CRYPTO 1 - TD 4
2024-2025

Énoncé

Exercice 1 :

Soit $\mathbb{G}$ un groupe commutatif (noté multiplicativement). Pour simplifier, on peut considérer que $\mathbb{G} = (\mathbb{Z}/n\mathbb{Z})^*$
Proposer un algorithme qui étant donnés t éléments $g_1, \ldots, g_t$ du groupe $\mathbb{G}$ et des entiers positifs $n_1, \ldots, n_t$ calcule le produit $g_1^{n_1} \ldots g_t^{n_t} \in \mathbb{G}$ en $O(\ell + 2^t)$ multiplications dans $\mathbb{G}$ (où ℓ est la taille en bits de $\max(n_1, \ldots, n_t)$).

Exercice 2 : Algorithme de Shanks

Considérons un groupe multiplicatif cyclique $\mathbb{G}$ engendré par $g \in \mathbb{G}$ d'ordre connu q (autrement dit, nous avons $\mathbb{G} = \{1, g, g^2, \ldots, g^{q-1}\}$). Proposer un algorithme de résolution de logarithme discret par compromis temps-mémoire de complexité $O(\sqrt{q})$ opérations de groupe en temps et $O(\sqrt{q})$ éléments de groupe en mémoire.

Indication. On pourra remarquer que pour tout élément $h = g^x \in \langle g \rangle$, l'entier x s'écrit sous la forme $x = x_1 T + x_0$ avec $0 \leq x_0 < T$ et $0 \leq x_1 < T$ pour $T = \lceil \sqrt{q} \rceil + 1$.

Exercice 3 :

Soient N un module RSA et e un nombre entier premier avec $\varphi(N)$. Considérons un algorithme $\mathcal{A}$ qui prend en entrée un élément de $\mathbb{Z}_N^*$ et retourne un élément de $\mathbb{Z}_N^*$, en temps τ (dans le pire des cas) où τ représente au moins le coût d'une exponentiation dans $\mathbb{Z}_N^*$.

Supposons qu'il existe un sous-ensemble E de $\mathbb{Z}_N^*$ avec $\#E \geq \epsilon N$ et $\epsilon \in]0, 1]$ pour lequel lorsque $\mathcal{A}$ est exécuté sur un élément $x \in E$, l'élément y retourné par $\mathcal{A}$ vérifie $y^e = x \bmod N$.

3.a] Montrer qu'il existe un algorithme $\mathcal{B}$ qui résout le problème RSA dans $\mathbb{Z}_N^*$ en un temps espéré $O(\tau/\epsilon)$.

Exercice 4 :

4.a] Montrer que le protocole de chiffrement RSA naïf n'est pas sémantiquement sûr sous une attaque à clairs choisis.

4.b] Montrer que le protocole de chiffrement RSA naïf est inversible sous une attaque à à un chiffré choisi.

Exercice 5 :

5.a] Décrire une attaque dans le protocole de mise en accord de clé Diffie-Hellman dans laquelle un attaquant *actif* (*i.e.* qui peut modifier les données pendant le protocole Diffie-Hellman) peut ensuite intercepter, déchiffrer et modifier toutes les communications qu'Alice ou Bob chiffrerait avec sa clé.

Exercice 6 :

L'algorithme de chiffrement d'ElGamal est un algorithme de cryptographie asymétrique basé sur le problème du logarithme discret. Il a été créé par T. ELGAMAL en 1985 :

Génération des clés : L'utilisateur choisit un groupe $\mathbb{G}$ d'ordre q dans lequel le problème du logarithme discret est jugé difficile et g un générateur de $\mathbb{G}$. Il tire uniformément aléatoirement $x \in \mathbb{Z}_q^*$ et calcule $y = g^x \in \mathbb{G}$. La clé publique est (q, g, y) et la clé secrète associée est x.

Chiffrement : étant donné un message clair $m \in \mathbb{G}$, l'algorithme de chiffrement tire uniformément aléatoirement $r \in \mathbb{Z}_q^*$ et calcule $c_1 = g^r \in \mathbb{G}$ et $c_2 = m \cdot y^r \in \mathbb{G}$. Le chiffré de m est le couple (c_1, c_2).

Déchiffrement : étant donné un chiffré $(c_1, c_2) \in \mathbb{G}^2$, l'algorithme de déchiffrement retourne (c_2/c_1^x).

6.a] Montrer que le protocole de chiffrement ElGamal n'est pas à sens-unique sous une attaque à un chiffré choisi.