

Sécurité des systèmes informatiques  
**Les fonctions de hachage et l'intégrité**

Olivier Markowitch

# Les fonctions de hachage

Une **fonction de hachage**,  $h$ , applique un string binaire de taille quelconque finie en un string binaire de taille fixe  $n$

Si les strings d'input sont de longueur  $> n$ , nous aurons plusieurs strings appliqués vers un même string résultat, nous avons alors des **collisions**

# Propriétés des fonctions de hachage

**Compression** : la fonction de hachage  $h$  applique un input  $x$  binaire de taille quelconque vers un output  $h(x)$  binaire de taille fixe  $n$

**Facilité de calcul** : étant donnée  $h$  et un input  $x$ ,  $h(x)$  doit être facile à calculer

Remarque : si  $y$  est tel que  $y = h(x)$ , alors  $x$  est dit être la *préimage* de  $y$

# Mécanismes

Les fonctions de hachage peuvent être utilisées dans :

- les *codes de détection de manipulation* (MDC) : gère l'intégrité
- les *codes d'authentification de message* (MAC) : gère l'intégrité ainsi que l'authentification de la source d'une donnée

Dans les MDCs nous trouvons deux grandes classes de fonctions de hachage les *one-way hash functions* (OWHF) et les *collision resistant hash functions* (CRHF)

# Propriétés additionnelles

Soient les inputs  $x$  et  $x'$  et les outputs correspondants  $y$  et  $y'$ , une fonction de hachage *peut* aussi respecter :

1. la **résistance de la préimage** : pour la plupart des outputs il est calculatoirement infaisable de trouver une préimage  $x'$  tel que  $h(x') = y$  pour tout  $y$  donné dont l'input correspondant n'est pas connu
2. la **résistance de la seconde préimage** : étant donné  $x$ , il est calculatoirement infaisable de trouver une deuxième préimage  $x' \neq x$  telle que  $h(x) = h(x')$
3. la **résistance à la collision** : il est calculatoirement infaisable de trouver deux inputs  $x$  et  $x'$  tel que  $h(x) = h(x')$

# Définitions

Une **one-way hash function** (OWHF) est une fonction de hachage qui respecte les propriétés additionnelles de résistance à la préimage et de résistance à la seconde préimage

Les one-way hash functions sont aussi parfois appelées *weak one-way hash functions*

Une **collision resistant hash function** (CRHF) est une fonction de hachage qui respecte les propriétés additionnelles de résistance à la seconde préimage et de résistance à la collision

Les collision resistant hash functions sont aussi parfois appelées *strong one-way hash functions*

# Fonction à sens unique

Une fonction  $f$  est dite à **sens unique** (one-way) si pour tout  $x$  appartenant au domaine de  $f$ , il est facile de calculer  $y = f(x)$ , mais pour la plupart des  $y$  de l'image de  $f$  il est calculatoirement infaisable de trouver un  $x$  tel que  $y = f(x)$

La propriété de compressivité est donc levée

# Keyed et unkeyed hash functions

Une code d'authentification de message (message authentication code : MAC) est une fonction de hachage paramétrée par une clé secrète  $k$ ,  $h_k()$ , qui respecte la facilité de calcul, la propriété de compression et qui est telle que :

pour tout  $k$  fixé et inconnu par un adversaire, il est calculatoirement infaisable pour cet adversaire de calculer une paire  $(x, h_k(x))$  à partir de sa connaissance d'un nombre quelconque de paires  $(x_i, h_k(x_i))$  où  $x$  est différent de tous les  $x_i$

Les codes de détection de manipulation (manipulation detection code : MDC) sont eux associés aux fonctions de hachage sans clé

# Fonction de hachage itérée

$$h(x) = g(H_t)$$

$$\begin{cases} H_0 = \text{valeur initiale} \\ H_i = f(H_{i-1}, x_i) \text{ avec } i \in [1, t] \end{cases}$$

$$x = x_1 \dots x_t \text{ avec } |x_i| = r \text{ pour } i \in [1, t]$$

# Sécurité

Une fonction de hachage non paramétrée par une clé (unkeyed) produisant un output de  $n$  bits est dit avoir une sécurité idéale si :

1. étant donné un output produire un préimage ou un deuxième préimage requiert approximativement  $2^n$  opérations
2. produire une collision requiert  $2^{\frac{n}{2}}$  opérations

# MDC en pratique

Les codes de détection de manipulation peuvent être construits :

- sur base de chiffrements par blocs
- sur mesure : MD4, MD5, SHA-1, RIPEMD-160
- sur base de l'arithmétique modulaire : MASH-1

# MAC en pratique

Les codes d'authentification de messages peuvent être construits :

- sur base sur de chiffrements par blocs

# CBC-MAC

Sans la partie optionnelle, nous pouvons avoir :

Soit  $x$  une donnée tenant sur 1 bloc

Soit  $M = \text{CBC-MAC}(x)$

$\text{CBC-MAC}(M) = \text{CBC-MAC}(x \parallel 0 \dots 0)$

où  $\parallel$  représente la concaténation et où la taille de  $x \parallel 0 \dots 0$   
= 2 blocs

# MAC en pratique

Les codes d'authentification de messages peuvent être construits :

- sur base sur de chiffrements par blocs
- à partir de MDCs

# MAC sur base de MDC

On inclut la clé  $k$  dans les données à hacher par le MDC

Comme un MDC itératif sur un message  $x = x_1 \dots x_t$  pourrait être construit ainsi :

$$\begin{cases} H_0 = \text{valeur initiale} \\ H_i = f(H_{i-1}, x_i) \text{ avec } i \in [1, t] \\ h(x) = H_t \end{cases}$$

Alors la construction :  $h_k(x) = h(k|x)$  est risquée

Mieux :  $h_k(x) = h(x|k)$  ou  $h_k(x) = h(k|x|k)$

# MAC en pratique

Les codes d'authentification de messages peuvent être construits :

- sur base sur de chiffrements par blocs
- à partir de MDCs
- sur mesure : MAA, MD5-MAC

# L'intégrité

**L'intégrité des données** assure qu'une donnée n'a pas été altérée de manière non-autorisée (quand elle est stockée ou transférée)

**L'authentification de l'origine des données** se base sur un partage d'une clé secrète. Mais on ne peut distinguer les parties partageant cette clé

Quand les techniques d'authentification de l'origine des données ne permettent pas de se prémunir de la réutilisation d'un message non modifié, on parle alors d'**authentification de message**

Lorsqu'on rajoute des mécanismes assurant l'unicité de l'usage d'une donnée, on obtient l'**authentification de transaction**

# Les techniques

L'intégrité s'obtient en utilisant :

- des mécanismes de détection et/ou correction d'erreurs
- des techniques de MAC
- des techniques de MDC associées à un canal authentique
- le chiffrement
- des techniques de MDC associées à du chiffrement
- des techniques de MAC associées à du chiffrement

# MAC associées à du chiffrement

Nous pouvons avoir :

$$h_k(x) = E_k(x|h_{k'}(x))$$

Mais risque si :

- $k = k'$ , et
- $h_{k'} = \text{CBC-MAC}$  sans partie optionnelle, et
- $E_k$  chiffrement en mode CBC identique à celui utilisé dans le CBC-MAC, et
- utilisation de la même valeur initiale

# Paradoxe des anniversaires

Sur une assemblée de 23 personnes la probabilité qu'au-moins deux d'entre-elles aient leur anniversaire le même jour (en ne tenant pas compte de l'année de naissance) est égale à 50 pourcents

# Paradoxe des anniversaires

Soit une fonction  $h$  de hachage :  $h : X \rightarrow Z$  où  $X$  et  $Z$  sont des ensembles finis tels que  $|X| > |Z|$ ; soient  $|X| = m$  et  $|Z| = n$

Prenons  $k$  messages  $x_i \in X$ , tirés aléatoirement, avec  $i \in [1, k]$  et étudions la probabilité que deux  $x_i$  différents aient la même image (ce qui revient à avoir une collision)

$$z_i = h(x_i) \text{ pour } i \in [1, k]$$

Nous pouvons considérer que les  $z_i$  sont des éléments aléatoires (ce qui est désirable pour une fonction de hachage) puisque les  $x_i$  sont tirés aléatoirement

# Paradoxe des anniversaires

La probabilité que les  $z_i$  soient tous différents est :

$$1 \left(1 - \frac{1}{n}\right) \left(1 - \frac{2}{n}\right) \dots \left(1 - \frac{k-1}{n}\right) = \prod_{i=1}^{k-1} \left(1 - \frac{i}{n}\right)$$

Où 1 est la probabilité de tirer  $z_1$ ,  $\left(1 - \frac{1}{n}\right)$  la probabilité de tirer  $z_2 \neq z_1$  (car il y a une chance sur  $n$  que  $z_1 = z_2$ ),  $\dots$ ,  $\left(1 - \frac{i}{n}\right)$  la probabilité de tirer  $z_i$  différent de  $z_1, \dots, z_{i-1}$

# Paradoxe des anniversaires

$$\prod_{i=1}^{k-1} \left(1 - \frac{i}{n}\right) \approx \prod_{i=1}^{k-1} e^{-\frac{i}{n}}$$

car  $e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} \dots$ , donc  $e^{-x} \approx 1 - x$  si  $x$  est petit, et ici  $x = \frac{i}{n}$  avec au plus  $x = \frac{k}{n}$  et  $k \leq n$

Donc  $1 - \frac{i}{n} \approx e^{-\frac{i}{n}}$

$$\prod_{i=1}^{k-1} \left(1 - \frac{i}{n}\right) \approx e^{-\frac{1}{n} \sum_{i=1}^{k-1} i} = e^{-\frac{(k-1)k}{2n}}$$

car

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}$$

# Paradoxe des anniversaires

Soit  $P'$  la probabilité qu'il n'y ait pas de collision, nous avons :

$$\prod_{i=1}^{k-1} 1 - \frac{i}{n} = P'$$

$$P' \approx e^{-\frac{(k-1)k}{2n}}$$

$$\ln(P') \approx -\frac{(k-1)k}{2n}$$

$$2n \ln(P') \approx -(k-1)k$$

$$2n \ln\left(\frac{1}{P'}\right) \approx k^2 - k$$

$$2n \ln\left(\frac{1}{P'}\right) \approx k^2 \text{ (en n\u00e9gligeant } k \text{ devant } k^2)$$

# Paradoxe des anniversaires

$$2n \ln\left(\frac{1}{P'}\right) \approx k^2, \text{ donc } \sqrt{2n \ln\left(\frac{1}{P'}\right)} \approx k$$

Soit  $P$  la probabilité qu'il y ait au-moins une collision :

$$P = 1 - P'$$

$$\sqrt{2n \ln\left(\frac{1}{1-P}\right)} \approx k$$

Si  $P = \frac{1}{2}$  (une chance sur deux de trouver au-moins une collision dans  $h(x_1) \dots h(x_k)$ ) :

$$\sqrt{2n \ln(2)} \approx k$$

$$\sqrt{1,386n} \approx k$$

$$1,177\sqrt{n} \approx k$$

Et  $k$  est en  $O(\sqrt{n})$

# Paradoxe des anniversaires

Si un hachage fait 64 bits ( $|Z| = 2^{64}$ ), il y a une chance sur deux de trouver une collision en testant plus ou moins  $2^{32}$  messages (ce qui est réalisable)

En toute généralité, si  $|Z| = 2^r$ , il y a une chance sur deux de trouver une collision en testant  $2^{\frac{r}{2}}$  messages

Ceci explique la valeur de la sécurité idéale d'une fonction de hachage (voir transparent du cours)

Et ainsi, il vaut mieux une fonction de hachage correctement construite et produisant des outputs de grande taille pour éviter que les collisions ne se trouvent trop facilement

# Paradoxe des anniversaires

Si  $X = \{\text{humains}\}$  et  $Z =$  chacun des 365 jours (ce sont les anniversaires),  $|Z| = 365 = n$ , nous avons :  
 $1,177\sqrt{n} = 1,177\sqrt{365} \approx 23$

Sur 23 personnes, il y a une chance sur deux d'en trouver au-moins deux qui auront leur anniversaire le même jour

Si on fixe  $P = \frac{3}{4}$ , nous avons  $k \approx 1,66\sqrt{n}$ , si  $n = 365$  cela donne  $k \approx 32$  (sur 32 personnes il y a trois chances sur quatre qu'au moins deux personnes aient leur anniversaire le même jour)

Si  $P = 99\%$ , alors  $k \approx 58$

# Paradoxe des anniversaires

Dans ce qui précède nous utilisons le fait que

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}$$

Quand  $n = 1$  nous avons  $1 = \frac{1 \cdot 2}{2}$

Supposons que la formule soit vraie pour  $n$ , vérifions-la pour  $n + 1$  :

$$\begin{aligned} \sum_{i=1}^{n+1} i &= \sum_{i=1}^n i + (n+1) = \frac{n(n+1)}{2} + (n+1) \\ &= \frac{n(n+1) + 2(n+1)}{2} = \frac{(n+1)(n+2)}{2} \end{aligned}$$

Ce qui vérifie et prouve la formule