Foire Aux Questions de GnuPG

Voici la FAQ à propos de GnuPG. La dernière version au format HTML est disponible ICI.

L index est généré automatiquement, il peut donc y avoir des erreurs. Des questions peuvent ne pas se trouver dans la bonne section. Les suggestions d amélioration de la structure de cette FAQ sont les bienvenues.

Envoyez les corrections et les ajouts au mainteneur. Si possible joignez également la réponse à ajouter. Merci de votre aide !

SVP, n envoyez pas de message du genre On devrait trouver la réponse à cette question dans la FAQ". Si la question n'a pas été posée auparavant, ce n'est pas une question posée si fréquemment ! Dans ce cas recherchez plutôt dans les archives de la liste de diffusion.

---

5. PROBLEMES DE COMPATIBILITE

6. PROBLEMES ET MESSAGES D'ERREURS

7. SUJETS AVANCES

1. GENERALITES

1.1) Qu'est−ce que GnuPG ?

GnuPG signifie GNU Privacy Guard, c est l utilitaire GNU permettant des communications et le stockage de données sécurisés. On peut s en servir pour chiffrer des données et pour créer des signatures numériques. Doté d un système évolué de gestion de clés, il respecte le futur standard OpenPGP décrit par la **RFC 2440**. Il s efforce ainsi d être compatible avec PGP de NAI, Inc.

1.2) GnuPG est−il compatible avec PGP ?

Généralement, oui. GnuPG et les versions récentes de PGP respectent le standard OpenPGP. Cependant quelques problèmes d interopérabilité subsistent. Pour plus de détails reportez−vous à la question **5.1.**

1.3) Peut−on utiliser librement GnuPG pour un usage personnel ou professionnel ?

Oui. GnuPG fait partie de la famille d outils et d applications GNU conçus et distribués en accord avec la General Public License (GPL) de la Free Software Foundation (FSF). En conséquence le logiciel peut être librement copié, utilisé, modifié et distribué tout en respectant les termes de cette licence. Pour plus d informations, référez−vous au fichier COPYING qui accompagne le logiciel.

1.4) Quelles conventions cette FAQ utilise−t−elle ?

Bien que GnuPG soit développé (souvent en parallèle) pour différents systèmes d exploitation, les conventions utilisées dans cette FAQ sont celles d un environnement shell d UNIX. Pour les utilisateurs de Win32, un prompt shell ('$') doit être interprété comme prompt DOS ('>'), un slash ('/') séparant les répertoires doit être converti en back slash ('\') et un tilde ('~') représente le répertoire "home" d un utilisateur (voyez l exemple de la question **4.18**)

Quelques lignes de commandes présentées dans cette FAQ sont si longues qu elles n auraient pas pu être correctement affichées dans la version Web de ce fichier. Il a donc fallu les répartir sur plusieurs lignes. Dans ce cas, ayez soin de taper tous les caractères sur une seule ligne sinon la commande échouera ou, au mieux, ne donnera pas le résultat escompté.

Gardez à l esprit que cette FAQ contient des informations pouvant ne pas s appliquer à la version de logiciel que vous possédez car de nouvelles fonctionnalités et des corrections de bogues sont continuellement ajoutées (lisez le fichier NEWS livré avec les sources ou le package pour voir les changements notables entre versions). Un point à noter : depuis la version 1.1.92 de GnuPG, le fichier contenant les options utilisateur ne se nomme plus "options" mais "gpg.conf". Les informations de la FAQ relatives au fichier options peuvent, dans beaucoup de cas, être interchangeables avec le nouveau fichier gpg.conf . Voir question **5.8** pour les détails.

2. SOURCES D INFORMATIONS

2.1) Où trouver plus d'informations sur GnuPG ?

Ressources en−ligne :

- La page de documentation se trouve à l adresse **http://www.gnupg.org/documentation/.** Voyez aussi les HOWTOs et le GNU Privacy Handbook (GPH, disponible en Anglais, Espagnol et Russe). Ce dernier constitue un guide détaillé d utilisation de GnuPG. Vous y trouverez aussi un document sur le passage de PGP 2.x à GnuPG.
- A l adresse **http://www.gnupg.org/documentation/mailing−lists.html** se trouve une archive en ligne des listes de diffusion de GnuPG. Parmi les plus intéressantes gnupg−users pour tous les problèmes des utilisateurs et gnupg−devel si vous voulez entrer en contact avec les développeurs.

De plus, on trouve des archives "cherchables" sur MARC, e.g.:
gnupg−users : **http://marc.theaimsgroup.com/?l=gnupg−users&r=1&w=2**
gnupg−devel : **http://marc.theaimsgroup.com/?l=gnupg−devel&r=1&w=2**

ATTENTION : Avant de poster dans une liste, lisez cette FAQ et la documentation disponible. De plus, recherchez dans l'archive de la liste, votre question a peut−être déjà été débattue. Ainsi vous aiderez les autres à se concentrer sur les problèmes qui n'ont pas encore été résolus.

- La distribution du source GnuPG contient un sous−répertoire :

```
./doc
```

avec de la documentation supplémentaire (surtout intéressante pour des hackers plutôt que pour l'utilisateur lambda).

2.2) Comment se procurer GnuPG ?

Vous pouvez télécharger GNU Privacy Guard depuis son serveur FTP principal **ftp://ftp.gnupg.org/gcrypt/** ou depuis les miroirs :

http://www.gnupg.org/download/mirrors.html

La version stable courante est la 1.2.1. Passez à cette version car elle intègre des caractéristiques, des fonctions et des correctifs de sécurité qui peuvent manquer aux versions précédentes.

3. INSTALLATION

3.1) Sur quels systèmes d'exploitation GnuPG fonctionne−t−il ?

Il devrait marcher sur la plupart des Unix, Windows (y compris Windows NT/2000) et Macintosh OS/X. Une liste de systèmes d'exploitation censés marcher se trouve à l'adresse :

http://www.gnupg.org/download/supported_systems.html

3.2) Quel générateur de nombres aléatoires utiliser ?

De "bons" nombres pseudo−aléatoires sont cruciaux pour la sécurité de vos chiffrements. Les différents systèmes d'exploitation fournissent des données aléatoires de plus ou moins bonne qualité. Linux et *BSD

fournissent des données aléatoires par /dev/random. C'est le meilleur choix sur ces systèmes. Les utilisateurs de Solaris avec le package SUNWski installé ont /dev/random. Dans les deux cas, utilisez l'option de configuration :

```
--enable-static-rnd=linux
```

Par ailleurs il y a le kernel random device d'Andi Maier **http://www.cosy.sbg.ac.at/~andi/**, mais il est en version bêta. A utiliser à vos risques et périls !

Sur d'autres systèmes, l' Entropy Gathering Daemon (EGD) est un bon choix. C'est un perl-daemon qui surveille l'activité du système et en déduit des données aléatoires. Voir la page **http://www.gnupg.org/download/** pour télécharger EGD. Utilisation :

```
--enable-static-rnd=egd
```

Si les options ci-dessus ne marchent pas, vous pouvez utiliser le générateur de nombre aléatoires "unix". C'est **très** lent et il vaut mieux l'éviter. Comme la qualité des nombres aléatoires n'est pas très bonne, ne l'utilisez pas pour des données sensibles.

3.3) Comment ajouter le support de RSA et IDEA ?

Le support de RSA est inclus depuis la version 1.0.3 de GnuPG.

La distribution officielle de GnuPG ne contient pas IDEA à cause d'un problème de brevet. Le brevet ne tombant dans le domaine public qu'en 2007, on ne doit pas s'attendre à un support officiel d'ici là.

Toutefois, il existe un module non-officiel à inclure, y compris dans les précédentes versions de GnuPG. Il est disponible depuis l'adresse **ftp://ftp.gnupg.dk/pub/contrib-dk/**. Recherchez :

```
idea.c.gz        (module C)

idea.c.gz.sig>    (fichier de signature)

ideadll.zip      (module C et DLL win32)

ideadll.zip.sig  (fichier de signature)
```

Les directives de compilation sont dans les entêtes de ces fichiers. Il vous faudra ajouter la ligne suivante dans votre fichier ~/.gnupg/gpg.conf ou ~/.gnupg/options :

```
load-extension idea
```

4. UTILISATION

4.1) Quelle est la taille de clés recommandée ?

1024 bits pour les signatures DSA ; même pour les signatures ElGamal normales. C'est suffisant parce que la taille du hachage est sans doute le maillon faible, du moment que la taille de clé est supérieure à 1024 bits. Les clés de chiffrement peuvent être plus grandes mais il faudrait alors vérifier leur empreinte :

```
$ gpg --fingerprint <user ID>
```

Comme algorithme de clés, vous devriez conserver celui par défaut (i.e., signature DSA et chiffrement ElGamal). Une clé de signature ElGamal présente les défauts suivants : signature plus grande, difficulté de créer une clé de signature résistant à certaines attaques, pas de gain de sécurité par rapport à DSA enfin

possibles problèmes de compatibilité avec certaines versions de PGP. Elle n'a été ajoutée que parce qu'il n'était pas clair à ce moment là que DSA ait été couvert par un brevet.

4.2) Pourquoi la génération des clés dure−t−elle parfois si longtemps ?

Le problème vient de ce que nous avons besoin de récolter beaucoup de données aléatoires (sur Linux le /dev/random device s'en charge). Il n'est vraiment pas aisé de remplir le buffer d'entropie de Linux. En en parlant à Ted Ts'o, il m'a dit que le meilleur moyen de remplir le buffer est de manipuler les touches du clavier. C'est le prix d'une bonne sécurité. Ce que je fais, c'est de taper plusieurs fois les touches Maj, Ctrl, Alt, et Verr−Maj parce que ces touches ne génèrent pas de caractères sur l'écran. Comme ça vous obtenez vos clés très rapidement (c'est pareil avec PGP2).

Un autre problème peut venir d'un autre programme qui consomme vos octets aléatoires en les lisant dans /dev/random.

4.3) En plus c'est très lent si je le fais sur une machine distante. Pourquoi ?

Ne faites surtout pas ça ! Vous ne devriez jamais créer des clés ou même utiliser GnuPG sur un système distant car vous n'avez généralement aucun contrôle physique sur votre trousseau de clés privées (la plupart du temps il est vulnérable aux attaques par dictionnaire). J'engage tout le monde à ne créer des clés que sur une machine locale (un portable déconnecté est sûrement le meilleur choix) et si vous le faites sur une machine connectée (je sais, on le fait tous), assurez−vous d'avoir un solide mot de passe à la fois sur votre compte réseau et sur votre clé privée et que vous pouvez faire confiance à l'administrateur système.

Quand je vérifie GnuPG sur une machine distante à travers ssh (je n'ai pas d'Alpha ici) ;−) c'est pareil. La création des clés est **très** longue, aussi j'utilise une option spéciale, −−quick−random, pour générer des clés peu sûres, justes bonnes pour des tests.

4.4) Quelle différence y a−t−il entre options et commandes ?

Si vous tapez : 'gpg −−help', vous aurez deux listes séparées. La première est la liste des commandes, la seconde est la liste des options. Quand vous lancez GPG, vous **devez** fournir exactement une commande (avec une exception, voir plus loin). Vous **pouvez** ajouter une ou plusieurs options. Par convention, la commande doit être à la fin de la ligne de commande après toutes les options. Si la commande concerne un fichier (toutes celles de base sont dans ce cas), le nom de fichier vient tout à la fin. Donc la manière basique de lancer gpg est :

```
$ gpg [−−option something] [−−option2] [−−option3 something]  command
file
```

Certaines options attendent des arguments. Par exemple l'option −−output (que l'on peut abréger en −o) attend un nom de fichier. L'argument doit suivre immédiatement l'option sinon gpg ne sait pas à quelle option l'argument correspond. En tant qu'option, −−output et le nom de fichier doivent être avant la commande. L'option recipient (−r) attend le nom ou l'identifiant du destinataire, il doit être tout de suite après l'argument −r. La commande −−encrypt (ou −e) vient après toutes les options, suivie par le nom du fichier à chiffrer. La ligne de commande sera donc :

```
$ gpg −r alice −o secret.txt −e test.txt
```

Il est plus facile de comprendre les options si elles sont écrites en entier :

```
$ gpg −−recipient alice −−output secret.txt −−encrypt test.txt
```

Si vous chiffrez vers un fichier texte d'extension ".txt", vous voulez sans doute avoir du texte ASCII avec armure dans le fichier (pas du binaire), pour cela vous devez ajouter l'option −−armor (−a) sans argument :

```
$ gpg --armor --recipient alice --output secret.txt  encrypt test.txt
```
Pour que ça devienne un peu plus clair, imaginez des crochets autour des options :

```
$ gpg [--armor] [--recipient alice] [--output secret.txt] --encrypt
test.txt
```
Les options peuvent être réarrangées comme vous voulez :

```
$ gpg --output secret.txt  recipient alice  armor  encrypt test.txt
```
Si le nom de votre fichier commence par un tiret ("−a.txt"), GnuPG prend ça pour une option et va protester. Pour éviter cela, soit vous utilisez "./−a.txt" ou alors vous arrêtez l'interprétation par deux tirets : "−− −a.txt".

Exception à la commande unique : signature et chiffrement en même temps. Vous pouvez combiner les deux commandes comme ceci :

```
$ gpg [--options] --sign --encrypt foo.txt
```
4.5) Je ne peux pas supprimer un identifiant utilisateur de mon trousseau de clés privées parce qu'il a déjà été supprimé de mon trousseau de clés publiques. Que faire ?

Comme vous ne pouvez sélectionner que depuis le trousseau de clés publiques, il n'y a pas de moyen direct de le faire. Cependant il n'est pas trop difficile d'y arriver. Créez un nouvel identifiant utilisateur avec exactement le même nom. Vous aurez alors deux identifiants utilisateurss identiques sur le trousseau de clés privées. Sélectionnez en un et supprimez−le. Les deux IDs sont effacés du trousseau de clés privées.

4.6) Je ne peux pas supprimer ma clé privée parce que ma clé publique a disparu. Que puis−je faire ?

Pour sélectionner une clé une recherche est toujours faite sur le trousseau de clés publiques, il n'est donc pas possible de sélectionner une clé privée sans avoir la clé publique. Normalement il ne devrait jamais arriver de disposer d'une clé privée sans la clé publique. La réalité étant ce qu'elle est, GnuPG fournit le moyen d'y remédier : Utilisez l'identifiant de clé long pour spécifier la clé à effacer. On peut l'avoir par l'option −−with−colons (cinquième champ des lignes commençant par "sec").

Si vous avez perdu votre clé publique et que vous deviez la recréer pour continuer à vous servir de votre clé privée utilisez gpgsplit comme décrit à la question **4.21**

4.7) C'est quoi trust, validity et ownertrust ?

Dans GnuPG, "ownertrust" est utilisé à la place de "trust" pour souligner que c'est la valeur attribuée à une clé pour exprimer la confiance que vous accordez à son propriétaire pour signer correctement (et donc vous présenter) d'autres clés. "validity" ou confiance calculée indique jusqu'à quel point GnuPG considère la clé valide (qu'elle appartient bien à celui qui le prétend). Voyez le chapitre "The Web of Trust" dans The GNU Privacy Handbook pour plus d'informations.

4.8) Comment signer un fichier de mise à jour ?

Utilisez "gpg clearsign not−dash−escaped". Le problème avec −−clearsign c'est que toute ligne commençant par un tiret est préfixée par "− " ; à l'évidence diff génère plein de lignes de ce genre qui sont alors préfixées ce qui n'est pas de bon goût pour un patch ;−). Pour traiter un fichier de patch sans devoir enlever la signature, on peut utiliser l'option spéciale −−not−dash−escaped pour supprimer le préfixage. Ne mettez pas un tel fichier patch dans un mail car les espaces et caractères de fin de ligne sont pris en compte

dans la signature et il n'est pas sûr qu'ils soient préservés. Si vous voulez envoyer un fichier par e−mail vous pouvez simplement le signer avec votre MUA (Mail User Agent).

4.9) Où est passée l'option "encrypt−to−self"?

Utilisez "−−encrypt−to votre_keyID". Vous pouvez utiliser plusieurs de ces options. Pour désactiver temporairement l'usage de cette clé additionnelle, utilisez l'option "−−no−encrypt−to".

4.10) Comment se débarrasser des en−têtes Version et Comment dans les messages ?

Utilisez "−−no−version comment " ". Notez la ligne vide requise par le protocole.

4.11) Que signifie "You are using the xxxx character set." ?

Ce message est affiché si une conversion UTF−8 doit avoir lieu. Assurez−vous que le jeu de caractères affiché est celui activé sur votre système. Comme l'"iso−8859−1" est le plus utilisé, c'est celui par défaut. Le jeu de caractères se change par l'option "−−charset". Si vous ne pouvez pas faire correspondre les jeux de caractères actif et affiché, restreignez−vous à l'ASCII 7bits et il n'y aura pas besoin de conversion.

4.12) Comment savoir par quelles clés a été chiffré un message ?

```
$ gpg −−batch −−decrypt −−list−only −−status−fd 1 2>/dev/null | awk
'/^[GNUPG:] ENC_TO / { print $3 }'
```

4.13) La nouvelle version de GnuPG ne déchiffre pas mes messages chiffrés symètriquement−seulement (−c).

Dans les versions de GnuPG antérieures à la 1.0.1, un bogue affectait les messages chiffrés symétriquement par 3DES ou Twofish (ça n'a jamais été le choix par défaut). Le bogue a été résolu mais pour conserver l'accès à d'anciens messages vous devriez lancer gpg avec l'option "−−emulate−3des−s2k−bug", déchiffrer puis re−chiffrer le message sans cette option. Cette option sera retirée de la version 1.1, donc re−chiffrez les messages affectés dès maintenant.

4.14) Comment utiliser GnuPG dans un environnement automatisé ?

Utilisez l'option −−batch sans utiliser de passphrase qui ne peut généralement pas être stockée plus en sécurité que dans le trousseau de clés privées lui−même. Voici le processus conseillé pour créer des clés pour un environnement automatisé :

Sur une machine sure :

1. Pour faire de la signature automatique, créez une sous−clé de signature pour votre clé (utilisez le menu interactif de modification de clé en tapant la commande : 'gpg edit−key keyID', tapez "addkey" et sélectionnez le type de clé DSA).
2. Assurez−vous d'utiliser une passphrase (nécessaire dans l'implémentation actuelle).
3. gpg −−export−secret−subkeys −−no−comment foo secring.auto
4. Copiez secring.auto et le trousseau de clés publiques dans un répertoire test.
5. Allez dans ce répertoire.
6. gpg −−homedir . −−edit foo et utilisez "passwd" pour retirer la passphrase des sous−clés. Profitez−en pour retirer les sous−clés inutiles.

7. Copiez secring.auto sur une disquette pour le transférer sur la machine cible.

Sur la machine cible :

1. Installez secring.auto comme trousseau de clés privées.
2. Vous pouvez lancer votre nouveau service. Il serait de bon ton d'installer une détection d'intrusion pour pouvoir révoquer les sous−clés installées sur cette machine et les remplacer par des neuves en cas d'intrusion réussie.

4.15) Quel client de messagerie utiliser avec GnuPG?

Le chiffrement d'e−mails est l'un des usages les plus appréciés de GnuPG. Certains clients de messagerie ou mail user agents (MUAs) supportent GnuPG à des degrés divers. En simplifiant, GnuPG peut chiffrer les mails de deux façons : "à l'ancienne" ASCII avec armure (i.e. cleartext encryption) ou dans le style RFC 2015 (anciennement PGP/MIME, OpenPGP maintenant). Ce dernier supporte complètement MIME. Certains MUAs ne supportant que l'un des deux, finalement celui que vous utiliserez dépendra de vos besoins et des capacités du destinataire. Le support peut être natif au MUA ou fournit par "plug−ins" ou des outils externes.

La liste suivante n'est pas exhaustive :

```
MUA            OpenPGP ASCII   How? (N,P,T)
-------------------------------------------------------------
Calypso           N      Y      P (Unixmail)
Elm               N      Y      T (mailpgp,morepgp)
Elm ME+           N      Y      N
Emacs/Gnus        Y      Y      T (Mailcrypt,gpg.el)
Emacs/Mew         Y      Y      N
Emacs/VM          N      Y      T (Mailcrypt)
Evolution         Y      Y      N
Exmh              Y      Y      N
GNUMail.app       Y      Y      P (PGPBundle)
GPGMail           Y      Y      N
KMail (<=1.4.x)   N      Y      N
KMail (1.5.x)     Y(P)   Y(N)   P/N
Mozilla           Y      Y      P (Enigmail)
Mulberry          Y      Y      P
Mutt              Y      Y      N
Sylpheed          Y      Y      N
Sylpheed−claws    Y      Y      N
TkRat             Y      Y      N
Xemacs/Gnus       Y      Y      T (Mailcrypt)
Xemacs/Mew        Y      Y      N
Xemacs/VM         N      Y      T (Mailcrypt)
Xfmail            Y      Y      N

N   Native, P − Plug−in, T − External Tool
```

La table ci−dessous liste les MUAs propriétaires. Le GNU Project conseille de ne pas les utiliser, ils sont listés pour l'interopèrabilité et par commodité.

```
MUA            OpenPGP ASCII   How? (N,P,T)
-------------------------------------------------------------
Apple Mail        Y      Y      P (GPGMail)
Becky2            Y      Y      P (BkGnuPG)
Eudora            Y      Y      P (EuroraGPG)
Eudora Pro        Y      Y      P (EudoraGPG)
Lotus Notes       N      Y      P
Netscape 4.x      N      Y      P
```

```
Netscape 7.x      Y      Y      P (Enigmail)
Novell Groupwise  N      Y      P
Outlook           N      Y      P (G-Data)
Outlook Express   N      Y      P (GPGOE)
Pegasus           N      Y      P (QDPGP,PM-PGP)
Pine              N      Y      T (pgpenvelope,(gpg|pgp)4pine)
Postme            N      Y      P (GPGPPL)
The Bat!          N      Y      P (Ritlabs)
```

Un bon aperçu du support d'OpenPGP se trouve aux adresses :
http://cryptorights.org/pgp-users/resources/pgp-mail-clients.html,
http://www.geocities.com/openpgp/courrier_en.html, et
http://www.bretschneidernet.de/tips/secmua.html.

Les utilisateurs de MUAs Win32 sans support OpenPGP peuvent essayer GPGrelay **http://gpgrelay.sourceforge.net**, un petit serveur-relais de courrier utilisant GnuPG pour permettre à des clients de messagerie d'envoyer et recevoir des e-mails conformes à PGP-MIME (RFC 2015).

4.16) Pourrions-nous avoir une bibliothèque gpg ?

Ca a été souvent demandé. Cependant, du point de vue des mainteneurs de GnuPG, cela conduirait à des problèmes de sécurité et ne sera donc pas implémenté dans un futur prévisible. Toutefois pour certains types d'applications gpgme pourrait convenir. Vous le trouverez à l'adresse :
ftp://ftp.gnupg.org/gcrypt/alpha/gpgme.

4.17) J'ai créé un certificat de révocation mais je ne sais pas comment l'envoyer aux serveurs de clés.

La plupart des serveurs de clés n'acceptent pas un simple certificat de révocation. Il vous faut d'abord importer le certificat dans gpg :

```
$ gpg --import my-revocation.asc
```
puis envoyer la clé révoquée aux serveurs de clés :

```
$ gpg --keyserver certserver.pgp.com  send-keys mykeyid
```
(ou utiliser une interface web de serveur de clés).

4.18) Comment mettre mon trousseau de clés dans un autre répertoire ?

GnuPG stocke plusieurs fichiers dans un répertoire à lui. Cela comprend les fichiers d'options, pubring.gpg, secring.gpg, trustdb.gpg, et d'autres. GnuPG crée et utilise toujours ces fichiers. Sur les systèmes UNIX le répertoire est généralement ~/.gnupg ; dans Windows "C:\gnupg".

Si vous voulez mettre vos trousseaux de clés ailleurs, utilisez l'option :

```
--homedir /mon/chemin/
```
pour que GnuPG crée tous ses fichiers dans ce répertoire. Votre trousseau de clés sera "/mon/chemin/pubring.gpg". Vous pouvez ainsi stocker vos secrets sur disquette. N'utilisez pas "--keyring" dont le but est de spécifier des fichiers de clés supplémentaires.

4.19) Comment vérifier des packages signés?

Avant de pouvoir vérifier la signature d'un package, vous devez importer dans votre trousseau de clés publiques celle du vendeur, de l'organisation ou de la personne qui publie le package. La clé doit aussi être validée (ou signée localement) faute de quoi GnuPG affichera des messages d'avertissement.

Il faudra aussi télécharger le fichier de signature séparée en même temps que le package. D'habitude ce fichier porte le même nom que le package avec l'extension binaire (.sig) ou ASCII avec armure (.asc).

Une fois la clé importée, le package et sa signature téléchargés, utilisez :

```
$ gpg --verify sigfile signed-file
```

Si le fichier signature porte le même nom de base que le package, celui-ci peut être vérifié en indiquant juste le fichier de signature, GnuPG déduira de lui-même le nom du package(en enlevant l'extension .sig ou .asc). Par exemple, pour vérifier un package nommé foobar.tar.gz à l'aide de son fichier de signature séparée, utilisez :

```
$ gpg --verify foobar.tar.gz.sig
```

4.20) Comment exporter seulement certaines signatures d'un trousseau de clés ?

Si vous désirez créer un trousseau de clés avec un sous-ensemble des signatures d'un trousseau maître (pour un club, un groupe d'utilisateurs ou un service de société par exemple), spécifiez simplement les clés que vous voulez exporter :

```
$ gpg --armor  export key1 key2 key3 key4 > keys1-4.asc
```

4.21) J'ai toujours ma clé privée mais j'ai perdu ma clé publique. Que faire?

Toutes les clés privées OpenPGP contiennent une copie de la clé publique correspondante et dans le pire des cas, vous pouvez recréer une nouvelle clé publique à partir de la clé privée.

Un outil convertissant une clé privée en clé publique est disponible (en fait une nouvelle option pour gpgsplit) dans les versions 1.2.1 et suivantes de GnuPG (ou peut être trouvé dans le CVS). Cela marche comme ceci :

```
$ gpgsplit --no-split  secret-to-public secret.gpg >publickey.gpg
```

On devrait essayer d'abord d'exporter la clé privée et de ne convertir que celle là. Ca devrait marcher aussi avec le trousseau complet. Ensuite, le fichier publickey.gpg peut être importé dans GnuPG comme d'habitude.

4.22) Les messages signés-clairs envoyés par mon compte de messagerie web ont une signature invalide. Pourquoi?

Assurez-vous que votre compte de messagerie web ne formate pas le message en HTML. Cela peut altérer le message avec des tags ou des espaces qui résultent en une signature invalide. Le destinataire doit pouvoir copier le message signé dans un fichier texte pour le vérifier ou le service d'e-mail doit vous permettre de transmettre le message signé comme pièce jointe si on ne peut utiliser du texte brut.

5. PROBLEMES DE COMPATIBILITE

5.1) Comment chiffrer un message avec GnuPG pour que PGP puisse le déchiffrer ?

Cela dépend de la version de PGP.

- PGP 2.x
  Vous ne pouvez pas car PGP 2.x utilise normalement IDEA non supporté par GnuPG parce qu'il est breveté (voir **3.3**). Cependant si vous avez une version modifiée de PGP essayez ceci :
  ```
  $ gpg --rfc1991 --cipher-algo 3des...
  ```
  N'envoyez pas à la suite les données à chiffrer vers gpg, utilisez un fichier sinon PGP 2 ne s'y retrouvera pas.

  Vous ne pouvez pas faire de chiffrement 'conventionnel' pour PGP 2.
- PGP 5.x et supérieurs
  Il vous faut deux options supplémentaires :
  ```
  --compress-algo 1 --cipher-algo cast5
  ```
  Vous pouvez aussi utiliser "3des" au lieu de "cast5", "blowfish" ne marche pas avec toutes les versions de PGP 5. Vous pouvez aussi mettre :

  ```
  Compress-algo 1
  ```
  dans votre fichier ~/.gnupg/options cela n'affecte pas le fonctionnement normal de GnuPG.

  C'est valable également pour le chiffrement 'conventionnel'.

5.2) Comment migrer de PGP 2.x vers GnuPG ?

PGP 2 utilise les algorithmes de chiffrement RSA et IDEA. Alors que depuis la version 1.0.3 GnuPG intègre RSA dont le brevet a expiré, l'algorithme IDEA est breveté jusqu'en 2007. Sous certaines conditions, vous pouvez utiliser IDEA dès à présent. Dans ce cas, référez-vous à la question **3.3** pour savoir comment ajouter le support d'IDEA à GnuPG et lisez **http://www.gnupg.org/gph/en/pgp2x.html** pour faire la migration.

5.3) (supprimé)

Vide puisqu'il a été supprimé ;-))

5.4) Pourquoi PGP 5.x ne chiffre-t-il pas les messages avec certaines clés ?

PGP, Inc. n'accepte pas les clés ElGamal de type 20 même en chiffrement. Ils ne supportent que le type 16 (identique au moins pour le déchiffrement). Pour être plus compatible, GnuPG (depuis la version 0.3.3) utilise aussi le type 16 pour la sous-clé ElGamal créée si l'algorithme de clé par défaut est choisi. Vous pouvez ajouter à votre clé publique une clé ElGamal type 16 puisque vos signatures de clé sont encore valides.

5.5) Pourquoi PGP 5.x n'arrive-t-il pas à vérifier mes messages ?

PGP 5.x n'accepte pas les signatures v4 pour les données alors que OpenPGP requiert la génération de signatures v4 pour tout type de données. C'est pourquoi GnuPG les utilise par défaut. Utilisez l'option "--force-v3-sigs" pour générer des signatures v3 pour les données.

5.6) Comment transferer les valeurs owner trust de PGP à GnuPG ?

Pour vous aider, il y a un script dans le répertoire tools. Après avoir importé le trousseau de clés de PGP, tapez cette commande :

```
$ lspgpot pgpkeyring | gpg --import-ownertrust
```

où pgpkeyring est le trousseau de clés original, pas celui de GnuPG que vous pourriez avoir créé à l'étape précédente.

5.7) PGP n'aime pas ma clé privée.

Certains paquets de commentaires utilisés par GnuPG vous font sûrement jeter par les plus vieux PGPs. Ces paquets sont parfaitement admis par OpenPGP mais PGP n'est pas réellement compatible OpenPGP. L'astuce consiste à exporter les clés privées avec cette commande :

```
$ gpg --export-secret-keys  no-comment -a your-KeyID
```

Autre possibilité : par défaut, GnuPG chiffre votre clé privée avec l'algorithme symétrique Blowfish. Les plus anciens PGPs ne comprennent que 3DES, CAST5, ou IDEA. Par la méthode suivante vous pouvez re-chiffrer votre clé privée gpg avec un autre algorithme :

```
$ gpg --s2k-cipher-algo=CAST5 --s2k-digest-algo=SHA1  compress-algo=1
--edit-key <username>
```

Puis utilisez passwd pour changer le mot de passe (réutilisez le même, cette fois il va chiffrer la clé avec CAST5).

Maintenant exportez-la et PGP devrait pouvoir la manipuler.

Avec PGP 6.x l'option suivante marche pour exporter une clé :

```
$ gpg --s2k-cipher-algo 3des --compress-algo 1 --rfc1991
--export-secret-keys <KeyID>
```

5.8) GnuPG n'installe plus de fichier ~/.gnupg/options. C'est un oubli ?

Non. Depuis la version 1.1.92 le fichier ~/.gnupg/options s'appelle ~/.gnupg/gpg.conf pour les nouvelles installations. Pour les mises à jour, si un fichier ~/.gnupg/options existe, il continue d'être utilisé, mais ce changement a été nécessaire pour avoir un nommage cohérent avec les outils à venir. Un fichier existant peut être renommé en gpg.conf si vous faites une mise à jour ou que vous recevez le message indiquant que l'"old default options file" est ignoré (cas où un fichier gpg.conf et un fichier options existent).

5.9) Comment exporter les clés GnuPG pour s'en servir avec PGP ?

Question soulevée assez souvent, voici comment faire :

PGP peut (pour la plupart des clés) utiliser des clés privées générées par GnuPG. Les problèmes surgissant parfois viennent surtout de ce que GnuPG supporte davantage de caractéristiques du standard OpenPGP. Si votre clé privée utilise une de ces caractéristiques, PGP la rejettera ou vous aurez des problèmes de communication plus tard. Notez que PGP ne crée pas de clés de signatures ElGamal, elles sont donc inutilisables dans toutes les versions.

Ces instructions devraient marcher avec GnuPG 1.0.7 et suivants et PGP 7.0.3 et suivants.

Commencez par modifier la clé. Les valeurs par défaut convenant, la plupart de ces options ne sont pas nécessaires, mais si vous aviez autre chose dans votre fichier d'options ça ne peut pas faire de mal.

```
$ gpg --s2k-cipher-algo cast5 --s2k-digest-algo sha1 --s2k-mode 3
--simple-sk-checksum --edit KeyID
```
Désactivez certaines fonctions. Mettez des types que PGP sache utiliser dans la liste des algorithmes préférés pour le chiffrement, le hachage et la compression. (Oui, je sais, c'est une drôle de liste mais c'est ce que PGP utilise, moins IDEA).

```
> setpref S9 S8 S7 S3 S2 S10 H2 H3 Z1 Z0
```
Ensuite appliquez la liste de préférences à la clé.

```
> updpref
```
<; >Enfin il vous faut déchiffrer et re-chiffrer la clé, en vous assurant de la chiffrer avec un codage du goût de PGP. C'est ce que vous avez préparé avec la commande edit ci-dessus, il suffit donc de changer la passphrase pour la faire prendre effet. Vous pouvez réutiliser la même passphrase ou en profiter pour la changer.

```
> passwd
```
Sauvegardez votre travail.

```
> save
```
Maintenant vous pouvez l'exporter normalement :

```
$ gpg --export KeyID > mypublickey.pgp --export-secret-key KeyID >
mysecretkey.pgp
```
Merci à David Shaw pour ces informations !

6. PROBLEMES ET MESSAGES D'ERREURS

6.1) Pourquoi ai-je ce message "gpg: Warning: using insecure memory!" ?

Sur beaucoup de systèmes GPG doit être installé en étant équivalent root. C'est nécessaire pour verrouiller des pages mémoire. Le verrouillage de pages mémoire empêche le système d'exploitation de les écrire sur disque et garde donc votre clé privée vraiment secrète. Si vous ne recevez pas ce type de message c'est que votre système d'exploitation supporte le verrouillage sans être root. Le programme abandonne les privilèges de root dès que la mémoire verrouillée est allouée.

Pour donner les droits équivalents root à l'exécutable GPG vous pouvez utiliser soit :

```
$ chmod U+s /path/to/gpg
```
ou

```
$ chmod 4755 /path/to/gpg
```
Pour des raisons de sécurité, certains interdisent d'utiliser setuid(root) sauf nécessité absolue. Voyez votre administrateur système si vous ne pouvez pas le faire vous-même.

Sur UnixWare 2.x et 7.x vous devez installer GnuPG avec le privilège 'plock' pour obtenir le même résultat :

```
$ filepriv -f plock /path/to/gpg
```
Si vous ne pouvez ou ne voulez pas donner les droits équivalents root à GPG utilisez l'option "--no-secmem-warning" ou ajoutez :

`no-secmem-warning`
dans votre fichier ~/.gnupg/options ou ~/.gnupg/gpg.conf (ça désactive l'avertissement).

Sur certains systèmes (e.g., Windows), GnuPG ne verrouille pas les pages mémoire et les versions antérieures à 1.0.4 de GnuPG émettent l'avertissement :

`gpg: Please note that you don't have secure memory`
Cet avertissement était considéré comme un problème trop grave pour pouvoir être désactivé par l'option ci-dessus. Finalement, comme il perturbait trop les utilisateurs, il a été ôté.

6.2) Large File Support ne marche pas...

LFS marche bien dans les versions postérieures à 1.0.4. Si configure ne le détecte pas, essayez un autre (meilleur) compilateur. Egcs 1.1.2 marche, certains autres gccs non. BTW, un certain nombre de problèmes de compilation de GnuPG 1.0.3 et 1.0.4 sur HP-UX ou Solaris étaient dus à un mauvais support de LFS.

6.3) Dans le menu edit, pourquoi les valeurs de confiance (trust) ne sont-elles pas affichées correctement après signature des identifiants utilisateurs ?

Cela se produit parce que certaines valeurs sont enregistrées immédiatement dans trustdb mais que le calcul des valeurs de confiance ne peut se faire qu'après la sauvegarde. C'est un défaut de conception difficile à éliminer mais il sera résolu dans une version ultérieure.

6.4) Que signifie "skipping pubkey 1: already loaded" ?

Depuis GnuPG 1.0.3, l'algorithme RSA est inclus. S'il reste une ligne "load-extension rsa" dans le fichier d'options, ce message apparaît.Retirez simplement cette ligne du fichier d'options.

6.5) GnuPG 1.0.4 ne crée pas ~/.Gnupg...

Bogue identifié et corrigé dans les versions plus récentes.

6.6) Depuis la version 1.0.2 une signature ElGamal ne se vérifie plus...

Utilisez l'option --emulate-md-encode-bug.

6.7) Les anciennes versions de GnuPG ne savent pas vérifier les signatures ElGamal

Passez à la version 1.0.2 ou plus récente de GnuPG.

6.8) Quand j'utilise --clearsign, le texte clair présente des tirets en trop. Pourquoi ?

C'est du texte dash-escaped et c'est exigé par OpenPGP. Ca se produit chaque fois qu'une ligne commence par un tiret ("-"), le but est de s'assurer que seules les lignes structurant le texte ou la signature (i.e., "-----BEGIN PGP SIGNATURE-----") commencent par 2 tirets.

Quand GnuPG traite ces messages il retire les tirets supplémentaires. Un bon client de messagerie les ôte

également quand il affiche ce type de message.

6.9) D'où sort ce "can't handle multiple signatures" ?

A cause de formats de messages différents, GnuPG ne parvient pas toujours à séparer sans ambiguïté les différentes parties d'un fichier avec signatures multiples. Ce message d'erreur indique que quelque chose ne va pas dans ce fichier.

Le seul moyen de mettre plusieurs signatures dans un fichier est d'utiliser le format OpenPGP avec one−pass−signature packets (qui est par défaut dans GnuPG) ou le format texte signé−clair.

6.10) Quand je soumets une clé à un serveur de clés rien ne se passe...

Vous utilisez sans doute GnuPG 1.0.2 ou plus ancien sur Windows. Cette fonction n'y est pas mais c'était une erreur de ne pas le signaler. Les nouvelles versions affichent un avertissement. Passez à la version 1.0.4 ou plus.

6.11) J'ai "gpg: waiting for lock..."

Une instance précédente de gpg s'est probablement terminée anormalement en laissant un fichier de verrouillage. Cherchez les fichiers *.lock dans ~/.gnupg et effacez−les.

6.12) Les anciens gpg (e.g., 1.0) ont des problèmes avec les clés des gpg récents...

Depuis la version 1.0.3, gpg crée des clés avec préférence à TWOFISH (et AES depuis la 1.0.4). Cela signifie qu'elles peuvent utiliser la nouvelle méthode de chiffrement MDC. Ce sera bientôt ajouté à OpenPGP et PGP 7 le supporte aussi. Cette nouvelle méthode protège d'une (pas très nouvelle) attaque contre tous les systèmes de chiffrement d'e−mail.

En contrepartie les versions d'avant la 1.0.3 de gpg rencontrent des difficultés avec les nouvelles clés. De toutes façons, pour des raisons de sécurité et de corrections de bogues, vous devriez maintenir votre GnuPG dans une version récente. A défaut vous pouvez forcer gpg à utiliser un algorithme plus ancien en mettant la ligne :

```
cipher−algo cast5
```
dans votre fichier d'options.

6.13) La version 1.0.4 me donne "this cipher algorithm is deprecated..."

Si vous venez de générer une clé et que vous receviez ce message pendant son chiffrement, félicitations, vous venez de voir un bogue de la version 1.0.4. Elle utilise le nouvel algorithme Rijndael qui est marqué à tord "désapprouvé". Ignorez l'avertissement. Les versions récentes de gpg ont été corrigées.

6.14) Certaines dates s'affichent ????−??−??. Pourquoi ?

Par suite de contraintes de la plupart des implémentations de libc, les dates après le19−1−2038 ne sont pas correctement affichées. Les systèmes d'exploitation 64−bits ne sont pas affectés. Plutôt que d'afficher des dates erronées, GnuPG met des points d'interrogation. Pour voir la valeur correcte utilisez les options −−with−colons et −−fixed−list−mode.

6.15) J'ai toujours un probleme. Comment signaler un bogue ?

Etes−vous sur qu'il n'est pas déjà mentionné dans les listes de diffusion ? Avez−vous jeté un Sil à la liste de bogues (il y a un lien vers la liste des bogues connus sur la page de documentation). Si vous n'êtes pas sûr que c'est un bogue, envoyez un mail à la liste gnupg−devel. Sinon utilisez le système de suivi de bogue de GUUG : **http://bugs.guug.de/Reporting.html**.

6.16) Pourquoi GnuPG ne supporte−t−il pas les certificats X.509 ?

Avant tout GnuPG est une implémentation du standard OpenPGP (RFC 2440) qui est une infrastructure concurrente et différente de X.509.

Ce sont tous deux des cryptosystèmes à clés publiques mais ces clés sont gérées différemment.

6.17) Pourquoi les caractères nationaux de mon identifiant utilisateur ont−ils l'air bizarre ?

En suivant OpenPGP, GnuPG encode l'identifiant utilisateur (et d'autres chaînes de caractères) en UTF−8. Dans ce codage Unicode, la plupart des caractères nationaux sont représentés par des suites de 2 ou 3 octets. Par exemple å (0xE5 en ISO−8859−1) devient Ã¥ (0xC3, 0xA5). Ce peut être la raison pour laquelle des serveurs de clés ne trouvent pas votre clé.

6.18) J'ai des erreurs 'sed' quand j'exécute ./configure sur Mac OS X...

Ce sera corrigé après mise à jour de GnuPG en autoconf−2.50. D'ici là recherchez la ligne CDPATH dans le script de configuration et mettez en dessous la ligne :

```
unset CDPATH
```

6.19) Pourquoi les trousseaux de clés de GnuPG 1.0.7 plantent−ils GnuPG 1.0.6 ?

GnuPG 1.0.6 a un petit bogue qui fait qu'il n'analyse pas bien les paquets de confiance. Si vous ne pouvez pas le mettre à jour, appliquez ce patch :

http://www.gnupg.org/developer/gpg−woody−fix.txt

6.20) Je suis passé à la version 1.0.7 de GnuPG, il met plus longtemps à charger mes trousseaux de clés. Que faire ?

Pour supporter les signatures v3, la méthode de stockage de l'état des signatures a changé. Vous pouvez utiliser la nouvelle commande de migration −−rebuild−keydb−caches qui accélère beaucoup d'opérations sur les trousseaux de clés existants.

7. SUJETS AVANCES

7.1) Comment tout cela marche−t−il ?

Pour générer une paire de clés publique/privée lancez :

```
$ gpg −−gen−key
```

et choisissez les valeurs par défaut.

Les données chiffrées par une clé publique peuvent seulement être déchiffrées par la clé privée correspondante. La clé privée est protégée par mot de passe mais pas la clé publique.

Pour envoyer un message à un ami, vous chiffrez votre message avec sa clé publique et il pourra le déchiffrer en donnant le mot de passe et en utilisant sa clé privée.

GnuPG permet aussi de signer. Les données chiffrées par la clé privée sont déchiffrées par la clé publique. Pour signer quelque chose, une empreinte des données est relevée et cette empreinte est en quelque sorte chiffrée par la clé privée. Quelqu'un disposant de la clé publique pourra vérifier que cela vient bien du propriétaire de la clé privée et que les données n'ont pas été altérées en vérifiant l'empreinte par la clé publique.

Un trousseau de clés est simplement un fichier qui stocke des clés. Vous avez un trousseau de clés publiques où se trouvent vos clés publiques et celles de vos amis. Vous avez aussi un trousseau de clés privées avec vos clés privées. Il faut en prendre grand soin. Jamais au grand jamais n'en laissez l'accès à quiconque et protégez son contenu par une ***bonne*** passphrase.

Vous pouvez chiffrer des données de manière 'conventionnelle' en utilisant l'option 'gpg −c'. Elles sont chiffrées en utilisant une passphrase, sans utilisation de clés publiques/privées. Pour déchiffrer, la personne à qui vous envoyez les données doit connaître la passphrase. C'est des plus utile pour chiffrer des choses pour vous−même, encore que vous puissiez faire la même chose avec votre clé publique. Ca devrait être utilisé avec des correspondants que vous connaissez et avec qui vous pouvez aisément échanger des passphrases (e.g. petit ami(e), femme). L'avantage est que vous pouvez changer la passphrase de temps en temps et ainsi réduire le risque que d'anciens messages soient lisibles par quelqu'un ayant eu accès à votre passphrase.

Avec les options 'gpg import' et 'gpg −−export' vous pouvez ajouter et copier des clés depuis ou vers votre trousseau de clés. 'gpg export−secret−keys' exporte vos clés privées. Normalement c'est inutile mais vous pouvez générer la clé sur une machine et la transférer vers une autre.

Avec l'option 'gpg −−edit−key' on peut signer des clés. Quand vous signez une clé, vous affirmez que vous êtes sûr que la clé appartient à la personne dont elle est censée provenir. Vous devez être parfaitement sûr que c'est la bonne personne : Vérifiez donc l'empreinte de la clé avec :

```
$ gpg −−fingerprint KeyID
```

par téléphone (si vous connaissez bien la voix de cette personne), lors d'une séance de signatures de clés (souvent organisée lors de salons informatiques), ou a une réunion de votre groupe local d'utilisateurs GNU/Linux User.

Hmm, quoi d'autre. Vous pouvez utiliser l'option '−o nom_de_fichier' pour forcer la sortie vers un fichier de ce nom(utilisez '−' pour forcer la sortie vers stdout). '−r' vous permet de spécifier le destinataire (dont vous utilisez la clé publique pour chiffrer) en ligne de commande au lieu de le taper interactivement.

Ah ouais, important. Les données sont chiffrées par défaut en format binaire plutôt bizarre. Si vous voulez les voir sous forme lisible en texte ASCII, ajoutez l'option '−a' option. Cependant la meilleure méthode c'est d'utiliser un lecteur d'e−mail compatible MIME (Mutt, Pine et plein d'autres).

Il existe une petite faille de sécurité dans le système OpenPGP (et donc dans GnuPG) ; pour l'éviter vous devriez toujours signer et chiffrer vos messages au lieu de juste les chiffrer.

7.2) Pourquoi certaines signatures avec clé ELG−E sont−elles valides ?

Ce sont des clés ElGamal générées par GnuPG en paquets v3 (RFC 1991). La proposition de standard OpenPGP a changé par la suite l'identificateur d'algorithme pour les clés ElGamal de chiffrement et signature de16 à 20. GnuPG génère des clés type 20 pour les nouvelles clés ElGamal mais continue d'accepter les type 16 (qui sont d'après OpenPGP "chiffrement seulement") si elles sont dans un paquet v3. Comme GnuPG est le seul programme à avoir utilisé ces clés ElGamal v3 cette supposition est assez sûre.

7.3) Comment marche le système de confiance ?

Plus ou moins comme dans PGP. La différence c'est que la confiance est calculée au moment où on en a besoin. C'est une des raisons de l'existence de trustdb qui contient une liste de signatures de clés valides. A moins d'être en mode batch, il vous sera demandé d'assigner un paramètre de confiance à une clé.

Vous pouvez voir la validité (valeur de la confiance calculée) par cette commande :

```
$ gpg −−list−keys −−with−colons
```

Si le premier champ est "pub" ou "uid", le second affiche la confiance :

```
o = Inconnu (cette clé est nouvelle dans le système)
e = Clé expirée
q = Non défini (pas de valeur assignée)
n = Ne pas faire confiance du tout à cette clé
m = Faire un peu confiance à cette clé
f = Confiance totale dans cette clé
u = Confiance extrême; seulement pour les clés dont la clé privée est disponible.
r = Clé révoquée
d = Clé désactivée
```

La valeur de la ligne "pub" est la meilleure des valeurs de toutes les lignes "uid". Vous pouvez avoir la liste des valeurs de confiance assignées (la confiance que vous accordez à son propriétaire pour bien signer la clé de quelqu'un d'autre) avec :

```
$ gpg −−list−ownertrust
```

Le premier champ est l'empreinte de la clé primaire, le second est la valeur assignée :

```
− = No pas de valeur assignée ou calculée.
n = Aucune confiance à ce porteur de clé pour vérifier les signatures des autres.
m = On peut un peu faire confiance à ce porteur de clé pour vérifier les signatures des autr
f = Assume que ce porteur de clé sait vraiment signer des clés.
u = Inutile de nous faire confiance à nous−mêmes, nous avons la clé privée.
```

Gardez confidentielles ces valeurs, elles expriment vos opinions sur les autres. PGP range ces informations dans le trousseau de clés, publier un trousseau de clés au lieu de l'exporter n'est donc pas une bonne idée. GnuPG lui, les stocke dans le fichier trustdb.gpg, on peut donc distribuer un trousseau de clés gpg (il a aussi la commande export).

7.4) Qu'est−ce c'est : "key C26EE891.298, uid 09FB: ...." ?

C'est la représentation interne d'un identifiant utilisateur dans le fichier trustdb. "C26EE891" est l'identifiant de clé, "298" l'identifiant local (numéro d'enregistrement dans trustdb) et "09FB" les 2 derniers octets d'un hachage ripe−md−160 de l'identifiant utilisateur pour cette clé.

7.5) Interprétation de certaines informations affichées

Quand il vérifie la validité d'une clé, GnuPG affiche parfois des informations précédées de l'indication de ce qu'il est en train de vérifier.

`"key 12345678.3456"`
C'est la clé d'identifiant 12345678 et de numéro interne 3456, numéro d'enregistrement dans l'espèce de répertoire de trustdb.

`"uid 12345678.3456/ACDE"`
identifiant utilisateur de la même clé. Pour identifier l' identifiant utilisateur les 2 derniers octets d'un hachage ripe−md−160 de l'identifiant utilisateur est affiché.

`"sig 12345678.3456/ACDE/9A8B7C6D"`
La signature par la clé d'identifiant 9A8B7C6D pour la clé et l'utilisateur ci−dessus, si c'est une signature directe sur une clé, la partie utilisateur est vide (..//..).

7.6) Les lignes d'en−tête d'un texte signé−clair font−elles partie de ce qui est signé ?

Non. Vous pouvez par exemple ajouter ou enlever des lignes de commentaires. Elles ont le même but que les lignes d'en−tête dans les e−mail. Toutefois une ligne "Hash:" est requise pour les signatures openPGP afin d'indiquer à l'analyseur quel algorithme de hachage utiliser.

7.7) C'est quoi la liste des algorithmes préférés ?

C'est une liste d'algorithmes de chiffrement, hachage et compression stockée dans l'auto−signature d'une clé pendant sa génération. Quand vous chiffrez un document, GnuPG utilise cette liste (qui fait alors partie d'une clé publique) pour savoir quels algorithmes utiliser. Plus simplement la liste indique aux autres quels algorithmes le destinataire peut gérer et dans quel ordre de préférence.

7.8) Comment changer la liste des algorithmes préférés ?

Avec la version 1.0.7 ou supérieure, utilisez le menu edit et établissez la nouvelle liste de préférences avec la commande "setpref"; la syntaxe de cette commande ressemble à la réponse de la commande "pref". Les préférences ne sont pas changées tout de suite mais seront utilisées quand un nouvel identifiant utilisateur sera créé. Pour mettre à jour les préférences des identifiants utilisateurss existants, sélectionnez celui ou ceux que vous voulez (ou aucun pour les mettre tous à jour) et tapez la commande "updpref". Notez que l'heure de l'auto−signature est augmentée d'une seconde quand on utilise cette commande.

8. REMERCIEMENTS

Tous les remerciements à Nils Ellmenreich qui a assuré la maintenance de cette FAQ pendant si longtemps, Werner Koch pour le fichier de FAQ original et à tous ceux qui postent dans gnupg−users et gnupg−devel. Ce sont eux qui ont fourni la plupart des réponses.

Merci aussi à Casper Dik qui a fourni un script pour générer cette FAQ (il s'en sert pour l'excellente FAQ Solaris2).

Written by Werner Koch (2002−11−03 11:00).
Generated with WML 2.0.8 (30−Oct−2001) on 2003−01−07 10:15:05
from source file `faqs.wml`, $Revision: 1.7 $, $Date: 2003/01/07 10:09:35 $
Site designed by LoLo

Pour toute question suivez ces instructions