Exercices et problèmes de cryptographie

Damien Vergnaud

Exercice complémentaire n° 5.10 (bis)

Les *attaques par canal auxiliaire* exploitent des fuites d'information révélées par un algorithme de cryptographie lors de son implantation. Les analyses de consommation de courant sur une carte à puce permettent par exemple de découvrir des informations secrètes lorsque certaines opérations, plus coûteuses, augmentent la consommation électrique du circuit. La courbe représentative de la consommation de courant lors d'une exponentiation discrète peut ainsi révéler de l'information sur les opérations effectuées pendant l'exponentiation (soit un carré, soit une multiplication).

Le but de l'exercice suivant est de proposer des algorithmes de calcul de logarithme discret lorsqu'une une information partielle sur la chaîne de « *carrés et multiplications* » utilisée lors du calcul de l'exponentiation est révélée. Notons que cette connaissance ne révèle pas immédiatement de l'information sur la représentation binaire de l'exposant et l'algorithme de l'exercice précédent ne s'applique donc pas immédiatement.

Exercice 5.10 (bis) LOGARITHME DISCRET AVEC INFORMATION PARTIELLE

Considérons un groupe multiplicatif cyclique $\mathbb{G}$ engendré par $g \in \mathbb{G}$ d'ordre premier connu q et soit $h = g^x$ un élément du sous-groupe engendré par g.

Nous supposons que la chaîne de « *carrés et multiplications* » lors d'une exponentiation discrète est représentée par un mot dans l'alphabet $\{C, M\}$. Par exemple, pour $x = 43$, nous obtenons la chaîne CCMCCMCM et les opérations successives

$$g \xrightarrow{C} g^2 \xrightarrow{C} g^4 \xrightarrow{M} g^5 \xrightarrow{C} g^{10} \xrightarrow{C} g^{20} \xrightarrow{M} g^{21} \xrightarrow{C} g^{42} \xrightarrow{M} g^{43} = h$$

En supposant qu'une attaque par canal auxiliaire a identifié la chaîne de « *carrés et multiplications* » de x sauf en n positions parmi lesquelles on sait que i sont des multiplications. Montrer qu'il est possible de retrouver x par un compromis temps-mémoire en $O(n^{i/2+1} \log q)$ opérations de groupe et $O(n^{i/2+1})$ éléments de groupe en mémoire.

Solution : L'algorithme naïf consiste à vérifier pour tous les choix possibles de i positions parmi n les éléments de la chaîne de « *carrés et multiplications* qui sont des multiplications et ceux qui sont des carrés. Cet algorithme a une complexité de $O(\binom{n}{i})$ exponentiations dans $\mathbb{G}$.

Une première approche consiste à remarquer que dans la chaîne de « *carrés et multiplications* », il n'y a jamais deux multiplications successives. Nous pouvons donc supposer que les éléments inconnus de la chaîne de « *carrés et multiplications* » se trouvent toujours après des carrés.

Notons ℓ la longueur totale de la chaîne et considérons (sans perte de généralité) le cas où i est pair et supposons que l'algorithme connaît la position $j \in \{1, \ldots, \ell\}$ de la chaîne de « *carrés et multiplications* » pour laquelle il y a $i/2$ multiplications à droite de cette position et $i/2$ multiplications à gauche. Notons n_1 et n_2 le nombre de symboles inconnus dans les parties gauche et droite de la chaîne (respectivement). La position j et le nombre de multiplications dans chaque partie de la chaîne étant connus, l'algorithme connaît la longueur en bits de l'exposant correspondant à chaque partie de la chaîne. Notons x_1 l'exposant correspondant à la partie gauche et x_2 l'exposant correspondant à la partie droite. Nous savons que $h = g^x = g^{x_1 2^\alpha + x_2}$ où α est la longueur en bits de la partie droite de la chaîne. Cette valeur α est le nombre de carrés dans la partie droite de la chaîne qui est égal à la somme du nombre de carrés connus dans la chaîne et de $n_2 - i/2$. L'algorithme est alors simplement une variante de l'algorithme de Shanks :

- l'algorithme calcule tous les entiers x_2 obtenus en complétant la chaîne à droite de la position j avec $i/2$ multiplications (placées non consécutivement) et $n_2 - i/2$ carrés ;
- il teste ensuite pour chaque entier x_1 obtenu en complétant la chaîne à gauche de la position j avec $i/2$ multiplications (placées non consécutivement) et $n_1 - i/2$ carrés, si $h \cdot (g^\alpha)^{-x_1}$ est une des valeurs g^{x_2} préalablement calculées.

La complexité de cet algorithme en nombre d'exponentiations dans $\mathbb{G}$ est de

$$\binom{n_2}{i/2} + \binom{n_1}{i/2} \leq \binom{n}{i/2} \leq n^{i/2}.$$

Puisque la position j est en réalité inconnue de l'algorithme, celui ci doit la « deviner » en faisant une recherche exhaustive parmi les $n - i/2$ premières positions inconnues. Le coût total de l'algorithme est donc en $O(n^{i/2+1})$ exponentiations dans $\mathbb{G}$. □