TD1 - Is My password safe ?

Loïc Rouquette
EPITA Research Laboratory (LRE)
loic.rouquette@epita.fr

> **Note**
> Les exercices ci-dessous sont dérivés du cours de Marine Minier, Professeure à l'Université de Lorraine ainsi que du SEED Labs « One-Way Hash Function and MAC », Wenliang Du, Syracuse University & Naval Postgraduate School, sous licence GNU Free Documentation License, Version 1.2.

Intégrité des documents

Votre serveur (http://loicrouquette.fr:4242) est corrompu, vous devez télécharger le fichier http://loicrouquette.fr:4242/test.txt et vous assurer que le fichier est correct avant d'utiliser son contenu.

Pour vérifier que vous avez reçu le bon fichier vous pouvez utiliser une fonction de condensation. En effet, le condensat représente l'empreinte du fichier, ainsi il est possible d'identifier une fichier sans avoir à retransmettre l'intégralité des informations mais uniquement son empreinte. Comme pour les empreintes digitales, il n'est pas impossible que deux personnes (ou deux fichiers) aient la même empreinte mais la probabilité que cela arrive est faible.

Le condensat du ficher correct est :

```
md5:    83cbba67006bf84788592ce7a1d6357f
sha1:   0d5388e80b951e2cfd56e283be4e2f452b00f48f
sha224: 0e569b924c9ff36df1471c2e94d2245e78e4e3bd06bd790b8608e9db
sha256: 9f01dfaba6bf7aba9d6e31fda1929342918f554f1a12b3dd28ef686bf8fac618
```

Question 1

Vérifiez que le fichier téléchargé est correct. Pour cela, il est possible d'utiliser la commande suivante :

```
md5sum <NOM DU FICHIER>
```

Vous pouvez également utiliser une des fonctions suivantes `sh sha1sum`, `sh sha224sum`, `sh sha256sum` de la même manière.

> **Astuce**
> Pour aller plus vite vous pouvez utiliser la commande suivante :
>
> ```
> rm -f test.txt; wget http://loicrouquette.fr:4242/test.txt; md5sum test.txt
> ```

> La commande `rm -f` va supprimer le fichier `test.txt` s'il existe. La commande `wget http://loicrouquette.fr:4242/test.txt` va télécharger le fichier sur le serveur et pour finir la commande `md5sum test.txt` va calculer le condensat du fichier téléchargé.

Question 2
Vous remarquez que les empreintes du fichier sont différentes, est-ce normal ? Quel analogie pouvez vous faire avec des empreintes humaines ?

Question 3
Quel est l'intérêt d'avoir des empreintes de tailles différentes ?

Sécurité des mots de passe

> **Astuce**
> Aidez vous d'internet pour répondre à question suivante.

Question 4
Calculez ou estimez le nombre de mots de passe possibles dans les cas suivants :

- le mot de passe est un prénom ;
- c'est un mot du dictionnaire ;
- il est composé de quatre chiffres ;
- il est composé de huit caractères alphanumériques (lettres + chiffres).

Question 5
Calculez ou estimez le nombre maximal d'essais que l'on peut autoriser pour rentrer un mot de passe si on veut limiter la probabilité de succès d'une attaque directe à 2^{-11} et que le mot de passe est choisi suivant un des types précédents.

Question 6
Calculez ou estimez le temps nécessaire pour pénétrer dans un système qui est protégé par un mot de passe de chacun des types précédents si on suppose que le système attend une seconde avant de vous redemander le mot de passe.

Il s'agit de calculer l'espérance mathématique.

Question 7
Combien de caractères doit comporter votre mot de passe si vous voulez une sécurité en 2^{80}, 2^{100} et 2^{128} calculs ?

> **Astuce**
> Le nombre de caractère nécessaires pour afficher un nombre N en base b est $log_b(N)$, arrondi au supérieur. Vous pouvez également utiliser la formule : $log_b(N) = ln(N)/ln(b)$.

Question 8
Comment doivent être choisis les mots de passe si vous souhaitez vraiment que la complexité des attaques soit celle que vous visez ?

Sécurtié des fonctions de condensation

Comme présenté en introduction, les fonctions de condensation doivent remplir certains critaires pour être qualifiés de *cryptographiques*. À savoir :

- *résistance à la pré-image :* pour toute valeur de hachage h, il est difficile de trouver un message m tel que $H(m) = h$.
- *résistance à la seconde pré-image :* pour une valeur m_1 donnée, il est difficile de trouver un message m_2 tel que $H(m_1) = H(m_2)$.
- *résistance aux collisions :* il est difficile de trouver deux message différents m_1 et m_2 tels que $H(m_1) = H(m_2)$. Une telle paire est appelée une collision de hachage cryptographique.

L'aléa des fonctions de condensation

Pour comprendre les propriétés des fonctions de condensat, vous verez cette question avec `MD5` et `SHA256`:

Question 9

1. Créez un fichier texte de n'importe quelle taille ;
2. Générez le hache H_1 pour ce fichier en utilisant l'algorithme de condensat spécifié ;
3. Inverser un bit du fichier d'entrée. Cette modification peut être effectuée à l'aide de `sh gedit` ;
4. Générez le hache H_2 pour le fichier modifié ;
5. Observez si H_1 et H_2 sont identiques ou non. Si non, les valeurs sont-elles proches (Vous pouvez compter le nombre de bits différents) ?

> **Astuce**
>
> (1.) Pour créer un nouveau fichier texte vous pouvez utiliser `gedit`.
>
> (3.) Les caractères sont représentés sous une forme binaire dans la machine. Pour modifier uniquement 1 bit il suffit remplacer une lettre du texte par une autre lettre dont le nombre de bit qui diffère est 1.
>
> Par exemple : la lettre `Y` est représentée par le code binaire `01011001` et la lettre `X` par le code `01011000`.
>
> Vous pouvez vous référer à la Table 1 pour avoir la correspondance entre les caractères et les valeurs binaires.

Résistance à la pré-image et résistance aux collisions

Dans cette tâche, nous allons étudier la différence entre les deux propriétés de la fonction de condensation : la résistance à la pré-image et résistance aux collisions. Nous utiliserons la méthode de force brute pour voir combien de temps il faut pour briser chacune de ces propriétés. Étant donné que la plupart des fonctions de hachage sont très résistantes à l'attaque par force brute sur ces deux propriétés, il nous faudrait des années pour les casser à l'aide de l'attaque par force brute. Pour rendre la tâche réalisable, vous avons fourni une fonction de hachage dont la taille de la sortie est paramétrable (*cf.* Figure 1). Vous pouvez télécharger cette fonction ici : https://loicrouquette.fr/teaching/24-25/JMI/is_my_password_safe/hash.zip.

```
def custom_hash(message, mask):
  hash = MD5(message)
  # & is the boolean "AND operator"
  hash = hash & mask
  return hash
```

Figure 1. – A custom (unsafe) hash function.

> **Astuce**
>
> Utiliser cette suite de commandes pour télécharger et décompresser le code.
>
> ```
> wget https://loicrouquette.fr/teaching/24-25/JMI/is_my_password_safe/hash.zip
> unzip hash.zip
> ```

> **Astuce**
>
> La fonction s'utilise comme ceci :
>
> ```
> python3 hash "Hello" -m 0b1111
> ```
>
> Le paramètre `sh -m` indique le nombre de bits que l'on souhaite conservé. Plus ce nombre est petit, plus il est facile de créer des collisions.

Question 10
Utiliser cette fonction de condensat fournie pour essayer de créer des collisions.

Nous souhaitons maintenant vérifier s'il est plus facile de calculer une pré-image ou de calculer des collisions. Pour ce faire nous vous fournissons un code d'exemple: https://loicrouquette.fr/teaching/24-25/JMI/is_my_password_safe/benchmark.zip.

> **Astuce**
>
> Utilisez cette suite de commandes pour télécharger et décompresser le code.
>
> ```
> wget          https://loicrouquette.fr/teaching/24-25/JMI/is_my_password_safe/
> benchmark.zip
> unzip benchmark.zip
> ```

La commande s'utilise comme ceci:

```
python3 benchmark -m 0b1111111 -l 2048
```

Comme pour la fonction `hash`, le paramètre `-m` indique le nombre de bits que nous souhaitons garder pour l'évaluation du hash, plus ce paramètre est petit et plus l'attaque est facile. Le paramètre `-l` indiqe le nombre d'essais effectués. Il sera nécessaire d'augmenter cette valeur si vous augmentez `-m`.

Question 11

1. Combien d'essais vous faudra-t-il pour casser la propriété à sens unique en utilisant la méthode de la force brute ? Vous devez répéter votre expérience plusieurs fois et indiquer votre nombre moyen d'essais.
2. Combien d'essais vous faudra-t-il pour briser la propriété d'absence de collision en utilisant la méthode de force brute ? De même, vous devez indiquer la moyenne.
3. D'après vos observations, quelle propriété est la plus facile à casser en utilisant la méthode de la force brute ?
4. Pouvez-vous expliquer mathématiquement la différence entre vos observations ?

Table 1. – ASCII Table

Letter	ASCII Code	Binary	Letter	ASCII Code	Binary
a	097	01100001	A	065	01000001
b	098	01100010	B	066	01000010
c	099	01100011	C	067	01000011
d	100	01100100	D	068	01000100
e	101	01100101	E	069	01000101
f	102	01100110	F	070	01000110
g	103	01100111	G	071	01000111
h	104	01101000	H	072	01001000
i	105	01101001	I	073	01001001
j	106	01101010	J	074	01001010
k	107	01101011	K	075	01001011
l	108	01101100	L	076	01001100
m	109	01101101	M	077	01001101
n	110	01101110	N	078	01001110
o	111	01101111	O	079	01001111
p	112	01110000	P	080	01010000
q	113	01110001	Q	081	01010001
r	114	01110010	R	082	01010010
s	115	01110011	S	083	01010011
t	116	01110100	T	084	01010100
u	117	01110101	U	085	01010101
v	118	01110110	V	086	01010110
w	119	01110111	W	087	01010111
x	120	01111000	X	088	01011000
y	121	01111001	Y	089	01011001
z	122	01111010	Z	090	01011010