Sécurité Appliquée
Fonctions de hachage

Jean-François COUCHOT
couchot[arobase]femto-st[point]fr

8 février 2022

1 Recherche de collisions

L'algorithme donné à la figure 1 présente une méthode générique de recherche de collisions pour une fonction de hachage f.

Algorithme 8 Recherche générique de collision

Sortie : x, x' tel que $f(x) = f(x')$

$j \leftarrow -1$
tant que *true* **faire**
  choisir aléatoirement $x_i \in \mathcal{X}$
  $y_i = f(x_i)$
  $j \leftarrow$ recherche(y_i)
  si $j \neq -1$ et $x_i \neq x_j$ **alors**
    return (x_i, x_j)
  fin si
  stocker (x_i, y_i)
fin tant que

FIGURE 1 – Extrait de https://www.ssi.gouv.fr/uploads/IMG/pdf/These_Reinhard.pdf

Exercice 1.1 (Implantation simple de l'algorithme donné à la figure 1).

1. *Dans cet algorithme, quel sont les paramètre implicites ?*
2. *Dans cet algorithme, comment implanter les instructions "rechercher(y_i)" et "stocker(x_i, y_i)" ?*
3. *Implanter cet algorithme.*

Exercice 1.2 (Collision sur les 32 premiers bits de sha256). *L'objectif de cet exercice est de détecter des collisions lorsqu'on considère la fonction de hachage* `monSHA256reduitA32bits(m)` *qui ne retourne que les 32 bits (ou de manière équivalente les 8 hexadécimaux) les plus à gauche de l'empreinte SHA256 du message m passé en paramètre. .*

1. *Implanter la fonction* `monSHA256reduitA32bits(m)`.
2. *Puisque la fonction de hachage génère une empreinte sur 32 bits, à partir de combien de messages va-t-on avoir une collision ?*
3. *En utilisant l'algorithme développé à l'exercice précédent, trouver deux messages qui entrent en collision avec* `monSHA256reduitA32bits(m)`.
4. *Etudier statistiquement le nombre d'essais qu'il est suffisant en moyenne de réaliser afin obtenir une collision.*