Les fonctions de hachage

1 Fonction de hachage

= fonction qui,

recevant *un ensemble des données d'une taille quelconque,*
fournit une chaîne de taille fixe (ex: 128 bits)

avec comme propriétés que

- la probabilité de trouver deux fois la même valeur résultante pour des messages différents est extrêmement faible (+/- nulle);
- la fonctions de hachage ne peut être inversée (ou, du moins, il faut un temps quasiment infini pour y parvenir) : il est donc impossible de retrouver la donnée qui a produit le hachage
- la distribution des valeurs produite est uniforme et chaotique.

Le résultat est encore appelé **digest** ou **empreinte** ou **hashage**.

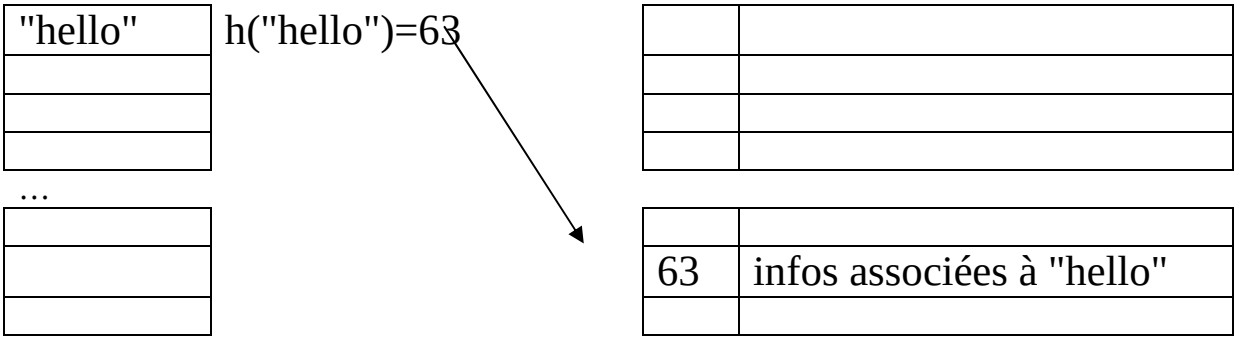
2. Exemple (simpliste)

$$h(\text{message}) = (\sum(\text{codes ASCII des caractères})) \ \% 67$$

Ex:
h("hello") = (104 + 101 + 108 + 108 + + 111) % 67 = 63

3. Utilisation

a) table de hashcoding (collisions admises)



b) le contrôle d'intégrité (collisions non souhaitées)
Le problème de l'**intégrité** des données transmises = *savoir si les données que l'on obtient par le réseau (par exemple) sont restées ce qu'elles étaient à leur envoi*.

Historiquement :
bit de parité → checksums (CRC) → message digests (MD5, SHA-1).

Un message est envoyé avec son digest. L'intégrité est vérifiée
- en calculant le message digest sur le texte reçu par réseau;
- en le comparant au message digest qui accompagnait le texte.

En cas d'égalité, l'intégrité est avérée.

c)

l'authentification (collisions non admises)
Dans un système d'authentification basé sur le principe user-password,

- ce n'est pas le password qui est envoyé, mais son digest;
- pour éviter la réutilisation malveillante d'un tel digest, on pratique un "**salage**" du digest = calculer celui-ci non seulement sur le password mais aussi sur un nombre envoyé par le serveur (= un "**challenge**") ou sur une date-heure ou un code PIN ou tout autre élément non reproductible.

En cas d'égalité du digest envoyé et du digest recalculé par le serveur, l'authentification est avérée.