

# Table des matières

<b>Introduction</b>	<b>3</b>
<b>1 Rappel sur les corps finis</b>	<b>5</b>
1.1 Structure de corps . . . . .	5
1.2 Définition de sous-corps . . . . .	6
1.3 Morphisme de corps . . . . .	7
1.4 Corps finis . . . . .	7
1.5 La caractéristique d'un corps . . . . .	8
1.6 Construction des corps finis . . . . .	10
1.7 Calculs dans un corps fini - Table de logarithmes . . . . .	11
<b>2 La cryptographie à clé publique</b>	<b>14</b>
2.1 Les principaux buts de la cryptographie . . . . .	15
2.2 Où utilise-t-on la cryptographie ? . . . . .	16
2.3 Un système cryptographique . . . . .	17
2.4 La cryptographie à clefs privées . . . . .	17
2.4.1 Alphabets désordonnés . . . . .	17
2.4.2 Le Chiffre de César . . . . .	19
2.5 Protocole d'échange de clés de Diffie-Hellman . . . . .	20
2.5.1 La signature . . . . .	21
2.6 Cryptographie à clefs publiques . . . . .	21

2.6.1	Algorithme de chiffrement à clé publique de RSA . . . . .	22
2.6.2	Algorithme de chiffrement à clé publique d'El Gamal . . . . .	27
<b>3</b>	<b>Étude sur la complexité des algorithmes R.S.A et EL GAMAL</b>	<b>30</b>
3.1	La complexité . . . . .	30
3.2	L'algorithme . . . . .	30
3.3	Complexité en temps et en espace . . . . .	31
3.4	Ordres de grandeurs . . . . .	31
3.4.1	Types de complexité algorithmique . . . . .	32
3.4.2	Ordres de grandeurs . . . . .	33
3.5	Complexité du système cryptographique du RSA . . . . .	34
3.6	Complexité du système cryptographique d'El Gamal . . . . .	34
	<b>Conclusion</b>	<b>38</b>

# Introduction

Dès que les hommes apprenent à communiquer, ils font des moyens pour assurer la confidentialité d'une partie de leurs communications : l'origine de la cryptographie remonte sans doute aux origines de l'homme.

En effet, le mot cryptographie du grec *kryptos* graphein -qui signifie caché l'écriture- est l'art de transformer un message pour tenter de le rendre illisible par toute autre personne que son destinataire. La cryptographie est l'étude des méthodes et techniques permettant de transmettre des données de manière confidentielle. Au cours des siècles, de nombreux systèmes de cryptage ont été inventés, tous de plus en plus perfectionnés, et il est vrai que l'informatique y a beaucoup contribué. Mais au commencement les algorithmes étaient loin d'être aussi complexes et astucieux qu'à notre époque. La majeure partie des méthodes classiques reposait sur deux principes fondamentaux : la substitution (remplacer certaines lettres par d'autres) et la transposition (permuter des lettres du message afin du brouiller). On distingue deux grands systèmes de cryptage : les cryptosystèmes symétriques où la méthode de cryptage (ou de décryptage) est connue par l'expéditeur et le destinataire, et les cryptosystèmes asymétriques (dit aussi à clefs publiques). L'idée de base des cryptosystèmes à clefs publiques a été proposée dans un article fondamental de Diffie et Hellman en 1976. Le principe fondamental est d'utiliser des clefs de chiffrement et de déchiffrement différentes, non reconstituables l'une à partir de l'autre :

- Une clef publique : connue par tout le monde, utilisée généralement pour crypter ou vérifier la signature des messages.
- Une clef secrète (privée) : connue uniquement par le détenteur, utilisée pour décrypter et signer des messages.

Dans ce mémoire de fin d'étude Master, on s'intéresse à l'étude de deux systèmes cryptographiques à clefs publiques qui sont R.S.A et EL GAMAL et la complexité de leurs algorithmes. Le premier système est basé sur la difficulté de factoriser un grand

nombre entier positif en produit de deux nombres premiers et le second système est basé sur le problème du logarithme discret dans un corps fini.

Ce mémoire est subdivisé en trois chapitres. Le premier chapitre est consacré aux outils algébriques nécessaires qui sont les corps finis et leurs propriétés pour la construction de systèmes de cryptage. Dans le second chapitre on donne en premier lieu un aperçu sur la cryptographie et les domaines de son utilisation. Ensuite, on présente les deux systèmes cryptographiques RSA et EL GAMAL. Le troisième chapitre porte sur la complexité des algorithmes en terminant le chapitre par une étude de la complexité des algorithmes RSA et EL GAMAL.

# Chapitre 1

## Rappel sur les corps finis

Dans ce chapitre, on fait un rappel sur les corps finis et leurs propriétés essentielles. La plupart des notations et résultats dans ce chapitre on pourra les consulter dans [6], [7] ou bien [11].

### 1.1 Structure de corps

#### Définition 1.1.1

Soit  $K$  un ensemble muni de deux lois  $+$  et  $\times$ . On dit que  $(K, +, \times)$  est un corps si :

- $(K, +, \times)$  est un anneau commutatif non réduit à  $\{0\}$ .
- Tout élément non nul de  $K$  ( $K^\times = K - \{0\}$ ) est inversible pour le produit.

Le neutre additif de  $K$  est noté  $0$  (ou  $0_K$ ), et le neutre multiplicatif de  $K$  est noté  $1$  (ou  $1_K$ ). Tout  $x \in K$  possède un opposé noté  $-x$ , et tout  $x \in K^\times$  possède un inverse noté  $x^{-1}$ .

#### Exemples et remarques 1.1.2

- 1)  $(\mathbb{Q}, +, \times)$ ,  $(\mathbb{R}, +, \times)$  et  $(\mathbb{C}, +, \times)$  sont des corps, mais pas  $(\mathbb{Z}, +, \times)$ .
- 2) Un corps est un cas particulier d'anneau intègre ( $xy = 0$  implique  $x = 0$  ou  $y = 0$ ).

3) Si  $(K, +, \times)$  est un corps,  $(K^2, +, \times)$  n'est pas un corps (idem avec  $K^n$ , si  $n \geq 2$ ).

### **Théorème 1.1.3**

Si  $p$  est un nombre premier, l'anneau  $\mathbb{Z}/p\mathbb{Z}$  est un corps à  $p$  éléments.

## **1.2 Définition de sous-corps**

### **Définition 1.2.1**

Soit  $(K, +, \times)$  un corps. On dit qu'une partie  $L$  de  $K$  est un sous-corps de  $(K, +, \times)$  si :

- $L$  est un sous anneau de  $(K, +, \times)$ .
- $\forall x \in L$ , avec  $x \neq 0$ ,  $x^{-1} \in L$ .
- Muni des lois induites,  $(L, +, \times)$  possède alors lui-même une structure de corps.

### **Proposition 1.2.2**

$L$  est un sous-corps de  $(K, +, \times) \Leftrightarrow$  :

- $1 \in L$ .
- $\forall (x, y) \in L^2, x - y \in L$ .
- $\forall (x, y) \in L^2$ , avec  $y \neq 0$ ,  $xy^{-1} \in L$ .

### **Remarques et exemples 1.2.3**

- 1) Si  $L$  est un sous-corps de  $(K, +, \times)$ , on dit que  $K$  est une extension de  $(L, +, \times)$ .
- 2) Dans  $(\mathbb{Q}, +, \times)$ ,  $(\mathbb{R}, +, \times)$ ,  $(\mathbb{C}, +, \times)$ , chacun est un sous-corps du suivant.
- 3) Le seul sous-corps de  $(\mathbb{Q}, +, \times)$  est lui-même.
- 4) Toute intersection de sous-corps d'un corps  $K$  est un sous-corps de  $K$ . L'intersection de tous les sous-corps de  $K$  est donc, au sens de l'inclusion, le plus petit sous-corps de  $K$ .

## 1.3 Morphisme de corps

### Définition 1.3.1

Soient  $(K, +, \times)$  et  $(L, +, \times)$  deux corps. On dit qu'une application  $f$  de  $K$  dans  $L$  est un morphisme de corps si  $f$  est un morphisme de l'anneau  $(K, +, \times)$  dans l'anneau  $(L, +, \times)$ , c'est-à-dire si :

- $f(1_K) = 1_L$
- $\forall (x, y) \in K^2, f(x + y) = f(x) + f(y)$
- $\forall (x, y) \in K^2, f(xy) = f(x)f(y)$

Si de plus  $f$  est bijective, on dit que  $f$  est un isomorphisme de corps.

## 1.4 Corps finis

**Définition 1.4.1** Un corps fini est un corps dont le cardinal est fini.

Un corps fini de  $q$  éléments est noté  $F_q$  ou  $GF(q)$ .

### Théorème 1.4.2

Si  $F_q$  est un corps fini, le groupe multiplicatif  $F_q^\times$  est cyclique.

Il en résulte que si  $F_q$  est un corps à  $q$  éléments, le groupe cyclique  $F_q^\times$  est d'ordre  $(q - 1)$ .

**Théorème 1.4.3 (Wedderburn)** Tout corps fini est commutatif.

### Définition 1.4.4

Un élément générateur du groupe  $F_q^\times$  est appelé **élément primitif** de  $F_q$ . L'importance d'un élément primitif  $a \in F_q^\times$  tient au fait qu'on peut décrire tous les éléments du groupe  $F_q^\times$  en termes des  $(q - 1)$  premières puissances positives de  $a$ .

## 1.5 La caractéristique d'un corps

### Définition 1.5.1

Soit  $K$  un corps (pas forcément fini). On appelle sous-corps premier de  $K$  le plus petit sous-corps de  $K$  contenant 1.

On va chercher à décrire ces sous corps premiers. Pour cela, on pose l'homomorphisme d'anneaux  $\varphi : \mathbb{Z} \rightarrow K$  défini par  $\varphi(n) = n \cdot 1 = 1 + 1 + \dots + 1$ .

$\ker \varphi$  est alors un idéal de  $\mathbb{Z}$ , donc de la forme  $p\mathbb{Z}$  (L'entier  $p$  est le plus petit entier positif tel que  $p \cdot 1 = 0$ ). D'autre part, les théorèmes d'isomorphisme donnent  $\mathbb{Z}/p\mathbb{Z} \sim \varphi(\mathbb{Z})$  qui est inclus dans  $K$  donc intègre. Donc deux cas :  $p = 0$  ou  $p$  est un nombre premier.

### Définition 1.5.2

Le nombre  $p$  est appelé la caractéristique du corps  $K$ . Il est noté  $\text{car}(K)$ .

### Remarques 1.5.3

- Si  $\text{car}(K) > 0$  alors  $px = 0$  pour tout  $x \in K$ .
- Si  $K$  est fini on a  $p = \text{car}(K) > 0$  alors le sous-corps premier de  $K$  est  $\mathbb{Z}/p\mathbb{Z}$  que l'on note aussi  $F_p$ . Donc  $|K| = p^m$ . Le cardinal d'un corps fini est une puissance d'un nombre premier (par exemple il n'y a pas de corps à 6 éléments).
- Soient  $K$  un corps et  $x \in K$ ,  $n \in \mathbb{Z}$ . Si la caractéristique de  $K$  est 0,  $(nx = 0)$  équivaut à  $((n = 0) \text{ ou } (x = 0))$ . Si la caractéristique de  $K$  est un nombre premier  $p$ ,  $(nx = 0)$  équivaut à  $((p \mid n) \text{ ou } (x = 0))$ .

### Proposition 1.5.4

Soit  $F_q$  un corps fini de caractéristique  $p$ .

1.  $\forall (x, y) \in F_q^2, (x + y)^p = x^p + y^p$ .
2.  $\forall (x, y) \in F_q^2, \forall i \geq 2, (x + y)^{p^i} = x^{p^i} + y^{p^i}$ .



$$3. F_p = \{x \in F_q / x = x^p\}.$$

**Proposition 1.5.5 (Frobenius)**

Soit  $K$  un corps de caractéristique  $p > 0$ . L'application  $x \in K \mapsto x^p \in K$  est un homomorphisme de corps appelé homomorphisme de Frobenius.

**Théorème 1.5.6**

Soit  $F_q$  un corps fini à  $q$  éléments, de caractéristique  $p$ .

1. Si  $n$  est la dimension de l'espace vectoriel  $F_q$  sur  $F_p$ , on a  $q = p^n$ .
2. Tout  $x \in F_q^\times$  vérifie  $x^{q-1} = 1$ , ce qui implique  $x^{-1} = x^{q-2}$ .
3. Tout  $x \in F_q$  vérifie  $x^q = x$ .
4. Dans l'anneau  $F_q[X]$ , on a l'égalité

$$X^{q-1} - 1 = \prod_{a \in F_q^\times} (X - a)$$

5. Soit  $a$  un élément primitif de  $F_q$ . La famille

$$\mathcal{B} = \{1, a, a^2, \dots, a^{n-1}\}$$

est une base de l'espace vectoriel  $F_q$  sur  $F_p$ .

**Théorème 1.5.7**

Soit  $K$  un corps de caractéristique  $p$ , à  $q = p^n$  éléments.

- 1) Le nombre d'éléments de tout sous-corps de  $K$  est de la forme  $p^r$ , où  $r$  divise  $n$ .
- 2) Réciproquement, pour tout diviseur  $r$  de  $n$ ,  $K$  possède un unique sous-corps à  $p^r$  éléments, c'est l'ensemble des  $x \in K$  vérifiant  $x^{p^r} = x$ .

## 1.6 Construction des corps finis

### Théorème 1.6.1

Pour tout nombre premier  $p$  et tout entier positif  $n$ , il existe un polynôme irréductible de degré  $n$  dans l'anneau  $F_p[X]$ .

### Théorème 1.6.2 (Existence de corps finis)

Soit  $n$  un entier positif et soit  $p$  un nombre premier, il existe un corps à  $p^n$  éléments. Plus précisément, soit  $P \in F_p[X]$  un polynôme irréductible de degré  $n$  et soit  $K$  le corps  $F_p[X]/\langle P \rangle$ , on désigne par  $\alpha$  la classe d'équivalence du polynôme  $X$  dans  $K$ .

1. Le corps  $K$  est constitué des éléments de la forme  $R(\alpha)$ , où  $R$  décrit l'espace vectoriel  $F_p[X]^{(n)}$  des polynômes de degré  $\leq n-1$  de  $F_p[X]$ .
2. Si  $\beta \in K$ , il existe **un seul** polynôme  $R \in F_p[X]^{(n)}$  tel que  $\beta = R(\alpha)$ .
3. La famille  $\{1, \alpha, \alpha^2, \dots, \alpha^{n-1}\}$  est une base de l'espace vectoriel  $K$  sur  $F_p$ .

### Exemple d'un corps à 4 éléments 1.6.3

Soit  $P = X^2 + X + 1$  le seul polynôme irréductible de degré 2 de  $F_2[X]$ .

On sait que  $F_2[X]/\langle P \rangle$  est un corps à  $p^n = 2^2 = 4$  éléments et si  $\alpha$  désigne la classe d'équivalence du polynôme  $X$  dans  $F_2[X]/\langle P \rangle$ ,

alors

$$F_4 = F_2[X]/\langle P \rangle = \{a + b\alpha \mid a, b \in F_2\} = \{0, 1, \alpha, 1 + \alpha\}.$$

La table de multiplication de  $F_2[X]/\langle P \rangle$  s'écrit, compte tenu de l'égalité  $P(\alpha) = 0$ , c'est-à-dire  $\alpha^2 = 1 + \alpha$

	0	1	$\alpha$	$1 + \alpha$
0	0	0	0	0
1	0	1	$\alpha$	$1 + \alpha$
$\alpha$	0	$\alpha$	$1 + \alpha$	1
$1 + \alpha$	0	$1 + \alpha$	1	$\alpha$

On y voit par exemple que l'inverse de  $\alpha$  est  $(1 + \alpha)$ .

## 1.7 Calculs dans un corps fini - Table de logarithmes

Le théorème 1.5.6 fournit un moyen efficace pour effectuer des calculs dans un corps fini  $K$  de caractéristique  $p$ , si on connaît un élément primitif  $a$ .

Soit  $q = p^n$  le nombre d'éléments de  $K$ . On sait en effet que chaque élément  $x \in K^\times$  s'écrit de façon unique sous chacune des deux formes suivantes :

- (1)  $x = R(a)$ , où  $R \in F_p[X]^{(n)}$ . On sait additionner deux éléments donnés sous cette forme.
- (2)  $x = a^i$ , où  $i \in \{0, 1, 2, \dots, q - 2\}$ . La multiplication de deux éléments donnés sous cette forme s'effectue en additionnant les exposants modulo  $(q - 1)$ .

La question est de pouvoir passer de la première à la seconde forme et réciproquement, c'est-à-dire d'établir la table des logarithmes de base  $a$ .

Le plus instructif est de donner un exemple.

On sait que le polynôme  $P = X^2 + 1$  est irréductible dans  $F_3[X]$ .

Le corps  $K = F_3[X]/\langle P \rangle$  possède 9 éléments, le groupe  $K^\times$  est d'ordre 8.

La classe  $\alpha$  de  $X$  dans  $K$  vérifie  $P(\alpha) = \alpha^2 + 1 = 0$ , c'est-à-dire  $\alpha^2 = -1 = 2$ .

On voit que  $\alpha$  n'est pas un élément primitif puisque  $\alpha^4 = 1$ .

Par contre, l'élément  $a = \alpha + 2$  est primitif puisque  $a^2 = \alpha^2 + 4\alpha + 4 = 2 + \alpha + 1 = \alpha$ , et que  $a^4 = \alpha^2 = -1$ . Remarquons que  $a$  vérifie la relation  $a^2 = 1 + a$ , qu'on utilisera pour le calcul des puissances successives de  $a$  dans la base  $\{1, a\}$  de  $K$ .

La table de logarithmes de base  $a$  s'écrit :

$$\left\{ \begin{array}{l} a^0 = 1, \\ a^1 = a, \\ a^2 = 1 + a, \\ a^3 = 1 + 2a, \\ a^4 = 2, \\ a^5 = 2a, \\ a^6 = 2 + 2a, \\ a^7 = 2 + a, \end{array} \right. \quad \text{c'est-à-dire} \quad \left\{ \begin{array}{l} \log_a(1) = 0, \\ \log_a(a) = 1, \\ \log_a(1 + a) = 2, \\ \log_a(1 + 2a) = 3, \\ \log_a(2) = 4, \\ \log_a(2a) = 5, \\ \log_a(2 + 2a) = 6, \\ \log_a(2 + a) = 7. \end{array} \right.$$

Connaissant la table des logarithmes de base  $a$ , **multiplier** entre eux deux éléments donnés sous la forme (1) revient à effectuer une addition modulo 8. Par exemple, pour calculer le produit  $(2 + a)(2 + 2a)$ , on consulte la table de logarithmes, on y trouve  $2 + a = a^7$  et  $2 + 2a = a^6$ , d'où

$$(2 + a)(2 + 2a) = a^{7+6} = a^{13} = a^5 = 2a.$$

**Additionner** deux éléments donnés sous la forme (2) se fait de façon symétrique, par exemple

$$a^3 + a^2 = (1 + 2a) + (1 + a) = 2 = a^4.$$

**Inverser** un élément  $x \neq 0$  donné sous la forme (1) est aussi automatique, compte tenu de ce que sous la forme (2), on a  $(a^k)^{-1} = a^{8-k}$ . Par exemple

$$(1 + 2a)^{-1} = (a^3)^{-1} = a^{8-3} = a^5 = 2a.$$

Remarquons que dans un corps  $K$  à  $q$  éléments, on peut toujours, en l'absence de table de logarithmes, écrire que  $x^{-1} = x^{q-2}$  puisque  $x^{q-1} = 1$ .

Dans un corps de très grande taille, d'élément primitif  $a$ , il existe des méthodes rapides, étant donné un entier  $m \geq 2$ , pour calculer l'élément  $a^m$ . Mais il est beaucoup plus difficile, en l'absence de table de logarithmes, d'effectuer le calcul inverse : connaissant  $x \in K$  sous la forme (1), déterminer l'entier  $m = \log_a(x)$  tel que  $x = a^m$ . Ce problème est connu sous le nom de problème du logarithme discret (ce logarithme ne prenant que des valeurs entières).

Certains algorithmes de cryptographie sont basés sur la difficulté à résoudre le problème du logarithme discret.

# Chapitre 2

## La cryptographie à clé publique

Dans ce chapitre, on présente quelques notions sur la cryptographie et les domaines de son utilisation. On s'intéresse plus particulièrement à l'étude des systèmes de cryptage public RSA et EL GAMAL. Pour plus de détails sur le sujet on pourra consulter [9], [11].

**La cryptographie**, du grec "kryptos" qui signifie caché, et "graphein" écriture, est l'art de transformer un message pour tenter de le rendre illisible par toute autre personne que son destinataire. Alors la cryptographie est l'étude des méthodes permettant de transmettre des données de manière confidentielle.

Les données lisibles et compréhensibles sans intervention spécifique sont considérées comme un **texte clair**. La méthode permettant de dissimuler un texte clair en masquant son contenu est appelée **chiffrement** (dans le langage courant on parle plutôt de cryptage et de ses dérivés : crypter, décrypter). Le cryptage consiste à transformer un texte clair en caractère inintelligible appelé **texte chiffré** (ou cryptogramme).

Cette opération permet de s'assurer que seules les personnes auxquelles les informations sont destinées pourront y accéder. Le processus inverse de transformation du texte chiffré vers le texte d'origine est appelé **déchiffrement** (décryptage). **La clé** (clef) : donnée supplémentaire permettant de construire les fonctions de chiffrement et de déchiffrement. Sans connaissance de la clé de déchiffrement, le déchiffrement doit être impossible.

**La cryptanalyse** est l'étude des procédés crypto afin de trouver des faiblesses pour décrypter les messages chiffrés, le but est de retrouver le message clair sans connaître la clé.

L'objectif fondamental de la cryptographie est de permettre à deux personnes, habituellement désignées sous le nom Alice et Bob, de communiquer a travers un canal peu sûr de tel manière qu'un adversaire, Oscar, ne peut pas comprendre ce qui est transmit. Ce canal peut être une ligne téléphonique ou un réseau informatique, par exemple : Alice veut envoyer à Bob un message, qui peut être un texte en anglais, ou des données numériques, ou quelque autre chose. Alice chiffre le texte clair, en utilisant une clef de chiffage prédéterminée, et envoie le texte chiffré a travers le canal de communication. Oscar, en voyant le texte chiffré dans le canal par l'écoute clandestine, ne peut pas déterminer le sens du texte ; mais Bob, qui possède la clef de déchiffage, peut obtenir le texte clair.

## 2.1 Les principaux buts de la cryptographie

### **Confidentialité**

Le premier but de la cryptographie a été la confidentialité, c'est-à-dire l'échange entre entités de messages chiffrés, sans clé de déchiffrement, sont inintelligibles. La confidentialité est obtenue principalement a travers des schémas de chiffrement, qu'ils soient à clé secrète ou à clé publique.

### **Intégrité**

Un autre but de la cryptographie est l'intégrité. Ceci désigne le fait qu'une personne veuille s'assurer que le message reçu n'a pas été modifié par une tierce personne. L'intégrité est gérée a travers l'aide des schémas de signature à clé publique.

## Authentification

Un dernier but de la cryptographie est l'authentification. Il s'agit là pour une personne de prouver qu'un message a bien été envoyé par elle. L'authentification se gère en cryptographie principalement à travers des schémas de signature à clé publique. C'est-à-dire qu'une signature valide d'un message donné ne puisse être contestée par le signataire.

## 2.2 Où utilise-t-on la cryptographie ?

1. Internet (confidentialité, anonymat, authentification (s'agit-il bien de ma banque ?)).
2. Signature électronique (vérifiable, authentique, non-répudiation (je n'ai jamais signé ce texte ...)).
3. Vote électronique (le résultat reflète le vote, chaque vote est confidentiel, on ne peut pas connaître des résultats partiels, seuls les électeurs peuvent voter et une seule fois).
4. Paiement par carte bancaire (est-ce qu'il s'agit d'une vraie carte ? est-ce que le montant débité sera égal au montant crédité ? est-ce que le code secret est bien protégé ?).
5. Décodeurs (vérification de l'abonné, impossibilité de retransmettre les données décodées à une tierce personne, mise à jour de l'abonnement).
6. Porte monnaie électronique (pas de création de fausse monnaie, pas de création de faux porte-monnaie).
7. Bases de données sécurisées (ex : carte vitale. Seules les personnes habilitées ont accès à la vue partielle à laquelle elles ont droit, les données peuvent être échangées entre un médecin, un laboratoire, un hôpital, mise à jour possible des données).



## 2.3 Un système cryptographique

**Définition 2.3.1** Un système cryptographique est un cinq-uplet  $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ , tels que :

1.  $\mathcal{P}$  est un ensemble fini des textes clairs (plaintexts) ;
2.  $\mathcal{C}$  est un ensemble fini des textes chiffrés (ciphertexts) ;
3.  $\mathcal{K}$  est un ensemble fini des clés (keys) ;
4. Pour chaque clé  $k \in \mathcal{K}$ , il y a une règle de chiffrement  $e_k \in \mathcal{E}$  et une règle correspondante de déchiffrement  $d_k \in \mathcal{D}$ . Chaque  $e_k : \mathcal{P} \rightarrow \mathcal{C}$  et  $d_k : \mathcal{C} \rightarrow \mathcal{P}$  sont des fonctions tels que  $d_k(e_k(x)) = x$  pour chaque élément  $x \in \mathcal{P}$ .

## 2.4 La cryptographie à clefs privées

La cryptographie à clefs privées, appelée aussi cryptographie symétrique (conventionnel ou à clé secrète) est utilisée depuis plusieurs siècles. C'est l'approche la plus authentique du chiffrement de données et mathématiquement la moins problématique.

La clef servant à chiffrer les données peut être facilement déterminée si l'on connaît la clef servant à déchiffrer et vice-versa. Dans la plupart des systèmes symétriques, la clef de cryptage et la clef de décryptage sont une seule et une même clef.

### 2.4.1 Alphabets désordonnés

La manière la plus classique de chiffrer des messages consiste à remplacer une lettre par une autre, en utilisant un alphabet désordonné. C'est un chiffre monoalphabétique.

Par exemple, on pourrait utiliser la grille de chiffrement ci-dessous :

Clair	A	B	C	D	E	F	G	H	I	J	K	L	M
Chiffré	B	T	U	E	Q	V	Z	A	R	W	G	O	N
Clair	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Chiffré	C	L	K	J	S	X	D	M	H	P	I	F	Y

### Construction horizontale

Un inconvénient est qu'il est difficile, à moins d'avoir une mémoire remarquable, de se souvenir de la grille de chiffrement. Pour pouvoir la reconstituer rapidement, on peut utiliser comme moyen mémotechnique un mot-clef. Les lettres le composant seront mises dans la deuxième ligne de la grille dans l'ordre d'apparition, après avoir supprimé les doublons. On ajoutera ensuite les lettres n'apparaissant pas dans le mot-clef par ordre alphabétique. Exemple avec le mot-clef "sesame ouvre toi" :

Clair	A	B	C	D	E	F	G	H	I	J	K	L	M
Chiffré	S	E	A	M	O	U	V	R	T	I	B	C	D
Clair	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Chiffré	F	G	H	J	K	L	N	P	Q	W	X	Y	Z

### Construction verticale

Une deuxième méthode consiste à écrire la clef puis, en dessous, les autres lettres de l'alphabet, par ordre alphabétique. On lit ensuite les lettres colonne par colonne. Par exemple la clef MAISON donne la table suivante :

M	A	I	S	O	N
B	C	D	E	F	G
H	J	K	L	P	Q
R	T	U	V	W	X
Y	Z				

On obtient ainsi l'alphabet de chiffrement suivant :

Clair	A	B	C	D	E	F	G	H	I	J	K	L	M
Chiffré	M	B	H	R	Y	A	C	J	T	Z	I	D	K
Clair	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Chiffré	U	S	E	L	V	O	F	P	W	N	G	Q	X

### 2.4.2 Le Chiffre de César

Le Chiffre de César est un décalage vers la droite ou la gauche des lettres de l'alphabet.

Jules César effectuait un décalage de 3 rangs vers la gauche pour chiffrer ses messages.

Clair	A	B	C	D	E	F	G	H	I	J	K	L	M
Chiffré	D	E	F	G	H	I	J	K	L	M	N	O	P
Clair	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Chiffré	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Jusqu'au milieu des années 70, les seuls cryptosystèmes connus étaient symétriques ; car la clé de chiffrement était la même que la clé de déchiffrement, ce obligeait à garder secrète la clé de chiffrement aussi. Se pose alors le problème crucial de l'échange de clé, impossible à résoudre dans le cas d'un système développé à grande échelle.

En 1976, W. Diffie et M. Hellman introduisirent le concept de cryptographie à clé publique (ou asymétrique) : dans ce type de système, la clé de chiffrement est publique, c'est-à-dire connue de tous. Seule la clé de déchiffrement reste secrète.

## 2.5 Protocole d'échange de clés de Diffie-Hellman

Le problème ici est lié à la génération des clés. Il s'agit pour deux personnes d'obtenir une clé qu'ils seront les seuls à posséder, clé qui leur servira à chiffrer leur correspondance en utilisant un chiffrement à clé secrète.

Ce protocole est basé sur la difficulté à résoudre le problème du logarithme discret dans des corps de grande taille.

Le protocole d'échange de clés de Diffie-Hellman permet, à partir d'une clé publique, à deux personnes désirant communiquer secrètement, de se fabriquer une clé secrète.

Soit  $F_q$  un corps à  $q$  éléments dans lequel le problème du logarithme discret est difficile, et soit  $g$  un élément primitif de  $F_q$ , le couple  $(F_q, g)$  est public.

Voici le protocole que doivent suivre Alice et Bob pour se confectionner une clé secrète au vu et au su de tout le monde :

- Alice choisit un entier  $a$  vérifiant  $1 < a < q - 1$  et transmet sa clé publique  $g^a$  à Bob.
- Bob choisit un entier  $b$  vérifiant  $1 < b < q - 1$  et transmet sa clé publique  $g^b$  à Alice.
- Alice élève  $g^b$  à la puissance  $a$ , obtenant  $\gamma = g^{ab}$ .
- Bob élève  $g^a$  à la puissance  $b$ , obtenant  $\gamma = g^{ab}$ , qui sera leur clé secrète commune.

Alice et Bob sont les seuls à connaître  $\gamma$  car Oscar peut intercepter  $g^a$  et  $g^b$  mais ne peut en déduire  $\gamma = g^{ab}$  qu'en connaissant  $a$  ou  $b$ , c'est-à-dire en ayant résolu le problème du logarithme discret de base  $g$  dans  $F_q$ .

### **Vulnérabilité du protocole de Diffie-Hellman - Nécessité d'une signature**

Le protocole de Diffie-Hellman est vulnérable à l'attaque suivante : Oscar choisit un entier  $c$  tel que  $1 < c < q - 1$  et calcule  $g^c$ . Il intercepte la clé publique  $g^a$  envoyée par Alice à Bob, lui substitue la clé  $g^c$  qu'elle envoie à Bob, lequel croit recevoir la clé publique d'Alice. Lorsque Bob envoie à Alice sa clé publique  $g^b$ , Oscar l'intercepte et envoie  $g^c$  à Alice.

La clé "secrète" d'Alice est donc  $\alpha = g^{ac}$ , et celle de Bob est  $\beta = g^{bc}$ , Oscar les connaît puisqu'il lui suffit d'élever chacune des clés publiques d'Alice et Bob à la puissance  $c$ .

Oscar intercepte alors tous les messages d'Alice à Bob, est capable de les déchiffrer, de les modifier avec la clé  $\alpha$  puis de les envoyer à Bob en les chiffrant avec la clé  $\beta$ .

Il procède de même avec les messages envoyés par Bob à Alice, qu'il déchiffre avec la clé  $\beta$  et renvoie à Alice en les rechiffrant à l'aide de la clé  $\alpha$ .

La vulnérabilité du protocole de Diffie-Hellman réside dans le fait qu'il ne permet pas d'authentifier les participants.

### 2.5.1 La signature

La signature numérique est un mécanisme permettant d'authentifier l'auteur d'un document électronique et de garantir son intégrité, par analogie avec la signature manuscrite d'un document papier. Un mécanisme de signature numérique doit permettre au lecteur d'un document d'identifier la personne ou l'organisme qui a apposé sa signature.

Pour cela, les conditions suivantes doivent être réunies :

1. Infalsifiable : la signature ne peut pas être falsifiée. Personne ne peut se faire passer pour une autre.
2. Irrévocable : la personne qui a signé ne peut le nier.

La signature électronique n'est devenue possible qu'avec la cryptographie asymétrique.

## 2.6 Cryptographie à clefs publiques

L'idée de base des cryptosystèmes à clefs publiques a été proposée dans un article fondamental de Diffie et Hellman en 1976. Le principe fondamental est d'utiliser des clefs de chiffrement et déchiffrement différentes, non reconstituables l'une à partir de l'autre :

- Une clef publique : connue par tout le monde, utilisée généralement pour crypter ou vérifier la signature des messages.

- Une clef secrète (privée) : connue uniquement par le détenteur, utilisée pour décrypter et signer des messages.

Ce système est basé sur une fonction à sens unique, soit une fonction facile à calculer dans un sens mais très difficile à inverser sans la clef privée. Pour faire une explication imagée, la clef publique joue le rôle d'un cadenas. Imaginons que seul Bob possède la clef (clef secrète), Alice enferme son message dans une boîte à l'aide du cadenas et l'envoie à Bob. Personne n'est en mesure de lire le message puisque seul Bob possède la clef du cadenas.

Le gros avantage de ce système est qu'il n'y ait pas besoin d'avoir partagé un secret au préalable pour s'échanger des messages cryptés. En revanche les implémentations de tels systèmes (RSA, ElGamal, ...) ont un inconvénient majeur : leur lenteur par rapport à leurs homologues à clefs secrètes qui tournent eux jusqu'à près de mille fois plus vite.

### 2.6.1 Algorithme de chiffrement à clé publique de RSA

Imaginé par les trois chercheurs américains Ronald **Rivest**, Adi **Shamir** et Leonard **Adleman** en 1977, dans la foulée de la découverte de Diffie et Hellman, l'algorithme RSA, nommé d'après leurs initiales, a révolutionné le domaine de la cryptographie.

Le plus étonnant de l'affaire réside dans le fait que le protocole de Diffie-Hellman et l'algorithme RSA reposent sur des propriétés mathématiques connues depuis le 18<sup>ème</sup> siècle, voir depuis Fermat au 17<sup>ème</sup> siècle. Mais il est vrai que la puissance de calcul nécessaire à la mise en œuvre de ces algorithmes n'était pas disponible à cette époque.

L'algorithme RSA est un algorithme de cryptage qui permet aussi de signer un message, c'est-à-dire de rendre possible l'authentification de l'expéditeur. Il repose sur le fait qu'en l'état actuel du savoir et de la technique, on ne sait effectuer rapidement aucune des deux opérations suivantes :

- Décomposer un "grand" entier en facteurs premiers.

- Étant donné un grand entier  $n$  et un entier  $e$ , inverser la fonction

$$\psi_e : \mathbb{Z}/n\mathbb{Z} \longrightarrow \mathbb{Z}/n\mathbb{Z} \text{ définie par } \psi_e(x) = x^e,$$

c'est-à-dire retrouver  $x(\bmod n)$  à partir de  $x^e(\bmod n)$ , ceci dans les cas, bien entendu, où cette fonction est injective.

**Convention 2.6.1.1** Étant donné un entier positif  $n$  et deux entiers  $a$  et  $b$ , il nous arrivera d'écrire  $a = b(\bmod n)$  pour signifier que  $a \equiv b(\bmod n)$  et que  $0 \leq b < n$ .

Cela revient à dire que  $b$  est le reste de la division euclidienne de  $a$  par  $n$ .

L'algorithme RSA est basé sur le résultat arithmétique suivant, corollaire du théorème d'Euler, qui permet sous certaines conditions d'inverser rapidement la fonction  $\psi_e$  ci-dessus.

**Lemme 2.6.1.2** Soit  $p$  et  $q$  deux nombres premiers distincts, et soit  $n = pq$ . Pour tout entier positif  $t$  tel que  $t \equiv 1(\bmod \varphi(n))$ , on a

$$\forall a \in \mathbb{Z}, a^t \equiv a(\bmod n), \text{ c'est-à-dire } \forall x \in \mathbb{Z}/n\mathbb{Z}, x^t = x.$$

**Preuve 2.6.1.3** On sait que  $\varphi(n) = (p-1)(q-1)$ .

Posons  $t = 1 + k\varphi(n)$ , avec  $k \in \mathbb{N}$ , et soit  $a \in \mathbb{Z}$ . Trois cas sont possibles.

1.  $(\text{pgcd}(a, p) = 1)$  et  $(\text{pgcd}(a, q) = 1)$ , ce qui équivaut à  $(\text{pgcd}(a, n) = 1)$ .
2.  $(\text{pgcd}(a, p) = 1)$  et  $a$  est multiple de  $q$  (resp  $(\text{pgcd}(a, q) = 1)$  et  $a$  est multiple de  $p$ ).
3.  $a$  est multiple de  $pq = n$ .

- Dans le premier cas, on a  $a^{\varphi(n)} \equiv 1(\bmod n)$  d'après le théorème d'Euler, donc

$$a^t = a^{1+k\varphi(n)} = a \left( a^{\varphi(n)} \right)^k \equiv a(\bmod n).$$

- Dans le second cas,  $a^{p-1} = a^{\varphi(p)} \equiv 1 \pmod{p}$  d'après le théorème d'Euler, donc

$$a^t = a^{1+k(p-1)(q-1)} = aa^{k(p-1)(q-1)} = a \left( a^{(p-1)} \right)^{k(q-1)} \equiv a \pmod{p}. \quad (1)$$

De plus, comme  $a$  est multiple de  $q$ , on a

$$a^t \equiv a \equiv 0 \pmod{q}. \quad (2)$$

On déduit de (1) et (2) que  $a^t - a$  est multiple de  $p$  et  $q$  donc de  $n$ , c'est-à-dire  $a^t \equiv a \pmod{n}$ .

- Dans le troisième cas, on a  $a \equiv 0 \pmod{n}$  donc  $a^t \equiv a \equiv 0 \pmod{n}$ .

**Corollaire 2.6.1.4** Soit  $p$  et  $q$  deux nombres premiers distincts et soit  $n = pq$ . Soit  $e$  un entier positif premier avec  $\varphi(n)$  et soit  $d$  l'inverse de  $e$  modulo  $\varphi(n)$ .

1. L'application  $\psi_e$  définie sur  $\mathbb{Z}/n\mathbb{Z}$  par  $\psi_e(x) = x^e$  est une bijection de  $\mathbb{Z}/n\mathbb{Z}$  sur lui-même, et la bijection réciproque est l'application  $\psi_d$  définie par  $\psi_d(x) = x^d$ .
2. On en déduit que les applications  $\pi_e$  et  $\pi_d$  définies sur l'ensemble  $\{2, 3, \dots, n-1\}$  par

$$\pi_e(a) = a^e \pmod{n} \quad \text{et} \quad \pi_d(a) = a^d \pmod{n}$$

sont des bijections réciproques de  $\{2, 3, \dots, n-1\}$ .

**Preuve 2.6.1.5** Pour tout  $x \in \mathbb{Z}/n\mathbb{Z}$ ,  $\psi_d \circ \psi_e(x) = \psi_e \circ \psi_d(x) = x^{ed} = x$  puisque  $ed \equiv 1 \pmod{\varphi(n)}$ . Pour le point 2, on remarque que  $\psi_d(0) = \psi_e(0) = 0$  et  $\psi_d(1) = \psi_e(1) = 1$ .

Chaque élément d'information à transmettre : chiffre, lettre, etc..., est représenté par un entier appartenant à un ensemble  $\{2, 3, \dots, C\}$ , où  $C \in \mathbb{N}$ , c'est ce qu'on appelle un **encodage**, cet encodage est connu de tous. Chaque utilisateur du cryptosystème RSA procède comme suit :

- Il choisit deux "grands" nombres premiers  $p$  et  $q$ . Il calcule le produit  $n = pq$  et l'indicatrice d'Euler  $\varphi(n) = (p-1)(q-1)$ . Remarquons que l'on a toujours  $n > C$ .
- Il choisit un entier  $d > 1$  **premier** avec  $\varphi(n)$  et calcule son inverse  $e$  modulo  $\varphi(n)$ .



- Il publie le couple  $(e, n)$ , qui est donc appelé sa **clé publique**, et conserve le couple  $(d, \varphi(n))$  qui est sa **clé secrète**.

Remarquons que, connaissant la clé publique  $(e, n)$ , quiconque voudrait reconstituer la clé secrète  $(d, \varphi(n))$  devrait calculer  $\varphi(n) = (p-1)(q-1)$ , donc connaître  $p$  et  $q$ , c'est-à-dire la décomposition de  $n$  en facteurs premiers.

- **Encryptage** : Si Alice veut envoyer le message secret  $m \in \{2, 3, \dots, C\}$  à Bob, elle prend connaissance de la clé publique  $(e, n)$  de ce dernier (dont la clé secrète est  $(d, \varphi(n))$ ), puis calcule l'entier  $M = m^e \pmod{n}$ . C'est le message crypté, qu'elle envoie à Bob.
- **Décryptage** : Bob calcule  $M^d \pmod{n}$  et récupère  $m$  puisque

$$M^d \equiv m^{ed} \pmod{n} = m \pmod{n}.$$

- **Signature** : L'algorithme RSA permet en outre à Alice de **signer** un message  $m$  de façon à ce que Bob soit certain que c'est bien elle qui l'a envoyé. Si  $(e_1, n_1)$  est sa clé publique et  $(d_1, \varphi(n_1))$  sa clé secrète, Alice calcule l'entier  $s = m^{d_1} \pmod{n_1}$  et fait parvenir à Bob le couple  $(m, s)$  qui est le **message signé**.
- **Vérification** : Recevant  $(m, s)$ , Bob calcule  $s^{e_1} \pmod{n_1}$ . Si  $s^{e_1} = m$ , il est certain que le message vient d'Alice puisque seule Alice connaît l'inverse  $d_1$  de  $e_1$  modulo  $\varphi(n_1)$ , c'est-à-dire la bijection réciproque  $\pi_{d_1}$  de la bijection  $\pi_{e_1}$  de l'ensemble  $\{2, 3, \dots, n_1 - 1\}$  dans lui-même.
- **Commentaire** : Dans notre cas, le message signé par RSA n'est pas secret, mais il est authentifié.

L'algorithme RSA est dit **asymétrique** au sens où

- pour chiffrer, Alice utilise la clé publique du destinataire Bob,
- pour déchiffrer, Bob utilise sa propre clé secrète.

**Exemple 2.6.1.6** Soit  $n = 7 \times 11 = 77$ . Alors  $\varphi(77) = 6 \times 10 = 60$ . Bob choisit  $e = 13$ , il utilise l'algorithme d'Euclide étendu pour calculer  $d$ , et obtient  $1 = 13 \times 37 + 60 \times (-8)$ , c'est-à-dire  $d = 37$ . Sa clé publique est donc  $(77, 13)$  et sa clé privée  $(60, 37)$ .

Pour envoyer à Bob le message  $m = 9$ , Alice calcule  $M = 9^{13} \pmod{77}$  en utilisant **un algorithme de calcul rapide des puissances** basé sur l'écriture en binaire de 13,

$$13 = 2^3 + 2^2 + 1 = 1101.$$

Pour calculer  $x^{13} = x \times x^4 \times x^8$ , il lui suffit de calculer successivement

$$x \longrightarrow x^2 \longrightarrow x^4 \longrightarrow x^8 \longrightarrow x^{12} = x^4 \times x^8 \longrightarrow x^{13} = x \times x^{12},$$

ce qui ne fait que 5 multiplications au lieu de 12. Pour cet algorithme de calcul rapide des puissances, le nombre des multiplications nécessaires pour calculer  $a^n$  est majoré par  $2 \log_2 n$ . Chacune des opérations est effectuée modulo 77, ce qui évite les trop grands nombres.

$$9^2 = 81 = 4 \pmod{77},$$

$$9^4 \equiv 4^2 = 16 \pmod{77},$$

$$9^8 \equiv 16^2 = 256 = 25 \pmod{77},$$

$$9^{12} = 9^8 \times 9^4 \equiv 25 \times 16 = 400 = 15 \pmod{77},$$

$$9^{13} = 9^{12} \times 9 \equiv 15 \times 9 = 135 = 58 \pmod{77}.$$

Recevant  $M = 58$ , Bob calcule  $58^{37} \pmod{77}$  de la façon suivante

$$\begin{aligned}
58^2 &= 3364 = 53(\bmod 77), \\
58^4 &\equiv 53^2 = 2809 = 37(\bmod 77), \\
58^8 &\equiv 37^2 = 1369 = 60(\bmod 77), \\
58^{16} &\equiv 60^2 = 3600 = 58(\bmod 77), \\
58^{32} &\equiv 58^2 = 3364 = 53(\bmod 77), \\
58^{36} &= 58^{32} \times 58^4 \equiv 53 \times 37 = 1961 = 36(\bmod 77), \\
58^{37} &= 58^{36} \times 58 \equiv 36 \times 58 = 2088 = 9(\bmod 77),
\end{aligned}$$

et récupère  $m = 9$  (en effectuant 7 multiplications au lieu de 36).

### 2.6.2 Algorithme de chiffrement à clé publique d'El Gamal

C'est un exemple de chiffrement à clé publique basé sur le problème du logarithme discret.

Soit  $F_q$  un corps à  $q$  éléments et soit  $g$  un élément primitif de  $F_q$ . On suppose que le problème du logarithme discret de base  $g$  dans  $F_q$  est difficile. Le **couple**  $(F_q, g)$  **est public**. Rappelons que pour tout  $x \in F_q$ , on a  $x^{-1} = x^{q-2}$  puisque  $x^{q-1} = 1$ .

L'encodage consiste à assigner à chaque élément de message un élément de  $F_q^*$  de façon injective.

- **Clé secrète, clé publique** : Si Bob veut permettre à un tiers de lui envoyer des messages secrets, il choisit un entier  $a$  tel que  $1 < a < q$ , qui sera sa **clé secrète**. Il calcule  $\alpha = g^a$ , qu'il publie, et qui sera sa **clé publique**.
- **Encryptage** : Pour envoyer à Bob le message  $m \in F_q$ , Alice **choisit aléatoirement** un entier  $x$  tel que  $1 < x < q$ , et lui transmet le couple

$$(\beta, \gamma) = (g^x, m\alpha^x) = (g^x, mg^{ax})$$

qui est le message crypté ou chiffré.

- **Décryptage** : Bob, grâce à sa clé secrète  $a$ , peut calculer l'inverse  $\delta = g^{-ax}$  de  $g^{ax}$ , en effet

$$\delta = g^{-ax} = g^{ax(q-2)} = (g^x)^{a(q-2)} = \beta^{a(q-2)},$$

il en déduit  $m = (mg^{ax})g^{-ax} = \gamma\delta$ .

Comme l'algorithme **RSA**, cet algorithme est asymétrique.

### Signature d'El Gamal

On considère un corps premier  $F_p$  et un élément primitif  $g$  de  $F_p$ . On identifie chaque élément de  $F_p$  à un entier compris entre 0 et  $p-1$ . Les entiers  $p$  et  $g$  sont choisis de telle sorte que le problème du logarithme discret de base  $g$  soit difficile à résoudre dans  $F_p$ . Le couple  $(F_p, g)$  est public.

- **Signature** : Soit  $a_1$  la clé secrète d'Alice et  $\alpha_1 = g^{a_1}$  sa clé publique. Pour signer le message  $m$ , elle procède comme suit :

- Elle **choisit aléatoirement** un entier positif  $e$  premier avec  $p-1$  et calcule son inverse  $d$  modulo  $p-1$ .
- Elle calcule l'entier  $s_1 = g^e \pmod{p}$  et l'entier  $s_2 = d(m - a_1 s_1)$ , c'est-à-dire tel que

$$m = es_2 + a_1 s_1 \pmod{p-1}; \quad (1)$$

- Le message signé est le triplet  $(m, s_1, s_2)$ .
- **Vérification** : Recevant le triplet  $(m, s_1, s_2)$ , Bob calcule  $s_1^{s_2} \alpha_1^{s_1}$  et vérifie l'égalité suivante dans  $F_p$

$$g^m = \alpha_1^{s_1} s_1^{s_2}, \quad (2)$$

qui résulte de l'égalité (1) puisque  $g^m = g^{es_2 + a_1 s_1} = \alpha_1^{s_2} \alpha_1^{s_1}$ .

- **Justification :** Étant donné que Bob connaît la clé publique  $\alpha_1$  d'Alice, pour authentifier son message, celle-ci doit prouver à Bob qu'elle connaît l'entier  $a_1 = \log_g(\alpha_1)$  qui est sa clé secrète. La façon la plus simple serait de lui communiquer  $a_1$ , ce qui constituerait une signature mais obligerait Alice à changer de clé, celle-ci n'étant plus secrète.

C'est ce qui se passerait si elle avait choisi  $e = d = 1$ , puisqu'à ce moment là, on aurait  $s_1 = g$  et  $s_2 = m - a_1$ , d'où on déduit immédiatement  $a_1$ . L'entier  $e$  est donc une clé secrète destinée à protéger la clé secrète  $a_1$ . Connaissant  $s_1 = g^e$ , Oscar ne peut en déduire  $e$ , donc  $d$ .

Connaissant  $s_2 = d(m - a_1 s_1)$ , elle ne peut donc en déduire  $a_1$ .

Mais comment l'égalité (2) peut-elle prouver à Bob qu'Alice connaît  $a_1$  ?

Remarquons que (2) équivaut à l'égalité (1) des logarithmes de base  $g$  :

$$m = es_2 + a_1 s_1 \pmod{p-1}.$$

Il en résulte qu'Alice, connaissant  $s_1$ ,  $s_2$ ,  $e$  et  $m$ , connaît nécessairement  $a_1$ .

Oscar, ne connaissant que  $s_1$  sans connaître  $e$ , et ignorant  $a_1$ , ne peut pas déterminer d'entier  $s_2$  vérifiant (1), même si elle connaît  $m$  ou  $m - a_1 s_1$ .

# Chapitre 3

## Étude sur la complexité des algorithmes R.S.A et EL GAMAL

Dans ce chapitre, on donne quelques notions sur la complexité des algorithmes. On s'intéresse plus particulièrement à la complexité des algorithmes RSA et EL GAMAL en se basant sur l'étude faite dans [5] et [9].

### 3.1 La complexité

La théorie de la complexité est un domaine des mathématiques, et plus précisément de l'informatique théorique, qui étudie formellement la quantité de ressources (en temps et en espace) nécessaire pour la résolution de problème au moyen de l'exécution d'un algorithme. Il s'agit donc d'étudier la difficulté intrinsèque de problèmes mathématiques.

### 3.2 L'algorithme

Un algorithme est une méthode pour résoudre un problème donné. La complexité d'un algorithme se mesure par son coût en temps de calcul, mémoire utilisée ou toute

autre unité significative, lors de son utilisation pour résoudre ce problème.

### 3.3 Complexité en temps et en espace

Les complexités qui peuvent être évaluées :

- **Complexité en temps** : il s'agit de savoir combien de temps prendra l'exécution d'un algorithme.
- **Complexité en espace** : il s'agit de savoir combien d'espace mémoire occupera l'exécution de l'algorithme.

Généralement, on s'intéresse essentiellement à **la complexité en temps**, cette complexité n'est pas la même selon les déroulements du traitement :

1. **Complexité au pire** : complexité maximum, dans le cas le plus défavorable.
2. **Complexité en moyenne** : il s'agit de la moyenne des complexités obtenues selon les issues du traitement.
3. **Complexité au mieux** : complexité minimum, dans le cas le plus favorable. En pratique, cette complexité n'est pas très utile.

Le plus souvent, on utilise la complexité au pire, car on veut borner le temps d'exécution.

### 3.4 Ordres de grandeurs

#### Définitions 3.4.1

Soient  $f$  et  $g$  deux fonctions de  $\mathbb{N}$  dans  $\mathbb{R}^+$ ,

1.  $f = O(g)$  si et seulement si il existe  $c$  dans  $\mathbb{R}^+$ , il existe  $n_0$  dans  $\mathbb{N}$  tel que quelque soit  $n > n_0$ ,  $f(n) \leq c.g(n)$ . Ainsi  $f = O(g)$  veut dire que  $f$  est dominée asymptotiquement par  $g$ . Par exemple :  $2n = O(n^2)$ , mais aussi  $2n = O(n)$ , il suffit de bien choisir la constante  $n_0$ .

Remarque : lorsqu'une complexité est donnée en termes de  $O(g)$ , il convient de donner la fonction  $g$  la plus «serrée» possible.

2.  $f = \Omega(g)$  si et seulement si il existe  $d$  dans  $\mathbb{R}^+$ , il existe  $n_0$  dans  $\mathbb{N}$  tel que quelque soit  $n > n_0$ ,  $d.g(n) \leq f(n)$ .
3.  $f = \Theta(g)$  si et seulement si  $f = O(g)$  et  $g = O(f)$  ou encore  $f = O(g)$  et  $f = \Omega(g)$ .  
C'est-à-dire qu'il existe  $c$  et  $d$  dans  $\mathbb{R}^+$ , il existe  $n_0$  dans  $\mathbb{N}$  tel que quelque soit  $n > n_0$ ,  $d.g(n) \leq f(n) \leq c.g(n)$ .

On dit que  $f$  et  $g$  sont de même ordre de grandeur asymptotique.

En d'autres termes,  $O()$  est une borne asymptotique supérieure,  $\Omega()$  est la borne asymptotique inférieure et  $\Theta()$  la borne asymptotique «exacte», cette dernière est beaucoup plus précise que les deux premières.

### Exemples 3.4.2

$$f(n) = \frac{n^3}{2} \quad g_1(n) = n^2 + 17n + 15 \quad g_2(n) = 5n^3$$

- $g_1 \in O(f)$  : on prend  $n_0 = 1$  et  $c = 2(1 + 17 + 15)$  et alors  $n^2 + 17n + 15 \leq c \cdot \frac{n^3}{2}$  si  $n \geq 1$ .
- $g_2 \in O(f)$  : on choisit  $c = 10$  et alors  $5n^3 \leq 10 \cdot \frac{n^3}{2}$ .
- $f \in O(g_2)$ .

### 3.4.1 Types de complexité algorithmique

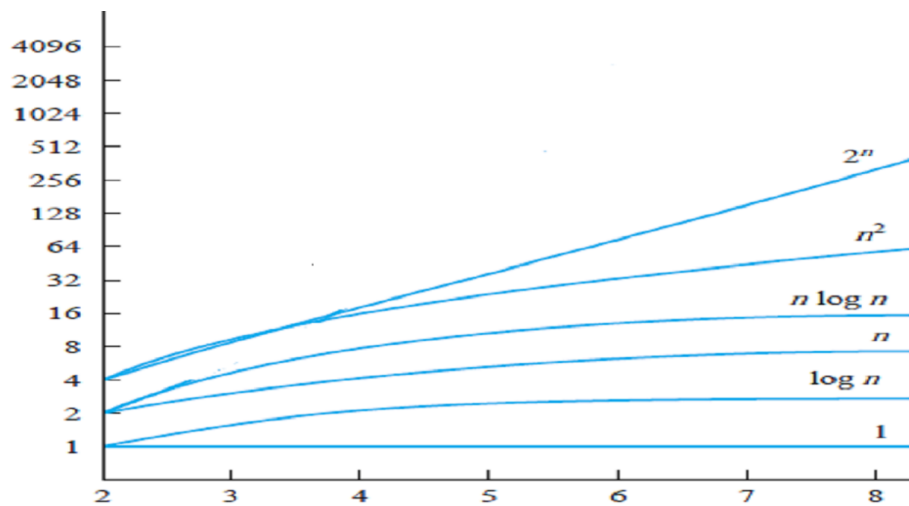
1. **Temps constant**,  $T(n) = O(1)$  : temps d'exécution indépendant de la taille des données à traiter.
2. **Temps logarithmique**,  $T(n) = O(\log(n))$  : on rencontre généralement une telle complexité lorsque l'algorithme casse un gros problème en plusieurs petits, de sorte que la résolution d'un seul de ces problèmes conduit à la solution du problème initial.



3. **Temps linéaire**,  $T(n) = O(n)$  : cette complexité est généralement obtenue lorsqu'un travail en temps constant est effectué sur chaque donnée en entrée.
4. **Temps quasi-linéaire**,  $T(n) = O(n \cdot \log(n))$  : l'algorithme scinde le problème en plusieurs sous-problèmes plus petits qui sont résolus de manière indépendante. La résolution de l'ensemble de ces problèmes plus petits apporte la solution du problème initial.
5. **Temps quadratique**,  $T(n) = O(n^2)$  : apparaît notamment lorsque l'algorithme envisage toutes les paires de données parmi les  $n$  entrées (ex. deux boucles imbriquées). Remarque :  $O(n^3)$  **temps cubique**,  $O(n^k)$  ,pour  $k > 3$ , **temps polynomiale**.
6. **Temps exponentiel**,  $T(n) = O(2^n)$  : souvent le résultat de recherche brutale d'une solution.

### 3.4.2 Ordres de grandeurs

$$O(1) \subset O(\log n) \subset O(n) \subset O(n \cdot \log n) \subset O(n^2) \subset O(n^3) \subset O(2^n)$$



### 3.5 Complexité du système cryptographique du RSA

Le chiffrement et le déchiffrement RSA nécessitent tous deux une exponentiation modulo  $n$ . Dans le cas général où les exposants  $d$  et  $e$  sont choisis aléatoirement, leur taille est donc celle de  $\varphi(n)$ , soit celle de  $n$  puisque  $\varphi(n) = (p-1)(q-1)$ . Avec les algorithmes usuels d'exponentiation, le chiffrement et le déchiffrement coûtent donc tous deux  $O(\log n)$  multiplications modulo  $n$ . Comme ces multiplications peuvent s'effectuer en temps  $O(\log^2 n)$ , on obtient une complexité de chiffrement et de déchiffrement de  $O(\log^3 n)$ .

Cette complexité, bien que polynomiale, n'est pas négligeable, notamment pour des environnements à puissance de calcul réduite. Aussi il est tentant de choisir soit un petit  $e$ , soit un petit  $d$ , de façon à accélérer le chiffrement ou le déchiffrement. Si  $d$  n'est pas petit, on peut légèrement accélérer le déchiffrement à l'aide des restes chinois. En effet, pour calculer  $m^d \bmod n$ , il suffit de calculer  $m^d \bmod p$  et  $m^d \bmod q$ , puis d'appliquer les restes chinois dont le coût est négligeable par rapport au calcul de  $m^d \bmod n$  si  $d$  est grand. Or  $m^d \equiv m^{d \bmod (p-1)} \pmod{p}$  et  $m^d \equiv m^{d \bmod (q-1)} \pmod{q}$ , de sorte que si l'on conserve les valeurs de  $p$  et  $q$ , on peut ramener le calcul de  $m^d \bmod n$  à deux exponentielles d'exposant de taille divisée par deux, avec un module de taille également divisée par deux. Comme la complexité d'une exponentielle est de  $O(\log^3 n)$ , on divise ainsi approximativement le temps de calcul global par quatre.

### 3.6 Complexité du système cryptographique d'El Gamal

Un petit exemple illustrera les calculs exécutés dans le système cryptographique d'El Gamal.

**L'exemple 3.6.1** on suppose que  $q = 2579$  et  $g = 2$ .

$g$  est un élément primitif modulo  $q$ . Soit  $a = 765$ ,  
alors

$$\alpha = 2^{765} \bmod 2579 = 949.$$

Maintenant, supposons qu'Alice souhaite envoyer le message  $m = 1299$  à Bob. Elle choisit aléatoirement l'entier  $x = 853$ . et calcule

$$\begin{aligned}\beta &= 2^{853} \bmod 2579 = 435, \\ \text{et } \gamma &= 1299 \times 949^{853} \bmod 2579 = 2396.\end{aligned}$$

Quand Bob reçoit le message chiffré  $(\beta, \gamma) = (435, 2396)$ , il calcule

$$m = 2396 \times (435^{765})^{-1} \bmod 2579 = 1299,$$

et c'est le message initial qu'Alice avait chiffré.

Il est clair que le système cryptographique d'El Gamal sera peu sûr si Oscar peut calculer la valeur  $a = \log_g \alpha$ , c.à.d. il peut déchiffrer des textes chiffrés exactement comme le fait Bob.

Par conséquent, une condition nécessaire pour la sécurité du système cryptographique d'El Gamal est que le problème du logarithme discret est infaisable. Ceci est généralement considéré comme être le cas si  $q$  est soigneusement choisi et  $g$  est un élément primitif modulo  $q$ . En particulier, là n'est aucun algorithme connu de temps polynomial pour cette version du problème du logarithme discret. Pour contrecarrer des attaques connues,  $q$  devrait avoir au moins 300 chiffres, et  $q - 1$  devrait avoir au moins un «grand» facteur premier.

### **Algorithmes pour le problème du logarithme discret**

Nous supposons que  $g$  est un élément primitif de  $F_q^*$  d'ordre  $q - 1$ . Par conséquent le problème du logarithme discret peut être exprimé sous la forme suivante : si on donne  $\alpha \in F_q$ , on trouve l'exposant unique  $a$ ,  $0 \leq a \leq q - 2$ , tel que  $g^a = \alpha$ .

Nous commençons en analysant quelques algorithmes élémentaires qui peuvent être employés pour résoudre le problème du logarithme discret. Dans nos analyses, nous supposons que le calcul d'un produit de deux éléments dans  $F_q$  exige constante temps (c.à.d.  $O(1)$ ).

D'abord, nous observons que le problème du logarithme discret peut être résolu par recherche approfondie dans les  $O(q)$  temps et  $O(1)$  espaces, simplement par le calcul  $g, g^2, g^3, \dots$ , jusqu'à  $\alpha = g^a$  est trouvé. (Chaque  $g^i$  est calculé en multipliant le précédent  $g^{i-1}$  par  $g$ , et par conséquent tout le temps requis est  $O(q)$ ).

Une autre approche est au precompute toutes les valeurs possibles  $g^i$ , et assortir alors la liste des paires commandées  $(i, g^i)$  en ce qui concerne leurs deuxièmes coordonnées. Puis, donné  $\alpha$ , nous pouvons effectuer une recherche dichotomique de la liste assortie afin de trouver la valeur  $a$  tel que  $g^a = \alpha$ . Ceci a besoin de le temps  $O(q)$  de precomputation de calculer les puissances  $q$  de  $g$ , et chronomètrant  $O(q \log q)$  pour assortir la liste de taille  $q$ . (l'étape de tri peut être faite à temps  $O(q \log q)$  si un algorithme de tri efficace est employé). Si nous négligeons des facteurs logarithmiques, comme est habituellement fait dans l'analyse de ces algorithmes, le temps de precomputation est  $O(q)$ . Le moment pour une recherche dichotomique d'une liste assortie de la taille  $q$  est  $O(\log q)$ . Si nous (encore) ignorons la limite logarithmique, alors nous voyons que nous pouvons résoudre le problème du logarithme discret en  $O(1)$  temps avec  $O(q)$  precomputation et  $O(q)$  mémoire.

En fin, on peut conclure que le déchiffrement d'EL Gamal, comme le déchiffrement du RSA, demande une exponentiation modulaire, la différence entre ces deux systèmes c'est que le chiffrement d'EL Gamal demande deux exponentiations modulaires : le calcul de  $\alpha^x \bmod q$  et celui de  $\beta = g^x \bmod q$ . Le chiffrement du RSA ne demande qu'une seule exponentiation modulaire. Mais les exponentiations pour le chiffrement d'EL Gamal sont indépendantes des messages en clair qui sont en train d'être chiffrés. Par conséquent, ces exponentiations peuvent être faites séparément, comme des pré-calculs. De la sorte, le

chiffrement ne demande plus qu'une seule multiplication modulaire et devient beaucoup plus efficace que le chiffrement du RSA. Cependant, les valeurs pré-calculées doivent être gardées secrètes, et doivent être stockées en lieu sûr, sur une carte à puce, par exemple.

# Conclusion

Nous avons présenté dans ce travail une étude sur les algorithmes de cryptage public et leurs complexités, ces algorithmes sont basés sur des problèmes mathématiques difficiles à résoudre. L'algorithme RSA est basée sur la difficulté de factorisation d'un grand nombre entier en facteurs premiers, et l'algorithme EL GAMAL est basée sur la difficulté du problème du logarithme discret dans un corps fini.

Si un jour à venir, on découvre des méthodes rapides (de complexité au plus polynomiale) pour :

- la factorisation d'un grand nombre entier  $n$  en facteurs premiers.

Ou bien

- Le calcul de logarithme discret dans un corps fini.

Les algorithmes RSA et EL GAMAL seront à ce moment inefficace pour le cryptage. Il faut donc penser à construire d'autres algorithmes de cryptage non seulement basés sur la complexité exponentielle d'autre problème.

# Bibliographie

- [1] BENOIT CHEVALLIER-MAMES : Cryptographie à clé publique : Constructions et preuves de sécurité, Thèse Doctorat, Paris, 2006.
- [2] C.ANTONINI, J.F.QUINT, P.BORGNAT, J.BÉRARD, E.LEBEAU, E.SOUCHE, A.CHATEAU, O.TEYTAUD : *Les Mathématiques pour l'Agrégation (Mai 2002)*
- [3] HERBERT S.WILF : *Algorithmes et complexité, Masson, Paris(1989)*
- [4] J.QUERRÉ : *Cours D'Algèbre, Masson, Paris (1976)*
- [5] J. STERN, L. GRANBOULAN, P. NGUYEN, D. POINTCHEVAL, *Conception et preuves d'algorithmes cryptographiques*, Cours de magistère M.M.F.A.I. École normale supérieure.
- [6] JEAN-MICHEL FERRARD : *Cours de Mathématiques, Groupe, Anneaux, Corps, Arithmétique.*
- [7] JEAN-PIERRE RAMIS ET ANDRE WARUSFEL : *Mathématiques, Tout-en-un pour la Licence, Dunod, Paris (2006)*
- [8] JOHANNES BUCHMANN : *INTRODUCTION À LA CRYPTOGRAPHIE, Cours et exercices corrigés, Dunod, Paris (2006)*
- [9] KENNETH.H.ROSEN : *CRYPTOGRAPHY, Theory and practice, third edition (2006)*
- [10] NICOLAS BRUYÈRE : *ÉLÉMENTS DE THÉORIE DES CORPS FINIS, Université de Rouen, Agrégation de Mathématique (2005-2006)*

- [11] PIERRE WASSEF : *ARITHMETIQUE, Application aux CODES CORRECTEURS et à la CRYPTOGRAPHIE*, Vuibert, France(Novembre 2009).
- [12] SITE D'INTERNET : <http://www.univ-biskra.dz/fac/fsesnv/cours/algo/Cours2.pdf>
- [13] SITE D'INTERNET : <http://www.e-campus.uvsq.fr/claroline/backends/download.php?url=L2NoYXA2LUNvbXBsZXhpdOkucGRm&cidReset=true&cidReq=IN100>
- [14] SITE D'INTERNET : <http://www.u-picardie.fr/~furst/docs/4-Complexite.pdf>