

Cryptographie

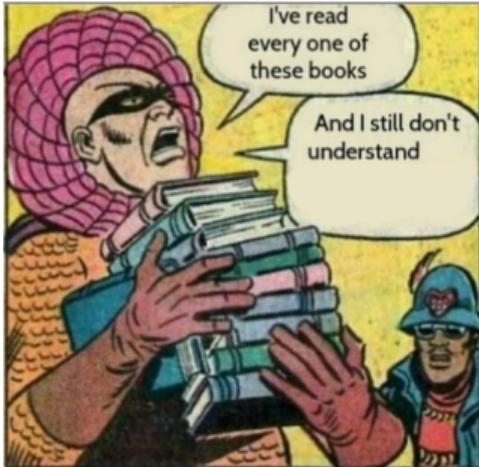
Introduction : définitions de sécurité
& méthodes de preuves

Léo COLISSON PALAIS
Master CSI 2024 – 2025

leo.colisson-palais@univ-grenoble-alpes.fr

<https://leo.colisson.me/teaching.html>

Quelques références



- Base de ce cours :
The Joy of Cryptography, Mike Rosulek
<https://joyofcryptography.com/>
- *Introduction to Modern Cryptography*, Jonathan Katz & Yehuda Lindell
- *Foundation of Cryptography*, Oded Goldreich

Planning

- 1 journée le 16 mai
- 3 journées du 18 au 20 juin

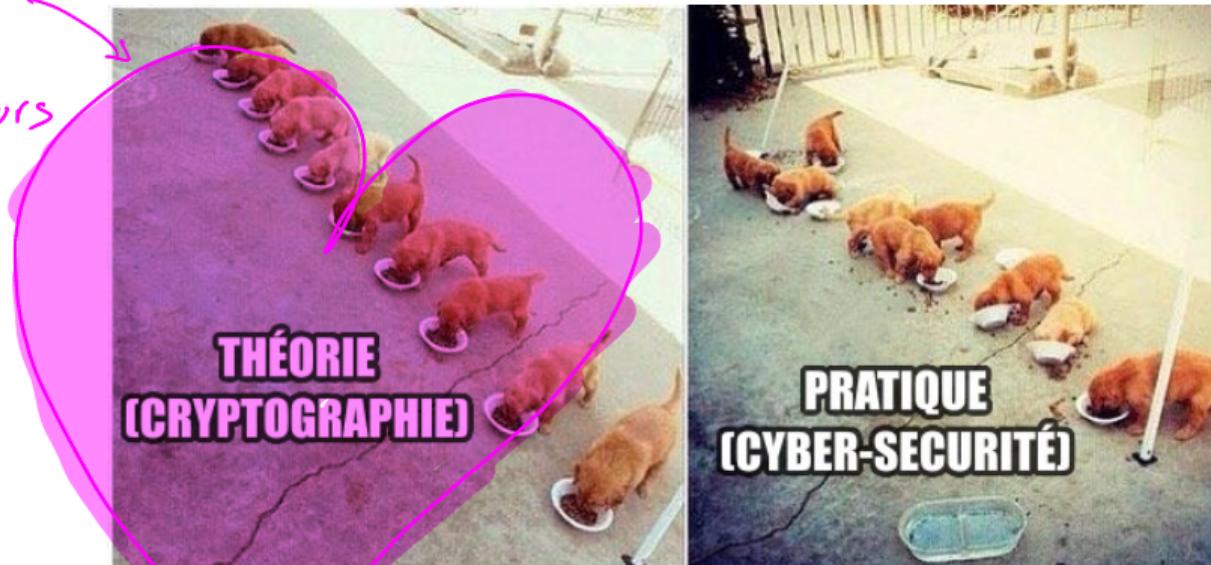
Cryptographie vs cyber-sécurité



Inspiré par <https://x.com/WauteletB/status/700980326215065600/photo/1>

Cryptographie vs cyber-sécurité

Coeur avec les doigts = ce cours



Inspiré par <https://x.com/WauteletB/status/700980326215065600/photo/1>

Buts

Buts:

- Ouvrir les boîtes : **comment sont définies** les primitives cryptographiques ?



https://www.youtube.com/watch?v=a_HIHG5Nvpk
(légèrement améliorée)

Buts

Buts:

- Ouvrir les boîtes : **comment sont définies** les primitives cryptographiques ?
- Définir précisément ce **que veut dire "sécurisé"** : modèles, hypothèses, définitions...



https://www.youtube.com/watch?v=a_HIHG5Nvpk
(légèrement améliorée)

Buts

Buts:

- Ouvrir les boîtes : **comment sont définies** les primitives cryptographiques ?
- Définir précisément ce **que veut dire "sécurisé"** : modèles, hypothèses, définitions...
- Comment **écrire des preuves de sécurité** formellement ?



UNBOXING NOUVEAUTÉS : On déballe TOUTES les primitives cryptographiques ensemble !

 Sananais 
3,022 M d'abonnés

Abonner

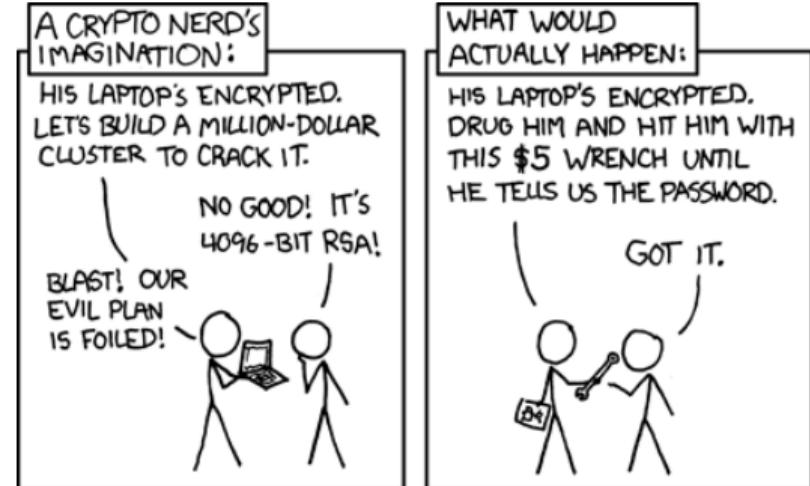
4,5 k Partager Enregistrer ...

https://www.youtube.com/watch?v=a_HIHG5Nvpk
(légèrement améliorée)

Buts

Buts:

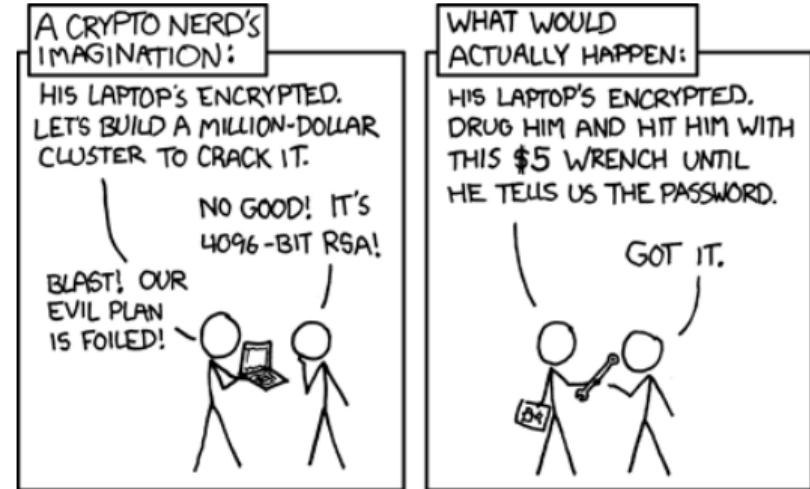
- Ouvrir les boîtes : **comment sont définies** les primitives cryptographiques ?
- Définir précisément ce **que veut dire "sécurisé"** : modèles, hypothèses, définitions...
- Comment **écrire des preuves de sécurité** formellement ?
- Comprendre les **implications de ces modèles** ?



Buts

Buts:

- Ouvrir les boîtes : **comment sont définies** les primitives cryptographiques ?
- Définir précisément ce **que veut dire "sécurisé"** : modèles, hypothèses, définitions...
- Comment **écrire des preuves de sécurité** formellement ?
- Comprendre les **implications de ces modèles** ?
- Les choses à **ne pas faire** !!



Cours moodle associé



<https://moodle.caseine.org/course/view.php?id=1317>

⇒ Faire le quiz

Notations

Notation

Signification

$x \xleftarrow{\$} X$ x est un élément aléatoire uniformément tiré dans l'ensemble X

$y \leftarrow A(x)$ Si A est un algorithme probabiliste ou une distribution, on exécute A sur l'entrée x et enregistre le résultat dans y

$x \stackrel{?}{=} y$ Retourne 1 (vrai) si x est égal à y , 0 (faux) sinon

$\text{negl}(\lambda)$ Une fonction arbitraire f qui est négligeable (= plus petite que n'importe quel polynôme), i.e. $\forall c \in \mathbb{N}, \lim_{\lambda \rightarrow \infty} \lambda^c f(\lambda) = 0$

$\text{poly}(\lambda)$ Fonction arbitraire f plus petite qu'un polynôme, i.e.
 $\exists c \in \mathbb{N}, \forall N \in \mathbb{N}, \forall \lambda > N, f(\lambda) \leq \lambda^c$

Quelles fonctions sont négligeables ?



- A $f(\lambda) = \frac{1}{2^\lambda}$
- B $f(\lambda) = \frac{1}{\lambda^{1000}}$
- C $f(\lambda) = 2^{-\log \lambda}$

Notations

Notation	Signification
$x \xleftarrow{\$} X$	x est un élément aléatoire uniformément tiré dans l'ensemble X
$y \leftarrow A(x)$	Si A est un algorithme probabiliste ou une distribution, on exécute A sur l'entrée x et enregistre le résultat dans y
$x \stackrel{?}{=} y$	Retourne 1 (vrai) si x est égal à y , 0 (faux) sinon
$\text{negl}(\lambda)$	Une fonction arbitraire f qui est négligeable (= plus petite que n'importe quel polynôme), i.e. $\forall c \in \mathbb{N}, \lim_{\lambda \rightarrow \infty} \lambda^c f(\lambda) = 0$
$\text{poly}(\lambda)$	Fonction arbitraire f plus petite qu'un polynôme, i.e. $\exists c \in N, \forall \lambda > N, f(\lambda) \leq \lambda^c$
NB: $\text{negl}(\lambda) + \text{negl}(\lambda) = \text{negl}(\lambda)$, $\text{negl}(\lambda) \times \text{negl}(\lambda) = \text{negl}(\lambda)$, $\text{poly}(\lambda)\text{negl}(\lambda) = \text{negl}(\lambda)$	

Sécurisé ? C'est à dire ?

Exemple du chiffrement symétrique

Principe de Kerckhoffs

Principe de Kerckhoffs

L'adversaire connaît tous les détails du protocole (mais ne peut pas connaître directement les valeurs tirées au sort lors de l'exécution du protocole)



Chiffrement symétrique vs asymétrique

→ = à clé privée

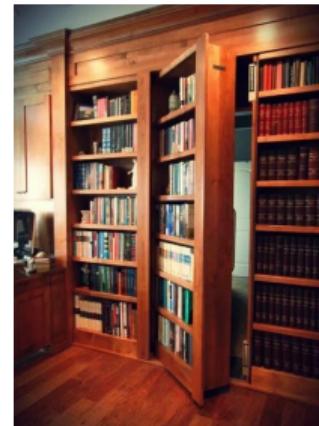
Chiffrement symétrique

Les deux participants partagent le même secret

→ = à clé publique

Chiffrement asymétrique

Une personne a une information additionnelle (**porte dérobée**) qui permet d'inverser une fonction facilement









m



m



c



m



m



c



m

Cryptographie symétrique vs asymétrique

Chiffrement asymétrique

- 😊 Pas besoin d'échanger de secrets (e.g. internet)
- 😱 Hypothèses plus fortes factorisation, LWE... (fonctions très structurées)
- 😱 Moins efficace
- 😱 Pas de sécurité statistique

Chiffrement symétrique

- 😱 Doit partager des secrets
- 😊 Plus faibles hypothèses (moins de structure)
- 😊 Plus efficace
- 😊 Sécurité statistique possible (mais pas pratique)

⇒ Systèmes hybrides: **combiner les deux** = meilleur des deux mondes (efficace + pas de secret à distribuer)

Modèles de sécurité

Quand on crée un système cryptographique, on souhaite dire:

“Le protocole XXX est **securisé**

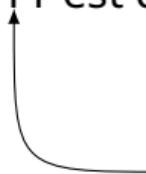
Modèles de sécurité

Quand on crée un système cryptographique, on souhaite dire:

“Le protocole XXX est **securisé**
que YYY est difficile.”

en supposant

Hypothèse calculatoire = qu'est-ce qui est supposé dur pour l'attaquant ?
E.g. DDH, factorisation, LWE...



Modèles de sécurité

Quand on crée un système cryptographique, on souhaite dire:

Hypothèse de setup = à quoi ont accès les participants honnêtes et malicieux
(e.g. oracle modélise/remplace une fonction de hash)

“Le protocole XXX est **securisé** dans le modèle plain/CRS/RO en supposant que YYY est difficile.”

Hypothèse calculatoire = qu'est-ce qui est supposé dur pour l'attaquant ?
E.g. DDH, factorisation, LWE...

Modèles de sécurité

Quand on crée un système cryptographique, on souhaite dire:

Hypothèse de setup = à quoi ont accès les participants honnêtes et malicieux
(e.g. oracle modélise/remplace une fonction de hash)

“Le protocole XXX est **securisé** dans le modèle plain/CRS/RO en supposant que YYY est difficile.”

???

Hypothèse calculatoire = qu'est-ce qui est supposé dur pour l'attaquant ?
E.g. DDH, factorisation, LWE...

Modèles de sécurité

Quand on crée un système cryptographique, on souhaite dire:

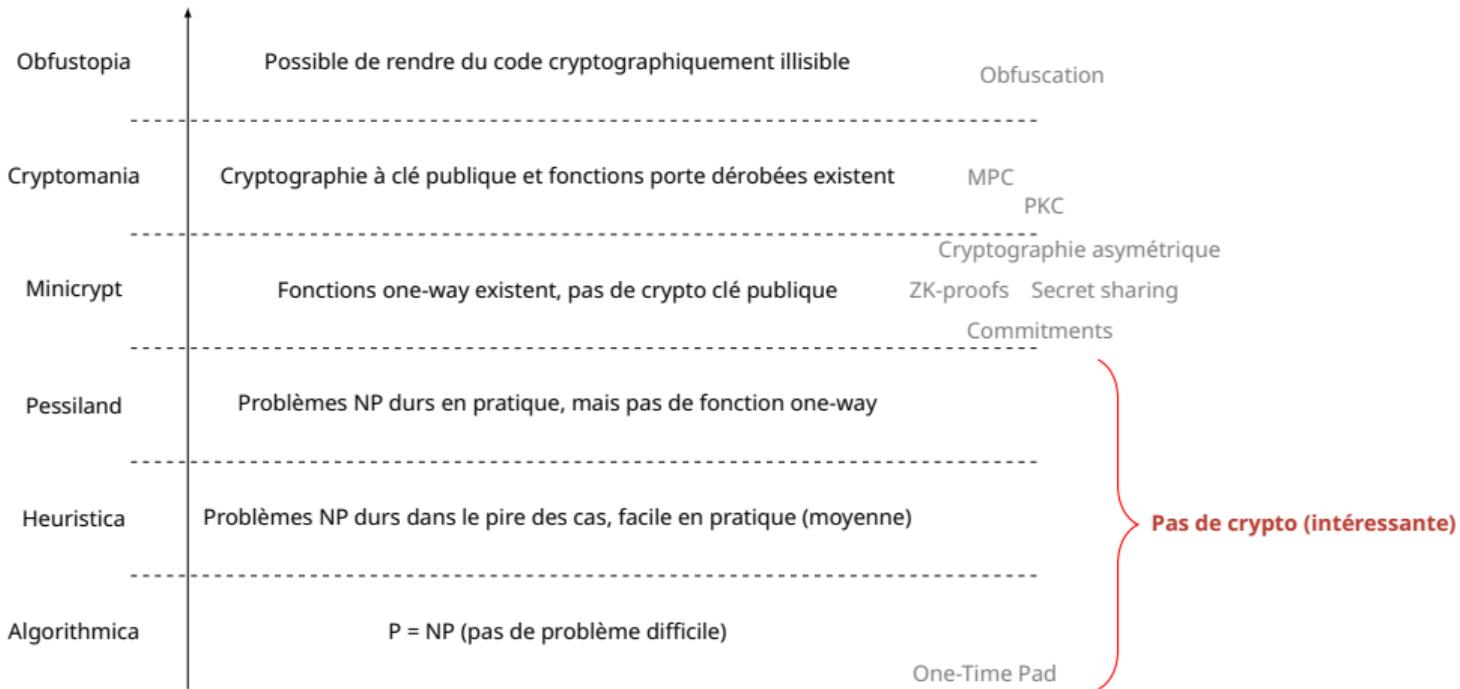
Hypothèse de setup = à quoi ont accès les participants honnêtes et malicieux
(e.g. oracle modélise/remplace une fonction de hash)

“Le protocole XXX est **securisé** dans le modèle plain/CRS/RO en supposant que YYY est difficile.”

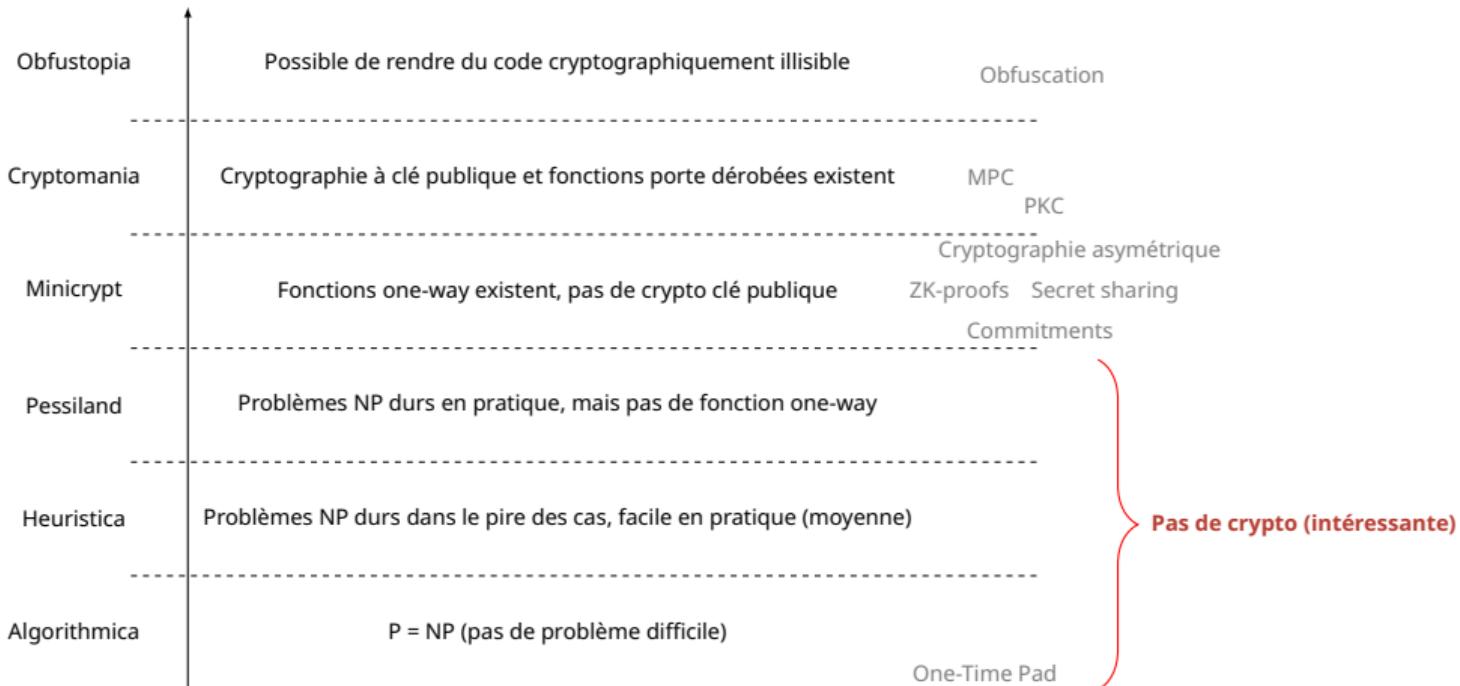
Modèle de sécurité = garanties (à prouver) en terme de sécurité
E.g. intuitivement “l’adversaire est incapable de trouver le message”

Hypothèse calculatoire = qu’est-ce qui est supposé dur pour l’attaquant ?
E.g. DDH, factorisation, LWE...

Hypothèses de sécurité : les mondes d'Impagliazzo

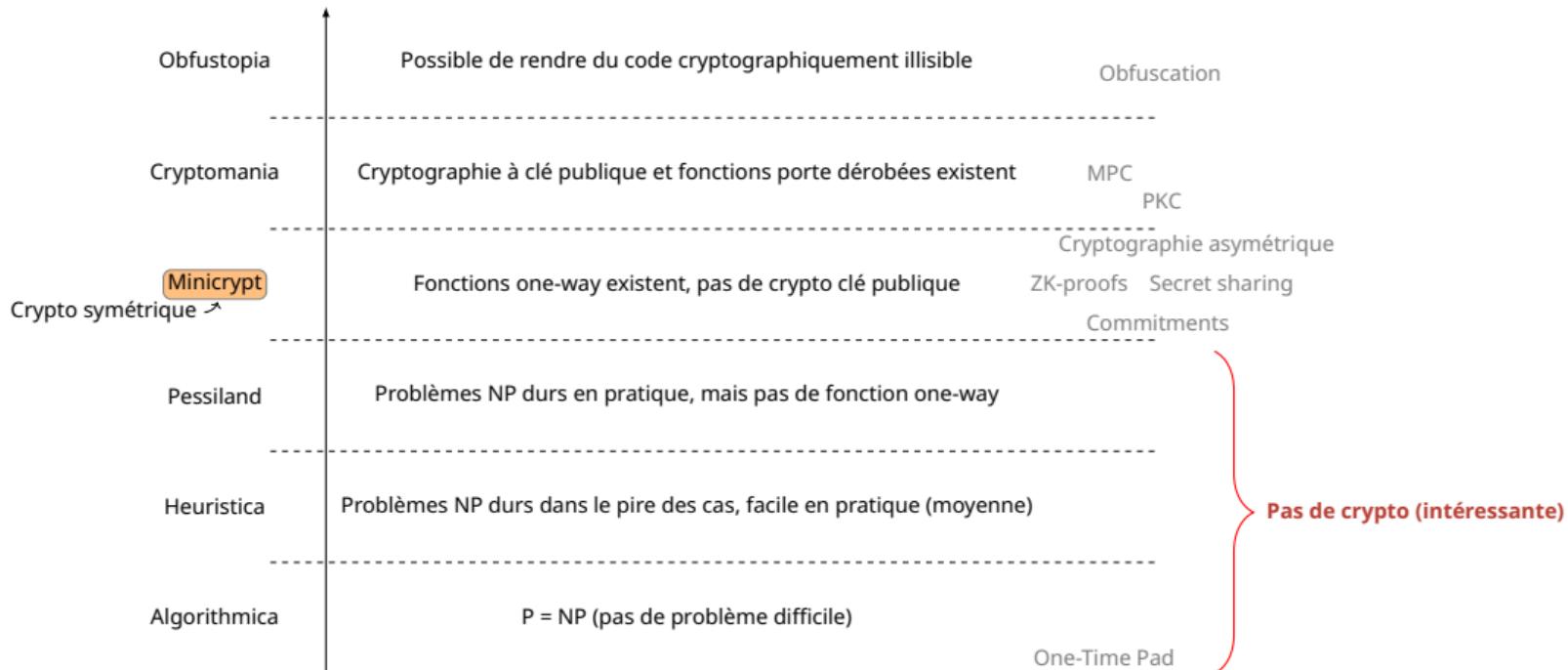


Hypothèses de sécurité : les mondes d'Impagliazzo



Grande question (plus dure que $P = NP$) : dans quel monde sommes nous ?

Hypothèses de sécurité : les mondes d'Impagliazzo



Grande question (plus dure que $P = NP$) : dans quel monde sommes nous ?

Pas de sécurité absolue

Puisque l'on ne sait pas dans quel monde on est = **pas de sécurité inconditionnelle** (sauf One-Time Pad) ⇒ toujours besoin **d'hypothèses**:

Hypothèses calculatoires

= l'adversaire ne peut pas ...

Factorisation/courbes elliptiques

Learning With Errors (LWE)

Crypto basée sur les codes

Existence de fonctions one-way (difficiles à inverser), pseudo-random permutations...

Indistinguishable Obfuscation (iO)...

Hypothèses de setup

= participants ont accès à

Plain model

Common Reference String (CRS)

Random Oracle (RO) model

Remplacer RO avec fonction de hash = heuristique
(pas de preuve que la sécurité est préservée)

Important de **clairement l'indiquer** et de comprendre leurs implications!

Pas de sécurité absolue

Puisque l'on ne sait pas dans quel monde on est = **pas de sécurité inconditionnelle** (sauf One-Time Pad) ⇒ toujours besoin **d'hypothèses**:

Hypothèses calculatoires

= l'adversaire ne peut pas ...

Factorisation/courbes elliptiques

(cassé contre ordinateur quantique)

Learning With Errors (LWE)

Crypto basée sur les codes

Existence de fonctions one-way (difficiles à inverser), pseudo-random permutations...

Indistinguishable Obfuscation (iO)...

Hypothèses de setup

= participants ont accès à

Plain model

Common Reference String (CRS)

Random Oracle (RO) model

Remplacer RO avec fonction de hash = heuristique
(pas de preuve que la sécurité est préservée)

Important de **clairement l'indiquer** et de comprendre leurs implications!

Modèles de sécurité

Quand on crée un système cryptographique, on souhaite dire:

Hypothèse de setup = à quoi ont accès les participants honnêtes et malicieux
(e.g. oracle modélise/remplace une fonction de hash)

“Le protocole XXX est **securisé** dans le modèle plain/CRS/RO en supposant que YYY est difficile.”

Modèle de sécurité = garanties (à prouver) en terme de sécurité
E.g. intuitivement “l’adversaire est incapable de trouver le message”

Hypothèse calculatoire = qu’est-ce qui est supposé dur pour l’attaquant ?
E.g. DDH, factorisation, LWE...

Modèles de sécurité

Facile de dire intuitivement ce que l'on souhaite **difficile de trouver un bon modèle** de sécurité qui capture tous les comportements indésirables :

E.g. pour chiffrement :



Essai 1 : "Étant donné un chiffrement de m , un adversaire doit être incapable de trouver m ". Est-ce une bonne définition de sécurité ? (si non, trouver un contre exemple)

- A Oui
- B Non

Facile de dire intuitivement ce que l'on souhaite **difficile de trouver un bon modèle** de sécurité qui capture tous les comportements indésirables :

E.g. pour chiffrement :

Essai 1 : "Étant donné un chiffrement de m , un adversaire doit être incapable de trouver m ". Est-ce une bonne définition de sécurité ? (si non, trouver un contre exemple)



- A Oui
- B Non Être capable de récupérer 3/4 du message est déjà un gros problème ! E.g.
 $m = ??????????????$, donc nous attaquons demain"

Security models



Essai 2 : "Étant donné un chiffrement de m , un adversaire devrait être incapable de retrouver un seul bit de m ". Est-ce une bonne définition de sécurité ? (si non, trouver un contre exemple)

- A Oui
- B Non

Security models

Essai 2 : "Étant donné un chiffrement de m , un adversaire devrait être incapable de retrouver un seul bit de m' ". Est-ce une bonne définition de sécurité ? (si non, trouver un contre exemple)



- A Oui X
- B Non ✓ Savoir quels groupes de bits diffèrent donne beaucoup d'info :



NE JAMAIS FAIRE ÇA

**UN CHIFFREMENT DOIT TOUJOURS ÊTRE
NON-DETERMINISTE!!!**

NE JAMAIS FAIRE ÇA

**UN CHIFFREMENT DOIT TOUJOURS ÊTRE
NON-DETERMINISTE!!!**

**NE FAITES JAMAIS VOTRE PROPRE CHIFFREMENT,
IL SERRA NON SÉCURISÉ!!!**



Meilleure solution

Plutôt que de demander à ce que l'adversaire soit incapable d'apprendre XXX sur m à partir de $\text{Enc}_k(m)$,



Mieux de dire qu'il est **incapable de distinguer** entre $\text{Enc}_k(m_0)$ et $\text{Enc}_k(m_1)$ pour m_0 et m_1 qu'il a choisi.



Ça **implique** en particulier qu'il est incapable d'apprendre XXX, sinon il pourrait choisir m_0 et m_1 avec XXX différent, récupérer XXX à partir de $\text{Enc}_k(m_b)$, et déterminer b en fonction de XXX.

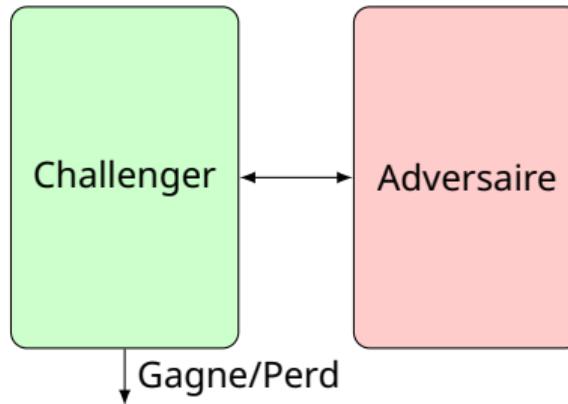
Comment formaliser cette intuition ?

Modèles de sécurité

Mais alors, comment définir un protocole/chiffrement sécurisé ? \Rightarrow Il n'y a pas une, mais plusieurs définitions de sécurité (avec des garanties différentes)

3 **classes** de modèles de sécurité :

1: Sécurité basée sur les jeux = Fixe un **challenger** (définit la sécurité) :



Stronger models

- General composability
- Sequential composability
- Game-based security

Sécurisé si pour chaque adversaire, **la probabilité de gagner est “basse”**
(souvent $1/2 + \text{negl}(\lambda)$ ou $0 + \text{negl}(\lambda)$ suivant le jeu)

Modèles de sécurité

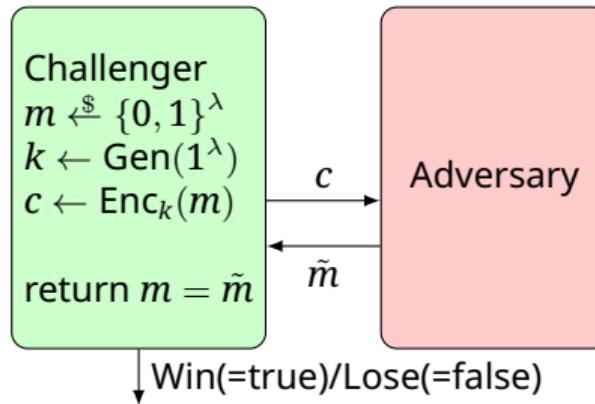
Mais alors, comment définir un protocole/chiffrement sécurisé ? \Rightarrow Il n'y a pas une, mais plusieurs définitions de sécurité (avec des garanties différentes)

3 **classes** de modèles de sécurité :

1: Sécurité basée sur les jeux = Fixe un **challenger** (définit la sécurité) :

Stronger models

- General composability
- Sequential composability
- Game-based security



Sécurisé si pour chaque adversaire, **la probabilité de gagner est “basse”**
(souvent $1/2 + \text{negl}(\lambda)$ ou $0 + \text{negl}(\lambda)$ suivant le jeu)

Modèles de sécurité

Mais alors, comment définir un protocole
plusieurs définitions de sécurité (avec des

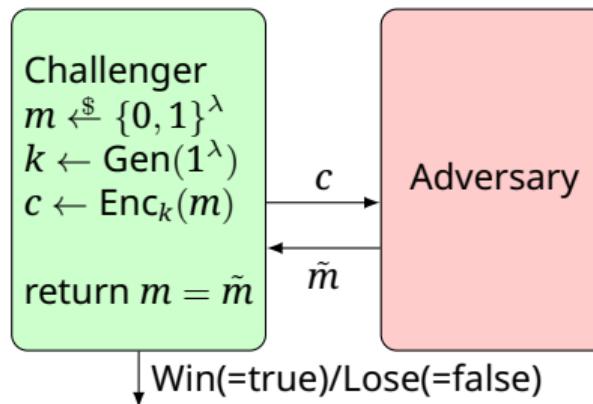
Q: Est-ce que ce challenger correspond à la définition:
(A) "n'apprends pas m "
(B) ou "n'apprends aucun bit sur m "?

3 **classes** de modèles de sécurité :

1: Sécurité basée sur les jeux = Fixe un **challenger** (définit la sécurité) :

Stronger models

- General composability
- Sequential composability
- Game-based security



Sécurisé si pour chaque adversaire, **la probabilité de gagner est "basse"**
(souvent $1/2 + \text{negl}(\lambda)$ ou $0 + \text{negl}(\lambda)$ suivant le jeu)

Modèles de sécurité

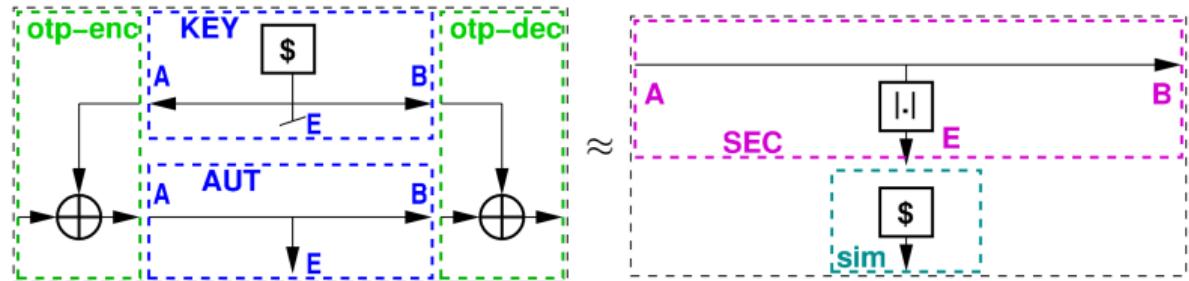
Mais alors, comment définir un protocole/chiffrement sécurisé ? \Rightarrow Il n'y a pas une, mais plusieurs définitions de sécurité (avec des garanties différentes)

3 **classes** de modèles de sécurité :

2 & 3: Frameworks composables = sécurité basée sur un **simulateur** qui traduit les attaques du monde réel pour attaquer une **fonctionalité** (personne tierce de confiance) dans un monde idéal, supposé sécurisé par définition :

Stronger models

- General composability
- Sequential composability
- Game-based security



Principaux frameworks : standalone security (sequential), Universal Composability [Can10], Abstract Cryptography [MR11, M12] (general)

Modèles de sécurité : comparaison

	Sécurité basée sur les jeux	Sécurité composable/ basée sur la simulation
Simple à comprendre	✓	✗
Simple de voir que c'est la "bonne" définition	✗	✓
Garanties plus fortes	✗	✓
Notions particulièrement adaptées au modèle	Signatures	MPC
Garanties quand les protocoles sont composés	✗	✓
Résultats d'impossibilité sont rares	✓	✗
Exemple de définition équivalentes	IND-CPA	Semantic-security

Modèles de sécurité : comparaison

Focus de ce cours

Sécurité basée sur les jeux

Simple à comprendre



Simple de voir que c'est la
"bonne" définition



Garanties plus fortes



Notions particulièrement
adaptées au modèle

Signatures

MPC

Garanties quand les
protocoles sont composés



Résultats d'impossibilité
sont rares



Exemple de définition
équivalentes

IND-CPA

Semantic-security

Sécurité basée sur les jeux

Le challenger modélise ce que l'adversaire est autorisé à faire et ce qui est considéré comme « mauvais » en termes de sécurité :

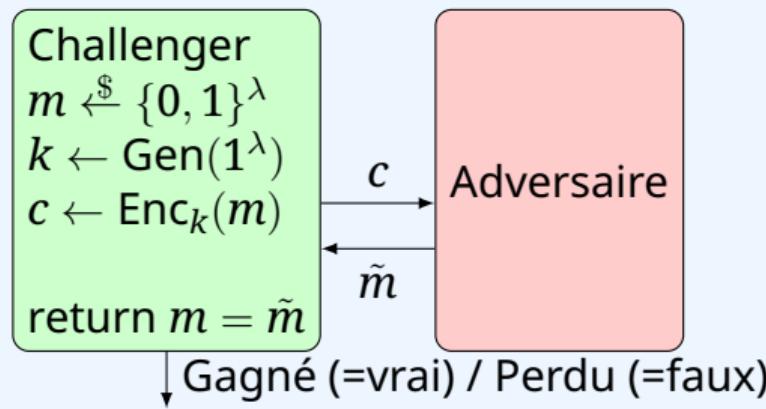
- Quels messages/fonctions l'adversaire peut-il lire/appeler ?
- Adversaire passif (= écouteur) ou actif (= man in the middle) ?
- Boîte noire ou accès physique à l'appareil ?
 - Attaques par canaux auxiliaires (= enregistrement de la consommation électrique, du bruit...)
 - Attaques par faute (ex. : envoi d'ondes magnétiques pour perturber un circuit...)
- Que doit-on garder secret ? (en fonction de la valeur de retour du challenger)

Questions

Ceci modélise :

- A un adversaire passif,
- B un adversaire actif ?

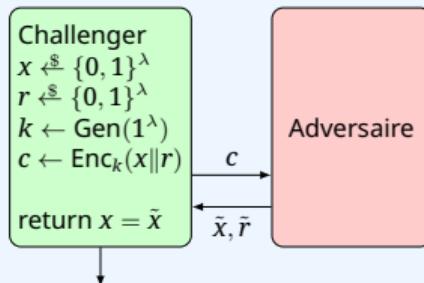
?



Questions

Supposons que pour tout adversaire \mathcal{A} , la probabilité de gagner ce jeu est négligeable. Traduire les implications en français :

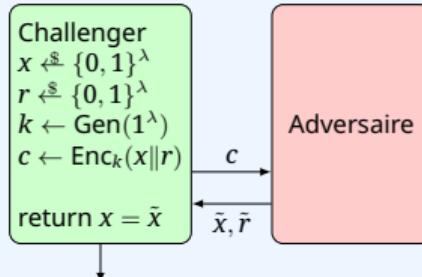
- A La probabilité de retrouver **entièrement** un message **aléatoire** à partir d'un chiffré est négligeable
- B La probabilité de retrouver la **première moitié** d'un message **aléatoire** à partir d'un chiffré est nulle
- C La probabilité de retrouver la **première moitié** de **n'importe quel** message à partir d'un chiffré est nulle



Questions

Supposons que pour tout adversaire \mathcal{A} , la probabilité de gagner ce jeu est négligeable. Traduire les implications en français :

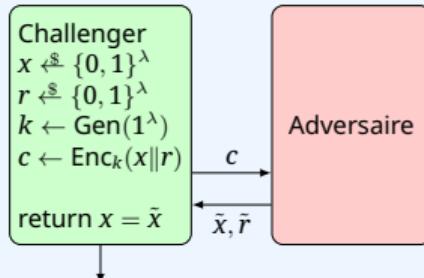
- A La probabilité de retrouver **entièrement** un message **aléatoire** à partir d'un chiffré est négligeable
- B La probabilité de retrouver la **première moitié** d'un message **aléatoire** à partir d'un chiffré est nulle ✓
- C La probabilité de retrouver la **première moitié** de **n'importe quel** message à partir d'un chiffré est nulle



Questions

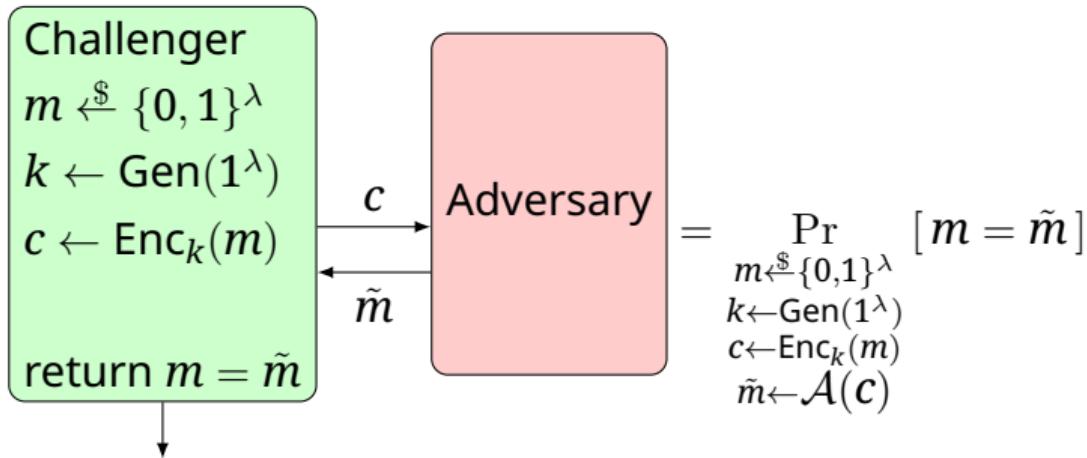
Supposons que pour tout adversaire \mathcal{A} , la probabilité de gagner ce jeu est négligeable. Traduire les implications en français :

- A La probabilité de retrouver **entièrement** un message **aléatoire** à partir d'un chiffré est négligeable
- B La probabilité de retrouver la **première moitié** d'un message **aléatoire** à partir d'un chiffré est nulle ✓
- C La probabilité de retrouver la **première moitié** de **n'importe quel** message à partir d'un chiffré est nulle

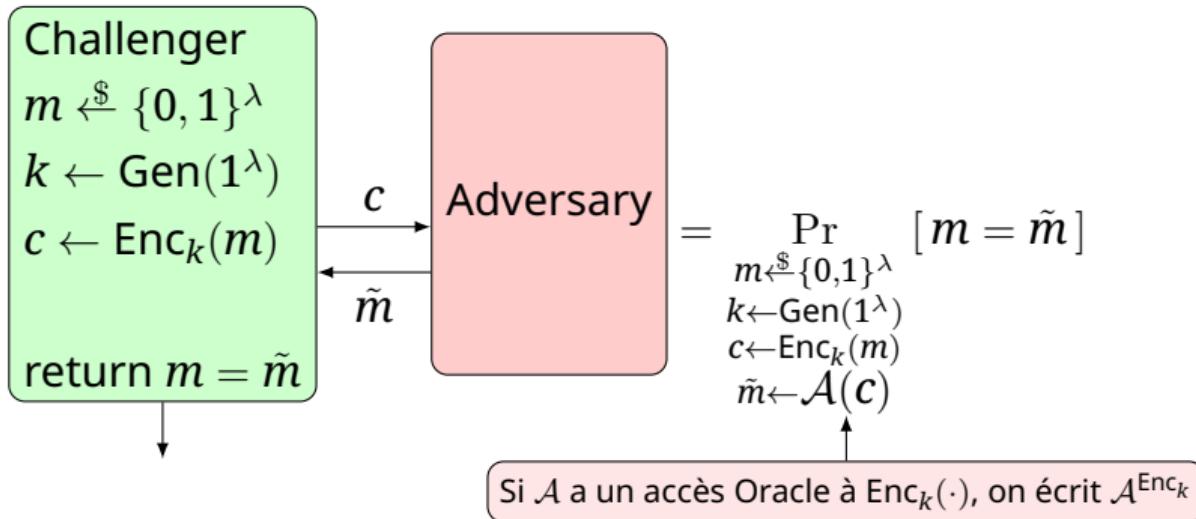


Il est donc peut-être facile de déchiffrer "Oui" et "Non" mais pas les autres messages!

Equivalent notations/formulations



Equivalent notations/formulations



Equivalent notations/formulations

Challenger

$$m \xleftarrow{\$} \{0,1\}^\lambda$$

$$k \leftarrow \text{Gen}(1^\lambda)$$

$$c \leftarrow \text{Enc}_k(m)$$

return $m = \tilde{m}$

```

graph LR
    C[Challenger] -- "c" --> A[Adversary]
    A -- "m-tilde" --> C
  
```

\mathcal{L}_m
$k \leftarrow \text{Gen}(1^\lambda)$
$c \leftarrow \text{Enc}_k(m)$
<u>GETC():</u>
return c

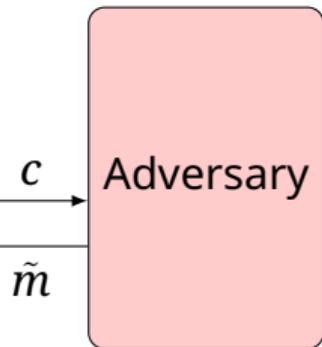
$$= \Pr_{\substack{m \xleftarrow{\$} \{0,1\}^\lambda \\ k \leftarrow \text{Gen}(1^\lambda) \\ c \leftarrow \text{Enc}_k(m) \\ \tilde{m} \leftarrow \mathcal{A}(c)}} [m = \tilde{m}] = \Pr_{\substack{m \xleftarrow{\$} \{0,1\}^\lambda \\ \tilde{m} \leftarrow \mathcal{A} \diamond \mathcal{L}_m}} [m = \tilde{m}]$$

$\mathcal{A} \diamond \mathcal{L}$ signifie que \mathcal{A} a un accès oracle à \mathcal{L} (= librairie), comme $\mathcal{A}^{\mathcal{L}}$ mais cette notation est utilisée dans *Joy of cryptography* et est pratique lorsque l'on chaine plusieurs librairies.

Equivalent notations/formulations

Challenger

```
 $m \xleftarrow{\$} \{0,1\}^\lambda$ 
 $k \leftarrow \text{Gen}(1^\lambda)$ 
 $c \leftarrow \text{Enc}_k(m)$ 
return  $m = \tilde{m}$ 
```



Verbeux, dure à manipuler
formellement

Plus standard mais souvent
difficile à manipuler et
vérifier formellement

\mathcal{L}_m

```
 $k \leftarrow \text{Gen}(1^\lambda)$ 
 $c \leftarrow \text{Enc}_k(m)$ 
GETC():
return  $c$ 
```

$$= \Pr_{\substack{m \xleftarrow{\$} \{0,1\}^\lambda \\ k \leftarrow \text{Gen}(1^\lambda) \\ c \leftarrow \text{Enc}_k(m) \\ \tilde{m} \leftarrow \mathcal{A}(c)}} [m = \tilde{m}] = \Pr_{\substack{m \xleftarrow{\$} \{0,1\}^\lambda \\ \tilde{m} \leftarrow \mathcal{A} \diamond \mathcal{L}_m}} [m = \tilde{m}]$$

Modèle de *Joy of cryptography*: **plus facile** à ré-utiliser et écrire/vérifier des preuves (dépendances explicites, petites réductions faciles à vérifier)

Mais **fondamentalement identique**, juste la présentation qui diffère !

Exercice évaluation de librairie

On considère les deux librairies suivantes :

\mathcal{A}_1
 $r_1 \leftarrow \text{RAND}(6)$
return $r_1 \stackrel{?}{=} 4$

\mathcal{L}_1
 $\text{RAND}(n):$
 $r \xleftarrow{\$} \mathbb{Z}_{[n/2]}$
return $2r$



Quelle est la valeur de $\Pr [\mathcal{A}_1 \diamond \mathcal{L}_1 = 1]^a$?

- A 0
- B 1/6
- C 1/3
- D 1

^aÀ partir de maintenant, on définit true = 1 et false = 0.

Exercice évaluation de librairie

On considère les deux librairies suivantes :

\mathcal{A}_1
 $r_1 \leftarrow \text{RAND}(6)$
return $r_1 \stackrel{?}{=} 4$

\mathcal{L}_1
 $\text{RAND}(n):$
 $r \xleftarrow{\$} \mathbb{Z}_{[n/2]}$
return $2r$



Quelle est la valeur de $\Pr [\mathcal{A}_1 \diamond \mathcal{L}_1 = 1]^a$?

- A 0
- B 1/6
- C 1/3 ✓
- D 1

^aÀ partir de maintenant, on définit true = 1 et false = 0.

Exercice des librairies bien définies



On considère les deux librairies suivantes : Quelle est la valeur de $\Pr[\mathcal{A}_1 \diamond \mathcal{L}_1 = 1]$?

- A 0
- B 1/2
- C 1
- D Ce n'est pas bien défini pour une raison
- E Ce n'est pas bien défini pour deux raisons

\mathcal{A}_1

```
a := 46
return sample()  $\stackrel{?}{=}$  c
```

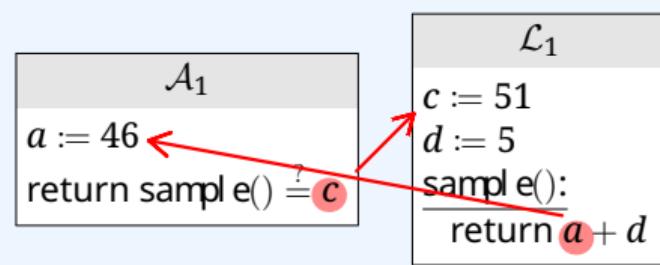
\mathcal{L}_1

```
c := 51
d := 5
sample():
    return a + d
```

Exercice des librairies bien définies

On considère les deux librairies suivantes : Quelle est la valeur de $\Pr[\mathcal{A}_1 \diamond \mathcal{L}_1 = 1]$?

- A 0
- B 1/2
- C 1
- D Ce n'est pas bien défini pour une raison
- E Ce n'est pas bien défini pour deux raisons ✓



Sécurité basée sur les jeux : puissance de l'adversaire

On peut aussi modéliser la puissance d'un adversaire (généralement modélisé comme une machine de Turing) dans la quantification de l'adversaire :

- "Pour tout adversaire \mathcal{A} **non borné**, la probabilité de gagner est faible" = sécurité statistique
- "Pour tout adversaire \mathcal{A} **borné polynomialement**, la probabilité de gagner est faible" = sécurité calculatoire

Si le temps d'exécution de $\mathcal{A}(n)$ est \sqrt{n} , \mathcal{A} est-elle polynomiale ?



- A Oui
- B Non

Sécurité basée sur les jeux : puissance de l'adversaire

On peut aussi modéliser la puissance d'un adversaire (généralement modélisé comme une machine de Turing) dans la quantification de l'adversaire :

- "Pour tout adversaire \mathcal{A} **non borné**, la probabilité de gagner est faible" = sécurité statistique
- "Pour tout adversaire \mathcal{A} **borné polynomialement**, la probabilité de gagner est faible" = sécurité calculatoire

Si le temps d'exécution de $\mathcal{A}(n)$ est \sqrt{n} , \mathcal{A} est-elle polynomiale ?



- A Oui
- B Non Elle doit s'exécuter en temps polynomial en la ($\log(n)$) de l'entrée (sinon, la factorisation devient efficace !)

Sécurité basée sur les jeux : puissance de l'adversaire

On peut aussi modéliser la puissance d'un adversaire (généralement modélisé comme une machine de Turing) dans la quantification de l'adversaire :

- "Pour tout adversaire \mathcal{A} **non borné**, la probabilité de gagner est faible" = sécurité statistique
- "Pour tout adversaire \mathcal{A} **borné polynomialement**, la probabilité de gagner est faible" = sécurité calculatoire



Si le temps d'exécution de $\mathcal{A}(1^\lambda)$ est λ^2 , \mathcal{A} est-elle polynomiale ?

- A Oui
- B Non

Sécurité basée sur les jeux : puissance de l'adversaire

On peut aussi modéliser la puissance d'un adversaire (généralement modélisé comme une machine de Turing) dans la quantification de l'adversaire :

- "Pour tout adversaire \mathcal{A} **non borné**, la probabilité de gagner est faible" = sécurité statistique
- "Pour tout adversaire \mathcal{A} **borné polynomialement**, la probabilité de gagner est faible" = sécurité calculatoire



Si le temps d'exécution de $\mathcal{A}(1^\lambda)$ est λ^2 , \mathcal{A} est-elle polynomiale ?

- A Oui car l'argument est spécifié en format unaire $1 \dots 1$
- B Non

Sécurité basée sur les jeux : puissance de l'adversaire

On peut aussi modéliser la puissance d'un adversaire (généralement modélisé comme une machine de Turing) dans la quantification : Qu'est-ce que « faible » ?

- "Pour tout adversaire \mathcal{A} **non borné**, la probabilité de gagner est faible" = sécurité statistique
- "Pour tout adversaire \mathcal{A} **borné polynomialement**, la probabilité de gagner est faible" = sécurité calculatoire



Si le temps d'exécution de $\mathcal{A}(1^\lambda)$ est λ^2 , \mathcal{A} est-elle polynomiale ?

- A Oui car l'argument est spécifié en format unaire 1...1
- B Non

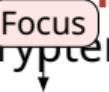
Recherche vs décision

Définition de “faible” = dépend du challenger, mais on distingue typiquement deux cas :

- **Problème de recherche** : l’adversaire doit retrouver une **chaîne de bits** (e.g. “décrypter ce message”) : faible = $\text{negl}(\lambda)$
- **Problème de décision** : l’adversaire doit retrouver un **bit unique** b (e.g. “est-ce un chiffré de m_0 ou m_1 ?”) : faible = $1/2 + \text{negl}(\lambda)$

Recherche vs décision

Définition de "faible" = dépend du challenger, mais on distingue typiquement deux cas :

- **Problème de recherche** : l'adversaire doit retrouver une **chaîne de bits** (e.g. "décrypter ce message") : faible = $\text{negl}(\lambda)$
A red rounded rectangle containing the word "Focus" with a small arrow pointing from it towards the word "décrypter" in the text above.
- **Problème de décision** : l'adversaire doit retrouver un **bit unique** b (e.g. "est-ce un chiffré de m_0 ou m_1 ?") : faible = $1/2 + \text{negl}(\lambda)$

Recherche vs décision

Définition de "faible" = dépend du challenger, mais on distingue typiquement deux cas :

- **Problème de recherche** : l'adversaire doit retrouver une **chaîne de bits** (e.g. "décrypter ce message") : faible = $\text{negl}(\lambda)$

- **Problème de décision** : l'adversaire doit retrouver un **bit unique** b (e.g. "est-ce un chiffré de m_0 ou m_1 ?") : faible = $1/2 + \text{negl}(\lambda)$

Définition (interchangeabilité)

Deux librairies \mathcal{L}_0 et \mathcal{L}_1 sont *interchangeables* (ou égales), noté $\mathcal{L}_0 \equiv \mathcal{L}_1$, si pour tout adversaire \mathcal{A} ,

$$\Pr [\mathcal{A} \diamond \mathcal{L}_0 = 1] = \Pr [\mathcal{A} \diamond \mathcal{L}_1 = 1]$$

Goal

Parfois, nous avons besoin d'une version plus relaxée ou l'adversaire est borné (temps polynomial) :

Définition (avantage et indistinguabilité)

Deux librairies \mathcal{L}_0 et \mathcal{L}_1 sont **indistinguables** (noté $\mathcal{L}_0 \approx \mathcal{L}_1$) si pour tout adversaire \mathcal{A} calculatoirement borné (temps polynomial), **l'avantage** $\text{Adv}_{\mathcal{A}}(\lambda)$ de \mathcal{A} est négligeable, avec:

$$\text{Adv}_{\mathcal{A}}(\lambda) := \left| \Pr \left[\mathcal{A}(1^\lambda) \diamond \mathcal{L}_0 = 1 \right] - \Pr \left[\mathcal{A}(1^\lambda) \diamond \mathcal{L}_1 = 1 \right] \right| \leq \text{negl}(\lambda)$$

Goal

Parfois, nous avons besoin d'une version plus relaxée où l'adversaire est borné (temps polynomial) :

Définition (avantage et indistinguabilité)

Deux librairies \mathcal{L}_0 et \mathcal{L}_1 sont **indistinguables** (noté $\mathcal{L}_0 \approx \mathcal{L}_1$) si pour tout adversaire \mathcal{A} calculatoirement borné (temps polynomial), **l'avantage** $\text{Adv}_{\mathcal{A}}(\lambda)$ de \mathcal{A} est négligeable, avec:

$$\text{Adv}_{\mathcal{A}}(\lambda) := \left| \Pr \left[\mathcal{A}(1^\lambda) \diamond \mathcal{L}_0 = 1 \right] - \Pr \left[\mathcal{A}(1^\lambda) \diamond \mathcal{L}_1 = 1 \right] \right| \leq \text{negl}(\lambda)$$

Notion **asymptotique!**

Nous avons enfin tous les outils pour définir la sécurité d'un chiffrement !



Antoine Daniel va pouvoir définir la sécurité d'un chiffrement

Definition (IND-CPA)

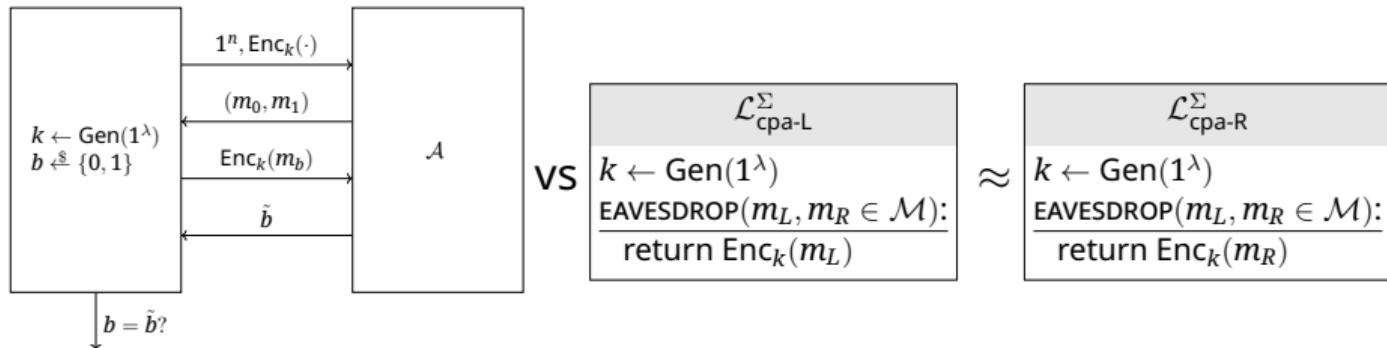
Un schéma de chiffrement $\Sigma = (\text{Gen}, \text{Enc}, \text{Dec})$ est indistinguorable contre des attaques à texte clair choisi (chosen-plaintext attack, **IND-CPA**) si:

$\mathcal{L}_{\text{cpa-L}}^{\Sigma}$
$k \leftarrow \text{Gen}(1^\lambda)$
$\text{EAVESDROP}(m_L, m_R \in \mathcal{M}):$ return $\text{Enc}_k(m_L)$

$\mathcal{L}_{\text{cpa-R}}^{\Sigma}$
$k \leftarrow \text{Gen}(1^\lambda)$
$\text{EAVESDROP}(m_L, m_R \in \mathcal{M}):$ return $\text{Enc}_k(m_R)$

Various definitions of IND-CPA

On rencontre parfois cette autre définition **équivalente** de IND-CPA :



- Au lieu d'un bit b , quand $b = 0$ on joue avec $\mathcal{L}_{\text{cpa-L}}^\Sigma$, sinon avec $\mathcal{L}_{\text{cpa-R}}^\Sigma$.
- Dans notre définition, pas d'accès à l'oracle $\text{Enc}_k(\cdot)$, mais on peut **le simuler** en appelant $\text{EAVESDROP}(m, m)$ (le même message deux fois).
- Dans notre définition, aucune restriction sur le nombre d'appels à EAVESDROP (= notion plus forte, tandis que dans l'autre on n'a qu'un seul message $\text{Enc}_k(m_b)$). Mais c'est équivalent (l'avantage est multiplié par le nombre maximal de requêtes faites par \mathcal{A} , mais reste négligeable) : preuve par une suite de **hybrides sur le nombre de requêtes**.

Comment prouver l'INsécurité ?

Comment prouver l'INsécurité

Pour prouver l'**insécurité** pour un jeu d'indistinguabilité entre \mathcal{L}_0 et \mathcal{L}_1 :

- ① fournir un attaquant \mathcal{A} donné
- ② calculer $\varepsilon = |\Pr[\mathcal{A} \diamond \mathcal{L}_0 = 1] - \Pr[\mathcal{A} \diamond \mathcal{L}_1 = 1]|$ (ε est toujours entre 0 et 1)
- ③ montrer qu'il existe $c \in \mathbb{N}$ tel que $\varepsilon > \frac{1}{\lambda^c}$

Comment prouver l'INsécurité

On considère le chiffrement $\text{Gen}(1^\lambda) := \text{return } 0$ et $\text{Enc}_k(m) := m \oplus \textcolor{red}{1} \dots \textcolor{red}{1}$. Ce schéma est-il IND-CPA secure, et si non, quel attaquant peut distinguer les deux librairies correspondantes, et avec quel avantage ?

$$\begin{array}{c|c} \mathcal{L}_{\text{cpa-L}}^{\Sigma} & \mathcal{L}_{\text{cpa-R}}^{\Sigma} \\ \hline k \leftarrow \text{Gen}(1^\lambda) & k \leftarrow \text{Gen}(1^\lambda) \\ \text{EAVESDROP}(m_L, m_R \in \mathcal{M}): & \text{EAVESDROP}(m_L, m_R \in \mathcal{M}): \\ \hline \text{return Enc}_k(m_{\textcolor{blue}{L}}) & \text{return Enc}_k(m_{\textcolor{blue}{R}}) \end{array} \approx ? \quad (1)$$



Comment prouver l'INsécurité

On considère le chiffrement $\text{Gen}(1^\lambda) := \text{return } 0$ et $\text{Enc}_k(m) := m \oplus \textcolor{red}{1} \dots \textcolor{red}{1}$. Ce schéma est-il IND-CPA secure, et si non, quel attaquant peut distinguer les deux librairies correspondantes, et avec quel avantage ?

$$\begin{array}{c|c} \mathcal{L}_{\text{cpa-L}}^{\Sigma} & \mathcal{L}_{\text{cpa-R}}^{\Sigma} \\ \hline k \leftarrow \text{Gen}(1^\lambda) & k \leftarrow \text{Gen}(1^\lambda) \\ \text{EAVESDROP}(m_L, m_R \in \mathcal{M}): & \text{EAVESDROP}(m_L, m_R \in \mathcal{M}): \\ \hline \text{return Enc}_k(m_{\textcolor{blue}{L}}) & \text{return Enc}_k(m_{\textcolor{blue}{R}}) \end{array} \approx ? \quad (1)$$

?

- 1 $\boxed{\begin{array}{c} \mathcal{A} \\ c := \text{EAVESDROP}(\textcolor{red}{0}^\lambda) \\ \text{return } c \oplus \textcolor{red}{1} \dots \textcolor{red}{1} \stackrel{?}{=} \textcolor{red}{0}^\lambda \end{array}}$, avantage 0 (A), 1/2 (B), $1/2 - \frac{1}{2^\lambda}$ (C) ou 1 (D)

- 2 $\boxed{\begin{array}{c} \mathcal{A} \\ c := \text{EAVESDROP}(\textcolor{red}{0}^\lambda) \\ \text{return } c \oplus c \stackrel{?}{=} \textcolor{red}{0}^\lambda \end{array}}$, avantage 0 (E), 1/2 (F), $1/2 - \frac{1}{2^\lambda}$ (G) ou $1 - \frac{1}{2^\lambda}$ (H)

Comment prouver l'INsécurité

On considère le chiffrement $\text{Gen}(1^\lambda) := \text{return } 0$ et $\text{Enc}_k(m) := m \oplus \textcolor{red}{1} \dots \textcolor{red}{1}$. Ce schéma est-il IND-CPA secure, et si non, quel attaquant peut distinguer les deux librairies correspondantes, et avec quel avantage ?

$$\begin{array}{c|c} \mathcal{L}_{\text{cpa-L}}^{\Sigma} & \mathcal{L}_{\text{cpa-R}}^{\Sigma} \\ \hline k \leftarrow \text{Gen}(1^\lambda) & k \leftarrow \text{Gen}(1^\lambda) \\ \text{EAVESDROP}(m_L, m_R \in \mathcal{M}): & \text{EAVESDROP}(m_L, m_R \in \mathcal{M}): \\ \hline \text{return Enc}_k(m_{\textcolor{blue}{L}}) & \text{return Enc}_k(m_{\textcolor{blue}{R}}) \end{array} \approx ? \quad (1)$$

?

- 1 \mathcal{A}
 $c := \text{EAVESDROP}(\textcolor{red}{0}^\lambda)$, avantage 0 (A), $1/2$ (B), $1/2 - \frac{1}{2^\lambda}$ (C) ou 1 (D ✓)
return $c \oplus \textcolor{red}{1} \dots \textcolor{red}{1} \stackrel{?}{=} \textcolor{red}{0}^\lambda$

- 2 \mathcal{A}
 $c := \text{EAVESDROP}(\textcolor{red}{0}^\lambda)$, avantage 0 (E), $1/2$ (F), $1/2 - \frac{1}{2^\lambda}$ (G) ou $1 - \frac{1}{2^\lambda}$ (H)
return $c \oplus c \stackrel{?}{=} \textcolor{red}{0}^\lambda$

Comment prouver l'INsécurité

Quel attaquant peut distinguer ces deux librairies, et avec quel avantage ?

$$\mathcal{L}_{\text{ots\$-real}}^{\Sigma}$$

$\text{CTXT}(m \in \{0, 1\}^{\lambda})$:

$$\frac{k \leftarrow \{0, 1\}^{\lambda} \quad // \Sigma.\text{KeyGen}}{c := k \& m \quad // \Sigma.\text{Enc}}$$

return c

$$\mathcal{L}_{\text{ots\$-rand}}^{\Sigma}$$

$\text{CTXT}(m \in \{0, 1\}^{\lambda})$:

$$\frac{c \leftarrow \{0, 1\}^{\lambda} \quad // \Sigma.C}{\text{return } c}$$


Comment prouver l'INsécurité

Quel attaquant peut distinguer ces deux librairies, et avec quel avantage ?

$$\mathcal{L}_{\text{ots\$-real}}^{\Sigma}$$

$\text{CTXT}(m \in \{0, 1\}^{\lambda})$:

$k \leftarrow \{0, 1\}^{\lambda}$ // $\Sigma.\text{KeyGen}$
 $c := k \& m$ // $\Sigma.\text{Enc}$
return c

$$\mathcal{L}_{\text{ots\$-rand}}^{\Sigma}$$

$\text{CTXT}(m \in \{0, 1\}^{\lambda})$:

$c \leftarrow \{0, 1\}^{\lambda}$ // $\Sigma.C$
return c



- 1 \mathcal{A}
 $c := \text{CTXT}(0^{\lambda})$, avantage $1/4$ (A), $1/2$ (B), $1/2 - \frac{1}{2^{\lambda}}$ (C) or $1 - \frac{1}{2^{\lambda}}$ (D)
return $c = 0^{\lambda}$

- 2 \mathcal{A}
 $c := \text{CTXT}(1^{\lambda})$, avantage $1/4$ (E), $1/2$ (F), $1/2 - \frac{1}{2^{\lambda}}$ (G) or $1 - \frac{1}{2^{\lambda}}$ (H)
return $c = 0^{\lambda}$

Comment prouver l'INsécurité

Quel attaquant peut distinguer ces deux librairies, et avec quel avantage ?

$$\mathcal{L}_{\text{ots\$-real}}^{\Sigma}$$

$\text{CTXT}(m \in \{0, 1\}^{\lambda})$:

$k \leftarrow \{0, 1\}^{\lambda}$ // $\Sigma.\text{KeyGen}$
 $c := k \& m$ // $\Sigma.\text{Enc}$
return c

$$\mathcal{L}_{\text{ots\$-rand}}^{\Sigma}$$

$\text{CTXT}(m \in \{0, 1\}^{\lambda})$:

$c \leftarrow \{0, 1\}^{\lambda}$ // $\Sigma.C$
return c



- 1 \mathcal{A}
 $c := \text{CTXT}(0^{\lambda})$, avantage $1/4$ (A), $1/2$ (B), $1/2 - \frac{1}{2^{\lambda}}$ (C) or $1 - \frac{1}{2^{\lambda}}$ (D ✓)

$$\mathcal{A}$$

- 2 \mathcal{A}
 $c := \text{CTXT}(1^{\lambda})$, avantage $1/4$ (E), $1/2$ (F), $1/2 - \frac{1}{2^{\lambda}}$ (G) or $1 - \frac{1}{2^{\lambda}}$ (H)

Pratiquer

Allez sur moodle faire les exercices (vous écrirez les distingueurs en python)

Sécurité asymptotique vs sécurité effective

En analyse théorique, la sécurité est asymptotique. En pratique : **comment choisir λ ?** Typiquement :

- A Étudier les meilleures attaques connues, **compter le nombre d'opérations T** et l'avantage ε (compromis temps/précision), et considérer que le nombre réel d'opérations est approximativement¹ T/ε .
⇒ ce protocole offre une sécurité de $\log(T/\varepsilon)$ bits.
- B Noter que :
 - 2^{40} opérations sont très faciles à faire (petit cluster de Raspberry Pi)
 - 2^{60} opérations sont faisables avec un gros cluster CPU/GPU
 - 2^{80} opérations sont faisables avec un cluster d'ASICs (minage de bitcoin)
 - 2^{128} opérations = **très difficile** (diapo suivante)

¹Plus de détails dans [Watanabe, Yasunaga 2021] et [Micciancio, Walter 2018].

Quelle est la taille de 2^{128} ?

Supposons que :

- le problème est parallélisable
- on a accès aux 500 meilleurs superordinateurs = 10 000 000 000 GFLOPS
(FLOPS = opérations flottantes par seconde)

Alors, il faudrait au total :

$$\frac{2^{128}}{10 \times 10^9 \times 10^9 \times 3600 \times 24 \times 365} \approx \boxed{1\,000\,000\,000\,000 \text{ années}}$$

(environ 4× l'âge de la Terre)

Comment écrire des preuves de sécurité

Propriétés de base

Propriétés (toujours vraies en remplaçant \approx par \equiv)

- **Transitivity:** $(\mathcal{L}_0 \approx \mathcal{L}_1) \wedge (\mathcal{L}_1 \approx \mathcal{L}_2) \Rightarrow \mathcal{L}_0 \approx \mathcal{L}_2$
- **Chainage:** $(\mathcal{L}_0 \approx \mathcal{L}_1) \Rightarrow ((\mathcal{L} \diamond \mathcal{L}_0) \approx (\mathcal{L} \diamond \mathcal{L}_1))$

Preuves: exercice

Propriétés de base

Propriétés (toujours vraies en remplaçant \approx par \equiv)

- **Transitivity:** $(\mathcal{L}_0 \approx \mathcal{L}_1) \wedge (\mathcal{L}_1 \approx \mathcal{L}_2) \Rightarrow \mathcal{L}_0 \approx \mathcal{L}_2$
- **Chainage:** $(\mathcal{L}_0 \approx \mathcal{L}_1) \Rightarrow ((\mathcal{L} \diamond \mathcal{L}_0) \approx (\mathcal{L} \diamond \mathcal{L}_1))$

Preuve transitivité (inégalité triangulaire): On suppose que $\mathcal{L}_0 \approx \mathcal{L}_1 \wedge \mathcal{L}_1 \approx \mathcal{L}_2$. Soit \mathcal{A} un adversaire polynomial. Alors par définition:

$$|\Pr[\mathcal{A} \diamond \mathcal{L}_0 = 1] - \Pr[\mathcal{A} \diamond \mathcal{L}_1 = 1]| \leq \text{negl}(\lambda) \wedge |\Pr[\mathcal{A} \diamond \mathcal{L}_1 = 1] - \Pr[\mathcal{A} \diamond \mathcal{L}_2 = 1]| \leq \text{negl}(\lambda)$$

Mais

$$\begin{aligned} & |\Pr[\mathcal{A} \diamond \mathcal{L}_0 = 1] - \Pr[\mathcal{A} \diamond \mathcal{L}_1 = 1]| \\ &= |\Pr[\mathcal{A} \diamond \mathcal{L}_0 = 1] - \Pr[\mathcal{A} \diamond \mathcal{L}_1 = 1] + \Pr[\mathcal{A} \diamond \mathcal{L}_1 = 1] - \Pr[\mathcal{A} \diamond \mathcal{L}_2 = 1]| \\ &\leq |\Pr[\mathcal{A} \diamond \mathcal{L}_0 = 1] - \Pr[\mathcal{A} \diamond \mathcal{L}_1 = 1]| + |\Pr[\mathcal{A} \diamond \mathcal{L}_1 = 1] - \Pr[\mathcal{A} \diamond \mathcal{L}_2 = 1]| \\ &\leq \text{negl}(\lambda) + \text{negl}(\lambda) \leq \text{negl}(\lambda) \end{aligned}$$

Propriétés de base

Propriétés (toujours vraies en remplaçant \approx par \equiv)

- **Transitivity:** $(\mathcal{L}_0 \approx \mathcal{L}_1) \wedge (\mathcal{L}_1 \approx \mathcal{L}_2) \Rightarrow \mathcal{L}_0 \approx \mathcal{L}_2$
- **Chainage:** $(\mathcal{L}_0 \approx \mathcal{L}_1) \Rightarrow ((\mathcal{L} \diamond \mathcal{L}_0) \approx (\mathcal{L} \diamond \mathcal{L}_1))$

Preuve chainage : On suppose que $\mathcal{L}_0 \approx \mathcal{L}_1$. soit \mathcal{A} un adversaire polynomial. Nous voulons montrer que $(\mathcal{L} \diamond \mathcal{L}_0) \approx (\mathcal{L} \diamond \mathcal{L}_1)$:

$$\begin{aligned} & |\Pr[\mathcal{A} \diamond (\mathcal{L} \diamond \mathcal{L}_0) = 1] - \Pr[\mathcal{A} \diamond (\mathcal{L} \diamond \mathcal{L}_2) = 1]| \\ \boxed{\mathcal{A}' := \mathcal{A} \diamond \mathcal{L}} \quad & \stackrel{=} {|\Pr[(\mathcal{A} \diamond \mathcal{L}) \diamond \mathcal{L}_0 = 1] - \Pr[(\mathcal{A} \diamond \mathcal{L}) \diamond \mathcal{L}_1 = 1]|} \\ & \stackrel{=}{|\Pr[\mathcal{A}' \diamond \mathcal{L}_0 = 1] - \Pr[\mathcal{A}' \diamond \mathcal{L}_1 = 1]|} \end{aligned}$$

puisque \mathcal{A} est polynomial, c'est aussi le cas de \mathcal{A}' . Par conséquent, on utilisant $\mathcal{L}_0 \approx \mathcal{L}_1$ l'expression ci-dessus est $\text{negl}(\lambda)$. □

Six méthodes principales :

- ① **Jeux hybrides** : Décomposer en une suite de jeux hybrides (pour faciliter les méthodes 2 à 6)
- ② **Probabilités** : Calculer explicitement les probabilités, et montrer l'égalité ou borner la distance statistique (valable uniquement pour la sécurité statistique)
- ③ **Égalité** : Montrer que les deux jeux font exactement la même chose (cas particulier de la méthode 2)
(par exemple : code simplement externalisé dans une sous-bibliothèque, code simplement “inlined”...)
- ④ **Réduction** : montrer que si on peut les distinguer, alors \mathcal{A} peut être utilisé pour casser un problème difficile (factorisation de nombres...)
- ⑤ **Théorème/hypothèse** : utiliser théorème vu en cours ou une hypothèse
- ⑥ **Chaînage** : montrer $\mathcal{L}_1 \approx \mathcal{L}_2$, puis $\mathcal{A} \diamond \mathcal{L}_1 \approx \mathcal{A} \diamond \mathcal{L}_2$

Nous détaillons maintenant les méthodes 1, 2, 3 et 4 (5 et 6 sont triviales).

Jeux hybrides

Preuve = séquence de jeux **hybrides**



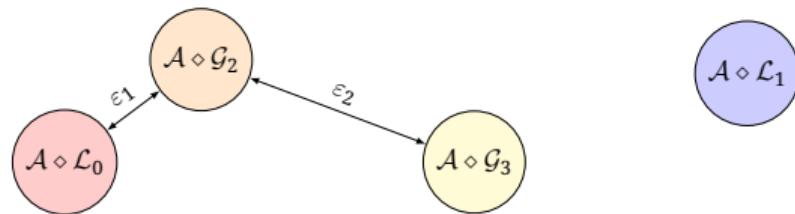
Jeux hybrides

Preuve = séquence de jeux **hybrides**



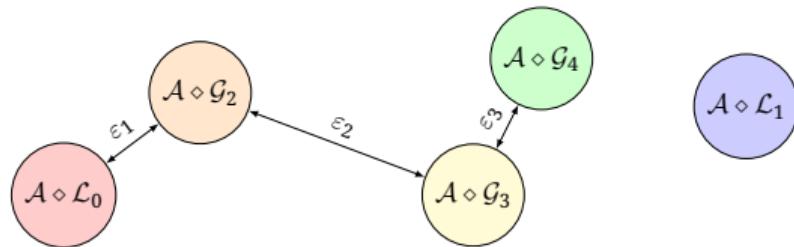
Jeux hybrides

Preuve = séquence de jeux **hybrides**



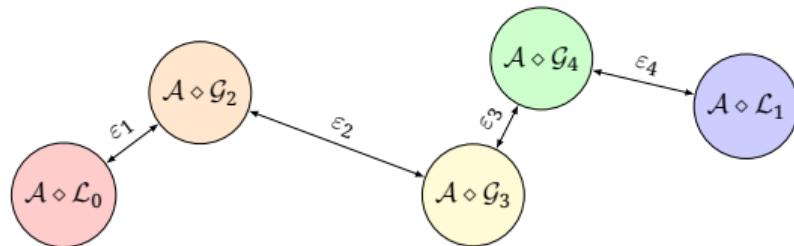
Jeux hybrides

Preuve = séquence de jeux **hybrides**



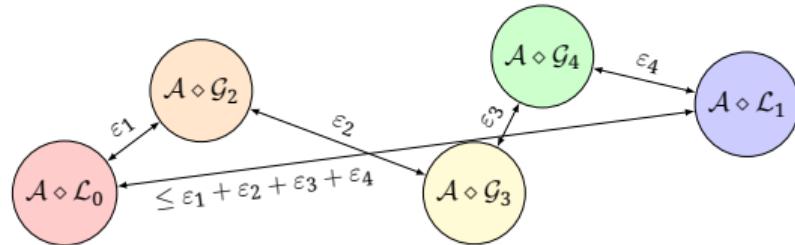
Jeux hybrides

Preuve = séquence de jeux **hybrides**



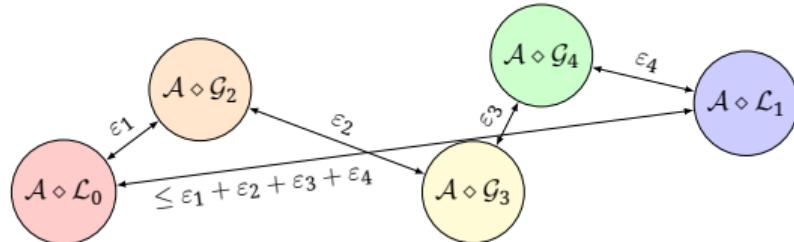
Jeux hybrides

Preuve = séquence de jeux **hybrides**



Jeux hybrides

Preuve = séquence de jeux **hybrides**



Par transitivité, if $\mathcal{L}_0 \approx \mathcal{G}_2 \approx \mathcal{G}_3 \approx \mathcal{G}_4 \approx \mathcal{L}_1$, then $\mathcal{L}_0 \approx \mathcal{L}_1$.

Égalité

On remarque simplement que deux bibliothèques font trivialement **exactement la même chose** (par exemple : externalizer un appel dans une sous-bibliothèque ou intégrer une sous-bibliothèque directement dans le code = “**inline**” à écrire dans les preuves)

ATTENTION: Assurez-vous que les variables sont toujours bien définies, sans collision de noms et bien **portées** (une sous-bibliothèque ne peut pas faire référence à une variable de la bibliothèque parente)

Égalité

Est-ce que ces deux librairies sont égales ?

?

$\text{CTXT}(m):$
 $k_1 \leftarrow \{0, 1\}^\lambda$
 $c_1 := k_1 \oplus m$
 $c_2 := \text{CTXT}'(c_1)$
return c_2

$\mathcal{L}_{\text{otp-rand}}$
 $\text{CTXT}'(m'):$
 $c \leftarrow \{0, 1\}^\lambda$
return c

$\text{CTXT}(m):$
 $k_1 \leftarrow \{0, 1\}^\lambda$
 $c_1 := k_1 \oplus m$
 $c_2 \leftarrow \{0, 1\}^\lambda$
return c_2

- A Oui
- B Non

Égalité

Est-ce que ces deux librairies sont égales ?

?

$\frac{\text{CTXT}(m):}{k_1 \leftarrow \{0, 1\}^\lambda}$
 $c_1 := k_1 \oplus m$
 $c_2 := \text{CTXT}'(c_1)$
return c_2

$\diamond \frac{\mathcal{L}_{\text{otp-rand}}}{\frac{\text{CTXT}'(m')}{c \leftarrow \{0, 1\}^\lambda}}$
return c

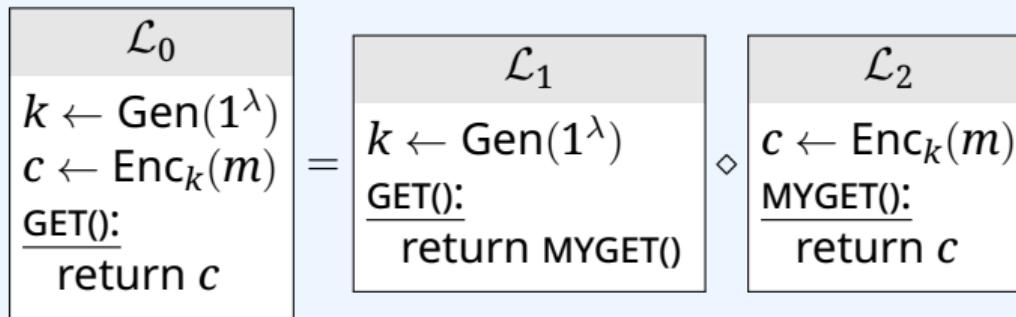
\equiv
 $\frac{\text{CTXT}(m):}{k_1 \leftarrow \{0, 1\}^\lambda}$
 $c_1 := k_1 \oplus m$
 $c_2 \leftarrow \{0, 1\}^\lambda$
return c_2

- A Oui Les variables sont bien définies dans leur scope, on a incliné une librairie
- B Non

Égalité

Est-ce que ces deux librairies sont égales ?

?



- A Oui
- B Non

Égalité

Est-ce que ces deux librairies sont égales ?

?

\mathcal{L}_0	$=$	\mathcal{L}_1	\diamond	\mathcal{L}_2
$k \leftarrow \text{Gen}(1^\lambda)$ $c \leftarrow \text{Enc}_k(m)$ <u>GET():</u> return c	$=$	$k \leftarrow \text{Gen}(1^\lambda)$ <u>GET():</u> return MYGET()	\diamond	$c \leftarrow \text{Enc}_k(m)$ <u>MYGET():</u> return c

- A Oui X
- B Non ✓ k n'est pas défini dans \mathcal{L}_2

Égalité

Est-ce que ces deux librairies sont égales ?

?

$$\begin{array}{c} \mathcal{L}_0 \\ k \leftarrow \text{Gen}(1^\lambda) \\ \underline{\text{GET()}:} \\ \text{return } 42 \end{array} \equiv \begin{array}{c} \mathcal{L}_1 \\ \underline{\text{GET()}:} \\ \text{return } 42 \end{array}$$

A

B

Égalité

Est-ce que ces deux librairies sont égales ?

?

$$\begin{array}{|c|} \hline \mathcal{L}_0 \\ \hline k \leftarrow \text{Gen}(1^\lambda) \\ \text{GET():} \\ \hline \text{return } 42 \\ \hline \end{array} \equiv \begin{array}{|c|} \hline \mathcal{L}_1 \\ \hline \text{GET():} \\ \hline \text{return } 42 \\ \hline \end{array}$$

- A k n'est jamais utilisé, on peut donc l'enlever
- B

Méthode : calcul de probabilités

Théorème (One-Time-Pad à chiffré uniforme)

$$\frac{\mathcal{L}_{\text{otp-real}}}{\begin{aligned} \text{OTENC}(m \in \{0, 1\}^\lambda) : \\ k \xleftarrow{\$} \{0, 1\}^\lambda \\ \text{return } k \oplus m \end{aligned}} = \frac{\mathcal{L}_{\text{otp-rand}}}{\begin{aligned} \text{OTENC}(m \in \{0, 1\}^\lambda) : \\ c \xleftarrow{\$} \{0, 1\}^\lambda \\ \text{return } c \end{aligned}}$$

Preuve Soit $m, \tilde{c} \in \{0, 1\}^\lambda$. Dans $\mathcal{L}_{\text{otp-rand}}$, $\Pr[\text{OTENC}(m) = \tilde{c}] = \frac{1}{2^\lambda}$ (tirage uniforme). Dans $\mathcal{L}_{\text{otp-real}}$:

$$\begin{aligned} \Pr[\text{OTENC}(m) = \tilde{c}] &= \Pr[k \oplus m = \tilde{c} \mid k \xleftarrow{\$} \{0, 1\}^\lambda] = \Pr[\tilde{c} \oplus m = k \mid k \xleftarrow{\$} \{0, 1\}^\lambda] \\ &= \Pr[C = k \mid k \xleftarrow{\$} \{0, 1\}^\lambda] = \frac{1}{2^\lambda} = \Pr[\text{OTENC}(m) = \tilde{c}] \end{aligned}$$

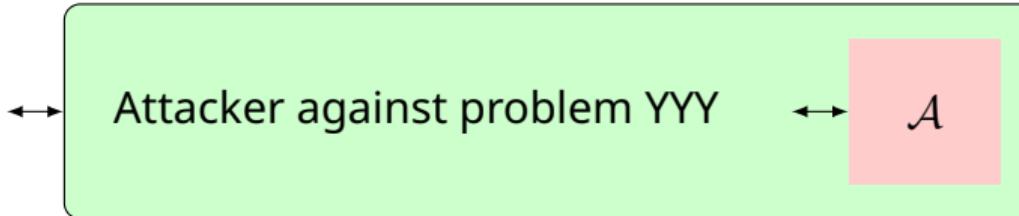
où $C := \tilde{c} \oplus m$. Donc $\mathcal{L}_{\text{otp-real}} = \mathcal{L}_{\text{otp-rand}}$



Méthode : réduction

Toutes les méthodes précédentes correspondent à l'interchangeabilité (indistinguabilité statistique). Mais qu'en est-il de l'indistinguabilité **computационnelle**? Soit on suppose directement que les deux bibliothèques sont difficiles à distinguer (il peut être nécessaire d'introduire une séquence hybride), soit :

Réduction !

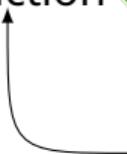


Idée: pour prouver que $\mathcal{L}_0 \approx \mathcal{L}_1$, supposons que $\mathcal{L}_0 \not\approx \mathcal{L}_1$, c'est-à-dire qu'il existe un adversaire polynomial \mathcal{A} tel que $|\Pr[\mathcal{A} \diamond \mathcal{L}_0 = 1] - \Pr[\mathcal{A} \diamond \mathcal{L}_1 = 1]|$.
On utilise \mathcal{A} comme sous-routine pour casser un problème difficile (on calcule explicitement la probabilité de succès) \Rightarrow contradiction !

Method: reduction

Option 1: single huge reduction:  hard to write and read

Option 2: hybrids + small reduction  Easier to read and verify



Often not even needed if the assumptions are already expressed as indistinguishable libraries

Méthode : réduction

Option 1 : une seule grande réduction : **X** difficile à écrire et à lire

Option 2 : jeux hybrides + petites réductions **✓** plus facile à lire et à vérifier

Souvent même pas nécessaires si les hypothèses sont déjà exprimées comme des bibliothèques indistinguables

Quelques théorèmes pratiques

Lemme des mauvais événements

Lemme des mauvais événements

Soit $\mathcal{L}_{\text{left}}$ et $\mathcal{L}_{\text{right}}$ deux librairies qui définissent une variable nommée bad, initialisée à 0. Si $\mathcal{L}_{\text{left}}$ et $\mathcal{L}_{\text{right}}$ ont des codes identiques sauf pour des parties atteignables seulement quand $\text{bad} = 1$ (e.g. à cause d'un "if $\text{bad} = 1$ "), alors:

$$|\Pr[\mathcal{A} \diamond \mathcal{L}_{\text{left}} = 1] - \Pr[\mathcal{A} \diamond \mathcal{L}_{\text{right}} = 1]| \leq \Pr[\mathcal{A} \diamond \mathcal{L}_{\text{left}} \text{ sets bad} = 1] \quad (2)$$

Preuve: Soit A_{left} l'événement " $\mathcal{A} \diamond \mathcal{L}_{\text{left}} = 1$ ", A_{right} l'événement " $\mathcal{A} \diamond \mathcal{L}_{\text{right}} = 1$ ", B_{left} l'événement $\mathcal{A} \diamond \mathcal{L}_{\text{left}} \text{ sets bad} = 1$, et B_{right} l'événement $\mathcal{A} \diamond \mathcal{L}_{\text{right}} \text{ sets bad} = 1$, et \cdot la négation de l'événement \cdot .

$$\begin{aligned} |\Pr[A_{\text{left}}] - \Pr[A_{\text{right}}]| &= |\Pr[B_{\text{left}}] \Pr[A_{\text{left}} | B_{\text{left}}] + \Pr[\bar{B}_{\text{left}}] \Pr[A_{\text{left}} | \bar{B}_{\text{left}}] \\ &\quad - \Pr[B_{\text{right}}] \Pr[A_{\text{right}} | B_{\text{right}}] - \Pr[\bar{B}_{\text{right}}] \Pr[A_{\text{right}} | \bar{B}_{\text{right}}]| \\ &\leq \Pr[\bar{B}_{\text{left}}] \underbrace{|\Pr[A_{\text{left}} | \bar{B}_{\text{left}}] - \Pr[A_{\text{right}} | \bar{B}_{\text{right}}]|}_{=0 \text{ (same code when bad is 0)}} + \Pr[B_{\text{left}}] \underbrace{|\Pr[A_{\text{left}} | B_{\text{left}}] - \Pr[A_{\text{right}} | B_{\text{right}}]|}_{\leq 1} \\ &\leq \Pr[B_{\text{left}}] \end{aligned}$$

Triangle ineq. & $\Pr[B_{\text{left}}] = \Pr[B_{\text{right}}]$ (code identique avant de modifier bad)



Application lemme des mauvais événements

Nous voulons montrer que

```
 $\mathcal{L}_{\text{left}}$ 
PREDICT( $x$ ):
 $s \xleftarrow{\$} \{0, 1\}^\lambda$ 
return  $x \stackrel{?}{=} s$ 
```

\approx

```
 $\mathcal{L}_{\text{right}}$ 
PREDICT( $x$ ):
return false
```

ces deux jeux hybrides:

et

```
 $\mathcal{G}_1$ 
bad := 0
PREDICT( $x$ ):
 $s \xleftarrow{\$} \{0, 1\}^\lambda$ 
if  $x \stackrel{?}{=} s$ :
    bad := 1
return false
```

```
 $\mathcal{G}_2$ 
bad := 0
PREDICT( $x$ ):
 $s \xleftarrow{\$} \{0, 1\}^\lambda$ 
if  $x \stackrel{?}{=} s$ :
    bad := 1
    return true
return false
```

. Un étudiant a déjà écrit

. Comment finir la preuve ?



- A $\mathcal{L}_{\text{left}} = \mathcal{G}_1 \approx \mathcal{G}_2 = \mathcal{L}_{\text{right}}$
- B $\mathcal{L}_{\text{left}} \approx \mathcal{G}_1 = \mathcal{G}_2 \approx \mathcal{L}_{\text{right}}$
- C $\mathcal{L}_{\text{left}} = \mathcal{G}_2 \approx \mathcal{G}_1 = \mathcal{L}_{\text{right}}$
- D $\mathcal{L}_{\text{left}} \approx \mathcal{G}_2 = \mathcal{G}_1 \approx \mathcal{L}_{\text{right}}$

Application lemme des mauvais événements

Nous voulons montrer que

$\mathcal{L}_{\text{left}}$
PREDICT(x):
 $s \xleftarrow{\$} \{0, 1\}^\lambda$
return $x \stackrel{?}{=} s$

$\approx \mathcal{L}_{\text{right}}$
PREDICT(x):
return false

ces deux jeux hybrides:

\mathcal{G}_1
bad := 0
PREDICT(x):
 $s \xleftarrow{\$} \{0, 1\}^\lambda$
if $x \stackrel{?}{=} s$:
 bad := 1
return false

et
 \mathcal{G}_2
bad := 0
PREDICT(x):
 $s \xleftarrow{\$} \{0, 1\}^\lambda$
if $x \stackrel{?}{=} s$:
 bad := 1
 return true
 return false

. Un étudiant a déjà écrit

. Comment finir la preuve ?



- A $\mathcal{L}_{\text{left}} = \mathcal{G}_1 \approx \mathcal{G}_2 = \mathcal{L}_{\text{right}}$
- B $\mathcal{L}_{\text{left}} \approx \mathcal{G}_1 = \mathcal{G}_2 \approx \mathcal{L}_{\text{right}}$
- C $\mathcal{L}_{\text{left}} = \mathcal{G}_2 \approx \mathcal{G}_1 = \mathcal{L}_{\text{right}}$ ✓ On utilise le lemme des mauvais événements pour montrer $\mathcal{G}_2 \approx \mathcal{G}_1$ ($\Pr[\text{bad} = 1] = \frac{1}{2^\lambda} = \text{negl}(\lambda)$)
- D $\mathcal{L}_{\text{left}} \approx \mathcal{G}_2 = \mathcal{G}_1 \approx \mathcal{L}_{\text{right}}$

Conclusion

Conclusion

- Cryptographie indissociable des **modèles** de sécurité et **preuves**
- **Beaucoup de paramètres** à considérer (borné ou non, hypothèses calculatoires, hypothèses setup, asymptotique ou non, modèle de sécurité...)
- Prouver la sécurité d'un protocole revient à **montrer que deux librairies sont indistinguables**
- Un exemple est la propriété de sécurité **IND-CPA**
- Nous avons vu un ensemble de **méthodes pour faire des preuves de sécurité**
- À l'inverse, **montrer l'insécurité** revient à exhiber un distingueur efficace (temps polynomial) qui arrive à distinguer les deux librairies avec un avantage non négligeable.