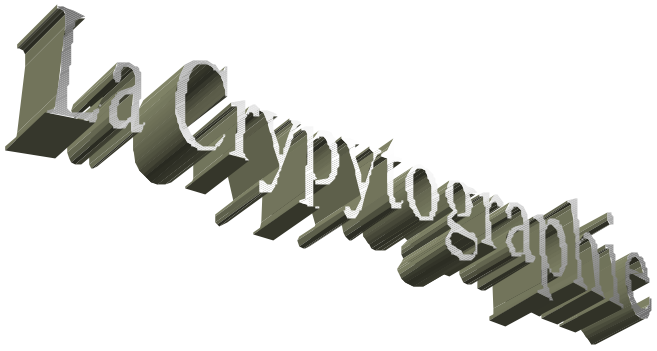

Bien souvent, quand deux interlocuteurs échangent des informations par ligne téléphonique, sur le réseau informatique, ils souhaitent que ces informations restent confidentielles, c'est à dire qu'un intrus qui intercepte ce message ne puisse pas comprendre celui-ci. Pour cela, l'expéditeur doit coder son message avant de l'envoyer et le récepteur le décoder avant de le lire. La nécessité de coder et de décoder le message a été ressentie depuis longtemps par tous ceux qui souhaitent communiquer secrètement : les militaires, les banques...

Le but de la cryptographie est de concevoir des méthodes de codage difficile à casser. Certaines méthodes de codage existantes sont connues, comme la méthode césarienne, la substitution, la transposition et la méthode RSA.

I. La méthode césarienne, la substitution et la transposition :

A. La méthode césarienne :

1. *Principe :*

Chaque lettre est décalée de quatre rang par rapport à sa place dans l'alphabet

2. *Exemple :*

bateau devient **fexiey** par cette méthode.

B. La méthode de substitution :

1. *Principe :*

On utilise un mot quelconque pour coder un message : pour cela, on additionne le rang de la 1ère lettre du mot à crypter avec le rang de la 1ère lettre du mot servant à coder, et ainsi de suite...

2. *Exemple :*

bateau devient **ccwfcx** avec la clef **abc** dans la méthode de substitution.

C. La méthode de transposition :

Le message est en général écrit sans division de mots en rangées de lettres disposées dans un bloc rectangulaire. Les lettres sont alors transposées selon un ordre prédéterminé, par exemple en colonne verticales, en diagonales, ou en spirales, ou bien suivant des systèmes plus complexes, comme le cheminement du cavalier, qui correspond aux trajectoires possibles du cavalier dans un jeu

d'échecs. La disposition des lettres dans le message dépend de la taille du bloc utilisé et du chemin suivi lors de l'inscription et de la transposition des lettres. Pour que le chiffre soit encore plus sûr, un mot clef ou un nombre clef peut être utilisé; par exemple, si l'on transpose en colonne verticales, le mot clef code nécessite que les colonnes soient prisent selon l'ordre 1-4-2-3, qui représente l'ordre alphabétique des lettres du mot code, plutôt que selon l'ordre normal 1-2-3-4.

D. Avantages et inconvénients :

La méthode césarienne possède un des plus gros défauts : on s'aperçoit que la fréquence des lettres les plus couramment utilisées dans une langue réapparaît avec la même fréquence dans le message crypté. Inutile de dire que ce système est désuet.

Le défaut de la méthode de substitution tient au fait que l'on ne peut coder un long message : ceci est dû à la périodicité du mot servant à coder et à la possibilité de redondance dans un texte. Par recoupement successifs, on peut arriver à décoder le message. Un des avantages est que pour un message assez court, il est difficile de décoder.

Dans la méthode de transposition les chiffres de type transposition peuvent être reconnus grâce aux fréquences d'apparition des lettres courantes du langage utilisé. On peut trouver la solution de ces chiffres sans clef, en redisposant les lettres selon des figures géométriques variées, tout en cherchant les anagrammes de mots probables, jusqu'à ce que la méthode de chiffrement soit découverte.

Une des méthodes les plus couramment utilisées et n'ayant pas ces inconvénients est la méthode RSA.

II. La méthode RSA :

Le système de cryptographie RSA est considéré comme le code public le plus sûr. Cette méthode est due à Rivest, Shamir et Adleman en 1977. On pensait alors que le message qu'ils avaient codé sous le nom de RSA129 ne serait décoder pas avant des milliards d'années. RSA129 est composé du message et d'une clef public à 129 chiffres.

1. Principe :

RSA est un standard de cryptographie. En effet, il est inconcevable d'envoyer des messages sur le réseau des réseaux (Internet) si tout le monde peut les visionner. D'où la nécessité de trouver un moyen qui offre la garantie du secret de votre message et ce grâce à la méthode RSA qui, depuis 1977, est toujours d'actualité et le restera encore pendent un long moment. Pour expliquer cette méthode nous avons besoin de la fonction d'Euler servant à coder/décoder les messages.

i. Fonction d'Euler, définition :

Si $n \in \mathbb{N}^*$, $\varphi(n)$ est le nombre de naturel inférieur à n et premier avec lui. Ex :$\varphi(10)=4$ (cf. :{1,3,7,9})$\varphi(1)=1$ car 0 est premier avec 1. Si p est premier $\varphi(p)=p-1$.

ii. Définition de la cryptographie à clefs publiques :

Il est relativement facile de concevoir un code secret dont le décryptage est rigoureusement impossible par qui ne possède pas la clef : il suffit pour cela de choisir une base r, et de tirer au hasard une suite (k_i) d'entiers entre 0 et r-1. Le message sera d'abord transformé en une suite (N_i)

d'entiers entre 0 et r-1 ; pour décoder, il suffit d'appeler (C_i) l'unique entier entre 0 et r-1 qui est congru à M_i+k_i [r]. Pour déchiffrer, connaissant C_i et k_i, il suffit de remarquer que M_i est l'unique entier entre 0 et r-1 qui est congru à C_i-k_i [r]. Si les k_i ont été choisi complètement au hasard, les C_i sont eux aussi complètement aléatoire, c'est à dire pour ceux qui ne connaissent pas les k_i, et il est donc impossible de décrypter le message sans connaître la clef (k_i). L'inconvénient de ce système est la transmission de la clef : celle-ci est aussi longue que le message, et ne peut être utilisée deux fois (sans quoi elle n'est plus inviolable).

Le principe de la cryptographie publique est d'utiliser un code pour chaque utilisateur : chacun doit pouvoir chiffrer un message mais seul son destinataire peut le déchiffrer. La clef de chiffrement est publique alors que la clef de déchiffrement est gardée secrète. On remarque que même l'émetteur n'est pas capable de décoder ce que lui-même envoie. En gros, il y a une clef de chiffrement pour chaque destinataire.

Voici donc en quoi consiste la méthode RSA. En effet il est facile de produire des grands nombres (environ 10^{100}) mais très difficile de les décomposer en facteurs premiers.

iii. Description de la méthode :

Le destinataire crée la clef de codage (publique). Pour cela, il choisit deux nombre premier p et q grand (environ 10^{50}) telle que la décomposition en facteurs premiers de (p-1)(q-1) soit connue. Celle-ci doit comporter un grand nombre premier. Le destinataire calcul r=p*q et la rend publique. Le destinataire connaît φ(r)=(p-1)(q-1), la décomposition de φ(r) en facteurs premiers et donc celle aussi de φ(φ(r)).

Le destinataire choisit alors s grand premier avec φ(r) et le rend public ; en secret il calcul $t \equiv s^{\varphi(\varphi(r))-1}$ [φ(r)].

Pour envoyer ce message au destinataire, on représente ce message sous la forme d'une suite finie (M_i) d'entiers compris entre 0 et r-1. L'auteur du message utilise alors la clef de codage (r,s) rendue publique, et calcule C_i qui est l'unique entier compris entre 0 et r-1, congrue à M_i^s [r]. Pour décoder le message, le destinataire utilise t car M_i est l'unique entier entre 0 et r-1 congrue à C_i^t [r].

On voit ainsi que pour chiffrer le message il suffit de connaître (r,s) et que pour le décrypter, il faut connaître t. Pour calculer, il faudrait arriver à calculer au préalable φ(r) ce qui est très difficile quand r est très grand.

2. Exemple :

Voir MAPLE.

Conclusion :

Le principe de la méthode RSA est basée sur le fait que l'on doit factoriser de très grands nombres composés de plus de 150 chiffres pour pouvoir trouver la clef donc il est très difficile actuellement de le faire. Pour le code RSA129 il faut savoir qu'il a été décrypter par Arjen Lenstra ,des Laboratoires Bell, aidé de trois passionnés d'informatique et de plus de 600 abonnés à Internet après huit moins de travail. Cette factorisation de RSA129 est l'un des calculs le plus compliqué à effectuer. Paul Leyland de l'Université d'Oxford a utiliser un filtre quadratique polynomial multiple pour décomposer le problème en de nombreux problèmes plus simples, ce qui lui a permit de travailler avec 1600 ordinateurs et donc le temps de calcul de ces 1600 ordinateurs. Les possesseurs d'abonnements Internet utilise la méthode RSA pour crypter leurs données pour quelles restent confidentielles, mais elles ne doivent en rien se préoccuper de la validité de cette méthode même si elle a été cassée sur un module de 129 chiffres car on ne connaît pas aujourd'hui d'algorithmes efficaces de factorisation des très grands nombres composés même s'ils existent.

ANNEXE :

Le petit théorème de Fermat généralisé :

Si m et n sont premiers entre eux (m<n), $m^{\varphi(n)}\equiv 1[n]$.

φ est la fonction d'Euler donnant le nombre des entiers inférieurs et premiers avec n.

Ici $\varphi(n)=(p-1)(q-1)$ et donc $m^{(p-1)(q-1)}\equiv 1[n]$ d'où $\forall k\in\mathbb{N}$, $m^{k(p-1)(q-1)}\equiv 1[n]$ et finalement on obtient $m^{k(p-1)(q-1)+1}\equiv m[n]$. Comme $p\equiv q\equiv 2[3]$, on a $2(p-1)(q-1)+1\equiv 0[3]$ et donc $\exists k\in\mathbb{N}$ tel que $2(p-1)(q-1)+1=3k$.

Théorème de Lagrange :

(G,x) un groupe fini tel que d=Card(G) alors :

$\forall x\in G$, $x^d=1$ (neutre G) (# l'ordre de x divise le cardinal de G)

Démonstration du théorème de Lagrange :

(G,x) de cardinal d. $g\in G$, n=ordre(g)=Card(Gr(g))=$\{1,g,\ldots,g^{n-1}\}$. On pose H=Gr(g). On définit une relation d'équivalence. Les classes d'équivalence sont toutes de cardinal=n.

Démonstration :

Cl(x)=$\{x,g^x,\ldots,g^{n-1}x\}$?

$\{x,g^x,\ldots,g^{n-1}x\}\subset$Cl(x), oui car $x(g^kx)^{-1}=g^{-k}\in H$

Cl(x)$\subset\{x,g^x,\ldots,g^{n-1}x\}$, oui car si $x\Re y$, $yx^{-1}\in H$, $yx^{-1}=g^k$, $y=g^kx$.

Enfin les $x,gx,\ldots,g^{d-1}x$ sont 2 à 2 différents car $1,g,\ldots,g^{d-1}$ le sont.

Les classes d'équivalence forme une partition de G :

ΣCard(Cl())=Card(G), nombre de classes*n=d donc n divise d.

Donc l'ordre de x divise le cardinal de G.

Application :

Pour p et q premiers différents, on a pour tout x non multiple de pq :

$x^{(p-1)(q-1)}\equiv 1[pq]$ (# autrement dit dans $\mathbb{Z}_{/pq\mathbb{Z}}$ $\xi^{(p-1)(q-1)}=1$)

Démonstration :

On travaille dans le groupe des inversibles de $\mathbb{Z}_{/pq\mathbb{Z}}$. Les restes r=0,1,2,…,pq-1 ne sont pas inversibles quand $r\wedge pq=1$, il s'agit donc (p et q premiers distincts) des multiples de p ou de q $\in[[0,pq-1]]$. Il s'agit donc :

$0.p,1.p,\ldots,(q-1)p$

$0.q,1.q,\ldots,(p-1)q$

ce qui donne 1+(q-1)+(p-1) valeurs distincts. Le cardinal des inversibles est donc :

$pq-(q+p-1)=q(p-1)-(p-1)=(q-1)(p-1)$

Donc pour x inversible [pq], d'après le théorème de Lagrange $\xi^{(p-1)(q-1)}=1$

et $x^{(p-1)(q-1)}\equiv 1[pq]$

Corollaire :

Si $es\equiv 1[(p-1)(q-1)]$

donc $es=1+k(p-1)(q-1)$ donc $x^{es}=x^1*x^{k(p-1)(q-1)}$

donc $x^{es}\equiv x*1[pq]$

alors $m\equiv x^e[n]\Rightarrow m^s\equiv x^{es}[n]\Rightarrow m^s\equiv x[n]$

Programmes réaliser grâce au logiciel de programmation BORLAND PASCAL v.7 :

Programme représentant le codage selon la méthode césarienne :

```
Program Cryptography_2;
Uses WinCrt;
Var
   mot_a_crypter:string;
   mot_crypter:string;
   mot_decrypter:string;
Procedure cryptage(mot:string;var crypt:string);
Var
   i,j:integer;
   place_lettre:integer;
   mot2:string;
Begin
     mot2:='';
     for i:=1 to length(mot) do
         begin
              if mot[i]<>' ' then
                 begin
                      place_lettre:=ord(mot[i])-96;
                      place_lettre:=place_lettre+4;
                      if place_lettre<=0 then
place_lettre:=26+place_lettre;
                      if place_lettre>26 then
place_lettre:=place_lettre-26;
                      mot2:=mot2+chr(place_lettre+96);
                 end else mot2:=mot2+' ';
         end;
     crypt:=mot2;
End;
Procedure decryptage(mot:string;var decrypt:string);
Var
   i,j:integer;
   place_lettre:integer;
   mot2:string;
Begin
     mot2:='';
     for i:=1 to length(mot) do
         begin
              if mot[i]<>' ' then
                 begin
                      place_lettre:=ord(mot[i])-96;
                      place_lettre:=place_lettre-4;
                      if place_lettre<=0 then
place_lettre:=26+place_lettre;
                      if place_lettre>26 then
place_lettre:=place_lettre-26;
                      mot2:=mot2+chr(place_lettre+96);
```

```
                end
                else mot2:=mot2+' ';
        end;
     decrypt:=mot2;
End;

Begin
     repeat
           clrscr;
           write('Entrer la phrase à crypter: ');
           readln(mot_a_crypter);
           cryptage(mot_a_crypter,mot_crypter);
           decryptage(mot_crypter,mot_decrypter);
           writeln('Message crypter selon la méthode
césarienne:');
           writeln(mot_crypter);
           writeln('Message décrypter selon la méthode
césarienne:');
           writeln(mot_decrypter);
     until readkey<>#13;
End.
```

Programme représentant le codage selon la méthode de substitution :

```
Program Cryptography_2;
Uses WinCrt;
Var
   mot_clef:string;
   mot_a_crypter:string;
   mot_crypter:string;
   mot_decrypter:string;
Procedure cryptage(mot:string;clef:string;var crypt:string);
Var
   i,j:integer;
   place_lettre:integer;
   place_lettre_clef:integer;
   mot2:string;
Begin
     j:=0;
     mot2:='';
     for i:=1 to length(mot) do
         begin
              j:=j+1;
              if j>length(clef) then j:=1;
              if mot[i]<>' ' then
                 begin
                      place_lettre:=ord(mot[i])-96;
                      place_lettre_clef:=ord(clef[j])-96;

place_lettre:=place_lettre+place_lettre_clef;
```

```
                    if place_lettre<=0 then
place_lettre:=26+place_lettre;
                    if place_lettre>26 then
place_lettre:=place_lettre-26;
                    mot2:=mot2+chr(place_lettre+96);
                end else mot2:=mot2+' ';
        end;
     crypt:=mot2;
End;
Procedure decryptage(mot:string;clef:string;var
decrypt:string);
Var
   i,j:integer;
   place_lettre:integer;
   place_lettre_clef:integer;
   mot2:string;
Begin
     j:=0;
     mot2:='';
     for i:=1 to length(mot) do
         begin
              j:=j+1;
              if j>length(clef) then j:=1;
              if mot[i]<>' ' then
                 begin
                      place_lettre:=ord(mot[i])-96;
                      place_lettre_clef:=ord(clef[j])-96;
                      place_lettre:=place_lettre-
place_lettre_clef;
                      if place_lettre<=0 then
place_lettre:=26+place_lettre;
                      if place_lettre>26 then
place_lettre:=place_lettre-26;
                      mot2:=mot2+chr(place_lettre+96);
                 end
                 else mot2:=mot2+' ';
         end;
     decrypt:=mot2;
End;

Begin
     repeat
           clrscr;
           write('Entrer la phrase à crypter: ');
           readln(mot_a_crypter);
           write('Entrer la clef de cryptage: ');
           readln(mot_clef);
           cryptage(mot_a_crypter,mot_clef,mot_crypter);
           decryptage(mot_crypter,mot_clef,mot_decrypter);
```

```
          writeln('Message crypter selon la méthode de
substitution:');
          writeln(mot_crypter);
          writeln('Message décrypter selon la méthode de
substitution:');
          writeln(mot_decrypter);
     until readkey<>#13;
End.
```

Programme représentant l'importance du code dans la méthode de substitution :

```
Program Cryptography_3;
Uses WinCrt;
Var
   mot_clef:string;
   mot_clef_2:string;
   mot_a_crypter:string;
   mot_crypter:string;
   mot_decrypter:string;
Procedure cryptage(mot:string;clef:string;var crypt:string);
Var
   i,j:integer;
   place_lettre:integer;
   place_lettre_clef:integer;
   mot2:string;
Begin
     j:=0;
     mot2:='';
     for i:=1 to length(mot) do
         begin
              j:=j+1;
              if j>length(clef) then j:=1;
              if mot[i]<>' ' then
                 begin
                      place_lettre:=ord(mot[i])-96;
                      place_lettre_clef:=ord(clef[j])-96;

place_lettre:=place_lettre+place_lettre_clef;
                      if place_lettre<=0 then
place_lettre:=26+place_lettre;
                      if place_lettre>26 then
place_lettre:=place_lettre-26;
                      mot2:=mot2+chr(place_lettre+96);
                 end else mot2:=mot2+' ';
         end;
     crypt:=mot2;
End;
Procedure decryptage(mot:string;clef:string;var
decrypt:string);
Var
```

```
   i,j:integer;
   place_lettre:integer;
   place_lettre_clef:integer;
   mot2:string;
Begin
     j:=0;
     mot2:='';
     for i:=1 to length(mot) do
         begin
              j:=j+1;
              if j>length(clef) then j:=1;
              if mot[i]<>' ' then
                 begin
                      place_lettre:=ord(mot[i])-96;
                      place_lettre_clef:=ord(clef[j])-96;
                      place_lettre:=place_lettre-
place_lettre_clef;
                      if place_lettre<=0 then
place_lettre:=26+place_lettre;
                      if place_lettre>26 then
place_lettre:=place_lettre-26;
                      mot2:=mot2+chr(place_lettre+96);
                 end
                 else mot2:=mot2+' ';
         end;
     decrypt:=mot2;
End;

Begin
     repeat
           clrscr;
           write('Entrer la phrase à crypter: ');
           readln(mot_a_crypter);
           write('Entrer la clef de cryptage: ');
           readln(mot_clef);
           repeat
                 clrscr;

cryptage(mot_a_crypter,mot_clef,mot_crypter);
                 writeln('Message crypter selon la méthode
de substitution:');
                 writeln(mot_crypter);
                 write('Entrer la clef pour voir le message:
');
                 readln(mot_clef_2);

decryptage(mot_crypter,mot_clef_2,mot_decrypter);
                 writeln('Message décrypter selon la méthode
de substitution:');
```

```
                writeln(mot_decrypter);
                if mot_clef_2<>mot_clef then readln;
           until mot_clef_2=mot_clef;
     until readkey<>#13;
End.
```

BIBLIOGRAPHIE :

Adresses Internet :

http://www.eleves.ens.fr:8080/home/madore

http://www.dmi.ens.fr/~www.grecc/Crypto/intro/.inti.dvi

Livres :

COGITATIONS : Quiz, mémos, formules et autres… en Sciences
Edition MASSON.

Un nouvel âge d'or, DEVLIN
Edition MASSON.

Utilitaires :

Encyclopédie® Microsoft® Encarta 97. © 1993-1996
Microsoft Corporation.

Encyclopædia Universalis France.

Borland Pascal v7.0 pour Windows™