COURS ID12 « INTRODUCTION À LA CRYPTOLOGIE »
COURS 6
CRYPTOGRAPHIE ASYMÉTRIQUE : SIGNATURE ET RSA EN DÉTAIL

PAUL ZIMMERMANN

2. Signature à clé publique

Référence : chapitre 11 de [1].

Signature digitale : nombre dépendant d'un secret connu seulement du signataire, et dépendant éventuellement aussi du message signé.

Propriétés : vérifiable (signataire niant avoir signé, ou fausse signature) par un tiers sans accéder à la clé secrète du signataire.

Applications : authentification, intégrité des données, non-répudiation. Application importante : certification de clé publique par un tiers de confiance (lien entre clé publique et personne physique).

Historique : problème identifié bien avant que des solutions soient connues. Première solution : signature par RSA, qui reste aujourd'hui la plus pratique.

Classification : deux classes d'algorithmes (à appendice ou à restitution de message). Dans chaque classe, algorithmes déterministes ou randomisés.

2.1. **Signature à appendice.** Les plus utilisées (DSA, ElGamal, Schnorr). On commence par hacher le message à signer, puis on signe la valeur de hachage :

$$m \in \mathcal{M} \longrightarrow h = H(m) \in \mathcal{M}_h \longrightarrow s = S_{A,k}(h),$$

et on transmet m en clair, et s.

Vérification : on vérifie que $V_A(h, s) = \text{true}$.

2.1.1. *Exemple : signature par ElGamal.* Initialisation : p nombre premier, α générateur de $\mathbb{Z}_p^*$, et $1 \leq a < p-1$. On calcule $y = \alpha^a \bmod p$. Clé publique (p, α, y), clé privée a.

Signature : on choisit au hasard $1 \leq k < p-1$ avec $\gcd(k, p-1) = 1$, puis on calcule $r = \alpha^k \bmod p$ et $k^{-1} \bmod (p-1)$, $s = k^{-1}(h(m) - ar) \bmod (p-1)$. La signature est la paire (r, s).

Vérification : vérifier que $1 \leq r < p$, calculer $v_1 = y^r r^s \bmod p$, $v_2 = \alpha^{h(m)} \bmod p$, et vérifier que $v_1 = v_2$.

Attaque : il faut trouver r et s. Si on choisit k au hasard, alors $r = \alpha^k \bmod p$, et il faut avoir $s = k^{-1}(h(m) - ar) \bmod (p-1)$ avec a inconnu. Si le problème du logarithme discret est difficile, on ne peut pas faire mieux que choisir s au hasard, avec probabilité $\frac{1}{p}$ de succès.

Attention, il faut choisir k différent pour chaque signature. Sinon, supposons $s_1 = k^{-1}(h(m_1) - ar) \bmod (p-1)$, et $s_2 = k^{-1}(h(m_2) - ar) \bmod (p-1)$, alors $(s_1 - s_2)k = h(m_1) - h(m_2) \bmod (p-1)$, donc si $s_1 - s_2 \neq 0 \bmod (p-1)$, alors $k = (s_1 - s_2)^{-1}(h(m_1) - h(m_2) \bmod (p-1)$. Une fois connu k, on retrouve facilement a à partir de $s = k^{-1}(h(m) - ar) \bmod (p-1)$.

Si $h(m) = m$ (identité), alors $s = k^{-1}(m - ar) \bmod (p-1)$. On peut alors fabriquer une attaque comme suit. Soit (u, v) avec $\gcd(v, p-1) = 1$, calculer $r = \alpha^u y^v \bmod p = \alpha^{u+av} \bmod p$ et $s = -rv^{-1} \bmod (p-1)$. La paire (r, s) est une signature valide pour le message $m = su \bmod (p-1)$.

Date: zimmerma@loria.fr.

2.2. **Signature à restitution de message.** Exemples : RSA, Rabin, Nyberg-Rueppel. Ici, on applique la fonction de signature à tout le message (doit être de taille fixée). La vérification se fait grâce à une fonction de redondance R.

$$m \in \mathcal{M} \longrightarrow m' = R(m) \longrightarrow s = S_{A,k}(m'),$$

et on transmet simplement s.

Vérification : on calcule $m' = V_A(s)$, on vérifie que $m' \in \mathcal{M}_R$, puis $m = R^{-1}(m')$.

Exemple de fonction de redondance : $R(m) = mm$ (on duplique m). Probabilité qu'un message aléatoire soit redondant : 2^{-n}.

Remarque : la fonction R doit être telle que R^{-1} est facile à calculer.

2.2.1. *Exemple : Signature par RSA.* Initialisation : On choisit $n = pq$ avec p et q premiers, $\phi = (p-1)(q-1)$, $1 < e < \phi$ avec $\gcd(e, \phi) = 1$, on en déduit par Euclide étendu $1 < d < \phi$ tel que $ed = 1 \bmod \phi$. Clé publique (n, e), clé privée d.

Signature : $m' = R(m)$, $0 \leq m' < n$, $s = (m')^d \bmod n$.

Vérification : $m' = s^e \bmod n$, vérifier que $m' \in \mathcal{M}_R$, puis $m = R^{-1}(m')$.

Attaques sur la signature par RSA. (i) factorisation de n (permet de retrouver d, donc d'imiter n'importe quelle signature de A), (ii) sur la propriété multiplicative de RSA : si $s_1 = m_1^d \bmod n$, et $s_2 = m_2^d \bmod n$, alors $s_1 s_2 = (m_1 m_2)^d \bmod n$, donc $s_1 s_2$ est une signature valide pour $m_1 m_2$ (cf ci-dessous).

2.2.2. *Exemple d'attaque sur la fonction de redondance.* La fonction R ne peut pas être choisie indépendamment de S. Voici un exemple d'attaque. Soit n un module RSA de k bits, d la clé privée correspondante, et $t < k/2$. Supposons que les messages sont dans l'intervalle $[1, n2^{-t} - 1]$, et que $R(m) = m2^t$. On applique l'algorithme d'Euclide étendu à n et $R(m) = m2^t$. Au cours de l'algorithme, on obtient x, y, r tels que $xn + yR(m) = r$. On peut montrer qu'à un moment on aura $|y| < n2^{-t}$ et $r < n2^{-t}$ comme $2^t \leq \sqrt{n}$. Soit alors $m_2 = r2^t$ et $m_3 = y2^t$ si $y > 0$. Supposons qu'on obtienne les signatures $s_2 = m_2^d \bmod n$ et $s_3 = m_3^d \bmod n$. Alors :

$$\frac{s_2}{s_3} = \frac{m_2^d}{m_3^d} = (r/y)^d = R(m)^d \bmod n.$$

3. RSA en détail

On considère un système de chiffrement RSA avec $n = p \cdot q$, une clé publique e, et une clé secrète d. On rappelle que e vérifie $0 < e < (p-1)(q-1)$, e est premier avec $(p-1)(q-1)$, et $d \equiv 1/e \bmod (p-1)(q-1)$.

3.1. **Équivalence entre attaque sur la clé et factorisation.** La connaissance de la factorisation de n permet de retrouver facilement la clé secrète. En effet, comme on a $d \equiv 1/e \bmod (p-1)(q-1)$, on a $d \equiv 1/e \bmod (p-1)$, donc on peut trouver $d \bmod (p-1)$ via un pgcd étendu entre e et $p-1$, qui sont par hypothèse premiers entre eux. On trouve de même $d \bmod (q-1)$, et par restes chinois $d \bmod (p-1)(q-1)$. Note : $p-1$ et $q-1$ ne sont pas forcément premiers entre eux, en toute rigueur les restes chinois donnent $d < \operatorname{lcm}(p-1, q-1)$.

Inversement, la connaissance de l'exposant privé d permet-il de trouver la factorisation de n ? La réponse est oui. Comme $ed \equiv 1 \bmod (p-1)(q-1)$, on a $ed - 1 = k(p-1)(q-1)$. Donc pour tout entier a, $0 < a < n$, $a^{ed-1} \equiv 1$ modulo p, et idem modulo q, donc $a^{ed-1} \equiv 1 \bmod n$. Soit $ed-1 = 2^s t$, avec t impair. On a donc $a^{2^s t} \equiv 1 \bmod n$. On peut montrer que $a^{2^{s-1}t} \neq \pm 1 \bmod n$ pour au moins la moitié des valeurs de a. Pour un tel a, le pgcd de $a^{2^{s-1}t} - 1$ et n est un facteur non trivial de n. Il suffit donc pour trouver p et q de choisir des a aléatoires, de calculer $b = a^{2^{s-1}t} \bmod n$, puis $\gcd(b-1, n)$; le nombre moyen d'essais est deux pour obtenir un facteur non trivial.

Ceci prouve l'équivalence, modulo des calculs polynomiaux, entre l'obtention de l'exposant privé et la factorisation du module RSA.

3.2. **Module commun.** Pour communiquer dans un groupe de personnes, on pourrait envisager l'utilisationn d'un module RSA n commun, avec des paires de clés distinctes (d_i, e_i). Ceci est non sûr, car on a vu que la connaissance de d permet de trouver la factorisation de n. À partir de là, n'importe quel membre du groupe peut donc calculer la clé privée d_i des autres membres.

3.3. **Attaque à petit exposant de chiffrement.** Supposons qu'Alice envoie un même message m à trois destinataires, qui ont des modules RSA différents n_1, n_2, n_3 (cf ci-dessus), mais un même exposant public $e = 3$. On a donc $c_1 \equiv m^3 \bmod n_1$, $c_2 =\equiv m^3 \bmod n_2$, $c_3 =\equiv m^3 \bmod n_3$. Il est raisonnable de penser que n_1, n_2, n_3 sont premiers entre eux. Soit $M = m^3$; comme $m < n_i$, on a $M < n_1 n_2 n_3$, et $M \equiv c_i \bmod n_i$. Via restes chinois, on peut donc reconstruire M. Une simple racine cubique (exacte) permet de retrouver m.

Une façon de se prémunir contre cette attaque est de légèrement modifier m pour chaque destinataire. Note : le même problème se pose pour un « petit » message, i.e. lorsque $m^e < n$, puisqu'une racine e-ième (entière) permet de retrouver m.

3.4. **Multiplicativité de RSA.** Pour m_1, m_2 deux messages clairs, et c_1, c_2 les chiffrés correspondants, on a $(m_1 m_2)^e \equiv m_1^e m_2^e \equiv c_1 c_2 \bmod n$. En d'autres termes, le chiffré correspondant à $m_1 m_2$ est $c_1 c_2$.

Charlie peut utiliser cette multiplicativité de RSA pour monter une attaque à couple clair-chiffré choisi. Soit c le message à déchiffrer. Charlie choisit x aléatoire, et demande à Alice de déchiffrer $y \equiv c x^e \bmod n$. Alice lui renvoie $z \equiv y^d \bmod n$.

Or $z \equiv (cx^e)^d \equiv c^d x^{ed} \equiv mx \bmod n$. Donc $z/x \equiv m \bmod n$, où z/x se calcule par un pgcd étendu.

3.5. **Messages invariants.** Un message m est dit *invariant* si le chiffré correspondant est identique à m, c'est-à-dire $m^e \equiv m \bmod n$. Il est clair (sic) qu'il vaut mieux éviter ce type de message, ou au moins avoir une faible proportion de messages invariants. Blakley et Borosh (1979) ont montré qu'il y a toujours au moins 9 messages invariants.

En effet, modulo p, les solutions de $x^e \equiv x \bmod p$ comprennent au moins $x = 0$, $x = 1$ et $x = -1$, car e est forcément impair, sinon e ne serait pas premier avec $p - 1$. Comme il en est de même modulo q, on obtient par restes chinois au moins 9 solutions de $x^e \equiv x \bmod n$. Si l'on exclut les messages clairs trivialement invariants 0, 1 et -1, il y a donc au moins 6 messages invariants non triviaux.

Par exemple, avec $n = 731 = 17 \cdot 43$, on a 15 solutions pour $e = 5$, 9 pour $e = 11$, 35 pour $e = 13$, 51 pour $e = 17$, 63 pour $e = 25$, 75 pour $e = 29$, ...

3.6. **Choix de p et q.** Comment choisir p et q pour avoir un bon niveau de sécurité ? Il faut évidemment se prémunir contre les algorithmes de factorisation dont la complexité dépend essentiellement de la taille du plus petit facteur premier de n, donc si possible choisir p et q de même taille. Il ne faut cependant pas les choisir trop proches, car alors une attaque exhaustive est possible : on suppose $q = p + k$ avec k un petit entier, et comme on connaît $n = pq$, on est ramené à résoudre une équation du second degré pour chaque valeur de k.

On impose usuellement que p et q soient des nombres premiers *forts* (*strong primes* en anglais). Un nombre premier p est dit *fort* lorsque (a) $p - 1$ a un grand facteur premier r, (b) $p + 1$ a un grand facteur premier, et (c) $r - 1$ a un grand facteur premier. La condition (a) permet de se prémunir contre la factorisation de p par l'algorithme $P - 1$ de Pollard. La condition (b) permet de se prémunir contre la factorisation de p par l'algorithme $P + 1$, attribué à Williams. Enfin, la condition (c) permet de se prémunir contre les attaques cycliques.

Références

1. Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone, *Handbook of applied cryptography*, CRC Press, 1997, freely available at `http ://www.cacr.math.uwaterloo.ca/hac/`.