Université Paris 8 J. Lavauzelle

Master 1 Mathématiques et applications — année 2022–23

Cryptographie à clé publique – Feuille de TD 5

24/02/2023

Le corrigé de certains exercices sera disponible à l'adresse suivante :

`www.math.univ-paris13.fr/∼lavauzelle/teaching/2022-23/clef-publique.html`

$(\star)$ exercice fondamental $(\star\star)$ pour s'entraîner $(\star\star\star)$ pour aller plus loin 💻 sur machine

Exercice 1. $(\star)$ Signature RSA : falsification sélective à message choisi.

On s'intéresse au schéma de signature RSA brut. Dans le cours, nous en avons vu une falsification existentielle à clef seule. Le but de cet exercice est de monter une falsification sélective à message choisi. Une falsification **sélective** signifie que l'attaquant fixe le message dont il veut falsifier la signature **avant** de monter son attaque (et donc, avant de demander au signataire d'autres signatures valides). C'est donc une attaque moins forte que la falsification universelle, mais plus forte que la falsification existentielle.

Dans l'exercice, on note $\mathtt{pk} = (n, e)$ et $\mathtt{sk} = d$ les clefs publique et privée du schéma de signature RSA brut.

Question 1.– Soient $m_1, m_2 \in (\mathbb{Z}/n\mathbb{Z})^\times$ deux messages, et s_1, s_2 leurs signatures correspondantes. Que vaut la signature s du message $m = m_1 m_2 \mod n$, en fonction de s_1 et s_2 ?

Question 2.– En déduire la falsification de la signature d'un message $m \in (\mathbb{Z}/n\mathbb{Z})^\times$ quelconque, après avoir demandé à Alice la signature de deux messages m_1 et m_2 (différents de m) judicieusement choisis.

Exercice 2. $(\star)$ Signatures et fonctions de hachage.

Soit H une fonction de hachage à valeurs dans $\{0,1\}^t$. On rappelle qu'une *collision* sur H est un couple de messages distincts $\boldsymbol{m} \neq \boldsymbol{m}'$ tels que $H(\boldsymbol{m}) = H(\boldsymbol{m}')$.

Question 1.– On peut obtenir une collision sur la fonction de hachage H par un compromis temps-mémoire, et exploiter ainsi le *paradoxe des anniversaires*. Décrire la méthode qui permet d'obtenir cette collision, et donner une approximation de sa complexité en fonction de t. Pour cela, on supposera que le coût d'évaluation de H, et le test d'appartenance d'un haché h à une liste de hachés L se font en temps constant.

On considère maintenant le schéma de signature DSA dans le groupe multiplicatif $\mathbb{F}_p^\times$, et on note g un générateur d'un sous-groupe d'ordre q de $\mathbb{F}_p^\times$, où q divise $(p-1)$. Pour simplifier, on suppose également que la fonction de hachage H est utilisée **sans schéma de remplissage** additionnel. Les algorithmes de signature et de vérification de DSA sont rappelés ci-dessous. On note $\mathcal{S} = (\mathbb{Z}/q\mathbb{Z})^\times \times (\mathbb{Z}/q\mathbb{Z})^\times$ l'espace des signatures.

Algorithme 1 : Signature DSA

Entrée : un message $\boldsymbol{m}$, la clé privée a
Sortie : une signature $s \in \mathcal{S}$

1 Calculer l'entier h associé à $H(\boldsymbol{m}) \in \{0,1\}^t$.
2 Choisir $k \in (\mathbb{Z}/q\mathbb{Z})^\times$ aléatoirement.
3 Calculer $b = (g^k \mod p) \mod q$.
4 Calculer $c = (h + ab)k^{-1} \mod q$.
5 Si b ou c n'est pas inversible $\mod q$, revenir à l'étape 2.
6 Sinon, retourner $s = (b,c)$.

Algorithme 2 : Vérification DSA

Entrée : une signature $s \in \mathcal{S}$, un message $\boldsymbol{m}$, la clé publique $\alpha = g^a$
Sortie : vrai ou faux

1 Calculer l'entier h associé à $H(\boldsymbol{m}) \in \{0,1\}^t$.
2 Calculer $x = g^{hc^{-1} \bmod q}\, \alpha^{bc^{-1} \bmod q}$.
3 Faire le test $x \equiv b \mod q$ et retourner le booléen associé.

Question 2.– Expliquer comment une collision sur H peut mener à une attaque sur le schéma de signature. On précisera la nature et les moyens de l'attaque.

Question 3.– En déduire la valeur de t minimale pour espérer obtenir une sécurité EUF-CMA (infalsifiabilité existentielle à message choisi) de 128 bits.

Exercice 3. $(\star\star)$ Signature de Lamport.

Dans cet exercice, on s'intéresse au schéma de signature de Lamport, qui ne présuppose que l'utilisation d'une fonction à sens unique.

Soient E et E' deux ensembles finis et $f : E \to E'$ une fonction à sens unique. Le schéma de signature est défini pour un certain paramètre entier $k \geq 1$.

Algorithme 3 : Génération des clefs

Entrée : k
Sortie : une paire de clés publique/privée

1 Tirer uniformément $2k$ éléments distincts de E, et les stocker dans une matrice de taille $(2 \times k)$:

$$\boldsymbol{A} = \begin{pmatrix} a_{0,1} & a_{0,1} & \dots & \dots & a_{0,k} \\ a_{1,1} & a_{1,2} & \dots & \dots & a_{1,k} \end{pmatrix} \in E^{2\times k}$$

2 Calculer la matrice $\boldsymbol{B}$ de taille $(2 \times k)$ sur E', constituée des $b_{i,j} = f(a_{i,j})$:

$$\boldsymbol{B} = \begin{pmatrix} f(a_{0,1}) & f(a_{0,1} & \dots & \dots & f(a_{0,k}) \\ f(a_{1,1}) & f(a_{1,2}) & \dots & \dots & f(a_{1,k}) \end{pmatrix} \in (E')^{2\times k}$$

3 La clé publique est $\boldsymbol{B}$, la clé privée est $\boldsymbol{A}$.

Algorithme 4 : Signature

Entrée : la clé privée $\boldsymbol{A}$, le message à signer $\boldsymbol{m} \in \{0,1\}^k$
Sortie : la signature $\boldsymbol{s}$

1 Pour tout $i \in \{1,\dots,k\}$, définir $s_i := a_{m_i,i}$.
2 Retourner la signature $\boldsymbol{s} = (s_1,\dots,s_k) \in E^k$

Question 1.– Expliquer pourquoi, si la fonction f n'est pas à sens unique, alors la signature n'est pas sûre.

Algorithme 5 : Vérification

Entrée : la clé publique $\boldsymbol{B}$, la signature $\boldsymbol{s}$, le message à signer $\boldsymbol{m} \in \{0,1\}^k$
Sortie : un booléen `true/false`

1 Vérifier que $f(s_i) = B_{m_i,i}$ pour tout $i \in \{1,\dots,k\}$.

Question 2.– Expliquer pourquoi la même paire de clés ne peut pas être utilisée pour signer deux messages.

Question 3.– A priori, le schéma semble construit de sorte que la longueur des messages, et le nombre de colonnes de la clé publique et de la clé privée sont égaux.

1. En quoi cela pose-t-il problème d'un point de vue pratique ?
2. Proposer une modification de la signature afin qu'elle puisse rester pratique pour n'importe quel message à signer.
3. En déduire (approximativement) la taille des clés de cette signature. Justifier.

Exercice 4. $(\star\star)$ Vérification simultanée de signatures RSA.

Soit $(n = pq, e)$ une clé publique RSA, et d la clé privée associée. On s'intéresse au schéma de signature RSA « brut ».

On suppose que n est de taille t bits.

Question 1.– En fonction de t, quel est le coût algorithmique (en nombre de multiplications et carrés dans $\mathbb{Z}/n\mathbb{Z}$) d'une signature RSA ?

Bob reçoit une série de $\ell \geq 2$ messages signés par Alice : $(m_1, s_1), \dots, (m_\ell, s_\ell)$.

Pour vérifier ces signatures RSA plus rapidement, Bob décide de multiplier tous les messages entre eux : il calcule ainsi

$$m = m_1 m_2 \cdots m_\ell \mod n \quad \text{et} \quad s = s_1 s_2 \dots s_\ell \mod n\,.$$

Puis, il décide d'accepter la série de messages signés par Alice si et seulement si $s^e = m \mod n$.

Question 2.– Démontrer que si tous les messages ont bien été signés par Alice, alors Bob a raison d'accepter la série de signatures d'Alice.

Question 3.– Quantifier le gain de calcul de Bob en utilisant cette méthode.

Question 4.– Charlie sait que Bob utilise cette méthode pour vérifier les signatures d'Alice. Charlie intercepte une série $(m_1, s_1), \dots, (m_\ell, s_\ell)$ de messages signés par Alice (Charlie n'a donc pas choisi les messages). Comment peut-il intégrer un autre message m' à la série pour faire croire à Bob qu'Alice a également signé m' ?

Exercice 5. $(\star)$ Signature ElGamal : réutilisation de l'aléa.

On s'intéresse au schéma de signature ElGamal, dans lequel le message est haché avant d'être signé. Plus précisément, si H est une fonction de hachage à valeurs dans le groupe $\mathbb{F}_p^\times$, voici l'algorithme de signature :

On suppose qu'**Alice réutilise le même aléa k pour toutes ses signatures**.

Question 1.– Soient $s = (b,c)$ et $s' = (b',c')$ les signatures de deux messages distincts m et m' (avec le même k). Comparer b et b', puis déterminer une égalité liant $h = H(\boldsymbol{m})$, $h' = H(\boldsymbol{m'})$, a, b, c et c'.

Algorithme 6 : Algorithme de signature d'ElGamal avec fonction de hachage

Entrée : un message $\boldsymbol{m} \in \{0,1\}^*$, la clé privée $a \in \{1,\dots,p-2\}$
Sortie : une signature $s \in \mathbb{F}_p^\times \times \{0,\dots,p-2\}$

1 Choisir $k \in \mathbb{Z}/(p-1)\mathbb{Z}$ inversible.
2 Calculer $b = g^k \mod p$.
3 Calculer $h = H(\boldsymbol{m})$.
4 Calculer $c = (h - ab)k^{-1} \mod (p-1)$.
5 Retourner $s = (b,c)$.

Question 2.– En déduire une attaque à message connu sur la clé privée d'Alice, qui réussit avec très bonne probabilité.

Exercice 6. 💻 $(\star\star)$ Calcul rapide de $g^b b^c$ dans la vérification d'ElGamal.

Dans l'algorithme de vérification de la signature ElGamal, on doit calculer la valeur $\alpha^b b^c$, où α est un élément de $\mathbb{F}_p^\times$, et $(b,c) \in \{1, p-2\}$ peuvent être considérés comme aléatoires.

L'algorithme *square-and-multiply* permet de calculer une exponentiation dans un groupe cyclique d'ordre ℓ, en $\log_2(\ell)/2$ multiplications et $\log_2(\ell)$ carrés en moyenne (l'exposant est supposé aléatoire). Si b et c sont aléatoires, le calcul de $g^b b^c$ requiert donc, en moyenne, approximativement $\log_2(p)$ multiplications et $2\log_2(p)$ carrés.

Le but de cet exercice est de calculer $\alpha^b b^c$ sensiblement plus rapidement grâce à l'Algorithme 7.

Algorithme 7 : Algorithme de calcul rapide de $\alpha^b b^c$.

Entrée : $\alpha \in \mathbb{F}_p^\times$, $b = \sum_{i=0}^{\ell-1} b_i 2^i$ et $c = \sum_{i=0}^{\ell-1} c_i 2^i \in \{1,\dots,p-2\}$
Sortie : $\alpha^b b^c$

1 Calculer $z \leftarrow \alpha b$.
2 Initialiser $x \leftarrow 1$.
3 **Pour tout** *i allant de* $\ell - 1$ *à* 0 **faire**
4 Calculer $x \leftarrow x^2$
5 **Si** $(b_i,c_i) = (1,1)$
6 Calculer $x \leftarrow xz$
7 **Si** $(b_i,c_i) = (1,0)$
8 Calculer $x \leftarrow x\alpha$
9 **Si** $(b_i,c_i) = (0,1)$
10 Calculer $x \leftarrow xb$
11 Retourner x.

Question 1.– Démontrer que pour tout $i = \ell - 1, \dots, 0$, à la fin de la boucle **Pour (...)** de l'Algorithme 7, on a

$$x = \alpha^{\sum_{j=i}^{\ell-1} b_j 2^{j-i}} b^{\sum_{j=i}^{\ell-1} c_j 2^{j-i}}.$$

En déduire que l'algorithme est correct.

Question 2.– Compter le nombre moyen de carrés et le nombre moyen de multiplications effectués par l'Algorithme 7, lorsque b et c sont des entiers de ℓ bits tirés aléatoirement.

Question 3.– Implanter l'Algorithme 7 et vérifier l'amélioration pratique qu'il procure, comparé aux calculs successifs de α^b et b^c par la méthode *square-and-multiply*.