TP 7 : GPG

1. GPG et le chiffrement symétrique

1.1. Vérifications

Le but de ce TP[1] est de vous familiariser avec la notion de signature électronique et de chiffrement clé publique / clé privée en utilisant le logiciel GPG. GPG est une version libre du logiciel PGP (Pretty Good Privacy) crée par Philip Zimmermann. Bien qu'il existe des clients graphique (GPA par exemple), nous allons utilisez l'outil le plus basique (mais aussi le plus puissant) : le programme gpg en mode commande.

Dans un terminal vérifiez que le logiciel est bien installé avec la commande

```
$ gpg --version
```

Output :

```
gpg (GnuPG) 2.2.27
libgcrypt 1.9.4
Copyright (C) 2021 Free Software Foundation, Inc.
License GNU GPL-3.0-or-later <https://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Home: /home/user/.gnupg
Supported algorithms:
Pubkey: RSA, ELG, DSA, ECDH, ECDSA, EDDSA
Cipher: IDEA, 3DES, CAST5, BLOWFISH, AES, AES192, AES256, TWOFISH,
      CAMELLIA128, CAMELLIA192, CAMELLIA256
Hash: SHA1, RIPEMD160, SHA256, SHA384, SHA512, SHA224
Compression: Uncompressed, ZIP, ZLIB, BZIP2
```

1.2. Chiffrement

Si vous partagez une clé secrète avec votre destinataire, vous pouvez utiliser GPG pour faire de la cryptographie symétrique. Pour crypter le fichier contenant du texte nommé "**mon_message.txt**", il faut utiliser la commande suivante

```
$ gpg --symmetric --armour mon_message.txt
```

Choisissez une « passphrase » sûre et dont vous vous rappellerez... Elle vous servira à chaque fois que vous aurez à utiliser votre clé privée.

Output généré :

[1] Source : https://clementbjornson.wordpress.com/wp-content/uploads/2013/04/tp-gpg.pdf

```
gpg: directory '/home/user/.gnupg' created
gpg: keybox '/home/user/.gnupg/pubring.kbx' created
```

- L'utilisation de l'option --**armour** permet de convertir l'output chiffré (qui serait normalement binaire) en texte ASCII en utilisant une conversion base64. Cela rend le fichier chiffré plus facile à partager dans des emails ou d'autres environnements qui pourraient ne pas gérer correctement les données binaires. Le texte résultant commence par -----BEGIN PGP MESSAGE----- et se termine par -----END PGP MESSAGE-----.
- **gpg: directory '/home/user/.gnupg' created** : GnuPG crée un nouveau répertoire .gnupg dans le dossier personnel de l'utilisateur (/home/user dans cet exemple). Ce répertoire est utilisé par GnuPG pour stocker ses fichiers de configuration, les clés privées et publiques de l'utilisateur, ainsi que d'autres données opérationnelles nécessaires à son fonctionnement.
- **gpg: keybox '/home/user/.gnupg/pubring.kbx' created** : Au sein de ce répertoire .gnupg, GnuPG crée un fichier nommé **pubring.kbx**, qui est une base de données de clés (keybox) où GnuPG stocke les clés publiques. Même pour un chiffrement symétrique où une clé publique n'est pas nécessairement utilisée pour chiffrer les données, GnuPG initialise cet environnement pour gérer toutes les opérations cryptographiques de manière cohérente.

Le fichier suivant " **mon_message.txt.asc**" sera créé contenant le fichier chiffré. Cette méthode est préférée quand vous voulez par exemple envoyer le fichier par email. Vous pouvez également créer un fichier crypté binaire. La commande correspondante est

```
$ gpg --symmetric mon_message.txt
```

Ceci créera un fichier binaire " **mon_message.txt.gpg**" contenant le fichier binaire chiffré.

Utilisez cette commande pour créer un fichier crypté. Vérifier qu'en modifiant un tout petit peu la clé ou le contenu du fichier, le contenu du fichier crypté change énormément. Vous pouvez pour ceci utiliser un gros fichier texte et regarder ce qui se passe dans le fichier crypté en l'ouvrant avec un éditeur de texte (gedit, vi ou autre).

1.3. Déchiffrement

Pour décrypter, il suffit d'utiliser :

```
$ gpg --decrypt mon_message.txt.asc
```

(Bien entendu, "mon_message.txt.asc" sera remplacé par "mon_message.txt.gpg" si le fichier a été crypté en binaire.)

1. Envoyez un fichier crypté à votre voisin , en mettant en copie l'adresse e-mail de votre professeur et en utilisant « Fichier crypté » comme objet de l'e-mail.
2. Essayez de décrypter le fichier en utilisant une clé erronée (une seule lettre de différence), que se passe-t-il ?
3. À votre avis, que peut-on faire pour déchiffrer le fichier si on a perdu la clé?

Déchiffrez le fichier qui a été crypté dans la question précédente et vérifiez que vous

obtenez le message correct. Échangez la clé de manière verbale. Le destinataire doit ensuite répondre au courriel précédent en joignant le message déchiffré et la clé correcte, préalablement communiquée par l'émetteur. Assurez-vous également de mettre votre professeur en copie du courriel.

2. Gestion des clés publique / privée

2.1. Création des clés

Les clés sont stockées dans un répertoire caché de votre répertoire personnel : **.gnupg**. Vous êtes la seule personne à avoir accès à ce fichier. De plus, vos clés sont protégées par un mot de passe pour renforcer la sécurité. Pour créer votre propre clé publique/clé privée, il faut utiliser la commande

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

- Créez vos clés en acceptant les choix par défaut, sauf pour la durée de validité de vos clés (30 jours)
- Pour notre TP, veuillez saisir uniquement **les initiales** de votre prénom et de votre nom, plutôt que vos noms complets, et indiquez « ENSET Training» dans le champ commentaire. Optez pour une **adresse email fictive** pour le test, par exemple **NUMORDREtest2025@SDIA.ma (ou NUMORDREtest2025@ICCN**.ma selon votre filiale**)**. Toutefois, dans une situation réelle, il est crucial de renseigner votre nom complet et une adresse email régulièrement consultée. Notez que la clé créée sera partagée sur un serveur de clés public, il est donc préférable d'utiliser des informations fictives pour ce TP.
- Choisissez une « passphrase » sûre et dont vous vous rappellerez... Elle vous servira à chaque fois que vous aurez à utiliser votre clé privée.

Lors de l'utilisation de la commande **gpg –full-gen-key** pour générer une nouvelle paire de clés, composée d'une clé publique et d'une clé secrète, un processus d'auto-signature est automatiquement mis en œuvre. En effet, la clé secrète signe la clé publique, attestant ainsi de l'authenticité de cette dernière par une signature auto-générée. Cette procédure d'auto-signature joue un rôle crucial au sein du modèle de confiance appelé *Web of Trust*, propre à GnuPG. Au sein de ce système, il est possible pour les utilisateurs de signer mutuellement leurs clés publiques une fois l'authenticité vérifiée, facilitant de ce fait la création d'un réseau fiable. La signature que vous appliquez sur votre propre clé marque ainsi votre entrée et votre participation active au Web of Trust.

Pour vérifier que les clés ont bien été créées, utilisez la commande

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

Output généré :

```
/home/user/.gnupg/pubring.kbx
```

```
------------------------------
pub   rsa3072/FF3FF5FE7884B4D3 2024-03-03 [SC] [expires: 2026-03-03]
      59CB8EFB55E865BE891940C419C1C7EE01057FC9
uid           [ultimate] monNom monPrenom <mon_email@gmail.com>
sub   rsa3072 2024-03-03 [E] [expires: 2026-03-03]
```

Cet output vous indique que vous avez une clé principale (ligne "pub") qui expire le 03 mars 2026 ; et une sous-clé (ligne "sub") qui expire aussi le 03 mars. La ligne "uid" vous donne l'identité de l'utilisateur correspondant. La clé principale est utilisée pour les signatures, et la sous-clé pour le chiffrement.

- Créez votre propre clé et vérifiez son existence via la commande gpg -k

ATTENTION : Il est impératif de **garder** votre clé secrète **confidentielle**. Dans le cas où une tierce personne obtiendrait l'accès à celle-ci, elle pourrait se faire passer pour vous et déchiffrer des messages qui vous sont destinés. La sécurisation de votre clé secrète repose entièrement sur votre passphrase, qui constitue votre unique ligne de défense si votre compte est compromis. Il est donc essentiel de choisir une passphrase robuste et de ne la partager avec personne. Cette précaution prend encore plus d'importance si votre clé publique a été partagée.

2.2. Certificat de révocation

Lors de la création d'une clé importante, il est impératif de créer un *certificat de révocation*. C'est ceci qui vous permettra de faire savoir que votre clé ne doit plus être utilisée... Un tel certificat pourra servir dans le cas où vous perdez votre clé privée, ou bien vous avez perdu votre passphrase, ou on vous a volé votre ordinateur...

Pour créer un certificat de révocation, il faut faire :

```
$ gpg --output revoke.txt --gen-revoke uid
```

où "uid" est l'identité de la clé concernée.

Cette commande génèrera un fichier : revoke.txt

Attention : ce fichier permet de supprimer votre clé... Générez un certificat de révocation de votre clé en exécutant la commande précédente et suivez les étapes.

Question 10 : D'après le message final de l'output généré, comment vous devez réagir avec la clé de révocation générée ?

Etape Informative à ne pas appliquer :

Si vous devez révoquer votre clé parce qu'elle a été compromise, perdue, ou si vous avez oublié votre passphrase, voici comment procéder :

- **Importer la clé de révocation** : Pour révoquer votre clé, vous devez importer la clé de

révocation dans votre trousseau GPG : gpg --import revoke.asc . Cette action révoquera la clé dans votre trousseau local et il sera tagué comme [revoked] quand vous essayez de lister vos clés avec la commande gpg --list-keys

- **Diffuser la clé révoquée :** Après avoir révoqué votre clé localement, il est important de s'assurer que la révocation soit connue de tous ceux qui utilisent votre clé. Pour cela, vous devez diffuser la clé révoquée. Si vous avez publié votre clé sur un serveur de clés, vous devez y téléverser la version révoquée de votre clé : gpg --send-keys mon_identifiant_clé . Cette commande envoie la clé révoquée vers les serveurs de clés configurés dans GPG, permettant ainsi à d'autres personnes de voir que la clé a été révoquée.

2.3. Partage des clés

2.3.1. À la main

Pour envoyer votre clé publique à quelqu'un, vous pouvez commencer par l'exporter avec la commande :

```
$ gpg --output cle.asc --export --armour uid
```

Où "**uid**" est l'identité (l'adresse email par exemple) de la clé concernée et "**cle.asc**" le nom du fichier qui contiendra la clé en ASCII.

- Lorsque vous générez plusieurs clés GPG avec la même adresse email mais des noms différents, GPG les traite comme des identités distinctes associées à la même adresse email. Cependant, lors de l'exportation d'une clé sans spécifier explicitement laquelle (en utilisant l'adresse email comme identifiant), GPG pourrait choisir la première clé correspondant à cette adresse dans sa base de données interne ou agir selon son algorithme de sélection par défaut, qui peut varier selon la version de GPG et la configuration spécifique de votre environnement.
  Pour comprendre à laquelle des clés l'output fait référence (la première ou la dernière clé générée ou autre), vous devriez examiner les détails de l'exportation. Si vous n'avez pas spécifié un identifiant unique (comme l'empreinte digitale de la clé qui est unique à chaque clé), GPG peut ne pas agir de manière intuitive.
  Pour vous assurer de manipuler la clé souhaitée, il est recommandé d'utiliser l'empreinte digitale (fingerprint) de la clé pour l'exportation, car elle est unique pour chaque paire de clés. Vous pouvez lister toutes vos clés et leurs empreintes digitales avec la commande : gpg --list-keys **--with-fingerprint**
  Ensuite, pour exporter une clé spécifique, utilisez son empreinte digitale : gpg --export -a "empreinte_digitale" > ma_cle_publique.asc

Si vous voulez exporter la clé en binaire, il faut utiliser :

```
$ gpg --output cle.gpg --export uid
```

À l'inverse, pour importer une clé (en binaire ou en ASCII) contenue dans le fichier "**cle.asc**", il suffit d'utiliser la commande :

```
$ gpg --import  cle_voisin.asc
```

1. Échangez vos clés avec votre voisin (par e-mail ayant pour objet «Partage des clés » comme objet, tout en mettant votre professeur en copie) en exportant la vôtre et important la sienne.
2. Vérifiez qu'une nouvelle clé apparaît dans la liste affichée par "$ gpg --list-keys".

2.3.2. Avec un annuaire

Pour partager les clés à grande échelle, on utilise plutôt un *serveur de clés*. Il existe de nombreux serveurs publics, comme par exemple celui du Ubuntu ou MIT. Ces serveurs de clés sont tous interconnectés, et il est pour cette raison impossible de pirater ces annuaires... (Il faudrait pour ceci arriver à tous les pirater en même temps !)

```
$ gpg --keyserver nom_du_serveur --send-key 0xnnnnnnnn
```

où **nnnnnnnn** est le numéro de votre clé, càd les chiffres apparaissant après le *rsa3072* sur la première ligne de votre entrée lors de la commande gpg --list-keys --keyid-format LONG . (*FF3FF5FE7884B4D3* dans mon cas)

Exemple :

```
$ gpg --keyserver https://keyserver.ubuntu.com/ --send-keys FF3FF5FE7884B4D3
```

- Consulter https://keyserver.ubuntu.com/ et vérifier l'existence de votre clé, vous pouvez lancer une recherche en spécifiant l'adresse mail, nom ou prénom fictif utilisé lors de la génération de votre clé, …..

Vous pouvez également télécharger des clés des autres personnes (physique ou morale) depuis le serveur avec :

```
$ gpg --keyserver https://keyserver.ubuntu.com/ --recv-key 0xnnnnnnnn
```

Exemple :

```
$ gpg --keyserver hkps://keys.openpgp.org --recv-keys E53D989A9E2D47BF
```

2.3.3. Vérifier et contre-signer une clé :

Chaque clé possède une « empreinte digitale ». Quand vous récupérez une clé, il est important de vérifier cette empreinte... Cette empreinte est suffisamment petite pour être facilement transmissible (carte de visite etc.)

Par exemple : 59CB 8EFB 55E8 65BE 8919 40C4 19C1 C7EE 0205 7FC9

Pour trouver l'empreinte d'une clé, vous pouvez utiliser $ gpg –fingerprint qui listera toutes les clés connues avec leur empreinte. Si vous mettez une chaîne de caractères à la fin de la commande, cela ne listera que les clés qui contiennent la chaîne en question. (Pratique quand vous avez beaucoup de clés.)

Une fois que vous avez vérifié une clé, vous pouvez l'authentifier pour dire « je fais confiance à cette clé... » On parle de *contre-signature*. La commande est simplement :

```
$ gpg --sign-key uid
```

où "**uid**" est l'identité de la clé à authentifier.

Exemple :

```
gpg --default-key FF3FF5FE7884B4D3 --sign-key E53D989A9E2D47BF
```

Vérifiez l'empreinte des clés que vous avez récupérées depuis le serveur de clé ubuntu, et authentifiez-les.

3.Signature Numérique

3.1. Signature

Maintenant que vous avez des clés, vous pouvez signer des messages. Pour cela, il faut utiliser la commande ci-dessous pour signer le fichier " **new_message.txt** ".

```
$ gpg --clearsign new_message.txt
```

Ceci créera un nouveau fichier "**new_message.txt**.**asc**" qui contiendra le fichier original avec une signature vous authentifiant.

La commande suivante permet de créer uniquement une signature (binaire) pour le fichier en question.

```
$ gpg --detach-sign new_message.txt
```

Cette signature (fichier " **new_message.txt**.**sig**") devra être envoyé avec le fichier original.

Pour un fichier texte, la première méthode est préférable. (Sauf si c'est un email et que votre logiciel gère les signatures en pièce jointe...) Pour un fichier binaire, il faut mieux utiliser la seconde méthode.

3.2. Vérification d'un fichier signé

Pour vérifier un fichier signé, on utilise

```
$ gpg --verify new_message.txt.asc
```

Ou bien

```
$ gpg --verify new_message.txt.sig new_message.txt
```

Il faut bien entendu pour cela disposer de la clé publique de la personne qui a signé le document.

1. Envoyez à votre voisin un document signé de votre part (en mettant, bien évidemment, le mail de votre professeur en copie avec « Vérification signature numérique » comme objet dudit mail)
2. Vérifiez la signature d'un document que votre voisin vous a envoyé via email et répondez au précédent mail pour informer votre voisin du résultat de votre vérification, n'oublier pas de mettre le mail de votre professeur en copie.
3. Vérifiez la signature d'un document qui a été modifié après signature. Que se passe- t-il ?

4.Chiffrement des fichiers signés

4.1. Chiffrement

Le principe est le même que pour la signature : on utilise

```
$ gpg --encrypt --armour fichier
```

pour obtenir un fichier ASCII "fichier.asc" contenant le fichier original crypté. GPG nous demandera les destinataires, à choisir parmi les gens dont on possède les clés publiques.

Si on veut obtenir un fichier binaire, la commande devient :

```
$ gpg --encrypt fichier
```

pour obtenir un fichier "fichier.gpg" contenant le fichier original crypté.

4.2. Déchiffrement

Toujours pareil : pour décrypter, on utilise

```
$ gpg --decrypt fichier
```

Testez le processus de chiffrement et de déchiffrement avec votre voisin en lui envoyant un fichier signé et chiffré. Utilisez l'e-mail avec votre professeur en copie, et indiquez comme objet du message : « Chiffrement d'un fichier signé ». Votre voisin devra ensuite déchiffrer le fichier, récupérer le message en clair, et vous le renvoyer par mail en répondant au message initial. Enfin, vous devrez confirmer si le contenu reçu correspond ou non à votre message initial en clair.

5. GPG en graphique sous Windows/Linux/MacOS

GPG est à la base un programme qui s'utilise dans une console, ce qui rebute les moins puristes. Mais il existe une multitude d'interfaces graphiques et de plugins utilisant GPG, peu importe que vous utilisiez **Outlook, Thunderbird ou Gmail** en direct, vous trouverez toujours le petit soft GPG qui vous permettra de chiffrer vos emails.

Listez les outils ou extensions disponibles en interface graphique, qu'ils soient inclus par défaut (dans un logiciel de messagerie client, par exemple) ou qu'ils nécessitent une installation supplémentaire. Ensuite, tentez d'utiliser l'un d'entre eux pour chiffrer et déchiffrer des messages, ainsi que pour signer et vérifier la signature de fichiers.

Listez brièvement, dans un compte-rendu avec vos propres captures d'écran de l'outil/extension choisi, les différentes étapes suivies (si votre outil les propose bien sûr) pour :

- Créer une paire de clés privée/publique
- Importer des clés
- Chiffrer un message avec la clé publique du destinataire
- Utiliser votre clé privée pour déchiffrer un message reçu (que votre voisin a précédemment chiffré avec votre clé publique)
- Signer un fichier
- Vérifier la signature d'un fichier précédemment signé par votre voisin