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

Introduction à la cryptologie
TD n° 3 : Signatures

Exercice 1 (Fonction de hachage et sécurité des signatures de Schnorr)**.** Soit $\mathbb{G}$ un groupe cyclique d'ordre q. Soit $\mathcal{H}$ une fonction de hachage. On rappelle le schéma de signature de Schnorr :

Génération des paramètres : Alice tire une clef secrète $x \in \mathbb{Z}_q^*$ uniformément aléatoirement. Elle publie la clef publique $y = g^x$.

Signature : Pour signer un message M, Alice tire $k \in \mathbb{Z}_q^*$ uniformément aléatoirement. La signature de M est $\sigma = (s, h)$ avec $h = \mathcal{H}(g^k, M)$ et $s = k - xh$.

Vérification : Étant donnés M et $\sigma = (s, h)$, on vérifie $\mathcal{H}(g^s y^h, M) = h$.

1. Montrer que ce schéma de signature est correct.
2. Montrer que si l'espace des messages à signer est $\mathbb{G}$ et qu'on choisit comme fonction de hachage :
$$\mathcal{H} : (M, r) \mapsto M \cdot r$$
alors le protocole de Schnorr n'est pas résistant aux contrefaçons existentielles.

Exercice 2 (Logarithme discret computationnel et décisionnel)**.** Soit p un nombre premier tel que $q = 2p + 1$ est premier également. Soit $\mathbb{G} = \mathbb{Z}_q^*$. On considère un élément h d'ordre p dans $\mathbb{G}$. Soit $\mathbb{H} = \langle h \rangle$.

1. On dit qu'un élément f de $\mathbb{G}$ est un carré ssi : $\exists g \in \mathbb{G}, f = g^2$. Donner un algorithme qui étant donné un élément $f \in \mathbb{G}$ renvoie 1 si f est un carré, 0 sinon.
Indication : on peut considérer la fonction sur $\mathbb{G}$ qui à x associe x^p.
2. Un élément h d'ordre p existe-t-il nécessairement ? Donner un algorithme pour trouver un tel g.
3. Supposons que nous avons un algorithme polynomial pour résoudre le logarithme discret sur $\mathbb{H}$. En déduire un algorithme polynomial pour résoudre le logarithme discret sur $\mathbb{G}$.
4. Soit A la distribution des triplets (g^a, g^b, g^{ab}) où a, b, sont tirés uniformément aléatoirement dans $\mathbb{Z}_q^*$. Soit B la distribution des triplets (g^a, g^b, g^c) où a, b, c sont tirés uniformément aléatoirement dans $\mathbb{Z}_q^*$. Le problème décisionnel de Diffie-Hellman sur $\mathbb{G}$ est de distinguer ces deux distributions. Donner un algorithme polynomial qui prend en entrée un triplet dans $\mathbb{G}$ et sort un bit, tel que que la différence entre la probabilité que ce bit soit 1 sur une entrée tirée dans A et une entrée tirée dans B est minorée par une constante non nulle.

Exercice 3 (Exercice philosophique)**.**

1. Rappeler le schéma d'échange de clef de Diffie-Hellman. Décrire une attaque par homme-au-milieu sur ce schéma.
2. Si deux entités souhaitent établir un canal sécurisé et pouvoir signer des messages l'une pour l'autre, sans aucune information partagée préalable, et en utilisant un canal non sécurisé, comment peuvent-elles faire ?
3. Lorsque vous vous connectez à un site de commerce en ligne sur votre navigateur, et que celui-ci vous indique que la connexion est sécurisée, à combien d'entités faites-vous confiance pour croire que la connexion est effectivement sécurisée ?

Exercice 4. Supposons que l'on a accès à m modules publiques N_i de signatures RSA sur n bits (où m est un grand nombre). À cause de générateurs d'aléa défaillant, on suppose que certaines de ces clefs publiques ont un facteur non-trivial commun. Pour simplifier, on suppose cependant que chaque clef partage au plus un de ses deux exposants avec d'autres clefs (et jamais les deux). On admet que la multiplication, division et PGCD entre deux entiers sur n bits coûtent $\tilde{O}(n)$ opérations de base (où la notation $\tilde{O}$ cache les facteurs polylogarithmiques, i.e. une fonction est $\tilde{O}(f)$ ssi elle est $O(f \cdot \log^k(f))$ pour un certain $k \in \mathbb{N}$).

1. Donner un algorithme qui, étant données deux clefs RSA distinctes avec un facteur commun, calcule la clef de signature de ces deux instances RSA en temps polynomial.
2. Donner un algorithme qui calcule $P = \prod N_i$ en temps $\tilde{O}(mn)$.
3. Donner un algorithme qui calcule $P \bmod N_i^2$ pour tout i en temps $\tilde{O}(mn)$.
4. En déduire un algorithme qui trouve tous les facteurs communs entre les m modules publiques en temps $\tilde{O}(mn)$. (On pourra remarquer que $\gcd(N_i, P/N_i) = \gcd(N_i, (P \bmod N_i^2)/N_i)$.