Le manuel de GNU Privacy Guard

Le manuel de GNU Privacy Guard

Merci d'envoyer les questions, suggestions et rapports de bug à propos de ce manuel au mainteneur, Mike Ashley (<jashley@acm.org>). Envoyez vos commentaires et vos corrections concernant la traduction à Jean-François Paris (<paris.jean-francois@bigfoot.com>). Merci de spécifier la version du manuel à laquelle vous faîtes référence. Utilisez la référence suivante: `$Name: $`.

Matthew Copeland, Joergen Grahn, et David A. Wheeler ont contribué a la rédaction de ce manuel. J Horacio MG a traduit ce manuel en espagnol. Merci à l'équipe de traduction de l'ALDIL, à Frédéric Delanoy pour la relecture. Que ceux que j'aurai oublié se manifestent.

Table des matières

Liste des illustrations

Chapitre 1. Pour démarrer.

GnuPG est un outil pour communiquer de manière sûre. Ce chapitre permet de couvrir l'ensemble des fonctionnalités importantes de GnuPG afin de pouvoir démarrer rapidement. Cela inclut la création, l'échange et la vérification des paires de clés, le chiffrement et le déchiffrement des documents, et pour finir l'authentification avec des signatures numériques. Il ne traite pas en détail des concepts qui sont derrière la cryptographie à clé publique, le chiffrement et les signatures numériques. Ceci est traité au chapitre 2. Il n'explique pas non plus comment utiliser GnuPG de manière avisée. Ceci est traité aux chapitres 3 et 4.

GnuPG utilise la cryptographie à clé publique de façon à ce que les utilisateurs puissent communiquer de manière sûre. Dans un système à clé publique, chaque utilisateur possède une paire de clés constituée d'une *clé privée* et d'une *clé publique*. La clé privée de l'utilisateur est gardée secrète, elle ne doit pas être révélée. La clé publique peut être distribuée à toute personne avec qui l'utilisateur souhaite communiquer. GnuPG utilise un système un peu plus sophistiqué dans lequel un utilisateur possède une paire de clés primaire et zéro ou plusieurs paires de clés additionnelles. La clé primaire et les clés additionnelles sont empaquetées pour faciliter la gestion des clés. Un paquet de clés peut être dans la plupart des cas considéré comme une simple paire de clés.

Générer une nouvelle paire de clés

L'option de ligne de commande `--gen-key` est utilisée pour créer une nouvelle paire de clés.

```
alice% gpg --gen-key
gpg (GnuPG) 0.9.4; Copyright (C) 1999 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.

Please select what kind of key you want:
   (1) DSA and ElGamal (default)
   (2) DSA (sign only)
   (4) ElGamal (sign and encrypt)
Your selection?
```

GnuPG est capable de créer de nombreux types différents de paires de clés, mais une paire de clés primaire doit être capable de générer des signatures. C'est la raison pour laquelle il n'y a que 3 choix possibles. Le choix numéro 1 crée en fait deux paires de clés. La paire de clés de type DSA est la paire de clés primaire, utilisable seulement pour signer. Une paire de clés additionnelle de type ElGamal est aussi créée pour le chiffrement. Le choix numéro 2 est similaire, à la différence que seule la paire de clés DSA est créée. Le choix numéro 4[1] crée une paire de clés ElGamal utilisable pour générer des signatures, mais aussi pour le chiffrement. Dans tous les cas, il est possible d'ajouter après coup des clés additionnelles pour le chiffrement et les signatures. Pour la plupart des utilisateurs, l'option par défaut est conseillée.

Vous devez aussi choisir une taille pour la clé. La taille d'une clé DSA doit être comprise entre 512 et 1024 bits, et la clé ElGamal peut être d'une taille quelconque. Cependant GnuPG requiert que toutes les clés aient une taille supérieure ou égale à 768 bits. Par conséquent, si vous avez choisi le choix 1 et une taille de clé supérieure à 1024 bits, la clé ElGamal sera de la taille demandée, mais la clé DSA sera de 1024 bits.

```
About to generate a new ELG-E keypair.
```

```
              minimum keysize is  768 bits
              default keysize is 1024 bits
    highest suggested keysize is 2048 bits
What keysize do you want? (1024)
```

Plus la clé est longue, plus elle sera résistante face à une attaque par la force brute, mais pour presque tous les usages, la taille de clé par défaut est adéquate puisqu'il coûterait moins cher de contourner le chiffrement que d'essayer de le casser. De plus, le chiffrement et le déchiffrement seront d'autant plus longs que la taille de la clé augmente, et une clé plus longue peut aussi influer sur la taille de la signature. Une fois choisie, la taille de la clé ne peut être changée.

Enfin, vous devez choisir une date d'expiration. Si le choix 1 est fait, la date d'expiration sera valable pour la paire de clés ElGamal et pour la paire de clés DSA.

```
Please specify how long the key should be valid.
         0 = key does not expire
      <n>  = key expires in n days
      <n>w = key expires in n weeks
      <n>m = key expires in n months
      <n>y = key expires in n years
Key is valid for? (0)
```

Pour la plupart des utilisateurs se satisferont d'une clé qui n'expire jamais. La date d'expiration doit être choisie avec soin, car, bien qu'il soit possible de la changer après que la clé ait été créée, il est difficile de faire parvenir la clé avec la date mise à jour aux utilisateurs qui possèdent déjà votre clé publique.

Vous devez fournir un identificateur d'utilisateur en plus des paramètres de la clé. Cet identificateur est utilisé pour associer la clé créée avec une personne physique.

```
You need a User-ID to identify your key; the software constructs the user id
from Real Name, Comment and Email Address in this form:
    "Heinrich Heine (Der Dichter) <heinrichh@duesseldorf.de>"

Real name:
```

Un seul identificateur d'utilisateur est créé au moment de la création de la clé, mais il est possible d'en créer d'autres. Si vous voulez utiliser la clé dans deux ou plusieurs contextes, par exemple comme un employée au travail et comme un activiste politique à côté. Une identité doit être créée avec soin, car elle ne peut plus être éditée ensuite.

GnuPG a besoin d'un mot de passe étendu[2] pour protéger les parties privées de la clé primaire et les clés additionnelles que vous avez en votre possession.

```
You need a Passphrase to protect your private key.

Enter passphrase:
```

Il n'y a pas de limite sur la taille du mot de passe étendu, et il doit être soigneusement choisi. Dans une perspective de sécurité, le mot de passe utilisé pour déverrouiller la clé privée est l'un des points faibles de GnuPG (ainsi que des autres systèmes de chiffrement à clé publique) car il est la seule protection dont vous disposez si un individu obtient votre clé privée. De manière idéale, le mot de passe étendu ne devrait pas utiliser des mots du dictionnaire, et devrait mélanger la casse des caractères alphabétiques et aussi utiliser des caractères non alphabétiques. Un bon mot de passe est crucial pour une utilisation sûre de GnuPG.

Générer un certificat de révocation

Une fois que votre paire de clés a été créée, vous devez immédiatement créer un certificat de révocation pour la clé principale en utilisant l'option `--gen-revoke`. Si vous oubliez votre mot de passe, ou si votre clé privée est compromise ou perdue, ce certificat de révocation peut être publié pour notifier aux autres que votre clé publique ne doit plus être utilisée. On peut toujours se servir d'une clé publique révoquée pour vérifier des signatures que vous avez faites par le passé, mais on ne peut s'en servir pour chiffrer de nouveaux messages à votre attention. Cela n'affecte pas non plus votre capacité à déchiffrer les messages qui vous ont été adressés précédemment, si vous avez toujours accès à votre clé privée.

```
alice% gpg --output revoke.asc --gen-revoke mykey
[...]
```

L'argument **`mykey`** doit être un *identificateur de clé*, soit celui de la clé principale de votre paire de clés, soit une partie d'un identificateur d'utilisateur qui identifie votre clé. Le certificat produit sera enregistré dans le fichier *`revoke.asc`*. Si l'option `--output` est omis, le résultat de la commande sera écrit sur la sortie standard. Comme le certificat est court, vous pouvez souhaiter en imprimer une copie pour la stocker dans un endroit sûr, comme votre coffre à la banque. Le certificat ne doit pas être stocké dans un endroit où d'autres pourraient accéder, car n'importe qui peut publier le certificat de révocation, rendant inutilisable la clé publique correspondante.

Échanger des clés

Pour communiquer avec les autres, vous devez échanger vos clés publiques. Pour afficher la liste des clés de votre trousseau de clés publiques utilisez l'option de ligne de commandes `--list-keys`.

```
alice% gpg --list-keys
/users/alice/.gnupg/pubring.gpg
---------------------------------------
pub  1024D/BB7576AC 1999-06-04 Alice (Judge) <alice@cyb.org>
sub  1024g/78E9A8FA 1999-06-04
```

Exporter une clé publique

Pour envoyer votre clé publique à un correspondant, vous devez d'abord l'exporter en utilisant l'option de ligne de commandes `--export`. Elle prend un argument supplémentaire : l'identificateur de la clé à exporter. Comme avec l'option `--gen-revoke`, soit l'identificateur de la clé, soit une partie de l'identificateur utilisateur peut être utilisé pour identifier la clé à exporter.

```
alice% gpg --output alice.gpg --export alice@cyb.org
```

La clé est exportée dans un format binaire, mais ceci peut être problématique si la clé doit être envoyée par email ou publiée sur une page web. C'est pourquoi GnuPG supporte une option de ligne de commandes `--armor`[3] qui provoque la génération des sorties dans un format ASCII-armored (blindage ASCII) similaire aux documents encodés avec l'algorithme UUE. En général, toutes les sorties de GnuPG, comme par exemple clés, documents chiffrés, et signatures, peuvent être exportés dans le format ASCII-armored en ajoutant l'option `--armor`.

```
alice% gpg --armor --export alice@cyb.org
```

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v0.9.7 (GNU/Linux)
Comment: For info see http://www.gnupg.org

[...]
-----END PGP PUBLIC KEY BLOCK-----
```

Importer une clé publique

Une clé publique peut être ajoutée à votre trousseau de clés publiques avec l'option `--import`.

```
alice% gpg --import blake.gpg
gpg: key 9E98BC16: public key imported
gpg: Total number processed: 1
gpg:               imported: 1
alice% gpg --list-keys
/users/alice/.gnupg/pubring.gpg
---------------------------------------
pub  1024D/BB7576AC 1999-06-04 Alice (Judge) <alice@cyb.org>
sub  1024g/78E9A8FA 1999-06-04

pub  1024D/9E98BC16 1999-06-04 Blake (Executioner) <blake@cyb.org>
sub  1024g/5C8CBD41 1999-06-04
```

Une fois que la clé a été importée, elle devrait être validée. GnuPG utilise un modèle de confiance puissant et flexible qui ne requiert pas que vous validiez personnellement chaque clé que vous importez. Toutefois, certaines clés le nécessitent. Une clé peut être validée en vérifiant son empreinte. En la signant, vous certifiez que c'est une clé valide. L'empreinte d'une clé peut être visualisée rapidement avec l'option de ligne de commandes `--fingerprint`, mais pour certifier la clé, vous devez l'éditer.

```
alice% gpg --edit-key blake@cyb.org

pub  1024D/9E98BC16  created: 1999-06-04 expires: never      trust: -/q
sub  1024g/5C8CBD41  created: 1999-06-04 expires: never
(1)  Blake (Executioner) <blake@cyb.org>

Command> fpr
pub  1024D/9E98BC16 1999-06-04 Blake (Executioner) <blake@cyb.org>
             Fingerprint: 268F 448F CCD7 AF34 183E  52D8 9BDE 1A08 9E98 BC16
```

L'empreinte d'une clé est vérifiée avec le propriétaire de la clé. Ce peut être fait en personne, au téléphone, ou par tout autre moyen, du moment que vous pouvez garantir que vous communiquez bien avec le vrai propriétaire de la clé. Si l'empreinte que vous obtenez est la même que celle que le propriétaire de la clé obtient, alors vous pouvez être sûr que vous avez une copie correcte de la clé.

Après avoir vérifié l'empreinte, vous pouvez signer la clé pour la valider. Étant donné que la vérification des clés est un point faible de la cryptographie à clé publique, vous devez être extrêmement prudent et toujours vérifier l'empreinte d'une clé avant de la signer.

```
Command> sign

pub  1024D/9E98BC16  created: 1999-06-04 expires: never      trust: -/q
             Fingerprint: 268F 448F CCD7 AF34 183E  52D8 9BDE 1A08 9E98 BC16

     Blake (Executioner) <blake@cyb.org>
```

```
Are you really sure that you want to sign this key
with your key: "Alice (Judge) <alice@cyb.org>"

Really sign?
```

Une fois signée, vous pouvez vérifier la clé pour lister ses signatures et voir celle que vous avez ajoutée. Chaque identificateur d'utilisateur de la clé aura une ou plusieurs auto-signatures et aussi une signature pour chaque utilisateur qui a validé cette clé.

```
Command> check
uid  Blake (Executioner) <blake@cyb.org>
sig!       9E98BC16 1999-06-04   [self-signature]
sig!       BB7576AC 1999-06-04   Alice (Judge) <alice@cyb.org>
```

Chiffrer et déchiffrer des documents.

La clé publique et la clé privée ont toutes deux un rôle spécifique quand vous chiffrez et déchiffrez des documents. Une clé publique peut être vue comme un coffre fort ouvert. Quand un correspondant chiffre un document en utilisant une clé publique, ce document est mis dans le coffre, celui-ci est fermé, et la serrure tournée de nombreuses fois. La clé privée correspondante est la combinaison qui peut rouvrir le coffre et lire le document. En d'autres termes, seule la personne qui détient la clé privée peut lire le document chiffré en utilisant la clé publique associée.

La procédure pour chiffrer et déchiffrer les documents est semblable à ce modèle. Si vous voulez chiffrer un message pour Alice, vous le faites en utilisant la clé publique d'Alice, qui le déchiffre avec sa clé privée. Si elle vous envoie un message, elle le chiffre en utilisant votre clé publique, et vous le déchiffrez avec votre clé privée.

Pour chiffrer un document, il faut utiliser l'option `--encrypt`. Vous devez avoir la clé publique de tous les destinataires. Le programme attend le nom du document à chiffrer en entrée ; s'il est omis, il lit l'entrée standard. Le résultat chiffré est placé sur la sortie standard ou comme spécifié en utilisant l'option `--output`. En plus du chiffrement, le document est compressé pour plus de sécurité.

```
alice% gpg --output doc.gpg --encrypt --recipient blake@cyb.org doc
```

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.

Pour déchiffrer un message, on utilise l'option `--decrypt`. Vous avez besoin de la clé privée pour laquelle le message a été chiffré. De la même manière que pour le chiffrement, le document à déchiffrer est l'entrée et le document déchiffré est la sortie.

```
blake% gpg --output doc --decrypt doc.gpg

You need a passphrase to unlock the secret key for
user: "Blake (Executioner) <blake@cyb.org>"
1024-bit ELG-E key, ID 5C8CBD41, created 1999-06-04 (main key ID 9E98BC16)
```

```
Enter passphrase:
```

Les documents peuvent aussi être chiffrés sans recourir à la cryptographie à clé publique. Au lieu de cela, vous utilisez un algorithme de chiffrement symétrique pour chiffrer le document. La clé utilisée pour l'algorithme de chiffrement symétrique est dérivée du mot de passe fourni au chiffrement du document, et pour une bonne sécurité, il ne doit pas être le même que celui que vous utilisez pour protéger votre clé publique. Le chiffrement symétrique est utile quand le mot de passe n'a pas besoin d'être communiqué aux autres. Un document peut être chiffré avec un algorithme de chiffrement symétrique en utilisant l'option `--symmetric`.

```
alice% gpg --output doc.gpg --symmetric doc
Enter passphrase:
```

Générer et vérifier des signatures

Une signature numérique certifie et date un document. Si le document est modifié d'une quelconque manière, une vérification de la signature échouera. Un signature numérique peut servir les mêmes besoins qu'une signature manuscrite, en outre elle ne peut pas être contrefaite. Les distributions du code source de GnuPG, par exemple sont signées de telle manière que les utilisateurs puissent vérifier qu'elles n'ont pas été modifiées depuis qu'elles ont été empaquetées.

La création et la vérification des signatures utilisent les paires de clés privée/publique pour une opération différente du chiffrement et du déchiffrement. Une signature est créée en utilisant la clé privée du signataire. La signature est vérifiée en utilisant la clé publique correspondante. Par exemple, Alice utilisera sa propre clé privée pour signer numériquement sa dernière contribution au Journal de la Chimie Non Organique. L'éditeur qui gérera sa soumission utilisera la clé publique d'Alice pour vérifier que l'article vient en effet d'Alice et qu'il n'a pas été modifié depuis qu'elle l'a envoyé. Une conséquence de l'utilisation des signatures numériques est qu'il est difficile d'infirmer que vous avez fait une signature numérique car cela impliquerait que votre clé privée a été compromise.

L'option de ligne de commandes `--sign` est utilisée pour générer une signature numérique. Le document à signer est l'entrée, et le document signé est la sortie.

```
alice% gpg --output doc.sig --sign doc

You need a passphrase to unlock the private key for
user: "Alice (Judge) <alice@cyb.org>"
1024-bit DSA key, ID BB7576AC, created 1999-06-04

Enter passphrase:
```

Le document est compressé avant d'être signé, et la sortie est au format binaire.

Pour un document signé donné, vous pouvez en vérifier la signature ou vérifier la signature et récupérer le document original. Pour vérifier la signature, utilisez l'option `--verify`. Pour vérifier la signature et extraire le document, utilisez l'option `--decrypt`. Le document à vérifier et à récupérer est l'entrée et le document récupéré est la sortie.

```
blake% gpg --output doc --decrypt doc.sig
gpg: Signature made Fri Jun  4 12:02:38 1999 CDT using DSA key ID BB7576AC
gpg: Good signature from "Alice (Judge) <alice@cyb.org>"
```

Les documents signés en clair

Un usage commun des signatures numériques est de signer les soumissions sur USENET, ou les messages email. Dans une telle situation, il est inopportun de compresser le document quand on le signe. L'option `--clearsign` entraîne que le document est suivi par une signature ASCII-armored, mais le document n'est pas modifié outre mesure.

```
alice% gpg --clearsign doc

You need a passphrase to unlock the secret key for
user: "Alice (Judge) <alice@cyb.org>"
1024-bit DSA key, ID BB7576AC, created 1999-06-04

-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

[...]
-----BEGIN PGP SIGNATURE-----
Version: GnuPG v0.9.7 (GNU/Linux)
Comment: For info see http://www.gnupg.org

iEYEARECAAYFAjdYCQoACgkQJ9S6ULt1dqz6IwCfQ7wP6i/i8HhbcOSKF4ELyQB1
oCoAoOuqpRqEzr4kOkQqHRLE/b8/Rw2k
=y6kj
-----END PGP SIGNATURE-----
```

Signatures détachées

Un document signé a une utilité limitée. Les autres utilisateurs doivent récupérer le document original à partir de la version signée, et même avec un document signé en clair, le document signé doit être modifié pour retrouver l'original. C'est pourquoi un troisième modèle de signature de document, crée une signature détachée du reste du document. Dans ce cas la signature est écrite dans un fichier séparé. Une signature détachée est créée en utilisant l'option `--detach-sig`.

```
alice% gpg --output doc.sig --detach-sig doc

You need a passphrase to unlock the secret key for
user: "Alice (Judge) <alice@cyb.org>"
1024-bit DSA key, ID BB7576AC, created 1999-06-04

Enter passphrase:
```

Le document et la signature détachée sont tous deux nécessaires pour vérifier la signature du document, à l'aide de l'option `--verify`.

```
blake% gpg --verify doc.sig doc
gpg: Signature made Fri Jun  4 12:38:46 1999 CDT using DSA key ID BB7576AC
gpg: Good signature from "Alice (Judge) <alice@cyb.org>"
```

Notes

1. Le choix numéro 3 permet de générer une paire de clés ElGamal, qui n'est utilisable que pour le chiffrement

2. ndt : en anglais pass phrase
3. Les options de lignes de commandes fréquemment utilisées peuvent êtres spécifiées dans un fichier de configuration.

Chapitre 2. Concepts

GnuPG fait usage de nombreux concepts de cryptographie dont les *algorithmes de chiffrement symétriques*, *algorithmes de chiffrement à clé publique*, et *hachage à sens unique*. Il n'est pas nécessaire de comprendre complètement ces concepts pour utiliser les fonctionnalités de GnuPG.

Ce chapitre est une introduction aux concepts cryptographiques de base utilisés par GnuPG. D'autres livres couvrent ces sujets plus en détails. Un bon livre pour approfondir cette étude est "Cryptographie Appliquée" (http://www.counterpane.com/applied.html) de Bruce Schneier (http://www.counterpane.com/schneier.html).

Algorithmes de chiffrement symétriques.

Un algorithme de chiffrement symétrique utilise la même clé pour le chiffrement et le déchiffrement. Les deux partis qui veulent communiquer en utilisant un algorithme de chiffrement symétrique doivent au préalable se mettre d'accord sur la clé. Une fois qu'elles le sont, l'émetteur chiffre le message en utilisant la clé, l'envoie au destinataire, et ce dernier le déchiffre en utilisant la même clé. Par exemple, la machine allemande Enigma utilisait un algorithme de chiffrement symétrique, et les clés quotidiennes étaient distribuées dans des livres de code. Chaque jour, les opérateurs radio qui émettaient ou recevaient devaient consulter leur copie du livre de codes, pour trouver la clé du jour. Le trafic radio pour la journée était chiffré et déchiffré en utilisant la clé du jour. Des exemples modernes d'algorithmes de chiffrement symétrique sont 3DES, Blowfish, ou IDEA.

Un bon procédé de chiffrement fait reposer la sécurité seulement sur la clef, nullement sur l'algorithme. En d'autres termes, ce ne sera d'aucune aide à un éventuel cryptanaliste de connaître l'algorithme utilisé. La connaissance de l'algorithme est nécessaire seulement s'il a obtenu la clé. Les procédés de chiffrement utilisés dans GnuPG vérifient cette propriété.

Comme toute la sécurité repose sur la clé, il est très important qu'il soit très difficile de deviner la clé. Autrement dit, l'ensemble des clés possibles, c'est à dire *l'espace des clés* se doit d'être grand. Lorsqu'il était à Los Alamos, Richard Feynman était célèbre pour sa capacité à forcer les coffres forts. Pour encourager le mythe, il transportait presque toujours avec lui un ensemble d'outils incluant un vieux stéthoscope. En réalité, il utilisait une quantité d'astuces pour réduire le nombre de combinaisons, il devait donc en essayer un petit nombre et continuait jusqu'à ce qu'il trouve la bonne. En d'autres termes, il réduisait la taille de l'espace des clés.

Les anglais utilisaient des machines pour découvrir les clés pendant la seconde guerre mondiale. La machine allemande Enigma avait un espace de clés très grand, mais les anglais ont construit des machines à calculer spécialisées dénommées les Bombes, pour essayer mécaniquement les clés, jusqu'à ce que la clé du jour soit trouvée. Cela signifie que parfois ils trouvaient la clé du jour pendant qu'elle était utilisée, mais cela signifie aussi que certains jours ils ne la trouvaient jamais. Les Bombes n'étaient pas des ordinateurs à proprement parler, mais étaient des précurseurs de nos ordinateurs modernes.

Aujourd'hui, les ordinateurs peuvent deviner des clés très rapidement : c'est la raison pour laquelle la taille des clés est très importante dans les cryptosystèmes modernes. Le procédé DES utilise une clé de 56 bits, ce qui signifie qu'il y a 2^{56} clés possibles. 2^{56} cela fait 72,057,594,037,927,936 clés. Cela fait beaucoup de clés, mais pour un ordinateur de base, essayer l'ensemble des clés est une question de jours. Un ordinateur spécialisé peut le faire en quelques heures. D'un autre côté, les procédés de chiffrement qui ont été conçus plus récemment, comme 3DES, Blowfish ou IDEA utilisent tous une clé de 128 bits, ce qui signifie qu'il y a 2^{128} clés possibles. C'est beaucoup, beaucoup trop de clés, et même si tous les ordinateurs de la planète coopéraient, il faudrait encore plus de temps que l'âge de l'univers pour trouver la clé.

Procédés de chiffrement à clé publique.

Le principal problème avec les procédés de chiffrement symétriques n'est pas leur sécurité, mais l'échange des clés. Une fois que l'émetteur et le récepteur ont échangé les clés, elles peuvent être utilisées pour communiquer de manière sécurisée, mais quel canal de communication sûr peut être utilisé pour communiquer la clé elle-même ? En particulier, il serait probablement plus facile pour un attaquant de travailler à intercepter la clé que d'essayer toutes les clés de l'espace des clés. Un autre problème est le nombre de clés nécessaires. S'il y a *n* personnes qui doivent communiquer, alors *n(n-1)/2* clés sont nécessaires pour que chaque couple de personnes puisse communiquer de manière privée. C'est peut être possible pour un petit nombre de personnes, mais ça devient rapidement ingérable pour un grand groupe de personnes.

Les procédés de chiffrement à clé publique ont été inventés pour éviter entièrement ce problème d'échange des clés. Un procédé de chiffrement à clé publique utilise une paire de clés pour envoyer des messages. Les deux clés appartiennent à la personne qui reçoit le message. Une des clés est la *clé publique* et peut être donnée à n'importe qui. L'autre clé est la *clé privée* et elle est gardée secrète par son propriétaire. L'émetteur chiffre un message en utilisant la clé publique et, une fois chiffré, seule la clé privée peut être utilisée pour le déchiffrer.

Ce protocole résout le problème d'échange des clés inhérent au procédé de chiffrement symétrique. Il n'est pas nécessaire pour l'émetteur et le récepteur de se mettre d'accord sur une clé. Il suffit qu'à un moment précédant la communication secrète, l'émetteur obtienne une copie de la clé publique du destinataire. De plus, la clé publique d'une personne peut être utilisée par toute personne désirant communiquer avec elle. Donc seules *n* paires de clés sont nécessaires à *n* personnes pour qu'elles puissent communiquer secrètement entre elles.

Les procédés de chiffrement à clé publique sont basés sur des fonctions à trappes à sens unique. Une fonction à sens unique est une fonction qui est aisée à calculer, mais dont l'inverse est dur à calculer. Par exemple, il est facile de multiplier deux nombres premiers entre eux pour obtenir un produit, mais il est difficile de factoriser un produit en deux nombres premiers qui le composent. Une fonction à trappe à sens unique est similaire, sauf qu'elle comporte en plus une trappe. C'est-à-dire que si une certaine information est connue, il devient facile de calculer l'inverse. Par exemple, si vous avez un nombre composé de deux facteurs premiers, alors la connaissance de l'un des facteurs rend le calcul du second facile. Considérons un procédé de chiffrement à clé publique basé sur la factorisation en nombres premiers. La clé publique contient un nombre obtenu par le produit de deux nombres premiers très grands, et l'algorithme de chiffrement utilise ce nombre pour chiffrer le message. L'algorithme de déchiffrement du message nécessite de connaître les facteurs premiers, donc le déchiffrement est facile si vous avez la clé privée contenant un des facteurs, mais extrêmement difficile si vous ne l'avez pas.

Comme avec un bon procédé de chiffrement symétrique, avec un bon procédé de chiffrement à clé publique, la sécurité repose entièrement sur la clé. C'est la raison pour laquelle la taille de la clé est une mesure de la sécurité du système, mais on ne peut pas comparer la taille des clés d'un système de chiffrement symétrique et d'un système de chiffrement à clé publique comme une mesure de leur sécurité relative. Dans une attaque par la force brute sur un procédé symétrique, avec une taille de clé de 80 bits, l'attaquant doit énumérer jusqu'à 2^{80} clés pour trouver la bonne. Dans une attaque par la force brute sur un procédé à clé publique, avec une taille de clé de 512 bits, l'attaquant doit factoriser un nombre encodé sur 512 bits (jusqu'à 155 chiffres). La charge de travail de l'attaquant est fondamentalement différente suivant le procédé de chiffrement qu'il attaque. Alors que 128 bits suffisent pour un procédé de chiffrement symétrique, étant donné la technologie de factorisation actuelle, des clés publiques de 1024 bits sont recommandées pour la plupart des usages.

Procédés de chiffrement hybride

Les procédés de chiffrement à clé publique ne sont pas la panacée. Beaucoup de procédés de chiffrements symétriques sont plus résistants du point de vue de la sécurité et le chiffrement et le déchiffrement à clé publique sont plus coûteux que les opérations correspondantes dans un système symétrique. Les procédés de chiffrement à clé publique sont néanmoins un outil efficace pour distribuer les clés des procédés symétriques, et c'est pourquoi ils sont utilisés dans les procédés de chiffrement hybride.

Un procédé de chiffrement hybride utilise à la fois un procédé de chiffrement symétrique et à clé publique. Cela fonctionne en utilisant un procédé de chiffrement à clé publique pour partager une clé qui sera utilisée pour le procédé de chiffrement symétrique. Le vrai message envoyé est chiffré en utilisant la clé puis transmis au destinataire. Comme l'échange de clés symétriques est sécurisé, la clé symétrique utilisée est différente pour chaque message envoyé. C'est pour cela qu'elle est aussi parfois appelée clé de session.

PGP et GnuPG utilisent tous les deux des procédés de chiffrement hybride. La clé de session, chiffrée en utilisant le procédé de chiffrement à clé publique, et le message envoyé, chiffré en utilisant le procédé de chiffrement symétrique, sont automatiquement combinés en un paquet. Le destinataire utilise sa clé privée pour déchiffrer la clé de session et la clé de session est ensuite utilisée pour déchiffrer le message.

Un procédé de chiffrement hybride est aussi résistant que le plus faible des procédés de chiffrement mis en œuvre. Dans PGP et GnuPG, le procédé de chiffrement à clé publique est probablement le plus faible des deux. Heureusement, même si un attaquant arrive à déchiffrer une clé de session, elle pourra seulement être utilisée pour lire le message qui a été chiffré avec cette clé de session. L'attaquant devra recommencer et déchiffrer une autre clé de session pour lire un autre message.

Signatures numériques

Une fonction de hachage est une fonction plusieurs donne un qui transforme son entrée en une valeur incluse dans un ensemble fini. Typiquement cet ensemble est un champ de nombres naturels. Une fonction simple de hachage est $f(x) = 0$ pour tout entier x. Une fonction de hachage plus intéressante est $f(x) = x \ mod\ 37$, qui fait correspondre x avec le reste de la division de x par 37.

La signature numérique d'un document est le résultat de l'application d'une fonction de hachage sur ce document. Toutefois, pour être utile, la fonction de hachage doit vérifier deux propriétés importantes. Premièrement, il doit être difficile de trouver deux documents qui une fois hachés donnent la même valeur. Deuxièmement, pour une valeur résultant d'une fonction de hachage, il doit être difficile de retrouver le document qui a produit cette valeur.

Quelques procédés de chiffrement à clé publique[1] peuvent être utilisés pour signer un document. Le signataire chiffre le document avec sa clé *privée*. Toute personne désireuse de vérifier la signature et de voir le document utilise simplement la clé publique du signataire pour déchiffrer le document. Cet algorithme satisfait les deux propriétés nécessaires pour une bonne fonction de hachage, mais en pratique, cet algorithme est trop lent pour être utile.

Une alternative est d'utiliser des fonctions de hachage conçues pour satisfaire ces deux propriétés importantes. SHA et MD5 sont des exemples de tels algorithmes. En utilisant un tel algorithme, un document est signé en le hachant et le résultat est la signature. Une autre personne peut vérifier la signature en hachant elle aussi sa copie du document, et en comparant le résultat du hachage avec le résultat du hachage du document original. Si elles correspondent, il est presque certain que les documents sont identiques.

Le problème maintenant est comment utiliser une fonction de hachage pour faire des signatures numériques sans permettre à un attaquant de fausser la vérification de la signature. Si le document et la signature sont envoyés non chiffrés, un attaquant pourrait modifier le document et générer la signature correspondante sans que le destinataire n'en soit conscient. Si le document seulement est chiffré, l'attaquant peut falsifier la signature et entraîner un échec de la vérification de la signature. Une troisième solution est d'utiliser un processus de chiffrement hybride à clé publique pour chiffrer à la fois le document et la signature. Le signataire utilise sa clé privée, et n'importe qui peut utiliser sa clé publique pour vérifier la signature et le document. Cela semble correct, mais en fait ne l'est pas. Si cet algorithme sécurise vraiment le document, il le protège aussi contre les altérations et il n'y aurait plus de raison pour la signature. Plus important, ceci ne protège ni le document ni la signature d'une altération. Avec cet algorithme, seule la clé de session pour le procédé de chiffrement symétrique est chiffrée en utilisant la clé privée du signataire. N'importe qui peut utiliser la clé publique pour récupérer la clé de session. C'est la raison pour laquelle, il est facile pour un attaquant de récupérer la clé de session et de l'utiliser pour chiffrer d'autres documents et les signatures correspondantes, et de les envoyer au nom de l'émetteur.

Une solution consiste à utiliser un algorithme à clé publique pour chiffrer seulement la signature. La valeur retournée par la fonction de hachage est chiffrée en utilisant la clé privée du signataire, et n'importe qui peut vérifier la signature en utilisant sa clé publique. Le document signé peut être envoyé en clair si le document est public ou en utilisant d'autres algorithmes de chiffrement. Si le document est modifié, la vérification de la signature va échouer, mais c'est précisément ce que la vérification d'une signature est censée détecter. Le standard de signature numérique (DSA) est un algorithme de signature à clé publique qui fonctionne comme celui que l'on vient de décrire. DSA est l'algorithme de signature utilisé par défaut dans GnuPG.

Notes

1. Le procédé doit avoir la propriété que la clé publique ou la clé privée peuvent être utilisées par l'algorithme de chiffrement comme clé publique. RSA est un exemple d'un tel algorithme, alors qu'ElGamal ne l'est pas.

Chapitre 3. La gestion des clés

La falsification des clés est une faiblesse majeure de la sécurité des cryptosystèmes à clé publique. Un espion peut falsifier les trousseaux de clés d'un utilisateur, ou fabriquer une clé publique pour un autre utilisateur et la publier pour que d'autres la téléchargent et l'utilisent. Par exemple, supposons que Chloé veuille surveiller les messages qu'Alice envoie à Bob. Elle pourrait monter ce que l'on appelle une attaque de *l'homme au milieu*. Pour cette attaque, Chloé crée une nouvelle paire de clés. Elle remplace la copie d'Alice de la clé publique de Bob par la nouvelle clé publique. Ensuite elle intercepte les messages qu'Alice envoie à Bob. Pour chaque message intercepté, elle le déchiffre en utilisant la nouvelle clé privée, le chiffre en utilisant la clé publique de Bob. Tous les messages envoyés par Alice à Bob peuvent maintenant être lus par Chloé.

Une bonne gestion des clés est cruciale pour être non seulement sûr de vos trousseaux de clés, mais aussi de l'intégrité des trousseaux de clés des autres. Le coœ de la gestion des clés dans GnuPG est la notion de signature des clés. La signature des clés a deux buts principaux : elle vous permet de détecter la modification de votre trousseau de clés, et de certifier qu'une clé appartient vraiment à la personne dont le nom est inscrit dans l'identifiant de la clé. Les signatures de clés sont aussi utilisées dans un schéma connu sous le nom de *toile de confiance*, utilisé pour étendre le mécanisme de certification aux clés qui ne sont pas signées directement par vous, mais par des gens en qui vous avez confiance. Les utilisateurs responsables qui pratiquent une bonne gestion des clés peuvent déjouer les attaques utilisant la falsification des clés.

Gérer votre paire de clés

Une paire de clés possède une clé publique et une clé privée. Une clé publique consiste en la portion publique de la clé principale de signature, la portion publique des clés secondaires de chiffrement et de signature et les identifiants utilisés pour associer la clé publique à une personne réelle. Chaque partie de la clé possède des informations sur elle même. Pour une clé, ces informations incluent ses identifiant d'utilisateurs, sa date de création, sa date d'expiration, etc. Pour un identifiant, ces données incluent le nom de la personne réelle qu'elle identifie, un commentaire[1] optionnel et une adresse email. La structure d'une clé privée est similaire, excepté qu'elle ne contient que les portions privées des clés, et qu'il n'y a pas d'informations concernant les identifiants d'utilisateur.

L'option de ligne de commandes `--edit-key` peut être utilisée pour visualiser une paire de clés. Par exemple :

```
chloe% gpg --edit-key chloe@cyb.org
Secret key is available.

pub  1024D/26B6AAE1  created: 1999-06-15 expires: never      trust: -/u
sub  2048g/0CF8CB7A  created: 1999-06-15 expires: never
sub  1792G/08224617  created: 1999-06-15 expires: 2002-06-14
sub   960D/B1F423E7  created: 1999-06-15 expires: 2002-06-14
(1)  Chloé (Jester) <chloe@cyb.org>
(2)  Chloé (Plebian) <chloe@tel.net>
Command>
```

La clé publique est affichée avec un drapeau indiquant si la clé privée est disponible. Les informations sur chaque composant de la clé sont ensuite listées. La première colonne indique le type de la clé. Le mot clé `pub` identifie la clé publique principale de signature, et le mot clé `sub` identifie une clé publique secondaire. La seconde colonne indique la longueur de la clé en bits, son type et son identifiant. Le type est `D` pour une clé DSA, `g` pour une clé de chiffrement seul ElGamal, et `G` pour une clé ElGamal qui peut être utilisée pour le chiffrement et la signature. Les dates de création et d'expiration sont données aux colonnes trois et quatre. Les identifiants d'utilisateur sont listés après les clés.

Plus d'informations sur la clé peuvent être obtenues avec les commandes interactives. La commande **toggle** bascule entre la composante privée et la composante publique de la paire de clés si elles sont effectivement disponibles.

```
Command> toggle

sec  1024D/26B6AAE1  created: 1999-06-15 expires: never
sbb  2048g/0CF8CB7A  created: 1999-06-15 expires: never
sbb  1792G/08224617  created: 1999-06-15 expires: 2002-06-14
sbb   960D/B1F423E7  created: 1999-06-15 expires: 2002-06-14
(1)  Chloé (Jester) <chloe@cyb.org>
(2)  Chloé (Plebian) <chloe@tel.net>
```

Les informations fournies sont similaires à celles affichées pour la clé publique. Le mot clé `sec` identifie la clé privée principale de signature, et le mot clé `sbb` identifie les clés privées secondaires. Les identifiants d'utilisateur de la clé publique associée sont aussi listés à sa convenance.

Intégrité des clés

Quand vous distribuez votre clé publique, vous distribuez les composantes publiques de votre clé principale et des clés secondaires qui lui sont associées, ainsi que les identifiants d'utilisateurs. Distribuer ces données seules est cependant un risque pour la sécurité car il est possible pour un attaquant de falsifier la clé. La clé publique peut être modifiée en ajoutant ou en substituant des clés ou en changeant ou en ajoutant les identifiants d'utilisateur. En modifiant l'identifiant d'utilisateur, l'attaquant peut changer son email et ainsi rediriger les emails vers lui même. En changeant une des clés de chiffrement, l'attaquant peut être capable de déchiffrer les messages redirigés vers lui.

L'utilisation des signatures numériques permet de résoudre ce problème. Quand des données sont signées par une clé privée, la clé publique correspondante est liée aux données signées. En d'autres termes, seule la clé publique correspondante peut être utilisée pour vérifier la signature et vérifier que les données n'ont pas été modifiées. Une clé publique peut être protégée contre les tentatives de falsification en utilisant la partie privée de la clé principale correspondante pour signer les composantes publiques et l'identifiant utilisateur. Elles seront ainsi liées à la partie publique de la clé principale. Signer les composantes publiques de la clé avec la partie privée de la clé principale est appelé une *auto-signature*, et une clé publique qui a des identifiants utilisateur auto-signés ainsi liés à elle est appelée un *certificat*.

Par exemple, Chloé a deux identifiants utilisateur et trois sous-clés. Les signatures des identifiants utilisateurs peuvent être vérifiées avec la commande **check** exécutée depuis le menu d'édition des clés.

```
chloe% gpg --edit-key chloe
Secret key is available.

pub  1024D/26B6AAE1  created: 1999-06-15 expires: never      trust: -/u
sub  2048g/0CF8CB7A  created: 1999-06-15 expires: never
sub  1792G/08224617  created: 1999-06-15 expires: 2002-06-14
sub   960D/B1F423E7  created: 1999-06-15 expires: 2002-06-14
(1)  Chloé (Jester) <chloe@cyb.org>
(2)  Chloé (Plebian) <chloe@tel.net>
```

```
Command> check
uid  Chloé (Jester) <chloe@cyb.org>
sig!    26B6AAE1 1999-06-15  [self-signature]
uid  Chloé (Plebian) <chloe@tel.net>
sig!    26B6AAE1 1999-06-15  [self-signature]
```

Comme on aurait pu le deviner, la clé utilisée pour les signatures est la clé principale de signature qui porte l'identifiant `0x26B6AAE1`. Les auto-signatures des sous-clés sont présentes dans la clé publique, mais elle ne sont pas affichées par l'interface de GnuPG.

Ajouter et supprimer des composantes à une clé

Vous pouvez rajouter des sous-clés et des identifiants d'utilisateur à votre paire de clés une fois qu'elle a été créée. Un identifiant d'utilisateur est ajouté en utilisant la commande **adduid**. On vous demande de saisir un nom, une adresse email et un commentaire comme lorsque vous avez créé votre paire de clés initiale. Une sous-clé est ajoutée en utilisant la commande **addkey**. L'interface est similaire à celle utilisée quand vous avez créé votre paire de clés initiale. La sous-clé peut être une clé de signature DSA, une clé de chiffrement ElGamal ou une clé ElGamal utilisable pour le chiffrement et les signatures. Lorsqu'une sous-clé ou un identifiant d'utilisateur est généré, ils sont signée en utilisant la clé principale de signature. C'est pour cette raison que vous devez saisir votre mot de passe quand la sous-clé est générée.

Des identifiants d'utilisateur supplémentaires sont utiles quand vous avez besoin de multiples identités. Par exemple, vous pouvez avoir besoin d'une identité pour votre emploi et une pour votre engagement politique. Vos collègues de travail vous reconnaîtrons par l'identifiant d'utilisateur de votre emploi, les autres membres de votre groupe politique par l'identifiant d'utilisateur créé pour votre engagement politique. Étant donné que les deux groupes de personnes peuvent être totalement distincts, chaque groupe peut ne pas faire confiance à l'autre identité d'utilisateur. C'est la raison pour laquelle les deux ID utilisateur sont nécessaires.

Les sous-clés supplémentaires sont aussi nécessaires. Les identifiants d'utilisateur associés à votre clé publique principale sont validés par les utilisateurs avec qui vous communiquez, et changer la clé principale nécessite de refaire les certifications. Ce peut être difficile et prendre beaucoup de temps si vous communiquez avec de nombreuses personnes. D'un autre côté, il est recommandé de changer périodiquement les clés de chiffrement. Si la clé est cassée, toutes les données qui sont chiffrées avec elle sont vulnérables. Par contre, en changeant les clés, seules les données chiffrées avec celle qui est cassée seront révélées.

Les sous-clés et les identifiants d'utilisateur peuvent aussi êtres effacés. Pour effacer une sous-clé ou un identifiant d'utilisateur, vous devez d'abord le sélectionner en utilisant respectivement les commandes **key** ou **uid**. Ces commandes sont des sélecteurs. Par exemple, la commande **key *2*** sélectionne la deuxième sous-clé, et lancer **key *2*** une seconde fois la désélectionne. Si aucun argument n'est fourni, toutes les sous-clés ou tous les identifiants d'utilisateur sont désélectionnés. Une fois que les identifiants d'utilisateur sont sélectionnés, la commande **deluid** efface l'identifiant d'utilisateur de votre clé. De manière similaire, la commande **delkey** efface toutes les sous-clés sélectionnées de votre clé publique et de votre clé privée.

Pour la gestion locale de clé, effacer les composantes des clés est un bon moyen de débarrasser les clés publiques de ce qui n'est pas nécessaire. Effacer les identifiants d'utilisateur et les sous-clés de votre propre clé n'est toutefois pas très avisé car cela complique la distribution des clés. Par défaut, quand un utilisateur importe votre clé publique mise à jour, elle sera fusionnée avec son ancienne copie de votre clé publique si elle était présente dans son trousseau de clés. Les composants des deux clés sont combinés lors de la fusion, et les composants que vous aviez effacés sont restaurés. Pour mettre à jour correctement la clé, l'utilisateur doit d'abord effacer la vieille version de votre clé et ensuite importer la nouvelle version. Cela fait du travail supplémentaire pour les personnes avec qui vous communiquez. De plus, si vous envoyez votre clé à un serveur de clés, la fusion a lieu quoi qu'il arrive, et toute personne qui téléchargera la clé depuis le serveur ne verra pas la clé avec les composants effacés. Par conséquent, pour mettre à jour votre clé, il vaut mieux révoquer les composants plutôt que de les effacer.

Révoquer les composants d'une clé

Pour révoquer une sous-clé , elle doit d'abord être sélectionnée. Une fois sélectionnée, elle peut être révoquée avec la commande **revkey**. La clé est révoquée en lui ajoutant une auto-signature de révocation. Contrairement à l'option de ligne de commandes `--gen-revoke`, l'effet est immédiat.

```
Command> revkey
Do you really want to revoke this key? y

You need a passphrase to unlock the secret key for
user: "Chloé (Jester) <chloe@cyb.org>"
1024-bit DSA key, ID B87DBA93, created 1999-06-28


pub  1024D/B87DBA93  created: 1999-06-28 expires: never      trust: -/u
sub  2048g/B7934539  created: 1999-06-28 expires: never
sub  1792G/4E3160AD  created: 1999-06-29 expires: 2000-06-28
rev! subkey has been revoked: 1999-06-29
sub   960D/E1F56448  created: 1999-06-29 expires: 2000-06-28
(1)  Chloé (Jester) <chloe@cyb.org>
(2)  Chloé (Plebian) <chloe@tel.net>
```

Un identifiant d'utilisateur est révoqué de manière différente. Normalement, un identifiant d'utilisateur collectionne des signatures qui attestent que celui-ci décrit bien la personne qui possède réellement la clé associée. En théorie, un identifiant d'utilisateur décrit une personne pour toujours, car cette personne ne changera jamais. En pratique, certains éléments de l'identifiant d'utilisateur tels que son adresse email ou d'autres composants peuvent changer avec le temps, rendant l'identifiant utilisateur invalide.

La spécification OpenPGP

* *Première référence à OpenPGP*

ne supporte pas la révocation des identifiant d'utilisateur, mais un identifiant d'utilisateur peut effectivement être révoqué en révoquant l'auto-signature de celui-ci. Pour des raisons de sécurité décrites précédemment, les correspondants ne feront pas confiance à un identifiant d'utilisateur sans auto-signature valide.

Une signature est révoquée en utilisant la commande **revsig**. Comme vous pouvez avoir signé un nombre quelconque d'identifiants d'utilisateur, l'interface utilisateur vous demande de décider pour chaque signature si elle doit être révoquée ou non.

```
Command> revsig
You have signed these user IDs:
     Chloé (Jester) <chloe@cyb.org>
   signed by B87DBA93 at 1999-06-28
     Chloé (Plebian) <chloe@tel.net>
   signed by B87DBA93 at 1999-06-28
```

```
user ID: "Chloé (Jester) <chloe@cyb.org>"
signed with your key B87DBA93 at 1999-06-28
Create a revocation certificate for this signature? (y/N)n
user ID: "Chloé (Plebian) <chloe@tel.net>"
signed with your key B87DBA93 at 1999-06-28
Create a revocation certificate for this signature? (y/N)y
You are about to revoke these signatures:
     Chloé (Plebian) <chloe@tel.net>
   signed by B87DBA93 at 1999-06-28
Really create the revocation certificates? (y/N)y

You need a passphrase to unlock the secret key for
user: "Chloé (Jester) <chloe@cyb.org>"
1024-bit DSA key, ID B87DBA93, created 1999-06-28


pub  1024D/B87DBA93  created: 1999-06-28 expires: never      trust: -/u
sub  2048g/B7934539  created: 1999-06-28 expires: never
sub  1792G/4E3160AD  created: 1999-06-29 expires: 2000-06-28
rev! subkey has been revoked: 1999-06-29
sub   960D/E1F56448  created: 1999-06-29 expires: 2000-06-28
(1)  Chloé (Jester) <chloe@cyb.org>
(2)  Chloé (Plebian) <chloe@tel.net>
```

Un identifiant d'utilisateur révoqué est indiqué par la signature de révocation sur l'identifiant d'utilisateur quand ses signatures sont listées.

```
Command> check
uid  Chloé (Jester) <chloe@cyb.org>
sig!    B87DBA93 1999-06-28  [self-signature]
uid  Chloé (Plebian) <chloe@tel.net>
rev!    B87DBA93 1999-06-29  [revocation]
sig!    B87DBA93 1999-06-28  [self-signature]
```

Pour révoquer des sous-clés ou des auto-signatures d'un identifiant d'utilisateur, GnuPG ajoute des auto-signatures de révocation à la clé. Étant donné que des signatures sont ajoutées et que rien n'est effacé, une révocation sera toujours visible par autrui quand votre clé publique mise à jour est distribuée et fusionnée avec d'anciennes copies de cette clé. C'est pourquoi la révocation garantit que tout le monde possède une copie intègre de votre clé publique.

Mettre à jour la date d'expiration d'une clé

La date d'expiration d'une clé peut être mise à jour avec la commande **expire** depuis le menu d'édition des clés. Si aucune clé est sélectionnée, c'est la date d'expiration de la clé principale qui est mise à jour. Dans les autres cas, la date d'expiration de la sous-clé sélectionnée est mise à jour.

La date d'expiration d'une clé est associée avec sa self-signature. La date d'expiration est mise à jour en effaçant l'ancienne self-signature et en ajoutant une nouvelle self-signature. Étant donné que les correspondants n'auront pas effacé l'ancienne self-signature, ils verront une self-signature supplémentaire de la clé quand ils mettront à jour leur copie de la clé. La dernière self-signature fait référence, donc, vos correspondants pourront connaître de manière non ambiguë la date d'expiration de vos clé.

Valider les clés des autres dans votre trousseau de clés publique

Dans le chapitre 1 nous avons donné une procédure pour valider les clés publiques de vos correspondants : la clé publique d'un correspondant est validée en vérifiant personnellement l'empreinte de sa clé et en signant ensuite sa clé publique avec votre clé privée. En vérifiant personnellement l'empreinte vous pouvez être sûr que la clé lui appartient vraiment, et comme vous avez signé la clé, vous pouvez être sûr de détecter toute modification dans le futur. Malheureusement, cette procédure est ingrate quand vous devez valider un grand nombre de clés ou quand vous devez communiquer avec des gens que vous ne connaissez pas personnellement.

GnuPG tente de résoudre ce problème avec un mécanisme connu sous le nom de *toile de confiance*[2]. Dans le modèle de la toile de confiance, la responsabilité pour valider les clés publiques est déléguée aux personnes en qui vous avez confiance. Par exemple, supposons que

- Alice a signé la clé de Blake, et
- Blake a signé les clés de Chloé et de Dharma.

Si Alice fait suffisamment confiance à Blake pour valider les clés qu'il signe, alors Alice considérera que les clés de Chloé et Dharma sont valides sans les avoir personnellement vérifiées. Elle utilise simplement sa copie validée de la clé publique de Blake, pour vérifier que les signatures faites par Blake sur les clés de Chloé et Dharma sont bonnes. En général, si on suppose qu'Alice fait confiance à tout le monde pour valider les clés qu'ils signent, alors toute clé signée par une clé valide est aussi considérée comme valide. La racine est la clé d'Alice, qui est considérée comme valide de manière axiomatique.

Confiance dans le propriétaire d'une clé

En pratique, la notion de confiance est subjective. Par exemple, la clé de Blake est valide pour Alice car elle l'a signée, mais elle peut aussi ne pas faire confiance à Blake pour valider les clés qu'il signe. Dans ce cas, elle ne considérera pas les clés de Chloé est de Dharma comme valides en se basant seulement sur la signature de Blake. Le modèle de toile de confiance prend ceci en compte en associant à chaque clé publique de votre trousseau une indication sur la manière dont vous faites confiance au propriétaire de la clé. Il y a quatre niveaux de confiance.

unknown

On ne sait rien sur la façon dont le propriétaire signe les clés. Les clés de votre trousseau de clés publiques ont par défaut ce niveau de confiance.

none

On sait que le propriétaire ne vérifie pas consciencieusement les clés avant de les signer.

marginal

Le propriétaire comprend l'implication de la signature des clés et valide les clés avant de les signer.

full

Le propriétaire comprend complètement les implications de la signature des clés, et sa signature sur une clé aurait la même valeur que votre signature.

Le niveau de confiance sur une clé est une chose personnelle que vous attribuez. Cette information est considérée comme privée. Ce n'est pas empaqueté avec la clé lorsque vous l'exportez ; elle est même enregistrée séparément de vos trousseaux, dans une base de donnée distincte.

L'éditeur de clés de GnuPG peut être utilisé pour définir le niveau de confiance que vous avez dans le propriétaire d'une clé. La commande **trust** est utilisée pour ce faire. Dans cet exemple, Alice édite le niveau de confiance qu'elle a en Blake et ensuite elle met à jour la base de données de confiance pour recalculer quelles clés sont valides en se basant sur le nouveau niveau de confiance assigné à Blake.

```
alice% gpg --edit-key blake

pub  1024D/8B927C8A  created: 1999-07-02 expires: never      trust: q/f
sub  1024g/C19EA233  created: 1999-07-02 expires: never
(1)  Blake (Executioner) <blake@cyb.org>

Command> trust
pub  1024D/8B927C8A  created: 1999-07-02 expires: never      trust: q/f
sub  1024g/C19EA233  created: 1999-07-02 expires: never
(1)  Blake (Executioner) <blake@cyb.org>

Please decide how far you trust this user to correctly
verify other users' keys (by looking at passports,
checking fingerprints from different sources...)?

 1 = Don't know
 2 = I do NOT trust
 3 = I trust marginally
 4 = I trust fully
 s = please show me more information
 m = back to the main menu

Your decision? 3

pub  1024D/8B927C8A  created: 1999-07-02 expires: never      trust: m/f
sub  1024g/C19EA233  created: 1999-07-02 expires: never
(1)  Blake (Executioner) <blake@cyb.org>

Command> quit
[...]
```

Le niveau de confiance dans le propriétaire de la clé et la validité de la clé sont indiqués à droite quand le clé est affichée. La confiance dans le propriétaire est affichée en premier et la validité de la clé est affichée ensuite[3]. Les quatre niveaux utilisés pour spécifier la confiance et la validité sont : unknown (`q`), none (`n`), marginal (`m`), et full (`f`). Dans ce cas, la clé de Blake est complètement valide car Alice l'a signée elle-même. Initialement la confiance accordée à Blake pour signer la clé des autres était non déterminée, mais elle a décidé de lui faire marginalement confiance.

Utiliser la confiance pour valider les clés

La toile de confiance autorise l'élaboration d'algorithmes plus élaborés pour valider une clé. Précédemment, une clé était considérée comme valide si vous l'aviez signée personnellement. Un algorithme plus flexible peut maintenant être utilisé : une clé *K* est considérée comme valide si elle remplit deux conditions :

1. elle est signée par suffisamment de clés valides, c'est-à-dire si

   - vous l'avez signée personnellement
   - elle a été signée par une clé à laquelle vous accordez toute votre confiance
   - elle a été signée par trois clés auxquelles vous accordez une confiance marginale

2. le chemin des clés signées conduisant de *K* jusqu'à votre propre clé mesure moins de cinq étapes.

La longueur du chemin, le nombre de clés auxquelles vous accordez une confiance marginale, et le nombre de clés auxquelles vous accordez une confiance totale nécessaire peuvent être modifiés. Les valeurs données ci-dessus sont les valeurs par défaut utilisées par GnuPG.

Figure 3-1 montre un exemple de toile de confiance dont Alice est la racine. Le graphe illustre qui a signé les clés de qui. Le tableau montre quelles clés Alice considère comme valides en se basant sur le niveau de confiance qu'elle accorde aux autres membres de la toile.

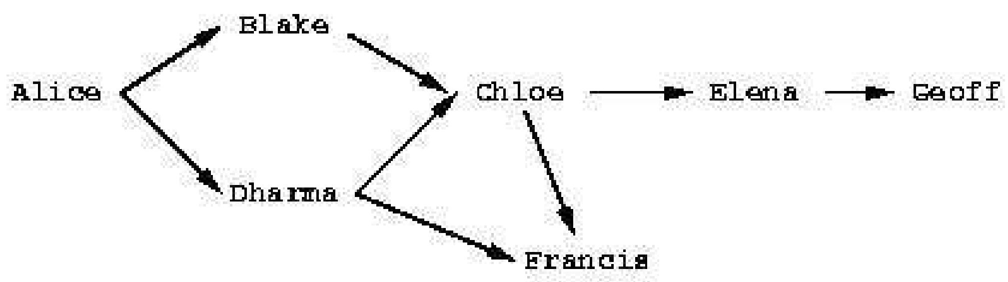
* *Bug potentiel: l'option de ligne de commande*`--completes-needed`*semble être ignorée quand on l'utilise conjointement avec l'option* `--update-trustdb`*. Les valeurs sont prises correctement en compte si elles sont placées dans le fichier de configuration.*

L'exemple considère que deux clés auxquelles on fait marginalement confiance et une en laquelle on a totalement confiance sont nécessaires pour valider une autre clé. La longueur maximum du chemin est fixée à trois.

Dans cet exemple, les clés de Blake et de Dharma sont toujours considérées comme valides, car elles sont signées directement avec la clé d'Alice. La validité des autres clés dépend de la confiance. Dans le premier cas, Alice a complètement confiance en Dharma, ce qui entraîne que les clés de Chloé et de Francis sont considérées comme valides. Dans le second exemple, Alice fait marginalement confiance à Blake et à Dharma. Étant donné que deux clés auxquelles on a marginalement confiance sont nécessaires pour valider une clé, celle de Chloé sera considérée comme pleinement valide, mais celle de Francis sera considérée comme marginalement valide. Dans le cas où on a marginalement confiance dans les clés de Chloé et de Dharma, celle de Chloé sera marginalement valide car celle de Dharma est pleinement valide. Toutefois, la clé de Francis sera considérée comme marginalement valide, car seule une clé valide peut être utilisée pour valider les autres clés, et la celle de Dharma est la seule clé pleinement valide utilisée pour signer la clé de Francis. Si de plus on ajoute une confiance marginale dans la clé de Blake, la clé de Chloé devient pleinement valide, et elle peut être utilisée pour valider pleinement la clé de Francis et valider marginalement la clé de Elena. Pour finir, même si on a pleinement confiance dans les clés de Blake, Chloé et Elena, çà ne suffit pas pour valider la clé de Geoff, car la longueur maximum du chemin pour valider une clé est de trois, alors que la longueur du chemin allant de la clé de Geoff jusqu'à celle d'Alice est de quatre.

Le modèle de toile de confiance est une approche flexible du problème de l'échange sécurisé de données avec des clés publiques. Il vous permet de personnaliser GnuPG pour vos besoins personnels. D'un côté, vous pouvez insister sur de multiples chemins courts allant de votre clé jusqu'à une autre clé *K* pour la valider. D'un autre côté, vous pouvez être satisfait par des chemins plus longs et même par un seul chemin allant de votre clé jusqu'à l'autre clé *K*. Demander de nombreux chemins courts est une forte garantie que la clé *K* appartient bien à celui à qui vous pensez qu'elle appartient. Ceci a un prix : il est plus difficile de valider les clés car vous devez signer plus de clés que si vous acceptiez de moins nombreux chemins, et de plus courts chemins.

Figure 3-1. Un exemple de toile de confiance



confiance		validité	
marginale	**complète**	**marginale**	**complète**
	Dharma		Blake, Chloé, Dharma, Francis
Blake, Dharma		Francis	Blake, Chloé, Dharma
Chloé, Dharma		Chloé, Francis	Blake, Dharma
Blake, Chloé, Dharma		Elena	Blake, Chloé, Dharma, Francis
	Blake, Chloé, Elena		Blake, Chloé, Elena, Francis

Distribution de clés

De manière idéale, vous devez distribuer vos clés en les donnant personnellement à vos correspondants. Par contre, en pratique, les clés sont souvent distribuées par email, ou par d'autre moyens électroniques de communication. La distribution par email est une bonne pratique quand vous avez seulement quelques correspondants, et même si vous avez de nombreux correspondants, vous pouvez utiliser d'autres moyens comme diffuser votre clé publique sur votre page Web. Ceci n'est pas acceptable si des personnes qui ont besoin de votre clé publique ne savent pas où la trouver sur le Web.

Pour résoudre ce problème, des serveurs de clés publiques sont utilisés pour collecter et distribuer les clés publiques. Une clé publique reçue par le serveur est soit ajoutée à la base de données du serveur soit fusionnée avec la clé existante si elle est déjà présente. Quand une requête de clé arrive au serveur, ce dernier consulte sa base de données et renvoie la clé publique s'il la trouve.

L'utilisation d'un serveur de clés est aussi intéressante quand de nombreuses personnes signent fréquemment les clés d'autres personnes. Sans l'utilisation d'un serveur de clés, quand Blake signe la clé d'Alice, il doit envoyer à Alice une copie de sa clé publique signée par lui pour qu'elle puisse ajouter la clé mise à jour à son trousseau, et la distribuer à tous ses correspondants. C'est la responsabilité de Alice et de Blake envers la communauté pour construire une toile de confiance resserrée et ainsi améliorer la sécurité de PGP. C'est néanmoins ennuyeux si la signature des clés est fréquente.

L'utilisation d'un serveur de clés rend le procédé plus facile. Quand Blake signe la clé d'Alice, il envoie la clé signée au serveur. Le serveur de clé ajoute la signature de Blake à sa copie de la clé publique d'Alice. Les personnes qui veulent mettre à jour leur copie de la clé de Alice consultent le serveur de clés quand ils le souhaitent pour récupérer la clé mise à jour. Alice n'est plus responsable de la distribution, et elle peut récupérer les signatures sur sa clé publique en interrogeant simplement le serveur de clés.

* `--keyserver` *doit apparaître avant* `--send-key` *ou* `--recv-key`*. Il s'agit d'un bug.*

Une ou plusieurs clés peuvent être envoyées à un serveur de clés en utilisant l'option de ligne de commandes `--send-keys`. Cette option prend un ou plusieurs sélecteurs de clés et envoie les clés spécifiées au serveur de clés. Le serveur de clés auquel les clés sont envoyées est spécifié avec l'option de ligne de commandes `--keyserver`. De manière similaire, l'option de ligne de commandes `--recv-keys` est utilisée pour récupérer les clés depuis un serveur de clés, mais cette option requiert l'utilisation d'un ID de clé pour spécifier la clé. Dans l'exemple suivant, Alice met à jour sa clé publique avec les nouvelles signatures depuis le serveur de clés *`certserver.pgp.com`* et envoie ensuite sa copie de la clé publique de Blake au même serveur de clés pour y ajouter toute nouvelle signature qu'elle y aurait ajoutée.

```
alice% gpg --keyserver certserver.pgp.com --recv-key 0xBB7576AC
gpg: requesting key BB7576AC from certserver.pgp.com ...
gpg: key BB7576AC: 1 new signature

gpg: Total number processed: 1
gpg:       new signatures: 1
alice% gpg --keyserver certserver.pgp.com --send-key blake@cyb.org
gpg: success sending to 'certserver.pgp.com' (status=200)
```

Il existe de nombreux serveurs de clés populaires en service à travers le monde. Les serveurs de clés les plus importants se synchronisent entre eux ; il est donc suffisant de sélectionner un serveur de clés proche de vous sur l'Internet et de l'utiliser régulièrement pour envoyer et recevoir des clés.

Notes

1. ndt : pourquoi pas votre surnom.
2. ndt : en anglais Web Of Trust
3. GnuPG utilise le mot “trust” pour signifier la confiance dans le propriétaire et la confiance dans la clé. Ceci peut être la source de confusion. Parfois la confiance dans le propriétaire est appelée *owner-trust* pour marquer la différence avec la confiance dans la clé. Dans ce manuel, “confiance” est utilisé pour signifier la confiance dans le propriétaire de la clé, et “validité” est utilisé pour signifier le fait que la clé appartient vraiment à la personne associée avec l'identifiant d'utilisateur de la clé.

Chapitre 4. Utilisation quotidienne de GnuPG

GnuPG est un outil complexe dont l'utilisation soulève des problèmes techniques, sociaux et légaux. Techniquement, il a été conçu pour être utilisé dans différentes situations nécessitant des besoins en sécurité complètement différents. Cela complique la gestion des clés. D'un point de vue social, utiliser GnuPG n'est pas une décision strictement personnelle. Pour utiliser effectivement GnuPG, les deux parties doivent l'utiliser. Enfin, en 1999, les lois régissant le chiffrement et en particulier si l'utilisation de logiciels tels que GnuPG est légale ou non, varient d'un pays à l'autre et sont actuellement débattues au sein de nombreux gouvernements nationaux.

Ce chapitre traite de ces problèmes. Il donne des conseils pratiques sur l'utilisation de GnuPG pour qu'il satisfasse vos besoins en matière de sécurité. Il explique aussi comment encourager vos correspondants à utiliser GnuPG pour communiquer de manière sécurisée. Finalement, le statut légal de GnuPG est décrit compte tenu des lois sur l'utilisation du chiffrement à travers le monde.

Définir vos besoins en matière de sécurité

GnuPG est un outil que vous utilisez pour protéger votre intimité. Votre intimité est protégée si vous pouvez correspondre avec les autres sans que des tiers puissent lire ces messages.

La façon dont vous devez utiliser GnuPG dépend de la détermination et des ressources de ceux qui peuvent vouloir lire les messages que vous chiffrez. La tierce personne peut être un administrateur système peu scrupuleux qui regarde négligemment vos emails, ou alors un espion industriel qui essaye de récupérer les secrets de votre compagnie, ou bien alors une agence gouvernementale qui essaye de vous poursuivre. L'utilisation de GnuPG pour vous protéger contre de l'espionnage occasionnel est différent de celle que vous en faites pour vous protéger contre un adversaire déterminé. Votre but ultime est de rendre plus coûteux le fait de récupérer les données chiffrées que ce qu'elles valent effectivement.

Adapter l'utilisation de GnuPG à vos besoins se résume à quatre problèmes :

- choisir la taille de votre paire de clés,
- protéger votre clé privée,
- définir les dates d'expiration et l'utilisation des clés secondaires,
- gérer votre toile de confiance.

Un taille de clé bien choisie vous protège contre une attaque par la force brute contre les messages chiffrés. Protéger votre clé privée empêche un attaquant d'utiliser simplement votre clé pour déchiffrer vos messages et en signer d'autres en votre nom. Gérer correctement votre toile de confiance empêche qu'un attaquant se fasse passer pour un de vos correspondants et s'interpose entre ce dernier et vous. Finalement, gérer ces différents problèmes en fonction de vos besoins en sécurité se résume à comment équilibrer la charge de travail supplémentaire requise pour utiliser GnuPG avec la protection que cela vous apporte.

Choisir la taille des clés

Le choix de la taille d'une clé dépend du type de clé. Dans OpenPGP, une paire de clés comporte souvent de multiples clés. Elle a au moins une clé principale pour les signatures, et probablement une ou plusieurs sous-clés additionnelles pour le chiffrement. En utilisant les paramètres par défaut de GnuPG, la clé principale sera une clé DSA, et les clés secondaires seront des clés ElGamal.

DSA permet une taille de clé allant jusqu'à 1024 bits. Ce n'est pas particulièrement bon étant donné les techniques de factorisation actuelles, mais c'est ce qui est spécifié par le standard. Vous devez choisir une clé DSA de 1024 bits, sans vous poser de question.

A l'opposé, les clés ElGamal peuvent être de n'importe quelle taille. Étant donné que GnuPG utilise un système hybride de chiffrement à clé publique, la clé publique est utilisée pour chiffrer la clé de session de 128 bits, et la clé privée est utilisée pour la déchiffrer. Néanmoins la taille de la clé a des répercussions sur la vitesse de chiffrement et de déchiffrement car le coût de ces algorithmes croit de manière exponentielle avec la taille de la clé. Les clés plus grosses prennent aussi plus de temps à générer et nécessitent plus d'espace pour le stockage. En fin de compte, cela diminue l'apport de sécurité d'une clé plus longue. Pour finir, si la clé est suffisamment grande pour résister à une attaque par la force brute, un espion aura sûrement recours à d'autres méthodes pour obtenir le texte déchiffré. Par exemple, il peut vous cambrioler chez vous ou au bureau ou vous agresser. Pour cette raison, 1024 bits est la taille de clé recommandée. Si vous avez vraiment besoin d'une clé plus grande, alors vous connaissez sûrement déjà tout ça, et vous devriez plutôt consulter un expert en sécurité informatique.

Protéger votre clé privée

La protection de votre clé privée est crucial pour bien utiliser GnuPG. Si quelqu'un obtient votre clé privée, alors tout ce qui a été chiffré à l'intention de cette clé pourra être déchiffré, et il pourra faire des signatures en votre nom. Si vous perdez votre clé privée, alors vous ne pourrez plus déchiffrer les documents chiffrés qui vous ont été envoyés ou qui vous seront envoyés, et vous ne pourrez plus signer de documents. Le fait de ne plus être le seul possesseur de votre clé privée est catastrophique.

Quelle que soit la façon dont vous utilisez GnuPG, vous devez sauvegarder le certificat de révocation de votre clé publique et une copie de sauvegarde de votre clé publique sur un support protégé en écriture stocké dans un lieu sûr. Par exemple, vous pouvez les graver sur un CD-ROM et les stocker dans un coffre à la banque dans une enveloppe scellée. Vous pouvez aussi les enregistrer sur une disquette et les cacher dans votre maison. Quoi que vous fassiez, ils doivent être enregistrés sur un support qui doit être physiquement capable de les mémoriser aussi longtemps que vous souhaitez utiliser la clé, et vous devez par conséquent les stocker plus soigneusement que la clé que vous utilisez tous les jours.

Pour vous aider à sauvegarder votre clé, GnuPG ne l'écrit pas directement sur le disque. Elle est chiffrée en utilisant un procédé symétrique de chiffrement. C'est la raison pour laquelle vous avez besoin d'un mot de passe pour accéder à la clé. De cette façon, un attaquant doit franchir deux barrières pour accéder à votre clé privée : (1) il doit s'emparer de la clé, et (2) il doit la déchiffrer.

Enregistrer de manière sûre votre clé privée est important, mais ceci a un prix. Idéalement, vous devez garder la clé privée sur un disque amovible protégé en écriture, comme une disquette, et vous devez l'utiliser sur une machine mono-utilisateur déconnectée du réseau. C'est peut-être dur voire carrément impossible à faire pour vous. Par exemple, vous pouvez ne pas posséder votre propre ordinateur et vous devez utiliser un ordinateur au travail ou à l'école, ou cela peut signifier que vous devez déconnecter votre ordinateur du réseau chaque fois que vous voulez utiliser GnuPG.

Cela ne signifie pas que vous ne devez pas utiliser GnuPG. Cela signifie seulement que vous avez décidé que les données que vous protégez sont suffisamment importantes pour que vous les chiffriez, mais pas assez pour mettre en place des mesures pour rendre la première barrière plus résistante. C'est votre choix.

Un bon mot de passe est critique pour utiliser GnuPG. Un attaquant qui accède à votre clé privée doit outrepasser le chiffrement utilisé pour protéger cette clé. Au lieu d'utiliser la force brute pour trouver la clé, l'attaquant essayera sûrement de deviner le mot de passe.

Il est plus facile de deviner le mot de passe faible, que de deviner une clé aléatoire de 128-bits. Si le mot de passe est un mot, il est bien moins coûteux d'essayer tous les mots des dictionnaires de tous les langages du monde. Même si les lettres du mot sont permutées, par exemple k3wldood, il est encore moins coûteux d'essayer les mots du dictionnaire avec un catalogue de permutations. Le problème est le même avec les citations. En général, les mots de passe basés sur le langage naturel sont des mots de passe faibles, car le langage naturel a beaucoup de redondance et peu d'entropie. Vous devez éviter le langage naturel si vous le pouvez.

Un mot de passe est bon si vous pouvez vous en rappeler et s'il est dur à deviner pour les autres. Il doit être composé de caractères issus de tout l'ensemble des caractères imprimables de votre clavier. Cela inclut les caractères alphabétiques, les nombres, et les caractères spéciaux comme } ou |. Soyez créatif et passez un peu de temps à considérer votre mot de passe, un bon choix est très important pour assurer votre protection.

Définition des dates d'expiration et utilisation des clés secondaires

Par défaut, la création d'une nouvelle paire de clés génère une clé DSA principale pour les signatures et une clé secondaire ElGamal pour le chiffrement. C'est convenable, car le rôle des deux clés est différent, et pour cette raison vous pouvez vouloir que les deux clés aient des durées de vie différentes. La clé principale est utilisée pour faire les signatures numériques, et elle accumule les signatures des autres qui confirment votre identité. La clé pour le chiffrement est seulement utilisée pour déchiffrer les documents chiffrés qui vous sont envoyés. En général, une signature numérique a une longue durée de vie, par exemple pour toujours, et vous pouvez aussi ne pas vouloir perdre les signatures sur votre clé que vous avez mis si longtemps à accumuler. D'un autre côté, la clé secondaire pour le chiffrement peut changer de manière périodique pour plus de sécurité, car si une clé de chiffrement est cassée, l'attaquant peut lire tous les documents qui ont été chiffrés à destination de cette clé dans le passé ou qui le seront dans le futur.

Dans presque tous les cas, vous ne voudrez pas que votre clé principale expire. Il y a deux raisons pour lesquelles vous pouvez choisir une date d'expiration. Premièrement, vous pourriez vouloir que la clé ait une durée de vie limitée. Par exemple, elle peut être utilisée pour un évènement particulier comme une campagne politique et elle ne sera d'aucune utilité quand la campagne sera finie. Une autre raison est que si vous perdez le contrôle de la clé et que vous n'avez pas de certificat de révocation, le fait d'avoir une date d'expiration sur la clé principale vous assure que la clé finira par ne plus pouvoir être utilisée.

Changer les clés secondaires de chiffrement coule de source, mais cela peut éventuellement poser un problème. Si vous générez une nouvelle paire de clés avec une date d'expiration pour la clé secondaire, cette clé secondaire finira par expirer. Peu avant son expiration, vous devez ajouter une nouvelle clé secondaire et publier votre clé publique que vous venez de mettre à jour. Une fois que la clé secondaire aura expiré, ceux qui désirent communiquer avec vous doivent récupérer la clé que vous venez de mettre à jour car ils ne pourront plus chiffrer des messages à votre attention avec la clé expirée. Cela peut poser problème suivant la façon dont vous distribuez votre clé. Heureusement, d'un autre côté, vous ne devez pas collecter de nouvelles signatures car la clé secondaire sera signée avec la clé principale, qui aura été précédemment validée par vos correspondants.

Cet inconvénient peut oui ou non valoir l'apport supplémentaire en sécurité que cela procure. Tout comme vous, un attaquant peut toujours lire tous les documents chiffrés à destination de la clé secondaire expirée. Changer la clé secondaire protège seulement les documents qui seront chiffrés ultérieurement. Pour lire les documents chiffrés avec la nouvelle clé secondaire, l'attaquant devra monter une nouvelle attaque telle que celle qu'il a utilisée la première fois.

Pour finir, il n'y a aucune raison à avoir plus d'une clé secondaire de chiffrement valide dans une clé. En avoir plusieurs n'apporte aucune sécurité supplémentaire. Il peut bien sûr y avoir un nombre quelconque de clés expirées dans une paire de clés donnée de manière à ce que les documents chiffrés dans le passé puissent toujours être déchiffrés, mais à un moment donné on a besoin que d'une seule clé secondaire active.

Gérer votre toile de confiance

La gestion de votre toile de confiance, tout comme la protection de votre clé privée, est un des aspects de l'utilisation de GnuPG. Il est nécessaire de trouver le juste équilibre entre la sécurité et la facilité d'utilisation. Si vous utilisez GnuPG pour vous protéger contre de l'espionnage ou une usurpation d'identité occasionnelle, vous pouvez accorder une confiance relative aux signatures faites par les autres. D'un autre côté, si vous avez à faire à quelqu'un de déterminé à envahir votre vie privée, vous devez moins faire confiance aux signatures des autres, et passer plus de temps à vérifier personnellement les signatures.

Quels que soient vos besoins en sécurité, vous devez *toujours être prudent* lorsque vous signez la clé de quelqu'un d'autre. Il est égoïste de signer une clé en étant juste assez sûr de la validité de la clé pour satisfaire vos propres besoins en matière de sécurité. D'autres, avec des besoins plus stricts, peuvent dépendre de votre signature. S'ils ne peuvent pas avoir confiance en vous, cela affaiblit la toile de confiance et la communication entre les utilisateurs de GnuPG sera plus difficile. Soyez aussi prudent lorsque vous signez des clés que vous souhaiteriez que les autres le soient si vous deviez dépendre de leurs signatures.

Dans la pratique, la gestion de votre toile de confiance se réduit à attribuer un degré de confiance aux autres et à régler les options `--marginals-needed` et `--completes-needed`. Toute clé que vous signez personnellement sera considérée comme valide, mais à part pour les petits groupes, il ne sera pas possible de signer personnellement les clés de toutes les personnes avec lesquelles vous devez communiquer. C'est la raison pour laquelle vous devez attribuer prudemment des niveaux de confiance.

Il est recommandé d'attribuer prudemment des niveaux de confiance et de faire attention lorsque vous réglez les options pour définir la façon dont GnuPG validera les clés. Prenons un exemple concret : vous pouvez faire pleinement confiance à un groupe d'amis très proches que vous savez être prudents lorsqu'ils valident les clés, et accorder une confiance limitée aux autres utilisateurs de votre trousseau. À partir de là, vous pouvez régler `--completes-needed` à `1` et `--marginals-needed` à `2`. Si vous êtes plus concerné par votre sécurité, vous pouvez régler ces valeurs respectivement à `1` et `3` ou à `2` et `3`. Si vous êtes moins sujet à des attaques concernant votre vie privée ou que vous souhaitez seulement être raisonnablement sûr de la validité des clés, réglez ces valeurs à `1` et `1`. En général, des nombres plus grands pour ces options signifient que plus de personnes seront nécessaires pour qu'une conspiration à votre égard puisse vous faire croire qu'une clé est valide alors qu'elle n'appartient pas à la personne que vous croyez.

Construisez votre réseau de confiance

Vouloir utiliser soi-même GnuPG ne suffit pas. Pour pouvoir communiquer de manière sécurisée avec d'autres personnes, vous devez avoir une toile de confiance. Toutefois, au premier abord c'est une tache décourageante. Les personnes avec qui vous communiquez doivent utiliser GnuPG[1], et il doit y avoir suffisamment de signatures pour considérer ces clés comme valides. Il ne s'agit pas de problèmes techniques, mais de problèmes sociaux. Quoiqu'il en soit, vous devez dépasser ces problèmes si vous voulez utiliser GnuPG.

Quand vous commencez à utiliser GnuPG, il est important de réaliser que vous n'avez pas besoin de communiquer de manière sécurisée avec tous vos correspondants. Commencez avec un petit nombre de personnes, peut-être juste vous et un ou deux de vos amis qui veulent utiliser leur droit à la protection de leur vie privée. Générez vos clés et signez mutuellement vos clés publiques. Ceci est votre toile de confiance initiale. En faisant ceci, vous apprécierez la valeur d'une toile de confiance, petite et robuste, et vous serez plus prudent quand vous agrandirez votre toile dans le futur.

En plus de votre toile de confiance initiale, vous pouvez souhaiter communiquer de manière sécurisée avec d'autres personnes qui utilisent GnuPG. Toutefois, ceci peut être gênant pour deux raisons : (1) on ne sait pas toujours quand quelqu'un utilise ou veut utiliser GnuPG et (2) si vous connaissez quelqu'un qui l'utilise, vous aurez encore des problèmes pour valider sa clé. La première raison à cela est que les gens ne font pas toujours de la publicité pour dire qu'ils utilisent GnuPG. Pour changer ce comportement, il faut montrer l'exemple et prévenir que vous utilisez GnuPG. Il y a au moins trois façons de le faire : vous pouvez signer les messages que vous envoyez aux autres ou que vous postez publiquement, vous pouvez diffuser votre clé publique sur votre page web ou si vous avez mis votre clé sur un serveur de clés, vous pouvez ajouter l'identifiant de votre clé dans votre signature d'email. Si vous promouvez votre clé, vous rendez la chose plus normale à accepter pour les autres. De plus, il sera plus facile pour les autres de commencer à communiquer de manière sécurisée avec vous car vous aurez pris l'initiative et rendu clair le fait que vous utilisez GnuPG.

Le problème de la validation des clés est plus difficile. Si vous ne connaissez pas personnellement la personne à qui appartient la clé que vous souhaitez signer, alors il n'est pas possible que vous signiez la clé vous-même. Vous devez vous reposer sur la signature des autres et espérer trouver une chaîne de signatures conduisant de la clé en question jusqu'à la vôtre. Pour avoir une chance de trouver une chaîne, vous devez prendre l'initiative et faire signer votre clé par d'autres personnes ne faisant pas partie de votre toile de confiance initiale. Pour accomplir ceci, participez à des "key signing parties". Si vous allez à une conférence, regardez à l'avance s'il y a une key signing party de prévue, et s'il n'y en a pas, proposez d'en organiser une (http://www.herrons.com/kb2nsx/keysign.html). Vous pouvez aussi être plus passif et avoir votre empreinte de clé avec vous pour des échanges de clés plus impromptus. Dans une telle situation, la personne à qui vous donnez l'empreinte la vérifiera et signera votre clé une fois qu'elle sera rentrée chez elle.

Gardez bien à l'esprit que tout ceci est optionnel. Vous n'êtes pas obligé de faire connaître votre clé ou de signer la clé des autres. La puissance de GnuPG réside dans le fait qu'il est suffisamment flexible pour s'adapter à vos besoins en sécurité quels qu'ils soient. Toutefois, en réalité, vous devrez prendre l'initiative si vous voulez agrandir votre toile de confiance et utiliser GnuPG pour effectuer une partie satisfaisante de votre communication.

Utiliser GnuPG légalement

Le statut légal du chiffrement de données varie d'un pays à l'autre, et les lois concernant le chiffrement évoluent rapidement. Bert-Japp Koops (http://rechten.kub.nl/koops/bertjaap.htm) tient à jour le Crypto Law Survey (http://rechten.uvt.nl/koops/cryptolaw/index.htm) auquel vous pouvez vous référer si vous souhaitez connaître le statut légal du chiffrement dans votre pays.

Notes

1. Dans cette partie, GnuPG fait référence à GnuPG en tant qu'implémentation de OpenPGP ou à toute autre telle le produit PGP de NAI.

Chapitre 5. Divers

Ce chapitre traite de divers sujets qui ne pouvaient pas être classés ailleurs dans le manuel. Au fur et à mesure que des sujets sont ajoutés, ils pourront être rassemblés dans des chapitres. Si vous souhaitez voir traité un sujet en particulier, suggérez-le. Mieux encore, portez-vous volontaire pour écrire un premier brouillon concernant ce sujet!

Écrire des interfaces utilisateur

Alma Whitten (http://www.cs.cmu.edu/~alma) et Doug Tygar (http://www.cs.berkeley.edu/~tygar) ont réalisé une étude (http://reports-archive.adm.cs.cmu.edu/anon/1998/abstracts/98-155.html) sur l'interface utilisateur du PGP 5.0 de NAI et sont arrivés à la conclusion que pour un utilisateur débutant, PGP peut paraître confus et frustrant. Dans leur étude sur des sujets humains, seuls quatre des douze sujets ont réussi à envoyer des mails chiffrés aux membres de leur équipe, et trois des douze ont envoyé le secret sans chiffrement. De plus, la moitié des sujets avaient des connaissances techniques.

Ces résultats ne sont pas surprenants. PGP 5.0 a une jolie interface utilisateur qui est excellente si vous êtes familier avec la façon dont fonctionne le chiffrement à clé publique et le modèle de gestion des clés de la toile de confiance de OpenPGP. Malheureusement, les utilisateurs débutants ne comprennent pas le chiffrement à clé publique et encore moins la gestion des clés, et l'interface utilisateur ne les aide pas beaucoup.

Si vous écrivez une interface utilisateur, vous devriez lire l'étude de Whitten et Tygar qui fournit des commentaires pour chacun des sujets du test, et ces détails sont très instructifs. Par exemple, il apparaît qu'une bonne partie des sujets croyaient qu'un message à envoyer devait être chiffré avec leur propre clé publique. En y réfléchissant, vous conviendrez que c'est une erreur facile à commettre. En général, les utilisateurs débutants ont des difficultés pour comprendre les différents rôles des clés publiques et privées dans GnuPG. En tant que concepteur d'interface graphique, vous devez essayer de rendre évident à tous moment quelle clé est utilisée. Vous pouvez aussi utiliser des assistants pour guider l'utilisateur lors des tâches ordinaires telle que la génération des clés, à l'intérieur desquelles des étapes annexes comme la génération du certificat de révocation et la réalisation d'une copie de sauvegarde qui sont essentielles pour utiliser GnuPG correctement. Le rapport comporte aussi les commentaires suivants :

- La sécurité est un objectif secondaire. Les utilisateurs veulent envoyer des mails, surfer, etc. Il ne faut pas s'imaginer que les utilisateurs seront motivés pour lire les manuels ou rechercher des contrôles de sécurité.
- La sécurité d'un ordinateur sur un réseau est aussi bonne que celle de son composant le plus faible. Les utilisateurs doivent être guidés pour considérer tous les aspects de leur sécurité, et surtout ne doivent pas êtres abandonnés à eux-mêmes pour procéder à des explorations aléatoires comme ils pourraient le faire avec un traitement de texte ou un tableur.
- Utilisez les mêmes termes pour décrire les mêmes choses. Ne pas utiliser alternativement des synonymes comme "chiffrer" et "encoder".
- Pour des utilisateurs inexpérimentés, simplifiez les affichages. Trop d'informations peuvent masquer l'information la plus importante. Dans la configuration initiale, un écran pourrait se contenter de donner à l'utilisateur une idée correcte de la relation entre les clés publiques et les clés privées et lui expliquer comment les obtenir et les distribuer.

Concevoir une interface utilisateur efficace pour la gestion des clés est encore plus difficile. La toile de confiance de OpenPGP est malheureusement plutôt obscure. Par exemple, la spécification impose trois niveaux de confiance pour un utilisateur : aucune, marginale et complète. La confiance effectivement accordée par l'utilisateur doit correspondre à un de ces trois niveaux. L'algorithme de validation des clés est difficile à comprendre pour les utilisateurs non informaticiens, en particulier les notions de "marginals needed" et de "completes needed". Étant donné que le modèle de toile de confiance est bien spécifié et qu'il ne peut être changé, vous allez devoir faire de votre mieux pour concevoir une interface utilisateur qui puisse le clarifier pour l'utilisateur. Une importante amélioration serait par exemple de générer un schéma relatant la façon dont la clé a été validée quand l'utilisateur le demande. Le rapport fait les commentaires suivants :

- Les utilisateurs vont probablement avoir du mal à cerner la façon et le moment où accorder des permissions.
- Accordez une grande importance au fait que les utilisateurs comprennent suffisamment leur sécurité pour les empêcher de commettre des erreurs qui pourraient leur coûter cher. De telles erreurs pourraient être : effacer accidentellement leur clé privée, publier ou révoquer accidentellement une clé, oublier leur mot de passe, ne pas sauvegarder leurs trousseaux.

Annexe A. GNU Free Documentation License

Version 1.1, March 2000

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.