Fonctions de Hachage

Plan

- Généralités
- MD5
- SHA

Généralités

- Une fonction de hachage peut satisfaire 2 objectifs
  - Vérifier l'intégrité d'un message
    - Fonctions de hachage sans clé (Unkeyed hash functions)
    - L'intégrité peut être vérifiée par tous
    - MDC : Modification Detection Codes
  - Vérifier l'authenticité d'un message
    - Fonction de hachage avec clé (Keyed hash functions)
    - L'intégrité ne peut être vérifiée qu'avec la connaissance de la clé
    - MAC : Message Authentication Codes

Principe d'une fonction de hachage

- Définition générale

$$h: \quad \{0,1\}^* \longrightarrow \{0,1\}^m$$
$$x \longrightarrow h(x)$$

  - Le message x est de taille quelconque, mais l'empreinte est de taille fixe m
  - h doit être facile à calculer, mais difficile à inverser (c'est souvent une fonction à sens unique, mais sans brèche secrète)
  - Déterminisme : une fonction de hachage renvoie toujours la même empreinte pour un x donné
  - Uniformité : l'ensemble des entrées doit produire des sorties uniformément réparties (ayant la même probabilité, entropie maximale)

- Propriétés exigées pour une utilisation en cryptographie
  - Résistance à la pré-image
    - Étant donné une sortie y, il est difficile de trouver x tel que : h(x) = y
    - C'est la définition d'une fonction à sens unique
  - Résistance à la seconde pré-image
    - Étant donné une entrée x, il est difficile de trouver x' tel que : h(x') = h(x)
  - Résistance à la collision
    - Il est très improbable de trouver aléatoirement x et x' tels que h(x)=h(x')

Un exemple de collision

Collision trouvée sur MD4

m1

839c7a4d7a92cb**5**678a5d5**b**9eea5a7573c8a74deb366c3dc20a083b69f
5d2a3bb3719dc69891e9f95e809fd7e8b23ba6318ed**d**45e51fe39708bf
9427e9c3e8b9

m2

839c7a4d7a92cb**d**678a5d5**2**9eea5a7573c8a74deb366c3dc20a083b69f
5d2a3bb3719dc69891e9f95e809fd7e8b23ba6318ed**c**45e51fe39708bf9
427e9c3e8b9

Bien sûr, $m_1 \neq m_1$

mais MD4(m1) = MD4(m2)=4d7e6a1defa93d2dde05b45d864c429b.

- Les fonctions de hachage sont souvent construites de manière à favoriser l'effet d'avalanche
  - Une différence (même infime) entre deux entrées produit deux empreintes radicalement différentes

Exemples

```
> echo "MD5 est une fonction de hachage cryptographique" |
md5sum
4a3efe18fa3a90af8b2fe1d0020bdfde
> echo "MD5 est une fonction de hachage cryptographiquE" |
md5sum
8e2bc55b3cefb44d656b79ad6ae84675
```

- Paradoxe des anniversaires
  - Soit k personnes dans une pièce
    - Quelle est la probabilité p pour que 2 personnes parmi les k aient le même jour anniversaire (parmi les 365 possibles) ?
    - Quelle est la valeur de k pour que cette probabilité soit environ d'une sur deux ?
  - Si k=2, la probabilité qu'elles soient nées des jours différents est 364/365
    - La 2ième personne a 364 autres possibilités d'être née un autre jour que la 1ière personne
    - La probabilité p d'être née un jour différent est P = 364/365 = 0,997 (presque 100% de chance d'être né un jour différent)

- Si k=3 ?
  - La 3ième personne n'a plus que 363 possibilités d'être née un jour différent p = (364/365)*(363/365)
- Pour k quelconque :

$$p = 1\left(\frac{364}{365}\right)\left(\frac{363}{365}\right)\ldots\ldots\left(\frac{365-k+1}{365}\right)$$

- K=10, p=0,883
- K=20, p=0,589
- K=23, p=0,493!!

- Pour K=23 on a plus qu'une chance sur 2 que deux personnes soient nées des jours différents
- La probabilité que deux personnes soient nées le même jour est 1-p
  - Pour K=10, 1-p = 1-0.883 = 0.117 soit 12% de chance que 2 des 10 personnes soient nées le même jour
  - Pour K=23, 1-p = 1-0,493 = 0,507 soit 50% !

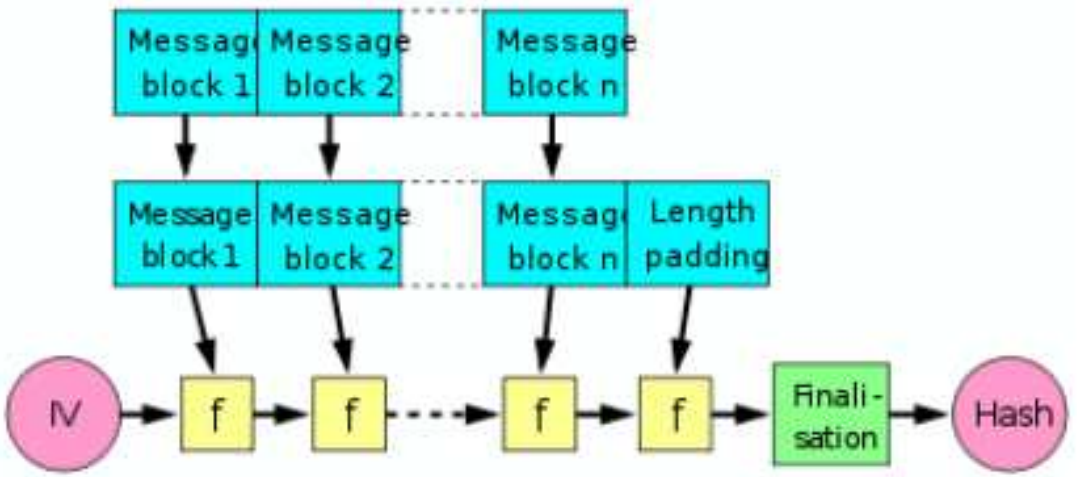
- Application aux fonctions de hachage
  - Si une empreinte est sur N bits, il faut environ $2^{\frac{N}{2}}$ empreintes pour avoir 50% de chance de créer une collision
  - Une empreinte de 128 bits n'offre plus aujourd'hui une sécurité suffisante (2^{64}hachés pour provoquer une collision constitue un effort de calcul raisonnable)

- Principe de construction de Merkle–Damgård
  - Le message est fractionné en blocs de taille fixe n (n peut être différent de m)
  - Si la taille du message n'est pas un multiple de n, il est complété
  - Chaque entrée de n bits est compressée pour obtenir une sortie de m bits
  - La fonction de compression peut être conçue spécialement pour le hachage ou peut réutiliser un algo de chiffrement par bloc (DES, AES, etc.)

- A l'itération i, le bloc d'entrée i est combiné avec la sortie i-1
- Le bloc 0 est combiné avec un vecteur d'initialisation

- Il est admis que si la fonction de compression choisie est résistante aux collisions, alors la fonction de hachage construite est aussi résistante

- Schéma de hachage:



MD5

- Évolution de MD4
- Proposé par Rivest en 1991 (RFC1321)
- Très utilisé pour stocker les mots de passe
  - Seules les empreintes sont stockées
  - Pour vérifier un mot de passe, on calcule son empreinte que l'on compare avec celle stockée
- Considéré actuellement comme non suffisamment sûre (empreinte de 128 bits)
  - Il existe des techniques performantes pour provoquer des collisions

MD5

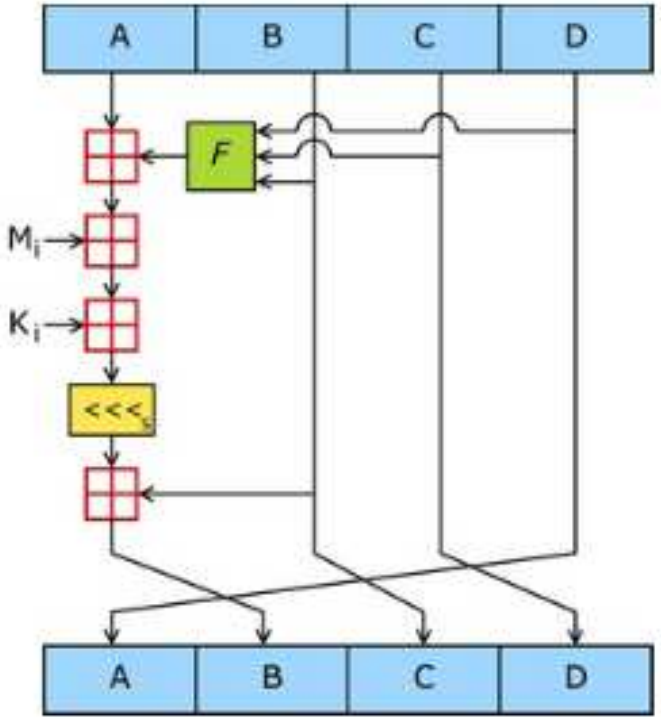
- MD5 est une fonction de hachage cryptographique, qui permet donc d'obtenir l'empreinte numérique d'un message.
- MD5 = Message Digest 5
  - Successeur de MD4
  - Proposé par Ronald Rivest en 1991
- MD5 associe à toute chaîne de caractères une
- généralement représentée sous forme de

Exemple

```
> echo "Salut" | md5sum
1560c8afac2fa4c0b61d6b7774406c8b
```

- Principe :
  - Traite des blocs de 512 bits
  - Le dernier bloc est complété avec 10000...T
    - C'est le « padding »
    - T est un mot de 64 bits représentant la taille du message initial
    - Le nombre de 0 est ajusté pour obtenir un bloc de 512 bits
    - T doit être codé en little-endian comme tous les entiers considérés dans MD5

- Hachage d'un bloc en 64 rondes
  - Le hachage d'un bloc modifie un 'état' composé de 4 mots de 32 bits (A,B,C,D)
- Vue schématique d'une ronde
  - M[i] : ième mot de 32 bits du bloc à hacher
  - K[i] : i constante précalculée de MD5
  - : Addition modulo 232
  - :Rotation vers la gauche de s bits
  - F : fonction non linéaire (effet d'avalanche)



- K[i] :
  - valeurs d'une sinusoïde pour des angles compris en 0 et 63 radians
  - Peuvent être précalculées une fois pour toute (ne dépend pas du message)

```
void MD5::initT() {
    for (int i=0; i<64; i++) {
        K[i] = (uint)(4294967296. * fabs(sin(i+1)));
    }
}
```

- Fonctions non linéaires :
  - Ronde de 0 à 15 : $F(B, C, D) = (B \wedge C) \vee (\bar{B} \wedge D)$
  - Ronde de 16 à 31 : $\mathrm{G}(B, C, D) = (B \wedge D) \vee (C \wedge \bar{D})$
  - Ronde de 32 à 47 : $\mathrm{H}(B, C, D) = \mathrm{B} \oplus C \oplus D$
  - Ronde de 48 à 63 : $I(B, C, D) = C \oplus (B \vee \bar{D})$

- Les valeurs de s dépendent de la ronde

```
s[ 0..15] := {7, 12, 17, 22,  7, 12, 17, 22,  7, 12, 17, 22,  7, 12, 17, 22}
s[16..31] := {5,  9, 14, 20,  5,  9, 14, 20,  5,  9, 14, 20,  5,  9, 14, 20}
s[32..47] := {4, 11, 16, 23,  4, 11, 16, 23,  4, 11, 16, 23,  4, 11, 16, 23}
s[48..63] := {6, 10, 15, 21,  6, 10, 15, 21,  6, 10, 15, 21,  6, 10, 15, 21}
```

- Exemple de la ronde 0 :

$$A = A + F(B, C, D) + M[0] + K[0]$$
$$A = A <\!\!\ll s[0]$$
$$A = A + B$$

Rotation vers la droite de A,B,C et D

- Vecteur d'initialisation de l'état :
  - A = 0x67452301
  - B = 0xEFCDAB89
  - C = 0x98BADCFE
  - D = 0x10325476

SHA

- Secure Hash Standard (SHA)
  - SHA-1 proposé par le NIST en 1993, lui même
  - inspiré de MD4/MD5
  - Document officiel : FIPS PUB 180-1
  - Empreinte 160 bits
  - Décliné ensuite en SHA-256 et SHA-512

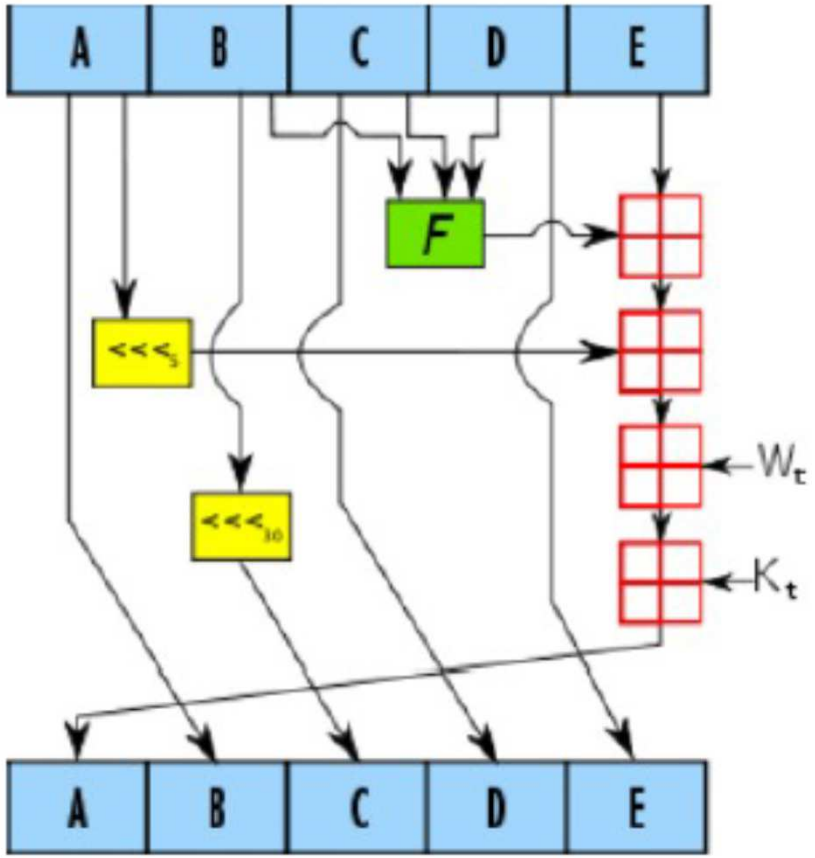
- Principe :
  - Mots codés en Big-endian
  - Traitement par blocs de 512 bits
  - Remplissage (padding) identique à MD5
    - 100....00L où L est la taille du message codée sur 64 bits en big-endian
    - Le nombre de 0 est ajusté pour compléter le dernier bloc de 512 bits
  - L'état est composé de 5 mots de 32 bits (A,B,C,D,E) soit 160 bits
  - Le bloc est composé de 16 mots de 32 bits : M[0] à M[15]

- Hachage d'un bloc en 80 rondes
  - Utilise 3 fonctions non linéaires
    - Ronde 0..19 : $F(B,C,D) = (B \wedge C) \vee (\bar{B} \wedge D)$
    - Ronde 20..39 : $\mathrm{G}(B,C,D) = \mathrm{B} \oplus C \oplus D$
    - Ronde 40..59 : $\mathrm{H}(B,C,D) = (B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$
    - Ronde 60..79 : $\mathrm{G}(B,C,D) = \mathrm{B} \oplus C \oplus D$
  - F est identique à F dans MD5 !
  - G est identique à H dans MD5 !

- Utilise 4 constantes :
  - Ronde 0..19 : K_0 = 0x5A827999
  - Ronde 20..39 : K_{20} = 0x6ED9EBA1
  - Ronde 40..59 : K_{40} = 0x8F1BBCDC
  - Ronde 60..79 : K_{60} = 0xCA62C1D6
- 80 mots de 32 bits sont construits à partir des 16 mots du bloc
  - Pour i de 0 à 15 :W[i] = M[i]
  - Pour i de 16 à 79 :

$$W[i] = (W[i-3] \oplus W[i-3] \oplus W[i-3] \oplus W[i-3]) \lll 1$$

- Vue schématique d'une ronde



- Exemple de la ronde 0 :

$$T = (A \lll 5) + F(B, C, D) + E + W[0] + K_0$$
$$E = D$$
$$D = C$$
$$C = (B \lll 30)$$
$$B = A$$
$$A = T$$

- Vecteur d'initialisation de l'état
  - A = 0x67453201
  - B = 0xEFCDAB89
  - C = 0x98BADCFE
  - D = 0x10325476
  - E = 0xC3D2E1F0