10. Le chiffrement asymétrique anonyme

10.1 Définitions

Un schéma de chiffrement à clé publique est défini par 3 algorithmes :
— $\mathcal{K}(1^k)$, qui génère un couple de clés $(\mathsf{pk}, \mathsf{sk})$, de chiffrement et de déchiffrement, en fonction du paramètre de sécurité k ;
— $\mathcal{E}(\mathsf{pk}; m)$, qui produit un chiffré c de m sous la clé pk (avec un ruban aléatoire) ;
— $\mathcal{D}(\mathsf{sk}; c)$, qui déchiffre c sous sk, et retourne donc, soit le clair m s'il existe, soit $\perp$ si le chiffré n'est pas correct.

La notion de sécurité pour le chiffrement est la *confidentialité* du clair, malgré la vue du chiffré, pour toute clé publique : aucun attaquant n'est en mesure d'avoir un avantage non-négligeable dans le jeu suivant, contre le challenger
— le challenger génère une liste de couples $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathcal{K}(1^k)$, et fournit les clés publiques pk_i à l'attaquant ;
— l'attaquant choisit une clé publique $\mathsf{pk} \in \{\mathsf{pk}_i\}$ et deux messages (m_0, m_1), qu'il envoie au challenger ;
— le challenger choisit un bit $b \leftarrow \{0, 1\}$, et chiffre m_b sous pk dans $c \leftarrow \mathcal{E}(\mathsf{pk}; m_b)$, qu'il envoie à l'attaquant ;
— l'attaquant retourne son avis b' au sujet de b.

On mesure l'avantage d'un attaquant $\mathcal{A}$, contre la *sécurité sémantique* définie par le jeu ci-dessus, par

$$\mathsf{Adv}(\mathcal{A}) = \Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0] = 2 \times \Pr[b' = b] - 1.$$

Q-1. Soit $\mathcal{B}$ un algorithme capable d'extraire le bit de poids faible de $m \in \{0, 1\}^n$ à partir de $c = \mathcal{E}(\mathsf{pk}; m)$ avec probabilité $1/2 + \varepsilon$:

$$\Pr[\mathcal{B}(\mathsf{pk}, c) = b | b \xleftarrow{R} \{0, 1\}, m \xleftarrow{R} \{0, 1\}^{n-1} \times \{b\}, c \leftarrow \mathcal{E}(\mathsf{pk}, m)] \geq \frac{1}{2} + \varepsilon.$$

Montrer comment on peut construire un attaquant $\mathcal{A}$ contre la sécurité sémantique, et exprimer l'avantage.

Masquer le contenu d'un message est important, mais parfois l'identité du destinataire est aussi une information sensible. Supposons donc qu'un certain nombre de clés publiques sont possibles, et on souhaite garantir l'*anonymat* du destinataire : aucune information sur la clé publique du destinataire ne doit fuir dans le chiffré (ce qui permettrait de lever un doute sur la clé publique utilisée, et ainsi connaître le destinataire).

Q-2. Définir le jeu d'un attaquant contre un challenger pour spécifier une telle notion d'anonymat.

10.2 Chiffrement ElGamal

On rappelle le chiffrement ElGamal, avec g un générateur de $\mathbb{G}$, d'ordre premier q :
— $\mathcal{K}(1^k)$: la clé privée est un scalaire $\mathsf{sk} = x \in \mathbb{Z}_q^\star$, et la clé publique est $\mathsf{pk} = y = g^x$;
— $\mathcal{E}(\mathsf{pk}, m)$: pour chiffrer un message $m \in \mathbb{G}$, on choisit $r \in \mathbb{Z}^\star$, puis on calcule

$$c_1 = g^r, \quad c_2 = y^r \cdot m.$$

— $\mathcal{D}(\mathsf{sk}, c)$: pour déchiffrer c, on calcule $m = c_2 / c_1^x$.

Q-3. Montrer que ce schéma garantit la sécurité sémantique (le jeu ci-dessus, sans aucun oracle de déchiffrement). On précisera sous quelle hypothèse algorithmique.

Q-4. Montrer qu'il garantit également l'anonymat du destinataire (votre notion de sécurité). On précisera sous quelle hypothèse algorithmique.

Q-5. En revanche, si l'adversaire a accès aux oracles de déchiffrement, montrer comment casser la sécurité sémantique.

Q-6. Montrer que de tels oracles permettent également de casser l'anonymat du destinataire.

10.3 Chiffrement RSA-OAEP

On rappelle le chiffrement RSA :

— $\mathcal{K}(1^k)$: la clé publique est un module RSA $n = pq$ sur k bits ($2^{k-1} < n < 2^k$) et l'exposant de chiffrement $\mathsf{pk} = (n, e)$, tandis que la clé privée est l'exposant de déchiffrement $\mathsf{sk} = d$;

— $\mathcal{E}(\mathsf{pk}, m)$: pour chiffrer un message $m \in \mathbb{Z}_n^\star$, on calcule

$$c = m^e \bmod n.$$

— $\mathcal{D}(\mathsf{sk}, c)$: pour déchiffrer c, on calcule $m = c^d \bmod n$.

Q-7. Montrer que ce schéma ne garantit pas la sécurité sémantique.

On applique donc le padding OAEP (sans redondance) pour le renforcer, où $k = k_1 + k_2 + 1$ (ainsi, $n > 2^{k_1+k_2}$), avec $G : \{0,1\}^{k_2} \to \{0,1\}^{k_1}$ et $H : \{0,1\}^{k_1} \to \{0,1\}^{k_2}$:

— $\mathcal{E}(\mathsf{pk}, m)$: pour chiffrer un message $m \in \{0,1\}^{k_1}$, on choisit un aléa $r \in \{0,1\}^{k_2}$, on calcule

$$x = \mathsf{OAEP}(m, r) = s\|t \text{ où } s = G(r) \oplus m, t = H(s) \oplus r, \quad c = \mathsf{RSA}(n, e, x) = x^e \bmod n.$$

— $\mathcal{D}(\mathsf{sk}, c)$: pour déchiffrer c, on calcule

$$x = c^d \bmod n = s\|t, \quad r = t \oplus H(s), \quad m = s \oplus G(r).$$

Q-8. Montrer que cette construction RSA-OAEP ne garantit toujours pas l'anonymat.

On considère les 6 applications suivantes de $\{0,1\}^k$ dans $\{0,1\}^k$:

$f^1_{n,e}(x)$
  `if` $x < n$ `then` $u \leftarrow x^e \bmod n$
  `else` $u \leftarrow x$
  `return` u

$f^2_{n,e}(u)$
  `if` $u < 2^k - n$ `then` $v \leftarrow u + n$
  `elseif` $2^k - n \le u < n$ `then` $v \leftarrow u$
  `else` $v \leftarrow u - n$
  `return` v

$f^3_{n,e}(v)$
  `if` $v < n$ `then` $y \leftarrow v^e \bmod n$
  `else` $y \leftarrow v$
  `return` y

$g^1_{n,d}(u)$
  `if` $u < n$ `then` $x \leftarrow u^d \bmod n$
  `else` $x \leftarrow u$
  `return` x

$g^2_{n,d}(v)$
  `if` $v < 2^k - n$ `then` $u \leftarrow v + n$
  `if` $2^k - n \le v < n$ `then` $u \leftarrow v$
  `else` $u \leftarrow v - n$
  `return` u

$g^3_{n,d}(y)$
  `if` $y < n$ `then` $v \leftarrow y^d \bmod n$
  `else` $v \leftarrow y$
  `return` v

Q-9. Montrer qu'il s'agit de 6 bijections, et que pour $i = 1, 2, 3$, $f^i_{n,e}$ et $g^i_{n,d}$ sont réciproques l'une de l'autre..

On remplace $\mathsf{RSA}(n, e, x) = x^e \bmod n$ pour $x \in \mathbb{Z}_n^\star$, par $\mathsf{RSACD}(n, e, x) = f_{n,e}^3(f_{n,e}^2(f_{n,e}^1(x)))$ pour $x \in \{0,1\}^k$. On va alors montrer que l'inversion de RSACD est aussi difficile que l'inversion de RSA.

Pour cela, on procède en 3 temps, en utilisant un algorithme $\mathcal{A}$ contre $\mathsf{RSACD}(n, e, x) = f_{n,e}^3(v)$, où $v = f_{n,e}^2(u)$ et $u = f_{n,e}^1(x)$, qui retrouve x avec probabilité ε.

Q-10. Montrer que si l'algorithme $\mathcal{A}$ retrouve x à partir de $y = \mathsf{RSACD}(n, e, x)$, avec probabilité ε_1 lorsque v est uniformément distribué dans $]0, n[$, alors on peut construire un algorithme $\mathcal{B}_1$ qui inverse RSA avec probabilité ε_1.

Q-11. Montrer que si l'algorithme $\mathcal{A}$ retrouve x à partir de $y = \mathsf{RSACD}(n, e, x)$, avec probabilité ε_2 lorsque v est uniformément distribué dans $]2^k - n, 2^k[$, alors on peut construire un algorithme $\mathcal{B}_2$ qui inverse RSA avec probabilité ε_2.

Q-12. En déduire un algorithme qui utilise $\mathcal{A}$ pour inverser RSA, et préciser la probabilité de succès.

10.4 Chiffrement RSACD-OAEP

Considérons désormais la construction suivante, avec $k = k_1 + k_2$:

— $\mathcal{K}(1^k)$: la clé publique est un module RSA $n = pq$ sur k bits ($2^{k-1} < n < 2^k$) et l'exposant de chiffrement $\mathsf{pk} = (n, e)$, tandis que la clé privée est l'exposant de déchiffrement $\mathsf{sk} = d$;

— $\mathcal{E}(\mathsf{pk}, m)$: pour chiffrer un message $m \in \{0,1\}^{k_1}$, on choisit un aléa $r \in \{0,1\}^{k_2}$, on calcule

$$x = \mathsf{OAEP}(m, r) \quad c = \mathsf{RSACD}(n, e, x).$$

— $\mathcal{D}(\mathsf{sk}, c)$: pour déchiffrer c, on calcule

$$x = \mathsf{RSACD}^{-1}(n, d, c) \quad (m, r) = \mathsf{OAEP}^{-1}(x).$$

Q-13. Discuter la propriété d'anonymat de ce schéma.