Cryptographie : introduction et outils mathématiques

A. Bonnecaze

Institut de Mathématiques de Marseille (I2M)

Polytech Marseille

Objectifs du cours

- Quelles sont les principales primitives de chiffrement
- Quelles sont les technos utilisées
- Comment évaluer la robustesse d'une primitive
- Comment elle doit être utilisée

Contenu du cours

1. Qu'est ce que la cryptologie ?
2. Le chiffrement symétrique
3. Le chiffrement asymétrique
    1. Rappels d'arithmétique
    2. Introduction aux CE
    3. Introduction aux codes correcteurs d'erreurs

References:
- Handbook of applied crypto (Menezes et al.)
- EC number theory and crypto (Washington)
- Lectures/articles from M. Bellare and P. Rogaway, W. Stalling, D. Boneh, V. Shoup

Qu'est-ce que la cryptologie ?

Définition (approximative)

- La cryptographie est l'art de communiquer confidentiellement au travers d'un canal non sûr.
- La cryptanalyse est l'art de décrypter (d'une manière non légitime) ces communications.
- La cryptologie est l'union de ces deux domaines (cryptos logos : « mot caché » en grec).

Outils : Théo de l'information, probabilité, théo algorithmique des nombres, codes correcteurs d'erreurs.

La cryptologie est une science très ancienne
Réservée aux militaires ou diplomates jusqu'aux années 80
Maintenant accessible à tous
Mais doit être utilisée correctement

Bref historique

Antiquité

Peu de gens savaient lire ...
scytale (Sparta, 475 BC), système de substitution (Caesar)

Jusqu'au 15eme siècle

Al-Kindi : encyclopedie avec une section dédiée à la cryptologie. Première étude de cryptanalyse (9th century)

Renaissance

Polyalphabetic substitution (Vigenere). Le secret à échanger devient non plus la technique mais la clé secrète

Le 19eme siècle

Kasiski : méthode pour attaquer Vigenere
Kerckhoffs : Définition d'un "**bon**" système cryptographique

20eme siècle

Automatisation de la crypto. One-time-pad, Vernam 1917
70's : DES, DH, RSA. 80's : ECC, Id-based,...

Modern Cryptography

In the digital world, security becomes a crucial issue (Internet, electronic commerce, ...)

Cryptography is a set of mathematical techniques, called security mechanisms, the aim being to ensure the following security services

- Data confidentiality
  protection of data from unauthorized disclosure
- Data integrity
  assurance that data received is as sent by an authorized entity
- Authentication
  assurance that the communicating entity is the one claimed
- Non repudiation
  protection against denial by one of the parties in a communication

Kerckhoffs: the security of a cryptosystem must depend only on the key, not on the secrecy of any other part of the system

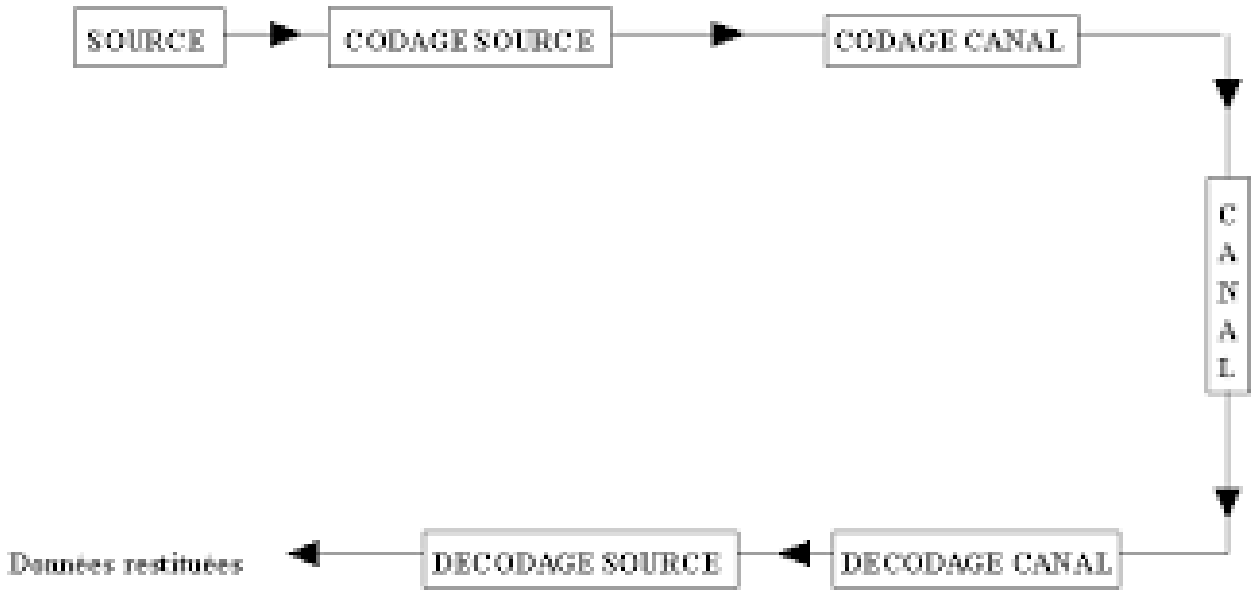
Cryptography is everywhere

- Secure communication:
  web traffic: HTTPS
  wireless traffic: 802.11i WPA2, GSM, Bluetooth
- Encrypting files on disk:
  EFS, TrueCrypt
- Content protection:
  DVD: CSS
  Blu-ray: AACS, BD+
- Internet purchasing:
  SET protocol, ...
- Money:
  bitcoin,...
- User authentication
  Kerberos, ...
- And much more ...

Petit rappel de théorie de l'information

- Développée par Claude Shannon et publié en 1949 au Bell Systems Technical Journal
- Codage, quantité d'information, compression, protection, mais ne s'intéresse pas à la sémantique de l'information
- "Communication Theory of Secrecy Systems" après le célèbre papier de 1948 "**A Mathematical Theory of Communication**"
- Permet de prédire si un chiffrement peut être cassé et avec combien de chiffrés si on utilise une attaque à texte chiffré.

Paradigme de Shannon



- Codage de source : élimine les redondances de la source afin de réduire le débit binaire
- Codage de canal : protection contre les erreurs
- Le canal est bruité (des contraintes matérielles peuvent engendrer des erreurs)

Volume d'information

- La théorie de l'information mesure le volume d'information dans un message par le nombre moyen de bits nécéssaire pour encoder tous les messages possibles
- $H(X) =$ volume d'information de X.
- $H(X)$ est appelé **entropie de X**
- $H(X) = 0$ si X est connu

Exemples

- X est une des quatre possibilités : Nord, Est, Ouest et Sud.
- Un encodage optimal requiert 2 bits : 00, 01, 10, et 11, donc $H(X) = 2$.
- $H(X) = 2$ même si X est encodé comme des mots : "Nord", "Sud", "Est ", "Ouest "
- Ce qui requiert 5*8 = 40 bits
- **38 bits sont gaspillés !**
- Que vaut $H(X)$ si X encode Sex = Male ou Female ?

Formule de l'entropie

$$\begin{aligned} H(X) &= -\sum_X p(X)\log_2(p(X)) \\ &= \sum_X p(X)\log_2(1/p(X)) \end{aligned}$$

Example

4 possibilités equiprobables :
$p(X) = 1/4,\ \ 1/p(X) = 4,\ \ \log_2(1/p(X)) = 2$

$$H(X) = 4 * [(1/4) * 2] = 2$$

Messages équiprobables

- X a n possibilités équiprobables $p(X) = 1/n$

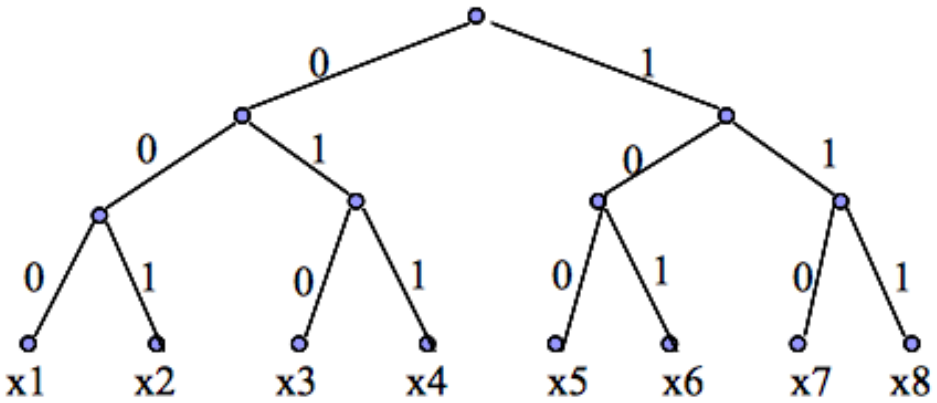
$$\begin{aligned} H(X) &= \sum_X p(X) \log_2(1/p(X)) \\ &= n * (1/n) \log_2(n) \\ &= \log_2(n) \end{aligned}$$

- X encode $\log_2(n)$ bits d'information.

L'encodage optimal est $\log_2(n)$ bits

Messages équiprobables

Example



L'encodage pour $n = 8$ valeurs, équiprobables

$$H(x) = log_2(8) = 3$$

L'entropie représente la hauteur de l'arbre
x_1 est encodé par 000, x_6 par 101, ...

Messages non équiprobables

Example

$n = 3, \quad p(A) = 1/2, \quad p(B) = 1/4, \quad p(C) = 1/4$

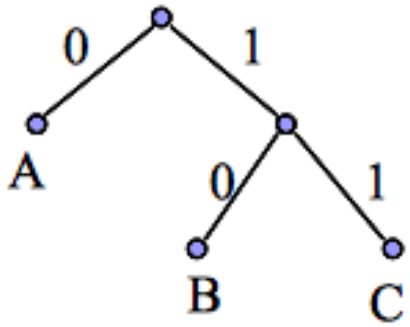
$$\begin{aligned} H(X) &= \sum_X p(X) \log_2(1/p(X)) \\ &= 1/2 \log_2(2) + 2 * (1/4) \log_2(4) \\ &= 1/2 + 1/2 * 2 = 0.5 + 1.0 = 1.5 \end{aligned}$$

Encodage optimal : A='0', B='10', C='11'
Encodage : AB AAC AB C (8 chars)
010 0011 01011 (12 bits)

En moyenne #bits/char est 1.5

Messages non équiprobables

Example

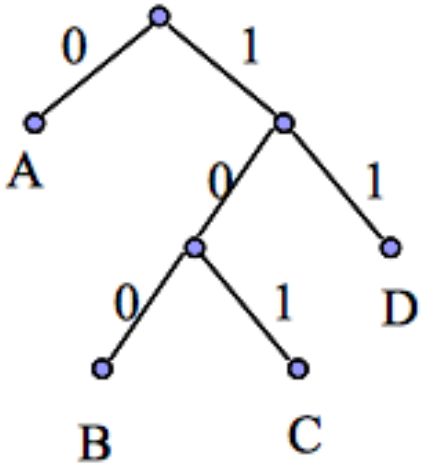


p = 1/2

p = 1/4

AB AAC AB C

010001101011



p = 1/2

p = 1/4

p = 1/8

n = 4

AB AD AC AD

01000110101011

Message connu

Example

Supposons que le message soit encodé sur 4 bits, mais que le message soit connu

$$\begin{aligned} H(X) &= \sum_X p(X) \log_2(1/p(X)) \\ &= 1 * \log_2(1) \\ &= 0 \end{aligned}$$

H(X) mesure l'incertitude (le nombre de bits inconnus)

Remarque : Le codage de Huffman permet d'obtenir un codage optimal

Entropie d'une source discrète

Example

26 lettres de l'alphabet (+ l'espace).

Si les 27 symboles sont équiprobables et indépendants :

- Entropie par lettre : $H_0 = \log_2(27) = 4,755$
- Exemple de tirage avec remise :
  XFOML RHKHJFFJUJZLPWCFWCKCYJFFJEYVKCQSGHYD
  QPAAMKBZAACIBZLHJQD
- Seules les lettres nous sont familières !

Entropie d'une source discrète

Example

On considère maintenant les probas d'apparition des lettres (anglais)

Entropie par lettre : $H_1 = -\sum P_i \log_2(p_i) = 4,029$

Le tirage donne :
OCRO HLI RGWR NMIEL VIS EU LL NBNESEBYA TH EEI ALHENHTTPA
OOBTTVA NAH BRL

Ça ressemble plus à du texte !

Remarque : en anglais, 't' est souvent suivi de 'h'

Entropie d'une source discrète

Example

On considère maintenant les probas des couples de lettres :

Entropie du couple est $H_2 = 3,318$

Le tirage donne :
ON IE ANTSOUTINYS ARE T INCORE ST BE S DEAMY ACHIN D ILONASIVE TUCOOWE AT TEASONARE FUSO TIZIN ANDY TOBE SEACE CTISBE

Remarques : proportion entre voyelles et consonnes, alternance entre voyelles et consonnes
On peut (presque) prononcer les mots

Entropie d'une source discrète

Example

On considère les probas des triplets...

$H_3 = 3,1$

Le tirage donne :
IS NO IST LAT WHEY CRATICT FROURE BIRS GROCID PONDENOME OF DEMONSTRURES OF THE REPTAGIN IS REGOACTIONA OF CRE

Remarques : des mots complets commencent à apparaître
Si on prend en compte 8 lettres consécutives, on obtient $H_8 = 1,86$.

L'information apportée par une lettre est de l'ordre de $1,86$ bit (en anglais)

Entropie d'une source discrète

Example

Entropie relative

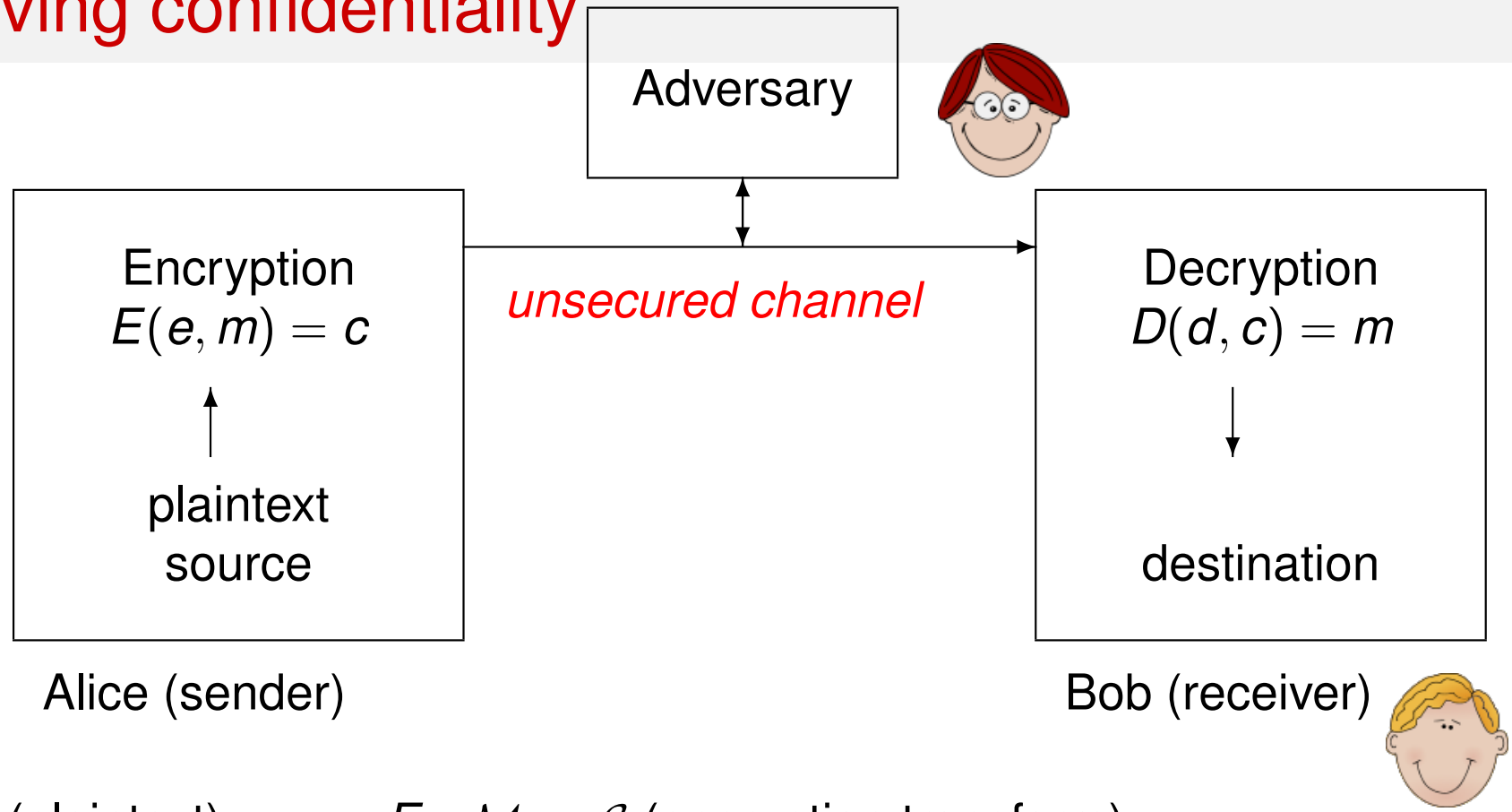
$H_8/H_0 = 1,86/4,75 = 40\%$

Dans le choix des lettres pour écrire un message en anglais, on est 40% aussi libre qu'on le serait si on se contentait de choisir de façon équiprobable et indépendante les lettres les unes après les autres.

La redondance est de 60%.

Dans un texte anglais
40% des lettres sont choisies librement
60% sont imposées par les règles de structure du langage
C'est sensiblement la même chose en français

Achieving confidentiality



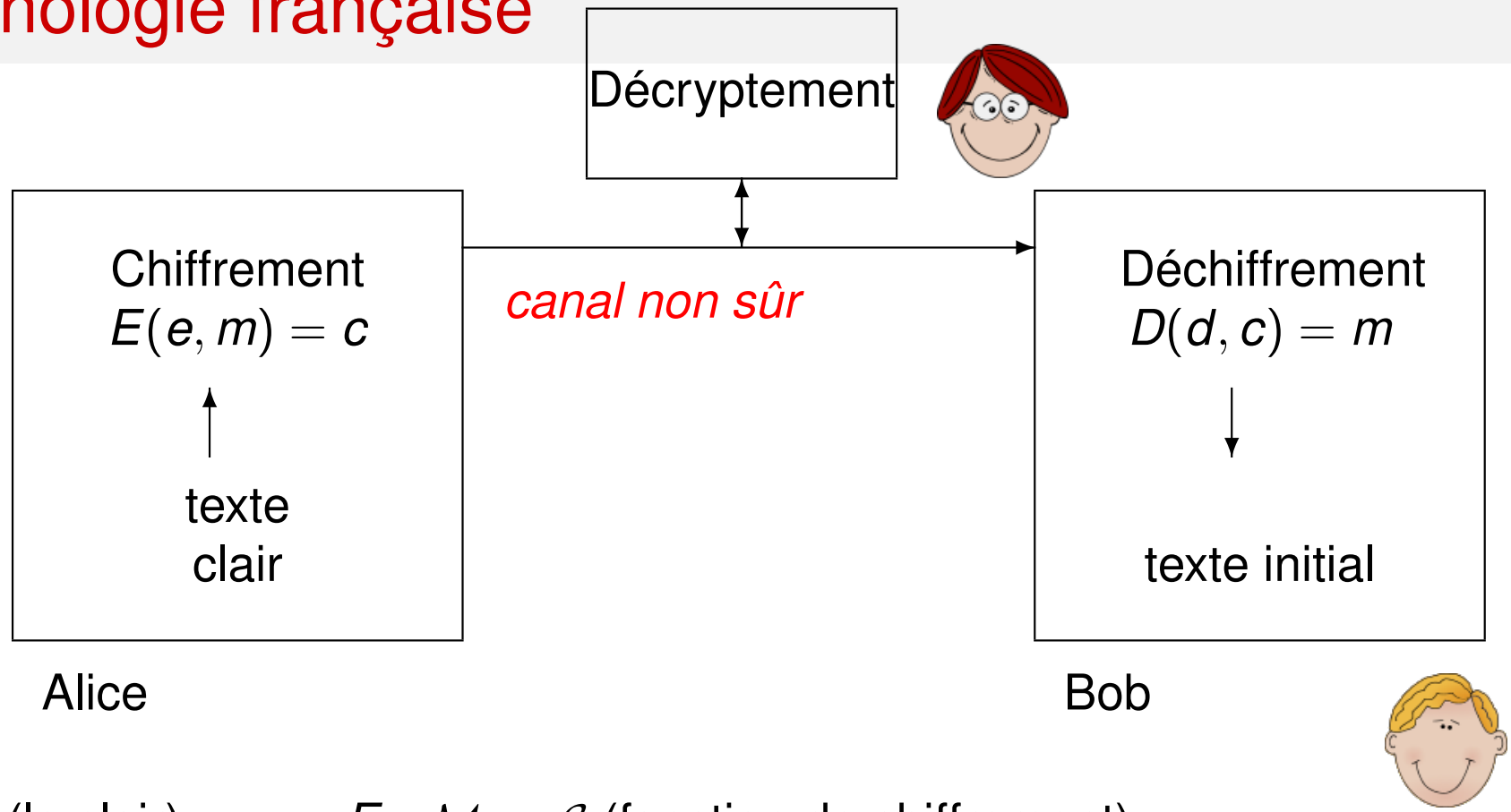
$m \in \mathcal{M}$ (plaintext) $\quad E : \mathcal{M} \to \mathcal{C}$ (encryption transform)

$c \in \mathcal{C}$ (ciphertext) $\quad D : \mathcal{C} \to \mathcal{M}$ (decryption transform)

$e, d \in \mathcal{K}$ (keys, d is secret)

$$D_d(E_c(m)) = m$$

Terminologie française



$m \in \mathcal{M}$ (le clair) $\quad E : \mathcal{M} \to \mathcal{C}$ (fonction de chiffrement)

$c \in \mathcal{C}$ (le chiffré) $\quad D : \mathcal{C} \to \mathcal{M}$ (fonction de déchiffrement)

$e, d \in \mathcal{K}$ (clés, d est secrète)

$$D_d(E_c(m)) = m$$

Entropie et cryptographie

$H(m) =$ incertitude sur le texte

$H(k) =$ incertitude sur la clef

$H(k) = n$ pour clef de n bits équiprobables

$H_c(k) =$ incertitude sur la clef, le chiffré étant donné

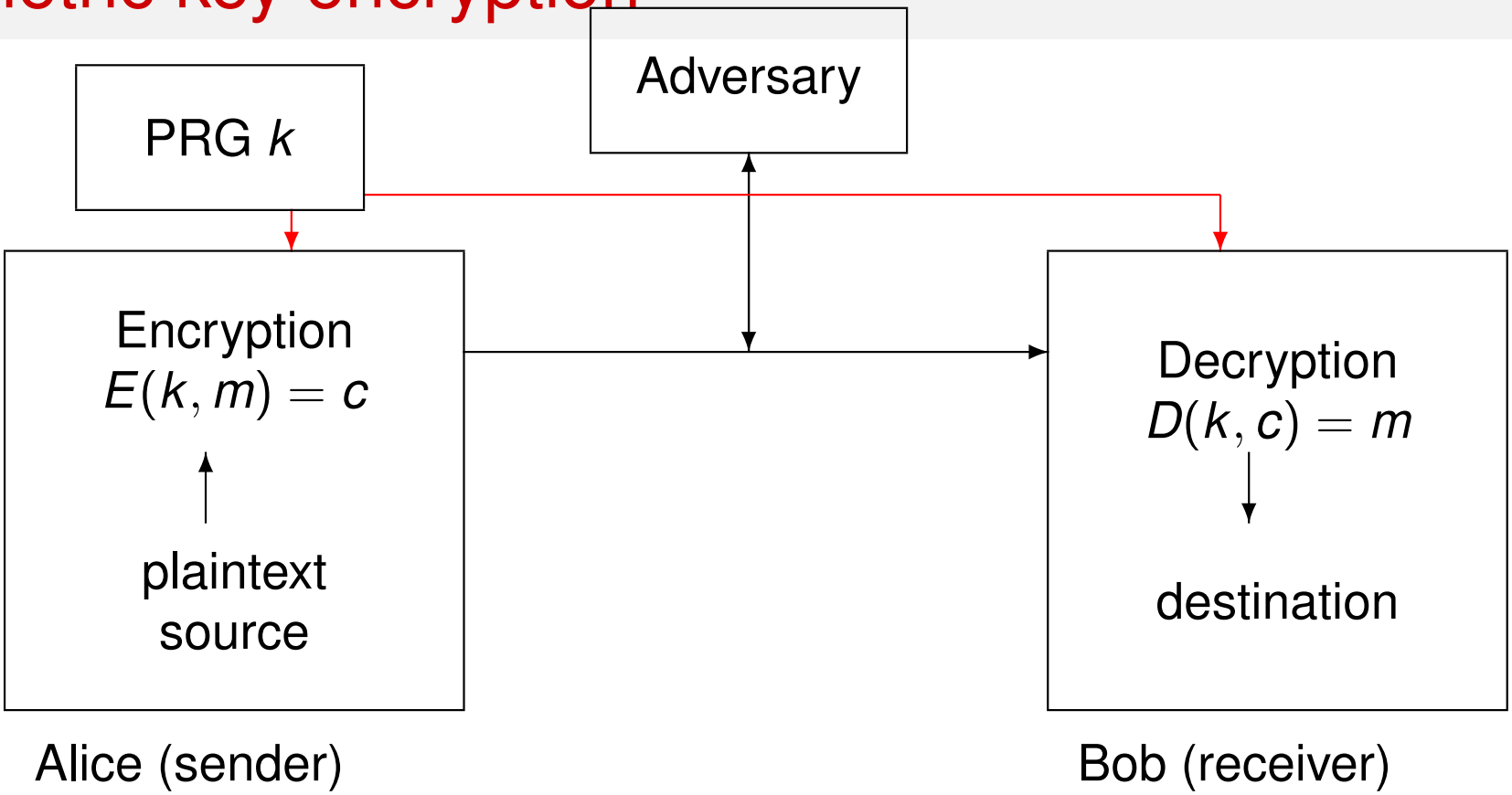
Perfect secrecy : $p_c(m) = p(m)$

c ne révèle pas d'information sur m

Confidentiality : hiding a text (many different ways)

Cher ami,
Je suis toute émue de vous dire que j'ai
bien compris l'autre jour que vous aviez
toujours une envie folle de me faire
danser. Je garde le souvenir de votre
baiser et je voudrais bien que ce soit
une preuve que je puisse être aimée
par vous. Je suis prête à montrer mon
affection toute désintéressée et sans cal-
cul, et si vous voulez me voir ainsi
vous dévoiler, sans artifice, mon âme
toute nue, daignez me faire visite,
nous causerons et en amis franchement
je vous prouverai que je suis la femme
sincère, capable de vous offrir l'affection
la plus profonde, comme la plus étroite
amitié, en un mot : la meilleure épouse
dont vous puissiez rêver. Puisque votre
âme est libre, pensez que l'abandon ou je
vis est bien long, bien dur et souvent bien
insupportable. Mon chagrin est trop
gros. Accourrez bien vite et venez me le
faire oublier. À vous je veux me sou-
mettre entièrement.
Votre poupée

Symmetric key encryption



Everything is public except the key k
Note that in symmetric crypto, encryption key = decryption key
Problems: How to generate k? How to distribute k?

Type of encryption (symmetric)

- Stream ciphers: One Time Pad (OTP), RC4, Salsa,...
  - bit-wise or byte-wise
  - very simple operation; generally XOR
  - very fast but length of key is length of message
- Block ciphers: (DES, 3-DES, AES, ...)
  - the message is divided into blocks of fixed length
    for ex. 64 bits (DES), 128 bits (AES)
  - transformation is applied on each block
  - fast (AES)
  - key length in bits: 56 for DES, 112 for 3-DES, 128, 192 or 256 for AES
  - DES not secure any more, standard is AES

Performance

Stream ciphers VS **block ciphers**

AMD Opteron, 2.2GHz (LINUX) Crypto++ 5.6.0 [Wei Dai]

Cipher	Block/key size	Speed (MB/sec)
RC4		126
Salsa20		643
Sosemanuk		727
3DES	64/168	13
AES	128/128	109

Terminologie

Domaines de chiffrement

- $\mathcal{A}$ est un ensemble fini appelé **alphabet de définition**. Par exemple $\mathcal{A} = \{0, 1\}$
- $\mathcal{M}$ est un ensemble appelé **espace des messages** . $\mathcal{M}$ consiste en des séquences de symboles d'un alphabet de définition. Un élément de $\mathcal{M}$ est appelé le clair (**plaintext** en anglais). Par exemple, binary strings, English text, computer code, etc.
- $\mathcal{C}$ est un ensemble appelé **espace des chiffrés**. $\mathcal{C}$ consiste en séquences de symboles d'un alphabet de définition, possiblement différent de celui de $\mathcal{M}$. Un élément de $\mathcal{C}$ est un chiffré (**ciphertext** en anglais).

Chiffrement symétrique

Définition

Un système de chiffrement symétrique est une paire d'algorithmes (E, D) efficaces définis sur $(\mathcal{K}, \mathcal{M}, \mathcal{C})$ où $E : \mathcal{K} \times \mathcal{M} \to \mathcal{C}$ $\quad \mathcal{D} : \mathcal{K} \times \mathcal{C} \to \mathcal{M}$ tq $\forall m \in \mathcal{M},\ k \in \mathcal{K}$,

$$D(k, E(k, m)) = m$$

E est (souvent) randomisée $\qquad$ D est déterministe

Par la suite, on va s'intéresser aux chiffrements anciens (Cesar, Vigenere, Vernam), puis aux chiffrements modernes

Chiffrements par substitutions et transpositions

Chiffrement par substitution

Cryptanalyse

1. non robuste si l'espace des substitutions est petit
2. Attention, si on chiffre du français, certaines lettres reviennent plus souvent que d'autres dans le texte :

lettre	probabilité	lettre	probabilité
A	0.082	D	0.043
B	0.015	E	0.127
C	0.028	...	...

Exemple : Chiffrement de Cesar : un message est composé de lettres

$$c = m + k, \quad \text{avec } k \in \{1, \ldots, 26\},$$

L'espace des clés est trop petit !

Polyalphabetic substitution ciphers

Definition

Un chiffrement par substitutions polyalphabetiques est un chiffrement par bloc (bloc de longueur t) sur un alphabet $\mathcal{A}$ ayant les propriétés suivantes :

1. $\mathcal{K}$ consiste en les ensembles ordonnés de t permutations $(p_1, p_2, \ldots, p_t)$, où chaque permutation p_i est défini sur $\mathcal{A}$;
2. le chiffrement du message $(m_1 m_2 \cdots m_t) \in \mathcal{M}$ avec la clé $e = (p_1, p_2, \ldots, p_t)$ est donné par $E_e(m) = (p_1(m_1), p_2(m_2), \ldots, p_t(m_t))$;
3. la clé de déchiffrement associée à e est $d = (p_1^{-1}, p_2^{-1}, \ldots, p_t^{-1})$.

Example

Vigenere cipher: $\mathcal{A} = \{A, B, C, \ldots, X, Y, Z\}$, $t = 3$, $e = (3, 7, 10)$. si

$$m = THI\ \ SCI\ \ PHE\ \ RIS\ \ CER\ \ TAI\ \ NLY\ \ NOT\ \ SEC\ \ URE, \text{ alors}$$

$$c = E_e(m) = WOS\ \ VJS\ \ SOO\ \ UPC\ \ FLB\ \ WHS\ \ QSI\ \ QVD\ \ VLM\ \ XYO$$

Polyalphabetic substitution ciphers

- Si $t = 1$, c'est un chiffrement de César : ce chiffrement n'est pas sûr car
  1. l'espace des clés est trop petit
  2. frequency attack (si le langage est connu)

- Si $t > 1$, et t petit comparé à la longueur du message. Pourquoi ce chiffrement n'est-il pas sûr ?
  1. Trouver la longueur de la clé
  2. frequency attack

Chiffrement par permutations

Definition

Un shéma de chiffrement par bloc (de longueur t) symétrique.
K l'ensemble de toutes les permutations sur $\{1, 2, \ldots, t\}$.
Pour chaque $e \in \mathcal{K}$, on définit la fonction de chiffrement

$$E_e(m) = (m_{e(1)} m_{e(2)} \cdots m_{e(t)}),$$

où $m = (m_1 m_2 \cdots m_t) \in \mathcal{M}$.
La clé de déchiffrement est $d = e^{-1}$.
Pour déchiffrer $c = (c_1 c_2 \cdots c_t)$, on calcule $D_d(c) = (c_{d(1)} c_{d(2)} \cdots c_{d(t)})$.

Ce chiffrement peut-il être facilement cryptanalysé ?

Compositions de chiffrements

Les chiffrements par substitutions ne sont pas robustes

Les chiffrements par permutations ne sont pas robustes

Mais la composition de chiffrements de substitutions et de permutations est plus difficile à casser

Par exemple, le DES (présenté plus tard) consiste en 16 tours, chacune composée de permutations et substitutions.

La **confusion** gomme les relations entre le texte en clair et le texte chiffré.
Exemple : substitutions (table-S dans DES)
La **diffusion** disperse la redondance du clair en la répartissant dans le chiffré
Exemple : permutations (initiale, p, finale dans DES)

One Time Pad et sa sécurité

One Time Pad (Vernam 1917)

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

$$c = E(k,m) = k \oplus m \qquad\qquad D(k,c) = k \oplus c$$

Très rapide mais pas pratique car la clé doit être une séquence binaire **aléatoire** et **aussi longue** que le message

Comment prouver sa sécurité ?

Idée : le chiffré ne doit révéler aucune information sur le texte clair

C'est la notion de sécurité parfaite...

Sécurité parfaite (Perfect secrecy)

Définition

(E, D) sur $(\mathcal{K}, \mathcal{M}, \mathcal{C})$ a une sécurité parfaite si
$\forall m_0, m_1 \in \mathcal{M}, \quad |m_0| = |m_1|, \quad \forall c \in \mathcal{C}$

$$Pr_k[E(k, m_0) = c] = Pr_k[E(k, m_1) = c]$$

où $k \xleftarrow{R} \mathcal{K}$ (k est choisi aléatoirement dans $\mathcal{K}$)

- c donné, impossible de savoir s'il est le chiffré de m_0 ou de m_1
- L'adversaire n'apprend rien du chiffré
- Donc une attaque utilisant seulement le chiffré n'est pas possible

Exercice : One Time Pad

Soit m, c de longueur n fixés, combien de clés permettent-elles de chiffrer m en c par OTP?

- dépend de m
- une infinité
- aucune
- 1
- 2

Que vaut $\#\mathcal{K}$?

$\forall m \in \mathcal{M}, \quad \forall c \in \mathcal{C}$
Que vaut : $\#\{cles \;\; k \;\; tq \;\; E(k,m) = c\}$?

OTP a une sécurité parfaite

Preuve

$\forall m \in \mathcal{M}, \quad \forall c \in \mathcal{C}$

$$Pr_k[E(k,m) = c] = \frac{\#\{cles \;\; k \;\; tq \;\; E(k,m) = c\}}{|\mathcal{K}|} = 1/2^n$$

Avec OTP, une attaque par chiffrés n'est pas possible
Mais d'autres attaques sont possibles ...
Peut-on garder une sécurité parfaite si on réduit la taille des clés?

Théorème

Sécu parfaite $\Rightarrow \quad |\mathcal{K}| = |\mathcal{M}|$

OTP non pratique !!

OTP non pratique

Le chiffrement par flot (stream cipher) utilise un XOR comme OTP

Mais les clés sont construites à partir de générateurs pseudo-aléatoires

Les clés ne sont pas aléatoires mais "ressemblent" à des clés aléatoires

Alice et Bob ont le même générateur qui à partir d'un **germe aléatoire** de petite longueur (par ex : 128 bits) construit une clé (presque) aussi longue qu'on veut.

Qu'est-ce qu'un PRG sûr ?
Comment s'assurer qu'un PRG est sûr ?
Quels sont les PRG les plus utilisés ?
Quelles sont les attaques connues ?

Pseudo Random Generators et leur sécurité

Pseudo Random Generator

Comment créer des clés aléatoires?
Existe-t-il un moyen de construire une clé aussi longue qu'on veut à partir d'une clé secrète (germe) ?

Definition

Un **PRG** est un algorithme déterministe G qui à partir d'une séquence aléatoire k de longueur m rend une séquence de longueur $l >> m$, qui "semble" aléatoire. k est appelé le **germe**
$G(k)$ est une **séquence pseudo-aléatoire**

Enigma (années 20) à partir d'une clé courte, génère une clé aussi longue que l'on veut qui parait avoir été tirée aléatoirement
Cassée par les alliés durant la deuxième guerre mondiale

Stream cipher : rendre pratique OTP

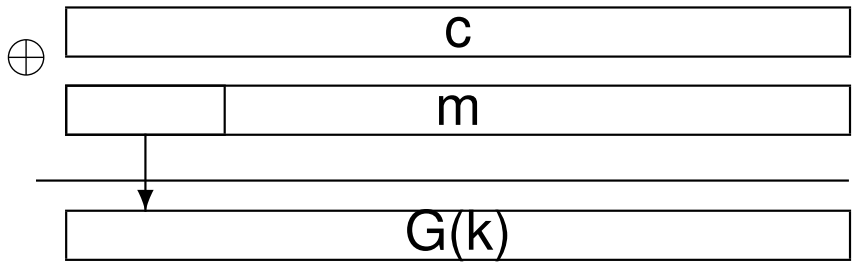
Les Stream Ciphers ne peuvent pas avoir de sécurité parfaite

- Besoin d'une autre définition de sécurité
- La sécurité dépendra du PRG
- Le PRG doit être imprédictible

Supposons PRG prédictible $\quad k =$seed, $G =$PRG

$$\exists i \quad G(k)|_{1,\ldots,i} \xrightarrow{Algo} G(k)|_{i+1,\ldots,n}$$

Alors



Le fait de pouvoir prédire le prochain bit est un problème !

PRG prédictible ou imprédictible

Définition : PRG prédictible

$G : \mathcal{K} \to \{0,1\}^n$ est prédictible si $\exists \mathcal{A}$ Algo efficace et $\exists 1 \leq i \leq n-1$ tq

$$Pr_{k \xleftarrow{R} \mathcal{K}}[\mathcal{A}(G(k)|_{1,\ldots,i}) = G(k)|_{i+1}] \geq 1/2 + \epsilon$$

pour ϵ "non négligeable" c.a.d. $\epsilon \geq 1/2^{30}$

Définition : un PRG est imprédictible s'il n'est pas prédictible

$\Rightarrow \forall i$, aucun adversaire ne peut prédire le prochain bit pour un ϵ non négligeable

Exercice : $G : \mathcal{K} \to \{0,1\}^n$ tq $\forall k \in \mathcal{K}, XOR(G(k)) = 1$. G est-il prédictible?

1. Oui, le 1er bit donné, je peux prédire le 2eme
2. G est imprédictible
3. Ca dépend de n
4. Oui car si je connais les $n-1$ premiers bits, je peux prédire le nième

Que veut dire négligeable?

En pratique ϵ est un scalaire

- ϵ non négligeable : $\epsilon \geq 1/2^{30}$ (l'événement peut survenir sur 1GB de données)
- ϵ négligeable : $\epsilon \leq 1/2^{88}$ (l'événement n'arrivera jamais dans toute la vie de la clé)

En théorie ϵ est une fonction

$$\epsilon \ : \ \mathbb{Z}^{\geq 0} \rightarrow \mathbb{R}^{\geq 0}$$

- ϵ non négligeable : $\exists d \ : \ \epsilon(\lambda) \geq 1/\lambda^d$ ($\epsilon \geq 1/Poly$, pour beaucoup de λ)
- ϵ négligeable : $\forall d, \ \exists \lambda_d, \ \lambda \geq \lambda_d \ : \ \epsilon(\lambda) \leq 1/\lambda^d$ ($\epsilon \leq 1/Poly$, pour λ grand)

Exercice : négligeable ou non négligeable?

- $\epsilon(\lambda) = 1/2^{\lambda}$
- $\epsilon(\lambda) = 1/\lambda^{1000}$
- $\epsilon(\lambda) = \begin{cases} 1/2^{\lambda} \text{ si } \lambda \text{ impair} \\ 1/\lambda^{1000} \text{ si } \lambda \text{ pair} \end{cases}$
- $\epsilon(\lambda) = 1/2^{\lambda} + 1/\lambda^{1000}$
- $\epsilon(\lambda) = 1/1000^{\lambda}$

PRG sûr

Un bon PRG doit se comporter (presque) comme un générateur aléatoire (RG)
Qu'est-ce que cela signifie?

Soit $G : \mathcal{K} \to \{0,1\}^n$ un PRG,

$$[k \xleftarrow{R} \mathcal{K}, \quad output\ G(k)]$$

doit être "indistinguable" de

$$[r \xleftarrow{R} \{0,1\}^n, \quad output\ r]$$

Remarque : L'espace des germes, input de $G()$, est beaucoup plus petit que $\{0,1\}^n$

Tests statistiques (voir NIST)

Comment décider qu'une séquence pseudo aléaoire est distinguable d'une séquence aléatoire ?

Un test sur $\{0,1\}^n$ est un algo (distingueur)

$$\{0,1\}^n \to \begin{cases} 0 \text{ l'output n'est pas aléatoire} \\ 1 \text{ le test est passé avec succès} \end{cases}$$

Exemples de tests :

- nombre de 1 dans la séquence
- nombre de runs
- longueur du plus grand run de 1
- ...

Avantage

Soit $G : \mathcal{K} \to \{0,1\}^n$ un PRG, A un test stat. sur $\{0,1\}^n$

Définition

$$Adv_{PRG}[A,G] := |Pr_{k \xleftarrow{R} \mathcal{K}}[A(G(k)) = 1] - Pr_{r \xleftarrow{R} \{0,1\}^n}[A(r) = 1]|$$

- L'avantage donne une valeur entre 0 et 1
- Adv proche de 1 $\Rightarrow$ A peut distinguer G d'un RG
- Adv proche de 0 $\Rightarrow$ A ne peut pas distinguer G d'un RG

Exemple

Soit $G : \mathcal{K} \to \{0,1\}^n$ un PRG, A un test stat. sur $\{0,1\}^n$
G satisfait $msb(G(k)) = 1$ pour $2/3$ des clés de $\mathcal{K}$

Définissons le test stat A par :
$A(x) = 1$ si $msb(x) = 1$
$A(x) = 0$ si $msb(x) = 0$

Quel est l'avantage de A?

$$Adv_{PRG}[A, G] := |Pr_{k \xleftarrow{R} \mathcal{K}}[A(G(k)) = 1] - Pr_{r \xleftarrow{R} \{0,1\}^n}[A(r) = 1]| = ?$$

Exemple

Soit $G : \mathcal{K} \to \{0,1\}^n$ un PRG, A un test stat. sur $\{0,1\}^n$
G satisfait $msb(G(k)) = 1$ pour $2/3$ des clés de $\mathcal{K}$

Définissons le test stat A par :
$A(x) = 1$ si $msb(x) = 1$
$A(x) = 0$ si $msb(x) = 0$

Quel est l'avantage de A?

$$Adv_{PRG}[A, G] := |Pr_{k \xleftarrow{R} \mathcal{K}}[A(G(k)) = 1] - Pr_{r \xleftarrow{R} \{0,1\}^n}[A(r) = 1]| = 1/6$$

L'avantage n'est pas négligeable donc A casse G avec avantage $1/6$

PRG sûr

Définition : PRG sûr

$G : \mathcal{K} \to \{0,1\}^n$ est sûr si pour tout test stat A, $Adv_{PRG}[A, G]$ est négligeable.

Existe-t-il des PRG dont la sécurité est prouvable? on ne sait pas (P=?NP)

Un PRG sûr est imprédictible

Preuve : par contraposé

On montre que PRG prédictible $\Rightarrow$ PRG non sûr

Un PRG sûr est imprédictible : preuve

Soit A un algo efficient tq

$$Pr_{k \xleftarrow{R} \mathcal{K}}[A(G(k)|_{1,\ldots,i}) = G(k)|_{i+1}] \geq 1/2 + \epsilon$$

pour ϵ "non négligeable" (par exemple $\epsilon = 1/1000$)
Définissons un test stat B :

$$B(x) = \begin{cases} \text{if } A(x)|_{1,\ldots,i} = x_{i+1} \text{ output 1} \\ \text{else output 0} \end{cases}$$

$$r \xleftarrow{R} \{0,1\}^n : \qquad Pr[B(r) = 1] = 1/2$$

$$k \xleftarrow{R} \mathcal{K} : \qquad Pr[B(G(k) = 1)] = 1/2 + \epsilon$$

$$\Rightarrow Adv_{PRG}[B, G] = \epsilon$$

avec ϵ non négligeable

Yao'82 : Un PRG imprédictible est sûr

Théorème

Soit $G : \mathcal{K} \to \{0,1\}^n$ un PRG. Si $\forall i \in \{0,\ldots,n-1\}$, G est imprédictible à la position i, alors G est sûr.

Cela signifie que si les prédicateurs du prochain bit ne peuvent pas distinguer G d'un RG, alors aucun test statistique ne peut le faire.

Exercices

1) Soit $G : \mathcal{K} \to \{0,1\}^n$ un PRG tq à partir des derniers $n/2$ bits de $G(k)$, il est facile de calculer les $n/2$ premiers bits

G est-il prédictible pour certains $i \in \{0, \ldots, n-1\}$?

2) Soit $G : \mathcal{K} \to \{0,1\}^n$ un PRG tq
$G(k)$ admet (à 1 près) autant de 0 que de 1.
Soit le test statistique A qui rend 1 si le nombre de 0 est égal (à 1 près) au nombre de 1 et qui rend 0 sinon.

Quel est l'avantage de A? G passe-t-il le test ? G est-il sûr ?

Indistinguabilité (calculatoire)

Soit P_1 et P_2 deux distributions sur $\{0,1\}^n$

Définition

On dit que P_1 et P_2 sont calculatoirement indistinguable ($P_1 \approx_p P_2$) si pour tout test statistique A

$$|Pr_{x \xleftarrow{R} P_1}[A(x) = 1] - Pr_{x \xleftarrow{R} P_2}[A(x) = 1]| < negl.$$

Ex : un PRG est sûr si $\{k \xleftarrow{R} \mathcal{K} \ : \ G(k)\} \approx_p uniform(\{0,1\}^n)$

Les PRG les plus utilisés

Le PRG construit une séquence binaire qui sert de clé dans un stream cipher

le chiffrement utilise toujours un XOR :

$$\text{message} \oplus k = \text{chiffré}$$

Linear Feedback Shift Register pour stream ciphers (LFSR)

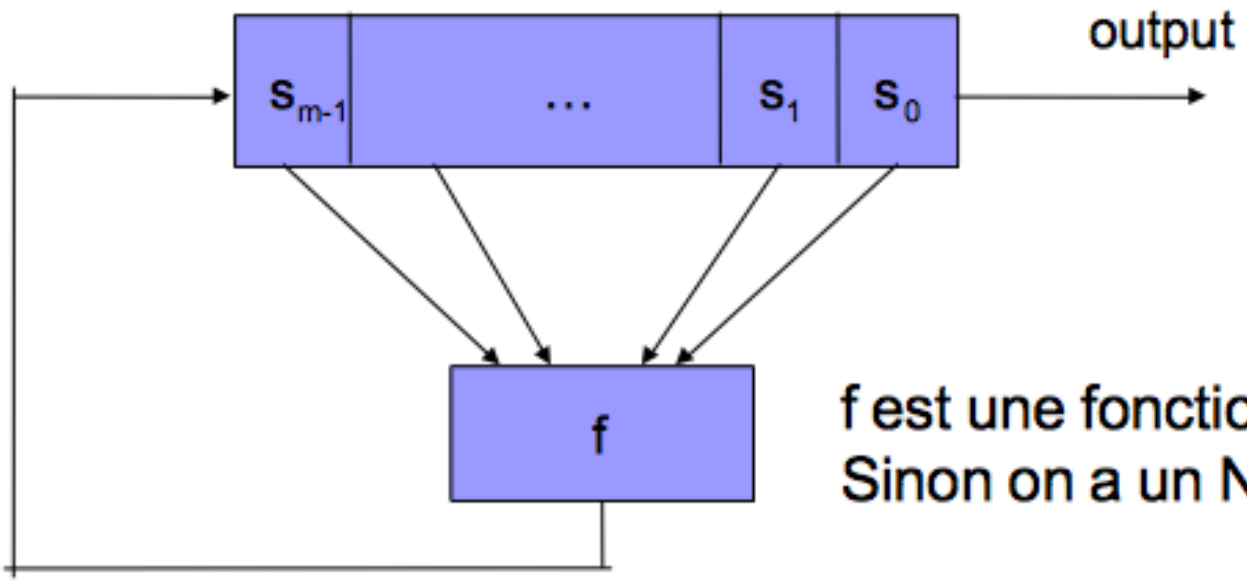
- Ils sont un composant de base de beaucoup de systèmes de chiffrement par flot
- Ils s'implémentent facilement en hardware
- Ils produisent des séquences qui ont de bonnes propriétés statistiques qui vont être utilisées comme des clés dans OTP
- Et pourtant, ils ne sont pas sûrs !

Un LFSR de longueur m sur $\mathbb{F}_2$ est un automate à état fini qui produit une séquence d'éléments de $\mathbb{F}_2$, $s = (s_t)_{t\geq 0} = s_0 s_1 \ldots$, satisfaisant une relation de recurrence linéaire de degré m sur $\mathbb{F}_2$

$$s_{t+m} = \sum_{i=1}^{m} c_i s_{t+m-i}, \quad \forall t \geq 0.$$

Les coefficients binaires c_i sont appelés **feedback coefficients** du LFSR.

LFSR



f est une fonction linéaire booléenne
Sinon on a un NLFSR

Le registre a m états contenant chacun un élément binaire. Initialement, les états sont $s_0 s_1 s_2 \ldots, s_{m-1}$. Le registre est controlé par une horloge externe. A chaque unité de temps, les s_i sont shiftés vers la droite. Le contenu de l'état le plus à droite s_t sort du registre et s_{t+m} rentre dans le registre par la gauche. s_{t+m} est le **feedback bit**. Il est obtenu par une combinaison linéaire des s_i

$$s_{t+m} = \sum_{i=1}^{m} c_i s_{t+m-i}.$$

LFSR

Exemple

LFSR de longueur 4 avec feedback coefficients $c_1 = c_2 = 0,\ c_3 = c_4 = 1$ et état initial $(s_0, s_1, s_2, s_3) = (1, 0, 1, 1)$. Quelle est la relation de récurrence ? Quel est la séquence produite ?

Les coefficients c_i d'un LFSR peuvent être représentés par un **feedback polynomial** (ou **connection polynomial**) défini par

$$P(x) = 1 - \sum_{i=1}^{m} c_i x^i.$$

On utilise aussi le **polynôme caractéristique** qui est le polynôme réciproque de P :

$$Q(x) = x^m P(1/x) \ = \ x^m - \sum_{i=1}^{m} c_i x^{m-i}.$$

LFSR pour chiffrer ?

- La **complexité linéaire** Λ est la longueur minimale du registre permettant d'obtenir une séquence donnée. Dans l'exemple précédent, c'est 4.
- Les LFSR ne devraient jamais être utilisés pour construire la clé d'un stream cipher :
  - Si on connait l'état du registre initial, alors on connait toute la clé
  - Si on connait une séquence de 2Λ éléments consécutifs, alors on connait toute la clé (Algorithme de Berlekamp-Massey, 1968)
- On peut combiner plusieurs LFSR pour essayer de construire une séquence ayant une complexité linéaire grande

Stream ciphers les plus utilisés

Stream ciphers utilisant des LFSR

- Encryption of DVD (CSS) utilise 2 LFSRs
- GSM encryption (A4/1,2) utilise 3 LFSRs
- Bluetooth (EO) uses 4 LFSRs

RC4 (1987) pour HTTPS et WEP

Faiblesses :

- biais sur l'output initial $Pr[2^{nd} byte = 0] = 2/256$ (au lieu de $1/256$)
- Prob. de $(0, 0)$ est $1/256^2 + 1/256^3$
- si les clés sont proches les unes des autres

tous cassés !

Les attaques

- 2-time pad
- MS-PPTP (Windows NT)
- 802.11b WEP
- Content Scramble System (CSS) chiffrement de DVD-video

2 time pad à éviter

Même si la clé est aléatoire, il ne faut jamais s'en resservir :

Exemple

$$c_1 = m_1 + PRG(k)$$
$$c_2 = m_2 + PRG(k)$$

L'attaquant calcule $c_1 + c_2 = m_1 + m_2$

Si les messages sont en français, il est facile de retrouver m_1 et m_2 grâce à la redondance dans le langage !

2 time pad à éviter

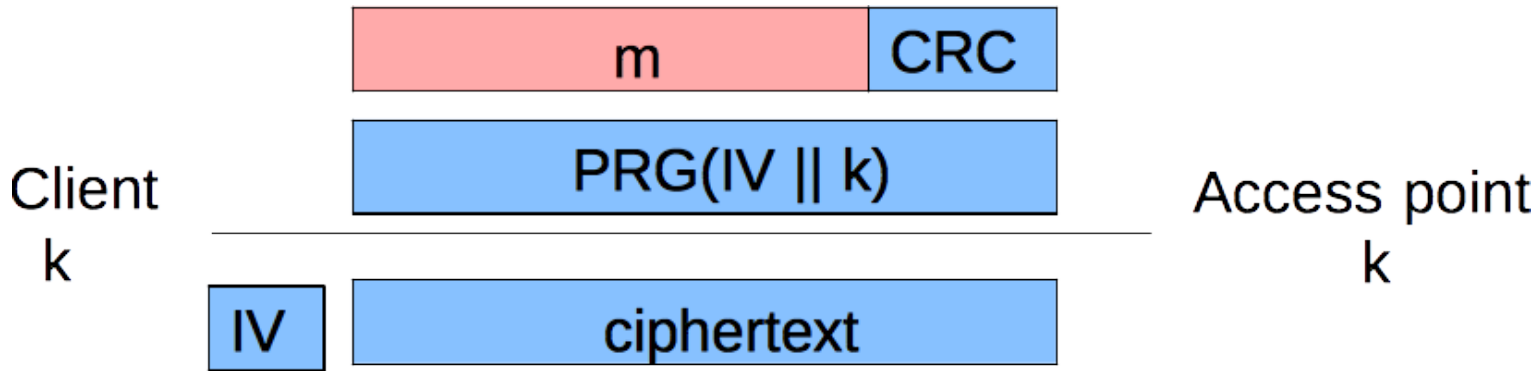
Même si la clé est aléatoire, il ne faut jamais s'en resservir :

Exemple : MS-PPTP (Windows NT)

Client		Server
k		k
$m_1 \rightarrow$		$\leftarrow s_1$
$m_2 \rightarrow$		$\leftarrow s_2$
$m_3 \rightarrow$		$\leftarrow s_3$
$(m_1 \mid\mid m_2 \mid\mid m_3 \mid\mid \ldots) + PRG(k)$		$(s_1 \mid\mid s_2 \mid\mid s_3 \mid\mid \ldots) + PRG(k)$

Il faudrait une clé pour le client et une autre pour le serveur.
Par ex : $K := (k_1, k_2)$, le client chiffre avec k_1 et le serveur chiffre avec k_2.
K est connu des 2 parties

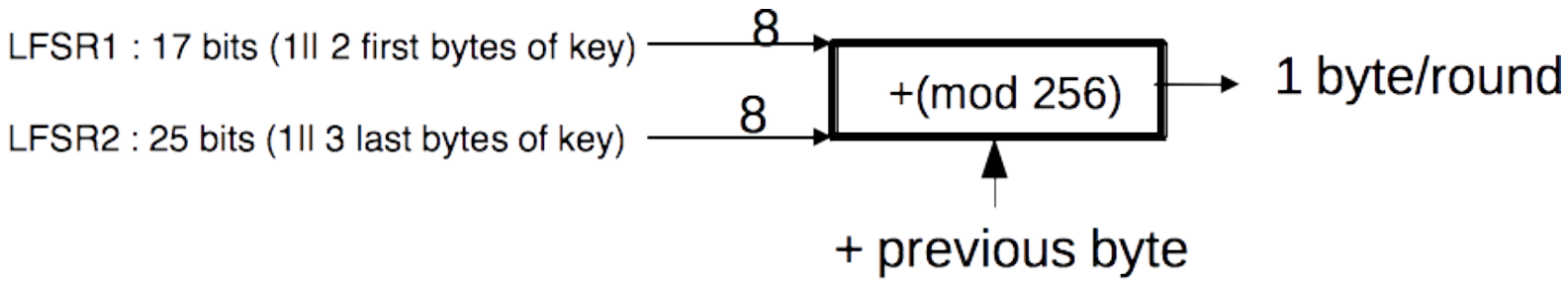
WEP 802.11b (PRG is RC4)



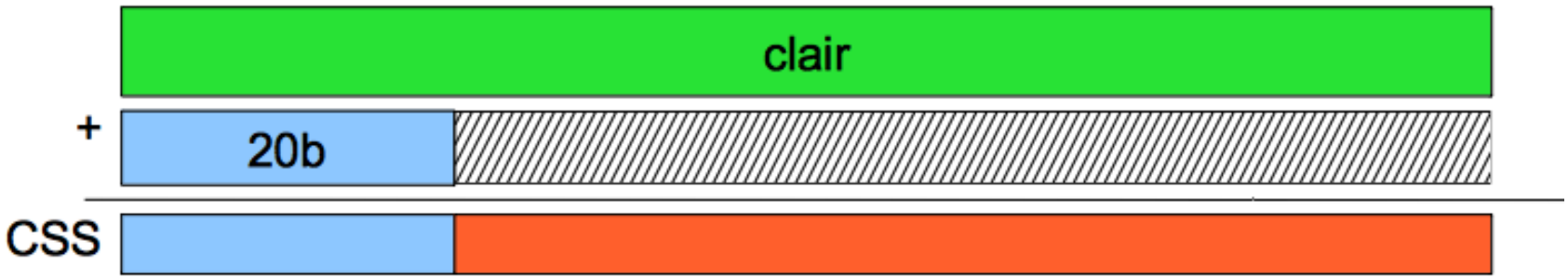
- longueur du vecteur d'initialisation IV = 24 bits ($2^{24} = 16M$ frames)
- sur certaines cartes, reset de l'IV aprés chaque utilisation
- IV est un compteur : $0, 1, 2, 3, \ldots$ et $|k| = 104$ bits. Donc les clés sont très proches les unes des autres
  2001 : attaque de RC4. Avec 10^6 paquets on peut pour retrouver la clé.
  En fait 40 000 paquets suffisent (quelques minutes)

Attaque sur CSS

Key = 5 bytes (US gov. Export regulation)



Msb=1 pour prévenir un cycle nul
L'utilisation de MPEG $\Rightarrow$ les 20 premiers bytes sont connus
Ca permet de casser CSS rapidement



Stream ciphers modernes : eStream (2008)

PRG:

$\{0,1\}^s \times r \rightarrow \{0,1\}^n \quad n >> s$

seed nonce ouput

- $E(k, m; r) = m \oplus PRG(k; r)$
- nonce est utilisé juste une seule fois
  - k peut être réutilisée
  - mais $(k; r)$ est toujours différent

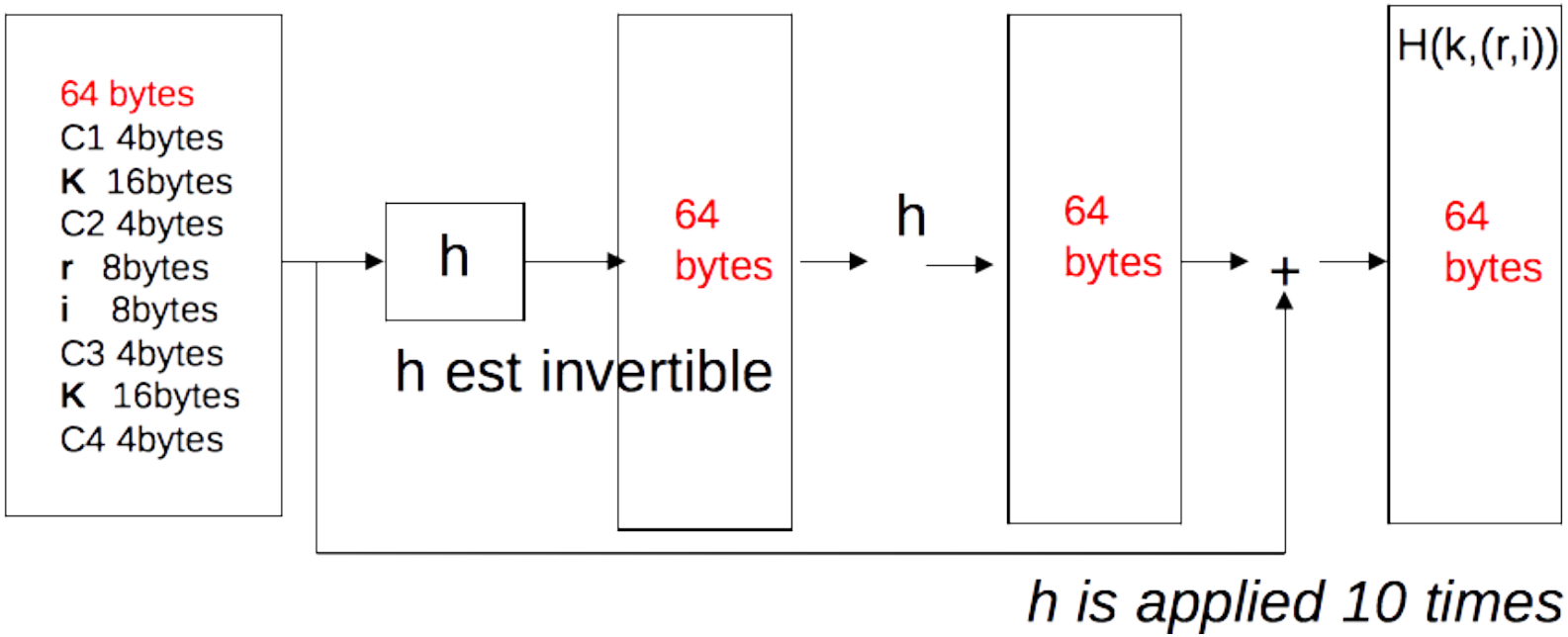
Exemple : Sosemanuk, Salsa20 (Daniel Bernstein, convient aussi bien pour hardware que software)

Salsa20

$\{0,1\}^{128 \text{ or } 256} \times \{0,1\}^{64} \rightarrow \{0,1\}^{n} \quad (max\ n = 273\ bits)$

seed — nonce — output

$$Salsa20(k;r) = H(k,(r,0))||H(k,(r,1))||\ldots$$



Chiffrement par flot et sécurité (clé à usage unique)

Sécurité sémantique

Qu'est-ce qu'un chiffrement sûr?

puissance de l'attaquant (pour l'instant) : il connait le chiffré

Possibles exigences de sécurité :

- l'attaquant ne peut pas retrouver la clé
  Exemple : $E(k, m) = m$ le chiffrement n'est pas sûr et pourtant on ne peut pas retrouver la clé
- l'attaquant ne peut retrouver le clair en entier
  Exemple : $E(k, m_0 \| m_1) = m_0 \| E(k, m_1)$
- Shannon : le chiffré ne doit donner aucune info sur le clair
  $H(m|c) = H(m)$ (H est l'entropie = degré d'incertitude)

Sécurité sémantique (suite)

Soit (E, D) un syst. de chiffrement sur $(\mathcal{K}, \mathcal{M}, \mathcal{C})$
Au lieu de considérer la définition

Définition

(E, D) sur $(\mathcal{K}, \mathcal{M}, \mathcal{C})$ a une sécurité parfaite si $\forall m_0, m_1 \in \mathcal{M}, \quad |m_0| = |m_1|$

$$\{E(k, m_0)\} \;=\; \{E(k, m_1)\} \text{ où } k \xleftarrow{R} \mathcal{K}$$

on préfère la définition

Définition

(E, D) sur $(\mathcal{K}, \mathcal{M}, \mathcal{C})$ a une sécurité sémantique si $\forall m_0, m_1 \in \mathcal{M}, \quad |m_0| = |m_1|$

$$\{E(k, m_0)\} \;\approx_p\; \{E(k, m_1)\} \text{ où } k \xleftarrow{R} \mathcal{K}$$

ET l'adversaire doit choisir explicitement m_0 et m_1

Sécurité sémantique (one time key)

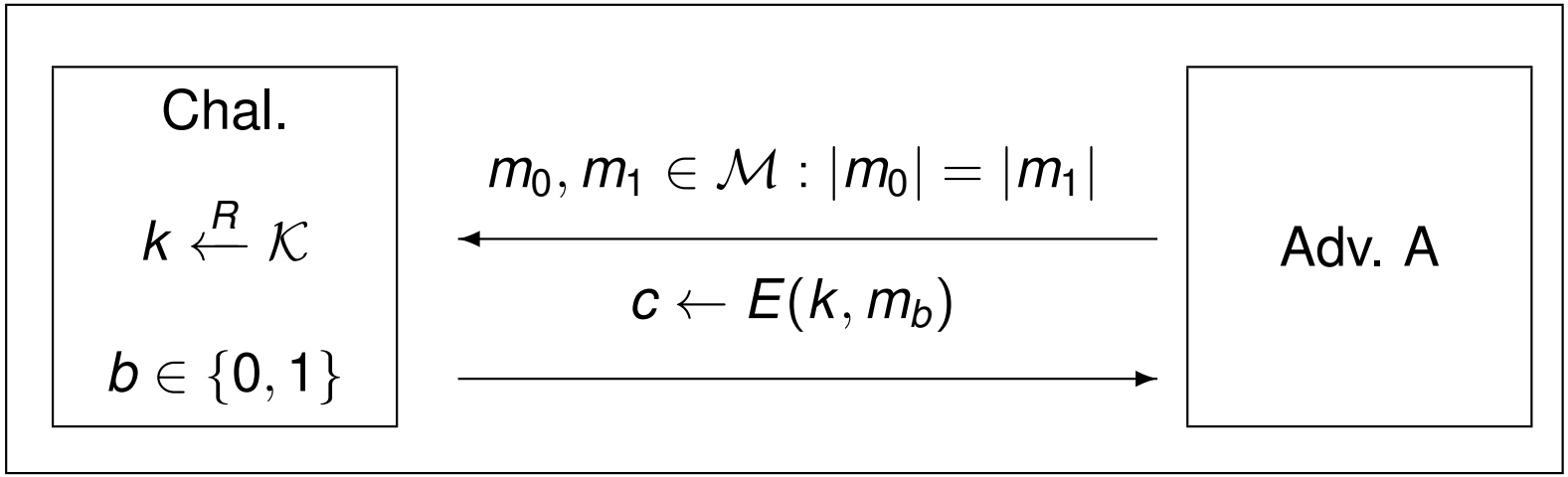
$\mathbb{E} = (E, D)$
Un adversaire (attaquant) A, un challenger Chal.
Chal. choisit une clé aléatoire dans $\mathcal{K}$
A choisit m_0 et m_1 dans $\mathcal{M}$ de même taille
A envoie m_0 et m_1 à Chal.
Chal. choisit b au hasard dans $\{0, 1\}$ et envoie le chiffré de $m_b = c$ à A
A doit deviner $b \in \{0, 1\}$
Pour $b = 0, 1$ on définit 2 expérimentations

- $EXP(0)$: le challenger a chiffré m_0
- $EXP(1)$: le challenger a chiffré m_1

W_b = événement tq $EXP(b) = 1$ (Chal. a chiffré m_b et A répond 1)

$$Adv_{SS}[A, \mathbb{E}] := |Pr[W_0] - Pr[W_1]| \in [0, 1]$$

Sécurité sémantique



$$Adv_{SS}[A, \mathbb{E}] = |Pr[EXP(0) = 1] - Pr[EXP(1) = 1]| \in [0, 1]$$

Sécurité sémantique

Définition

$\mathbb{E} = (E, D)$ est sémantiquement sûr si pour tout algo A efficient,

$$Adv_{SS}[A, \mathbb{E}]$$

est négligeable

$\Rightarrow$ les distributions des chiffrés $\{E(k, m_0)\}$ et $\{E(k, m_1)\}$ sont indistinguables

Exemple

Soit un algo A efficient qui peut toujours déduire LSB du clair à partir du chiffré
Montrer que $\mathbb{E}$ n'est pas sémantiquement sûr

Preuve

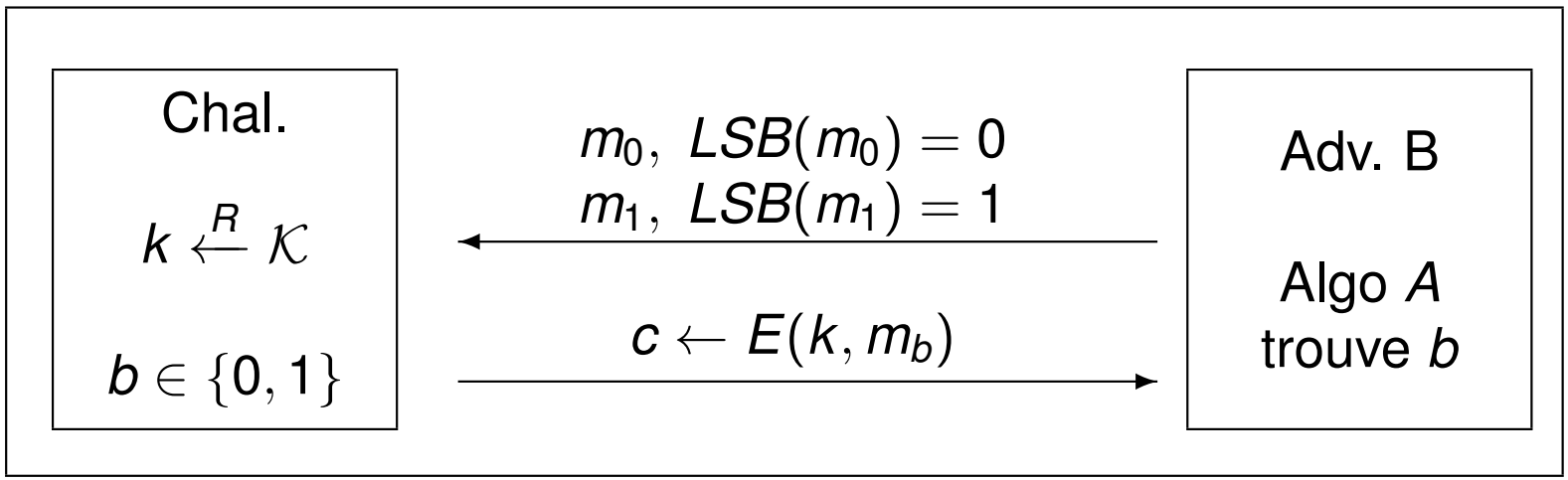
Challenger
$Adv.\ B$
$EXP(0)$, $EXP(1)$
$Pr[EXP(0) = 1]$?, $Pr[EXP(1) = 1]$?

$Adv_{SS}[B, \mathbb{E}]$?

Preuve

Exemple

Soit un algo A efficient qui peut toujours déduire LSB du clair à partir du chiffré
Montrer que $\mathbb{E}$ n'est pas sémantiquement sûr



$$Adv_{SS}[B, \mathbb{E}] = |Pr[EXP(0) = 1] - Pr[EXP(1) = 1]| = |0 - 1| = 1$$

OTP est sémantiquement sûr

Preuve

Challenger
Adv. *A*
$Pr[A(k \oplus m_0) = 1]$?, $Pr[A(k \oplus m_1) = 1]$?
Rappel : les distributions de $\{k \oplus m_0\}$ et de $\{k \oplus m_1\}$ sont identiques

$Adv_{SS}[A, \mathbb{E}]$?

OTP est sémantiquement sûr contre n'importe quel attaquant car les distributions des chiffrés sont égales (pas possible de les distinguer)