Examen ISEC / 11 mai 2023 / durée 2h

— Documents autorisés.
— Il y a deux exercices indépendants. Ce sujet est recto-verso.
— **Toute affirmation devra être justifiée** (répondre juste « oui » ou « non » vaut zéro).

1 KEM basé sur Elgamal

Un *Key Encapsulation Mechanism* (KEM) est un mécanisme cryptographique à clef publique constitué de quatre algorithmes :

Setup À partir d'un paramètre de sécurité λ, produit des *paramètres publics* pp. Ceux-ci définissent des ensembles de clefs publiques $\mathcal{PK}$, de clefs privées $\mathcal{SK}$ de clefs « encapsulées » $\mathcal{K}$ et de chiffrés $\mathcal{C}$. Les trois autres algorithmes décrits ci-dessous reçoivent pp en argument de manière implicite.

KeyGen Produit une paire de clefs $(\mathsf{pk}, \mathsf{sk}) \in \mathcal{PK} \times \mathcal{SK}$.

Encaps À partir d'une clef publique $\mathsf{pk} \in \mathcal{PK}$, renvoie une paire $(\mathsf{k}, \mathsf{c}) \in \mathcal{K} \times C$.

Decaps À partir d'une clef secrète $\mathsf{sk} \in \mathcal{SK}$ et d'un chiffré $\mathsf{c} \in \mathcal{C}$, renvoie $k \in \mathcal{K}$.

Moralement, l'opération **Encaps**(pk) génère un message aléatoire k, puis en effectue le chiffrement avec la clef publique pk pour obtenir un chiffré c — cela renvoie à la fois k et c.

Pour que le KEM soit correct, il faut que pour toute paire de clefs $\mathsf{pk}, \mathsf{sk} \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, et pour toute paire $\mathsf{k}, \mathsf{c} \leftarrow \mathsf{Encaps}(\mathsf{pk})$, on ait bien $\mathsf{Decaps}(\mathsf{sk}, \mathsf{c}) = \mathsf{k}$.

▷ **Question 1:** Expliquez comment on peut combiner un KEM et un algorithme de chiffrement symétrique pour réaliser du *chiffrement à clef publique.*

▷ **Question 2:** Proposez une manière de réaliser un KEM en utilisant la fonction RSA (indice : le **Setup** ne fait rien).

On s'intéresse maintenant à un KEM basé sur le chiffrement Elgamal. L'opération **setup** produit un entier g d'ordre q modulo p (comme pour le chiffrement Elgamal normal). L'opération **Encaps** fonctionne en tirant r uniformément au hasard modulo q, puis calcule $\mathsf{k} \leftarrow g^r$ et $\mathsf{c} \leftarrow h^r$ (h est la clef publique).

▷ **Question 3:** Expliquez comment fonctionnent **KeyGen** et **Decaps**.

▷ **Question 4:** Quels sont les ensembles $\mathcal{PK}, \mathcal{SK}, \mathcal{K}$ et $\mathcal{C}$?

▷ **Question 5:** Justifiez que le KEM proposé est correct.

▷ **Question 6:** Quel est l'intérêt de cette construction par rapport au chiffrement Elgamal normal ?

▷ **Question 7:** En 2023, quels doivent être les tailles minimales (en bits) de p et q ?

▷ **Question 8:** Expliquez le fonctionnement détaillé de **Setup**.

▷ **Question 9:** Pour un adversaire, à quels problèmes algorithmiques correspondent : 1) le fait de calculer sk à partir de pk (et pp bien sûr) ; 2) le fait de calculer k à partir de c et pk (et pp bien sûr), sachant que $\mathsf{k}, \mathsf{c} \leftarrow$ **Encaps**(pk).

La *sécurité sémantique* du KEM est définie par le « jeu » suivant, qui est paramétré par un bit b :

— Un *challenger* génère des paramètres publics $\mathsf{pp} \leftarrow$ **Setup**(λ) et calcule $\mathsf{pk}, \mathsf{sk} \leftarrow$ **KeyGen**(pp)
— L'adversaire reçoit pp et pk ; il peut ensuite adresser des *requêtes* au *challenger*. Pour chacune d'entre elles :
  — Le challenger calcule $\mathsf{k}_0, \mathsf{c} \leftarrow$ **Encaps**(pk), puis il tire uniformément au hasard k_1 dans $\mathcal{K}$.
  — L'adversaire reçoit $(\mathsf{k}_b, \mathsf{c})$.
— L'adversaire émet un bit $\hat{b}$ (moralement, il « gagne » si $b = \hat{b}$).

L'*avantage* de l'adversaire est défini par $\Delta = \left|\Pr(\hat{b} = 1 \mid b = 1) - \Pr(\hat{b} = 1 \mid b = 0)\right|$. On considère que le KEM est *sémantiquement sûr* si tout adversaire efficace (qui fonctionne en temps polynomial en λ) n'a qu'un avantage négligeable, c'est-à-dire asymptotiquement plus faible que l'inverse de n'importe quel polynôme en λ.

▷ **Question 10:** Expliquez en quelques phrases l'intérêt de cette notion de sécurité.

▷ **Question 11:** Montrez que si l'adversaire pouvait *prévoir* la valeur de k dans la prochaine invocation de **Encaps**, alors le KEM ne pourrait pas être sémantiquement sûr (ceci justifie l'idée que les k sont aléatoires).

▷ **Question 12:** Quel lien y a-t-il entre la sécurité sémantique du KEM basé sur Elgamal décrit ci-dessus et le problème Diffie-Hellman *décisionnel* (justifiez).

2 Fonction de hachage de Chaum, van Heijst et Pfitzmann

Cet exercice étudie une fonction de hachage « prouvablement sûre » (mais jamais utilisée en pratique).

Considérons un nombre premier q de n bits, choisi de telle sorte que $p = 2q + 1$ est premier lui aussi. On prend également deux générateurs de α et β du groupe $\mathbb{Z}_p^\times$. On considère la famille de fonctions (indexée par p, α, β) qui prend en entrée deux nombres x et y dans $\mathbb{Z}_q$ et qui renvoie :

$$H(x, y) = \alpha^x \beta^y \bmod p$$

▷ **Question 13:** Quelle sont les tailles (en bits) de l'entrée de H et de sa sortie ?

▷ **Question 14:** Quelle est (en fonction de n) la complexité asymptotique du de l'évaluation de H ?

▷ **Question 15:** Comment faire pour hacher des messages arbitrairement longs sans augmenter n ?

▷ **Question 16:** Expliquez de façon détaillée comment faire pour produire les « paramètres » (p, α, β).

Comme on a supposé que α était une racine primitive, alors on sait qu'il existe λ tel que $\beta = \alpha^\lambda \bmod p$.

▷ **Question 17:** Pour un adversaire à qui (p, α, β) sont imposés, déterminer λ est-il facile ?

▷ **Question 18:** Montrez que si on connaît la valeur de λ, alors on peut produire des *collisions* sur H efficacement (rappel : une collision, ce sont deux entrées différentes qui produisent la même sortie).

L'objectif des questions suivantes est de montrer la réciproque : si on arrive à trouver *une seule* collision, alors on est capable de calculer λ efficacement.

▷ **Question 19:** Expliquez pourquoi ceci garantit de manière « prouvée » la *résistance aux collisions* de la fonction.

▷ **Question 20:** Supposons qu'on ait une collision sur H, c'est-à-dire deux paires $(x, y) \neq (u, v)$ telles que $H(x, y) = H(u, v)$. Justifiez qu'on a alors $\lambda(v - y) - (x - u) \equiv 0 \bmod p - 1$.

▷ **Question 21:** On pose $d \leftarrow \text{PGCD}(v - y, p - 1)$. Montrez que d est strictement inférieur à q (on peut supposer sans perte de généralité que $v - y > 0$).

▷ **Question 22:** En utilisant le lemme de Gauss, justifier que d vaut soit 1, soit 2 (Rappel. Lemme de Gauss : si a divise bc et que a est premier avec b, alors a divise c).

▷ **Question 23:** Dans le cas où $d = 1$, justifiez qu'il n'y a qu'une seule valeur de λ possible, et expliquez comment on peut la calculer facilement.

On traite maintenant le cas moins agréable où $d = 2$.

▷ **Question 24:** Montrez que $\lambda = 0$ et $\lambda = 1$ sont des solutions de $\lambda(v - y) - (x - u) = 0 \bmod 2$.

▷ **Question 25:** Concluez-en (à l'aide du théorème des restes chinois) qu'il y a deux valeurs de λ qui sont solutions modulo $p - 1$. Comment fait-on pour calculer celle qui est telle que $\beta = \alpha^\lambda$?