Cryptographie
Chiffrement par blocs

Carlos Aguilar
carlos.aguilar@enseeiht.fr

IRIT-IRT

Références

Livres électroniques

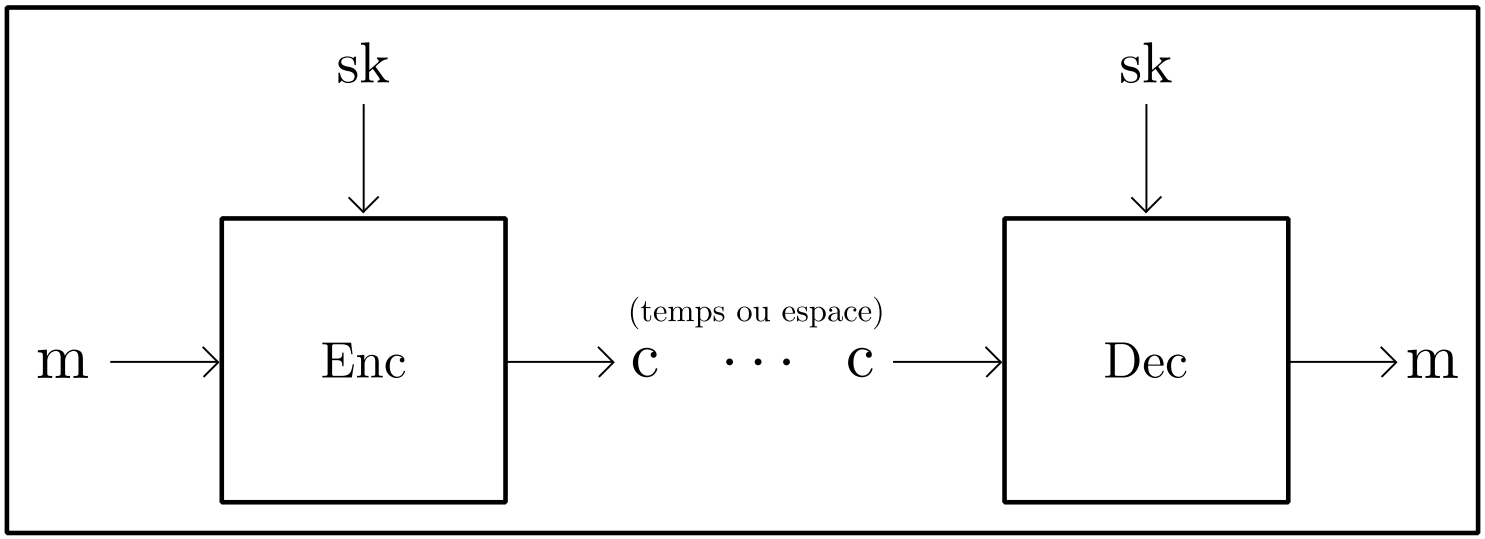
Cornell, https://www.cs.cornell.edu/courses/cs4830/2010fa/lecnotes.pdf
Bristol, http://www.cs.umd.edu/~waa/414-F11/IntroToCrypto.pdf
Stanford, http://crypto.stanford.edu/~dabo/cryptobook/

Cours fortement inspiré de :

[1] Stanford, https://www.coursera.org/course/crypto
[2] Univ. of Virginia, http://www.udacity.com/view#Course/cs387/

Plan

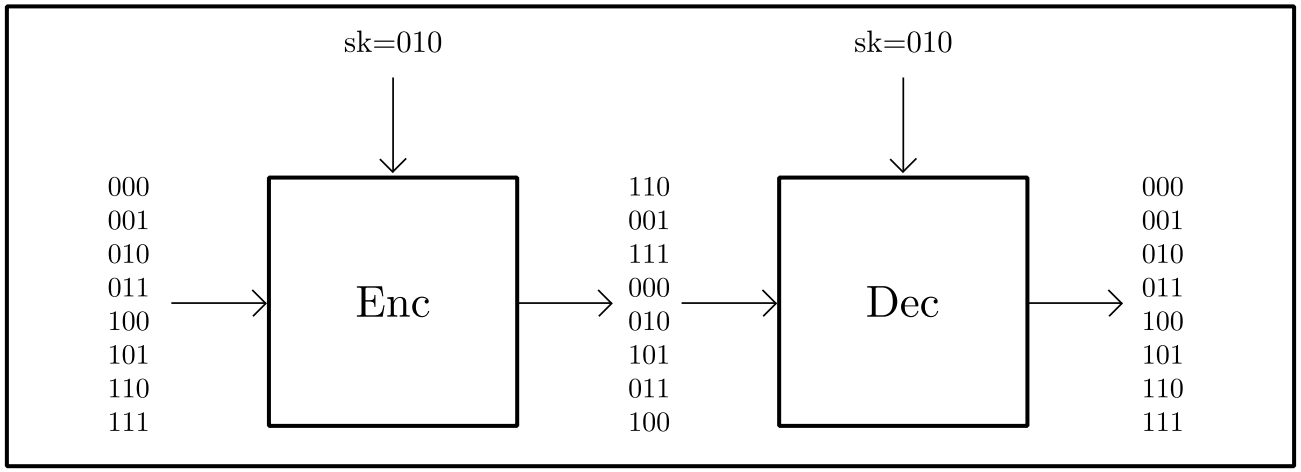
Les algorithmes de chiffrement symétrique



Tailles

On aimerait que la taille de la clé aie un lien avec la sécurité mais ne limite pas la quantité de données que l'on peut chiffrer

Le chiffrement par blocs



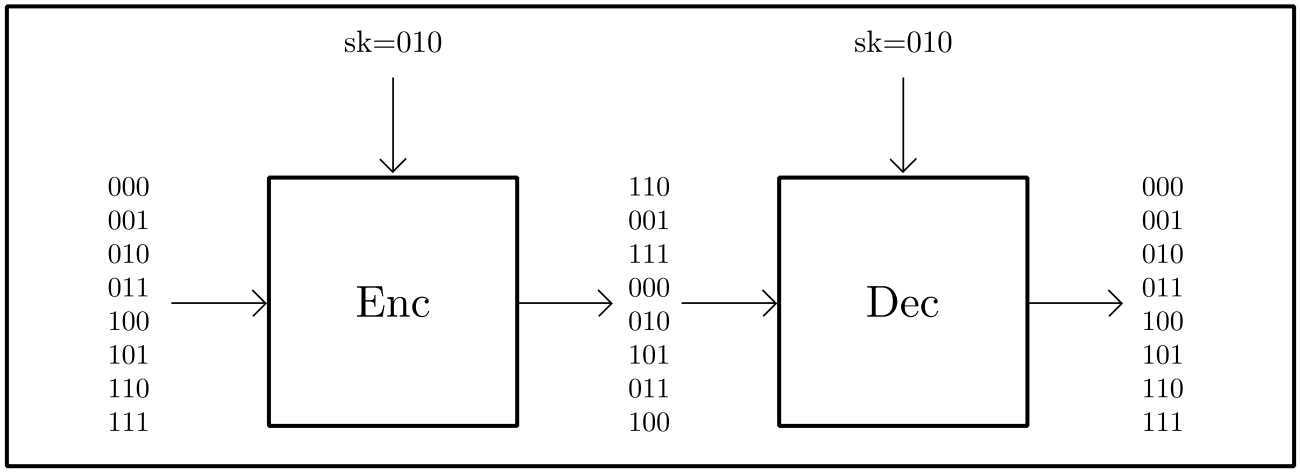
Principe

Clé de k bits (typiquement 128 ou 256)
On chiffre les données par blocs de bits (généralement k aussi)

- Chaque bloc de k bits du message donne un chiffré de k bits
- Pour une clé donnée Enc fait une perm. de $\{0,1\}^k$ et Dec la perm. inverse

Si le message fait moins de k bits on fait du padding
S'il fait plus, on chiffre chaque bloc séparément

Le chiffrement par blocs



Des opérations en boucle sur 10-20 tours

- On dérive une sous clé à partir de la clé avec une fonction dép. de l'itération
- On fait un XOR d'une partie du message avec une partie de la sous clé
- On remplace des blocs avec des tables (inversibles) de substitution
- On permute les bits du message en fonction des bits de la sous clé

Objectif : c'est permutation pseudo-aléatoire (PRP, indistingable d'aléatoire)

Sécurité et attaques

Définition de sécurité

C'est une permutation pseudo-aléatoire (PRP)
De façon plus terre à terre : il faut résister à une découverte de la clé

Attaques

- ciphertext only : l'attaquant ne dispose que de chiffrés
- chosen plaintext : l'attaquant dispose de paires clair/chiffré de son choix

Complexité

- une attaque à clairs choisis doit avoir une complexité de l'ordre de 2^k
- sinon il est cassé d'un point de vue cryptographique
- en pratique tant qu'on ne passe pas sous 2^{100} ...

Algorithmes à connaître (1/2)

Data Encryption Standard (DES, 1975)

Clés de 64 bits mais un bit par octet de parité $\rightarrow$ 56 bits
Coût attaques :

- Annonces théoriques 1977 (20M$), 1993 (1M$)
- 1998 (250K$, 2 jours), pratique réalisé par la EFF
- 2008 (150K$, 7 heures), pratique machine composée de 128 FPGAs (en vente)

Triple DES (3DES)

Chiffrement $\mathrm{Enc}(k_1) \circ \mathrm{Dec}(k_2) \circ \mathrm{Enc}(k_3)$ $\rightarrow$ Déchiffrement $\mathrm{Dec}(k_3) \circ \mathrm{Enc}(k_2) \circ \mathrm{Dec}(k_1)$
Attaque Meet-in-the-Middle (partant d'un couple clair=m/chiffré=c)
- Stocker 2^{56} chiffrés de m
- Déchiffrer c partiellement (un Dec un Enc) 2^{112} fois
- Chercher à chaque fois si le déchiffré partiel est dans la table de chiffrés
$\rightarrow$ 112 bits de sécurité

Pourquoi pas faire un Double DES ?

Algorithmes à connaître (1/2)

Data Encryption Standard (DES, 1975)

Clés de 64 bits mais un bit par octet de parité $\rightarrow$ 56 bits
Coût attaques :

- Annonces théoriques 1977 (20M$), 1993 (1M$)
- 1998 (250K$, 2 jours), pratique réalisé par la EFF
- 2008 (150K$, 7 heures), pratique machine composée de 128 FPGAs (en vente)

Triple DES (3DES)

Chiffrement $\mathrm{Enc}(k_1) \circ \mathrm{Dec}(k_2) \circ \mathrm{Enc}(k_3)$ $\rightarrow$ Déchiffrement $\mathrm{Dec}(k_3) \circ \mathrm{Enc}(k_2) \circ \mathrm{Dec}(k_1)$
Attaque Meet-in-the-Middle (partant d'un couple clair=m/chiffré=c)
- Stocker 2^{56} chiffrés de m
- Déchiffrer c partiellement (un Dec un Enc) 2^{112} fois
- Chercher à chaque fois si le déchiffré partiel est dans la table de chiffrés
$\rightarrow$ 112 bits de sécurité

Pourquoi pas faire un Double DES ?

Meet-in-the-Middle $\rightarrow$ 56 bits de sécurité

Algorithmes à connaître (2/2)

Standard Actuel : Advanced Encryption Standard (AES)

Concours ouvert en 1997 (pas comme pour DES), choix final en 2001
Organisé par le National Institute of Standards and Technology (NIST)
15 soumission $\rightarrow$ 5 finalistes $\rightarrow$ Rjindael

Taille de bloc (clair/chiffré) fixe de 128 bits
Tailles de clés/tours : 128bits/10, 192bits/12, ou 256bits/14
Meilleure attaque connue en 2^{k-2} (attaque biclique 2011)

Autres algorithmes

RC5, IDEA, Blowfish

Modes

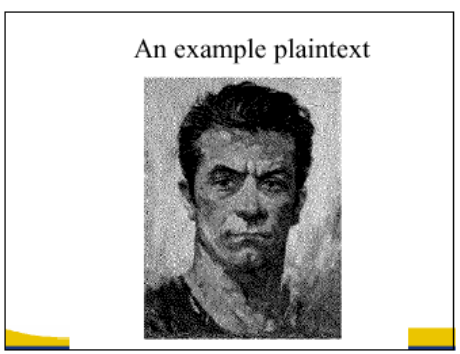
Utilisation

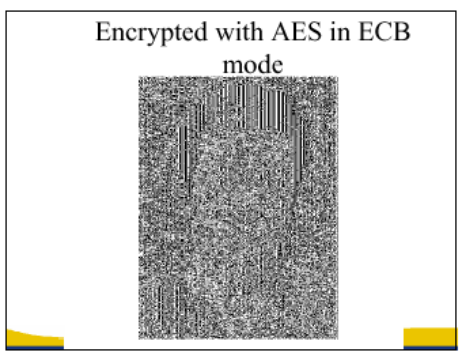
Pour un algorithme il faut choisir une taille de clé et un mode de fonctionnement

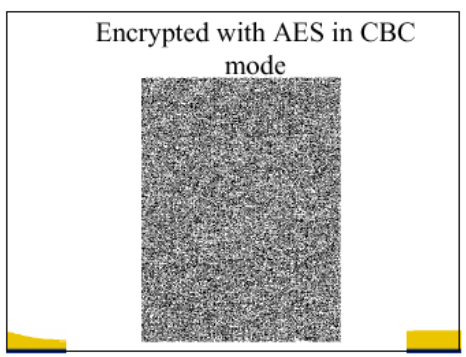
Les modes de fonctionnement

Pour une clé secrète donnée on a un chiffrement déterministe
Mode ECB : deux fois le même clair $\rightarrow$ deux fois le même chiffré

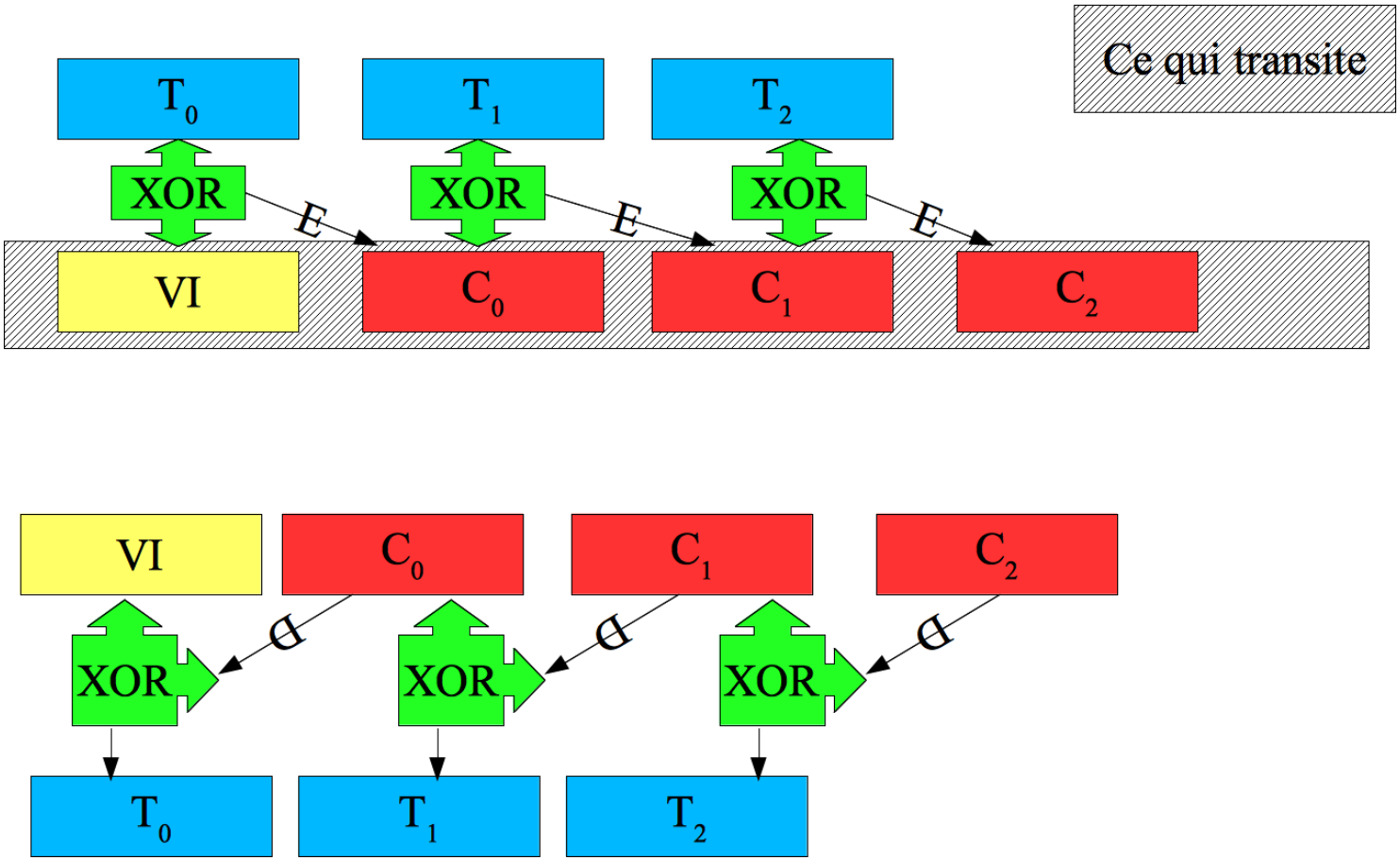
Modes sûrs CBC, CTR, etc.







CBC


Ce qui transite
T0
T1
T2
XOR
XOR
XOR
E
E
E
VI
C0
C1
C2
VI
C0
C1
C2
XOR
XOR
XOR
T0
T1
T2

Définition

Définition d'une permutation pseudo-aléatoire (PRP)

C'est une famille de fonctions telle que le choix uniforme d'un élément est indistingable du choix uniforme d'une permutation parmi toutes les permutations possibles

Indistingable ?

Informellement : tout test **exploitable** et permettant de dire que nous avons à faire à un cas ou l'autre, prends un temps de calcul **prohibitif**

Complexité : temps/algorithme polynomial

Définition

On dit qu'un algorithme est (en temps) polynomial, si son temps d'exécution suit un polynôme en la taille de l'entrée

Remarques :

- c'est la notion en complexité d'un algorithme **efficace**
- n taille de l'entrée : n, n^3, $10^9 * n^{10^6}$ sont des temps polynomiaux
- définit par opposition aux temps exponentiel (e.g. 2^n) et quasi-exponentiel (e.g. $2^{\sqrt{n}}$) qui sont considérés **prohibitifs**

Quiz : quelles opérations sont réalisables en temps polynomial ?

1. chiffrer/dechiffrer par OTP un message de taille n
2. essayer toutes les clés de taille n pour un déchiffrement
3. multiplier deux nombres de taille n
4. calculer a^b pour a, b de taille n
5. calculer $a^b \mod c$ pour a, b, c de taille n

Complexité : temps/algorithme polynomial

Définition

On dit qu'un algorithme est (en temps) polynomial, si son temps d'exécution suit un polynôme en la taille de l'entrée

Remarques :

- c'est la notion en complexité d'un algorithme **efficace**
- n taille de l'entrée : n, n^3, $10^9 * n^{10^6}$ sont des temps polynomiaux
- définit par opposition aux temps exponentiel (e.g. 2^n) et quasi-exponentiel (e.g. $2^{\sqrt{n}}$) qui sont considérés **prohibitifs**

Quiz : quelles opérations sont réalisables en temps polynomial ?

1. chiffrer/dechiffrer par OTP un message de taille n
2. essayer toutes les clés de taille n pour un déchiffrement
3. multiplier deux nombres de taille n
4. calculer a^b pour a, b de taille n
5. calculer $a^b \mod c$ pour a, b, c de taille n

Complexité : fonction négligeable

Définition : comportement asymptotique

On dit que quelque chose est vrai asymptotiquement pour une fonction ou un algorithme quand au delà d'une certaine limite c'est tout le temps vrai

Exemple : x^2 est supérieur à $10^{12} + x$ asymptotiquement

Définition : fonction négligeable

Une fonction est dite négligeable si pour tout polynôme p la valeur absolue de la fonction est majorée asymptotiquement par $1/p(n)$, n étant l'entrée de la fonction

Quiz : quelles fonctions sont négligeables en n ?

1. $1/2^n$
2. $1/(10^9 * n^{10^6})$
3. $1/1.0000001^{\sqrt{n}}$
4. f qui vaut $1/n$ si l'entrée est divisible par 10^9 et $1/2^{\log^2 n}$ sinon

Complexité : fonction négligeable

Définition : comportement asymptotique

On dit que quelque chose est vrai asymptotiquement pour une fonction ou un algorithme quand au delà d'une certaine limite c'est tout le temps vrai

Exemple : x^2 est supérieur à $10^{12} + x$ asymptotiquement

Définition : fonction négligeable

Une fonction est dite négligeable si pour tout polynôme p la valeur absolue de la fonction est majorée asymptotiquement par $1/p(n)$, n étant l'entrée de la fonction

Quiz : quelles fonctions sont négligeables en n ?

1. $1/2^n$
2. $1/(10^9 * n^{10^6})$
3. $1/1.0000001^{\sqrt{n}}$
4. f qui vaut $1/n$ si l'entrée est divisible par 10^9 et $1/2^{\log^2 n}$ sinon

Complexité : avantage non-négligeable

Définition : avantage non-négligeable

Un algorithme devant trouver un résultat parmi m, a un avantage ϵ par rapport à deviner s'il peut trouver le bon résultat avec une probabilité supérieure à $1/m + \epsilon$

L'avantage d'un algorithme est dit non-négligeable si il suit une fonction qui n'est pas négligeable en n, pour des entrées de taille n

Remarques

- un algorithme dont l'avantage est non-négligeable en la taille des entrées est considéré en complexité **exploitable**
- même si $P[\text{succes}] = 1/2 + 1/(10^9 * n^{10^6})$ l'algorithme est utilisable
- définit par opposition aux avantages exponentiel (e.g. 2^n) et quasi-exponentiel (e.g. $2^{\sqrt{n}}$) qui sont considérés **non-significatifs**

Complexité : avantage non-négligeable

Quels algorithmes ont un avantage non-négligeable ?

1. pour un disque chiffré avec une clé de n bits, essayer n^{10} clés puis si on a pas trouvé la bonne tenter une au hasard parmi les restantes
2. même algo mais en essayant $2^{n/2}$ clés

Les distingueurs

Jeu d'indistingabilité

Soient A_0 et A_1 des algorithmes randomisés prenant uniquement en entrée une source d'aléa parfaite. On définit généralement le jeu suivant :

1. On tire $b \leftarrow \{0, 1\}$
2. On donne une sortie de A_b à l'attaquant
3. L'attaquant étudie la sortie et propose un bit b'

L'objectif de l'attaquant est d'avoir $b' = b$
L'avantage de l'attaquant à ce jeu est la quantité Adv $= | P[b' = b] - 1/2 |$

Définition : indistingabilité

S'il n'existe pas d'algorithme en temps polynomial qui distingue A_0 de A_1 avec un avantage non-négligeable, on dit que les (distributions de sortie des) deux algorithmes sont indistingables

Chiffrement par blocs sûr

Chiffrement par bloc indistingable d'une permutation aléatoire

Utilité d'un distingueur et amplification

Phénomène d'amplification

Supposons que l'algorithme polynomial A permet de distinguer deux algorithmes A_0 et A_1 mais avec une toute petite probabilité (e.g. $1/n^3$ pour n de plusieurs centaines)
Alors il existe un algorithme polynomial B qui distingue A_0 et A_1 avec une probabilité très proche de 1 (i.e. la différence avec 1 est negligeable)

Amplification par 2

Si B exécute A deux fois et fait un choix par majorité quel est son avantage ?

Non prouvé

B exécute le jeu d'indistingabilité en utilisant l'algorithme A un nombre polynomial de fois $\rightarrow$ probabilité exponentiellement proche de 1

Utilité d'un distingueur et amplification

Phénomène d'amplification

Supposons que l'algorithme polynomial A permet de distinguer deux algorithmes A_0 et A_1 mais avec une toute petite probabilité (e.g. $1/n^3$ pour n de plusieurs centaines)
Alors il existe un algorithme polynomial B qui distingue A_0 et A_1 avec une probabilité très proche de 1 (i.e. la différence avec 1 est negligeable)

Amplification par 2

Si B exécute A deux fois et fait un choix par majorité quel est son avantage ?
$P[\text{succès}] =$
$P[A \text{ réussit 1}] * P[A \text{ réussit 2}] + 1/2 * (2 * P[A \text{ réussit 1}] * P[A \text{ rate 2}]) =$
$(1/2 + \epsilon)^2 + (1/2 + \epsilon) * (1/2 - \epsilon) = 1/4 + 2\epsilon + \epsilon^2 + 1/4 - \epsilon^2 = 1/2 + 2\epsilon$

Non prouvé

B exécute le jeu d'indistingabilité en utilisant l'algorithme A un nombre polynomial de fois $\rightarrow$ probabilité exponentiellement proche de 1

Complexité et cryptographie

Principe

Trouver des primitives cryptographiques (chiffrement, déchiffrement, etc.)

- avec un coût polynomial en la taille des entrées
- avec des attaques ayant un coût super-polynomial en cette même taille

Puis choisir une taille d'entrée telle que les attaques soient infaisables

Bits de sécurité et attaques

Algorithme avec k bits de sécurité $\Leftrightarrow$ Meilleure attaque en 2^k operations
Si l'attaque est en 1.01^n avec n la taille de l'entrée il faut prendre $n = 200 * k$

Repères

$k = 30$ 1s-ordi, $k = 55$ 1an-ordi, $k = 85$ 1an-OST, $k = 100$ 32000ans-OST,
$k = 115$ 1age-univers-OST, $k = 240 + 45 + 30 = 315$ 1age-univers-ADU
Sécurités considérées aujourd'hui $k = 100, 112, 128, 256$