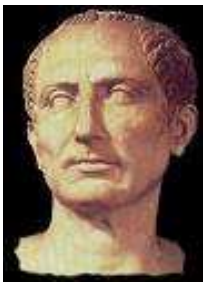
Substitution mono alphabétique : 1ère partie.

César & Co.

Pour ce premier véritable article de cryptologie, nous allons étudier des cryptogrammes assez simples, mais qui vous demanderont déjà d'appréhender de nouvelles notions mathématiques. Rassurez-vous : rien de trop compliqué ! Pour le moment, commençons par le commencement et rendons à César ce qui lui appartient…

I. Chiffrement de César



«*L'expérience, voilà le maître en toutes choses.***»**

[Jules César] - *De bello Civili*

Je vous ne ferai pas l'affront de vous présenter Jules César ! Sans doute l'homme politique le plus important du premier siècle avant Jésus Christ, il a notamment conquis la Gaule grâce à sa victoire sur Vercingétorix à Alésia. Durant cette campagne, afin de communiquer avec ses généraux dans le plus grand secret, il utilisa un procédé de chiffrement qui porte maintenant son nom. Ce chiffre est expliqué par Suétone dans son ouvrage « *Les vies des 12 César* », écrit en 121 après J.C.

Ce chiffre consiste à décaler chaque lettre de 3 emplacements dans l'alphabet. Ci-dessous, vous avez donc l'alphabet clair sur la 1ère ligne et l'alphabet chiffré sur la 2nde. Par convention, le texte clair est toujours en *minuscules* et le texte chiffré toujours en *majuscules*.

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

Pour déchiffrer, on procède évidemment dans le sens contraire…

Par exemple, le texte clair « attaquezalesia » devient « DWWDTXHCDOHVLD ».

II. Chiffrement par décalage

Par une méthode similaire, il est bien sûr possible de décaler d'une autre valeur que 3. Cette valeur est évidemment un nombre entier compris entre 0 et 25, ce qui nous donne 26 possibilités de chiffrement par décalage.

Afin d'exprimer mathématiquement ce chiffrement, il convient à présent de vous expliquer une notion simple mais indispensable : l'arithmétique modulaire (N.B : cette notion est abordée en Terminale S).

L'entier *m* étant fixé, à tout nombre entier *a*, on peut faire correspondre un unique entier *b*, compris entre 0 et *m* – 1. *b* est alors le reste de la division euclidienne de *a* par *m*.
On note alors :

$$a \equiv b \ mod \ m$$

(qui se lit « *a* est congru à *b* modulo *m* »)

Exemple :

$83 = 3 \times 26 + 5$ On effectue la division euclidienne de 83 par 26.

Alors $83 \equiv 5 \ mod \ 26$

Procédé de chiffrement par décalage :

1. A chaque lettre de l'alphabet clair, on fait correspondre un nombre :

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

2. Pour effectuer un décalage de 11 lettres (par exemple), on utilise alors la fonction de chiffrement suivante :

$$c_{11}(x) = x + 11 \quad mod \ 26$$

et on assigne à chaque nouvelle valeur la lettre correspondante de l'alphabet.

Procédé de déchiffrement par décalage :

On utilise le même procédé que ci-dessus avec la fonction de déchiffrement suivante :

$$d_{11}(y) = y - 11 \quad mod \ 26$$

Exemple de chiffrement par décalage

Réalisons le chiffrement du message « Ave Caesar » avec un décalage de 11 lettres.

texte clair	a	v	e	c	a	e	s	a	r
nombre correspondant	0	21	4	2	0	4	18	0	17
+ 11 *mod* 26	11	6	15	13	11	15	3	11	2
texte chiffré	L	G	P	N	L	P	C	L	B

Ainsi le cryptogramme est : « LGPNLPCLB ».

Je vous laisse vérifier que le déchiffrement fonctionne correctement ☺

Dans le cadre d'un chiffrement par décalage, on a vu qu'il n'y a que 26 clefs possibles, voire seulement 25 si on exclut la valeur 0 qui ne modifie pas le texte clair, ce qui n'est pas très utile en cryptologie ! C'est ridiculement peu : il suffit, au pire, de faire 26 essais avant de retrouver le texte clair. Essayer toutes les clefs se nomme « méthode exhaustive », et fonctionne ici parfaitement bien. En moyenne, il faudra seulement 13 essais pour casser un message.

Autant dire que la sécurité d'un cryptogramme chiffré par décalage est quasi-nulle. Nous verrons ainsi qu'il est nécessaire que l'espace des clefs soit grand, mais comme on peut l'imaginer, cette condition n'est pas suffisante.

III. Chiffrement affine

Afin d'accroître le nombre de clefs possibles, le chiffrement par décalage peut-être amélioré en chiffrement affine. Les lecteurs ayant tous suivi avec assiduité le cours de maths de 3ème (!) se doutent déjà de la technique que je vais décrire.

Le procédé reste identique à celui du chiffrement par décalage, mis à part évidemment les fonctions de chiffrement et de déchiffrement qui sont… des fonctions affines bien sûr !

$$c(x) = ax + b \quad mod\ 26$$

et

$$d(y) = a^{-1}(y - b) \quad mod\ 26$$

Quelques explications mathématiques s'imposent ici…

- La clef est ici représentée par le couple (*a* ; *b*).
  a et *b* sont 2 entiers naturels compris entre 0 et 25, vu que nous travaillons en arithmétique modulo 26.

- On démontre (mais pas ici !) que *a* doit être premier avec 26, i.e qu'il peut prendre les 12 valeurs suivantes : 1,3,5,7,9,11,15,17,19,21,23,25. On notera ici que l'on retrouve le chiffrement par décalage quand *a* = 1.
  Le paramètre *b*, quant à lui, peut être quelconque.

- Le chiffrement affine admet donc 12 x 26 = 312 clefs possibles. La sécurité est meilleure que le chiffrement par décalage, mais le nombre de clefs reste bien trop petit pour résister longtemps… surtout face à un programme informatique !

- Enfin, que signifie ce « a^{-1} » ?
  Il s'agit de l'inverse de a, i.e l'unique entier naturel entre 0 et 25 tel que :
  $$a.a^{-1} \equiv a^{-1}.a \equiv 1 \quad mod\ 26$$

  Exemples :
  $$1^{-1} = 1\ ; \qquad 3^{-1} = 9\ ; \qquad 5^{-1} = 21$$

  Attention ! Ces résultats sont valables dans l'ensemble des entiers compris entre 0 et 25, noté $\mathbb{Z}_{26}$.

Exemple de chiffrement affine

Prenons la clef de chiffrement (7 ; 3).

On note que : $7^{-1} = 15 \quad mod\ 26$

La fonction de chiffrement est :

$$c(x) = 7x + 3 \quad mod\ 26$$

Et la fonction de déchiffrement est :

$$d(y) = 15(y - 3) \quad mod\ 26$$

$$d(y) = 15y + 7 \quad mod\ 26$$

Vérifions que la fonction *d* est bien l'inverse de la fonction *c* :

Pour tout $x \in \mathbb{Z}_{26}$:

$$\begin{aligned} d\big(c(x)\big) &= d(7x + 3) \\ &= 15(7x + 3) + 7 \\ &= 105x + 45 + 7 \\ &= x + 52 \qquad \text{(on est en } mod \text{ 26 !)} \\ &= x \end{aligned}$$

Chiffrons à présent le mot « alpha ».

1. On convertit les lettres en nombres :

a	l	p	h	a
0	11	15	7	0

2. On chiffre en utilisant la fonction *c* :

$$c(0) = 7 \times 0 + 3 \quad mod\ 26 = 3 \quad mod\ 26 \equiv 3$$
$$c(11) = 7 \times 11 + 3 \quad mod\ 26 = 80 \quad mod\ 26 \equiv 2$$
$$c(15) = 7 \times 15 + 3 \quad mod\ 26 = 108 \quad mod\ 26 \equiv 4$$
$$c(7) = 7 \times 7 + 3 \quad mod\ 26 = 52 \quad mod\ 26 \equiv 0$$

3. On obtient alors le texte chiffré :

3	2	4	0	3
D	C	E	A	D

Je vous laisse vérifier que l'on retrouve bien le mot « alpha » en déchiffrant « DCEAD » ☺

Et voilà, les chiffrements par décalage et affine n’ont plus de secrets pour vous ! Mais comme vous vous en êtes rendu compte, ces chiffres (au sens cryptologique du terme) ont une sécurité lamentable, du fait du faible nombre de clefs possibles.

Rassurez-vous, dans le prochain article, nous étudierons un chiffre qui autorise plus de 4×10^{26} clefs possibles !

Je vous donne donc rendez-vous dans le prochain article de cryptologie pour découvrir le chiffrement par substitution !