

Cryptographie à clé publique

Cours 3

Julien Lavauzelle

Université Paris 8

Master 1 mathématiques et applications – parcours ACC

10/02/2023

Vu à la **séance précédente** :

- Présentation d'un chiffrement à clé publique
- Types d'attaques : cassage total, cassage partiel, distinction de l'aléa (IND)
- Modes d'attaques : chiffré seul, clair connu, clair choisi (CPA), chiffré choisi (CCA)
- Présentation de RSA
- Quelques cryptanalyses de RSA : factorisation de n , attaque de Wiener, attaque de Coppersmith
- En TD : Håstad et autres problèmes de sécurité sémantique.

Questions ?

1. Sécurité sémantique pour RSA

2. D'autres systèmes à base de factorisation

Cryptosystème de Rabin

Cryptosystème de Goldwasser–Micali

Cryptosystème de Blum–Goldwasser

1. Sécurité sémantique pour RSA

2. D'autres systèmes à base de factorisation

Cryptosystème de Rabin

Cryptosystème de Goldwasser–Micali

Cryptosystème de Blum–Goldwasser

Rappel. Dans le cours précédent, on a vu que pour une paire de clés RSA ($\text{pk} = (n, e), \text{sk} = d$) :

- si le clair m est trop petit, alors $m^e < n$ donc un attaquant peut retrouver m
- si les messages sont liés, alors on peut éventuellement retrouver m ,
- ...

Ces problèmes sont liés au fait que le chiffrement est **déterministe** : un même message chiffré deux fois, donne deux fois le même chiffré.

On voudrait obtenir (au moins) une **sécurité sémantique**, autrement dit, que le chiffré soit **indistinguable** d'un élément aléatoire.

Définition : problème d'indistinguabilité du chiffré (IND).

Instance. Une fonction de chiffrement $f : \mathcal{X} \rightarrow \mathcal{Y}$, deux clairs x_1, x_2 et un chiffré $c = f(x_i)$ où $i \in \{1, 2\}$

Question. Déterminer i .

Définition : problème d'indistinguabilité du chiffré (IND).

Instance. Une fonction de chiffrement $f : \mathcal{X} \rightarrow \mathcal{Y}$, deux clairs x_1, x_2 et un chiffré $c = f(x_i)$ où $i \in \{1, 2\}$

Question. Déterminer i .

Pour rendre ce problème difficile, on **doit** introduire de l'aléa pour rendre la fonction de chiffrement non-déterministe.

Une solution générique : OAEP (*Optimal asymmetric encryption padding*).

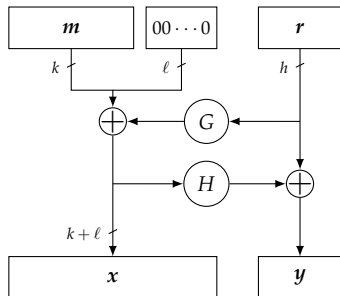


Optimal asymmetric encryption. M. Bellare and P. Rogaway. EUROCRYPT. 1994.

Idée : ajouter un aléa au message et le hacher, pour le rendre « presque uniforme » dans l'espace des clairs.

Soit m un message à chiffrer, de taille k bits. On suppose que l'on dispose :

1. d'un **générateur d'aléa**, produisant des séquences aléatoires r (appelées **nonces**) de longueur h
2. de deux **fonctions de hachage** cryptographiques, qui sont **publiques** :
 - la fonction G , produisant des hachés de taille $k + \ell$,
 - la fonction H , produisant des hachés de taille h .



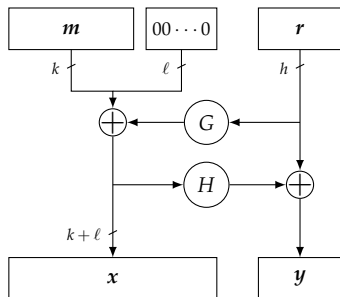
Le **nonce** r de longueur h est tiré aléatoirement.

On a ensuite :

- $x = G(r) \oplus (m \parallel 0)$,
- $y = r \oplus H(x) = r \oplus H(G(r) \oplus (m \parallel 0))$.

La **sortie d'OAEP** est la concaténation $z = x \parallel y$.

Ensuite, on fournit z en entrée de l'algorithme de chiffrement (de RSA, par exemple).



Le **nonce** r de longueur h est tiré aléatoirement.

On a ensuite :

- $x = G(r) + (m \parallel 0)$,
- $y = r + H(x) = r + H(G(r) + (m \parallel 0))$.

La sortie est $z = x \parallel y$.

Question. Comment retrouver m à partir de z ?

Solution.

1. on calcule $H(x)$
2. on retranche $H(x)$ à y pour retrouver r
3. on retranche $G(r)$ à x pour obtenir $(m \parallel 0)$, puis m .

Remarque. On appelle cette technique un **procédé de remplissage** (*padding*).

Remarque. OAEP s'inspire des réseaux de Feistel (voir cryptographie symétrique).


Propriété (énoncé informel). Pour h et ℓ suffisamment grands, si les fonctions G et H sont cryptographiquement sûres et si le générateur d'aléa a suffisamment d'entropie, la sortie z d'OAEP est indistinguable d'une séquence aléatoire.

Dans ce cas, combiné avec une fonction à sens unique (par exemple, la fonction RSA), le **schéma de chiffrement devient sémantiquement sûr (IND-CPA)**.

Remarque. On obtient également un mode de chiffrement *all-or-nothing* (« tout ou rien »), signifiant qu'en l'absence du chiffré entier, même Alice n'obtient pas d'information sur le clair.

Important. Il s'avère que OAEP fournit également une sécurité IND-CCA1 (attaque à chiffré choisi non-adaptative), mais il n'est pas encore prouvé qu'il fournit une sécurité IND-CCA2 (adaptative) pour n'importe quel schéma de chiffrement.

Pour **RSA**, c'est bien le cas. Voir par exemple :

 *Provable security for public key schemes*. D. Pointcheval. Advanced Courses on Contemporary Cryptology. 2005.

D'ailleurs, la seconde version du standard RSA PKCS#1 utilise OAEP.

1. Sécurité sémantique pour RSA

2. D'autres systèmes à base de factorisation

Cryptosystème de Rabin

Cryptosystème de Goldwasser–Micali

Cryptosystème de Blum–Goldwasser

1. Sécurité sémantique pour RSA

2. D'autres systèmes à base de factorisation

Cryptosystème de Rabin

Cryptosystème de Goldwasser–Micali

Cryptosystème de Blum–Goldwasser

Proposé en 1979 par Michael Rabin (rappel : RSA en 1977).

Historiquement, premier système admettant une **réduction** vers le problème de la factorisation.

RABIN : GÉNÉRATION DE CLÉS

1. Choisir aléatoirement deux grands nombres premiers p et q .
2. Calculer $n = pq$.
3. La **clé publique** est n , la **clé privée** est (p, q) .

RABIN : CHIFFREMENT

Pour **chiffrer** un message $m \in \{0, \dots, n-1\}$:

1. Le **chiffré** est $c = m^2 \pmod n$.

RABIN : DÉCHIFFREMENT

Pour **déchiffrer** c :

1. Calculer l'ensemble des racines carrées de c modulo n .
2. Retrouver m parmi ces racines carrées.

Problème (extraction de racine carrée modulaire) : étant donnés p, q tel que $pq = n$, et $y = x^2 \pmod n$, retrouver x

Pour calculer une **racine carrée modulo** $n = pq$, on utilise le théorème des restes chinois

$$\begin{array}{ccc} \mathbb{Z}/n\mathbb{Z} & \simeq & \mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/q\mathbb{Z} \\ y \pmod n & \longleftrightarrow & (y \pmod p, y \pmod q) \end{array}$$

Puis, on trouve les racines carrées de y dans \mathbb{F}_p (notons-les r_p et $-r_p$).

- Il existe des algorithmes efficaces pour cela (Tonelli–Shanks, Cipolla).
- Remarque : si $p \equiv 3 \pmod 4$, il suffit de calculer $y^{(p+1)/4}$.

Idem dans \mathbb{F}_q (les racines sont notées r_q et $-r_q$).

On calcule les coefficients de Bezout a_p et a_q associés à p et q :

$$a_p \cdot p + a_q \cdot q = 1.$$

Enfin, on reconstruit les quatre racines carrées :

$$\left\{ \begin{array}{l} x_1 = a_p \cdot p \cdot r_q + a_q \cdot q \cdot r_p \\ x_2 = n - x_1 \\ x_3 = a_p \cdot p \cdot r_q - a_q \cdot q \cdot r_p \\ x_4 = n - x_3 \end{array} \right.$$

Question : comment retrouver le bon message parmi les quatre racines carrées?

Solution 1. Ajouter une structure particulière au message.

Par exemple : $m \mapsto m \parallel 00 \dots 0$ (*padding*)

→ avec grande probabilité, les autres racines n'auront pas cette forme.

Solution 2. Blum & Williams ont proposé

- de choisir p, q congrus à 3 modulo 4 (on dit que p, q sont des nombres premiers de Blum),
- de prendre comme espace de messages m le groupe des résidus quadratiques non-nuls modulo n , que l'on note QR_n^\times .

Dans ce cas, la fonction

$$f : \begin{array}{ccc} \text{QR}_n^\times & \rightarrow & \text{QR}_n^\times \\ m & \mapsto & m^2 \pmod n \end{array}$$

est bijective.

Preuve : en exercice.

Théorème. Si l'on dispose d'un algorithme A , d'entrées (c, n) , qui retrouve le message m associé à un chiffré $c = m^2 \pmod n$ en un temps polynomial, alors on peut factoriser n en un temps polynomial.

Preuve du théorème. En exercice.

Conséquence. Attaquer le cryptosystème de Rabin est aussi difficile que factoriser n .

Même s'il admet une réduction de sécurité pour sa fonction à sens unique, le cryptosystème de Rabin, vu comme tel, **n'est pas sémantiquement sûr.**

→ lui appliquer OAEP

1. Sécurité sémantique pour RSA

2. D'autres systèmes à base de factorisation

Cryptosystème de Rabin

Cryptosystème de Goldwasser–Micali

Cryptosystème de Blum–Goldwasser

En 1982, Goldwasser et Micali proposent un système de chiffrement sémantiquement sûr. Le schéma est **probabiliste** : des chiffrements successifs d'un même clair sont potentiellement différents \implies facilite la preuve de sécurité IND-CPA.

GOLDWASSER–MICALI : GÉNÉRATION DE CLÉS

1. Choisir aléatoirement deux grands nombres premiers p et q .
2. Calculer $n = pq$.
3. Choisir $x \in \{1, \dots, n-1\}$ tel que $\left(\frac{x}{p}\right) = -1$ et $\left(\frac{x}{q}\right) = -1$ (x est un non-résidu modulo p et q).
4. La **clé publique** est (x, n) , la **clé privée** est (p, q) .

Exemple. Si $p \equiv 3 \pmod{4}$ et $q \equiv 3 \pmod{4}$, on peut choisir $x = n - 1$ (exercice : pourquoi?)

Remarque. Si on souhaite un x aléatoire, on procède de la manière suivante :

- calculer a et b deux non-résidus quadratiques modulo p et q ,
- avec les restes chinois, reconstruire $x \in \mathbb{Z}/n\mathbb{Z}$ tel que $x \equiv a \pmod{p}$ et $x \equiv b \pmod{q}$.

GOLDWASSER–MICALI : CHIFFREMENT

On souhaite chiffrer un message $m = (m_1, \dots, m_k) \in \{0, 1\}^k$ constitué de k bits.

1. Pour tout $i = 1, \dots, k$:
 - 1.1 choisir aléatoirement $y_i \in (\mathbb{Z}/n\mathbb{Z})^\times$
 - 1.2 définir $c_i = y_i^2 x^{m_i} \pmod n$
2. Le chiffré est $c = (c_1, \dots, c_k) \in (\mathbb{Z}/n\mathbb{Z})^k$

GOLDWASSER–MICALI : DÉCHIFFREMENT

On souhaite déchiffrer $c = (c_1, \dots, c_k) \in (\mathbb{Z}/n\mathbb{Z})^k$.

1. Pour tout $i = 1, \dots, k$:
 - 1.1 on calcule $a = c_i \pmod p$.
 - 1.2 si a est un carré modulo p , on affecte $m_i = 0$
 - 1.3 sinon, on affecte $m_i = 1$
2. Le message déchiffré est $m = (m_1, \dots, m_k)$.

Validité du déchiffrement.

- Si $m_i = 0$, alors $c_i = y_i^2$ est un carré modulo n , donc modulo p (et modulo q).
- Sinon, $c_i = y_i^2 x$ n'est pas un carré modulo p . En effet, $\left(\frac{c_i}{p}\right) = 1 \times \left(\frac{x}{p}\right) = -1$.

Fait. On peut **réduire la sécurité** du chiffrement de Goldwasser et Micali au problème de la résiduosit  quadratique.

Probl me de r siduosit  quadratique. Soit x tel que $\left(\frac{x}{n}\right) = 1$. D terminer si x est un r sidu quadratique modulo n .

On appelle **pseudo-carr ** un  l ment $x \in \mathbb{Z}/n\mathbb{Z}$ tel que $\left(\frac{x}{n}\right) = 1$ mais x n'est pas un carr  modulo n .

Dans le syst me de Goldwasser–Micali, chaque bit c_i du chiffr  est de la forme

$$c_i = y_i^2 x^{m_i} \pmod{n}.$$

Alors,

- si $m_i = 0$, alors c_i est un r sidu quadratique **al atoire** modulo n
- si $m_i = 1$, alors c_i est un pseudo-carr  **al atoire** modulo n

Ainsi, c_i est une instance al atoire du probl me de r siduosit  quadratique.

Probl me majeur de Goldwasser–Micali : son **facteur d'expansion**.

→ Pour un message de k bits, le chiffr  est de taille $k \log_2 n$.

1. Sécurité sémantique pour RSA

2. D'autres systèmes à base de factorisation

Cryptosystème de Rabin

Cryptosystème de Goldwasser–Micali

Cryptosystème de Blum–Goldwasser

En 1984, Blum et Goldwasser proposent le système de chiffrement suivant.

BLUM–GOLDWASSER : GÉNÉRATION DE CLÉS

1. Choisir aléatoirement deux grands nombres premiers p et q congrus à 3 modulo 4.
2. Calculer $n = pq$.
3. Calculer a et b les coefficients de Bezout de p et q , i.e.

$$ap + bq = 1$$

4. La **clé publique** est n , la **clé privée** est (a, b, p, q) .

On note $k = \lfloor \log_2 n \rfloor$ et $h = \lfloor \log_2 k \rfloor$.

Remarque. On donne ici une version « accélérée » du cryptosystème de Blum–Goldwasser. Il en existe une version lente, avec $h = 1$.

BLUM–GOLDWASSER : CHIFFREMENT

Pour chiffrer un message m :

1. Découper m en blocs (m_1, \dots, m_t) de h bits chacun.
2. Choisir aléatoirement $x_0 \in \mathbb{QR}_n^\times$
3. Pour tout $i = 1, \dots, t$:
 - calculer $x_i = x_{i-1}^2 \pmod n$,
 - extraire $p_i \in \{0, 1\}^h$, les h bits de poids faible de x_i ,
 - calculer $c_i = p_i \oplus m_i$.
4. Calculer $x_{t+1} = x_t^2 \pmod n$.
5. Retourner $(c_1, \dots, c_t, x_{t+1})$.

Remarque. L'idée est de réaliser un chiffrement de type « flot », où la suite chiffrante est générée par un générateur de Blum-Blum-Shub (BBS).

On rappelle que $k = \lfloor \log_2 n \rfloor$ et $h = \lfloor \log_2 k \rfloor$. La clé privée est (p, q, a, b) où $ap + bq = 1$.

BLUM–GOLDWASSER : DÉCHIFFREMENT

Pour déchiffrer un chiffré $(c_1, \dots, c_t, x_{t+1})$:

1. Calculer $b_p = \left(\frac{p+1}{4}\right)^{t+1} \bmod (p-1)$ et $b_q = \left(\frac{q+1}{4}\right)^{t+1} \bmod (q-1)$.
2. Calculer $u_p = x_{t+1}^{b_p} \bmod p$ et $u_q = x_{t+1}^{b_q} \bmod q$.
3. Calculer $x_0 = apu_p + bqv_q \bmod n$
4. Pour tout $i = 1, \dots, t$:
 - calculer $x_i = x_{i-1}^2 \bmod n$,
 - extraire $p_i \in \{0, 1\}^h$, les h bits de poids faible de x_i ,
 - calculer $m_i = p_i \oplus c_i$.
5. Retourner m .

Validité. Il suffit de montrer que x_0 est bien calculé :


$$\text{On a } x_{t+1}^{(p+1)/4} \equiv x_t^{(p+1)/2} \equiv x_t \bmod p.$$

$$\text{Donc } x_{t+1}^{b_p} \equiv x_{t+1}^{((p+1)/4)^{t+1}} \equiv x_0 \bmod p.$$

De même, $x_{t+1}^{b_q} \equiv x_0 \bmod q$, et les restes chinois donnent le résultat.

Exemple pour Blum–Goldwasser

Exemple tiré de

 *Handbook of Applied Cryptography*. A. Menezes, P. van Oorschot and S. Vanstone. CRC Press. 1996.

Paramètres : $p = 499, q = 547 \implies n = 272953$

On trouve $a = -57$ et $b = 52$ tels que $ap + bq = 1$.

On a $k = \lfloor \log_2 n \rfloor = 18$ et $h = \lfloor \log_2 k \rfloor = 4$. Soit le message à chiffrer

$$m = 1001|1100|0001|0000|1100$$

CHIFFREMENT, avec $pk = n$:

On choisit aléatoirement $x_0 = 159201 \bmod n$.

On calcule alors

i	$x_i = x_{i-1}^2 \bmod n$	p_i	$c_i = p_i \oplus m_i$
1	180539	1011	0010
2	193932	1100	0000
3	245613	1101	1100
4	130286	1110	1110
5	40632	1000	0100

et on obtient $x_6 = x_5^2 \bmod n = 139680$.

Le chiffré est donc

$$((0010|0000|1100|1110|0100), 139680)$$

DÉCHIFFREMENT, avec $sk = (a, b, p, q)$:

Pour déchiffrer, on calcule

$$b_p = ((p+1)/4)^6 \bmod (p-1) = 463$$

$$b_q = ((q+1)/4)^6 \bmod (q-1) = 337$$

$$u_p = x_6^{463} \bmod p = 20$$

$$u_q = x_6^{337} \bmod q = 24$$

$$x_0 = apu_p + bqvu_q \bmod n = 159201$$

puis on déchiffre comme dans le tableau.

La **sécurité** du cryptosystème se réduit à celle du générateur de pseudo-aléa BBS.

Proposition. Le schéma de chiffrement de Blum–Goldwasser est sûr sémantiquement (IND-CPA), mais vulnérable à des attaques à chiffré choisi.

Par rapport à Goldwasser–Micali, l'**expansion** du cryptosystème de Blum–Goldwasser est réduite : le chiffré contient « seulement » $\log(n)$ bits **supplémentaires** par rapport au clair.

Efficacité. D'un point de vue calculatoire, les temps de chiffrement et de déchiffrement sont comparables à RSA (à un petit facteur constant près).

Questions ?