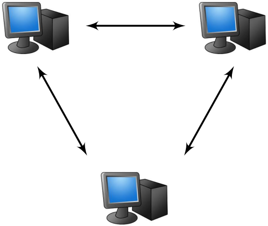
Les preuves de sécurité en cryptographie

Yannick Seurin

ANSSI

7 avril 2015 — Université de Cergy-Pontoise

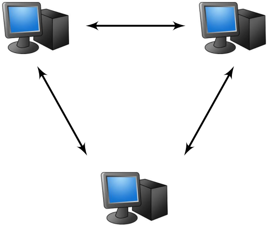
Introduction



Un cryptosystème

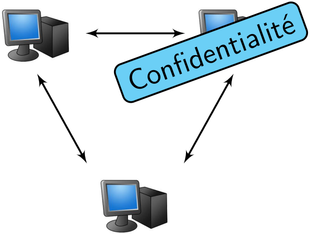
Introduction

Un attaquant



Un cryptosystème

Introduction

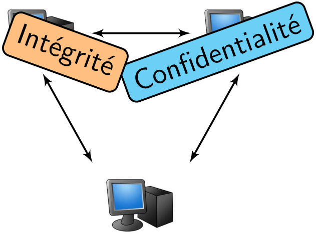


Un attaquant

Un cryptosystème

Introduction

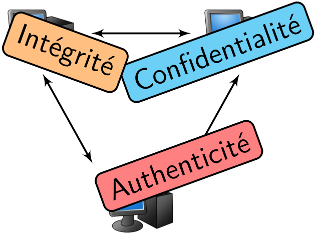
Un attaquant



Un cryptosystème

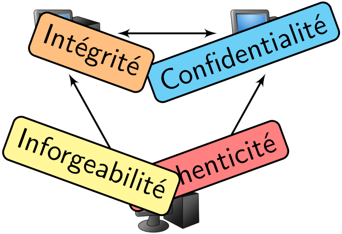
Introduction

Un attaquant



Un cryptosystème

Introduction

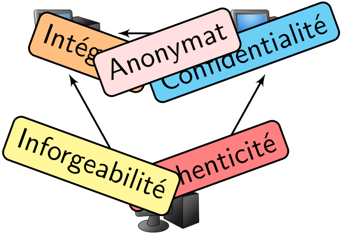


Un attaquant

Un cryptosystème

Introduction

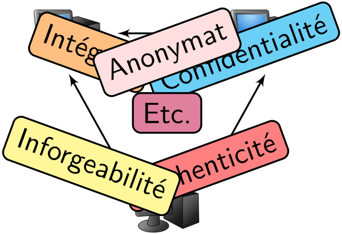
Un attaquant



Un cryptosystème

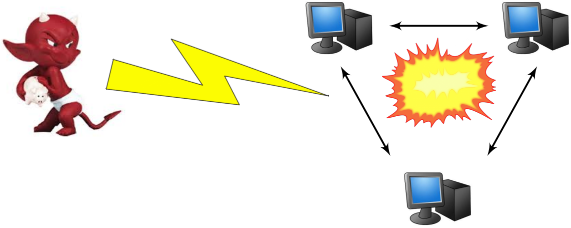
Introduction

Un attaquant



Un cryptosystème

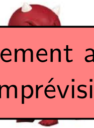
Introduction



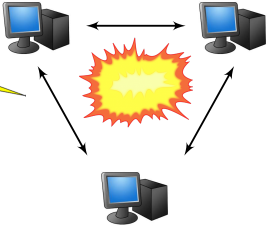
Un attaquant

Un cryptosystème

Introduction



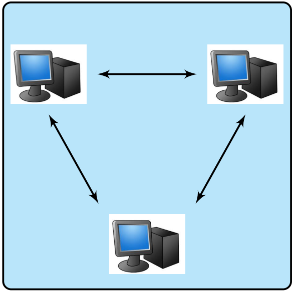
Comportement arbitraire et imprévisible



Un attaquant

Un cryptosystème

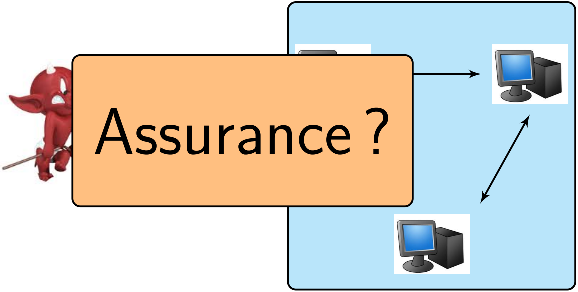
Introduction



Un attaquant
inoffensif

Un cryptosystème
sûr

Introduction



Un attaquant
inoffensif

Un cryptosystème
sûr

Plan

Plan

Généralités

Sécurité inconditionnelle (informationnelle)

Sécurité calculatoire (réductionniste)

Qu'est-ce qu'une preuve de sécurité ?

« *A proof is whatever convinces me.* »

Shimon Even, répondant à la question d'un étudiant

Comment (se) convaincre qu'un cryptosystème est sûr ?

1. absence d'attaques connues :
    - méthode itérative par « essais et erreurs » peu fiable
    - un cryptosystème reçoit peu d'attention tant qu'il n'est pas largement utilisé
    - soupçons de faiblesses connues mais non divulguées
2. argument mathématique :
    - formel : hypothèses $\Rightarrow$ conclusions
    - hypothèses = modèle de sécurité (+ problème difficile)
    - conclusion = attaque impossible

Qu'est-ce qu'une preuve de sécurité ?

« *A proof is whatever convinces me.* »

Shimon Even, répondant à la question d'un étudiant

Comment (se) convaincre qu'un cryptosystème est sûr ?

1. absence d'attaques connues :
    - méthode itérative par « essais et erreurs » peu fiable
    - un cryptosystème reçoit peu d'attention tant qu'il n'est pas largement utilisé
    - soupçons de faiblesses connues mais non divulguées
2. argument mathématique :
    - formel : hypothèses $\Rightarrow$ conclusions
    - hypothèses = modèle de sécurité (+ problème difficile)
    - conclusion = attaque impossible

Qu'est-ce qu'une preuve de sécurité ?

« *A proof is whatever convinces me.* »

Shimon Even, répondant à la question d'un étudiant

Comment (se) convaincre qu'un cryptosystème est sûr ?

1. absence d'attaques connues :
   - méthode itérative par « essais et erreurs » peu fiable
   - un cryptosystème reçoit peu d'attention tant qu'il n'est pas largement utilisé
   - soupçons de faiblesses connues mais non divulguées
2. argument mathématique :
   - formel : hypothèses $\Rightarrow$ conclusions
   - hypothèses $=$ modèle de sécurité ($+$ problème difficile)
   - conclusion $=$ attaque impossible

Les ingrédients d'une preuve de sécurité

Formaliser :

1. le cryptosystème
   - spécifier les fonctions qui le composent, leur cadre d'utilisation normal, la gestion des clés, etc.
2. l'objectif de sécurité ($\simeq$ opposé du but de l'attaquant) :
   - propriété que l'on cherche à garantir (confidentialité et/ou authenticité d'un message, inforgeabilité d'une signature, etc.)
3. le modèle d'attaque (ressources de l'attaquant) :
   - puissance de calcul, informations disponibles (souvent modélisées par des oracles), etc.
4. éventuellement, l'hypothèse de complexité sur laquelle va reposer la sécurité du cryptosystème
   - factorisation, log discret, résistance aux collisions (hachage), etc.

Les ingrédients d'une preuve de sécurité

Formaliser :

1. le cryptosystème
   - spécifier les fonctions qui le composent, leur cadre d'utilisation normal, la gestion des clés, etc.
2. l'objectif de sécurité ($\simeq$ opposé du but de l'attaquant) :
   - propriété que l'on cherche à garantir (confidentialité et/ou authenticité d'un message, inforgeabilité d'une signature, etc.)
3. le modèle d'attaque (ressources de l'attaquant) :
   - puissance de calcul, informations disponibles (souvent modélisées par des oracles), etc.
4. éventuellement, l'hypothèse de complexité sur laquelle va reposer la sécurité du cryptosystème
   - factorisation, log discret, résistance aux collisions (hachage), etc.

Les ingrédients d'une preuve de sécurité

Formaliser :

1. le cryptosystème
   - spécifier les fonctions qui le composent, leur cadre d'utilisation normal, la gestion des clés, etc.
2. l'objectif de sécurité ($\simeq$ opposé du but de l'attaquant) :
   - propriété que l'on cherche à garantir (confidentialité et/ou authenticité d'un message, inforgeabilité d'une signature, etc.)
3. le modèle d'attaque (ressources de l'attaquant) :
   - puissance de calcul, informations disponibles (souvent modélisées par des oracles), etc.
4. éventuellement, l'hypothèse de complexité sur laquelle va reposer la sécurité du cryptosystème
   - factorisation, log discret, résistance aux collisions (hachage), etc.

Les ingrédients d'une preuve de sécurité

Formaliser :

1. le cryptosystème
   - spécifier les fonctions qui le composent, leur cadre d'utilisation normal, la gestion des clés, etc.
2. l'objectif de sécurité ($\simeq$ opposé du but de l'attaquant) :
   - propriété que l'on cherche à garantir (confidentialité et/ou authenticité d'un message, inforgeabilité d'une signature, etc.)
3. le modèle d'attaque (ressources de l'attaquant) :
   - puissance de calcul, informations disponibles (souvent modélisées par des oracles), etc.
4. éventuellement, l'hypothèse de complexité sur laquelle va reposer la sécurité du cryptosystème
   - factorisation, log discret, résistance aux collisions (hachage), etc.

Deux types de preuves de sécurité

1. sécurité inconditionnelle :
    - théorie de l'information (puissance de calcul de l'attaquant ∞)
    - ☺ garantie de sécurité forte
    - ☹ très inefficace (voire impossible)
2. sécurité calculatoire :
    - repose sur une hypothèse de complexité (problème difficile)
    - ☺ nouvelles fonctionnalités possibles
    - ☹ sécurité dépendante des avancées algorithmiques

Deux types de preuves de sécurité

1. sécurité inconditionnelle :
    - théorie de l'information (puissance de calcul de l'attaquant ∞)
    - ☺ garantie de sécurité forte
    - ☹ très inefficace (voire impossible)
2. sécurité calculatoire :
    - repose sur une hypothèse de complexité (problème difficile)
    - ☺ nouvelles fonctionnalités possibles
    - ☹ sécurité dépendante des avancées algorithmiques

Rapide historique

Un développement rapide à partir des années 80 :

- 1949 : Shannon (chiffrement parfait) [Sha49]
- 1976 : Diffie-Hellman (établissement de clé) [DH76]
- 1978 : Rivest-Shamir-Adleman (chiffrement à clé publique et signature) [RSA78]
- 1979 : Rabin (chiffrement à clé publique et signature reposant sur la factorisation) [Rab79]
- 1984 : Goldwasser-Micali (formalisation rigoureuse du chiffrement à clé publique) [GM84]
- 1988 : Goldwasser-Micali-Rabin (formalisation rigoureuse de la signature) [GMR88]
- 1993 : Bellare-Rogaway (modèle de l'oracle aléatoire) [BR93]

Rapide historique

Un développement rapide à partir des années 80 :

- 1949 : Shannon (chiffrement parfait) [Sha49]
- 1976 : Diffie-Hellman (établissement de clé) [DH76]
- 1978 : Rivest-Shamir-Adleman (chiffrement à clé publique et signature) [RSA78]
- 1979 : Rabin (chiffrement à clé publique et signature reposant sur la factorisation) [Rab79]
- 1984 : Goldwasser-Micali (formalisation rigoureuse du chiffrement à clé publique) [GM84]
- 1988 : Goldwasser-Micali-Rabin (formalisation rigoureuse de la signature) [GMR88]
- 1993 : Bellare-Rogaway (modèle de l'oracle aléatoire) [BR93]

Rapide historique

Un développement rapide à partir des années 80 :

- 1949 : Shannon (chiffrement parfait) [Sha49]
- 1976 : Diffie-Hellman (établissement de clé) [DH76]
- 1978 : Rivest-Shamir-Adleman (chiffrement à clé publique et signature) [RSA78]
- 1979 : Rabin (chiffrement à clé publique et signature reposant sur la factorisation) [Rab79]
- 1984 : Goldwasser-Micali (formalisation rigoureuse du chiffrement à clé publique) [GM84]
- 1988 : Goldwasser-Micali-Rabin (formalisation rigoureuse de la signature) [GMR88]
- 1993 : Bellare-Rogaway (modèle de l'oracle aléatoire) [BR93]

Rapide historique

Un développement rapide à partir des années 80 :

- 1949 : Shannon (chiffrement parfait) [Sha49]
- 1976 : Diffie-Hellman (établissement de clé) [DH76]
- 1978 : Rivest-Shamir-Adleman (chiffrement à clé publique et signature) [RSA78]
- 1979 : Rabin (chiffrement à clé publique et signature reposant sur la factorisation) [Rab79]
- 1984 : Goldwasser-Micali (formalisation rigoureuse du chiffrement à clé publique) [GM84]
- 1988 : Goldwasser-Micali-Rabin (formalisation rigoureuse de la signature) [GMR88]
- 1993 : Bellare-Rogaway (modèle de l'oracle aléatoire) [BR93]

Rapide historique

Un développement rapide à partir des années 80 :

- 1949 : Shannon (chiffrement parfait) [Sha49]
- 1976 : Diffie-Hellman (établissement de clé) [DH76]
- 1978 : Rivest-Shamir-Adleman (chiffrement à clé publique et signature) [RSA78]
- 1979 : Rabin (chiffrement à clé publique et signature reposant sur la factorisation) [Rab79]
- 1984 : Goldwasser-Micali (formalisation rigoureuse du chiffrement à clé publique) [GM84]
- 1988 : Goldwasser-Micali-Rabin (formalisation rigoureuse de la signature) [GMR88]
- 1993 : Bellare-Rogaway (modèle de l'oracle aléatoire) [BR93]

Rapide historique

Un développement rapide à partir des années 80 :

- 1949 : Shannon (chiffrement parfait) [Sha49]
- 1976 : Diffie-Hellman (établissement de clé) [DH76]
- 1978 : Rivest-Shamir-Adleman (chiffrement à clé publique et signature) [RSA78]
- 1979 : Rabin (chiffrement à clé publique et signature reposant sur la factorisation) [Rab79]
- 1984 : Goldwasser-Micali (formalisation rigoureuse du chiffrement à clé publique) [GM84]
- 1988 : Goldwasser-Micali-Rabin (formalisation rigoureuse de la signature) [GMR88]
- 1993 : Bellare-Rogaway (modèle de l'oracle aléatoire) [BR93]

Rapide historique

Un développement rapide à partir des années 80 :

- 1949 : Shannon (chiffrement parfait) [Sha49]
- 1976 : Diffie-Hellman (établissement de clé) [DH76]
- 1978 : Rivest-Shamir-Adleman (chiffrement à clé publique et signature) [RSA78]
- 1979 : Rabin (chiffrement à clé publique et signature reposant sur la factorisation) [Rab79]
- 1984 : Goldwasser-Micali (formalisation rigoureuse du chiffrement à clé publique) [GM84]
- 1988 : Goldwasser-Micali-Rabin (formalisation rigoureuse de la signature) [GMR88]
- 1993 : Bellare-Rogaway (modèle de l'oracle aléatoire) [BR93]

Plan

Généralités

Sécurité inconditionnelle (informationnelle)

Sécurité calculatoire (réductionniste)

Chiffrement symétrique : masque jetable (Vernam)

clé K (n bits)
message M (n bits)

clé K (n bits)

- hypothèse : K uniformément distribuée et indépendante de M
- $\Rightarrow M \oplus K$ uniformément distribué $\forall$ distribution de M
- l'attaquant n'obtient aucune information sur M
- clé aussi longue que le message

Chiffrement symétrique : masque jetable (Vernam)

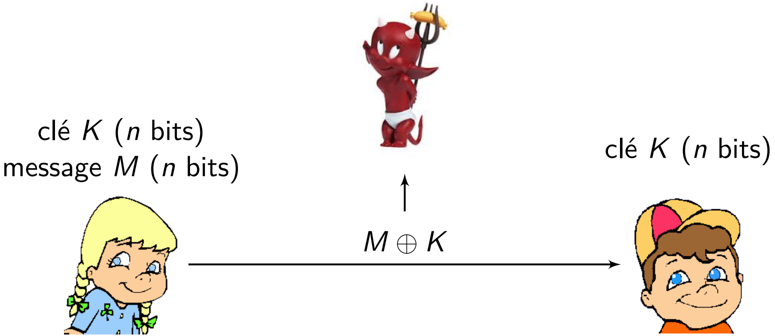
clé K (n bits)
message M (n bits)

$\xrightarrow{M \oplus K}$

clé K (n bits)

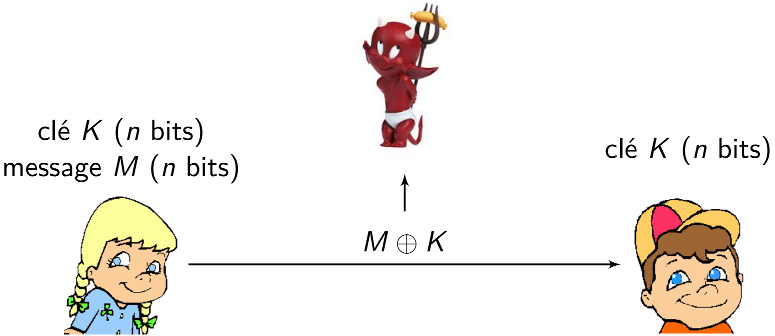
- hypothèse : K uniformément distribuée et indépendante de M
- $\Rightarrow M \oplus K$ uniformément distribué $\forall$ distribution de M
- l'attaquant n'obtient aucune information sur M
- clé aussi longue que le message

Chiffrement symétrique : masque jetable (Vernam)



- hypothèse : K uniformément distribuée et indépendante de M
- $\Rightarrow M \oplus K$ uniformément distribué $\forall$ distribution de M
- l'attaquant n'obtient aucune information sur M
- clé aussi longue que le message

Chiffrement symétrique : masque jetable (Vernam)



- hypothèse : K uniformément distribuée et indépendante de M
- $\Rightarrow M \oplus K$ uniformément distribué $\forall$ distribution de M
- l'attaquant n'obtient aucune information sur M
- clé aussi longue que le message

Théorème d'impossibilité de Shannon

Soit X une variable aléatoire à valeur dans un ensemble S.

- entropie (incertitude sur la v.a. X) :

$$H(X) = \sum_{x \in S} -p(x) \log p(x)$$

- entropie conditionnelle (incertitude sur Y connaissant X)

$$H(Y|X) = H(XY) - H(X) = \mathbb{E}_X [H(Y|X = x)]$$

- information mutuelle entre X et Y (information que X donne sur Y et réciproquement)

$$I(X;Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(XY)$$

Théorème d'impossibilité de Shannon

Soit X une variable aléatoire à valeur dans un ensemble S.

- entropie (incertitude sur la v.a. X) :

$$H(X) = \sum_{x \in S} -p(x) \log p(x)$$

- entropie conditionnelle (incertitude sur Y connaissant X)

$$H(Y|X) = H(XY) - H(X) = \mathbb{E}_X \left[H(Y|X = x) \right]$$

- information mutuelle entre X et Y (information que X donne sur Y et réciproquement)

$$I(X;Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(XY)$$

Théorème d'impossibilité de Shannon

Soit X une variable aléatoire à valeur dans un ensemble S.

- entropie (incertitude sur la v.a. X) :

$$H(X) = \sum_{x \in S} -p(x) \log p(x)$$

- entropie conditionnelle (incertitude sur Y connaissant X)

$$H(Y|X) = H(XY) - H(X) = \mathbb{E}_X \left[H(Y|X = x) \right]$$

- information mutuelle entre X et Y (information que X donne sur Y et réciproquement)

$$I(X;Y) = H(Y) - H(Y|X) = H(X) + H(Y) - H(XY)$$

Théorème d'impossibilité de Shannon

- message M = v.a. sur un ensemble $\mathcal{M}$
- clé K = v.a. sur une ensemble $\mathcal{K}$
- chiffré $C = E(K, M)$

Définition (Chiffrement parfait)

Un chiffrement symétrique est dit parfait si $I(C; M) = 0$ (le chiffré n'apporte aucune information sur le message clair).

Théorème (Shannon, 1949)

Un chiffrement parfait nécessite $H(K) \geq H(M)$. En particulier, si M est uniforme,

$$\log_2(|\mathcal{K}|) \geq \log_2(|\mathcal{M}|) \Rightarrow |K| \geq |M|.$$

$\Rightarrow$ Le masque jetable est optimal.

Théorème d'impossibilité de Shannon

- message $M =$ v.a. sur un ensemble $\mathcal{M}$
- clé $K =$ v.a. sur une ensemble $\mathcal{K}$
- chiffré $C = E(K, M)$

Définition (Chiffrement parfait)

Un chiffrement symétrique est dit parfait si $I(C; M) = 0$ (le chiffré n'apporte aucune information sur le message clair).

Théorème (Shannon, 1949)

Un chiffrement parfait nécessite $H(K) \geq H(M)$. En particulier, si M est uniforme,

$$\log_2(|\mathcal{K}|) \geq \log_2(|\mathcal{M}|) \Rightarrow |K| \geq |M|.$$

$\Rightarrow$ Le masque jetable est optimal.

Théorème d'impossibilité de Shannon

- message $M =$ v.a. sur un ensemble $\mathcal{M}$
- clé $K =$ v.a. sur une ensemble $\mathcal{K}$
- chiffré $C = E(K, M)$

Définition (Chiffrement parfait)

Un chiffrement symétrique est dit parfait si $I(C; M) = 0$ (le chiffré n'apporte aucune information sur le message clair).

Théorème (Shannon, 1949)

Un chiffrement parfait nécessite $H(K) \geq H(M)$. En particulier, si M est uniforme,

$$\log_2(|\mathcal{K}|) \geq \log_2(|\mathcal{M}|) \Rightarrow |K| \geq |M|.$$

$\Rightarrow$ Le masque jetable est optimal.

Authentification symétrique

clé K (k bits)
message M (m bits)

clé K (k bits)

- Alice veut transmettre M sans que l'attaquant puisse le modifier
- $f(K, M)$ = Message Authentication Code (MAC) de n bits
- l'attaquant ne doit pas pouvoir calculer $f(K, M')$ pour $M' \neq M$

Authentification symétrique

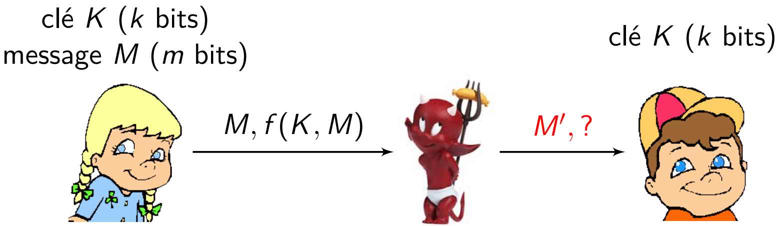
clé K (k bits)
message M (m bits)

$$M, \overbrace{f(K,M)}^{MAC} \longrightarrow$$

clé K (k bits)

- Alice veut transmettre M sans que l'attaquant puisse le modifier
- $f(K,M) =$ Message Authentication Code (MAC) de n bits
- l'attaquant ne doit pas pouvoir calculer $f(K,M')$ pour $M' \neq M$

Authentification symétrique



- Alice veut transmettre M sans que l'attaquant puisse le modifier
- $f(K, M) =$ Message Authentication Code (MAC) de n bits
- l'attaquant ne doit pas pouvoir calculer $f(K, M')$ pour $M' \neq M$

Authentification symétrique : Wegman-Carter

Définition (Hachage XOR-universel)

Une fonction $H(K, M)$ à valeurs dans $\{0,1\}^n$ est ε-XOR-universelle si pour tous messages $M \neq M'$ et tout $y \in \{0,1\}^n$,

$$\Pr_K \left[H(K, M) \oplus H(K, M') = y \right] \leq \varepsilon.$$

Exemple (Hachage polynomial)

Soit $K \in \{0,1\}^n$ et $M := (M_1, \ldots, M_b) \in (\{0,1\}^n)^b$. Alors

$$H(K, M) \stackrel{\text{def}}{=} \sum_{i=1}^{b} M_i \odot K^i \quad \text{(mult. sur corps fini GF}(2^n))$$

est ε-XU avec $\varepsilon = b2^{-n}$.

Preuve : $P(K) := H(K, M) \oplus H(K, M') \oplus y$ a au plus b racines.

Authentification symétrique : Wegman-Carter

Définition (Hachage XOR-universel)

Une fonction $H(K, M)$ à valeurs dans $\{0,1\}^n$ est ε-XOR-universelle si pour tous messages $M \neq M'$ et tout $y \in \{0,1\}^n$,

$$\Pr_K \left[H(K, M) \oplus H(K, M') = y \right] \leq \varepsilon.$$

Exemple (Hachage polynomial)

Soit $K \in \{0,1\}^n$ et $M := (M_1, \ldots, M_b) \in (\{0,1\}^n)^b$. Alors

$$H(K, M) \stackrel{\text{def}}{=} \sum_{i=1}^{b} M_i \odot K^i \quad \text{(mult. sur corps fini GF}(2^n))$$

est ε-XU avec $\varepsilon = b2^{-n}$.

Preuve : $P(K) := H(K, M) \oplus H(K, M') \oplus y$ a au plus b racines.

Authentification symétrique : Wegman-Carter

Définition (Hachage XOR-universel)

Une fonction $H(K, M)$ à valeurs dans $\{0,1\}^n$ est ε-XOR-universelle si pour tous messages $M \neq M'$ et tout $y \in \{0,1\}^n$,

$$\Pr_K \left[H(K, M) \oplus H(K, M') = y\right] \leq \varepsilon.$$

Exemple (Hachage polynomial)

Soit $K \in \{0,1\}^n$ et $M := (M_1, \ldots, M_b) \in (\{0,1\}^n)^b$. Alors

$$H(K, M) \stackrel{\text{def}}{=} \sum_{i=1}^{b} M_i \odot K^i \quad \text{(mult. sur corps fini GF}(2^n))$$

est ε-XU avec $\varepsilon = b2^{-n}$.

Preuve : $P(K) := H(K, M) \oplus H(K, M') \oplus y$ a au plus b racines.

Authentification symétrique : Wegman-Carter

Définition (MAC de Wegman-Carter)

Soit $(K, K') \in (\{0,1\}^n)^2$ la clé. Le MAC de M est défini par

$$f\Big((K, K'), (M_1, \ldots, M_b)\Big) = \underbrace{\left(\sum_{i=1}^{b} M_i \odot K^i\right)}_{H(K,M)} \underbrace{\oplus K'}_{\text{masque}}$$

- si H est ε-XU, un attaquant a une probabilité au plus ε de réussir à calculer le MAC d'un autre message $M' \neq M$
- messages multiples $\rightarrow$ renouveler seulement K'
- taille de clé indépendante de $b = |M|/n$!
- construction essentiellement optimale

Authentification symétrique : Wegman-Carter

Définition (MAC de Wegman-Carter)

Soit $(K, K') \in (\{0,1\}^n)^2$ *la clé. Le MAC de M est défini par*

$$f\Big((K,K'),(M_1,\ldots,M_b)\Big) = \underbrace{\left(\sum_{i=1}^{b} M_i \odot K^i\right)}_{H(K,M)} \oplus \underbrace{K'}_{\text{masque}}$$

- si H est ε-XU, un attaquant a une probabilité au plus ε de réussir à calculer le MAC d'un autre message $M' \neq M$
- messages multiples $\to$ renouveler seulement K'
- taille de clé indépendante de $b = |M|/n$!
- construction essentiellement optimale

Authentification symétrique : Wegman-Carter

Définition (MAC de Wegman-Carter)

Soit $(K, K') \in (\{0,1\}^n)^2$ *la clé. Le MAC de M est défini par*

$$f\Big((K, K'), (M_1, \dots, M_b)\Big) = \underbrace{\left(\sum_{i=1}^{b} M_i \odot K^i\right)}_{H(K,M)} \oplus \underbrace{K'}_{\text{masque}}$$

- si H est ε-XU, un attaquant a une probabilité au plus ε de réussir à calculer le MAC d'un autre message $M' \neq M$
- messages multiples $\rightarrow$ renouveler seulement K'
- taille de clé indépendante de $b = |M|/n$!
- construction essentiellement optimale

Authentification symétrique : Wegman-Carter

Définition (MAC de Wegman-Carter)

Soit $(K, K') \in (\{0,1\}^n)^2$ *la clé. Le MAC de* M *est défini par*

$$f\Big((K, K'), (M_1, \ldots, M_b)\Big) = \underbrace{\left(\sum_{i=1}^{b} M_i \odot K^i\right)}_{H(K,M)} \oplus \underbrace{K'}_{\text{masque}}$$

- si H est ε-XU, un attaquant a une probabilité au plus ε de réussir à calculer le MAC d'un autre message $M' \neq M$
- messages multiples $\rightarrow$ renouveler seulement K'
- taille de clé indépendante de $b = |M|/n$!
- construction essentiellement optimale

Authentification symétrique : Wegman-Carter

Définition (MAC de Wegman-Carter)

Soit $(K, K') \in (\{0,1\}^n)^2$ *la clé. Le MAC de M est défini par*

$$f\Big((K,K'),(M_1,\ldots,M_b)\Big) = \underbrace{\left(\sum_{i=1}^{b} M_i \odot K^i\right)}_{H(K,M)} \oplus \underbrace{K'}_{\text{masque}}$$

- si H est ε-XU, un attaquant a une probabilité au plus ε de réussir à calculer le MAC d'un autre message $M' \neq M$
- messages multiples $\rightarrow$ renouveler seulement K'
- taille de clé indépendante de $b = |M|/n$!
- construction essentiellement optimale

Autres exemples de sécurité inconditionnelle

- distribution de clé quantique (en théorie ☹)

Autres exemples de sécurité inconditionnelle

- distribution de clé quantique (en théorie ☹)

Plan

Problèmes calculatoirement difficiles

Définition

Un problème est dit calculatoirement difficile si la complexité du meilleur algorithme de résolution connu augmente vite en fonction de la taille de l'entrée

- vite = exponentiel, à défaut super-polynomial
- en général, pas de preuve « absolue » de la complexité minimale du meilleur algorithme de résolution
- permet de dépasser les limites intrinsèques de la sécurité inconditionnelle
  (ex : chiffrement à clé publique → impossible de façon inconditionnellement sûre)

Problèmes calculatoirement difficiles

Définition

Un problème est dit calculatoirement difficile si la complexité du meilleur algorithme de résolution connu augmente vite en fonction de la taille de l'entrée

- vite = exponentiel, à défaut super-polynomial
- en général, pas de preuve « absolue » de la complexité minimale du meilleur algorithme de résolution
- permet de dépasser les limites intrinsèques de la sécurité inconditionnelle
  (ex : chiffrement à clé publique → impossible de façon inconditionnellement sûre)

Problèmes calculatoirement difficiles

Définition

Un problème est dit calculatoirement difficile si la complexité du meilleur algorithme de résolution connu augmente vite en fonction de la taille de l'entrée

- vite = exponentiel, à défaut super-polynomial
- en général, pas de preuve « absolue » de la complexité minimale du meilleur algorithme de résolution
- permet de dépasser les limites intrinsèques de la sécurité inconditionnelle
  (ex : chiffrement à clé publique → impossible de façon inconditionnellement sûre)

Problèmes calculatoirement difficiles

Exemple (Factorisation)

- étant donné $n = pq$, $p \neq q$ premiers, calculer (p, q)
- meilleur algo : $T = \exp(C|n|^{1/3} \log^{2/3} |n|)$
  ($|n|$ taille de n en bits)
  $\rightarrow$ super-polynomial, sous-exponentiel

Exemple (Logarithme discret)

- soit G un groupe cyclique fini de taille n, g un générateur
- étant donné $X \in G$, trouve $x \in \mathbb{Z}_n$ tel que $X = g^x$
- dans certains groupes (courbes elliptiques) meilleur algo requiert $T = 2^{|n|/2}$ (exponentiel)

Problèmes calculatoirement difficiles

Exemple (Factorisation)

- étant donné $n = pq$, $p \neq q$ premiers, calculer (p, q)
- meilleur algo : $T = \exp(C|n|^{1/3} \log^{2/3} |n|)$
  ($|n|$ taille de n en bits)
  $\rightarrow$ super-polynomial, sous-exponentiel

Exemple (Logarithme discret)

- soit G un groupe cyclique fini de taille n, g un générateur
- étant donné $X \in G$, trouve $x \in \mathbb{Z}_n$ tel que $X = g^x$
- dans certains groupes (courbes elliptiques) meilleur algo requiert $T = 2^{|n|/2}$ (exponentiel)

Notion de réduction

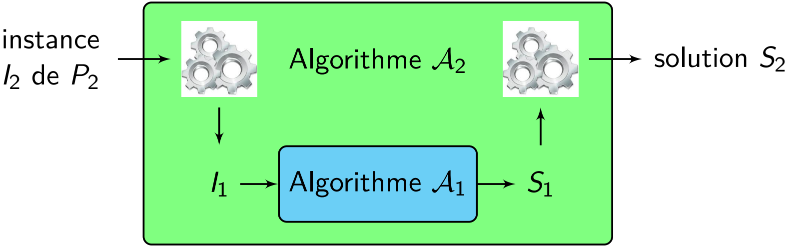
- issu de la théorie de la complexité (NP-complétude)
- à partir de $\mathcal{A}_1$ qui résout problème P_1 en temps t_1 avec proba ε_1
- on construit $\mathcal{A}_2$ qui résout P_2 en temps t_2 avec proba ε_2
- on a « réduit » le problème P_2 au problème P_1

Notion de réduction

- issu de la théorie de la complexité (NP-complétude)
- à partir de $\mathcal{A}_1$ qui résout problème P_1 en temps t_1 avec proba ε_1
- on construit $\mathcal{A}_2$ qui résout P_2 en temps t_2 avec proba ε_2
- on a « réduit » le problème P_2 au problème P_1

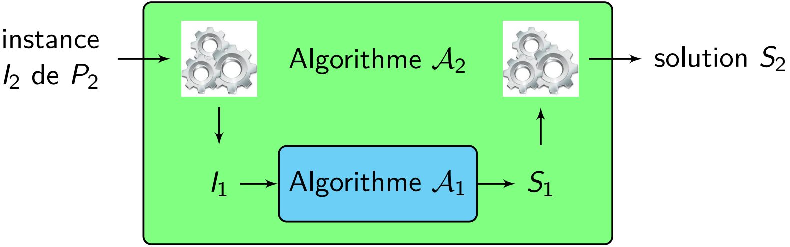
Notion de réduction

- issu de la théorie de la complexité (NP-complétude)
- à partir de $\mathcal{A}_1$ qui résout problème P_1 en temps t_1 avec proba ε_1
- on construit $\mathcal{A}_2$ qui résout P_2 en temps t_2 avec proba ε_2
- on a « réduit » le problème P_2 au problème P_1



Notion de réduction

- issu de la théorie de la complexité (NP-complétude)
- à partir de $\mathcal{A}_1$ qui résout problème P_1 en temps t_1 avec proba ε_1
- on construit $\mathcal{A}_2$ qui résout P_2 en temps t_2 avec proba ε_2
- on a « réduit » le problème P_2 au problème P_1



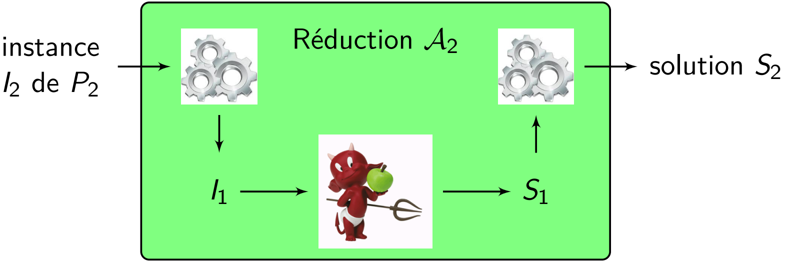
Réduction cryptographique

- problème P_1 = casser le cryptosystème
- problème P_2 = problème difficile (factorisation, log discret. . .)
- $\exists$ attaquant $\mathcal{A}_1 \Rightarrow \exists$ algorithme de résolution de P_2
- $\nexists$ algorithme de résolution de $P_2 \Rightarrow \nexists$ attaquant $\mathcal{A}_1$
- la réduction $\mathcal{A}_2$ doit parfois « simuler » les oracles auxquels l'attaquant a accès (oracle de signature, de déchiffrement, etc.)

$I_1 \longrightarrow$  $\longrightarrow S_1$

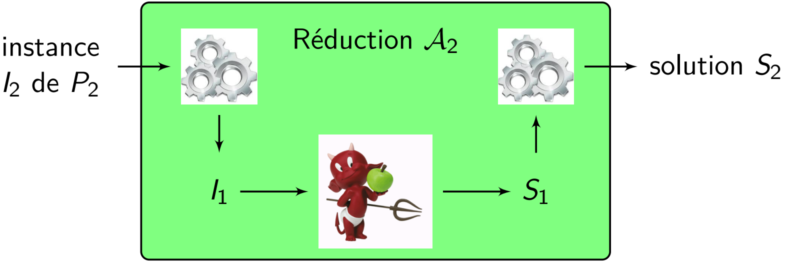
Réduction cryptographique

- problème P_1 = casser le cryptosystème
- problème P_2 = problème difficile (factorisation, log discret...)
- $\exists$ attaquant $\mathcal{A}_1 \Rightarrow \exists$ algorithme de résolution de P_2
- $\nexists$ algorithme de résolution de $P_2 \Rightarrow \nexists$ attaquant $\mathcal{A}_1$
- la réduction $\mathcal{A}_2$ doit parfois « simuler » les oracles auxquels l'attaquant a accès (oracle de signature, de déchiffrement, etc.)



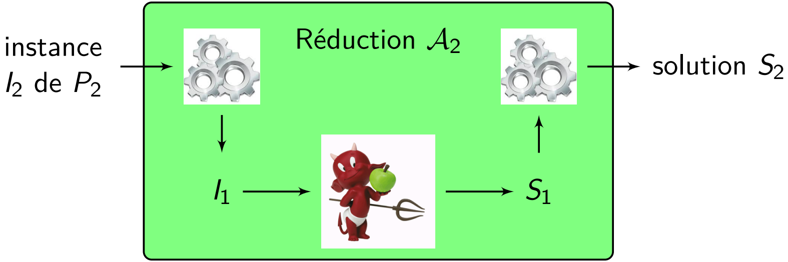
Réduction cryptographique

- problème P_1 = casser le cryptosystème
- problème P_2 = problème difficile (factorisation, log discret...)
- $\exists$ attaquant $\mathcal{A}_1 \Rightarrow \exists$ algorithme de résolution de P_2
- $\nexists$ algorithme de résolution de $P_2 \Rightarrow \nexists$ attaquant $\mathcal{A}_1$
- la réduction $\mathcal{A}_2$ doit parfois « simuler » les oracles auxquels l'attaquant a accès (oracle de signature, de déchiffrement, etc.)



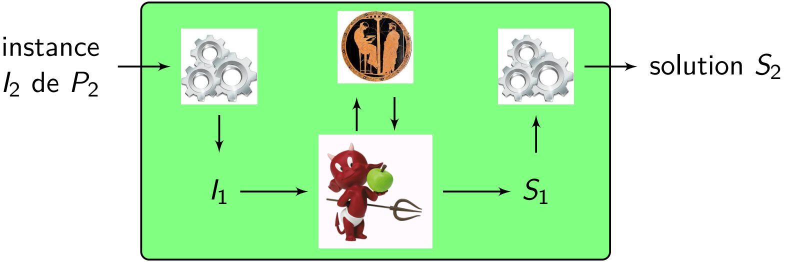
Réduction cryptographique

- problème P_1 = casser le cryptosystème
- problème P_2 = problème difficile (factorisation, log discret...)
- $\exists$ attaquant $\mathcal{A}_1 \Rightarrow \exists$ algorithme de résolution de P_2
- $\nexists$ algorithme de résolution de $P_2 \Rightarrow \nexists$ attaquant $\mathcal{A}_1$
- la réduction $\mathcal{A}_2$ doit parfois « simuler » les oracles auxquels l'attaquant a accès (oracle de signature, de déchiffrement, etc.)



Réduction cryptographique

- problème P_1 = casser le cryptosystème
- problème P_2 = problème difficile (factorisation, log discret...)
- $\exists$ attaquant $\mathcal{A}_1 \Rightarrow \exists$ algorithme de résolution de P_2
- $\nexists$ algorithme de résolution de $P_2 \Rightarrow \nexists$ attaquant $\mathcal{A}_1$
- la réduction $\mathcal{A}_2$ doit parfois « simuler » les oracles auxquels l'attaquant a accès (oracle de signature, de déchiffrement, etc.)



Finesse de la réduction

Efficacité de la réduction $\mathcal{A}_2$:

- idéalement, $t_2 \simeq t_1$ et $\varepsilon_2 \simeq \varepsilon_1$
- une réduction est dite fine si

$$\frac{t_2}{\varepsilon_2} \leq C \frac{t_1}{\varepsilon_1}$$

  où C est une petite constante indépendante des paramètres (taille de clé, etc.)
- une réduction fine permet de fixer les paramètres du cryptosystème au plus prêt de ceux pour lesquels le problème sous-jacent est infaisable

Finesse de la réduction

Efficacité de la réduction $\mathcal{A}_2$:

- idéalement, $t_2 \simeq t_1$ et $\varepsilon_2 \simeq \varepsilon_1$
- une réduction est dite fine si

$$\frac{t_2}{\varepsilon_2} \leq C \frac{t_1}{\varepsilon_1}$$

  où C est une petite constante indépendante des paramètres (taille de clé, etc.)

- une réduction fine permet de fixer les paramètres du cryptosystème au plus prêt de ceux pour lesquels le problème sous-jacent est infaisable

Finesse de la réduction

Efficacité de la réduction $\mathcal{A}_2$:

- idéalement, $t_2 \simeq t_1$ et $\varepsilon_2 \simeq \varepsilon_1$
- une réduction est dite fine si

$$\frac{t_2}{\varepsilon_2} \leq C \frac{t_1}{\varepsilon_1}$$

  où C est une petite constante indépendante des paramètres (taille de clé, etc.)
- une réduction fine permet de fixer les paramètres du cryptosystème au plus prêt de ceux pour lesquels le problème sous-jacent est infaisable

Exemple de réduction : racine carrée vs. factorisation

Racine carrée modulaire :

- n-résidu quadratique = élément y de $\mathbb{Z}_n^*$ qui est un carré :

$$\exists x \in \{0, \ldots, n-1\} : x^2 \equiv y \bmod n$$

- problème calculatoire : étant donné un n-résidu y, calculer une racine carrée de y
- « facile » quand n est premier : temps $\mathtt{poly}(|n|_2)$ (probabiliste si $n \equiv 1 \bmod 8$, déterministe sinon)
- n composite ?

Exemple de réduction : racine carrée vs. factorisation

Racine carrée modulaire :

- n-résidu quadratique = élément y de $\mathbb{Z}_n^*$ qui est un carré :

$$\exists x \in \{0, \ldots, n-1\} \; : \; x^2 \equiv y \bmod n$$

- problème calculatoire : étant donné un n-résidu y, calculer une racine carrée de y
- « facile » quand n est premier : temps $\texttt{poly}(|n|_2)$ (probabiliste si $n \equiv 1 \bmod 8$, déterministe sinon)
- n composite ?

Exemple de réduction : racine carrée vs. factorisation

Racine carrée modulaire :

- n-résidu quadratique = élément y de $\mathbb{Z}_n^*$ qui est un carré :

$$\exists x \in \{0, \ldots, n-1\} \; : \; x^2 \equiv y \bmod n$$

- problème calculatoire : étant donné un n-résidu y, calculer une racine carrée de y
- « facile » quand n est premier : temps $\texttt{poly}(|n|_2)$ (probabiliste si $n \equiv 1 \bmod 8$, déterministe sinon)
- n composite ?

Exemple de réduction : racine carrée vs. factorisation

Racine carrée modulaire :

- n-résidu quadratique = élément y de $\mathbb{Z}_n^*$ qui est un carré :

$$\exists x \in \{0, \ldots, n-1\} : x^2 \equiv y \bmod n$$

- problème calculatoire : étant donné un n-résidu y, calculer une racine carrée de y
- « facile » quand n est premier : temps $\texttt{poly}(|n|_2)$ (probabiliste si $n \equiv 1 \bmod 8$, déterministe sinon)
- n composite ?

Exemple de réduction : racine carrée vs. factorisation

- soit $n = pq$, p et q premiers distincts
- tout n-résidu a exactement 4 racines, $\pm x_1, \pm x_2$

Théorème

Si n est composite, calculer une racine carrée d'un n-résidu y est « aussi dur » que factoriser n.

Preuve (factorisation ⇒ racine).

Si on sait factoriser n, on peut calculer les 4 racines carrées mod n en combinant les racines mod p et q par le théorème des restes chinois. □

Exemple de réduction : racine carrée vs. factorisation

- soit $n = pq$, p et q premiers distincts
- tout n-résidu a exactement 4 racines, $\pm x_1, \pm x_2$

Théorème

Si n est composite, calculer une racine carrée d'un n-résidu y est « aussi dur » que factoriser n.

Preuve (factorisation $\Rightarrow$ racine).

Si on sait factoriser n, on peut calculer les 4 racines carrées mod n en combinant les racines mod p et q par le théorème des restes chinois. □

Exemple de réduction : racine carrée vs. factorisation

- soit $n = pq$, p et q premiers distincts
- tout n-résidu a exactement 4 racines, $\pm x_1, \pm x_2$

Théorème

Si n est composite, calculer une racine carrée d'un n-résidu y est « aussi dur » que factoriser n.

Preuve (factorisation $\Rightarrow$ racine).

Si on sait factoriser n, on peut calculer les 4 racines carrées mod n en combinant les racines mod p et q par le théorème des restes chinois. $\square$

Exemple de réduction : racine carrée vs. factorisation

Preuve (racine $\Rightarrow$ factorisation).

Supposons qu'il existe un algorithme $\mathcal{A}$ qui, étant donné n et un n-résidu y, retourne une racine carrée x de y. Soit $\mathcal{B}$ l'algorithme qui tire un x aléatoire et calcule $y = x^2 \bmod n$ qu'il donne à $\mathcal{A}$. Si $\mathcal{A}$ renvoie une racine $x' \neq \pm x \bmod n$, alors

$$\begin{aligned} x^2 \equiv x'^2 \bmod n &\Leftrightarrow (x - x')(x + x') \equiv 0 \bmod n \\ &\Rightarrow \gcd(x - x', N) = p \text{ or } q \end{aligned} \qquad \square$$

On a réduit le problème de la factorisation de n au problème de calculer des racines carrées mod n. La réduction est fine :

$$t_{\mathcal{B}} \simeq \underbrace{t_{x^2} + t_{\mathrm{gcd}}}_{\ll t_{\mathcal{A}}} + t_{\mathcal{A}} \simeq t_{\mathcal{A}}, \qquad \varepsilon_{\mathcal{B}} \simeq \frac{\varepsilon_{\mathcal{A}}}{2}.$$

Exemple de réduction : racine carrée vs. factorisation

Preuve (racine $\Rightarrow$ factorisation).

Supposons qu'il existe un algorithme $\mathcal{A}$ qui, étant donné n et un n-résidu y, retourne une racine carrée x de y. Soit $\mathcal{B}$ l'algorithme qui tire un x aléatoire et calcule $y = x^2 \bmod n$ qu'il donne à $\mathcal{A}$. Si $\mathcal{A}$ renvoie une racine $x' \neq \pm x \bmod n$, alors

$$\begin{aligned} x^2 \equiv x'^2 \bmod n &\Leftrightarrow (x - x')(x + x') \equiv 0 \bmod n \\ &\Rightarrow \gcd(x - x', N) = p \text{ or } q \end{aligned} \qquad \square$$

On a réduit le problème de la factorisation de n au problème de calculer des racines carrées mod n. La réduction est fine :

$$t_{\mathcal{B}} \simeq \underbrace{t_{x^2} + t_{\text{gcd}}}_{\ll t_{\mathcal{A}}} + t_{\mathcal{A}} \simeq t_{\mathcal{A}}, \qquad \varepsilon_{\mathcal{B}} \simeq \frac{\varepsilon_{\mathcal{A}}}{2}.$$

Schéma de signature de Rabin

Signature de Rabin-Williams, version « basique »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \mathbb{Z}_n^*$:
  - calculer une racine carrée σ de $M \bmod n$
  - si M est un non-résidu, considérer αM, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = M \bmod n$?

Sécurité :

- un attaquant ne connaissant pas (p, q) ne peut pas calculer de signature pour un message M quelconque
- mais il peut choisir σ, et calculer $M = \sigma^2 \bmod n$ $\Rightarrow (M, \sigma)$ est une paire message/signature valide
- M « non contrôlé » par l'attaquant = forge existentielle

Schéma de signature de Rabin

Signature de Rabin-Williams, version « basique »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \mathbb{Z}_n^*$:
  - calculer une racine carrée σ de $M \bmod n$
  - si M est un non-résidu, considérer αM, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = M \bmod n$?

Sécurité :

- un attaquant ne connaissant pas (p, q) ne peut pas calculer de signature pour un message M quelconque
- mais il peut choisir σ, et calculer $M = \sigma^2 \bmod n$
  $\Rightarrow (M, \sigma)$ est une paire message/signature valide
- M « non contrôlé » par l'attaquant = forge existentielle

Schéma de signature de Rabin

Signature de Rabin-Williams, version « basique »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \mathbb{Z}_n^*$:
    - calculer une racine carrée σ de $M \bmod n$
    - si M est un non-résidu, considérer αM, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = M \bmod n$?

Sécurité :

- un attaquant ne connaissant pas (p, q) ne peut pas calculer de signature pour un message M quelconque
- mais il peut choisir σ, et calculer $M = \sigma^2 \bmod n$
  $\Rightarrow (M, \sigma)$ est une paire message/signature valide
- M « non contrôlé » par l'attaquant = forge existentielle

Schéma de signature de Rabin

Signature de Rabin-Williams, version « basique »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \mathbb{Z}_n^*$:
  - calculer une racine carrée σ de $M \bmod n$
  - si M est un non-résidu, considérer αM, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = M \bmod n$?

Sécurité :

- un attaquant ne connaissant pas (p, q) ne peut pas calculer de signature pour un message M quelconque
- mais il peut choisir σ, et calculer $M = \sigma^2 \bmod n$
  $\Rightarrow (M, \sigma)$ est une paire message/signature valide
- M « non contrôlé » par l'attaquant = forge existentielle

Schéma de signature de Rabin

Signature de Rabin-Williams, version « basique »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \mathbb{Z}_n^*$:
  - calculer une racine carrée σ de $M \bmod n$
  - si M est un non-résidu, considérer αM, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = M \bmod n$?

Sécurité :

- un attaquant ne connaissant pas (p, q) ne peut pas calculer de signature pour un message M quelconque
- mais il peut choisir σ, et calculer $M = \sigma^2 \bmod n$
  $\Rightarrow (M, \sigma)$ est une paire message/signature valide
- M « non contrôlé » par l'attaquant = forge existentielle

Schéma de signature de Rabin

Signature de Rabin-Williams, version « basique »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \mathbb{Z}_n^*$:
  - calculer une racine carrée σ de $M \bmod n$
  - si M est un non-résidu, considérer αM, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = M \bmod n$?

Sécurité :

- un attaquant ne connaissant pas (p, q) ne peut pas calculer de signature pour un message M quelconque
- mais il peut choisir σ, et calculer $M = \sigma^2 \bmod n$
  $\Rightarrow (M, \sigma)$ est une paire message/signature valide
- M « non contrôlé » par l'attaquant = forge existentielle

Schéma de signature de Rabin

Signature de Rabin-Williams, version « basique »

- clé privée (p, q) $(p \equiv 3 \bmod 8,\ q \equiv 7 \bmod 8)$
- clé publique $n = pq$
- signature d'un message $M \in \mathbb{Z}_n^*$:
  - calculer une racine carrée σ de M mod n
  - si M est un non-résidu, considérer αM, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = M \bmod n$?

Sécurité :

- un attaquant ne connaissant pas (p, q) ne peut pas calculer de signature pour un message M quelconque
- mais il peut choisir σ, et calculer $M = \sigma^2 \bmod n$
  $\Rightarrow (M, \sigma)$ est une paire message/signature valide
- M « non contrôlé » par l'attaquant = forge existentielle

Formalisation de la sécurité d'un schéma de signature

Objectif de sécurité (on veut empêcher que…) :

- inforgeabilité universelle (IF-Univ) : l'attaquant puisse forger une signature pour (quasiment) n'importe quel message, non choisi par l'attaquant
- inforgeabilité sélective (IF-Sel) : l'attaquant puisse forger une signature pour un message qu'il choisit avant de connaitre la clé publique
- inforgeabilité existentielle (IF-Ex) : il existe un message pour lequel l'attaquant puisse forger une signature

$$\text{IF-Univ} \Longleftarrow \text{IF-Sel} \Longleftarrow \text{IF-Ex}$$

$$\xrightarrow{\text{propriété plus forte}}$$

Formalisation de la sécurité d'un schéma de signature

Objectif de sécurité (on veut empêcher que. . .) :

- inforgeabilité universelle (IF-Univ) : l'attaquant puisse forger une signature pour (quasiment) n'importe quel message, non choisi par l'attaquant
- inforgeabilité sélective (IF-Sel) : l'attaquant puisse forger une signature pour un message qu'il choisit avant de connaitre la clé publique
- inforgeabilité existentielle (IF-Ex) : il existe un message pour lequel l'attaquant puisse forger une signature

$$\text{IF-Univ} \Longleftarrow \text{IF-Sel} \Longleftarrow \text{IF-Ex}$$

$$\xrightarrow{\text{propriété plus forte}}$$

Formalisation de la sécurité d'un schéma de signature

Objectif de sécurité (on veut empêcher que. . .) :

- inforgeabilité universelle (IF-Univ) : l'attaquant puisse forger une signature pour (quasiment) n'importe quel message, non choisi par l'attaquant
- inforgeabilité sélective (IF-Sel) : l'attaquant puisse forger une signature pour un message qu'il choisit avant de connaitre la clé publique
- inforgeabilité existentielle (IF-Ex) : il existe un message pour lequel l'attaquant puisse forger une signature

$$\text{IF-Univ} \Longleftarrow \text{IF-Sel} \Longleftarrow \text{IF-Ex}$$

$$\xrightarrow{\text{propriété plus forte}}$$

Formalisation de la sécurité d'un schéma de signature

Objectif de sécurité (on veut empêcher que. . .) :

- inforgeabilité universelle (IF-Univ) : l'attaquant puisse forger une signature pour (quasiment) n'importe quel message, non choisi par l'attaquant
- inforgeabilité sélective (IF-Sel) : l'attaquant puisse forger une signature pour un message qu'il choisit avant de connaitre la clé publique
- inforgeabilité existentielle (IF-Ex) : il existe un message pour lequel l'attaquant puisse forger une signature

$$\text{IF-Univ} \Longleftarrow \text{IF-Sel} \Longleftarrow \text{IF-Ex}$$

$$\xrightarrow{\text{propriété plus forte}}$$

Formalisation de la sécurité d'un schéma de signature

Modèle d'attaque :

- attaques sans messages (ASM) : l'attaquant ne connait que la clé publique de l'utilisateur
- attaque à messages connus (AMK) : l'attaquant a accès à la signature de messages connus (aléatoires)
- attaque à messages choisis (AMC) : l'attaquant peut demander la signature de messages de son choix à un oracle de signature

$$\text{ASM} \Longleftarrow \text{AMK} \Longleftarrow \text{AMC}$$

$$\xrightarrow{\text{attaque plus forte}}$$

Formalisation de la sécurité d'un schéma de signature

Modèle d'attaque :

- attaques sans messages (ASM) : l'attaquant ne connait que la clé publique de l'utilisateur
- attaque à messages connus (AMK) : l'attaquant a accès à la signature de messages connus (aléatoires)
- attaque à messages choisis (AMC) : l'attaquant peut demander la signature de messages de son choix à un oracle de signature

$$\text{ASM} \Longleftarrow \text{AMK} \Longleftarrow \text{AMC}$$

$$\xrightarrow{\text{attaque plus forte}}$$

Formalisation de la sécurité d'un schéma de signature

Modèle d'attaque :

- attaques sans messages (ASM) : l'attaquant ne connait que la clé publique de l'utilisateur
- attaque à messages connus (AMK) : l'attaquant a accès à la signature de messages connus (aléatoires)
- attaque à messages choisis (AMC) : l'attaquant peut demander la signature de messages de son choix à un oracle de signature

$$\text{ASM} \Longleftarrow \text{AMK} \Longleftarrow \text{AMC}$$

$$\xrightarrow{\text{attaque plus forte}}$$

Formalisation de la sécurité d'un schéma de signature

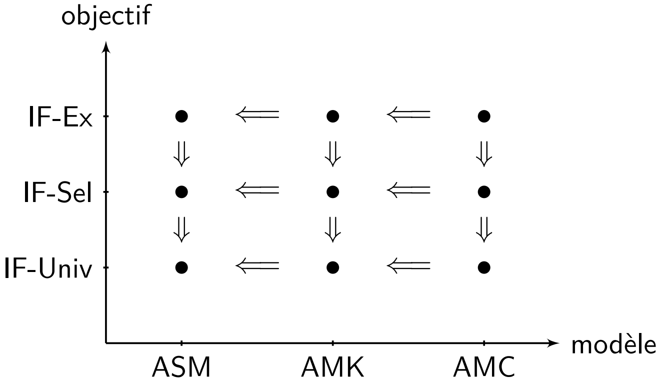
Modèle d'attaque :

- attaques sans messages (ASM) : l'attaquant ne connait que la clé publique de l'utilisateur
- attaque à messages connus (AMK) : l'attaquant a accès à la signature de messages connus (aléatoires)
- attaque à messages choisis (AMC) : l'attaquant peut demander la signature de messages de son choix à un oracle de signature

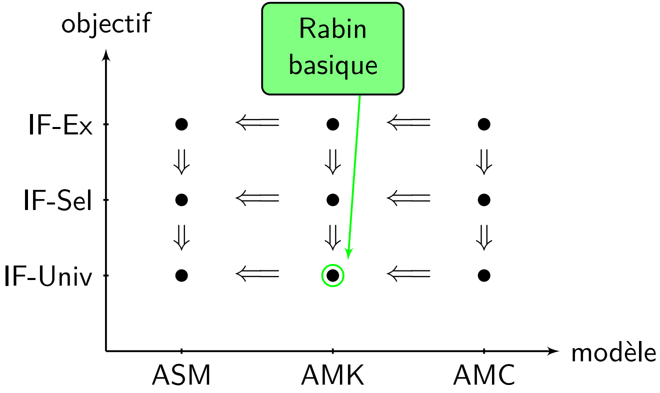
$$\text{ASM} \Longleftarrow \text{AMK} \Longleftarrow \text{AMC}$$

$$\xrightarrow{\text{attaque plus forte}}$$

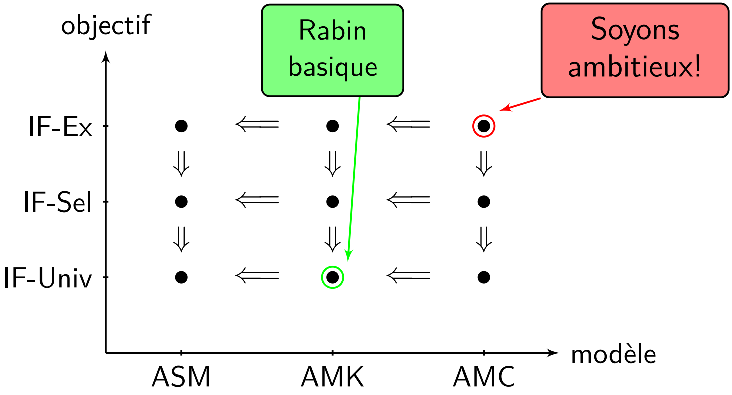
Relation entre les notions de sécurité



Relation entre les notions de sécurité



Relation entre les notions de sécurité



IF-Ex-AMC = Inforgeabilité existentielle contre les attaques à messages choisis

Schéma de signature de Rabin amélioré

Définition (Fonction de hachage)

Fonction H de $\{0,1\}^*$ vers $\{0,1\}^m$ vérifiant à minima :

- *résistance à la pré-image : étant donné $y \in \{0,1\}^m$, trouver x tel que $H(x) = y$ doit demander $\sim 2^m$ évaluations de H*
- *résistance à la seconde pré-image : étant donné x, trouver $x' \neq x$ tel que $H(x') = H(x)$ doit demander $\sim 2^m$ évaluations de H*
- *résistance aux collisions : trouver x, x' tels que $H(x) = H(x')$ doit demander $\sim 2^{m/2}$ évaluations de H*
- *idéalement, H doit se comporter comme un oracle aléatoire*

Schéma de signature de Rabin amélioré

Définition (Fonction de hachage)

Fonction H de $\{0,1\}^*$ vers $\{0,1\}^m$ vérifiant à minima :

- *résistance à la pré-image : étant donné $y \in \{0,1\}^m$, trouver x tel que $H(x) = y$ doit demander $\sim 2^m$ évaluations de H*
- *résistance à la seconde pré-image : étant donné x, trouver $x' \neq x$ tel que $H(x') = H(x)$ doit demander $\sim 2^m$ évaluations de H*
- *résistance aux collisions : trouver x, x' tels que $H(x) = H(x')$ doit demander $\sim 2^{m/2}$ évaluations de H*
- *idéalement, H doit se comporter comme un oracle aléatoire*

Schéma de signature de Rabin amélioré

Définition (Fonction de hachage)

Fonction H de $\{0,1\}^*$ vers $\{0,1\}^m$ vérifiant à minima :

- *résistance à la pré-image : étant donné $y \in \{0,1\}^m$, trouver x tel que $H(x) = y$ doit demander $\sim 2^m$ évaluations de H*
- *résistance à la seconde pré-image : étant donné x, trouver $x' \neq x$ tel que $H(x') = H(x)$ doit demander $\sim 2^m$ évaluations de H*
- *résistance aux collisions : trouver x, x' tels que $H(x) = H(x')$ doit demander $\sim 2^{m/2}$ évaluations de H*
- *idéalement, H doit se comporter comme un oracle aléatoire*

Schéma de signature de Rabin amélioré

Définition (Fonction de hachage)

Fonction H de $\{0,1\}^*$ vers $\{0,1\}^m$ vérifiant à minima :

- *résistance à la pré-image : étant donné $y \in \{0,1\}^m$, trouver x tel que $H(x) = y$ doit demander $\sim 2^m$ évaluations de H*
- *résistance à la seconde pré-image : étant donné x, trouver $x' \neq x$ tel que $H(x') = H(x)$ doit demander $\sim 2^m$ évaluations de H*
- *résistance aux collisions : trouver x, x' tels que $H(x) = H(x')$ doit demander $\sim 2^{m/2}$ évaluations de H*
- *idéalement, H doit se comporter comme un oracle aléatoire*

Schéma de signature de Rabin amélioré

Signatures de Rabin-Williams « *Full Domain Hash* »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \{0, 1\}^*$:
  - calculer une racine carrée σ de $H(M) \bmod n$
  - si $H(M)$ est un non-résidu, considérer $\alpha H(M)$, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = H(M) \bmod n$?

Sécurité :

- l'attaque sur la version précédente ne marche plus : il faudrait inverser H
- preuve de sécurité ?

  $\Rightarrow$ possible en modélisant H comme un oracle aléatoire [BR93]

Schéma de signature de Rabin amélioré

Signatures de Rabin-Williams « *Full Domain Hash* »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \{0,1\}^*$:
  - calculer une racine carrée σ de $H(M) \bmod n$
  - si $H(M)$ est un non-résidu, considérer $\alpha H(M)$, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = H(M) \bmod n$?

Sécurité :

- l'attaque sur la version précédente ne marche plus : il faudrait inverser H
- preuve de sécurité ?
  $\Rightarrow$ possible en modélisant H comme un oracle aléatoire [BR93]

Schéma de signature de Rabin amélioré

Signatures de Rabin-Williams « *Full Domain Hash* »

- clé privée (p, q) $(p \equiv 3 \bmod 8,\ q \equiv 7 \bmod 8)$
- clé publique $n = pq$
- signature d'un message $M \in \{0,1\}^*$:
  - calculer une racine carrée σ de $H(M)$ mod n
  - si $H(M)$ est un non-résidu, considérer $\alpha H(M)$, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = H(M) \bmod n$?

Sécurité :

- l'attaque sur la version précédente ne marche plus : il faudrait inverser H
- preuve de sécurité ?
  $\Rightarrow$ possible en modélisant H comme un oracle aléatoire [BR93]

Schéma de signature de Rabin amélioré

Signatures de Rabin-Williams « *Full Domain Hash* »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \{0, 1\}^*$:
  - calculer une racine carrée σ de $H(M) \bmod n$
  - si $H(M)$ est un non-résidu, considérer $\alpha H(M)$, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = H(M) \bmod n$?

Sécurité :

- l'attaque sur la version précédente ne marche plus : il faudrait inverser H
- preuve de sécurité ?
  $\Rightarrow$ possible en modélisant H comme un oracle aléatoire [BR93]

Schéma de signature de Rabin amélioré

Signatures de Rabin-Williams « *Full Domain Hash* »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \{0,1\}^*$:
  - calculer une racine carrée σ de $H(M) \bmod n$
  - si $H(M)$ est un non-résidu, considérer $\alpha H(M)$, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = H(M) \bmod n$?

Sécurité :

- l'attaque sur la version précédente ne marche plus : il faudrait inverser H
- preuve de sécurité ?
  $\Rightarrow$ possible en modélisant H comme un oracle aléatoire [BR93]

Schéma de signature de Rabin amélioré

Signatures de Rabin-Williams « *Full Domain Hash* »

- clé privée (p, q) ($p \equiv 3 \bmod 8$, $q \equiv 7 \bmod 8$)
- clé publique $n = pq$
- signature d'un message $M \in \{0,1\}^*$:
  - calculer une racine carrée σ de $H(M) \bmod n$
  - si $H(M)$ est un non-résidu, considérer $\alpha H(M)$, $\alpha \in \{\pm 1, \pm 2\}$
- vérification d'une signature : $\sigma^2 = H(M) \bmod n$?

Sécurité :

- l'attaque sur la version précédente ne marche plus : il faudrait inverser H
- preuve de sécurité ?
  $\Rightarrow$ possible en modélisant H comme un oracle aléatoire [BR93]

Schéma de signature de Rabin : preuve de sécurité

- simulation de $H(M)$: tirer x aléatoire, et « programmer »

$$H(M) := x^2 \bmod n$$

- simulation de $\mathrm{Sign}(M)$: répondre x
- pour une requête $\widehat{M}$ aléatoirement choisie, répondre $H(\widehat{M}) = y$

n ⟶

⟶ (M^*, σ^*)

Schéma de signature de Rabin : preuve de sécurité

- simulation de $H(M)$: tirer x aléatoire, et « programmer »

$$H(M) := x^2 \bmod n$$

- simulation de Sign(M) : répondre x
- pour une requête $\widehat{M}$ aléatoirement choisie, répondre $H(\widehat{M}) = y$

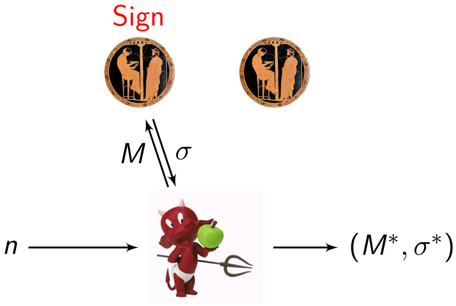


Schéma de signature de Rabin : preuve de sécurité

- simulation de $H(M)$: tirer x aléatoire, et « programmer »

$$H(M) := x^2 \bmod n$$

- simulation de $\mathrm{Sign}(M)$: répondre x
- pour une requête $\widehat{M}$ aléatoirement choisie, répondre $H(\widehat{M}) = y$

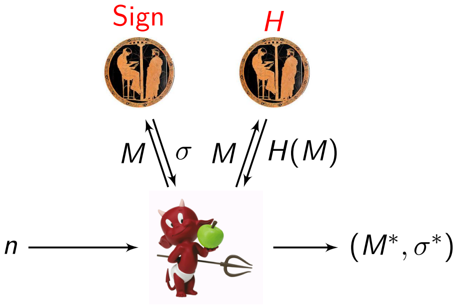


Schéma de signature de Rabin : preuve de sécurité

- simulation de $H(M)$: tirer x aléatoire, et « programmer »

$$H(M) := x^2 \bmod n$$

- simulation de $\mathrm{Sign}(M)$: répondre x
- pour une requête $\widehat{M}$ aléatoirement choisie, répondre $H(\widehat{M}) = y$

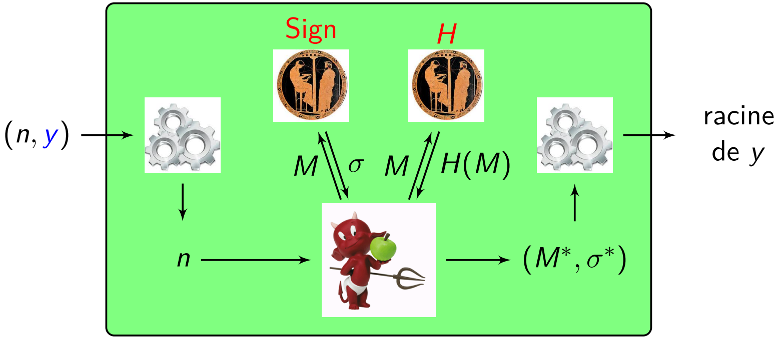


Schéma de signature de Rabin : preuve de sécurité

- simulation de $H(M)$: tirer x aléatoire, et « programmer »

$$H(M) := x^2 \bmod n$$

- simulation de $\mathrm{Sign}(M)$: répondre x
- pour une requête $\widehat{M}$ aléatoirement choisie, répondre $H(\widehat{M}) = y$

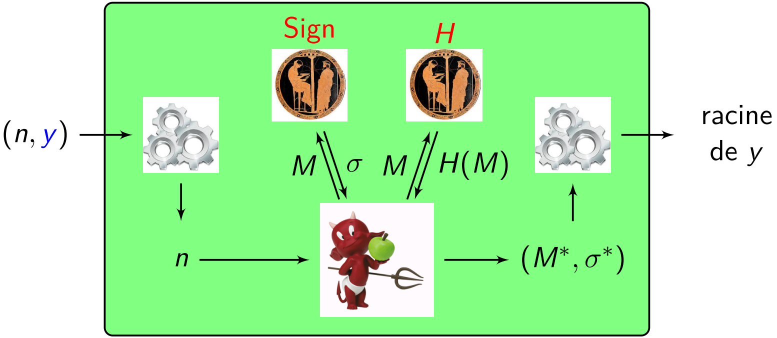


Schéma de signature de Rabin : preuve de sécurité

- simulation de $H(M)$: tirer x aléatoire, et « programmer »

$$H(M) := x^2 \bmod n$$

- simulation de $\mathrm{Sign}(M)$: répondre x
- pour une requête $\widehat{M}$ aléatoirement choisie, répondre $H(\widehat{M}) = y$

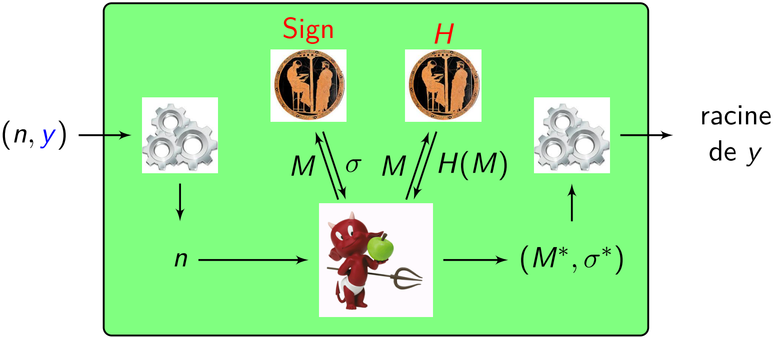


Schéma de signature de Rabin : preuve de sécurité

- simulation de $H(M)$: tirer x aléatoire, et « programmer »

$$H(M) := x^2 \bmod n$$

- simulation de $\mathrm{Sign}(M)$: répondre x
- pour une requête $\widehat{M}$ aléatoirement choisie, répondre $H(\widehat{M}) = y$

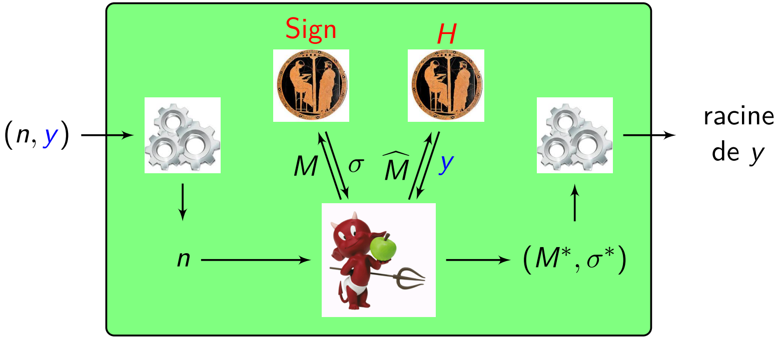


Schéma de signature de Rabin : preuve de sécurité

- si $M^* = \widehat{M}$, alors la signature forgée σ^* est une racine de y :

$$(\sigma^*)^2 = H(\widehat{M}) = y \bmod N$$

- se produit avec proba $1/q_h$, q_h nombre de requêtes à H
- $\exists$ une réduction perdant seulement un facteur q_s (nombre de requêtes de signature) [Cor00]

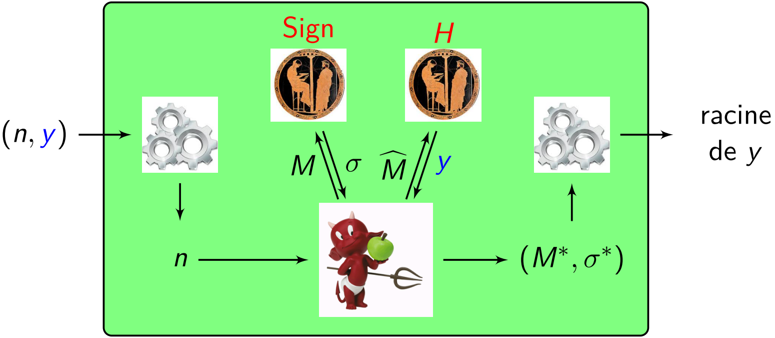


Schéma de signature de Rabin : preuve de sécurité

- si $M^* = \widehat{M}$, alors la signature forgée σ^* est une racine de y :

$$(\sigma^*)^2 = H(\widehat{M}) = y \bmod N$$

- se produit avec proba $1/q_h$, q_h nombre de requêtes à H
- $\exists$ une réduction perdant seulement un facteur q_s (nombre de requêtes de signature) [Cor00]

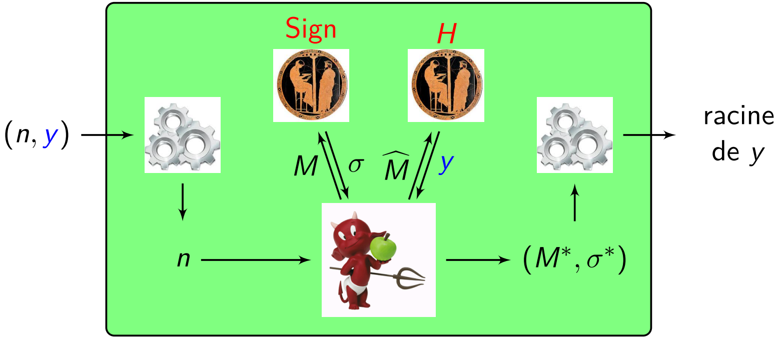
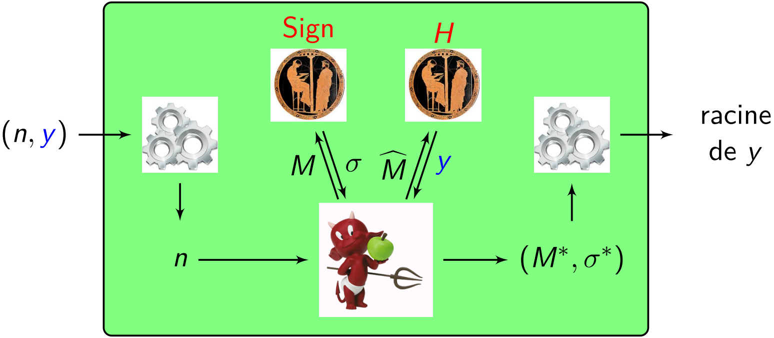


Schéma de signature de Rabin : preuve de sécurité

- si $M^* = \widehat{M}$, alors la signature forgée σ^* est une racine de y :

$$(\sigma^*)^2 = H(\widehat{M}) = y \bmod N$$

- se produit avec proba $1/q_h$, q_h nombre de requêtes à H
- $\exists$ une réduction perdant seulement un facteur q_s (nombre de requêtes de signature) [Cor00]



Problème de résiduosité quadratique décisionnel (RQD)

- p premier : symbole de Legendre

$$\left(\frac{y}{p}\right) = y^{\frac{p-1}{2}} \bmod p = 1 \text{ si } y \text{ est un } p\text{-résidu}$$
$$= -1 \text{ si } y \text{ est un } p\text{-non-résidu}$$

- $n = pq$, p, q premiers distincts : symbole de Jacobi

$$\left(\frac{y}{n}\right) = \left(\frac{y}{p}\right)\left(\frac{y}{q}\right)$$

$\left(\frac{y}{p}\right) \setminus \left(\frac{y}{q}\right)$	1	-1
1	1	-1
-1	-1	1

- problème : étant donné y tel que $\left(\frac{y}{n}\right) = 1$, y est-il un n-résidu ?
- pas de meilleure algorithme connu que factoriser n, puis calculer les deux symboles de Legendre
- mais pas de réduction de factorisation à RQD
  $\Rightarrow$ potentiellement plus facile que factoriser !

Problème de résiduosité quadratique décisionnel (RQD)

- p premier : symbole de Legendre

$$\left(\frac{y}{p}\right) = y^{\frac{p-1}{2}} \bmod p = 1 \text{ si } y \text{ est un } p\text{-résidu}$$
$$= -1 \text{ si } y \text{ est un } p\text{-non-résidu}$$

- $n = pq$, p, q premiers distincts : symbole de Jacobi

$$\left(\frac{y}{n}\right) = \left(\frac{y}{p}\right)\left(\frac{y}{q}\right)$$

$\left(\frac{y}{p}\right) \setminus \left(\frac{y}{q}\right)$	1	-1
1	1	-1
-1	-1	1

- problème : étant donné y tel que $\left(\frac{y}{n}\right) = 1$, y est-il un n-résidu ?
- pas de meilleure algorithme connu que factoriser n, puis calculer les deux symboles de Legendre
- mais pas de réduction de factorisation à RQD
  $\Rightarrow$ potentiellement plus facile que factoriser !

Problème de résiduosité quadratique décisionnel (RQD)

- p premier : symbole de Legendre

$$\left(\frac{y}{p}\right) = y^{\frac{p-1}{2}} \bmod p = 1 \text{ si } y \text{ est un } p\text{-résidu}$$
$$= -1 \text{ si } y \text{ est un } p\text{-non-résidu}$$

- $n = pq$, p, q premiers distincts : symbole de Jacobi

$$\left(\frac{y}{n}\right) = \left(\frac{y}{p}\right)\left(\frac{y}{q}\right)$$

$\left(\frac{y}{p}\right) \setminus \left(\frac{y}{q}\right)$	1	-1
1	1	-1
-1	-1	1

- problème : étant donné y tel que $\left(\frac{y}{n}\right) = 1$, y est-il un n-résidu ?
- pas de meilleure algorithme connu que factoriser n, puis calculer les deux symboles de Legendre
- mais pas de réduction de factorisation à RQD
  $\Rightarrow$ potentiellement plus facile que factoriser !

Problème de résiduosité quadratique décisionnel (RQD)

- p premier : symbole de Legendre

$$\left(\frac{y}{p}\right) = y^{\frac{p-1}{2}} \bmod p = 1 \text{ si } y \text{ est un } p\text{-résidu}$$
$$= -1 \text{ si } y \text{ est un } p\text{-non-résidu}$$

- $n = pq$, p, q premiers distincts : symbole de Jacobi

$$\left(\frac{y}{n}\right) = \left(\frac{y}{p}\right)\left(\frac{y}{q}\right)$$

$\left(\frac{y}{p}\right) \setminus \left(\frac{y}{q}\right)$	1	-1
1	1	-1
-1	-1	1

- problème : étant donné y tel que $\left(\frac{y}{n}\right) = 1$, y est-il un n-résidu ?
- pas de meilleure algorithme connu que factoriser n, puis calculer les deux symboles de Legendre
- mais pas de réduction de factorisation à RQD
  $\Rightarrow$ potentiellement plus facile que factoriser !

Problème de résiduosité quadratique décisionnel (RQD)

- p premier : symbole de Legendre

$$\left(\frac{y}{p}\right) = y^{\frac{p-1}{2}} \bmod p = 1 \text{ si } y \text{ est un } p\text{-résidu}$$
$$= -1 \text{ si } y \text{ est un } p\text{-non-résidu}$$

- $n = pq$, p, q premiers distincts : symbole de Jacobi

$$\left(\frac{y}{n}\right) = \left(\frac{y}{p}\right)\left(\frac{y}{q}\right)$$

$\left(\frac{y}{p}\right) \setminus \left(\frac{y}{q}\right)$	1	-1
1	1	-1
-1	-1	1

- problème : étant donné y tel que $\left(\frac{y}{n}\right) = 1$, y est-il un n-résidu ?
- pas de meilleure algorithme connu que factoriser n, puis calculer les deux symboles de Legendre
- mais pas de réduction de factorisation à RQD
  $\Rightarrow$ potentiellement plus facile que factoriser !

Schéma de chiffrement de Goldwasser-Micali

Chiffrement de Goldwasser-Micali

- clé privée : (p, q), premiers distincts
- clé publique : $n = pq$, y non-résidu tel que $\left(\frac{y}{n}\right) = 1$
- chiffrement d'un bit b :
  - tirer $r \leftarrow_{\$} \mathbb{Z}_n^*$ aléatoire,
  - calculer $c = r^2 y^b \bmod n$
- déchiffrement de c :
  - si c est un n-résidu $\Rightarrow b = 0$
  - si c est un n-non-résidu $\Rightarrow b = 1$

Sécurité :

- un attaquant qui intercepte un chiffré doit distinguer un résidu d'un non-résidu
  $\rightarrow$ il doit résoudre le problème RQD
- mais problème si accès à un oracle de déchiffrement

Schéma de chiffrement de Goldwasser-Micali

Chiffrement de Goldwasser-Micali

- clé privée : (p, q), premiers distincts
- clé publique : $n = pq$, y non-résidu tel que $\left(\frac{y}{n}\right) = 1$
- chiffrement d'un bit b :
  - tirer $r \leftarrow_{\$} \mathbb{Z}_n^*$ aléatoire,
  - calculer $c = r^2 y^b \bmod n$
- déchiffrement de c :
  - si c est un n-résidu $\Rightarrow b = 0$
  - si c est un n-non-résidu $\Rightarrow b = 1$

Sécurité :

- un attaquant qui intercepte un chiffré doit distinguer un résidu d'un non-résidu
  $\rightarrow$ il doit résoudre le problème RQD
- mais problème si accès à un oracle de déchiffrement

Schéma de chiffrement de Goldwasser-Micali

Chiffrement de Goldwasser-Micali

- clé privée : (p, q), premiers distincts
- clé publique : $n = pq$, y non-résidu tel que $\left(\frac{y}{n}\right) = 1$
- chiffrement d'un bit b :
  - tirer $r \leftarrow_{\$} \mathbb{Z}_n^*$ aléatoire,
  - calculer $c = r^2 y^b \bmod n$
- déchiffrement de c :
  - si c est un n-résidu $\Rightarrow b = 0$
  - si c est un n-non-résidu $\Rightarrow b = 1$

Sécurité :

- un attaquant qui intercepte un chiffré doit distinguer un résidu d'un non-résidu
  $\rightarrow$ il doit résoudre le problème RQD
- mais problème si accès à un oracle de déchiffrement

Schéma de chiffrement de Goldwasser-Micali

Chiffrement de Goldwasser-Micali

- clé privée : (p, q), premiers distincts
- clé publique : $n = pq$, y non-résidu tel que $\left(\frac{y}{n}\right) = 1$
- chiffrement d'un bit b :
    - tirer $r \leftarrow_{\$} \mathbb{Z}_n^*$ aléatoire,
    - calculer $c = r^2 y^b \bmod n$
- déchiffrement de c :
    - si c est un n-résidu $\Rightarrow b = 0$
    - si c est un n-non-résidu $\Rightarrow b = 1$

Sécurité :

- un attaquant qui intercepte un chiffré doit distinguer un résidu d'un non-résidu
  $\rightarrow$ il doit résoudre le problème RQD
- mais problème si accès à un oracle de déchiffrement

Schéma de chiffrement de Goldwasser-Micali

Chiffrement de Goldwasser-Micali

- clé privée : (p, q), premiers distincts
- clé publique : $n = pq$, y non-résidu tel que $\left(\frac{y}{n}\right) = 1$
- chiffrement d'un bit b :
  - tirer $r \leftarrow_{\$} \mathbb{Z}_n^*$ aléatoire,
  - calculer $c = r^2 y^b \bmod n$
- déchiffrement de c :
  - si c est un n-résidu $\Rightarrow b = 0$
  - si c est un n-non-résidu $\Rightarrow b = 1$

Sécurité :

- un attaquant qui intercepte un chiffré doit distinguer un résidu d'un non-résidu
  $\rightarrow$ il doit résoudre le problème RQD
- mais problème si accès à un oracle de déchiffrement

Schéma de chiffrement de Goldwasser-Micali

Chiffrement de Goldwasser-Micali

- clé privée : (p, q), premiers distincts
- clé publique : $n = pq$, y non-résidu tel que $\left(\frac{y}{n}\right) = 1$
- chiffrement d'un bit b :
  - tirer $r \leftarrow_{\$} \mathbb{Z}_n^*$ aléatoire,
  - calculer $c = r^2 y^b \bmod n$
- déchiffrement de c :
  - si c est un n-résidu $\Rightarrow b = 0$
  - si c est un n-non-résidu $\Rightarrow b = 1$

Sécurité :

- un attaquant qui intercepte un chiffré doit distinguer un résidu d'un non-résidu
  $\rightarrow$ il doit résoudre le problème RQD
- mais problème si accès à un oracle de déchiffrement

Sécurité d'un schéma de chiffrement à clé publique

Objectif de sécurité :

- but informel : l'attaquant ne doit pas pouvoir apprendre d'information (même partielle) sur le clair à partir du chiffré
- sécurité sémantique (SS) : pour tout adversaire $\mathcal{A}$, toute distribution $\mathcal{D}$ sur $\mathcal{M}$, toute fonction $I : \mathcal{M} \to \{0,1\}^*$, et tout prédicat $f : \mathcal{M} \to \{0,1\}$, il existe un attaquant $\mathcal{A}'$ tel que

$$\left|\Pr[\mathcal{A}(E(sk, M), I(M)) = f(M)] - \Pr[\mathcal{A}'(I(M)) = f(M)]\right| \leq \varepsilon$$

- indistinguabilité (IND) : l'attaquant choisit deux messages m_0, m_1, de même taille, et reçoit le chiffré de m_b, $b \in \{0,1\}$ aléatoire ; il doit deviner b
- SS et IND sont équivalents [GM84]

Sécurité d'un schéma de chiffrement à clé publique

Objectif de sécurité :

- but informel : l'attaquant ne doit pas pouvoir apprendre d'information (même partielle) sur le clair à partir du chiffré
- sécurité sémantique (SS) : pour tout adversaire $\mathcal{A}$, toute distribution $\mathcal{D}$ sur $\mathcal{M}$, toute fonction $I : \mathcal{M} \to \{0,1\}^*$, et tout prédicat $f : \mathcal{M} \to \{0,1\}$, il existe un attaquant $\mathcal{A}'$ tel que

$$\left|\Pr[\mathcal{A}(E(sk, M), I(M)) = f(M)] - \Pr[\mathcal{A}'(I(M)) = f(M)]\right| \leq \varepsilon$$

- indistinguabilité (IND) : l'attaquant choisit deux messages m_0, m_1, de même taille, et reçoit le chiffré de m_b, $b \in \{0,1\}$ aléatoire ; il doit deviner b
- SS et IND sont équivalents [GM84]

Sécurité d'un schéma de chiffrement à clé publique

Objectif de sécurité :

- but informel : l'attaquant ne doit pas pouvoir apprendre d'information (même partielle) sur le clair à partir du chiffré
- sécurité sémantique (SS) : pour tout adversaire $\mathcal{A}$, toute distribution $\mathcal{D}$ sur $\mathcal{M}$, toute fonction $I : \mathcal{M} \rightarrow \{0,1\}^*$, et tout prédicat $f : \mathcal{M} \rightarrow \{0,1\}$, il existe un attaquant $\mathcal{A}'$ tel que

$$\left|\Pr[\mathcal{A}(E(sk, M), I(M)) = f(M)] - \Pr[\mathcal{A}'(I(M)) = f(M)]\right| \leq \varepsilon$$

- indistinguabilité (IND) : l'attaquant choisit deux messages m_0, m_1, de même taille, et reçoit le chiffré de m_b, $b \in \{0,1\}$ aléatoire ; il doit deviner b
- SS et IND sont équivalents [GM84]

Sécurité d'un schéma de chiffrement à clé publique

Objectif de sécurité :

- but informel : l'attaquant ne doit pas pouvoir apprendre d'information (même partielle) sur le clair à partir du chiffré
- sécurité sémantique (SS) : pour tout adversaire $\mathcal{A}$, toute distribution $\mathcal{D}$ sur $\mathcal{M}$, toute fonction $I : \mathcal{M} \to \{0,1\}^*$, et tout prédicat $f : \mathcal{M} \to \{0,1\}$, il existe un attaquant $\mathcal{A}'$ tel que

$$\left|\Pr[\mathcal{A}(E(sk, M), I(M)) = f(M)] - \Pr[\mathcal{A}'(I(M)) = f(M)]\right| \leq \varepsilon$$

- indistinguabilité (IND) : l'attaquant choisit deux messages m_0, m_1, de même taille, et reçoit le chiffré de m_b, $b \in \{0,1\}$ aléatoire ; il doit deviner b
- SS et IND sont équivalents [GM84]

Sécurité d'un schéma de chiffrement à clé publique

Autres objectifs de sécurité :

- sens unique (SU) : étant donné un chiffré C, l'attaquant essaie de retrouver le message clair M correspondant
- non-malléabilité (NM) : étant donné le chiffré C d'un message M inconnu, l'attaquant essaie de créer le chiffré C' d'un message M' relié à M par une certaine fonction f [DDN00]

Sécurité d'un schéma de chiffrement à clé publique

Autres objectifs de sécurité :

- sens unique (SU) : étant donné un chiffré C, l'attaquant essaie de retrouver le message clair M correspondant
- non-malléabilité (NM) : étant donné le chiffré C d'un message M inconnu, l'attaquant essaie de créer le chiffré C' d'un message M' relié à M par une certaine fonction f [DDN00]

Sécurité d'un schéma de chiffrement à clé publique

Modèle d'attaque :

- attaque à clairs choisis (CPA) : l'attaquant ne connait que la clé publique (oracle de chiffrement inutile)
- attaque à chiffrés choisis non-adaptative (CCA1) : l'attaquant a accès à un oracle de déchiffrement avant de recevoir le défi (message chiffré)
- attaque à chiffrés choisis adaptative (CCA2) : l'attaquant a accès à un oracle de déchiffrement avant et après avoir reçu le défi (message chiffré)

$$\text{CPA} \Longleftarrow \text{CCA1} \Longleftarrow \text{CCA2}$$

$$\xrightarrow{\text{attaque plus forte}}$$

Sécurité d'un schéma de chiffrement à clé publique

Modèle d'attaque :

- attaque à clairs choisis (CPA) : l'attaquant ne connait que la clé publique (oracle de chiffrement inutile)
- attaque à chiffrés choisis non-adaptative (CCA1) : l'attaquant a accès à un oracle de déchiffrement avant de recevoir le défi (message chiffré)
- attaque à chiffrés choisis adaptative (CCA2) : l'attaquant a accès à un oracle de déchiffrement avant et après avoir reçu le défi (message chiffré)

$$\text{CPA} \Longleftarrow \text{CCA1} \Longleftarrow \text{CCA2}$$

$$\xrightarrow{\text{attaque plus forte}}$$

Sécurité d'un schéma de chiffrement à clé publique

Modèle d'attaque :

- attaque à clairs choisis (CPA) : l'attaquant ne connait que la clé publique (oracle de chiffrement inutile)
- attaque à chiffrés choisis non-adaptative (CCA1) : l'attaquant a accès à un oracle de déchiffrement avant de recevoir le défi (message chiffré)
- attaque à chiffrés choisis adaptative (CCA2) : l'attaquant a accès à un oracle de déchiffrement avant et après avoir reçu le défi (message chiffré)

$$\text{CPA} \Longleftarrow \text{CCA1} \Longleftarrow \text{CCA2}$$

$\xrightarrow{\text{attaque plus forte}}$

Sécurité d'un schéma de chiffrement à clé publique

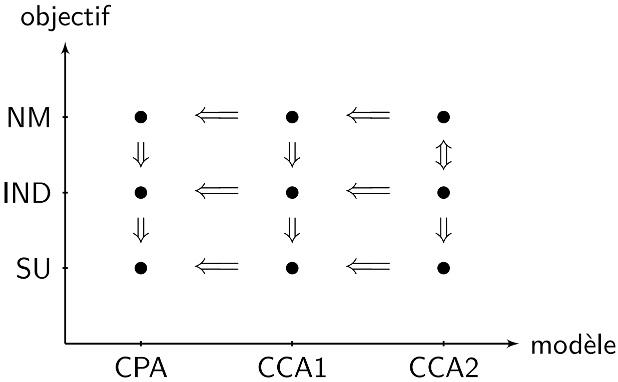
Modèle d'attaque :

- attaque à clairs choisis (CPA) : l'attaquant ne connait que la clé publique (oracle de chiffrement inutile)
- attaque à chiffrés choisis non-adaptative (CCA1) : l'attaquant a accès à un oracle de déchiffrement avant de recevoir le défi (message chiffré)
- attaque à chiffrés choisis adaptative (CCA2) : l'attaquant a accès à un oracle de déchiffrement avant et après avoir reçu le défi (message chiffré)

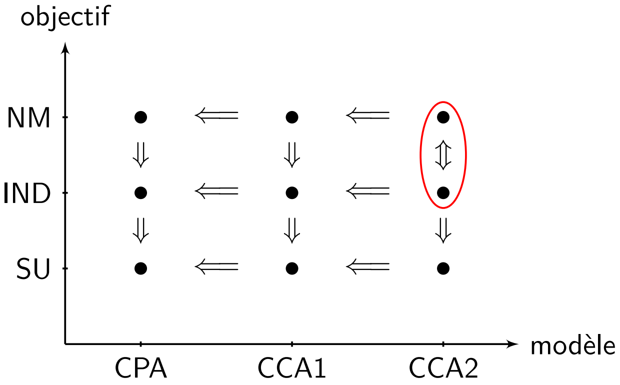
$$\text{CPA} \Longleftarrow \text{CCA1} \Longleftarrow \text{CCA2}$$

$$\xrightarrow{\text{attaque plus forte}}$$

Relation entre les notions de sécurité

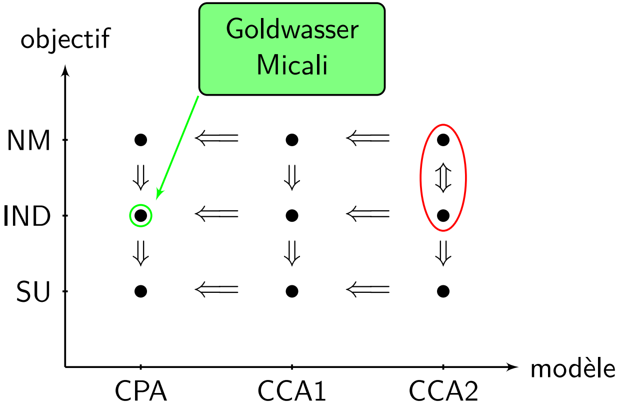


Relation entre les notions de sécurité



IND-CCA2 = Indistinguabilité contre les attaques à chiffrés choisis adaptatives

Relation entre les notions de sécurité



IND-CCA2 = Indistinguabilité contre les attaques à chiffrés choisis adaptatives

Cryptosystèmes asymétriques modernes

Principaux cryptosystèmes possédant une preuve de sécurité :

1. chiffrement :
   - RSA-OAEP (PKCS #1 v2.0)
     $\Rightarrow$ IND-CCA2 sous l'hypothèse RSA + modèle oracle aléatoire
   - DHIES (variante d'ElGamal)
     $\Rightarrow$ IND-CCA2 sous l'hypothèse Diffie-Hellman + modèle oracle aléatoire
2. signature :
   - RSA-PSS (PKCS #1 v2.1)
     $\Rightarrow$ IF-Ex-AMC sous l'hypothèse RSA + modèle oracle aléatoire (réduction fine)
   - (EC)DSA
     $\Rightarrow$ repose sur le log discret mais pas de preuve de sécurité
   - (EC)-Schnorr (variante de DSA)
     $\Rightarrow$ IF-Ex-AMC sous l'hypothèse log discret + modèle oracle aléatoire

Cryptosystèmes asymétriques modernes

Principaux cryptosystèmes possédant une preuve de sécurité :

1. chiffrement :
   - RSA-OAEP (PKCS #1 v2.0)
     $\Rightarrow$ IND-CCA2 sous l'hypothèse RSA + modèle oracle aléatoire
   - DHIES (variante d'ElGamal)
     $\Rightarrow$ IND-CCA2 sous l'hypothèse Diffie-Hellman + modèle oracle aléatoire
2. signature :
   - RSA-PSS (PKCS #1 v2.1)
     $\Rightarrow$ IF-Ex-AMC sous l'hypothèse RSA + modèle oracle aléatoire (réduction fine)
   - (EC)DSA
     $\Rightarrow$ repose sur le log discret mais pas de preuve de sécurité
   - (EC)-Schnorr (variante de DSA)
     $\Rightarrow$ IF-Ex-AMC sous l'hypothèse log discret + modèle oracle aléatoire

Sécurité calculatoire en cryptographie symétrique

Comment dépasser le théorème d'impossibilité de Shannon ?

Sécurité calculatoire en cryptographie symétrique

Comment dépasser le théorème d'impossibilité de Shannon ?

Fonction à sens unique (e.g. log discret)

Sécurité calculatoire en cryptographie symétrique

Comment dépasser le théorème d'impossibilité de Shannon ?

Fonction à sens unique (e.g. log discret)

$\downarrow$ Håstad *et al.* [HILL99]

Générateur pseudo-aléatoire

Sécurité calculatoire en cryptographie symétrique

Comment dépasser le théorème d'impossibilité de Shannon ?

Fonction à sens unique (e.g. log discret)

↓ Håstad *et al.* [HILL99]

Générateur pseudo-aléatoire

↓ Goldreich *et al.* [GGM86]

Fonction pseudo-aléatoire

Sécurité calculatoire en cryptographie symétrique

Comment dépasser le théorème d'impossibilité de Shannon ?

Fonction à sens unique (e.g. log discret)

↓ Håstad *et al.* [HILL99]

Générateur pseudo-aléatoire

↓ Goldreich *et al.* [GGM86]

Fonction pseudo-aléatoire

↓ Luby-Rackoff [LR88]

Permutation pseudo-aléatoire
(Chiffrement par blocs)

Sécurité calculatoire en cryptographie symétrique

Comment dépasser le théorème d'impossibilité de Shannon ?

Fonction à sens unique (e.g. log discret)

↓ Håstad *et al.* [HILL99]

Générateur pseudo-aléatoire

↓ Goldreich *et al.* [GGM86]

Fonction pseudo-aléatoire

↓ Luby-Rackoff [LR88]

Permutation pseudo-aléatoire
(Chiffrement par blocs)

Constructions trop complexes en pratique $\Rightarrow$ AES

Modes opératoires

Hypothèse : AES (par ex.) est un chiffrement par bloc « sûr »

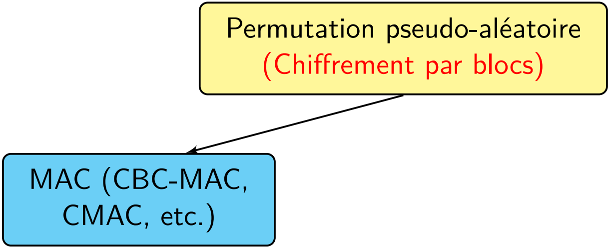
Modes opératoires

Hypothèse : AES (par ex.) est un chiffrement par bloc « sûr »

Permutation pseudo-aléatoire
(Chiffrement par blocs)

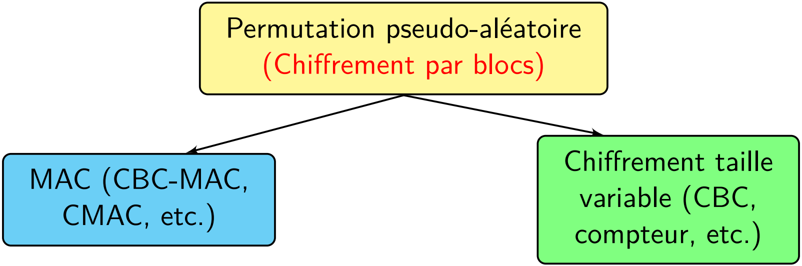
Modes opératoires

Hypothèse : AES (par ex.) est un chiffrement par bloc « sûr »



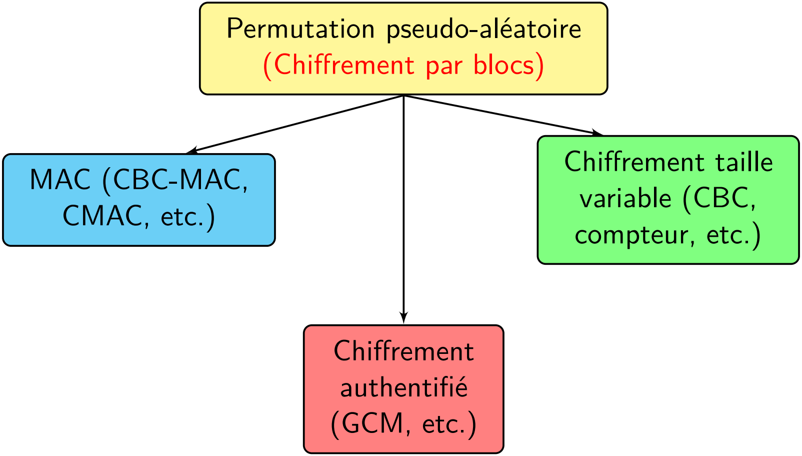
Modes opératoires

Hypothèse : AES (par ex.) est un chiffrement par bloc « sûr »



Modes opératoires

Hypothèse : AES (par ex.) est un chiffrement par bloc « sûr »



Conclusion

Une preuve de sécurité n'est pas une garantie absolue :

- attaquants en dehors du modèle (canaux cachés, etc.)
- interprétation « pratique » de l'objectif de sécurité parfois spécieuse
- repose sur des hypothèses qui peuvent s'avérer fausses (générateur aléatoire parfait, etc.)

Perspectives

- élargissement du modèle d'attaque (prise en compte des canaux cachés, etc.)
- affaiblissement des hypothèses (problème difficile plus fort, suppression du modèle de l'oracle aléatoire, etc.)
- preuves formelles assistées (EasyCrypt, CryptoVerif)

Conclusion

Une preuve de sécurité n'est pas une garantie absolue :

- attaquants en dehors du modèle (canaux cachés, etc.)
- interprétation « pratique » de l'objectif de sécurité parfois spécieuse
- repose sur des hypothèses qui peuvent s'avérer fausses (générateur aléatoire parfait, etc.)

Perspectives

- élargissement du modèle d'attaque (prise en compte des canaux cachés, etc.)
- affaiblissement des hypothèses (problème difficile plus fort, suppression du modèle de l'oracle aléatoire, etc.)
- preuves formelles assistées (EasyCrypt, CryptoVerif)

Conclusion

Une preuve de sécurité n'est pas une garantie absolue :

- attaquants en dehors du modèle (canaux cachés, etc.)
- interprétation « pratique » de l'objectif de sécurité parfois spécieuse
- repose sur des hypothèses qui peuvent s'avérer fausses (générateur aléatoire parfait, etc.)

Perspectives

- élargissement du modèle d'attaque (prise en compte des canaux cachés, etc.)
- affaiblissement des hypothèses (problème difficile plus fort, suppression du modèle de l'oracle aléatoire, etc.)
- preuves formelles assistées (EasyCrypt, CryptoVerif)

Conclusion

Une preuve de sécurité n'est pas une garantie absolue :

- attaquants en dehors du modèle (canaux cachés, etc.)
- interprétation « pratique » de l'objectif de sécurité parfois spécieuse
- repose sur des hypothèses qui peuvent s'avérer fausses (générateur aléatoire parfait, etc.)

Perspectives

- élargissement du modèle d'attaque (prise en compte des canaux cachés, etc.)
- affaiblissement des hypothèses (problème difficile plus fort, suppression du modèle de l'oracle aléatoire, etc.)
- preuves formelles assistées (EasyCrypt, CryptoVerif)

Conclusion

Une preuve de sécurité n'est pas une garantie absolue :

- attaquants en dehors du modèle (canaux cachés, etc.)
- interprétation « pratique » de l'objectif de sécurité parfois spécieuse
- repose sur des hypothèses qui peuvent s'avérer fausses (générateur aléatoire parfait, etc.)

Perspectives

- élargissement du modèle d'attaque (prise en compte des canaux cachés, etc.)
- affaiblissement des hypothèses (problème difficile plus fort, suppression du modèle de l'oracle aléatoire, etc.)
- preuves formelles assistées (EasyCrypt, CryptoVerif)

Conclusion

Une preuve de sécurité n'est pas une garantie absolue :

- attaquants en dehors du modèle (canaux cachés, etc.)
- interprétation « pratique » de l'objectif de sécurité parfois spécieuse
- repose sur des hypothèses qui peuvent s'avérer fausses (générateur aléatoire parfait, etc.)

Perspectives

- élargissement du modèle d'attaque (prise en compte des canaux cachés, etc.)
- affaiblissement des hypothèses (problème difficile plus fort, suppression du modèle de l'oracle aléatoire, etc.)
- preuves formelles assistées (EasyCrypt, CryptoVerif)

The end...

Merci de votre attention !

Commentaires ou questions ?

References I

- Mihir Bellare and Phillip Rogaway. Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.
- Jean-Sébastien Coron. On the Exact Security of Full Domain Hash. In Mihir Bellare, editor, *Advances in Cryptology - CRYPTO 2000*, volume 1880 of *LNCS*, pages 229–235. Springer, 2000.
- Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable Cryptography. *SIAM Journal on Computing*, 30(2):391–437, 2000.
- Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.
- Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.
- Shafi Goldwasser and Silvio Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

References II

Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIAM J. Comput.*, 17(2):281–308, 1988.

Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A Pseudorandom Generator from any One-way Function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

Michael Luby and Charles Rackoff. How to Construct Pseudorandom Permutations from Pseudorandom Functions. *SIAM Journal on Computing*, 17(2):373–386, 1988.

Michael O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report 212, MIT Laboratory for Computer Science, 1979.

Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

References III

Claude Shannon. Communication Theory of Secrecy Systems. *Bell System Technical Journal*, 28(4):656–715, 1949.