ÉTUDE MATHÉMATIQUE

ÉLÉMENTS DE CRYPTOGRAPHIE ARITHMÉTIQUE

Par Alain SATABIN, professeur honoraire

Sommaire de l'article :

1. CHIFFREMENT À CLÉ RÉVÉLÉE

Nous examinons ici quelques procédés de chiffrement utilisant un calcul arithmétique. Les textes sont supposés ne contenir que des lettres non accentuées (aucun chiffre, ni ponctuation, ni espace). Chaque lettre est identifiée à un nombre compris entre 0 et 25 (du A au Z) :

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Les procédés peuvent évidemment s'étendre à un jeu de caractères plus complet en adaptant les fonctions arithmétiques utilisées.

1.1. Chiffre de César (– 101 / – 44)

Il s'agit là d'un simple glissement de l'alphabet. Chaque lettre est remplacée par sa translatée d'une certaine constante dans l'alphabet bouclé (du Z, on revient au A).

La fonction arithmétique correspondante est donc du type $n \longrightarrow n+k\ [26]$. Le déchiffrement consiste évidemment à appliquer l'opération $n \longrightarrow n\text{-}k\ [26]$.

Par exemple, avec $k = 15$, le texte **BONJOUR** se chiffre **QDCYDJB** ; et avec $k = 7$, **ZHSBA** se déchiffre **SALUT**.

Ce type de fonction peut être affine du type $n \longrightarrow pxq\ [26]$ où p est premier avec 26 afin de posséder un inverse p' dans z / 26z. Sa réciproque, permettant le déchiffrement, est alors $n \longrightarrow p'x(n\text{-}q)\ [26]$.

Par exemple, avec $k = 15$, le texte **BONJOUR** se chiffre **QDCYDJB** ; et avec $k = 7$, **ZHSBA** se déchiffre **SALUT**.

Pour sa part, César utilisait une constante égale à 3 et l'alphabet latin ne comportait que 20 lettres. Le fait qu'il n'existe que 25 chiffres de César rend ce procédé extrêmement vulnérable !

1.2. Chiffre de substitution

Dans ce procédé, chaque lettre est remplacée par une autre lettre. Cette permutation de l'alphabet constitue la clé de chiffrement et doit évidemment être convenue entre émetteur et récepteur. Ce mélange peut provenir tout simplement d'une succession des 26 lettres mélangées, ou d'un mélange issu d'un mot clé, ou encore d'une fonction arithmétique bijective de $\mathbb{Z}$ /n $\mathbb{Z}$ dans lui-même.

Ce type de fonction peut être affine du type $n \longrightarrow pxq\ [26]$ où p est premier avec 26 afin de posséder un inverse p' dans z / 26z. Sa réciproque, permettant le déchiffrement, est alors $n \longrightarrow p'x(n\text{-}q)\ [26]$.

En prenant par exemple $p = 5$ et $q = 17$ (donc $p' = 21$), la correspondance mono-alphabétique est :

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

et **BONJOUR** se chiffre **WJEKJNY**, alors que **DRUNI** se déchiffre **SALUT**.

retour au sommaire

1.3. Chiffre de Vigénère (1523 - 1596)

Les chiffres de substitution mono-alphabétique ont l'inconvénient de toujours remplacer une même lettre par un même lettre, ce qui les rend vulnérables et facilement cassables par l'analyse des fréquences des lettres du texte chiffré.
Vigénère pallia cet inconvénient en introduisant un mot clé qui, écrit de façon cyclique sous le texte clair, donne, pour chaque lettre, le décalage à lui faire subir (au sens *Chiffre de César*).
Si, par exemple, la lettre **D** du texte clair se trouve associé à la lettre **O** de la clé, la lettre **D** sera chiffrée en utilisant un procédé de César où la lettre **A** est remplacée par un **O** … et donc le **D** par un **R**. Cela revient dans ce cas à décaler l'alphabet de 14 lettres.
Voici un exemple où la clé de chiffrement est **SOL** (qui ne connait pas la clé de sol ?) :

Texte clair	**A**	**D**	**E**	**M**	**A**	**I**	**N**
Clé	S	O	L	S	O	L	S
Décalage	18	14	11	18	14	11	18
Texte chiffré	**S**	**R**	**P**	**E**	**O**	**T**	**F**

La fonction arithmétique de chiffrement est du type $n \longrightarrow n - q \times k_{clef}\ [26]$ où k évolue en fonction de la position du caractère n dans le texte clair.

BONJOUR TOUT LE MONDE avec la clé **VOUTE** se chiffre **WCHCSPFNHYOZYFSI** et le déchiffrement de **UHLGISXVUE** avec la clé **CHAMPS** donne **SALUT A VOUS**.

On voit sur ces exemples qu'une même lettre n'est pas toujours chiffrée de la même façon … à condition de ne pas retomber en face de la même lettre du mot clé. Le risque sera d'autant plus faible que le mot clé est long. Il est d'ailleurs possible de prendre comme mot clé un proverbe ou un texte classique, et le procédé devient incassable si la clé est un texte de même longueur que le texte clair.

1.4. Chiffre de Jefferson (1743 - 1826)

Les procédés de chiffrement utilisés par le président américain Jefferson étaient des variantes du chiffre de Vigénère dans lequel étaient introduits des coefficients multiplicatifs.
Le premier s'appuyait sur la fonction arithmétique de chiffrement du type $n \longrightarrow n + q \times k_{clef}\ [26]$.
Par exemple, avec le mot clé **PLATE** et $q = 11$, le texte clair **BONJOUR** se chiffre de la façon suivante en **KFNKGDI** :

Texte clair	**B**	**O**	**N**	**J**	**O**	**U**	**R**
n	1	14	13	9	14	20	17
clé	P	L	A	T	E	P	L
$k_{clé}$	15	11	0	19	4	15	11
$n + 11 \times k_{clé}$	10	5	13	10	6	3	8
Texte chiffré	**K**	**F**	**N**	**K**	**G**	**D**	**I**

Le déchiffrement utilise évidemment la fonction $n \longrightarrow n - q \times k_{clef}\ [26]$.
Par exemple, toujours avec la clé (**PLATE**;11), **BRLVLJKIPFB** signifie **SALUTATIONS**.
Le second procédé utilisé par Jefferson correspondait à la fonction $n \longrightarrow n + q \times k_{clef}\ [26]$ où p était premier avec 26 pour pouvoir posséder un inverse p' modulo 26 et le déchiffrement se faisait grâce à la fonction $n \longrightarrow p' \times (n - k_{clef})\ [26]$.
Par exemple, avec le mot clé **DEHUIT** et $p = 11$ (et donc $p' = 19$)), **BONJOUR** se chiffre **OCUPGFI** et **TEYGJTEOFHY** signifie **SALUTATIONS**.

retour au sommaire

1.5. Généralisation

Tous les procédés analysés jusqu'ici ne sont que des cas particuliers de la fonction de chiffrement

$$n \in \mathbb{Z}/26\mathbb{Z} \rightarrow p \times n + q \times k_{clef}\,[26]\; p \wedge 26 = 1$$

dont la fonction de déchiffrement est :

$$n \in \mathbb{Z}/26\mathbb{Z} \rightarrow p' \times (n - q \times k_{clef})\,[26]\; p \times p' = 1\,[26]$$

pour lesquels la clé de chiffrement est le triplet (*motclef ; p; q*).
Par exemple, avec la clé (**OPATRE**;7;12), l'inverse de 7 modulo 26 étant 15, **BONJOUR** se chiffre **TSNFQGB** et **IYZEZWPCUHS** signifie **SALUTATIONS**.
On notera les cas particuliers :

- si la clé est du type (B ; 1 ; *k*), c'est un chiffre de César
- si la clé est du type (B ; *p* ; *q)*, c'est un chiffre affine
- si la clé est du type (*mot clef* ; 1 ; 1), c'est un chiffre de Vigénère
- si la clé est du type (*mot clef* ; 1 ; *q*), c'est un chiffre de Jefferson type 1
- si la clé est du type (*mot clef* ; *p* ; 1), c'est un chiffre de Jefferson type 21.6. Le chiffre MIAS

Ce procédé consiste à multiplier modulo 26 le caractère clair par le caractère clé :
$n \longrightarrow n + q \times k_{clef}\,[26]$.
On voit tout de suite l'inconvénient : avec la clé **MATITE**, le texte **A DEMAIN MATIN** devient **AAYSAGAAAWWA** et le déchiffrement pose un réel problème. Par ailleurs, un **A** est toujours chiffré par un **A** avec ce procédé.
Pour pouvoir déchiffrer, il faudrait que chaque caractère du mot clé soit de rang premier avec 26, ce qui ne laisse que 12 lettres possibles … dont aucune voyelle !
Pour pallier cet inconvénient, on peut étendre le jeu de caractères à 37 (nombre premier) en y adjoignant les chiffres et l'espace :

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	G	H	I
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
J	K	L	M	N	P	P	Q	R	S	T	U	V	W	X	Y	Z	-	
19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	

Seul le caractère **0** n'est plus inversible et est interdit dans le mot clé.
Le chiffrement est alors la fonction $n \longrightarrow n + q \times k_{clef}\,[37]$
et le déchiffrement $n \longrightarrow n + (q \times k_{clef})^{-1}\,[37]$ où l'inverse est évidemment considéré modulo 37.
Par exemple, les caractères du mot clé **MATITE** ont pour rangs 22/10/29/18/29/14 et leurs inverses modulo 37 sont respectivement 32/26/23/35/23/8. Déchiffrer un message crypté avec la clé **MATITE** revient à le chiffrer avec la clé **WQNZN8**.
Ainsi, toujours avec la clé **MATITE**, le texte clair **A DEMAIN 8H** se chiffre **ZR7U9TQ88XC**, et le texte crypté **WPJ4U F6CIY** se déchiffre **PLUTOT 8H15**.

2. LE PROBLÈME DE LA TRANSMISSION DES CLÉS

2.1. Le talon d'Achille

La puissance grandissante de l'outil informatique fournit des moyens efficaces aux briseurs de chiffre. Les procédés examinés dans le paragraphe précédent sont loin d'être infaillibles … et sont même brisés, pourvu que le cryptanalyste possède un texte chiffré assez long.

retour au sommaire

Les mêmes moyens informatiques ont offert aux développeurs la possibilité de mettre au point des procédés de chiffrement à clé secrète de plus en plus sophistiqués et qui, eux, résistent aux briseurs les plus tenaces.

Citons pour mémoire le procédé Lucifer, développé chez IBM par Horst Feistel dans les années 70, qui crypte les messages par brouillages itérés. Chaque caractère du texte est remplacé par l'octet correspondant à son code ASCII (de 0 à 255). Cette succession de 1 et de 0 (des bits) est ensuite mélangée par application à chaque bloc de 32 bits de permutations successives. Le battage de ce jeu de cartes repose sur un nombre clé dont la connaissance permettra au récepteur de retrouver l'ordre originel, et donc le texte clair. La fiabilité et l'inviolabilité de ce procédé en fit un standard adopté officiellement aux États-Unis en 1976, sous le nom de DES (*Data Encryption System*). Il demeure encore aujourd'hui une norme américaine de chiffrement pour les échanges commerciaux.

Mais tous ces procédés de chiffrement ont en commun un inconvénient majeur : une clé secrète. Le problème de sa transmission entre les personnes utilisant le procédé est sérieux. Aussi sophistiqué soit-il, un chiffre ne présente plus aucun intérêt si l'ennemi intercepte sa clé.

2.2. Naissance d'un remède

En 1976, Whitfield Diffie et Martin Hellman eurent l'idée de résoudre le problème d'échange des clés grâce à une fonction arithmétique impossible à inverser dans un temps raisonnable.

Il s'agit des fonctions du type $n \longrightarrow a^n \ [p]$ où p est un nombre premier et a un entier différent de 0 et de 1.

Prenons par exemple $a = 53$ et $p = 21\,997\,783$. Sur mon ordinateur, un programme a mis $1{,}3 \times 10^{-4}$ seconde seconde à calculer $a^{15\,634\,410}$ modulo p pour trouver le résultat 100928. Mais sur le même ordinateur, avec un programme tapé dans le même langage, la résolution de l'équation en n : $53^n = 100928\ [21997783]$ a pris 39 secondes, soit 300000 fois plus!

Il est intéressant de se pencher sur la raison de ce décalage.

Prenons l'exemple du calcul de a^{106}. La première idée est que 105 multiplications sont nécessaires. Mais c'est compter sans le soutien du système binaire: l'écriture 106 = 2+8+32+64 fournit $a^{106} = a^2 \times a^8 \times a^{32} \times a^{64}$. On commence donc par calculer de a^2 à a^{64} par élévations au carré successives (6 multiplications) et puis on multiplie entre elles les puissances concernées (trois multiplications). Ainsi le calcul de a^{106} n'a demandé que 9 multiplications … et non pas 105.

De façon analogue, le calcul de $53^{15634410}$ ne nécessite que 38 multiplications.

Par contre, les restes de 53^n modulo 21 997 783 ne présentent aucune régularité et ne vont pas croissants. Pour résoudre l'équation $53^n = 100\,928\ [21\,997\,783]$, il n'est d'autre possibilité que de faire évoluer n de 1 en 1 jusqu'à tomber sur le résultat.

Avant d'aboutir à la solution $n = 15634410$, il aura fallu… 15634409 multiplications !

On imagine qu'avec un nombre premier de plusieurs centaines de chiffres et une puissance assez élevée, l'inversion de la fonction, bien que possible, demande des milliards d'années de calcul !

2.3. L'administration du remède

L'idée d'utiliser ces fonctions pour échanger publiquement des clés fait partie du trait de génie de Whitfield Diffie et Martin Hellman.

Imaginons qu'Alice et Bernard disposent d'un procédé de chiffrement connu fondé sur le choix d'une clé numérique (comme le DES par exemple). Par téléphone ou par courrier, ils conviennent d'un nombre p premier et d'un nombre a, $1 < a < p$. Comme exemple, prenons les nombres a et p du paragraphe précédent.

Alice choisit un nombre secret $\alpha = 3660$ et Bernard un nombre secret $\beta = 4307$..

Alice calcule $a^{\alpha} = 11375235\ [p]$ et le transmet à Bernard.

Bernard fait de même en envoyant à Alice $\alpha^{\beta} = 2913121\ [p]$.

Bernard calcule élève alors le nombre envoyé par Alice à la puissance β modulo p, et trouve 10928.

Alice fait de même en élevant le nombre envoyé par Bernard à la puissance α modulo p et trouve également 10928 puisque $(a^{\alpha})^{\beta} = (a^{\beta})^{\alpha} = a^{(\alpha \times \beta)}$.

retour au sommaire

Ils utilisent alors ce résultat pour chiffrer leurs messages avec le procédé convenu.
Imaginons qu'un malfaisant Igor espionne les communication. Il connait a, p, a^{α} [p] et a^{β} [p].
Comme nous l'avons vu précédemment, cela ne lui permettra pas de restituer ni α, ni β, ce qui serait indispensable pour connaitre la clé de chiffrement $a^{(\alpha \times \beta)}$ [p].
Sans l'échanger concrètement, Alice et Bernard ont pu convenir d'une clé de chiffrement commune.

2.4. La faille du remède

Dans un procédé d'encryptage donné, l'utilisation répétée d'une même clé présente un danger : celui de fournir à un espion trop de textes chiffrés exactement de la même façon. Pour des questions de sécurité, il est bon de changer les paramètres de chiffrement régulièrement (pendant le dernière guerre, la clé de la machine *Enigma* changeait tous les jours).
Imaginons qu'Alice et Bernard résident dans deux pays à fort décalage horaire et ne puissent facilement être en ligne en même temps. Alice veut envoyer un message crypté à Bernard. Elle lui envoie les nombres qu'elle a choisis : a, p et a^{α}. Bernard consulte sa boite quelques heures plus tard, renvoie à Alice son nombre a^{β} et calcule la clé. Encore plus tard, Alice relève sa boite aux lettres et peut alors, elle aussi, calculer la clé. Elle envoie alors à Bernard le message chiffré que ce dernier lira lorsqu'il se réveillera, encore plus tard.
Whitfield Diffie et Martin Hellman sentaient que cette multiplicité d'échanges avant de pouvoir communiquer créait un handicap. Qui plus est, la fuite et l'intoxication restent possibles avec ce procédé : n'importe qui peut se faire passer pour l'un des deux en piratant sa boite à lettres sans que l'autre ne se doute de quoi que ce soit.
Ils cherchèrent mieux : un moyen pour qu'Alice puisse directement laisser sur la boite à lettres de Bernard un message crypté en étant sûre que lui seul pourrait le déchiffrer. Il fallait donc que la clé de chiffrement de Bernard soit connue de tous …, mais que lui seul possède la clé de déchiffrement. Cela serait comparable à un cadenas que tout le monde peut fermer, mais que seul le détenteur de la clé peut ouvrir. N'importe qui pourrait acheter en magasin un cadenas ouvert de type *Bernard*, enfermer le message dans une boite et l'envoyer à Bernard, sachant que lui seul possède la clé pour ouvrir ce type de cadenas.
La solution fut trouvée en 1977 par Ronald Rivest, Adi Shamir et Leonard Adleman qui inventèrent le RSA (initiales de leurs inventeurs), protocole de chiffrement à clé révélée.

3. LE GRAAL : UN CHIFFREMENT À CLÉ RÉVÉLÉE

3.1. Le but de la manœuvre

Outre le fait de conserver la confidentialité d'un message, un procédé de chiffrement doit aussi éviter les intoxications. C'est à dire que l'émetteur doit être sûr que seul le destinataire pourra le déchiffrer, et le récepteur doit être sûr que le message a bien été envoyé par l'expéditeur présumé.

3.2. Les vertus d'une clé révélée

Un tel procédé est fondé sur l'existence de fonctions mathématiques bijectives f qui servent à chiffrer les messages et qui sont connues de tous, mais dont, pour chacune d'entre elles, la réciproque f^{-1} n'est connue que d'une personne (le propriétaire de la fonction), et impossible à déterminer à partir de f.
Pour en comprendre l'intérêt, supposons qu'il existe un tel procédé. Dans un annuaire, nous allons trouver la fonction f_A (resp. f_B) attribuée à Alice (resp. Bernard), dont seule Alice (resp. Bernard) connait la fonction réciproque f_A^{-1} (resp. f_B^{-1}).
Alice veut envoyer le message T (texte clair) à Bernard. Elle se procure dans l'annuaire f_B, calcule $T' = f_B \circ f_A^{-1}(T)$ et fait parvenir T' à Bernard.
Pour déchiffrer ce message, il faudra lui appliquer $f_A \circ f_B^{-1}$.
Alice est donc sûre que seul Bernard pourra le lire puisque lui seul connait f_B^{-1}.
Mais quand Bernard aura appliqué f_B^{-1} au message crypté reçu, il n'obtiendra pas un message clair. Comme le message est sensé provenir d'Alice, Bernard se procurera la clé publique f_A et l'appliquera au résultat

retour au sommaire

obtenu. S'il obtient un message clair *T*, il saura alors que celui-ci provient bien d'Alice puisqu'elle seule pouvait le crypter avec f_A^{-1} .

3.3. Où poussent de telles fonctions ?

Le système repose encore cette fois sur des manipulations facilement réalisables dans un sens et extrêmement longues dans l'autre sens, donc considérées comme impossibles.
En fait Alice va partir de f_A^{-1} pour construire f_A (manipulation aisée) qu'elle communiquera à tout le monde, sachant qu'il faudrait des années pour retrouver f_A^{-1} à partir de f_A.
Évidemment la technologie évolue et un calcul nécessitant un siècle à un moment donné peut très bien se voir réduit à quelques secondes cinq ans plus tard. Mais cela est-il vraiment grave que les documents soient décryptés cinq ans plus tard ? Et les mathématiques permettent souvent d'augmenter la difficulté en choisissant des nombres plus grands pour augmenter les temps de calcul de façon exponentielle, ou en découvrant des nouvelles fonctions pathologiques au comportement exaspérant.
Nous analyserons dans les sections suivantes deux exemples de chiffrement à clef révélée.

3.4. Codage numérique d'un texte

Le chiffrement par fonction mathématique suppose déjà un codage numérique des caractères. Le code ASCII étendu (de 0 à 255) permet de transformer en nombre les caractères alphanumériques d'un texte, ainsi que les ponctuations et les lettres accentuées. Nous utiliserons donc dans la suite le code ASCII, chaque caractère du texte étant alors représenté par un octet.

4. CHIFFREMENT PAR EMPILEMENT

4.1. Un problème difficile

Considérons un ensemble de 16 pièces dont les hauteurs sont données par le vecteur *A*' :
A' = (65455 ; 51479 ; 9551 ; 33078 ; 52180 ; 24929 ; 63834 ; 34261 ; 68522 ; 71589 ; 35795 ; 6135 ; 77725 ; 78122 ; 70285 ; 61139), et j'empile certaines de ces pièces sans vous dire lesquelles en vous livrant la hauteur totale obtenue : 279 095. A votre charge de retrouver les pièces utilisées !
Nous admettrons pour l'instant qu'une seule configuration peut donner l'empilement dont il est question. L'algorithme le plus rapide connu pour résoudre ce problème consiste à envisager chacune des 2^{16} possibilités. Dans notre exemple, cela reste accessible à tout ordinateur … mais imaginez ce que cela peut donner avec un vecteur à 1000 coordonnées : le nombre de cas à envisager dépasse le nombre d'atomes actuellement estimé dans l'univers !

4.2. Un problème facile

Reprenons le même problème que dans le paragraphe précédent, mais cette fois avec le vecteur *A*' = (1 ; 2 ; 5 ; 9 ; 19; 38 ; 75; 151 ; 302 ; 603 ; 1208 ; 2415 ; 4831 ; 9713 ; 19381 ; 38762) et une pile de taille 23765.
Comme vous l'avez tout de suite remarqué au premier coup d'œil, le vecteur *A*' est très particulier : chacune de ses coordonnées est supérieure à la somme des précédentes.
La pièce 16 (38762) n'a visiblement pas été prise et on peut affirmer que la pièce 15 (19381) a été utilisée. En effet, si nous ne la prenons pas, la somme de toutes les autres ne l'excédant pas, il sera impossible d'arriver à la hauteur voulue.
Il nous reste à composer la hauteur 23765 - 19381 = 4834 et par un raisonnement analogue, nous trouvons que la pièce n°12 (2415) figure dans la pile.
En continuant ainsi, on trouve le vecteur *X* indiquant les pièces prises :

X = (0 ; 1 ; 1 ; 0 ; 0 ; 0 ; 0 ; 1 ; 0 ; 1 ; 1 ; 1 ; 0 ; 0 ; 1 ; 0)

On vérifie aisément que le produit scalaire *A'*. *X* vaut bien 23765.

retour au sommaire

4.3. Comment transformer un problème difficile en un problème facile ?

Vous l'avez compris : le vecteur *A* n'est pas tout à fait, et même pas du tout, aléatoire. Ce n'est qu'un brouillage du vecteur *A'* : il est issu du vecteur *A'* via une bijection difficilement inversible.

Considérons les entiers $m = 79431$ et $w = 65455$. Ces deux nombres sont premiers entre eux. Donc *w* est inversible dans $\mathbf{Z}/m\mathbf{Z}$ et son inverse w^{-1} est obtenu via l'algorithme d'Euclide et l'identité de Bezout.

Ainsi, l'application $\Phi : \; a \in \mathbb{Z}/m\mathbb{Z} \to a \times w\,[m]$ est une bijection. Je vous laisse le soin de vérifier que $\Phi(A') = A$.

De plus, *m* étant supérieur à la somme des coordonnées de *A'*, le travail dans $\mathbf{Z}/m\mathbf{Z}$ permet de discriminer simplement toutes les piles qu'on peut obtenir avec *A'*.

Une personne connaissant *A* ne peut pas retrouver le vecteur *A'* s'il ne connait pas les nombres *m* et *w*.

Nous verrons dans la suite que *A* est la clef révélée de ce procédé et que *m* et *w* en constituent la clef secrète, encore appelée la gâche du chiffrement par empilement.

4.4. Mathématisons un peu

Pour $n \in \mathbb{N}^*$ soit $\mathcal{E} = \{0\,;\,1\}^n$ l'ensemble des vecteurs *x* à coordonnées binaires et posons :

$$\mathcal{A}' = (a'_1\,;a'_2\,;\cdots;a'_n) \in (\mathbb{N}^*)^n, \forall k \in \{2\,;3\,;\cdots;n\}, a_k > \sum_{i=1}^{i=k-1} a_i$$

$$\text{Soit } m \in \mathbb{N}, m > \sum_{i=1}^{i=n} a_1 \;,$$

soit $w \in \{2\,;3\,;\cdots;m-1\}, w \wedge m = 1, \alpha \in \mathbb{Z}/m\mathbb{Z} \xrightarrow{\phi} \alpha \times w\,[m], w' = w^{-1}\,[m]$,

posons $\mathcal{A} = \phi(\mathcal{A}') = (a_1\,;a_2\,;\cdots;a_n), \forall i \in \{1\,;2\,;\cdots;n\}, a_i = \phi(a'_i)$.

Propriété 4.4.i :

L'application $X \in \mathcal{E} \xrightarrow{\Psi} \mathcal{A}' \cdot X = \sum_{i=1}^{i=n} a'_i \times x_i \in \mathbb{N}$ est injective.

Démonstration : Soient $X, Y \in \mathcal{E}, X \neq Y, \Psi(X) = \Psi(Y)$

$k = max(i \in 1\,;2\,;\ldots n), x_i \neq y_i$ et posons $k = max\, i \in (1\,;2\,;\ldots;n), x_i \neq y_i$

On a donc $\forall i \in \{k+1\,;\cdots;n\}, x_i = y_i$ et $x_k \neq y_k$.

Supposons $x_k = 1, y_k = 0$.

Soit $Z = (0\,;0\,;\cdots;0\,;x_{k+1}\,;\cdots;x_n) = (0\,;0\,;\cdots;0\,;y_{k+1}\,;\cdots;y_n)$

Par propriété du produit scalaire, on a $\Psi(X - Z) = \Psi(Y - Z)$

Si k = 1, cela signifie $a' = 0$, ce qui est faux, et si k>1, cela conduit à

$$a'_k \leq \sum_{i=1}^{i=k-1} x_i \times a'_i + 1 \times a'_k = \sum_{i=1}^{i=k-1} y_i \times a'_i + 0 \times a'_k < \sum_{i=1}^{i=k-1} a'_i$$

ce qui est contradictoire avec la construction du vecteur A'.

Donc l'application Ψ est bien injective.

retour au sommaire

Propriété 4.4.ii :

L'application $\mathcal{X} \in \mathscr{E} \xrightarrow{\Psi} \mathcal{A}\ '\cdot\mathcal{X}\ [m] \in \mathbb{Z}/m\mathbb{Z}$ est injective.

Démonstration : L'image de Ψ étant contenue dans [0 ; *m*-1] par choix de l'entier $m > \sum_{i=1}^{i=n} a_i$, cette propriété est une conséquence de la précédente.

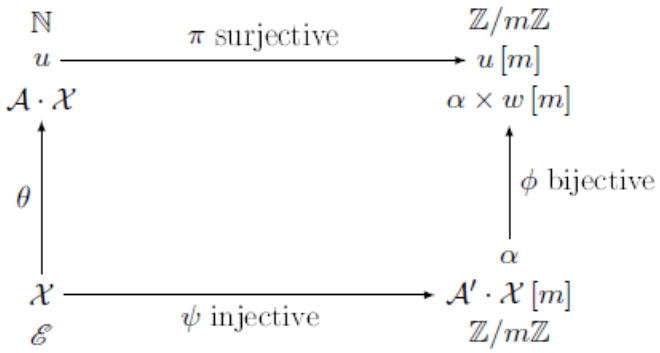
Propriété 4.4.iii :

L'application $\alpha \in \mathbb{Z}/m\mathbb{Z} \xrightarrow{\phi} \alpha \times w\,[m] \in \mathbb{Z}/m\mathbb{Z}$ est bijective.

Cela est simplement dû au fait que w est inversible dans $\mathbb{Z}/m\mathbb{Z}$.

Propriété 4.4.iv :

Dans le schéma suivant :



Cela provient directement de la définition de *A* à partir de *A'* :

$$\phi\circ\psi(\mathcal{X}) = \left(\sum_{i=1}^{i=n} a'_i \times x_i\right) \times w[m] = \left(\sum_{i=1}^{i=n} a'_i \times w \times x_i\right)[m] = \left(\sum_{i=1}^{i=n} a_i \times x_i\right)[m] = \pi\circ\theta(\mathcal{X})$$

Propriété 4.4.v :

L'application $\mathcal{X} \in \mathscr{E} \xrightarrow{\theta} \mathcal{A}\cdot\mathcal{X} \in \mathbb{N}$ est injective.

Démonstration : $\Phi_o\Psi$, composée d'une bijection et d'une injection, est injective.
Avec [4.4.*iv*], on en déduit que $\pi_o\theta$ est injective, et donc cela entraine que θ est injective.

Conséquences : Cela établit qu'un entier obtenu comme somme d'éléments de *A* ne peut l'être que d'une seule façon.

Cela nous fournit aussi un moyen de retrouver $\mathcal{X}$ à partir de $N = \mathcal{A}\cdot\mathcal{X} = \theta(\mathcal{X})$ lorsqu'on connait *m* et *w* :

- considérer $N' = w^{-1} \times N[m] = \phi^{-1}\circ\pi(N)$
- nous avons donc $N' = \phi^{-1}\circ\pi\circ\theta(\mathcal{X}) = \psi(\mathcal{X})$
- or, nous avons vu au §[4.2] comment retrouver $\mathcal{X}$ à partir de $\psi(\mathcal{X}) = \mathcal{A}\ '\cdot\mathcal{X}$

Reprenons l'exemple du §[4.1] avec *N* = 279095.

Nous avions vu au §[4.3] que $\mathcal{A}$ était obtenu à partir de $\mathcal{A}'$ avec *m* = 79431 et *w* = 65455.
Nous avions également mentionné que w^{-1} = 1813 dans $\mathbb{Z}/m\mathbb{Z}$.
Prenons $N' = w^{-1} \times N[m]$ = 1813x279095 [79431] = 23765 .

Le §[4.2] détaille la décomposition de 23765 suivant les coordonnées de $\mathcal{A}'$.

Et nous obtenons ainsi le vecteur $\mathcal{X} = (0;1;1;0;0;0;0;1;0;1;1;1;0;0;1;0)$ tel que $\mathcal{A}\cdot\mathcal{X} = 279095$.

retour au sommaire

4.5. Et le chiffrement dans tout ça ?

Le texte à chiffrer est scindé en groupes de deux caractères (espaces et ponctuations sont des caractères). Chaque caractère pouvant être représenté par un octet (voir table ASCII), chaque groupe de deux caractères correspond, en les accolant, à un vecteur *x* de 16 coordonnées binaires.
Par exemple, le groupe **ar** correspond au vecteur *X* =(0 ; 1 ; 1 ; 0 ; 0 ; 0 ; 0; 1 ; 0 ; 1 ; 1 ; 1 ; 0 ; 0 ; 0) du §[4.2] puisque le code ASCII du **a** est 97 = $[01100001]_2$ et celui du **r** vaut 114 = $[01110010]_2$.

Imaginons que Alice ait choisi en secret les nombres *m* et *w* déjà évoqués, ainsi que le vecteur *A'*. Elle a alors calculé *A* = *w*x*A'* [*m*] et l'a publié dans un annuaire (clé révélée).
Bernard veut envoyer un texte clair à Alice et trouve dans l'annuaire sa clé révélée *A*.Il chiffre **ar** par le produit scalaire *A*x*X* = 279095.
Pour déchiffrer la signification de ce nombre dans le message reçu, Alice utilisera sa clé secrète (*m* ; w^{-1} ; *A'*) qu'elle est la seule à connaitre, selon le protocole expliqué dans l'exemple ci-dessus.

Si vous voulez tester un algorithme de chiffrement par cette méthode, vous pourrez vérifier que le texte **J'arriverai le 12 à 7 h** donne la séquence chiffrée `412834 – 279095 – 404190 – 429516 – 326465 – 183266 – 384784 – 112620 – 142258 – 162280 – 201448 – 375137 ;` et que le message chiffré `414138 – 201192 – 301098 – 310678 – 453736 – 330902` signifie : **Je t'attends**

5. LE CHIFFRE RSA

5.1. Un problème difficile

Décomposer un nombre dont on sait que c'est le produit de deux premiers.
Essayez pour voir avec 29 083. Il faut déjà obtenir tous les premiers à concurrence de sa racine (crible d'Eratosthène). Ici donc de 2 à 170.
Supposons qu'on en ait la liste. Il vous faudra bien une bonne heure à la main pour tester les divisibilités et aboutir au 31e nombre premier qui est 127 et trouver que 29083 = 127x229.

Sur de grands nombres (quelques centaines de chiffres), le temps de calcul informatique croît de façon exponentielle et peut atteindre des dizaines de siècles.

5.2. Un problème facile

Calculer le produit de deux nombres premiers.
À la main, la multiplication de 127 et 229 prend une paire de minutes.
Pour des nombres premiers comportant un grand nombre de chiffres (quelques centaines), un ordinateur effectue le calcul en quelques secondes.

5.3. Un résultat arithmétique utile

Propriété 5.3.i :

Soient *p* et *q* deux nombres premiers, et posons *n* = *p*x*q* et *m* = (*p*-1)x(*q*-1) .

Soit $e \in 2\,;3\,;\ldots;m-1\,,e\wedge m=1$ et *d* l'inverse de *e* modulo *m*.

Soient a et b dans {1; 2 ;...; n-1}. Alors on a : $b \equiv a^e[n] \Leftrightarrow a \equiv b^d[n]$

Démonstration :
Le fait qu'on a aussi $e = d^{-1}[m]$ et que *d* est aussi premier avec *m* rend cette équivalence symétrique et il suffit donc de démontrer une seule implication.

$$b \equiv a^e[n] \Rightarrow b^d \equiv a^{ed}[n]$$

$$e \times d \equiv 1[m] \Rightarrow \exists k \in \mathbb{N}, e \times d = k \times m + 1$$

$$b \equiv a^e[n] \Rightarrow b^d \equiv a^{km} \times a[n]$$

retour au sommaire

Si a $a \wedge n = 1$, alors *a* appartient au groupe multiplicatif des éléments inversibles de $\mathbb{Z}/n\mathbb{Z}$; comme $n = p \times q$ avec *p* et *q* premiers, ce groupe contient (*p*-1)(*q*-1) éléments (voir fonction d'Euler) ; *a* étant dans un groupe multiplicatif d'ordre *m*, on $a^m \equiv 1[n]$, et donc $a^{km} \equiv 1[n]$ et finalement on obtient : $a^{km} \times a \equiv a[n]$

- Si $a \wedge n \neq 1$, comme $n = p \times q$ avec *p* et *q* premiers, cela signifie que *a* est un multiple de *p* ou de *q,* et $a < n$ implique que *a* ne peut être multiple à la fois de *p* et *q.*
  Disons par exemple que *a* est multiple de *p* mais pas de *q* ; alors *a* est premier avec *q,* et dans le groupe multiplicatif dans le groupe multiplicatif $(\mathbb{Z}/q\mathbb{Z})^*$, on a $aq^{-1} \equiv 1[q]$
  et donc $a^m = a^{(p-1)(q-1)} \equiv 1[q]$ et a fortiori $a^{km} \equiv 1[q]$, ce qui signifie que $a^{km} - 1$ est un multiple de *q* ; et comme *a* est un multiple de *p*, cela induit que $(a^{km} - 1) \times a$ est un multiple de *pq*, c'est à dire *n*.
  Finalement cela donne : $a^{km} \times a - a \equiv 0[n]$ et donc, là encore, $a^{km} \times a \equiv a[n]$

  L'implication $b \equiv a^e[n] \rightarrow b^d \equiv a[n]$ est ainsi démontrée, et par symétrie la propriété aussi.

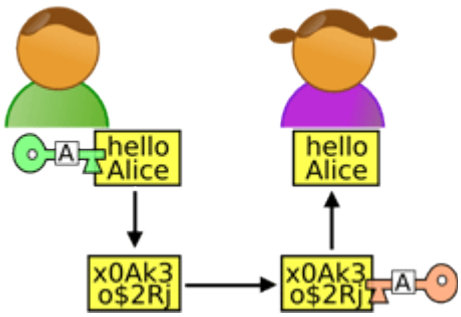
5.4. Chiffrer en RSA

Alice prend par exemple *p* = 479 et *q* = 541 et obtient *n* = 259139 et *m* = 258120.
Elle choisit *e* = 359, premier avec *m*, et calcule son inverse modulo *m* : *d* = 719.
Elle publie dans un annuaire sa clé publique (*n* ; *e*).
Bernard veut crypter **ar** dans un message destiné à Alice. Il accole les deux nombres décimaux (formatés sur 3 chiffres) représentant ces deux caractères en code ASCII et obtient a = 097114..
Remarquons qu'il obtiendra toujours un nombre inférieur à 255255, donc inférieur à *n*.
Il calcule a^e modulo *n* et obtient *b* = 156229, et ce nombre cryptera **ar** dans le message.

Avec la clé publique d'Alice, le message **J'arriverai le 12 à 7 h** va ainsi être chiffré
`195483 – 156229 – 133380 – 228039 – 31701 – 39311 – 87362 – 123219 – 67354 – 159742 – 154974 – 7403.`

5.5. Déchiffrer du RSA

Alice veut déchiffrer la séquence cryptée par *b* = 111116
En vertu de la propriété [5.3.i], elle va calculer b^d modulo *n,* va trouver *a* = 74101et identifier le caractère **J** par son code ASCII 74 et **e** par son code ASCII 101.
Par exemple, le message crypté `111116 – 140003 – 181943 – 4277 – 58734 – 32095` sera déchiffré par Alice en : **Je t'attends**
Nous voyons là que pour déchiffrer, la connaissance de *d* est indispensable. Or *d* ne peut être déterminé qu'en connaissant *m*, et donc en connaissant *p* et *q*. Pour retrouver *p* et *q* à partir de leur produit, le temps de calcul serait astronomique si les nombres premiers choisis par Alice comportaient plusieurs centaines de chiffres.



retour au sommaire