Sécurité des systèmes d'information

Mehdi Haddad
mehdi.haddad@u-pec.fr

2017 - 2018

Qu'est ce que la cryptographie

- La cryptographie ou science du secret est un art très ancien, c'est :
  l'art de remplacer un secret encombrant par un secret miniature
- Le secret encombrant est le contenu du message il est remplacé par un petit secret qui est la clef de déchiffrement dont la taille est en général de quelques centaines à quelques milliers de bits à comparer aux mégabits d'un message.

Qu'est ce que la cryptographie

- La cryptographie est l'art de rendre inintelligible, de crypter, de coder, un message pour ceux qui ne sont pas habilités à en prendre connaissance. Le chiffre, le code est le procédé, l'algorithme, la fonction, qui permet de crypter un message.
- La cryptanalyse est l'art pour une personne non habilitée, de décrypter, de décoder, de déchiffrer, un message. C'est donc l'ensemble des procédés d'attaque d'un système cryptographique.
- La cryptologie est l'ensemble formé de la cryptographie et de la cryptanalyse.

Vocabulaire

- Texte clair : information dont la confidentialité n'est pas protégée
- Texte chiffré : information protégée
- Clef : paramètre secret d'un algorithme cryptographique
- Protocole : protocole qui garantit des fonctions de sécurité via l'utilisation de primitives cryptographiques.

Buts principaux de la cryptographie

- Confidentialité : Seuls les légitimes ont accès à l'information
- Intégrité : Le message n'a pas été altéré
- Authenticité : On communique bien à la bonne personne
- Non-répudiation : On ne peut pas nier sa participation

D'autres mécanismes participent également à ces fonctions, l'usage de la cryptographie seule est inutile.

Principes de Kerckhoffs

- Auguste Kerckhoffs (1835-1903) posa les principes de la cryptographie moderne.
- Ces principes et en particulier le second stipulent entre autre que la sécurité d'**un cryptosystème ne doit pas reposer sur le secret de l'algorithme** de codage mais qu'elle doit uniquement reposer sur la clef secrète du cryptosysème qui est un paramètre facile à changer, de taille réduite (actuellement de 64 à 2048 bits suivant le type de code et la sécurité demandée) et donc assez facile à transmettre secrètement.

Méthodes historiques de cryptographie

- Codes à répertoire
- Codes de permutation
- Codes mono-alphabétique

Codes à répertoire

- Ils consistent en un dictionnaire qui permet de remplacer certains mots par des mots différents.
- Ils sont très anciens et ont été utilisés intensivement jusqu'au début du $20^{ème}$ siècle.
- Ils ont fait l'objet d'une critique sévère de A. Kerchkoffs dans son article fondateur.

Codes à répertoire : Exemple

- On peut par exemple créer le dictionnaire suivant :
  rendez-vous $\leftrightarrow$ 175
  demain $\leftrightarrow$ oiseaux
  midi $\leftrightarrow$ à vendre
  Créteil $\leftrightarrow$ au marché
- La phrase en clair : RENDEZ VOUS DEMAIN MIDI CRÉTEIL
- Devient : 175 OISEAUX À VENDRE AU MARCHÉ

Codes à répertoire

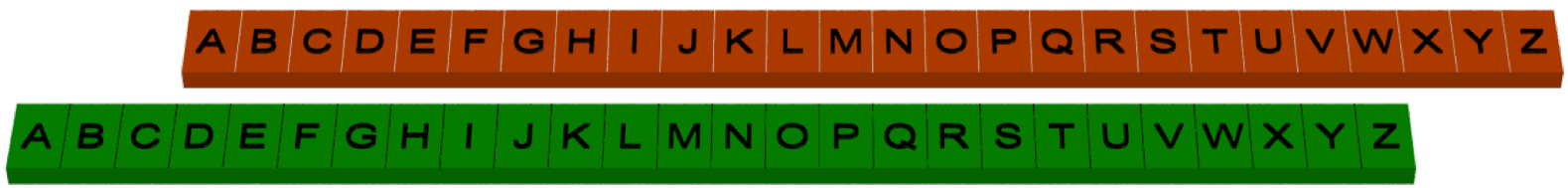
- Il faut disposer de dictionnaires qui prévoient toutes les possibilités.
- Donc, sauf si on se restreint à transmettre des informations très limitées, la taille du dictionnaire s'accroit démesurément.
- Tout changement du code nécessitait l'envoi de documents volumineux avec un risque d'interception non négligeable.
- Ces codes manquent de souplesse ils ne permettent pas de coder des mots nouveaux sans un accord préalable entre l'expéditeur et le destinataire.
- Pour cela il faut qu'ils échangent des documents ce qui accroît le risque d'interception du code.
- Ils ne sont pas adaptés à des usages intensifs entre de nombreux correspondants.

Codes de substitution

- Dans les codes de substitution par flots ou par blocs l'ordre des lettres est conservé mais on les remplace par des symboles d'un nouvel alphabet suivant un algorithme précis.
- Exemple : code de César.
- D'après Suetone, dans son ouvrage "Vie des douze Césars", Jules César pendant la guerre des Gaules avait utilisé le code de substitution par flot suivant :
  lettre codée= lettre claire+3 modulo 26

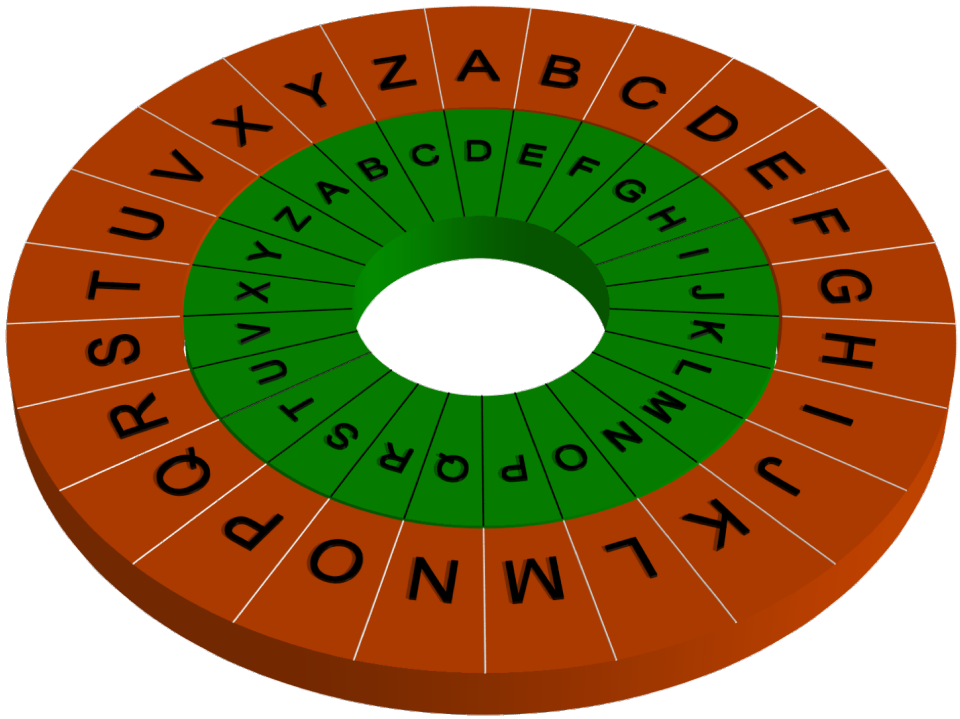
Le chiffrement de César

- Jules César a-t-il vraiment prononcé la célèbre phrase : DOHD MDFWD HVW
- En fait César, pour ses communications importantes à son armée, cryptait ses messages.
- Ce que l'on appelle le chiffrement de César est un décalage des lettres : pour crypter un message, A devient D , B devient E , C devient F ,..
- Voici une figure avec l'alphabet d'origine en haut et en rouge , en correspondance avec l'alphabet pour le chiffrement en-dessous et en vert .



Le chiffrement de César

- Pour prendre en compte aussi les dernières lettres de l'alphabet, il est plus judicieux de représenté l'alphabet sur un anneau.
- Ce décalage est un décalage circulaire sur les lettres de l'alphabet.



Le chiffrement de César

- Pour déchiffrer le message de César, il suffit de décaler les lettres dans l'autre sens, D se déchiffre en A, E en B,...
- Et la célèbre phrase de César est : ALEA JACTA EST
- Traduite du latin cette phrase donne « Les dés sont jetés ».

Modélisation mathématique

- Il est plus facile de manipuler des nombres que des lettres, aussi nous passons à une formulation mathématique.
- Nous associons à chacune des 26 lettres de A à Z un nombre de 0 à 25.
- En termes mathématiques, nous définissons une bijection :

$$f : \{A, B, C, \ldots, Z\} \longrightarrow \{0, 1, 2, \ldots, 25\}$$

$$A \longmapsto 0 \quad B \longmapsto 1 \quad C \longmapsto 2 \quad \ldots \quad Z \longmapsto 25$$

- Ainsi "A L E A" devient "0 11 4 0".
- Le chiffrement de César est un cas particulier de **chiffrement mono-alphabétique**, c'est-à-dire un chiffrement lettre à lettre.

Rappel : modulo

- Soit n $>$ 2 un entier fixé. On dit que a est congru à b modulo n, si n divise b - a.
- On note alors : $a \equiv b \pmod{n}$.
- Exemples :
  - Pour nous n = 26. Ce qui fait que $28 \equiv 2 \pmod{26}$, car 28 - 2 est bien divisible par 26.
  - De même $85 = 3 \times 26 + 7$ donc $85 \equiv 7 \pmod{26}$.
- On note $\mathbb{Z}/26\mathbb{Z}$ l'ensemble de tous les éléments de $\mathbb{Z}$ modulo 26.
- Cet ensemble peut par exemple être représenté par les 26 éléments $\{0, 1, 2, ..., 25\}$.
- En effet, puisqu'on compte modulo 26 : 0, 1, 2, . . . , 25, puis $26 \equiv 0$, $27 \equiv 1$, $28 \equiv 2$, . . . , $52 \equiv 0$, $53 \equiv 1$, . . . et de même $-1 \equiv 25$, $-2 \equiv 24$,...
- Plus généralement $\mathbb{Z}/n\mathbb{Z}$ contient n éléments.

Rappel modulo : addition

- De façon naturelle l'addition et la multiplication d'entiers se transposent dans $\mathbb{Z}/n\mathbb{Z}$.
- Par exemple dans $\mathbb{Z}=26\mathbb{Z}$, 15 + 13 égale 2. En effet $15 + 13 = 28 \equiv 2 \pmod{26}$.
- Autre exemple : que vaut 133 + 64 ?
- $133+64 = 197 = 7 \times 26+15 \equiv 15 \pmod{26}$.
- Mais on pourrait procéder différemment : tout d'abord $133 = 5\times26+3 \equiv 3 \pmod{26}$ et $64 = 2 \times 26 + 12 \equiv 12 \pmod{26}$.
- Et maintenant sans calculs : $133 + 64 \equiv 3 + 12 \equiv 15 \pmod{26}$.

Rappel modulo : multiplication

- De façon naturelle l'addition et la multiplication d'entiers se transposent dans $\mathbb{Z}/n\mathbb{Z}$.
- Par exemple 3×12 donne 10 modulo 26, car $3 \times 12 = 36 = 1 \times 26 + 10 \equiv 10 \pmod{26}$.
- De même : $3 \times 27 = 81 = 3 \times 26 + 3 \equiv 3 \pmod{26}$.
- Une autre façon de voir la même opération est d'écrire d'abord $27 = 1 \pmod{26}$ puis $3 \times 27 \equiv 3 \times 1 \equiv 3 \pmod{26}$.

Chiffrer et déchiffrer

- Le chiffrement de César est simplement une addition dans $\mathbb{Z}/26\mathbb{Z}$.
- Fixons un entier k qui est le décalage (par exemple k = 3 dans l'exemple de César précédent).
- La fonction de chiffrement de César de décalage k qui va de l'ensemble $\mathbb{Z}/26\mathbb{Z}$ dans lui-même est définie comme suit :

$$C_k : \begin{cases} \mathbb{Z}/26\mathbb{Z} & \longrightarrow & \mathbb{Z}/26\mathbb{Z} \\ x & \longmapsto & x+k \end{cases}$$

- Par exemple, pour k = 3 :$C_3(0) = 3$, $C_3(1) = 4$. . .

Chiffrer et déchiffrer

- La fonction de déchiffrement de César de décalage k qui va de l'ensemble $\mathbb{Z}/26\mathbb{Z}$ dans lui-même est définie comme suit :

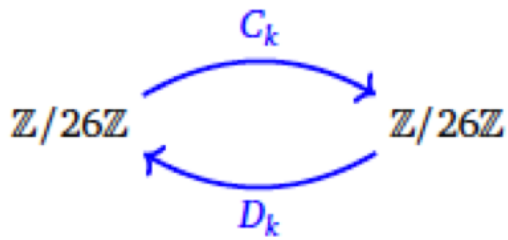
$$D_k : \begin{cases} \mathbb{Z}/26\mathbb{Z} & \longrightarrow & \mathbb{Z}/26\mathbb{Z} \\ x & \longmapsto & x-k \end{cases}$$

- Si 1 a été chiffré en 4, par la fonction C_3 alors $D_3(4) = 4 - 3 = 1$. On retrouve bien le nombre originel.
- Mathématiquement, D_k est la bijection réciproque de C_k, ce qui implique que pour tout x $\in \mathbb{Z}/26\mathbb{Z}$:

$$D_k(C_k(x)) = x$$

Chiffrer et déchiffrer

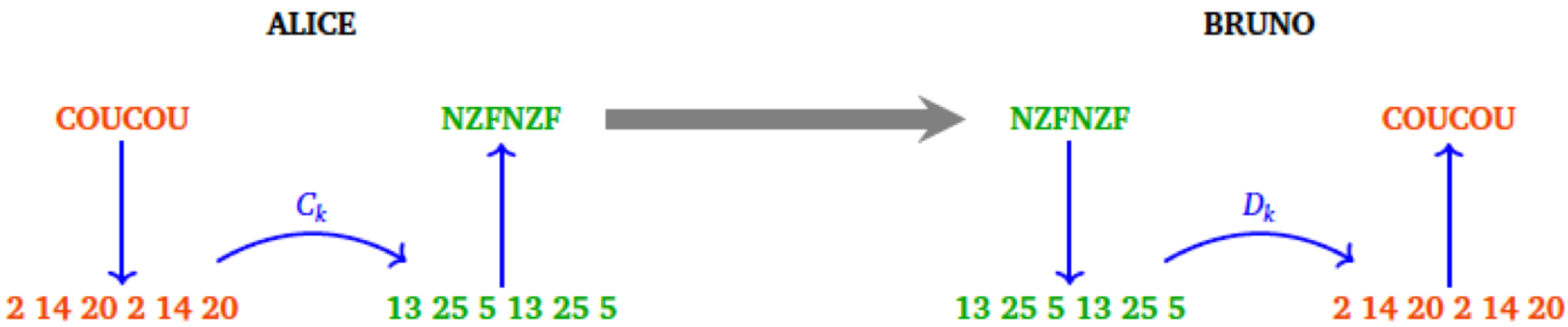
- En d'autres termes, si x est un nombre, on applique la fonction de chiffrement pour obtenir le nombre crypté y = C_k(x).
- La fonction de déchiffrement fait bien ce que l'on attend d'elle D_k(y) = x, on retrouve le nombre original x.



Exemple

- Alice veut envoyer des messages secrets à Bruno.
- Ils se sont d'abord mis d'accord sur une clé secrète k, par exemple k = 11.
- Alice veut envoyer le message "COUCOU" à Bruno.

Exemple


ALICE
BRUNO
COUCOU
NZFNZF
NZFNZF
COUCOU
C_k
D_k
2 14 20 2 14 20
13 25 5 13 25 5
13 25 5 13 25 5
2 14 20 2 14 20

Espace des clés et attaque

- Combien existe-t-il de possibilités de chiffrement par la méthode de César ?
- Il y a 26 fonctions C_k différentes, k = 0,1, . . . ,25.
- Encore une fois, k appartient à $\mathbb{Z}/26\mathbb{Z}$, car par exemple les fonctions C_{29} et C_3 sont identiques.
- Le décalage k s'appelle la clé de chiffrement, c'est l'information nécessaire pour crypter le message. Il y a donc 26 clés différentes et l'espace des clés est $\mathbb{Z}/26\mathbb{Z}$.

Espace des clés et attaque

- Il est clair que ce chiffrement de César est d'une sécurité très faible.
- Si Alice envoie un message secret à Bruno et que Chloé intercepte ce message.
- Il sera facile pour Chloé de le décrypter même si elle ne connaît pas la clé secrète k.
- L'attaque la plus simple pour Chloé est de tester ce que donne chacune des 26 combinaisons possibles et de reconnaître parmi ces combinaisons laquelle donne un message compréhensible.

Chiffrement mono-alphabétique

- Le chiffrement de César présente une sécurité très faible car l'espace des clés est trop petit : il y a seulement 26 clés possibles.
- On peut attaquer un message chiffré en testant toutes les clés à la main.
- Dans un chiffrement mono-alphabétique, au lieu de faire correspondre circulairement les lettres, on associe maintenant à chaque lettre une autre lettre (sans ordre fixe ou règle générale).

Exemple

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

- Pour crypter le message : ETRE OU NE PAS ETRE TELLE EST LA QUESTION
- On regarde la correspondance et on remplace la lettre E par la lettre X, puis la lettre T par la lettre G, puis la lettre R par la lettre K...
- Le message crypté est alors : XGKX DR SX OFV XGKX GXWWX XVG WF ZRXVGPDS
- Pour le décrypter, **en connaissant les substitutions**, on fait l'opération inverse.

Chiffrement mono-alphabétique

- Avantage : l'espace des clés est gigantesque ce qui rend l'énumération toutes les possibilités très coûteux.
- Inconvénients : la clé à retenir est beaucoup plus longue, puisqu'il faut partager la clé constituée des 26 lettres "FQBMX...". Mais surtout, nous allons voir que ce protocole de chiffrement est relativement simple à « craquer ».

Espace de clés

- Mathématiquement, le choix d'une clé revient au choix d'une bijection de l'ensemble $\{A, B, ..., Z\}$ vers le même ensemble $\{A, B, ..., Z\}$.
- Il y a 26 ! choix possibles. En effet pour la lettre A de l'ensemble de départ, il y a 26 choix possibles (nous avions choisi F), pour B il reste 25 choix possibles (tout sauf F qui est déjà choisi), pour C il reste 24 choix... enfin pour Z il ne reste qu'une seule possibilité, la seule lettre non encore choisie.
- Au final il y a : $26 \times 25 \times 24 \times ... \times 2 \times 1$ soit 26 ! choix de clés.
- Ce qui fait environ 4×10^{26} clés. Il y a plus de clés différentes que de grains de sable sur Terre !
- Si un ordinateur pouvait tester 1 000 000 de clés par seconde, il lui faudrait alors 12 millions d'années pour tout énumérer.

Attaque statistique

- La principale faiblesse du chiffrement mono-alphabétique est qu'une même lettre est toujours chiffrée de la même façon.
- Par exemple, ici E devient X. Dans les textes longs, les lettres n'apparaissent pas avec la même fréquence.
- Ces fréquences varient suivant la langue utilisée. En français, les lettres les plus rencontrées sont dans l'ordre : E S A I N T R U L O D C P M V Q G F H B X J Y Z K W
- Exemple de fréquence (qui peut dépendre de l'échantillon) :

E	S	A	I	N	T	R	U	L	O	D
14.69%	8.01%	7.54%	7.18%	6.89%	6.88%	6.49%	6.12%	5.63%	5.29%	3.66%

Attaque statistique

- Dans le texte crypté, on cherche la lettre qui apparaît le plus, et si le texte est assez long cela devrait être le chiffrement du E, la lettre qui apparaît ensuite dans l'étude des fréquences devrait être le chiffrement du S, puis le chiffrement du A...
- On obtient des morceaux de texte clair sous la forme d'une texte à trous et il faut ensuite deviner les lettres manquantes.
- Par exemple, déchiffrons la phrase :
  LHLZ HFQ BC HFFPZ WH YOUPFH MUPZH

Attaque statistique

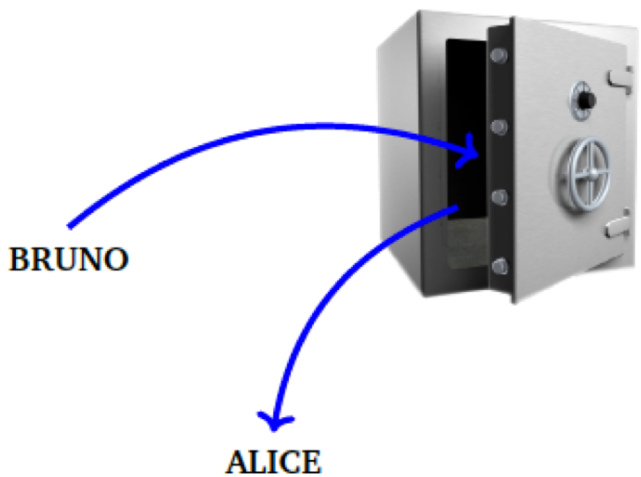
- On compte les apparitions des lettres : H : 6 F : 4 P : 3 Z : 3
- On suppose donc que le H crypte la lettre E, le F la lettre S, ce qui donne : *E** ES* ** ESS** *E ***SE ****E
- D'après les statistiques P et Z devraient se décrypter en A et I (ou I et A). Le quatrième mot "HFFPZ", pour l'instant décrypté en "ESS**", se complète donc en "ESSAI" ou "ESSIA". La première solution semble correcte ! Ainsi P crypte A, et Z crypte I.
- La phrase est maintenant : *E*I ES* ** ESSAI *E ***ASE **AIE
- En réfléchissant un petit peu, on décrypte le message : CECI EST UN ESSAI DE PHRASE VRAIE

Cryptographie modene

- Chiffrement à clé privée
- Chiffrement à clé publique

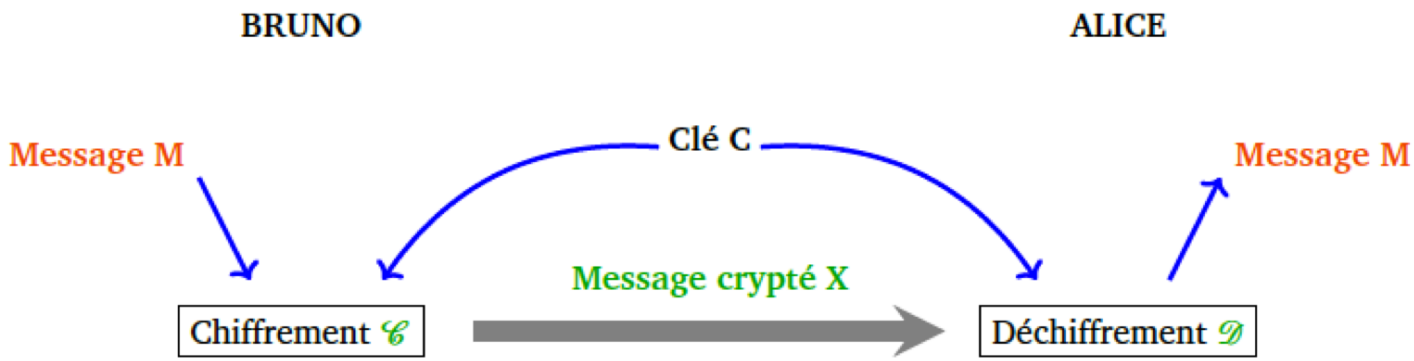
Chiffrement à clé privée

- Les protocoles précédents étaient des chiffrements à clé privée. De façon imagée, tout se passe comme si Bruno pouvaient déposer son message dans un coffre fort pour Alice.
- Alice et Bruno étant les seuls à posséder la clé du coffre.



Chiffrement à clé privée

- les deux interlocuteurs se partagent une même clé qui sert à chiffrer (et déchiffrer) les messages.
- Cela pose bien sûr un problème majeur : Alice et Bruno doivent d'abord se communiquer la clé.



Chiffrement à clé privée : DES

- Le DES (Data Encryption Standard) est un protocole de chiffrement par blocs.
- Il a été, entre 1977 et 2001, le standard de chiffrement pour les organisations du gouvernement des États-Unis et par extension pour un grand nombre de pays dans le monde.
- Nous allons présenter une version très simplifiée

Chiffrement à clé privée : DES

- Pour changer, nous allons travailler modulo 10. Lorsque l'on travaille par blocs, les additions se font bit par bit.
- Par exemple : $[1\ 2\ 3\ 4] \oplus [7\ 8\ 9\ 0] = [8\ 0\ 2\ 4]$ car $(1 + 7 \equiv 8 \pmod{10}$, $2 + 8 \equiv 0 \pmod{10}$,...)
- Notre message est coupé en blocs, pour nos explications ce seront des blocs de longueur 8. La clé est de longueur 4.
- Voici le message (un seul bloc) : $M = [1\ 2\ 3\ 4\ 5\ 6\ 7\ 8]$ et voici la clé : $C = [3\ 1\ 3\ 2]$.

DES

Étape 0. Initialisation. On note $M_0 = M$ et on découpe M en une partie gauche et une partie droite

$$M_0 = [G_0 \parallel D_0] = [1\ 2\ 3\ 4 \parallel 5\ 6\ 7\ 8]$$

Étape 1. Premier tour. On pose

$$M_1 = [D_0 \parallel C \oplus \sigma(G_0)]$$

où σ est une permutation circulaire.
On effectue donc trois opérations pour passer de M_0 à M_1 :

1. On échange la partie droite et la partie gauche de M_0 :

$$M_0 \longmapsto [5\ 6\ 7\ 8 \parallel 1\ 2\ 3\ 4]$$

2. Sur la nouvelle partie droite, on permute circulairement les nombres :

$$\longmapsto [5\ 6\ 7\ 8 \parallel 2\ 3\ 4\ 1]$$

3. Puis on ajoute la clé secrète C à droite (ici $C = [3\ 1\ 3\ 2]$) :

$$\longmapsto [5\ 6\ 7\ 8 \parallel 5\ 4\ 7\ 3] = M_1$$

DES

On va recommencer le même processus. Cela revient à appliquer la formule de récurrence, qui partant de $M_i = [G_i \parallel D_i]$, définit

$$M_{i+1} = [D_i \parallel C \oplus \sigma(G_i)]$$

Étape 2. Deuxième tour. On part de $M_1 = [5\ 6\ 7\ 8 \parallel 5\ 4\ 7\ 3]$.

1. On échange la partie droite et la partie gauche de M_0 :

$$M_0 \longmapsto [5\ 4\ 7\ 3 \parallel 5\ 6\ 7\ 8]$$

2. Sur la nouvelle partie droite, on permute circulairement les nombres.

$$\longmapsto [5\ 4\ 7\ 3 \parallel 6\ 7\ 8\ 5]$$

3. Puis on ajoute la clé secrète C à droite.

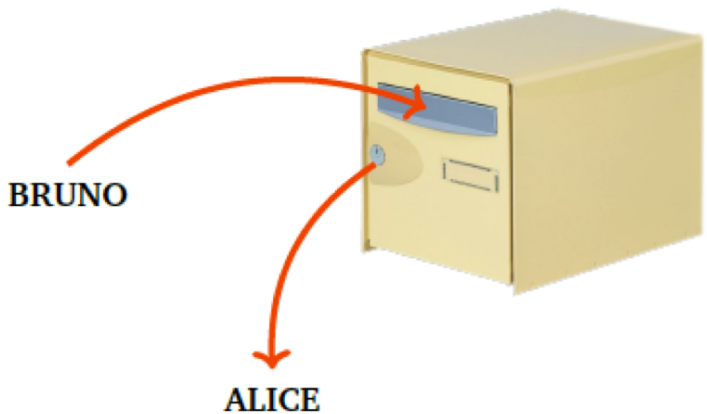
$$\longmapsto [5\ 4\ 7\ 3 \parallel 9\ 8\ 1\ 7] = M_2$$

DES

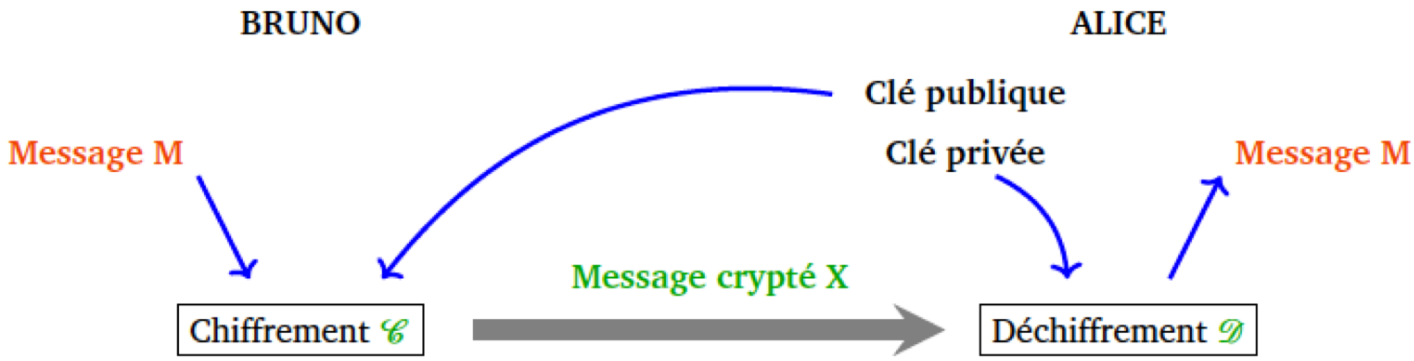
- On peut décider de s'arrêter après ce tour et renvoyer le message crypté X = M2 = [5 4 7 3 9 8 1 7].
- Comme chaque opération élémentaire est inversible, on applique un protocole inverse pour déchiffrer.
- Dans le vrai protocole du DES, la clé principale est de taille 64 bits, il y a plus de manipulations sur le message et les étapes mentionnées ci-dessus sont effectuées 16 fois (on parle de tours).
- À chaque tour, une clé différente est utilisée. Il existe donc un préambule à ce protocole : générer 16 clés secondaires (de longueur 48 bits) à partir de la clé principale.

Chiffrement à clé publique

- De façon imagée, si Bruno veut envoyer un message à Alice, il dépose son message dans la boîte aux lettres d'Alice.
- Seule Alice pourra ouvrir sa boîte et consulter le message. Ici la clé publique est symbolisée par la boîte aux lettre.
- Tout le monde peut y déposer un message, la clé qui ouvre la boîte aux lettres est la clé privée d'Alice, que Alice doit conserver à l'abri.



Chiffrement à clé publique



L'arithmétique pour RSA

- Pour un entier n, sachant qu'il est le produit de deux nombres premiers, il est difficile de retrouver les facteurs p et q tels que $n = pq$.
- Le principe du chiffrement RSA, chiffrement à clé publique, repose sur cette difficulté.
- Dans cette partie nous mettons en place les outils mathématiques nécessaires pour le calcul des clés publique et privée ainsi que les procédés de chiffrement et déchiffrement RSA.

L'arithmétique pour RSA

- L'algorithme d'Euclide qui repose sur le principe que pgcd(a, b) = pgcd(b, a (mod b)).

```
def euclide(a,b):
    while b !=0 :
        a , b = b , a % b
    return a
```

L'arithmétique pour RSA

- Calcul des coefficients de Bézout u, v tels que au + bv = pgcd(a, b).

```
def euclide_etendu(a,b):
    x = 1 ; xx = 0
    y = 0 ; yy = 1

   while b != 0 :
       q = a // b
       a , b = b , a % b
       xx , x = x - q*xx , xx
       yy , y = y - q*yy , yy
   return (a,x,y)
```

- Cet algorithme renvoie d'abord le pgcd, puis les coefficients u, v tels que au + bv = pgcd(a, b).

L'arithmétique pour RSA

▶ Inverse modulo n

Soit $a \in \mathbb{Z}$, on dit que $x \in \mathbb{Z}$ est un *inverse de a modulo n* si $\boxed{ax \equiv 1 \pmod{n}.}$
Trouver un inverse de a modulo n est donc un cas particulier de l'équation $ax \equiv b \pmod{n}$.

Proposition 1. • *a admet un inverse modulo n si et seulement si a et n sont premiers entre eux.*
- *Si $au + nv = 1$ alors u est un inverse de a modulo n.*

En d'autres termes, trouver un inverse de a modulo n revient à calculer les coefficients de Bézout associés à la paire (a, n).

L'exponentiation rapide

- Nous aurons besoin de calculer rapidement des puissances modulo n.
- Pour cela il existe une méthode beaucoup plus efficace que de calculer d'abord ak puis de le réduire modulo n.
- Il faut garder à l'esprit que les entiers que l'on va manipuler ont des dizaines voir des centaines de chiffres.
- Voyons la technique sur l'exemple de 511 (mod 14). L'idée est de seulement calculer 5, $5^2, 5^4, 5^8$... et de réduire modulo n à chaque fois. Pour cela on remarque que $11 = 8 + 2 + 1$ donc $5^{11} = 5^8 \times 5^2 \times 5^1$.

L'exponentiation rapide : Exemple

On souhaite calculer $5^{11} \pmod{14}$

$$5 \equiv 5 \pmod{14}$$
$$5^2 \equiv 25 \equiv 11 \pmod{14}$$
$$5^4 \equiv 5^2 \times 5^2 \equiv 11 \times 11 \equiv 121 \equiv 9 \pmod{14}$$
$$5^8 \equiv 5^4 \times 5^4 \equiv 9 \times 9 \equiv 81 \equiv 11 \pmod{14}$$

$$5^{11} \equiv 5^8 \times 5^2 \times 5^1 \equiv 11 \times 11 \times 5 \equiv 11 \times 55 \equiv 11 \times 13 \equiv 143 \equiv 3 \pmod{14}$$

Le chiffrement RSA

Pour crypter un message on commence par le transformer en un –ou plusieurs– nombres. Les processus de chiffrement et déchiffrement font appel à plusieurs notions :

- On choisit deux nombres premiers p et q que l'on garde secrets et on pose $n = p \times q$. Le principe étant que même connaissant n il est très difficile de retrouver p et q (qui sont des nombres ayant des centaines de chiffres).
- La clé secrète et la clé publique se calculent à l'aide de l'algorithme d'Euclide et des coefficients de Bézout.
- Les calculs de cryptage se feront modulo n.
- Le déchiffrement fonctionne grâce à une variante du petit théorème de Fermat.

Le chiffrement RSA

Dans cette section, c'est Bruno qui veut envoyer un message secret à Alice. La processus se décompose ainsi :

1. Alice prépare une clé publique et une clé privée,
2. Bruno utilise la clé publique d'Alice pour crypter son message,
3. Alice reçoit le message crypté et le déchiffre grâce à sa clé privée.

Calcul de la clé publique et de la clé privée

Alice effectue, une fois pour toute, les opérations suivantes (en secret) :

- elle choisit deux nombres premiers distincts p et q (dans la pratique ce sont de très grand nombres, jusqu'à des centaines de chiffres),
- Elle calcule n = p × q,
- Elle calcule φ(n) = (p - 1) × (q - 1).

Exemples

Exemple 1.

- p = 5 et q = 17
- n = p × q = 85
- φ(n) = (p - 1) × (q - 1) = 64

Le calcul de φ(n) n'est possible que si la décomposition de n sous la forme p × q est connue. D'où le caractère secret de φ(n) même si n est connu de tous.

Exemple 2.

- p = 101 et q = 103
- n = p × q = 10 403
- φ(n) = (p - 1) × (q - 1) = 10 200

Choix d'un exposant et calcul de son inverse

Alice continue :

- elle choisit un exposant e tel que pgcd(e, φ(n)) = 1,
- elle calcule l'inverse d de e module φ(n) : d $\times$ e $\equiv$ 1 (mod φ(n)).
- Ce calcul se fait par l'algorithme d'Euclide étendu.

Exemples

Exemple 1.

- Alice choisit par exemple e = 5 et on a bien pgcd(e, φ(n)) = pgcd(5, 64) = 1,
- Alice applique l'algorithme d'Euclide étendu pour calculer les coefficients de Bézout correspondant à pgcd(e, φ(n)) = 1. Elle trouve 5 × 13 + 64 × (-1) = 1.
- Donc 5 × 13 ≡ 1 (mod 64) et l'inverse de e modulo φ(n) est d = 13.

Exemple 2.

- Alice choisit par exemple e = 7 et on a bien pgcd(e, φ(n)) = pgcd(7, 10 200) = 1,
- L'algorithme d'Euclide étendu pour pgcd(e, φ(n)) = 1 donne 7 × (-1457) + 10 200 × 1 = 1. Mais -1457 ≡ 8743 (mod φ(n)), donc pour d = 8743 on a d × e ≡ 1 (mod φ(n)).

Clé publique

- La clé publique d'Alice est constituée des deux nombres : **n et e**
- Et comme son nom l'indique Alice communique sa clé publique au monde entier.
- Exemple 1 : n = 85 et e = 5
- Exemple 2 : n = 10 403 et e = 7

Clé privée

- Alice garde pour elle sa clé privée : **d**
- Alice détruit en secret p, q et φ(n) qui ne sont plus utiles. Elle conserve secrètement sa clé privée.
- Exemple 1. d = 13
- Exemple 2. d = 8743

Chiffrement du message

Bruno veut envoyer un message secret à Alice. Il se débrouille pour que son message soit un entier (quitte à découper son texte en bloc et à transformer chaque bloc en un entier).

- Le message est un entier m, tel que $0 \leq m < n$.
- Exemple 1. Bruno veut envoyer le message $m = 10$.
- Exemple 2. Bruno veut envoyer le message $m = 1234$.

Bruno récupère la clé publique d'Alice : n et e avec laquelle il calcule, à l'aide de l'algorithme d'exponentiation rapide, le message chiffré :

$$x \equiv m^e (mod\ n)$$

Exemple 1

m = 10, n = 85 et e = 5
Donc : $x \equiv m^e (mod\ n) \equiv 10^5 (mod\ 85)$
On peut ici faire les calculs à la main :

- $10^2 \equiv 100 \equiv 15 (mod\ 85)$
- $10^4 \equiv (10^2)^2 \equiv 15^2 \equiv 225 \equiv 55 (mod\ 85)$
- $x \equiv 10^5 \equiv 10^4 \times 10 \equiv 55 \times 10 \equiv 550 \equiv 40 (mod\ 85)$

Le message chiffré est donc x = 40.

Exemple 2

m = 1234, n = 10 403 et e = 7
Donc : $x \equiv m^e (mod\ n) \equiv 1234^7 (mod 10\ 403)$
On utilise l'ordinateur pour obtenir que x = 10 378.

Déchiffrement du message

Lemme de déchiffrement
Soit d l'inverse de e modulo φ(n).

$$Si\ x \equiv m^e (mod\ n)\ alors\ m \equiv x^d (mod\ n).$$

Alice reçoit le message x chiffré par Bruno, elle le décrypte à l'aide de sa clé privée d, par l'opération :

$$m \equiv x^d (mod\ n)$$

Exemple 1

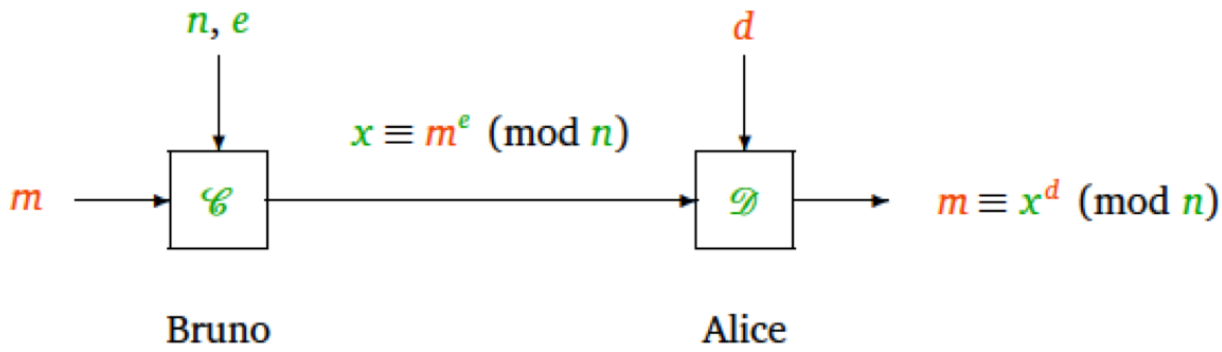
c= 40, d = 13, n = 85
Donc : $x^d \equiv (40)^{13}(mod\ 85)$.
$40^{13} \equiv (mod\ 85)$ on note que $13 = 8 + 4 + 1$, donc
$40^{13} = 40^8 \times 40^4 \times 40$.
$40^2 \equiv 1600 \equiv 70(mod\ 85)$
$40^4 \equiv (40^2)^2 \equiv 70^2 \equiv 4900 \equiv 55(mod\ 85)$
$40^8 \equiv (40^4)^2 \equiv 55^2 \equiv 3025 \equiv 50(mod\ 85)$

Donc $x^d \equiv 40^{13} \equiv 40^8 \times 40^4 \times 40 \equiv 50 \times 55 \times 40 \equiv 10(mod\ 85)$
qui est bien le message m de Bruno.

Exemple 2

c = 10 378, d = 8743, n = 10 403.
On calcule par ordinateur $x^d \equiv (10378)^{8743} (mod\ 10403)$ qui vaut exactement le message original de Bruno m = 1234.

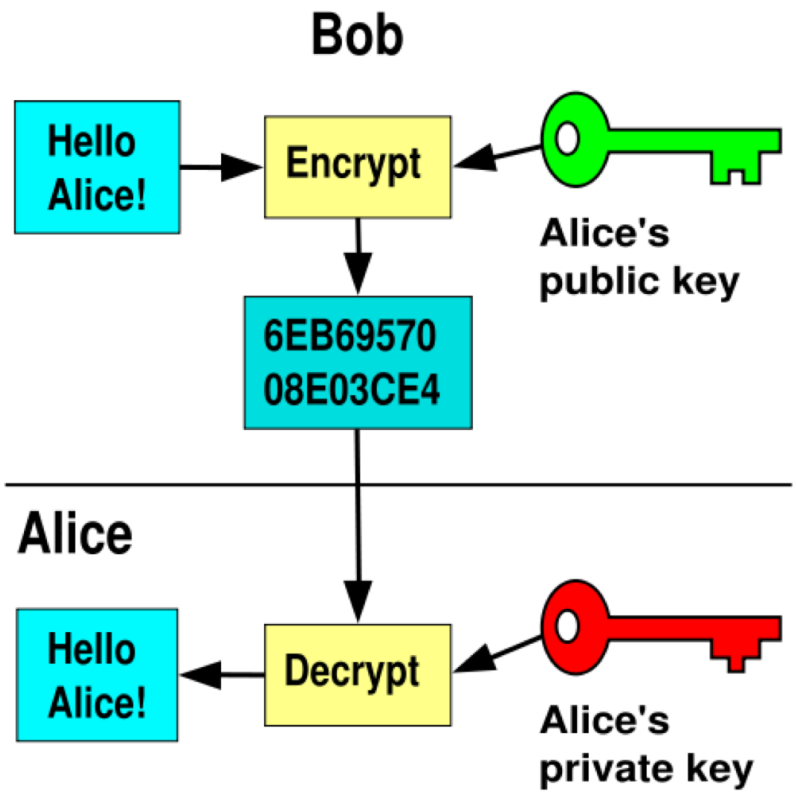
Pour résumer



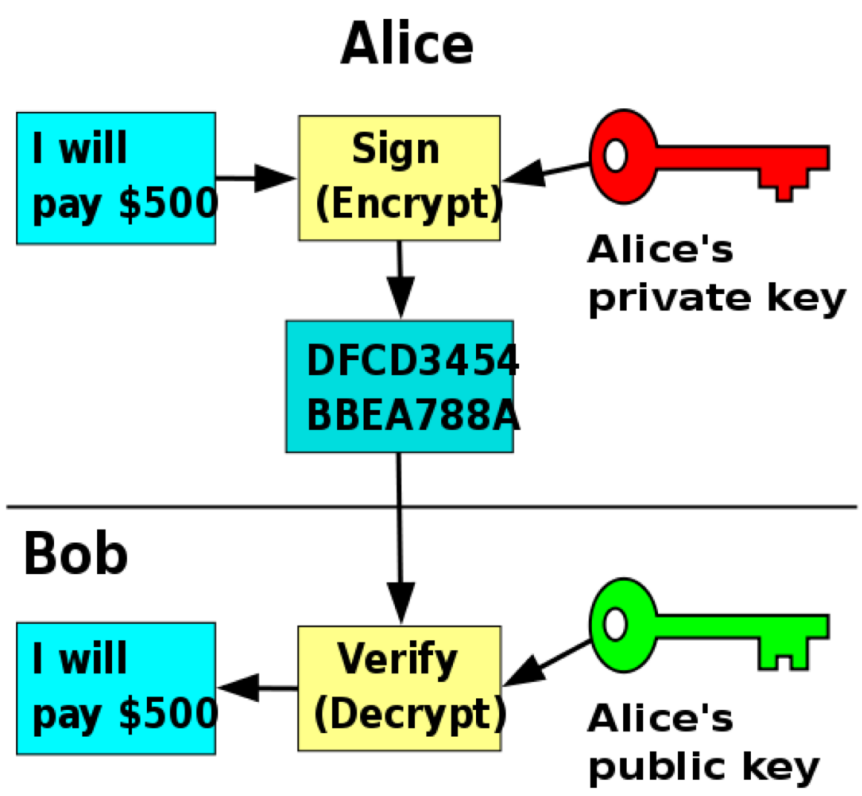
Clés d'Alice :

- publique : n, e
- privée : d

Chiffrement et signature



Chiffrement



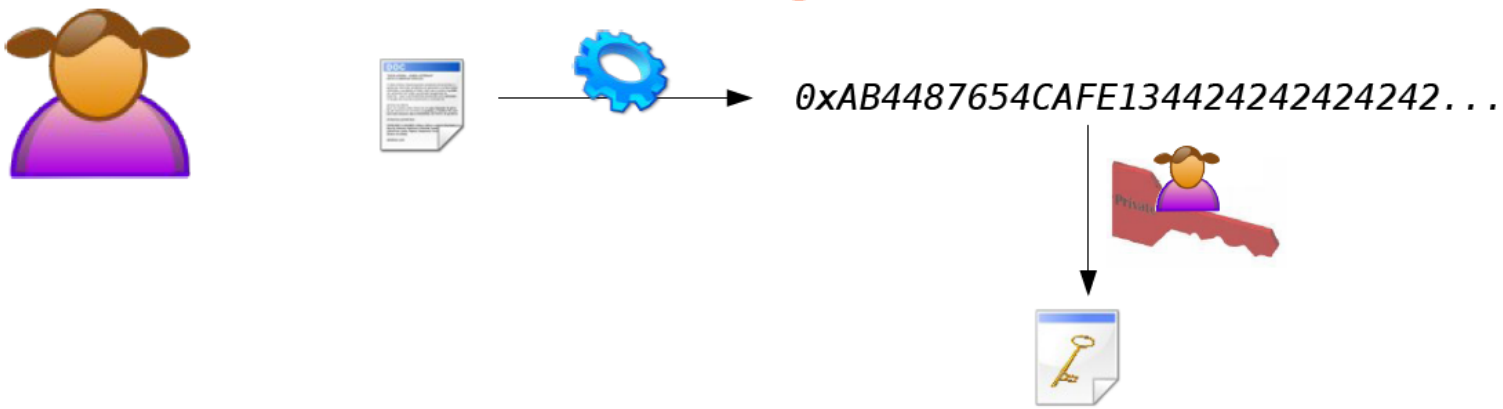
SIgnature

Fonctions de hachage

- Une fonction de hachage est un procédé à sens unique permettant d'obtenir une suite d'octets (une empreinte) caractérisant un ensemble de données.
- Pour tout ensemble de données de départ, l'empreinte obtenue est toujours la même.

Signer un document

- La signature d'un document utilise à la fois la cryptographie asymétrique et les fonctions de hachage
- Imaginons que Alice souhaite envoyer un document signé à Bob.
  - Tout d'abord, elle génére l'empreinte du document au moyen d'une fonction de hachage.
  - Puis, elle crypte cette empreinte avec sa clé privée.



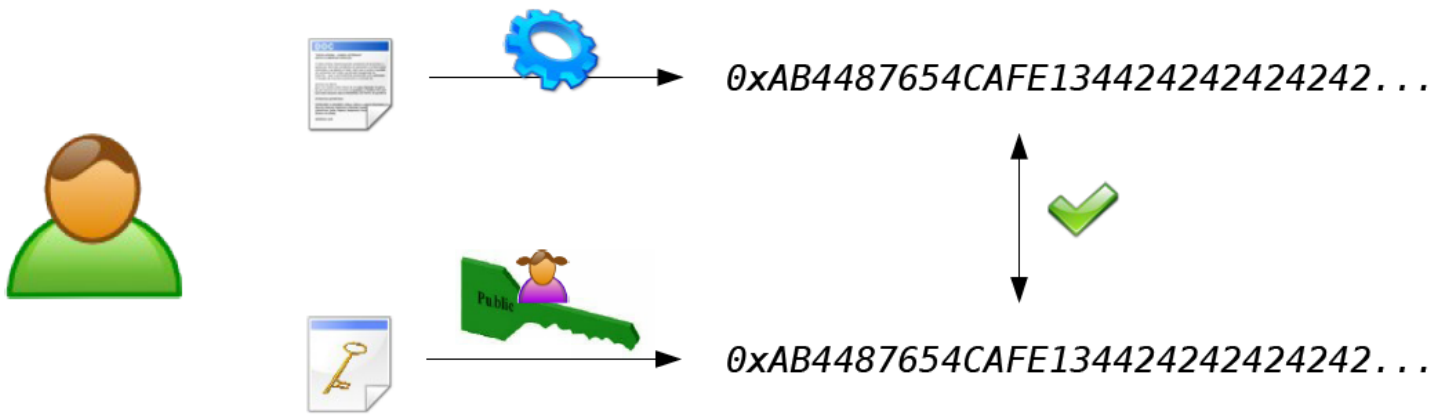
Signer un document

- Elle obtient ainsi la signature de son document. Elle envoie donc ces deux éléments à Bob

- Pour vérifier la validité du document, Bob doit tout d'abord déchiffrer la signature en utilisant la clé publique d'Alice. Si cela ne fonctionne pas, c'est que le document n'a pas été envoyé par Alice.
- Ensuite, Bob génère l'empreinte du document qu'il a reçu, en utilisant la même fonction de hachage qu'Alice (On supposera qu'ils suivent un protocole établi au préalable).

Signer un document

- Puis, il compare l'empreinte générée et celle issue de la signature.



- Si les deux empreintes sont identiques, la signature est validée

OpenSSL

OpenSSL est une boite à outils cryptographiques qui offre :

- une bibliothèque de programmation en C permettant de réaliser des applications client/serveur sécurisées s'appuyant sur SSL/TLS.
- une commande en ligne (openssl) permettant
- la création de clés RSA, DSA (signature)
- la création de certificats X509 ;
- le calcul d'empreintes (MD5, SHA, RIPEMD160, . . .) ;
- le chiffrement et déchiffrement (DES, IDEA, RC2, RC4, Blowfish, . . .) ;
- la réalisation de tests de clients et serveurs SSL/TLS ;
- la signature et le chiffrement de courriers (S/MIME)

Pour connaître toutes les fonctionnalités de openSSL, consultez le manuel (man openssl ou openssl [commande] -help). La syntaxe générale de la commande openssl est : $openssl commande <options>

Génération de clés

On peut générer une paire de clés RSA avec la commande genrsa (pour generate an RSA private key) :

```
$openssl genrsa -out fichierdesortie taille
```

où fichierdesortie est le nom du fichier de sauvegarde de la clé, et taille est la taille souhaitée (exprimée en bits) du modulus de la clé.

Visualisation des clés RSA

La commande rsa permet de visualiser le contenu d'un fichier contenant une paire de clés RSA :

```
$openssl rsa -in cleaVisualiser -text -noout
```

où cleaVisualiser est le fichier de sauvegarde des clés au format .pem par défaut.
Si l'on souhaite visualiser uniquement la clé publique, alors il faudra rajouter l'option -pubin.
L'option -text demande l'affichage décodé de la paire de clés.
L'option -noout supprime la sortie normalement produite par la commande rsa. Les différents éléments de la clé sont affichés en hexadécimal (hormis l'exposant public).
On peut distinguer le modulus, l'exposant public (qui par défaut est toujours 655371), l'exposant privé, les nombres premiers facteurs du modulus, ainsi que trois autres nombres qui servent à optimiser l'algorithme de déchiffrement.

Chiffrement d'un fichier de clés RSA

Il est possible de chiffrer une paire de clés avec la commande rsa. L'algorithme de chiffrement symétrique utilisé est spécifié en option (voir le manuel pour connaîtres les algorithmes de chiffrements possibles) :

```
$openssl rsa -in maCle <algoChiffrement> -out maCle
```

Un mot de passe est demandé deux fois pour générer une clé symétrique protégeant l'accès à la clé. On peut aussi directement générer une clé dans un fichier crypté, en utilisant la commande genrsa :

```
$openssl genrsa <algoChiffrement> -out maCle taille
```

Exportation de la partie publique

Afin de communiquer sa clé publique (sans compromettre sa clé privée), on exporte la clé publique à partir d'une paire de clés avec la commande rsa et l'option -pubout :

```
$openssl rsa -in maCle -pubout -out maClePublique
```

Signature de fichiers

Avant de signer un message, il est essentiel de calculer son empreinte. La commande dgst calcule l'empreinte du message en utilisant la fonction de hachage précisée en option (voir le manuel pour connaitre les fonctions de hachage possibles). On fera bien-sûr attention à ne pas utiliser -md5 ni -sha1 pour des applications cryptographiques !

```
$openssl dgst <hachage> -out fichierHashSortie
fichierAHacher
```

Signature de fichiers

Pour signer un document, on signe son empreinte. Pour celà, on utilise l'option -sign de la commande rsautl (pour RSA utility) :

```
$openssl rsautl -sign -in fichierMessageHash -inkey
fichierCle  -out fichierSignature
```

et on utilise l'option verify de la commande rsautl pour vérifier la signature :

```
$openssl rsautl -verify -in fichierSignature -pubin
-inkey fichierCle  -out fichierHashResultat
```

Il reste ensuite à vérifier que l'empreinte ainsi produite est la même que celle du message, que l'on calcule avec dgst. On pourra par exemple utiliser la commande diff de linux pour comparer les deux fichiers ainsi calculés. L'option -pubin indique que le fichier clé utilisé pour la vérification est celui contenant uniquement la partie publique de la clé utilisée pour la signature.