Université de Paris - UFR Mathématiques et Informatique

Projet de Mathématique S2

Grands nombres premiers et système RSA

Réalisé par :
M. Amine KEBOUCHE
M^me^ Malek AMRI

Encadrants :
M. Nasser TEBBACHE

Version du
11 mai 2020

Table des matières

Table des figures

1 Introduction

La quintessence de l'entrepreneuriat se base sur la création d'un besoin ou bien répondre à un besoin donné. Trois décennies auparavant, l'humanité ne disposait pas d'internet dans le domaine publique. L'usage des dispositifs de communication et l'outil informatique se démocratise et se répandent progressivement. De nos jours, on peut s'assurer de l'usage sécurisé d'un traitement de texte online sur Google en jetant un simple coup d'oeil sur l'adresse url.

L'explosion du volume d'échange d'information a engendré des prises de précautions pour stocker, accéder, diffuser et sécuriser les données. L'aspect sécuritaire est primordial ce qui a boosté les recherches dans le domaine de cryptographie.

En effet, la sécurité informatique, ou la Cybersécurité s'appuie sur la cryptographie à fin de protéger les espaces cyber contre d'éventuelles actions malveillantes. Ceci nécessite la mise en place d'un ensemble de stratégies dont des algorithmes pour évaluer les risques et définir les objectifs de sécurité à atteindre.

Ainsi, la sécurité informatique est un ensemble de moyens techniques, organisationnels, juridiques et humains utilisés pour garantir la sécurité des systèmes manipulés. Notamment la sécurité des données et des communications qui est assurée principalement par l'utilisation de la Cryptographie (Science de codage), considérée comme étant le noyau de la sécurité informatique.

2 Histoire et concepts de base

La cryptographie est la science qui utilise les mathématiques pour chiffrer et déchiffrer les données. Elle permet la protection des données stockées ou transmises, à travers des réseaux non sûrs (comme Internet), de telle sorte qu'elles ne puissent être interceptées à l'exception des destinataires convenus. la confidentialité, l'intégrité des données, l'authentification et la non-répudiation constituent les bases fondamentaux de la cryptographie.

À travers L'histoire de la civilisation humaine, le recours au cryptages des messages a évolué en fonction de l'avancée des sciences et essentiellement des mathématiques. 2000 ans Ac, les Égyptiens utilisaient le cryptage par mot clé pour sécuriser leur communications ; il suffisait de disposer dudit mot clé pour pouvoir crypter et décrypter les messages. De plus, Jule César avait recours à la même technique durant l'empire romaine, d'où l'algorithme de césar qui est utilisé dans la cryptographie classique.

L'avènement des ordinateurs a accéléré l'évolution de la cryptographie et des algorithmes inhérentes . Les techniques d'intrusion ont aussi connu un essor remarquable qui ne cessent d'accroître en efficacité. Par conséquent, la sécurité a connu l'émergence des techniques associées à la cryptographie, telles que le hachage, la signature et la gestion de clés.

On distingue deux types de chiffrement qui reposent sur deux stratégies différents quand à l'usage de clés de cryptage et de décryptage :

2.1 Les algorithmes symétriques :

Autrement dit algorithmes à clé secrète qui reposent sur le principe de partager la même clé secrète entre les interlocuteurs afin d'échanger de différentes informations confidentiellement.

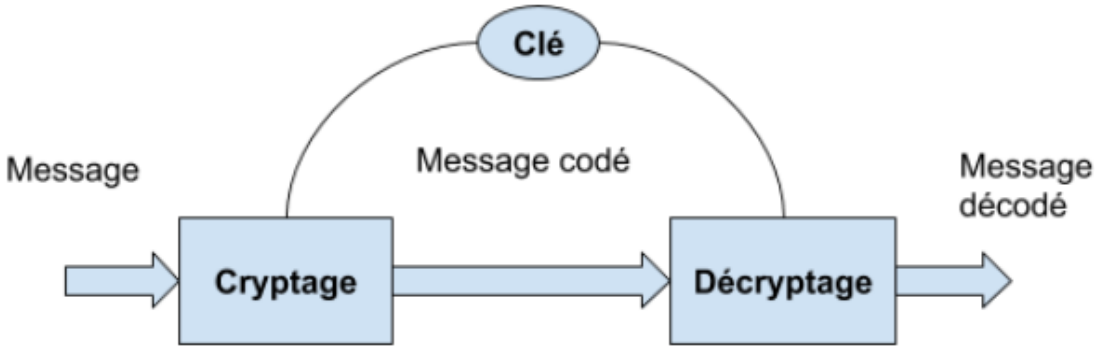

FIGURE 1 – Système de cryptage symétrique

2.2 Les algorithmes asymétriques :

Ou algorithmes à clé publique qui consiste à générer deux clé distinctes.Une clé publique qui est mise à la disposition de toute personne désirant envoyer une information au propriétaire,et une clé privée qui est utilisé uniquement par le destinataire pour le déchiffrement des messages.

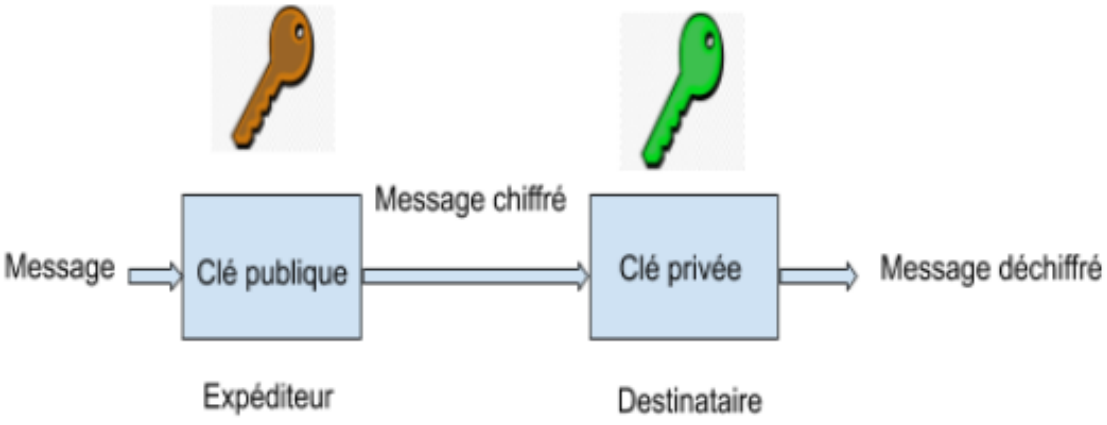

FIGURE 2 – Système de cryptage asymétrique

2.3 Systeme cryptographique RSA :

Le principe de clé publique a été proposé en 1976 par Whitfield Diffie et Martin Hellman, et implémenté en 1978 par Rivest, Shamir et Adleman sous la forme de l'algorithme RSA qui est le fruit des recherches académique de l' MIT, appliqué dans plusieurs domaines de sécurité.

L'algorithme RSA est un systhème de chiffrement asymétrique qui se base sur la difficulté de la factorisation d'un nombre entier en produit de nombres premiers, et la fonction à sens unique qui est une fonction "puissance".

Avant de détailler cet algorithme, quelques notions mathématiques sur lesquelles RSA est construit doivent être introduites.

3 Prérequis mathématiques :

Dans cette partie on note l'enssemble des nombres relatifs par Z, et l'enssemble des nombres positifs par Z_n.

3.1 Généralités sur les nombres premiers :

Définition1. Soient **a** et **b** deux entiers relatifs ; on dit que **b** divise **a** noté **b | a**, s'il existe un entier **c** tel que **a = c.b**.
Exemple : 6 | 12 car il existe 2 tel que : 12=6×2

Définition2. Un nombre premier est un entier naturel supérieur à 1 dont les seuls diviseur sont 1 et lui-même.

Théorème 1. (D'après EUCLIDE)
Pour tout ensemble fini **E** de nombres premiers, il existe un nombre premier **p** qui n'appartient pas à **E**.
Autrement dit, il existe une infinitée de nombres premiers.

Théorème 2.(Décomposition en produit de facteurs premiers)
Tout nombre entier naturel s'écrit comme un produit de nombres premiers ; cette

décomposition en facteurs premiers est unique (exceptée par réarrangement des facteurs).
Exemple : 600= $5^2 \times \mathbf{2^3} \times 3$
et il n'admet aucune autre factorisation sous forme de produits de nombres premiers, excepté par réarrangement des facteurs 2,3 et 5.

La décomposition en facteurs premiers est aisée pour de petits nombres, mais elle est très longue pour de grands nombres ; La sécurité du système asymétrique RSA est basée sur **la difficulté de la factorisation de grands nombres.**

3.1.1 Test de primalité :

Definition 3 : Un test de primalité est un algorithme qui détermine si un nombre entier est premier.

Algorithme1. (Test de primalité) Vérifie si un entier N est divisible par l'un des entiers entre 2 et N-1, le cas contraire indique que N est premier.

Fonction premier(N)
Verifier que N ≥ 2
pour i compris entre 2 *et* $N - 1$
si i divise N
Retourner Faux
Retourner Vrai

Le crible d'Eratosthène

Le crible d'Eratosthène décrit un moyen de calculer la liste des nombres premiers inférieurs à un entuer fixé. La méthode consiste à créer dans un premier temps la liste des entiers compris entre 2 et un entier **n** puis itérer le processus suivant :

1- Récupere le premier entier de la liste (Ce nombre est premier)
2- Retirer de la liste restante les multiples de ce nombre
3- Tant que la liste n'est pas vide, recommencer à l'étape 1

3.1.2 Fonction totient d'Euler :

La fonction est définie par : $\phi(n) = card(\{m \leq n | PGCD(m,n) = 1\})$

En théorie des nombres, la fonction phi d'Euler donne le nombre d'entiers positifs jusqu'à un entier donné n qui sont relativement premiers à n. Autrement dit, c'est le nombre d'entiers k dans l'intervalle 1 k n pour lequel le plus grand commun diviseur pgcd(n, k) est égal à 1.
- Si n est premier, alors $\phi(n) = n1$.
$-$ *Deplus, soient p etq deux nombres premiers et* $n = p.q$, *alors* :

$$\phi(n) = \phi(pq) = \phi(p)\phi(q) = (p-1)(q-1)$$

3.2 Algorithme d'EUCLIDE étendu :

Définition4.(PGCD)
Le PCGD de deux entiers naturels **a** et **b** est le plus grand élément de l'ensemble de leurs diviseurs communs ; on le note **PGCD(a,b)**.

Propriété1 : Soient **a** et **b** deux entiers positifs non nuls (a>b) et **r** le reste de la division euclidienne de **a** sur **b** : **a = b.q + r** alors : **PGCD(a,b) = PGCD(b,r)**

On calcule le PGCD de deux nombres en utilisant l'algorithme d'EUCLID
Algorithme2.(EUCLID) Soient **a** et **b** deux entiers naturels avec **a** $\geq b$:

Fonction Euclide1(a,b)
tant que b $\neq 0$
$t \leftarrow b$
$b \leftarrow a$ *modulo* b $(a\%b)$
$a \leftarrow t$
retourner a

Fonction Euclide2 (a, b)
tant que a $\neq b$
si $a > b$
$a \leftarrow a - b$
sinon
$b \leftarrow b - a$
retourner a

Euclide1 : Se base sur la **Propriété1** et la divisions successives
Euclide2 : Soustractions successives
Définition5.(Deux nombres premiers entre eux)
Soient **a** et **b** deux entiers positifs non nuls, on dit que **a** et **b** sont prmiers entre eux si **PGCD(a, b) = 1**
Exemple : Calculons le PGCD de 61 et 16

$61 = 16 \times 3 + 13$
$16 = 13 \times 1 + 3$
$13 = 3 \times 4 + \mathbf{1}$
$3 = 1 \times 3 + \mathbf{0}$

61 *et* 16 *sont premiers entre eux car* $PGCD(61, 16) = 1$

3.2.1 Théorème de Bézout :

Théorème3. Soient **a** et **b** des entiers, il existe des entiers **u, v** $\in \mathbb{Z}$ *tels que* :

a.u + b.v = PGCD(a,b)

u,v sont les coeficients de Bézout, il s'obtiennent en remontant l'algorithrme d'Euclide.
Exemple : Calculons les coefficients de Bézout correspondant à PGCD (61,16)

$61 = 16 \times 3 + 13$
$16 = 13 \times 1 + 3$
$13 = 3 \times 4 + \mathbf{1}$
$3 = 1 \times 3 + \mathbf{0}$

$1 = 61 \times \mathbf{5} + 16 \times (\mathbf{-19})$
$\Uparrow 1 = (61 - 16 \times 3) \times 5 - 16 \times 4$
$\Uparrow 1 = 13 \times 5 - 16 \times 4$
$\Uparrow 1 = 13 - (16 - 13 \times 1) \times 4$
$\Uparrow 1 = 13 - 3 \times 4$

Nous avons vu précédemment l'algorithme d'Euclide qui permet de calculer le PGCD de deux nombres entiers, et comment calculer les coeffiecient de Bézout à la main en remontant cet algorithme de bas en haut.

L'algorithme d'EUCLIDE étendu permet de calculer les coefficients de Bézout qui correspondent à deux nombres entiers **a** et **b** d'une manièr automatique.

Pour cela, on définit deux suites qui vont aboutir au coefficients de Bézout de **a** et **b (a > b)**

Soient U_n *et* $\mathbb{V}$ *deux suites relles d f inies par leurs premiers termes et leurs relations de rccurences*.

Pour tout entier $n > 1$:

$$\begin{cases} U_0 & = 1 \\ U_1 & = 0 \\ U_{n+1} & = U_{n-1} - q_n.U_n \end{cases} \qquad \begin{cases} V_0 & = 0 \\ V_1 & = 1 \\ V_{n+1} & = V_{n-1} - q_n.V_n \end{cases}$$

Où q_n *est le cotient de la division euclidienne de* $a_n$par b_n

Algorithme3.(EUCLID étendu) :

Fonction Euclide-etendu (a,b)

U $\leftarrow 1$; $UU \leftarrow 0$

$V \leftarrow 0$; $VV \leftarrow 1$

Tant que $b \neq 0$

$q \leftarrow a/b$; $c \leftarrow a$

$a \leftarrow b$; $b \leftarrow c\%b$

$d \leftarrow UU$; $UU \leftarrow x - q.UU$

$U \leftarrow d$; $f \leftarrow VV$

$VV \leftarrow V - q.VV$

$V \leftarrow f$

$if\ a < 0$

$a \leftarrow -a$; $U \leftarrow -U$; $V \leftarrow -V$

retourner (a, U, V)

3.3 Inversabilité sur Z_n

3.3.1 Congruences :

Définition6. Soit un entier n > 2. On dit que a **congru** b **modulo** n si n divise a-b

et on écrit a $\equiv b[n]$

Autrement dit : a $\equiv b[n] \iff \exists k \in Z\ a = k.n + b$

Proposition :

- $a \equiv a(mod\ n)$,
- *si* $a \equiv b(mod\ n)$ *alors* $b \equiv a(mod\ n)$
- *si* $a \equiv b(mod\ n)$ *et* $b \equiv c(mod\ n)$ *alors* $a \equiv c(mod\ n)$
- *Si* $a \equiv b(mod\ n)$ *et* $c \equiv d(mod\ n)$ *alors* $a + c \equiv b + d(mod\ n)$
- *Si* $a \equiv b(mod\ n)$ *et* $c \equiv d(mod\ n)$ *alors* $a \times c \equiv b \times d(mod\ n)$
- *Si* $a \equiv b(mod\ n)$ *alors pour tout* $k \geq 0$, $a^k \equiv b^k(mod\ n)$

Exemple : $17 \equiv 5[6], \quad 3 \equiv -11[7]$

3.3.2 Inverse modulo n

Définition7. Soit **a** $\in \mathbb{Z}$, *on dit que* **x** $\in \mathbb{Z}$ *est un* **inverse de a[n]** *si*

$ax \equiv 1[n]$

Proposition :

- *Si* **a** *et* **n** *sont premiers entre eux , alors a admet un inverse modulo n*
- *Si* $au + nv = 1, (PGCD(a, n) = 1)$, *alors* **u** *est un inverse de* $a[n]$

Autrement dit : trouver un inverse de a modulo n revient à calculer les coefficients de Bezout associé à (a, n)

Algorithme4.(Renvoi l'inverse)

```
Fonction inverse(a, n)
    c, u, v = Euclide-etendu(a, n)
    si c ≠ 1
        retourner 0
    sinon
    retourner u modulo n
```

Exemple : Prenons l'exemple de du Théorème3 : En aplliquant l'algorithme d'Euclide étendu sur (61, 16) on a trouvé : 1= 61×**5** + 16 × (**-19**)
telque $PGCD(61,16) = 1$ *et*5, *et* -19 *sont les coefficients de Bezout*
On lit 5 *est* **l'inverse de 61 modulo 16**.

3.3.3 Petit théorème de Fermat :

Théorème4. Si **p** est un nombre premier et **a** $\in \mathbb{Z}$, *alors* :

$$\mathbf{a}^p \equiv a[p]$$

Proposition : Si **p** ne divise pas **a**, alors :

$$\mathbf{a}^{p-1} \equiv 1[p]$$

Théorème de Fermat amélioré : Soient **p** et **q** deux nombres premiers distincts, et soit **n=p.q**. Pour tout a$\in \mathbb{Z}$ *tel que* $PGCD(a,n) = 1$ *alors* :

$$\mathbf{a}^{(p-1)(q-1)} \equiv 1[n]$$

Le principe du déchiffrement RSA sera basé sur ce petit théorème de Fermat amélioré (Extait du Théorème4), ainsi que la notion de l'inversibilité dans Z_n.
À partir de ce théorème amélioré de Fermat, on déduit cette proposition qui permet le déchiffrement d'un message codé :
Proposition : Soit **d** l'inverse de **e** modulo $\phi(n)$,

Si C $\equiv M^e[n]$ *alors* $M \equiv C^d[n]$

3.4 L'exponentiation rapide modulaire :

L'exponentiation rapide est utilisée pour calculer des puissances modulo n. Pour manipuler des entiers ayant des dizaines, voir des centaines de chiffres, cette methode rapide et efficace sera utilisé dans RSA pour chiffrer et déchiffrer les messages.

Exemple : Appliquons cette methode sur l'exemple $40^{13}(mod\ 85)$
L'ide est de seulement calculer $40, 40^2, 40^4, 40^8, ...$
et de reduire modulo n chaque fois On remarque que $13 = 8 + 4 + 1$ *donc* :

$$40^{13} = 40^1 \times 40^4 \times 40^8$$

Calculons donc les $40^{2^i}(mod\ 85)$:

$$40 \equiv 40(mod\ 85)$$

$$40^2 = 1600 \equiv 70(mod\ 85)$$
$$40^4 = (40^2)^2 \equiv 70^{40} \equiv 4900 \equiv 55\ (mod\ 85)$$
$$40^8 = (40^4)^2 \equiv 55^2 \equiv 3025 \equiv 50(mod\ 84)$$
$$40^{13} \equiv 40^{8+4+1} \equiv 40^8 \times 40^4 \equiv 40 \equiv 50 \times 55 \times 40 \equiv 10(mod\ 85)$$

4 Algorithme de RSA :

Remarque : Afin de simplifier les calcules, les exemples seront illustrés avec des petits nombres, cependant, en pratique le systhème RSA ne manipule que des grands nombres (centaines de chiffres voir mille).

4.1 Fonctionnement et principe de RSA :

L'algorithme RSA est constitué de cinq parties essentielles : la génération des clés, distribution de la clé publique , le chiffrement et le déchiffrement du message et enfin la signature numérique qui permet d'identifer l'émetteur. Le message doit étre numérisé, autrement dit, l'emeteur transforme son text en nombres (chaque lettre sera représentée par le numero de son ordre alphbétique par exemple, son code ASCII ...)

Admettant qu'une entité **B** souhaite envoyer un message secret à l'entité **A**, alors le protocole sera comme suit :

1- **A** prépare une clé publique qui sera mise à la disposition de tout autre utilisateur, ainsi qu'une clé privée qui doit rester secte.

2- **B** numérise d'abord son message **M** en utilisant l'une des méthodes évoquée précédemment, puis le code en utilisant la clé publique.

3-**A** reçoit le message crypté et le déchiffre avec sa clé privée.

4.1.1 Calcul de la clé publique et de la clé privée :

Choix de deux nombres premier assez grands :

L'utilisateur **A** effectue ces calcules secretement :

- Choisir deux nombres premiers p et q (assez grands dans la pratique).

- Calcule leur produit **n=p.q**

- Calcule la fonction d'Euler $\phi(n) = (p-1)(q-1)$

La factorisation de $\phi(n)$ *ne sera possible que si l'on connais les facteurs de* **n**

Exemple :

$\circ p = 37$ *et* $q = 43$

$\circ\ n = 37 \times 43 = 1591$

$\circ\ \phi(n) = (37-1) \times (43-1) = 36 \times 42 = 1512$

Choix d'un exposant et calcul de son inverse :

L'entité **A** continue :

- Choisir un exposant **e** tel que PGCD(e, $\phi(n)) = 1$

- Calculer l'inverse **d** de e modulo $\phi(n)$, *tel que* $d \times e \equiv 1[\phi(n)]$.

(Cela se fait en utilisant l'algorithme d'Euclide étendu.)

Exemple :

$\circ PGCD(e, \phi(n)) = 1, \implies PGCD(e, 1512) = 1$

$\circ\ E = \{11, 13, 19, 23, 887, 895...\}$ *(Possibilits verifiant* $PGCD(e, 1512) = 1$.*)*

$\circ$ *On choisit un* $e \in E$

$\circ\ e = 23$

$\circ\ d \times e \equiv 1[\phi(n)]$ *(d inverse de e)* $\implies d \times 23 \equiv 1[1512]$

$\circ\ d = 263$

Clé publique :

La clée publique est constituée des deux nombres **n** et **e**. L'entité A la met à la disposition de tout autre utilisateur, leur permettant de crypter les données envoyées .

Exemple :

clé publique est (n , e) = (1591,23)

Clé privée :

Le créateur des clés garde secretement sa clé privée constituée du nombre **d** qui permet le déchiffrement du message codé par la clé publique. Il détruit les nombres **p**, **q** et $\phi(n)$ *qui ne sont plus utile.*

Exemple :

cle prive est $d = 263$

4.1.2 Chiffrement du message :

Une entité **B** souhaite envoyer un message secret **m** au destinataire **A**. Pour cela, elle :
1- Numérise d'abord son texte **m** en remplçant chaque lettre par son rang alphabetique (d'autre méthodes de numérisation sont possibles).
2- Découpe le message numérisé en blocs de même taille de sorte que chaque bloc **M** soit inferieur à **n**. Cette étape est nécessaire car plus le bloc est assez long, plus il est difficile de l'attaquer.
3- L'entité **B** partage le processus de découpage, ainsi que la methode choisi pour la numérisation du message avec l'entité **A**
Message :
Le message est un entier **M** tel que $0 \leq M < n$.
3- L'utilisateur **B** récupère la clé publique **(n,e)** de **A** avec laquelle il va chiffrer chaque bloc **M** de son message numérisé grace à l'algortithme d'exponotiation rapide comme suit :

$$\mathbf{C} \equiv M^e[n].$$

4- Le message codé composé des blocs **C** sera ensuite transmit à l'entité **A**.
Exemple :
$\circ m = RSA \implies 18\ 19\ 01$
$\circ$ *Decoupage en morceaux de meme taille* $3 \implies M = 181\ 901$
$\circ$ *Chiffrement des blocs*
$C_1 \equiv (181)^{23}[1591] = 382$
$C_2 \equiv (901)^{23}[1591] = 520$
On obtient le code $C = 382520$

4.1.3 Déhiffrement du message :

L'entité **A** reçoit le message chiffré envoyé par l'utilisateur **B**. Pour le décrypter, elle utilise sa clé privée **d**, ainsi que l'algorithme de l'exponentiation rapide. À l'aide de la proposition du Théorème de Fermat amélioré évoquée précedemment, l'entité **A** retrouve finalement le message **M** :

$$\mathrm{M} \equiv C^d[n]$$

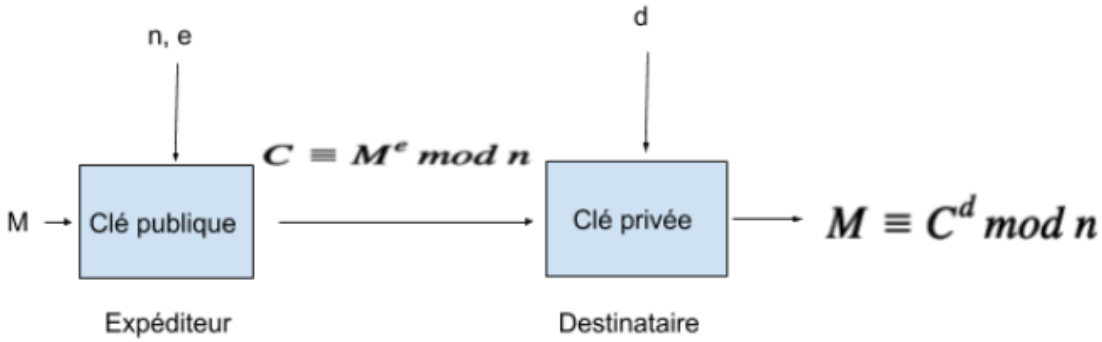

FIGURE 3 – Fonctionnement du système RSA

Exemple :
Déchiffrer le code C= 382520
$\circ$*Dcoupage en morceaux de meme taille* 3 $\Longrightarrow$ 382 520
$\circ$ *Dchiffrement du code C* :
$M_1 \equiv (382)^{263}[1591] = 181$
$M_2 \equiv (520)^{263}[1591] = 901$
Le rsultat est : 181901
$\circ$ *Dcoupage en morceaux de taille* 2 *pour reconstruire le message* 18 19 01 $\Longrightarrow$ *R S A*.

4.1.4 La signature RSA :

La signature électronique a le même rôle que la signature manuscrite utilisée quotidiennement dans les documents officiels. Il s'agit d'un procédé assurant l'authenticité du signataire (auteur), autrement dit : garantir que l'information provient de la bonne source, ainsi que l'intégrité de son document, c'est à dire que ce dérnier n'a pas été modifié ou altéré par d'autres entités non autorisées ou inconnues.
Comme tout les procédés de signature numérique, le procédé de la signature RSA est composé d'un **algorithme de signature** et d'un **algorithme de vérification**. L'expiditeur **A** signe son message **M** suivant l'algorithme secret de la fonction **S**, ensuite le recepteur **B** vérifie le résultat **R=S(M)** grace à l'algorithme publique de la fonction **ver** qui retourne **VRAI** si R est une authentique signature de **M**, **FAUX** si non.

Soit p , q nombres premiers et n=p.q, (n,e), d sont les clés publique et privée (respictivement) crée par l'algorithme RSA.

-Pour signer son message, l'entité **A** utilise la fonction suivante :

$$\text{S(M)} \equiv M^d[n].$$

- Pour identifier l'expediteur, l'entité **B** utilise la fonction de vérification associée :

$$\text{ver(M, R)}= \begin{cases} Vrai\ si\ M\ \equiv R^e[n] \\ Faux\ si\ M \not\equiv R^e[n] \end{cases}$$

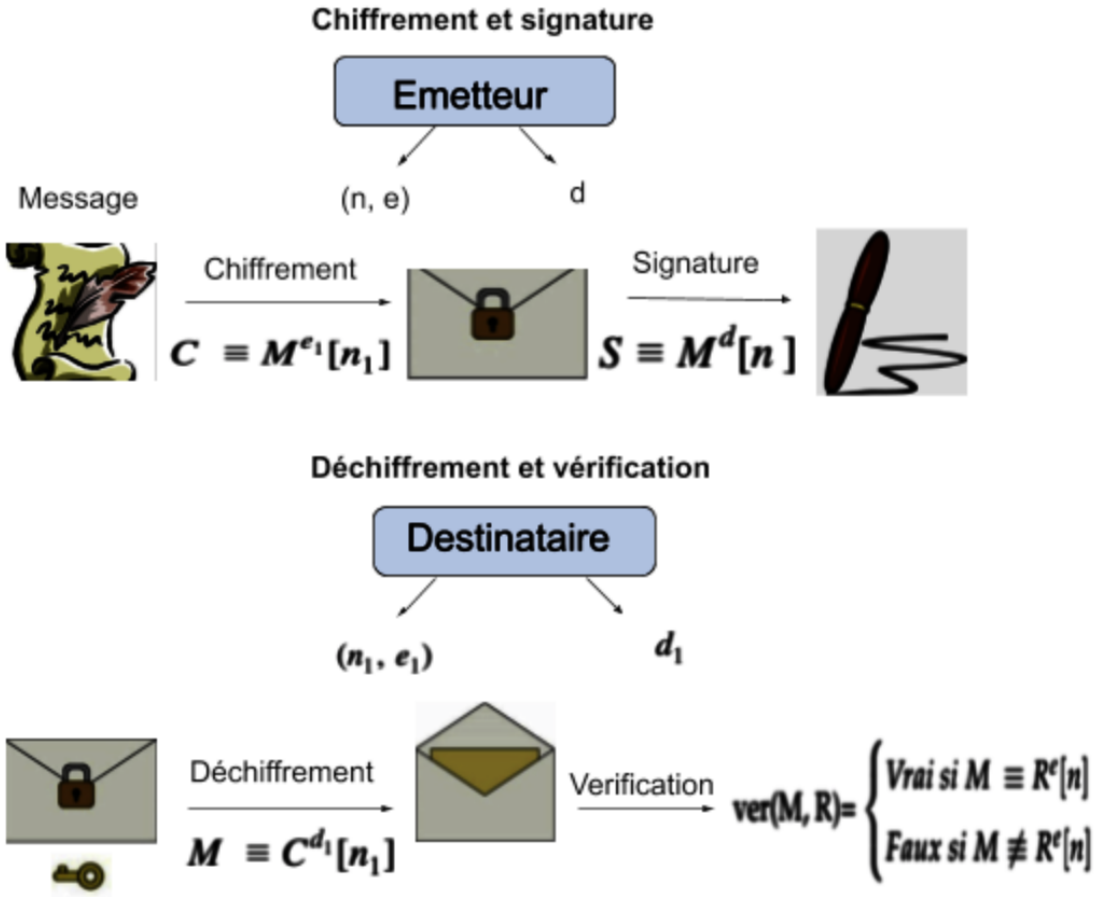

FIGURE 4 – Fonctionnement de la signature RSA

5 Sécurité et robustesse de RSA :

La sécurité du système RSA repose principalement sur la difficultée de la factorisation du nombre **n** en deux nombres premiers **p** et **q**, ainsi que le temps nécessaire pour effectuer cette opération qui croit exponentiellement en fonction de la longueur de **n**. En effet, si un utilisateur réussit à factoriser **n** il en déduira ensuite O(n) et donc calculera facilement la clé secrète **d**. La confiance accordée au système RSA est dû aux échecs de toutes tentatives pour le détruire. Neanmoins, ces échecs redondantes ont conduit à la formulation des nouvelles recommandations pour le choix des paramètres **p** et **q**. Par exemple, les experts recommandent l'usage des nombres **n** entre 1024 et 2048 bits dans le cas de donneés ayant une grande importance.

La fameuse factorisation reste toujours trés difficile et lente même s'il existe des méthodes plus efficaces que les méthodes classiques (test de primalité de Fermat, crible d'Erastothène vu ci-dessus, etc ...).

Les meuilleurs algorithmes de factorisation actuels peuvent factoriser des nombres jusqu'à 230 chiffres en plusieurs mois de calculs en utilisant des centaines de machines très puissantes. Cela implique que plus le nombre est grand, plus le temps du calcul des facteurs est important, d'où le conseil des experts de choisir des nombres assez grands. En appliquant les méthodes connus à ce jour, il semble que la factorisation des nombres à 1000 chiffres restera impossible, ce qui met en évidence la robustesse du systeme RSA. Parmis les fameux algorithmes qui ont essayé de briser ce système depuis son apparition, la crible algébrique, l'attaque de Wiener..

L'utilisation de nombres assez grands rend RSA plus sûr et réduit les risques d'attaques, en revanche, le procédé du chiffrement et déchiffrement, etant basé sur ces grands nombres, devient trés lent et compliqué quand il s'agit de transmettre des informations à taille considèrable. Pour cela, dans la pratique on s'en sert le plus souvent de transmettre des clés d'autre système plus rapide (symetriques) servant à déchiffrer ces informations.

6 Bibliographie

DELAHAYE, Jean Paul. La Cryptographie RSA Vings Ans Après . Jan. 2000. http ://www.lifl.fr/ jdelahay/dnalor/RSA.pdf

BODIN, Arnaud et RECHER, François. Cryptographie. Exo7
http ://exo7.emath.fr/cours/ch$_c$*rypto.pdf*

SystmeRSA www.cryptage.org/rsa.html

BUCHMANN, Johannes LaFactorisation Des Grands Nombres.Sept.1998.
https : //www.apprendre − en − ligne.net/crypto/rsa/251_088_096*.pdf*

VIGOUREUX, Pierre.Liberts individuelles et codes secrets. Mars 2006

STINSON, Douglas.CRYPTOGRAPHIE.Thorieetpratique.2000