Cours 4. Fonctions de hachage (cryptographiques)

HAI709I – Cryptographie

Bruno Grenet

Université de Montpellier – Faculté des Sciences

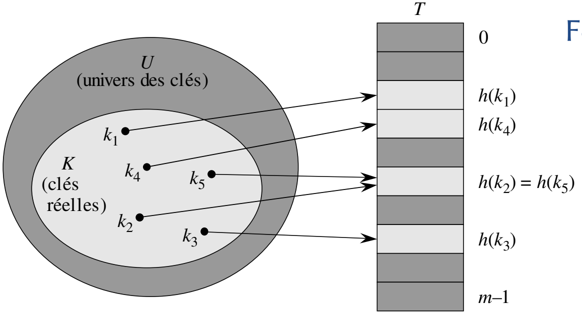
1. Fonctions de hachage et leur sécurité

2. Construction de Merkle-Damgård

3. Attaques : le paradoxe des anniversaires

4. Constructions pratiques

Au commencement : tables de hachage



Fonction de hachage

- Fonction $h : U \to \{0, \ldots, m-1\}$
- Algorithmique :
  pas trop de *collisions*

En cryptographie

- On oublie les tables $\to$ seules les fonctions nous intéressent
- La fonction doit paraître aléatoire
  - $\to$ difficile de *revenir en arrière* : trouver x à partir de $h(x)$
  - $\to$ difficile de trouver des collisions : $x \neq x'$ tels que $h(x) = h(x')$

Source : T. H. Cormen, C. Leiserson, R. Riverst, et C. Stein, Introduction à l'algorithmique. 2ème éd. Dunod, 2004.

Les fonctions de hachage cryptographique

Définition

- Une fonction de hachage avec taille de sortie $\ell(n)$ est un couple d'algorithmes polynomiaux $\mathcal{H} = (\mathrm{Gen}, H)$ tels que
  - $\mathrm{Gen}(1^n)$ renvoie une clef aléatoire s (n = paramètre de sécurité)
  - H est déterministe, et $H^s(x) \in \{0,1\}^{\ell(n)}$ pour tout $x \in \{0,1\}^*$
- Une fonction de compression est une fonction de hachage telle que $H^s : \{0,1\}^{\ell'(n)} \to \{0,1\}^{\ell(n)}$ avec $\ell'(n) > \ell(n)$.

Remarques

- Clef non secrète, souvent omise notée en exposant
- On dit souvent que H est une fonction de hachage, en oubliant Gen
- En pratique, pas de clef : H est une fonction déterministe fixée
  - ex. : MD5, SHA-1, SHA-2, SHA-3, …
  - pas de résultat théorique : H fixée $\to$ toute question se résout (théoriquement) en $O(1)$
  - résultats pratiques : on ne sait pas les casser (sauf MD5, SHA-1, …)

Sécurité d'une fonction de hachage

Résistance aux collisions : il est difficile de trouver $x \neq x'$ tels que $H^s(x) = H^s(x')$

Expérience de sécurité

Entrée : paramètre de sécurité 1^n

1. Protocole : $s \leftarrow \mathrm{Gen}(1^n)$
2. Attaquant : reçoit s, et produit x et x'

Succès de l'attaquant si $x \neq x'$ et $H^s(x) = H^s(x')$

Définition

Une fonction de hachage $\mathcal{H} = (\mathrm{Gen}, H)$ est résistante aux collisions si pour tout APP, $\Pr\left[H^s(x) = H^s(x')\right] \leq \mathrm{negl}(n)$.

Autres notions de sécurité

Définitions

- Une fonction de hachage est **résistante à la préimage** si étant donné s et $y = H^s(x)$ avec x choisi uniformément, aucun APP ne peut trouver x' tel que $H^s(x') = y$ avec probabilité non négligeable
- Une fonction de hachage est **résistante à la 2^nde^ préimage** si étant donné s et x choisi uniformément, aucun APP ne peut trouver x' tel que $H^s(x') = H^s(x)$ avec probabilité non négligeable

Théorème

- Une fonction résistante aux collisions est résistante à la 2^nde^ préimage.
- Une fonction résistante à la 2^nde^ préimage est résistante à la préimage.

« résistante aux collisions > résistante à la 2^nde^ préimage > résistante à la préimage »

$\rightarrow$ Preuve en TD

Applications des fonctions de hachage

Empreinte numérique

- Idée : comparer deux fichiers en comparant leurs hachés
- Applications :
  - Stockage sur un serveur : l'utilisateur garde en mémoire le haché, pour vérifier que le serveur renvoie bien le bon fichier
  - Détection de virus : base de données de hachés de virus
  - Déduplication : détecter des fichiers identiques sur un serveur
  - Pair-à-pair : haché comme identifiant des fichiers

Hachage de mots de passe

- Ne pas stocker les mots de passe en clair, mais hachés… voire *salés* !
  s aléatoire ; $h_{mdp} \leftarrow H(s, mdp)$; stockage de (s, h_{mdp})

Dérivation de clef

- Deux utilisateurs se mettent d'accord sur un secret s (mot de passe, etc.)
- Pour obtenir une clef secrète (quasi-)aléatoire : $k \leftarrow H(s)$

Application : la mise en gage

Exemple : PILE OU FACE à distance

1. Alice choisit PILE OU FACE et annonce son choix
2. Bob lance une pièce et annonce son résultat
3. Alice gagne si elle a correctement prédit la pièce de Bob

$\rightarrow$ Si Alice et Bob sont à distance, Bob peut tricher ; si on inverse 1. et 2., Alice peut tricher

Solution : la mise en gage

1. Alice *met en gage* son choix b : elle tire r uniformément et annonce $h = H(b\|r)$
2. Bob lance une pièce et annonce son résultat
3. Alice annonce b' et r' $\rightarrow$ désignation du vainqueur
4. Bob vérifie que $h = H(b'\|r')$ $\rightarrow$ vérification de l'honnêteté

Sécurité

- Difficile pour Alice de trouver r' et $b' \neq b$ tels que $H(b'\|r') = H(b\|r)$
- Difficile pour Bob, étant donné h, de trouver b

1. Fonctions de hachage et leur sécurité

2. Construction de Merkle-Damgård

3. Attaques : le paradoxe des anniversaires

4. Constructions pratiques

Des fonctions de compression aux fonctions de hachage

Rappel

- Fonction de hachage : $H^s : \{0,1\}^* \to \{0,1\}^{\ell(n)}$
- Fonction de compression : $H^s : \{0,1\}^{\ell'(n)} \to \{0,1\}^{\ell(n)}$ avec $\ell'(n) > \ell(n)$

Constat

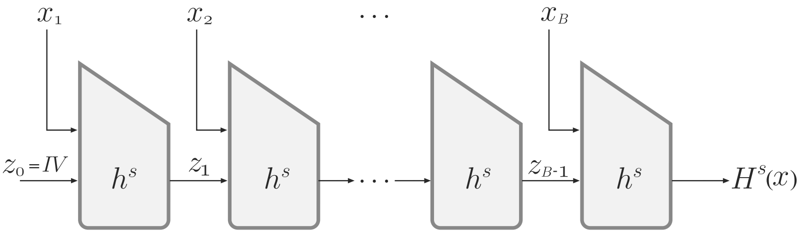
- Les fonctions de compression sont plus simples à concevoir
- Mais les fonctions de hachage sont plus utiles

Comment obtenir une fonction de hachage à partir d'une fonction de compression ?

Hypothèse

- (Gen, h) est une fonction de compression avec $h^s : \{0,1\}^{n+n'} \to \{0,1\}^n$ et $n' \geq n$
- On construit une fonction de hachage (Gen, H) Gen inchangé

Construction de Merkle-Damgård



Construction

- On fixe $\ell < n'$ et $IV \in \{0,1\}^n$
- Calcul de $H^s(x)$ pour $x \in \{0,1\}^*$ de longueur $L < 2^\ell$:
  1. $x_{pad} \leftarrow x \| 10 \cdots 0$ de longueur $Bn' - \ell$, B minimal
  2. $x_1 \| \ldots \| x_B \leftarrow x_{pad} \| L$ avec L écrit en binaire sur ℓ bits
  3. $z_0 \leftarrow IV$
  4. Pour $i = 1$ à B: $z_i \leftarrow h^s(z_{i-1} \| x_i)$
  5. Renvoyer z_B

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

La construction est sûre

Théorème

Si (Gen, h) est résistante aux collisions, (Gen, H) l'est également.

1. Fonctions de hachage et leur sécurité

2. Construction de Merkle-Damgård

3. Attaques : le paradoxe des anniversaires

4. Constructions pratiques

Attaquer une fonction de hachage

$$H^s : \{0,1\}^* \to \{0,1\}^\ell$$

Objectifs

- Trouver une collision : $x \neq x'$ tels que $H^s(x) = H^s(x')$
- Trouver une préimage : étant donné y, trouver x tel que $H^s(x) = y$
- Trouver une 2^{nde} préimage : étant donné x, trouver $x' \neq x$ tel que $H^s(x') = H^s(x)$

Recherche exhaustive

- Si on calcule $H^s(x)$ pour $2^\ell + 1$ valeurs de x $\to$ collision !
- Si $H^s : \{0,1\}^{\ell'} \to \{0,1\}^\ell$, calcul des $2^{\ell'}$ valeurs de $H^s(x)$

Remarque

- En chiffrement symétrique, meilleure attaque *générique* en $O(2^n)$

Il existe une attaque (bien) plus rapide sur les fonctions de hachage !

Le paradoxe des anniversaires

Théorème

Soit $y_1, \ldots, y_q$ tirés uniformément et indépendamment dans un ensemble de taille N, $q \leq \sqrt{2N}$. Alors

$$\frac{q(q-1)}{4N} \leq 1 - e^{-q(q-1)/2N} \leq \boxed{\Pr\left[\exists i \neq j, y_i = y_j\right]} \leq \frac{q(q-1)}{2N}$$

Exemple

- Dans un groupe de 23 personnes, il y a au moins 50% de chances que deux soient nées le même jour ! Dans cette salle de … personnes, la probabilité est $\geq \ldots$

Utilisation pour les fonctions de hachage

- On tire $x_1, \ldots, x_q$ aléatoirement, et on calcule $y_i \leftarrow H(x_i)$ pour $1 \leq i \leq q$
- Si H est aléatoire, $\Pr\left[\exists i \neq j, y_i = y_j\right] \geq q(q-1)/2^{\ell+2} \rightarrow q = \Omega(2^{\ell/2})$ suffit !
- Si H est une *vraie* fonction de hachage : probabilité de collision plus forte

Preuve du paradoxe des anniversaires

Attaques basées sur le paradoxe des anniversaires

- Tirer $\Omega(2^{\ell/2})$ valeurs suffit à trouver une collision avec probabilité constante

Construire des collisions utiles

- Trouver deux messages m_0 et m_1, de significations opposées, tels que $H(m_0) = H(m_1)$
  - Exemple : « Je dois 1000€ à Bruno » et « Bruno me doit 1000€ »
- Idée : produire de nombreuses *variantes* de m_0 et de m_1
  - m_0 : « J'ai une dette de 1000€ envers Bruno », « Je dois rendre 1000€ à Bruno », ...
  - m_1 : « Bruno a une dette envers moi, de 1000€ », ...
- On *devrait* trouver une collision, avec suffisamment de variantes

Problème de la mémoire

- Stocker tous les x_i et y_i est coûteux
- Algorithme du lièvre et de la tortue :
  1. Tirer x_0 uniformément
  2. Pour $i = 1, 2, \ldots$: $(x_i, x_{2i}) \leftarrow (H(x_{i-1}), H(H(x_{2(i-1)})))$
  3. Dès que $x_i = x_{2i}$: trouver j entre 1 et i tel que $x_j = x_{i+j}$
- Stockage de deux valeurs uniquement

1. Fonctions de hachage et leur sécurité

2. Construction de Merkle-Damgård

3. Attaques : le paradoxe des anniversaires

4. Constructions pratiques

Deux grands types de constructions pratiques

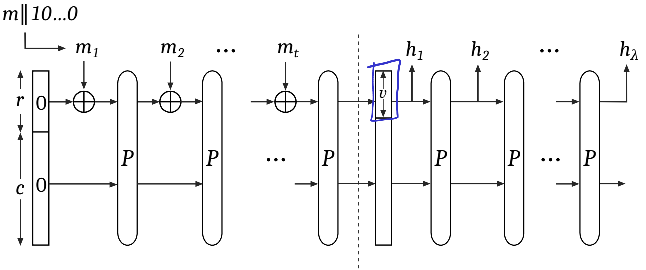
À partir d'un chiffre par blocs F_k

- Fonction de compression + construction de Merkle-Damgård $\rightarrow$ fonction de hachage
- Constructions de fonction de compression :
  - Davies-Meyer : $h(z_{i-1}, x_i) = F_{x_i}(z_{i-1}) \oplus z_{i-1}$
  - Matyas-Meyer-Oseas : $h(z_{i-1}, x_i) = F_{z_{i-1}}(x_i) \oplus x_i$
- Nombreux exemples : MD4, MD5, SHA-1 (cassées), SHA-2
- Preuve générique : nécessite un chiffre par bloc *idéal* $\rightarrow$ preuves *à la main*

À partir d'une permutation aléatoire

- Construction en éponge $\rightarrow$ directement une fonction de hachage
- Exemple : SHA-3 (Keccak)
  - vainqueur d'une compétition NIST (2008-2012)
  - meilleur choix actuel de fonction de hachage
  - permutation Keccak $-f(1600)$: permutation de $\{0,1\}^{1600}$
- Preuve générique basée sur une permutation *vraiment* aléatoire

La construction en éponge



Construction

- P : permutation de $\{0,1\}^\ell$; on fixe r, c, v, $\lambda \geq 1$ tels que $r + c = \ell$, $v \leq \ell$
- Calcul de $H(m)$, pour $m \in \{0,1\}^*$:
  0. $m_1 \| \cdots \| m_t \leftarrow m \| 10 \ldots 0$ de longueur multiple de r
  1. $y_0 \leftarrow 0^\ell$ *Phase d'absorption*
     Pour $i = 1$ à t: $y_i \leftarrow P(x_i)$ où $x_i = y_{i-1} \oplus (m_i \| 0^c)$
  2. $y_1^* \leftarrow y_t$ *Phase d'essorage*
     Pour $i = 2$ à λ : $y_i^* \leftarrow P(y_{i-1}^*)$
  3. Renvoyer $h_1 \| \cdots \| h_\lambda$ où $h_i = {y_i^*}_{[0,v[}$

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

Sécurité de la construction en éponge (1/3)

Théorème

Si P est une permutation aléatoire et $\lambda = 1$, un attaquant effectuant q requêtes à P ou P^{-1} ne peut trouver une collision dans H qu'avec probabilité $\leq \frac{q^2}{2^v} + \frac{q(q+1)}{2^c}$.

Preuve

- Soit A qui produit (m^0, m^1) tel que $H(m^0) = H(m^1)$
- Après *padding*, $m^0 \to m^0_1 \| \cdots \| m^0_{t^0}$ et $m^1 \to m^1_1 \| \cdots \| m^1_{t^1}$
- Calcul de $H(m^0)$: $y^0_0 \to x^0_1 \to \cdots \to y^0_{t^0}$ avec $x^0_i = y_{i-1} \oplus (m^0_i \| 0^c)$ et $y^0_i = P(x^0_i)$
- Calcul de $H(m^1)$: $y^1_0 \to x^1_1 \to \cdots \to y^1_{t^1}$ avec $x^1_i = y_{i-1} \oplus (m^1_i \| 0^c)$ et $y^1_i = P(x^1_i)$
- On suppose que
  - A interroge P sur des x^0_i ou x^1_i et P^{-1} sur des y^0_i ou y^1_i
  - A n'interroge pas P ou P^{-1} de manière redondante
  - A interroge $P(x^0_1), \ldots, P(x^0_{t^0})$ et $P(x^1_1), \ldots, P(x^1_{t^1})$

Sécurité de la construction en éponge (2/3)

Lemme 1

Si l'attaquant produit une collision, alors un des trois évènements arrive :

- E_1: A fait 2 requêtes à P dont les résultats sont égaux sur leurs v premiers bits
- E_2: A fait une requête à P ou P^{-1} dont le résultat finit par 0^c
- E_3: A fait 2 requêtes (à P ou P^{-1}) dont les résultats sont égaux sur leurs c derniers bits

Cas 1 A a fait une requête à P^{-1}

0^ℓ

- Si $P^{-1}(y_1^b)$ → il obtient $x_1^b = y_0^b \oplus (m_1^b \| 0^c)$ E_2
- Sinon $P^{-1}(y_i^b) = x_i^b$ et $P(x_{i-1}^b) = y_{i-1}^b$; or $x_i^b = y_{i-1}^b \oplus (m_i^b \| 0^c)$ E_3

Cas 2 Requêtes uniquement à P mais $y_{t^0}^0 \neq y_{t^1}^1$

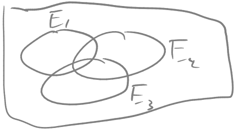
$x_{t^0}^0 \neq x_{t^1}^1$ mais $P(x_{t^0}^0) = y_{t^0}^0$ et $P(x_{t^1}^1) = y_{t^1}^1$ partagent les m̂ v premiers bits E_1

Cas 3 Requêtes uniquement à P et $y_{t^0}^0 = y_{t^1}^1$.

Soit i maximal tq $y_{t^0-j}^0$ et $y_{t^1-j}^1$ aient les m̂ c derniers bits pour $0 \leq j \leq i$

- Si $i < t^0$: $x_{t^0-i}^0 \neq x_{t^1-i}^1$ E_3
- Si $i = t^0$ et $t^0 < t^1$: $P(x_{t^1-i}^1) = y_{t^1-i}^1$ $\Rightarrow E_2$
- Si $i = t^0 = t^1$: $y_{j-1}^0 = y_{j-1}^1$ mais $x_j^0 \neq x_j^1$ E_3

Sécurité de la construction en éponge (3/3)



Lemme 2

La permutation P étant aléatoire, $\Pr[E_1 \vee E_2 \vee E_2] \leq \frac{q^2}{2^v} + \frac{q(q+1)}{2^c}$.

- $\Pr[E_1 \vee E_2 \vee E_3] \leq \Pr[E_1] + \Pr[E_2] + \Pr[E_3]$
- $\Pr[E_2] \leq q/2^c$ car à chaque requête, la proba est $1/2^c$
- $C_{i,j}$: résultats des $i^{ème}$ et $j^{ème}$ requêtes ont les m̂ v premiers bits

$$\Pr[E_1] = \Pr\Big[\bigvee_{i<j} C_{i,j}\Big] \leq \sum_{i<j} \Pr[C_{i,j}] \leq \binom{q}{2} \cdot \frac{2}{2^v} \leq \frac{q^2}{2^v}$$

$$\Pr[C_{i,j}] \leq \frac{2^{\ell-v}}{2^\ell - 1}$$

- $\Pr[E_3] \leq q^2/2^c$ ~> m̂ preuve que pour E_1.

Conclusion

- Fonctions de hachage (cryptographique) : primitive extrêmement utile !
  - Hachage de mots de passe
  - Codes d'authentification de message *cours 5*
  - Signature électronique *2^{nde} partie*
  - Version sécurisée de RSA (RSA-OAEP) *2^{nde} partie*
  - Blockchain
  - …
- Constructions basées sur des chiffres par blocs
  - Fonction de compression + construction de Merkle-Damgård
  - Exemples : ~~MD4~~, ~~MD5~~, ~~SHA-1~~, SHA-2
- Constructions basées sur des permutations aléatoires
  - Construction en éponge
  - Exemple : SHA-3

Si vous avez besoin d'une fonction de hachage, utilisez SHA-3 (éventuellement SHA-2) !