École normale supérieure **2018-2019**
Département d'informatique

Introduction à la cryptologie
TD n° 6 :Correction.

Exercice 1 (Partage de secret pour toute structure d'accès)**.** On rappelle le protocole de partage de secret (n, n) où n utilisateurs partagent un secret, et ce secret ne peut être retrouvé que si les k utilisateurs collaborent ensemble. Pour cela, considérons un secret $S \in \{0,1\}^\lambda$. On tire de manière uniformément aléatoire $(S_1, \ldots, S_n) \in (\{0,1\}^\lambda)^n$ conditionné à : $\sum S_i = S$. Ici la somme est sur $\mathbb{F}_2^\lambda$: c'est un XOR. De manière équivalente, on tire uniformément aléatoirement $S_1, \ldots, S_{n-1}$ et on fixe $S_n = S - \sum_{i<n} S_i$: cette distribution est identique à la précédente. On voit que les n utilisateurs peuvent retrouver S en calculant la somme de leurs parts. Par contre, pour un sous-ensemble strict des n utilisateurs, la distribution des valeurs qu'il connaissent est uniformément aléatoire et indépendante ; en particulier ell est indépendante de S.
Nous arrivons à la question de l'exercice. Soient $I_1, \ldots, I_m$ les éléments minimaux pour l'inclusion dans $\mathcal{A}$. L'idée est simplement de réaliser, pour chaque I_k, une instance indépendante du protocole ci-dessus. Si un ensemble $A \subseteq [\![1, n]\!]$ d'utilisateurs est dans $\mathcal{A}$, il existe $I_k \subseteq A$ et les utilisateurs dans I_k peuvent calculer le secret grâce à l'instance correspondante. Par contre, pour $A \notin \mathcal{A}$, la distribution des valeurs connues par les utilisateurs de A, dans toute les instances, est un ensemble de valeurs indépendantes et uniformément aléatoires, en particulier indépendantes de S.

Exercice 2 (Graphes et partage de secret)**.** Soit $S \in \{0,1\}^n$ le secret. On associe un label dans $\{0,1\}^n$ à chaque sommet de G de la manière suivante. Le sommet 1 a le label $\ell_1 = 0$. Le sommet n a le label $\ell_n = S$. Pour $1 < i < n$, le sommet i a un label ℓ_i tiré uniformément aléatoirement et de manière indépendante dans $\{0,1\}^n$. On associe ensuite à l'arête (i, j) la valeur $a_{i,j} = \ell_i \oplus \ell_j$. L'utilisateur associé à l'arête (i, j) reçoit cette valeur.
On identifie les utilisateurs avec les arêtes du graphe. Si un ensemble A d'utilisateurs contient un chemin du sommet 1 au sommet n, il suffit de calculer la somme des $a_{i,j}$ le long de ce chemin pour trouver S. Réciproquement, si un ensemble d'utilisateurs A ne contient pas de tel chemin, alors considérons le graphe dont les arêtes sont dans A. Dans ce graphe, les sommets 1 et n sont dans des composantes connexes distinctes (sinon il existerait un chemin). Soit C la composante connexe contenant n. Prenons une valeur δ arbitraire dans $\{0,1\}^n$. Définissons la fonction f_δ sur les labels des sommets telle que $f_\delta((\ell_i)_{i\leq n}) = (\ell'_i)_{i\leq n}$ avec :

$$\ell'_i = \begin{cases} \ell_i \oplus \delta \text{ si } i \in C \\ \ell_i \text{ sinon.} \end{cases}$$

Cette fonction translate donc les labels des sommets par δ sur C, et laisse les autres labels inchanges. On remarque que cette fonction conserve les valeurs $a_{i,j}$, quel que soit le choix de δ. Par contre elle correspond au secret $S' = S \oplus \delta$ et non plus S. Le point crucial est les suivant : f_δ est une bijection entre l'ensemble des labels ℓ_i correspondant au secret S et compatibles avec les valeurs $a_{i,j}$ connues des utilisateurs de A, et l'ensemble des ℓ'_i correspondant au secret S' et compatibles avec les (mêmes) valeurs $a_{i,j}$ connues des utilisateurs de A. Comme toutes les distributions sont uniformes, il s'ensuit que la probabilité que le secret soit S ou soit S' conditionné à la connaissance des utilisateurs de A est la même. Les secrets S et S' ont donc la même probabilité, pour n'importe quel S' ($\delta = S' \oplus S$ étant arbitraire) : les utilisateurs de A n'apprennent donc rien sur la valeur de S.

Exercice 3 (Partage de secret pour linear-span programs)**.** Soit $S \in \{0,1\}^n$ le secret à partager. On tire n valeurs $S_1, \dots, S_n$ uniformément aléatoirement dans $\{0,1\}^n$ conditionné à : $\sum S_i = S$. Soit $\vec{S}$ le vecteur $(S_1, \dots, S_n)$. Chaque utilisateur U_i correspondant à un vecteur $\vec{u_i} \in \mathbb{Z}_p^n$ reçoit la valeur $V_i = \vec{u_i} \cdot \vec{S}$, où $\cdot$ dénote le produit scalaire. Si un ensemble d'utilisateurs possèdent des vecteurs $\vec{u_{i_1}}, \dots, \vec{u_{i_k}}$ tels qu'il existe une combinaison linéaire $\sum \lambda_j \vec{u_{i_j}} = (1, \dots, 1)$, alors on $\sum \lambda_j V_{i_j} = \sum \lambda_j \vec{u_{i_j}} \cdot \vec{S} = (1, \dots, 1) \cdot \vec{S} = S$, donc on peut retrouver le secret. On indique brièvement une manière de montrer la réciproque. Supposons qu'on a un ensemble d'utilisateurs associés aux vecteurs $\vec{u_{i_1}}, \dots, \vec{u_{i_k}}$ tel que $(1, \dots, 1) \notin \langle \vec{u_{i_1}}, \dots, \vec{u_{i_k}} \rangle$. Alors comme dans l'exercice précédent, si on fixe les valeurs V_{i_j} connues des utilisateurs, on peut créer une bijection entre les choix de $\vec{S}$ donnant lieu à ces valeurs et tels que $\sum S_i = S$, et les choix de $\vec{S}$ donnant lieu à ces même valeurs et tels que $\sum S_i = S'$ pour un $S' \neq S$ arbitraire. Pour cela, on peut voir $\vec{S}$ comme l'application linéaire $\mathcal{L}$ qui à chaque élément de la base canonique (e_i) associe le vecteur S_i (vu comme un vecteur de $\mathbb{F}_2^n$). Les utilisateurs connaissent l'image de cette application sur un sous-espace strict E de $\{0,1\}^n$; puisque le vecteur $(1, \dots, 1)$ est dans le complémentaire de cet espace, on peut modifier la valeur de $\mathcal{L}$ sur $(1, \dots, 1)$ arbitrairement, en laissant fixe sa valeur sur E.

Exercice 4 (Problème de la demande en mariage)**.** Une manière de faire est la suivante.

1. Alice et Bob génèrent une clef ElGamal partagée suivant le protocole vu en cours, puis chiffrent le message « mariage ! » avec la clef publique, pour obtenir un chiffré c_0.
2. Si Alice veut se marier, elle re-randomise le chiffré c_0 en un c_1 (on rappelle qu'ElGamal permet de rerandomiser le chiffré d'un message m, sans modifier le message et sans connaître la clef : il suffit de multiplier le composant de gauche du chiffré par g^s, et le composant de droite par y^s, où y est la clef publique et s est quelconque). Si elle ne veut pas se marier, elle remplace les deux composants du chiffré par des valeurs aléatoires uniformes.
3. Bob fait de même en partant de c_1 pour obtenir le chiffré c_2.
4. Alice et Bob déchiffrent de manière conjointe le chiffré c_2 suivant le protocole vu en cours. S'ils obtiennt le message « mariage ! », c'est que les deux souhaitent se marier.

Exercice 5 (Sécurité du protocole de signature de Groth)**.**

1. On demande les chiffrées des messages $m = 0$ et $m = 1$. On obtient r_1, r_2, r_1', r_2' tels que :

   $$y_1^{r_1} y_2^{r_2} = y_3 \qquad g y_1^{r_1'} y_2^{r_2'} = y_3$$

   On peut alors forger une signature pour un message quelconque m, en effet on a :

   $$y_1^{(m-1)r_1} y_2^{(m-1)r_2} = y_3^{m-1}$$
   $$g^m y_1^{mr_1'} y_2^{mr_2'} = y_3^m$$

   donc en combinant :

   $$g^m y_1^{mr_1' - (m-1)r_1} y_2^{mr_2' - (m-1)r_2} = y_3.$$

   La signature $(mr_1' - (m-1)r_1, mr_2' - (m-1)r_2)$ est donc valide pour le message m.
2. L'équation (1) de l'énoncé implique qu'une signature (r_1, r_2) ne peut être valide que pour un unique message m (le log discret de $y_3 y_1^{-r_1} y_2^{-r_2}$ en base g).
3. (a) Les variables y_1, y_2, y_3 sont indépendantes (chacune est uniformément aléatoire), seule y_3 dépend de c_s donc nous pouvons limiter notre attention à y_3. Il suffit de remarquer que la distribution de y_3 conditionnée à $c_s = c$, pour tout c fixé, reste uniformément aléatoire à cause du choix uniforme de b_s, en particulier elle ne dépend pas de c.

(b) On vérifie que la signature $(b_s - m - a_s, c_s)$ est valide. D'autre part on a vu que c_s est uniformément aléatoire même conditionné à la clef publique, et $r_1 = b_s - m - a_s$ est l'unique exposant donnant une signature correcte pour $r_2 = c_s$. C'est donc la même distribution qu'une signature véritable : en effet le même relation existe entre r_1 et r_2 dans une signature générée en suivant le protocole (le protocole tel qu'il est écrit tire r_1 uniformément et déduit r_2, mais on voit que les rôles de r_1 et r_2 sont complètement symétriques et qu'on peut faire l'inverse).

(c) Supposons que $\mathcal{A}$ produit une contrefaçon sur un message $m^* \neq m$. Il produit donc r_1^*, r_2^* tels que :

$$\begin{aligned} g^{m^*} y_1^{r_1^*} y_2^{r_2^*} &= y_3 \\ g^{m^*} g^{a_s r_1^*} h^{r_2^*} &= g^{b_s} h^{c_s} \\ g^{(m^* + a_s r_1^* - b_s)(c_s - r_2^*)^{-1}} &= h. \end{aligned}$$

L'inversion $(c_s - r_2^*)^{-1}$ est valide parce qu'on a supposé $r_2^* \neq r_2 = c_s$.

4. Comme déjà remarqué plus haut, les rôles de r_1 et r_2 sont complètement symétriques dans ce protocole, donc on peut réécrire le même raisonnement en échangeant les rôles de y_1 (et ses exposants) et y_2 (et ses exposants).

5. Soit $\mathcal{B}$ l'algorithme qui tire $b \leftarrow \{0, 1\}$ uniformément aléatoirement et qui exécute $\mathcal{B}_b$ (qui fait lui-même appel à $\mathcal{A}$). On a vu que la distribution de la clef publique, et de la signature du message demandé par $\mathcal{A}$ sont identiques à celle des clefs publiques et signatures légitimes (en particulier elles sont identiques pour les deux choix de b). L'algorithme $\mathcal{A}$ produit donc une signature forgée (r_1^*, r_2^*) pour un message m^* avec probabilité ϵ, après avoir éventuellement demandé la signature (r_1, r_2) d'un message choisi m. Nous avons vu que $(r_1^*, r_2^*) \neq (r_1, r_2)$, donc $r_1 \neq r_1^*$ ou $r_2 \neq r_2^*$, donc avec probabilité au moins 50% l'hypothèse de l'algorithme $\mathcal{B}_b$ est satisfaite. Avec probabilité au moins $\epsilon/2$ on retrouve donc le logarithme discret de h en base g.