**École normale supérieure** **2022-2023**
**Département d'informatique**

# Introduction à la cryptologie
# TD n° 3 : Partage de secret et cryptologie distribuée.

**Exercice 1** (Partage de secret pour toute structure d'accès)**.** Soient $n$ utilisateurs indexés par les entiers $[\![1, n]\!]$. On considère la structure d'accès définie par $\mathcal{A}$, qui est un ensemble d'ensembles d'utilisateurs : $\mathcal{A} \subseteq \mathcal{P}([\![1, n]\!])$. On suppose que $\mathcal{A}$ est clos par inclusion. Proposer un protocole de partage de secret (non efficace) pour la structure d'accès définie par $\mathcal{A}$, c'est-à-dire : un sous-ensemble d'utilisateurs $i_1, \ldots, i_k$ peut recouvrer le secret si et seulement si $\{i_1, \ldots, i_k\} \in \mathcal{A}$.

**Exercice 2** (Graphes et partage de secret)**.** Soit $n \geq 2$ un entier et considérons le graphe complet non orienté $G = (V, E)$ avec $V = \{1, \ldots, n\}$ et $E = \{\{i, j\}, 1 \leq i < j \leq n\}$. Nous considérons $\mathcal{A}$ l'ensemble formé par tous les ensembles d'arêtes de $V$ contenant un chemin dans $G$ du sommet 1 au sommet $n$. Proposer un protocole de partage de secret (d'un bit) pour la "structure d'accès" $\mathcal{A}$ (*i.e.* l'information possédée par tout ensemble d'arêtes de $\mathcal{A}$ permet de reconstruire le secret mais celle possédée par un ensemble d'arêtes qui n'appartient pas à $\mathcal{A}$ ne révèle aucune information sur le secret).

**Exercice 3** (Partage de secret pour signature à vérificateur designé)**.** Considérons un groupe $\mathbb{G}$ d'ordre premier $q$ généré par $g$. Soit $y = g^x \in \mathbb{G}$. Considérons le protocole suivant par lequel Alice veut prouver sa connaissance de $x$ ($g$, $y$ étant publics).

**Engagement :** Alice tire uniformément aléatoirement $k \in \mathbb{Z}_q^*$ et calcule $r = g^k \in \mathbb{G}$. Elle envoie $r$ à Bob.

**Challenge :** Bob répond en envoyant un élément $c \in \mathbb{Z}_q$ tiré uniformément aléatoirement.

**Réponse :** Alice répond en envoyant $s = k - cx \bmod q$. Bob accepte si $r = g^s y^c$ dans le groupe $\mathbb{G}$.

Nous allons construire un schéma de signature à vérificateur designé à partir des signatures Schnorr. Une signature à vérificateur designé doit convaincre uniquement le vérificateur désigné.

1. Rappelez comment on peut construire un schéma de signature à partir de la preuve zero-knowledge ci-dessus.
2. Rappelez comment on peut simuler un transcript de la preuve ci-dessus (pour prouver la propriété zero-knowledge).
3. Construisez une preuve de connaissance d'un de deux logarithmes discrets de $y_0 = g^{x_0}$ ou $y_1 = g^{x_1}$. Quel est le langage associé ? Quel est un témoin pour ce langage ? **Indication.** Partager le challenge $c$ en deux parties $c_0, c_1$ telles que $c = c_0 \oplus c_1$. Réaliser le protocole plus haut pour la connaissance de $x_0$ et pour la connaissance de $x_1$, mais en utilisant cet unique challenge $c$. La question précédente pourra être utile.
4. Prouver que votre schéma est correcte, robuste et zero-knowledge.
5. Construisez un schéma de signature avec clé publique $y_0$ pour un verificateur designé dont la clé publique est $y_1 = g^{x_1}$. Pourquoi le schéma cache-t-il le témoin utilisé ? Pourquoi le schéma est-il à vérificateur designé ?

**Exercice 4** (Problème de la demande en mariage)**.** Alice et Bob sont en couple depuis de nombreuses années et se posent la question du mariage... Cependant, aucun des deux n'ose faire sa demande de peur de subir l'affront d'un refus de son partenaire. En utilisant le protocole d'ElGamal partagé à deux joueurs, proposer un protocole sécurisé pour ce problème où Alice et Bob apprennent si ils sont d'accord pour se marier mais n'obtiennent aucune information sur le choix de leur partenaire (autrement dit, Alice a un bit $x \in \{0,1\}$, Bob a un bit $y \in \{0,1\}$ et à la fin du protocole les deux partenaires apprennent la valeur de $xy \in \{0,1\}$, mais aucune autre information).

**Exercice 5** (Sécurité du protocole de signature de Groth)**.** Soit $\mathbb{G}$ un groupe d'ordre premier $q$ engendré par $g$ où le problème du logarithme discret est supposé difficile. Nous considérons le protocole de signature suivant qui a été proposé par Jens Groth en 2006.

**Génération des clés :** le signataire tire uniformément aléatoirement trois entiers $x_1, x_2$ et $x_3$ dans $\mathbb{Z}_q^*$ et calcule $y_1 = g^{x_1}$, $y_2 = g^{x_2}$ et $y_3 = g^{x_3}$. La clé publique est $(y_1, y_2, y_3) \in \mathbb{G}^3$ et la clé secrète est le triplet $(x_1, x_2, x_3) \in \mathbb{Z}_q^3$.

**Signature :** pour signer un message $m \in \mathbb{Z}_q$, le signataire tire uniformément $r_1$ dans $\mathbb{Z}_q$ et calcule $r_2 = (x_3 - m - x_1 r_1)/x_2 \bmod q$. La signature est le couple $(r_1, r_2) \in \mathbb{Z}_q^2$.

**Vérification :** étant donné un message $m \in \mathbb{Z}_q$ et une signature supposée $(r_1, r_2)$, l'algorithme accepte la signature si et seulement si

$$g^m \cdot y_1^{r_1} \cdot y_2^{r_2} = y_3. \tag{1}$$

1. Montrer que ce protocole de signature numérique n'est pas résistant aux contrefaçons universelles sous une attaque à deux messages choisis.

Le but de cet exercice est de montrer que ce protocole de signature numérique est résistant aux contrefaçons existentielles sous une attaque à un message choisi sous l'hypothèse de la difficulté du problème du logarithme discret dans $\mathbb{G}$.
Nous considérons un algorithme (probabiliste) $\mathcal{A}$ qui retourne une contrefaçon existentielle sous une attaque à un message choisi avec une probabilité $\varepsilon$ en temps $\tau$. Nous allons construire explicitement un algorithme probabiliste $\mathcal{B}$ qui résout le problème du logarithme discret dans $\mathbb{G}$ avec une probabilité $\varepsilon' \geq \varepsilon/2$ en temps $\tau' \simeq \tau$. L'algorithme $\mathcal{B}$ prend en entrée un élement $h \in \mathbb{G}$ et doit retourner l'entier $x \in \mathbb{Z}_q$ tel que $h = g^x$.

2. Notons $m$ le message pour lequel $\mathcal{A}$ demande une signature et $(r_1, r_2)$ la signature qu'il obtient. Montrer que si l'algorithme $\mathcal{A}$ produit une contrefaçon existentielle $(r_1^*, r_2^*)$ sur un message $m^* \neq m$, alors $(r_1^*, r_2^*) \neq (r_1, r_2)$.
3. Dans cette question, nous considérons uniquement des adversaires $\mathcal{A}$ qui retournent une contrefaçon telle que $r_2^* \neq r_2$.
   (a) L'algorithme $\mathcal{B}_1$ va exécuter l'algorithme $\mathcal{A}$ sur la clé publique $(y_1, y_2, y_3) \in \mathbb{G}^3$ formée de la façon suivante :
   $$y_1 = g^{a_s}, y_2 = h \text{ et } y_3 = g^{b_s} h^{c_s}$$
   où $a_s, b_s, c_s$ sont tirés uniformément aléatoirement dans $\mathbb{Z}_q^*$. Montrer que la distribution de $(y_1, y_2, y_3)$ est identique à celle produite par l'algorithme de génération de clés du protocole et que l'algorithme $\mathcal{A}$ n'a aucune information sur la valeur de $c_s$.
   (b) Montrer que lorsque l'algorithme $\mathcal{A}$ demande une signature sur le message $m$, l'algorithme $\mathcal{B}_1$ peut retourner une signature distribuée de façon identique à celles produites par l'algorithme de signature.
   (c) Montre que si l'algorithme $\mathcal{A}$ produit une contrefaçon existentielle valide sur un message $m^* \neq m$, l'algorithme $\mathcal{B}$ peut en déduire le logarithme discret de $h$ avec probabilité 1.
4. Construire de même un algorithme $\mathcal{B}_2$ qui résout le problème du logarithme discret à partir d'un algorithme $\mathcal{A}$ qui retourne une contrefaçon telle que $r_1^* \neq r_1$.
5. Conclure.