Cryptographie

François Ducrot
http://math.univ-angers.fr

Décembre 2012

Terminologie

Cryptographie

Étude des méthodes permettant de transmettre des données de façon confidentielle.

Cryptanalyse

Étude des procédés cryptographiques du point de vue de l'attaquant.

La cryptologie regroupe ces deux activités, fortement indissociables.

Contenu

Deux exemples seront expérimentés grâce au logiciel libre de calcul formel Xcas :

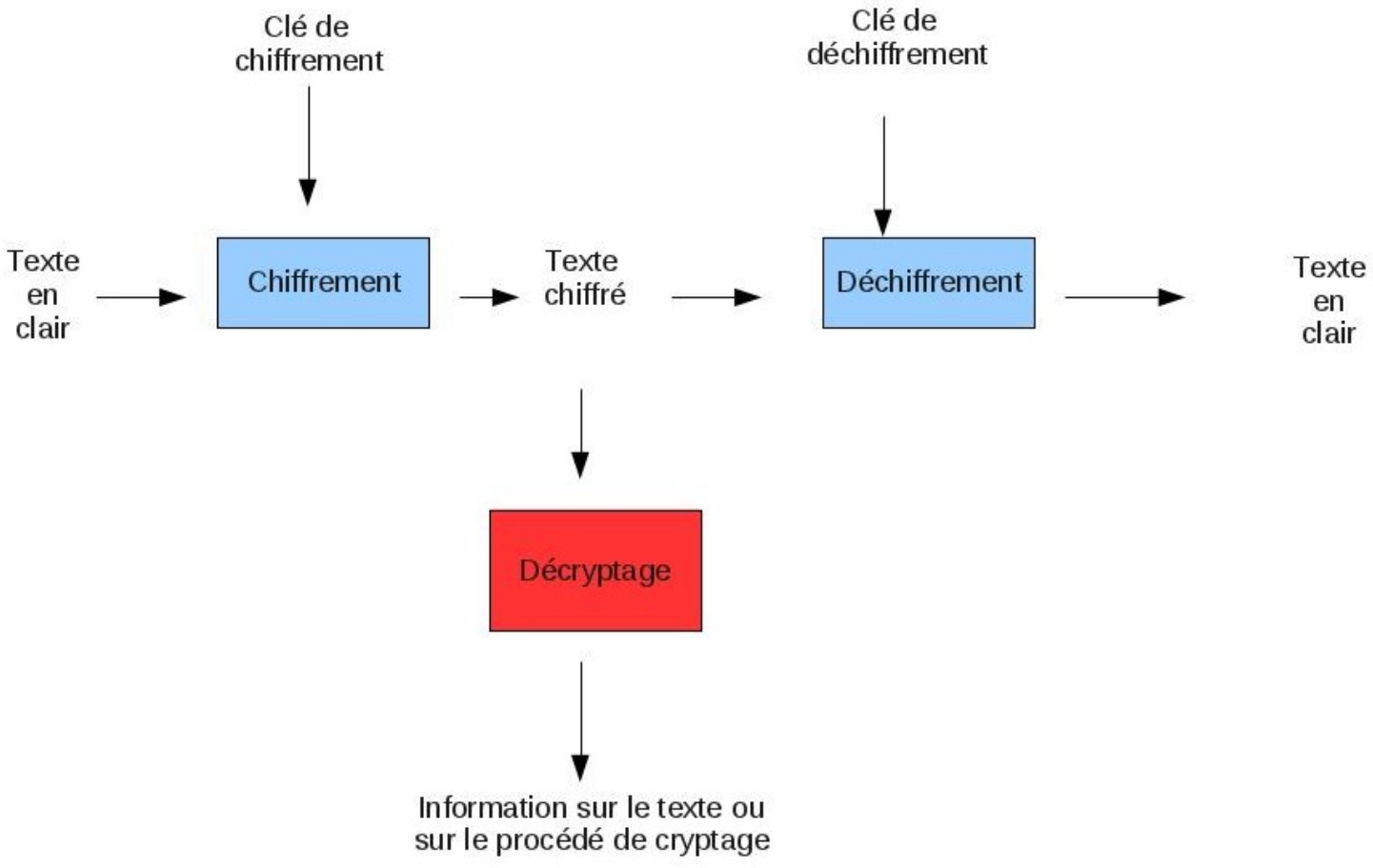
`http ://www-fourier.ujf-grenoble.fr/~parisse`

Questions traitées par la cryptographie

La cryptographie traite des problèmes de natures différentes :

- Cryptage d'un message
- Authentification d'un utilisateur
- Certification d'intégrité
- Signature

Cryptage



Chiffrement : transmettre un texte crypté.

Attaque : découvrir le texte en clair, ou la clé de décryptage.

Authentification

Exemple : les cartes de crédits.
La puce de la carte contient un certain nombre d'informations cryptées ; ces informations, plus le code rentré par l'utilisateur, certifient à la banque qu'il s'agit de l'utilisateur en titre.
Attaque : usurpation d'dentité
Principe : repose sur la notion de couple clé privée / clé publique.
Autre application : déclaration d'impots par internet

Certification d'intégrité

On récupère un fichier sur internet, par exemple un programme.
On veut être sûr que ce fichier n'a pas été modifié par une personne mal intentionnée, par exemple pour y introduire un virus ou un cheval de Troie.
Principe : repose sur la notion de fonction de hachage.

Signature

Deux interlocuteurs échangent un document (contrat).
Elles veulent qu'une tierce personne, supposée impartiale (notaire), puisse certifier que le document a été émis par l'un des deux interlocuteurs, et qu'il n'a pas été modifié depuis, ni par son auteur, ni par celui qui l'a reçu.
De plus, on peut souhaiter que la tierce personne n'ait pas connaissance du contenu du document.
Principe : repose sur la notion de fonction à sens unique.

Le chiffrement symétrique

Dans le chiffrement symétrique, la personne qui code un message et le destinataire qui va décoder le message codé partagent une même information secrète, la clé du code, qui doit être absolument gardée à l'abri des regards indiscrets.
Exemples :

- Le chiffre de César (codage mono-alphabétique)
- Les codages poly-alphabétiques
  - La machine enigma
  - Le codage D.E.S.
  - Le chiffre de Vernam

Le chiffre de César

On décale chaque lettre de cinq crans dans l'alphabet.

a	b	c	d	e	...	v	w	x	y	z
e	f	g	h	i	...	z	a	b	c	d

BONJOUR est transformé en FSRNSYV

Les chiffres mono-alphabétiques

- Le chiffre de César est l'exemple le plus simple de chiffre mono-alphabétique : une lettre est toujours envoyée sur la même lettre.
  Le code est entièrement déterminé par sa clé : le décalage utilisé (5 dans l'exemple). Il y a donc 26-1 = 26 clés utilisables.
- On peut envisager d'autres chiffres mono-alphabétiques. Un tel code aura au plus $26\,! = 403291461126605635584000000$ clés (nombre de façons de permuter 26 éléments).
- Les chiffres mono-alphabétiques sont très faciles à casser par analyse des fréquences des lettres.

Les chiffres poly-alphabétiques

- Les lettres sont regroupées par ensembles de k lettres. Dans le codage ascii, chaque symbole de l'anglais nord-américain est codé sur 8 bits, soit par un nombre compris entre 0 et $2^8 - 1 = 127$.
- Il y a donc $n = 128^k$ groupes possibles de k lettres. On associe donc à chacun de ces groupes un nombre compris entre 0 et $128^k - 1$.
- La clé secrète est le choix d'un permutation σ de $E = \{0, 1, \ldots, 128^k - 1\}$ et sa permutation inverse σ^{-1}. Il y a $(128^k)!$ permutations possibles.

Un exemple de code poly-alphabétique :

Prenons ici $k = 3$, et donc $n = 128^3 = 2097152$.

- On choisit un nombre p premier avec n. Le codage secret consiste ici à remplacer un nombre x par $y = p.x$ modulo n.
- On apprend en arithmétique à trouver un nombre q tel que $pq \equiv 1$ modulo n.

Expérimentation avec Xcas : Préparation

```
n:=128^3 ;
        2097152
p:=nextprime(123456);
        123457
q:=powmod(p,-1,n)
        994753
irem(p*q,n)
        1
```

Expérimentation avec Xcas : Codage ASCII

```
message:="Bonjour" :
decoupe(message);
        ["Bon","jou","r"]
asc("Bon")
        [66,111,110]
convert(asc("Bon"),base,128)
        1816514
convert(asc("jou"),base,128)
        1931242
convert(asc("r"),base,128)
        114
```

Le message `Bonjour` est donc traduit en la suite `x:=[1816514,1931242,114]`.

Expérimentation avec Xcas : Codage secret

On code le message numérique `x` en le multipliant par `p`, et en prenant le reste modulo `n`.

```
y:=irem(p*x,n);
        [322626,132714,1491186]
```

C'est le message codé. Le destinataire peut le décoder :

```
z:=irem(q*y,n)
        [1816514,1931242,114]
```

Expérimentation avec Xcas : traduction ASCII inverse

Il reste à revenir au message textuel

```
convert(z[0],base,128)
          [66,111,110]
char(convert(z[0],base,128))
          Bon
```

ou en mettant tout ensemble :

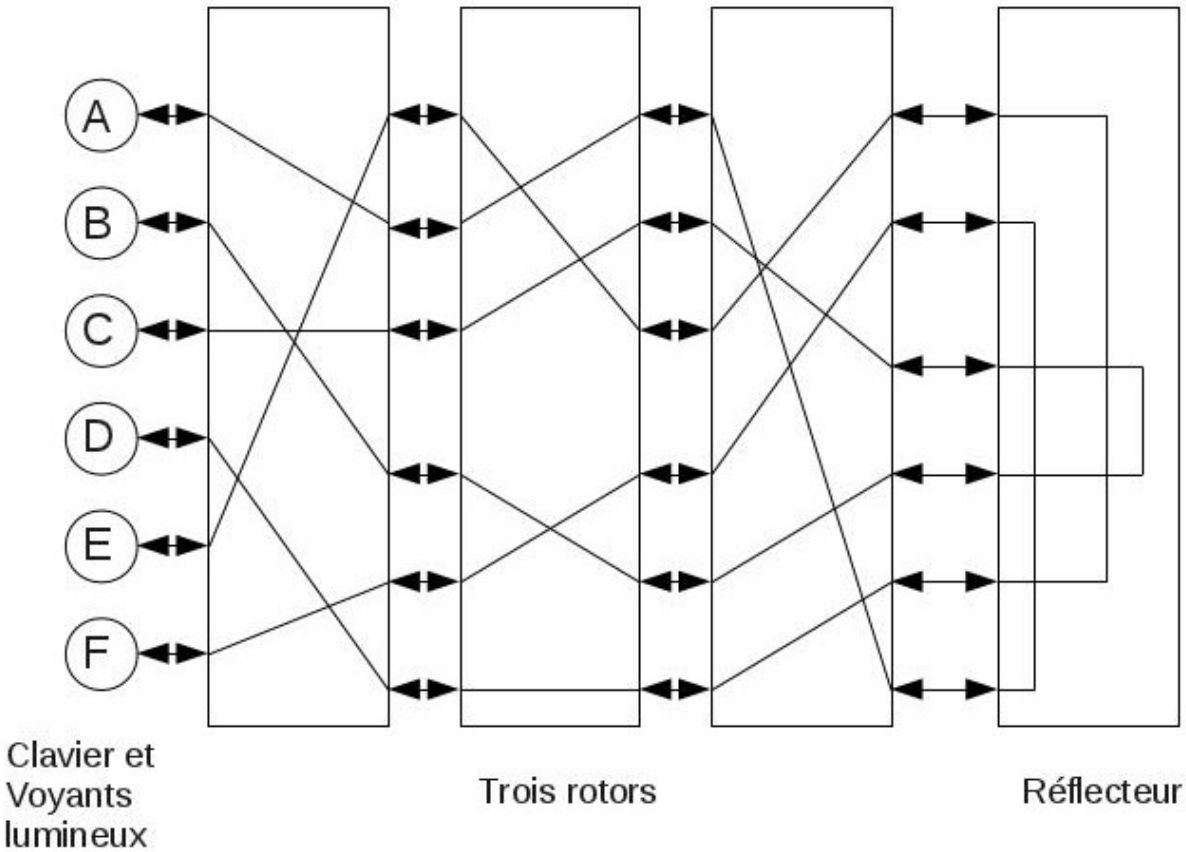
```
char(convert(z[0],base,128))+char(convert(z[1],base,128)) \
   +char(convert(z[2],base,128))

           Bonjour
```

La machine ENIGMA

- Inventée en 1919 et adoptée par l'armée allemande.
- Cryptanalysée par les services secrets polonais dès 1930.
- Utilisée par les Allemands pendant la seconde guerre mondiale, et décryptée systématiquement par les alliés, grâce aux travaux de mathématiciens comme A. Turing, et à l'utilisation de machines électromécaniques.

La machine ENIGMA - Schéma très simplifié



Ici, A est envoyé sur F, et réciproquement.

La machine ENIGMA - Description

- Utilise un procédé électromécanique
- La même machine effectue codage et décodage.
- A chaque frappe d'une lettre, un rotor tourne d'un cran, comme dans un compteur kilométrique ; au bout de 26^3 frappes, la machine revient à son état initial : chiffre poly-alphabétique à mots de 26^3 lettres.
- Clé secrète : La position initiale des trois rotors, soit 17576 clés possibles.
- Toutes les machines sont identiques, et donc vites connues par l'ennemi.
- Faiblesse : tout réside dans la transmission de la clé.

Le code DES

- Appel d'offre lancé par la NSA dans les années 70 pour un système de codage fiable et universel.
- Le D.E.S. (Data Encryption Standard), basé sur l'algorithme Lucifer de IBM, est un standard mondial jusqu'à récemment.
- Message découpé en blocs de 64 bits. Chaque bloc est codé indépendamment.
- Utilise une clé de codage de 64 bits (en réalité 56 bits utilisés, soit $\simeq 10^{17}$ clés possibles).
- Basé sur des permutations de bits et des ou exclusifs (XOR) avec des parties de la clé, le tout effectué 16 fois de suite.
- Clé jugée maintenant trop petite. Remplacement par d'autres protocoles...
- Mathématiques sous-jacentes : théorie des groupes

Code de Vernam à masque jetable

- La clé est une suite très longue de 0 et 1, dont chacun n'est utilisé qu'une fois (masque jetable).
- Le message codé est obtenu par ou exclusif à partir du message en clair et de la clé.

message en clair	1	0	1	1	1	0	0	1	0	1	0	1	0	0
clé	1	1	0	0	1	0	1	0	0	0	1	0	1	1
message codé	0	1	0	1	0	0	1	1	0	1	1	1	1	1

- La même clé est utilisée pour le décodage.
- C'est le seul code non cryptanalysable.
- Mais problème de communication de la clé.
- La clé doit être parfaitement aléatoire !
- Inventé en 1917. Utilisé pour sécuriser le "téléphone rouge", avec échange de clés par la valise diplomatique.

Codages symétriques et asymétriques

- Dans les codes décrits précédemment, la même clé est utilisée pour le codage et le décodage. C'est un secret partagé par l'émetteur et le receveur. Ces codes sont dits symétriques, ou à clé secrète.
- Le problème essentiel des codes symétriques est l'échange des clés entre interlocuteurs.
- A l'inverse, on peut envisager qu'il y ait deux clés : une clé publique connue de tous, permettant de coder et une clé privée permettant à son possesseur, et à lui seulement, de décoder les messages qui lui sont adressés.
- On parle alors de codes asymétriques. Il n'y a plus de problème d'échange des clés.

Fonctions de hachage

- Soit X l'ensemble de tous les messages envisagés.
- Une fonction de hachage sur X est une fonction $f : X \longrightarrow Y$, à valeur dans un ensemble Y de petite taille qui vérifie :
  - f est facile à calculer par tous (avec un ordinateur)
  - Si x est un élément de X, on ne sait pas pratiquement trouver un élément x' tel que $f(x) = f(x')$.
- Si je veux envoyer un message x de telle façon que mon interlocuteur soit assuré que ce message n'a pas été modifié par quelqu'un d'autre, je calcule $y = f(x)$, puis j'envoie séparément x et y.
- Mon interlocuteur reçoit un message x' par le réseau. Il calcule alors $f(x')$.
  - si $f(x') = y$, il peut raisonnablement accepter le message.
  - si $f(x') \neq y$, il est assuré que le message x' est faux
- Les fonctions de hachage sont utilisées pour le stockage des mots de passe sur un ordinateur

Fonction de hachage : la somme MD5

- MD5 associe à un message de taille quelconque une chaîne de 32 caractères hexadécimaux (128 bits), appelée empreinte du message.
- L'agorithme MD5 est entièrement publique.
- Bien sur, il y a beaucoup plus de messages que de chaînes de 128 bits ; donc plusieurs messages ont la même empreinte.
- Repose sur le fait qu'on ne sait pas pratiquement construire deux objets qui ont la même empreinte (ou plutôt la confiance en ce fait).
- MD5 n'est plus considéré comme suffisamment fiable : on a réussi à construire deux messages x et x' qui ont la même empreinte (on parle de collision).
- MD5 est progressivement remplacé par SHA-256, RIPEMD-160 ou Whirlpool, qui utilisent des empreintes plus longues.

Un exemple d'utilisation de MD5

Un exemple :

```
 -> du -sh csgCrypto.tex
8,0K csgCrypto.tex
  ->md5sum csgCrypto.tex
d667a49b0b628a97a7527ebfc4a48a9b  csgCrypto.tex
```

Ajoutons maintenant un espace blanc à la fin du fichier :

```
 -> echo " " >> csgCrypto.tex
 -> du -sh csgCrypto.tex
8,0K csgCrypto.tex
 -> md5sum csgCrypto.tex
d7f4ea6af88c35cb4a1704ebfcacf1a0  csgCrypto.tex
```

Fonctions à sens unique

- **Fonction à sens unique** : c'est une fonction $f : X \longrightarrow Y$ bijective telle qu'il est facile de calculer l'image $y = f(x)$ d'un élément de X, mais qu'il est algorithmiquement long de chercher un élément x de X tel que $f(x) = y$.
- **Fonction à sens unique et à information cachée** : c'est une fonction $f : X \longrightarrow Y$ telle qu'il est facile de calculer l'image $y = f(x)$ d'un élément de X, mais qu'il est algorithmiquement long de chercher un élément x de X tel que $f(x) = y$, à moins de connaitre une information supplémentaire I_f sur la fonction f.
- Une telle fonction permet de définir un code asymétrique : Alice fabrique une telle fonction f et son information cachée I_f. Elle publie f (clé publique) et garde I_f secrète (clé privée).
- Si Bob veut envoyer un message x à Alice, il calcule $y = f(x)$, et envoie y à Alice. Seule Alice peut alors retrouver x à partir de y à l'aide de sa clé privée.

Jouer à pile ou face au téléphone

- Alice et Bob jouent à pile ou face au téléphone :
  - Alice lance la pièce et annonce "J'ai lancé la pièce"
  - Bob annonce "Face"
  - Alice lui répond "Pas de chance, c'était Pile !"
- Comment Bob peut-il être sûr que Alice n'a pas triché ?
- On peut proposer une méthode grâce à une fonction à sens unique !

Jouer à pile ou face : fonction à sens unique

- Alice et Bob conviennent ensemble d'un grand ensemble d'entiers X et d'une fonction à sens unique $f : X \longrightarrow Y$.
- Alice choisit un nombre x dans X. Elle calcule $y = f(x)$ et envoie y à Bob.
- Bob choisit "Pair" ou "Impair" et l'annonce à Alice.
- Alice lui dit si x était pair ou impair. Pour justifier sa réponse, elle envoie x à Bob.
- Bob vérifie quelle n'a pas triché en calculant $f(x)$ et en le comparant à y.

Encore faut-il que

- X soit assez grand et contienne autant de nombres pairs que de nombres impairs,
- la connaissance de $f(x)$ ne permette pas d'en déduire la parité de x.

Le code RSA

Ce code a été imaginé par Rivest, Shamir, Adelman en 1978. Il est utilisé pour

- l'authentification des cartes bancaires : la clé secrète est contenue dans la puce de la carte
- permettre à deux interlocuteurs de s'échanger une clé secrète. Avec cette clé secrète, ils vont continuer à dialoguer en utilisant un procédé symétrique.

Il est la base du protocole SSH de connexion sécurisée sur internet.

Pour aller plus loin, il est indispensable de parler un peu d'arithmétique.

L'outil mathématique qui fait marcher tout ça est l'arithmétique.

Arithmétique modulaire

$\mathbb{Z}/n\mathbb{Z}$: ensemble des restes de la division par n.

Exemple : $\mathbb{Z}/5\mathbb{Z} = \{0, 1, 2, 3, 4\}$

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

$4 + 4 = 3 + 5$

x	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

$3 \times 4 = 2 + 2 \times 5$

Tous les éléments non nuls ont un inverse : $1 \to 1$, $2 \to 3$, $3 \to 2$, $4 \to 4$

Un exemple plus compliqué : $\mathbb{Z}/15\mathbb{Z}$

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
2	0	2	4	6	8	10	12	14	1	3	5	7	9	11	13
3	0	3	6	9	12	0	3	6	9	12	0	3	6	9	12
4	0	4	8	12	1	5	9	13	2	6	10	14	3	7	11
5	0	5	10	0	5	10	0	5	10	0	5	10	0	5	10
6	0	6	12	3	9	0	6	12	3	9	0	6	12	3	9
7	0	7	14	6	13	5	12	4	11	3	10	2	9	1	8
8	0	8	1	9	2	10	3	11	4	12	5	13	6	14	7
9	0	9	3	12	6	0	9	3	12	6	0	9	3	12	6
10	0	10	5	0	10	5	0	10	5	0	10	5	0	10	5
11	0	11	7	3	14	10	6	2	13	9	5	1	12	8	4
12	0	12	9	6	3	0	12	9	6	3	0	12	9	6	3
13	0	13	11	9	7	5	3	1	14	12	10	8	6	4	2
14	0	14	13	12	11	10	9	8	7	6	5	4	3	2	1

Seuls 1, 2, 4, 7, 8, 11, 13, 14 ont des inverses. Ce sont les éléments qui ne sont divisibles ni par 3, ni par 5 ; ils sont premiers avec 15.

Une fonction à sens unique

On remarque qu'il est très facile, et très rapide, de calculer une puissance, même élevée, d'un élément de $\mathbb{Z}/n\mathbb{Z}$.
Exemple : Dans $\mathbb{Z}/563\mathbb{Z}$, calculons 253^{1024}.
Comme $1024 = 2^{10}$, il suffit de calculer :

$$253^2 = 390\,,\ 390^2 = 90\,,\ 90^2 = 218\,,\ 218^2 = 232\,,\ 232^2 = 339$$

$$339^2 = 69\,,\ 69^2 = 257\,,\ 257^= = 178\,,\ 178^2 = 156\,,\ 156^2 = 127$$

Donc $253^{1024} = 127$ dans $\mathbb{Z}/563\mathbb{Z}$.
Si n est un nombre très grand, fixons un entier r, la fonction

$$f_r\ :\ \mathbb{Z}/n\mathbb{Z} \to \mathbb{Z}/n\mathbb{Z}\ ,\ x \mapsto x^r$$

se calcule très facilement.

Une fonction à sens unique - suite

En revanche, si on connait seulement n et $y \in \mathbb{Z}/n\mathbb{Z}$, pour trouver un élément $x \in \mathbb{Z}/n\mathbb{Z}$ tel que $f_r(x) = y$, il faut tester successivement toutes les valeurs de x, jusqu'à trouver celle qui convient. Quand n est très grand, c'est impraticable !

f_r est donc une fonction à sens unique.

Si on choisit astucieusement n et r, f_r est une fonction à sens unique et à information cachée :
On choisit deux nombres premiers p et q très grands, et on calcule $n = pq$. A partir de la donnée de p et q, on calcule $\phi = (p-1)(q-1)$, et on choisit un nombre r premier avec ϕ.
En appliquant un peu d'arithmétique, on trouve alors facilement in nombre s tel que

$$\text{pour tout } x \in \mathbb{Z}/n\mathbb{Z},\ (x^r)^s = x^{rs} = (x^s)^r = x$$

Donc si je connais p et q, à partir de $y = x^r$, je sais retrouver x (en calculant y^s).

Le codage RSA

- Je choisis les nombres premiers p et q. Je calcule $\phi = (p-1)(q-1)$, et je choisis un entier r premier avec ϕ.
- Je rends public le couple (n, r). C'est la clé publique.
- A l'aide de ϕ, je trouve l'entier s que je garde secret. C'est ma clé secrète.
- N'importe qui peut coder un message en utilisant la fonction f_r, mais seul moi suis capable de décoder le message, en utilisant f_s.
- Tout ceci repose sur le fait qu'on ne sait pas pratiquement retrouver les nombres p et q à partir $n = pq$.
- Si la technologie des ordinateur change, ou si de nouvelles méthodes mathématiques sont découvertes, tout peut changer...

Expérimentation avec Xcas : clé publique

Je prends deux grands nombres premiers :

```
p:=nextprime(157796525649746346341546 4) ;
                157796525649746346341555 3
q:=nextprime(685974568721666566326515 7) ;
                685974568721666566326522 7
n:=p*q
      1082444036283621460851342895067417077726795587553 1
r:=nextprime(7764647975649654)
                7764647975649701
```

Ma clé publique est la donnée de (n, r).

Expérimentation avec Xcas : clé privée

Je fabrique maintenant ma clé privée.

```
phi:=(p-1)*(q-1)
      10824440362836214608513420512963227063138829194752
s:=iegcd(r,phi)[0]
      32654092461740943679525898315606068274670672220 61
```

Ma clé privée est le nombre s.

Expérimentation avec Xcas : codage/décodage

```
message:="Bonjour":
liste:=asc(message)
      [66,111,110,106,111,117,114]
messagenum:=convert(liste,base,128)
      505427412105154
messagecode:=powmod(messagenum,r,n)
    641935888361853276903482006372026579406469741327 3
messagedecode:=powmod(messagecode,s,n)
      505427412105154
listedecode:=convert(messagedecode,base,128)
    [66,111,110,106,111,117,114]
char(listedecode)
    Bonjour
```

Dans la vraie vie, il faut travailler avec des nombres beaucoup plus grands : clé de 2048 bits, soit deux nombres premiers p et q à 300 chiffres chacun !

Une clé ssh

Voici une clé de 2048 bits, fabriquée par `openssh` :

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpQIBAAKCAQEA28BnhnkffDd5pBKBOOHBF3UVWpN8vIergh2kw8VxL8dDn5/Y
JEpHs7+anlU+/JExLYMhKqiyhWocQzdRwuW9jEdsuRvIniimMt7J4RehBxIFki7m
uJUHVVcxiBHjjc4yUJXXK2HuJCWz77OG/OQ8hVTTlSbpGYgqrCSNY16THoZ8swDn
V1QJmbHspRRsEp2P3aPZ3UrUUrmfaxaVGHyxTLd1oBl6yMziRY54g2vXLGsARCfh
FFGsootV+FR8NDB8wB8jW0cFOB86HOAmSRnMpr9mFHYhBtQUicCHR2fEXw3piOAa
uRgfLbKAEItQ8B53HSEpuSKQA8dbMyxPTmq7NQIDAQABAoIBAQCIucWv3VzBpKKC
eyD9BYRmW+rVWD6b7q2/X0JMrGQoHgrtHj8VFUJXjDdfnBp8ernmX1gbeqrYZ5T+
HUU5xkeRKD91TNwdy5zcudFRw6nlBEmDErtqzQ6rAn1G9Bpo+IpFqXUfAOdyhiLW
WRT78+XT71tVycfuotIqGPP/h7BxpIR5rhtybuhwaOKAz9gdz7YqUa7ClnwmyjCK
uKDR2z5dzPJqVrcP5LcxGgg8sNcAOoy2h4ZZI7NIg76qGjJUcgXlTOFifyc6drpA
J/k5tMbRgG9SaQVKoqRp1jQcsVl4OrA7rylknXp9Y2uLYDrQusowEQ6FnPACYZ4L
pwt2gEdhAoGBAO7s1gaLEDGYhZwMqT/FMFPu/amELpvWe+aeX5biMA7zGlAqhobh
swt14jLi9a/WsShKwztKc8OHsPhikX9wB5cSz7yAWT6D1mXO/TUa2ONXGCuc2/Q/
8mEoyeboclL3QkjCGnn+PMMLLoek9nEDznxtksYrZwhw0W2OqWAejIJ3AoGBAOt0
ySg3IV73EccESd4dqcf/rTazLpxzj/7iQRdcCyN5PDWi67FP8sTx0ZTKV8tPo48X
DMW6VZIJlNvD6eJoBMUvbWUNDPUxrYCI8BexdJcaRKFclCZTNmMXEKf8ruM08OR9
pbPkc2YC4vDxteJgzPj1zS4Ff7nN0Jt+LZtGfw6zAoGBAJMuqpnVSOCEiKoc0pZK
Lv9FllW668tsUV876L7WD6zl7Su9RqoIl3dT40hdr+PmabPcenAn6TBmgopAjkjn
BoX2PKpulouAQ67r1+fC5iEGIgmb05OfYPyhV987ClI5p0eS6ySDjYLJfJxq7rWo
FqdoCKazTgFRHnJqfHyN1Rj5AoGAIRmI0cN2SIBZGm4q6DOpU3lI42CrE8lXG9C4
r8z7r9/yK+QcWA/SJAeOm+kaEHXAm3P1ErLQJrh8Gazf6zkOOp4hNcxw3uM97czS
W7JRnpBNx5VZKk4WXjW23Laz6zLdDnHvYmuiIVwUhSSlmg5bokADuQ2C2B7rzYAe
by6o2+sCgYEAiBXEOhcbZTap230N2TWk2mNqnfldgPaV5xqzvTjimlJ6ddhMxMhj
cwHMa4iy4ru0nDr65sDn368WX41Nq4XzrcY62/e/zDhukJgiKBbeRRLuGHsouKEN
Gyyv+VC1MvLYKXXKsUHFidHPc/5oLDZd68t0lPlEjejV5kqe1DKW120=
-----END RSA PRIVATE KEY-----
```