RFC 9580 : OpenPGP

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 1 août 2024. Dernière mise à jour le 4 janvier 2025

Date de publication du RFC : Juillet 2024

https://www.bortzmeyer.org/9580.html

Le logiciel PGP est synonyme de cryptographie pour beaucoup de gens. Un des plus anciens et des plus utilisés pour les fonctions de confidentialité mais aussi d'authentification. Le format de PGP, OpenPGP, est normalisé dans ce RFC, qui remplace le RFC 4880 [1] (il n'y a pas de changement crucial, juste une mise à jour longtemps attendue).

PGP a vu son format de données normalisé pour la première fois en août 1996, dans le RFC 1991. Cette norme a été révisée par la suite, dans le RFC 2440, puis le RFC 4880, puis par des RFC ponctuels (comme les RFC 5581 et RFC 6637, désormais inclus) et notre RFC est la dernière version, qui synthétise tout. Sa gestation a été longue et douloureuse et a suscité des controverses, menant à un projet concurrent, LibrePGP <https://librepgp.org/> (qui prévoit son propre RFC, actuellement draft-koch-librepgp, que j'avoue n'avoir pas lu).

Cette normalisation permet à diverses mises en œuvre de PGP d'interopérer. La plus connue aujourd'hui est le logiciel libre, GNU Privacy Guard (qui n'existait pas encore au moment de la publication du premier RFC) mais il y en a d'autres comme Sequoia <https://sequoia-pgp.org/> (qui a annoncé <https://sequoia-pgp.org/blog/2024/12/23/202412-sequoia-openpgp-2.0.0-alpha.0/> qu'il suivrait notre récent RFC). Il ne faut donc pas confondre le **logiciel** PGP, écrit à l'origine par Phil Zimmermann, et qui est non-libre, avec le **format** OpenPGP que décrit notre RFC (cf. section 1.1) et que des logiciels autres que PGP peuvent lire et écrire.

Le principe du chiffrement avec PGP est simple. Une **clé de session** (le terme est impropre puisqu'il n'y a pas de session au sens de TLS mais c'est celui utilisé par le RFC) est créée pour chaque destinataire, elle sert à chiffrer le message et cette clé est chiffrée avec la clé publique du destinataire (section 2.1 du RFC).

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc4880.txt

Pour l'authentification, c'est aussi simple conceptuellement. Le message est condensé et le condensé est chiffré avec la clé privée de l'émetteur (section 2.2 du RFC).

Le format OpenPGP permet également la compression (qui améliore la sécurité en supprimant les redondances) et l'encodage en Base64 (RFC 4648), baptisé "*ASCII armor*", pour passer à travers des logiciels qui n'aiment pas le binaire (la section 6 détaille cet encodage).

La section 3 explique les éléments de base utilisés par le format PGP. L'un des plus importants est le concept d'entier de grande précision (MPI pour "*Multi-Precision Integers*"), qui permet de représenter des entiers de très grande taille, indispensables à la cryptographie, sous forme d'un doublet longueur + valeur.

Enfin les sections 4 et 5 expliquent le format lui-même. Un message PGP est constitué de **paquets** (rien à voir avec les paquets réseau). Chaque paquet a un type, une longueur et un contenu. Par exemple, un paquet de type 1 est une clé de session chiffrée, un paquet de type 2 une signature, un paquet de type 9 du contenu chiffré, etc.

La section 7 du RFC décrit un type de message un peu particulier, qui n'obéit pas à la syntaxe ci-dessus, les messages en clair mais signés. Ces messages ont l'avantage de pouvoir être lus sans avoir de logiciel PGP. Ils nécessitent donc des règles spéciales.

On notera que gpg permet d'afficher les paquets présents dans un message PGP, ce qui est pratique pour l'apprentissage ou le débogage. Voyons un exemple avec un fichier `test.txt` de 17 octets, signé mais non chiffré (j'ai un peu simplifié la sortie du logiciel) :

```
% gpg --list-packets test.gpg
:compressed packet: algo=2
:onepass_sig packet: keyid 3FA836C996A4A254
 version 3, sigclass 0x00, digest 10, pubkey 1, last=1
:literal data packet:
 mode b (62), created 1721800388, name="test.txt",
 raw data: 17 bytes
:signature packet: algo 1, keyid 3FA836C996A4A254
 version 4, created 1721800388, md5len 0, sigclass 0x00
 digest algo 10, begin of digest 2b d9
 hashed subpkt 33 len 21 (issuer fpr v4 C760CAFC6387B0E8886C823B3FA836C996A4A254)
 hashed subpkt 2 len 4 (sig created 2024-07-24)
 subpkt 16 len 8 (issuer key ID 3FA836C996A4A254)
 data: [4096 bits]
```

Malheureusement, gpg n'affiche pas les valeurs numériques des types, telles que listées par le RFC. Mais les noms qu'il utilise sont les mêmes que dans le RFC, on peut donc facilement trouver la section qui explique ce qu'est un "onepass_sig packet" (section 5.4).

Avec un message chiffré, on obtient :

```
% gpg --list-packets test.txt.gpg
gpg: encrypted with 4096-bit RSA key, ID 9045E02757F02AA1, created 2014-02-09
      "Stéphane Bortzmeyer (Main key) <stephane@bortzmeyer.org>"
gpg: encrypted with 2048-bit RSA key, ID 516CB37B336525BB, created 2009-12-15
      "ISC Security Officer <security-officer@isc.org>"
gpg: decryption failed: No secret key
```

```
:pubkey enc packet: version 3, algo 1, keyid 516CB37B336525BB
 data: [2046 bits]
:pubkey enc packet: version 3, algo 1, keyid 9045E02757F02AA1
 data: [4096 bits]
:encrypted data packet:
 length: 78
 mdc_method: 2
```

Conformément au principe d'agilité cryptographique (RFC 7696), le format OpenPGP n'est pas lié à un algorithme cryptographique particulier, et permet d'en ajouter de nouveaux. La section 15 détaille comment demander à l'IANA d'ajouter de nouveaux paramètres dans les registres PGP <https://www.iana.org/assignments/openpgp/openpgp.xml>.

L'annexe B, quant à elle, énumère les principaux changements depuis le RFC 4880. Ce dernier RFC a été publié il y a plus de seize ans mais son remplacement par notre nouveau RFC 9580 a été une opération longue et difficile, et les changements se sont accumulés (Daniel Huigens en a fait un bon résumé <https://proton.me/blog/openpgp-crypto-refresh>). Ceci dit, le format OpenPGP ne change pas radicalement, et l'interopérabilité avec les anciens programmes est maintenue. Parmi les principales modifications :

— des nouveaux algorithmes cryptographiques de signature comme Ed25519, ou bien ECDSA avec les courbes Brainpool (RFC 5639),
— des nouveaux algorithmes cryptographiques de chiffrement, comme X25519 ; comme pour ceux de signature, ils étaient parfois mentionnés dans le RFC 4880 sans que leur utilisation dans OpenPGP soit complètement spécifiée, et parfois avaient été intégrés via un RFC spécifique, comme le RFC 6637, sur certains algorithmes à courbe elliptique,
— des nouveaux modes pour le chiffrement intègre, comme GCM (nouveaux pour OpenPGP, mais anciens en cryptographie),
— des nouvelles fonctions de dérivation de clé comme Argon2 (RFC 9106) ou de condensation comme SHA-3,
— au contraire, certains algorithmes sont désormais marqués comme dépassés et ne devant plus être utilisés pour de nouveaux messages ; c'est le cas de DSA, ElGamal, MD5, SHA-1…(rappelez-vous que PGP peut avoir à lire des messages anciens et qu'on ne peut donc pas retirer purement et simplement ces algorithmes, d'autant plus qu'autrefois ils étaient parfois les seuls obligatoires pour une mise en œuvre standard d'OpenPGP),
— nouvelle version (version 6) pour plusieurs types de paquets,
— réduction de l'en-tête de la protection ASCII des messages (OpenPGP est du binaire mais peut être transcrit en "*ASCII armor*", par exemple pour le courrier électronique) ; vous ne verrez plus le « Version : » dans cet en-tête,
— redressement de la terminologie, qui était parfois trop floue.

Enfin, un grand nombre d'errata <https://www.rfc-editor.org/errata_search.php?rfc=4880&rec_status=15&presentation=table> ont été traités (la liste complète est dans l'annexe D).

Parmi les mises en œuvre de cette nouvelle version du format, on peut citer rsop <https://crates.io/crates/rsop/> ou rPGP <https://crates.io/crates/pgp>.