# Téléchargement et vérification d'une image iso

## Téléchargement

Supposons qu'on veuille installer une Debian sur une machine (physique ou virtuelle, par exemple virtualbox). On va sur le site `https://www.debian.org/`, on cherche le lien vers le fichier de l'image d'installation par le réseau et on le télécharge :

```
$ wget https://cdimage.debian.org/debian-cd/current/amd64/iso-cd/debian-11.3.0-amd64-netinst.iso
```

## Vérification du hash

Afin de vérifier que le fichier a correctement été téléchargé, on télécharge le fichier qui contient sa somme de contrôle :

```
$ wget https://cdimage.debian.org/debian-cd/current/amd64/iso-cd/SHA512SUMS
```

Puis on vérifie que le hash SHA512 est le bon :

```
$ sha512sum debian-11.3.0-amd64-netinst.iso
2810f894afab9ac2631ddd097599761c1481b85e629d6a3197fe1488713af048d37241eb85def681ba86e62b406dd9b891ee

$ cat SHA512SUMS
2810f894afab9ac2631ddd097599761c1481b85e629d6a3197fe1488713af048d37241eb85def681ba86e62b406dd9b891ee
bf872c5353cd062f604bcafcbadd56468cc8d187db29d938efae5c3aee0d460f07a553bfb85a3ea12a9210d15b65e8f76647
8c22a699cc71a121b2a6f27be880541fdd4bb44f8de5848d491cc455ad5386b179c7690c10510075a85b109276ac6655e801
```

Remarque : pour vérifier l'égalité de deux hashes à la main, il est utile de les aligner et de les mettre une ligne juste au dessus de l'autre dans un fichier par exemple. Il est aussi possible de faire vérifier l'égalité des hashes avec l'option `-c` de sha512sum :

```
$ sha512sum -c SHA512SUMS
debian-11.3.0-amd64-netinst.iso: Réussi
sha512sum: debian-edu-11.3.0-amd64-netinst.iso: Aucun fichier ou dossier de ce type
...
```

À ce stade, on a vérifié que le fichier qu'on a téléchargé est bien celui qui se trouve sur le serveur.

## Vérification de la signature du fichier contenant le hash

Mais il est possible que le serveur qui distribue les fichiers soit corrompu de sorte que l'image `iso` téléchargée ne soit pas la bonne et que le fichier `SHA512SUMS` corresponde au mauvais fichier (un attaquant qui aurait modifié le fichier sur le serveur peut aussi avoir modifié le fichier `SHA512SUMS`).

Afin de se prémunir d'une telle attaque, il est possible de vérifier la signature du fichier `SHA512SUMS`. Pour cela, on installe le paquet `debian-keyring` qui fournit les clefs GPG des développeur·es et responsables Debian :

```
$ sudo apt install debian-keyring
```

Ce paquet fournit entre autres le trousseau de clefs `/usr/share/keyrings/debian-role-keys.gpg`.

On télécharge la signature du fichier `SHA512SUMS` :

```
$ wget https://cdimage.debian.org/debian-cd/current/amd64/iso-cd/SHA512SUMS.sign
```

On vérifie la signature du fichier `SHA512SUMS` :

```
$ gpg --keyring /usr/share/keyrings/debian-role-keys.gpg --verify SHA512SUMS.sign
...
Bonne signature de « Debian CD signing key <debian-cd@lists.debian.org> » [inconnu]
...
```

Remarque : `gpg` se plaint que la clef n'est pas certifiée avec une signature de confiance. En général, la confiance en une clef téléchargée se fait de proche en proche en tissant une toile de confiance (si A fait confiance en la clef publique de B et qu'il possède la signature de par B de la clef publique de C, alors A peut faire confiance en la clef publique de C). Ici, la confiance vient du fait que lorsqu'on installe un paquet avec `apt`, la signature de ce paquet est vérifiée avant son installation. Ainsi, si on considère que la distro installée sur notre machine est intègre, on peut considérer qu'il existe un fil de vérifications entre nous et le signataire du fichier `SHA512SUMS` (voir le cours de crypto pour les histoires de toile de confiance).

Remarque : comment a-t-on pu savoir que le trousseau de clefs avait comme chemin `/usr/share/keyrings/debian-role-k`
?

On regarde la liste des fichiers fournis par ce paquet :

```
$ dpkg -L debian-keyring
/.
/usr
/usr/share
/usr/share/doc
/usr/share/doc/debian-keyring
/usr/share/doc/debian-keyring/NEWS.Debian.gz
/usr/share/doc/debian-keyring/README.gz
/usr/share/doc/debian-keyring/changelog.gz
/usr/share/doc/debian-keyring/changelog.old.gz
/usr/share/doc/debian-keyring/copyright
/usr/share/keyrings
/usr/share/keyrings/debian-keyring.gpg
/usr/share/keyrings/debian-maintainers.gpg
/usr/share/keyrings/debian-nonupload.gpg
/usr/share/keyrings/debian-role-keys.gpg
```

Ce paquet est assez simple, il contient de la doc `/usr/share/doc/debian-keyring/*` et des trousseaux de clefs (`/usr/share/keyrings/*.gpg`).

Le fichier de doc qui semble intéréssant est le README, il est compréssé , on peut le lire en le décompréssant directement :

```
$ zless /usr/share/doc/debian-keyring/README.gz
```

On voit une section "What the keyrings are" qui décrit les 4 trousseaux de clefs. Seul `debian-role-keys.gpg` correspond à ce que l'on cherche.

Remarque : comment on fait si on connait pas la commande `zless` ?

D'une part, on peut se demander quel est le format du fichier `README.gz` :

```
$ file /usr/share/doc/debian-keyring/README.gz
/usr/share/doc/debian-keyring/README.gz: gzip compressed data, max compression, from Unix,
```

D'autre part, on peut chercher à quoi correspond l'extension `.gz` :

```
$ man -k gz
diff2patches (1)     - Extraire les correctifs qui ne s'appliquent pas à debian/ dans des f
gzip (1)             - Compresser ou décompresser des fichiers
...
```

Dans les deux cas, on voit qu'il faut chercher du côté de `gzip`, on regarde son manuel :

```
$ man gzip
```

On voit que `gzip` est une commande pour comprésser, et que pour décomprésser, on peut utiliser la commande `gunzip` ou l'option `--decompress`, et que pour pouvoir envoyer le résultat dans la sortie standard plutôt que dans un fichier, on peut utiliser l'option `--stdout`. Ainsi, on peut accéder à cette doc avec la commande :

```
$ gzip --decompress --stdout /usr/share/doc/debian-keyring/README.gz | less
```