TD 4 : Fonctions de hachage et signature

Exercice 1 (Signature naïve avec RSA **)**.** Prenons $n = pq$, où p et q sont deux nombres premiers différents, et une paire de clés RSA :

Clé publique : (e, n)

Clé secrète : d (où $d = e^{-1} \mod \phi(n)$ et $pgcd(e, \phi(n)) = 1$)

Rappel de l'algorithme de signature avec RSA :

- Signature : $\sigma = m^d \mod n$
- Vérification : $\sigma^e = m \mod n$

1. Montrez qu'un attaquant peut forger la signature d'un message à partir de deux signatures qu'il a interceptées.
2. Soit la clé publique $(e, n) = (5, 91)$, calculer le message ainsi signé à partir des deux signatures 76 et 58 respectivement pour les messages 6 et 11.

Exercice 2 (Signature *)**.** Soient p et q deux (grands) nombres premiers tels que q divise $p - 1$, g un générateur de $\mathbb{Z}/p\mathbb{Z}$ et H une fonction de hachage. Ces paramètres sont partagés par tous les utilisateurs.
Alice se munit d'une paire de clefs $(s_k, p_k = g^{s_k} \mod p)$, avec $0 < s_k < q$. Elle utilise à présent le schéma de signature DSA (Digital Signature Algorithm) ci-dessous :

- Choisir k aléatoire vérifiant $0 < k < q$
- Calculer $r = (g^k \mod p) \mod q$ et $t = k^{-1} \cdot (H(m) + s_k \cdot r) \mod q$ (comme q est premier, k est forcément inversible dans $\mathbb{Z}/q\mathbb{Z}$)
- Signature : $\sigma = (r, t)$

La vérification s'effectue comme suit :

- Vérifier $0 < r < q$ et $0 < t < q$
- Calculer $H(m)$ et $t^{-1} \mod q$ (comme q est premier, t est forcément inversible dans $\mathbb{Z}/q\mathbb{Z}$)
- Vérifier que $(g^{H(m) \cdot t^{-1}} \cdot p_k^{r \cdot t^{-1}} \mod p) \mod q$ est égal à r

1. Vérifiez que la procédure de vérification marche, autrement dit qu'on a bien $r = (g^{H(m) \cdot t^{-1}} \cdot p_k^{r \cdot t^{-1}} \mod p) \mod q$.
2. Alice signe un message m, dont la signature est (r, t). Elle révèle par inadvertance la valeur k associée. Trouvez une attaque. Autrement dit, montrez qu'un attaquant peut calculer s_k.
3. En conséquence, que doit faire Alice de la valeur de k après avoir généré la signature ?

Exercice 3. **

Une *fonction de hachage* est une fonction qui permet de calculer, à partir de n'importe quel message, une chaîne de caractères de longueur limitée, appelée le *haché* du message.

Une fonction de hachage est *déterministe*, c'est-à-dire que le résultat du calcul pour deux messages identiques est toujours le même. Cette propriété est utilisée pour l'authentification par mot de passe. En effet, au lieu de stocker les mots de passe en clair sur un serveur, il est souhaitable de stocker uniquement leurs hachés. Ainsi, l'utilisateur reste le seul à connaître son mot de passe. Pour l'authentification, le serveur calcule le haché du mot de passe entré par l'utilisateur, et vérifie qu'il est bien identique à celui qui est stocké dans sa base de données.
Par ailleurs, une fonction de hachage doit posséder plusieurs propriétés de sécurité. Par exemple, comme l'illustre cette énigme, une fonction de hachage doit être *résistante à la collision*, c'est-à-dire qu'il ne doit pas être facile de trouver deux messages distincts qui ont le même haché.

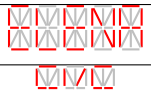
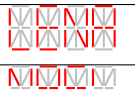
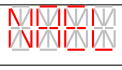
login	H(login)	login	H(login)	login	H(login)
	25		28		22
	24		16		22

FIGURE 1 – Logins et logins hachés.

FIGURE 2 – Alphabet sur un afficheur 14 segments.

- Dans cette énigme, ce ne sont pas les mots de passe des utilisateurs qui sont hachés, à l'aide d'une certaine fonction notée H, mais leurs identifiants. Le tableau de la figure 1 montre les valeurs des hachés de certains identifiants. La figure 2 donne les représentations des 26 lettres de l'alphabet utilisées dans cette énigme. À partir de ces informations, saurez-vous découvrir le calcul auquel correspond la fonction de hachage H, calculer le haché de JAMES et trouver un autre prénom qui provoque une collision avec lui ?

Exercice 4 (Collision fonction de hachages naïves *)**.** En 2013, une des bases de données de la société Adobe, qui contenait plus de 153 millions de mots de passe, a fuité sur Internet. Heureusement, les mots de passe des utilisateurs n'étaient pas stockés en clair, mais *hachés*.
Dans cette énigme, la fonction de hachage est la suivante[1] : un mot de passe W est formé de caractères : chiffres, lettres minuscules ou majuscules, sans accent, sans espace. Le haché $H(W)$ du mot de passe W est la somme des valeurs en ASCII de chacun de ses caractères. Ainsi, le haché du mot de passe `Pi314` est :

$$\begin{aligned} H(\texttt{Pi314}) &= \texttt{ASCII(P)} + \texttt{ASCII(i)} + \texttt{ASCII(3)} + \texttt{ASCII(1)} + \texttt{ASCII(4)} \\ &= 80 + 105 + 51 + 49 + 52 = 337. \end{aligned}$$

Au lieu de stocker sur le serveur, dans une base de données, le mot de passe `Pi314` en clair, il est souhaitable de stocker uniquement son haché, 337. Ainsi, l'utilisateur reste le seul à connaître son

1. Les *fonctions de hachage* utilisées dans la vie réelle sont plus compliquées.

Table de conversion ASCII

Chiffres	ASCII
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55
8	56
9	57

Lettres	ASCII
A	65
B	66
C	67
D	68
E	69
F	70
G	71
H	72
I	73
J	74
K	75
L	76
M	77

Lettres	ASCII
N	78
O	79
P	80
Q	81
R	82
S	83
T	84
U	85
V	86
W	87
X	88
Y	89
Z	90

Lettres	ASCII
a	97
b	98
c	99
d	100
e	101
f	102
g	103
h	104
i	105
j	106
k	107
l	108
m	109

Lettres	ASCII
n	110
o	111
p	112
q	113
r	114
s	115
t	116
u	117
v	118
w	119
x	120
y	121
z	122

mot de passe. Pour l'authentification, lorsque l'utilisateur entre un mot de passe, le serveur en calcule le haché, puis le compare à celui qui est stocké dans la base. Si les deux valeurs sont différentes, il est certain que le mot de passe entré n'est pas correct, et la connexion est refusée. Dans le cas contraire, la connexion est autorisée.

Le hachage des mots de passe est une précaution élémentaire dans la gestion d'une base de données d'utilisateurs. Malheureusement, dans le cas de la fuite, en 2013, de la base de données de la société Adobe, pour chaque login était aussi stocké, en clair, dans la base de données, un *indice*. Cette information était destinée à permettre à l'utilisateur de retrouver son mot de passe en cas d'oubli, mais représentait potentiellement une faille de sécurité, comme illustré dans cette énigme.

- Grâce à la base de données de la figure 3, saurez-vous découvrir trois mots de passe différents de certains utilisateurs ?

LOGIN	INDICE	H(password)
Alice	Yellow	709
Arnaud	Incassable	555
Bart	Élément 74	431
Blaise	Musique	637
Bob	Numbers	964
Camille	Bataille Stalingrad Cubisme	824
Carlton	1 to 9	122
David	Electric	709
Édouard	Noces de Figaro	637
Étienne	Love	169
Ève	Pokémon	709
François	Métal	404
Jules	Mon prénom	515
Lisa	Mendeleiev 74	349
Lucie	< 3	169
Matthieu	Flûte Enchantée	637
Nadia	Naissance Hawking Demoiselles d'Avignon	824
Nathalie	Nintendo	709
Paul	Longueur 16	161
Philippe	Réponse universelle Guernica	824
Rémi	Mendeleiev Mo Peintre	824
Robert	Amadeus	637
Sonia	Hard Rock	666
Stéphane	Mort Celsius Pablo	824
Valéry	Compositeur	637
William	Numbers - 0	779
Xavier	Marche Turque	637

FIGURE 3 – Une base de données de mots de passe avec indices.