Université Paris 8 Année 2021–2022

M1 Mathématiques et applications, parcours ACC et CSSD

Cryptographie à clef publique – Devoir semaine 4

devoir du 18/02/2022
à rendre jusqu'au 04/03/2022

Exercice 1. Cryptosystème de Merkle–Hellman.

En 1978, Merkle et Hellman ont proposé un cryptosystème à clef publique dont la sécurité est fondée sur la difficulté du problème de la somme de sous-ensemble (SUBSET-SUM). Définissons ce problème.

Problème de décision SUBSET-SUM :

> **Instance :** une séquence $S = (s_1, \dots, s_n)$ d'entiers naturels non-nuls et un entier $a \geq 1$.
> **Objectif :** existe-t-il un sous-ensemble $I \subseteq \{1, \dots, n\}$ tel que $\sum_{i \in I} s_i = a$?

Le problème de recherche associé à SUBSET-SUM demande de **calculer** un tel ensemble I. Il a été prouvé que le problème SUBSET-SUM est NP-complet ; par conséquent, on ne connaît pas à l'heure actuelle d'algorithme déterministe qui résout toutes les instances de ce problème en temps polynomial en la taille des entrées.

Néanmoins, pour certaines instances, le problème devient aisé. C'est notamment le cas lorsque la séquence entiers $S = (s_1, \dots, s_n)$ vérifie les contraintes suivantes :

— $0 < s_1 < \cdots < s_n$,
— pour tout $j \in \{1, \dots, n\}$, on a $\sum_{i=1}^{j-1} s_i < s_j$.

Lorsque ces contraintes sont vérifiées, on dit alors que S est **super-croissante**. Par exemple, la séquence $(1, 3, 5, 10, 22)$ est super-croissante, mais $(2, 7, 10, 18, 41)$ ne l'est pas car $18 \leq 2 + 7 + 10$.

Question 1.– Démontrer que si $S = (s_1, \dots, s_n)$ est une séquence super-croissante avec $0 < s_1 < \cdots < s_n$, alors on a $s_i \geq 2^{i-1}$ pour tout $i \geq 1$.

On considère un algorithme « glouton » présenté dans l'Algorithme 1. On pourrait vérifier que cet algorithme résout bien le problème SUBSET-SUM **dans le cas où S est super-croissante**.

Question 2.– Quelle est la complexité de l'Algorithme 1, en nombre d'opérations élémentaires sur les entiers ?

Merkle et Hellman ont proposé un cryptosystème asymétrique, avec comme idée de « cacher » une instance super-croissante du problème en la multipliant par un élément inversible modulo un grand nombre entier. Une description précise de la génération de clefs, du chiffrement et du déchiffrement est donnée dans les Algorithmes 2, 3 et 4.

Question 3.– Vérifier que le système de chiffrement est valide, c'est-à-dire que le déchiffrement d'un chiffré retourne bien le clair d'origine.

Algorithme 1 : Algorithme glouton pour la résolution de SUBSET-SUM dans le cas super-croissant.

Entrée : une séquence $S = (s_1, \ldots, s_n)$ super-croissante et un entier $a \geq 1$
Sortie : un sous-ensemble $I \subseteq \{1, \ldots, n\}$ tel que $\sum_{i \in I} s_i = a$ s'il existe, et « pas de solution » sinon

$x \leftarrow a$
$I = \varnothing$
Pour *i allant de n à* 1 **faire**
    Si $x \geq s_i$
        $I \leftarrow I \cup \{i\}$
        $x \leftarrow x - s_i$
Si $x = 0$
    Retourner I
Sinon
    Retourner « pas de solution »

Algorithme 2 : Génération de clés dans le cryptosystème de Merkle–Hellman

Entrée :
Sortie : une paire de clés publique/privée

Choisir aléatoirement une séquence $S = (s_1, \ldots, s_n)$ super-croissant et un entier $\ell > \sum_{i=1}^{n} s_i$
Choisir un entier e (assez grand) tel que $\text{pgcd}(e, \ell) = 1$
Calculer $d = e^{-1} \mod \ell$
Calculer $B = (b_1, \ldots, b_n)$ où $b_i = e\, s_i \mod \ell$ pour tout $i \in \{1, \ldots, n\}$.
La **clé publique** est B, la **clé privée** est (S, ℓ, d).

Algorithme 3 : Chiffrement dans le cryptosystème de Merkle–Hellman

Entrée : un mot $\boldsymbol{m} = (m_1, \ldots, m_n) \in \{0,1\}^n$, la clé publique $B = (b_1, \ldots, b_n)$
Sortie : un chiffré $c \in \mathbb{N}$

Calculer et retourner $c = \sum_{i=1}^{n} m_i b_i$.

Algorithme 4 : Déchiffrement dans le cryptosystème de Merkle–Hellman

Entrée : la clé privée $(S = (s_1, \ldots, s_n), \ell, d)$ et un chiffré $c \in \mathbb{N}$
Sortie : un message $\boldsymbol{m} = (m_1, \ldots, m_n) \in \{0,1\}^n$

Calculer $a = dc \mod \ell$.
Appliquer l'Algorithme 1 sur l'instance (S, a). Il retourne un ensemble $I \subseteq \{1, \ldots, n\}$.
Retourner $\boldsymbol{m} = (m_1, \ldots, m_n)$ où $m_i = 1$ si $i \in I$, et $m_i = 0$ sinon.

Question 4.– Dans le cas où la clé privée est $S = (4, 7, 13, 31, 58)$, $\ell = 114$ et $d = 37$, retrouver le message associé au chiffré $c = 72$.

Question 5.– On suppose que dans la génération de la paire de clefs, on choisit $s_i \leq 2^{i+1}$ et $\ell \leq 2s_n$.

1. Donner une borne supérieure (en fonction de n), sur le nombre total de clefs privées possibles.
2. En déduire la valeur minimale de n pour qu'une attaque exhaustive sur la clef privée coûte au moins 2^{80} tests de clefs (une attaque exhaustive consiste à tester toutes les clefs possibles).
3. En déduire une borne inférieure sur la taille des clefs pour avoir 80 bits de sécurité.

Question 6.– Démontrer que si un attaquant sait résoudre le problème SUBSET-SUM dans un cadre général, alors il peut déchiffrer n'importe quel message sans connaissance de la clef privée.

Question 7.– Supposons que, pour une clé privée fixée, un attaquant connaisse la valeur des chiffrés des messages $\boldsymbol{m}^{(1)} = (1, 0, \ldots, 0)$, $\boldsymbol{m}^{(2)} = (0, 1, 0, \ldots, 0)$, $\ldots$, $\boldsymbol{m}^{(n)} = (0, \ldots, 0, 1)$. Comment peut-il en déduire le chiffré d'un message $\boldsymbol{m}$ quelconque ? En déduire une attaque à chiffré choisi sur le système.