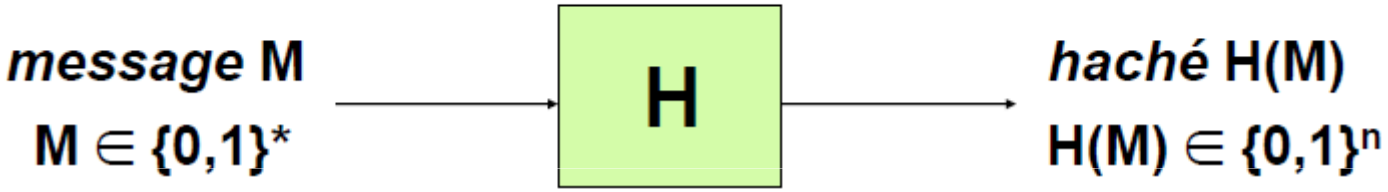
Révision sur les méthodes de hachage

Définition



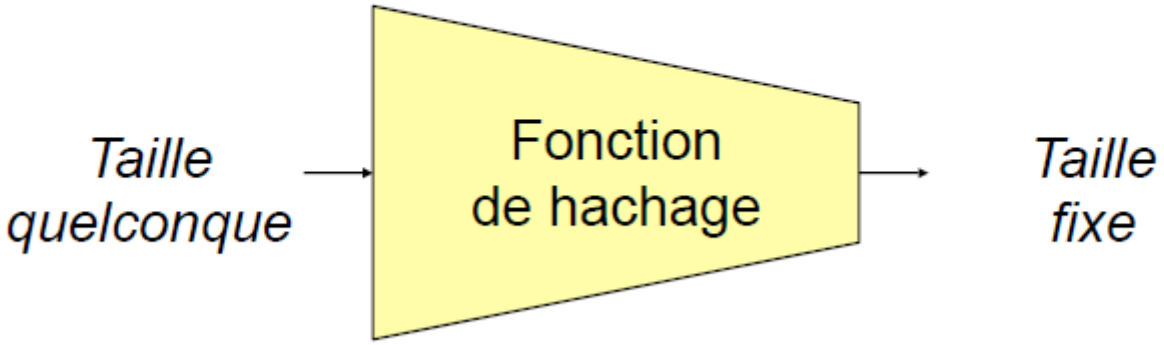
Une fonction de hachage H calcule un haché de n bits à partir d'un message arbitraire M

$$H : \{0,1\}^* \rightarrow \{0,1\}^n$$

Critères de conception

La fonction de hachage doit

- réduire la taille des entrées à n bits (compression)
- être facile à calculer



Fonctions classiques

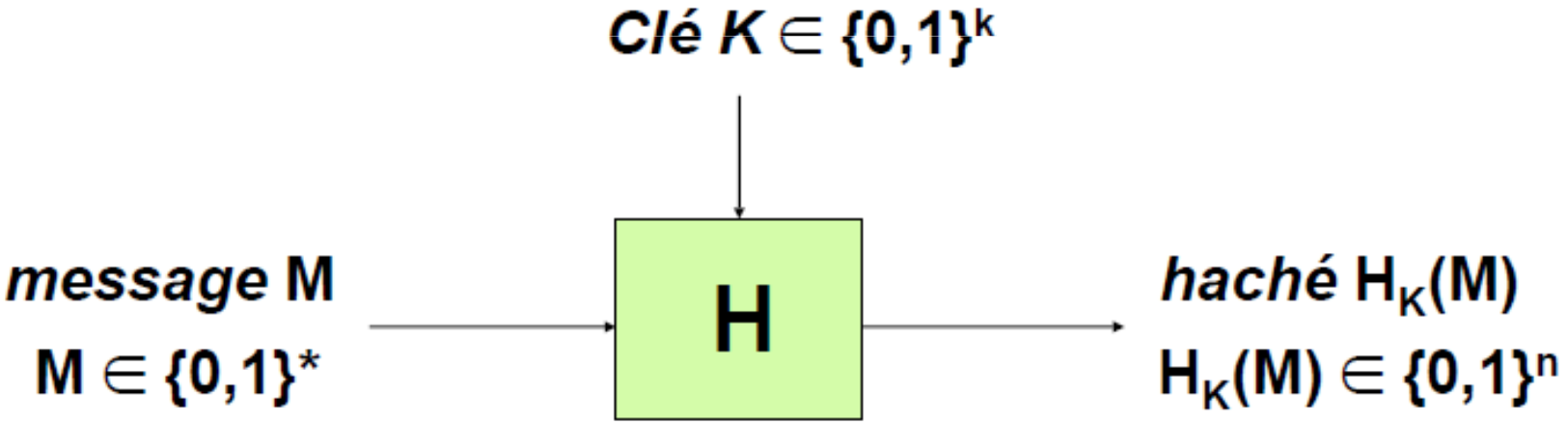
Nom	n	Auteur
MD2	128	RSA
MD4	128	RSA
MD5	128	RSA
SHA-1	160	NIST
SHA-256	256	NIST

Suite

Nom	n	h	Auteur
MD2	128	128	RSA
MD4	512	128	RSA
MD5	512	128	RSA
SHA-1	512	160	NIST
SHA-256	512	256	NIST

Présence d'une clé

Toutefois, dans certaines situations, on peut être amené à utiliser des fonctions de hachage avec clé :



Utilisation

1. Intégrité de fichier
2. Stockage de mots de passe
3. Intégrité de communications
4. Signature numérique

Intégrité

Idée : on veut vérifier qu'un fichier n'a pas été modifié
→ On calcule son empreinte

```
// Fichier code.c

#include <stdio.h>
#include <stdlib.h>

int main(int argc, char** argv)
{
  if (argc <2)
    {
      …
    }
}
```

SHA-1 →

Valeur hachée
de 160 bits :

SHA-1 (`code.c`) =
A51F 07BB 62EC 44A3 F118

Mots de passe

- Au lieu de stocker un mot de passe « en clair » sur une machine, on stocke son haché (ex : systèmes d'exploitation)

$$h = H(password)$$

- Pour s'authentifier, un utilisateur envoie h
- Cela rend la base de données moins sensible

<u>Résistance mathématique</u>

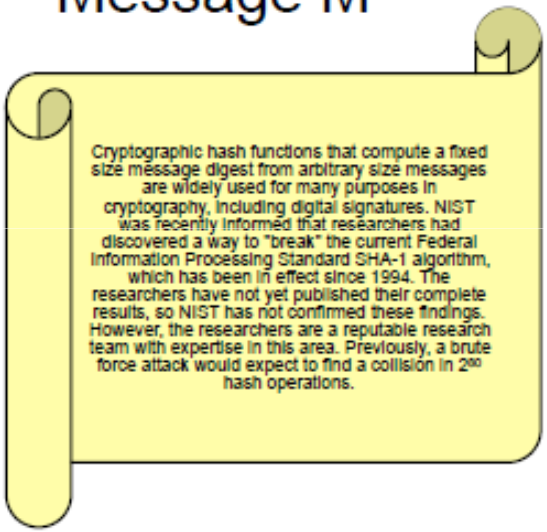
– A partir du haché h, difficulté de trouver un antécédent x tel que

$$h = H(x) = H(password)$$

– Est-il nécessaire que x == *password* ?

Signature

Message M



Clé de signature K_s

Algorithme de signature $Sign_{KS}(.)$

Fonction de hachage H(.)

Signature du message M :

$$\sigma = Sign_{KS}(H(M))$$

Sécurité

Les notions suivantes sont apparues :

- Résistance aux collisions

  trouver M_1 et M_2 tel que $H(M_1) = H(M_2)$

- Résistance aux secondes préimages

  avec M_1 donné, trouver M_2 tel que $H(M_1) = H(M_2)$

- Résistance aux préimages

  avec x donné, trouver M tel que $H(M) = x$

Attaques génériques

- Résistance aux collisions

  une attaque naïve coûte $2^{n/2}$

- Résistance aux secondes préimages

  une attaque naïve coûte $2^{n/2} < ? < 2^n$

- Résistance aux préimages

  une attaque naïve coûte 2^n