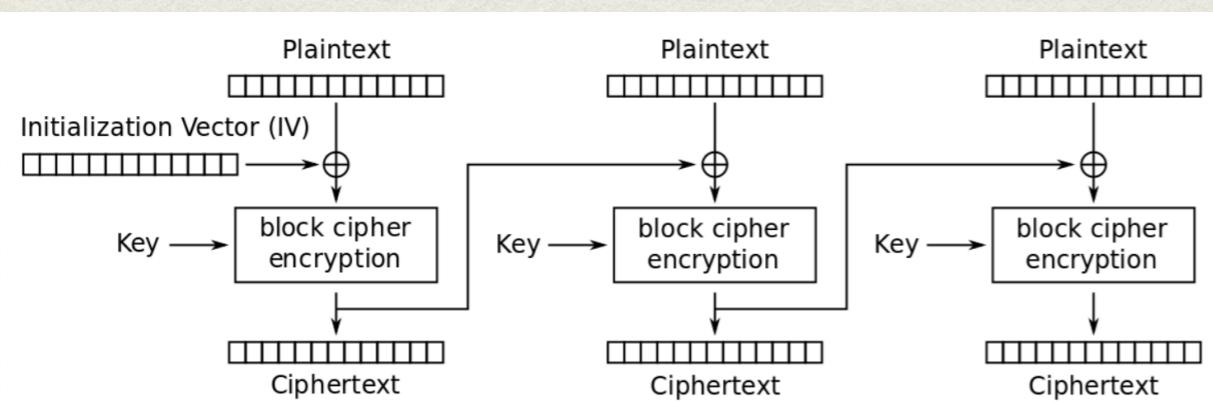


Image : Wikimedia

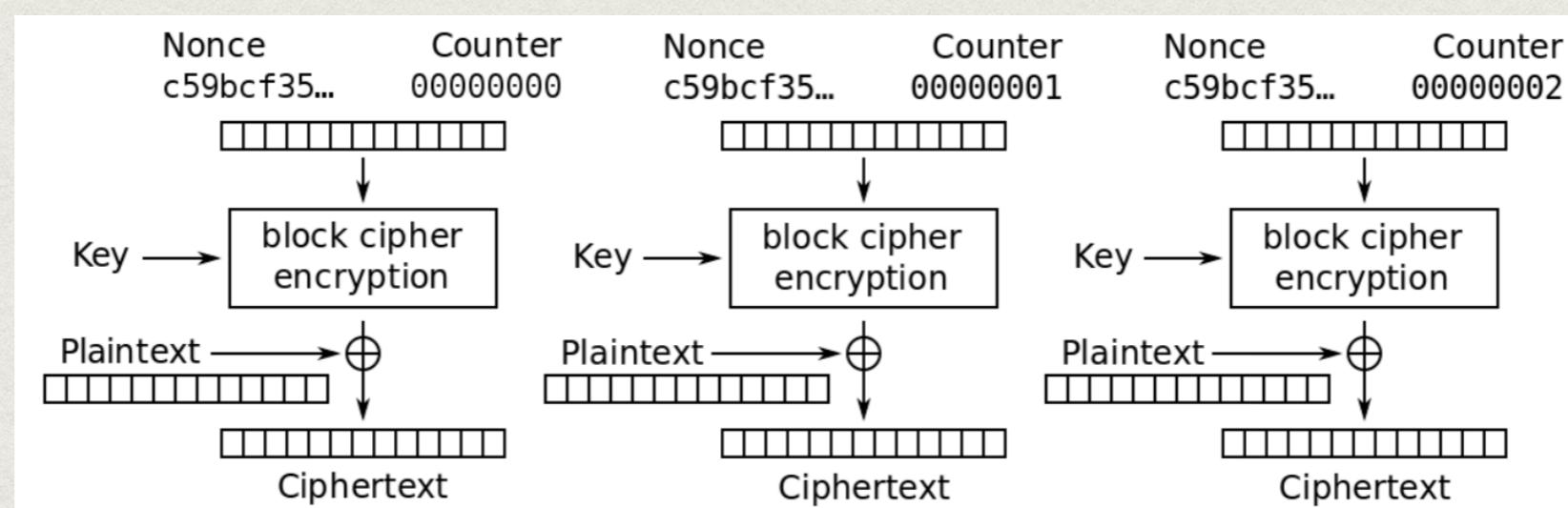
CIPHER BLOCK CHAINING (CBC)

- On découpe le clair m en blocs de 64 (ou 128) bits $m = (m_0, m_1, \dots, m_{n-1})$
- Chaque entrée est XORée, la première avec un testeur d'initialisation IV qui doit être aléatoire pour sécurité sémantique, les suivantes avec c_{i-1}
 $c_i = ENC(c_{i-1} \oplus m_i, k)$ avec $c_{-1} = IV$
- Probabiliste mais non parallélisable



COUNTER (CTR)

- On découpe le clair m en blocs de 64 (ou 128) bits $m = (m_0, m_1, \dots, m_{n-1})$
- On utilise un nombre aléatoire (*nonce*) concaténé (\odot) à un compteur
- On chiffre ce *nonce* et on XOR au m_i :
 $c_i = m_i \oplus \text{ENC}(\text{nonce} \odot i, k)$
- Probabiliste **et** parallélisable



Images : Wikimedia

CHIFFREMENT PAR FLUX

- Pour des messages de longueur arbitraire
- Masques jetables mais avec une chaîne pseudo-aléatoire pour clé
- Algorithme générique
 1. Flux de clé généré à partir de la clé et d'un IV
$$G_k(IV) = r_0r_1r_2r_3 \dots$$
 2. Le message est XORé au flux de clé
$$\forall \text{ bit } i : c_i = m_i \oplus r_i$$
 3. Le chiffré transmis inclut IV
$$c = (IV, c_0, \dots, c_{n-1})$$

SÉCURITÉ D'UN CHIFFREMENT PAR FLUX

SÉCURITÉ D'UN CHIFFREMENT PAR FLUX

- Dépend fortement du générateur pseudo-aléatoire
 - **Indistinguabilité** : propriété selon laquelle il doit être *difficile* de distinguer $G_k(IV)$ d'une séquence aléatoire
 - **Malléabilité** :
un ajout/perte d'un bit du chiffré rend le déchiffrement difficile/ impossible
une inversion de bit du chiffré inverse le clair
un adversaire peut modifier le clair déchiffré sans être détecté
- ex: $c = m \oplus G(s)$ modifié en $c' = c \oplus \Delta$
déchiffré en $m' = c' \oplus G(s) = m \oplus \Delta$
- Vérifier en exercice
- $m = \text{Alice}$ $m' = \text{Molly}$
 $\Delta = \text{Alice} \oplus \text{Molly}$

INSÉCURITÉ D'UN CHIFFREMENT PAR FLOTS

- Exemple de RC4 : « Masques jetables » avec une chaîne pseudo aléatoire générée par un PRG (Pseudo-Random-Generator) donné
- Très utilisé : 802.11b WEP, WPA, SSL, Windows NT PPTP, etc.
 - NT: Two-time-pad - la clé était réutilisée ...
 - PRG faible : sous certaines conditions on peut deviner le 3e octet de la clé ...
 - PRG faible 2 : il existe une corrélation entre certains bits de clé
- Conclusion 1 : **ne plus utiliser RC4**
- Conclusion 2 : faire de **bons PRG** est *1-essentiel et 2-difficile !*
bon=secure **et** rapide

EN CONCLUSION

- Un bon chiffrement par blocs + un mode approprié + un bon IV (un bon PRG)
- Chiffrement symétrique assez efficace (1-10Mo/s PC std)
- Plus on utilise une même clé, plus la sécurité se détériore
 - Ne pas utiliser une même clé trop longtemps
 - Générer et échanger de nouvelles clés (rencontre, canal aux., crypto. asymétrique, etc.)
- **ET SURTOUT NE JAMAIS
IMPLÉMENTER SA PROPRE CRYPTO !!!**

