# Sécurité des Systèmes d'Information


| | |
|---|---|
| Eric Alata | eric.alata@laas.fr |
| Yves Deswarte | yves.deswarte@laas.fr |
| Vincent Nicomette | vincent.nicomette@laas.fr |
| Benoît Morgan | benoit.morgan@enseeiht.fr |

INSA de Toulouse - ENSEEIHT


6 octobre 2020

## Cours magistraux

| S. | Description |
|---|---|
| 3 | Introduction sécurité des systèmes d'information |
| 2 | Cryptologie |
| 2 | Sécurité du logiciel + rappels d'assembleur `x86` |
| 2 | Sécurité réseau |

## Travaux pratiques

| S. | Description |
|---|---|
| 2 | Conception de *shellcodes* 64-bit et dépassement de tampons |
| 2 | Filtrage réseau avec `netfilter` et `iptables` |
| 2 | Isolation réseau `IPSEC` (racoon + setkey) |

## Examen

Écrit d'1h30

### Système d'information

Un système d'information est l'ensemble des éléments participant au traitement, à la gestion et à la transmission d'informations entre les membres d'une communauté.

### Sécurité

La sécurité des systèmes d'information est l'ensemble des moyens permettant d'assurer les propriétés de confidentialité, d'intégrité et de disponibilité des informations.

Un peu d'histoire

Les propriétés de la sécurité

Les attaques

Les défenses

La protection des systèmes informatiques

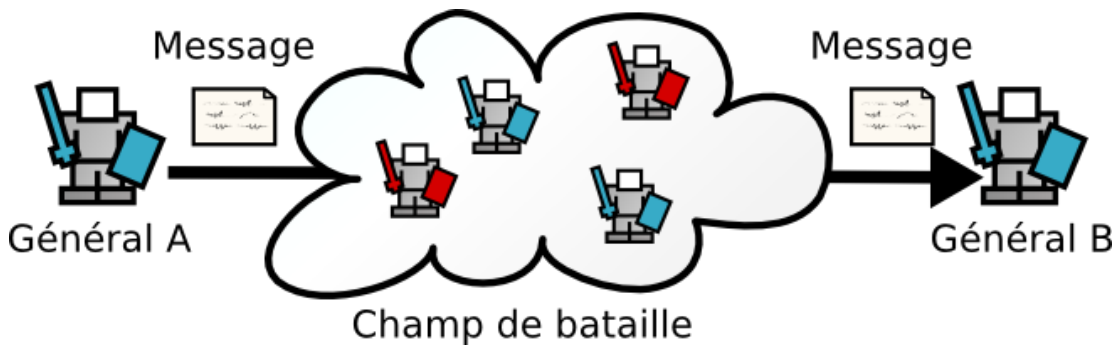
# Sommaire

# Sommaire

# Sommaire

# La scytale – v$^{e}$ *siècle avant N.E.*

- Les spartiates, contexte militaire
- Besoin en communication durant la bataille
  - Permettre aux généraux d'échanger des messages à travers le champ de bataille
  - Empêcher les ennemis de lire le contenu des messages



# La scytale – v$^{e}$ *siècle avant N.E.*

- Description dans une œuvre de Plutarque[9]

Je dois dire ce que c'est que la scytale. Quand un général part pour une expédition de terre ou de mer, les éphores prennent deux bâtons ronds, d'une longueur et d'une grandeur si parfaitement égales, qu'ils s'appliquent l'un à l'autre sans laisser entre eux le moindre vide. Ils

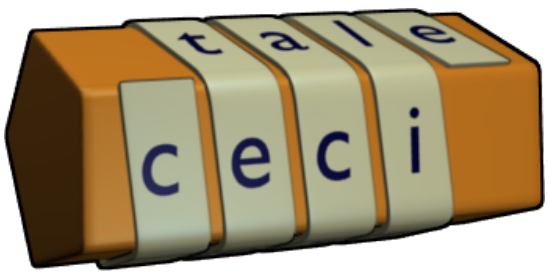


gardent l'un de ces bâtons et donnent l'autre au général ; ils appellent ces bâtons scytales. Lorsqu'ils ont quelque secret important à faire passer au général, ils prennent une bande de parchemin, longue et étroite comme une courroie, la roulent autour de la scytale qu'ils ont gardée, sans y laisser le moindre intervalle, en sorte que la surface du bâton est entièrement couverte. Ils écrivent ce qu'ils veulent sur cette bande ainsi roulée, après quoi ils la déroulent, et l'envoient au général sans le bâton. Quand celui-ci la reçoit, il ne peut rien lire, parceque les mots tous séparés et épars, ne forment aucune suite. Il prend donc la scytale qu'il a emportée, et roule autour la bande de parchemin, dont les différents tours, se trouvant alors réunis, remettent les mots dans l'ordre où ils ont été écrits, et présentent toute la suite de la lettre. On appelle cette lettre scytale, du nom même du bâton, comme ce qui est mesuré prend le nom de ce qui lui sert de mesure.

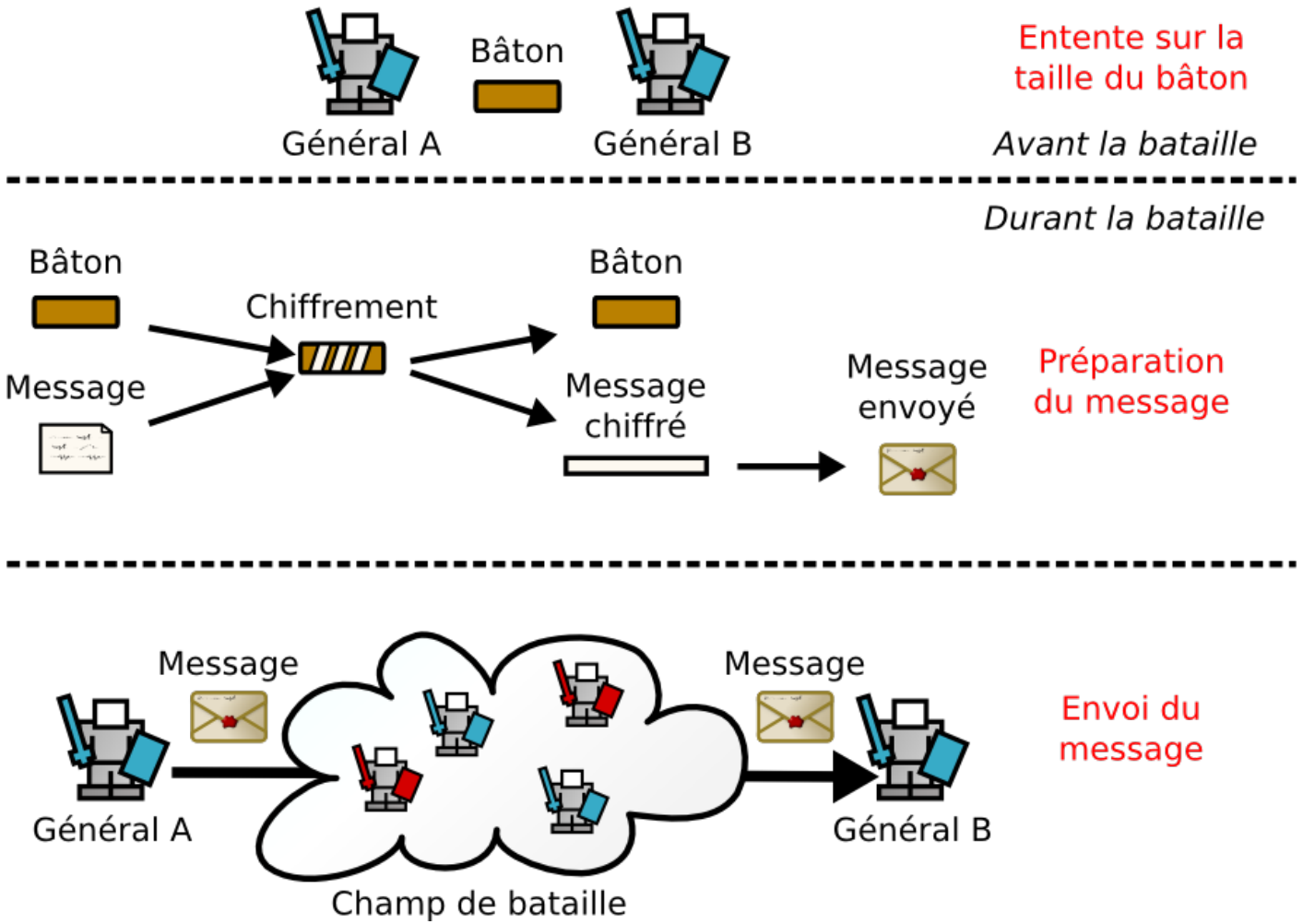
# La scytale – v$^{e}$ *siècle avant N.E.*

- Dispositif de chiffrement par transposition
  - Bâton ~ clef
  - Lanière de cuir ~ support du message
- Exemple avec une scytale à 5 caractères par contour

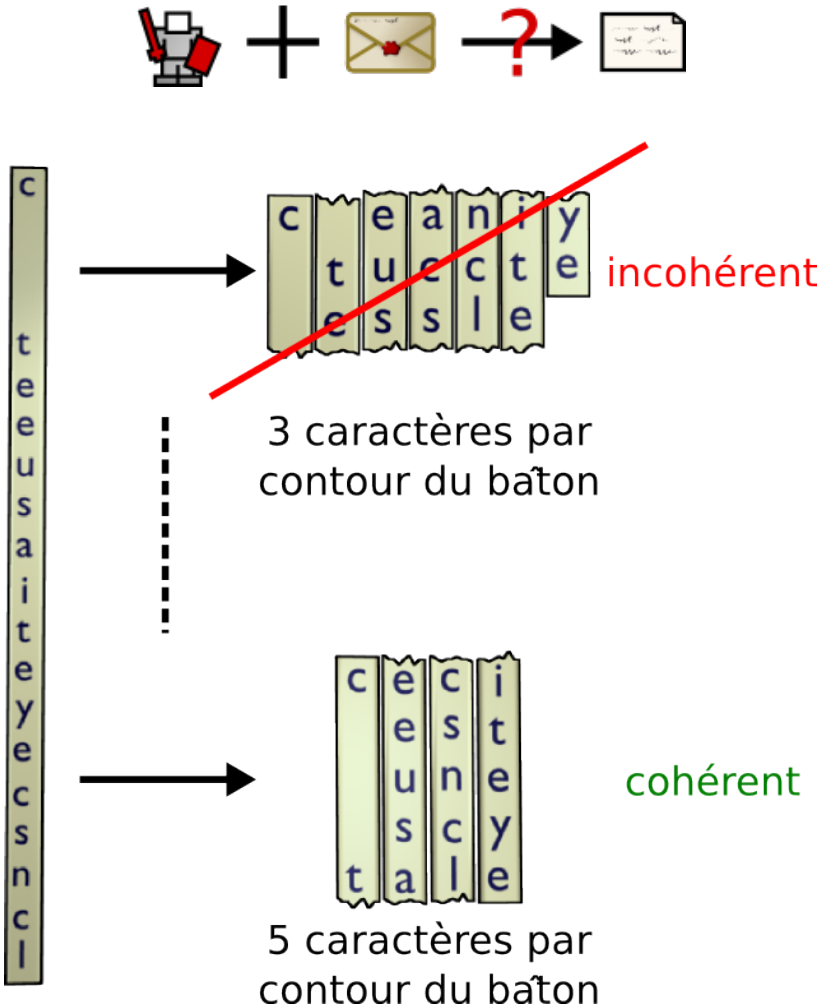
| Message clair | `ceci est une scytale` |
|---|---|
| Message chiffré | `c teeusacsncliteye` |




# La scytale – v$^{e}$ *siècle avant N.E.*



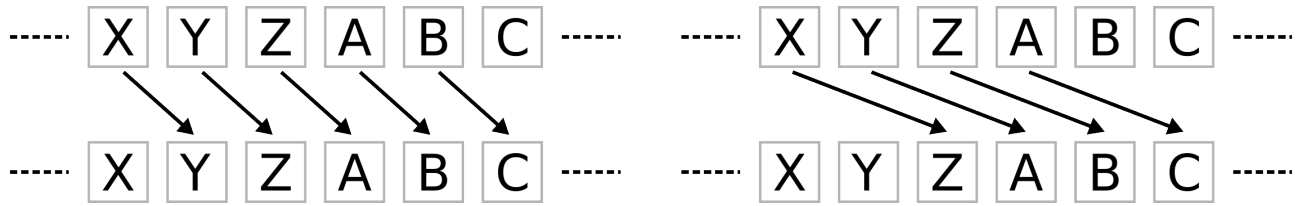
# La scytale – v$^{e}$ *siècle avant N.E.*



# Sommaire

# Le chiffre de César – i$^{e}$ *siècle avant N.E.*

- Décalage de chaque lettre du message clair d'une distance fixe




- Soient $n$ la distance de décalage, $x$ la lettre à coder/décoder

$$\begin{array}{ll} \text{Codage} & C_n(x) = (x + n) \mod 26 \\ \text{Décodage} & D_n(x) = (x - n) \mod 26 \end{array}$$

- Exemple

| Message clair | `chiffre de cesar` |
|---|---|
| Distance 1 | `dijggsf ef dftbs` |
| Distance 2 | `ejkhhtg fg eguct` |

- Utilisé par Jules César lors de la Guerre des Gaules avec $n = 3$

# Le chiffre de César – i$^{e}$ *siècle avant N.E.*

- Analyse fréquentielle possible (fonction de la langue)
  - Certaines lettres sont plus employées que d'autres
  - La fréquence d'une lettre dans un message égale la fréquence de son image dans le message chiffré

$$f(x, M) = f(C_n(x), C_n(M))$$

  - Liste des lettres alphabétiques de la plus fréquente à la moins fréquente dans un texte français

`EAISTNRULODMPCVQGBFJHZXYKW`

  - Exemple

`atnqf zs fzywj hmnkkwj ij hjxfw xn kfhnqj f hfxxjw yjm !`

# Le chiffre de César – i$^{e}$ *siècle avant N.E.*

- Analyse fréquentielle possible (fonction de la langue)
  - Certaines lettres sont plus employées que d'autres
  - La fréquence d'une lettre dans un message égale la fréquence de son image dans le message chiffré

$$f(x, M) = f(C_n(x), C_n(M))$$

  - Liste des lettres alphabétiques de la plus fréquente à la moins fréquente dans un texte français

`EAISTNRULODMPCVQGBFJHZXYKW`

  - Exemple

```
atnqf zs fzywj hmnkkwj ij hjxfw xn kfhnqj f hfxxjw yjm !
voila un autre chiffre de cesar si facile a casser teh !
```

# Sommaire

# Le chiffre de Vigenère – xvi$^{e}$ *siècle après N.E.*

- Blaise de Vigenère (1523 – 1596), diplomate francais
- Amélioration du chiffre de César ⇒ substitution polyalphabétique
  - Une lettre de l'alphabet peut être chiffrée de plusieurs manières différentes
  - La clé est représentée par une chaîne de caractères
  - Un caractère de la clef = une distance

    A=0 B=1 C=2 . . .
  - Répétition de la clé, si nécessaire
- Exemple

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Message clair* | c | h | i | f | f | r | e | d | e | v | i | g | e | n | e | r | e |
| *Clé* | u | n | e | c | l | e | u | n | e | c | l | e | u | n | e | c | l |
| *Décalage* | 20 | 13 | 4 | 2 | 11 | 4 | 20 | 13 | 4 | 2 | 11 | 4 | 20 | 13 | 4 | 2 | 11 |
| *Message chiffré* | w | u | m | h | q | v | y | q | i | x | t | k | y | a | i | t | p |

# Le chiffre de Vigenère – xvi$^{e}$ *siècle après N.E.*

Comment obtenir le message clair avec, seulement, le message chiffré ?

*Message chiffré* esmhqvgxipeexpgnpgjtjhcifpzkrippvg

- Trois motifs : vg ($\Delta = 27$), ip ($\Delta = 21$) et pg ($\Delta = 3$)
- La distance entre les répétitions d'un motif est multiple de 3

$\Rightarrow$ La clé a vraissemblablement une taille de 3

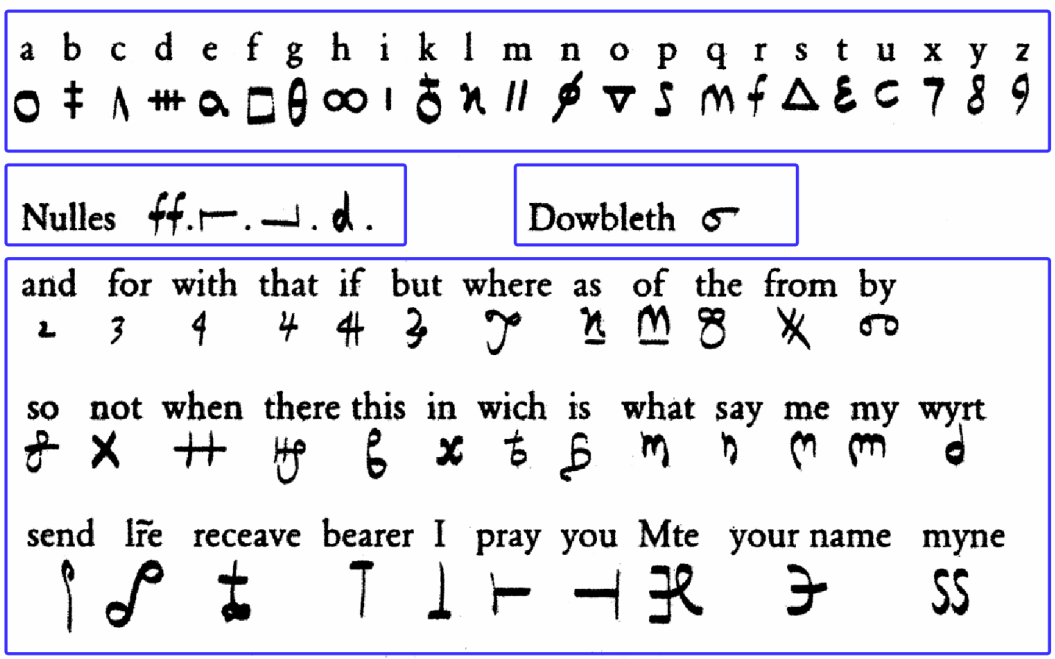
| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Analyse fréquentielle* | e | h | g | p | x | n | j | h | f | k | p | g |
| | s | q | x | e | p | p | t | c | p | r | p | |
| | m | v | i | e | g | g | j | i | z | i | v | |

*Message clair* chiffrementaveclechiffredevigenere
*Clé* cleclecleclecleclecleclecleclecleclec
*Message chiffré* esmhqvgxipeexpgnpgjtjhcifpzkrippvg

# Sommaire

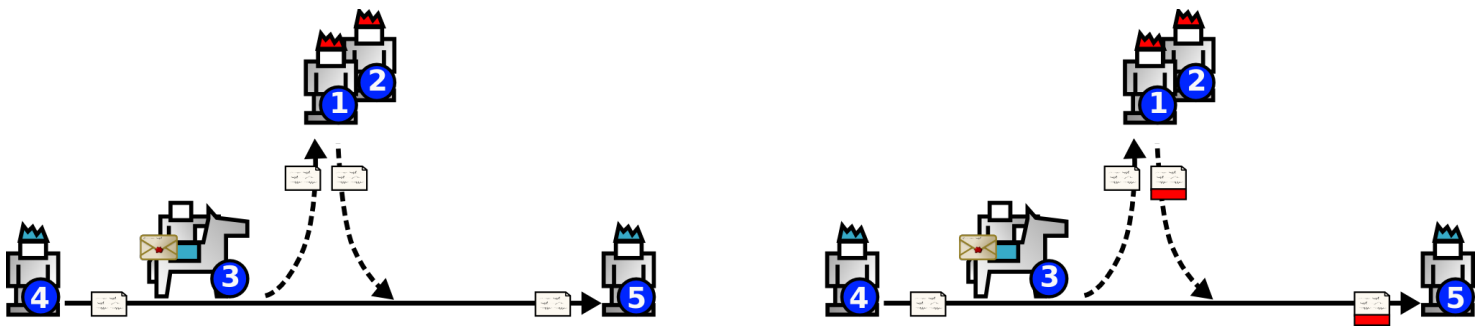
# Le chiffre de Marie Stuart – xvi$^{e}$ *siècle après N.E.*

- Tentative d'assassinat de la reine d'Angleterre
  - Marie Stuart (en prison) et ses complices, conspirateurs
- Chiffrement des correspondances avec une nomenclature
  - 1 symbole pour chaque lettres de l'alphabet (sauf j, v et w)
  - 36 symboles pour les mots usuels
  - 4 symboles nuls
  - 1 symbole indiquant la répétition du symbole suivant



# Le chiffre de Marie Stuart – xvi$^{e}$ *siècle après N.E.*

- Complot déjoué[3][7]
  1. Francis Walsingham : maître-espion de la reine d'Angleterre
  2. Thomas Phelippes : chiffreur, déchiffreur du maître-espion
  3. Gilbert Gifford : messager de Marie Stuart et agent double
  4. Marie Stuart : comploteur
  5. Anthony Babington : comploteur
- Comment identifier tous les comploteurs ?

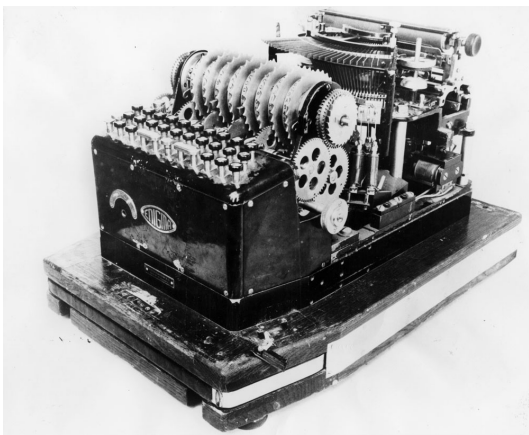



*"Francis Walsingham est l'homme au millieu"*

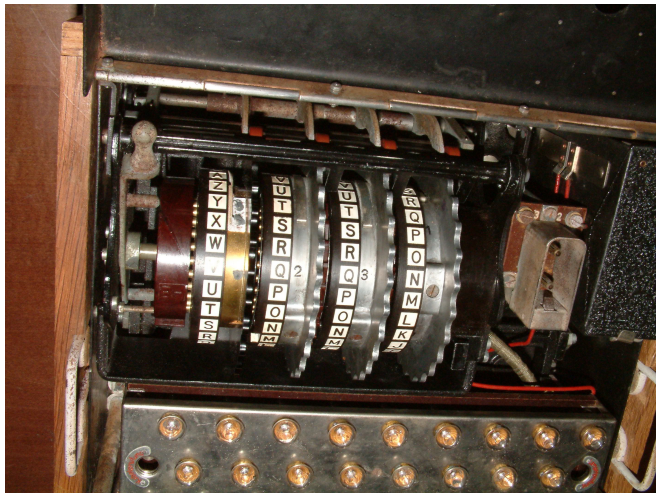
# Sommaire

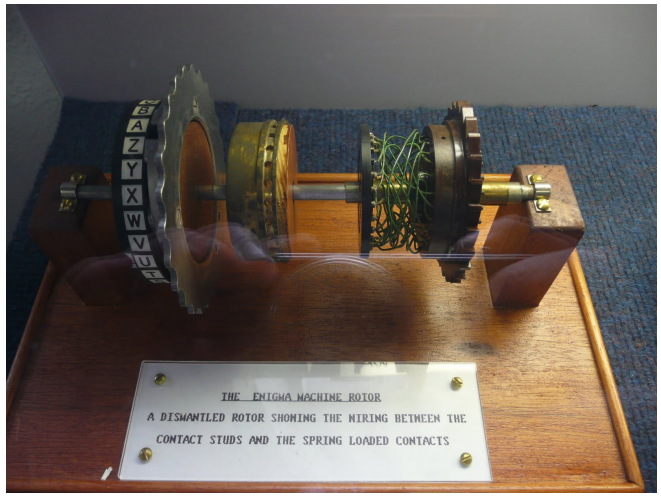
# La machine Enigma – $xx^e$ *siècle après N.E.*

- Arthur Scherbius (1878 – 1929), ingénieur en électricité allemand
- Machine électromécanique inventée en 1919, à but commercial
- Employée par l'armée allemande (U-Bot)
- Automatisation du chiffrement par substitution
  - Un clavier de 26 lettres
  - Un cadran lumineux de 26 lettres
  - Des rotors (en général 3)
  - Un réflecteur



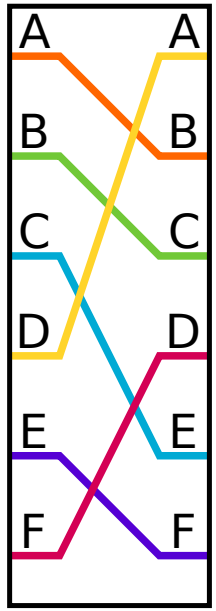
# La machine Enigma – $xx^e$ *siècle après N.E.*



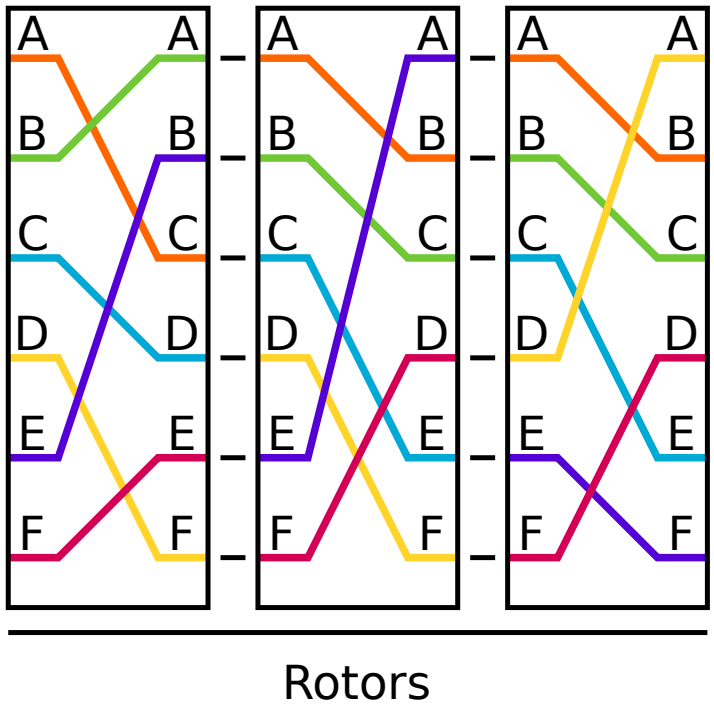



- Clé de chiffrement
  - Disposition et orientation des rotors
  - Connexions entre lettres de l'alphabet : tableau de permutations
- A chaque pression d'une touche
  - Une des ampoules du cadran lumineux s'allume
  - Le rotor le plus à droite pivote
  - A chaque tour complet d'un rotor, le rotor à sa gauche pivote
- La bombe de Turing : découverte de la clé en 1 heure

# La machine Enigma – $xx^e$ *siècle après N.E.*– Fonctionnement

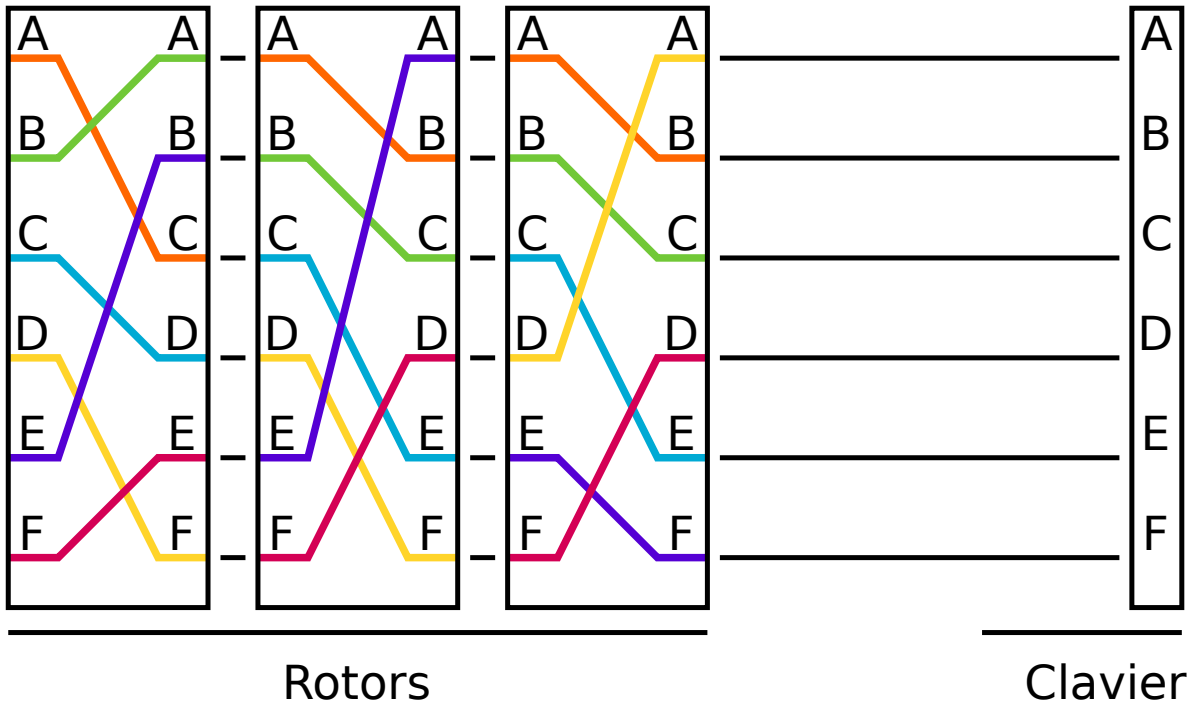


# La machine Enigma – $xx^e$ *siècle après N.E.*– Fonctionnement



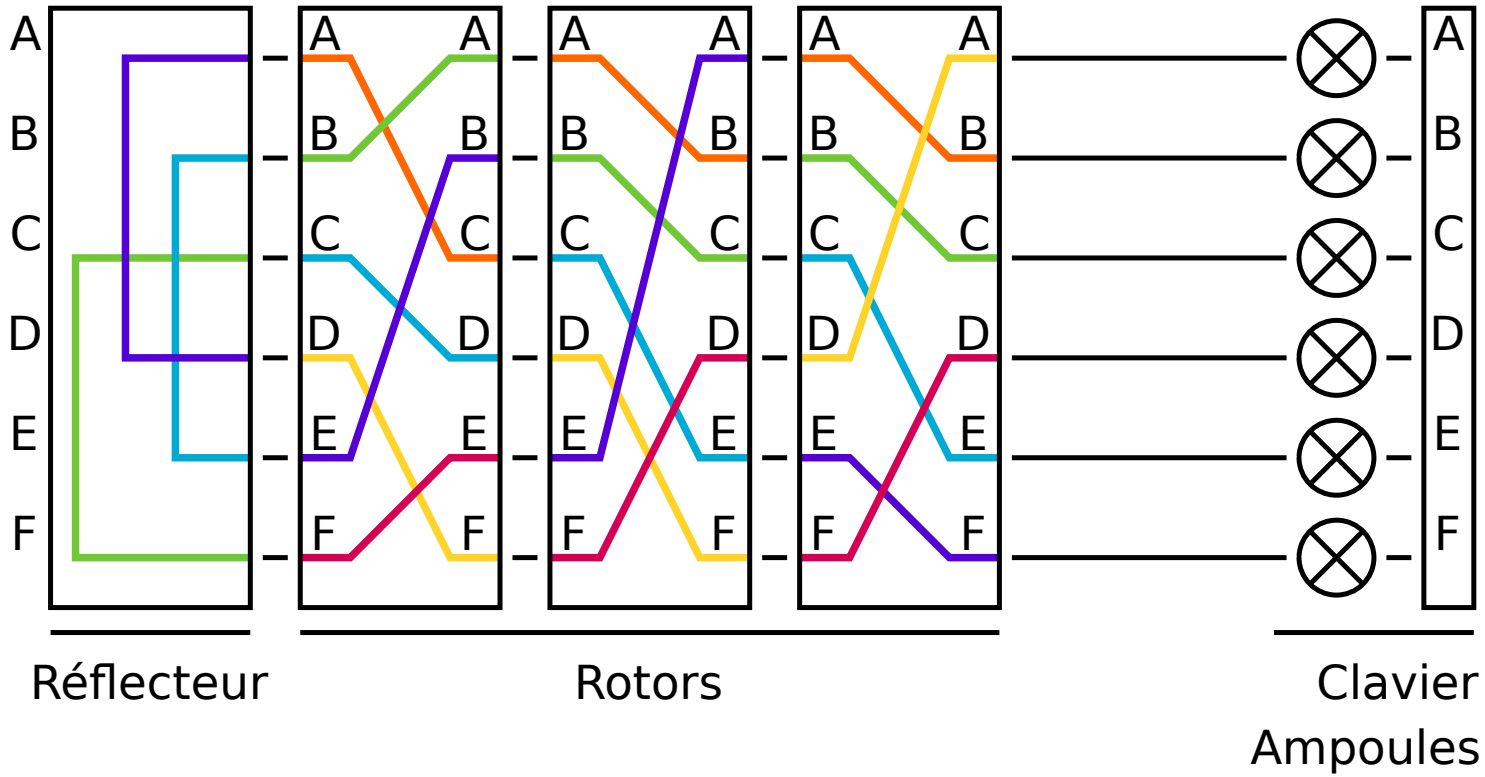

- Les rotors sont interconnectés

# La machine Enigma – $xx^e$ *siècle après N.E.*– Fonctionnement



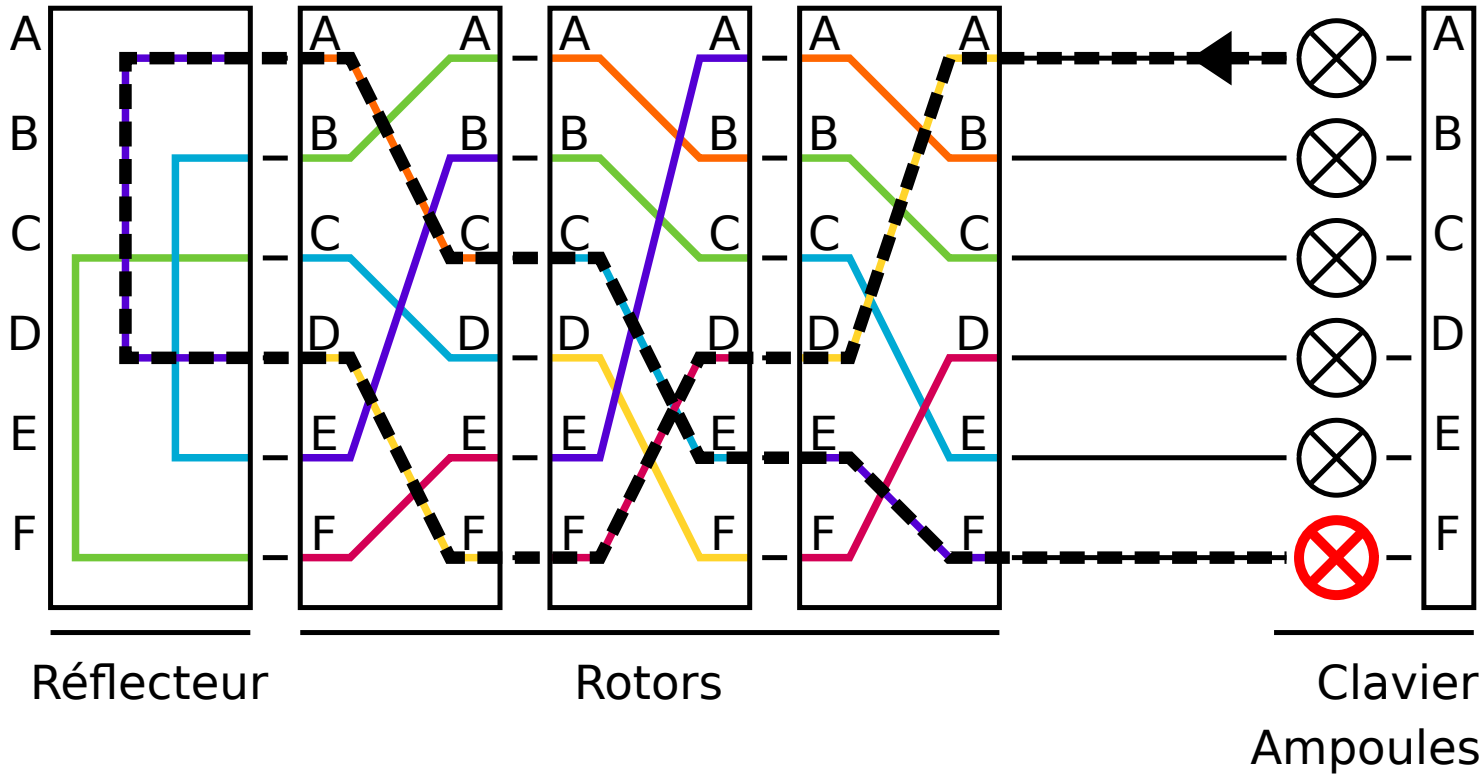

- Les rotors sont interconnectés
- Le circuit électrique est fermé à l'aide d'un clavier

# La machine Enigma – $xx^e$ *siècle après N.E.*– Fonctionnement



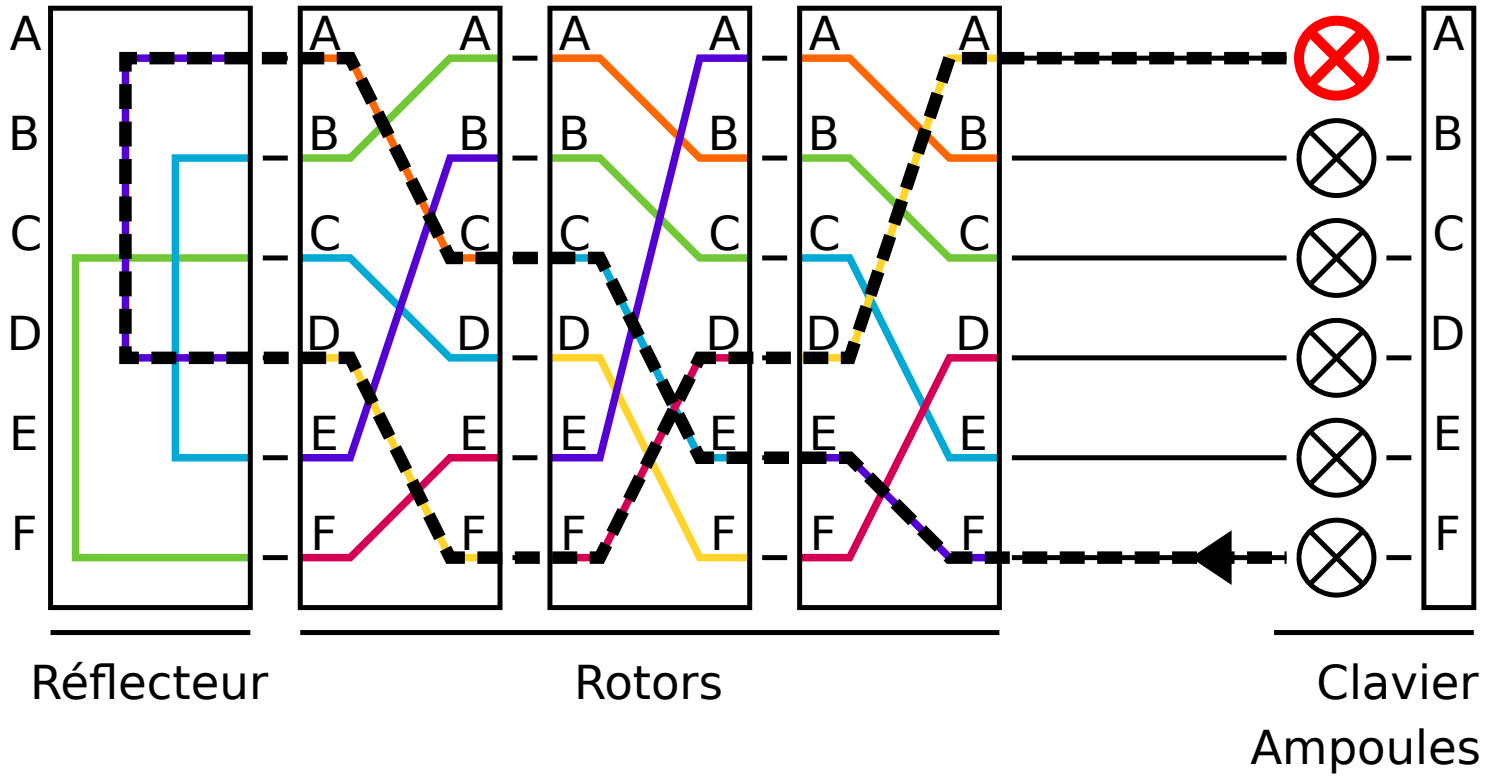

- Les rotors sont interconnectés
- Le circuit électrique est fermé à l'aide d'un clavier
- Le réflecteur rend le systèmes symétrique
- Lecture du chiffré se fait à l'aide de l'ampoule allumée

# La machine Enigma – $xx^{e}$ *siècle après N.E.*– Fonctionnement



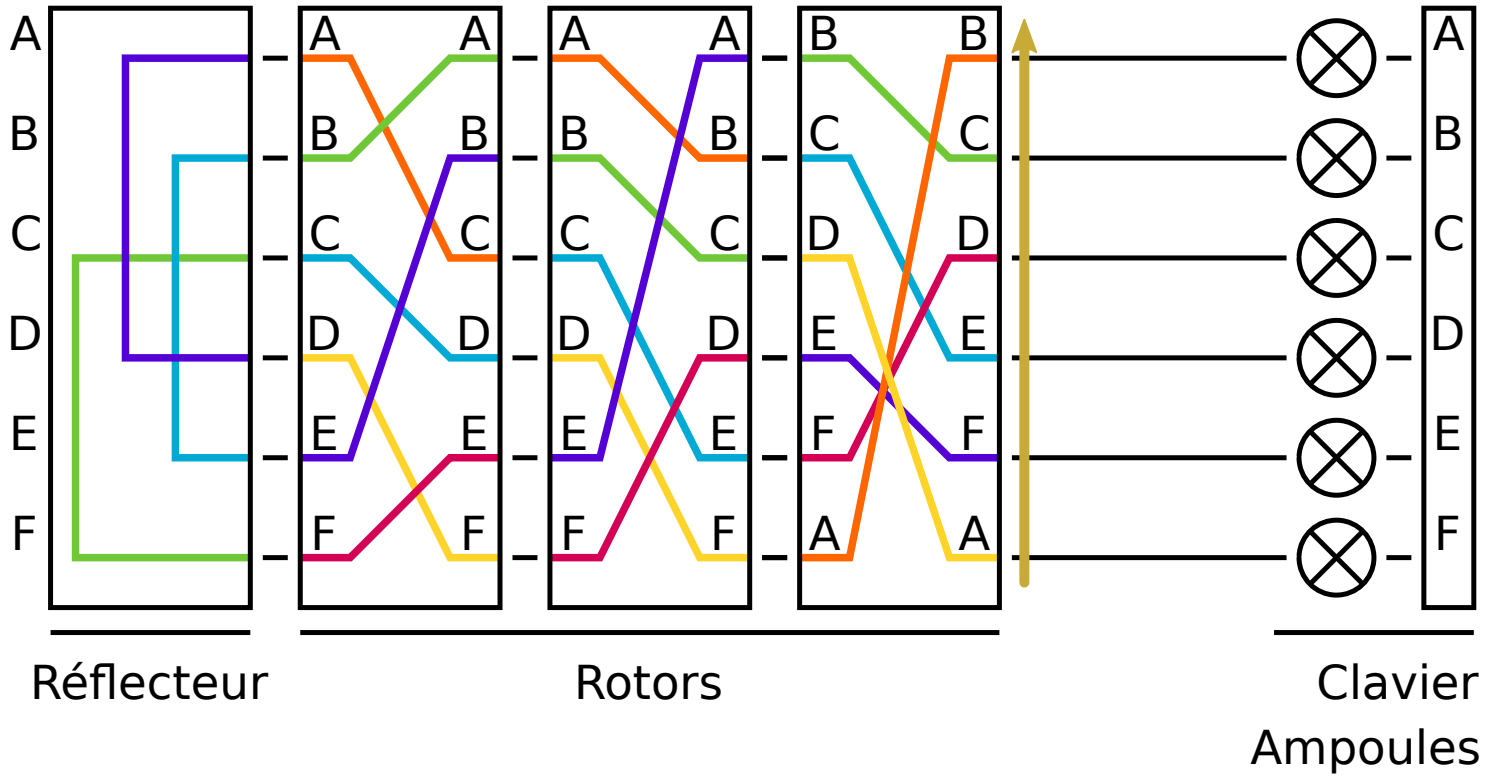

- Les rotors sont interconnectés
- Le circuit électrique est fermé à l'aide d'un clavier
- Le réflecteur rend le systèmes symétrique
- Lecture du chiffré se fait à l'aide de l'ampoule allumée

# La machine Enigma – $xx^e$ *siècle après N.E.*– Fonctionnement



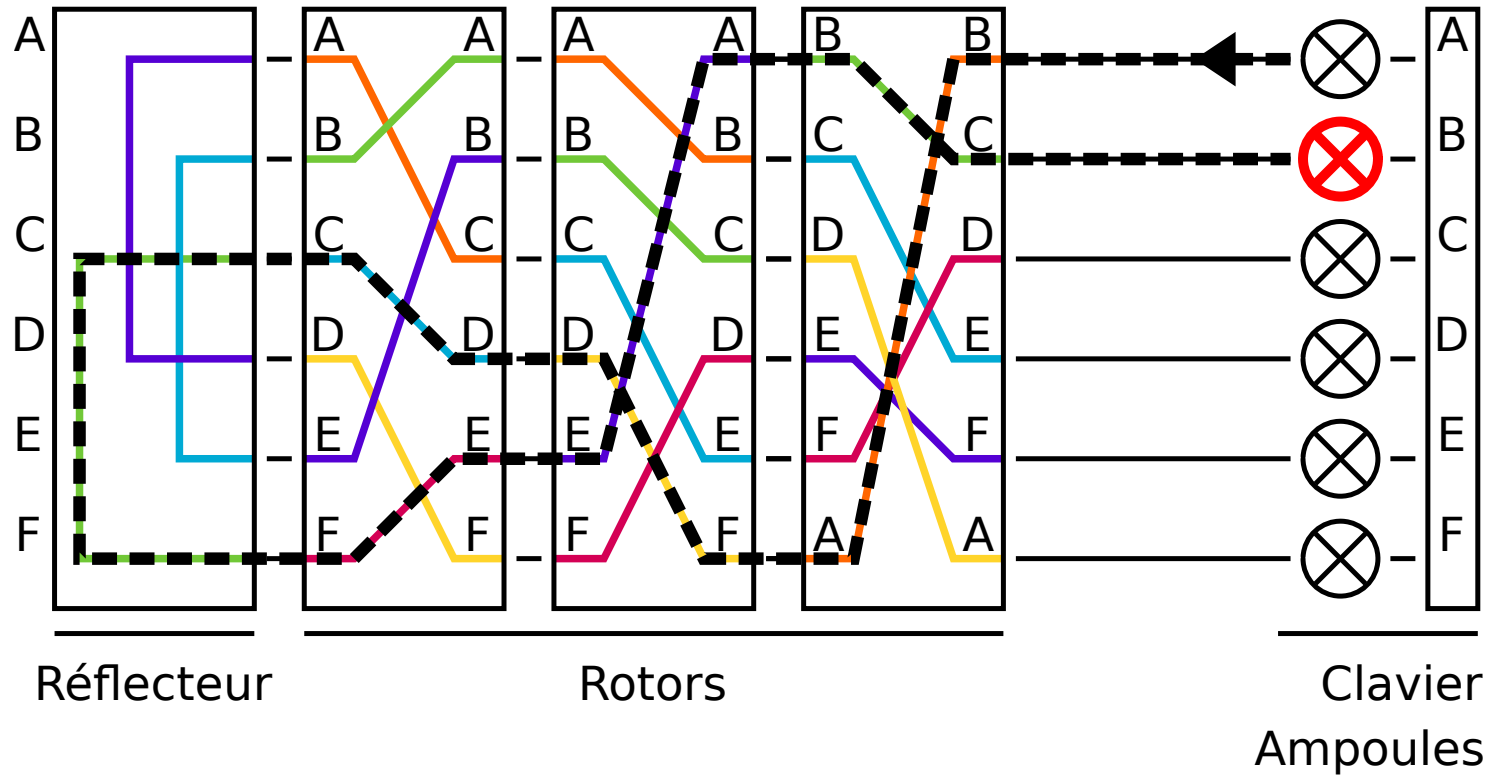

- Les rotors sont interconnectés
- Le circuit électrique est fermé à l'aide d'un clavier
- Le réflecteur rend le systèmes symétrique
- Lecture du chiffré se fait à l'aide de l'ampoule allumée

# La machine Enigma – $xx^{e}$ *siècle après N.E.*– Fonctionnement




- Les rotors sont interconnectés
- Le circuit électrique est fermé à l'aide d'un clavier
- Le réflecteur rend le systèmes symétrique
- Lecture du chiffré se fait à l'aide de l'ampoule allumée
- Les rotors sont ensuite incrémenté de droite à gauche

# La machine Enigma – $xx^e$ *siècle après N.E.*– Fonctionnement




- Les rotors sont interconnectés
- Le circuit électrique est fermé à l'aide d'un clavier
- Le réflecteur rend le systèmes symétrique
- Lecture du chiffré se fait à l'aide de l'ampoule allumée
- Les rotors sont ensuite incrémenté de droite à gauche

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Polonais

## Section allemande du bureau du chiffre Polonais – 1926 - 1939

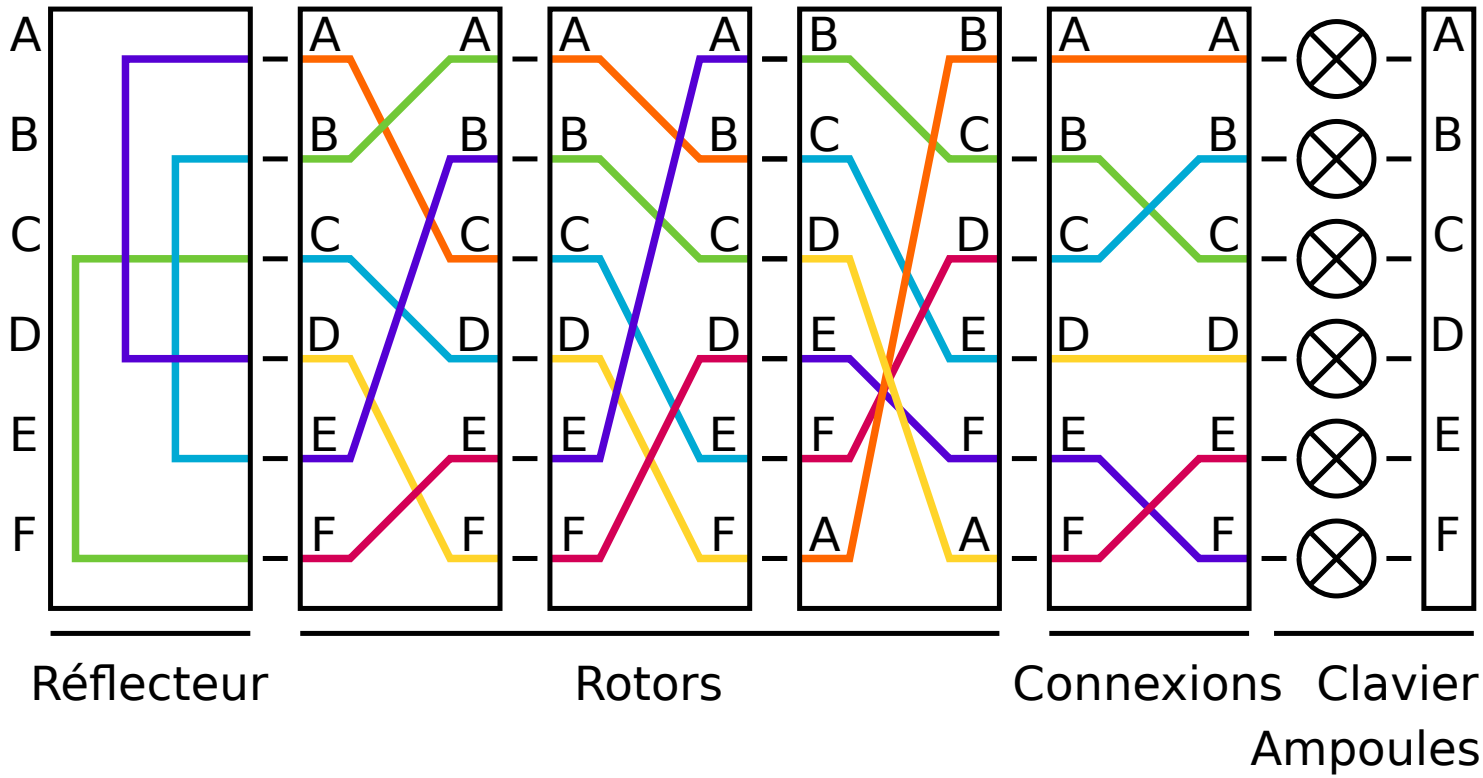
- Travaille sur l'Enigma à partir de 1926 (sans permutations)
  - Utilisation de schémas linguistiques

  → Quelques messages déchiffrés
- En 1932, ajout d'un tableau de 6 permutation et recrutement du mathématicien Marian Rejewski (23 ans)
  - Carnets d'utilisation de l'espion d'Hans-Thilo Schmidt

  → 6 première lettres d'un chiffré = $E(K_m.K_m, K_j)$
  - 6 équations aux multiples inconnues...
  - Aide du général Gustave Bertrand : configuration de 2 mois de 1932

  → Résolution du câblage en 1932
- Premières attaques sur les clés : Bomba et feuilles de Zyglaski
- En janvier 1939 : tableau de permutations à 10 → Turing

# La machine Enigma – $xx^e$ *siècle après N.E.*– Permutations




- Ajout du tableau des 10 permutations possibles

# La machine Enigma – $xx^e$ *siècle après N.E.*– La clé

## Clé de chiffrement en détails (modèle I)

1. Sélection de 3 rotors parmi 5
   $\binom{5}{3} = 10$
2. Placement des rotors entre eux
   $3! = 6$
3. Position initiale des rotors (et des anneaux de lettres)
   $26^3 = 17576$
4. Configuration du tableau de permutations de 10 lettres †
   $\frac{26!}{6! \times 10! \times 2^{10}} \approx 10^{14}$

## Combinatoire

- Sans †, avant la guerre $10 \times 6 \times 17576 \approx 10^6$ clés

→ Beaucoup donc robuste ?

- Avec †, pendant la guerre $10^6 \times 10^{14} \approx 10^{20}$

→ Encore plus beaucoup, donc encore plus robuste ? :)

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

**Objectif** : retrouver la position des rotors et du tableau de permutations

## Faiblesses de l'Enigma : attaque par mot probable ($\simeq$ clair connu)

1. Une lettre ne peut être substituée par elle même
2. Rigueur des Allemand dans les messages envoyés
   - Bulletin météo, formules de politesse

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

**Objectif** : retrouver la position des rotors et du tableau de permutations

## Faiblesses de l'Enigma : attaque par mot probable ($\simeq$ clair connu)

1. Une lettre ne peut être substituée par elle même
2. Rigueur des Allemand dans les messages envoyés
   - Bulletin météo, formules de politesse

## Alignement du mot probable avec un chiffré

L'attaquant Eve sait qu'à 12h, Bob envoie tous les jours à Alice un message court, chiffré avec Enigma, contenant systématiquement la formule suivante :

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

**Objectif** : retrouver la position des rotors et du tableau de permutations

## Faiblesses de l'Enigma : attaque par mot probable ($\simeq$ clair connu)

1. Une lettre ne peut être substituée par elle même
2. Rigueur des Allemand dans les messages envoyés
   - Bulletin météo, formules de politesse

## Alignement du mot probable avec un chiffré

L'attaquant Eve sait qu'à 12h, Bob envoie tous les jours à Alice un message court, chiffré avec Enigma, contenant systématiquement la formule suivante :

 À ce soir ma chérie

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

**Objectif** : retrouver la position des rotors et du tableau de permutations

## Faiblesses de l'Enigma : attaque par mot probable ($\simeq$ clair connu)

1. Une lettre ne peut être substituée par elle même
2. Rigueur des Allemand dans les messages envoyés
   - Bulletin météo, formules de politesse

## Alignement du mot probable avec un chiffré

L'attaquant Eve sait qu'à 12h, Bob envoie tous les jours à Alice un message court, chiffré avec Enigma, contenant systématiquement la formule suivante :

`xacesoirmacherie`

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

**Objectif** : retrouver la position des rotors et du tableau de permutations

## Faiblesses de l'Enigma : attaque par mot probable ($\simeq$ clair connu)

1. Une lettre ne peut être substituée par elle même
2. Rigueur des Allemand dans les messages envoyés
   - Bulletin météo, formules de politesse

## Alignement du mot probable avec un chiffré

L'attaquant Eve sait qu'à 12h, Bob envoie tous les jours à Alice un message court, chiffré avec Enigma, contenant systématiquement la formule suivante :

`xacesoirmacherie`

`maamjacpubttxqtavylmvsbcdzq`

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

**Objectif** : retrouver la position des rotors et du tableau de permutations

## Faiblesses de l'Enigma : attaque par mot probable ($\simeq$ clair connu)

1. Une lettre ne peut être substituée par elle même
2. Rigueur des Allemand dans les messages envoyés
   - Bulletin météo, formules de politesse

## Alignement du mot probable avec un chiffré

L'attaquant Eve sait qu'à 12h, Bob envoie tous les jours à Alice un message court, chiffré avec Enigma, contenant systématiquement la formule suivante :

```
 xacesoirmacherie
maamjacpubttxqtavylmvsbcdzq
```

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

**Objectif** : retrouver la position des rotors et du tableau de permutations

## Faiblesses de l'Enigma : attaque par mot probable ($\simeq$ clair connu)

1. Une lettre ne peut être substituée par elle même
2. Rigueur des Allemand dans les messages envoyés
   - Bulletin météo, formules de politesse

## Alignement du mot probable avec un chiffré

L'attaquant Eve sait qu'à 12h, Bob envoie tous les jours à Alice un message court, chiffré avec Enigma, contenant systématiquement la formule suivante :

```
  xacesoirmacherie
maamjacpubttxqtavylmvsbcdzq
```

Pas de substitution impossible → attaque à clair connu

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

## Caractérisation d'une clé par cycles de transformations

- Impossible de bruteforcer toutes les combinaisons de clé + permutations donnant notre mot probable (complexité en $10^{20}$)
- **Idée** : former des cycles de transformations caractéristiques de la clé $P$ que l'on veut trouver

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

## Caractérisation d'une clé par cycles de transformations

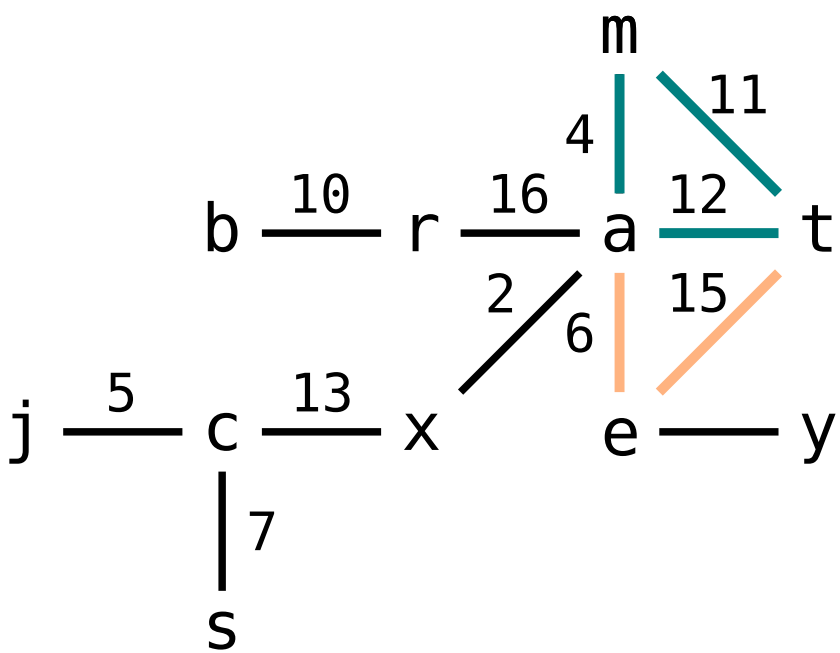
- Impossible de bruteforcer toutes les combinaisons de clé + permutations donnant notre mot probable (complexité en $10^{20}$)
- **Idée** : former des cycles de transformations caractéristiques de la clé $P$ que l'on veut trouver

| P + | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | | 12 | | 14 | | 16 | | 18 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | x | a | c | e | s | o | i | r | m | a | c | h | e | r | i | e |
| | a | m | j | a | c | p | u | b | t | t | x | q | t | a | v | y |

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

## Caractérisation d'une clé par cycles de transformations

- Impossible de bruteforcer toutes les combinaisons de clé + permutations donnant notre mot probable (complexité en $10^{20}$)
- **Idée** : former des cycles de transformations caractéristiques de la clé $P$ que l'on veut trouver



# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

**Hypothèse** : on connait les $\leq 10$ permutations choisies

## Principe d'un circuit de recherche de la clé $P$ (position des rotors)

- On "branche" des Enigma en série pour toutes les boucles trouvées
- Les lettres (sommets) sont les connexions à effectuer entre Enigma
- Les nombres (arêtes) sont les Enigma dont la position des rotors est placée à $P+$ étiquette

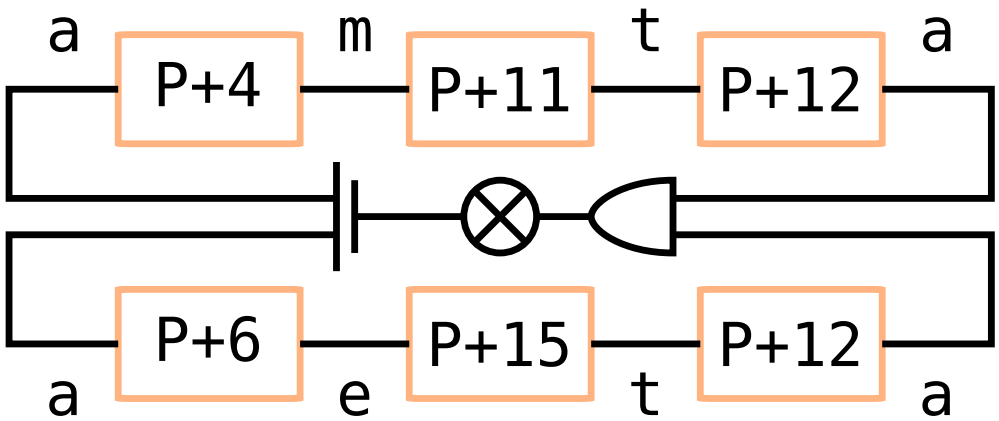
1. On choisit une lettre de départ, source du courant
2. On branche une ampoule sur la lettre de sortie (même que celle d'entrée)

$\infty$ On incrémente toutes les $P$ sur toutes les Enigma.

- Si l'ampoule s'allume, on s'arrête → clé probable à tester

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

Vue conceptuelle d'une bombe (pas tout à fait de Turing)




Machine d'automatisation de recherche des cycles

Enigma complète
rotors + reflecteur + permutations

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

**Retrait de l'hypothèse** : on ne connait plus les permutations choisies

## Observation de Turing

$\alpha$ est transformée en 'a' puis en $\alpha \rightarrow$ identité

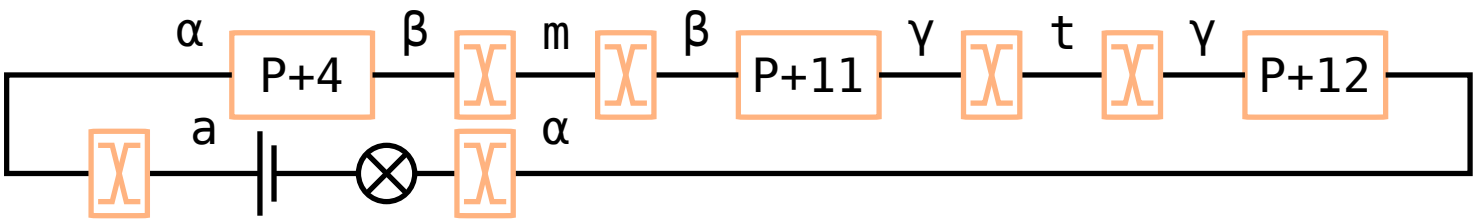


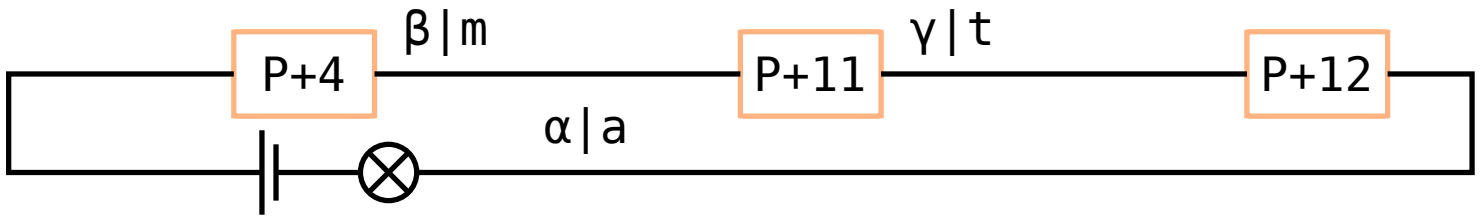

Tableau de permutations Rotors seuls

# La machine Enigma – $xx^e$ *siècle après N.E.*– Les Anglais

**Retrait de l'hypothèse** : on ne connait plus les permutations choisies

## Observation de Turing

$\alpha$ est transformée en 'a' puis en $\alpha \to$ identité




Rotors seuls

On peut retirer les tableaux de permutations et connecter $\alpha$, $\beta$ et $\gamma$, la lumière s'allume toujours. Mais *quid* de ces inconnues ?

# La machine Enigma – xx$^e$ *siècle après N.E.*– Les Anglais

## Idée de turing

Modification sur la machine précédente :

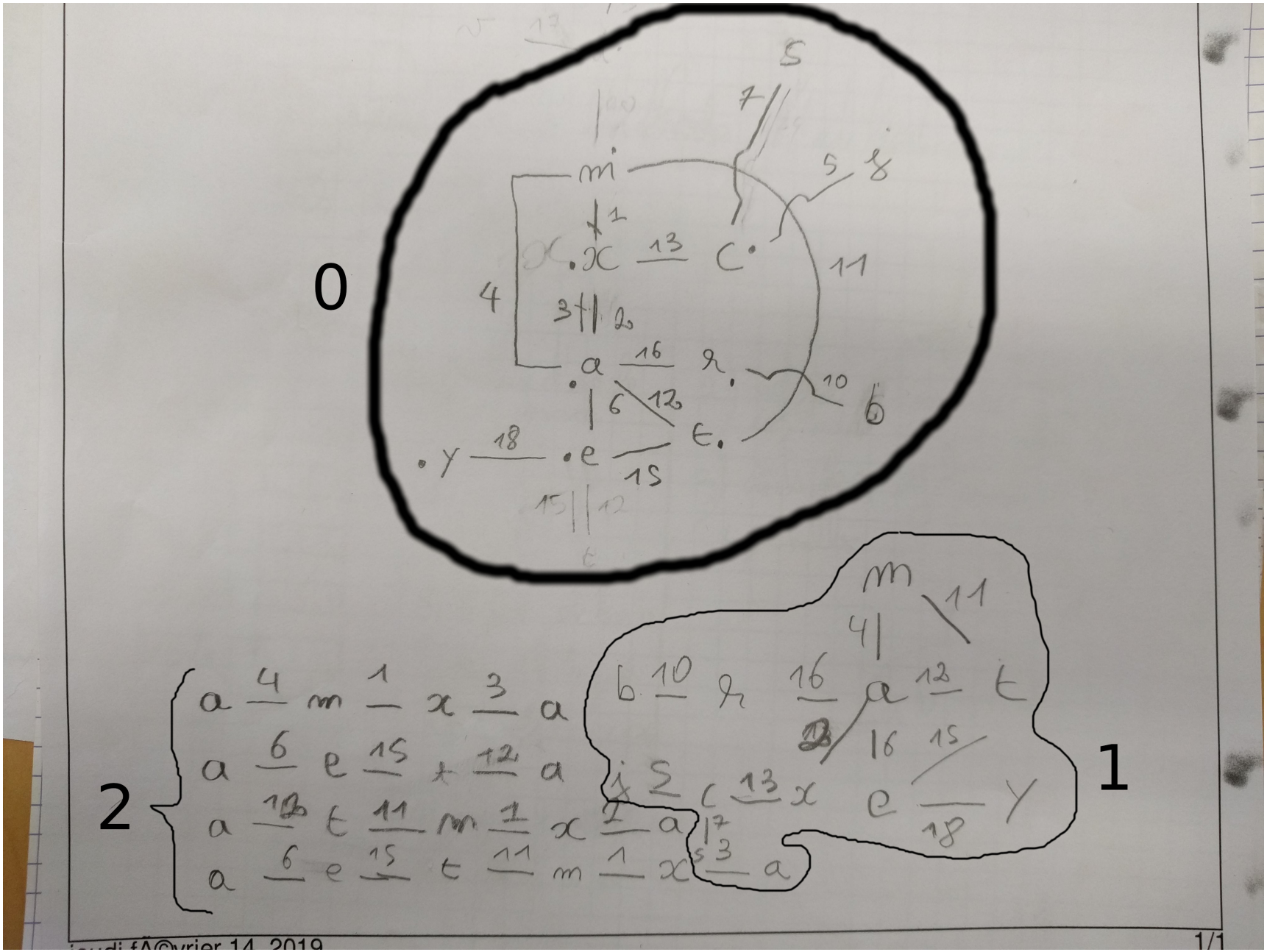
1. Brancher de toutes les sorties des Enigma sur leur suivant respectif sauf pour la section de test (au niveau d'$\alpha$ par exemple)
2. Choisir $\alpha \in [a - z]$ comme entrée et lettre permutée (ou non) pour a
3. Tester la clé $P$ courante pour tout $\alpha$
4. Si la lampe s'allume, on teste la cohérence de cette hypothèse du point de vue du tableau de permutation pour les lettres du cycle
5. Si cela implique qu'une lettre est permutée plus d'une fois, on jette la clé et on continue

→ **Bombe de Turing**

# La machine Enigma – $xx^e$ *siècle après N.E.*– Faiblesses

1. Fiabilité des communications et tolérance aux fautes
   - Répétitions dans le clair
2. Problématique de l'échange des clés
   - Stabilité des rotors et des modèles de machine ($26^{2+3} \approx 300 \times 10^6$)
   - Stabilité du cablage du tambour d'entrée ($26! \approx 4 \times 10^{26}$)
3. Symétrisation du système à l'aide du miroir
   - Annule certaines transformation lors de la cryptanalyse
4. Réflecteur non identité : implique une traduction différente
   - Permet les attaques par clair connu
5. Utilisation faite par les Allemands
   - Attaques à clair connu, parfois à clair choisi

# La machine Enigma – xx$^{e}$ *siècle après N.E.*– Démo

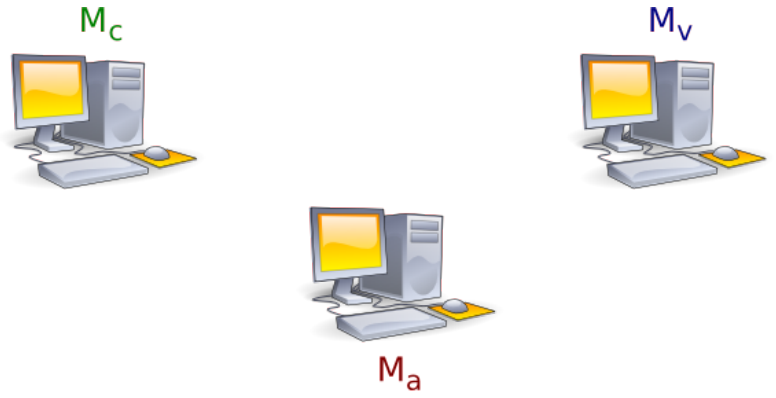


# Sommaire

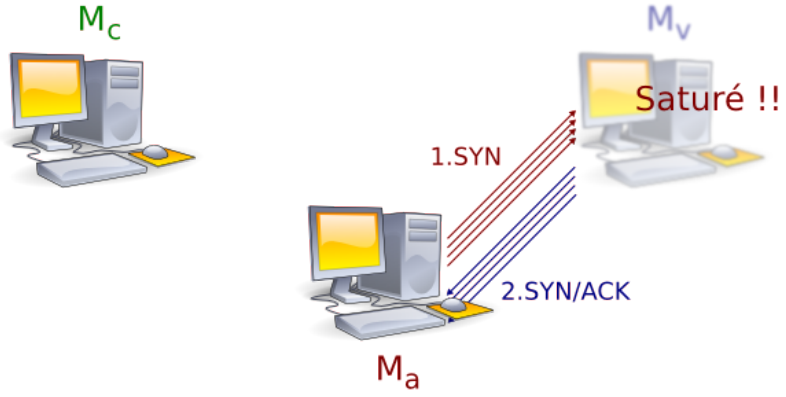
# Kevin Mitnick – *25 décembre 1995*

- Attaque du 25 décembre 1994
- Protagonistes de l'affaire
  - Kevin Mitnick : pirate informatique américain
  - Tsutomu Shimomura : expert américain en sécurité informatique
- Configuration
  - $M_a$, attaquant, San Francisco
  - $M_v$, victime, San Diego
  - $M_c$, cible, San Diego
  - Liaison de confiance $M_v \rightarrow M_c$
- Objectif
  - Empêcher $M_v$ de dialoguer avec $M_c$
  - Permettre à $M_a$ de répondre à la place de $M_c$
- Techniques employées
  - Déni de service ou saturation de service (*SYN Flooding*)
  - Usurpation d'adresse IP (*IP Spoofing*)

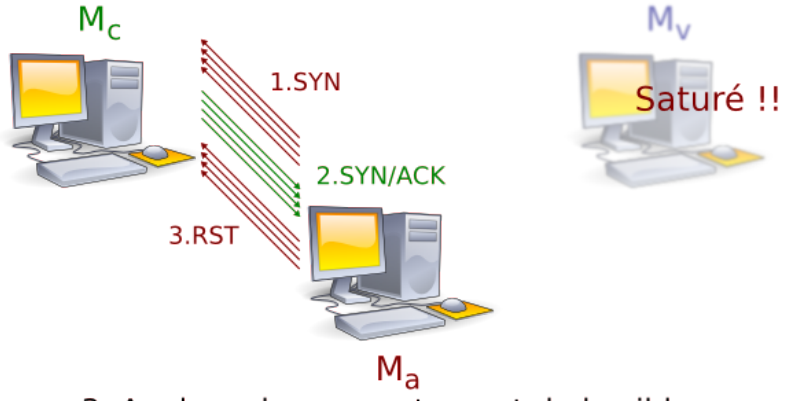
# Kevin Mitnick – *25 décembre 1995*



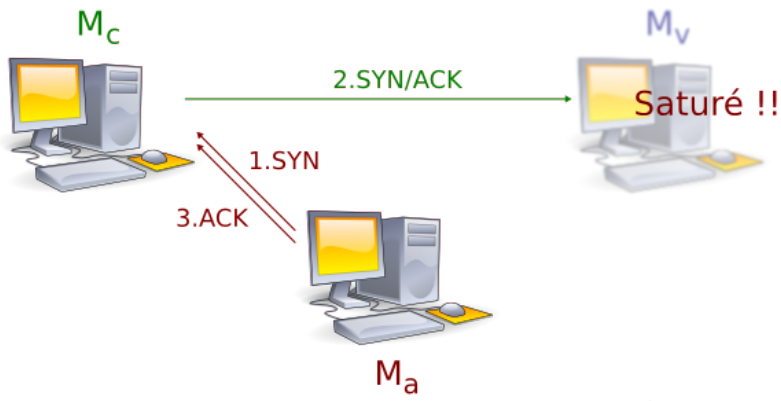

1. Etat initiale



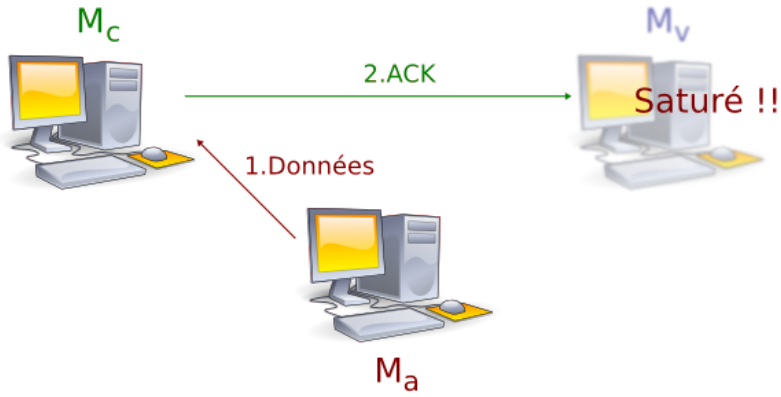

2. Saturation de la victime



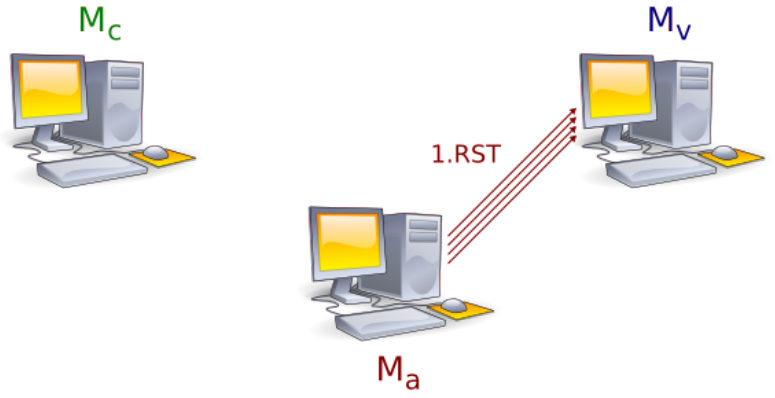

3. Analyse du comportement de la cible




4. Ouverture d'une connection avec la cible, à l'aveugle




5. Exploitation de la cible, à l'aveugle




6. Libération de la victime

# Sommaire

# A nos jours

1949 John Von Neumann, logiciels autocopiés
1960 ingénieurs des laboratoires Bell, Core war
1984 Scientific American, guide pour fabriquer ses propres virus
1986 Les frères Alvi, virus Brain
1988 Robert Morris, fraude informatique
2000 Virus "I Love You"

Plus de détails [1]

# Sommaire

# Sommaire

## Les propriétés de la sécurité

# La sûreté de fonctionnement informatique

*Discipline dont les concepts englobent la sécurité informatique.*

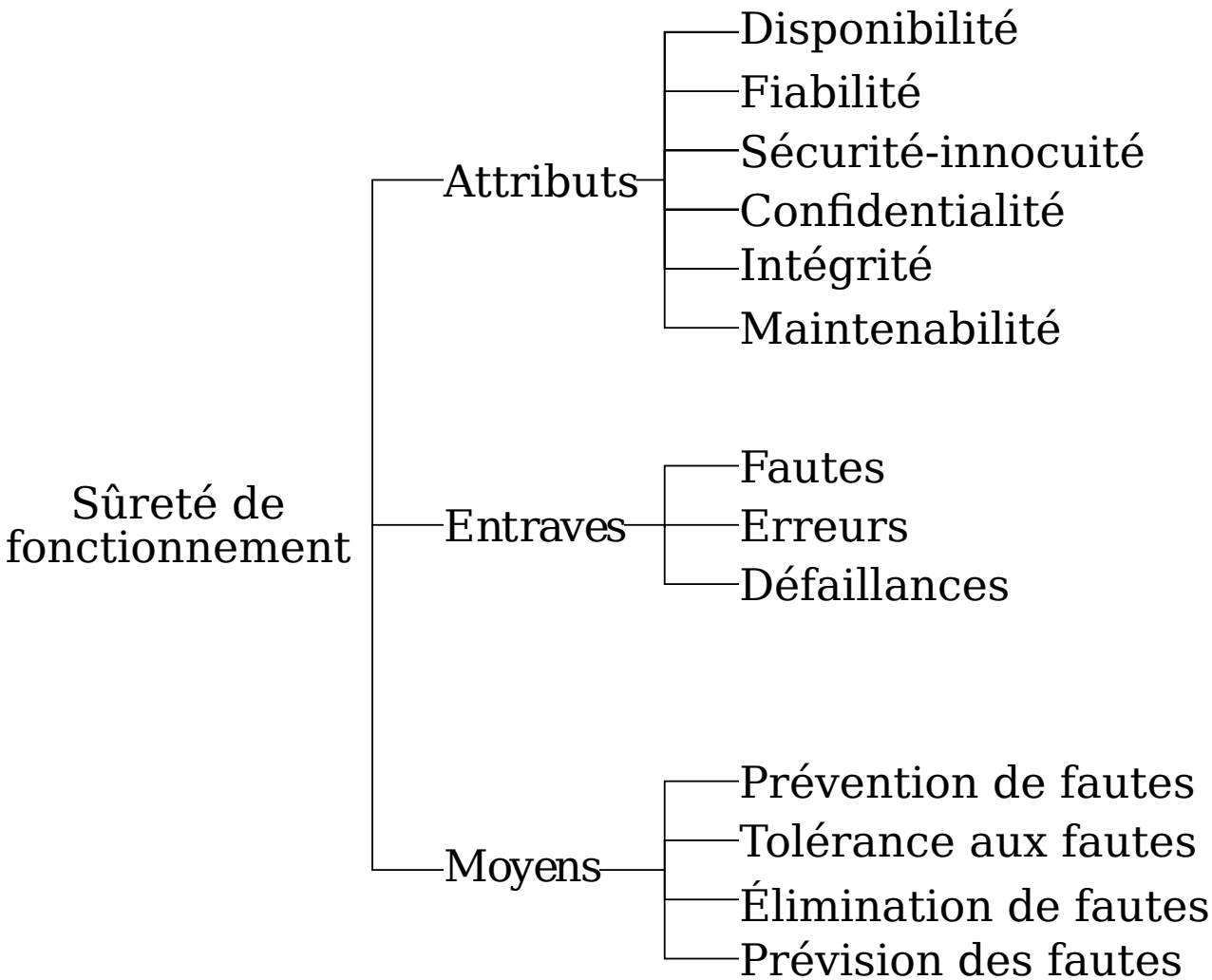
## Définition

L'aptitude d'un système à délivrer un service de confiance justifiée

## Attributs

- **Disponibilité** : capacité d'un système à être prêt à l'utilisation.
- **Fiabilité** : continuité du service.
- **Sécurité-innocuité** : non-occurrence de conséquences catastrophiques pour l'environnement.
- **Confidentialité** : non-occurrence de divulgations non autorisées de l'information.
- **Intégrité** : non-occurrence d'altérations inappropriées de l'information.
- **Maintenabilité** : aptitude d'un système à être réparé ou à subir des évolutions.

# La sûreté de fonctionnement informatique



# Entraves à la sûreté de fonctionnement

- Une **défaillance** survient lorsque le service délivré dévie de l'accomplissement de la fonction du système.
- Une **erreur** est la partie de l'état du système qui est susceptible d'entraîner une défaillance.
- Une **faute** est la cause adjugée ou supposée d'une erreur.

Chaîne fondamentale

. . . faute → erreur → défaillance → faute → . . .

# Les moyens pour la sûreté de fonctionnement informatique

*Éviter les fautes*

## Prévention des fautes

Comment empêcher que des fautes surviennent ou soient introduites

## Élimination des fautes

Comment réduire la présence (en nombre ou en gravité) des fautes

*Accepter les fautes*

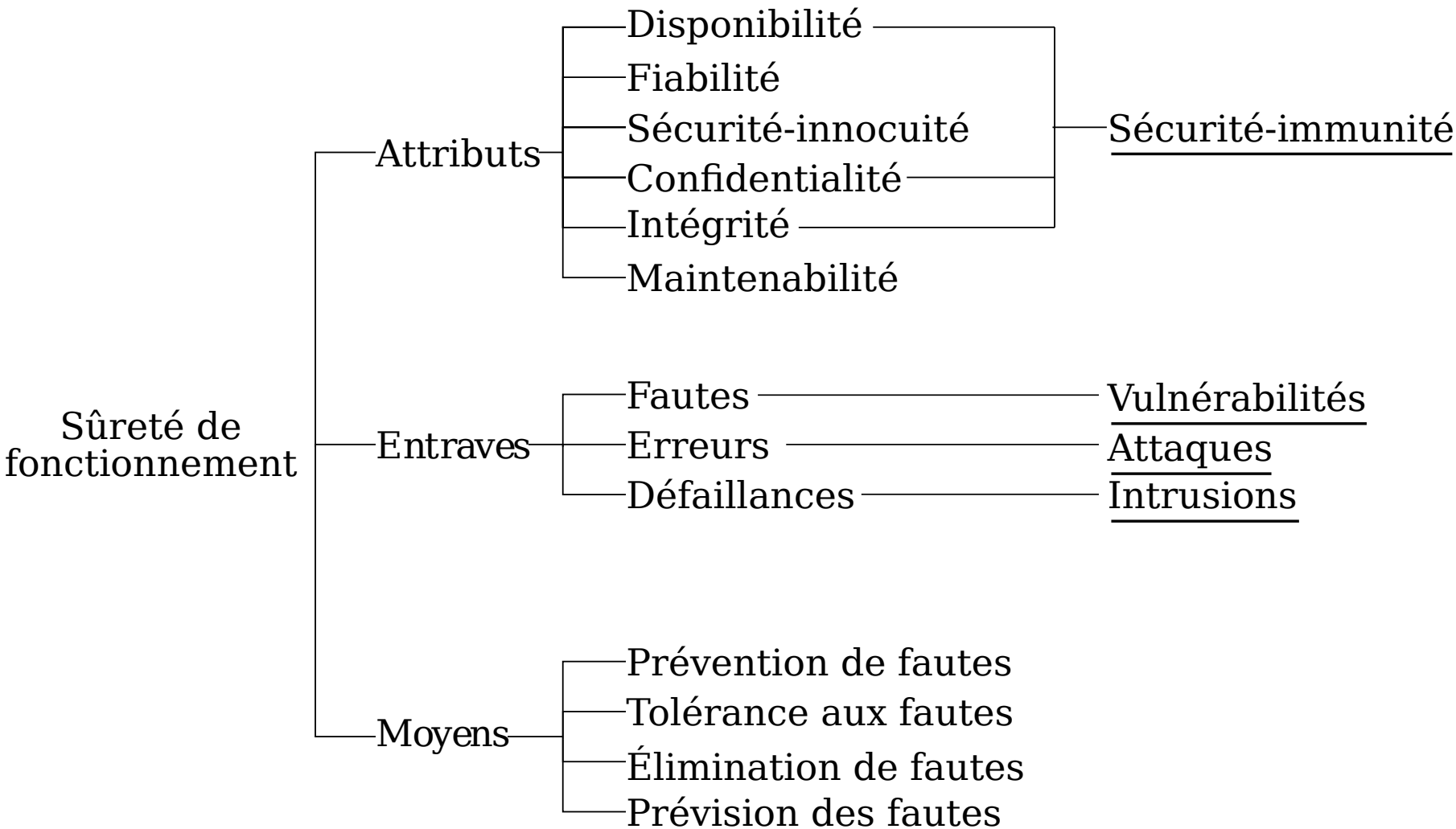
## Tolérance aux fautes

Comment fournir un service conforme à la fonction en dépit des fautes

## Prévision des fautes

Comment estimer la présence, la création et les conséquences des fautes

# La sûreté de sécurité-immunité



# Sécurité des systèmes d'information

## Definition

Sécurité(-immunité) = confidentialité + intégrité + disponibilité
Vis-à-vis des fautes intentionnelles dites malveillances
Malveillances = logiques malignes + intrusions

- Perte de confidentialité = divulgation non autorisée d'information
- Perte d'intégrité = altération non autorisée de l'information
- Perte de disponibilité = capacité d'un système à être prêt à l'utilisation

**vis-à-vis de malveillances**

*For most distributed systems, the security objectives of confidentiality, integrity, and availability of information apply. A loss of confidentiality is the unauthorized disclosure of information. A loss of integrity is the unauthorized modification or destruction of information. A loss of availability is the disruption of access to or use of information or an information system.[2]*

# Entraves à la sécurité-immunité

- **Une attaque** est une faute d'interaction externe au système, dont le but est de violer un ou plusieurs des attributs de sécurité. Elle peut être aussi définie comme une tentative d'intrusion.
- **Une vulnérabilité** est une faute qui peut être accidentelle, intentionnelle malveillante ou non malveillante placée dans les exigences, la spécification, la conception ou la configuration du système, ou dans la manière dont il est utilisé.
- Une vulnérabilité peut être exploitée avec une attaque pour créer une **intrusion**. Une intrusion est donc une faute malveillante, initiée depuis l'extérieur pendant l'utilisation du système.

Chaîne fondamentale

. . . vulnérabilité → attaque → intrusion → vulnérabilité → . . .

# Les moyens pour la sécurité-immunité

*Éviter les fautes intentionnelles*

## Prévention des fautes

Prévention des vulnérabilités ; prévention des attaques ; prévention d'intrusion

## Élimination des fautes

Élimination des vulnérabilités

*Accepter les fautes intentionnelles*

## Tolérance aux fautes

Tolérance aux intrusions

## Prévision des fautes

Prévisions des vulnérabilités ; prévision des attaques ; prévision des intrusions

# L'information

## Definition

Une information est composée de données et méta-données.

- Données : captées ou générées, traitées, stockées, transmises, affichées
- Méta-données : créées et utilisées par les services sous-jacents

Une méta-donnée est une donnée à un niveau inférieur

# Autres propriétés

## Anonymat

Confidentialité de (identité de l'utilisateur)

## Protection de la vie privée

Confidentialité de (identité de l'utilisateur + données personnelles)

## Authenticité d'un message

Intégrité de (contenu + identité de l'émetteur + date + . . . )

## Authenticité d'un document

Intégrité de (contenu + identité du créateur + date + . . . )

## Authenticité d'un utilisateur

Intégrité de (identité)

# Autres propriétés

## Imputabilité

Disponibilité de (qui + quoi + quand + où + . . . ) d'une action

## Non-répudiation d'origine

Disponibilité de (identité de l'émetteur + . . . ) +
intégrité du (contenu)

## Non-répudiation de réception

Disponibilité de (identité du récepteur + . . . ) +
intégrité du (contenu)

## Protection de la propritété intellectuelle

Confidentialité de (contenu) +
intégrité du (contenant)

# Besoins de sécurité selon les secteurs

- Défense, gouvernement
  Confidentialité ≫ intégrité, disponibilité
- Finance
  Intégrité ≫ disponibilité > confidentialité
- Autres (industrie, administrations, médecine, . . . )
  Ça dépend

⇒ Besoin de définir les spécificité de l'application
⇒ Politique de sécurité

# Sommaire

# Sommaire

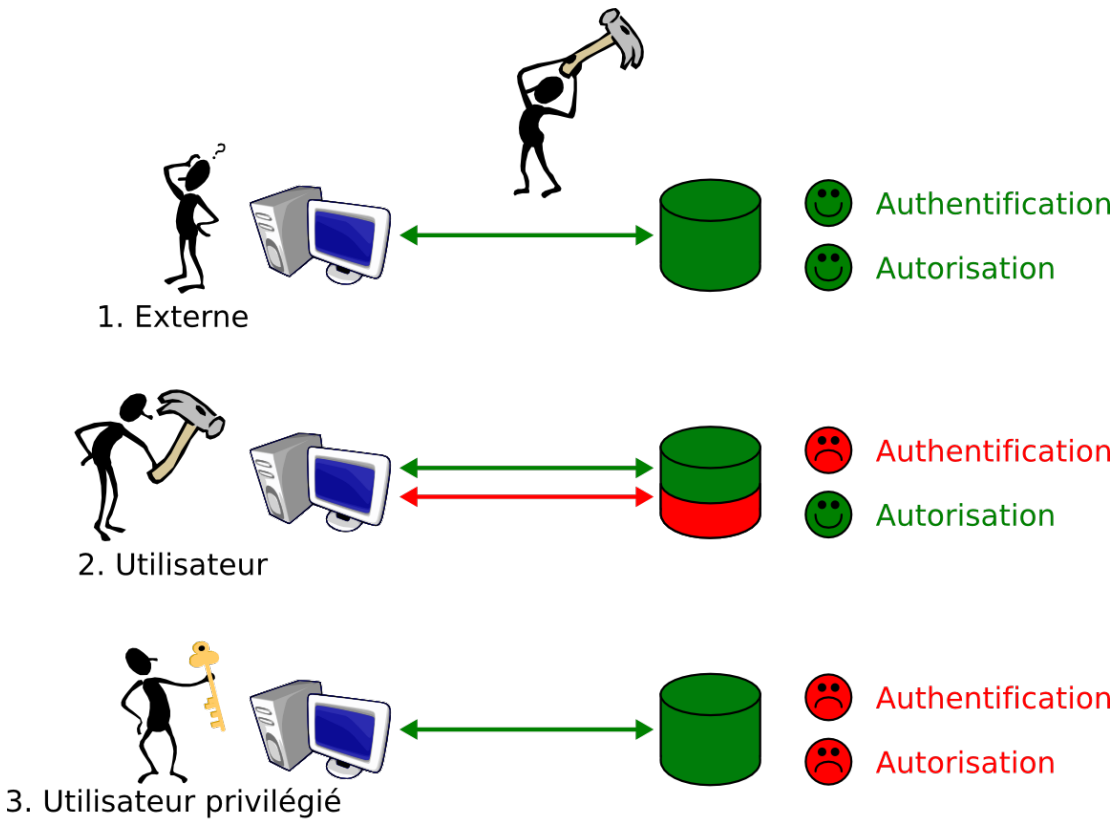
# Sommaire

# Les attaquants et leurs motivations

- Jeu : explorer les limites, éprouver et étendre ses connaissances, découvrir de nouvelles failles, améliorer la sécurité : “hackers”
- Emulation, sectarisme : groupe de hackers : “exploits”
- Vandalisme : montrer sa force, punir : “web defacing”, virus, vers, . . .
- Politique, idéologie : ex. CCC, 600 sites danois “défigurés” en février 2006
- Vengeance : ex. SCORES
- Profit : espionnage, extorsion de fonds : concurrence déloyale, crime organisé, espionnage international (attaques probablement chinoises contre des sites gouvernementaux des USA, GB, Allemagne, France, . . . )
- Guerre informatique, terrorisme : 2007 DDoS contre des sites estoniens, 2008 contre des sites georgiens, . . .
- Sensibilisation, lobbying
- Protection abusive : ex. SONY

# Les attaquants et leurs motivations

Qui sont les “intrus” ?



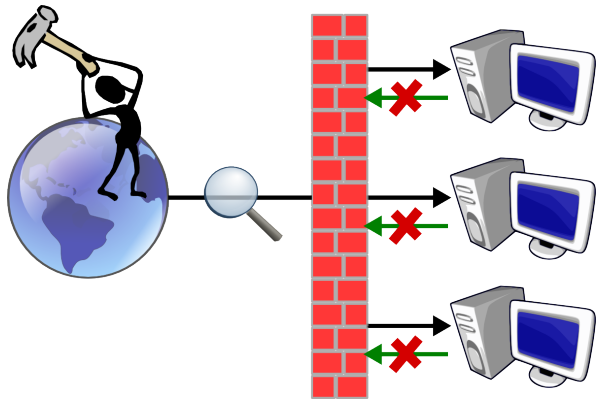

80% des fraudes sont "autorisées"

# Les attaquants et leurs motivations

[fragile]

- Organisation
  - Seul ? Groupe ?
- Compétence
  - Novice ? Averti ? Expert ?
- Comportement
  - Discret ? Ostensible ?

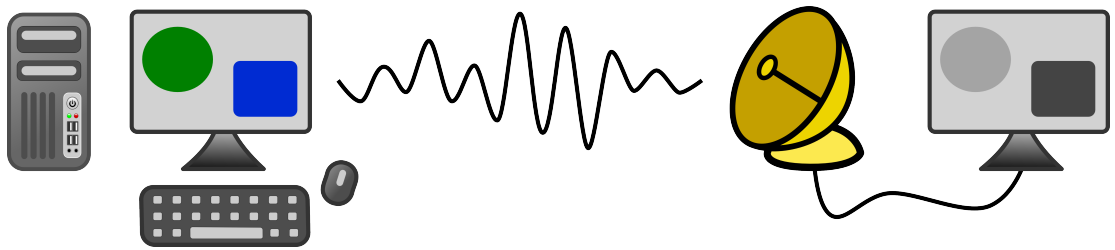
$\Rightarrow$ Utilisation de “pots de miel”



# Sommaire

# Classification des attaques(1)

- Ecoute passive (confidentialité)
  Accès sans modification à des informations générées, transmises, stockées ou affichées sur des composants vulnérables
  - *sniffing*, *snooping*, *eavesdroping*, *wiretapping*, réutilisation de mémoire (buffers, fichiers temporaires, supports magnétiques), analyse de trafic, effet Van Heck (TEMPEST), *key logger*, . . .
    Câbles de claviers
    Câbles de contrôlleurs d'affichage



*Effet Van Heck*

# Classification des attaques(2)

- Interception (intégrité)
  Modification d'informations transmises
  - Modification de messages, rejeu, éblouissement
- Cryptanalyse (confidentialité)
  Obtenir des informations secrètes (messages en clair, clés, algorithme de chiffrement) à partir des informations publiques (cryptogrammes)
  - Identification de collisions dans MD5 en 2004[11], utilisé pour signer, par exemple, les fichiers téléchargés
- Répudiation (intégrité)
  Refuser de reconnaître une opération qu'on a effectuée
  - Répudiation d'origine, de réception

# Classification des attaques(3)

- Déduction par inférence, furetage (confidentialité)
  Obtenir des informations secrètes (par exemple, des données personnelles) à partir des informations auxquelles on a accès (par exemple, statistiques)
- Déguisement (masquerade) (intégrité)
  Se faire passer pour quelqu'un d'autre
  (tromper l'authentification, s'il y en a . . . )
  - *Phishing* → obtenir des renseignements personnels
  - *Scam* (Fraude 4-1-9) → escroquer
  - *WiPhishing* : hotspots WiFi ouverts

  - Attaques *homographes* → cacher les *vraies* URL
  - *IP spoofing* (contre-mesure : *ingress filtering*)
  - *DNS poisoning* : URL ↔ @IP
  - *ARP poisoning* : @MAC ↔ @IP

# Cas particulier de déguisement : *phishing*

- Hameçonnage de mots de passe : autrefois par téléphone
  Maintenant : (courriel, blog, news, IRC, MSN Messenger. . . )
  avec URL cliquable pointant vers une autre URL, ou (SMS)
- 49 084 sites actifs en juin 2009, durée de vie moyenne : 1 jour à 1 mois

  `http://www.antiphishing.org` et `http://phishery.internetdefence.net`

- En 2007, 3,6 $m^{ns}$ d'américains piégés, 3,2 $m^{ds}$ de dollars de perte
- Exemple, en 2005 : Croix-rouge et cyclone Katrina, grippe aviaire, . . .
- Octobre 2009 : Des fraudeurs se font passer pour les services des impôts sur internet[8]

| *Avril 2009* | | *Mai 2009* | | *Juin 2009* | |
|---|---|---|---|---|---|
| USA | 66,24% | USA | 68,65% | Suède | 46,18% |
| Chine | 8,00% | Chine | 6,33% | USA | 42,39% |
| Suède | 7,76% | Canada | 6,15% | Canada | 3,52% |
| Canada | 2,67% | Allemagne | 2,24% | Chine | 1,57% |
| Allemagne | 1,97% | Royaume-Uni | 1,60% | Allemagne | 0,88% |
| Royaume-Uni | 1,02% | Suède | 1,29% | Royaume-Uni | 0,54% |
| Pays-Bas | 0,98% | Russie | 1,23% | France | 0,53% |
| Corée | 0,90% | France | 1,19% | Corée | 0,40% |
| France | 0,86% | Corée | 1,17% | Pays-Bas | 0,39% |
| Russie | 0,85% | Pays-Bas | 1,01% | Russie | 0,28% |

# Cas particulier de déguisement : *phishing*

From remboursement@impots.gouv.fr Mon Oct 5 09:17:46 2009
Return-Path: <remboursement@impots.gouv.fr>
Reply-To: <remboursement@impots.gouv.fr>
From: "L'administration Fiscale"<remboursement@impots.gouv.fr>
Subject: Notification d'impôt
Date: Mon, 5 Oct 2009 02:55:50 -0400
Content-Type: text/html; charset="Windows-1251"
X-Spam-Status: Yes
X-Spam-Score: 9.706 (*********)

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

DIRECTION GENERALE DES FINANCES PUBLIQUES 05/10/2009

Notification d'impôt - Remboursement

---

Après les derniers calculs annuels de l'exercice de votre activité, nous avons déterminé que vous êtes admissible à recevoir un remboursement d'impôt de € 178,80.

S'il vous plaît soumettre la demande de remboursement d'impôt et nous permettre de 10 jours ouvrables pour le traitement.

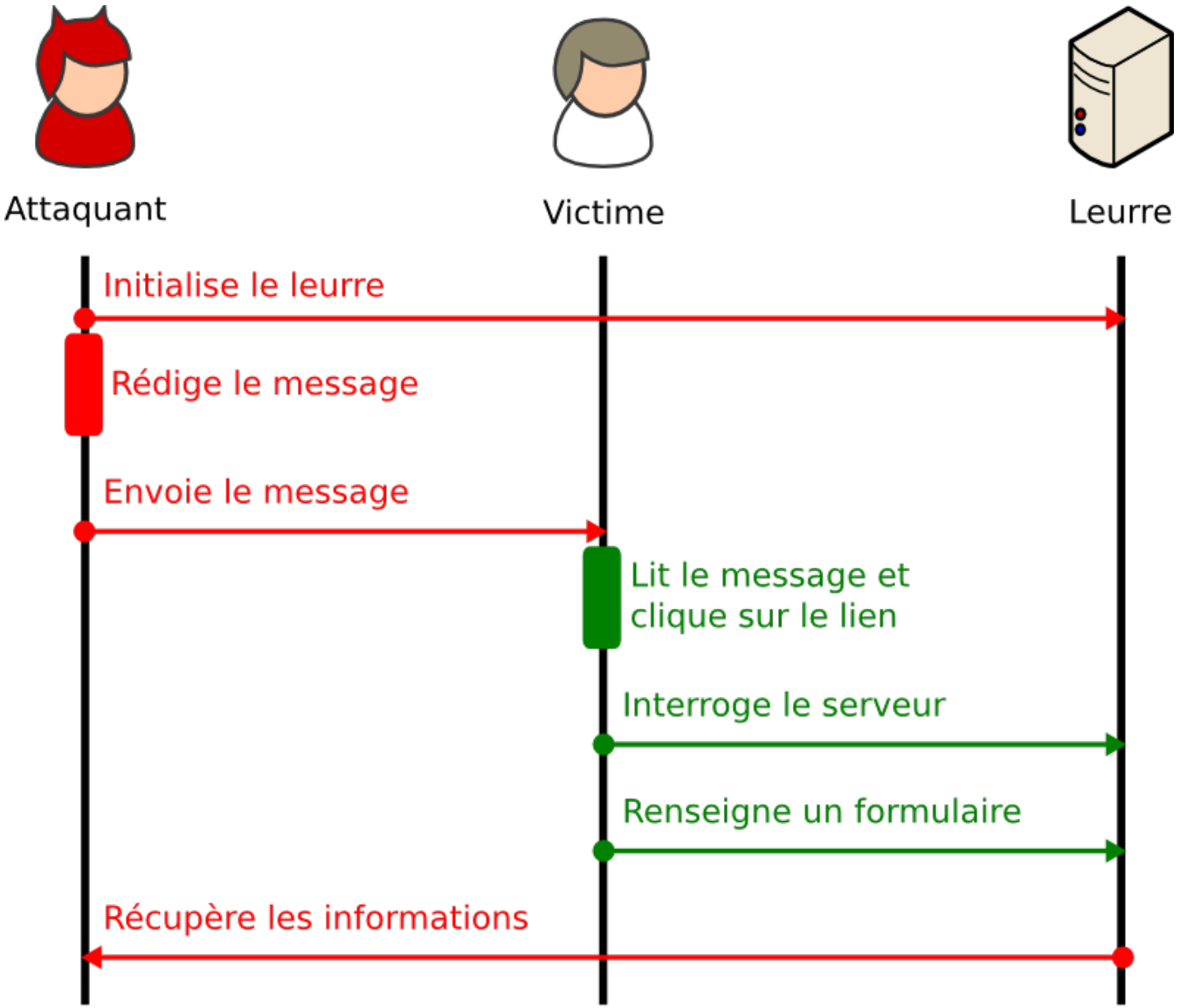
Pour accéder au formulaire pour votre remboursement d'impôt, cliquez ici

Un remboursement peut être retardé pour diverses raisons. Par exemple la soumission des dossiers non valides ou inscrivez après la date limite.

Le Conciliateur fiscal adjoint

Philippe BERGER

# Cas particulier de déguisement : *phishing*



# Comment cacher la vraie URL ?

- *Open Redirect* : redirection vers une URL sans validation

  `http://unsite.com/redirect?url=http://unstie.com`

  `http://cgi4.ebay.com/ws/eBayISAPI.dll?MfcISAPICommand=RedirectToDomain&DomainUrl=http%3A%2F%2F%32%31%31%2E%31%37%32%2E%39%36%2E%37%2FUpdateCenter%2FLogin%2F%3FMfcISAPISession%3DAAJbaQqzeHAAeMWZlHhlWXS2AlBXVShqAhQRfhgTDrferHCURstpAisNRqAhQRfhgTDrferHCURstpAisNRpAisNRqAhQRfhgTDrferHCUQRfqzeHAAeMWZlHhlWXh`

- *Pharming* : tromper la traduction DNS
  - *DNS poisoning*
  - Modifier la table /etc/hosts
- *Phishing* et HTTPS
  - MD5 *cassé* ⇒ possibilité de forger des certificats
  - Vérifier la présence du cadenas ne suffit plus

# Comment cacher la vraie URL ?

- Attaques homographes : obtenir un nom de domaine en caractères UNICODE non-latins (ex. : cyrillique) qui s'affiche comme le site cible : eBay.com ≠ eBay.com
- Attaques dites d'erreur typographiques : par exemple, `goggle.com` ou `googgle.com`
  Plus subtil, pages d'erreur *DNS* redirigées par le *FAI* vers un site publicitaire hacké, *Wildcard DNS*
  `www.schneier.com/blog/archives/2008/04/hacking_isp_err.html`
- Attaques par modification de logiciel
  - Noyau ou interception des appels systèmes
  - Modification des applications : browser (java, plug-ins), mail.

# Attaques dites d'erreur typographiques

ryanair.com → http://www.ryanair.com/site/FR/

# Attaques dites d'erreur typographiques

`ryamair.com` → `http://www.searchnut.com/?domain=ryamair.com`

**ryamair.com** Search the Web: Go

Ryanair | Airline | Aer Lingus | Airline Tickets | Ryanair Cheap Flights | Low Cost Airlines | Cheap Flights

**Ryanair**

- Airline
- Aer Lingus
- Airline Tickets
- Ryanair Cheap Flights
- Low Cost Airlines
- Cheap Flights
- Aviation
- Plane
- Europe Airline
- Air Travel

**Aviation**

- Commercial Aviation
- Aviation Services
- Aero
- Aviation Information
- Aviation Service

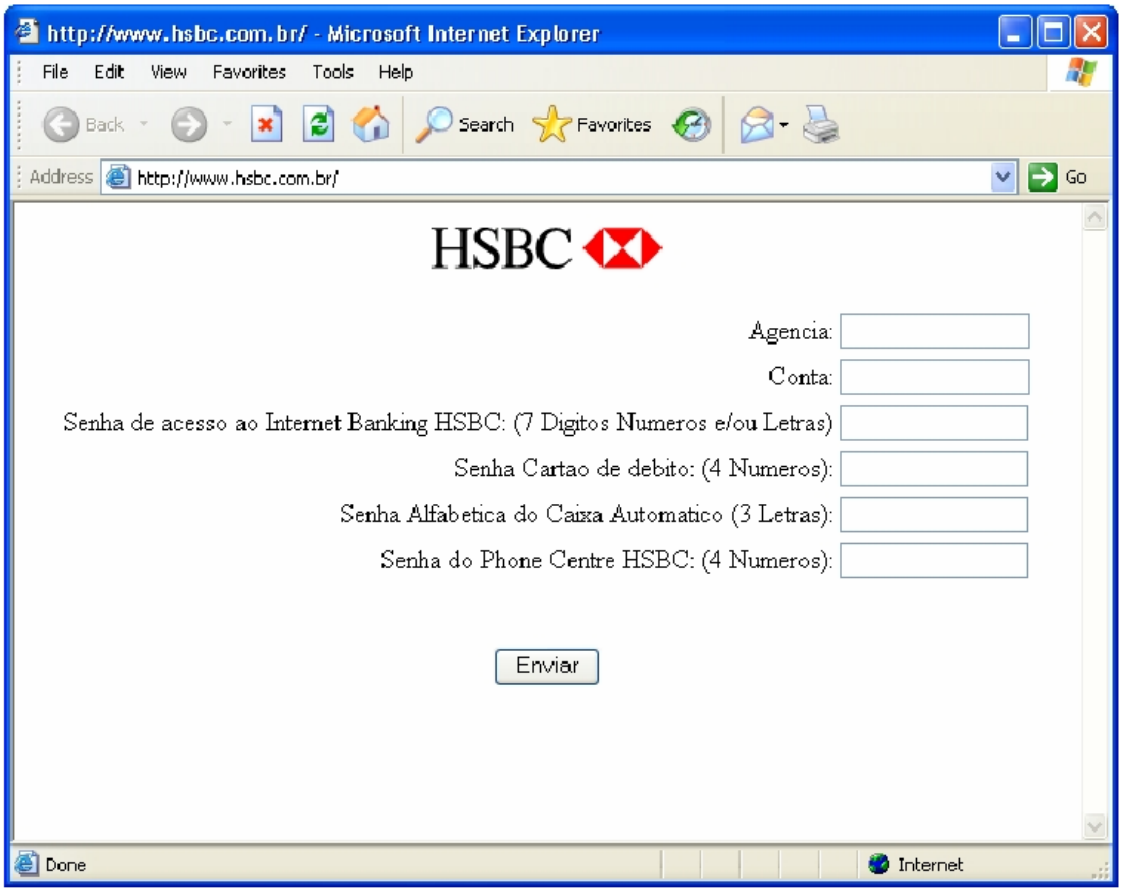
**Plane**

- Pilot
- Airline Tickets
- Train
- Jet
- Helicopter

**Low Cost Airlines**

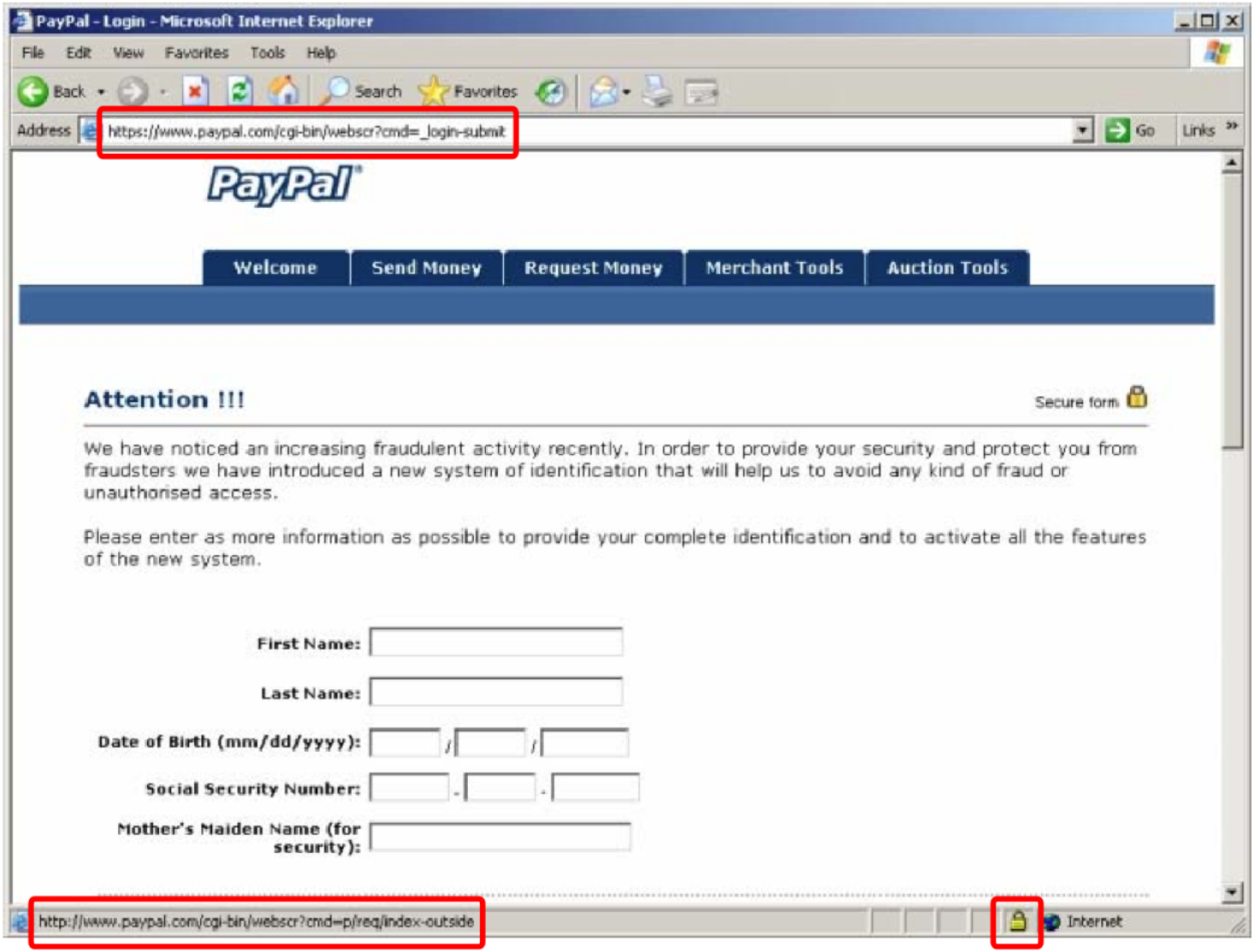
- Discount Airline
- Europe Airline
- Low Cost Air
- Cheap Flights
- Cheap Airline

# *Pharming* : tromper la traduction DNS

# Attaques par modification de logiciel

https://www.paypal.com




http://www.paypal.com

# Cas particulier de déguisement : *Scam*

Date: Thu, 24 Sep 2009 11:31:29 -0700 (PDT)
From: Linda Fastus <ms_lindafastus@info2link.biz>
Subject: FROM MRS LINDA FASTUS SCHWARZ
X-Spam-Status: Yes
X-Spam-Score: 12.088 (**********)

FROM MRS LINDA FASTUS SCHWARZ
ABIDJAN COTE D'IVOIRE
PLEASE EMAIL BACK

DEAREST IN CHRIST,

KNOW THAT THIS MAIL MAY REACH YOU BY SURPRISE.AS WE DONT KNOW OURSELF PREVIOUSLY.I AM THE ABOVE NAME PERSON FROM INDIA. I AM MARRIED TO MR FASTUS SCHWARZ ;WHO WAS THE AMBASSADOR OF JAMAICA FOR NINETEN YEARS IN COTE DIVOIRE.WE WERE MARRIED FOR FIFTEN YEARS WITHOUT A CHILD. HE DIED IN DECEMBER 27TH 2004 AFTER A BRIEF ILLNESS THAT LASTED FOR ONLY TWO WEEKS

BEFORE HIS DEATH WE ARE HAPPY HUSBAND AND WIFE CHRISTIAN FAMILY. SINCE HIS DEATH IDECIDED NOT TO REMARRY OR GET A CHILD OUTSIDE MY MATRIMONIAL HOME WHICH THE BIBLE IS AGAINST. WHEN MY LATE HUSBAND WAS ALIVE, HE DEPOSITED THE SUM OF ( USD $12.7MILLION ) TWELVE MILLION SEVEN HUNDRED THOUSAND U.S.DOLLARS INTO A BOX FOR SECURITY REASON AND THE MONEY STILL WITH THE SECURITY COMPANY HERE IN ABIDJAN COTE D'IVOIRE.

MEANWHILE, I HAVE NOT TELL ANY BODY THE CONTENT OF THIS DEPOSIT IN THE SECURITY COMPANY.I AM TELLING YOU THE CONTENT REASON THAT I WANT YOU TO ASSIST ME USE THE FUND FOR THE WORK OF GOD. EVEN DO YOU ARE NOT A CHRISTAIN.THAT IS NOT A PROBLEM .WHAT I WANT IS FOR YOU TO USE IT AND HELP THE HELPLESS PEOPLE AROUND YOU .TO HELP THE ORPHANAGES, WIDOWS, AND MOTHERLES CHILDRENS

RECENTLY, MY DOCTOR TOLD ME THAT I HAVE SERIOUS SICKNESS WHICH IS CADIAC PROBLEM.THE ONE THAT DISTURBS ME MOST IS MY STROKE SICKNESS HAVING KNOWN MY CONDITION I DECIDED TO DONATE HIS FUND TO YOU TO UTILIZE THIS MONEY ACCORDING TO MY DIRECTION AND THE WILL OF GOD.

PLEASE DO GIVE URGENT RESPONSE TO THIS MAIL WITHOUT ANY DELAY.

I WANT TO GIVE YOU NUMBER TO CALL ME BUT I DONT WANT IN A WAY MY HUSBAND RELATIONS WILL KNOW THAT I AM GIVING YOU THIS MONEY. I HAVE SISTER NURSE WHO IS FEARFUL TO THE LORD THAT WILL BE HELPING GIVING YOU INFORMATION OF THIS DEPOSIT.HER NAME ID SISTER (CHANTAL KONE )

SO PLEASE I AM WAITING FOR YOUR URGENT REPLY SO THAT I CAN GIVE YOU ALL THE INFORMATION ABOUT THIS MONEY AND THE SECURITY COMPANY WERE IT WAS DEPOSITED BY MY LATE HUSBAND ,

REMAIN BLESSED ALWAYS
YOURS SISTER IN CHRIST
MRS LINDA FASTUS SCHWARZ

# Classification des attaques(4)

- Canaux cachés (covert channels) (confidentialité)
  Communiquer (*high* → *low*) par des moyens non-surveillés
  - Canaux de stockage (ou canaux mémoires)
  - Canaux temporels
  - Autres : stéganographie, modulation analogique, canaux de fuite indirects, . . .
- Canaux de fuite (side channels) (confidentialité)
  Obtenir des informations cachées (*high*) de façon détournée
  - Exemples avec les cartes à puce : analyse de la consommation de courant d'alimentation (simple SPA, différentielle DPA)
  - Captation : microscope à balayage, micro-sondes, rayonnement électromagnétique, . . .
  - Injection de faute : micro-sondes, impulsions électro-magnétiques (y compris lumineuses), rayonnement nucléaire, . . .
  - Analyse logicielle (fuites, analyse temporelle)

# Classification des attaques(5)

- Porte dérobée (Trapdoor / Backdoor)
  (confidentialité, intégrité, disponibilité)
  Contourner les mécanismes de protection
  - Authentification (Turing Award de Ken Thompson), autorisation
  - Exemple : *l'œuf du coucou*, Clifford Stoll, 1986
  - *Rootkits*
    - Utilisation d'une porte dérobée pour devenir *root* (escalade de privilège)
    - Modification du noyau, appels systèmes ou commandes (ps, w, netstat, . . . )
    - Installation d'une porte dérobée pour un accès plus facile (ex. à distance)
    - Installation de logiciels malveillants, invisibles au niveau utilisateur

# Classification des attaques(6)

- *Spyware* (confidentialité, disponibilité)
  80% des PC professionnels infectés (estimation Webroot août 2005)
  - *Keyloggers*, *screen grabbers*, *sniffers*, analyse de fichiers, . . .
  - Installés par des vers, des pages Webs minées, du *phishing*, freeware, . . .
  - Octobre 2005 : Intermix paie 7,5 $m^{ns}$ de dollars pour un retrait de plainte pour *spyware*
- *Spyware légitime* ?
  - Rapports d'anomalies
  - Mises à jour automatiques
  - Vérification de versions (anti-piratage)
  - Détection de tricherie (World of Warcraft)

# Classification des attaques(7)

- Bombe logique (confidentialité, intégrité, disponibilité)
  Déclencher des dégâts sur un événement particulier
  - Divulgation d'information confidentielle (ex. virus SirCam)
  - Destruction, modification de données/programmes (disques, audit)
  - Diffusion de fausses informations (ex. diagnostic)
  - Dégâts matériels (ex. virus Tchernobyl)
  - Installer un *zombie*, *spammer*, *spyware*, . . .
- Logiciels malveillants (confidentialité, intégrité, disponibilité)
  (*malware* / maliciels : *rootkits*, *zombies*, . . . )
  - Furtivité (*stealth*)
  - Escalade de privilèges (jusqu'à *root*)
  - Installation de portes dérobées, de bombes logiques, de *spyware*, . . .
  - Pages Web *minées* (→ firewalls)
  - Exécution de scripts HTTP

→ Correction des failles pour protéger le zombie contre d'autres pirates !

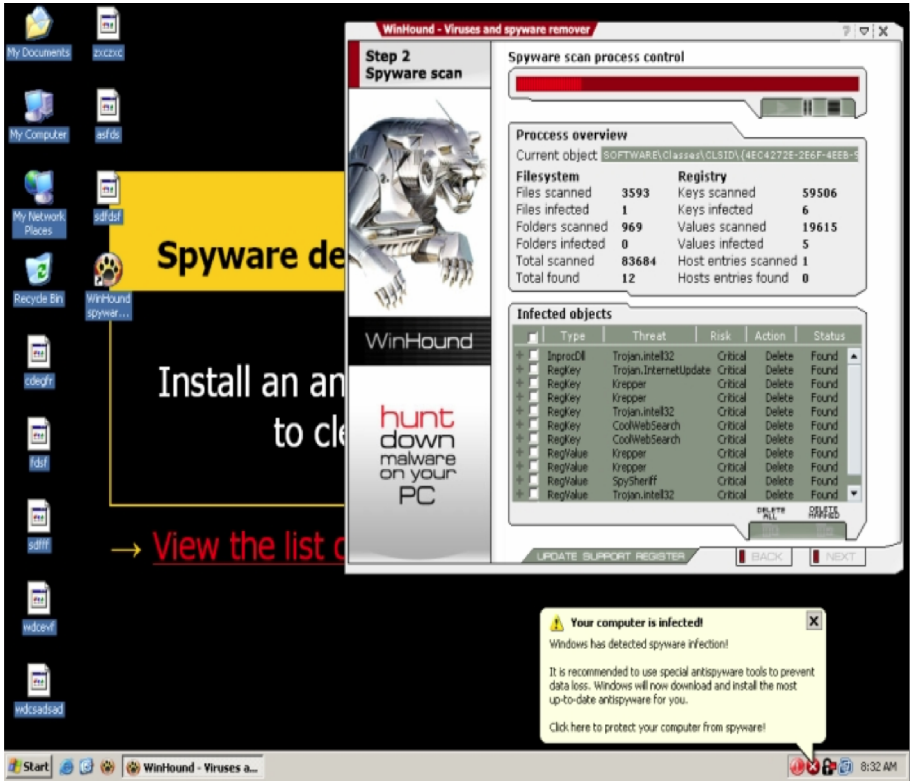
# Exemple de pages Web minées

- Décembre 2005 : 2 *zero-day exploits* (MS05-054 et MS06-001) sur plus de 1500 sites Web : *Broad Proliferation of Crimeware Sites Exploiting WMF Image-Handling Vulnerabilities*

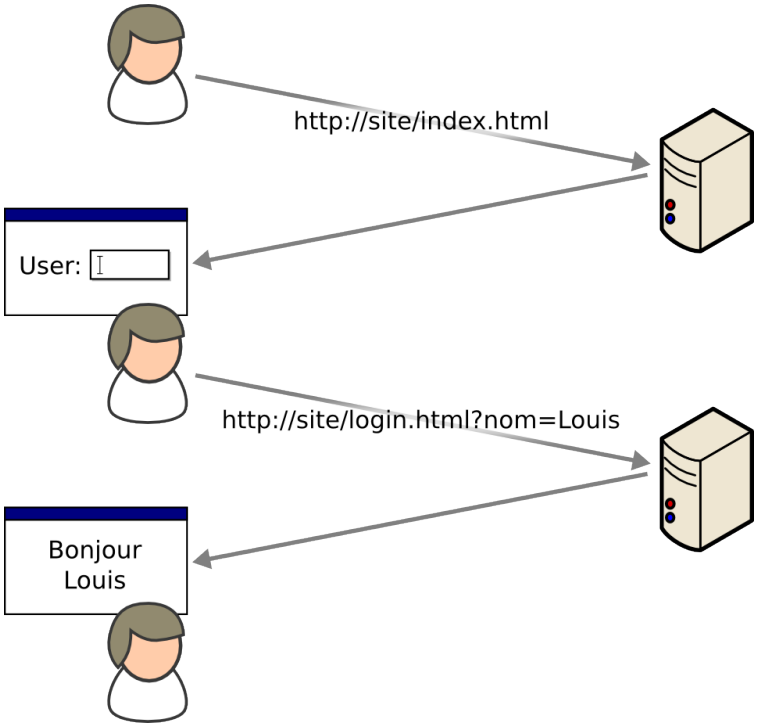
  *The websites . . . are using the exploit to distribute Spyware applications and other Potentially Unwanted Software. The user's desktop background is replaced with a message warning of a spyware infection and a "spyware cleaning" application is launched. This application prompts the user to enter credit card information in order to remove the detected spyware. . . In addition, a mail relay is installed on the infected computer and it will begin sending thousands of SPAM messages.*

  `www.antiphishing.org`

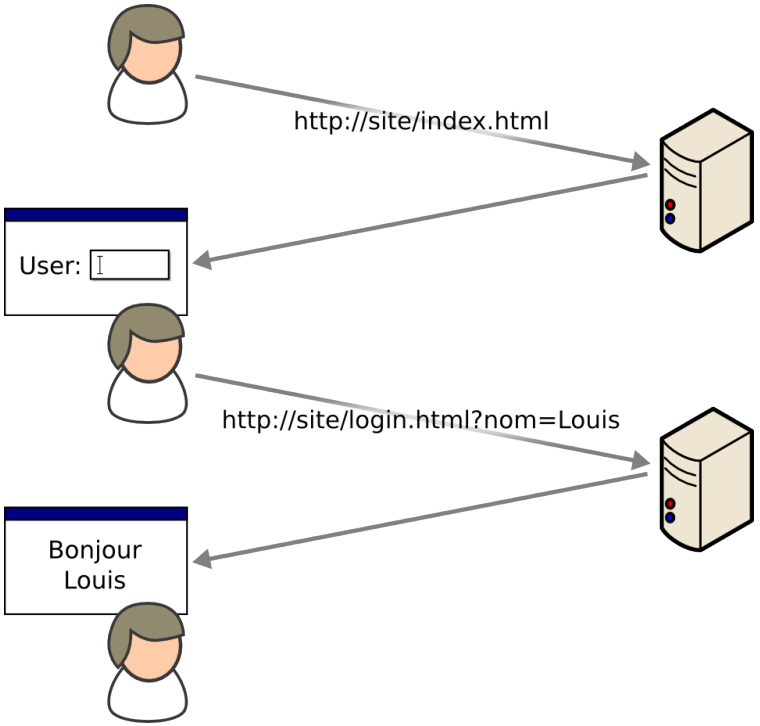
# Exemple de pages Web minées

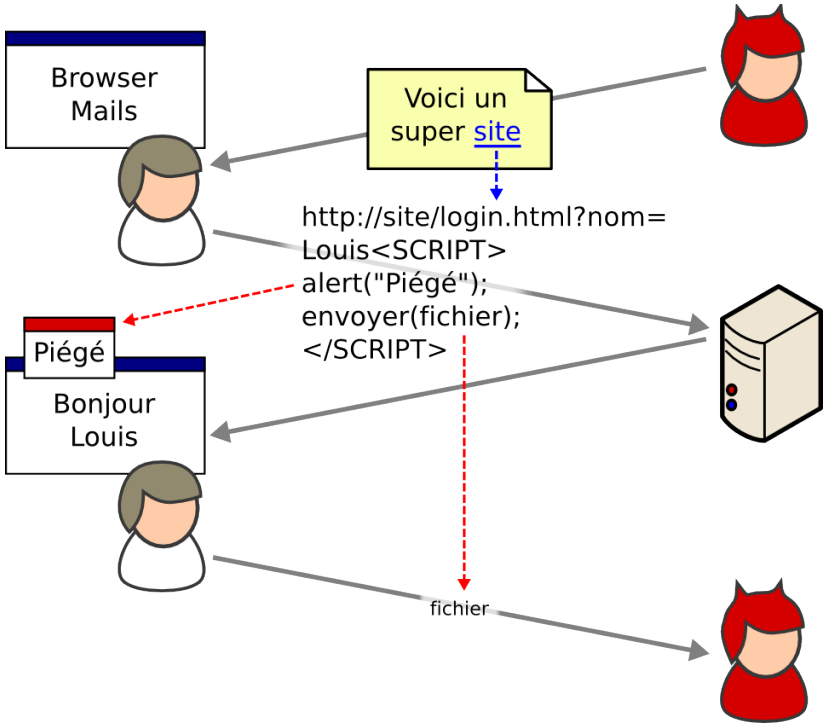



# Le *Cross Site Scripting* (*XSS*)



# Le *Cross Site Scripting* (*XSS*)




`www.cert.org/advisories/CA-2000-02.html`

- Le pirate peut communiquer le lien soit par *phishing* soit indirectement via, par exemple, un *blog* ou *forum* d'un serveur innocent
- La victime lit le message avec un navigateur configuré pour permettre l'exécution de scripts

# Classification des attaques(8)

- Cheval de Troie (confidentialité, intégrité, disponibilité)
  Fonction illicite cachée dans un programme apparemment bénin
  - Divulgation (*SoBig*, *P2P-Winny*) ou modification d'information, bombe logique
  - Exemples : disquette AIDS (1989), PGPCoder (ransomware), pages web *minées*

Sur `www.hifocus.net`, le 05/03/05 :

*. . . , je suis allé télécharger winrar sur* `http://www.01net.com/telecharger/windows/Utilitaire/compression_et_decompression/fiches/2257.html` *. . . et j'ai mis une heure pour désinfecter le pc. Pour un logiciel payant, ce n'est quand même pas normal de choper un virus. Il y avait entre autres Trojan-Spy.Banker.EA . . .*

# Informations sur *Trojan.Banker.FA*

*Trojan.Banker.FA est un cheval de Troie voleur de mots de passe prenant pour cible les clients d'une banque brésilienne.*

*Trojan.Banker.FA surveille l'accès Internet de l'utilisateur. Lorsque certains sites bancaires sont visités sur Internet, le cheval de Troie affiche un faux écran de connexion afin de tromper l'utilisateur pour qu'il saisisse ses détails. Trojan.Banker.FA transmet ensuite les détails qu'il a subtilisés à une adresse électronique brésilienne.*

*Trojan.Banker.FA peut aussi télécharger et installer d'autres logiciels associés. Lorsqu'il est exécuté, Trojan.Banker.FA se copie dans le dossier système Windows sous le nom de* `CARTAO.EXE` *et, pour être exécuté au démarrage du système, paramètre l'entrée de registre suivante :*

`HKCU\Software\Microsoft\Windows\CurrentVersion\Runcartao<système>\cartao.exe`

# *Phishing* utilisé pour télécharger un cheval de Troie

*In July, Websense® Security Labs discovered a new malicious website, which distributed malicious code that installs a Trojan Horse on end-users' machines. This potentially occurs without user interaction.*
*The site appeared to be mirroring a World Cup 2006 Soccer website with the exception that they have a lead story regarding the now infamous, Zinedine Zidane head butt incident from the World Cup final against Italy.*
*Upon visiting any of the pages on the site, end-users were potentially infected with a Trojan Horse downloader. This Trojan Horse downloads additional payload code from the site. The site was using the underground "Web Attacker" toolkit (discussed in an earlier alert* `http://www.websense.com/securitylabs/alerts/alert.php?AlertID=472` *).*
*The Web Attacker toolkit is sold on a Russian website and costs anywhere from $20 to $300. This toolkit allows users to install code that exploits users based on their browser types. The installed code includes one of five different variants, including exploits for old and new vulnerabilities.*
*This site was hosted in the United States.*

`www.antiphishing.org` – *Juillet 2006*

# Exemple de pages Web minées

FIFA WORLD CUP GERMANY 2006

BERNAMA WORLD CUP 2006 SPECIAL PAGE

Main | News List | Match Schedule | Results

**World Cup 2006 Top Story**

## What did Materazzi say to Zidane?

**PARIS - *The Zinedine Zidane mystery is not quite solved yet.***

In his first, highly awaited comments since the World Cup final, the French soccer star only partly explained what caused him to react in fury and head-butt an Italian opponent: repeated harsh insults about his mother and sister.

But Zidane didn't go into specifics about what Marco Materazzi said. Materazzi swears he never insulted Zidane's mother. And FIFA is still investigating.

**FIFA World Cup 2006 Champion**
Italy

**Second Place**
France

**Third Place**
Germany

**Fourth Place**
Portugal

**Teams that did not qualify**
Brazil
England
Ukraine
Argentina
Spain
Ghana
Switzerland
Australia
Netherlands
Ecuador
Mexico
Sweden
Poland
Costa Rica
Paraguay

# Classification des attaques(9)

- Virus (confidentialité, intégrité, disponibilité)
  Segment de code qui, lorsqu'il est exécuté, se reproduit en s'attachant à un autre programme (système ou application), éventuellement porteur d'une bombe logique
  - Période d'incubation
  - Seulement sur des fichiers de programme
  - Propagation par échange de support ou par réseau
  - Exemples : *Brain*, *Vendredi 13*, *macrovirus*, etc.
- Ver (*worm*) (confidentialité, intégrité, disponibilité)
  Programme autonome, capable de se répliquer et de se propager, éventuellement porteur d'une bombe logique
  - Mail (sendmail, outlook, majordomo, etc.) débordement de buffer (IIS, SQL Server, LSASS, etc.), Instant Messaging (IRC, AOL, Yahoo !, MSN, etc.), P2P, etc.
  - Exemples : Xerox, CHRISMAS, Robert T. Morris, ILOVEYOU, Code Red, Slammer, etc.

# Classification des attaques(10)

- Déni de service (*DoS* : *denial of service*) (disponibilité)
  Empêcher les utilisateurs légitimes d'accéder aux informations ou aux services auxquels ils ont droit
  - *flooding*, *smurfing* (*ICMP echo requests*), DDoS (par *botnets*)
    - Ver de Morris (novembre 1988)
    - DDoS : février 2000 (Amazon, CNN, eBay, etc.), octobre 2002 (DNS), juin 2004 (Akamai), février 2007 (DNS),
    - Février 2006 : un espagnol condamné à 2 ans de prison et 1,4 $m^{ns}$ d'euros pour un DDoS ayant affecté 1/3 des utilisateurs d'ISP espagnols en 2003 (vengeance contre son exclusion d'un IRC)
    - Octobre 2006 : 4 russes condamnés à 8 ans de prison pour avoir extorqué plus de 4 $m^{ns}$ de dollars à des casinos et bookmakers sous menace de DDoS
  - *Spamming*
    - Escror britannique aux noms de domaines : 1,6 $m^{ns}$ de livres détournés, mille à 5 $m^{ns}$ de mails par victimes, 6 ans de prison
    - Un spammer condamné à payer 5,6 $m^{ns}$ de dollars à AOL

# Classification des attaques(10)

- Attaques complexes
  - *Spam*/*Blogs* → Vers/Chevaux de Troie → Botnets (IRC, P2P, etc.) → *spamming*, *phishing*, DDoS, chantage, serveur illégal, etc.
    - Octobre 2005 : 3 Hollandais arrêtés pour avoir pris le contrôle de 1.500.000 machines (zombies) avec un ver
    - Mai 2006 : 1 californien de 20 ans, Jeanson James Ancheta, condamné à 57 mois de prison pour avoir créé 400.000 zombies (contrôlés par IRC) et s'être fait payer 100.000 dollars par des sites publicitaires
    - En 2007, *Storm Worm* : estimation : 1 à 50 $m^{ns}$ de zombies, utilisés pour spam (record : 57 $m^{ns}$ le 22/08/07) + DDoS de sites de spam, anti-spam ou de contre-mesures `www.schneier.com/blog/archives/2007/10/the_storm_worm.html`
  - Attaques ciblées (harponnage, *spear phishing*) : un cheval de Troie spécifique, difficile à détecter, visant une compagnie (exemple : espionnage industriel en Israël), ou une agence gouvernementale particulière, etc.
    Autre exemple : cibler une banque particulière : Auto-*XSS* + *session hijackiing*

# Exemple d'harponnage (1)

**December 4, 2007**

**MI5 Sounds Alarm on Internet Spying from China**

Someone in MI5 is pissed off at China:

> In an unprecedented alert, the Director-General of MI5 sent a confidential letter to 300 chief executives and security chiefs at banks, accountants and legal firms this week warning them that they were under attack from "Chinese state organisations."
>
> [...]
>
> Firms known to have been compromised recently by Chinese attacks are one of Europe's largest engineering companies and a large oil company, The Times has learnt. Another source familiar with the MI5 warning said, however, that known attacks had not been limited to large firms based in the City of London. Law firms and other businesses in the regions that deal even with only small parts of Chinese-linked deals are being probed as potential weak spots, he said.
>
> A security expert who has also seen the letter said that among the techniques used by Chinese groups were "custom Trojans", software designed to hack into the network of a particular firm and feed back confidential data. The MI5 letter includes a list of known "signatures" that can be used to identify Chinese Trojans and a list of internet addresses known to have been used to launch attacks.
>
> A big study gave warning this week that Government and military computer systems in Britain are coming under sustained attack from China and other countries. It followed a report presented to the US Congress last month describing Chinese espionage in the US as so extensive that it represented "the single greatest risk to the security of American technologies."

www.schneier.com/blog/archives/2007/12/mi5_sounds_alar.html

# Exemple d'harponnage (2)

Publié le 04/01/2008 à 08:28 Le Point.fr

## L'armée sud-coréenne subit des attaques informatiques

Guerric Poncet

Le ministère de la Défense de la Corée du Sud est en ébullition : des soldats ont subi des attaques informatiques sur leurs ordinateur et des pirates ont réussi à voler des données personnelles. Selon le porte-parole du ministre, cité par l' AFP , "aucune donnée militai n'a été dérobée". Mais rien ne permet d'en être sûr, les soldats ayant pu violer les règles de sécurité informatique en enregistrant des données confidentielles sur leurs postes personnels.

Grâce à un e-mail, intitulé avec intelligence "État actuel des capacités militaires de la Corée du Nord" et infecté par un cheval de Troie (programme malicieux destiné à "ouvrir les portes" de l'ordinateur), les pirates ont parfaitement réussi leur coup. Officiellement, l'attaque provenait "de l'étranger", sans plus de précisions. Plus bavarde, la presse ne cesse de parler de la Chine, qu développe une véritable force d'attaque cybernétique depuis plusieurs années. En novembre 2006, un rapport du Congrès des États-Unis s'inquiétait de la formation par Pékin d'unités de combat informatique (NET Force), capables d'infiltrer n'importe quel ordinateur, de Wall Street au Pentagone. Un cri d'alarme à prendre toutefois avec des pincettes, dans un contexte où le Congrès hésitait à accorder des crédits supplémentaires pour la protection de la sécurité nationale.

Immédiatement, le ministère de la Défense sud-coréen a rappelé à ses troupes les règles élémentaires de la sécurité informatique : utiliser un antivirus à jour ainsi qu'un pare-feu, et ne pas ouvrir n'importe quel contenu. Dans ce pays où 70 % de la population utili Internet (contre 40 % en Europe et 12 % en Chine), cette affaire a créé un scandale et personne ne veut en rester là.

Toutes les structures étatiques du monde sont soumises à des attaques informatiques. Parfois l'intensité ou la gravité de celles-ci permettent de parler d'acte de guerre informatique. Par exemple, la mise à genoux de l'infrastructure informatique estonienne en 20 a été considérée comme le premier événement du genre. Les attaques répétées menées contre des soldats de l'armée sud-coréenne c derniers jours sont inquiétantes mais ne peuvent pour l'heure être qualifiées d'actes de guerre. Cela ressemble plutôt à un entraînement.

# Les gains finaciers

- Pour les pirates qui contrôlent un ordinateur
  - Utilisation de numéros de cartes de crédit
  - Chantage, extorsion de fonds, espionnage industriel, etc.
  - Spéculation en bourse : *pump and dump scams* (*spam*, VoIP), exemple : `www.investopedia.com/ask/answers/05/061205.asp`
  - Connexion à des lignes téléphoniques payantes
  - Accès à des comptes (banques, retraites, paypal, e-Bay, FAI, opérateurs téléphoniques, hotspots, etc.)
  - Vente d'adresses e-mails : exemple 28 000 dollars pour 92 $m^{ns}$ d'adresses mail (AOL)
  - Services payants (exemples : porno, films piratés, etc.) + spammers, etc.
  - *Click fraud* (relais de publicité) : exemple : 100 mille dollars avec 400.000 zombies
  - Location de botnets, spammers, etc.

# Les principales failles exploitables

`cwe.mitre.org/top25`

- API : Débordement de buffers, heap, stack (exemple : return-to-libc attacks), entiers, etc.
- API : pas de contrôle de type ou vérification de format insuffisante : SQL injection, PHP, etc.
- Utilisation non-prévue → Fuzzing
- Race conditions
- Contrôle d'origine insuffisant : *cross-scripting*, applets, plug-ins, extensions, certificats, etc.

# Sommaire

# Sommaire

# Différents volets de la sécurité

- Sécurité physique
  Protection des locaux contre incendie, inondation, etc.
  Contrôle des accès physiques
- Sécurité du personnel (pas la CHS)
  Règles liées aux conditions de travail pour les personnels internes (employés, intérimaires, stagiaires, etc.) et externes (visiteurs, maintenance, sous-traitants, etc.), y compris dans des circonstances particulières : embauche, départ, grève, etc.

# Différents volets de la sécurité

- Sécurité procédurale
  Procédures pour la gestion du SIC : enregistrement (et effacement) des utilisateurs, sauvegardes, maintenance, installation et mises à jour de matériels et de logiciels, etc.
- Sécurité technique
  C'est tout ce qu'on va voir maintenant.

# Sommaire

# Terminologie

- Cryptologie = cryptographie + cryptanalyse
  - Cryptographie, du grec *kruptos* (caché) et *graphein* (écrire)
    Ecrire des messages incompréhensibles par des tiers
  - Cryptanalyse
    Découvrir le(s) secret(s), décrypter
- A ne pas confondre avec stéganographie
  - Du grec *stegano* (dissimuler)
  - Encre sympathique
  - Filigranes (tatouages)
- Chiffre, chiffrement (pas chiffrage ni cryptage), déchiffrement, clair, cryptogramme

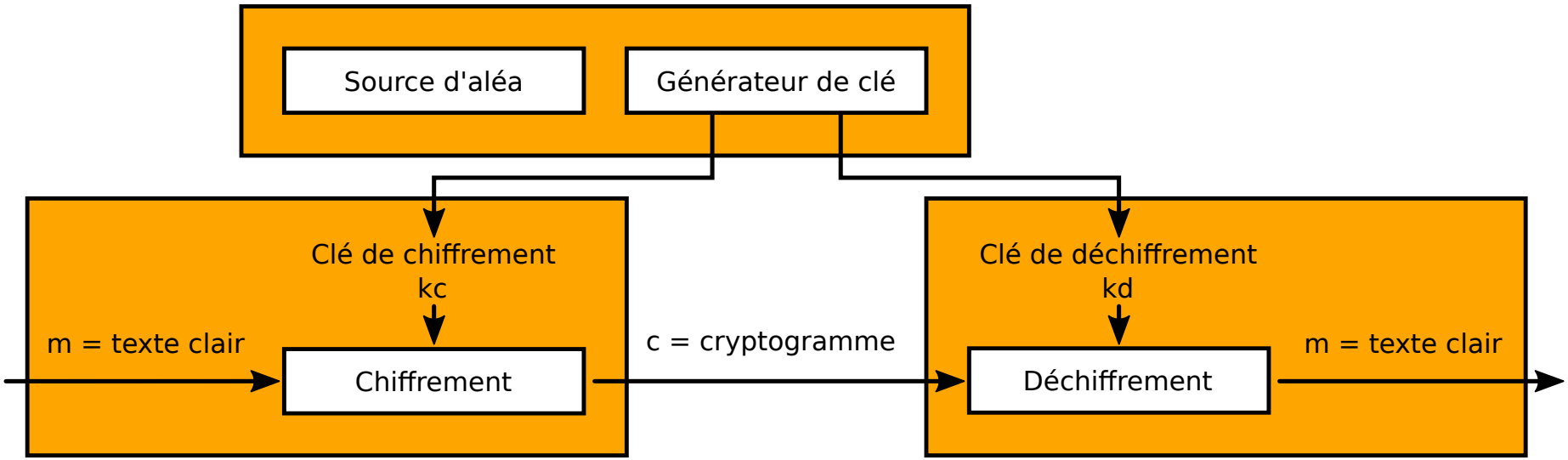
# Propriétés couvertes par la cryptographie

- **Confidentialité** de l'information
  Exemple : écoute passive
- **Integrité** / **authenticité** de l'information
  Exemple : homme dans le milieu
- **Authentification** des entités
  Exemple : déguisement
- **Non-prépudiation** d'origine et de destination
  Exemple : preuves, matériel juridique

# Définition fondamentales et notations

- **Clair**, $m \in M$ : message non chiffré, l'information est accessible
- **Chiffré**, $c \in C$ : message chiffré ou cryptogramme, l'information n'est pas accessible
- **Clé**, $k \in K$ : secret indispensable pour transformer un clair en chiffré ou un chiffré en clair. On parle respectivement de clé de chiffrement et de clé de déchiffrement
- **Générateur de clé** : génération des clés
- **Chiffrement** $\{\}$ ou $E()$ : transformation d'un clair en chiffré pour une clé de chiffrement donnée
- **Déchiffrement** $[]$ ou $D()$ : transformation d'un chiffré en clair pour une clé de déchiffrement donnée

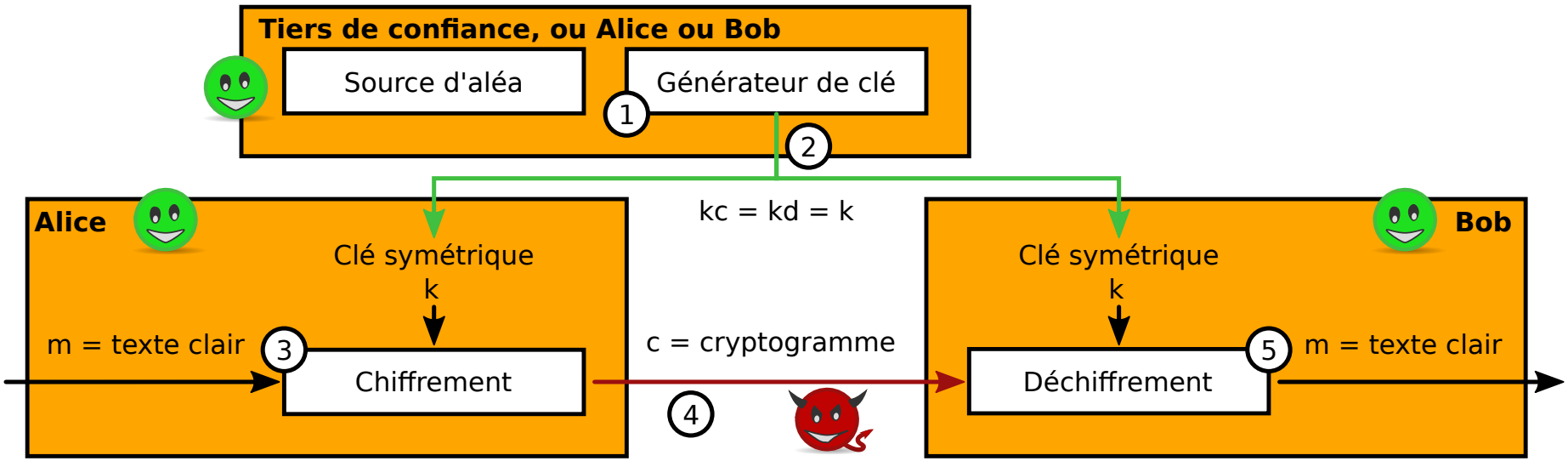
# Constructions fondamentales : chiffrement




## Notation

- Chiffrement : $C = \{M\}_{k_c}$ ou $C = E_{k_c}(M)$
- Déchiffrement : $M = [C]_{k_d}$ ou $M = D_{k_d}(C)$

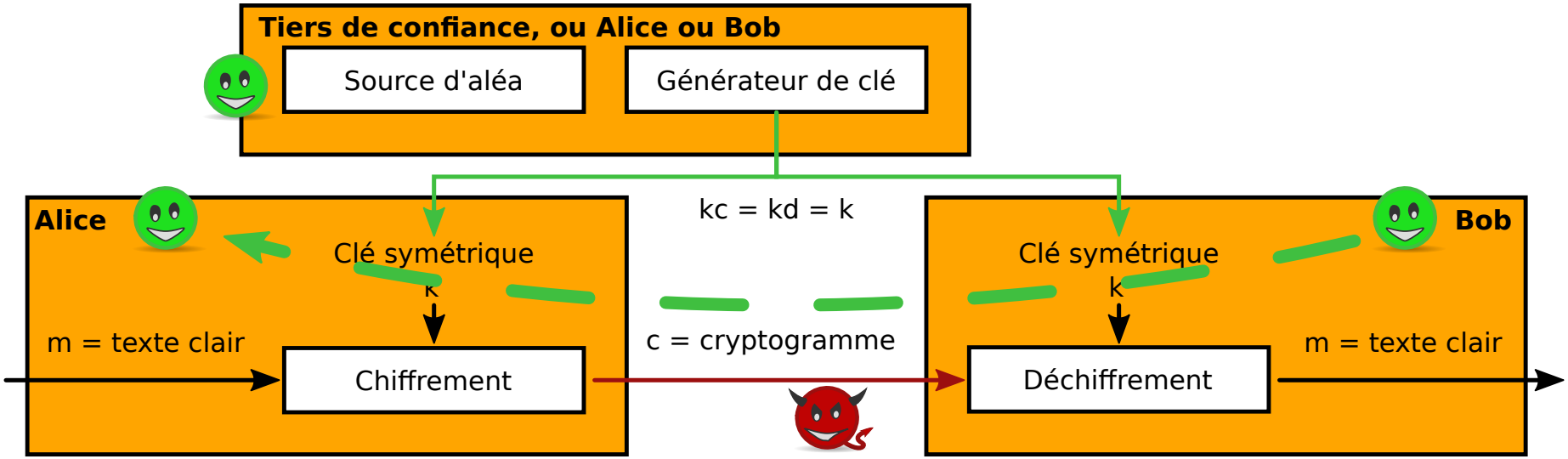
# Constructions fondamentales : chiffrement symétrique 1/2




## Procédure

1. Alice ou Bob génère une clé secrète unique : $K$
2. Distribution de la clé à l'aide d'un canal sécurisé
3. Alice chiffre le message avec la clé secrète $K$
4. Le message est transmis au travers d'un canal non sécurisé
5. Bob déchiffre le message avec la clé secrète $K$

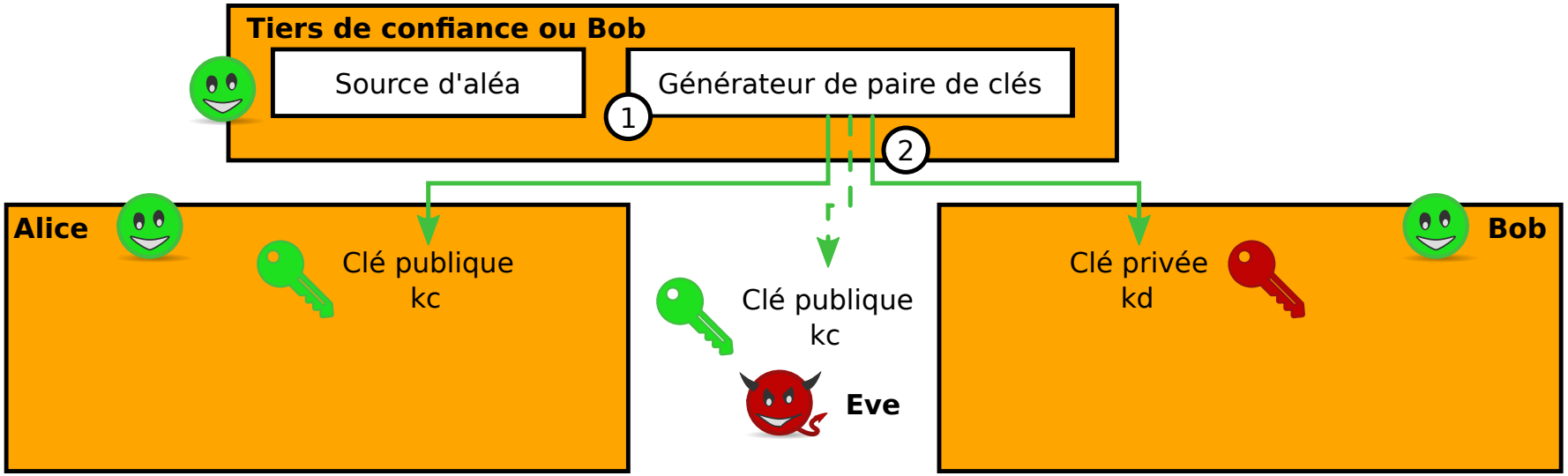
# Constructions fondamentales : chiffrement symétrique 2/2




## Propriétés

- $k_c = k_d = K$
- Authentification de l'origine
- $M$ confidentiel

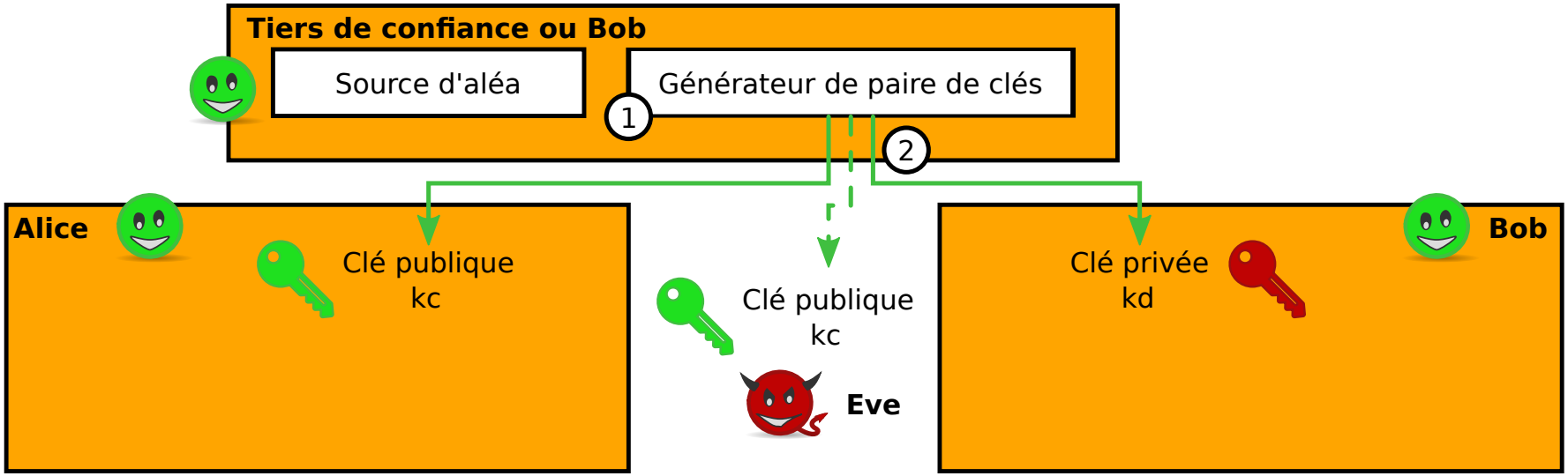
# Constructions fondamentales : chiffrement asymétrique 1/6




## Procédure : distribution des clés

1. Bob génère une paire de clés unique : $(k_c, k_d)$
2. Distribution de $k_d$ à Bob l'aide d'un canal sécurisé
3. Distribution de $k_c$ à Alice et au monde

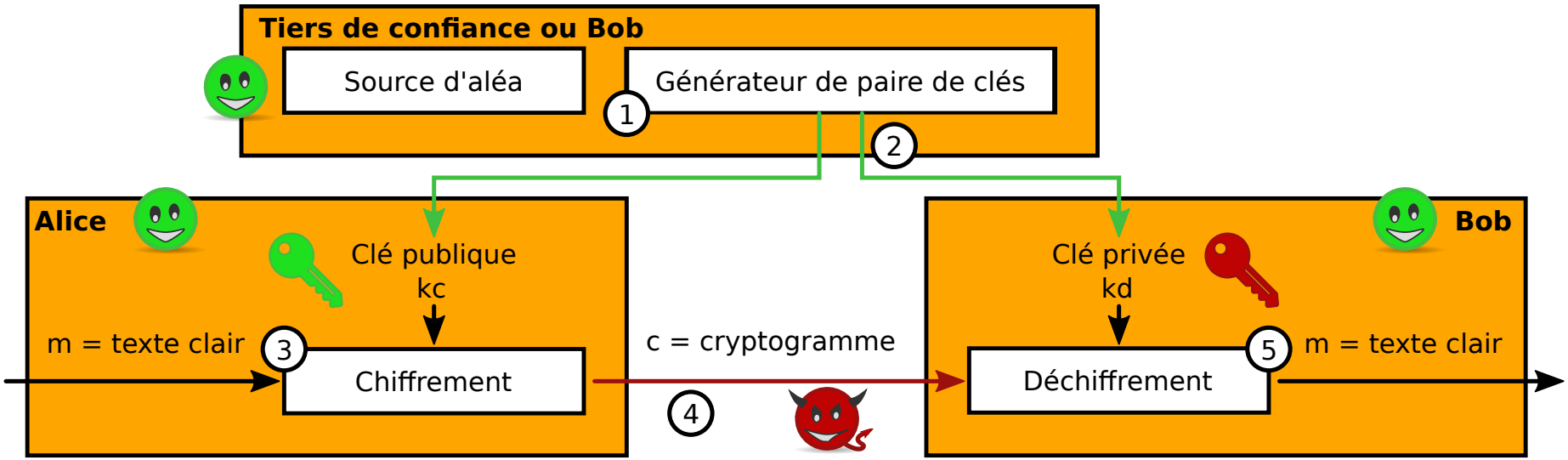
# Constructions fondamentales : chiffrement asymétrique 2/6




## Propriétés chiffrement asymétrique

- $k_c \neq k_d$
- $\exists$ une unique paire $(k_c, k_d) | M = D_{k_d}(E_{k_c}(M))$
- $k_c$ est connue à la fois d'Alice et Bob, mais aussi de l'attaquant Eve

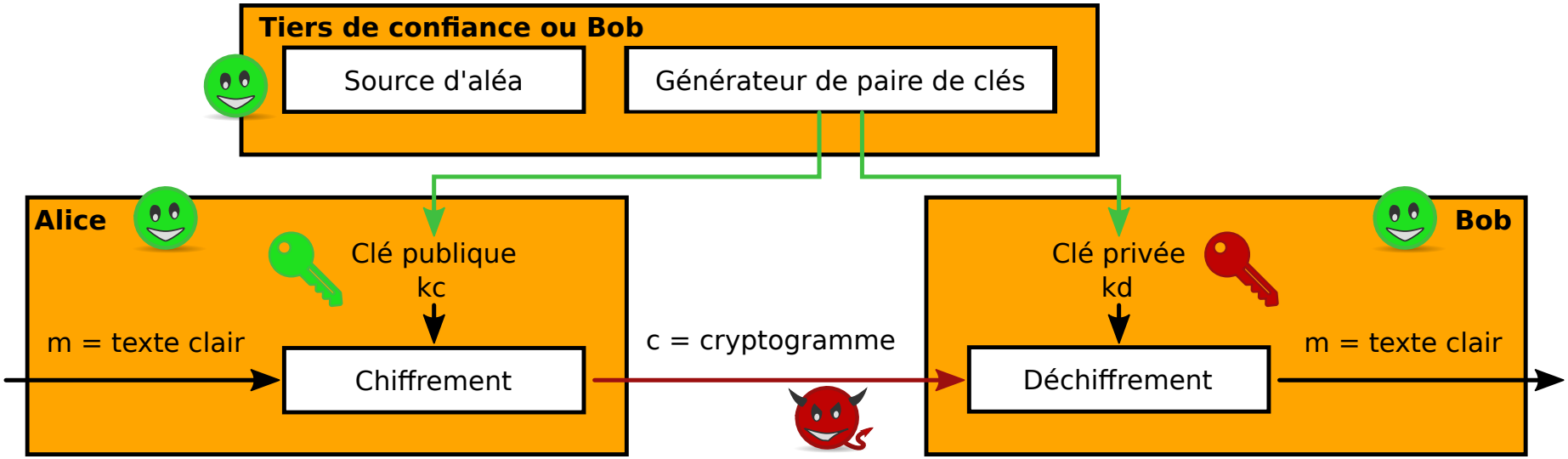
# Constructions fondamentales : chiffrement asymétrique 3/6




## Procédure : chiffrement $k_c \rightarrow k_d$

3. Alice chiffre le message avec la clé publique $k_c$
4. Le message est transmis au travers d'un canal non sécurisé
5. Bob déchiffre le message avec la clé secrète $k_d$

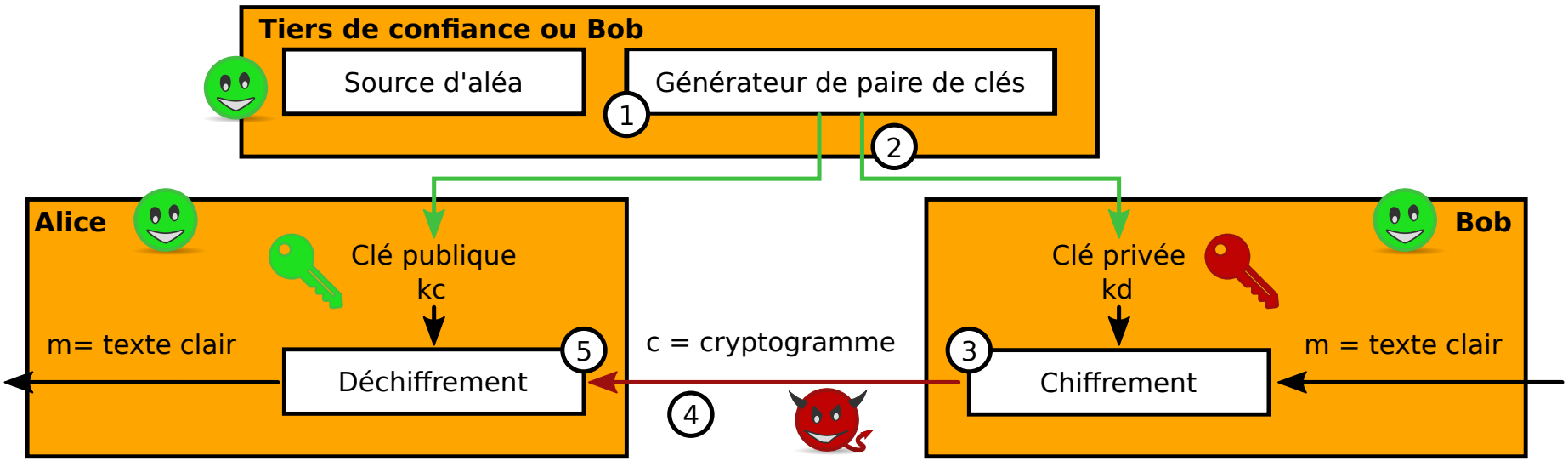
# Constructions fondamentales : chiffrement asymétrique 4/6




## Propriétés chiffrement $k_c \rightarrow k_d$

- $M$ confidentiel

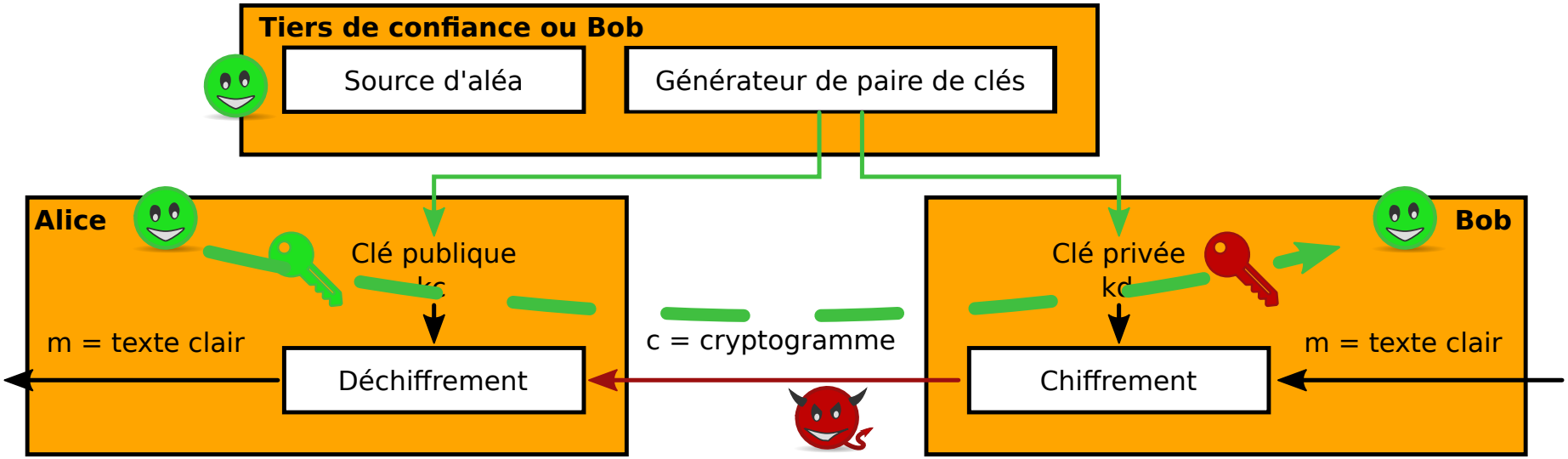
# Constructions fondamentales : chiffrement asymétrique 5/6




## Procédure : chiffrement $k_c \rightarrow k_d$

3. Bob chiffre le message avec sa clé privée $k_d$
4. Le message est transmis à Alice et au monde au travers d'un canal non sécurisé
5. Alice et le monde déchiffre le message avec la clé secrète $k_d$

# Constructions fondamentales : chiffrement asymétrique 6/6




## $k_c \rightarrow k_d$ : propriétés

- Authentification de l'entité Bob : seul Bob peut calculer $E_{k_d}(M)$
- $M$ non confidentiel

# Notions de déterminisme et d'aléa

## Chiffrement et déchiffrement

- Algorithmes déterministes
- Fonctions : $\exists$ une seule image y $\forall$ antécédent x de l'algorithme
- Propriété de cohérence : $M = D_{k_d}(E_{k_c}(M))$

## Algorithmes aléatoires

- $\exists$ plus d'une image y $\forall$ antécédent x de l'algorithme
- Soit $A$ l'ensemble des sorties possibles pour un algorithme
- Distribution, $\mathcal{D}$ : on associe une probabilité d'occurrence à chaque élément de $A$ ($\Omega$)
- Distribution uniforme : $\forall y \in A, \mathcal{D}(y) = 1/|A|$

# Conception d'un bon algorithme de chiffrement symétrique

## Fonctions attendues

- Primitive de chiffrement
- Primitive de déchiffrement
- Propriété de cohérence

## Sécurité d'un algorithme de chiffrement symétrique

- Sans connaître $k_d$, il doit être "impossible" de retrouver $M$

→ Le chiffré ne doit révéler aucune information sur le clair ni le chiffré

- Il doit être "impossible" de trouver $k_d$, même connaissant $C$ et $M$
- Il doit être "impossible" de trouver $k_d$, même choisissant $M$

## Étude d'un exemple

*One Time Pad* (OTP)

# Propriétés du ou exclusif $\oplus$

Ou exclusif ou addition binaire (sans la retenue)

## Table de vérité

| $x$ | $y$ | $x \oplus y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

## Propriétés

1. $x \oplus 0$
2. $x \oplus x$
3. $y \oplus x \oplus x$

# Propriétés du ou exclusif $\oplus$

Ou exclusif ou addition binaire (sans la retenue)

## Table de vérité

| $x$ | $y$ | $x \oplus y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

## Propriétés

1. $x \oplus 0 = x$
2. $x \oplus x$
3. $y \oplus x \oplus x$

# Propriétés du ou exclusif $\oplus$

Ou exclusif ou addition binaire (sans la retenue)

## Table de vérité

| $x$ | $y$ | $x \oplus y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

## Propriétés

1. $x \oplus 0 = x$
2. $x \oplus x = 0$
3. $y \oplus x \oplus x$

# Propriétés du ou exclusif $\oplus$

Ou exclusif ou addition binaire (sans la retenue)

## Table de vérité

| $x$ | $y$ | $x \oplus y$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

## Propriétés

1. $x \oplus 0 = x$
2. $x \oplus x = 0$
3. $y \oplus x \oplus x = y$

# Un candidat : le *One Time Pad* (OTP, Vernam, 1917)

Masque jetable en français

## Définition

- $M = C = K = \{0,1\}^n$
- Chiffrement : $c = E_k(m) = k \oplus m$
- Déchiffrement : $m = D_k(c) = k \oplus c$
- $E_k \Leftrightarrow D_k$

## Propriétés

- Cohérent
- Performant, mise en œuvre simple

## Contrainte

$|\mathrm{K}| = |\mathrm{M}|$ ...

# Sécurité du *One Time Pad*

## Propriété de sécurité 1

$M = K = C = \{0,1\}^n$
$c_i \in c$ à la position $i$
$P[c_i = 1]$ ?

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

# Sécurité du *One Time Pad*

## Propriété de sécurité 1

$M = K = C = \{0,1\}^n$
$c_i \in c$ à la position $i$
$P[c_i = 1]$ ?

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

1. $P[c_i = 1] = P[k_i = 0 \cap m_i = 1] + P[k_i = 1 \cap m_i = 0]$

# Sécurité du *One Time Pad*

## Propriété de sécurité 1

$M = K = C = \{0,1\}^n$
$c_i \in c$ à la position $i$
$P[c_i = 1]$ ?

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

1. $P[c_i = 1] = P[k_i = 0 \cap m_i = 1] + P[k_i = 1 \cap m_i = 0]$
2. $P[c_i = 1] = P[k_i = 0] \times P[m_i = 1] + P[k_i = 1] \times P[m_i = 0]$

# Sécurité du *One Time Pad*

## Propriété de sécurité 1

$M = K = C = \{0,1\}^n$
$c_i \in c$ à la position $i$
$P[c_i = 1]$ ?

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

1. $P[c_i = 1] = P[k_i = 0 \cap m_i = 1] + P[k_i = 1 \cap m_i = 0]$
2. $P[c_i = 1] = P[k_i = 0] \times P[m_i = 1] + P[k_i = 1] \times P[m_i = 0]$
3. $P[c_i = 1] = 1/2 \times 1/2 + 1/2 \times 1/2 = 1/2$

# Sécurité du *One Time Pad*

## Propriété de sécurité 1

$M = K = C = \{0,1\}^n$
$c_i \in c$ à la position $i$
$P[c_i = 1]$ ?

```
mi : 0 1 0 1
ki : 0 0 1 1
-----------
ci : 0 1 1 0
```

1. $P[c_i = 1] = P[k_i = 0 \cap m_i = 1] + P[k_i = 1 \cap m_i = 0]$
2. $P[c_i = 1] = P[k_i = 0] \times P[m_i = 1] + P[k_i = 1] \times P[m_i = 0]$
3. $P[c_i = 1] = 1/2 \times 1/2 + 1/2 \times 1/2 = 1/2$

$\Rightarrow P[c_i = x]$ est uniforme

# Sécurité du *One Time Pad*

## Propriété de sécurité 2

$M = K = C = \{0,1\}^2$

```
m : 0 1 | 0 0 | 1 1 | 1 0
k : 0 0 | 0 1 | 1 0 | 1 1
-------------------------
c : 0 1 | 0 1 | 0 1 | 0 1
```

# Sécurité du *One Time Pad*

## Propriété de sécurité 2

$M = K = C = \{0,1\}^2$

```
m : 0 1 | 0 0 | 1 1 | 1 0
k : 0 0 | 0 1 | 1 0 | 1 1
-------------------------
c : 0 1 | 0 1 | 0 1 | 0 1
```

Pour un chiffré donné, tout clair peut être un antécédent

# Sécurité parfaite 1/2

Pour $n \in \mathbb{N} \Rightarrow |\mathrm{K}| = |\mathrm{M}| = |\mathrm{C}|$ de taille quelconque

## Définition

$\forall m, \in M, \forall c \in C$ on a $P[M = m|C = c] = Pr[M = m]$

- L'attaquant n'apprend rien du chiffré
- Une attaque utilisant seulement le chiffré est impossible

## Stragégie de preuve, intuition

Montrer que $P[M = m]$ et $P[C = c]$ sont indépendants
$\rightarrow P[M = m|C = c] \Leftrightarrow P[M = m \cap C = c]/P[C = c] = P[M = m] \times P[C = c]/P[C = c] = P[M = m]$

## Indiscernabilité parfaite

$\forall m_0, m_1 \in M, \forall c \in C$ on a $P[E_k(m_0) = c] = P[E_k(m_1) = c]$
avec $k \in K$ variable aléatoire

# Sécurité parfaite 2/2

## Intuition

(1) Pour $c \in C$ et $m \in M$ fixés, combien de clé peuvent chiffrer $m$ en $c$ ?

(2) Quel est le cardinal de $K$, $|K|$ ?

# Sécurité parfaite 2/2

## Intuition

(1) Pour $c \in C$ et $m \in M$ fixés, combien de clé peuvent chiffrer $m$ en $c$ ?
Réponse : 1

(2) Quel est le cardinal de $K$, $|K|$ ?
Réponse : $2^n$

# Sécurité parfaite 2/2

## Intuition

(1) Pour $c \in C$ et $m \in M$ fixés, combien de clé peuvent chiffrer $m$ en $c$ ?
Réponse : 1

(2) Quel est le cardinal de $K$, $|K|$ ?
Réponse : $2^n$

## Indiscernabilité parfaite

$P[E_k(m) = c] = (1)/(2)$
$P[E_k(m) = c] = 1/2^n = P[E_k(m_0) = c] = P[E_k(m_1) = c]$

# Limites 1/2

## Maléabilité

Soit $m \in M, c \in C, k \in K$
$m = E_k(m)$
Et après une attaque : $c_2|c_2 = c \oplus x$
$D_k(c_2) = c_2 \oplus k = c \oplus k \oplus x \oplus k = c \oplus x$
L'attaquant $\oplus$ directement le clair ! !

## Réutilisation de la clé impossible

Soit $m_1, m_2 \in M, c_1, c_2 \in C, k \in K$
$c_1 = E_k(m_1)$ et $c_2 = E_k(m_2)$
$c_1 \oplus c_2 = m_1 \oplus k \oplus m_2 \oplus k = m_1 \oplus m_2$
Ou exclusif des clairs ! !

# Limites 2/2

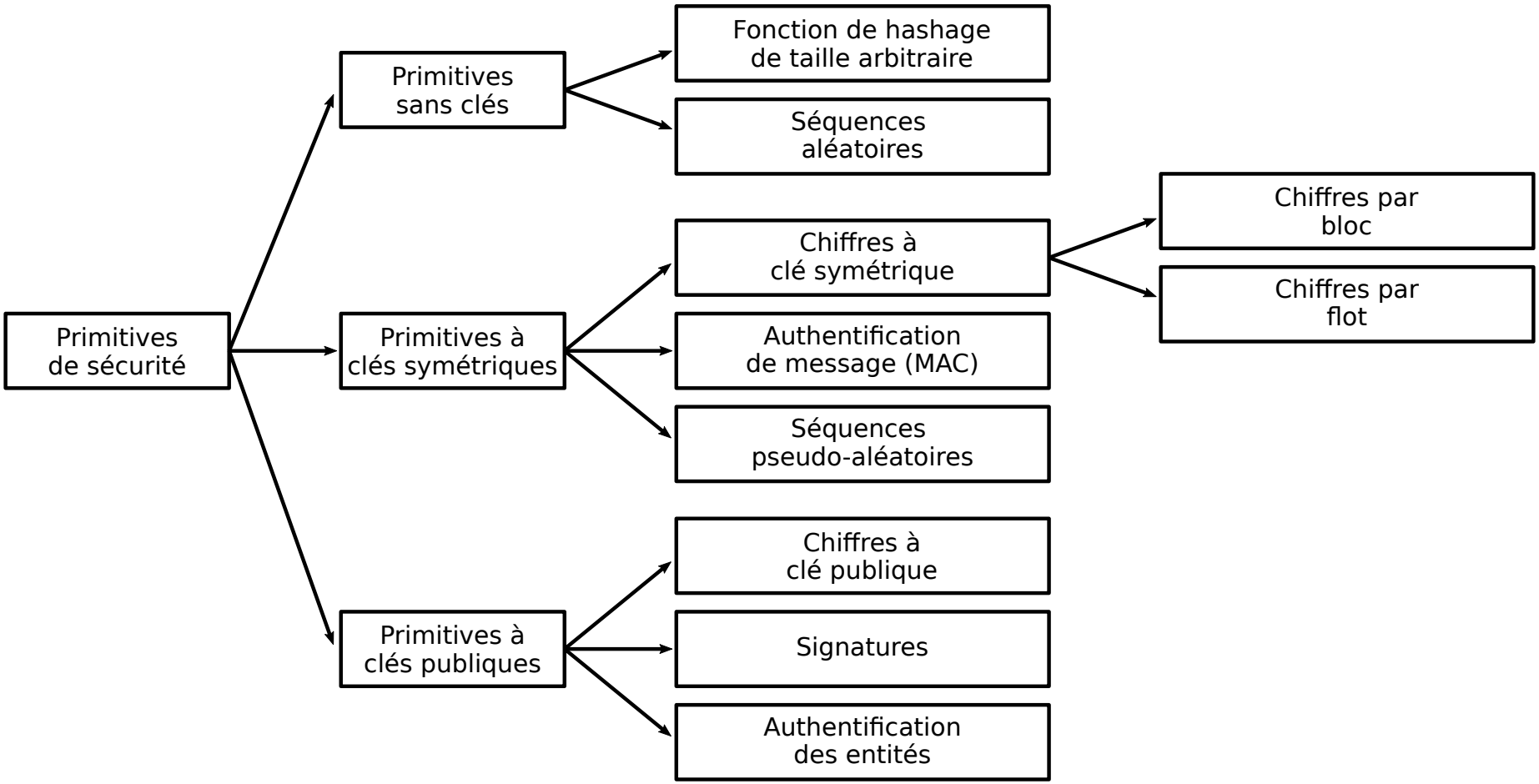
## Réutilisation de la clé impossible 2

Soit $m_1, m_2 \in M, c_1, c_2 \in C, k \in K$
$c_1 = E_k(m_1)$ et $c_2 = E_k(m_2)$
Si $m_1$ est connu par l'attaquant
$m_1 \oplus c_1 = m_1 \oplus m_1 \oplus k = k$
L'attaquant peut déchiffrer $m_2$

## $|K| >= |M|$

On montre que $P[M = m] = 1/2^n$

# Terminologie : *mindmap*




Nomenclature simplifiée des primitives de sécurité cryptographiques [5]

# Chiffrement symétriques : $k_c = k_d$

- Tous les chiffres connus jusqu'en 1976 !
- Chiffre par bloc
  - Texte découpé en blocs de taille fixe pour traitement
  - Souvent associé à un mode d'opération
  - Certains modes transforment en primitive par flot (CTR, CFB)
- Chiffre par flot
  - Génération indépendante de la clé
  - Puis application d'une fonction réversible sur le clair ($\oplus$)
  - Texte clair de taille arbitraire
- Exemples :

Bloc :

- DES (1976)
  Clés de 56 bits (plus 8 bits de parité)
  Blocs de 64 bits
- AES (2000)
  Clés de 128, 192, 256 bits
  Blocs de 128 bits

Flot :

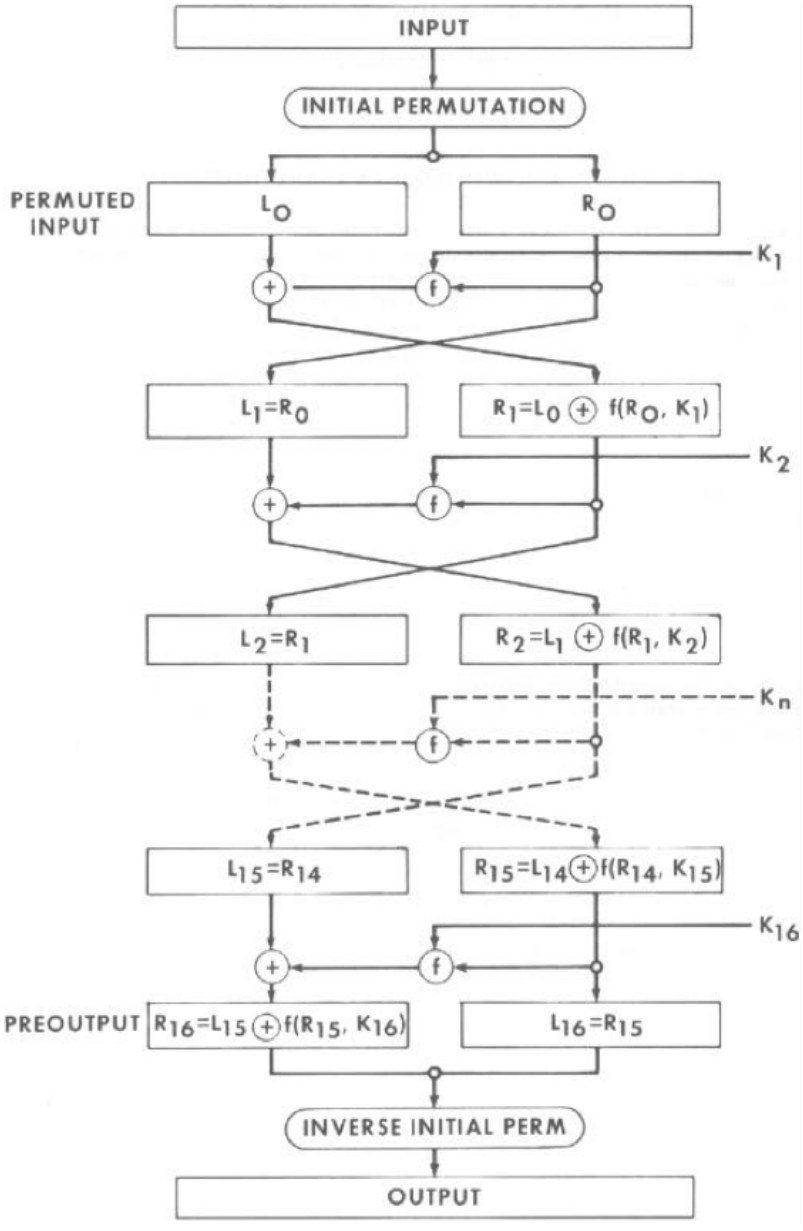
- RC4 (1987)
  Ronald Rivest
- Salsa20 (2005)
  Daniel J. Bernstein
- AES-CTR
  Pré-calcul de la clé

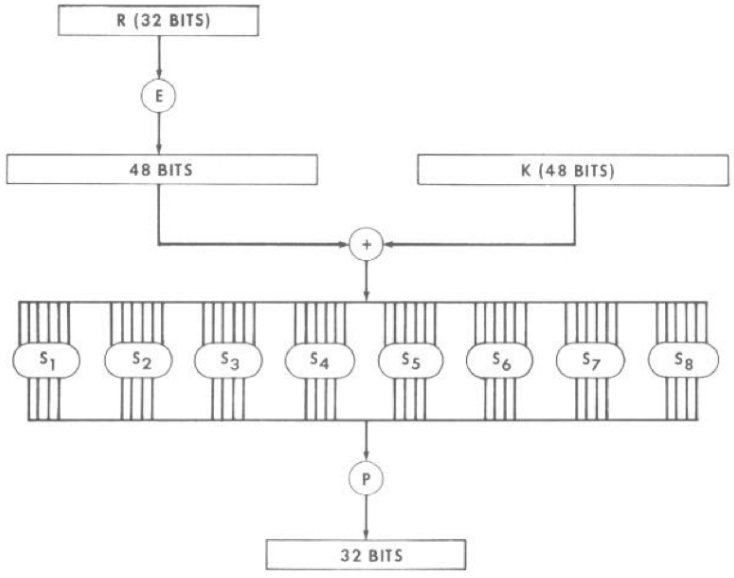
# DES, Data Encryption Standard

`csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf`

1. Diversification de la clé → 16 sous clés $K_{1..16}$ de 48 bits
   *Chaque $K_i$ est composé de 48 bits de $K$ pris dans un certain précis*
2. Fractionnement du texte en blocs $B_{1..n}$ de 64 bits
3. Pour chaque bloc $B_j$
   1. Permutation initiale du bloc $B_j$
   2. Découpage du bloc $B_j$ en parties gauche $G_0$ et droite $D_0$
   3. Pour chaque sous clé, $K_i$
      1. $G_i = D_{i-1}$
      2. $D_i = G_{i-1} \oplus f(D_{i-1}, K_i)$
   4. Reconstitution du bloc $B'_j$ à partir de $G_{16}$ et $D_{16}$
   5. Permutation initiale inverse du bloc $B'_j$

# DES, Data Encryption Standard




*Fonction $f(R, K)$*

***IP***

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 58 | 50 | 42 | 34 | 26 | 18 | 10 | 2 |
| 60 | 52 | 44 | 36 | 28 | 20 | 12 | 4 |
| 62 | 54 | 46 | 38 | 30 | 22 | 14 | 6 |
| 64 | 56 | 48 | 40 | 32 | 24 | 16 | 8 |
| 57 | 49 | 41 | 33 | 25 | 17 | 9 | 1 |
| 59 | 51 | 43 | 35 | 27 | 19 | 11 | 3 |
| 61 | 53 | 45 | 37 | 29 | 21 | 13 | 5 |
| 63 | 55 | 47 | 39 | 31 | 23 | 15 | 7 |

*Permutation initiale*

# Cryptanalyse : niveaux d'attaques

Niveau de puissance de l'attaquant

1. Attaque à texte chiffré : → récupérer le clair, voire la clé
   - Possède des messages chiffrés
2. Attaque à clair connu :
   - Possède des couples de message clair / chiffré
3. Attaque à clair choisi :
   - Construit des couples de message clair / chiffré
   - Choisi le clair à chiffrer, Chiffre en mode boite noire
3. Attaque à chiffré choisi :
   - Construit des couples de message clair / chiffré
   - Choisi le chiffré à chiffrer, Chiffre en mode boite noire

# Cryptanalyse : types d'attaques

Précédent la cryptographie moderne

- Analyse fréquentielle (texte chiffré)
- Indice de coïncidence (texte chiffré)
- Mot probable (clair connu)
- Force Brute

Cryptographie moderne

- Cryptanalyse linéaire (clair connu)
- Cryptanalyse différentielle (clair choisi)
- Canal auxiliaire (temps, consommation, e.m.)

# Cryptanalyse : modèles de sécurité

Modèle pour caractériser le niveau de sécurité d'un chiffre

- Sécurité inconditionnelle (*perfect secrecy*)
  - Théorie de l'information Shannon
  - $H(M) = H(M|C)$, $H(X)$ entropie de $X$, incertitude
  - Taille de la clé nécessairement aussi grande que le message
    One-time pad
- Sécurité prouvable
  - *Équivalence du chiffre avec un problème difficile connu*

  → Réduction à un problème NP (ex : réseaux euclidiens)
- Sécurité par complexité de calcul (*computational security*)
  - Hypothèse sur la puissance de calcul de l'attaquant ($O(2^{80})$)
  - *Quantité de calcul à exécuter avec la meilleure méthode connue*
  - Concerne la plupart des chiffres modernes
  - ex : bits de clés (chiffre symétrique), RSA, logarithme discret

# Cryptanalyse : mise en œuvre des chiffres

De la théorie vers la pratique

- La mise en œuvre des chiffres est non triviale
- Protection des secrets en mémoire (TEE, HSM)
- Gestion de l'aléa (matériel quantique / chaotique, post traitement)
- Protection contre les attaques intrusives
- Protection contre le canaux auxiliaires
- Et bien d'autres...

## À retenir

- Mettre en œuvre de la cryptographie est très difficile

→ Préférer les projets ouverts, de spécialistes et à l'état de l'art

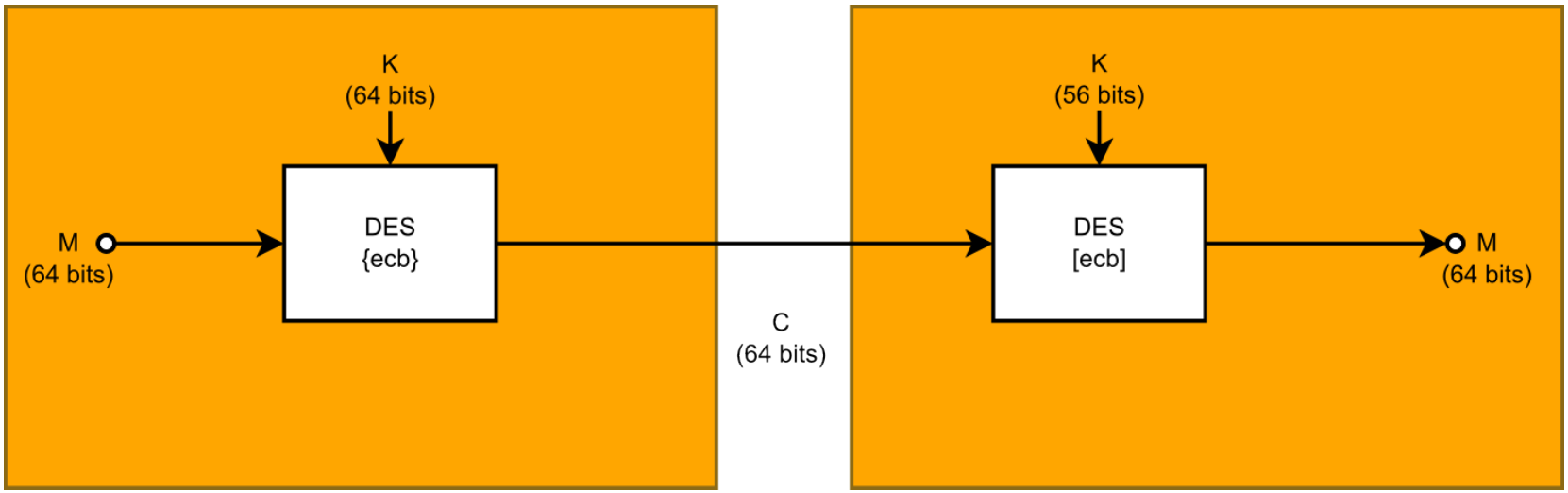
- NaCL, libsodium, etc (Daniel J. Bernstein)
- openssl : largement éprouvé par une communauté

# Cryptanalyse : niveau de sécurité

Cible de sécurité : combinaison subtile des paramètres suivants :

- La configuration du chiffre (taille des clés, etc.)
- Niveau d'attaque / attaques à considérer
- Le modèle de sécurité considéré (qui évolue : bits de sécurité)
- La qualité de la mise en œuvre
- Niveau et durée d'évaluation du chiffre
- Niveau et durée évaluation de la mise en œuvre
- D'autres paramètres d'environnement : qualité de l'aléa, etc.
- Et bien d'autres...

# *DES*, mode *ECB* (*Electronic-Code-Book*)




Mode d'opération par bloc

## Avantages

- Parallélisme
- Accès aléatoire

# *DES*, mode *ECB* (*Electronic-Code-Book*)

## Faiblesses

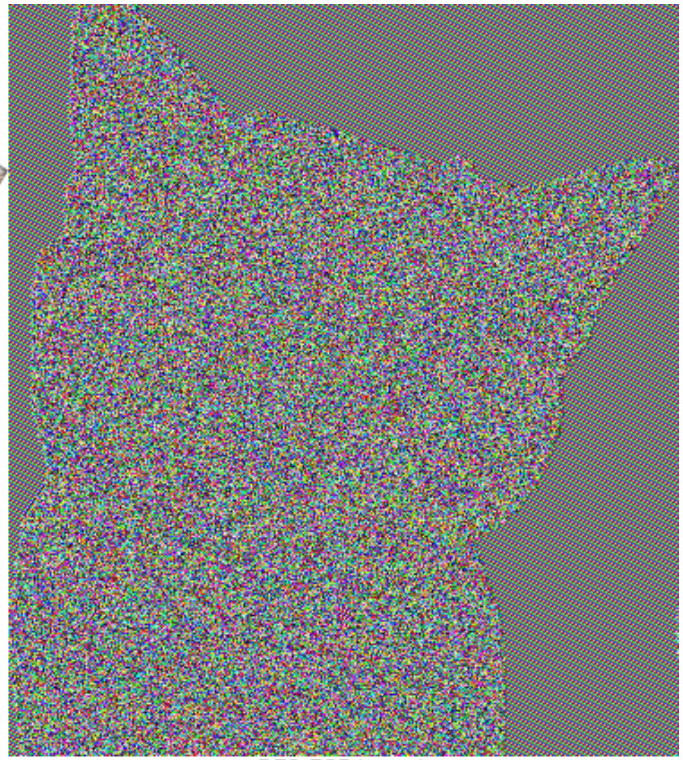
- Tout bloc clair dans un texte, donnera systématiquement le même bloc chiffré

# *DES*, mode *ECB* (*Electronic-Code-Book*)

## Faiblesses

- Tout bloc clair dans un texte, donnera systématiquement le même bloc chiffré

DES-ECB is weak



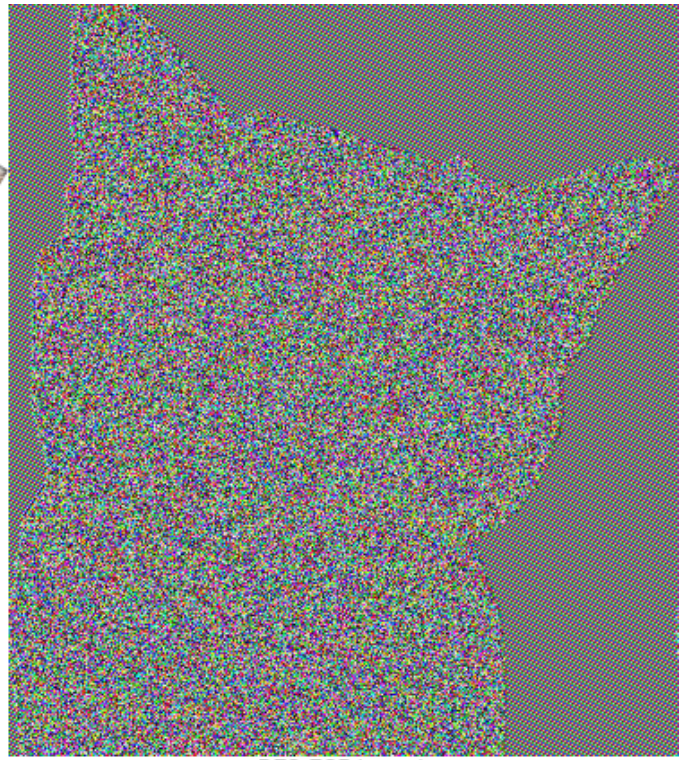
DES-ECB is weak

# *DES*, mode *ECB* (*Electronic-Code-Book*)

## Faiblesses

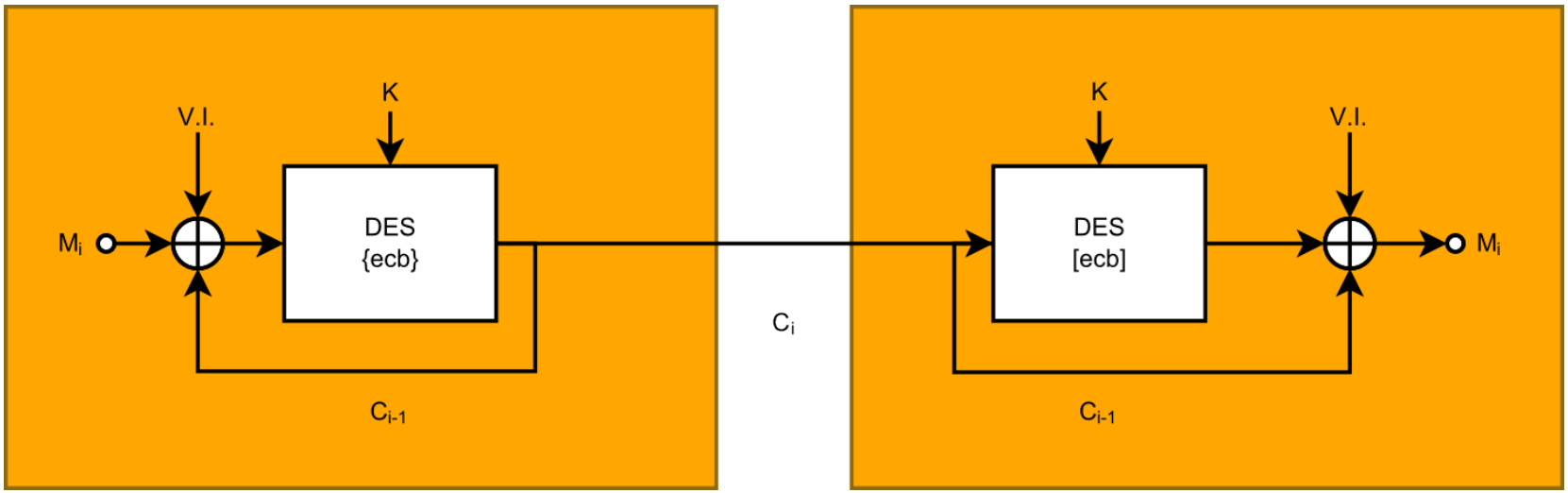
- Tout bloc clair dans un texte, donnera systématiquement le même bloc chiffré


DES-ECB is weak


DES-ECB is weak

## Déconseillé

# *DES*, mode *CBC* (*Cipher-Block-Chaining*)




Mode d'opération par bloc

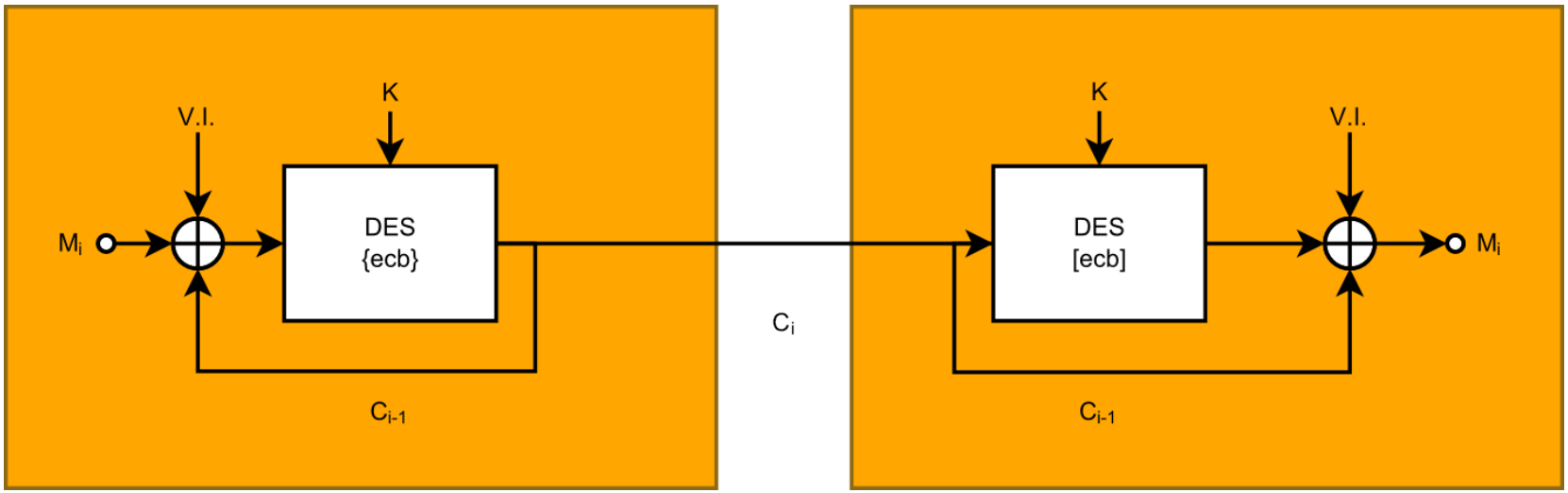
## Avantages

- Accès aléatoire
- Déchiffrement en parallèle

## Désavantages

- Chiffrement séquentiel

# *DES*, mode *CBC* (*Cipher-Block-Chaining*)




Mode d'opération par bloc

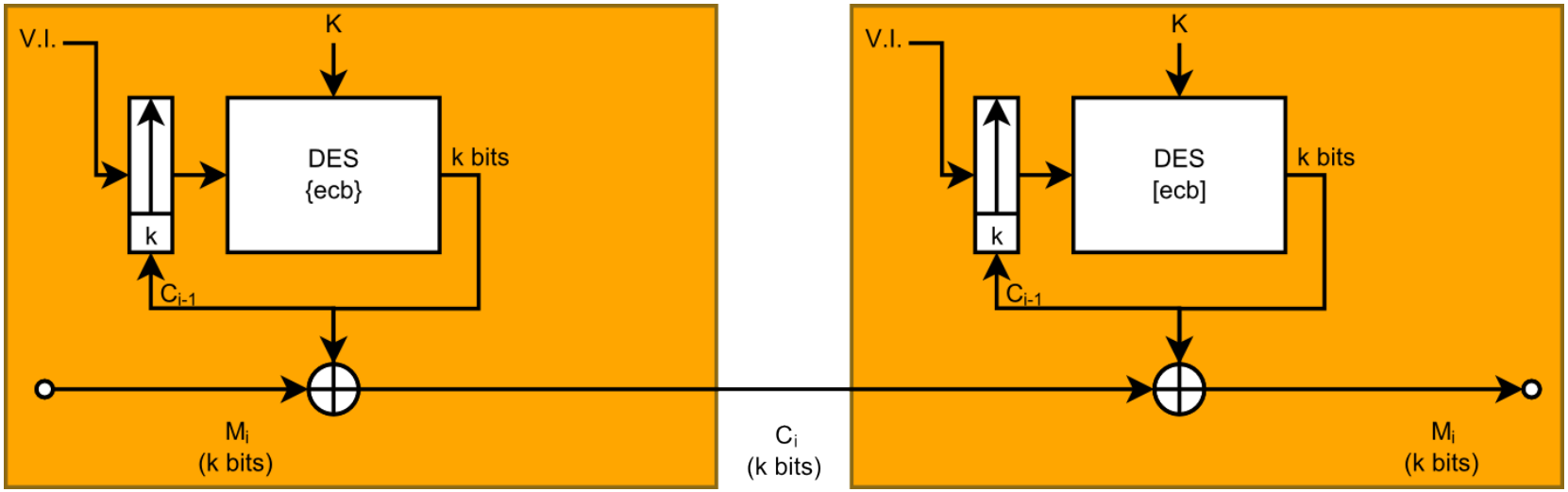
## Forces

- Deux même blocs de texte clair seront deux blocs chiffrés différents (*idem* autres modes)

## Faiblesses

- Bourrage nécessaire (attaque POODLE SSLv3)

# *DES*, mode *OFB* (*Output-Feedback-Block*)




Mode d'opération par flot synchrone

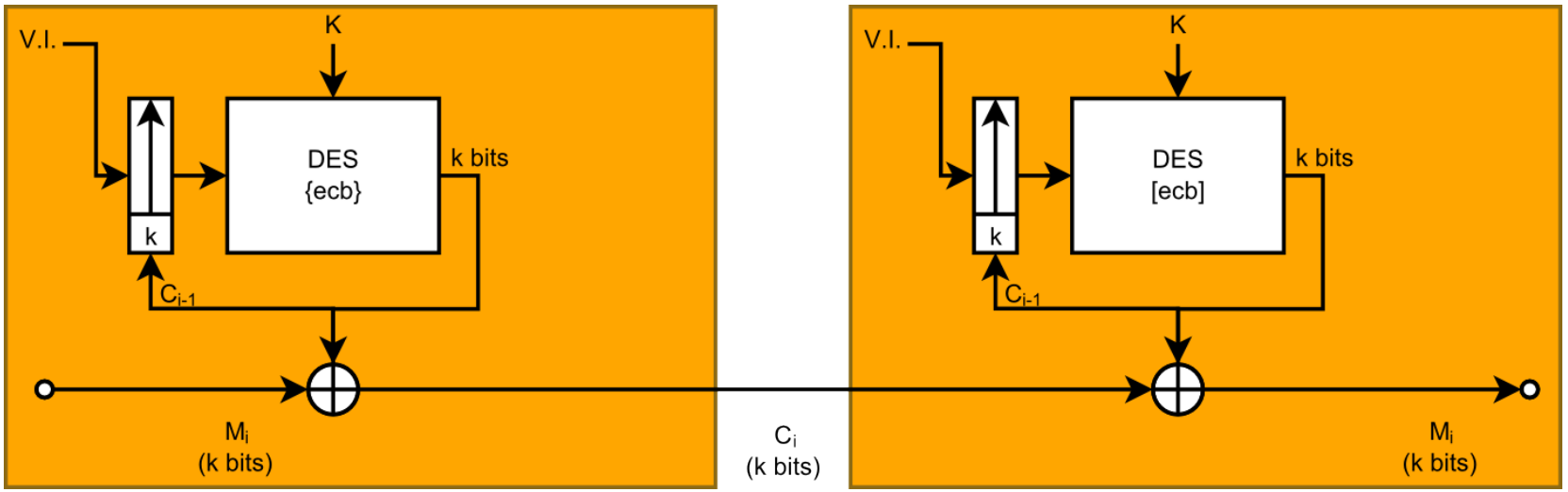
## Avantages

- Même circuit de chiffrement et déchiffrement
- $\exists$ codes correcteurs d'erreurs sont applicables sur le chiffré (*bit flips*)

## Désavantages

- Chiffrement Séquentiel
- Déchiffrement Séquentiel
- Besoin de synchronisation parfaite (client / serveur)

# *DES*, mode *OFB* (*Output-Feedback-Block*)




Mode d'opération par flot synchrone

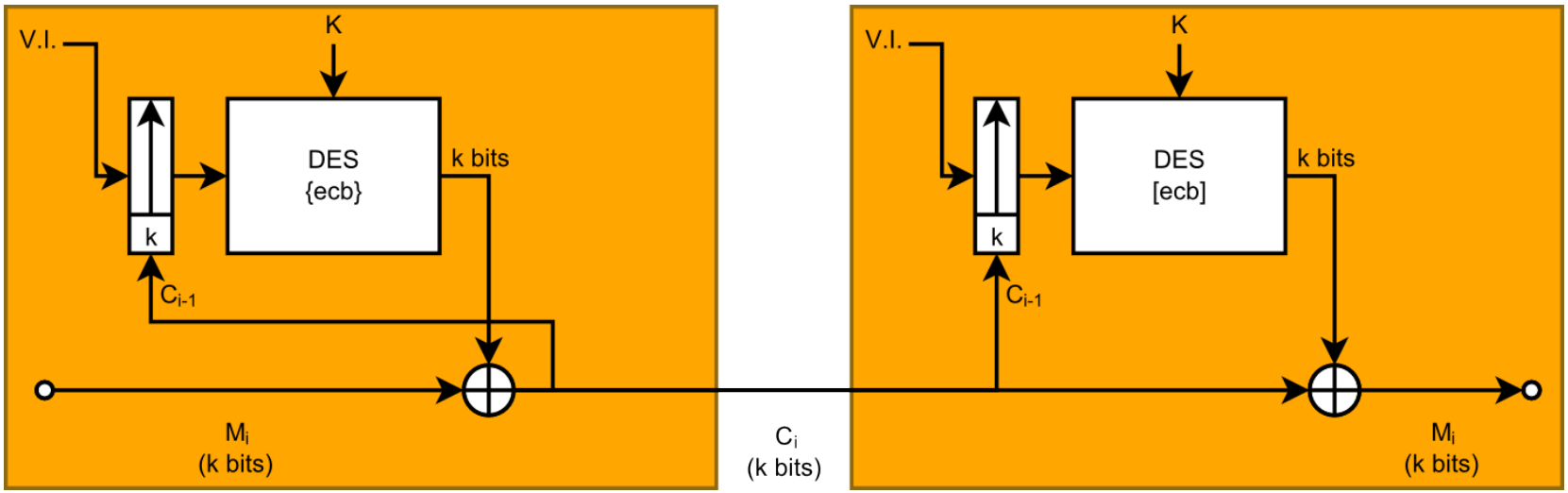
## Forces

- En plus ?

## Faiblesses

- Attaque active facilitée : 1 *bit flip* clair = 1 *bit flip* chiffré
- Attaque à clair connu : 1 $IV$ + 1 $K$ = 1 keystream

# *DES*, mode *CFB* (*Cipher-Feedback-Block*)




Mode d'opération par flot auto synchrone (si $k$ bits perdus ou ajoutés)

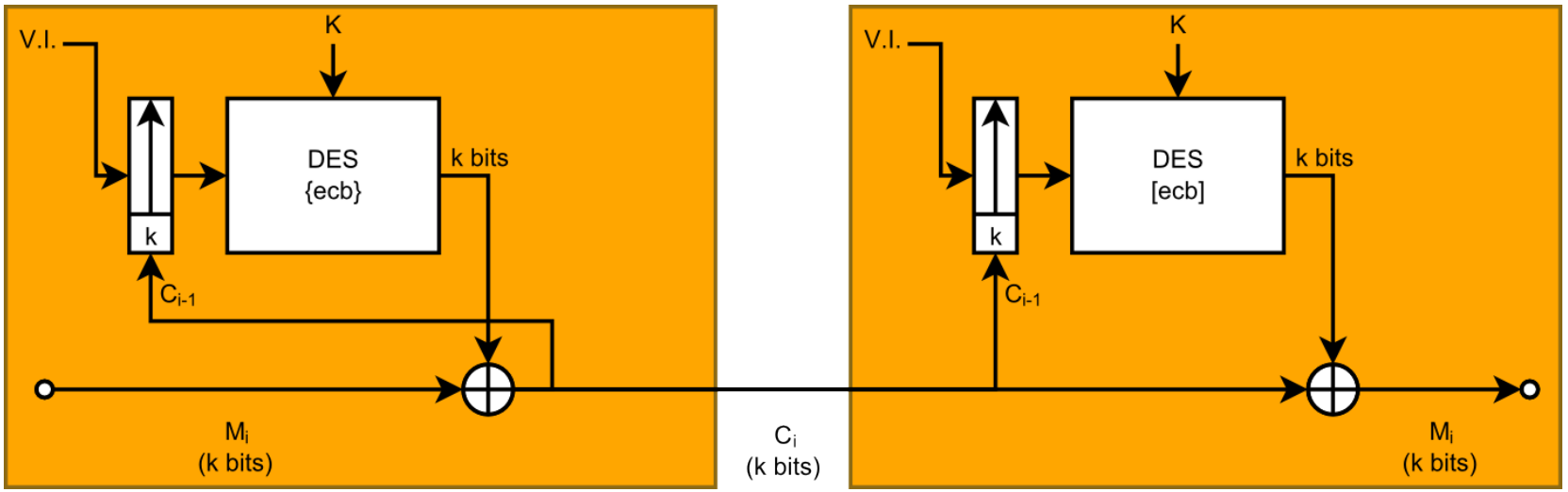
## Avantages

- Accès aléatoire
- Déchiffrement en parallèle
- Synchronisation modulo $k$ (client / serveur)

## Désavantages

- Chiffrement séquentiel

# *DES*, mode *CFB* (*Cipher-Feedback-Block*)




Mode d'opération par flot auto synchrone (si $k$ bits perdus ou ajoutés)

## Forces

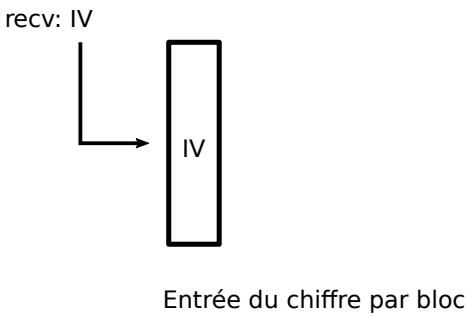
- En plus ?

## Faiblesses

- ?

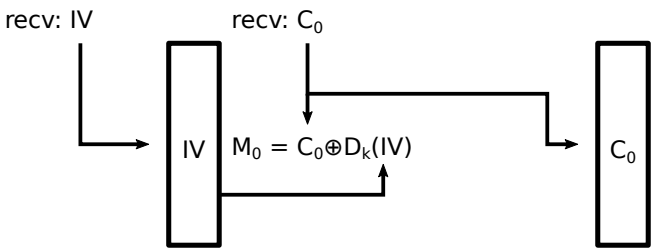
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Fonctionnement normal sans *shift register*



# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

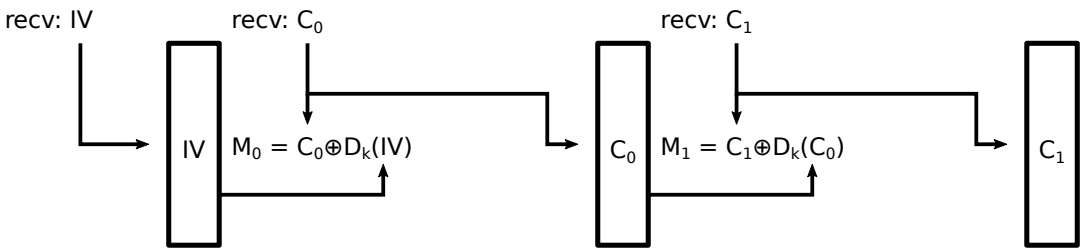
## Fonctionnement normal sans *shift register*




Entrée du chiffre par bloc

# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

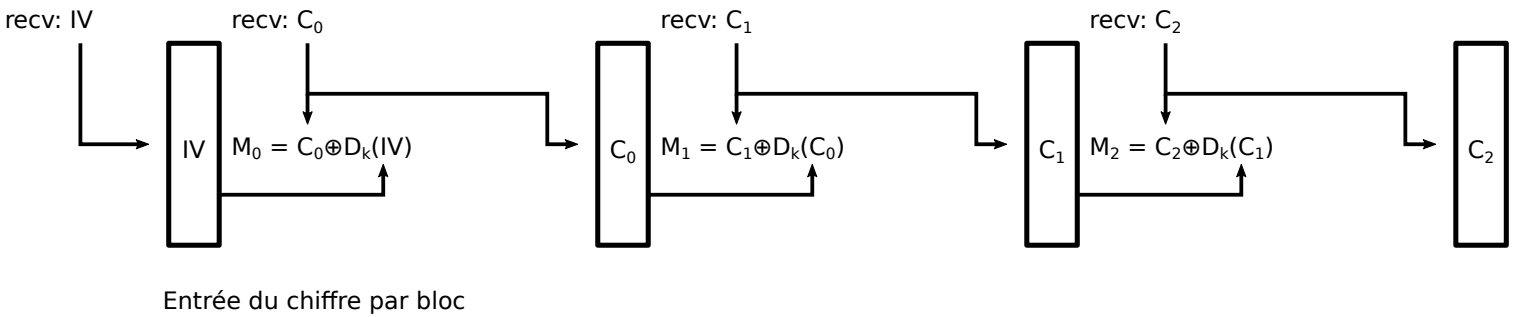
## Fonctionnement normal sans *shift register*




Entrée du chiffre par bloc

# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

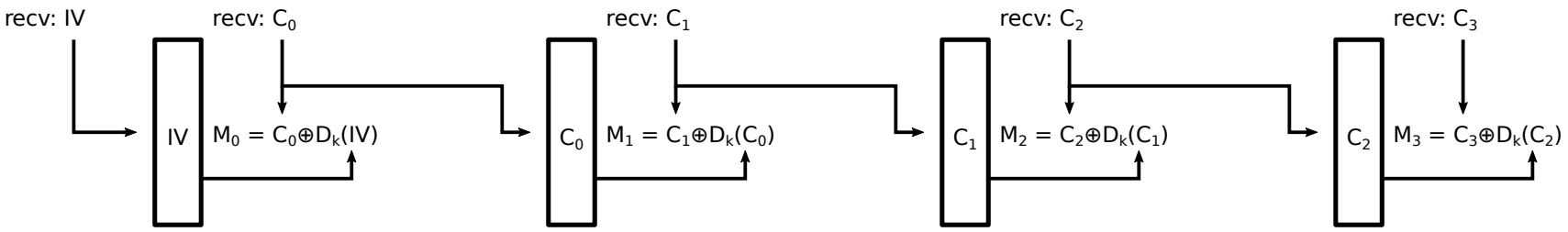
## Fonctionnement normal sans *shift register*




Entrée du chiffre par bloc

# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

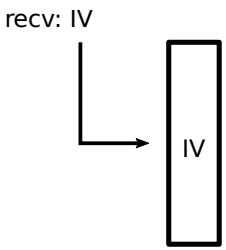
## Fonctionnement normal sans *shift register*




Entrée du chiffre par bloc

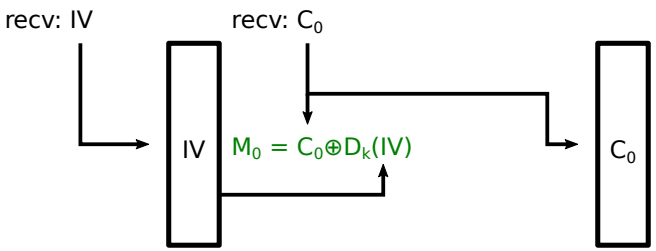
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Perte de $n$ bits et resynchronisation après $n$ bits reçus



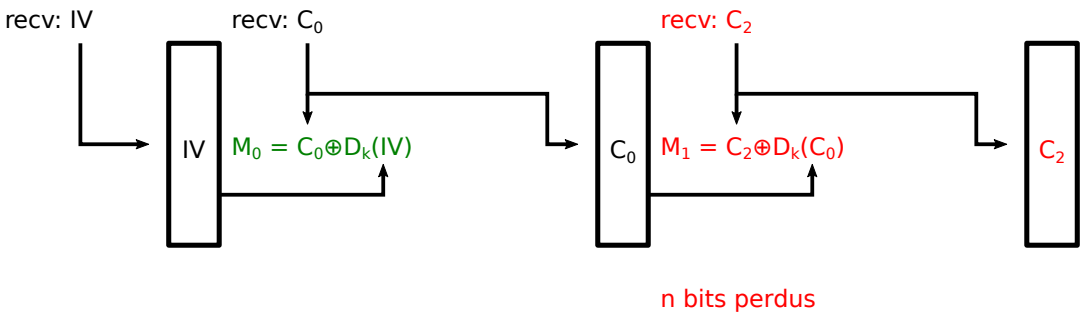
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Perte de $n$ bits et resynchronisation après $n$ bits reçus



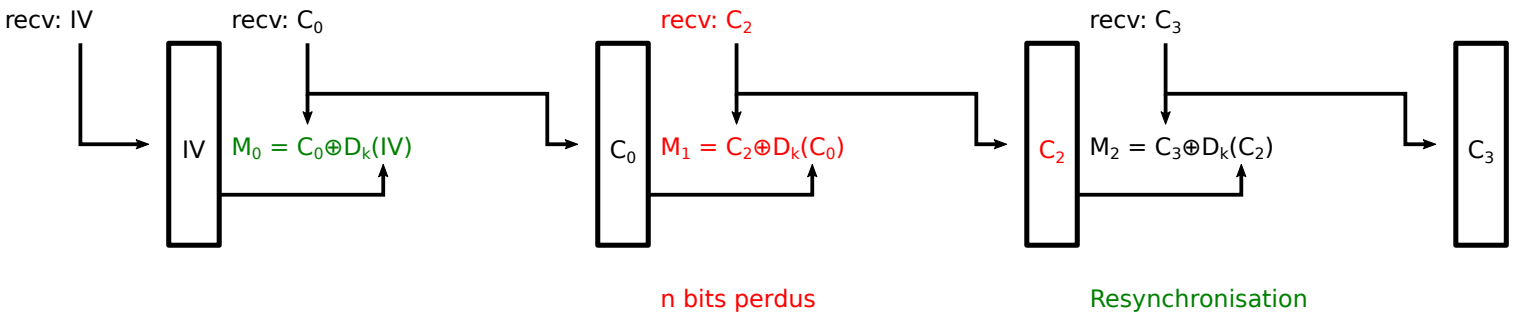
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Perte de $n$ bits et resynchronisation après $n$ bits reçus



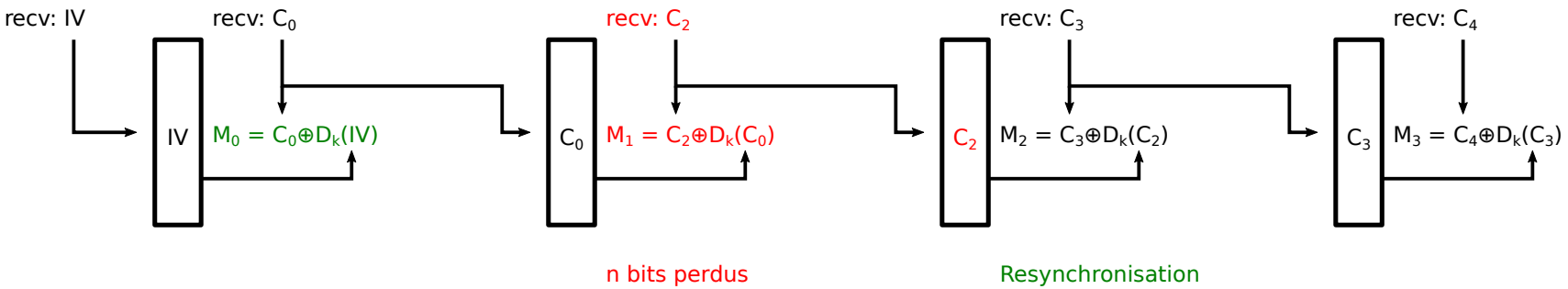
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Perte de $n$ bits et resynchronisation après $n$ bits reçus



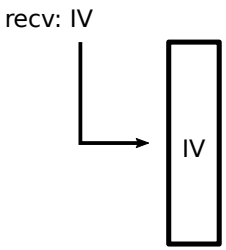
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

## Perte de $n$ bits et resynchronisation après $n$ bits reçus



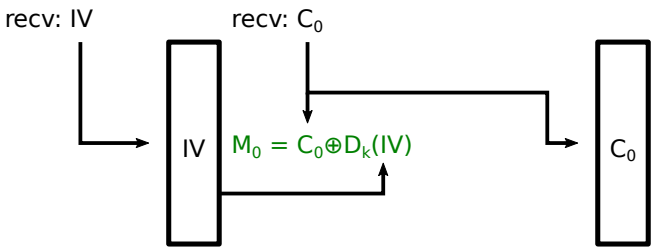
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Désynchronisation totale après perte de $n/2$ bits



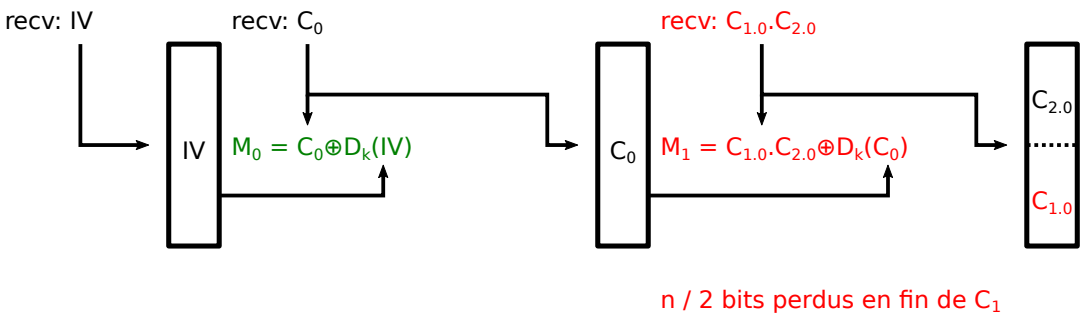
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Désynchronisation totale après perte de $n/2$ bits



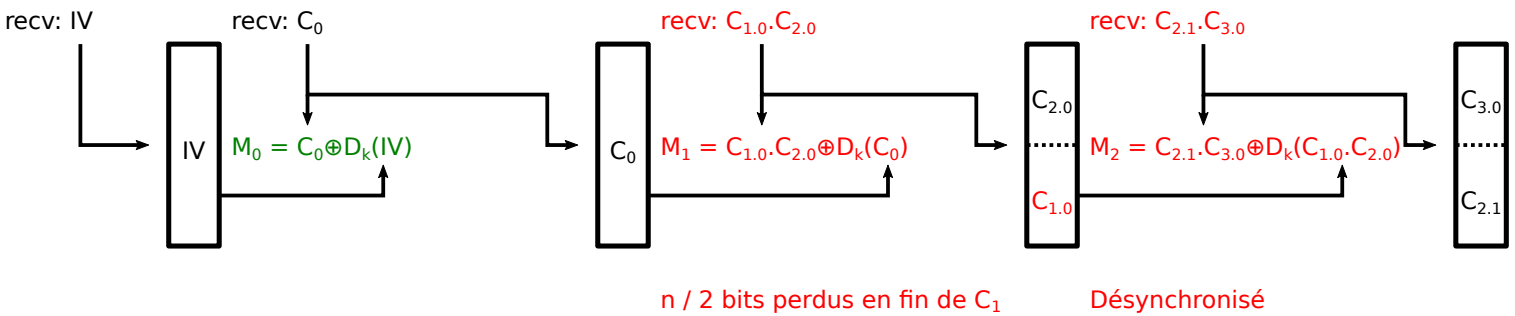
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

## Désynchronisation totale après perte de $n/2$ bits



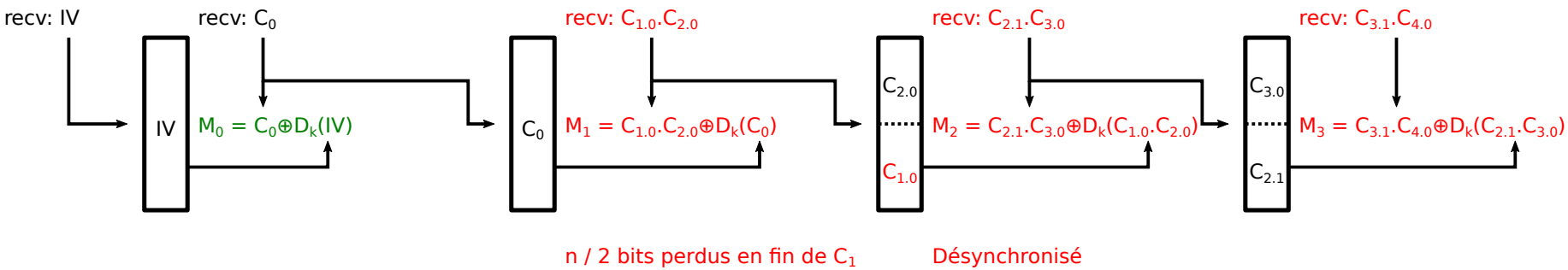
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

## Désynchronisation totale après perte de $n/2$ bits



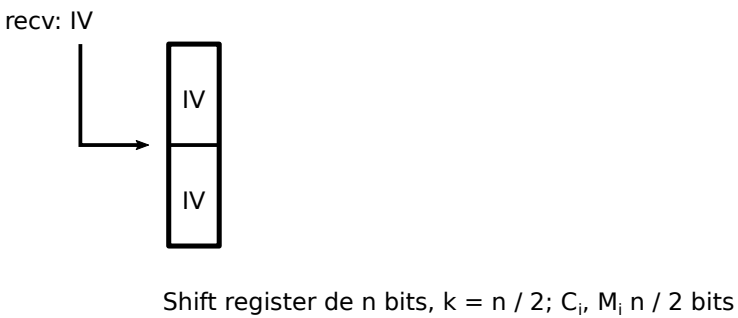
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

## Désynchronisation totale après perte de $n/2$ bits



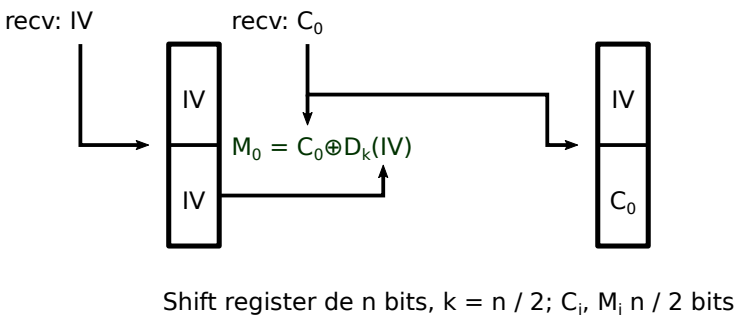
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Fonctionnement normal avec un *shift register* décalant de $k = n/2$ bits



# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

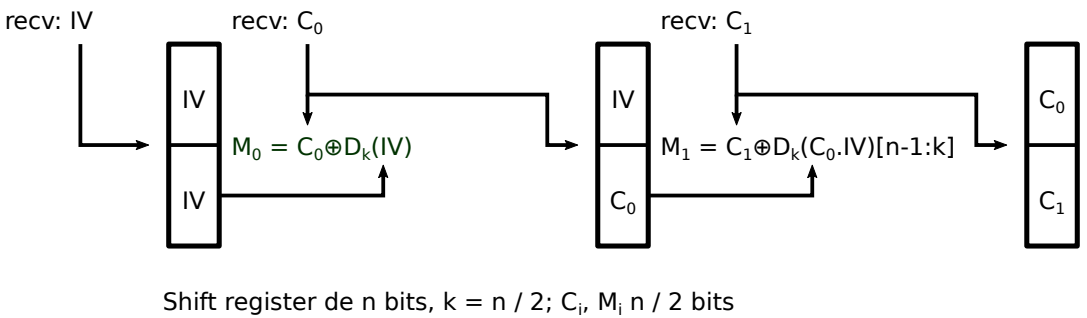
Fonctionnement normal avec un *shift register* décalant de $k = n/2$ bits




Shift register de n bits, k = n / 2; $C_i$, $M_i$ n / 2 bits

# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

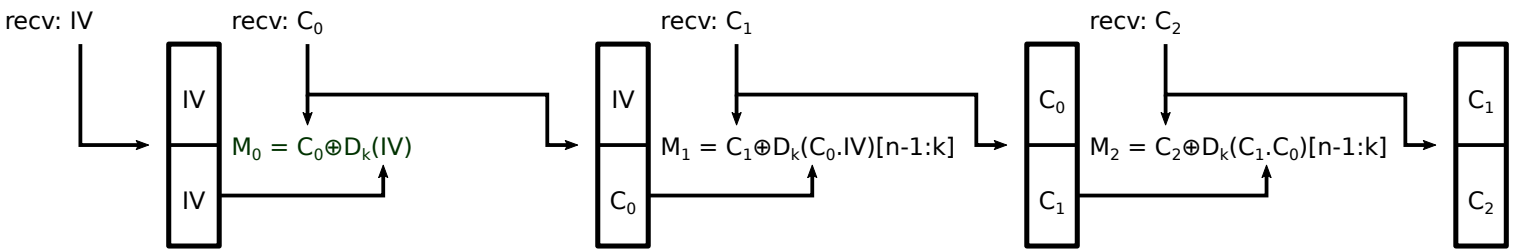
Fonctionnement normal avec un *shift register* décalant de $k = n/2$ bits




Shift register de n bits, k = n / 2; $C_i$, $M_i$ n / 2 bits

# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

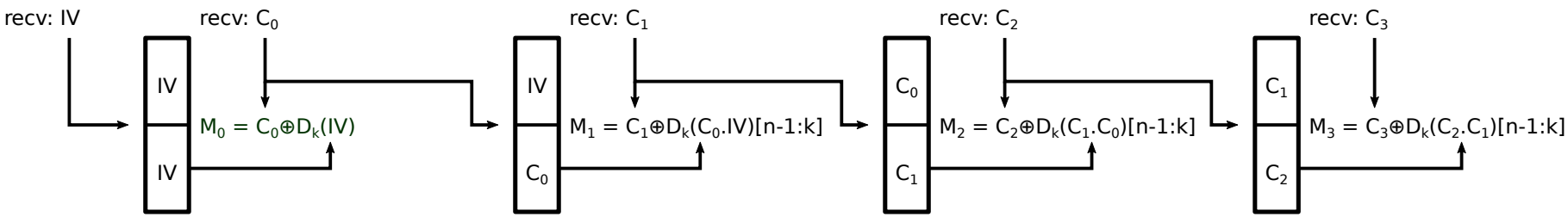
Fonctionnement normal avec un *shift register* décalant de $k = n/2$ bits




Shift register de n bits, k = n / 2; $C_i$, $M_i$ n / 2 bits

# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

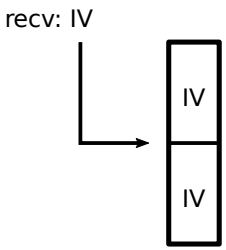
Fonctionnement normal avec un *shift register* décalant de $k = n/2$ bits




Shift register de n bits, k = n / 2; $C_i$, $M_i$ n / 2 bits

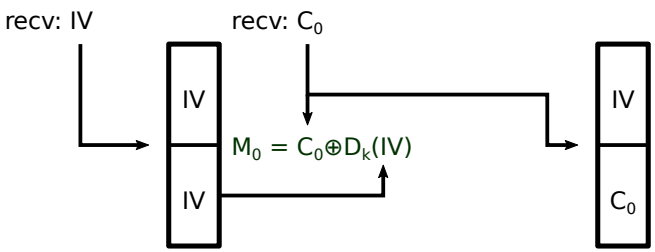
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Perte de $n/2$ bits et resynchronisation après $n$ bits reçus



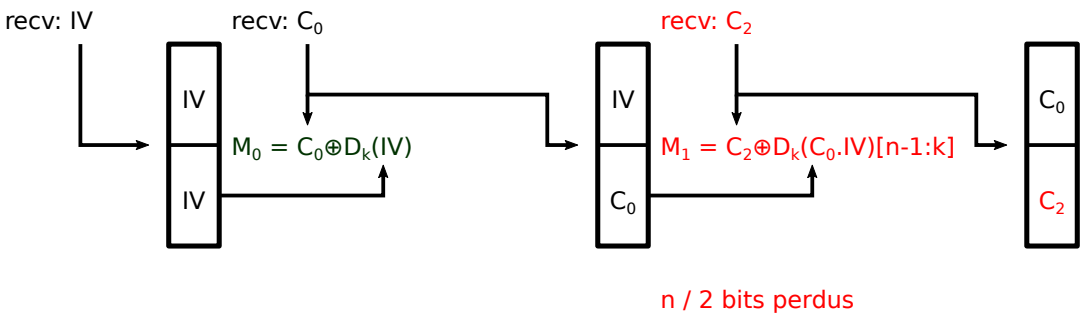
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Perte de $n/2$ bits et resynchronisation après $n$ bits reçus



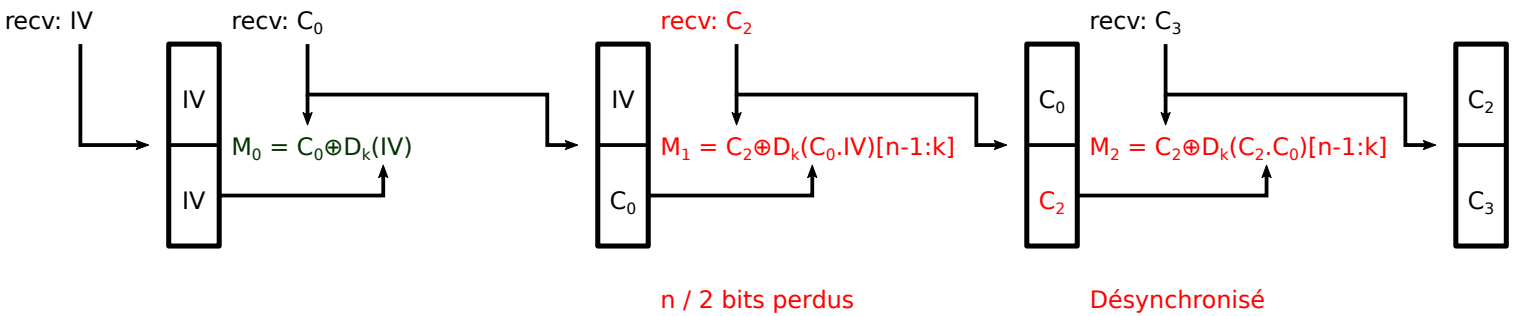
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Perte de $n/2$ bits et resynchronisation après $n$ bits reçus



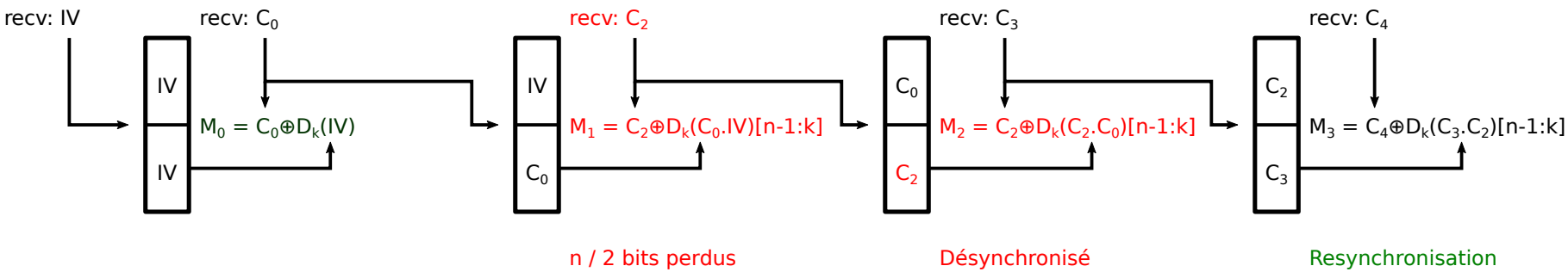
# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

Perte de $n/2$ bits et resynchronisation après $n$ bits reçus



# *DES*, mode *CFB* (*Cipher-Feedback-Block*)

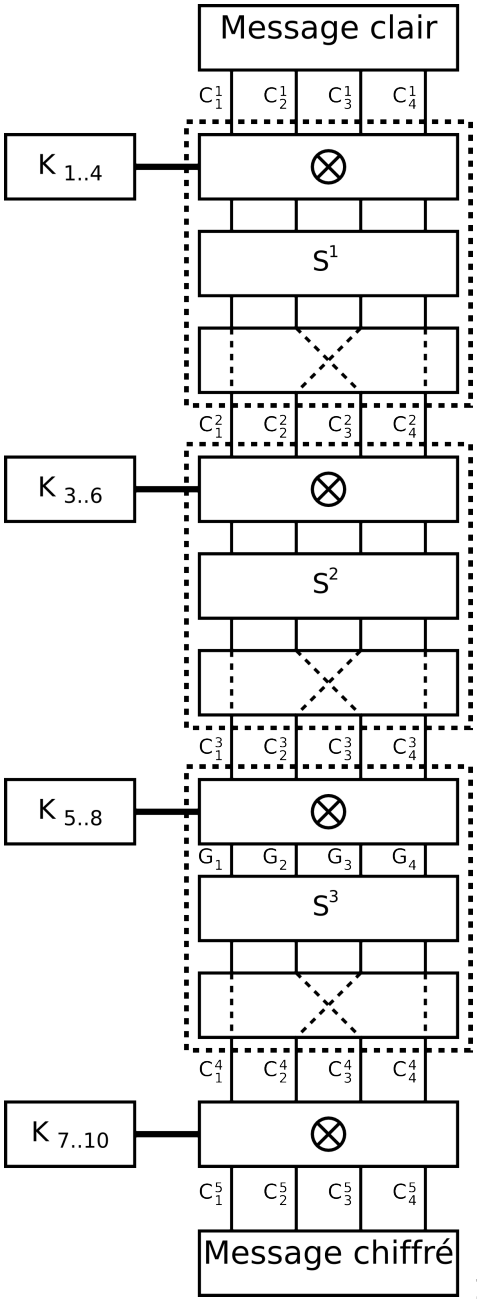
Perte de $n/2$ bits et resynchronisation après $n$ bits reçus



# Cryptanalyse linéaire

- Attaque à clairs connus
  - $\mathcal{T} = \{(\text{message}_i, \text{ciphertext}_i)\}$
- Sur les réseaux de substitution-permutation (SPN)
  - AES, DES,. . .
- L'attaquant dispose de l'algorithme et recherche la clé
- Exploitation d'un manque de non linéarité pour établir des approximations

  $\Rightarrow$ Réduction de l'espace de recherche
  Être plus rapide que la force brute



# Cryptanalyse linéaire – non linéaire ?

- Si substitution $S^i$ est non linéaire $\Rightarrow$ robuste à cette attaque. . .
- $\forall y = S^i(x), \mathcal{P}(\bigoplus_{i=1}^{n} x_i = \bigoplus_{i=1}^{n} y_i) = 1/2$
- Parfois non vérifié pour des sous parties des vecteurs $y$ et $x$
- ex : si $(1)$ $x_4 = y_4 \oplus y_3 \oplus y_2$ vérifié 14 fois sur 16
- Biais $\epsilon = |\mathcal{P}[(1)] - 1/2| = 3/8$

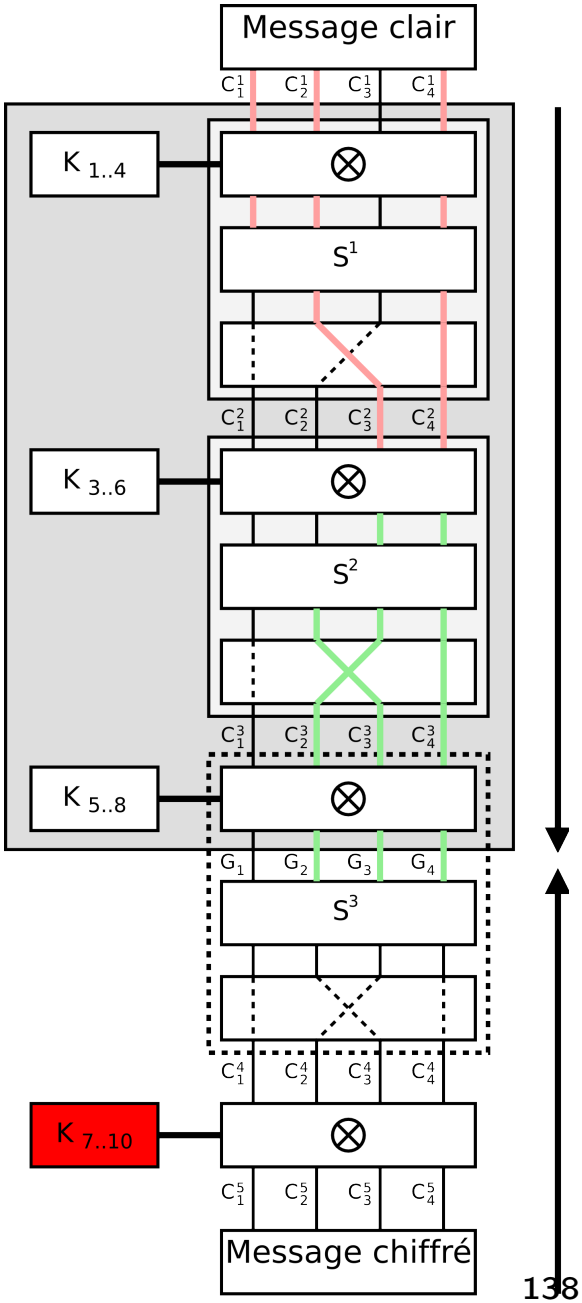
$\Rightarrow$ Utiliser $\epsilon$ comme distingueur

- C'est à dire, pour un grand nombre de vecteurs $x$ et $y$, si l'équation est vérifiée avec un biais de 3/8, $y$ a été bien généré par $S^i(x)$, pour les bits concernés
- **Intuition :** vérifier le biais sur tous les couples (clair, chiffré) est plus rapide qu'appliquer la subsitution
- Sur une approximation globale de l'algorithme → exctraction plus rapide de bits de clés qu'une attaque par force brute sans approximation

# Cryptanalyse linéaire – démarche

- Approximation d'une partie de l'algorithme
  - Indépendante de la clé, pour réduire l'espace de recherche
  - Dépendante des messages clairs et de l'entrée du dernier bloc (tous les *rounds* sauf le dernier)
  - **Rapide à exécuter**
- Attaque par force brute sur la partie non approximée de l'algorithme
  - Première étape du déchiffrement avec une clé candidate

⇒ Trouver une partie de la clé $K$, correspondant au dernier *round*



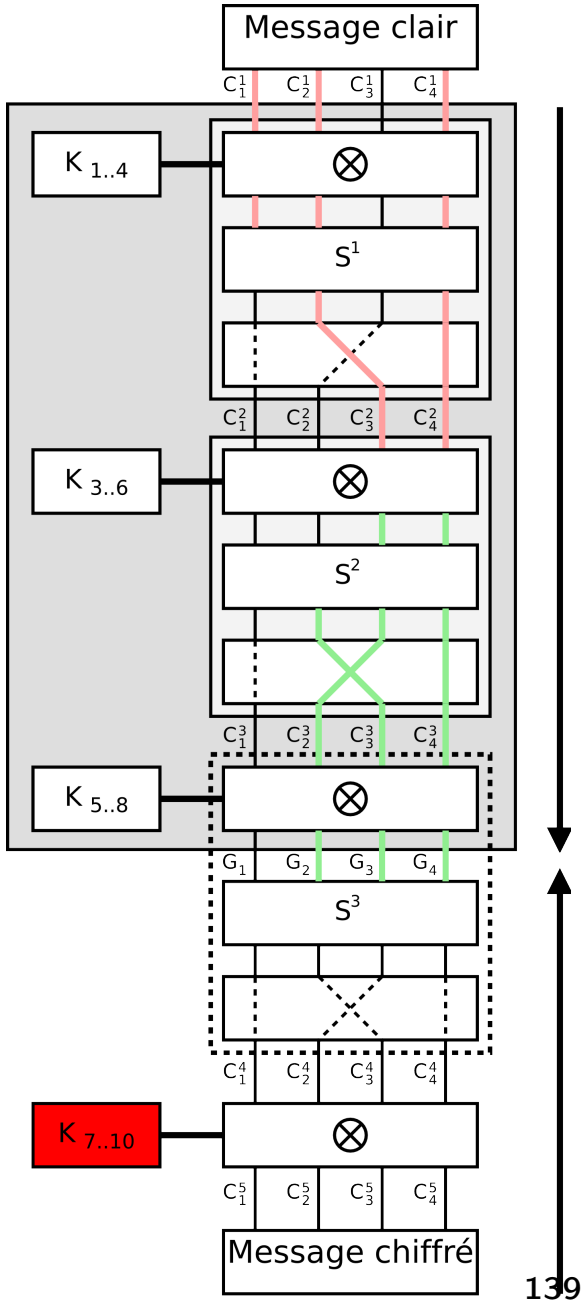
# Cryptanalyse linéaire – algorithme

*Version simplifiée*

**for** candidat = 0 .. $2^4 - 1$ **do**
  $\text{score}_{\text{candidat}} \leftarrow 0$
  **for all** (message, ciphertext) $\in \mathcal{T}$ **do**
    valeur $\leftarrow$ ciphertext $\oplus$ candidat
    valeur $\leftarrow (\text{Permute}^3)^{-1}$(valeur)
    valeur $\leftarrow (S^3)^{-1}$(valeur)
    *Test si l'approximation est vérifiée par le couple valeur, message*
    **if** Approximation_Avérée(valeur, message) **then**
      $\text{score}_{\text{candidat}} \leftarrow \text{score}_{\text{candidat}} + 1$
    **end if**
  **end for**
**end for**
résultat $\leftarrow \underset{x}{\text{argmax}}\ \text{abs}(\text{score}_x - |\mathcal{T}|/2)$

*On sélectionne le candidat qui a le biais le plus proche :*

$$\epsilon = \left| \frac{\text{score}_{\text{candidat}} - |\mathcal{T}|/2}{|\mathcal{T}|} \right|$$



# Cryptanalyse linéaire

- Approximation linéaire des premiers *rounds*
- **Objectif :** obtenir une équation de la forme :

$$(\bigoplus_{i=1}^{4} a_i \wedge C_i^1) \oplus (\bigoplus_{i=1}^{4} b_i \wedge G_i^4) \oplus \text{Constante} = 0$$

$$\mathcal{P}[(\bigoplus_{i=1}^{4} a_i \wedge C_i^1) \oplus (\bigoplus_{i=1}^{4} b_i \wedge G_i^4) \oplus \text{Constante} = 0] = 1/2 + \epsilon$$

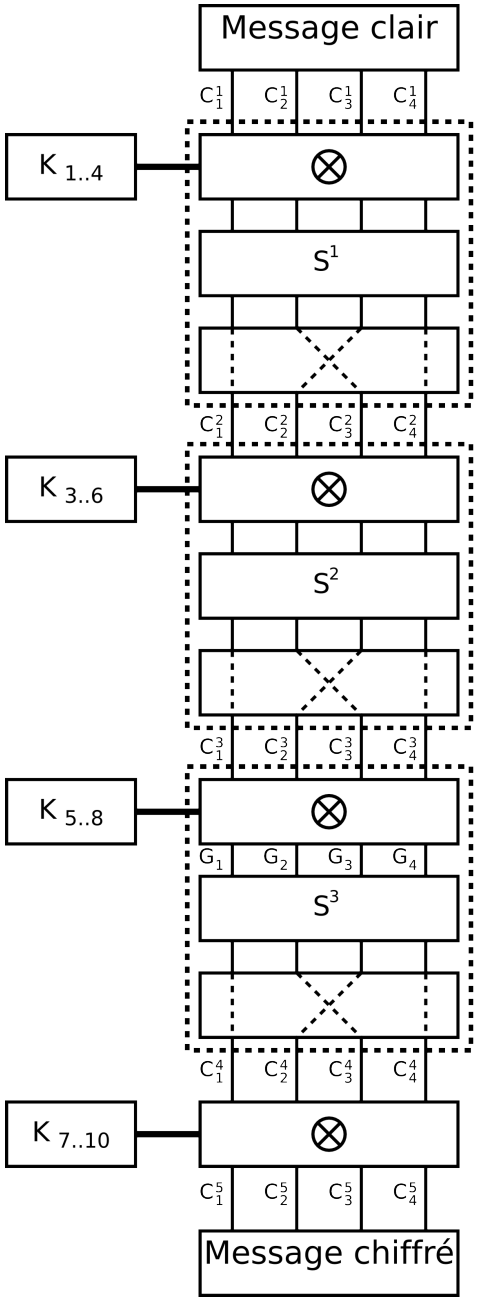
$a_i$ et $b_i$ sont des masques

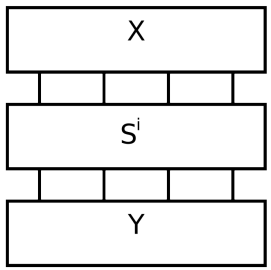
- **Objectif :** associer le biais attendu :
- Le biais attendu est calculé par combinaison des $\epsilon$ de chaque round
- Utilisation du lemme *Piling-Up* produit par Mitsuru Matsui
- Constante : clé de chiffrement $\rightarrow$ biais positif ou négatif ($\oplus = 1$ ou $0$)

# Cryptanalyse linéaire

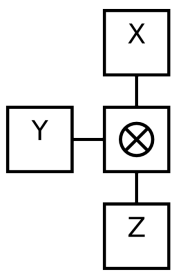
- Approximation linéaire des premiers *rounds*
  - Dernier *round* non approximé : toute ou partie de la clé correspondante va être recherchée
  - Pour chaque clé candidate au dernier *round*, il est possible de calculer $C_i^4$ correspondant, puis $G_i^4$
  - La clé vraissemblablement utilisée pour chiffrer les $\text{message}_i$ en $\text{ciphertext}_i$ est celle pour laquelle l'approximation est la meilleure ...

# Cryptanalyse linéaire – exemple



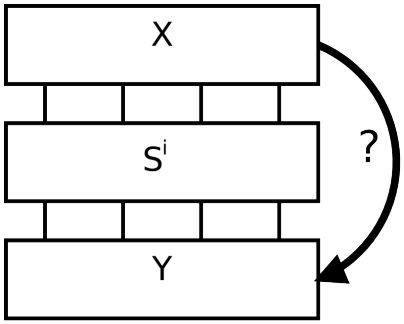

| $X_1X_2X_3X_4$ | $Y_1Y_2Y_3Y_4$ |
|---|---|
| 0000 | 0011 |
| 0001 | 1000 |
| 0010 | 1001 |
| 0011 | 0101 |
| 0100 | 1011 |
| 0101 | 0001 |
| 0110 | 0010 |
| 0111 | 0100 |
| 1000 | 0000 |
| 1001 | 1111 |
| 1010 | 0111 |
| 1011 | 0110 |
| 1100 | 1110 |
| 1101 | 1010 |
| 1110 | 1100 |
| 1111 | 1101 |




| XY | Z |
|---|---|
| 00 | 0 |
| 01 | 1 |
| 10 | 1 |
| 11 | 0 |

- Plusieurs *rounds*
- Fonction non linéaire $S^i$, ne peut pas être exprimée sous la forme de xor

# Cryptanalyse linéaire – exemple, table des approximations




$$N(\,(a_1, a_2, a_3, a_4), (b_1, b_2, b_3, b_4)\,) = |\{(X_1, X_2, X_3, X_4), (Y_1, Y_2, Y_3, Y_4) / \bigoplus_{i=1}^{4} a_i X_i \bigoplus_{i=1}^{4} b_i Y_i = 0\}|$$

$(a_1, a_2, a_3, a_4)$ \ $(b_1, b_2, b_3, b_4)$

| | 0000 | 0001 | 0010 | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | 1011 | 1100 | 1101 | 1110 | 1111 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0000 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0001 | 0 | 0 | -2 | 2 | 2 | -2 | 0 | 0 | 0 | 0 | -2 | 2 | 2 | 6 | 0 | 0 |
| 0010 | 0 | 0 | -2 | 2 | 4 | 0 | 2 | 2 | -2 | -2 | 4 | 0 | 2 | -2 | 0 | 0 |
| 0011 | 0 | 0 | 0 | 0 | -2 | -2 | -2 | 6 | 2 | 2 | 2 | 2 | 0 | 0 | 0 | 0 |
| 0100 | 0 | -2 | 0 | 2 | 0 | 2 | 4 | 2 | 2 | 0 | -2 | 0 | -2 | 0 | -2 | 4 |
| 0101 | 0 | -2 | 2 | 0 | 2 | 0 | 0 | -2 | 2 | 0 | 0 | 6 | 0 | -2 | 2 | 0 |
| 0110 | 0 | 2 | 2 | 4 | 0 | 2 | -2 | 0 | 0 | 2 | -2 | 0 | 4 | -2 | -2 | 0 |
| 0111 | 0 | 2 | 0 | -2 | 2 | 0 | -2 | 0 | 4 | -2 | 0 | -2 | 2 | 0 | 2 | 4 |
| 1000 | 0 | -2 | 2 | 0 | 4 | -2 | -2 | 0 | 2 | 0 | 0 | -2 | -2 | 0 | -4 | -2 |
| 1001 | 0 | -2 | -4 | -2 | 2 | 0 | -2 | 0 | -2 | 4 | -2 | 0 | 0 | -2 | 0 | 2 |
| 1010 | 0 | -2 | 0 | 2 | 0 | -2 | 0 | 2 | 0 | -2 | -4 | -2 | 0 | -2 | 4 | -2 |
| 1011 | 0 | -2 | -2 | 4 | -2 | 0 | -4 | -2 | 0 | -2 | 2 | 0 | -2 | 0 | 0 | 2 |
| 1100 | 0 | 0 | 2 | -2 | 0 | 0 | -2 | 2 | -4 | -4 | -2 | 2 | 0 | 0 | -2 | 2 |
| 1101 | 0 | 0 | 0 | 0 | -2 | -6 | 2 | -2 | 0 | 0 | 0 | 0 | 2 | -2 | -2 | 2 |
| 1110 | 0 | 4 | -4 | 0 | 0 | 0 | 0 | 0 | 2 | -2 | -2 | 2 | -2 | -2 | -2 | -2 |
| 1111 | 0 | -4 | -2 | -2 | -2 | 2 | 0 | 0 | 2 | -2 | 0 | 0 | 4 | 0 | -2 | -2 |

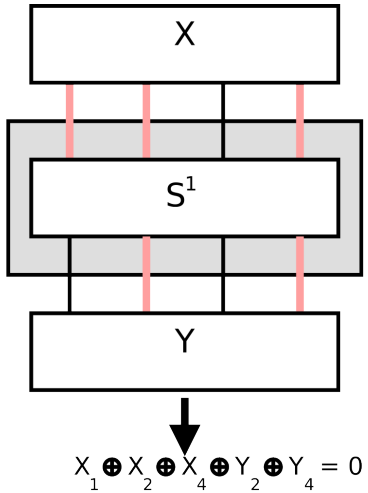
| XXXX (1 2 3 4) | YYYY (1 2 3 4) | XXXYY (1 2 4 2 4) | ⊕ |
|---|---|---|---|
| 0000 | 0011 | 00001 | 1 |
| 0001 | 1000 | 00100 | 1 |
| 0010 | 1001 | 00001 | 1 |
| 0011 | 0101 | 00111 | 1 |
| 0100 | 1011 | 01001 | 0 |
| 0101 | 0001 | 01101 | 1 |
| 0110 | 0010 | 01000 | 1 |
| 0111 | 0100 | 01110 | 1 |
| 1000 | 0000 | 10000 | 1 |
| 1001 | 1111 | 10111 | 0 |
| 1010 | 0111 | 10011 | 1 |
| 1011 | 0110 | 10110 | 1 |
| 1100 | 1110 | 11010 | 1 |
| 1101 | 1010 | 11100 | 1 |
| 1110 | 1100 | 11010 | 1 |
| 1111 | 1101 | 11111 | 1 |

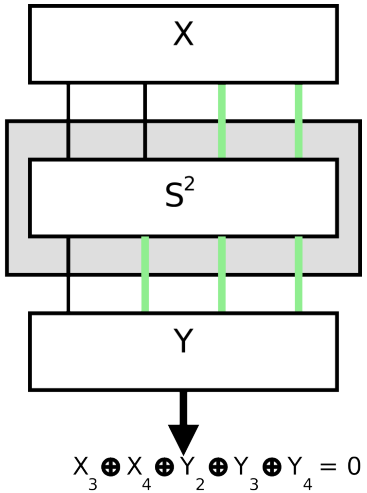
| XXXX (1 2 3 4) | YYYY (1 2 3 4) | XXYYY (3 4 2 3 4) | ⊕ |
|---|---|---|---|
| 0000 | 0011 | 00011 | 0 |
| 0001 | 1000 | 01000 | 1 |
| 0010 | 1001 | 10001 | 0 |
| 0011 | 0101 | 11101 | 0 |
| 0100 | 1011 | 00011 | 0 |
| 0101 | 0001 | 01001 | 0 |
| 0110 | 0010 | 10010 | 0 |
| 0111 | 0100 | 11100 | 1 |
| 1000 | 0000 | 00000 | 0 |
| 1001 | 1111 | 01111 | 0 |
| 1010 | 0111 | 10111 | 0 |
| 1011 | 0110 | 11110 | 0 |
| 1100 | 1110 | 00110 | 0 |
| 1101 | 1010 | 01010 | 0 |
| 1110 | 1100 | 10100 | 0 |
| 1111 | 1101 | 11101 | 0 |

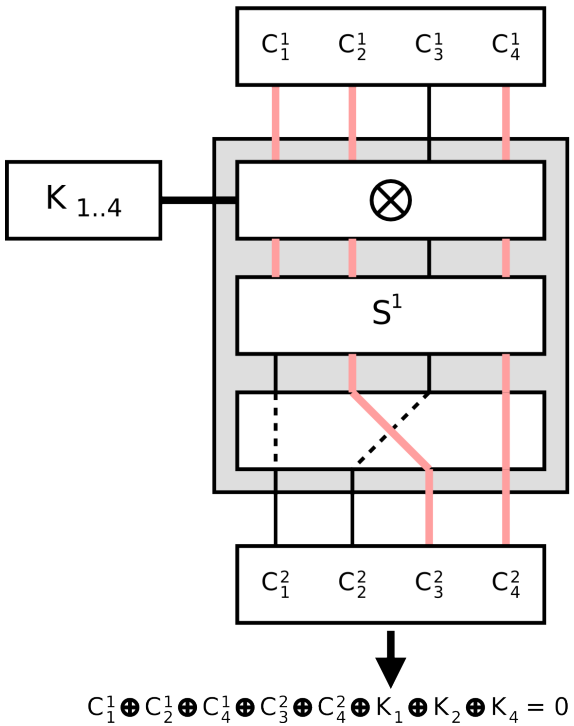
| XXXX (1 2 3 4) | YYYY (1 2 3 4) | XXYYY (2 4 1 3 4) | ⊕ |
|---|---|---|---|
| 0000 | 0011 | 00011 | 0 |
| 0001 | 1000 | 01100 | 0 |
| 0010 | 1001 | 00101 | 0 |
| 0011 | 0101 | 01001 | 0 |
| 0100 | 1011 | 10111 | 0 |
| 0101 | 0001 | 11001 | 1 |
| 0110 | 0010 | 10010 | 0 |
| 0111 | 0100 | 11000 | 0 |
| 1000 | 0000 | 00000 | 0 |
| 1001 | 1111 | 01111 | 0 |
| 1010 | 0111 | 00011 | 0 |
| 1011 | 0110 | 01010 | 0 |
| 1100 | 1110 | 10110 | 1 |
| 1101 | 1010 | 11110 | 0 |
| 1110 | 1100 | 10100 | 0 |
| 1111 | 1101 | 11101 | 0 |

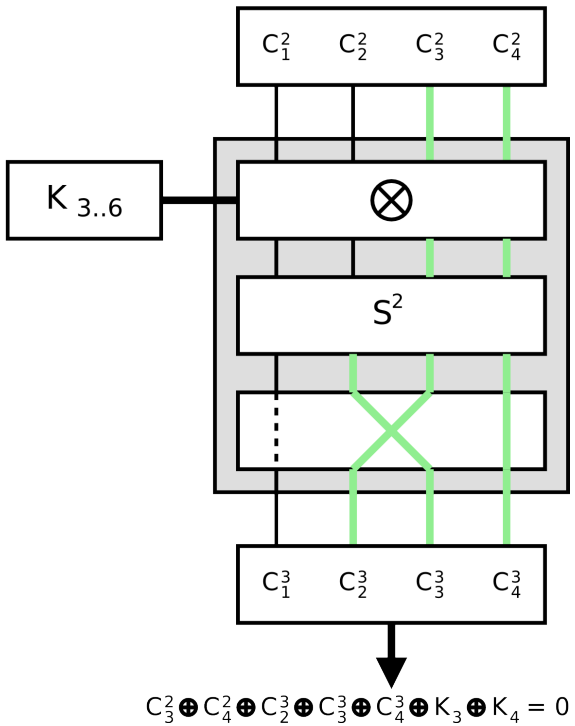
| XXXX (1 2 3 4) | YYYY (1 2 3 4) | XYYY (4 1 2 4) | ⊕ |
|---|---|---|---|
| 0000 | 0011 | 0001 | 0 |
| 0001 | 1000 | 1100 | 0 |
| 0010 | 1001 | 0101 | 0 |
| 0011 | 0101 | 1011 | 0 |
| 0100 | 1011 | 0101 | 0 |
| 0101 | 0001 | 1001 | 1 |
| 0110 | 0010 | 0000 | 0 |
| 0111 | 0100 | 1010 | 0 |
| 1000 | 0000 | 0000 | 0 |
| 1001 | 1111 | 1111 | 0 |
| 1010 | 0111 | 0011 | 0 |
| 1011 | 0110 | 1010 | 0 |
| 1100 | 1110 | 0110 | 1 |
| 1101 | 1010 | 1100 | 0 |
| 1110 | 1100 | 0110 | 0 |
| 1111 | 1101 | 1111 | 0 |

# Cryptanalyse linéaire – exemple, approximation

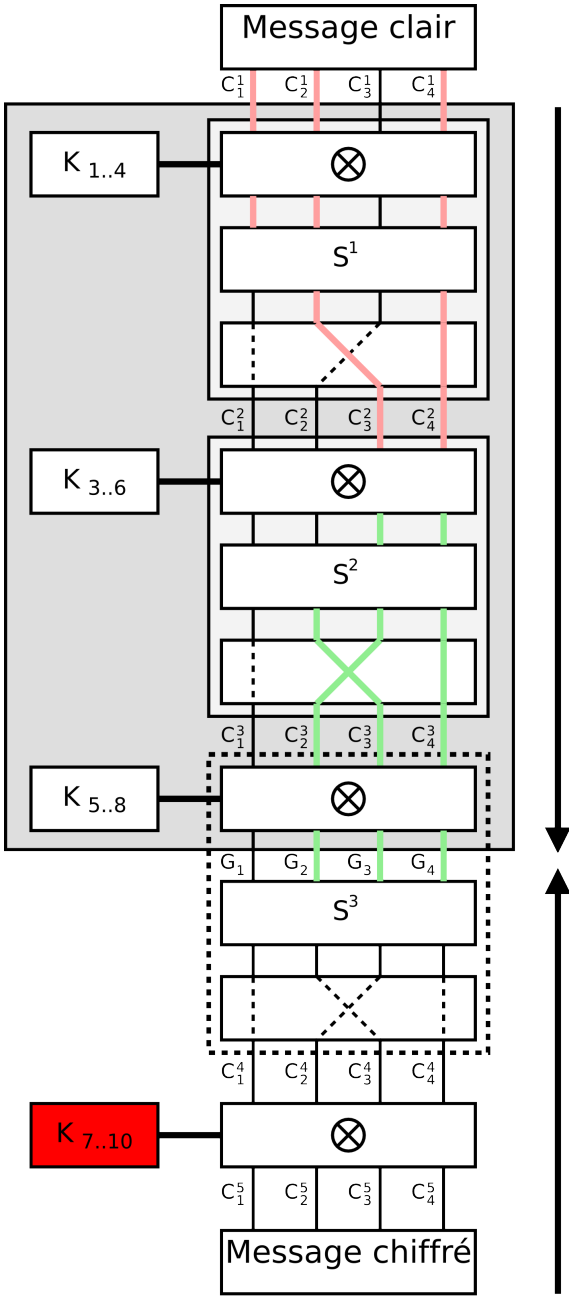


# Cryptanalyse linéaire – exemple, algorithme




$$C_1^1 \oplus C_2^1 \oplus C_4^1 \oplus C_3^2 \oplus C_4^2 \oplus K_1 \oplus K_2 \oplus K_4 = 0$$

$$C_3^2 \oplus C_4^2 \oplus C_2^3 \oplus C_3^3 \oplus C_4^3 \oplus K_3 \oplus K_4 = 0$$

$$C_1^1 \oplus C_2^1 \oplus C_4^1 \oplus \cancel{C_3^2} \oplus \cancel{C_4^2} \oplus K_1 \oplus K_2 \oplus K_4$$

$$\oplus \cancel{C_3^2} \oplus \cancel{C_4^2} \oplus \cancel{C_2^3} \oplus \cancel{C_3^3} \oplus \cancel{C_4^3} \oplus K_3 \oplus K_4 \oplus$$

$$\cancel{C_2^3} \oplus \cancel{C_3^3} \oplus \cancel{C_4^3} \oplus G_2 \oplus G_3 \oplus G_4 \oplus K_6 \oplus K_7 \oplus K_8 = 0$$

$$C_1^1 \oplus C_2^1 \oplus C_4^1 \oplus G_2 \oplus G_3 \oplus G_4$$

$$\boxed{\oplus K_3 \oplus K_4 \oplus K_1 \oplus K_2 \oplus K_4 \oplus K_6 \oplus K_7 \oplus K_8} = 0$$

*Constante*

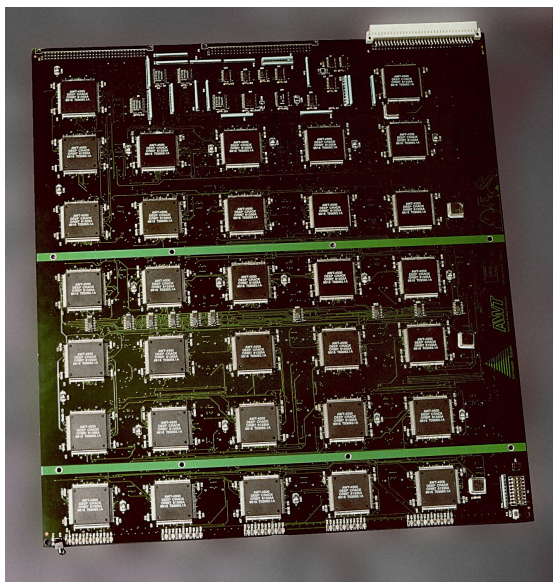
$$C_1^1 \oplus C_2^1 \oplus C_4^1 \oplus G_2 \oplus G_3 \oplus G_4 = 0$$

**for** $k = 0 \,..\, 2^4 - 1$ **do**
  $s_k \leftarrow 0$
  **for all** $(m, c) \in \mathcal{T}$ **do**
    $g \leftarrow c \oplus k$
    $g \leftarrow (\text{Permute}^3)^{-1}(g)$
    $g \leftarrow (S^3)^{-1}(g)$
    **if** $g_2 \oplus g_3 \oplus g_4 \oplus m_1 \oplus m_2 \oplus m_4 = 0$ **then**
      $s_k \leftarrow s_k + 1$
    **end if**
  **end for**
**end for**
$r \leftarrow \underset{x}{\text{argmax}}\ \text{abs}(s_x - |\mathcal{T}|/2)$

# *DES*, cryptanalyse

- Clé sur 56 bits $\Rightarrow 2^{56}$ clés possibles
- Possibilité d'attaques par brute force
  - *Deep Crack* – Cryptography Research, Advanced Wireless Technologie, EFF
  - 29 cartes de 64 puces (1 856 puces spécialisées pour le DES)
  - 90 $m^{ds}$ de clés testées par seconde

  $\Rightarrow$ Environ 5 jours pour tester toutes les possibilités



# Avantages des chiffres symétriques

- Rapides
  - Exemple avec AES
    - Jusqu'à 100 Gb/s sur du matériel spécifique
    - Jusqu'à 250 Mb/s avec du logiciel (MavBook Pro)
- Clés *courtes* : typiquement 80 bits pour résister aux attaques *brutales* (aujourd'hui)

  `www.rsa.com/rsalabs/node.asp?id=2103`
  - DES (ECB) cassé en octobre 1997 (22h avec un matériel spécifique)
  - RC5-56 cassé en octobre 1997 (250j sur Internet)
  - RC5-64 cassé en juillet 2002 (1757j sur Internet)
- Pratiques pour chiffrer des fichiers personnels
  → pas de clé à partager

# Problèmes des chiffres symétriques

- Communication : clé secrète partagée
  Il faut que l'émetteur et le récepteur se fassent confiance, et gardent soigneusement la clé secrète
- Comment distribuer/renouveler la clé ?
  - Chiffrer la nouvelle clé de session avec l'ancienne
  - Chiffrer la clé de session avec une clé spécifique de chaque matériel
    $\Rightarrow$ site de confiance (répertoire)
  - Cryptographie quantique
  - Utiliser un système à clé publique (Diffie-Hellman)

# Chiffres à clé publique : $k_c \neq k_d$

- Connaissant $k_c$, il est "impossible" de trouver $k_d$
  - $k_d$ est "privé" : seul celui qui connaît $k_d$ peut déchiffrer
  - $k_c$ est public : tout le monde peut chiffrer → répertoire de clés publiques
- Exemples
  - RSA (1978) → difficulté de factoriser de grands nombres[10]
  - El Gamal (1985) → difficulté de calcul des logarithmes discrets[4]

# *RSA* – Rivest, Shamir, Adleman

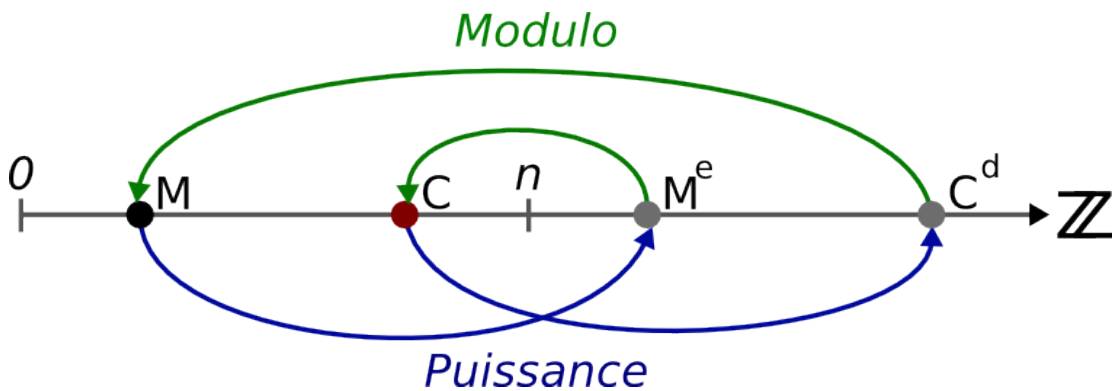
- Création des clés
  - Choisir $p$ et $q$ deux nombres premiers distincts

  ⇒ Calculer le *module de chiffrement* $n$, $n = p \cdot q$

  ⇒ Calculer l'*indicatrice d'Euleur* de $n$, $\phi(n) = (p-1) \cdot (q-1)$
  - Choisir l'*exposant de chiffrement* $e$, un entier premier avec $\phi(n)$,

  ⇒ Calculer l'*exposant de déchiffrement* $d$, $e \cdot d \equiv 1 \mod \phi(n)$
  *Algorithme d'Euclide étendu*

  ⇒ $k_c = \{e, n\}$ $k_d = \{d, n\}$
- Chiffrement : $C = M^e \mod n$, avec $M < n$
- Déchiffrement : $M = C^d \mod n$
- Décomposition de $n$ en produit de facteurs premiers → $p$ et $q$ $\mathcal{O}(e^n)$



# *RSA* – Rivest, Shamir, Adleman

- Exemple
  - Création des clés
    - $p = 5,\ q = 11$

    $\Rightarrow n = p \cdot q = 5 \cdot 11 = 55$

    $\Rightarrow \phi(n) = (p-1) \cdot (q-1) = (5-1) \cdot (11-1) = 4 \cdot 10 = 40$

    - $e = 3$

    $\Rightarrow d = 27$ Vérification : $d \cdot e \stackrel{?}{\equiv} 1 \mod \phi(n)$

    $d \cdot e = 3 \cdot 27 = 81 = 2 \cdot 40 + 1 \equiv 1 \mod 40$

    $\Rightarrow k_c = \{n, e\} = \{55, 3\} \quad k_d = \{n, d\} = \{55, 27\}$
  - Chiffrement de $M = 19$
    - $C = M^c \mod n = 19^3 \mod 55 = 6859 \mod 55 = 39$
  - Déchiffrement de $C = 39$
    - $M = C^d \mod n = 39^{27} \mod 55 = 39$
    - $39^{27} = 909377887614652551975371341130628025063947 9$

# Avantages des chiffres à clé publique

- Pas de confiance mutuelle entre émetteur et récepteur
- Gestion de clé "facile"
  - Répertoire public de clés publiques ou distribution entre pairs
  - La clé privée ne doit "jamais" être transmise
- Possibilité d'utilisations nouvelles : distribution de clés symétriques, signatures, certificats, etc.

# Distribution de clés symétriques

- Exemple : Alice génère aléatoirement une clé de session $K$ (symétrique) et la chiffre avec la clé publique de Bob
- Exemple : Diffie-Hellmann
  - Alice (A) et Bob (B) souhaitent communiquer (ex : groupe fini $\mathbb{Z}/p\mathbb{Z}$)

| | | | |
|---|---|---|---|
| A | $\leftrightarrow$ | B | Alice et Bob se mettent d'accord sur un nombre premier $p$ |
| A | $\leftrightarrow$ | B | Alice et Bob conviennent d'une racine primitive $g$ |
| A | ⏲ | B | Alice choisi un nombre secret $0 \leq a \leq p-1$ |
| A | $\rightarrow$ | B | Alice envoie la valeur $g^a \mod p$ à Bob |
| A | ⏲ | B | Bob choisi un nombre secret $0 \leq b \leq p-1$ |
| A | $\leftarrow$ | B | Bob envoie la valeur $g^b \mod p$ à Alice |
| A | ⏲ | B | Alice calcule la clé secrète $K = (g^b \mod p)^a \mod p$ |
| A | ⏲ | B | Bob calcule la clé secrète $K = (g^a \mod p)^b \mod p$ |

  - Eve écoute les transmissions
    - Eve connait $p$, $g$, $g^a \mod p$, $g^b \mod p$
    - Peut-il calculer $a$ et $b$ ?
    - $a = \log_g(g^a)$ et $b = \log_g(g^b) \mod p$
    - Problème du logarithme discret

# Distribution de clés symétriques

**Question** : Est-ce qu'Alice est sûre d'échanger une clé avec Bob ?

| | | | |
|---|---|---|---|
| A | $\rightarrow$ | B | Alice envoie la valeur $g^a \mod p$ à Bob |
| A | $\leftarrow$ | B | Bob envoie la valeur $g^b \mod p$ à Alice |
| A | ⏱ | B | Alice calcule la clé secrète $K = (g^b \mod p)^a \mod p$ |
| A | ⏱ | B | Bob calcule la clé secrète $K = (g^a \mod p)^b \mod p$ |

# Distribution de clés symétriques

**Question** : Est-ce qu'Alice est sûre d'échanger une clé avec Bob ?

| | | | |
|---|---|---|---|
| A | $\rightarrow$ | B | Alice envoie la valeur $g^a \mod p$ à Bob |
| A | $\leftarrow$ | B | Bob envoie la valeur $g^b \mod p$ à Alice |
| A | ⏱ | B | Alice calcule la clé secrète $K = (g^b \mod p)^a \mod p$ |
| A | ⏱ | B | Bob calcule la clé secrète $K = (g^a \mod p)^b \mod p$ |

**Réponse** : authenticité de $g^b \mod p = B$ ?

# Distribution de clés symétriques

**Question** : Est-ce qu'Alice est sûre d'échanger une clé avec Bob ?

| | | | |
|---|---|---|---|
| A | $\rightarrow$ | B | Alice envoie la valeur $g^a \mod p$ à Bob |
| A | $\leftarrow$ | B | Bob envoie la valeur $g^b \mod p$ à Alice |
| A | ⏱ | B | Alice calcule la clé secrète $K = (g^b \mod p)^a \mod p$ |
| A | ⏱ | B | Bob calcule la clé secrète $K = (g^a \mod p)^b \mod p$ |

**Réponse** : authenticité de $g^b \mod p = B$ ?

Eve en homme dans le milieu peut envoyer $g^c \mod p = C$ à Alice.
$\rightarrow$ $g$ et $p$ étant publics.

# Distribution de clés symétriques

**Question** : Est-ce qu'Alice est sûre d'échanger une clé avec Bob ?

| | | | |
|---|---|---|---|
| A | $\rightarrow$ | B | Alice envoie la valeur $g^a \mod p$ à Bob |
| A | $\leftarrow$ | B | Bob envoie la valeur $g^b \mod p$ à Alice |
| A | ⏱ | B | Alice calcule la clé secrète $K = (g^b \mod p)^a \mod p$ |
| A | ⏱ | B | Bob calcule la clé secrète $K = (g^a \mod p)^b \mod p$ |

**Réponse** : authenticité de $g^b \mod p = B$ ?

Eve en homme dans le milieu peut envoyer $g^c \mod p = C$ à Alice.
$\rightarrow$ $g$ et $p$ étant publics.

**Solution** : authentification de Bob. Primitives de signature
**Intuition et exemple** : $k_c \neq k_d$, $E_{k_d}(B)$ peut être calculé seulement par Bob qui possède la clé secrète $k_d$

# Problèmes des chiffres à clé publique

- Calculs complexes : lents ($\backsim$ 1 Mbits/s), clé longue (1024 ou 2048 bits), sauf avec des courbes elliptiques ($\backsim$ 160 bits)
  Records actuels
  - RSA 200, 200 chiffres (2005) : 663 bits (BSI, U.Bonn, CWI)
  - RSA 640/173 (2005) : 4,5 mois à 80 opteron 2,2 GHz (BSI, U.Bonn)
  - Logarithme discret 613 bits (2005) : 17 jours à 64 Itanium2 (Bull, U. Versailles)
  - Certicom ECC2-109 (2004) : 15 mois à 2900 calculateurs
- Problèmes spécifiques
  - Intégrité des répertoires de clés publiques
  - Durée de vie des clés
  - Révocation
  - Nécessité de partager des clés privées ?
  - Limitation des algorithmes, par exemple : chiffrer un petit $M$ par RSA

# Factorisation – Défis

`www.rsa.com/rsalabs/node.asp?id=2092`

`www.crypto-world.com/FactorWorld.html`

| Nombre | Nombre de décimales | Date | Vitesse | Algorithme |
|---|---|---|---|---|
| C116 | 116 | 1990 | 275 MIPS années | mpqs |
| RSA-120 | 120 | 06/1993 | 830 MIPS années | mpqs |
| RSA-129 | 129 | 04/1994 | 5000 MIPS années | mpqs |
| RSA-130 | 130 | 04/1996 | 1000 MIPS années | gnfs |
| RSA-140 | 140 | 02/1999 | 2000 MIPS années | gnfs |
| RSA-155 | 155 | 08/1999 | 8000 MIPS années | gnfs |
| C158 | 158 | 01/2002 | 3,4 Pentium 1GHz années | gnfs |
| RSA-160 | 160 | 03/2003 | 2,7 Pentium 1GHz années | gnfs |
| RSA-576 | 174 | 12/2003 | 13,2 Pentium 1GHz années | gnfs |
| C176 | 176 | 05/2005 | 48,6 Pentium 1GHz années | gnfs |
| RSA-200 | 200 | 05/2005 | 121 Pentium 1GHz années<br>55 Opteron 2,2GHz années | gnfs |
| RSA-640 | 193 | 11/2005 | 30 Opteron 2,2GHz années | gnfs |

# Factorisation – Défis *RSA-640*

- Durée : 4,5 mois
- Matérielle : 80 Opterons 2,2GHz
- Vitesse : 30 Opteron 2,2GHz années
- Le nombre : 193 chiffres – 640 bits
- La factorisation

3107418240490043721350750035888567930037346022842
7275457201619488232064405180815045563468296717232
8678243791627283803341547107310850191954852900733
77248227835257423864540146917366024776523466 09

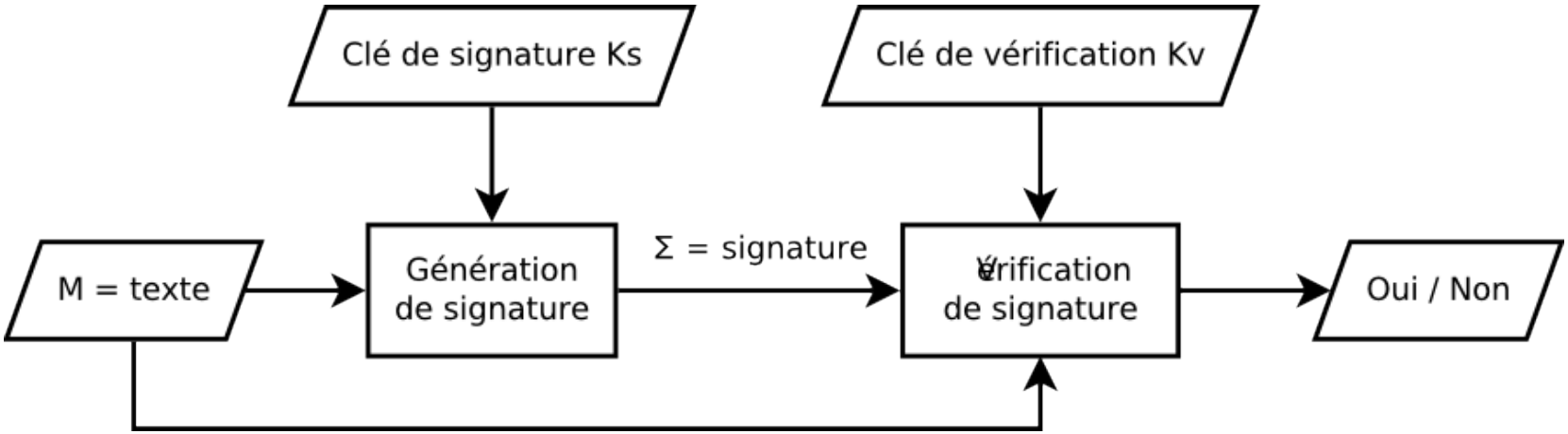
=

1634733645809253848443133883865090859841783670033
0923121811108523893331001045081512121181675115 79

x

1900871281664822113126851573935413975471896789968
5154936666385390880271038021044989571912614655 71

# Signature (intégrité)



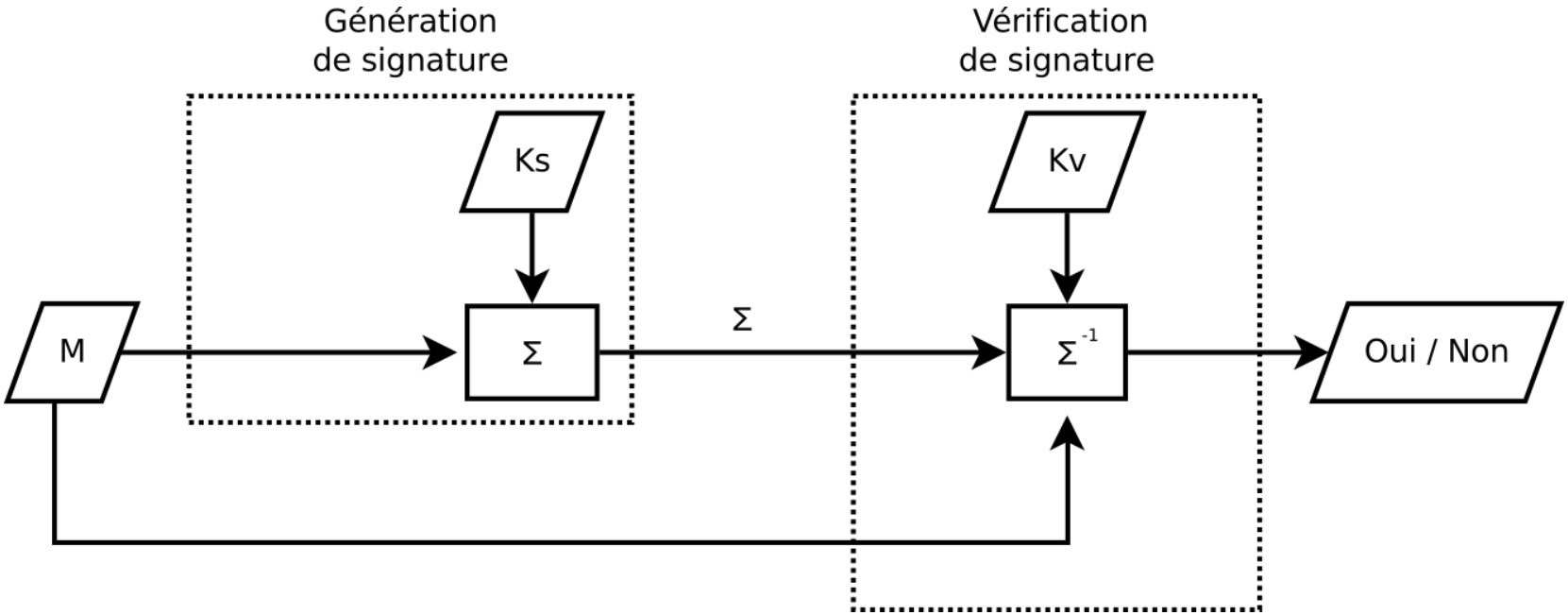

- $k_s$ = clé de signature $k_v$ = clé de vérification
- Intégrité
  - Sans connaître $k_s$, "impossible" de générer une signature valide
  - Il est "impossible" de trouver $k_s$, connaissant $M$ et $\Sigma$ (clair connu)
  - Il est "impossible" de trouver $k_s$, choisissant $M$ (clair choisi)
- Pratique : $\Sigma$ est de taille fixe et relativement petit, quelque soit la taille de $M$

# Signature symétriques – $k_s = k_v$ : secrètes !

- *MAC* : *Message Authentication Code*
- Exemples
  - *CBC-MAC* : Dernier bloc du *DES* en mode *CBC*
  - $\Sigma = \{M\}_{k_s} \stackrel{?}{=} \Sigma' = \{M\}_{k_v}$
- Inconvénients
  - Signataire et vérificateur doivent se faire confiance
  - Répudiation possible $\Rightarrow$ la signature n'est pas valable devant un juge

# Signatures à clé publique – $k_s \neq k_v$

## Principe




## Limitations

**Exemple de RSA** : $\Sigma = E_{k_d}(M)$. $M$ petit ! $|M| < |\text{module } n|$

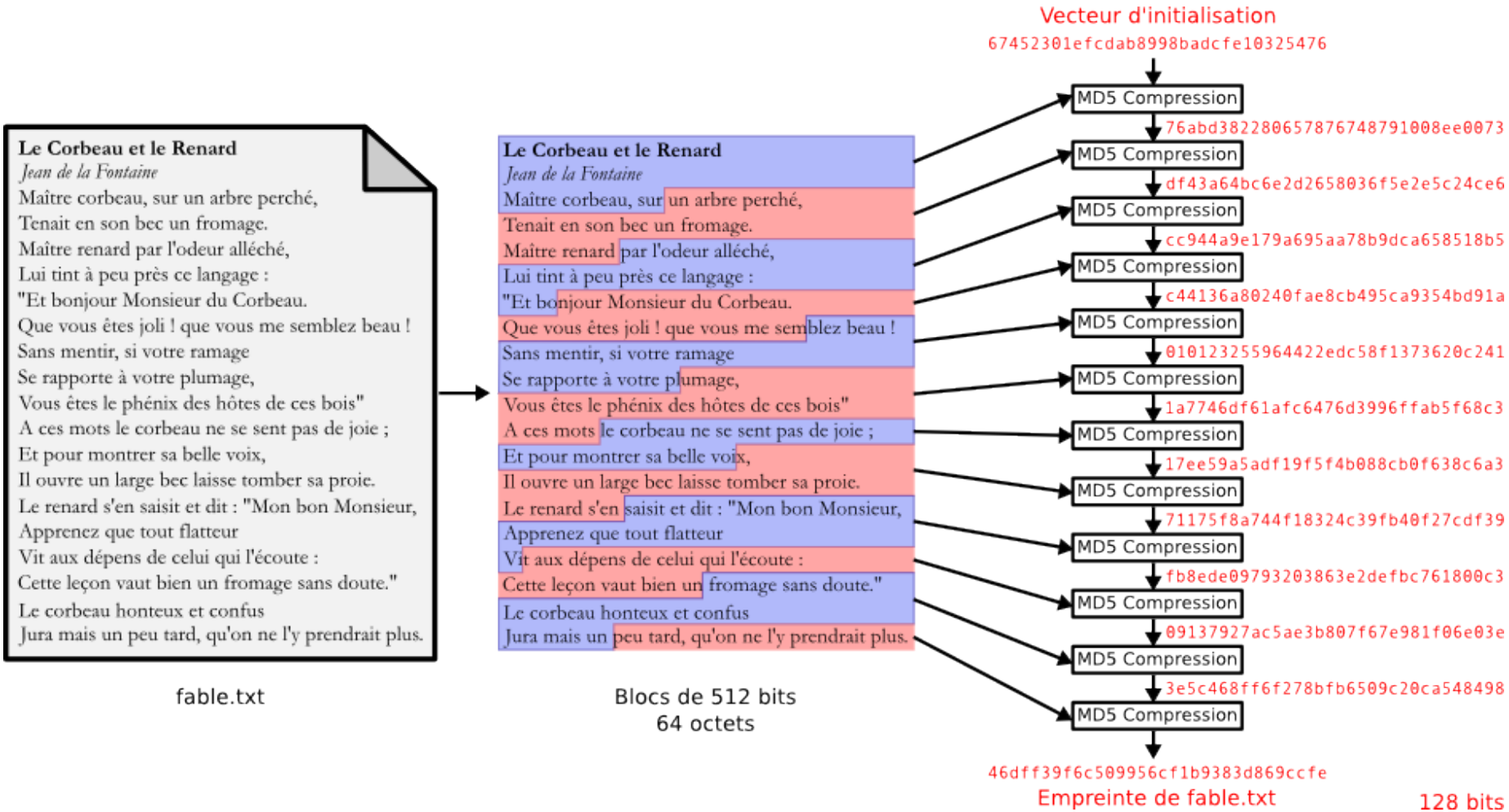
→ Fonctions de hashage !

# Fonctions de hachage → empreinte, condensat

- *One-way hash function* $\mathcal{H}$
  - L'empreinte $\mathcal{H}(M)$ est de taille fixe $n$, quelque soit la longueur de $M$
  - Si 1 bit de $M$ est changé, environ $n/2$ bits de $\mathcal{H}(M)$ changent
  - Connaissant $M$, il est facile de calculer $\mathcal{H}(M)$
  - Collisions : $M \neq M', \mathcal{H}(M) = \mathcal{H}(M')$
    $|M|$ non borné et $|\mathcal{H}(M)|$ fixe $\Rightarrow \exists$ un nombre infini de collisions
  - Sécurité : sauf attaque par force brute ($\backsim 2^n$ essais)
    - **Préimage** : connaissant $x < 2^n$, il est "impossible" de trouver $M$ tel que $\mathcal{H}(M) = x$
    - **Seconde préimage** : connaissant $M$, il est "impossible" de trouver $M'$ tel que $M \neq M'$ et $\mathcal{H}(M) = \mathcal{H}(M')$
    - **Collision** : il est très difficile ($\backsim 2^{n/2}$ essais) de trouver $M$ et $M'$ tel que $M \neq M'$ et $\mathcal{H}(M) = \mathcal{H}(M')$
- Exemples : *DES – CBC* (64 bits), *MD5* (128 bits), *SHA-1* (160 bits)

# MD5

- $M$ est découpé en $z$ blocs de 512 bits, $m_1$, $m_2$, . . . , $m_z$
  $h_1 = \mathcal{F}(constante, m_1)$ ; $h_2 = \mathcal{F}(h_1, m_2)$ ; . . . ; $h_z = \mathcal{F}(h_{z-1}, m_z) = \mathcal{H}(M)$



# Fonctions de hachage – Application : Intégrité

- Communications : contre interception et modification
  Transmettre le message et l'empreinte par des canaux indépendants
- Fichiers : détection de modification (*Tripwire*[6])
  - Sur une machine correcte, calculer les empreintes des fichiers stables (OS, programmes, configuration, etc.) et les stocker de manière sûre (par exemple, chiffrées)
  - Périodiquement, ou en cas de doute, ou au démarrage, recalculer les empreintes et les comparer (sur une machine saine)
- Faiblesse découverte récemment (MD5, etc.)
  - `www.stachliu.com/research_collisions.html`
  - 2006 : collision MD5 en 3/4 d'heure sur Pentium 4 1,6GHz ($\backsim$ 50 bits)
  - 2009 : collision SHA-1 en $\backsim 2^{52}$ essais (théoriquement)
- RIPEMD (128 bits, 160 bits, 256 bits), SHA-256, SHA-512, Whirlpool (512 bits)

# Signature symétriques et fonctions de hashage

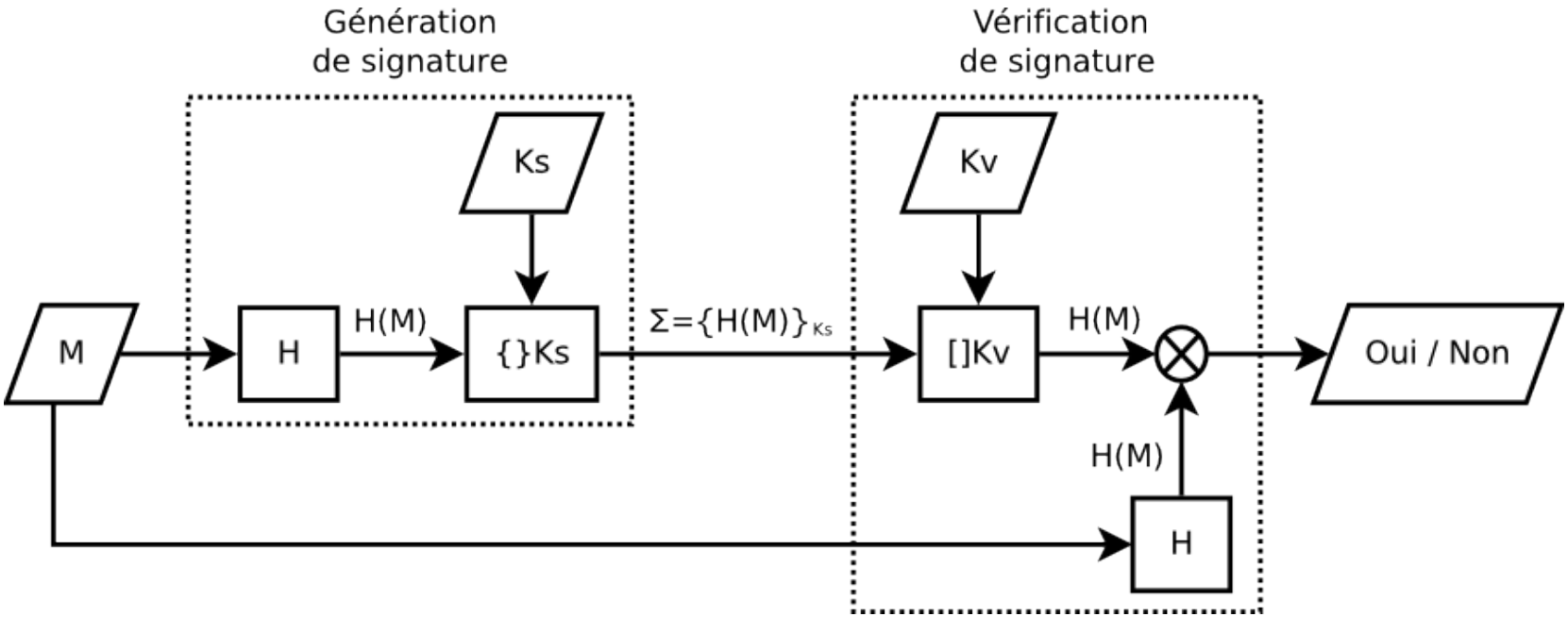
Constructions de signatures symétriques avec fonctions de hashage

- *H-based MAC* : $\Sigma = \mathcal{H}(k_s \cdot M) \quad \stackrel{?}{\rightarrow} \quad \Sigma' = \mathcal{H}(k_v \cdot M)$
  - Vulnérabilités au attaques de *length extension* : *SHA-1, MD5*
  - $\mathcal{H}(K \cdot M \cdot \text{bad}) = \mathcal{H}(\mathcal{H}(K \cdot M) \cdot \text{bad})$
  - Faiblesse
- Variante (HMAC) :

  $\Sigma = \mathcal{H}(k_s \cdot \mathcal{H}(k_s \cdot M)) \quad \stackrel{?}{\rightarrow} \quad \Sigma' = \mathcal{H}(k_s \cdot \mathcal{H}(k_s \cdot M))$

# Signatures à clé publique – $k_s \neq k_v$

- Exemple : *RSA*
  - $k_s$ = clé de signature = clé de chiffrement $k_c$ privée
  - $k_v$ = clé de vérification = clé de déchiffrement $k_d$ publique



# Propriétés des signatures à clé publique

- Vérifiables par des tiers : preuve de responsabilité du signataire
  *la clé de signature ne doit jamais être transmise*
- Peuvent servir à sécuriser les répertoires de clés publiques
  Infrastructure de gestion de clés (*IGC* ou *PKI*)
  - Chaque entrée de répertoire est signée par une *autorité de certification* (*AC* ou *CA*)
  - Les clés publiques des autorités de certification sont dans une répertoire, chacune signée par une *AC* de plus haut niveau, etc.
- Attention : être sûr de ce qu'on signe !
  *What you sign is not necessarily what you see*

# Propriétés des signatures à clé publique

Julius. Caesar
Via Appia 1
Rome, The Roman Empire

May, 22, 2005

To Whom it May Concern:

Alice Falbala fulfilled all the requirements of the Roman Empire intern position. She was excellent at translating roman into her gaul native language, learned very rapidly, and worked with considerable independence and confidence.

Her basic work habits such as punctuality, interpersonal deportment, communication skills, and completing assigned and self-determined goals were all excellent.

I recommend Alice for challenging positions in which creativity, reliability, and language skills are required.

I highly recommend hiring her. If you'd like to discuss her attributes in more detail, please don't hesitate to contact me.

Sincerely,

Julius Caesar

Julius. Caesar
Via Appia 1
Rome, The Roman Empire

May, 22, 2005

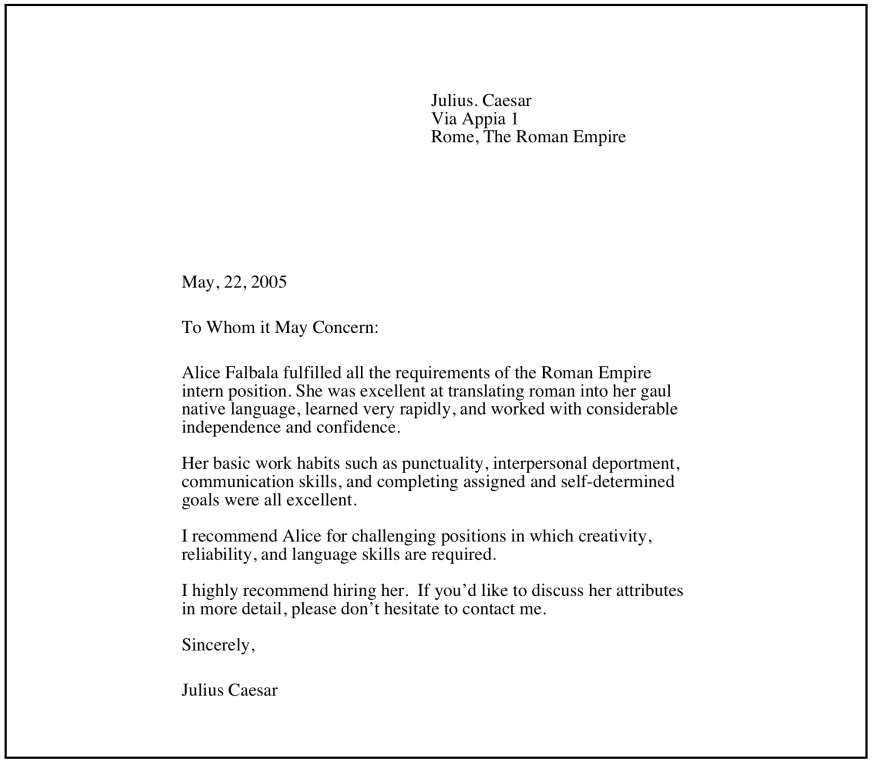
Order:

Alice Falbala is given full access to all confidential and secret information about GAUL.

Sincerely,

Julius Caesar

# Propriétés des signatures à clé publique



Julius. Caesar
Via Appia 1
Rome, The Roman Empire

May, 22, 2005

To Whom it May Concern:

Alice Falbala fulfilled all the requirements of the Roman Empire intern position. She was excellent at translating roman into her gaul native language, learned very rapidly, and worked with considerable independence and confidence.

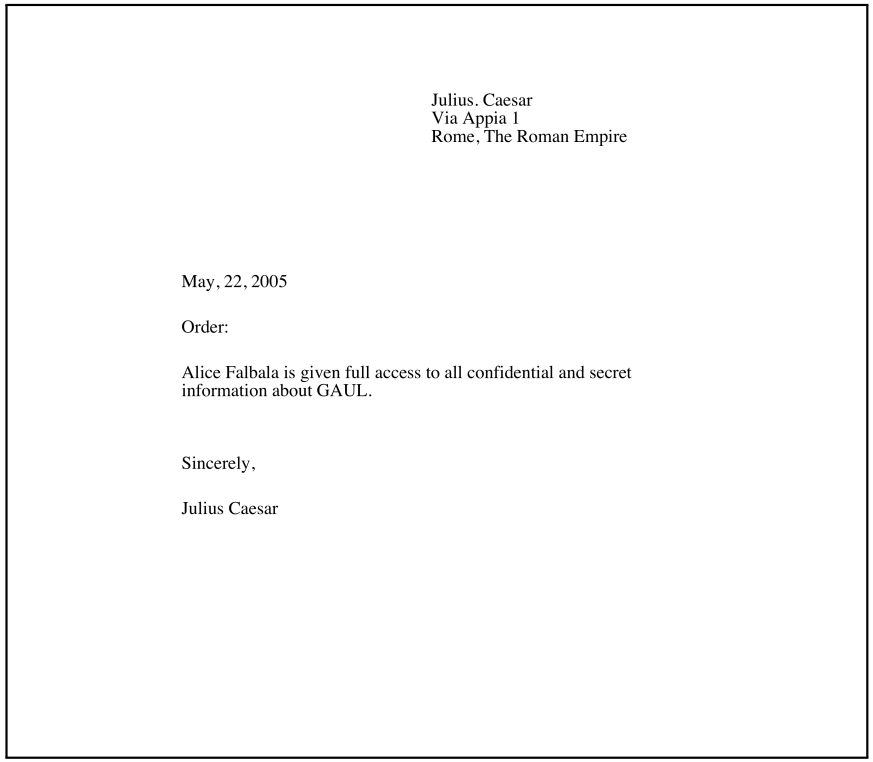
Her basic work habits such as punctuality, interpersonal deportment, communication skills, and completing assigned and self-determined goals were all excellent.

I recommend Alice for challenging positions in which creativity, reliability, and language skills are required.

I highly recommend hiring her. If you'd like to discuss her attributes in more detail, please don't hesitate to contact me.

Sincerely,

Julius Caesar



Julius. Caesar
Via Appia 1
Rome, The Roman Empire

May, 22, 2005

Order:

Alice Falbala is given full access to all confidential and secret information about GAUL.

Sincerely,

Julius Caesar

```
$ cat letter_of_rec.ps | openssl md5
a25f7f0b29ee0b3968c860738533a4b9
$ cat order.ps | openssl md5
a25f7f0b29ee0b3968c860738533a4b9
$ diff order.ps letter_of_rec.ps
Binary files order.ps and letter_of_rec.ps differ
```

# Constructions fondamentales : modèles de confiance

**Question** : confiance dans la distribution des clés publiques ($k_c$)

- Root of trust / Pinned
- TOFU
- Web of trust
- PKI

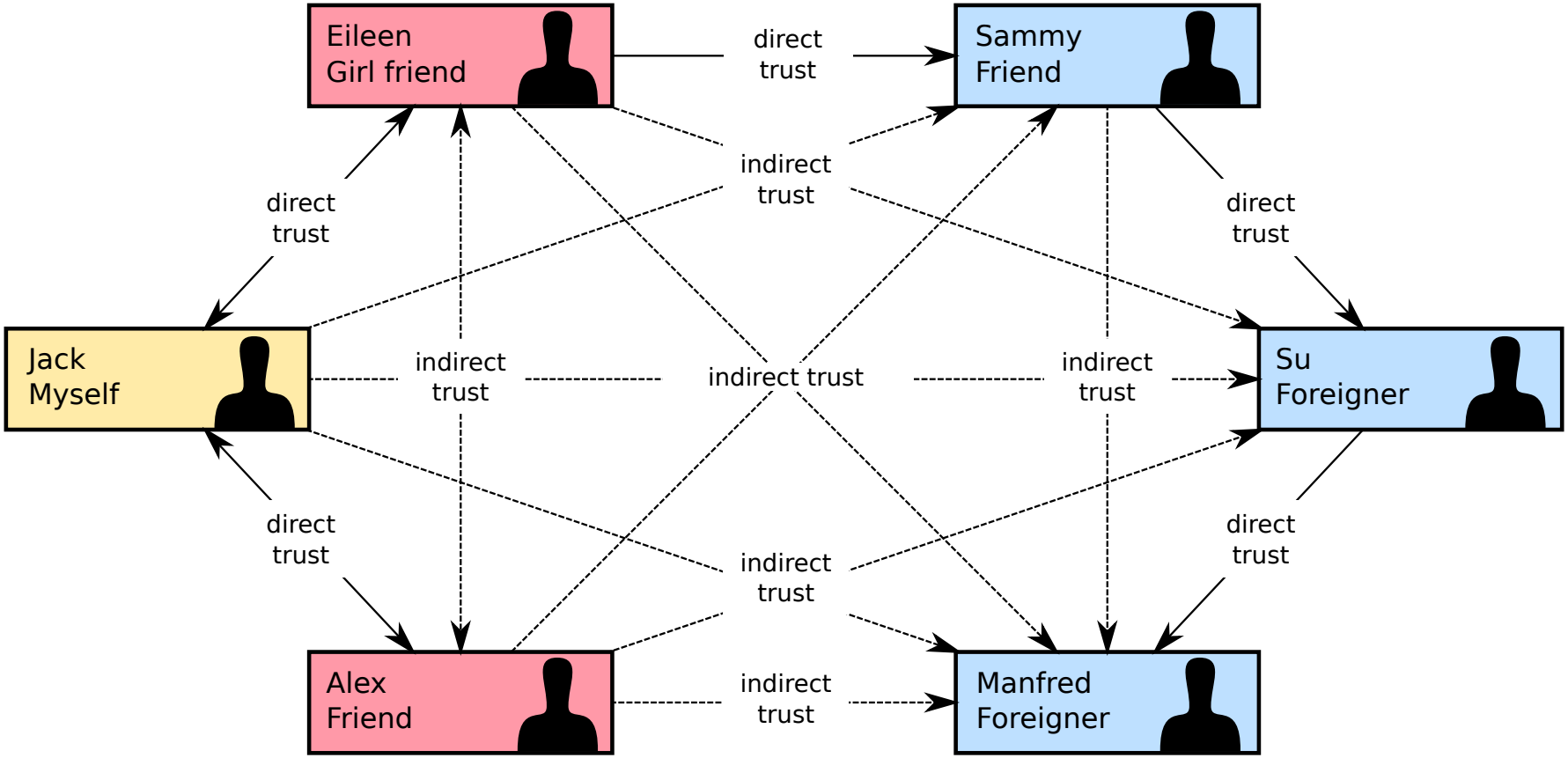
# *Trust On First Use (TOFU)*

## Modèle de *Secured SHell* (SSH)

```
$ ssh m0rgan.net
The authenticity of host 'm0rgan.net (89.234.156.200)' can't be established.
ECDSA key fingerprint is SHA256:43XUsUoU2Bh7WaYN/t7STAwlr9bsRqTEwOl2hadVWMk.
Are you sure you want to continue connecting
(yes/no/[fingerprint])?
```

# *Web of Trust*

Modèle de confiance de *Pretty Good Proviacy* (PGP)




https://en.wikipedia.org/wiki/Web_of_trust#/media/File:Web_of_Trust-en.svg

# *Trust On First Use (TOFU)*

## Modèle de *Secured SHell* (SSH)

```
$ cat ~/.ssh/known_hosts
gitlab.enseeiht.fr,147.127.176.52 ecdsa-sha2-nistp256 AAAAE2V..
iritfs.enseeiht.fr,147.127.80.15 ecdsa-sha2-nistp256 AAAAE2Vj..
176.158.13.37 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHA..
maison ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAA..
10.10.10.140 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAy..
147.127.133.144 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzd..
192.168.0.2 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyN..
enseeiht.fr,193.48.203.34 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQE..
yAuSEjjFqA6QcIxmVnMjDWAgmPHhAxkXsLHmHYycq4XAzQ/xmIeWn/1AaCCHC..
192.168.1.3 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyN..
canardscitrons.nohost.me,80.67.176.23 ecdsa-sha2-nistp256 AAA..
cyclope.enseeiht.fr,147.127.133.163 ecdsa-sha2-nistp256 AAAAE..
```

## Vulnérable ?

# *Trust On First Use (TOFU)*

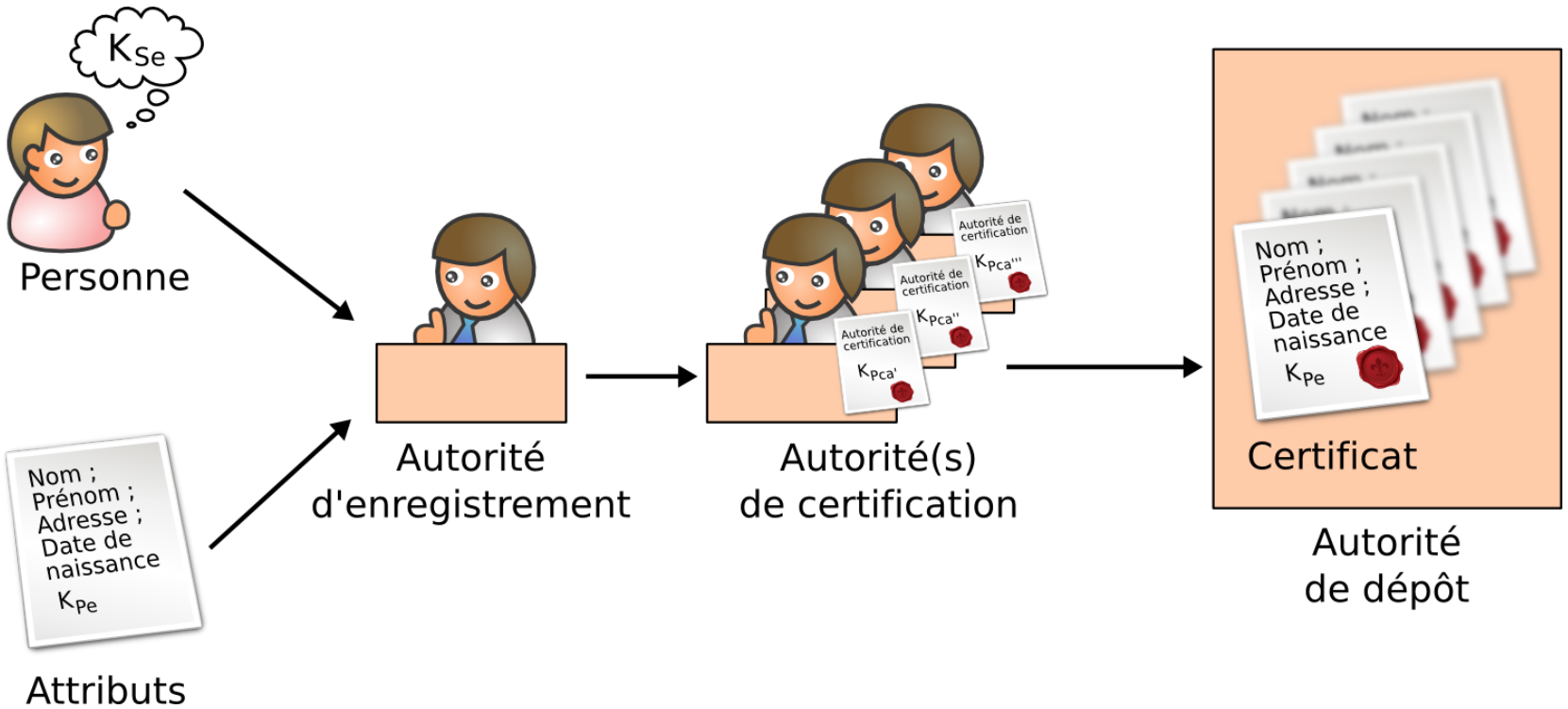
## Modèle de *Secured SHell* (SSH)

```
$ cat ~/.ssh/known_hosts
gitlab.enseeiht.fr,147.127.176.52 ecdsa-sha2-nistp256 AAAAE2V..
iritfs.enseeiht.fr,147.127.80.15 ecdsa-sha2-nistp256 AAAAE2Vj..
176.158.13.37 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHA..
maison ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAA..
10.10.10.140 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAy..
147.127.133.144 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzd..
192.168.0.2 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyN..
enseeiht.fr,193.48.203.34 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQE..
yAuSEjjFqA6QcIxmVnMjDWAgmPHhAxkXsLHmHYycq4XAzQ/xmIeWn/1AaCCHC..
192.168.1.3 ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyN..
canardscitrons.nohost.me,80.67.176.23 ecdsa-sha2-nistp256 AAA..
cyclope.enseeiht.fr,147.127.133.163 ecdsa-sha2-nistp256 AAAAE..
```

**Vulnérable ?**

**Réponse** : homme dans le milieu au *first use* !

# Certificats et PKI – exemple *X509*



# Autres fonctions cryptographiques – sujets

- Cryptographie à clés publiques basées sur l'identité
- Stéganographie
- *Watermarking*
- Génération de nombres aléatoires et pseudo-aléatoires
- Générateur de nombres premiers
- Ecrous (*key escrow*)
- Vote
- Horodatage
- Cryptanalyse
- Protocoles

# Sommaire

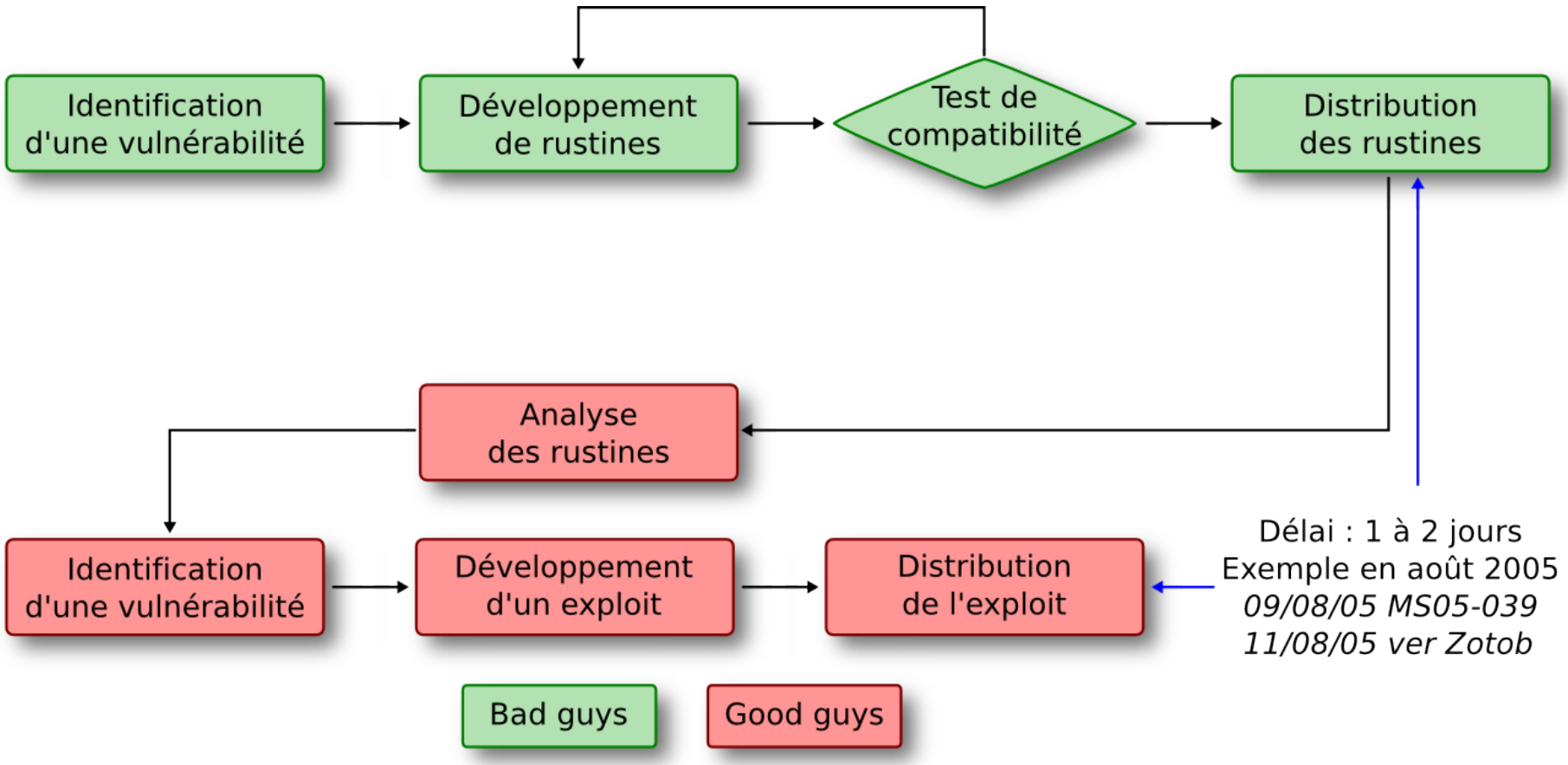
# Prévention des vulnérabilités

- Vulnérabilités = fautes de conception ou de configuration
- Les systèmes commerciaux actuels sont trop complexes pour être sans fautes
- Objectifs divergents
  - Disponibilité / sécurité (*TCP/IP*)
  - Rentabilité-efficacité / sécurité
- Il existe des outils pour éviter d'introduire des vulnérabilités classiques (par exemple des débordements de tampons)

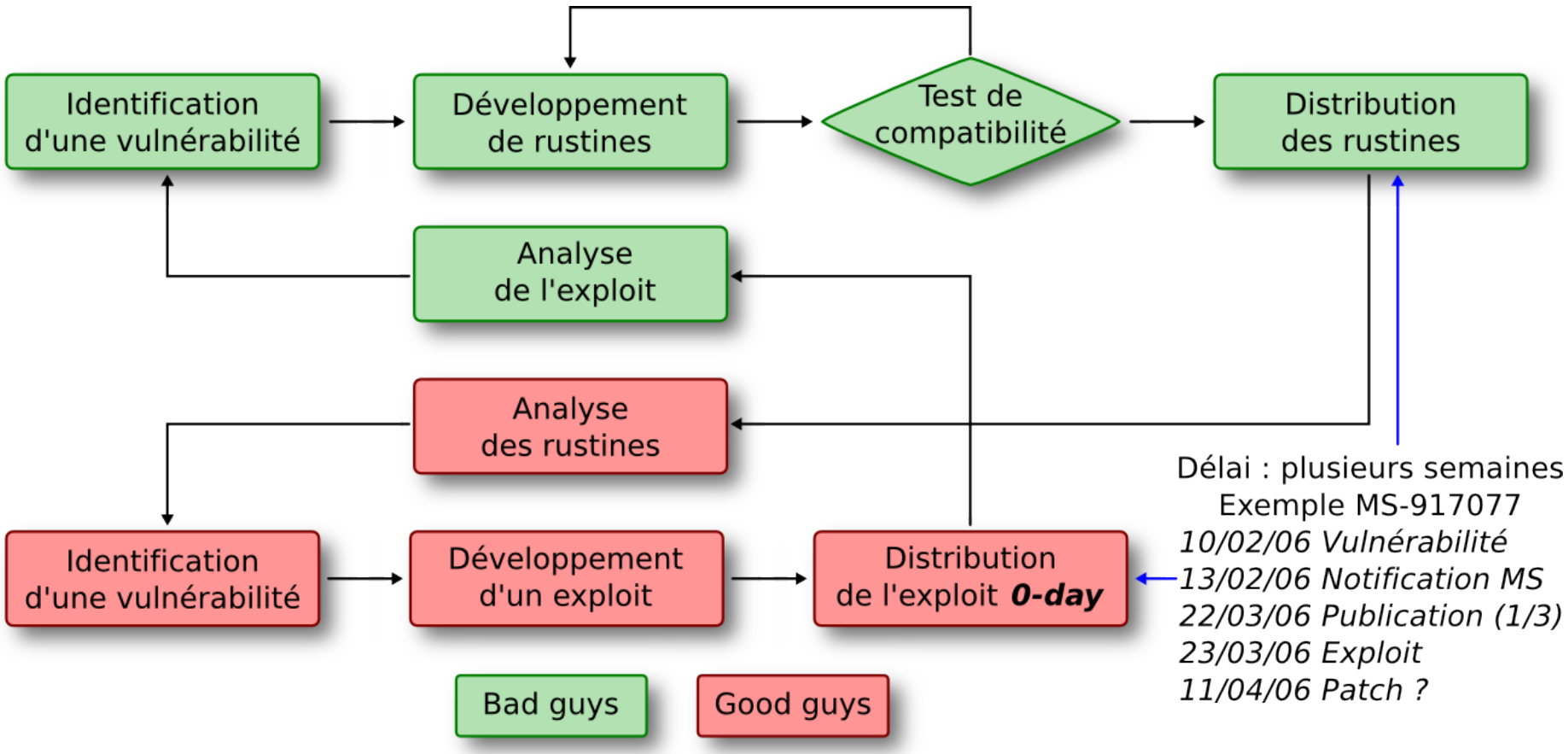
# Elimination des vulnérabilités

- Cycle habituel
  - Identification d'une nouvelle vulnérabilité
  - *Exploit*
  - *Patches* (rustines)
  - Nouvelle version
- Mais
  - Nombreuses alertes → quelles sont celles qui sont pertinentes ?
  - Certains *patches* sont imparfaits → élimination d'une fonctionnalité indispensable
  - Certaines applications indispensables ne sont plus compatibles

# Cycle de vie des patches



# Cycle de vie des exploits



# Sommaire

# Cloisonnement

- Empêcher tout communication/interaction qui n'est pas nécessaire
  - Isoler les systèmes de développement des systèmes opérationnels, les systèmes de surveillance des systèmes surveillés
  - Fragmenter et disséminer l'information, séparer les pouvoirs
- Pare-feux
  - Filtrer les adresses sources/destination (*IP* + n° port), entrée/sortie
  - Traduction d'adresse (*NAT*)
  - Mandataire d'application (*proxy*) pour vérifier les protocoles d'application
    - Liaison avec *IDS stateful*
    - Option : outil *anti-reconnaissance*, *Intrusion Prevention System* (*IPS*)

# Sommaire

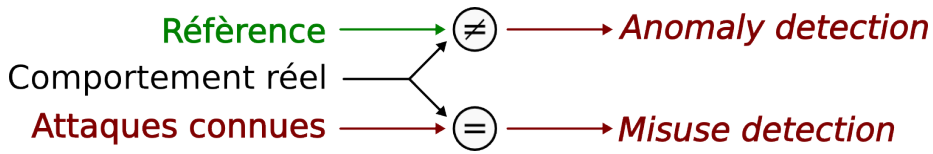
# Audit – journalisation

- Enregistrer toutes les opérations liées à la sécurité (réussies ou non)
  - Connexion/déconnexion d'utilisateurs
  - Création/modification/destruction d'informations de sécurité
    - Droits d'accès
    - Mots de passe
    - Enregistrements d'audit
    - . . .
  - Changement de privilèges
- Informations enregistrées
  - Date, heure
  - Identité de l'utilisateur
  - Type d'opération, référence des objets
  - . . .

# Sommaire

# Détection d'intrusion – *IDS*

- Principe : détection d'erreurs dues à des intrusions
- Deux familles de techniques : analogie avec les détecteurs de virus




  - *Anomaly detection* : par discrimination entre les comportements normaux (utilisateurs non-malveillants) et les comportements anormaux (intrus) : profils statistiques, systèmes experts, systèmes immunitaires, etc.
  - *Misuse detection* : par reconnaissance de signatures correspondant à des attaques connues (*stateless*, *stateful*)
- Implémenté dans chaque calculateur (*host-based IDS*) ou sur des machines observant le réseau (*netword-based IDS*)
- Problème
  - Taux de fausses alarmes (*false positives*)
  - Taux de non détection (*false negatives*)
- Les autres mécanismes de détection d'erreurs peuvent aussi être efficaces vis-à-vis des intrusions

# Sommaire

# Sommaire

# Sommaire

# Politiques de sécurité

## Politique de sécurité

Une politique de sécurité est l'ensemble des lois, règles et pratiques qui régissent la façon dont l'information sensible et les autres ressources sont gérées, protégées et distribuées à l'intérieur d'un système spécifique.[?]

- Modèle de sécurité → formalisme mathématique

# Politiques de sécurité

- Objectifs à satisfaire, par exemple :
  - Confidentialité : le dossier médical ne peut être consulté que par le patient et son ou ses médecins traitants
  - Intégrité : un chèque de plus de 1000 € doit être signé par le Président et le Trésorier
  - Disponibilité : si la carte et le *PIN* sont valides, le distributeur de billets doit fournir l'argent dans les 30 secondes
- Règles, par exemple :
  - Un fichier ne peut être lu que par les utilisateurs autorisés par le propriétaire du fichier
  - Un message de type *chèque de plus de 1000 €* n'est valide que s'il est signé par $P_1$ et $T_2$ et que les signatures sont valides
  - L'insertion d'une carte lance automatiquement l'action, etc.

# Cohérence d'une politique

- La politique est cohérente si, partant d'un état quelconque où les objectifs sont satisfaits, il n'est pas possible d'atteindre, en respectant les règles, un état où ils ne sont plus satisfaits
- Intérêts d'un modèle formel
  - Décrire de manière précise les objectifs et les règles
  - Prouver des propriétés sur la politique (cohérence, complétude, etc.) et sur son implémentation par le système informatique

# Politique, protection et contrôle d'accès

- Les règles doivent être mises en œuvre par des mécanismes (matériels, logiciels)
- Facile à imaginer pour les règles du type "*il est permis de . . .*" ou "*il est interdit de . . .*" → mécanismes de protection
  instructions privilégiées, contrôle d'accès à la mémoire, contrôle à l'ouverture de fichiers, etc.
  → autorisation : confidentialité, intégrité
- Difficile pour les règles du type "*il est obligatoire de . . .*" ou "*il est recommandé de . . .*"
  → actions automatiques, gestion des ressouces, etc : intégrité, disponibilité

# Politique d'autorisation

- Un sujet a un droit d'accès sur un objet
  $\Leftrightarrow$ le sujet est autorisé à exécuter la méthode d'accès sur cet objet
  - Sujet : processus qui s'exécute pour le compte d'un utilisateur
  - Utilisateur : personne physique ou service identifié dans le système
  - Objet : conteneur d'information, défini par un nom, un état et des méthodes, par exemple : fichier, périphérique, processus, etc.

# Modèle *HRU*

- L'état de sécurité du système est défini par :
  - $D$ : l'ensemble de tous les droits
  - $S$ : l'ensemble des sujets courants
  - $O$ : l'ensemble des objets courants, $S \subseteq O$
  - $A$ : l'ensemble des droits courants de chaque sujet sur chaque objet
    $A$ est représenté par une matrice avec une ligne par sujet $s_i$ et une colonne pour chaque objet $o_j$
    $A_{ij} = d_{ij}$ avec $d_{ij} \subseteq D$

$$(s_i, o_j, d_k) \text{ est vrai} \Leftrightarrow s_i \text{ a le droit } d_k \text{ sur } o_j$$
$$d_{ij} = \{d_k \in D | (s_i, o_j, d_k)\}$$

# Politique d'autorisation discrétionnaires

*DAC : Discretionary Access Control*

- Les droits d'accès à chaque information sont manipulés par le responsable de l'information (généralement le propriétaire), à sa discrétion
- Les droits peuvent être accordées à chaque utilisateur individuellement ou à des groupes d'utilisateurs ou les deux

# Politique d'autorisation discrétionnaires

- Exemple : protection des fichiers UNIX
  - Règles
    1. Un utilisateur peut créer librement des fichiers dont il devient propriétaire
       $(A, F, \text{créer}) \xrightarrow{A} (A, F, \text{propriétaire}) \wedge (A, F, \text{écrire}) \wedge (A, F, \text{lire})$
    2. Les droits d'accès à un fichier sont définis librement par le propriétaire : par exemple, il peut décider quels utilisateurs sont autorisés à lire le fichier
       $(A, F, \text{propriétaire}) \xrightarrow{A} (B, F, \text{lire})$
  - Objectif
    - *Un utilisateur non-autorisé à lire un fichier ne peut obtenir aucune information contenue dans le fichier (même avec la complicité d'un utilisateur autorisé)* → impossible à garantir
  - Exemple
    3. $S = \{s_1, s_2, s_3\}$
    4. $O = \{f_1, f_2\}$
    5. $D = \{\text{propriétaire}, \text{lire}, \text{écrire}\}$
    6. $A = \{(s_1, f_1, \text{propriétaire})\}$

(2 et 6) 7. $(s_1, f_1, \text{propriétaire}) \xrightarrow{s_1} (s_2, f_1, \text{lire})$

(1 et 2) 8. $(s_2, f_2, \text{créer}) \xrightarrow{s_2} (s_2, f_2, \text{écrire}) \wedge (s_3, f_2, \text{lire})$

(7 et 8) 9. $(s_2, f_1, \text{lire}) \wedge (s_2, f_2, \text{écrire}) \wedge (s_3, f_2, \text{lire}) \xrightarrow{s_2} (s_3, k(f_1), \text{lire})$

# Inconvénient des politiques *DAC*

- Possibilité d'abus de pouvoir
  (par malveillance ou par maladresse)
  - S'il est possible pour un utilisateur légitime d'accéder à certains objets ou d'en modifier les droits d'accès, il est possible qu'un *Cheval de Troie* en fasse de même
  - Si un utilisateur a le droit de lire une information, il a (en général) automatiquement le droit de la divulger à n'importe qui
  - Il est difficile de corriger les effets d'une divulgation

# Politique d'autorisation obligatoires

*MAC* : *Mandatory Access Control*

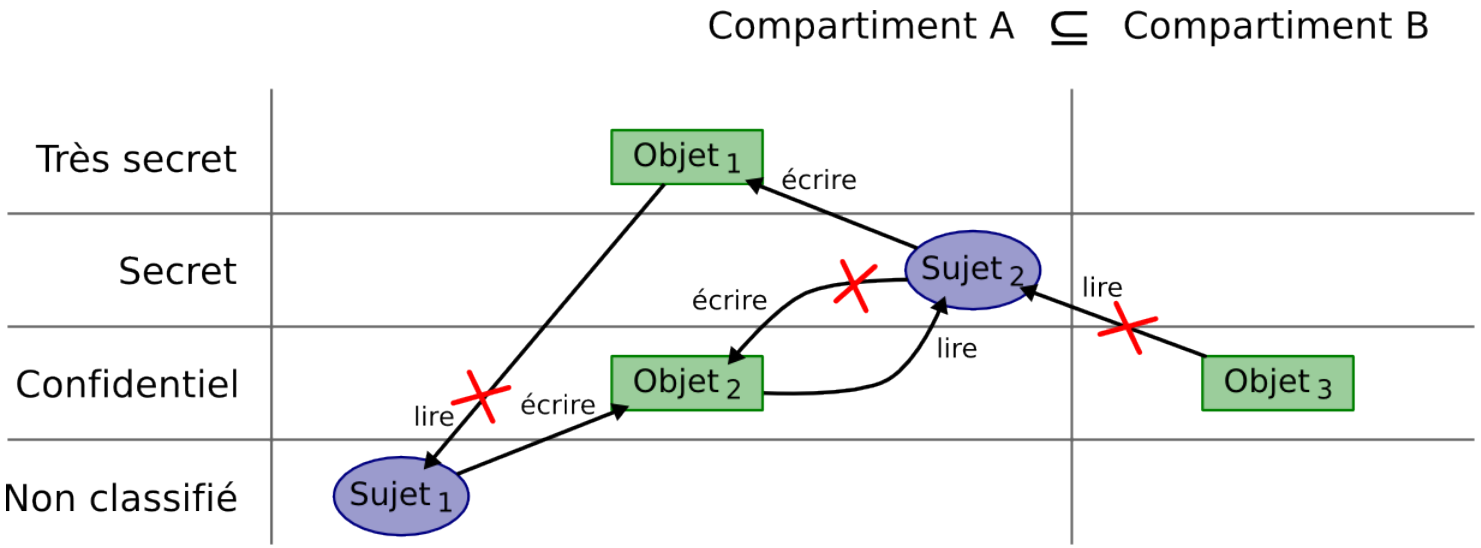
- Des règles incontournables sont imposées, en plus des règles discrétionnaires
- Exemple : politique de confidentialité multi-niveau *militaire*
  Des classes sont assignées aux utilisateurs (habilitation) et aux objets (classification)
  Une classe est définie par :

| Un niveau (ordonné) | Un compartiment = {catégories} |
|---|---|
| Non-classifié | Cryptographie |
| Confidentiel | Nucléaire |
| Secret | OTAN |
| Très secret | Irak |
| | . . . |

# Politique de *Bell-LaPadula* (condifentialité)

- A chaque sujet $s_i$ correspond une habilitation $h(s_i)$ avec un niveau $n_i$ et un compartiment $\Sigma_i$
- A chaque objet $o_j$ correspond une classification $c(o_j)$ avec un niveau $n_j$ et un compartiment $\Sigma_j$
- Règle simple :
  $(s_i, o_j, \text{lire}) \Rightarrow n_j \leq n_i \wedge \Sigma_j \subseteq \Sigma_i$ $\quad (h(s_i)$ *domine* $c(o_j))$
- Règle étoile :
  $(s_i, o_j, \text{lire}) \wedge (s_i, o_k, \text{écrire}) \Rightarrow n_j \leq n_k \wedge \Sigma_j \subseteq \Sigma_k$ $\quad (c(o_k)$ *domine* $c(o_j))$
- Propriété :
  Si $h(s_n)$ *ne domine pas* $c(o_i)$, il n'existe pas de suite telle que
  $(s_l, o_i, \text{lire}) \wedge (s_l, o_j, \text{écrire}) \wedge (s_m, o_j, \text{lire}) \wedge \ldots \wedge (s_x, o_k, \text{écrire}) \wedge (s_n, o_k, \text{lire})$
  Interdire à tout sujet d'obtenir des informations d'un objet de niveau supérieur à son habilitation
  $\Rightarrow$ Pas de fuite d'information possible

# Inconvénients de Bell-LaPadula

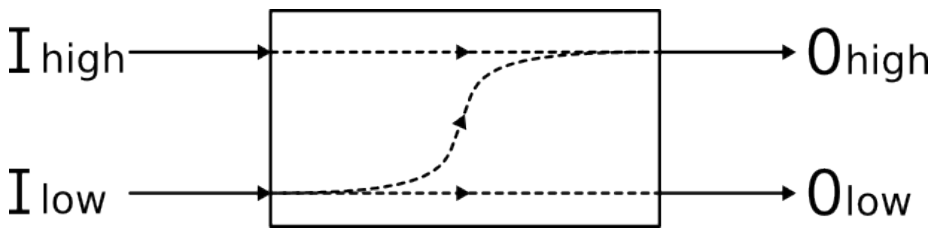



- Surclassification : au fur et à mesure que l'information est traitée, sa classification augmente

⇒ *Trusted process* pour déclassifier

# Autres politiques de confidentialité

- *Muraille de Chine*
  Chez les agents de change (conflits d'intérêt)
- Modèle de non interférence : $O_{\text{low}}$ ne dépend pas de $I_{\text{high}}$

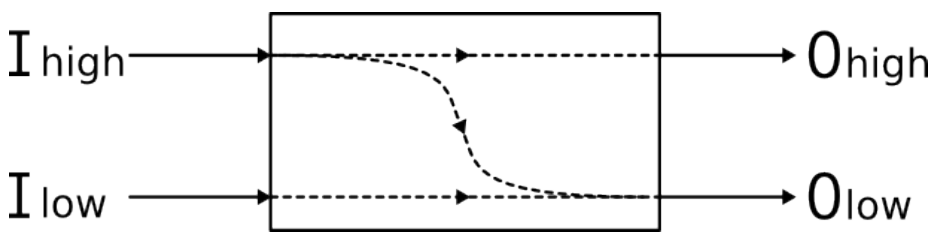


# Politique de *Biba* (intégrité)

- Multiple niveaux d'intégrité (crédibilité, vérification, etc.)
  - A chaque sujet $s_i$ correspond un niveau $is(s_i)$
  - A chaque objet $o_j$ correspond un niveau $io(o_j)$
- Règles
  - $(s_i, o_j, \text{observer}) \Rightarrow is(s_i) \leq io(o_j)$
  - $(s_i, o_j, \text{modifier}) \Rightarrow io(o_j) \leq is(s_i)$
  - $(s_i, s_j, \text{invoquer}) \Rightarrow is(s_j) \leq is(s_i)$
- Propriété : empêcher la *contamination* des niveaux élevés : diffusion de fausses informations, propagation d'erreur, etc.
- Inconvénient : dégradation progressive des niveaux d'intégrité

# Autres politiques d'intégrité

- Politique de *Clark-Wilson* (transactions financières)
  - Deux niveaux d'intégrité
    - *UDI* (données non contraintes)
    - *CDI* (données contraintes), vérifiables par des *IVP* (*Integrity Verification Procedures*), ne pouvant être manipulées que par des *TP* (*Transformation Procedures*) certifiées
  - Règles : listes de relations autorisées
    - *CDI* $\leftrightarrow$ *TP*, Utilisateurs $\leftrightarrow$ *TP* (avec éventuellement *separation of duty*), *UDI* $\leftrightarrow$ *CDI* par (*TP* + *IVP*)
- Modèle de non-interférence : $O_{\text{high}}$ ne dépend pas de $I_{\text{low}}$



# Politiques basées sur les rôles

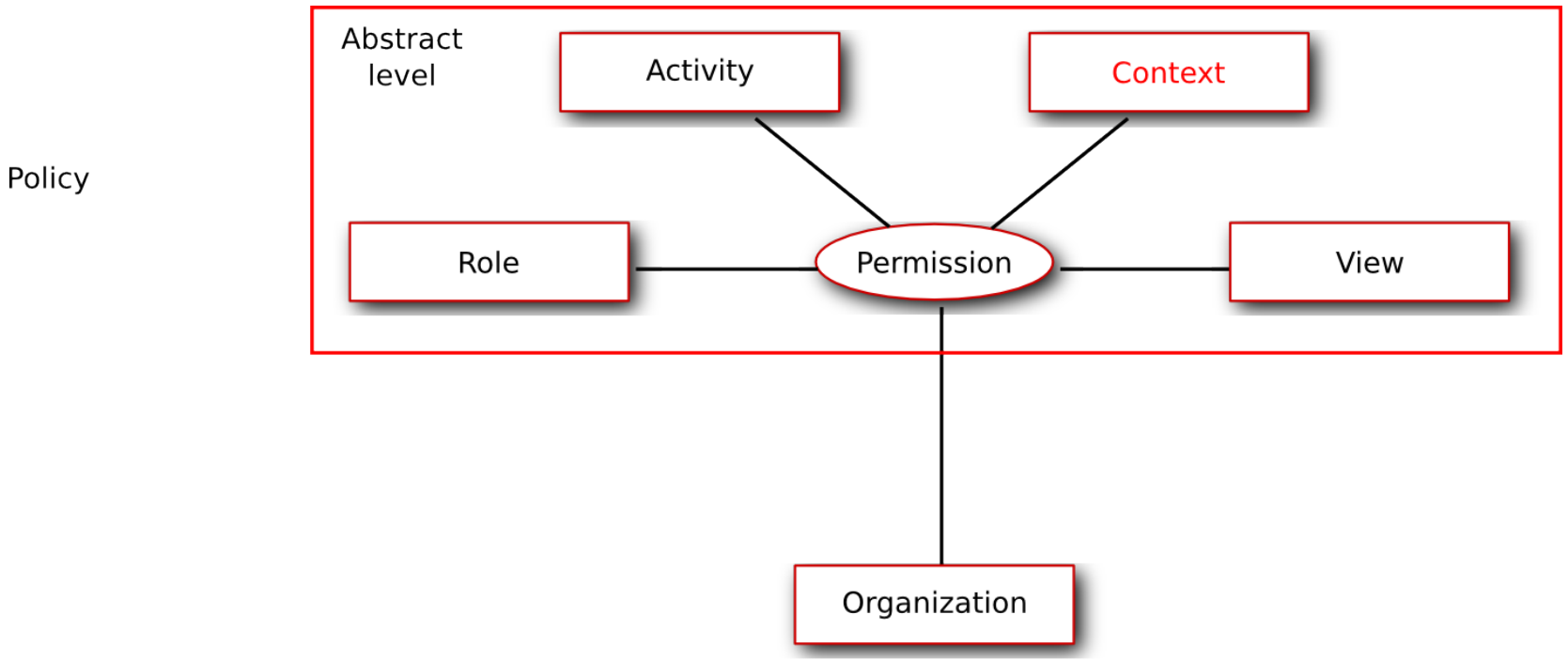
*RBAC* : *Role-Based Access Control*

- On définit des rôles, pour représenter des fonctions dans l'organisation
- On associe à chaque rôle les privilèges (ensemble de droits) nécessaires pour remplir la fonction
- On associe à chaque utilisateur le(s) rôle(s) qu'il peut jouer dans l'organisation
- Administration facilitée
  - Les rôles et leurs privilèges changenent rarement
  - Il suffit d'identifier les rôles que peut jouer un utilisateur

# *OrBAC – Organization-Based Access Control*

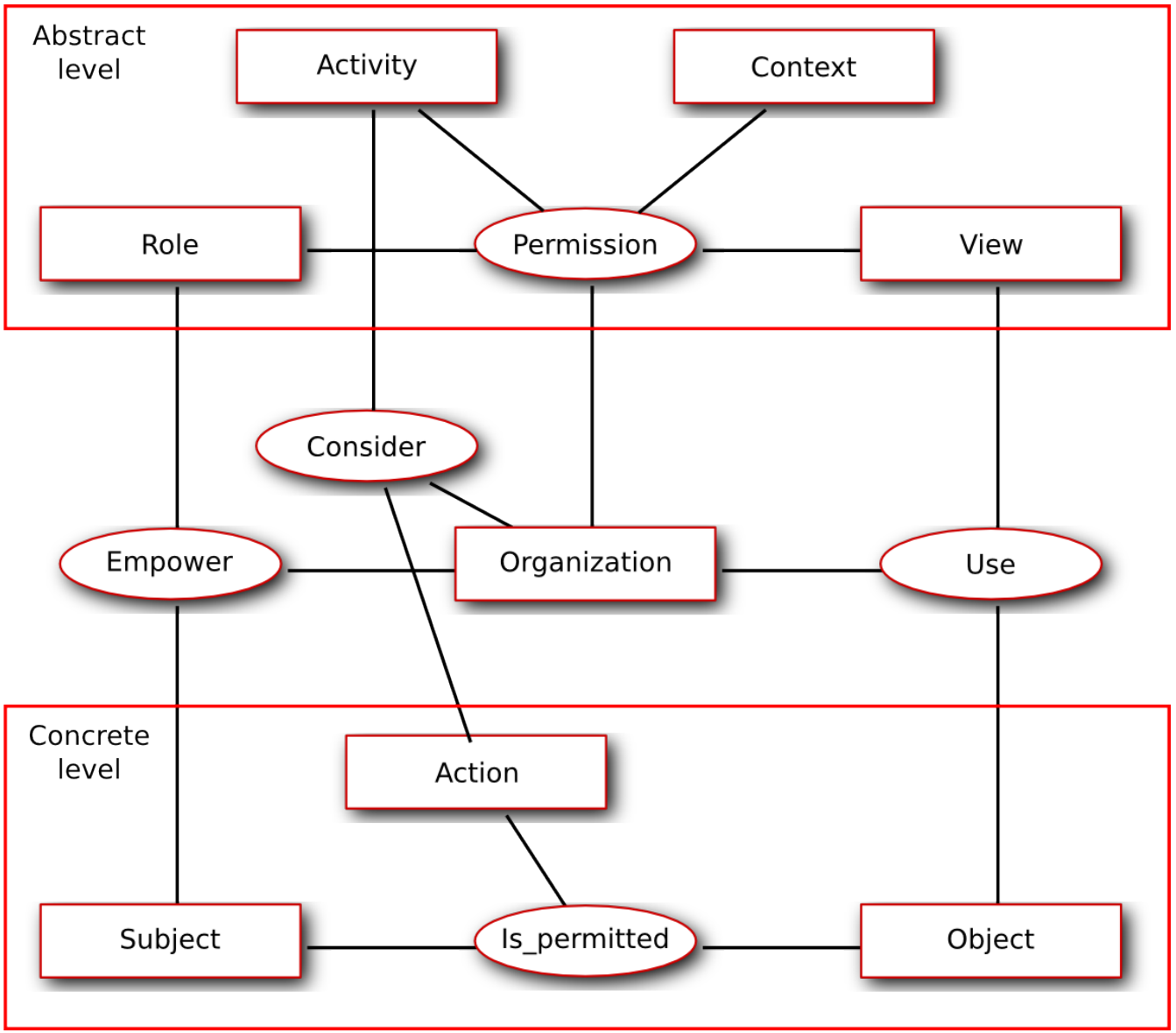
- Modèle conçu dans *MP6*
  Modèles et politiques de sécurité des systèmes d'information et de communication en santé et social
  *Ernst & Young*, *IRIT*, *LAAS-CNRS*, *ONERA*, *ENST-Bretagne*, etc.
- Abstractions
  - User → rôle
  - Objet → vue
  - Action → activité
- Liaisons entre niveaux abstrait (politique) et concret (mécanismes de contrôle d'accès) : définies par l'organisation
- Règles
  - Définies au niveau abstrait
  - Permissions/interdictions + obligations
  - Validées par le contexte (concret)

# OrBAC – Organization-Based Access Control




- La politique est définie au niveau abstrait → définition de règles
  - Permission (Organization, Role, Activity, View, $B$(context))
  - Interdiction (Organization, Role, Activity, View, $B$(context))
  - Obligation (Organization, Role, Activity, View, $B$(context))

# OrBAC – Organization-Based Access Control



# Sommaire

# Authentification des utilisateurs

*Nécessaire pour l'autorisation et l'audit*

- Authentification = identification + vérification de l'identité
- Identité = information (non confidentielle) spécifique à une personne, connue au moins par elle et par le vérificateur : nom, numéro, etc.
- Vérification de la correspondance entre l'identité et la personne, en utilisant :
  - Quelque chose qu'elle connaît (mot de passe, informations personnelles, etc.) ou qu'elle sait faire (reconnaissance de forme, association d'idées, etc.)
  - Quelque chose qu'elle possède : badge, carte à puce, etc.
  - Quelque chose qui lui est propre (biométrie) : empreinte digitale, signature, voix, fond de l'œil, iris, forme de la main, etc.
  - Ou plusieurs de ces moyens : carte à puce + PIN, etc.

# Méthodes de vérification

- Secret partagé entre la personne et le vérificateur (mot de passe, informations personnelles, etc.)
- Secret correspondant à une caractéristique biométrique (stockée par le vérificateur ou non), non falsifiable, non rejouable
- Secret connu par la personne, vérifié par des informations ou protocoles publics (sans apport de connaissance, *zero-knowledge*

# Qualité de l'authentification

- La qualité des systèmes d'authentification dépend du taux d'acceptation à tort (*false acceptance rate*, *FAR*) et tu taux de rejet à tort (*false rejection rate*, *FRR*)

  | | | | |
  |---|---|---|---|
  | Empreinte digitale | $FAR \approx 10^{-6}$ | $FRR \approx 10^{-3}$ | (SAGEM, Compaq, NEC) |
  | Iris | $FAR \approx 10^{-12}$ | $FRR \approx 10^{-4}$ | (Sensar, IriScan) |

- Il faut distinguer si la victime dont on prend l'identité est consentante (par exemple, transmet volontairement son mot de passe) ou non : supériorité des systèmes biométriques
- Mais, les systèmes biométtriques ont des limitations : falsification (prothèses), handicapés, acceptation sociale, difficulté de révocation, etc.

# Références I


La petite histoire des virus informatiques...
Site Internet.
https://graphism.fr/la-petite-histoire-des-virus-informatiques%E2%80%A6/.

National institute of standards and technology.
Site Internet.
http://www.itl.nist.gov/lab/bulletns/bltnjun06.htm.

M. de Marlès.
Histoire de Marie Stuart, Reine d'Ecosse.

Taher El Gamal.
A public key cryptosystem and a signature scheme based on discrete logarithms.
In Proceedings of CRYPTO 84 on Advances in cryptology, pages 10–18, New York, NY, USA, 1985. Springer-Verlag New York, Inc.

# Références II


Jonathan Katz, Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone.
Handbook of applied cryptography.
CRC press, 1996.


Gene H. Kim and Eugene H. Spafford.
The design and implementation of tripwire : a file system integrity checker.
In CCS '94 : Proceedings of the 2nd ACM Conference on Computer and communications security, pages 18–29, New York, NY, USA, 1994. ACM.


A. Labanoff.
Lettres, instructions et mémoire de Marie Stuart, Reine d'Ecosse.


L'Expansion.com.
Des fraudeurs se font passer pour les services des impots sur internet, 6 Octobre 2009.

# Références III



Plutarque.
Vie de lysandre.
In Les Vies des hommes illustres.


R. L. Rivest, A. Shamir, and L. Adleman.
A method for obtaining digital signatures and public-key cryptosystems.
Commun. ACM, 21(2) :120–126, 1978.


Xiaoyun Wang, Dengguo Feng, Xuejia Lai, and Hongbo Yu.
Collisions for hash functions MD4, MD5, HAVAL–128 and RIPEMD, 2004.
URL : `http://eprint.iacr.org/2004/199/`.