Cours 2. Chiffrement symétrique

HAI709I – Cryptographie

Bruno Grenet

Université de Montpellier – Faculté des Sciences

Théorie de l'information ou théorie de la complexité

Chiffrement de Vernam

- Inconditionnellement sûr $\rightarrow$ garantie de *théorie de l'information*
- Aucune hypothèse sur la puissance de calcul de l'attaquant
- Mais irréaliste en pratique (taille des clefs, etc.)

Théorie de l'information ou théorie de la complexité

Chiffrement de Vernam

- Inconditionnellement sûr $\rightarrow$ garantie de *théorie de l'information*
- Aucune hypothèse sur la puissance de calcul de l'attaquant
- Mais irréaliste en pratique (taille des clefs, etc.)

Sécurité calculatoire

1. Sécurité garantie uniquement face à des attaquants efficaces qui calculent pendant un temps raisonnable
2. L'attaquant peut potentiellement réussir, mais avec une probabilité négligeable

Théorie de l'information ou théorie de la complexité

Chiffrement de Vernam

- Inconditionnellement sûr $\rightarrow$ garantie de *théorie de l'information*
- Aucune hypothèse sur la puissance de calcul de l'attaquant
- Mais irréaliste en pratique (taille des clefs, etc.)

Sécurité calculatoire

1. Sécurité garantie uniquement face à des attaquants efficaces qui calculent pendant un temps raisonnable
2. L'attaquant peut potentiellement réussir, mais avec une probabilité négligeable

Deux approches

- Approche concrète
- Approche asymptotique

L'approche concrète

Un schéma est dit (t, ε)-*sûr* si un attaquant disposant d'un temps de calcul $\leq t$ a une probabilité $\leq \varepsilon$ de réussir à casser le schéma.

L'approche concrète

Un schéma est dit (t, ε)-*sûr* si un attaquant disposant d'un temps de calcul $\leq t$ a une probabilité $\leq \varepsilon$ de réussir à casser le schéma.

- En général : temps = nombre de cycles CPU
- « La probabilité de casser tel schéma en 2^{80} cycles est $\leq 1/2^{60}$ »

Quelques chiffres

- Clefs de taille $n \to |\mathcal{K}| = 2^n$; force brute $\to$ proba. $\propto t/2^n$
- Processeur 16 cœurs à 4 GHz : 2^{64} cycles en $2^{64}/(16 \times 4 \cdot 10^9)$ s $\simeq 9$ ans
- Si plusieurs processeurs… plus rapide ! Clefs recommandées de taille 128
- Si proba. $1/2^{60}$ de réussir *chaque seconde* : 1 réussite en ≥ 30 milliards d'années !

L'approche concrète

Un schéma est dit (t, ε)-*sûr* si un attaquant disposant d'un temps de calcul $\leq t$ a une probabilité $\leq \varepsilon$ de réussir à casser le schéma.

- En général : temps = nombre de cycles CPU
- « La probabilité de casser tel schéma en 2^{80} cycles est $\leq 1/2^{60}$ »

Quelques chiffres

- Clefs de taille $n \to |\mathcal{K}| = 2^n$; force brute $\to$ proba. $\propto t/2^n$
- Processeur 16 cœurs à 4 GHz : 2^{64} cycles en $2^{64}/(16 \times 4 \cdot 10^9)$ s $\simeq 9$ ans
- Si plusieurs processeurs… plus rapide ! Clefs recommandées de taille 128
- Si proba. $1/2^{60}$ de réussir *chaque seconde* : 1 réussite en ≥ 30 milliards d'années !

- Important en pratique
- Mais difficile à estimer : évolution du matériel, proba. pour $t = 2$ ans ne dit rien sur $t = 10$ ans, etc.

L'approche asymptotique

- Fondée sur la théorie de la complexité
- Introduit un *paramètre de sécurité* n (entier)

Sécurité asymptotique

1. Attaquant efficace : algorithme probabiliste de complexité polynomiale en n
2. Probabilité négligeable : pour tout polynôme p, proba. $< 1/p(n)$ pour n assez grand

L'approche asymptotique

- Fondée sur la théorie de la complexité
- Introduit un *paramètre de sécurité* n (entier)

Sécurité asymptotique

1. Attaquant efficace : algorithme probabiliste de complexité polynomiale en n
2. Probabilité négligeable : pour tout polynôme p, proba. $< 1/p(n)$ pour n assez grand

L'approche asymptotique

- Fondée sur la théorie de la complexité
- Introduit un *paramètre de sécurité* n (entier)

Sécurité asymptotique

1. Attaquant efficace : algorithme probabiliste de complexité polynomiale en n
2. Probabilité négligeable : pour tout polynôme p, proba. $< 1/p(n)$ pour n assez grand

Quelques chiffres

- Si un attaquant peut casser un schéma avec proba. $2^{40}/2^n$ en n^3 minutes,
  - si $n = 40$, proba. 1 en temps $40^3 \simeq 6$ semaines : oups !
  - si $n = 50$, proba. $1/1000$ en 3 mois : acceptable ?
  - si $n = 500$, proba. $1/2^{460}$ en 200 ans : sans doute suffisant…
- Temps pour chiffrer $10^6 n^2$ cycles ; attaquant en $10^8 n^4$ cycles avec proba. $1/2^{n/2}$
  - Avec 2GHz et $n = 80 \rightarrow$ chiffrement : 3, 2s ; attaque : 3 semaines ; proba. $1/2^{40}$
  - Avec 8GHz et $n = 160 \rightarrow$ chiffrement : 3, 2s. ; attaque : 13 semaines ; proba. $1/2^{80}$

Définition des asymptotiques

Un schéma de chiffrement est (asymptotiquement) sûr si un attaquant polynomial probabiliste a une probabilité négligeable de réussir à casser le schéma.

Polynomial

$f(n) = \text{poly}(n)$ s'*il existe* c et e tq $f(n) \leq cn^e$ pour n suffisamment grand

- $\log n$, $n \log n$, $\sqrt{n}$, n^{1000}, … sont $\text{poly}(n)$
- 2^n, $2^{\sqrt{n}}$, $n^{\log n}$, $(\log n)^{\log n}$, … ne sont pas $\text{poly}(n)$

Négligeable

$g(n) = \text{negl}(n)$ si *pour tout* c et e, $g(n) < 1/cn^e$ pour n suffisamment grand

- $1/2^n$, $1/2^{\sqrt{n}}$, $1/n^{\log n}$, $1/(\log n)^{\log n}$… sont $\text{negl}(n)$
- $1/\log n$, $1/n \log n$, $1/\sqrt{n}$, $1/n^{1000}$, … ne sont pas $\text{negl}(n)$

- $g(n) = \text{negl}(n) \iff 1/g(n) \neq \text{poly}(n)$
- $\text{negl}_1(n) + \text{negl}_2(n)$ et $\text{negl}_1(n) \times \text{poly}(n)$ sont $\text{negl}(n)$

1. Sécurité calculatoire

2. Sécurité calculatoire du chiffrement

3. EAV-Sécurité et générateurs pseudo-aléatoires

Formalisation du chiffrement à clef privée

Définition

Un schéma de chiffrement à clef privée est un triplet d'algorithmes probabilistes polynomiaux $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ où

- Gen prend en entrée 1^n (paramètre de sécurité en unaire) et renvoie une clef k
- Enc prend en entrée k et un message $m \in \{0,1\}^\star$ et renvoie un chiffré c
- Dec prend en entrée k et un chiffré c et renvoie un message m ou une erreur

tels que pour tout $k \leftarrow \mathrm{Gen}(1^n)$ et tout $c \leftarrow \mathrm{Enc}_k(m)$, $\mathrm{Dec}_k(c) = m$.

Formalisation du chiffrement à clef privée

Définition

Un schéma de chiffrement à clef privée est un triplet d'algorithmes probabilistes polynomiaux $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ où

- Gen prend en entrée 1^n (paramètre de sécurité en unaire) et renvoie une clef k
- Enc prend en entrée k et un message $m \in \{0,1\}^\star$ et renvoie un chiffré c
- Dec prend en entrée k et un chiffré c et renvoie un message m ou une erreur

tels que pour tout $k \leftarrow \mathrm{Gen}(1^n)$ et tout $c \leftarrow \mathrm{Enc}_k(m)$, $\mathrm{Dec}_k(c) = m$.

Remarques

- On suppose toujours que $|k| \geq n$
- Si pour $k \leftarrow \mathrm{Gen}(1^n)$, Enc_k n'est défini que sur $\{0,1\}^{\ell(n)}$, $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ est un schéma de chiffrement à longueur fixée $\ell(n)$.

EAV-sécurité

Modèle de sécurité

- Modèle de menace : capacités (puissance de calcul, …) de l'attaquant
- Objectif de sécurité : qu'entend-on par *casser* le système ?

EAV-sécurité

Modèle de sécurité

- Modèle de menace : capacités (puissance de calcul, …) de l'attaquant
- Objectif de sécurité : qu'entend-on par *casser* le système ?

Modèle d'EAV-sécurité

- Menace : oreille indiscrète (*eavesdropper*)
  - observe *un* chiffré
  - algorithme polynomial probabiliste
- Objectif : aucun bit d'information supplémentaire
  - sécurité sémantique : difficile
  - indistinguabilité : relier un chiffré à un message clair parmi deux

EAV-sécurité

Modèle de sécurité

- Modèle de menace : capacités (puissance de calcul, …) de l'attaquant
- Objectif de sécurité : qu'entend-on par *casser* le système ?

Modèle d'EAV-sécurité

- Menace : oreille indiscrète (*eavesdropper*)
  - observe *un* chiffré
  - algorithme polynomial probabiliste
- Objectif : aucun bit d'information supplémentaire
  - sécurité sémantique : difficile
  - indistinguabilité : relier un chiffré à un message clair parmi deux

Remarque

- Un seul chiffré : autres notions de sécurité plus complexes

L'expérience d'EAV-sécurité

Entrée : Paramètre de sécurité 1^n

1. Attaquant : produit deux messages m_0 et m_1 de même taille
2. Protocole : $k \leftarrow \mathrm{Gen}(1^n)$; $b \leftarrow_R \{0,1\}$; $c \leftarrow \mathrm{Enc}_k(m_b)$
3. Attaquant : étant donné c, renvoie un bit b'

Succès de l'attaquant si $b' = b$

Définition

(Gen, Enc, Dec) est EAV-sûr si pour tout attaquant polynomial probabiliste (APP),
$\Pr[b' = b] \leq \frac{1}{2} + \mathrm{negl}(n)$

L'expérience d'EAV-sécurité

Entrée : Paramètre de sécurité 1^n

1. Attaquant : produit deux messages m_0 et m_1 de même taille
2. Protocole : $k \leftarrow \mathrm{Gen}(1^n)$; $b \leftarrow_R \{0,1\}$; $c \leftarrow \mathrm{Enc}_k(m_b)$
3. Attaquant : étant donné c, renvoie un bit b'

Succès de l'attaquant si $b' = b$

Définition

(Gen, Enc, Dec) est EAV-sûr si pour tout attaquant polynomial probabiliste (APP),
$\Pr[b' = b] \leq \frac{1}{2} + \mathrm{negl}(n)$

Remarque

- La longueur du message clair n'est pas cachée

Aucun bit n'est dévoilé

Théorème

Si $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *est de longueur fixée* $\ell(n)$ *et EAV-sûr, alors pour tout* APP $\mathcal{A}$, $n > 0$ *et* $0 \leq i < \ell(n)$, $\Pr\left[\mathcal{A}(1^n, \mathsf{Enc}_k(m)) = m_{[i]}\right] \leq \frac{1}{2} + \mathsf{negl}(n)$, *où* k *et* m *sont uniformes.*

$\mathcal{A}'$: 1. $\mathcal{A}'$ choisit $m_0 \in \Pi_0$ et $m_1 \in \Pi_1$ tq $m_{0[i]} = 0$ et $m_{1[i]} = 1$ uniformément où $\Pi_0 = \{m \in \Pi : m_{[i]} = 0\}$ $\Pi_1 = \{m \in \Pi : m_{[i]} = 1\}$

2. Protocole choisit b et calcule $c \leftarrow \mathsf{Enc}_k(m_b)$

3. $\mathcal{A}'$ utilise $\mathcal{A}$ ↝ Calcule $b' \leftarrow \mathcal{A}(1^n, c)$

$\mathcal{A}'$ répond 0 si $b' = 0$ et 1 si $b' = 1$

$\forall m \in \{0,1\}^{\ell(n)}$ $\Pr\left[m_b = m\right] = 1/2^{\ell(n)}$

Puisque m_b est uniforme, $\Pr\left[\mathcal{A}' \text{ succès}\right] = \Pr\left[\mathcal{A} \text{ succès}\right]$.

Généralisation à plusieurs chiffrés

Entrée : Paramètre de sécurité 1^n

1. Attaquant : produit deux **listes** de messages $\vec{m}_0 = (m_{0,1}, \ldots, m_{0,t})$ et $\vec{m}_1 = (m_{1,1}, \ldots, m_{1,t})$ de mêmes tailles
2. Protocole : $k \leftarrow \mathrm{Gen}(1^n)$; $b \leftarrow_R \{0,1\}$; $c_i \leftarrow \mathrm{Enc}_k(m_{b,i})$ pour $1 \leq i \leq t$
3. Attaquant : étant donné les c_i, renvoie un bit b'

Succès de l'attaquant si $b' = b$

Définition

$(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ est EAV-sûr **pour les chiffrements multiples** si pour tout APP,
$\Pr[b' = b] \leq \frac{1}{2} + \mathrm{negl}(n)$

1. Sécurité calculatoire

2. Sécurité calculatoire du chiffrement

3. EAV-Sécurité et générateurs pseudo-aléatoires

Comment construire un chiffrement EAV-sûr ?

Idée

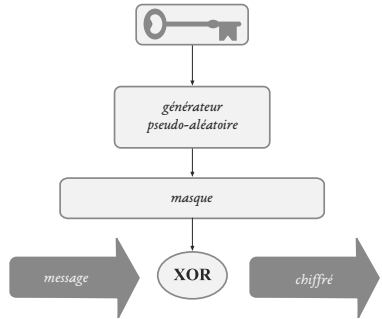
- Le chiffre de Vernam est *sûr* si on tire une clef aléatoire aussi longue que le message

Difficultés

- Il faut une clef aussi longue que le message
- Elle doit être aléatoire

Solution

- Produire une clef *pseudo-aléatoire*
- Utiliser une petite *graîne* comme clef



Source : J. Katz, Y. Lindell. Introduction to modern cryptography. 3rd ed, CRC Press, 2021. (modif.)

Générateur pseudo-aléatoire

Algorithme déterministe, basé sur une graîne aléatoire, qui produit des bits qui semblent aléatoires à n'importe quel algorithme polynomial probabiliste

- La suite de bits doit être *indistinguable* d'une suite aléatoire pour un algo. poly. proba.
  $\rightarrow$ comportement identique pour une suite aléatoire ou pseudo-aléatoire

Générateur pseudo-aléatoire

Algorithme déterministe, basé sur une graîne aléatoire, qui produit des bits qui semblent aléatoires à n'importe quel algorithme polynomial probabiliste

- La suite de bits doit être *indistinguable* d'une suite aléatoire pour un algo. poly. proba.
  $\rightarrow$ comportement identique pour une suite aléatoire ou pseudo-aléatoire

Définition

Un **générateur pseudo-aléatoire** G est un algorithme déterministe tel que

1. Expansion Pour $s \in \{0,1\}^n$, $G(s) \in \{0,1\}^{\ell(n)}$ où $\ell(n) > n$
2. Pseudo-aléa Pour tout algo. poly. proba. D, (distingueur)

$$\left|\Pr_{s\in\{0,1\}^n}[D(G(s)) = 1] - \Pr_{r\in\{0,1\}^{\ell(n)}}[D(r) = 1]\right| \leq \mathrm{negl}(n)$$

Remarques sur les générateurs pseudo-aléatoires

Un contre-exemple

- $G(s) = s\| \oplus_{i=0}^{n-1} s_i \to \ell(n) = n+1$
- Distingueur D : $D(r) = 1$ ssi $r_n = \oplus_{i<n} r_i$

Remarques sur les générateurs pseudo-aléatoires

Un contre-exemple

- $G(s) = s\| \oplus_{i=0}^{n-1} s_i \to \ell(n) = n+1$
- Distingueur $D : D(r) = 1$ ssi $r_n = \oplus_{i<n} r_i$

Remarques

- $\ell(n)$ doit être $\gg n$ pour éviter les attaques brute-force
- $G : \{0,1\}^n \to \{0,1\}^{\ell(n)}$ déterministe $\to 2^n$ chaînes peuvent être produites
- Sans borne de temps, trivial de distinguer (vrai) aléa et pseudo-aléa

Remarques sur les générateurs pseudo-aléatoires

Un contre-exemple

- $G(s) = s \| \oplus_{i=0}^{n-1} s_i \to \ell(n) = n+1$
- Distingueur $D : D(r) = 1$ ssi $r_n = \oplus_{i<n} r_i$

Remarques

- $\ell(n)$ doit être $\gg n$ pour éviter les attaques brute-force
- $G : \{0,1\}^n \to \{0,1\}^{\ell(n)}$ déterministe $\to 2^n$ chaînes peuvent être produites
- Sans borne de temps, trivial de distinguer (vrai) aléa et pseudo-aléa

Existe-t-il des générateurs pseudo-aléatoires ?

- En théorie oui, sous certaines hypothèses de complexité (*fonctions à sens unique*)
- En pratique oui, mais sans démonstration formelle (*chiffrement par flot*)

EAV-sécurité grâce à un générateur pseudo-aléatoire

Définition

Étant donné un générateur pseudo-aléatoire G d'expansion $\ell(n)$, on définit un schéma de chiffrement à longueur fixée $\ell(n)$:

- $\mathrm{Gen}(1^n)$ choisit uniformément $k \in \{0,1\}^n$
- $\mathrm{Enc}_k(m) = G(k) \oplus m$
- $\mathrm{Dec}_k(m) = G(k) \oplus c$

EAV-sécurité grâce à un générateur pseudo-aléatoire

Définition

Étant donné un générateur pseudo-aléatoire d'expansion $\ell(n)$, on définit un schéma de chiffrement à longueur fixée $\ell(n)$:

- $\mathrm{Gen}(1^n)$ choisit uniformément $k \in \{0,1\}^n$
- $\mathrm{Enc}_k(m) = G(k) \oplus m$
- $\mathrm{Dec}_k(m) = G(k) \oplus c$

Remarques

- $\mathcal{K} = \{0,1\}^n$, $\mathcal{M} = \mathcal{C} = \{0,1\}^{\ell(n)}$
- $\mathrm{Dec}_k(\mathrm{Enc}_k(m)) = m$ car G est déterministe

Preuve de sécurité

Théorème

Si G est un générateur pseudo-aléatoire, le schéma de chiffrement est EAV-sûr pour les messages de longueur $\ell(n)$.

On suppose un attaquant A_0. On construit D qui sur une entrée $w \in \{0,1\}^{\ell(n)}$, détermine si w est aléatoire ou pseudo-aléatoire.

D :
1. D utilise A_0 pour créer m_0 et m_1
2. D tire un bit aléatoire b et calcule $c = w \oplus m_b$
3. Calcule $b' = A_0(1^n, c)$
4. D renvoie 1 si $b = b'$, 0 sinon.

- Si $w = G(k)$, D simule l'expérience d'EAV-sécurité $\rightsquigarrow$ proba[succès] $> 1/2 + \text{negl}(n)$
- Sinon, $c = w \oplus m_b$ est uniforme dans $\{0,1\}^{\ell(n)}$.

$$P_r[A_0(1^n, c) = b] = \frac{1}{2}\left(P_r[A_0(1^n, c) = 0] + P_r[A_0(1^n, c) = 1]\right) = 1/2.$$

Sécurité face aux attaques à clair choisi (CPA-sécurité)

Attaquant plus fort

- L'attaquant ne peut pas chiffrer lui-même…
- … mais imaginons qu'il force le protocole à chiffrer des messages de son choix
- Simule aussi les informations (partielles) dont dispose l'attaquant

Idée de la définition de CPA-sécurité

- Similaire à EAV-sécurité
- À tout moment, l'attaquant peut demander le chiffrement de messages $\neq m_0, m_1$

Comment construire la CPA-sécurité ?

Basé sur des *permutations pseudo-aléatoires*

- $F : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^\ell$
- Pour tout k, F_k est une permutation
- Pseudo-aléa : indistinguable par un algo. poly. proba.

Plus loin : attaques à chiffrés choisis

- CCA sécurité : l'attaquant peut demander le *déchiffrement* de chiffrés de son choix

En pratique

Comment construire des générateurs et permutations pseudo-aléatoires ?

Chiffrement par flot

- Implantation des générateurs pseudo-aléatoires
- Plus flexibles : suites *infinies* de bits
- Pas de preuves formelles mais testés empiriquement

Chiffrement par bloc

- Implantation des permutations pseudo-aléatoires
- Pas plus de preuves !
- Modes d'opérations : comment chiffrer plusieurs blocs à la suite ?

Conclusion

Points principaux

- Principe de la sécurité calculatoire
- Définition d'EAV-sécurité
- Définition de générateurs pseudo-aléatoires
- EAV-sécurité basé sur les générateurs pseudo-aléatoires

Pour la culture

- Notions plus fortes de sécurité

Anticipation

- Chiffrement par flot et blocs (cours 3)
- Existence théorique de générateurs pseudo-aléatoires (cours 5)