

I. Contexte de la cryptographie asymétrique

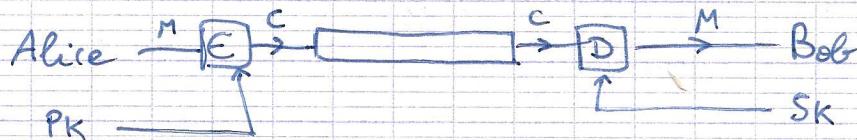
Problèmes de la crypto symétrique :

1) m utilisateurs $\rightarrow \frac{m(m-1)}{2}$ clés.

2) non répudiation / impossible en crypto symétrique
signature

\rightarrow invention de la crypto asymétrique

(Diffie, Hellman, 1976) "New directions in cryptography".



$$\begin{cases} C = E_{PK}(M) \\ M = D_{SK}(E_{PK}(M)) \end{cases} \quad \forall M, M = D_{SK}(E_{PK}(M))$$

$$\begin{cases} M = D_{SK}(C) \\ D_{SK} \circ E_{PK} = Id \end{cases} \quad D_{SK} \circ E_{PK} = Id$$

II. Définition de RSA

1977 : Rivest, Shamir et Adleman

(Merkle avait proposé un autre système quelques mois avant)

$$\begin{cases} SK = (p, q) \text{ où } p \text{ et } q \text{ sont des nombres entiers premiers et } p \neq q \\ PK = n = p \times q \end{cases}$$

Si un attaquant cherche à retrouver SK à partir de PK, il doit résoudre le problème de la factorisation à partir de $p \times q$, retrouver p et q .

exemple : 1) Diviser n successivement par 2, 3, 4, 5, 6...

~~$O(\sqrt{n})$~~

$p \times q = n$ si $p > \sqrt{n}$ et $q > \sqrt{n}$, on aurait $\underbrace{pq}_{n} > n$

2) Diviser par $2, 3, 5, 7, 11, 13, 17 \dots$ (nb premiers)
 \rightarrow complexité $O\left(\frac{\sqrt{m}}{\ln m}\right)$

3) Meilleure méthode connue
GNFS (General Number Field Sieve) 1994

↪ cible algébrique général

$$O\left(\exp\left(c(\ln m)^{1/3}(\ln \ln m)^{2/3}\right)\right)$$

$c \approx 1.92$

$$\textcircled{1} \quad O(\sqrt{m}) = O\left(\exp\left(\frac{1}{2} \ln(m)\right)\right)$$

$$\textcircled{2} \quad O\left(\frac{\sqrt{m}}{\ln m}\right) = O\left(\exp\left(\frac{1}{2} \ln(m - \ln \ln m)\right)\right)$$

$$\textcircled{3} \quad O\left(\exp\left(c(\ln m)^{1/3}(\ln \ln m)^{2/3}\right)\right) \geq 2^{80}$$

$m \text{ fait au moins } 1024 \text{ bits} \quad m \geq 2^{1024}$



Théorème: $f: \begin{cases} \mathbb{Z}/m\mathbb{Z} & \longrightarrow \mathbb{Z}/m\mathbb{Z} \\ x \longmapsto y = x^e \pmod{m} \end{cases}$ est une bijection si on suppose $\text{pgcd}(e, \varphi(m)) = 1$
(où $\varphi(m) = (p-1)(q-1)$)

et $f^{-1}: \begin{cases} \mathbb{Z}/m\mathbb{Z} & \longrightarrow \mathbb{Z}/m\mathbb{Z} \\ y \longmapsto x = y^d \pmod{m} \end{cases}$ avec $d = e^{-1} \pmod{\varphi(m)}$

Rappels: Théorème de Bézout : $\text{PGCD}(a, b) = 1$
 $\Leftrightarrow \exists u, v \text{ tq } au + bv = 1$

pair car

$(q-1) \text{ et } (p-1)$ pairs

Théorème de Fermat : Soit p premier et a entier $a \neq 0 \pmod p$
alors $a^{p-1} \equiv 1 \pmod p$

Format Démonstration : $\phi \left\{ (\mathbb{Z}/p\mathbb{Z})^* \rightarrow (\mathbb{Z}/p\mathbb{Z})^* \right.$ est une bijection

$$\left. \begin{array}{l} z \mapsto az \\ \end{array} \right.$$

Montrons que ϕ est injective, c'est à dire : $\phi(z) = \phi(z') \Rightarrow z = z'$

$$\phi(z) = \phi(z') \Leftrightarrow az = az' \pmod{p}$$

comme $a \neq 0 \pmod{p}$, a admet un inverse b tq $ab = 1 \pmod{p}$.

$$\Rightarrow abz = abz' \pmod{p} \Rightarrow z = z' \pmod{p}.$$

Comme l'ensemble de départ $(\mathbb{Z}/p\mathbb{Z})^*$ et l'ensemble d'arrivée $(\mathbb{Z}/p\mathbb{Z})^*$ ont le même cardinal on en déduit que ϕ est une bijection.

$$\begin{aligned} (p-1)! &= 1 \times 2 \times 3 \times \dots \times (p-1) = \phi(1) \times \phi(2) \times \phi(3) \times \dots \times \phi(p-1) \pmod{p} \\ &= (a \times 1) \times (a \times 2) \times \dots \times (a \times (p-1)) \pmod{p} \\ (p-1)! &= a^{p-1} \times 1 \times 2 \times 3 \times \dots \times (p-1) \pmod{p} \\ \text{car } 1, 2, 3, \dots &\quad \Rightarrow a^{p-1} = 1 \pmod{p}. \\ \text{inversibles} \end{aligned}$$

Démonstration RSA :

$$f: x \mapsto y = x^e \pmod{n} \text{ bijection ?}$$

On va montrer que $\forall x, (x^e)^d = x \pmod{n}$.

$$\begin{aligned} 1) \text{ Montrons que } \forall x, (x^e)^d &= x \pmod{p} \\ (x^e)^d &= x^{ed} = x^{1+\lambda \varphi(n)} = x \times x^{\lambda(p-1)(q-1)} = x \times \underbrace{(x^{p-1})^{\lambda(q-1)}}_{=1 \pmod{p}} \\ &\quad \text{si } x \neq 0 \pmod{p} \end{aligned}$$

$$2) \text{ De même } \forall x, (x^e)^d = x \pmod{q}$$

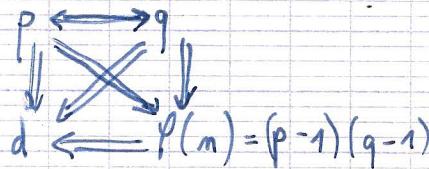
$$3) \text{ on en déduit } \left. \begin{array}{l} (x^e)^d = x \pmod{p} \\ (x^e)^d = x \pmod{q} \end{array} \right\} \Rightarrow (x^e)^d = x$$

divisible par $p \times q = n$

$(x^e)^d = x \pmod{p}$
 vrai même si
 $x = 0 \pmod{p}$

RSA : clé secrète $p, q, d, \varphi(n)$
 clé publique $n (= p \times q), e$

$$d = e^{-1} \bmod (p-1)(q-1)$$



si on connaît $\varphi(n)$ on peut retrouver p et q (cf feuille TD)
 si on connaît d , on peut retrouver p et q (admis)

III. Sécurité de RSA.

Objectifs : 1) Trouver la clé secrète \rightarrow factorisation \rightarrow meilleur algo
 $O(\exp(\alpha (\ln n)^{1/3} (\ln \ln n)^{2/3}))$

2) Retrouver x à partir de $y (= x^e \bmod n) \rightarrow$ extraire une racine e ième modulo n .

$$y = x^d \bmod n$$

La seule méthode connue : faire RSA

a) factoriser n pour trouver p et q et $\varphi(n)$

b) calculer $d (= e^{-1} \bmod \varphi(n))$ puisque e est public

$$x = y^d \bmod n$$

Remarque : Le problème de la factorisation deviendrait facile sur un ordinateur quantique.

IV. Implémentation de RSA.

On doit pouvoir calculer "en temps raisonnable"

e doit vérifier $\text{pgcd}(e, \varphi(n)) = 1$ typiquement pair

$$\begin{cases} e = 3 \\ e = 17 \\ e = 257 \end{cases}$$

demande $O(\exp(\alpha (\ln n)^{1/3} (\ln \ln n)^{2/3})) \geq 2^{80}$

$\Leftrightarrow n$ fait au moins 1024 bits.

CMINF115

Suite chapitre 4.

02/01/19

$$\begin{aligned}y &= x^e \bmod m \\x &= y^d \bmod m \\d &= e^{-1} \bmod \varphi(m)\end{aligned}$$

$m \rightarrow 1024$ bits
 $p, q \rightarrow 512$ bits
 $e \rightarrow$ quelques bits
 $d \rightarrow 1024$ bits.
 $\varphi(m) \rightarrow 1024$ bits

1^{ere} idée : on calcule d'abord y^d , puis on réduit modulo m .

$$\begin{aligned}d &\approx 2^{1024} \\y^d &\approx y^{(2^{1024})} = \underbrace{\left(\left(\left(\left(y^2 \right)^2 \right)^2 \cdots \right)^2}_{\text{taille}} = 1024 \times 2 \times 2 \times \cdots \times 2 = 1024 \times 2^{1024} \text{ bits!}\end{aligned}$$

2^e idée : Algorithme "Square and Multiply"

Calculer $x = y^d \bmod m$

$$d = d_{k-1} \cdot 2^{k-1} + d_{k-2} \cdot 2^{k-2} + \cdots + d_2 \cdot 2^2 + d_1 \cdot 2^1 + d_0$$

k bits

$$x = 1$$

Pour i de ~~2048 bits~~ à 0
 $x = x^2 \bmod m$ 1024 bits
 si $(d_i = 1)$ alors $x = \overbrace{x \times y}^{1024 \text{ bits}} \bmod m$.

exemple : $d = 11 = \{1011\}_2$

$$d = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$$

$$k=4 \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad d_3 \quad d_2 \quad d_1 \quad d_0$$

i	x
/	1
3	1
2	y^2
1	y^4
0	$y^{10} \times y = y^{11}$

$$y^{10} \times y = y^{11}$$

II. Utilisation de RSA

1) Confidentialité \rightarrow chiffrement RSA.

M message clair

C message chiffré

$$C = M^e \bmod n$$

$$M = C^d \bmod n$$

Problèmes :

$$1) \text{ Si } M < \sqrt[3]{n} \quad (e=3) \quad \text{ alors } C = M^3 \bmod n$$

$\approx \frac{1024}{3}$ bits

$$C = M^3 \Rightarrow M = \sqrt[3]{C}$$

2) Si l'ensemble des messages clairs possibles est "petit".

On sait que $M \in \mathcal{M}$ ex: $\mathcal{M} = \{\text{oui}, \text{non}\}$

Système déterministe

Le problème est que $M \mapsto C = M^e \bmod n$ est déterministe.

3) Si on chiffre le même message pour plusieurs destinataires Δ

ex: $e=3$ Alice

$$\begin{aligned} c_1 &\rightarrow \text{Bob } (m_1, e_3) = P_{\text{Bob}} \rightarrow SK_{\text{Bob}}(p_1, q_1, d_1) \\ c_2 &\rightarrow \text{Bertrand } (m_2, e_3) = P_{\text{Bertrand}} \rightarrow SK_{\text{Bertrand}}(p_2, q_2, d_2) \\ c_3 &\rightarrow \text{Bill } (m_3, e_3) = P_{\text{Bill}} \rightarrow SK_{\text{Bill}}(p_3, q_3, d_3) \end{aligned}$$

$$C_1 = M^3 \pmod{m_1}$$

$$C_2 = M^3 \pmod{m_2}$$

$$C_3 = M^3 \pmod{m_3}$$

Théorème chinois des restes :

Si $\text{PGCD}(n, n') = 1$

$\Psi : \mathbb{Z}/nn'\mathbb{Z} \rightarrow \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n'\mathbb{Z}$ est une bijection.

$$z \mapsto (\underbrace{z \pmod{n}}_{=a}, \underbrace{z \pmod{n'}}_{=b})$$

De plus, $\boxed{\Psi^{-1}(a, b) = umb + nm'a}$

$$\frac{umb}{1-nm'} + \frac{nm'a}{1-nm'} = b + (a-b)nm' = b \pmod{n'}$$

$$= a + (b-a)nm \equiv a \pmod{n}$$

$$\Rightarrow \Psi(umb + nm'a) = (a, b)$$

$$\left. \begin{cases} C_1 = M^3 \pmod{m_1} \\ C_2 = M^3 \pmod{m_2} \\ C_3 = M^3 \pmod{m_3} \end{cases} \right\} \xrightarrow{\text{Charlie obtient } M^3 \pmod{(m_1 \cdot m_2)}} \text{à condition que } \text{PGCD}(m_1, m_2) = 1$$

si $\text{PGCD}(m_1, m_2) \neq 1$ Charlie peut factoriser m_1 et m_2

$\xrightarrow{\quad} \left\{ M^3 \pmod{(m_1 \cdot m_2 \cdot m_3)} \right\}$ à condition que $\text{PGCD}(m_1 \cdot m_2, m_3) = 1$

avec $0 \leq M^3 < m_1 \cdot m_2 \cdot m_3$

\rightarrow Charlie connaît M^3

Exemple de méthode de chiffrement :

- PKCS (Public Key Cryptographic Standards)
(RSA Data Security)

PKCS #1 v 1.5 $C = (\mu(m))^e \bmod n$

$$\mu(m) = \text{padding} \parallel \text{m} \parallel \text{padding} \parallel M$$

147 octets max
 $n = 1024$ bits
= 128 octets.

valeur aléatoire d'au moins 8 octets qui sont tous non nuls.

Problème 1 : si $M < \sqrt{n}$ $\mu(M)$ n'aura pas ce défaut

Problème 2 : il n'y a plus de déterminisme (à cause de M)

Problème 3 : l'alea est différent pour chaque destinataire

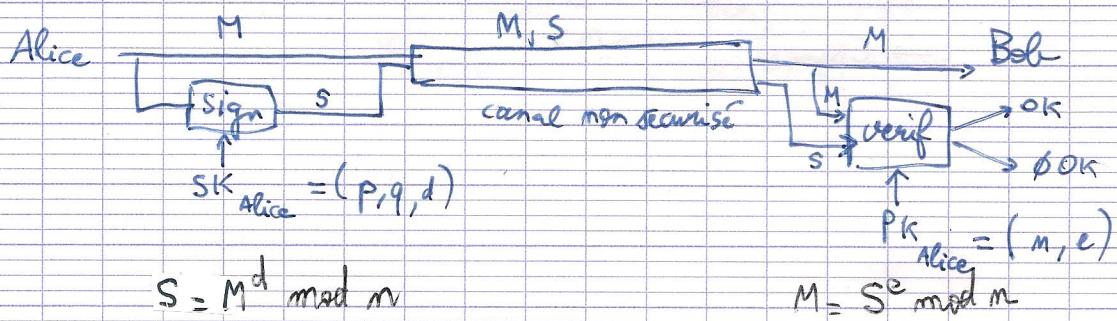
Déchiffrement (dans le cas PKCS #1 v 1.5)

Alice $c = (\mu(m))^e \bmod n \rightarrow$ Bob $\rightarrow c^d \bmod n = \text{padding} \parallel m \parallel \text{padding}$
 ↓
 ≥ 8 octets.

- OAEP

2) Intégrité et authenticité.

→ Signature électronique



Problèmes qui apparaissent si on pose : $s = M^d \bmod n$

Problème 1: On doit imposer $0 \leq M < n$

(si on ne met aucune condition par exemple M et $M+n$ auraient la même signature)

Remarque : Si M est "grand"

on peut le diviser en blocs $M_1 || M_2 || \dots || M_k$ avec $\forall i : 0 \leq M_i < n$

Alice $\xrightarrow{M, s_1, \dots, s_k}$ Bob puis calculer $\forall i : s_i = M_i^d \bmod n$.
(Charlie peut échanger des blocs M_i et s_i)

Problème 2: Supposons qu'Alice envoie 2 messages à Bob

Alice $\xrightarrow{M_1, s_1 (= M_1^d \bmod n)}$ Bob Sign : $S = M^d \bmod n$
 $\xrightarrow{M_2, s_2 (= M_2^d \bmod n)}$ Bob Verif : $S^e = M \bmod n$

Posons $M_3 = M_1 \times M_2 \bmod n$

M_3 a pour signature $S_3 = M_3^d \bmod n = (M_1 \times M_2)^d \bmod n = M_1^d \times M_2^d \bmod n$
 $\Rightarrow S_3 = s_1 \times s_2 \bmod n$.

Méthode pour signer :

PKCS #1 v 1.5

$$S = (\mu(M))^d \bmod n$$

$$\mu(M) = \phi\phi\phi1FF\dotsFF\phi\phi||c_n||h(M)$$

1024 bits

h : fonction de hachage.

Identifiant pour connaître le nom de la fonction de hachage utilisée.

Vérification: $S^e \bmod n = \phi\phi\phi1FF\dotsFF\phi\phi||c_n||h(M)$

Problème 1: $\forall M$, $h(M)$ a une taille fixe "petite".

Problème 2: La multiplicité disparaît car
en général $h(M_1 \times M_2) \neq h(M_1) \times h(M_2)$