Initiation à PGP : GnuPG

par Bourgeois Morgan

SOMMAIRE

1 PRÉSENTATION

PGP (Pretty Good Privacy) est un programme de cryptage créé par Philip Zimmermann. La première version de PGP date de 1991. Il est très rapidement devenu l'un des cryptosystèmes les plus populaires du monde. La version la plus récente PGP 9.0 est distribuée par la PGP Corporation et peut être trouvée sur le site www.pgp.com. PGP offre toutes les fonctionnalités d'un cryptosystème complet : cryptage, décryptage, signature, certificat. Aux yeux des utilisateurs, il fonctionne comme un cryptosystème à clefs publiques alors qu'il s'agit d'un cryptosystème hybride.

Dans un premier temps, nous retracerons l'histoire de PGP depuis sa création.
Puis, nous nous attacherons aux aspects strictement techniques de son fonctionnement, ce qui nous conduira à quelques rappels lapidaires sur les principaux types de cryptosystèmes. Enfin, dans une partie plus orientée vers la pratique, nous montrerons comment utiliser concrètement les fonctionnalités offertes par PGP au travers d'exemples utilisant GPG, une implémentation gratuite d'Open PGP, le standard de PGP.

2 L'HISTOIRE

L'histoire de PGP est très complexe et les sources, bien que nombreuses, apparaissent souvent incomplètes, voire contradictoires. Toutefois 1983, quatre ans après la création de l'algorithme RSA (Rivest Shamir Adeleman), est considérée comme le point de départ de PGP.
L'informaticien Charlie Merrit, qui travaille sur une implémentation de RSA, prend contact avec Philip Zimmermann et lui demande d'effectuer des recherches concernant la possibilité d'implémenter RSA pour des ordinateurs privés. Charlie Meritt présente alors Philipp Zimmermann à Jim Bidzos le président de RSA Data Security. On sait peu de chose de cet entretien mais on peut affirmer qu'il fut primordial pour l'avenir de PGP : Zimmermann prétend que lors de cette réunion, Bidzos lui offrit des licences gratuites sur l'utilisation de l'algorithme RSA. Lorsque la première version de PGP est achevée, Zimmermann contact Bidzos pour obtenir ces licences mais ce dernier refuse, prétextant que celles-ci vont à l'encontre de la politique de la société. Zimmermann se retrouve confronté à un grave problème. Son logiciel utilise l'algorithme RSA alors qu'il n'en a pas obtenu les droits commerciaux.
De plus, peu de temps après ce refus de Bidzos, le sénat des Etats-Unis vote une loi anti-criminalité obligeant tout fabricant de cryptosystème à inclure dans son produit une "porte dérobée" c'est-à-dire un moyen pour le gouvernement de retrouver de façon systématique le texte en clair depuis un texte crypté. Zimmermann, qui se refuse à ajouter une porte dérobée à PGP, se retrouve confronté à une double difficulté : un litige financier avec RSA Data Security et un risque imminent de voir son produit devenir illégal. Il s'empresse ainsi de

mettre PGP à disposition du public en l'offrant en libre téléchargement sur son site Internet. La première version publiquement connue de PGP (1.0) voit le jour en 1991.
Bidzos menace alors Zimmermann de poursuites judiciaires et ce dernier accepte de ne plus mettre PGP à disposition du public tant qu'il n'a pas obtenu explicitement l'accord de RSA Data Security. Cette décision arrive trop tard, PGP a déjà fait son chemin sur internet.

L'algorithme original de PGP (appelé Bass-O-Matic) développé par Zimmermann lui-même présente des défauts de sécurité et est rapidement abandonné au profit d'un autre algorithme développé en Suisse et connu sous le nom d'IDEA (International Data Encryption Algorithm) qui est inclus dès la version 2.0 de PGP.

En 1993, le gouvernement des Etats-Unis d'Amérique lance des poursuites judiciaires contre Zimmermann pour une violation des restrictions liées à l'export des produits cryptographiques, PGP pouvant être téléchargé librement partout dans le monde.
Après trois ans d'enquête, l'investigation est abandonnée sans que le gouvernement n'en explique le motif.
Dans le même temps, Zimmermann, qui cherche à rendre PGP légitime, fonde la société Viacrypt et produit Viacrypt PGP qui devient la première version légale de PGP. Afin de compenser le prix des licences RSA il est vendu aux alentours de 150 $ US. Toutefois, une version gratuite de PGP pour usage non commercial est systématiquement mise à disposition par Viacrypt.
En 1996, la société PGP Inc. est créée et absorbe Viacrypt.
En janvier 1998 c'est au tour de Network Associates d'acquérir PGP Inc. Mais ceux-ci rompent avec la tradition de PGP voulant que le code source soit mis à disposition du public afin qu'il puisse s'assurer de la qualité du produit et de l'absence de portes dérobées dans le logiciel.
En 1998 est proposé un standard de l'IETF (Internet Engineering Task Force) nommé Open-PGP et décrit dans la RFC 2440. Il décrit les formats des messages, signatures ou clefs, qui peuvent être envoyés par des programmes cryptosystèmes en se basant sur PGP.
On peut désormais considérer que PGP implémente cette norme tout en offrant des fonctionnalités supplémentaires. Mais également, que d'autres logiciels peuvent dès lors implémenter la même norme que PGP et ce, gratuitement, à condition ne pas utiliser d'algorithme breveté payant.
En 2002, Network Associates vend ses activités dans PGP à PGP Corporation qui, la même année, rend de nouveau le code source de PGP accessible pour des revues par les pairs.
En 2006, PGP Corporation est toujours propriétaire de PGP, et la version actuelle PGP 9.0 n'a plus beaucoup de points communs avec PGP 1.0. D'un système de cryptage en ligne de commande offrant peu de possibilités, PGP est devenu une offre logicielle complexe. Elle se compose toujours du logiciel mais il est désormais muni d'une interface graphique avancée et de diverses possibilités telles que le cryptage de disque dur, de mails via des plug-ins pour les différents clients de messagerie etc. Des bibliothèques de fonctions sont également vendues et permettent d'incorporer les fonctionnalités de PGP au sein des développements logiciels.

3 FONCTIONNEMENT DE PGP

3.1 Crypter et décrypter avec PGP

3.1.1 Rappels sur la cryptographie

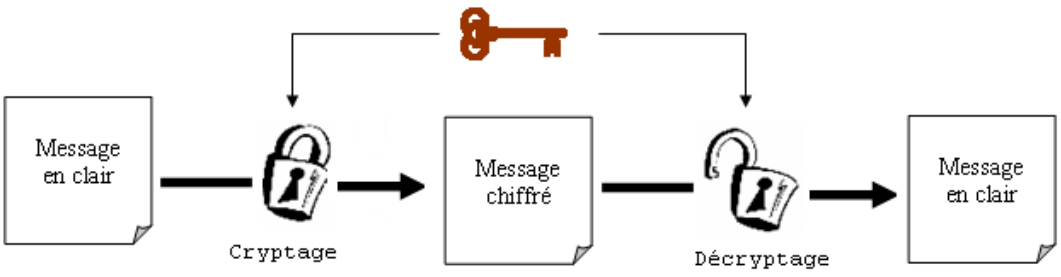
La cryptographie est l'ensemble des techniques qui permet de rendre un message inintelligible. L'action de coder le message initial (plaintext) en un message inintelligible, appelé « cryptogramme » (cipher text), se nomme « chiffrement » (ou cryptage). L'opération inverse est le « déchiffrement » (ou décryptage).

Le chiffrement (respectivement le déchiffrement) s'effectue au moyen d'une clef de chiffrement (respectivement une clef de déchiffrement).

PGP étant une technique de chiffrement hybride basée sur le chiffrement symétrique et le chiffrement asymétrique, un brève rappel de ces notions est nécessaire.

Le chiffrement symétrique (ou chiffrement à clefs privées)

Le chiffrement symétrique consiste à utiliser la même clef pour chiffrer et déchiffrer un message.



Cryptographie à clef privée : la même clef est utilisée pour crypter et décrypter.

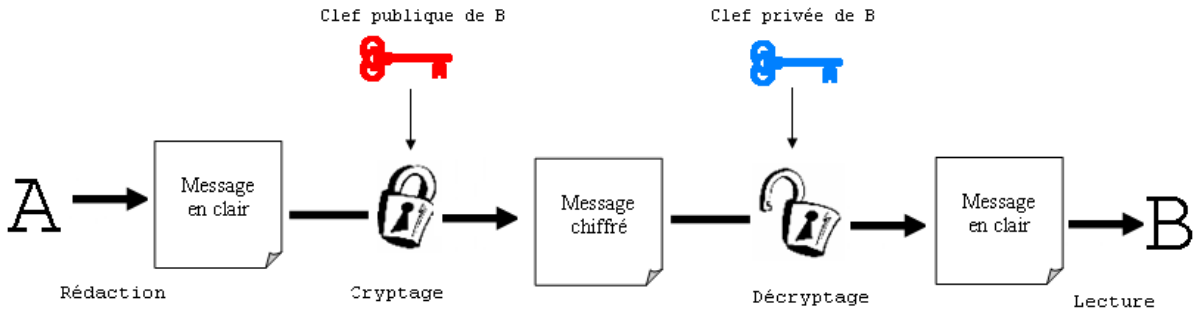
Schéma 1 : Chiffrement symétrique

Son principal inconvénient réside dans l'échange des clefs qui doit s'exécuter via un canal sécurisé (ex : de la main à la main sur une disquette)

Le chiffrement asymétrique (ou chiffrement à clefs publiques)

Les chiffrements asymétriques utilisent un système de paires de clefs (des bi-clefs). L'utilisateur génère une clef aléatoire dite clef privée qu'il est seul à connaître. De cette clef est déduite la seconde, appelée clef publique, que l'utilisateur distribue par exemple via un serveur de clefs.

L'expéditeur qui souhaite envoyer un message chiffré au destinataire doit récupère la clef publique du destinataire sur le serveur. Puis il chiffre le document avec cette clef comparable à un cadenas. Lorsque le destinataire reçoit le document, il le déchiffre grâce à sa clef privée, la seule clef capable d'ouvrir le cadenas.



Cryptographie à clef publique : le message est chiffré avec la clef publique du destintaire et décrypté avec sa clef privée.

Schéma 2 : Chiffrement asymétrique

Avec le déchiffrement asymétrique le problème de la distribution de la clef est résolu. Cependant cette technique est moins rapide et nécessite de s'assurer que la clef récupérée est bien celle du destinataire du message.

3.1.2 Chiffrement de PGP

Le système de PGP est un système hybride que l'on peut classer dans les systèmes "à clef de session", c'est-à-dire un système qui utilise à la fois le principe du chiffrement à clef privée et le principe du chiffrement à clef publique.
Considérons les différentes étapes du transfert d'un message crypté avec PGP de l'expéditeur X vers le destinataire Y.

(i) X doit envoyer le message crypté à Y.

(ii) Y crée une paire de clef via l'algorithme RSA. Il transmet sa clef publique à X.
(iii) X saisit le texte en clair à envoyer. Ce texte est tout d'abord compressé ce qui offre un double avantage :

- la taille des données à transférer est réduite,
- les risques de décryptage sont minimisés (la plupart des techniques de cryptanalyse se base sur le texte en clair obtenu. Si le texte obtenu est un texte compressé il est, par exemple, plus difficile de calculer la probabilité de retrouver telle ou telle lettre).

Il faut noter que la compression n'est pas systématique. Si le taux de compression d'un fichier n'est pas satisfaisant ou si le fichier est trop petit, cette étape n'est pas réalisée.

(iv) Puis, X crée aléatoirement une clef secrète IDEA. (Nous reviendrons sur les clefs IDEA dans la partie pratique). L'expéditeur chiffre le texte avec cette clef IDEA. Le texte ainsi chiffré pourra être déchiffré avec la même clef. Dans PGP, le message est alors crypté selon un système symétrique (à clef secrète).

(v) Le destinataire Y ne connaissant pas cette clef, elle va lui être envoyée avec le message crypté. Toutefois pour éviter qu'elle soit interceptée, la clef sera également cryptée à l'aide de la clef publique de Y. La clef privée IDEA est cryptée avec la clef publique de Y selon un système asymétrique (à clef publique).

Finalement, le résultat obtenu contient :

- le texte chiffré avec la clef IDEA,
- la clef IDEA chiffrée avec la clef publique RSA du destinataire.

A réception du message, le destinataire utilise sa clef privée RSA pour retrouver la valeur de la clef IDEA. Il utilise la clef obtenue pour déchiffrer le message reçu.

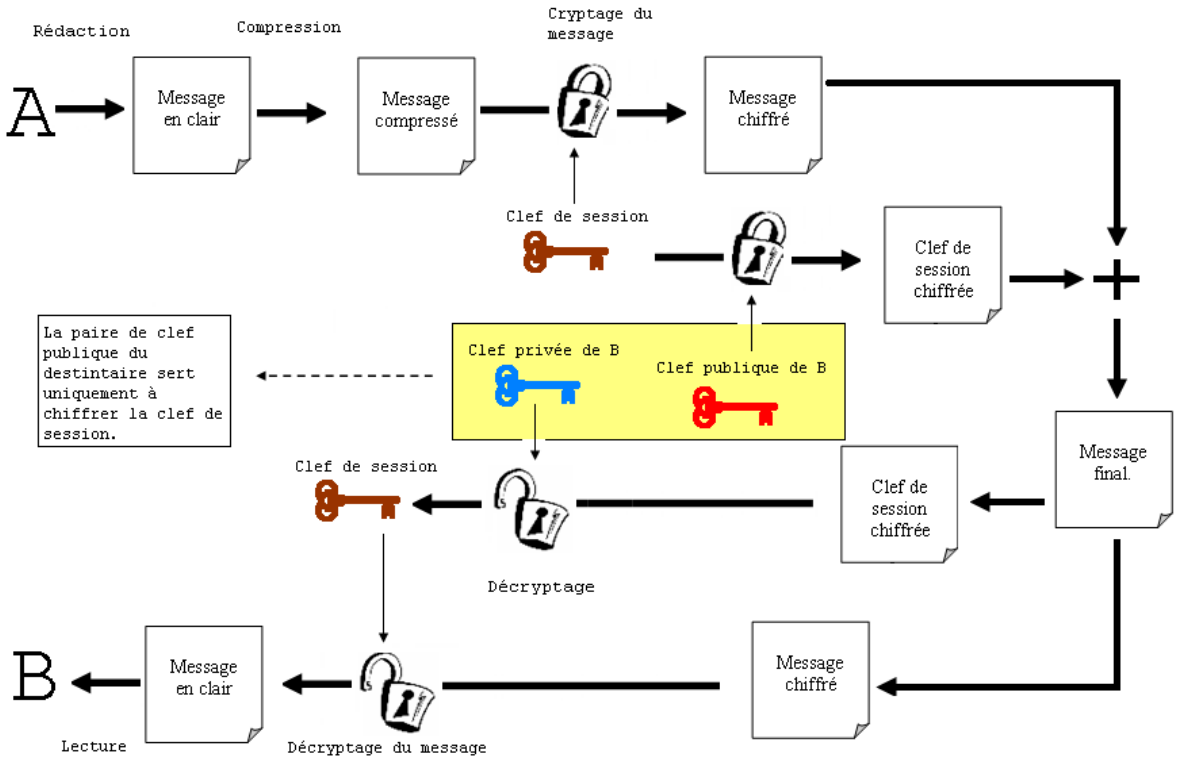


Schéma 3 : Chiffrement hybride de PGP

PGP possède plusieurs avantages :

- La rapidité : le message est chiffré par un cryptage symétrique. La clef IDEA est chiffrée de façon asymétrique. Toutefois le volume de données que représente cette clef est négligeable par rapport au volume de données que représente le message. Par conséquent, le temps de chiffrage global est proche de celui d'un système symétrique.
- Une plus haute sécurité qu'un système à clef symétrique. En effet, si l'on peut considérer que le niveau de sécurité du système PGP est celui de son maillon le plus faible - le système à clef privée servant à coder le message - il faut toutefois nuancer cette conclusion. Dans un système à clef privée standard, le canal d'échange de la clef est le point faible du système. Si l'on désire changer de clef afin de minimiser les risques, cela nécessite de définir un moyen d'échanger cette nouvelle clef, opération difficile à mettre en pratique. Dans PGP en revanche, la clef utilisée pour coder le message est nouvelle pour chaque message. Ce qui implique que pour effectuer une attaque il est nécessaire de casser au choix :
    - autant de clefs privées que de messages,
    - le système de clefs RSA, ce qui rend finalement PGP plus résistant qu'un système à clef privée classique.

3.2 Autres fonctionnalités de PGP

3.2.1 La signature des données

En matière de signature des données, PGP utilise un scellement de données. Il applique une fonction de hachage au texte en clair à signer. Puis le condensé obtenu, de taille fixe, est signé avec la clef privée de l'expéditeur. Le sceau ainsi obtenu est joint au texte en clair.
A la réception du message, le destinataire (i) applique la fonction de hachage au texte en clair, (ii) utilise la clef publique de l'expéditeur pour retrouver la valeur du condensé joint au texte en clair et (iii) compare les deux condensés. Il s'agit d'un système de scellement des plus classiques, seule la fonction de hachage utilisée est propre à PGP.

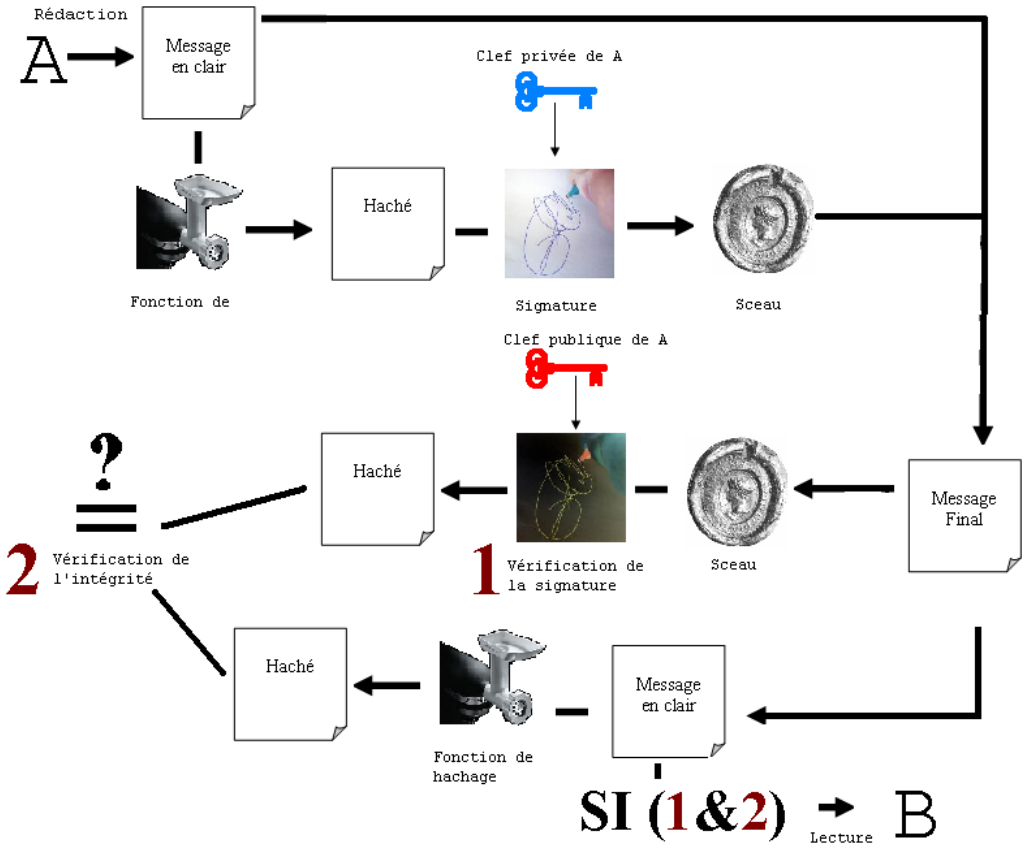


Schéma 4 : Le scellement de données

3.2.2 Les certificats

Rappelons que l'un des points clefs des cryptosystèmes est la vigilance que l'utilisateur apporte à la vérification de l'appartenance de la clef au bon propriétaire.

Ces systèmes sont en effet sensibles aux attaques dites de 'l'homme au milieu' qui consiste pour l'attaquant à créer une clef qu'il fait passer pour la clef d'une autre personne, s'accordant ainsi la possibilité de lire tous les messages sensément destinés à la personne dont il a usurpée l'identité. Pour éviter cela, on crée des certificats numériques dont le but est d'apporter la preuve de la validité d'une clef et de son appartenance à un propriétaire donné.
Ces certificats peuvent être considérés comme les cartes d'identité des clefs et se composent de trois parties :

- la clef publique pour laquelle le certificat est généré,
- des informations sur le détenteur de la clef (nom d'utilisateur, mails etc.) et sur le certificat lui-même (durée de validité …),
- d'une ou plusieurs signatures de personnes attestant de la validité de ces informations et de la clef.

En règle générale, une autorité appelée « autorité de certification » est seule habilitée à produire des certificats et les signer à l'aide de sa clef privée. Dans PGP, le système retenu pour les certifications numériques ne se base pas sur une autorité de certification centralisée mais sur un système de confiance.

Il est possible à chaque personne de signer de sa clef privée un certificat. Contrairement à un système centralisé, un certificat PGP pourra contenir une multitude de signatures numériques.

Le format d'un certificat PGP comprend entre autres les informations suivantes :

- Le numéro de la version de PGP utilisée pour créer la clef associée à ce certificat
- La clef publique sur laquelle porte ce certificat ainsi que l'algorithme employé pour la générer
- Des informations sur le propriétaire de cette clef (nom, mail, nom d'utilisateur etc.)
- La signature numérique du propriétaire effectuée à l'aide de la clef privée qui correspond à la clef publique du certificat
- La période de période de validité du certificat
- Les éventuelles signatures effectuées par d'autres utilisateurs.

PGP reconnaît également les certificats X. 509. Il est possible de produire ses propres certificats PGP. Pour les certificats X. 509, il faut effectuer la demande auprès d'une autorité de certification. La structure des certificats est normalisée par le standard X.509v3 de l' HYPERLINK "http://www.itu.int" UIT (Union Internationale des Télécommunications).

3.2.3 Les niveaux de confiance

Avec le système des certificats à signatures multiples se pose le problème de déterminer si les signatures apportées aux certificats sont dignes de confiance. Pour pallier cet inconvénient, un système basé sur les niveaux de confiance et de validité est mis en place.
De base, il existe dans PGP trois niveaux de validité :

- valide,

- semi – valide,
- invalide,

ainsi que trois niveaux de confiance :

- confiance totale,
- confiance moyenne,
- aucune confiance.

Lorsque l'utilisateur génère sa paire de clef dans PGP, celle-ci se voit implicitement attribuer les niveaux maximums pour ces deux domaines.

Considérons maintenant que l'utilisateur importe la clef de X dans son système. Il valide la clef de X en signant son certificat et lui accorde une confiance maximum.
La clef de X à désormais valeur d'autorité de certification dans le système ainsi lorsqu'une clef signée par X est importée, elle sera considérée comme valide dans le système.

Si, désormais, une confiance moyenne est accordée à X et Y et que l'utilisateur importe une clef, deux cas de figure se présentent :

- si la clef est seulement signée par X ou Y, elle ne sera pas valide dans le système
- si la clef est signée par X et Y, elle sera valide.

En résumé, pour qu'une clef soit validée dans le système PGP de l'utilisateur, elle devra avoir reçu soit :

- la signature de ce dernier,
- la signature d'une clef à laquelle une confiance totale aura été accordée,
- la signature d'au moins deux clefs auxquelles une confiance moyenne aura été apportée.

Les signatures effectuées par des clefs qui ont le niveau de confiance 'Aucune Confiance' ne sont tout simplement pas prises en compte.

3.2.4 Les empreintes

Les niveaux de confiance permettent d'automatiser la validation de clefs grâce à des clefs qui sont déjà valides. Reste qu'en haut de cette pyramide, il faut commencer par valider des clefs. C'est pourquoi il est nécessaire de pouvoir s'assurer de l'intégrité de celles-ci. C'est le rôle des empreintes.
Une empreinte est un condensé obtenu en appliquant une fonction de hachage à un certificat. Bien entendu, comme tout condensé, il est unique.
Dans PGP, les empreintes de certificats peuvent être affichées sous forme hexadécimale ou sous la forme d'une série mots biométriques (« biometric words »).

Les empreintes sont utilisées pour s'assurer auprès du détenteur de la clef que l'on veut valider l'authenticité du certificat qui la contient, en comparant cette empreinte à son homologue dans le système du propriétaire de la clef.

3.2.5 La révocation

Un certificat possède une durée de validité. A l'issue de celle–ci, la clef de ce certificat n'est plus considérée comme valide. Il peut toutefois être nécessaire d'invalider une clef avant la fin de son certificat : si celle-ci a été cassée ou le mot de passe lié a cette clef a été perdu.

Dans X509, révoquer sa signature sur un certificat consiste à enlever sa signature d'un certificat d'authenticité, c'est-à-dire à indiquer qu'on n'apporte plus son crédit à ce certificat.

La révocation PGP va plus loin en apportant la possibilité d'invalider entièrement un certificat (et plus seulement d'en enlever sa signature). Toutefois, une telle révocation n'est possible que pour :

- le propriétaire de ce certificat
- une personne considérée par ce propriétaire comme une autorité de certification, c'est à dire une personne à laquelle il a attribué un niveau de confiance totale.

4 UTILISATION PRATIQUE

Dans cette partie, nous traiterons de l'utilisation pratique de PGP. A cette fin, nous n'utiliserons pas le logiciel PGP de la PGP Corporation (cf. www.pgp.com) mais une autre implémentation du standard Open PGP appelée GPG ou GnuPG (Gnu Privacy Guard) libre de droit et reconnue par Philip Zimmermann. De plus ce logiciel s'avère plus éducatif car il fonctionne entièrement en ligne de commande.

IDEA est un algorithme de génération de clef privée propriétaire qui est utilisé dans le logiciel PGP de la PGP Corporation, à ce titre, il n'est pas utilisé dans GnuPG. La clef qui sert à crypter le message dans GnuPG n'est donc pas une clef IDEA, mais reste une clef privée répondant à la spécification de la recommandation Open-PGP

4.1 Télécharger et installer Gnu PG

GnuPG (également appelé GPG) peut-être téléchargé à cette adresse :
http://www.gnupg.org/(fr)/download/index.html

http://www.gnupg.org/(fr)/download/index.html

La version actuelle pour Windows (utilisée pour réaliser cet article) est munie d'un utilitaire d'installation. Il faut ajouter le chemin d'accès à gpg à votre variable path afin de pouvoir encoder depuis n'importe quel endroit.
Bien que les captures d'écran qui vont suivre aient été effectuées sous MS-DOS, les commandes ont la même syntaxe sous Linux.

4.2 Gérer les clefs

Dans cette partie, nous présenterons les types de clefs pris en charge par GnuPG ainsi que les algorithmes disponibles pour les générer. Nous verrons les commandes pour générer, éditer, importer, exporter et publier ces clefs

4.2.1 Clefs V3/V4

Rappelons que l'envoi d'un message nécessite:
- la clef publique du destinataire,
- une clef privée pour signer le message.

Dans ce système, la clef publique utilisée pour
- vérifier la signature d'un correspondant,
- chiffre un message pour le correspondant

est la même. Cette paire de clef unique est appelée une paire de clefs "V3". GnuPG prend également en charge des paires de clefs dites V4.

Dans un système de paire de clefs V4, on ne génère plus 1 mais 2 paires de clefs utilisées de la manière suivante :

Paire de clefs principales :
- la clef publique sert au correspondant à vérifier les signatures du propriétaire,
- la clef privée sert au propriétaire à signer les messages.

Paire de clefs secondaires :
- la clef publique sert au correspondant à chiffrer des fichiers à l'attention du propriétaire (en réalité elle sert à signer la clef utilisée au chiffrement des à destination du propriétaire)
- la clef privée sert au propriétaire à déchiffrer les fichiers qui lui ont été envoyés (en réalité, elle sert à déchiffrer la clef utilisée au chiffrement des messages à destination du propriétaire)

Dans un système de clefs V4, les différents rôles des clefs sont clairement distingués. De plus, ces clefs présentent l’avantage suivant : lorsque l'une des paires de clefs est compromise,

seule l'une des fonctions est également compromise (la signature ou le chiffrement). C'est pourquoi il est conseillé d'utiliser ce type de clefs.

En résumé, le schéma complet représentant l'envoie d'un message de X vers Y et d'une réponse de Y vers X est le suivant :

4.2.2 Les algorithmes pris en charge

Les clefs créées dans les exemples suivants sont des clefs V4. Il s'agit de deux paires de clefs.

Afin de bien comprendre les choix proposés par GPG, il convient de préciser les algorithmes pris en charge pour la signature ainsi que le chiffrement. Toutefois, rappelons que GPG permet d'importer des clefs RSA V3 mais pas d'en créer.

1) La signature :

DSA : Digital Signature Algorithm, conforme au Digital Signature Standard, concurrent de RSA. Les clefs sont limitées à 1024 Bits.

RSA : deux versions de RSA sont implémentées pour la signature.
La version RSA pour les clefs V3 : lorsqu'une unique paire de clefs est utilisée à la fois pour le chiffrement et la signature (1).
La version RSA-S pour les clefs V4 : elle est utilisée pour générer une paire de clefs RSA destinée uniquement à la signature dans un système de clefs V4.

2) Le chiffrement

RSA : deux versions de RSA sont implémentées pour le chiffrement.
La version RSA pour les clefs V3 : la paire de clefs étant unique, on retrouve la paire générée en (1).
La version RSA-E pour clefs V4 : elle est utilisée pour générer une paire de clefs RSA destinée au chiffrement dans un système de clefs V4.

Elgamal : algorithme de Taher Elgamal également appelé "DH" pour Diffie-Hellman dans PGP.

Finalement, puisque l'on ne peut générer que des clefs V4, les combinaisons possibles sont :

SIGNATURE	**CHIFFREMENT**
DSA	ELGAMAL
DSA	RSA-E
RSA-S	ELGAMAL

RSA-S	RSA-E

4.2.3 Génération des clefs

La génération des clefs est réalisée via la commande :

gpg --gen-key

```
C:\>gpg --gen-key
gpg (GnuPG) 1.4.2.2; Copyright (C) 2005 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

Sélectionnez le type de clé désiré:
   (1) DSA et Elgamal (par défaut)
   (2) DSA (signature seule)
   (5) RSA (signature seule)
Votre choix ?
```

Les clefs peuvent être générées à l'aide de trois types d'algorithme selon le fonctionnement suivant :

1) DSA et Elgamal créeront automatiquement deux paires de clefs, une première paire DSA pour la signature et une secondaire paire Elgamal pour le chiffrement.
2) Les choix <**2**> (DSA) et <**5**> (RSA) créeront seulement une paire de clefs de signature, respectivement DSA ou RSA-S. L'ajout d'une seconde paire de clefs pour le chiffrement sera alors nécessaire.

Dans cet exemple nous allons créer une paire V4 **RSA-S/Elgamal** :

On choisit 5 pour créer la paire RSA-S. Le système demande de renseigner la taille de la clef.

```
Votre choix ? 5
les clés RSA peuvent faire entre 1024 et 4096 bits de longueur.
Quelle taille de clé désirez-vous ? (2048) _
```

Plus la longueur est importante plus elle offre de sécurité. En contre partie, les temps de calcul augmentent corrélativement. Pour un usage privé, 1024 est suffisant.

La seconde étape consiste à déterminer la durée de vie de la paire de clefs :

```
Spécifiez combien de temps cette clé devrait être valide.
         0 = la clé n'expire pas
      <n>  = la clé expire dans n jours
      <n>w = la clé expire dans n semaines
      <n>m = la clé expire dans n mois
      <n>y = la clé expire dans n années
La clé est valide pour ? (0) _
```

Afin d'identifier la paire de clefs, certaines informations personnelles doivent être renseignées :

```
Nom réel: Bourgeois Morgan
Adresse e-mail: morganbourgeois@gmail.com
Commentaire: ma clef d'exposé_
```

L'étape suivante est primordiale. Gpg demande une phrase de passe qui sera utilisée à chaque fois qu'une opération sera effectuée avec la paire de clefs. En cas de perte, il sera impossible de générer vous-même un certificat de révocation pour cette paire de clefs.

Une fois la clef générée, vous en obtenez l'empreinte ainsi que l'uid (user Id) correspond :

```
    Empreinte de la clé = 8E2B F2BE BB8B AAE9 B2A3  7413 041E 2EC9 7A48 191B
uid                  Bourgeois Morgan (ma clef d'exposé) <morganbourgeois@gmail.
com>
```

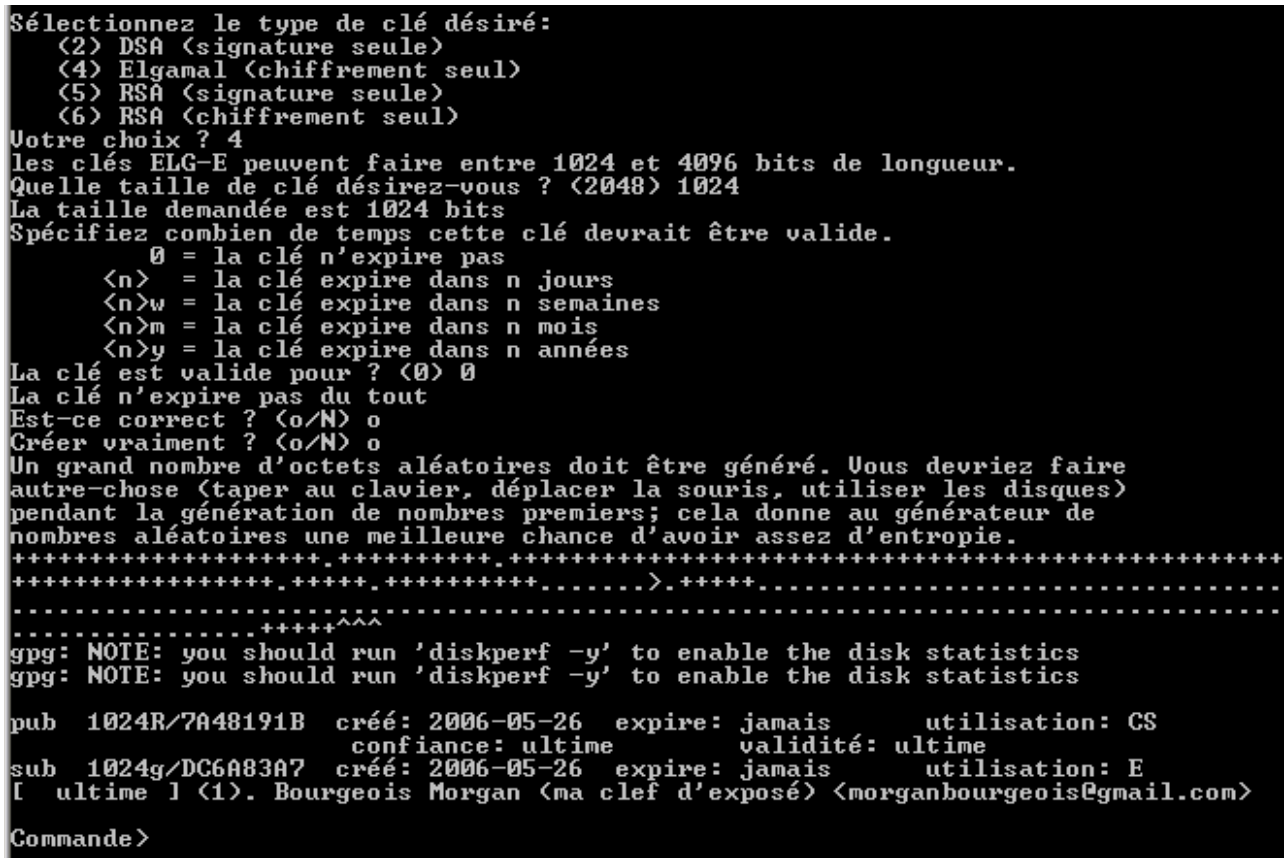
Reste la paire Elgamal pour la signature.
On édite la clef à l'aide de la commande **gpg --edit-key**, puis on ajoute une paire avec la commande **addkey** :

```
C:\Documents and Settings\              >gpg --edit-key 7A48191B
gpg (GnuPG) 1.4.2.2; Copyright (C) 2005 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

La clé secrète est disponible.

pub  1024R/7A48191B  créé: 2006-05-26  expire: jamais      utilisation: CS
                     confiance: ultime        validité: ultime
[  ultime ] (1). Bourgeois Morgan (ma clef d'exposé) <morganbourgeois@gmail.com>


Commande> addkey
La clé est protégée.

Vous avez besoin d'une phrase de passe pour déverrouiller la
clé secrète pour l'utilisateur: « Bourgeois Morgan (ma clef d'exposé) <morganbou
rgeois@gmail.com> »
clé de 1024 bits RSA, ID 7A48191B, créée le 2006-05-26

Entrez la phrase de passe: _
```

Gpg demande :

- la phrase passe,
- le type de paire de clefs ajoutée (4) pour notre exemple,
- les renseignements nécessaires à la génération de cette paire de clefs,
- une confirmation.

Le système génère la clef puis renvoie sur le prompt des commandes d'édition de clefs :



L'enregistrement des modifications et le retour au prompt DOS s'effectue à l'aide de la commande **save** :

4.2.4 Exportation /Importation /Publication

L'exportation de la clef publique s'effectue par la commande :

Gpg -a --export > Nomdefichier

L'option **-a** évite les problèmes de codage des caractères dans le fichier.
Nomdefichier est le fichier destination qui sera créé par la commande.

Dans le cas traité ici, il faut exporter les deux clefs publiques de la bi-paire de clefs :

```
C:\PROGRA~1\GNU\GnuPG>gpg -a --export 7A48191B > signatureMorgan.gpg
C:\PROGRA~1\GNU\GnuPG>edit signatureMorgan.gpg
C:\PROGRA~1\GNU\GnuPG>gpg -a --export DC6A83A7 > chiffreMorgan.gpg
C:\PROGRA~1\GNU\GnuPG>_
```

Nous obtenons deux fichiers identiques. Lorsque l'on exporte une bi-paire de clefs, GnuPG exporte automatiquement les clefs publiques de chacune des paires.
Les commandes gpg --a –export 7A48191B et gpg --a –export DC6A83A7 produisent ainsi le même effet.

L'étape suivante consiste à distribuer les clefs aux correspondants de deux manières. Il est possible, soit de leur envoyer par mail le fichier .gpg créé, soit d'utiliser un serveur pour y déposer la clef.

Dans le premier cas, lors de la réception d'un tel fichier, il faut demander au système d'ajouter les clefs qu'il contient à votre trousseau via la commande **gpg –import FICHIER**

Dans le second cas, il existe deux techniques pour déposer une clef et la retrouver :

1) Sur la page web du serveur (ci-dessous le serveur http://pgp.mit.edu)

On choisit de déposer une clef (submitting a Key).
Après ouverture du fichier texte dans lequel la clef a été exportée (signatureMorgan.gpg), on effectue un copier coller du fichier :

Submit a key

Enter ASCII-armored PGP key here:

```
Version: GnuPG v1.4.2.2 (MingW32)

mIsERHbVhgEEAMQKX8AcZtoUE2Zg73vzXJmP5Fd9F3x3zFGgSIu3LLlXcjWibPtE
ok1o1qa+ygmSjQc22FCaCPo6nRYi1EOGh1uX2VgcxmcVdZupqNEW4HYy5MgOgQAN
JHRUMM9/lgSHZoPaxkxF7zhICC8TOR/5s5TZZqkOAp7DaavXOrPWwl+HAAYptEBC
b3VyZ2VvaXMgTW9yZ2FuIChtYSBjbGVmIGQnZXhwb3PDqSkgPG1vcmdhbmJvdXJn
ZW9pcOBnbWFpbC5jb2O+iLYEEwECACAFAkR21YYCGwMGCwkIBwMCBBUCCAMEFgID
AQIeAQIXgAAKCRAEHi7JekgZG6jmA/OY+Trwt9DIgvGnGtyFY8nFnbd987YCg2rf
fqe1rbWQWvZrV3QFe9Wv2ATr8Ne41iYiHakUvBin6FLQJz8J9UrYOcVGHvlg4zU+
TYeZ56hdiRxjcQIUBcFLi7GDn7V58OAN3DL4dJEI1ZmUe7EMB1eEHNq1fw8FgTgn
GsLwZUyAFbkBDQREdtqkEAQAlvxFO2g6hQcq6HaFIl1htyOO7LsVtI3c/dBsyBiT
7HsyOEN2DOZG/1p12Tw5uUpaMkiRp9p3YDgwZ4kQXOkWniJtVVE2YOEMKXPO/65W
X85MW+frkh3TBktSav3IlI6sN2vAknTHBMBIm8dfCxFU8eDS+M94UhbLNUHpqyH+
Dh8AAwUD/O1NwT1OJBdMklQ4rTMGT4BH1dnE7y9A2peUOO2elBQ+JiQtvp8jsdST
l9NBGygAWnqCRu7utthwqa+xgZMtGWVB2CRireH1VT5rSrsrblVFqj41WBPOx6dc
sOnv2iUvEOfLSjR3Tv+OjLB4AyDYThKGUdSWTHpFamuWe67ad6BPiJ8EGAECAAkF
AkR22qQCGwwACgkQBB4uyXpIGRsJAAP8D8FtrCSONOYuRztvct7KaDrgLe5KzpA7
ZRWSctcDembMc3vS8pShZAXD8WyhV81VXEwxcRjd8fJHZxYpllribd9X2tL27C55
vSjwzkOCC9oaE8vq2Q2zpHKPyOl4OmD1V17jzvUDwM8Z9D4ONqMtDRIJQQjzf5qX
fmRrmZ61mfO=
```

Clear | Submit this key to the keyserver!

Afin de vérifier le bon fonctionnement de la manipulation, on va rechercher la clef générée :

Extract a key

Search String: morganbourgeois | Do the search!

Voici la clef telle qu'elle apparaît sur le serveur :

Public Key Server -- Index ``morganbourgeois ''

```
Type bits /keyID    Date       User ID
pub  1024R/E9ED5F1D 2006/05/26 Bourgeois Morgan (ma clef d'exposÃ©) <morganbourgeois@gmail.com>
```

Pour récupérer une clef en allant sur la page web du serveur, il suffit de procéder à une recherche comme précédemment puis de cliquer sur le lien "KeyId".

Le contenu de la clef est alors affiché. Il ne reste plus qu'à effectuer un copier-coller dans un fichier texte et d'importer la clef via gpg –import :

Public Key Server -- Get ``0xE9ED5F1D ''

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: PGP Key Server 0.9.6

mIsERHbVhgEEAMQKX8AcZtoUE2Zg73vzXJmP5Fd9F3x3zFGgSIu3LLlXcjWibPtE
ok1o1qa+ygmSjQc22FCaCPo6nRYi1EOGh1uX2VgcxmcVdZupqNEW4HYy5MgOgQAN
JHRUMM9/lgSHZoPaxkxF7zhICC8TOR/5s5TZZqkOAp7DaavXOrPWwl+HAAYptEBC
b3VyZ2VvaXMgTW9yZ2FuIChtYSBjbGVmIGQnZXhwb3PDqSkgPG1vcmdhbmJvdXJn
```

2) Sur un serveur

Il est également possible d'envoyer (de recevoir) une clef sur un serveur directement depuis la ligne de commande **gpg --keyserver SERVEUR --send-keys LACLEF** :

```
C:\PROGRA~1\GNU\GnuPG>gpg --keyserver pgp.mit.edu --send-keys DC6A83A7
gpg: envoi de la clé 7A48191B au serveur hkp pgp.mit.edu
```

Pour recevoir :

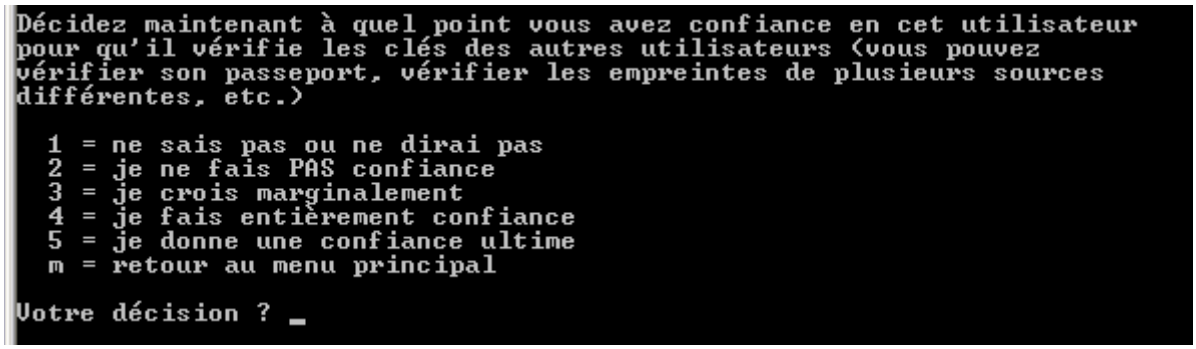
```
C:\Documents and Settings\Ometz Fabienne>gpg --keyserver gpg.linuq.org --recv-keys 8C31CB95
gpg: requête de la clé 8C31CB95 du serveur hkp gpg.linuq.org
gpg: clé 8C31CB95: clé publique « MOI A A@A;C (q) <A@A.c> » importée
gpg:        Quantité totale traitée: 1
gpg:                     importée: 1
```

A noter que le fait de demander la réception ou l'envoi d'une clef déjà présente dans votre trousseau (ou sur le serveur) entraîne sa mise à jour si, par exemple, une paire de clefs secondaire a été ajoutée.

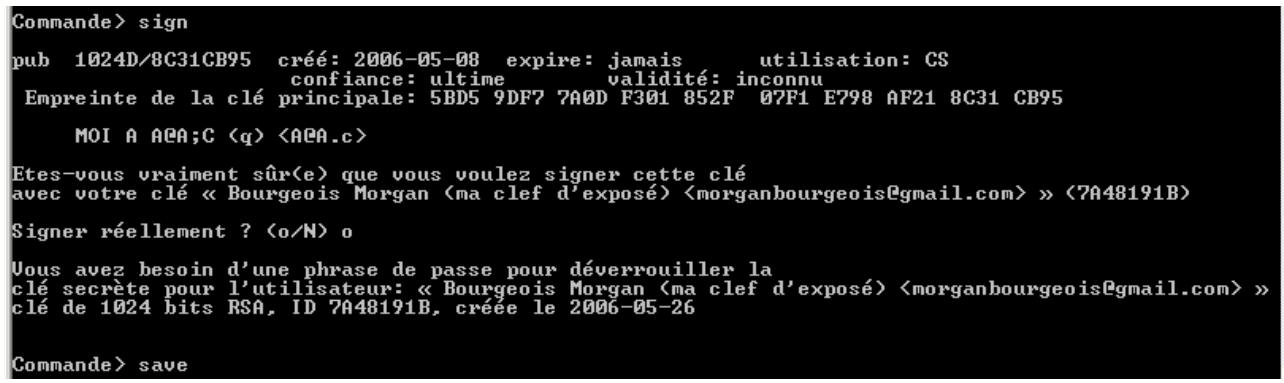
4.2.5 Signer les clefs.

Lorsque qu'une clef est importée, par défaut, aucune confiance ne lui est accordée. La commande **Gpg --edit-key LACLEF** affiche le niveau de confiance de **LACLEF** ce qui permet de le vérifier.

Toujours en mode édition, un niveau de confiance peut être attribué à la clef via la commande **trust** suivie d'une sauvegarde (**save**):



Notons, qu'il faut bien réfléchir au niveau de confiance accordé. Rappelons que lorsque l'on importe une clef signée par une autre clef déjà présente dans le trousseau, selon le niveau de confiance de cette dernière, la clef nouvellement importée pourra automatiquement être considérée comme digne de confiance.
Trust fixe le niveau de confiance d'une clef, pas sa validité. Pour cela, il faut la signer (commande **sign**)



Cette opération utilisant votre clef privée, elle nécessite votre phrase de passe.

4.2.6 Générer des certificats de révocation :

Il est important de créer un certificat de révocation DES lorsqu'une paire de clefs est crée. Ce certificat permettra d'informer gpg (et vos correspondants) lorsque l'une de vos paires de clefs ne doit plus être utilisée. Toutefois, la phrase de passe de cette paire de clef est nécessaire pour générer le certificat. Si vous oubliez cette phrase, la paire de clef ne pourra plus être utilisée mais le certificat ne pourra plus être généré non plus.

C'est pourquoi il est recommandé de générer ce certificat dès que vous créez une paire de clef, et de le stocker dans un autre emplacement en cas de besoin.
La génération du certificat se fait à l'aide de la commande :
gpg --gen-revoke CLEF > FICHIER

4.2.7 Afficher l'empreinte

L'empreinte est le moyen utilisé afin de vérifier qu'une clef appartient effectivement au propriétaire supposé. Pour afficher l'empreinte d'une clef on utilise la commande :

Gpg –fingerprint LACLEF

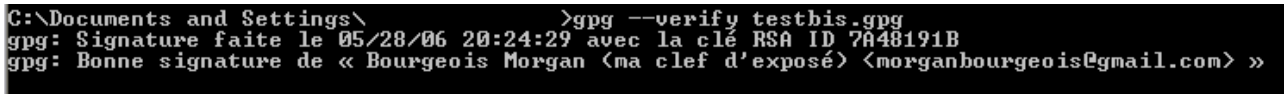
4.3 Signer /chiffrer / déchiffrer des messages

Les commandes suivantes permettent de signer électroniquement des messages, de les chiffrer et les déchiffrer.

- Signer et vérifier les signatures :

Pour signer un fichier on utilise: **gpg –-sign FICHIER**
La commande requiert la phrase de passe. Elle génère le fichier **FICHIER.gpg**

Pour vérifier les signatures d'un fichier : **gpg –-verify FICHIER**
La liste des clefs avec lesquelles le fichier a été signé s'affiche :



- Chiffrer un message

Le chiffrement d'un message est réalisé via la commande :
gpg –r CLEF –e –a –o [FICHIER CRYPTE] [FICHIER DE DEPART]

où

- **CLEF** est la clef publique du destinataire
- **FICHIER CRYPTE** est le fichier de destination. Par convention, on lui ajoute l'extension .gpg mais cela reste facultatif.
- FICHIER DEPART est le fichier qui contient le message en clair.

- Déchiffrer un message

La commande est :
gpg –d –a –o [FICHIER DECRYPTE] [FICHIER DE DEPART]

Il faudra bien entendu connaître la phrase de passe de la paire de clefs pour laquelle le message a été chiffré.

- Chiffre et Signer en une fois :

La commande suivante permet d'effectuer signature et chiffrement en une seule étape :
gpg --sign --armor --encrypt [FICHIER DE DEPART] > [FICHIER CRYPTE]

Le nom du destinataire et la phrase de passe sont requis.

4.4 Aller plus loin

Les commandes présentées dans cet article sont les plus utilisées mais ne couvrent pas l'étendue des possibilités de GnuPG. Il est par exemple possible de préciser les algorithmes supportés par une paire de clef et l'ordre de préférence de ces algorithmes.

Le manuel complet d'utilisation est disponible a cette adresse :
http://www.gnupg.org/gph/fr/manual.html

Pour conclure cet exposé, je vous propose un petit exercice.
Le but est de m'envoyer vos commentaires concernant l'article à l'adresse suivante :
morganbourgeois@gmail.com

Ces commentaires devront être chiffrés en utilisant ma clef publique qui est disponible sur
http://pgp.mit.edu/

Le nom d'utilisateur est : Bourgeois Morgan
Le commentaire : ma clef d'exposé
Le mail : morganbourgeois@gmail.com

L'empreinte : 8E2B F2BE BB8B AAE9 B2A3 7413 041E 2EC9 7A48 191B