GnuPG

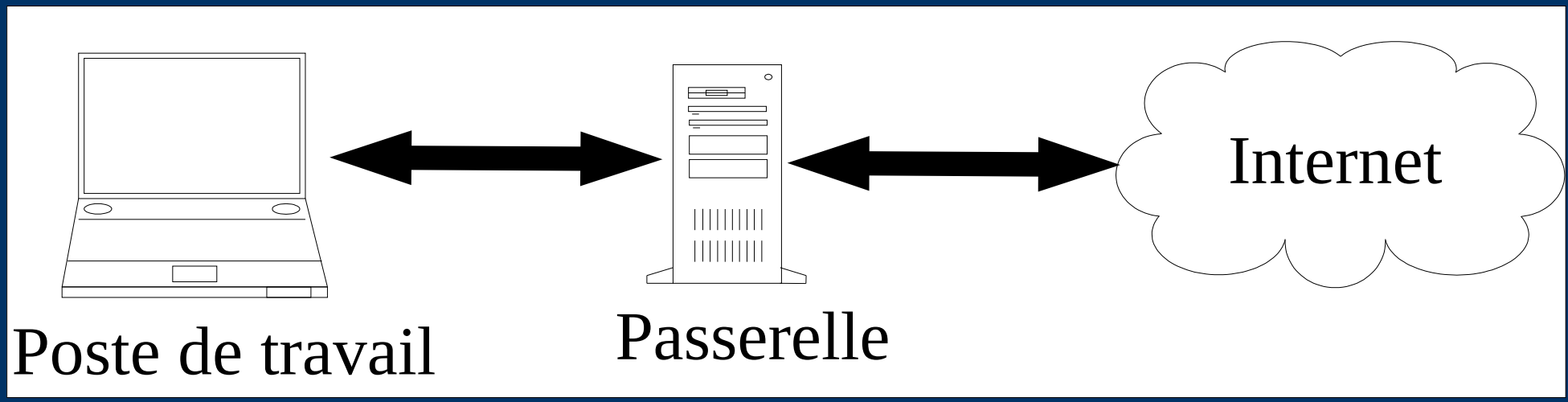
Chiffrement et signature du courrier électronique

Jeudi 21 Octobre - Thomas Petazzoni

Pourquoi chiffrer ?

Envoi de courrier électronique

- Protocole **SMTP**
  Simple Mail Transport Protocol
- Port 25
- RFC 821
- Transit des **données en clair !**

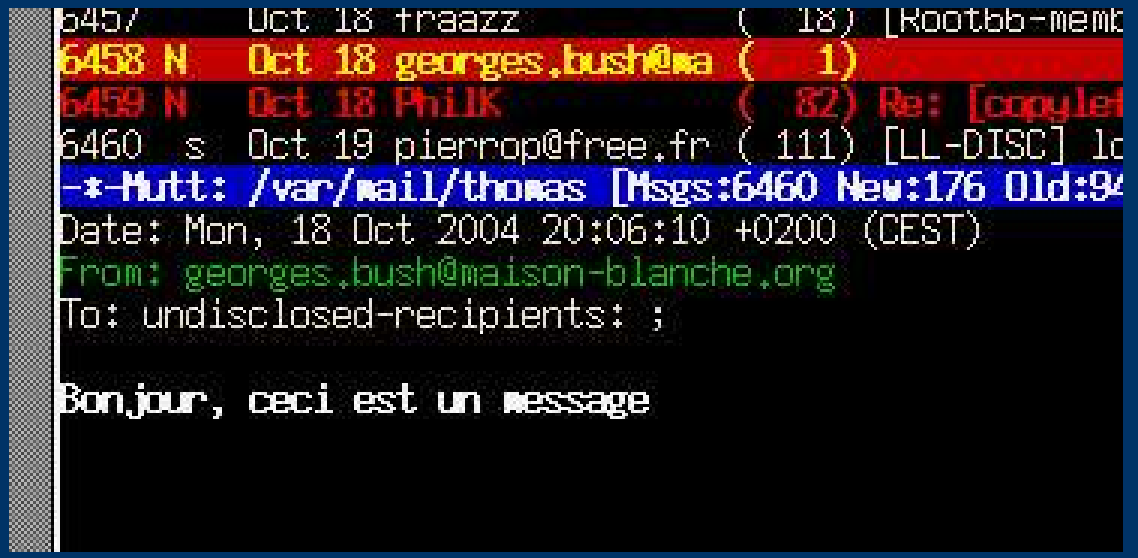


Pourquoi chiffrer ? (2/2)

[...]
T 192.168.1.2:32801 -> 213.228.0.169:25 [AP]
  MAIL FROM:<gnupgtest1@free.fr> SIZE=487..
[...]
T 192.168.1.2:32801 -> 213.228.0.169:25 [AP]
  RCPT TO:<thomas.petazzoni@enix.org>..
[...]
T 192.168.1.2:32801 -> 213.228.0.169:25 [AP]
  DATA..
[...]
T 192.168.1.2:32801 -> 213.228.0.169:25 [AP]
  Message-ID: <417401D7.3080302@free.fr>..Date: Mon, 18 Oct 2004 19:48:07 +0200..From: GnuPG Test 1 <gnupgtest1@free.fr>..User-Agent: Mozilla Thunderbird 0.8 (X11/20040926)..X-Accept-Language: en-us, en..MIME-Version: 1.0..To: thomas.petazzoni@enix.org..**Subject: Nouvelles !**..X-Enigmail-Version: 0.86.1.0..X-Enigmail-Supports: pgp-inline, pgp-mime..Content-Type: text/plain; charset=ISO-8859-1; format=flowed..Content-Transfer-Encoding: 8bit....**Salut !....Je suis . GULLIVER !.... Thomas.....**
[...]
T 192.168.1.2:32801 -> 213.228.0.169:25 [AP]
  QUIT..
T 213.228.0.169:25 -> 192.168.1.2:32801 [AP]
  221 Bye..

Pourquoi signer ?

```
thomas2@crazy:~$ telnet smtp.free.fr 25
Trying 213.228.0.62...
Connected to postfix4-1.free.fr.
Escape character is '^]'.
220 postfix4-1.free.fr ESMTP Postfix
EHLO pouet.net
250-postfix4-1.free.fr
250-PIPELINING
250-SIZE 100000000
250-VRFY
250-ETRN
250 8BITMIME
MAIL FROM:<georges.bush@maison-blanche.org>
250 Ok
RCPT TO:<thomas.petazzoni@enix.org>
250 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
Bonjour, ceci est un message
.
250 Ok: queued as 4498C1F3234
QUIT
221 Bye
```



Cryptographie symétrique

- Une solution simple : utilisation d'un secret partagé
- Secret = algorithme ou mot de passe
- Opération entre la clé et le texte pour générer le texte chiffré : ROT13, XOR, DES, 3DES, AES, Blowfish, etc...
  - En ROT13, espace des clés = 26
  - En AES, 2^128 clés

  «L'âge de l'univers étant de 1010 années, si on suppose qu'il est possible de tester 1000 milliards de clefs par seconde (soit 3.2 1019 clefs par an) il faudra encore plus d'un milliard de fois l'âge de l'univers »

Problème : comment faire si les deux parties ne peuvent se contacter physiquement ?

Cryptographie asymétrique

Basée sur l'utilisation de fonctions à **sens unique**

Principe : Diffie et Hellman

Implémentation : RSA : Rivest, Samir et Adelman

Chaque partie possède deux clés :

- une clé publique
- une clé privée

Cryptographie asymétrique
Chiffrement

Alphonse chiffre son message avec la clé **publique** de Bob

Bob déchiffre son message avec sa clé **privée**

Cryptographie asymétrique
Signature

Alphonse signe son message avec sa clé **privée**

Bob vérifie la signature du message avec la clé **publique** d'Alphonse

En pratique ... générer sa clé

Création d'une clé :

```
$ gpg --gen-key
```

Générer un certificat de révocation :

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

Lister les clés du trousseau :

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

En pratique ... identifier une clé

```
thomas2@crazy:~$ gpg --list-keys --fingerprint
pub   1024D/27AF8427 2004-10-18 GnuPG Test 1
      (GnuPG Test 1) <gnupgtest1@free.fr>
 Empreinte de la clé = 856D 549D 29E6 1D3A 5BB6
                       4AAC 5B8A 3D98 27AF 8427
sub   1024g/3E4DEF09 2004-10-18
```

ID de la clé : 27AF8427

Empreinte ou fingerprint : 856D 549D 29E6 1D3A 5BB6
4AAC 5B8A 3D98 27AF 8427

En pratique ... export et import de clés

Exporter sa clé :

```
$ gpg --output moi.gpg --armor
      --export user@domain.org
```

Envoi du fichier *moi.gpg* aux correspondants.

Importer une clé :

```
$ gpg --import toto.gpg
```

Échange des clés peu pratique : il faut échanger des fichiers

En pratique ... crypter et décrypter

Pour crypter un fichier :

```
$ gpg --output doc.gpg --encrypt
      --recipient user@domain.org
      doc
```

Pour décrypter un fichier :

```
$ gpg --output doc
      --decrypt doc.gpg
```

En pratique ... signer et vérifier

Pour signer un fichier :

```
$ gpg --output doc.sig
      --detach-sig doc
```

Pour vérifier une signature :

```
$ gpg --verify doc.sig doc
```

En pratique ... serveurs de clés

Pour faciliter l'échange de clés, il existe des serveurs, par exemple *pgp.mit.edu*

Envoyer sa clé sur un serveur :

```
$ gpg --keyserver pgp.mit.edu
      --send-key moi@moi.org
```

Importer une clé depuis un serveur :

```
$ gpg --keyserver pgp.mit.edu
      --search-key user@domain.org
```

En pratique ... utilisation

Chiffrement et **signature** automatiques avec de nombreux clients mails :

- ThunderBird
- Sylpheed
- Kmail
- Evolution
- Mutt
- Gnus

Signature : une preuve formelle ?

- Signature prévue pour identifier l'expéditeur.
- Pourtant, il est possible de créer un couple clé privée/publique avec le nom d'une autre personne !
- **Solution : réseau de confiance**

Signature : réseau de confiance

Les utilisateurs signent les clés d'autres utilisateurs dans lesquels ils ont confiance.

« Si vous avez confiance en *Alphonse*, et que *Alphonse* a confiance en *Robert*, alors vous pouvez avoir confiance en *Robert* »

Solution reposant sur le facteur humain, elle est donc faillible.

Autre solution : tierce partie.

Signer des clés

1. Récupérer la clé dans son trousseau
2. Vérifier l'empreinte ou *fingerprint*
3. Signer la clé
4. Envoyer la signature sur le serveur

```
$ gpg –edit-key toto@toto.com
   > fpr
   > sign
   > quit

$ gpg --keyserver pgp.mit.edu
      --send-key toto@toto.com
```

Mettre à jour son trousseau

Pour récupérer les nouvelles signatures :

```
$ gpg --keyserver pgp.mit.edu
      --refresh-keys
```

Références

http://www.gnupg.org

http://www.gnupg.org/gph/fr/manual.html

http://fr.wikipedia.org