GnuPG et chiffrement de fichiers

Franck Jeannot

Montreal, Canada, Février 2017, G192 ; v1.1 Oct. 2017

Abstract

A brief introduction to GPG with a focus on ways to create GPG certificates and crypting emails with some quick overview on history and weaknesses.

Keywords: Cryptography, gpg, PGP, GnuPG, PGP/MIME

1. Introduction

Cet article donne un aperçu sur GPG et détaille des étapes pour son usage, puis fournit des solutions permettant l'usage d'emails *cryptés*[1]. Des références intéressantes sont [2] et [3].
Des éléments sont aussi fournis pour détailler des étapes principales visant à vérifier des signatures de logiciels avec GPG.

1. Les termes chiffrer, crypter, encrypter seront utilisés sans revue rigoriste de sémantique, voulant signifier dans tous les cas de rendre inintelligible un texte pour des personnes non destinataires
2. https://xato.net/getting-started-with-pgp-in-10-minutes-or-less-eebf645df59d
3. https://alexcabal.com/creating-the-perfect-gpg-keypair/

2. Présentation de GPG

GnuPG (ou GPG, de l'anglais GNU Privacy Guard) est un logiciel de **cryptographie**.
C'est l'implémentation GNU du standard OpenPGP défini dans la **RFC 4880**[4][5][6], distribuée selon les termes de la licence publique générale GNU. Il a été initié par Werner Koch en 1997.
GnuPG[7] permet de crypter et signer des données et communications, dispose d'un système polyvalent de gestion de clés, ainsi que des modules d'accès pour tous les types de répertoires de clés publiques.

OpenPGP a été écrit sur la base de **PGP** (Pretty Good Privacy) 5.x développé par Philip R. Zimmermann[8].

Pour le cryptage, PGP et GPG se basent notamment sur l'algorithme **RSA** : « *Encryption using GnuPG is based on the so-called RSA algorithm. RSA is derived from the last names of Ron Rivest, Ami Shamir und Ben Adleman, who discovered this algorithm in 1978. This algorithm uses a type of arithmetic which is called arithmetic with residue classes or "modular (or modulo) arithmetic"* » (Source : [1] p 144).

4. historiquement RFC2440
5. https://tools.ietf.org/html/rfc2440
6. https://www.ietf.org/rfc/rfc4880.txt
7. https://www.gnupg.org
8. https://en.wikipedia.org/wiki/Phil_Zimmermann

3. PGP : l'historique

C'est en 1991 que Philip R. Zimmermann publiait la premiere version de PGP, et en 1994, dans le contexte d'une année noire pour la cryptographie (Clipper...etc)[2] que son usage se popularisait. Il publia finalement en 1995 au MIT Press son « *Pgp : Source Code and Internals* » [3]. C'est en 1997, que Zimmermann proposa au Internet Engineering Task Force (IEFT) le standard appelé OpenPGP. Les apports de Zimmermann lui ont permis d'être élu en 2012 comme *Innovateur* dans le **Internet Hall of Fame** [9]. Une entrevue [4] et rétrospective sur PGP, avec Zimmermann, de 2013, rappelait toute l'utilité des outils de cryptographie de nos jours.

4. PGP : le principe simplifié

Sur la partie gauche on décrit le cryptage de données qui part d'une génération de données aléatoires et se poursuit avec le cryptage des données en utilisant **la clé publique** du destinataire. Sur la partie droite le destinataire décrypte les données avec sa **clée privée**.

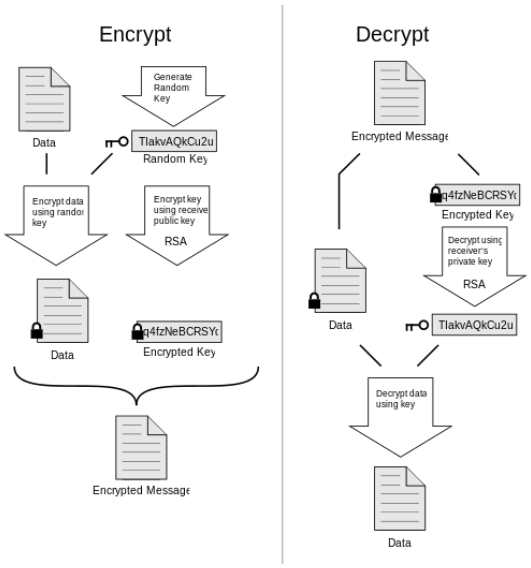

FIGURE *(1): Principe simplifié du chiffrage de données avec PGP. Source : gigaom.com*

9. https://en.wikipedia.org/wiki/Internet_Hall_of_Fame

5. GPG et vulnérabilités (CVE)

Comme tout logiciel, les implémentations de GPG et libgcrypt [5] (librairie basée sur le code de GPG) ne sont pas exempts de vulnérabilités [10] :

Quelques exemples (CVE) :

- **CVE-2016-6313** [6316] (CVSS 5.0) : The mixing functions in the random number generator in Libgcrypt before 1.5.6, 1.6.x before 1.6.6, and 1.7.x before 1.7.3 and GnuPG before 1.4.21 make it easier for attackers to obtain the values of 160 bits by leveraging knowledge of the previous 4640 bits « *Felix Dörre and Vladimir Klebanov from the Karlsruhe Institute of Technology found a bug in the mixing functions of Libgcrypt's random number generator : An attacker who obtains 4640 bits from the RNG can trivially predict the next 160 bits of output. This bug exists since 1998 in all GnuPG and Libgcrypt versions.* » [11].
- **CVE-2013-4351** (CVSSv3 Base 5.3) GnuPG 1.4.x, 2.0.x, and 2.1.x treats a key flags subpacket with all bits cleared (no usage permitted) as if it has all bits set (all usage permitted), which might allow remote attackers to bypass intended cryptographic protection mechanisms by leveraging the subkey [12].

6. GPG et faiblesses diverses

Au delà des vulnérabilités (CVE) il y aussi des ***faiblesses diverses*** directes ou indirectes (criticité à quantifier).
Un exemple, de Juillet 2016, de chercheurs tchèques (Masaryk University), pointait des "leakages" possibles sur les origines de clés RSA laissant entrevoir de nouvelles attaques possibles « *accurate identification of originating library or smartcard*

10. https://www.cvedetails.com/vulnerability-list/vendor_id-4711/Gnupg.html
11. https://lists.gnupg.org/pipermail/gnupg-announce/2016q3/000395.html
12. http ://seclists.org/oss-sec/2013/q3/599

is possible based only on knowledge of the public keys » (p 66)

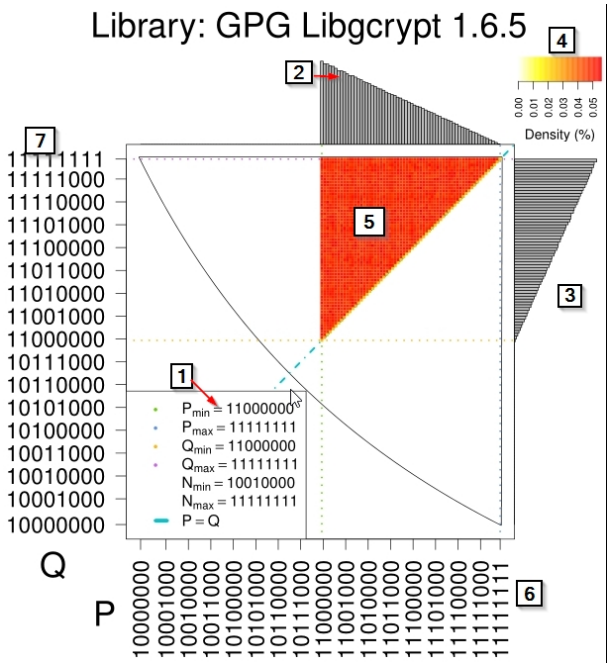

FIGURE *(2): "Libraries : the practical square region" (Source : [6] page fig 4 page 24 ; [7]). [1] légende associant les points de couleur du graphe et leurs valeurs respective P/Q/N [2]distribution marginale de Q ; [5]"The colour scheme expresses the likelihood that primes of a randomly generated key will have specific high order bytes, ranging from white (not likely) over orange to red (more likely)"*

Selon [6] (*commentaire fig. 4 page 24*), l'usage spécifique des nombres premiers, par exemple dans les librairies de cryptographie pour la génération de clés RSA, permet de remonter aux librairies à l'origine de leur génération « *Libgcrypt 1.6.5 (used by GPG) and PGP SDK 4 sort the primes in the opposite order* ($p < q$). *Other libraries do not manipulate with the order of the primes.* »

7. GPG et versions

Selon gpg4win.org, la dernière version disponible en Octobre 2017 est la Version 3.0.0 (released 2017-09-19) avec comme contenu :

— GnuPG 2.2.1
— Kleopatra 3.0.0
— GPA 0.9.9
— GpgOL 2.0.1
— GpgEX 1.0.5

La version testée ici est Gpg4win 2.3.3 (Released : 2016-08-18) :

— GnuPG 2.0.30
— Kleopatra 2.2.0-gitfb4ae3d
— GPA 0.9.9

```
C:\Users\u1> gpg --version
gpg (GnuPG) 2.0.30 (Gpg4win 2.3.3)
libgcrypt 1.6.6
Supported algorithms:
Pubkey: RSA, RSA, RSA, ELG, DSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH,
 AES, AES192, AES256, TWOFISH,
CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: MD5, SHA1, RIPEMD160, SHA256,
 SHA384, SHA512, SHA224
Compression: Uncompressed, ZIP, ZLIB, BZIP2
```

Selon gnupg.org « 2.0.30 is the stable version from an often used branch. This branch will reach end-of-life on 2017-12-31. Project Gpg4win provides a Windows version of the old GnuPG stable. »

La plateforme et l'usage envisagé conduiront à utiliser donc une de ces versions en prenant bien en compte la fin de vie proche annoncée de la branche 2.0.30. L'utilisation d'**ECC (Elliptic curve cryptography**) impose d'utiliser une implémentation Linux sur branche 2.1.18 dite version "moderne" par exemple.

8. Windows : installer GPG et générer une key-pair

Il **n'est pas recommandé d'utiliser Windows** comme plateforme sécurisée, mais à défaut d'autre plateforme, mieux vaut crypter des emails sous Windows que de ne rien faire : « *It is not recommended to use Windows as a secure communication platform. While Windows can be locked down to provide a more secure environment than is provided by default, the tendencies in Windows lean towards very lax security.* » [13]
Les étapes essentielles sont donc :

13. https://riseup.net/ca/security/message-security/openpgp/gpg-keys

1. Télécharger la dernière version de **gpg4win** [14] qui est l'implémentation de GPG pour Windows et l'installer en veillant à sélectionner GPA (GNU Privacy Assistant) qui n'est pas par défaut.
2. Lancer **Kleopatra** qui est un logiciel venant avec le package gpg4win, vous allez pouvoir soit importer des clés existantes [15] ou bien en créer de nouvelles. [16]
3. Sous **Kleopatra**, considérant que vous créez une nouvelle **key-pair** allez dans File -> New Certificate.
4. Dans *Advanced Settings*, paramétrez vos options de choix. Je choisirais du RSA 4096 bits + RSA 4096 bits (subkey, voir [17]) avec capacité de Signature (Signing), d'authentification (Authentication) et une expiration sous 1 an. Vous devez fournir au moins un nom et un email à titre d'identification. Vous devez fournir une passphase de qualité, à la fin du process votre key-pair est générée. Il n'est plus concevable d'utiliser des clés RSA de 2048 bits (et encore moins DSA), revoir [18]. Il est évident que la passphrase ne doit jamais être révélée ou stockée, à défaut utiliser un outil de gestion de mots de passe sécurisé...
5. Il est important de pouvoir sauvegarder les différentes clés en lieu sûr. Idéalement la clé privée doit être stockée (entête BEGIN PGP PRIVATE KEY BLOCK) sur un système soit physiquement protégé, soit « cryptographiquement protégé » sinon cela perd de son intérêt... Sous gpg4win, cochez et utilisez l'option *ASCII Armor* (protection).
6. Il est important de générer **un certificat de révocation** en ligne de commande, permettant, le cas échéant de pouvoir révoquer la key-pair à tout moment avant son expiration. Sous windows la ligne de commande sera (avec mykey à remplacer par le **key-id** de votre **key-pair**) :

```
gpg --output revoke.asc
    --gen-revoke mykey
```

nda : [A formater sur une seule ligne !]

7. Une clé publique doit être accessible aux correspondants. Dans un premier temps il faut exporter la clé publique. En éditant la clé publique on remarquera l'entête précisant : BEGIN PGP PUBLIC KEY BLOCK.
Une solution fournie et par défaut est par exemple : clic droit-> Export Certificates to Server –> Envoi a **keys.gnupg.net** (ou **pgp.mit.edu** ou autre serveur). Le message de succès est "OpenPGP certificates exported successfully". Il est possible aussi d'aller directement sur http://keys.gnupg.net. Autre exemple, en soumettant la clé publique sur https://pgp.mit.edu.
8. Ayez votre clé certifiée par un utilisateur de votre connaissance qui ira chercher votre clé publique sur un serveur public de confiance et ira la certifier (Clic Droit –>certify certificate).
9. Vérifier les paramètres de votre certificat et chiffer le contenu souhaité, le résultat sera une copie du fichier original avec une extension **.gpg**. Sous windows, on fera attention à se débarasser de l'original par un "secure delete", avec **sdelete** [19].
Par exemple (f.txt est le fichier original) :

```
gpg --encrypt --sign -- armor
--recipient <your-key-id>
--recipient <their-key-id> f.tx
```

14. https://www.gpg4win.org/download.html
15. https://www.gpg4win.org/doc/en/gpg4win-compendium_25.html
16. https://www.gpg4win.org/doc/en/gpg4win-compendium_12.html
17. https://wiki.debian.org/Subkeys?action=show&redirect=subkeys
18. https://alexcabal.com/creating-the-perfect-gpg-keypair/
19. https://technet.microsoft.com/en-us/sysinternals/bb897443.aspx

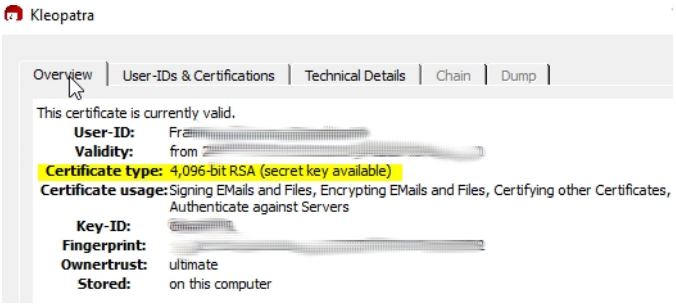

FIGURE *(3): Certificat-Clé GPG et sa clé RSA de 4096 bits*

9. Chiffrer ses emails : normes

Aucune des normes **OpenPGP** ou **PGP/MIME** ne peut crypter les en-têtes de courrier - y compris le sujet de l'email [8].

10. Echanger des emails chiffrés

Si l'on veut recevoir des emails chiffrés il faut que l'émetteur connaisse la clé publique du destinataire, donc au final les deux partenaires ont chacun besoin d'accéder à la clé publique de l'autre [20]. Typiquement avec Kleopatra on va exporter sa clé en format texte (.asc) : File» Export Certificate.

11. Chiffrer ses emails avec Thunderbird et Enigmail

1. Configurer un client lourd pour email, typiquement **Thunderbird**. [21]
2. Télécharger l'addon **Enigmail** (avant de l'installer appliquez la procédure de vérification de signature) [22].

12. Vérifier la signature d'un logiciel

12.1. Windows : GPG et Kleopatra

Dans la majorité des cas, un logiciel, package, binaire, distribution, iso sera accompagné d'un fichier signature (.sig, .asc, .p7s ou .pem). Une première étape de vérification va consister à :

1. Télécharger le logiciel, sa signature et sa clé de signature (certifcat) dans un même répertoire,
2. Importer le certificat dans **Kleopatra** (File » Import certificates),
3. Vérifier le fichier de signature dans Kleopatra (File » Decrypt/Verify files),
4. Vérifier la signature avec la « Toile de confiance OpenPGP » ou « OpenPGP Web of Trust ».

Références

[1] M. J. H. I. K. u. D. F. W. Ute Bahn, Karl Bihlmeier, Gpg4win compendium - secure e-mail and file encryption using gnupg for windows.
URL http://wald.intevation.org/frs/download.php/1385/gpg4win-compendium-en-3.0.0.pdf

[2] S. Garfinkel, PGP: Pretty Good Privacy, Encryption for everyone, O'Reilly Media, Incorporated, 1995.
URL https://books.google.ca/books?id=cSe_0OnZqjAC

[3] P. Zimmermann, Pgp: Source Code and Internals, MIT Press, 1995.
URL https://books.google.ca/books?id=xR4ZAQAAIAAJ

[4] O. Malik, Zimmermann's law: Pgp inventor and silent circle co-founder phil zimmermann on the surveillance society.
URL https://gigaom.com/2013/08/11/zimmermanns-law-pgp-inventor-and-silent-circle-co-founder-phil-zimmermann-on-the-surveillance-society

[5] Libgcrypt.
URL https://www.gnupg.org/(fr)/related_software/libgcrypt/index.html

[6] P. S. et al., The million-key question - investigating the origins of rsa public keys, in : FI MU Report Series, FIMU-RS-2016-03, Masaryk University, 2016.
URL https://crocs.fi.muni.cz/_media/public/papers/usenixsec16_1mrsakeys_trfimu_201603.pdf

[7] Crocs wiki : The million-key question – investigating the origins of rsa public keys.
URL https://crocs.fi.muni.cz/public/papers/usenix2016#datasets_and_tools

[8] Faq enigmail.
URL https://enigmail.wiki/FAQ

20. https://www.gpg4win.org/doc/en/gpg4win-compendium_13.html
21. https://www.mozilla.org/en-US/thunderbird/
22. https://www.enigmail.net/index.php/en/verify-signature