[Note : Ce qui suit est la documentation originale pour PGP 2.6.2 du MIT, incluse ici dans une version non modifiée. Pour une explication sur la manière dont PGP 2.6.3i diffère de la version 2.6.2, voyez le fichier « readme.1st. »]

Phil's Pretty Good Software
Présente

PGP™

Pretty Good™ Privacy
Cryptographie à Clé Publique pour Tous

Guide de l'utilisateur de PGP™

Volume I : Questions Essentielles

par Philip Zimmermann
Révisé le 11 Octobre 94
(Traduction : news:fr.misc.cryptologie, 1998)

PGP Version 2.6.2 – 11 Oct 94
Logiciel par
Philip Zimmermann, et beaucoup d'autres.

Résumé : PGP™ utilise la cryptographie à clé publique pour protéger les e-mails et les fichiers de données. Communiquez en toute sécurité avec des gens que vous n'avez jamais rencontrés, sans avoir besoin de canaux sécurisés pour échanger les clés préalablement. PGP est rapide et possède de bonnes fonctionnalités, avec une gestion des clés sophistiquée, des signatures numériques, une compression des données, et une bonne ergonomie.

« Quoi que vous ferez, ce sera insignifiant, mais il est très important que vous le fassiez. » – Mahatma Gandhi ["Whatever you do will be insignificant, but it is very important that you do it." – Mahatma Gandhi]

Contenu

Rapide survol

Pretty Good™ Privacy (PGP) [ce qui signifie « Assez bonne confidentialité »], de Phil's Pretty Good Software, est un logiciel cryptographique de haute sécurité pour MSDOS, Unix, VAX/VMS, et d'autres ordinateurs. PGP permet aux gens d'échanger des fichiers avec confidentialité, authentification, et commodité. Confidentialité signifie que seuls ceux destinés à recevoir un message peuvent le lire. Authentification signifie que les messages qui semblent émaner d'une personne précise ne peuvent émaner que de cette personne. Commodité signifie que la confidentialité et l'authentification sont assurés sans les tracas de la gestion des clés accompagnant un logiciel de cryptographie conventionnelle. Aucun canal sécurisé n'est nécessaire pour échanger les clés entre les utilisateurs, ce qui rend PGP beaucoup plus facile à utiliser. Cela vient du fait que PGP est fondé sur une puissante nouvelle technologie appelée cryptographie « à clé publique ».

PGP combine la commodité du cryptosystème à clé publique Rivest-Shamir-Adleman (RSA) avec la vitesse de la cryptographie conventionnelle, les contractions de message pour les signatures numériques, la compression des données avant chiffrement, une bonne ergonomie, et une gestion des clés sophistiquée. Et PGP exécute les fonctions de clé publique plus vite que la plupart des autres logiciels. PGP c'est la cryptographie à clé publique pour tous.

PGP ne fournit aucune fonctionnalité permettant la communication par modem. Vous devez utiliser un logiciel séparé pour cela.

Ce document, « Volume I : Questions essentielles », explique seulement les notions essentielles pour utiliser PGP, et devrait être lu par tous les utilisateurs de PGP. « Volume II : Questions Spéciales » couvre les fonctionnalités avancées de PGP et d'autres questions spéciales, et peut être lu par les utilisateurs plus sérieux de PGP. Aucun des volumes n'explique les détails techniques des algorithmes cryptographiques et de la structure des données.

Pourquoi avez-vous besoin de PGP ?

C'est personnel. C'est privé. Et cela ne regarde personne d'autre que vous. Vous pouvez être en train de préparer une campagne électorale, discuter de vos impôts, ou avoir une affaire illicite. Ou vous pouvez être en train de faire quelque chose que vous pensez ne pas être illégal et qui l'est. Quoi qu'il en soit, vous ne voulez pas que votre courrier électronique (e-mail) ou vos documents confidentiels soient lus par quelqu'un d'autre. Il n'y a rien de mal dans la défense de votre intimité. L'intimité est le bien le plus précieux de la Constitution.

Peut-être pensez-vous que votre courrier électronique que vous recevez est assez légitime pour que le chiffrement ne se justifie pas. Si vous êtes vraiment un citoyen au dessus de tout soupçon, pourquoi n'envoyez-vous pas toujours votre correspondance papier sur des cartes postales ? Pourquoi ne vous soumettez-vous pas aux tests de consommation de drogue sur

simple demande ? Pourquoi exigez-vous un mandat de perquisition pour laisser la police fouiller votre maison ? Essayez-vous de cacher quelque chose ? Vous devez être un [élément] subversif ou un trafiquant de drogue si vous cachez votre courrier dans des enveloppes. Ou peut-être un paranoïaque aigu. Est-ce que les citoyens honnêtes ont un quelconque besoin de chiffrer leurs e-mails ?

Que se passerait-il si tout le monde estimait que les citoyens honnêtes devraient utiliser des cartes postales pour leur courrier ? Si une bonne âme s'avisait alors d'imposer le respect de son intimité en utilisant une enveloppe, cela attirerait la suspicion. Peut-être que les autorités ouvriraient son courrier pour voir ce que cette personne cache. Heureusement, nous ne vivons pas dans ce genre de société car chacun protège la plupart de son courrier avec des enveloppes. Aussi personne n'attire la suspicion en protégeant son intimité avec une enveloppe. La sécurité vient du nombre. De la même manière, ce serait excellent si tout le monde utilisait la cryptographie de manière systématique pour tous ses e-mails, qu'ils soient innocents ou non, de telle sorte que personne n'attirerait la suspicion en protégeant l'intimité de ses e-mails par la cryptographie. Pensez à le faire comme une forme de solidarité.

Aujourd'hui, si le Gouvernement désire violer l'intimité de citoyens ordinaires, il doit consentir une certaine dépense d'argent et de travail pour intercepter, ouvrir et lire les lettres, et écouter, et si possible transcrire, le contenu des conversations téléphoniques. Cette méthode, coûteuse en travail, n'est pas praticable sur une grande échelle. Cela est fait seulement dans les cas importants, quand cela en vaut la peine.

De plus en plus, nos messages sont transmis par le biais de canaux électroniques. L'e-mail est en train de remplacer progressivement le courrier traditionnel sur papier. Les messages e-mail sont justement beaucoup trop faciles à intercepter et à soumettre à une recherche systématique par mots clés. Ceci peut être fait facilement, systématiquement, automatiquement, et de manière indécelable sur une grande échelle. Les télex internationaux sont déjà interceptés de cette manière sur une grande échelle par la NSA américaine.

Nous nous dirigeons vers un futur où les pays seront sillonnés de réseaux de fibres optiques à haute capacité reliant ensemble la totalité de nos ordinateurs personnels sans cesse plus nombreux. L'e-mail sera la norme pour chacun, et non plus la nouveauté qu'il est aujourd'hui. Les Etats protégeront nos e-mails avec les protocoles de chiffrement conçus par les Etats. Il est probable que la plupart des gens accepteront cela. Mais peut-être que certains préféreront leurs propres mesures de protection.

En 1991 aux Etats-Unis, le projet de loi 266 du Sénat, un texte anti-criminalité, comportait une disposition troublante cachée à l'intérieur du texte. Si cette résolution était devenue une véritable loi, cela aurait contraint les fabricants d'équipements de communications sécurisées à insérer des « portes dérobées » spéciales dans leurs produits, de telle sorte que le gouvernement puisse lire les messages chiffrés par n'importe qui. Le texte disait : « La recommandation du Sénat est que les fournisseurs de services de communications électroniques et les fabricants d'équipements de communication électronique devront s'assurer que les systèmes de communication permettent au gouvernement d'obtenir le contenu en clair des communications vocales, des données, et des autres communications dans les cas prévus par la loi ». Cette mesure fut rejetée après une protestation massive des militants des droits civiques et des groupes industriels.

En 1992, la « Digital Telephony wiretap proposal » du FBI fut introduite au Congrès américain. Ce projet de loi prévoyait que tout fabricant de matériel de communication devait installer dans ses équipements des portes d'entrée spéciales permettant au FBI de mettre sur table d'écoute, à distance, n'importe quelle forme de communication électronique. Bien

qu'elle n'ait jamais attiré de sympathisants au Congrès en 1992 à cause de l'opposition des citoyens, elle a été réintroduite en 1994.

La plus alarmante de toutes est l'initiative incluse dans la politique de la Maison Blanche, développée à la NSA après le départ de l'Administration Bush, et révélée le 16 avril 1993. La pièce centrale de ce dispositif est le microprocesseur construit par le gouvernement et appelé puce « Clipper », contenant un algorithme de la NSA classé top secret. Le gouvernement est en train d'encourager l'industrie privée à l'insérer dans leurs équipements de communications sécurisées, comme les téléphones sécurisés, les fax sécurisés, etc. AT&T insère dès à présent la « Clipper » dans ses équipements vocaux sécurisés. Ce que cela cache : au moment de la fabrication, chaque puce « Clipper » sera chargée avec sa propre clé, et le gouvernement en gardera une copie, placée entre les mains d'un tiers. Il n'y a pas à s'inquiéter, cependant : le gouvernement a promis qu'il utiliserait ces clés pour lire le trafic des citoyens uniquement dans les cas dûment autorisés par la loi. Bien sûr, pour rendre la « Clipper » complètement efficace, la prochaine étape devrait être de mettre hors-la-loi tout autre système.

Si l'intimité est mise hors la loi, seuls les hors-la-loi auront une intimité. Les agences de renseignement ont accès à une bonne technologie cryptographique. De même les trafiquants d'armes et de drogue. Egalement les entreprises sous contrat avec la Défense Nationale, les sociétés pétrolières et les autres grandes multinationales. Mais les gens ordinaires et les organisations politiques de base n'avaient pas accès à une technologie cryptographique de « qualité militaire » abordable. Jusqu'à présent.

PGP donne aux gens le pouvoir de prendre en main leur intimité. Il y a un besoin social croissant pour cela. C'est pourquoi j'ai écrit PGP.

Comment ça marche

La connaissance préalable des notions de cryptographie en général et de cryptographie à clé publique en particulier serait un atout. Néanmoins, voici quelques remarques d'introduction à la cryptographie à clé publique.

Tout d'abord, quelques rappels terminologiques élémentaires. Supposons que je veuille vous envoyer un message, mais de telle sorte que vous soyez le seul à pouvoir le lire. Je peux « crypter » ou « chiffrer » le message, ce qui veut dire que je le brouille d'une manière extrêmement compliquée, le rendant illisible pour quiconque d'autre que vous, destinataire convenu du message. J'utilise une « clé » cryptographique pour chiffrer le message, et vous aurez à utiliser la même clé pour le déchiffrer ou le « décrypter ». C'est ainsi que les choses se passent dans les cryptosystèmes conventionnels à « clé unique ».

Dans les cryptosystèmes conventionnels, tels que le Data Encryption Standard (DES), le standard fédéral américain, une clé unique est utilisée aussi bien pour le chiffrement que pour le déchiffrement. Cela veut dire que la clé doit être préalablement transmise par des canaux sûrs de sorte que les parties puissent la connaître avant que les messages chiffrés puissent être envoyés par des canaux peu sûrs. Cela peut ne pas être pratique. Et si vous disposez déjà d'un canal sûr pour l'échange des clés, pourquoi auriez-vous besoin de recourir à la cryptographie pour le reste ?

Dans les cryptosystèmes à clé publique, chacun possède deux clés complémentaires, une clé publique, connue de tous, et une clé secrète (souvent aussi appelée clé privée). Chaque clé décode ce qui a été codé par l'autre. La connaissance de la clé publique ne permet pas d'en déduire la clé secrète. La clé publique peut être publiée et largement diffusée à travers un réseau de communication. Ce protocole assure la confidentialité sans avoir besoin de recourir aux canaux sécurisés exigés par les cryptosystèmes conventionnels.

N'importe qui peut utiliser la clé publique d'un destinataire pour chiffrer un message à son intention, et ce destinataire utilise sa propre clé secrète pour le déchiffrer. Personne d'autre que le destinataire ne peut le déchiffrer, parce que personne d'autre que lui n'a accès à cette clé secrète. Même la personne qui a chiffré le message ne peut pas le déchiffrer.

L'authentification des messages est aussi possible. La clé secrète de l'expéditeur peut être utilisée pour chiffrer un message, en le « signant » de cette façon. Cela crée une signature numérique d'un message, que le destinataire (ou n'importe qui d'autre) peut vérifier en utilisant la clé publique de l'expéditeur pour la déchiffrer. Cela prouve que l'expéditeur était le véritable auteur du message, et que le message n'a pas été altéré par la suite par quelqu'un d'autre, puisque seul l'expéditeur possède la clé secrète qui a fait cette signature. La contrefaçon d'un message signé est infaisable, et l'expéditeur ne peut plus par la suite désavouer sa signature.

Ces deux possibilités peuvent être combinées de manière à permettre à la fois confidentialité et authentification en signant d'abord le message avec votre propre clé secrète, et ensuite en chiffrant le message signé avec la clé publique du destinataire. Le destinataire refait ces étapes à l'envers en déchiffrant d'abord le message avec sa propre clé secrète, puis en vérifiant la signature qui y est contenue avec votre clé publique. Ces opérations sont réalisées automatiquement par le logiciel du destinataire.

En raison de la lenteur de l'algorithme de chiffrement à clé publique par rapport au chiffrement conventionnel à clé unique, le chiffrement est plus efficacement effectué en utilisant un algorithme de chiffrement conventionnel à clé unique rapide et de grande qualité pour chiffrer le message. Le message originel non chiffré est appelé « texte clair » [ou plus loin « fichier clair »]. Dans un processus invisible pour l'utilisateur, une clé temporaire aléatoire, créée uniquement pour cette seule « session », est utilisée pour chiffrer conventionnellement le fichier clair. Ensuite la clé publique du destinataire est utilisée pour chiffrer cette clé temporaire aléatoire conventionnelle. Cette clé de « session » conventionnelle chiffrée avec la clé publique est envoyée avec le texte chiffré [ou plus loin « fichier chiffré »] au destinataire. Le destinataire utilise sa propre clé secrète pour récupérer cette clé de session temporaire, puis utilise cette clé pour exécuter l'algorithme conventionnel rapide à clé unique qui déchiffrera le message contenant le texte chiffré.

Les clés publiques sont conservées dans des « copies de clés » individuelles qui contiennent l'ID utilisateur du propriétaire de la clé (qui est le nom de cette personne), la date de la création de la paire de clés, et la clé publique proprement dite. Les copies de clés publiques contiennent la clé publique elle-même, tandis que les copies de clés secrètes contiennent la clé secrète elle-même. Chaque clé secrète est aussi chiffrée avec son propre mot de passe, pour le cas où on la perdrait. Un fichier de clés, ou « trousseau de clés », contient une ou plusieurs de ces copies de clés. Le trousseau de clés publiques contient les copies de clés publiques, et le trousseau de clés secrètes contient les copies de clés secrètes.

Les clés sont également référencées en interne par une « clé ID », qui est une « abréviation » de la clé publique (les 64 bits les moins significatifs de la clé publique). Lorsque cette clé ID est affichée, seuls les derniers 32 bits apparaissent, pour faire plus court. Alors que plusieurs

clés peuvent partager la même ID utilisateur, il n'existe pas, en pratique, deux clés qui partagent la même clé ID.

PGP utilise des « contractions de message » pour produire des signatures. Une contraction de message est une condensation sur 128 bits, cryptographiquement solide, du message, produite par l'application d'une fonction de hachage à sens unique. C'est quelque chose d'analogue à une « somme de contrôle » ou CRC code de contrôle d'erreur, en ce qu'elle « représente » le message sous forme ramassée et est utilisée pour détecter des modifications du message. A la différence d'un CRC, cependant, il est mathématiquement impraticable pour un attaquant d'élaborer un message de substitution qui produirait une contraction de message identique. La contraction de message est chiffrée par la clé secrète pour former une signature.

Les documents sont signés en les faisant précéder de certificats de signature, qui contiennent la clé ID de la clé utilisée pour signer, une contraction signée du document, et la date à laquelle la signature a été faite. La clé ID est utilisée par le destinataire pour retrouver la clé publique de l'expéditeur pour vérifier la signature. Le logiciel du destinataire retrouve automatiquement la clé publique de l'expéditeur et l'ID utilisateur de l'expéditeur dans son trousseau de clés publiques.

Les fichiers chiffrés sont précédés de la clé ID de la clé publique utilisée pour les chiffrer. Le destinataire utilise cette clé ID pour retrouver la clé secrète requise pour déchiffrer le message. Le logiciel du destinataire retrouve automatiquement la clé secrète de déchiffrement dans son trousseau de clés secrètes.

Ces deux types de trousseaux sont le principal moyen de stockage et de gestion des clés publiques et secrètes. Plutôt que de les conserver dans des fichiers de clé séparés, elles sont regroupées en trousseaux pour faciliter la recherche automatique des clés, que ce soit par le biais de la clé ID ou par celui de l'ID utilisateur. Chaque utilisateur conserve sa propre paire de trousseaux. Une copie de la clé publique personnelle peut être provisoirement conservée dans un fichier séparé, le temps de l'envoyer à votre ami qui pourra alors l'ajouter à son trousseau.

Installation de PGP

L'archive de la version MSDOS de PGP est livrée dans un fichier compressé avec un nom de fichier de cette forme : PGPxx.ZIP (chaque version a un nombre différent pour le « xx » dans le nom). Par exemple, l'archive de la version 2.6 est appelée PGP26.ZIP. L'archive peut être décompressée avec l'utilitaire shareware de décompression PKUNZIP pour MSDOS [ou WINZIP pour Windows], ou l'utilitaire Unix « unzip ». Quand l'archive de PGP est décompressée, plusieurs fichiers en émergent. Un de ces fichiers, appelé README.DOC, devrait toujours être lu avant d'installer PGP. Ce fichier contient les dernières informations sur ce qu'il y a de nouveau dans cette version de PGP, aussi bien que ce qu'il y a dans les autres fichiers inclus dans la livraison.

Si vous avez déjà une version plus ancienne de PGP, vous devriez la renommer ou l'effacer, pour éviter les conflits de nom avec le nouveau PGP.

Pour les détails complets sur la manière d'installer PGP, voyez le fichier séparé Guide d'installation de PGP, dans le fichier SETUP.DOC inclus avec cette livraison. Il décrit entièrement la manière de préparer le répertoire PGP et votre fichier AUTOEXEC.BAT et comment utiliser PKUNZIP pour l'installer. Nous résumerons brièvement ici simplement les instructions d'installation, au cas où vous seriez trop impatient pour lire tout de suite le manuel d'installation plus détaillé.

Pour installer PGP sur votre système MSDOS, vous devez copier le fichier archive compressé PGPxx.ZIP dans un répertoire adéquat sur votre disque dur (comme C:\PGP), et le décompresser avec PKUNZIP. Pour de meilleurs résultats, vous devriez aussi modifier votre fichier AUTOEXEC.BAT, comme décrit dans ce manuel, mais vous pouvez faire cela plus tard, après avoir joué un peu avec PGP et lu un peu plus ce manuel. Si vous n'avez pas lancé PGP avant, la première étape après l'installation (et la lecture de ce manuel) est de créer une paire de clés pour vous-même en lançant la commande « pgp -kg » de génération de clé PGP. Lisez d'abord le chapitre « Génération de la clé RSA » de ce manuel.

L'installation sous Unix et VAX/VMS est généralement similaire à celle sous MSDOS, mais vous devez d'abord compiler le code source. Un « makefile » Unix est fourni à cette fin avec le code source.

Comment utiliser PGP

Voir un résumé de l'utilisation

Pour voir un rapide résumé des commandes de PGP, tapez simplement :

```
pgp -h
```

Chiffrer un message

Pour chiffrer un fichier clair avec la clé publique du destinataire, tapez :

```
pgp -e fichier son_ID_utilisateur
```

Cette commande produit un fichier chiffré appelé fichier.pgp. Un exemple explicite est :

```
pgp -e lettre.txt Alice
```

ou :

```
pgp -e lettre.txt "Alice S"
```

Le premier exemple recherche dans votre trousseau de clés publiques « pubring.pgp » toute clé contenant la chaîne [de caractères] « Alice » quelque part dans le champ ID utilisateur. Le deuxième exemple trouverait tout ID utilisateur contenant la chaîne « Alice S ». Vous ne pouvez pas utiliser d'espaces sur la ligne de commande, sauf si vous mettez toute la chaîne [en contenant] entre guillemets. La recherche n'est pas sensible à la casse. Si la clé publique correspondante est trouvée, elle est utilisée pour chiffrer le fichier clair « lettre.txt », produisant un fichier chiffré appelé « lettre.pgp ».

PGP essaye de compresser le fichier clair avant de le chiffrer, augmentant ainsi considérablement la résistance à la cryptanalyse. Le fichier chiffré sera donc bien plus petit que le fichier clair.

Si vous voulez envoyer ce message chiffré à travers des canaux e-mail, convertissez-le au format imprimable ASCII « radix-64 » en ajoutant l'option -a, comme expliqué plus loin.

Chiffrer un message pour plusieurs destinataires

Si vous voulez envoyer le même message à plus d'une personne, vous pouvez spécifier un chiffrement pour plusieurs destinataires, chacun d'eux pourra déchiffrer le même fichier chiffré. Pour spécifier plusieurs destinataires, ajoutez simplement plus d'ID utilisateurs sur la ligne de commande, comme ceci :

pgp -e lettre.txt Alice Bob Carol

Cela créera un fichier chiffré appelé lettre.pgp qui pourra être déchiffré par Alice, Bob ou Carol. N'importe quel nombre de destinataires peut être spécifié.

Signer un message

Pour signer un fichier clair avec votre clé secrète, tapez :

pgp -s fichier [-u votre_ID_utilisateur]

Notez que les [crochets] indiquent un champ optionnel, donc ne tapez pas de vrais crochets.

Cette commande produit un fichier signé appelé fichier.pgp. Un exemple explicite est :

pgp -s lettre.txt -u Bob

Cela recherche dans votre trousseau de clés secrètes « secring.pgp » toute clé secrète contenant la chaîne « Bob » quelque part dans le champ ID utilisateur. Votre nom est bien Bob, n'est-ce pas ? La recherche n'est pas sensible à la casse. Si la clé secrète correspondante est trouvée, elle est utilisée pour signer le fichier clair « lettre.txt », produisant un fichier signé appelé « lettre.pgp ».

Si vous omettez le champ ID utilisateur, la première clé secrète de votre trousseau de clés secrètes sera utilisée comme clé secrète par défaut pour votre signature.

PGP essaye de compresser le message après l'avoir signé. Le fichier signé sera donc bien plus petit que le fichier d'origine, ce qui est appréciable pour l'archivage. Cependant, cela rend le fichier illisible, même si le message originel était du texte ASCII ordinaire. Ce serait bien si vous pouviez créer un fichier signé qui demeurerait lisible. Ce serait particulièrement utile si vous vouliez envoyer un message signé comme e-mail.

Pour signer des messages e-mail, dont vous voulez qu'ils restent lisibles, il est probablement plus approprié d'utiliser la fonctionnalité CLEARSIG, expliquée plus loin. Cela permet d'appliquer une signature imprimable à la fin du texte, et désactive aussi la compression du texte. Cela signifie que le texte demeure lisible pour le destinataire, même si celui-ci n'utilise pas PGP pour vérifier la signature. Ceci est expliqué en détail dans le chapitre intitulé « CLEARSIG – Enable Signed Messages to be Encapsulated as Clear Text » dans le Volume II « Questions Spéciales ». Si vous ne pouvez pas attendre de lire ce chapitre du manuel, vous pouvez voir à quoi ressemble un message e-mail signé, avec cet exemple :

pgp -sta message.txt

Cela créera un message signé dans le fichier « message.asc », englobant le texte originel, toujours lisible, auquel a été ajoutée une signature en ASCII, prêt à être envoyé par e-mail. Cet exemple présume que vous utilisez les réglages normaux pour activer le paramètre CLEARSIG dans le fichier de configuration.

Signer et ensuite chiffrer

Pour signer un fichier avec votre clé secrète, et ensuite le chiffrer avec la clé publique du destinataire :

pgp -es fichier son_ID_utilisateur [-u votre_ID_utilisateur]

Notez que les [crochets] indiquent un champ optionnel, donc ne tapez pas de vrais crochets.

Cet exemple produit un fichier chiffré gigogne appelé fichier.pgp. Votre clé secrète pour signer est automatiquement retrouvée dans votre trousseau de clés secrètes via votre_ID_utilisateur. La clé publique [du destinataire] est automatiquement retrouvée dans votre trousseau de clés publiques via son_ID_utilisateur. Si vous omettez le champ ID utilisateur, il vous sera demandé.

Si vous omettez votre propre champ ID utilisateur, la première clé secrète de votre trousseau de clés secrètes sera utilisée comme clé secrète par défaut pour votre signature.

Notez que PGP essaye de compresser le fichier avant de le chiffrer.

Si vous voulez envoyer ce message chiffré à travers des canaux e-mail, convertissez-le au format imprimable ASCII « radix-64 » en ajoutant l'option -a, comme expliqué plus loin.

Plusieurs destinataires peuvent être spécifiés, en ajoutant plus d'ID utilisateurs sur la ligne de commande.

Utiliser seulement le chiffrement conventionnel

Parfois vous avez simplement besoin de chiffrer un message à l'ancienne manière, avec une clé unique conventionnelle. Cette approche est intéressante pour protéger des fichiers archivés qui seront conservés sans être envoyés à personne. Puisque c'est la même personne qui aura chiffré qui déchiffrera, il n'est pas nécessaire de recourir à la cryptographie à clé publique.

Pour chiffrer un fichier de manière conventionnelle seulement, tapez :

pgp -c fichier

Cet exemple chiffre le fichier appelé fichier, produisant un fichier chiffré appelé fichier.pgp, sans utiliser la cryptographie à clé publique, les trousseaux, les ID utilisateurs, ni toutes ces sortes de choses. Il vous est demandé une phrase de passe comme clé conventionnelle pour chiffrer le fichier. Cette phrase de passe n'a pas besoin d'être (et, vraiment, ne DEVRAIT PAS être) la même que celle que vous utilisez pour protéger votre propre clé secrète [afin de signer ou déchiffrer]. Notez que PGP essaye de compresser le fichier avant de le chiffrer.

PGP ne chiffrera pas deux fois le même fichier de la même façon, même si vous utilisez la même phrase de passe à chaque fois.

Déchiffrer et vérifier les signatures

Pour déchiffrer un fichier chiffré, ou pour vérifier l'intégrité d'un fichier signé :

pgp fichier_chiffré [-o fichier_en_clair]

Notez que les [crochets] indiquent un champ optionnel, donc ne tapez pas de vrais crochets.

Le nom du fichier chiffré est présumé posséder une extension « .pgp » par défaut. L'option de nom de fichier déchiffré spécifie l'endroit où mettre celui-ci. Si aucun nom n'est spécifié, le nom du fichier chiffré est utilisé, sans aucune extension. Si une signature se trouve dans le fichier chiffré, il est automatiquement déchiffré et son intégrité vérifiée. L'ID utilisateur du signataire est affichée.

Notez que le « déballage » du fichier chiffré est complètement automatique, peu importe que le fichier soit seulement signé, seulement chiffré, ou les deux. PGP utilise [l'indication de] la clé ID contenue dans le fichier chiffré pour retrouver automatiquement la clé secrète de déchiffrement correspondante dans votre trousseau de clés secrètes. Si une signature est trouvée à l'intérieur [du fichier chiffré], PGP utilise [l'indication de] la clé ID dans la signature pour retrouver automatiquement la clé publique correspondante dans votre trousseau de clés publiques afin de vérifier la signature. Si toutes les clés requises se trouvent déjà dans vos trousseaux, aucune intervention de l'utilisateur n'est demandée, sauf pour entrer la phrase de passe pour utiliser votre clé secrète si nécessaire. Si le fichier chiffré était chiffré de manière conventionnelle sans cryptographie à clé publique, PGP s'en aperçoit et vous demande d'entrer la phrase de passe pour le déchiffrer de manière conventionnelle.

Gestion des clés

Depuis l'époque de Jules César, la gestion des clés a toujours été la partie la plus difficile de la cryptographie. Une des principales caractéristiques distinguant PGP est sa gestion sophistiquée des clés.

Génération de la clé RSA

Pour générer votre propre et unique paire de clés publique/secrète d'une taille spécifique, tapez :

pgp -kg

PGP vous montre un menu des tailles de clés recommandées (commercial de bas niveau, commercial de haut niveau, ou niveau « militaire ») et vous demande quelle longueur vous voulez, jusqu'à bien plus d'un millier de bits [maximum : 2048 bits]. Plus la clé est grande, plus vous gagnez en sécurité, mais vous le payez d'une baisse de la vitesse [de traitement].

Il vous est aussi demandé une ID utilisateur. Une bonne idée est d'utiliser votre nom complet comme ID utilisateur, parce qu'il y a ainsi moins de risque pour les autres d'utiliser la mauvaise clé publique pour chiffrer des messages à votre attention. Les espaces et les signes de ponctuation sont autorisés dans l'ID utilisateur. Il serait souhaitable de mettre votre adresse e-mail entre <crochets> après votre nom, comme cela :

Jean Dupont <dupont@toto.fr>

Si vous n'avez pas d'adresse e-mail, utilisez votre numéro de téléphone ou une autre information personnelle qui pourrait aider à s'assurer que votre ID utilisateur est unique.

PGP demande aussi une « phrase de passe » pour protéger votre clé secrète au cas où elle tomberait en de mauvaises mains. Personne ne peut utiliser votre clé secrète sans cette phrase de passe. La phrase de passe est comme un mot de passe, excepté que ce peut être une phrase entière ou une locution avec plusieurs mots, des espaces, des signes de ponctuation, ou tout

autre chose que vous voulez y mettre. Ne perdez pas cette phrase de passe – il n'y aucun moyen de la retrouver si vous la perdez. Cette phrase de passe sera nécessaire chaque fois que vous utiliserez votre clé secrète. La phrase de passe tient compte de la casse, et ne devrait pas être trop courte ou facile à deviner. Elle n'est jamais affichée à l'écran. Ne la laissez pas par écrit quelque part où quelqu'un d'autre pourrait la voir, et ne la stockez pas sur votre ordinateur. Si vous ne voulez pas de phrase de passe (vous êtes fou !), appuyez simplement sur RETURN (ou ENTREE) lorsqu'il vous est demandé de taper la phrase de passe.

La paire de clés publique/secrète est dérivée de grands nombres véritablement aléatoires principalement en mesurant les intervalles entre vos frappes des touches clavier avec un minuteur rapide. Le logiciel vous demandera d'entrer du texte au hasard pour l'aider à accumuler des bits aléatoires pour les clés. A sa demande, vous devrez frapper des touches de manière raisonnablement aléatoire, et ce serait une bonne chose que les caractères eux-mêmes que vous tapez le soient aussi. Une part de l'aléa provient de l'imprévisibilité du contenu de ce que vous tapez. Aussi ne tapez surtout pas les mêmes séquences de caractères.

Notez que la génération de la clé RSA est un processus très lent. Cela peut prendre quelques secondes pour une petite clé sur un processeur rapide, ou plusieurs minutes pour une grande clé sur un vieux IBM PC/XT. PGP indiquera visuellement la progression pendant la génération des clés.

La paire de clés générée sera placée dans vos trousseaux de clés publiques et secrètes. Vous pouvez plus tard utiliser l'option de commande -kx pour extraire (copier) votre nouvelle clé publique depuis votre trousseau de clés publiques et la mettre dans un fichier de clé séparé pour la distribuer à vos amis. Le fichier contenant la clé publique [que vous aurez extraite] peut être envoyé à vos amis pour être intégré à leur trousseau de clés publiques. Naturellement, vous gardez votre clé secrète avec vous, et vous devrez l'inclure dans votre trousseau de clés secrètes. Chaque clé secrète d'un trousseau de clés est individuellement protégée par sa propre phrase de passe.

Ne donnez jamais votre clé secrète à quelqu'un. Pour la même raison, ne générez pas de paires de clés pour vos amis. Chacun devrait générer lui-même sa propre paire de clés. Gardez toujours le contrôle physique de votre clé secrète, et ne prenez pas le risque de la divulguer en l'entreposant sur un ordinateur distant et/ou partagé avec d'autres. Gardez-la sur votre propre ordinateur personnel.

Si PGP se plaint de ne pas trouver le guide de l'utilisateur de PGP sur votre ordinateur, et refuse de générer une paire de clés sans lui, ne paniquez pas. Lisez l'explication du paramètre NOMANUAL dans le chapitre « Réglage des paramètres de configuration » (« Setting Configuration Parameters ») dans le volume « Questions Spéciales » du Guide de l'utilisateur de PGP [Volume II].

Ajouter une clé à votre trousseau de clés

Parfois, vous voudrez ajouter à votre trousseau une clé que quelqu'un vous aura donnée, sous la forme d'un fichier de clé.

Pour ajouter le contenu d'un fichier de clé publique ou secrète à votre trousseau de clés publiques ou secrètes (notez que les [crochets] représentent un champ de commande optionnel) :

```
pgp -ka fichier_de_clé [trousseau_de_clés]
```

L'extension du fichier de clé par défaut est « .pgp ». Le nom du fichier trousseau de clés est par défaut « pubring.pgp » ou « secring.pgp », selon que le fichier [de clé] contient une clé

publique ou secrète. Vous pouvez spécifier un nom de trousseau de clés différent, l'extension demeurant « .pgp » par défaut.

Si la clé est déjà dans votre trousseau de clés, PGP ne l'ajoutera pas une nouvelle fois. Toutes les clés contenues dans le fichier de clés sont ajoutées au trousseau, sauf celles qui s'y trouvent déjà.

Plus loin dans le manuel, nous expliquerons la notion de certification des clés au moyen des signatures. Si la clé ajoutée comporte des signatures attachées qui la certifient, les signatures seront ajoutées avec la clé. Si la clé se trouve déjà dans votre trousseau, PGP ajoute seulement celles des nouvelles signatures certifiant cette clé que vous n'aviez pas encore dans votre trousseau.

PGP a été conçu à l'origine pour gérer de petits trousseaux de clés. Si vous voulez gérer des trousseaux vraiment gros, lisez le chapitre « Gérer de gros trousseaux de clés publiques » (« Handling Large Public Keyrings ») dans le volume « Questions Spéciales » du Guide de l'utilisateur de PGP [Volume II].

Effacer une clé ou un ID utilisateur de votre trousseau de clés

Pour effacer une clé ou un ID utilisateur de votre trousseau de clés :

pgp -kr ID_utilisateur [trousseau_de_clés]

Cela recherche l'ID utilisateur spécifié dans votre trousseau, et l'efface si est trouvée une clé y correspondant. Rappelez-vous qu'une partie d'un ID utilisateur suffit. Le nom optionnel du trousseau est censé être « pubring.pgp ». Il peut ne pas être précisé, ou bien vous pouvez spécifier « secring.pgp » si vous voulez effacer une clé secrète. Vous pouvez spécifier un nom de trousseau différent. L'extension par défaut du trousseau de clés est « .pgp ».

S'il existe plus d'un ID utilisateur pour cette clé, il vous sera demandé si vous voulez effacer seulement l'ID utilisateur spécifié, tout en laissant la clé et ses autres ID intacts.

Extraire (copier) une clé de votre trousseau de clés

Pour extraire (copier) une clé de votre trousseau de clés publiques ou secrètes :

pgp -kx ID_utilisateur fichier_de_clé [trousseau_de_clés]

Cela copie, sans la détruire, la clé spécifiée par son ID utilisateur depuis votre trousseau de clés publiques ou secrètes vers le fichier de clé spécifié. Cela est particulièrement utile si vous voulez donner une copie de votre clé publique à quelqu'un.

Si des signatures la certifiant sont attachées à la clé dans votre trousseau de clés, elles seront copiées avec la clé.

Si vous voulez que la clé extraite sorte en fichier ASCII imprimable pouvant être envoyé par e-mail, utilisez l'option -kxa.

Voir le contenu de votre trousseau de clés

Pour voir le contenu de votre trousseau de clés publiques :

pgp -kv[v] [ID_utilisateur] [trousseau_de_clés]

Cela liste toutes les clés du trousseau de clés qui correspondent à l'ID utilisateur indiqué. Si vous omettez l'ID utilisateur, toutes les clés du trousseau sont listées. Le nom du fichier du

trousseau est présumé être « pubring.pgp ». Il peut être omis ou vous pouvez spécifier « secring.pgp » si vous voulez lister les clés secrètes. Vous pouvez spécifier un nom de trousseau différent. L'extension par défaut du trousseau est « .pgp ».

Plus loin dans le manuel, nous expliquerons la notion de certification des clés au moyen des signatures. Pour voir toutes les signatures attachées à chaque clé, utilisez l'option -kvv :

pgp -kvv [ID_utilisateur] [trousseau_de_clés]

Si vous voulez spécifier un nom de fichier de trousseau particulier, mais que vous voulez voir toutes les clés qui s'y trouvent, essayez l'approche alternative :

pgp fichier_de_clé

En ne spécifiant aucune option de commande, PGP liste toutes les clés contenues dans fichier_de_clé.pgp, et essaye aussi de les ajouter à votre trousseau si elles ne s'y trouvent pas déjà.

Comment protéger les clés publiques de la falsification

Dans un cryptosystème à clé publique, vous n'avez pas à protéger les clés publiques de la divulgation. En fait, il est mieux qu'elles soient largement disséminées. Mais il est important de protéger les clés de la falsification, pour être sûr que la clé publique appartient réellement à la personne à qui elle semble appartenir. C'est peut-être la plus importante vulnérabilité des cryptosystèmes à clé publique. Voyons d'abord un désastre potentiel, avant de voir comment l'éviter sûrement avec PGP.

Supposons que vous vouliez envoyer un message privé à Alice. Vous téléchargez la clé publique d'Alice depuis un BBS [quelconque, ou un site Internet inconnu]. Vous chiffrez votre lettre à Alice avec cette clé publique et vous la lui envoyez par e-mail.

Malheureusement, à votre insu ou à l'insu d'Alice, un autre utilisateur appelé Charlie a infiltré le BBS et a lui-même généré une clé publique avec l'ID utilisateur « Alice » attaché à cette clé. Il a secrètement substitué cette fausse clé à la véritable clé d'Alice. Vous utilisez sans le savoir cette fausse clé appartenant [en réalité] à Charlie au lieu de la clé publique d'Alice. Tout semble normal parce que cette fausse clé affiche « Alice » comme ID utilisateur. Maintenant, Charlie peut déchiffrer le message destiné à Alice parce qu'il a la clé secrète correspondante. Il peut même chiffrer à nouveau le message préalablement déchiffré avec la vraie clé publique d'Alice et le lui envoyer pour que personne ne se doute de la fraude. Pire encore, il peut même faire des signatures, en apparence authentiques, d'Alice avec sa [fausse] clé secrète parce que tout le monde utilisera la fausse clé publique pour vérifier la signature d'Alice.

La seule façon d'empêcher ce désastre est d'empêcher que qui ce soit puisse falsifier les clés publiques. Si vous avez obtenu la clé publique d'Alice directement d'Alice, il n'y a pas de problème. Mais cela peut être difficile si Alice est à des milliers de kilomètres de là, ou si elle est actuellement injoignable.

Peut-être pourriez-vous vous procurer la clé publique d'Alice par l'intermédiaire de David, un ami commun en qui vous avez tous les deux confiance, et qui sait qu'il détient une copie authentique de la clé publique d'Alice. David pourrait signer la clé publique d'Alice, se portant ainsi garant de l'intégrité de la clé publique d'Alice. David créerait cette signature avec sa propre clé secrète.

Cela créerait une signature de la clé publique, et prouverait que la clé d'Alice n'a pas été falsifiée. Cela exige de disposer d'une copie reconnue authentique de la clé publique de

David pour vérifier sa signature. Peut-être que David pourrait aussi fournir à Alice une copie signée de votre clé publique. De cette manière, David sert de « certificateur » entre vous et Alice.

Cette signature de la clé publique d'Alice pourrait être mise en ligne par David ou Alice sur un BBS, et vous pourriez la télécharger ultérieurement. Vous pourriez alors vérifier la signature via la clé publique de David et être ainsi assuré qu'il s'agit réellement de la clé publique d'Alice. Aucun imposteur ne peut vous duper en vous faisant accepter sa propre fausse clé comme étant la clé d'Alice parce que personne ne peut contrefaire la signature créée par David.

Une personne largement reconnue comme digne de confiance pourrait même se spécialiser dans ce service [consistant à] « certifier » les utilisateurs les uns aux autres en signant leurs clés publiques. Cette personne de confiance pourrait être considérée comme un « serveur de clés », ou une « Autorité Certifiante ». On aurait l'assurance que toute clé publique portant la signature du serveur de clés appartient réellement à la personne à qui elle semble appartenir. Tout utilisateur intéressé n'aurait dès lors besoin que d'une copie reconnue authentique de la clé publique du serveur de clés, de sorte que les signatures du serveur de clés puissent être vérifiées [sur les clés publiques des utilisateurs].

Un serveur de clés centralisé fiable, ou Autorité Certifiante, est particulièrement adapté aux grandes institutions contrôlées depuis un centre unique comme les grandes entreprises ou les administrations. Quelques milieux institutionnels recourent au modèle de telles Autorités Certifiantes.

Pour les milieux de base plus décentralisés, moins « professionnels » et au style plus « guérilla », un modèle permettant à tous les utilisateurs d'agir comme certificateurs fiables pour leurs amis se révélera probablement mieux adapté qu'un serveur de clés centralisé. PGP tend à privilégier cette approche non institutionnelle à organigramme décentralisé. Cela reflète mieux la façon naturelle dont les êtres humains interagissent à un niveau personnel, et permet aux gens de mieux choisir à qui ils font confiance pour la gestion de leurs clés.

Toute cette affaire de la protection des clés publiques contre la falsification est le problème le plus délicat à résoudre pour les applications pratiques de la cryptographie à clé publique. C'est le talon d'Achille de la cryptographie à clé publique, et une grande partie de la complexité du logiciel est liée à la résolution de ce seul problème.

Vous ne devriez utiliser une clé publique qu'après vous être assuré qu'il s'agit d'une clé publique authentique qui n'a pas été falsifiée, et qui appartient réellement à la personne à qui la clé prétend appartenir. Vous pouvez en être sûr si vous tenez cette clé publique directement de son propriétaire, ou si elle est signée par quelqu'un en qui vous avez confiance, dont vous détenez déjà une clé publique authentique. Aussi, l'ID utilisateur devrait être le nom complet du propriétaire de la clé, et non pas seulement son nom de famille.

Peu importe combien vous pouvez être tenté – et vous serez tenté – ne cédez jamais, JAMAIS, à la facilité en faisant confiance à une clé publique que vous avez téléchargée depuis un BBS, à moins qu'elle ne soit signée par quelqu'un en qui vous avez confiance. Cette clé non certifiée pourrait avoir été falsifiée, peut-être même par l'administrateur système du BBS.

Si on vous demande de signer la clé publique de quelqu'un d'autre, assurez-vous qu'elle appartient réellement à la personne nommée dans l'ID utilisateur de cette clé publique. Et cela parce que votre signature sur sa clé est votre promesse que cette clé publique lui appartient réellement. D'autres personnes qui vous font confiance accepteront sa clé parce qu'elle porte votre signature. Il peut être malavisé de se fier au ouï-dire – ne signez pas sa clé

publique sauf si vous avez une connaissance indépendante et de première main qu'elle lui appartient vraiment. De préférence, vous ne devriez la signer que si vous l'obtenez directement [de la personne].

Pour signer une clé publique, vous devez être encore bien plus certain de l'appartenance de cette clé que si vous vouliez simplement utiliser cette clé pour chiffrer un message. Pour être convaincu qu'une clé a un aloi suffisant pour être utilisée, les signatures par des certificateurs fiables devraient suffire. Mais pour signer une clé vous-même, vous devriez recourir à votre connaissance directe, personnelle et indépendante du propriétaire de cette clé. Peut-être que vous pourriez téléphoner au propriétaire de la clé et lui lire le fichier de clé pour qu'il confirme que la clé que vous détenez est réellement sa clé – assurez-vous que vous parlez réellement à la bonne personne. Voyez le chapitre intitulé « Vérifier une clé publique par téléphone » dans le volume « Questions Spéciales » pour plus de détails.

Gardez présent à l'esprit que votre signature sur une clé publique ne garantit pas l'intégrité de cette personne, mais seulement l'intégrité (l'appartenance) de la clé publique de cette personne. Vous ne risquez pas de compromettre votre crédibilité en signant la clé publique d'un débile mental, si vous êtes absolument sûr que la clé lui appartient réellement. D'autres personnes accepteront cette clé parce que vous l'avez signée (en admettant qu'elles vous fassent confiance), mais elles n'auront pas confiance en la personne du propriétaire de cette clé. Avoir confiance en une clé n'est pas la même chose que d'avoir confiance en la personne du propriétaire de la clé.

La confiance n'est pas nécessairement transitive ; j'ai un ami dont je sais qu'il n'est pas un menteur. C'est une personne crédule, qui croit que le Président n'est pas un menteur. Cela ne signifie pas que je crois que le Président n'est pas un menteur. Ce n'est que du bon sens. Si je fais confiance à la signature d'Alice sur une clé, et qu'Alice fait confiance à la signature de Charlie sur une clé, cela n'implique pas que je doive avoir confiance en la signature de Charlie sur une clé.

Ce serait une bonne idée de garder sous la main une copie de votre propre clé publique signée par de nombreux « certificateurs », dans l'espoir que beaucoup de gens feront confiance à au moins un des certificateurs qui se sont portés garants de la validité de votre propre clé. Vous pourriez poster votre clé avec sa collection de signatures sur divers BBS. Si vous signez la clé publique d'autres personnes, renvoyez-la leur avec votre signature de telle sorte qu'elles puissent l'ajouter à leur propre collection de garants de leur propre clé publique.

PGP garde la trace de celles des clés de votre trousseau de clés publiques qui sont convenablement signées par des certificateurs en qui vous avez confiance. Tout ce que vous avez à faire est de dire à PGP qui vous jugez fiables en tant que certificateurs, et de certifier leurs clés vous-même avec votre propre clé la plus certifiée. PGP peut se servir de ça, et valider automatiquement toutes les autres clés qui ont été signées par ceux que vous avez désignés comme certificateurs. Et bien sûr, vous pouvez directement signer d'autres clés vous-même. On verra cela plus tard.

Assurez-vous que personne ne peut falsifier votre propre trousseau de clés. La vérification d'une nouvelle signature certifiant une clé publique doit dépendre en dernier ressort de l'intégrité des clés publiques certifiées qui se trouvent déjà dans votre propre trousseau de clés publiques. Gardez un contrôle physique de votre trousseau de clés publiques, de préférence sur votre propre ordinateur personnel plutôt que sur un système distant et/ou partagé, exactement comme vous le feriez pour votre clé secrète. Ceci pour le protéger de la falsification, non de la divulgation. Gardez une copie de sauvegarde fiable de vos trousseaux de clés publiques et secrètes sur un support protégé en écriture.

Dans la mesure où votre propre clé publique certifiée est utilisée comme référence pour certifier directement ou indirectement toutes les autres clés de votre trousseau, c'est celle qu'il faut protéger avec le plus grand soin de la falsification. Pour détecter une falsification de votre propre clé publique la plus certifiée, PGP peut être réglé pour comparer automatiquement votre clé publique avec une copie de sauvegarde sur un support protégé en écriture. Pour plus de détails, voyez la description de la commande de vérification « -kc » dans le Volume II « Questions Spéciales ».

D'une manière générale, PGP présume que vous conserverez le contrôle physique de votre système et de vos trousseaux de clés, ainsi que de votre copie de PGP elle-même. Si un intrus peut accéder à votre disque, alors en théorie il peut falsifier PGP lui-même, remettant en cause l'efficacité des dispositifs de sécurité dont dispose PGP pour détecter une falsification des clés.

Une méthode plus complexe pour protéger votre propre trousseau de toute falsification est de signer ce trousseau entier avec votre propre clé secrète. Vous pouvez le faire en créant une signature détachée de votre trousseau de clés publiques, en signant le trousseau avec les options « - sb » (voyez le chapitre intitulé « Separing Signatures from Messages » [« Détacher les signatures des messages »] dans le Guide de l'utilisateur de PGP, Volume II « Questions Spéciales »). Malheureusement, vous devrez toujours conserver sous la main une copie fiable séparée de votre propre clé publique pour vérifier la signature que vous avez faite. Vous ne pouvez pas compter sur votre propre clé publique contenue dans votre trousseau de clés publiques pour vérifier la signature que vous avez faite pour le trousseau entier, parce que y étant elle-même contenue, elle est une partie du tout que vous cherchez à vérifier.

Comment PGP sait quelles clés sont valides ?

Avant de lire ce chapitre, assurez-vous d'avoir lu le chapitre précédent « Comment protéger les clés publiques de la falsification ».

PGP sait quelles clés dans votre trousseau de clés publiques sont convenablement certifiées par les signatures des certificateurs en qui vous avez confiance. Tout ce que vous avez à faire est de dire à PGP qui sont les gens fiables en tant que certificateurs, et de certifier leurs clés avec votre propre clé la plus certifiée. PGP peut utiliser cette information, validant automatiquement toutes les autres clés qui ont été signées par les certificateurs. Et bien sûr, vous pouvez directement signer d'autres clés vous-même.

PGP utilise deux critères bien distincts pour apprécier la qualité d'une clé publique – ne les confondez pas :

1. Est-ce que la clé appartient réellement à la personne à qui elle semble appartenir ? En d'autres termes, a-t-elle été certifiée avec une signature digne de confiance ?
2. Est-ce qu'elle appartient à quelqu'un en qui vous pouvez avoir confiance pour certifier d'autres clés ?

PGP peut évaluer la réponse à la première question. Pour répondre à la deuxième question, PGP doit être explicitement informé par vous, l'utilisateur. Quand vous répondez à la question 2, PGP peut ensuite évaluer la réponse à la question 1 pour les autres clés signées par le certificateur que vous avez désigné comme fiable.

Les clés qui ont été certifiées par un certificateur fiable sont considérées comme valides par PGP. Les clés appartenant aux certificateurs fiables doivent être certifiées soit par vous soit par un autre certificateur fiable.

PGP offre aussi la possibilité d'établir des nuances quant au crédit que méritent les certificateurs. Votre confiance en la personne des propriétaires de clés pour agir en tant que certificateurs ne reflète pas seulement votre estimation de leur intégrité personnelle – cela devrait refléter également la sagacité que vous leur supposez dans la compréhension de la gestion des clés et dans celle de signer les clés à bon escient. Vous pouvez désigner à PGP une personne comme inconnue, non fiable, marginalement fiable, ou complètement fiable pour certifier les autres clés publiques. Cette information sur la fiabilité est conservée avec leurs clés dans votre trousseau, mais quand vous demandez à PGP de copier [extraire] une clé de votre trousseau, PGP ne copiera pas l'information sur la fiabilité avec la clé, parce que vos opinions personnelles sur la fiabilité sont considérées comme confidentielles.

Quand PGP évalue la validité d'une clé publique, il examine le niveau de fiabilité de toutes les signatures attachées. Il calcule un résultat pondéré de la validité – deux signatures marginalement fiables sont considérées comme équivalentes à une [signature] complètement fiable. L'évaluation critique de PGP est modulable – par exemple, vous pouvez régler PGP pour exiger deux signatures complètement fiables ou trois signatures marginalement fiables pour décider qu'une clé est valide.

Votre propre clé est « axiomatiquement » valide pour PGP, n'ayant pas besoin de la signature d'un certificateur pour prouver sa validité. PGP sait quelles clés publiques sont les vôtres, en regardant la clé secrète correspondante dans le trousseau de clés secrètes. PGP présume également que vous vous considérez vous-même comme complètement fiable pour certifier d'autres clés.

Avec le temps, vous accumulerez des clés d'autres personnes que vous pouvez vouloir désigner comme certificateurs fiables. Chacun choisira ses propres certificateurs fiables. Et chacun accumulera progressivement et distribuera avec sa clé une collection de signatures d'autres personnes, dans l'espoir que parmi ceux qui en détiendront une copie, il s'en trouvera pour faire confiance à au moins une ou deux des signatures. Cela permettra l'émergence d'un réseau de confiance décentralisé, à tolérance d'erreurs, pour toutes les clés publiques.

Cette approche originale par la base tranche nettement avec les schémas de la norme gouvernementale de gestion des clés publiques, comme le « Internet Privacy Enhanced Mail » (PEM), qui est basé sur un contrôle et une obligation de confiance centralisés. Le modèle normatif repose sur une hiérarchie d'Autorités Certifiantes qui décident [à votre place] à qui vous devez faire confiance. La méthode probabiliste décentralisée de PGP pour déterminer l'aloi des clés publiques est la poutre maîtresse de l'architecture de son modèle de gestion des clés. PGP vous laisse choisir vous-même ceux qui méritent votre confiance, vous plaçant au sommet de votre propre pyramide personnelle de certification. PGP est destiné aux gens qui préfèrent plier eux-mêmes leur propre parachute.

Comment protéger ses clés secrètes de la divulgation

Protégez votre propre clé secrète et votre phrase de passe soigneusement. Vraiment, vraiment soigneusement. Si jamais votre clé secrète est compromise, vous feriez mieux de le faire savoir à toutes les parties concernées (bonne chance) avant qu'on l'utilise pour signer en votre nom. Par exemple, on pourrait l'utiliser pour créer de fausses vraies signatures, qui pourraient créer des problèmes à beaucoup de monde, surtout si votre signature est largement considérée comme fiable. Et bien sûr, une compromission de votre propre clé secrète compromettrait tous les messages qui vous sont envoyés.

Pour protéger votre clé secrète, vous pouvez commencer par toujours conserver le contrôle physique de votre clé secrète. Il est bon de la conserver sur votre ordinateur personnel à la maison, ou sur un ordinateur portable que vous pouvez emmener avec vous. Si vous devez utiliser au bureau un ordinateur dont vous n'avez pas en permanence le contrôle physique, alors gardez vos trousseaux de clés publiques et secrètes sur une disquette protégée en écriture, et ne l'oubliez pas en quittant le bureau. Ce ne serait pas une bonne idée de conserver votre clé secrète sur un ordinateur distant et/ou partagé, comme un système de type Unix connecté en permanence. Quelqu'un pourrait intercepter la ligne de votre modem et capturer votre phrase de passe, et ensuite se procurer votre clé secrète depuis le système distant. Vous ne devriez utiliser votre clé secrète que sur une machine placée sous votre contrôle physique.

Ne conservez pas votre phrase de passe sur l'ordinateur sur lequel se trouve votre clé secrète. Conserver ensemble la clé secrète et la phrase de passe sur le même ordinateur est aussi dangereux que de garder votre code secret de carte bancaire dans le même portefeuille que la carte. Vous ne voulez pas que quelqu'un mette la main sur votre disque contenant à la fois la phrase de passe et le fichier de clé secrète. Il serait plus sûr de simplement mémoriser votre phrase de passe et de ne pas la conserver ailleurs que dans votre cerveau. Si vous sentez que vous devez écrire votre phrase de passe, protégez-la bien, peut-être mieux encore que la clé secrète.

Et conservez des copies de sauvegarde de votre trousseau de clés secrètes – rappelez-vous, vous détenez l'unique exemplaire de votre clé secrète, et la perdre rendra inutilisables toutes les copies de votre clé publique que vous avez diffusées à travers le monde.

L'approche décentralisée non institutionnelle utilisée par PGP pour gérer les clés publiques a ses avantages, mais malheureusement elle signifie aussi qu'on ne peut pas compter sur une liste centralisée unique des clés compromises. Cela rend beaucoup plus difficile de limiter les dégâts causés par une compromission de clé secrète. Vous ne pouvez que le faire savoir et espérer que tout le monde en entendra parler.

Si le pire des cas survient – votre clé secrète et votre phrase de passe sont toutes les deux compromises (espérons que vous vous en apercevrez) – vous devrez émettre un certificat de « révocation de clé compromise ». Ce type de certificat est utilisé pour prévenir les gens d'arrêter d'utiliser votre clé publique. Vous pouvez utiliser PGP pour créer un tel certificat en utilisant la commande « -kd ». Ensuite, vous devez d'une manière ou d'une autre envoyer ce certificat de révocation à tout le monde sur la planète, ou au moins à tous vos amis et à leurs amis, etc. Leur propre logiciel PGP installera ce certificat de révocation dans leur trousseau de clés publiques et les empêchera automatiquement d'utiliser votre clé publique à l'avenir. Vous pouvez alors générer une nouvelle paire de clés secrète/publique et publier la nouvelle clé publique. Vous pourriez diffuser un « lot » contenant votre nouvelle clé publique et le certificat de révocation de votre ancienne clé.

Révoquer une clé publique

Supposons que votre clé secrète et votre phrase de passe soient d'une manière ou d'une autre toutes les deux compromises. Vous devez le faire savoir au reste du monde, de sorte qu'on arrête d'utiliser votre clé publique. Pour ce faire, vous aurez à émettre un certificat de « révocation de clé compromise », ou certificat de « révocation de clé » pour révoquer votre clé publique.

Pour générer un certificat pour révoquer votre propre clé, utilisez la commande -kd :

```
pgp -kd votre_ID_utilisateur
```

Ce certificat porte votre signature, faite avec la clé que vous êtes en train de révoquer. Vous devriez largement diffuser ce certificat de révocation de clé aussitôt que possible. Les personnes qui le recevront peuvent l'ajouter à leur trousseau de clés publiques, et leur logiciel PGP les empêchera alors automatiquement d'utiliser votre ancienne clé publique à l'avenir. Vous pouvez alors générer une nouvelle paire de clés secrète/publique et publier la nouvelle clé publique.

Vous pouvez choisir de révoquer votre clé pour d'autres raisons que la compromission de la clé secrète. Si c'est le cas, vous aurez à utiliser la même procédure pour la révoquer.

Que faire si vous perdez votre clé secrète ?

Normalement, si vous voulez révoquer votre propre clé secrète, vous pouvez utiliser la commande « -kd » pour émettre un certificat de révocation, signé avec votre propre clé secrète (voir « Révoquer une clé publique »).

Mais que pouvez-vous faire si vous perdez votre clé secrète, ou si votre clé secrète est détruite ? Vous ne pouvez pas la révoquer vous-même, parce que vous devez utiliser votre propre clé secrète pour la révoquer, et vous ne l'avez plus. Une future version de PGP offrira une méthode de substitution pour révoquer des clés dans ces circonstances, autorisant des certificateurs fiables à certifier qu'une clé publique a été révoquée. Mais pour l'instant, vous aurez à le faire savoir par tout moyen, en demandant aux autres utilisateurs de « désactiver » votre clé publique dans leurs propres trousseaux personnels de clés publiques.

Les autres utilisateurs pourront désactiver votre clé publique dans leurs propres trousseaux de clés publiques en utilisant la commande « -kd ». Si un ID utilisateur est spécifié qui ne correspond pas à une clé secrète dans le trousseau de clés secrètes, la commande « -kd » cherchera cet ID utilisateur dans le trousseau de clés publiques, et marquera cette clé publique comme désactivée. Une clé désactivée ne sera pas utilisée pour chiffrer des messages, et ne peut pas être extraite du trousseau avec la commande -kx. Elle peut toujours servir à vérifier des signatures, mais un avertissement est affiché. Et si l'utilisateur essaye d'ajouter de nouveau la même clé, cela n'aboutira pas parce que la clé désactivée se trouve déjà dans le trousseau. Ces fonctionnalités combinées aideront à entraver la diffusion d'une clé désactivée.

Si la clé publique spécifiée est déjà désactivée, la commande -kd vous demandera si vous voulez réactiver la clé.

Questions Avancées

La plupart de ces « Questions Avancées » sont couvertes dans le « Guide de l'utilisateur de PGP, Volume II : Questions Spéciales ». Mais voici quelques questions qui peuvent être mentionnées ici.

Envoyer du texte chiffré à travers les canaux e-mail : le format Radix-64

La plupart des systèmes d'e-mail n'autorisent que les messages en texte ASCII, et non le binaire brut en 8-bit dont est constitué le texte chiffré. Pour contourner ce problème, PGP supporte le format ASCII radix-64 pour les messages chiffrés, identique au format du Privacy Enhanced Mail (PEM), ou au format Internet MIME. Ce format spécial représente des données binaires en utilisant uniquement des caractères ASCII imprimables, ce qui est utile pour transmettre des données binaires chiffrées à travers les canaux 7-bit ou pour envoyer des données chiffrées comme du texte e-mail normal. Ce format agit comme une forme d'« armure de transport », le protégeant contre la corruption lorsqu'il voyage à travers les passerelles intersystèmes d'Internet. PGP annexe aussi un CRC pour détecter les erreurs de transmission.

Le format Radix-64 convertit le texte clair en étendant les groupes de 3 octets binaires de 8-bit en 4 caractères ASCII imprimables, aussi le fichier s'accroît-il d'environ 33%. Mais l'expansion n'est pas si mauvaise quand vous considérez que le fichier était probablement compressé davantage que cela par PGP avant d'avoir été chiffré.

Pour produire un fichier chiffré au format ASCII radix-64 format, ajoutez simplement l'option « a » en chiffrant ou en signant un message, comme cela :

```
pgp -esa message.txt son_ID_utilisateur
```

Cet exemple produit un fichier chiffré appelé « message.asc » qui contient les données dans un format ASCII radix-64 ressemblant au format MIME. Ce fichier peut être aisément récupéré dans un éditeur de texte à travers les canaux 7-bit pour une transmission comme un e-mail normal sur Internet ou tout autre réseau d'e-mails.

Déchiffrer le message en armure de transport radix-64 n'est pas différent d'un déchiffrement normal. Par exemple :

```
pgp message
```

PGP recherche automatiquement le fichier ASCII « message.asc » avant de chercher le fichier binaire « message.pgp ». Il reconnaît que le fichier est au format radix-64 et le reconvertit dans son format binaire avant de procéder comme il le fait normalement, produisant ainsi un fichier chiffré « .pgp » de forme binaire. Le résultat final est dans une forme de texte clair normal, exactement comme il l'était dans le fichier original « message.txt ».

La plupart des infrastructures e-mail d'Internet interdisent l'envoi de messages de plus de 50000 ou 65000 octets. Les messages plus longs doivent être coupés en plus petits morceaux qui peuvent être envoyés séparément. Si votre message chiffré est très grand, et que vous demandez un format radix-64, PGP le coupera automatiquement en morceaux qui sont chacun assez petits pour être envoyés par e-mail. Les morceaux sont mis dans des fichiers d'extension « .as1 », « .as2 », « .as3 », etc. Le destinataire doit concaténater ces morceaux séparés dans le bon ordre à l'intérieur d'un gros fichier avant de le déchiffrer. En déchiffrant, PGP ignore tout texte superflu dans les en-têtes d'e-mail qui ne sont pas compris dans les blocs du message en radix-64.

Si vous voulez envoyer une clé publique au format radix-64 à quelqu'un, ajoutez juste l'option -a en extrayant la clé de votre trousseau de clés.

Si vous avez oublié d'utiliser l'option -a quand vous avez chiffré un fichier ou extrait une clé, vous pouvez encore directement convertir le fichier binaire au format radix-64 en utilisant

simplement l'option -a toute seule, sans aucun chiffrement spécifié. PGP le convertit en un fichier « .asc ».

Si vous signez un fichier en clair sans le chiffrer, PGP le compressera normalement après l'avoir signé, le rendant illisible. C'est une façon commode de stocker des fichiers signés dans des applications d'archivage. Mais si vous voulez envoyer le message signé comme e-mail, et que le fichier original est sous forme de texte (et non en binaire), il y a moyen de l'envoyer à travers un canal e-mail de telle sorte que le texte clair ne soit pas compressé, et que l'armure ASCII ne soit appliquée que sur la signature binaire, mais pas sur le message en clair. Cela permet au destinataire de lire le message signé, sans l'aide de PGP. Bien sûr, PGP reste nécessaire pour vérifier la signature. Pour plus d'informations sur cette fonctionnalité, voir l'explication du paramètre CLEARSIG dans le chapitre « Setting Configuration Parameters » [« Réglage des paramètres de configuration »] du Volume II « Questions Spéciales ».

Parfois, vous pouvez vouloir envoyer un fichier de données binaires à travers un canal e-mail sans le chiffrer ou le signer avec PGP. Des gens utilisent l'utilitaire Unix uuencode dans ce but. PGP peut aussi être utilisé à cet effet, en utilisant simplement l'option -a toute seule, et il fait mieux le travail que l'utilitaire uuencode. Pour plus de détails, voir le chapitre « Using PGP as a Better Uuencode » [« Utiliser PGP comme un Uuencode amélioré »] du Volume II « Questions Spéciales ».

Variable d'environnement du nom de chemin

PGP utilise plusieurs fichiers spéciaux pour ses besoins, ainsi vos trousseaux de clés normaux « pubring.pgp » et « secring.pgp », le fichier alimentant le générateur de nombres pseudo aléatoires « randseed.bin », le fichier de configuration de PGP « config.txt » (ou « pgp.ini », ou « .pgprc »), et le fichier « language.txt » traduisant les instructions [du logiciel] en langue étrangère. Ces fichiers spéciaux peuvent être placés dans n'importe quel répertoire, en réglant la variable d'environnement « PGPPATH » sur le nom de chemin désiré. Par exemple, sous MSDOS, la commande de shell :

SET PGPPATH=C:\PGP

fait que PGP considère que le nom de votre trousseau de clés publiques est « C:\PGP\pubring.pgp ». En présumant, bien sûr, que ce répertoire existe. Utilisez votre éditeur de texte favori pour modifier le fichier MSDOS AUTOEXEC.BAT pour régler automatiquement cette variable chaque fois que vous démarrez votre système. Si PGPPATH n'est pas défini, ces fichiers spéciaux sont considérés comme se trouvant dans le répertoire courant.

Régler les paramètres dans le fichier de configuration de PGP

PGP dispose d'un certain nombre de paramètres réglables par l'utilisateur qui peuvent être définis dans un fichier texte spécial de configuration de PGP appelé « config.txt », dans le répertoire vers lequel fait pointer la variable d'environnement PGPPATH. Disposer d'un fichier de configuration permet à l'utilisateur de définir diverses instructions et paramètres pour PGP sans le fardeau d'avoir à toujours définir ces paramètres dans la ligne de commande de PGP.

Dans le but de se conformer aux conventions de nom propres aux systèmes, pour les systèmes Unix ce fichier de configuration peut aussi être nommé « .pgprc », et sur les systèmes MSDOS il peut aussi être nommé « pgp.ini ».

Avec ces paramètres de configuration, par exemple, vous pouvez contrôler à quel endroit PGP stocke ses fichiers temporaires, ou vous pouvez sélectionner quelle langue étrangère PGP utilisera pour afficher ses messages de diagnostic et ses demandes à l'utilisateur, ou vous pouvez ajuster le niveau de l'évaluation critique de PGP dans la détermination de la validité d'une clé selon le nombre de signatures certifiées qu'elle possède.

Pour plus de détails sur le réglage de ces paramètres de configuration, voir le chapitre approprié du Guide de l'utilisateur de PGP, Volume II « Questions Spéciales ».

Vulnérabilités

Aucun système de sécurité des données n'est impénétrable. PGP peut être circonvenu par une variété de biais. Les vulnérabilités potentielles dont vous devez être conscients incluent la compromission de votre phrase de passe ou de votre clé secrète, la falsification de clé publique, les fichiers que vous avez effacés mais qui sont encore quelque part sur le disque, les virus et les chevaux de Troie, les brèches dans votre sécurité physique, les émissions électromagnétiques, la divulgation sur des systèmes multi-utilisateurs, l'analyse de trafic, et peut-être même la cryptanalyse directe.

Pour une discussion détaillée sur ces problèmes, voir le chapitre « Vulnérabilités » du Guide de l'utilisateur de PGP, Volume II « Questions Spéciales ».

Prenez garde à « l'Huile de Serpent »

Quand on examine un logiciel de cryptographie, la question qui se pose toujours est : pourquoi devriez-vous faire confiance à ce produit ? Même si vous examinez le code source par vous-même, tout le monde n'a pas l'expérience cryptographique pour en apprécier la sécurité. Même si vous êtes un cryptographe expérimenté, de subtiles faiblesses dans les algorithmes peuvent toujours vous échapper.

Quand j'étais au collège, au début des années 70, j'avais conçu ce que je croyais être un schéma de chiffrement génial. Un simple flux pseudo aléatoire était ajouté au flux de texte clair pour créer un texte chiffré. Cela devait apparemment contrecarrer toute analyse de fréquence sur le texte chiffré, et être incassable même pour les services de renseignement du Gouvernement disposant des plus grandes ressources qui soient. Je me sentais si suffisant à propos de mon exploit. Sûr comme un coq.

Des années plus tard, je découvris le même schéma dans de nombreux textes d'introduction à la cryptographie et des articles de cours. Comme c'était charmant. Les autres cryptographes avaient pensé au même schéma. Malheureusement, le schéma était présenté comme un simple devoir d'écolier sur comment utiliser des techniques cryptographiques élémentaires pour les craquer banalement. Autant pour mon schéma génial.

De ma modeste expérience, j'ai appris combien il est facile de verser dans une conception erronée de la sécurité quand on conçoit un algorithme de chiffrement. La plupart des gens ne réalisent pas combien il est diablement difficile de concevoir un algorithme de chiffrement qui puisse résister à une attaque prolongée et déterminée par un adversaire possédant de grandes ressources. Beaucoup d'ingénieurs informaticiens sur grands systèmes ont développé des schémas de chiffrement aussi naïfs (souvent même exactement le même schéma), et certains d'entre eux ont été incorporés dans des logiciels de chiffrement commerciaux et vendus contre argent sonnant et trébuchant à des milliers d'utilisateurs ne soupçonnant rien.

C'est comme vendre des ceintures de sécurité d'automobile qui ont une bonne apparence et semblent efficaces, mais s'ouvrent d'un coup même au plus petit test d'accident. Compter sur elles peut être pire que de ne pas porter de ceinture du tout. Personne ne suspecte qu'elles sont mauvaises jusqu'à l'accident réel. Compter sur un logiciel de cryptographie faible peut faire mettre inconsciemment en danger des informations sensibles. Vous ne l'auriez pas fait si vous n'aviez pas eu du tout de logiciel de cryptographie. Peut-être ne découvrirez vous jamais que vos données ont été compromises.

Parfois, les logiciels commerciaux utilisent le standard fédéral américain Data Encryption Standard (DES), un assez honnête algorithme conventionnel recommandé par le Gouvernement américain pour l'utilisation commerciale (mais pas pour l'information classée secret défense, chose curieuse – hmmm). Il y a plusieurs « modes d'opération » que le DES peut utiliser, certains d'entre eux sont meilleurs que d'autres. Le Gouvernement recommande expressément de ne pas utiliser le mode le plus simple et le plus faible pour les messages, le mode Electronic Codebook (ECB). En revanche, on recommande les modes plus résistants et plus complexes Cipher Feedback (CFB) ou Cipher Block Chaining (CBC).

Malheureusement, la plupart des logiciels commerciaux de cryptographie que j'ai examinés utilisent le mode ECB. Quand j'en ai parlé aux auteurs de plusieurs de ces réalisations, ils ont dit qu'ils n'avaient jamais entendu parler des modes CBC ou CFB, et qu'ils ne savaient rien au sujet de la faiblesse du mode ECB. Le fait même qu'ils n'aient jamais étudié assez de cryptographie pour connaître ces concepts élémentaires n'est pas rassurant. Et ils gèrent parfois leurs clés DES d'une manière inadéquate ou non sûre. De même, ces logiciels incluent souvent un second algorithme plus rapide qui peut être utilisé à la place du DES plus lent. L'auteur du logiciel pense souvent que son algorithme propriétaire plus rapide est aussi sûr que le DES, mais après l'avoir questionné je découvre habituellement que c'est juste une variation de mon génial schéma des jours de collège. Ou peut-être ne révélera-t-il jamais comment son schéma de chiffrement propriétaire fonctionne, mais il m'assure que c'est un schéma génial et que je devrais lui faire confiance. Je suis sûr qu'il croit que son algorithme est génial, mais comment puis-je le savoir sans le voir ?

En toute justice, je dois signaler que dans la plupart des cas ces produits lamentables ne proviennent pas de sociétés qui se spécialisent dans la technologie cryptographique.

Même les très bons logiciels, qui utilisent le DES dans le mode d'opération correct présentent encore des problèmes. Le standard DES utilise une clé de 56 bits, ce qui est trop petit pour les normes actuelles, et peut maintenant être aisément cassée par des recherches exhaustives de la clé sur des machines ultra rapides spéciales. Le DES a atteint la fin de sa vie utile, et voilà pourtant encore des logiciels qui y font appel.

Il y a une société appelée AccessData (87 East 600 South, Orem, Utah 84058, phone 1-800-658-5199) qui vend un ensemble pour $185 qui craque le schéma de chiffrement intégré utilisé par WordPerfect, Lotus 1-2-3, MS Excel, Symphony, Quattro Pro, Paradox, et MS Word 2.0. Il ne recherche pas simplement les mots de passe – il fait vraiment de la cryptanalyse. Des gens l'achètent quand ils ont oublié leur mot de passe pour leurs propres fichiers. Les services de police judiciaire l'achètent aussi, ainsi peuvent-ils lire les fichiers qu'ils saisissent. J'ai parlé à Eric Thompson, l'auteur, et il a dit que son programme prend seulement une demi seconde pour les craquer, mais qu'il a intégré une boucle retardatrice pour le ralentir de sorte que cela ne semble pas trop facile au client. Il m'a aussi dit que la fonction de chiffrement par mot de passe des fichiers PKZIP peut souvent être cassée facilement, et que ses clients de la police judiciaire recourent régulièrement à ce genre de services fournis par un autre vendeur.

D'un certain point de vue, la cryptographie est comme la pharmacie. Sa qualité peut être absolument cruciale. La mauvaise pénicilline a la même apparence que la bonne. Vous pouvez juger que votre tableur est mauvais, mais comment juger que votre logiciel de cryptographie est faible ? Le texte chiffré produit par un algorithme de chiffrement faible paraît aussi bon que le texte chiffré produit par un algorithme de chiffrement résistant. Il y a beaucoup d'huile de serpent là-dedans. Beaucoup de remèdes de charlatan. Contrairement aux colporteurs de remèdes de charlatans, ces programmeurs de logiciels ne savent habituellement même pas que leur truc est de l'huile de serpent. Ils sont peut-être de bons ingénieurs informaticiens, mais ils n'ont habituellement même pas lu d'ouvrages universitaires de cryptographie. Mais ils croient quand même qu'ils peuvent écrire de bons logiciels de cryptographie. Et pourquoi pas ? Après tout, cela semble intuitivement facile à faire. Et leurs logiciels semblent bien marcher.

Quiconque croit avoir inventé un schéma de chiffrement incassable est, soit un véritable génie, soit un naïf inexpérimenté. Malheureusement, j'ai quelquefois affaire à ces prétendus cryptographes qui veulent apporter des « améliorations » à PGP en lui ajoutant des algorithmes de chiffrement de leur cru.

Je me souviens d'une conversation avec Brian Snow, un cryptographe haut placé et ancien de la NSA. Il me dit qu'il ne ferait jamais confiance à un algorithme de chiffrement conçu par quelqu'un qui ne s'était pas « fait les dents » en passant d'abord beaucoup de temps à casser des codes. Cela tombait sous le sens. J'observai que pratiquement personne dans le monde de la cryptographie commerciale n'était qualifié selon ce critère. « Oui », répondit-il avec un sourire entendu, « Et cela rend notre travail à la NSA tellement plus facile. » Une réflexion à vous glacer le sang. Je n'en aurais pas jugé autrement.

Le Gouvernement américain a également colporté l'huile de serpent. Après la Seconde Guerre mondiale, les USA vendirent les machines à chiffrer allemandes Enigma aux gouvernements du Tiers monde. Mais ils ne leur dirent pas que les Alliés avait cassé le code Enigma pendant la guerre, un fait qui resta classé secret défense pendant de nombreuses années. Aujourd'hui encore, de nombreux système Unix dans le monde entier utilisent l'algorithme d'Enigma pour le chiffrement de fichiers, en partie parce que le Gouvernement a dressé des obstacles légaux contre l'utilisation de meilleurs algorithmes. Ils essayèrent même d'empêcher la publication initiale de l'algorithme RSA en 1977. Et ils ont brisé dans l'œuf toutes les tentatives [de l'industrie] pour développer des téléphones réellement sécurisés pour le grand public.

Le principal travail de la NSA (National Security Agency) du Gouvernement américain est de recueillir des renseignements, principalement en enregistrant secrètement les communications privées des gens (voir le livre de James Bamford, « The Puzzle Palace »). La NSA a

accumulé des compétences et des ressources considérables pour casser des codes. Quand les gens ne peuvent pas disposer de bonne cryptographie pour se protéger, cela rend le travail de la NSA plus facile. La NSA a également la responsabilité d'approuver et recommander des algorithmes de chiffrement. Des critiques soutiennent que c'est une source de conflits d'intérêts, comme mettre le renard à garder le poulailler. La NSA a poussé en avant un algorithme de chiffrement conventionnel qu'elle avait conçu, et elle ne dira à personne comment il fonctionne parce que c'est classé secret défense. Elle veut qu'on lui fasse confiance et qu'on l'utilise. Mais n'importe quel cryptographe vous dira qu'un algorithme bien conçu n'a pas à être classé secret défense pour rester sûr. Seules les clés pourraient avoir besoin de protection. Comment fait-on pour savoir vraiment si l'algorithme classé secret défense de la NSA est sûr ? Il n'est pas difficile pour la NSA de concevoir un algorithme de chiffrement qu'elle seule peut craquer, si personne ne peut examiner l'algorithme. Est-elle délibérément en train de vendre de l'huile de serpent ?

Il y a trois facteurs principaux qui ont miné la qualité des logiciels commerciaux de cryptographie aux USA. Le premier est le manque virtuellement universel de compétence des programmeurs de logiciels commerciaux de cryptographie (quoique cela commence à changer depuis la sortie de PGP). Chaque ingénieur informaticien se prend pour un cryptographe, ce qui a conduit à la prolifération de logiciels de crypto vraiment mauvais. Le second est que la NSA a délibérément et systématiquement éliminé toutes les bonnes technologies commerciales de chiffrement, par l'intimidation légale et la pression économique. Une partie de cette pression a été portée à son maximum par les rigoureux contrôles à l'exportation sur les logiciels de cryptographie ce qui, vu l'aspect financier du marketing logiciel, a eu pour résultat d'éliminer les logiciels de chiffrement domestiques. L'autre principale méthode d'élimination vient de la concession de tous les brevets portant sur tous les algorithmes de chiffrement à clé publique à une seule société, constituant un goulot d'étranglement pour empêcher l'extension de cette technologie. Le résultat tangible de tout cela est qu'avant la sortie de PGP, il n'y avait presque pas de logiciels de chiffrement de haute sécurité disponibles aux USA.

Je ne suis pas aussi certain de la sécurité de PGP que je l'étais autrefois de celle de mon génial schéma de chiffrement du collège. Si je l'étais, cela serait mauvais signe. Mais je suis à peu près sûr que PGP ne contient pas de faiblesses manifestes (bien qu'il puisse contenir des bogues). Les algorithmes de crypto ont été développés par des personnes hautement qualifiées dans les milieux de la cryptographie universitaire civile, et chacun d'eux a été soumis à un examen approfondi étendu. Le code source est disponible pour faciliter l'examen approfondi de PGP et aider à dissiper les craintes de certains utilisateurs. Il est raisonnablement bien étudié, et cela a pris des années pour le faire. Et je ne travaille pas pour la NSA. J'espère qu'on n'aura pas à recourir à un « acte de foi » pour croire à la sécurité de PGP.

Notice pour les utilisateurs de Macintosh

PGP a été développé à l'origine pour les machines sous MSDOS et Unix. Il y a aussi une version pour Apple Macintosh de PGP. Ce manuel est écrit pour les versions MSDOS/Unix de PGP, qui utilisent une interface en mode caractères pour toutes les fonctions de PGP. Sur

le Mac, toutes les fonctions de PGP sont accessibles à travers des menus déroulants et des boîtes de dialogues. Il y a aussi une aide en ligne sur le Mac pour savoir comment utiliser MacPGP, et il devrait y avoir une documentation spécifique au Mac incluse dans l'archive de la version MacPGP, en plus de ce manuel.

La plupart des bons logiciels pour Mac sont écrits directement pour le Mac et avec son style, et non simplement portés depuis un autre système d'exploitation. Malheureusement, l'actuelle version Mac de PGP n'a pas été conçue pour le Mac dans cet esprit. Elle a été portée depuis la version MSDOS/Unix vers le Mac par Zbigniew Fiedorwicz. Comme la version MSDOS/Unix de PGP n'a pas été conçue pour une GUI (interface graphique utilisateur), la porter pour le Mac n'était pas une tâche facile, et beaucoup de bogues demeurent. Une toute nouvelle version de PGP est en développement, conçue pour une adaptation facile à une GUI. Une nouvelle version Mac sera développée depuis ce nouveau code source de PGP. Cela ressemblera plus à du Mac, et ce sera plus fiable. En dépit des bogues de la version actuelle de MacPGP, il est important de noter que si Zbigniew avait attendu que cette toute nouvelle version de PGP soit développée pour porter PGP sur le Mac, le monde aurait été privé d'une version Mac de PGP pendant trop longtemps.

Aide-mémoire de PGP

Voici un rapide résumé des commandes de PGP.

Pour chiffrer un fichier clair avec la clé publique du destinataire :
pgp -e fichier son_ID_utilisateur

Pour signer un fichier clair avec votre clé secrète :
pgp -s fichier [-u votre_ID_utilisateur]

Pour signer un fichier clair de texte ASCII avec votre clé secrète, produisant un message signé de texte clair pouvant être envoyé par e-mail :
pgp -sta fichier [-u votre_ID_utilisateur]

Pour signer un fichier clair avec votre clé secrète, puis le chiffrer avec la clé publique du destinataire :
pgp -es fichier son_ID_utilisateur [-u votre_ID_utilisateur]

Pour chiffrer un fichier clair de manière conventionnelle seulement, tapez :
pgp -c fichier

Pour déchiffrer un fichier chiffré, ou pour vérifier l'intégrité d'un fichier signé :
pgp fichier_chiffré [-o fichier_en_clair]

Pour chiffrer un message à l'intention de plusieurs destinataires :
pgp -e fichier ID_utilisateur1 ID_utilisateur2 ID_utilisateur3

- **Commandes de gestion des clés :**

Pour générer votre propre paire de clés publique/secrète :
pgp -kg

Pour ajouter le contenu d'un fichier de clés à votre trousseau de clés publiques ou secrètes :
pgp -ka fichier_de_clés [trousseau_de_clés]

Pour extraire (copier) une clé de votre trousseau de clés publiques ou secrètes :
pgp -kx ID_utilisateur fichier_de_clé [trousseau_de_clés]
ou : pgp -kxa ID_utilisateur fichier_de_clé [trousseau_de_clés]

Pour visualiser le contenu de votre trousseau de clés publiques :
pgp -kv[v] [ID_utilisateur] [trousseau_de_clés]

Pour visualiser l'« empreinte » d'une clé publique, afin de la vérifier au téléphone avec son propriétaire :
pgp -kvc [ID_utilisateur] [trousseau_de_clés]

Pour visualiser le contenu et vérifier les signatures de votre trousseau de clés publiques :
pgp -kc [ID_utilisateur] [trousseau_de_clés]

Pour éditer [modifier] l'ID_utilisateur ou la phrase de passe de votre clé secrète :
pgp -ke ID_utilisateur [trousseau_de_clés]

Pour éditer [modifier] les paramètres de fiabilité d'une clé publique :
pgp -ke ID_utilisateur [trousseau_de_clés]

Pour effacer une clé ou seulement un ID_utilisateur de votre trousseau de clés publiques :
pgp -kr ID_utilisateur [trousseau_de_clés]

Pour signer et ainsi certifier la clé publique de quelqu'un dans votre trousseau de clés publiques :
pgp -ks son_ID_utilisateur [-u votre_ID_utilisateur] [trousseau_de_clés]

Pour enlever certaines signatures d'un ID utilisateur dans un trousseau de clés :
pgp -krs ID_utilisateur [trousseau_de_clés]

Pour révoquer définitivement votre propre clé, émettant un certificat de clé compromise :
pgp -kd votre_ID_utilisateur

Pour désactiver ou réactiver une clé publique dans votre propre trousseau de clés publiques :
pgp -kd ID_utilisateur

- **Commandes ésotériques :**

Pour déchiffrer un message [signé] et y laisser la signature qui s'y trouve intacte :
pgp -d fichier_chiffré

Pour créer une signature détachée du document :
pgp -sb fichier [-u votre_ID_utilisateur]

Pour détacher la signature du message signé :
pgp -b fichier_chiffré

- **Options de commande qui peuvent être utilisées en combinaison avec d'autres options de commande (parfois même en épelant certains mots intéressants !) :**

Pour produire un texte chiffré au format ASCII radix-64, ajoutez simplement l'option -a lors du chiffrement ou de la signature d'un message ou de l'extraction [copie] d'une clé :
pgp -sea fichier son_ID_utilisateur
ou : pgp -kxa ID_utilisateur fichier_de_clé [trousseau_de_clés]

Pour détruire le fichier clair après la production du texte chiffré, ajoutez simplement l'option -w (wipe) lors du chiffrement ou de la signature d'un message :

pgp -sew message.txt son_ID_utilisateur

Pour spécifier que le fichier clair contient du texte ASCII, et pas du binaire, et devrait être converti suivant les règles des sauts de ligne du destinataire, ajoutez l'option -t (texte) aux autres options :

pgp -seat message.txt son_ID_utilisateur

Pour visualiser la sortie du fichier déchiffré sur votre écran (dans le genre de la commande Unix « more »), sans l'écrire dans un fichier, utilisez l'option -m (more) en déchiffrant :

pgp -m fichier_chiffré

Pour spécifier que le fichier déchiffré par le destinataire sera SEULEMENT affiché sur son écran et ne pourra pas être sauvegardé sur le disque, ajoutez l'option -m :

pgp -steam message.txt son_ID_utilisateur

Pour récupérer le nom originel du fichier clair en le déchiffrant, ajoutez l'option -p :

pgp -p fichier_chiffré

Pour utiliser un filtre dans le genre des commandes Unix, lisant depuis l'entrée standard et écrivant vers la sortie standard, ajoutez l'option -f :

pgp -feast son_ID_utilisateur <fichier_entrée> fichier_sortie

Questions légales

Pour des informations détaillées sur la licence, la distribution, les copyrights, les brevets, marques déposées, limitations de responsabilité, et contrôle de l'exportation en dehors des USA, voyez le chapitre « Questions légales » dans le « Guide de l'utilisateur de PGP, Volume II : Questions Spéciales ».

PGP utilise un algorithme à clé publique couvert par un brevet américain (U.S. patent #4,405,829). Les droits exclusifs de licence de ce brevet sont détenus par une compagnie appelée Public Key Partners (PKP), et vous pouvez enfreindre ses droits si vous utilisez PGP aux USA sans une licence. Ces questions sont détaillées dans le Volume II du manuel, et dans la licence RSAREF qui accompagne la version freeware de PGP. PKP a accordé certains droits sur la licence à d'autres, dont la compagnie connue sous le nom de ViaCrypt, à Phoenix, Arizona, aux USA. ViaCrypt vend une version avec licence complète de PGP. [Viacrypt a été racheté par PGP, Inc. en 1996, qui a été racheté par Network Associates, Inc. en 1997 : http://www.nai.com]

PGP est un freeware de « guérilla », et cela ne me gêne pas que vous le distribuiez largement. Juste une chose : ne me demandez pas de vous en envoyer une copie. Vous pouvez le trouver sur de nombreux BBS et sites FTP d'Internet. Mais avant de le distribuer [depuis les USA ou le Canada], il est essentiel que vous compreniez les contrôles américains sur l'exportation des logiciels de cryptographie.

Remerciements

De formidables obstacles et des forces puissantes ont été alignés pour arrêter PGP. Les personnes à qui il est dédié ont aidé et aident encore à surmonter ces obstacles. PGP a obtenu la notoriété en tant que « logiciel souterrain », et faire sortir PGP des catacombes pour en faire un logiciel freeware avec une licence en règle a requis de la patience et de la ténacité. Je voudrais remercier tout particulièrement Hal Abelson, Jeff Schiller, Brian LaMacchia, et Derek Atkins du MIT pour leur efforts déterminés. Je voudrais aussi remercier Jim Bruce et David Litster de l'administration du MIT et Bob Prior et Terry Ehling des Presses du MIT. Et je voudrais remercier toute l'équipe de mes défenseurs dont le travail n'est pas encore terminé. J'avais l'habitude de faire un tas de plaisanteries sur les avocats avant de rencontrer des exemples aussi positifs dans l'équipe de mes défenseurs, la plupart d'entre eux ayant travaillé bénévolement.

Le développement de PGP a tourné au phénomène social significatif, dont l'appel politiquement original à le soutenir a inspiré les efforts collectifs d'un nombre grandissant de programmeurs bénévoles. Vous rappelez-vous l'histoire pour les enfants appelée la « Soupe aux cailloux » ?

J'aimerais remercier les personnes suivantes pour leurs contributions à la création de Pretty Good Privacy. Bien que je sois l'auteur de PGP version 1.0, la plus grande partie des dernières versions de PGP fut réalisée grâce à un effort de coopération internationale impliquant un grand nombre de contributeurs, sous ma maîtrise d'œuvre.

Branko Lankester, Hal Finney et Peter Gutmann contribuèrent tous en accordant une part énorme de leur temps à l'ajout de fonctionnalités à PGP 2.0, et l'ont porté sur les variantes d'Unix.

Hugh Kennedy l'a porté sur VAX/VMS, Lutz Frank l'a porté sur Atari ST, et Cor Bosman et Colin Plumb l'ont porté sur Commodore Amiga.

La traduction de PGP en langues étrangères a été faite par Jean-loup Gailly en France, Armando Ramos en Espagne, Felipe Rodriquez Svensson et Branko Lankester aux Pays-Bas, Miguel Angel Gallardo en Espagne, Hugh Kennedy et Lutz Frank en Allemagne, David Vincenzetti en Italie, Harry Bush et Maris Gabalins en Lettonie, Zygimantas Cepaitis en Lituanie, Peter Suchkow et Andrew Chernov en Russie, et Alexander Smishlajev en Espéranto. Peter Gutmann a offert de le traduire en anglais de Nouvelle-Zélande, mais nous avons finalement décidé que PGP ferait l'affaire en anglais américain.

Jean-Loup Gailly, Mark Adler, et Richard B. Wales ont publié le code [source] de compression ZIP, et accordé la permission de l'inclure dans PGP. Les routines MD5 ont été développées et placées dans le domaine public par Ron Rivest. L'algorithme IDEA™ a été développé par Xuejia Lai et James L. Massey à l'ETH de Zurich, et a été utilisé dans PGP avec la permission de Ascom-Tech AG.

Charlie Merritt m'a appris à l'origine comment faire une horloge arithmétique multiprécision décente pour la cryptographie à clé publique, et Jimmy Upton a contribué à rendre plus rapide l'algorithme [de multiplication modulo un entier]. Thad Smith a réalisé un algorithme [de ce genre] encore plus rapide. Zhahai Stewart a contribué par beaucoup de suggestions pertinentes quant au format des fichiers de PGP et d'autres choses, dont le fait d'avoir plus

d'un nom d'utilisateur par clé. J'ai trouvé l'idée des certificateurs fiables chez Whit Diffie. Kelly Goen a fait le gros du travail pour la publication initiale de PGP 1.0.

Des contributions variées à l'effort d'écriture du code [source] sont aussi venues de Colin Plumb, Derek Atkins, et Castor Fu. Les autres contributions à cet effort, d'écriture du code ou d'autres choses, sont venues de Hugh Miller, Eric Hughes, Tim May, Stephan Neuhaus, et de beaucoup trop d'autres personnes pour que leurs noms me reviennent immédiatement à l'esprit. Zbigniew Fiedorwicz a fait le premier portage pour Macintosh.

Depuis la réalisation de PGP 2.0, beaucoup d'autres programmeurs ont envoyé des correctifs et des corrections de bogues et amélioré le portage pour les autres ordinateurs. Ils sont trop nombreux pour être remerciés individuellement ici.

Exactement comme dans l'histoire de la « Soupe aux cailloux », il devient de plus en plus dur de voir, à travers la soupe épaisse, tout au fond, les cailloux que j'avais lancés dedans pour tout commencer.

A propos de l'auteur

Philip Zimmermann est un ingénieur informaticien consultant avec une expérience de 19 ans, spécialement dans le domaine des systèmes en temps réel, de la cryptographie, de l'authentification, et des communications de données. Cette expérience intègre la conception et la réalisation de systèmes d'authentification pour les réseaux d'informations financières, la sécurité des réseaux de données, les protocoles de gestion de clés, l'intégration d'exécutables multitâches, les systèmes d'exploitation, et les réseaux locaux.

e-mail : prz@acm.org

[Pour toute autre version de PGP, contacter :

[Network Associates, Inc. (USA) http://www.nai.com

[Network Associates BV (Pays-Bas) http://www.pgpinternational.com

[International PGP home page http://www.pgpi.com