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

Introduction à la cryptologie
TD n° 10 : hypothèses minimales.

Exercice 1 (Multi-party Computation)**.** Pour $\vec{v} \in \mathbb{Z}_2^n$ et $i \in \{1, \ldots, n\}$, on note v_i la i-ième coordonnée du vecteur $\vec{v}$. Pour $\vec{v}, \vec{w} \in \mathbb{Z}_2^n$, $\vec{v} + \vec{w}$ désigne le vecteur somme, où la somme est calculée coordonnée par coordonnée modulo 2. On écrira parfois aussi $\vec{v} - \vec{w}$, qui est équivalent. Dans la suite, on dit qu'un triplet $(\vec{a}, \vec{b}, \vec{c})$ *encode* un secret $\vec{s}$ ssi $\vec{a} + \vec{b} + \vec{c} = \vec{s}$.

Alice, Bob et Charlie partagent un secret $\vec{s} \in \mathbb{Z}_2^n$: Alice possède $\vec{a} \in \mathbb{Z}_2^n$, Bob possède $\vec{b} \in \mathbb{Z}_2^n$, et Charlie possède $\vec{c} \in \mathbb{Z}_2^n$, tels que $\vec{a} + \vec{b} + \vec{c} = \vec{s}$. Seule Alice connaît $\vec{a}$, seul Bob connaît $\vec{b}$, seul Charlie connaît $\vec{c}$. Le triplet $(\vec{a}, \vec{b}, \vec{c})$ a été tiré uniformément parmi les triplets dans $(\mathbb{Z}_2^n)^3$ qui satisfont la condition $\vec{a} + \vec{b} + \vec{c} = \vec{s}$. (De manière équivalente : $\vec{a}$ et $\vec{b}$ ont été tirés uniformément et indépendamment dans $\mathbb{Z}_2^n$, et $\vec{c} = \vec{s} - \vec{a} - \vec{b}$.)

1. Montrer que si Alice et Bob mettent en commun leur connaissance de $\vec{a}$ et $\vec{b}$, ils n'apprennent rien sur $\vec{s}$, au sens où la distribution de $\vec{s}$ reste uniforme de leur point de vue.

Alice, Bob et Charlie veulent maintenant se partager un secret qui encode la somme $s_i + s_j$ de deux bits de $\vec{s}$. Pour cela, Alice calcule $a' = a_i + a_j$, Bob calcule $b' = b_i + b_j$, et Charlie calcule $c' = c_i + c_j$.

1. Montrer que (a', b', c') est uniforme parmi les triplets (x, y, z) qui encodent $s_i + s_j$.
   Indication : de manière équivalente, on peut montrer que x et y sont uniformes et indépendants, et $z = s_i + s_j - x - y$.

De cette manière, Alice, Bob et Charlie peuvent « calculer » une somme de bits du secret $\vec{s}$, au sens où ils se partagent un nouveau secret qui encode la somme. Le trio souhaite maintenant « calculer » une *multiplication* $s_i s_j$, dans le même sens. Pour cela, Alice, Bob et Charlie tirent uniformément un bit r_a, r_b, r_c respectivement. Alice envoie (r_a, a_i, a_j) à Charlie, Charlie envoie (r_c, c_i, c_j) à Bob, Bob envoie (r_b, b_i, b_j) à Alice. Alice calcule $a' = a_i a_j + a_i b_j + a_j b_i + r_a - r_b$. Bob calcule $b' = b_i b_j + b_i c_j + b_j c_i + r_b - r_c$. Charlie calcule $c' = c_i c_j + c_i a_j + c_j a_i + r_c - r_a$.

1. Montrer que (a', b', c') est à nouveau uniforme parmi les triplets qui encodent $s_i s_j$. Que se passe-t-il si on enlève les r_i ?
2. Proposer une manière de calculer la négation d'un bit de secret s_i, au même sens qu'on a calculé l'addition et la multiplication dans les questions précédentes.
3. En itérant les idées précédentes, proposer un protocole qui permet à Alice, Bob et Charlie de calculer $F(\vec{s})$ pour une fonction arbitraire F. La fonction F est décrite sous la forme d'un circuit booléen (publique, connu de tous). Au terme du calcul, Alice, Bob et Charlie doivent connaître le résultat du calcul, mais ne doivent rien apprendre sur le secret $\vec{s}$, à part $F(\vec{s})$. (On ne demande pas de preuve.)

Exercice 2 (Post-quantum Signatures)**.** On fixe une fonction de hachage H. Pour $\vec{x} \in \mathbb{Z}_2^n$ uniforme, on suppose qu'étant donné $\vec{y} = \mathsf{H}(\vec{x})$, et seulement $\vec{y}$, trouver une préimage de $\vec{y}$ est un problème difficile (c'est-à-dire : trouver $\vec{x'} \in \mathbb{Z}_2^n$, non nécessairement distinct de $\vec{x}$, tel que $\mathsf{H}(\vec{x'}) = \vec{y}$ est difficile). Dans la suite, on va utiliser une technique d'engagement (*commitment* en anglais), comme en cours. En bref, s'engager sur une valeur x signifie tirer une valeur uniforme r, et envoyer $\mathsf{H}(x \parallel r)$, où $\parallel$ dénote la concaténation. Ouvrir l'engagement signifie révéler x et r.

Sylvie tire $\vec{s} \in \mathbb{Z}_2^n$ uniformément, et publie $\vec{y} = \mathsf{H}(\vec{s})$. Sylvie souhaite effectuer une preuve zero-knowledge de connaissance d'une préimage de $\vec{y}$.
Pour prouver à Thomas sa connaissance d'une préimage de $\vec{y}$, Sylvie utilise le protocole suivant.

(a) Sylvie partage son secret $\vec{s}$ entre trois entités virtuelles Alice, Bob et Charlie, en tirant $\vec{a}$, $\vec{b}$, $\vec{c}$ tels que $\vec{a}+\vec{b}+\vec{c}=\vec{s}$, comme dans l'exercise précédente. Noter qu'Alice, Bob et Charlie ne sont que des personnes virtuelles imaginées par Sylvie : en réalité, Sylvie connaît les secrets de tout le monde. Elle effectue ensuite le calcul de $\mathsf{H}(\vec{s})$ en suivant le processus de calcul multipartite de l'exercise précédente. Elle envoie à Thomas les informations suivantes : un engagement sur le secret $\vec{a}$ d'Alice, un engagement sur toutes les valeurs calculées en sortie de porte logique par Alice (les valeurs a' des questions 2 et 3), ainsi que les bits aléatoires r_a utilisés (question 3) ; de même pour Bob et Charlie. [1] Sylvie envoie également les valeurs $\vec{a}_f$, $\vec{b}_f$, $\vec{c}_f$, obtenues par Alice, Charlie et Bob au terme du calcul (et telles que $\mathsf{H}(\vec{s}) = \vec{a}_f + \vec{b}_f + \vec{c}_f$).

(b) Thomas choisit une personne parmi Alice, Bob et Charlie, et envoie son choix à Sylvie.

(c) Sylvie ouvre tous les engagements relatifs aux deux personnes qui n'ont pas été choisies par Thomas.

(d) Thomas accepte la preuve si tous les calculs qu'il peut vérifier à partir des informations qu'il reçoit sont corrects. (Pour plus de détail : il vérifie que (1) les valeurs révélées par Alice en (c) correspondent bien aux engagements en (a) ; (2) chaque porte logique calculée par Alice, Charlie, et Bob a été calculée correctement, lorsque Thomas connaît les entrées et sorties de la porte ; (3) la sortie finale du calcul est bien égale à $\vec{a}_f$, $\vec{b}_f$, $\vec{c}_f$, pour les deux participants virtuels non-choisis à l'étape (b) ; (4) $\vec{y} = \vec{a}_f + \vec{b}_f + \vec{c}_f$.)

1. Supposons que Thomas choisit Alice à l'étape (b). Montrer que Thomas peut vérifier tous les calculs effectués par un des autres participants virtuels (lequel ?).
2. Supposons temporairement que Sylvie ne connaît en fait pas de préimage de $\vec{y}$. Décrire une stratégie qui permet à Sylvie de faire quand même accepter la preuve à Thomas, avec probabilité de succès 1/3.
3. Réciproquement, argumenter que si Sylvie sait faire accepter une preuve avec probabilité 1 (relativement au choix de Thomas en (b)), alors elle connaît nécessairement une préimage.
4. Esquisser une preuve que le protocole précédent est *zero-knowledge*.
5. Supposons qu'on utilise la technique de Fiat-Shamir (cf. cours) pour transformer ce protocole de connaissance en un protocole de signature. Brièvement, que peut-on dire sur la résistance de cette technique de signature face aux ordinateurs quantiques ?

Exercice 3 (Zero-knowledge Foundations)**.** This exercise is about zero-knowledge proofs. In this exercise, we assume that it has the properties :

— correctness : honestly generated proofs verify
— soundness : it is hard to output a proof for $x \notin L$ that verifies
— zero-knowledge : there is a simulator that can simulate the view of the verifier for $x \in L$

1. Sketch a non-interactive zero-knowledge proof for some language $L \in \mathsf{BPP}$. That is, the prover sends a single message π.
2. Suppose that L has a non-interactive ZK proof. Show that $L \in \mathsf{BPP}$.
   Hint : use the simulator S to construct an adversary that decides L. Note that the view of the verifier is (π, r), where π is the proof and r is the verifiers randomness.
3. We have seen that with the Fiat-Shamir transform, we can make ZK proofs non-interactive in the random-oracle model. Also, we can construct NIZKs for the graph isomorphism problem (GIP). Does this mean that $\mathsf{GIP} \in \mathsf{BPP}$?
4. Suppose that L has a ZK proof in which the verifier is deterministic. Then $L \in \mathsf{BPP}$.

Exercice 4 (Commitments from OWFs)**.** An interactive bit-commitment is two stage protocol.

— Commit Stage : Alice has a bit b to which she wishes to commit to Bob. She and Bob exchange messages. At the end of the stage Bob has some information that represents b.

1. Noter que Sylvie ne s'engage pas sur les valeurs (r_c, c_i, c_j) *reçues* par Alice lors d'une porte multiplicative (question 3), mais seulement sur la sortie a'.

— Reveal Stage : At the end of this stage Bob knows b.

Further, Bob should not be able to guess b before the reveal stage with probability noticably higher than $1/2$ (hiding), and Alice should not be able to reveal two 0 and 1 after the commit stage (binding). Let us assume that we can construct a pseudorandom generator G_F that maps n bits to $3n$ bits, based on any one-way function F.

1. Show that given some one-way function F, we can construct a bit-commitment.
   Hint : Bob sends a random $r \leftarrow \{0,1\}^{3n}$ in the first round, then Alice evaluates $G_F(s)$ with a random seed s to construct a message that commits to b.
2. Let us assume some setup function (e.g., $g, h \leftarrow \mathbb{G}$ for Pedersen commitments). Show that non-interactive bit-commitments with setup imply one-way functions, i.e., the commit stage is a single message from Alice to Bob.
3. Modify the interactive bit-commitment from the first question to construct a secure non-interactive bit-commitment with setup function.
4. Can you construct a commitment to arbitrary messages (with a length upper bound by some n) based on OWFs ?