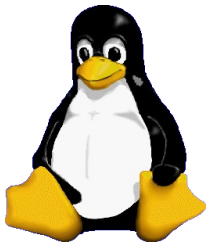


Sécurité GNU/Linux

GPG/PGP

By sharevb

Sommaire

I. Introduction

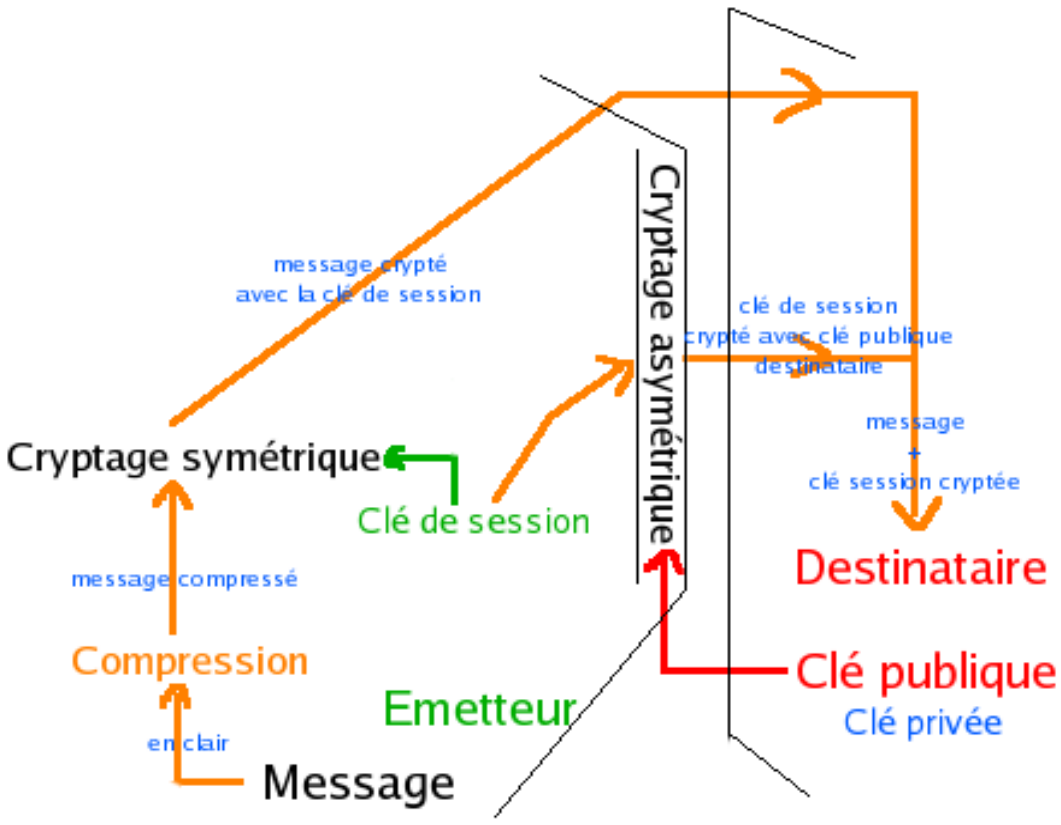
Le principe des algorithmes cryptographiques à clés asymétrique repose sur deux clés. L'une privée, qui sera l'information secrète, l''autre publique, que l'on distribue à tout le monde. Ainsi, une information cryptée avec la clé public ne pourra être décryptée qu'avec la clé privée et inversement. Par exemple si un utilisateur A veut envoyer un message crypté à B, il va encrypté le message avec la clé public de B, et B sera le seul à pouvoir décrypté le message car il est l'unique possesseur de sa clé privée.

II. Fonctionnement

a) Encryptage

Le fonctionnement de PGP/GPG est le suivant pour l'encryptage :

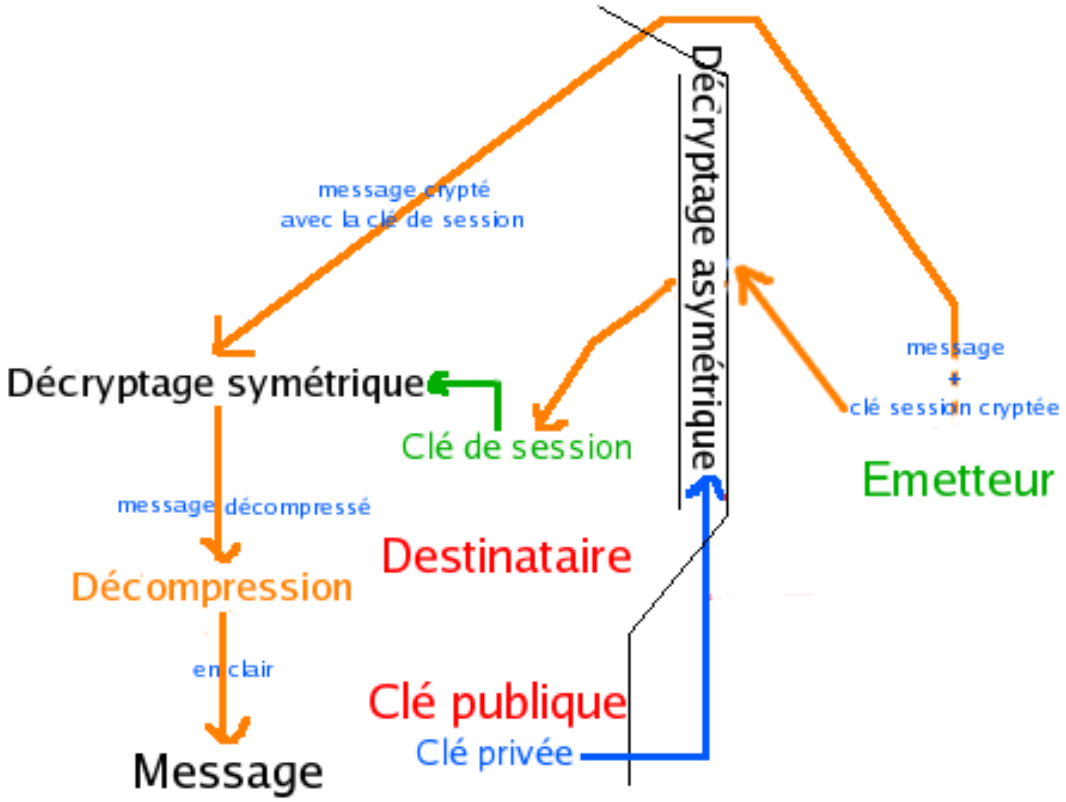
- le message est d'abord compressé afin de réduire la possibilité de trouver des motifs de texte clair dans le cas d'une cryptanalyse
- il génère une clé aléatoire appelée clé de session
- il crypte le message avec cette clé et un algorithme symétrique
- il crypte la clé de session avec la clé publique de la personne à qui on envoie le message



b) Décryptage

Le fonctionnement de PGP/GPG est le suivant pour le décryptage :

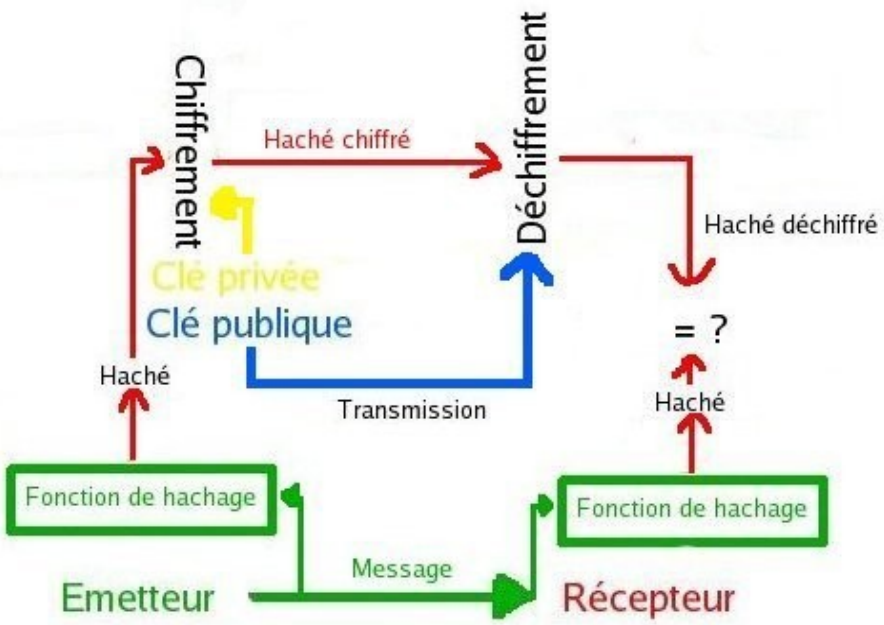
- on décrypte la clé de session avec sa clé privée
- on décrypte le message avec la clé de session décrypté avec l'algorithme symétrique



c) Signature

Le fonctionnement de PGP/GPG est le suivant pour la signature d'un message :

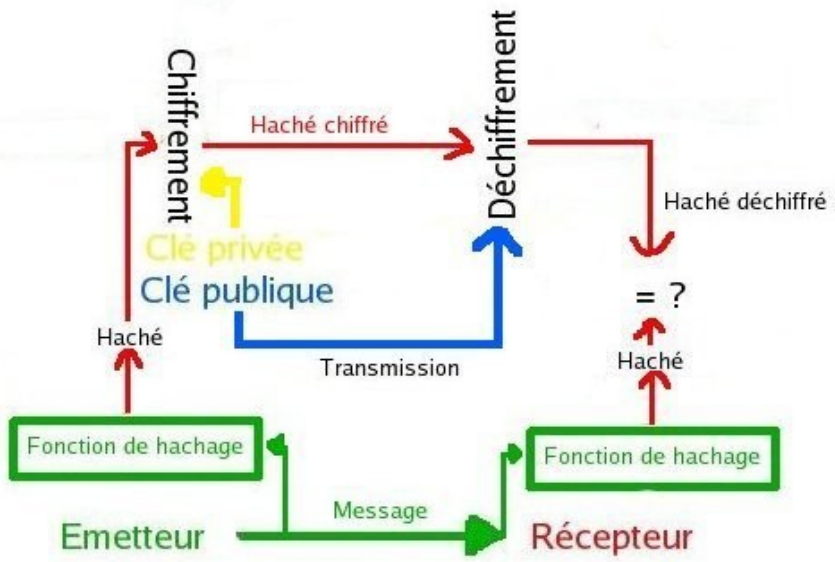
- le message est passé à une fonction de hachage type MD5 ou SHA-1 qui génère un digest de 128bits ou plus (qui n'est pas le même si un seul caractère est changé dans le message)
- le digest généré est crypté avec votre clé privée de sorte que l'on peut être sûr que c'est vous qui avez envoyé le message.



d) Vérification de signature

Le fonctionnement de PGP/GPG est le suivant pour la vérification de la signature d'un message :

- le digest est décrypté avec la clé publique de l'expéditeur
- on recalcule le digest du message reçu
- on compare les deux digests :
  - s'ils sont égaux : le message provient bien de son expéditeur (sauf si le pirate a réussi à remplacer la clé publique de l'expéditeur par la sienne avant que vous la trouviez sur un serveur de clé ou d'annuaire)
  - s'ils ne sont pas égaux : soit le message a été modifié avant de vous parvenir, soit le message ne provient pas de la personne que vous croyez



e) Clé de session

PGP/GPG utilise un système à clé de session pour plusieurs raisons :

- si le pirate arrive à décrypter la clé de session, il ne pourra lire que le message qui lui est attaché. Il faudra qu'il recommence à chaque nouveau message.
- Les systèmes à clés publiques sont beaucoup plus lents (1000 fois) que les systèmes à clés privées
- Les systèmes à clés privées sont beaucoup plus résistants à la cryptanalyse

Il existe essentiellement deux outils utilisés pour cela et qui sont compatibles entre eux :

- PGP (Pretty Good Privacy), qui est le premier, datant de 1991. On peut trouver des logiciels gratuit et payant de PGP sur le site www.pgp.com.
- GPG (GNU Privacy Guard), la version libre de PGP, qui existe sous Windows et sous Linux, c'est cette version que nous utiliseront.

III. Installation

GPG fait partie du rpm `gnupg`.

IV. Gérer ses clés et celles des autres

a) Génération de votre paire de clé

La commande utilisée pour gpg est : **gpg**. Ces clés sont stockées dans le répertoire `.gnupg`.

```
[utilisateur]$ gpg –gen-key
```

On choisit d'abord la méthode de chiffrement (par défaut) et la longueur de clé (par défaut).

Il faut ensuite entrer un nom, un email et un commentaire pour la clé, par exemple, votre nom ou/et votre email.

Il vous faut, ensuite, entrer une passphrase qui sert à crypter la clé privée (pour la protéger). Cette phrase vous sera demandé pour toutes les opérations sur ou avec la clé (modification, signature, cryptage...). Elle doit être relativement longue et ne doit pas pouvoir être devinée. Si vous n'oubliez, vous ne pourrez plus décrypter les messages que vous recevez (cryptés avec la clé publique associée).

Une fois toutes les informations entrées, le calcul de votre paire de clé commence. Le générateur de nombres pseudo-aléatoires a besoin d'un très grand nombre de données aléatoires, ce qui est difficile sur un ordinateur. Vous pouvez améliorer la qualité des résultats du générateur de nombres pseudo-aléatoires en générant vous même un peu de hasard, par exemple en bougeant votre souris, en tapant sur votre clavier, en exécutant des applications, etc. L'utilisation de pseudo-hasard est

nécessaire pour s'assurer qu'il n'est pas possible d'obtenir votre clé privée en effectuant le même calcul que vous.

b) Liste des clés que l'on possède

Pour afficher la liste des clés contenues dans son porte clé, on utilise l'option `--list-keys`.

```
[utilisateur]$ gpg –list-keys
/home/utilisateur/.gnupg/pubring.gpg
---------------------------------
pub   1024D/A43C54A7 2006-02-02
uid                  utilisateur (ma clé) <utilisateur@machine>
sub   2048g/BC928B9F 2006-02-02
```

c) Exportation de clé et transfert à vos amis

Maintenant il faut générer un fichier contenant sa clé public pour pouvoir le donner à ses amis. Pour cela il faut utiliser l'option `--export` de `gpg`. Attention par défaut la clé est affichée au format binaire, pour l'extraire au format texte il faut ajouter l'option `-a`.

```
[utilisateur]$ gpg -a –export nom_clé > fichier.pub
```

Enregistre la clé publique *nom_clé* dans le fichier *fichier*.pub au format ascii. Il suffit ensuite de l'envoyer aux personnes avec qui on veut communiquer.

On peut également publier sa clé public à travers des serveurs de clés avec la commande :

```
[utilisateur]$ gpg --send-keys --keyserver wwwkeys.pgp.net KEYID
```

Ainsi les personnes peuvent directement récupérer votre clé publique. (Remplacez KEYID par votre identifiant de clé).

d) Validation d'une clé reçue

Pour vérifier que la clé reçue est bien celle envoyée par son propriétaire et avant de la signer, son émetteur doit vous envoyer le fingerprint (par un canal sécurisé, comme le téléphone ou fax) de la clé qu'il vous envoit et vous devez faire la comparaison à la main. Pour obtenir l'empreinte d'une clé :

```
[utilisateur]$ gpg –finger nom_clé
```

Le fingerprint (empreinte) est un nombre hexadécimal.

e) Importation des clés de vos amis

Pour importer une clé que quelqu'un vous a donné, il faut utiliser l'option `--import` de `gpg`.

```
[utilisateur]$ gpg –import fichier.pub
```

Importe la clé publique *fichier*.pub dans votre trousseau de clés. La clé importée ne sera pas importée comme sûre. Pour la certifier, il vous faudra la signer :

```
[utilisateur]$ gpg –sign-key nom_clé
```

f) Certificat de révocation

Pour générer un certificat de révocation d'une paire de clé :

```
[utilisateur]$ gpg –gen-revoke nom_clé > fichier_revoke
```

Vous devez alors indiquer la raison de la révocation (menu) et, éventuellement, une indication de la raison de révocation. Il faudra ensuite, importer le *fichier_revoke* comme une clé normale.

V. Signatures, sceaux et messages cryptés

Les signatures GPG sont en fait des sceaux.

a) Signatures (sceaux)

Une signature sert uniquement à s'assurer que la personne qui vous a envoyé un message est bien la bonne, il n'y a pas d'encryption de message. Pour envoyer une signature créer un fichier texte contenant le message que l'on veut envoyer, et le signer avec la commande `gpg --sign -a`, qui va générer un fichier texte .asc contenant la signature.

Il existe trois type de signature :

- option `--sign` : génère le message crypté et signé dans un fichier `.sig` (binaire) ou `.asc` (ascii option `-a`). Pour vérifier la signature, on utilise l'option `--verify`. Pour vérifier la signature et décrypter le message en même temps, on utilise l'option `--decrypt`.

```
[utilisateur]$ gpg --sign -a fichier_message > fichier.asc
```

ou

```
[utilisateur]$ gpg --sign fichier_message > fichier.sig
```

```
[utilisateur]$ gpg --verify fichier.sig #ou fichier.asc
[utilisateur]$ gpg --decrypt fichier.asc ou fichier.sig > fichier_message
```

- option `--detach-sig` : génère un fichier `.asc` (option `-a`) ou `.sig` contenant uniquement la signature (et non le message crypté avec). Ceci est utile pour conserver le

document original tel quel et joindre la signature dans un fichier à part (contrairement à `--clearsign`).

```
[utilisateur]$ gpg --sign -a fichier_message > fichier.asc
ou
[utilisateur]$ gpg --sign fichier_message > fichier.sig

[utilisateur]$ gpg --verify fichier.sig fichier
[utilisateur]$ gpg --verify fichier.asc fichier
```

- option `--clearsign` : génère un document contenant le document original en clair suivi de sa signature au format ascii. Ceci est utile pour l'envoi de mail ou USENET non cryptés signés. Pour retrouver l'original, il faut toutefois modifier le message reçu.

```
[utilisateur]$ gpg --clearsign fichier_message
génère un fichier_message.asc contenant le message signé en clair.
[utilisateur]$ gpg --verify fichier.asc #ou fichier.sig
```

b) Encrypter un message

Pour encrypter un message et l'envoyer à quelqu'un qui va le decrypter, il faut utiliser les commandes suivantes :

```
[utilisateur]$ gpg --sign --armor --encrypt –recipient
<destinataire> <filename>
[utilisateur]$ gpg --decrypt -o <output> <filename>
```

Pour chiffrer un document, il faut utiliser l'option *--encrypt*. Vous devez avoir la clé publique de tous les destinataires. En plus du chiffrement, le document est compressé pour plus de sécurité.

```
[utilisateur]$ gpg --sign --armor --encrypt –recipient
nom_clé_destinataire fichier_à_crypter
```

L'option *--recipient* est utilisée une fois pour chaque destinataire du message, et prend un argument supplémentaire spécifiant la clé publique pour laquelle le document doit être chiffré. Le document chiffré peut seulement être déchiffré par quelqu'un possédant une clé privée qui correspond à une des clés publiques des destinataires. En particulier, vous ne pouvez pas déchiffrer un document chiffré par vous, à moins que vous ayez inclus votre clé publique dans la liste des destinataires.

c) Décrypter un message

Pour déchiffrer un message, on utilise l'option *--decrypt*. Vous avez besoin de la clé privée pour laquelle le message a été chiffré.

```
[utilisateur]$ gpg --decrypt fichier_à_decrypter > fichier_en_clair
```

VI. Configuration avec un client de messagerie

a) Enigmail

On peut utiliser gpg directement avec un logiciel de courrier éléctronique. Evolution gère normalement en natif les mails cryptés tandis qu'il faut installer Enigmail si l'on utilise Thunderbird.

Voir http://enigmail.mozdev.org/ ou http://fr.wikipedia.org/wiki/Enigmail.

b) Autres solutions

Voici une liste (non exhaustive) de plugins :

1 UNIX en console

- mutt et son fichier .muttrc
- pine : pgp4pine, magicpgp, pinepgp
- emacs : par défaut

2 UNIX sous X

- Xemacs : par défaut
- Xfmail gpg natif
- Evolution : nativement
- Thunderbird : Enigmail
- Sylpheed : gpgmpe

3 Windows

Installation de GnuPG puis :

- Eudora :
- Outlook : GnuPG Plugin
- Outlook express : GPG-OE
- ThunderBird : Enigmail

4 Autre

GPGrelay et PGPSendmail permettent d'utiliser PGP/GPG de façon transparente pour l'utilisateur.

Toutes sortes d'informations peuvent être trouvées sur la page GnuPG dédiée aux frontends.

VII. Version pour Windows

Il existe des versions de PGP/GPG pour windows, pour cela regardez aux adresses web suivantes :

- http://www.pgp.com
- http://openpgp.vie-privee.org/gnupg-win.htm

VIII. Résumé des commandes

Les options entres crochets, ne sont pas obligatoires.

a) Liste de serveurs de clés

.

Network	Server	Lang	Alias
	cryptonomicon.mit.edu	Engl	pgp.mit.edu, pgpkeys.mit.edu
	dir2.es.net	Engl	pgp.es.net
	moon.rediris.es	Engl Span	pgp.rediris.es
These servers are not on any network	nicpgp2.nic.ad.jp	Engl Japn	pgp.nic.ad.jp
	pgp.eteo.mondragon.edu	Span	
	pgp.uk.demon.net	Engl	
	pgp.zdv.uni-mainz.de	Engl	pgp.uni-mainz.de
	pks.aaiedu.hr	Croat	
	the.earth.li	Engl	wwwkeys.uk.pgp.net
Keyserver.Net	**kies.mcbone.net**	Engl	keyserver.topnet.de
SKS	calvin.lk.etc.tu-bs.de	?	
	dannyj.dynip.com	?	
	keyserver.hadiko.de	Engl	
	keyserver.mine.nu	Engl	
	lorien.prato.linux.it	Engl	keyserver.linux.it
	sks.keyserver.penguin.de	Engl	
PGPnet	blackhole.pca.dfn.de	Germ	wwwkeys.de.pgp.net
	minsky.surfnet.nl	Engl Dutch	keys.pgpi.net, pgp.surfnet.nl, wwwkeys.nl.pgp.net
	rex.citrin.ch	Engl	wwwkeys.ch.pgp.net, pgp.keyserver.ch
	stinkfoot.org	Engl	wwwkeys.stinkfoot.us.pgp.net
	wwwkeys.eu.pgp.net	?	
	wwwkeys.cz.pgp.net	?	pks.ms.mff.cuni.cz, wwwkeys.gpg.cz

b) Exporter une clé dans un fichier

`gpg [--armor] [--output` *`NOM-FICHIER`*`.asc] --export` *`NOM-clé`*

c) Exporter une clé sur un serveur

```
gpg [--keyserver HÔTE] --send-keys NOM-clé
```

d) Importer une clé depuis un fichier

```
gpg --import FICHIER
```

e) Importer une clé depuis un serveur

```
gpg [--keyserver HÔTE] --recv-keys ID-DE-clé
```

f) Rafraichir toutes vos clés depuis un serveur

```
gpg [--keyserver HÔTE] --refresh-keys
```

g) Créer une signature détachée

```
gpg --detach-sign FICHIER
```

h) Vérifier une signature détachée

```
gpg --verify FICHIER.sig [FICHIER]
```

i) Créer une signature en clair

```
gpg --clearsign FICHIER
```

j) Vérifier une signature en clair

```
gpg --verify FICHIER
```

k) Chiffrer un message/document

```
gpg --recipient clé-DESTINATAIRE [--armor] --encrypt FICHIER
```

l) Déchiffrer un message/document

```
gpg --decrypt FICHIER.asc [> FICHIER]
```

IX. Bibliographie

The **GNU Privacy Guard** – GnuPG.org

Mac GNU Privacy Guard

GnuPG pour Windows (OpenPGP en français)

GNU Privacy Guard - Wikipedia, the free encyclopedia

PGP Corporation - Home Page

Pretty Good Privacy - Wikipédia

mozdev.org - enigmail: index