École normale supérieure **2019-2020**
Département d'informatique

Introduction à la cryptologie
TD n° 3 : Correction

Exercice 1 (Multi-exponentiation)**.** L'idée est d'adapter l'algorithme d'exponentiation rapide. Pour tout entier n, on a $n = 2\lfloor n/2 \rfloor + (n \bmod 2)$ (où mod est vu comme une opération dans $\mathbb{N}$ qui choisit le représentant positif minimal). D'où :

$$\begin{aligned}\prod_i g_i^{n_i} &= \prod_i g_i^{n_i \bmod 2} g_i^{2\lfloor n_i/2 \rfloor} \\ &= \left(\prod_i g_i^{n_i \bmod 2}\right) \cdot \left(\prod_i g_i^{\lfloor n_i/2 \rfloor}\right)^2\end{aligned}$$

Le terme de gauche dans le produit ci-dessus ne peut prendre que 2^t valeurs distinctes. Le terme de droite (sous le carré) est du même type que le terme qu'on veut calculer, mais les exposants ont un bit de moins.
On en déduit l'algorithme récursif suivant. On précalcule $\prod_i g_i^{b_i}$ pour chaque $(b_1, \ldots, b_t) \in \{0,1\}^t$; on enregistre les résultats dans une table T. Pour calculer $\prod_i g_i^{n_i}$, on procède de manière récursive. Si tous les exposants sont nuls on renvoie 1. Sinon on utilise l'égalité plus haut, en choisissant le terme approprié dans la table T, et en faisant un appel récursif pour calculer $\prod_i g_i^{\lfloor n_i/2 \rfloor}$. Cet algorithme fait $O(2^t)$ précalculs (on peut se débrouiller pour que le calcul de chaque nouvelle entrée de la table ne coûte qu'une multiplication dans le groupe, par exemple avec un code de Gray) et $O(\ell)$ appels récursifs coûtant chacun deux multiplications dans $\mathbb{G}$.

Exercice 2 (Algorithmes génériques de logarithme discret)**.**

A. Algorithme naïf. On essaie tous les exposants x possibles jusqu'à avoir $g^x = h$.

B. Algorithme de Shanks. On construit une table de hachage T qui à chaque élément hg^{-mq} pour $q < m$ associe q. On calcule ensuite g^r pour chaque $r < m$. Si cette valeur apparaît dans la table T, on retrouve la valeur q associée et on renvoie $x = mq + r$. On voit que l'algorithme est correct : il renvoie x tel que $g^x = h$. D'autre part l'algorithme trouve nécessairement la solution puisque par division euclidienne, le x cherché admet une décomposition de la forme $mq + r$ avec $q, r < m$. La construction de la table et le calcul des g^r coûtent $O(\sqrt{p})$ opérations.

C. Algorithme ρ de Pollard.

1. On procède par récurrence : ces égalités sont vérifiées lors de la première entrée dans la boucle, et elles sont préservées par une itération de la boucle.
2. Supposons que la boucle termine avec $i < p$. On note que les égalités de la question précédentes restent vraies à la sortie de la boucle. On a alors :

$$\begin{aligned}g^x h^i &= g^y h^{2i} \\ g^{x-y} &= h^i.\end{aligned}$$

   Comme $1 \leq i < p$, i inversible modulo p ; soit i^{-1} cet inverse, on obtient $g^{(x-y)i^{-1}} = h$ (on rappelle que $g^p = 1$). La sortie de l'algorithme est donc correcte.
3. Le groupe $\mathbb{G}$ contient p éléments. Les éléments γ_i pour $1 \leq i \leq p+1$ sont au nombre de $p+1$, donc deux au moins d'entre eux doivent être égaux (principe des tiroirs). Il s'ensuit que $j \leq p+1$.
   Soit $\ell = j - k$. On remarque que pour $s, t \geq k$, $\gamma_s = \gamma_t$ ssi $|t - s|$ est un multiple de ℓ. La boucle s'arrête lorsque $\gamma_i = \gamma_{2i}$. Pour cela il (faut et) suffit que $i \geq k$ et $2i - i = i$ est un multiple de ℓ. Comme il y a ℓ valeurs consécutives entre γ_k et γ_{j-1} incluses, l'une d'entre elles doit être un multiple de ℓ. L'algorithme s'arrête sur cette valeur, donc il s'arrête bien pour $i < j$.
4. La question précédente permet de conclure que l'algorithme conclut au bout de $O(p)$ étapes, correspondant à $O(p)$ multiplications dans $\mathbb{G}$, ce qui n'est pas très intéressant : la force brute coûte $O(p)$ multiplications aussi. Par contre si F est uniformément aléatoire, tant que les éléments de le suite (γ_i) restent distincts, chaque nouvel élément est uniformément aléatoire. On est donc dans la situation du

paradoxe des anniversaires : la première collision a lieu au bout de $O(\sqrt{p})$ nouveaux tirages en moyenne, i.e. l'espérance de j est $O(\sqrt{p})$. Par la question précédente, l'algorithme s'arrête donc en $O(\sqrt{p})$ étapes en moyenne, chacune coûtant $O(1)$ multiplications dans $\mathbb{G}$.

Par rapport à l'algorithme de Shanks, noter que le coût en mémoire est en $O(1)$ et non $O(\sqrt{p})$, ce qui est un gain majeur.

Exercice 3 (Logarithme discret de petit poids de Hamming)**.** Le nombre de multiplications effectuées par l'algorithme d'exponentiation dichotomique dépend du nombre de 1 dans le développement en base 2 de l'exposant considéré. Il a donc été suggéré pour rendre les protocoles cryptographiques plus efficaces d'utiliser des clés secrètes où ce nombre, le *poids de Hamming* de l'exposant, est relativement petit. Pour se prémunir d'attaque par recherche exhaustive de la clé secrète, il est nécessaire que le nombre de tels exposants soit suffisamment grand. Le nombre d'exposants de ℓ bits de poids w est donné par le coefficient binomial $\binom{\ell}{w}$. L'exercice montre une adaptation de l'algorithme de Shanks pour résoudre le problème du logarithme discret de petit poids de Hamming en $O(\ell\binom{\lceil \ell/2 \rceil}{\lceil w/2 \rceil})$ exponentiations dans le groupe considéré.
Considérons un groupe multiplicatif cyclique $\mathbb{G}$ engendré par $g \in \mathbb{G}$ d'ordre connu q un nombre premier de ℓ bits (*i.e.* $2^{\ell-1} < q < 2^{\ell}$). Soit w un entier dans $\{1, \ldots, \ell\}$. Nous supposerons que ℓ et w sont pairs.

1. Donner un algorithme pour calculer le logarithme discret dans $\mathbb{G}$ d'un élément h dont le poids de Hamming du logarithme discret est égal à w en $O(\binom{\ell}{w/2})$ exponentiations dans le groupe et dont la complexité en mémoire est $O(\binom{\ell}{w/2})$ éléments de groupe.

Soient N un entier et ℓ et w deux entiers pairs. Un *système de décomposition* de type (N, ℓ, w) est un couple $(X, \mathcal{B})$ tel que
- X est un ensemble de cardinal ℓ ;
- $\mathcal{C}$ est une famille de N sous-ensembles de X de cardinaux $\ell/2$;
- pour tout sous ensemble A de X de cardinal w, il existe un sous-ensemble $C \in \mathcal{C}$ tel que $|A \cap C| = w/2$.

2. Montrer que s'il existe un système de décomposition de type (N, ℓ, w) alors le problème du logarithme discret de poids de Hamming égal à w peut être résolu en $O(N\binom{\ell/2}{w/2})$ exponentiations dans le groupe et en stockant $O(\binom{\ell/2}{w/2})$ éléments de groupe.
3. Posons $X = \mathbb{Z}_\ell$. Montrer que la famille $\mathcal{C} = \{C_i, 0 \leq i \leq \ell/2 - 1\}$, définie par

$$C_i = \{i + j \bmod \ell, 0 \leq j \leq \ell/2 - 1\} \quad \text{pour } 0 \leq i \leq \ell/2 - 1$$

   est un système de décomposition de type $(\ell/2, \ell, w)$.
4. Conclure.

Exercice 4 (Auto-réductibilité du problème du logarithme discret)**.** Soit $\mathbb{G}$ un groupe fini cyclique d'ordre p et g un générateur de $\mathbb{G}$. Considérons un algorithme $\mathcal{A}$ qui prend en entrée un élément de $\mathbb{G}$ et retourne un entier, en temps τ (dans le pire des cas) où τ représente au moins le coût d'une exponentiation dans $\mathbb{G}$.
Supposons qu'il existe un sous-ensemble E de $\mathbb{G}$ avec $|E| \geq \epsilon|\mathbb{G}|$ et $\epsilon \in]0, 1]$ pour lequel lorsque $\mathcal{A}$ est exécuté sur un élément $h \in E$, l'entier retourné par $\mathcal{A}$ est le logarithme discret de h en base g. Considérons l'algorithme $\mathcal{B}$ défini à partir de $\mathcal{A}$ dans l'algorithme (1).

Algorithme 1 Algorithme $\mathcal{B}$

Entrée: $g, h \in \mathbb{G}$
Sortie: $x \in \mathbb{Z}_p$ tel que $h = g^x$.
  tant que VRAI **faire**
    $c \xleftarrow{R} \mathbb{Z}_p$ (c est tiré uniformément aléatoirement dans $\mathbb{Z}_p$)
    $h' \leftarrow g^c$
    $w \leftarrow \mathcal{A}(h \cdot h')$
    si $g^w = h \cdot h'$ **alors**
      retourner $w - c \bmod p$
    fin si
  fin tant que

Montrer que l'algorithme $\mathcal{B}$ résout le problème du logarithme discret dans $\mathbb{G}$ en temps espéré $O(\tau/\epsilon)$.

Exercice 5 (Logarithme discret et Diffie-Hellman)**.** Soit M un entier. Soit $\mathbb{G} = \langle g \rangle$ un groupe fini cyclique d'ordre premier p tel que $p-1$ soit M-friable et soit $h = g^x \in \mathbb{G}$.

1. Donner un algorithme probabiliste qui retourne $\mathfrak{g} \in \mathbb{Z}_p^*$ un générateur de $\mathbb{Z}_p^*$.
2. Montrer que si $x \neq 0 \bmod p$, il existe $y \in \mathbb{Z}_{p-1}$ tel que $x = \mathfrak{g}^y \bmod p$. En déduire que pour retrouver x, il suffit de calculer y modulo tous les diviseurs de $p-1$ qui sont puissances d'un nombre premier.
3. Soit q un diviseur premier de $p-1$. Donner un algorithme qui prenant en entrée h et $\mathfrak{g}$ et disposant d'un oracle qui résout le problème calculatoire de Diffie-Hellman dans $\mathbb{G}$ retourne $y \bmod q$ en faisant au plus $O(\log p)$ requêtes à l'oracle et $O(\sqrt{q})$ exponentiations dans $\mathbb{Z}_{p-1}$ et $\mathbb{G}$.
4. Adapter l'algorithme précédent pour qu'il retourne $y \bmod q^e$ lorsque q^e est une puissance d'un nombre premier divisant $p-1$.
5. Conclure.