GNUPG

- Objet : GnuPG, Fonctionnement, Installation, Utilisation
- Niveau requis :
  débutant, avisé
- Commentaires : *Pour communiquer en toute sécurité, et pour chiffrer vos fichiers*
- Débutant, à savoir : Utiliser GNU/Linux en ligne de commande, tout commence là !
- Suivi :
  à-tester
  - Création par mattux le 05/07/2007
  - Testé par .... le ....
- Commentaires sur le forum : C'est ici[1)]

Introduction

Cet article n'a pas pour vocation d'être une sorte de copie des excellents howtos fournis sur le site officiel GnuPG, mais plutôt d'en être un résumé, un rappel (peut-être un peu plus...) apportant quelques détails supplémentaires pouvant être intéressants.

Avertissement :

Attention, toute les opérations décrites dans cet article sont à faire dans un terminal.

Si vous n'aimez pas le terminal, tant pis pour vous... heu... nan... vous pouvez toujours utiliser une interface du genre kgpg *avec* kmail *ou* thunderbird *avec* enigmail*, mais dans ce cas, ne comptez pas sur moi pour vous décrire toute la procédure.*

Pourquoi chiffrer ses données ?

À l'heure actuelle, internet est tout, sauf un endroit de confiance.
En ayant acquis les connaissances techniques (il suffit de lire quelques documentations) tout le monde peut lire, voire modifier les données envoyée par internet.
Bref, prenez en main votre intimité !

Vue ainsi, il semble nécessaire de protéger **l'authenticité, l'intégrité ainsi que la confidentialité** de nos données.
Pour cela, Philip Zimmermann à créé PGP. Le logiciel que nous allons utiliser s'appelle gnupg, il implémente **OpenPGP** qui est le standard **PGP**.

Nota

Si vous voulez en savoir plus sur Mr. Zimmermann, c'est par là...

Fonctionnement de PGP

Bien sûr, avant de faire joujou, il faut comprendre ce que l'on fait, sinon ça ne sert à rien !

PGP utilise un *système de chiffrage asymétrique*, c'est-à-dire que ce système utilise une paire[2)] de clefs de chiffrage.
Nous verrons dans la partie pratique que nous disposons aussi de plusieurs algorithmes possibles.

Nous disposerons donc de 2 clefs :

1. Une **clef privée** personnelle à ne pas divulguer et même à protéger soigneusement par les droits.
2. Une **clef publique** que l'on indiquera au(x) destinataire(s) du message qui devront comme vous la rendre inabordable au tout venant en en dissimulant l'accès.



Un message chiffré avec la *clef privée* ne sera déchiffrable qu'avec la *clef publique* associée, et vice et versa.

Merci à **Thom1** *pour ses suggestions et améliorations !*

Installation sous Debian

GnuPG est installé dans le système de base de debian, il n'y a donc rien à faire.

Installation sous FreeBSD

Je n'ai pu m'empêcher d'indiquer comment on fait l'installation sous FreeBSD, désolé :-p

```
cd /usr/ports/security/gnupg && make install clean
```

Via les sources

Pour tous les autres systèmes, je vous conseille de passer par le système de packages de votre OS. Sinon vous pouvez toujours tenter une installation via les sources.

Pour plus d'information :

- cette page du manuel de gnupg

Génération d'une clef

Fini le blabla théorique, passons aux choses sérieuses

Génération des clefs

Là on vous demande de choisir un algorithme :

```
gpg --full-gen-key
```

séquence interactive

```
gpg (GnuPG) 1.4.7; Copyright (C) 2006 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

Please select what kind of key you want:
   (1) DSA and Elgamal (default)
   (2) DSA (sign only)
   (5) RSA (sign only)
Your selection?
```

1. Le premier permet de chiffrer et de signer les messages.
2. Les deux autres sont utiles uniquement pour signer.

Pour plus d'information sur ces algorithmes :

- http://fr.wikipedia.org/wiki/RSA|RSA
- http://fr.wikipedia.org/wiki/Digital_Signature_Algorithm|DSA
- http://fr.wikipedia.org/wiki/Cryptosystème_de_ElGamal|ElGamal.

C'est le premier qui nous intéresse car c'est le seul qui permet de chiffrer nos données.

Taille de votre clef.

```
DSA keypair will have 1024 bits.
ELG-E keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048)
```

Plus c'est gros mieux c'est (la mienne est plus grosse que la tienne). :-p
Donc → 4096

Ensuite, on vous demande si vous voulez une date d'expiration pour votre clef ainsi que quelques données personnelles qui figureront sur la clef publique, tel que votre nom, votre mail évidemment... Et surtout on vous demandera une **passphrase**, un mot de passe à ne surtout pas oublier !



Rappel: un bon mot de passe fait plus de 8 caractères, contient des chiffres, des lettres minuscules ET majuscules.

Après, la paire de clefs se génère (Faut patienter quoi)... :P

Et voilà, vous vous trouvez en possession d'une paire de clefs !

Gérer son trousseau de clef

Maintenant qu'on a une paire de clefs, se serait chouette de pouvoir envoyer la clef publique a ses amis et de pouvoir prendre la leur par la même occasion, non ?

Les opérations de base

Extraire la clef publique :

Pour extraire la clef public on fait :

```
gpg --export -a
```

On peut rediriger l'output de *gpg* avec l'option -o :

```
gpg --export -a -o /repertoire/maclef.asc
```

Importer la clef publique

Une fois qu'on a la clef de ses amis, on l'importe ainsi :

```
gpg --import fichier_de_la_clef_du_copain.asc
```

Accéder au trousseau de clefs

Et pour voir le trousseau de clefs :

```
gpg --list-keys
```

Sauvegarder la clef privée



Attention ! Cette clef permet de déchiffrer tous vos fichiers. Il est primordial de la conserver dans un endroit sécurisé et de ne pas la diffuser. Ne confondez pas avec la

clef publique.

En cas de réinstallation, ou pour l'exporter vers un deuxième PC, vous pouvez sauvegarder votre clef privée. La commande suivante va lister les clefs privées installées:

```
gpg --list-secret-keys
```

On obtient un résultat de ce genre:

retour de la commande

```
--------------------------------
sec   2048R/75262C01 2011-04-07
uid                  philippe (yeah!) <k@k.com>
ssb   2048R/527644F3 2011-04-07
```

Il suffit de noter le chiffre après “sec 2048R/” (dans cet exemple, il s'agit de “75262C01”) et de le rentrer dans la commande suivante:

```
gpg --armor --export 75262C01 > clef-privee.key
```

Le fichier apparaît dans votre dossier personnel. Pour l'importer sur un autre PC, il suffit de l'importer comme une clef publique.

Autre solution : copier le contenu du dossier /home/utilisateur/.gnupg du pc1 dans un répertoire identique du pc2. Vérifier sur le pc2 par :

```
gpg --list-keys
```

suivi de :

```
gpg --list-secret-keys
```

Modifier une clef

Si vous voulez modifier des informations sur votre clef ou n'importe laquelle de votre trousseau :

```
gpg --edit-key mon.mail@monfai.org
```

Vous obtiendrez quelque chose comme ça :

retour de la commande

```
pub  1024D/4231DE26  created: 2007-03-31  expires: never       usage:
SC
                     trust: ultimate      validity: ultimate
```

```
sub  4096g/316AD2BB  created: 2007-03-31  expires: never       usage: E
[ultimate] (1). T.Gaudin <t.gaudin@hotmail.com>

Command>
```

Vous pouvez obtenir la liste des commandes avec la commande help.

Exemple d'utilisation

Voici un exemple simple, comment ajouter une adresse à sa clef, puis de l'enlever :

J'allais oublier de préciser :
Ce que **GnuPG** appelle `uid`, c'est simplement une adresse **e-mail**.

```
gpg --edit-key mon.mail@mon.fai
```

retour de la commande

```
gpg (GnuPG) 1.4.7; Copyright (C) 2006 Free Software Foundation, Inc.
This program comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. See the file COPYING for details.

gpg: WARNING: using insecure memory!
gpg: please see http://www.gnupg.org/faq.html for more information
Secret key is available.

pub  1024D/4231DE26  created: 2007-03-31  expires: never       usage:
SC
                     trust: ultimate      validity: ultimate
sub  4096g/316AD2BB  created: 2007-03-31  expires: never       usage: E
[ultimate] (1). Albator <mon.mail@mon.fai>

Command> adduid
Real name: K.Facile
Email address: k.facile@kameleon-facile.org
Comment:
You selected this USER-ID:
    "K.Facile <k.facile@kameleon-facile.org>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? O

You need a passphrase to unlock the secret key for
user: "Albator <mon.mail@mon.fai>"
1024-bit DSA key, ID 4231DE26, created 2007-03-31

pub  1024D/4231DE26  created: 2007-03-31  expires: never       usage:
SC
```

```
                     trust: ultimate      validity: ultimate
sub  4096g/316AD2BB  created: 2007-03-31  expires: never       usage: E
[ultimate] (1)  ALbator <mon.mail@mon.fai>
[ unknown] (2). K.Facile <k.facile@kameleon-facile.org>

Command> uid 2

pub  1024D/4231DE26  created: 2007-03-31  expires: never       usage:
SC
                     trust: ultimate      validity: ultimate
sub  4096g/316AD2BB  created: 2007-03-31  expires: never       usage: E
[ultimate] (1)  Albator <mon.mail@mon.fai>
[ unknown] (2)* K.Facile <k.facile@kameleon-facile.org>

Command> deluid
Really remove this user ID? (y/N) y

pub  1024D/4231DE26  created: 2007-03-31  expires: never       usage:
SC
                     trust: ultimate      validity: ultimate
sub  4096g/316AD2BB  created: 2007-03-31  expires: never       usage: E
[ultimate] (1)  Albator <mon.mail@mon.fai>

Command> save
```

2015-02-26 : gpg –edit-key répond à présent par

```
 gpg>
```

dans l'exemple ci-dessus 4231DE26 :

```
gpg --edit-key 4231DE26
```

retour de la commande

```
gpg (GnuPG) 1.4.12; Copyright (C) 2012 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

La clef secrète est disponible
pub  1024D/4231DE26  created: 2007-03-31  expires: never       usage:
SC
                     trust: ultimate      validity: ultimate
sub  4096g/316AD2BB  created: 2007-03-31  expires: never       usage: E
[ultimate] (1)  Albator <mon.mail@mon.fai>
  gpg>
```

si l'on veut intervenir sur une UID particulière, il faudra indiquer à gpg le numéro qui nous intéresse,

par exemple :

```
gpg> uid 1
```

après validation par ↵ Entrée une astérisque sera placée après (1):

```
[  ultime ] (1)* Albator <mon.mail@mon.fai>
```

par défaut, il n'y a pas de sous-clé (sub) sélectionnée et l'on travaille sur la clé primaire (pub). pour intervenir sur la sous-clé :

```
gpg>key 1
```

déplacera le focus * sur la clé sub :

retour de la commande

```
pub  1024D/4231DE26  created: 2007-03-31  expires: never       usage:
SC
                     trust: ultimate      validity: ultimate
sub*  4096g/316AD2BB  created: 2007-03-31  expires: never       usage:
E
[ultimate] (1)  Albator <mon.mail@mon.fai>
```

modifier la date d'expiration, une fois entré dans le edit-key de la clé concernée :

```
gpg>expire
```

séquence interactive

```
Modification de la date d'expiration de la clef principale.
Veuillez indiquer le temps pendant lequel cette clef devrait être
valable.
         0 = la clef n'expire pas
      <n>  = la clef expire dans n jours
      <n>w = la clef expire dans n semaines
      <n>m = la clef expire dans n mois
      <n>y = la clef expire dans n ans
Pendant combien de temps la clef est-elle valable ? (0) 40y
La clef expire le 2055-02-07
Est-ce correct ? (o/N) o

Une phrase de passe est nécessaire pour déverrouiller la clef secrète
de
l'utilisateur : « Albator <mon.mail@mon.fai> »

clef RSA de 4096 bits, identifiant 4231DE26, créée le 2015-02-17
```

```
pub  4096R/4231DE26  créé : 2015-02-17  expire : 2015-02-17
utilisation : SC
                     confiance : ultime        validité : ultime
sub  4096R/316AD2BB  créé : 2015-02-17  expire : 2015-02-17
utilisation : E
[  ultime ] (1). Albator <mon.mail@mon.fai>
```

NB: si gpg répond :

```
 la clef secrète est nécessaire pour faire cela
```

c'est que la phase d'import de la clé secrète ne s'est pas passée correctement et qu'il faut la refaire 😀

Chiffrer, déchiffrer et signer

Je pense qu'un exemple sera plus clair qu'une liste de commande 😀

Rédiger un message :

Soit, notre message

```
echo "GnuPG rocks!" > msg
```

Pour ceux qui ne comprennent pas cette commande echo : elle écrit simplement “GPG rocks!” dans le fichier msg.

On pourrait très bien faire pareil en utilisant gedit ou kate... Mais j'allais quand même pas ouvrir une interface graphique pour si peu 😀

Chiffrer et signer le message :

On le chiffre et signe le message ainsi :

```
gpg --encrypt --sign -a -r destinaire@mail.org msg
```

GnuPG vous demandera simplement votre passphrase.

Et après, on remarque qu'on a un *zouli* msg.asc qui doit ressembler un peu à ça : 😀

```
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1.4.7 (FreeBSD)

hQIOA/m11uzGat4pEAgAsyUaQomOlSFw7VMpVY+R0PEOow+fLwqPDQTyrNrYjmy9
LBN4jfH5ObzlQZ50Q3VYJ49BZqhAHgLOWs1niWr/Q5ZHKw1tsS9lT4ThyHFjfbF2
```

```
vACcCrxL/uURppxAfmtcrWamNIGvjf0Co5TJ6cfhk0RyT9uUry3ZjnGIQkObeoAD
jhu+Vk45wn43y/EMnpyUE8oHmdS2pW0h6e4aMAIN2jwyFUzG/9CU8VFnspA+dPES
hey8Qro+WR4SEJWuW3y+6DDeka75Z71uqYeLHYOdii2rZ1h/asHCpi8ixal1jFVI
uHoDwXwXgdH+o6C7LJ0LqXlLE5sapjDJAfs1F+35EQf/YRnPUA45FuircCP4M9sa
Ms7gduOyROhQKvyCSDsrq8F1lXHhvo6fzXVNRcTZIlrxisXBDoif+szzTZmXkkcB
3uMvda435JVMoVkq4tmiIomvb7uNMNd24uFB3JJAuWuS8i7ITmdVF5VAxyNCESf/
zR01H7KaxFPeXikhS96RbrJ+sJJWs0QB0kznJgILDTOTqxoeT+Ch4WzFuPVB2a4t
tn+ivsdnIo4F2G1br7bOrhQVt/EtLNYddQkbZYcNmOT8ffeGtnU+PzJeHYFMh+zH
JQnh3GTukwpACzXbFbbqvqLcu8idQ64pqn3nFU4PGoYolIk4SHZtb6uYKeuUxWqt
gNKVAeXcafEMW3yk93me6dnfwNfIfnoKxGPA1uIzGxzV1gv6yRv1K3Jw4wRw8TC2
yUpPbjtzFw1QBmbrRZncXxSy+I+FyzcS2Lz2Fjbk+v6TLRVOSvdyHyz9AOjwSYdq
YRsK6jr+1St7pPCvgVCMscx8omayjsP5whMdVE/ZQLc0ZZBcVl6bpJf8YkERSEkD
tacrU3kXEMo=
=aahZ
-----END PGP MESSAGE-----
```

On comprend plus rien hein ?
C'est le but justement 😃

Vous n'avez plus qu'à l'envoyer à votre destinataire : destinataire@mail.org

Réception d'un message :

Maintenant imaginons que notre cher destinataire@mail.org ait répondu.

On place son message dans le fichier *msg*

```
gpg -d msg
```

retour de la commande

```
gpg: WARNING: using insecure memory!
gpg: please see http://www.gnupg.org/faq.html for more information

You need a passphrase to unlock the secret key for
user: "Albator <mon.mail@mon.fai>"
4096-bit ELG-E key, ID 316AD2BB, created 2007-03-31 (main key ID
4231DE26)

gpg: encrypted with 4096-bit ELG-E key, ID 316AD2BB, created 2007-03-31
      "Albator <mon.mail@mon.fai>"
Tu l'as dit, bouffi ! ^^
```

Le message apparait à la fin :

```
"Tu l'as dit, bouffi ! ^^ "
```

Utiliser un serveur de clef

Révoquer une clef

[prévu pour après bientot :-p]

Conclusion

[prévu pour après après bientôt]

Links

- ...Chiffrement
- ...Clé_de_chiffrement
- ...RSA
- ...Digital_Signature_Algorithm
- ...Cryptosystème_de_ElGamal
- interface GNOME pour GnuPG
- Enigmail ;: Chiffrement GPG avec Icedove

- [[http://biblioweb.samizdat.net/article.php3?id_article=4|En savoir plus sur le créateur de PGP]] - **Lien Obsolète**

1)
N'hésitez pas à y faire part de vos remarques, succès, améliorations ou échecs !
2)
c'est à dire 2 clefs couplées

From:
http://debian-facile.org/ - **Documentation - Wiki**

Permanent link:
http://debian-facile.org/doc:systeme:gnupg

Last update: **11/08/2018 14:11**