Sécurité, fiabilité, sûreté des systèmes informatisés

1. Rappels mathématiques

Théorème : $\forall a \in Z$ et $\forall n \in N$ il existe $(q,r) \in Z^2$ tels que a = qn + r avec $0 \le r < n$.
q est le quotient de la division
r = a mod n est le reste de la division
donc a / n $\Leftrightarrow$ a mod n = 0

Définition : si a mod n = b mod n on écrira $a \equiv b(\mathrm{mod}\, n)$ et on dira que a est congru modulo n autrement dit $a \equiv b(\mathrm{mod}\, n)$ si a et b produisent le même reste quand on les divise par n.

Définition : soit $\mathbf{Z}_m = \{0,1,2,\ldots,m-1\}$ un ensemble muni de deux opérations + et * ; celles-ci satisfont les règles suivantes :

- *l'addition est interne :* $(a,b) \in Z_m^2 \Rightarrow a+b \in Z_m$
- *l'addition est commutative :* $(a,b) \in Z_m^2 \Rightarrow a+b = b+a$
- *l'addition est associative :* $(a,b,c) \in Z_m^3 \Rightarrow (a+b)+c = a+(b+c)$
- *0 est neutre pour l'addition :* $a \in Z_m \Rightarrow a+0 = 0+a = a$
- *l'opposé de tout a de Zm est m – a :* $a \in Z_m \Rightarrow a+(m-a) = (m-a)+a = 0$
- *la multiplication est interne :* $(a,b) \in Z_m^2 \Rightarrow a*b \in Z_m$
- *la multiplication est commutative :* $(a,b) \in Z_m^2 \Rightarrow a*b = b*a$
- *la multiplication est associative :* $(a,b,c) \in Z_m^3 \Rightarrow (a*b)*c = a*(b*c)$
- *1 est neutre pour la multiplication :* $a \in Z_m \Rightarrow a*1 = 1*a = a$
- *la multiplication est distributive sur l'addition :* $(a,b,c) \in Z_m^3 \Rightarrow (a+b)*c = a*c+b*c$

Définition : on définit a-b dans $\mathbf{Z}_m$ comme étant (a + m – b) mod m

Exemple : 11 – 18 dans $\mathbf{Z}_{31}$
11 + 31 – 18 = 24
24 mod 31 = 24
or 11 – 18 = -7 d'où -7 mod 31 = 24

Définition : soit $a \in Z_m$, l'inverse de a est un élément noté $a^{-1} \in Z_m \left(\neq \frac{1}{a} \notin Z_m \right)$ tel que $\boxed{a * a^{-1} \equiv 1(\mathrm{mod}\, m)}$.

Exemple : calculer 3^{-1} dans $\mathbf{Z}_{26}$
on cherche $x \in Z_{26}$ tel que $x*3 \equiv 1(\mathrm{mod}\, 26)$; on sait que $a \equiv b(\mathrm{mod}\, m) \Rightarrow m/(a-b)$
d'où $26/(x*3-1) \Rightarrow x = 9$ donc $3^{-1} = 9$

Théorème : a de $\mathbf{Z}_m$ admet un inverse modulo m si et seulement si pgcd(a,m) = 1, c'est-à-dire a et m sont premiers entre eux.

Propriétés :

- $b = a^{-1} \Rightarrow a = b^{-1}$
- $m \text{ premier} \Rightarrow \forall a \in Z_m \text{ existe et est unique}$

Définition : un nombre premier est un entier naturel différent de 1 qui n'est divisible que par 1 et lui-même.

Théorème : il existe un nombre infini de nombres premiers.
Preuve : supposons qu'il existe un nombre fini de nombres premiers $p_1,p_2,\ldots,p_n$
posons $m = p_1 * p_2 * \ldots * p_n + 1$ alors $m > 1$ et m n'est pas premier car $m > p_i \ \forall i = 1...n$
d'où $\exists q \in N$ et $j \in \{1,...,n\}$ tel que $m = q * p_j$

$$q = \frac{m}{p_j} = \frac{p_1 * p_2 \ldots p_j * p_{j+1} \ldots p_n + 1}{p_j} = \frac{p_1 * p_2 \ldots p_j * p_{j+1} \ldots p_n}{p_j} + \frac{1}{p_j} = p_1 * p_2 \ldots p_{j-1} * p_{j+1} \ldots p_n + \frac{1}{p_j}$$

or $q \in N$ et $\frac{1}{p_j} \notin N$ donc contradiction → il existe un nombre infini de nombres premiers.

Théorème : tout entier n > 1 est soit un nombre premier soit le produit de nombres premiers.

2. Indicateur d'Euler

Définition : l'indicateur d'Euler noté φ(p) est le nombre d'éléments inversibles dans $\mathbf{Z}_p$, c'est-à-dire le nombre d'éléments inférieurs à p et premiers avec p.

Exemple : φ(3) = Card{1,2} = 2 ; φ(5) = Card {1,2,3,4} = 4

Remarque : si p est premier alors φ(p) = p – 1.

Propriétés :

- *si p est premier alors* $\varphi(p^k) = p^k * \left(1 - \frac{1}{p}\right) = p^{k-1} * (p-1)$
- *pour tout p on a* $\varphi(p_2) = p * \varphi(p)$
- *si* $p_1,\ldots,p_n$ *sont des facteurs premiers de p alors* $\varphi(p) = p * \left(1 - \frac{1}{p_1}\right)..\left(1 - \frac{1}{p_n}\right)$
- *si p et q sont premiers entre eux alors* $\varphi(p*q) = \varphi(p) * \varphi(q)$
- *si* $d_1,\ldots,d_n$ *sont les diviseurs de p alors* $\varphi(d_1) + \ldots + \varphi(d_n) = p$

Théorème d'Euler : si a est inversible dans $\mathbf{Z}_p$ (pgcd(a,p) = 1) alors $a^{\varphi(p)} \equiv 1 (\text{mod } p)$
Preuve : soit A l'ensemble des éléments inversibles dans $\mathbf{Z}_p$; $A = \{a_1,\ldots,a_{\varphi(p)}\}$

A est bien sûr un ensemble fini. Il faut remarquer que A est stable pour la multiplication c'est-à-dire $(a,b) \in A^2 \Rightarrow (a * b \bmod p) \in A$.
soit f : $\mathbf{Z}_p \rightarrow \mathbf{Z}_p$
x $\rightarrow$ a*x avec $a \in A$ c'est-à-dire a est inversible ; f(x) = a * x
a est inversible donc la bijectivité de f est évidente d'où f(A) = A
considérons alors le produit M = $f(a_1) * f(a_2) \ldots f(a_{\varphi(p)})$ où φ(p) est l'indicateur d'Euler
on a donc M = $a * a_1 * a * a_2 \ldots a * a_{\varphi(p)} = a^{\varphi(p)} * (a_1 * a_2 \ldots a_{\varphi(p)})$
or par bijectivité de f on sait que chaque élément de f(A) possède un seul antécédent dans A

Remarque :
- *si p est premier alors φ(p) = p – 1 et dans ce cas* $a^{p-1} \equiv 1 (\text{mod } p)$
- *si p et q sont premiers alors φ(pq) = φ(p) * φ(q) = (p-1) * (q-1) et dans ce cas* $a^{\varphi(pq)} \equiv 1 (\text{mod } pa)$ *et* $a^{(p-1)(q-1)} \equiv 1 (\text{mod } pq)$

Théorème de Fermat : si p est premier alors $\forall a \in N$ $a^p \equiv a (\text{mod } p)$
Preuve :
si pgcd(a,p) = 1 d'après la remarque on a $a^{p-1} \equiv 1 (\text{mod } p) \Rightarrow a^{p-1} * a \equiv a (\text{mod } p) \Rightarrow a^p \equiv a (\text{mod } p)$
si pgcd(a,p) ≠ 1 dans ce cas p / a car p est premier et donc a mod p = 0 ; $a^p \bmod p = 0 \Rightarrow a^p \equiv a (\text{mod } p)$

Théorème de Bezout : soit $(a,b) \in Z^2$ et d = pgcd(a,b) alors d est le plus petit entier positif vérifiant : $\boxed{ax + by = d \ \forall (x,y) \in Z^2}$
Preuve : soit s le plus petit entier positif vérifiant s = ax + by ; il faut montrer que s = d ; en effet soit q le quotient de la division de a par s on a alors a = qs + r donc r = a – qs = a mod s
il vient a mod s = a – q (ax + by) = a (1 – qx) + b (-qy) = aX + bY
donc a mod s est aussi une combinaison linéaire de a et b puisque a mod s < s
or s est le plus petit entier vérifiant ax + by = s ; il s'ensuit que a mod s = 0 donc s / a.
par un résonnement identique on trouve s / b ; comme s / a et s / b alors s est diviseur commun de a et b. Ainsi pgcd(a,b) = d ≥ s

Rappel : si l / a et l / b alors l / ax + by $\forall (x,y) \in Z^2$ (1)
d'après (1) d = pgcd(a,b) / s = ax + by donc pgcd(a,b) ≤ s
finalement d = pgcd(a,b) ≥ et d ≤ s donc $\boxed{d = s}$ (2) et donc d = ax + by $\forall (x,y) \in Z^2$

Remarque :
- si pgcd(a,b) = 1 dans ce cas $\exists (x,y) \in Z^2$ tels que ax + by = 1
- si l / a et l / b alors l / pgcd(a,b)

Théorème de Gauss : soit $(a,b,c) \in Z^3$, si pgcd(a,b) = 1 alors a / bc $\Rightarrow$ a / c
Preuve : pgcd(a,b) = 1 alors $\exists (x,y) \in Z^2$ tels que ax + by = 1 $\Rightarrow$ axc + byc = c (3)
or a / bc $\Rightarrow$ a / bcy $\Rightarrow \exists \alpha \in Z$ tel que bcy = αa ; on a aussi a / ac $\Rightarrow$ a / acx $\Rightarrow \exists \beta \in Z$ tel que acx = βa (4)
de (3) et (4) on tire αa + βa = c $\Rightarrow$ c = (α + β)a $\Rightarrow$ a / c

3. Algorithme d'Euclide

Il permet de calculer rapidement le pgcd de deux nombres r_0 et r_1 avec $r_0 > r_1$; il se fonde sur la propriété suivante.

Théorème : $\forall n \in Z$ on a pgcd(a,b) = pgcd(a,b + an)

L'algorithme d'Euclide consiste à effectuer la suite de divisions suivantes :
$r_0 = q_1 * r_1 + r_2 \qquad 0 \leq r_2 < r_1$
$r_1 = q_2 * r_2 + r_3 \qquad 0 \leq r_3 < r_2$
…
$r_{m-2} = q_{m-1} * r_{m-1} + r_m$
$r_{m-1} = q_m * r_m + 0$

$pgcd(r_0,r_1) = pgcd(r_1,r_0 - q_1r_1) = pgcd(r_1,r_2) = pgcd(r_2,r_1 - q_2r_2) = pgcd(r_2,r_3) = \ldots = pgcd(r_{m-1},r_m) = r_m$
$\Rightarrow$ *le pgcd de deux nombres r_0 et r_1 en suivant l'algorithme d'Euclide est le dernier reste non nul de la division.*

Exploitation de l'algorithme :
Soit $(a,b) \in Z_*^2$ et d = pgcd(a,b) ; posons $a = r_0$ et $b = r_1$, on a les relations suivantes :
1° ligne : $r_0 = q_1r_1 + r_2$ c'est-à-dire $a = q_1b + r_2 \Rightarrow r_2 = a - q_1b$ donc r_2 s'écrit comme $r_2 = ax_2 + by_2$ avec $x_2 = 1$ et $y_2 = -q_1$
2° ligne : $r_3 = r_1 - q_2r_2 = b - q_2(a - q_1b) = -q_2a + b(1 + q_1q_2) = ax_3 + by_3$ avec $x_3 = -q_2$ et $y_3 = 1 + q_1q_2$
…
pour chaque étape on aura $ax_k + by_k = r_k$ (1) et $r_{k-2} - q_{k-1} * r_{k-1} = r_k$ (2)
en remplaçant dans (2) les r_k par leurs valeurs d'après (1) on obtient :
$ax_k + by_k = ax_{k-2} + by_{k-2} - q_{k-1}(ax_{k-1} + by_{k-1})$
il vient

$\boxed{x_k = x_{k-2} - q_{k-1} * x_{k-1} \text{ et } y_k = y_{k-2} - q_{k-1} * y_{k-1}}$

donc on peut trouver x_k et y_k de la suite à partir de x_0, x_1, y_0, y_1
dans la pratique on pose $x_0 = 0$, $x_1 = 1$, $y_0 = 1$, $y_1 = 0$.

Application de l'algorithme :
a) trouver x et y dans la relation de Bezout
b) trouver l'inverse d'un élément a
soit $(a,b) \in Z^2$ avec pgcd(a,b) = 1 donc d'après Bezout $\exists (x,y)$ tels que ax + by = 1
si nous posons $r_0 = a$ et $r_1 = b$ dans l'algorithme on aura $ax_m + by_m = 1 = r_m$
dans $\mathbf{Z}_b$ cette relation devient $ax_m = 1 \bmod b$ car $by_m \bmod b = 0$
$\Rightarrow$ x_m est l'inverse de a dans $\mathbf{Z}_b$.

Cryptographie

Définition : un système cryptographique est un quintuplet (P,C,K,E,D) satisfaisant :

- *P est un ensemble fini de blocs de textes clairs*
- *C est un ensemble fini de blocs de textes chiffrés*
- *K est un ensemble fini de clefs*
- $\forall k \in K$, *il y a une règle de chiffrement* e_k *de E et une règle de chiffrement correspondante* d_k *de D ; on a* $e_k : P \rightarrow C$ *et* $d_k : C \rightarrow P$ *telles que* $d_k(e_k(x)) = x$ $\forall x \in P$

Principe : Alice et Bob choisissent secrètement une clé k qui définit les règles de e_k et d_k. Supposons que Alice souhaite communiquer un message à Bob.
Ce message étant une chaîne $x = x_1\ x_2 \ldots x_n$ $\forall x_i \in P$, chaque xi est chiffré en utilisant la règle e_k. Ainsi Alice calcule $y_i = e_k(x_i)$ $\forall i \in [1, n]$; la chaîne chiffrée obtenue $y = y_1\ y_2 \ldots y_n$ est envoyée à Bob qui déchiffre le message en utilisant $d_k(y_i) = x_i$.

1. Chiffrement RSA (Rivest, Shamir, Adleman – 1978)

Principe : soit n = p*q où p et q sont chacun premiers ; P = C = $\mathbf{Z_n}$.
On définit K = {(n,p,q,a,b) : n = p*q, a*b = 1 mod φ(n) avec φ(n) l'indicateur d'Euler.
$\forall k \in K$ on définit $e_k(x) \equiv x^b \pmod n$ et $d_k(y) \equiv y^a \pmod n$ $\forall (x, y) \in Z^2$.
Les valeurs n,b sont publiques tandis que a,p,q sont privées.

1.1. Mise en œuvre de RSA

Si Bob désire que l'on puisse communiquer avec lui de façon discrète, il effectue les opérations suivantes :

a) *Bob engendre deux grands nombres premiers p et q (cf test de primalité)*
b) *Bob calcule n = p*q et φ(n) = φ(p*q) = (p – 1)*(q – 1)*
c) *Bob choisit un b aléatoire tel que 1 < b < φ(n) et pgcd(b, φ(n)) = 1 (cf algorithme d'Euclide)*
d) *Bob calcule* $a \equiv b^{-1} \pmod{\varphi(n)}$ *où* $a*b \equiv 1 \pmod{\varphi(n)}$ *(cf algorithme d'Euclide)*
e) *Bob publie b,n dans un répertoire et garde a,p,q qui forment la clé privée*

Supposons que Alice veuille envoyer le message M à Bob, alors elle calcule $\boxed{M^b \bmod n = C}$ et envoie C à Bob ; il calcule alors $\boxed{C^a \bmod n = M}$ pour retrouver le message original.

Preuve
Rappel : soit n = p*q et p,q premiers un à un, alors on a $\forall a \in]0, n[\ \ a^{1+(p-1)(q-1)} \equiv a \pmod n$
puisque pgcd(b,φ(n)) = 1 et a est l'inverse de b modulo φ(n) alors d'après le théorème de Bezout $\exists \alpha \in Z$ tel que a*b + α*φ(n) = 1
$\Rightarrow$ a*b = 1 – α*φ(n) = 1 + λ*φ(n) avec λ = -α
on sait que $C = M^b \bmod n$

donc $C^a = M^{ab} \bmod n = M^{1+\lambda(p-1)(q-1)} \bmod n$
or on a $1 + \lambda(p-1)(q-1) = 1 + (p-1)(q-1) + (\lambda-1)(p-1)(q-1)$
d'où $C^a = M^{1+(p-1)(q-1)} * M^{(\lambda-1)(p-1)(q-1)} \bmod n = M * M^{(\lambda-1)(p-1)(q-1)} \bmod n$
$= M^{1+(\lambda-1)(p-1)(q-1)} \bmod n$
$\Rightarrow$ en appliquant λ-1 fois le rappel on obtient $C^a = M \bmod n$.

1.2. Comment choisir un nombre premier

Supposons pour le moment que l'on connaisse un test permettant de déterminer si un grand nombre est premier ; on pourrait choisir au hasard un grand nombre n et tester s'il est premier. S'il n'est pas premier on teste si n+1 est premier, et ainsi de suite jusqu'à ce que l'on tombe sur un nombre premier.
Nous allons montrer que ce n'est pas une bonne méthode.

Théorème : il existe des suites arbitrairement longues de nombres entiers consécutifs qui ne sont pas premiers.
En effet soit la suite suivante : n!+2, n!+3, …, n!+n
on a $\forall q \in]2, n[$ p / n! et p / n!+p car n! = p*k $\Rightarrow$ n' + p = p*k + p = p*(k + 1)

Conclusion : la bonne technique de recherche d'un nombre aléatoire est de tester sa primalité.

Théorème (raréfaction des nombres premiers) : soit N un grand nombre alors le nombre (noté $\Pi(N)$) d'entiers inférieurs ou égaux à N est :

$$\Pi(N) = \frac{N}{\ln(N)}$$

Considérons l'ensemble E = {1,2,…,N} avec N un grand nombre. Si l'on veut prendre des entiers de 100 chiffres on va prendre $N = 10^{101}$, dans ce cas $\Pi(N) = \frac{N}{\ln(N)} = \frac{10^{101}}{101 * \ln 10}$.
Donc si l'on choisit au hasard un nombre n, la probabilité que n soit premier est :
$P(npremier) = \frac{N / \ln N}{N} = \frac{1}{\ln N} = \frac{1}{101 * \ln 10} \approx \frac{1}{232}$ soit une chance sur 232
On peut améliorer nos chances en choisissant un nombre impair ; il suffit de choisir le chiffre de la dernière décimale dans l'ensemble {1,3,5,7,9}. Dans ce cas on a environ une chance sur 93 d'obtenir un nombre premier.

Conclusion : bien que 93 soit un nombre relativement petit, il reste important par rapport au temps de calcul du test choisi, d'où la nécessité d'optimiser les tests de primalité.

1.3. Test de primalité

Test naïf : pour déterminer si un grand nombre n est premier, il suffit de tester successivement tous les diviseurs possibles de n.

Question : existe-t-il des astuces pour réduire ce test ?

Théorème : si n∈ **N** s'écrit comme un produit de (a,b)∈ $\mathbf{N}^2$, c'est-à-dire n = a*b avec a < b, alors $a \leq \sqrt{n}$.
Preuve (par l'absurde) : supposons que $a > \sqrt{n}$ on sait que b > a donc $a * b > \sqrt{n} * \sqrt{n} = n$ d'où a*b ≠ n ce qui n'est pas possible donc $a \leq \sqrt{n}$.

Conclusion : pour tester su un grand nombre n est premier, il suffit de tester les diviseurs a tels que $a \leq \sqrt{n}$.

Remarque : on sait qu'un nombre premier est forcément impair, donc si 2 / n alors n est pair donc n n'est pas premier.

Conclusion : après avoir testé 2, on ne teste que les nombres impairs ⇒ le temps de calcul est divisé par 2.

Algorithme :
```
entrée n
si n mod 2 = 0 et n ≠ 2
   n n'est pas premier
sinon pour (d = 3 ; d² < n ; d = d + 2)
      si n mod d = 0
         n n'est pas premier
      finsi
   finpour
   n est premier
finsi
```

Complexité : ce test de primalité exécute, si n est premier, $\frac{\sqrt{n}}{2}$ divisions.

Exemple : pour des entiers codés sur 32 bits (232), la complexité est de $\frac{\sqrt{2^{32}}}{2} = 2^{15}$ divisions.

Test de Fermat : ce test repose sur le théorème de Fermat.
Rappel : si n est premier alors $\forall a \in]0; n[$ on a $a^{n-1} \equiv 1 \pmod n$ donc on sait que :

a) *si n est premier, on a toujours $a^{n-1} \equiv 1 \pmod n$ quelque soit la base a comprise entre 0 et n*
b) *si n n'est pas premier, il se peut que pour certaines valeurs de la base a, $a^{n-1} \equiv 1 \pmod n$ soit quand même vérifiée*

Test : soit $n \in N$, on cherche à déterminer, avec une probabilité d'erreur aussi petite que possible, si n est premier :

a) *générer un nombre aléatoire dans l'ensemble {2,3,…,n-1}*
b) *calculer $a^{n-1} \bmod n$:*
   - *si $a^{n-1} \bmod n \neq 1$ alors n n'est pas premier*
   - *si $a^{n-1} \bmod n = 1$ alors n a de grandes chances d'être premier*

Remarque : il existe des nombres n non premiers pour lesquels l'égalité est vérifiée pour certaines valeurs de a ; ce sont des nombres « pseudo-premiers ».
Exemple : 341 qui passe avec 2, mais pas avec 3.

Remarque : certains nombres non premiers vérifient même l'égalité pour tout a : ce sont des nombres « pseudo-premiers absolus » ou de Carmichaël ; le plus petit d'entre eux est 561 (11*3*17).

1.4. Test de Lucas (comment fabriquer des nombres premiers)

Définition : les nombres de Mersenne sont les nombres M_p tels que $M_p = 2^p - 1$ avec p premier ; on construit une suite S_n en posant $S_1 = 4$ et $S_n = S_{n-1}^2 - 2$

Théorème : pour $p > 2$ on a M_p est premier si et seulement si M_p divise S_p
$\Rightarrow$ le plus grand nombre premier de Mersenne est $M_{1346697}$ et il possède 4053946 chiffres !

Calcul rapide de $a^x \bmod n$
si x est pair alors $a^x \bmod n = (a^2)^{x/2} \bmod n$
si x est impair alors $a^x \bmod n = a*a^{x-1} \bmod n = a*(a^2)^{(x-1)/2} \bmod n$

Algorithme de Hörner : soit l'écriture de x en base 2 : $x = x_{p-1}*2^{p-1} + x_{p-2}*2^{p-2} + \ldots + x_0 2^0$; les x_p prennent les valeurs 0 ou 1 :

```
entrée a,x,n
sortie a^x mod n
si x_{p-1} = 1
   r = a
sinon r = 1
finsi
pour i de p – 2 à 0 faire
   r = r^2 mod n
   si x_i = 1
      r = r*a mod n
   finsi
finpour
```

Exemple : $22 = (10110)_2$

i	bits	r
4	1	a
3	0	$a^2 \bmod n$
2	1	$(a^2)^2*a \bmod n = a^5 \bmod n$
1	1	$(a^5)^2*a \bmod n = a^{11} \bmod n$
0	0	$(a^{11})^2 \bmod n = a^{22} \bmod n$

Complexité : le nombre total de multiplications est égal au nombre de chiffres de l'exposant x en numération binaire plus le nombre de bits égaux à 1.

Signature électronique

Une signature idéale possède les propriétés suivantes :
- *elle ne peut être imitée*
- *elle n'appartient qu'à un seul document*
- *le document signé ne peut être modifié*
- *elle peut être contrôlée par un tiers*
- *elle ne peut être reniée*

Objectif : présenter une méthode de signature électronique applicable à un document et qui possède ces 5 propriétés.

Définition : un procédé de signature est un quintuplet (P,A,K,S,V) vérifiant :
- *P est un ensemble fini de messages*
- *A est un ensemble fini de signatures*
- *K est un ensemble fini de clés*
- *pour chaque k de K il y a une fonction de signature sig_k de S et une fonction de vérification ver_k de V ; les fonctions $sig_k : P \rightarrow A$ et $ver_k : A \rightarrow \{vrai,faux\}$ vérifient pour tout (x,y) de P*A :* $ver(x,y) = vrai$ *si* $y = sig(x)$; $ver(x,y) = faux$ *si* $y \neq sig(x)$

1. Signature RSA

Alice veut envoyer le message M signé et crypté à Bob :
1) *elle choisit ses clés publiques (n_A,b_A) et la clé privée a_A*
2) *elle calcule la signature $y = sig_A(M) = M^{aA} \bmod n_A$; c'est la seule personne à pouvoir calculer y puisqu'elle est la seule à connaître a_A*
3) *elle chiffre (M,y) en utilisant la clé publique de Bob b_B*
4) *le texte chiffré z est transmis à Bob*
5) *à la réception Bob déchiffre z à l'aide de sa clé secrète a_B et obtient (M,y)*
6) *il vérifie la signature y d'Alice à l'aide de sa clé publique b_A par $y^{bA} \bmod n_A$ et doit obtenir M*

2. Chiffrement et signature d'El Gamal

Le chiffrement d'El Gamal est basé sur l'algorithme de chiffrement discret ; on décrit d'abord le problème dans le corps $\mathbf{Z}_p$ où est p est premier.

Rappel : $\mathbf{Z}_p = \{0,1,\ldots,p-1\}$ avec $\mathbf{Z}_p^*$ l'ensemble des éléments inversibles dans $\mathbf{Z}_p$

Définition : soit p un nombre premier ; un nombre x est primitif modulo p si $x^i \neq 1 \bmod p$ $\forall i \in]0; p-1 = \varphi(p)[$ et $x^{p-1} \equiv 1 \pmod p$.

Problème du logarithme discret : instance du problème I = (p,d,B) où p est premier, α de $\mathbf{Z}_p$ primitif modulo p et β est dans $\mathbf{Z}_p^*$.

Question : trouver l'unique a tel que $0 < a \leq p-2$ et $\alpha^a \equiv \beta \pmod p$
$\Rightarrow$ on note cet entier $\log_\alpha \beta$.

Remarques :
- *ce problème est plus difficile à résoudre si p est convenablement choisi (ex : 150 chiffres)*
- *on ne connaît aucun algorithme polynomial pour résoudre ce problème*

2.1. Chiffrement d'El Gamal

Bob choisit :
- *p premier tel que le problème du logarithme discret soit difficile dans $\mathbf{Z}_p$*
- *α primitif modulo p dans $\mathbf{Z}_p$ et un entier a $0 < a \leq p-2$ tel que $\beta \equiv \alpha^a \pmod p$*

Les valeurs (p,α,β) sont publiques et a est privée.
Alice souhaite transmettre le message M à Bob :
- *elle choisit au hasard k de $\mathbf{Z}_{p-1}$, calcule $y_1 = \alpha^k \bmod p$ et $y_2 = M*\beta^k \bmod p$*
- *elle envoie le couple (y_1,y_2) à Bob*
- *Bob reçoit (y_1,y_2) et calcule $y_2*(y_1^a)^{-1}$ et doit obtenir M*

En effet $y_2*(y_1^a)^{-1} \bmod p = M*\beta^k*\alpha^{-ak} = M*\alpha^{ak}*\alpha^{-ak} = M$.

Remarque : le chiffrement d'El Gamal est non déterministe car l'opération de chiffrement dépend de M et d'une variable choisie aléatoirement par Alice. Il y a donc plusieurs textes chiffrés qui correspondent à un même texte clair.

Alice souhaite signer un message crypté M :
- *elle choisit ses clés $(p_A,\alpha_A,\beta_A,a_A)$ et Bob fait de même $(p_B,\alpha_B,\beta_B,a_B)$*
- *elle choisit un k de $\mathbf{Z}_{p-1}^*$ (privé), calcule $\gamma = \alpha^k \bmod p$ et $\delta = (C - a*\gamma)*k^{-1} \bmod p-1$*
- *elle envoie (C,γ,δ) à Bob (C : message crypté)*
- *Bob reçoit (C,γ,δ), calcule $\beta^\gamma*\gamma^\delta \bmod p$ et doit trouver α^C (vérification de la signature)*

En effet $\beta_A^{\gamma}*\gamma^{\delta} = \alpha_A^{aA\gamma}*\alpha_A^{k\delta} = \alpha_A^{aA\gamma}*\alpha_A^{k(C-aA\gamma)k-1} = \alpha_A^{aA\gamma+C-aA\gamma} = \alpha_A^{C}$.

Fonction de hachage

C'est une fonction qui transforme un message de longueur arbitraire en une empreinte numérique de taille fixée. Celle-ci est ensuite signée :

a) message x (longueur arbitraire)
b) empreinte numérique z = h(x) (α bits fixés)
c) signature $y = sig_K(z)$

Principe : lorsque l'on veut signer un message x, on calcule d'abord l'empreinte z = h(x), puis on signe avec y et on transmet (x,y).

1. Fonction de hachage à collisions difficiles

Il faut prendre quelques précautions pour que l'emploi des fonctions de hachage n'affaiblisse pas la sécurité du procédé.

Définition : une fonction de hachage est à collisions difficiles si étant donné un message x, il est difficile d'obtenir un message $x' \neq x$ de manière calculatoire avec $h(x') = h(x)$.

Définition : Une fonction de hachage est à sens unique si étant donné une empreinte z, il est difficile de trouver un message x de manière calculatoire tel que $h(x) = z$.

2. Construction

Supposons que (P,C,K,E,D) soit un système cryptographique avec $P = C = K = \mathbf{Z_2^n}$; soit x un message de longueur n.

Notation : x || y représente la concaténation de x et y.

On peut alors décomposer x en la concaténation suivante : $x = x_1 \| x_2 \| \ldots \| x_n$ avec $x_i \in \mathbf{Z_2^n}$.
L'idée de la construction consiste à fixer une valeur g_0 et à calculer successivement $g_1, g_2, \ldots, g_m$ de telle sorte que $g_i = f(x_i, g_{i-1})$ où f est une fonction qui utilise la règle de chiffrement.

$\Rightarrow$ le résultat final du hachage est $g_m = h(x)$.

Transmission et sécurité de l'information

1. Incertitude

Définition : soit X une variable aléatoire d'un nombre fini d'événements $\{x_1,x_2,\ldots,x_n\}$; l'incertitude sur X est définie par l'entropie :

$$H(X) = -\sum_{i=1}^{n} P(X = x_i) * \log_2 P(X = x_i)$$

Cette valeur est aussi appelée quantité d'information, une fois l'incertitude levée.

Définition : soient X et Y deux variables aléatoires avec $X = \{x_1 \ldots x_n\}$ et $Y = \{y_1 \ldots y_n\}$; l'incertitude conditionnelle sur X sachant Y est définie par

$$H(X/Y) = \sum_{x,y} P(X = x/Y = y) * \log_2 P(X = x/Y = y)$$

Elle représente l'incertitude qu'il reste sur X lorsque l'on connaît Y.

2. Application à la cryptographie

On peut modéliser un système cryptographique de la manière suivante :

- *le message clair M est défini comme une variable aléatoire à valeurs dans l'ensemble des messages clairs possibles*
- *le message chiffré C est défini comme une variable aléatoire à valeurs dans l'ensemble des messages chiffrés possibles*
- *un cryptanalyste (attaquant) doit obtenir une information de l'ordre de H(M) pour retrouver le message clair M*

Théorème : un système cryptographique est dit parfait si

$$H(M/C) = H(M)$$

Autrement dit la connaissance du message chiffré C n'apporte aucune information sur le message clair.