INSTALLATIONS SOUS GNU/LINUX-DEBIAN GESTION DE PAQUETS

Jean-Claude CATY
[Animateur GNU/Linux du club informatique de Loriol-sur-Drôme, auteur du projet EDIGEO sur Adullact]

Vous utilisez windows. L'installation de logiciels est souvent fastidieuse. Il faut trouver, le plus souvent sur internet, un programme installant le logiciel souhaité. Premier problème : ce programme d'installation est-il sécurisé ? N'est-il pas infecté par un virus ? Deuxième problème : L'installation est quelquefois compliquée et demande des précisions souvent inutile à ce stade et il faut cliquer n fois sur un bouton « suivant ». Rien de tout cela sous Linux ! S'agissant le plus souvent de logiciels libres, le processus peut être centralisé et donc être grandement sécurisé et simplifié. L'objectif principal de cet article est de montrer la philosophie sous système debian, les principes sous-jacents et savoir se débrouiller à minima avec le système de gestion de paquets apt.

MOTS-CLÉS : DEBIAN, PAQUET, APT, INSTALLATION, MISE À JOUR

INTRODUCTION

Cet article sur *la gestion des paquets* sous debian est inspirée du cours que je donne au club informatique de Loriol. Les aspirants linuxiens n'étant pas tous très expérimentés avec la ligne de commande, nous utilisons l'interface graphique *synaptic*. Nous verrons cependant les commandes correspondantes pour ceux qui souhaiteraient par exemple gérer avec la ligne de commande et intégrer certaines commandes d'installation dans leurs scripts.

Sous le système d'exploitation debian et ses dérivés comme ubuntu, mint, etc., les mises à jours logicielles, l'installation de logiciels ou d'autres types de paquets sont gérées via le système *apt* (**A**dvanced **P**ackaging **T**ool – Outil avancé de gestion de paquets).

Je vous propose dans cet article de faire un tour de ce système de manière interactive via des exemples concrets par la simple observation sur internet et quelques déductions logiques.

Cet article est le résultat de mes propres recherches, consultations de documentation internet … Il est donc possible que diverses fonctionnalité aient pu m'échapper. Cependant, compte tenu du fait que j'ai réussi à me débrouiller et répondre aux questions des aspirants linuxiens jusqu'à ce jour, il est fort probable que vous puissiez vous aussi y trouver réponses à vos interrogations.

En titre de cet article , j'ai bien parlé de *gestion de paquets* et non d'*installation de logiciels*. Normal car un paquet n'est pas seulement un logiciel. Citons rapidement quelques types de paquets disponibles : logiciels bien sur, mais aussi librairies logicielles, traductions, données, documentations…

PRINCIPES

Dépôts

Les dépôts contiennent des paquets. Nous éclaircirons dans les paragraphes suivants ce que recouvrent dépôts et paquets.

Il est important de signaler que les dépôts sont sécurisés. Il serait en effet destructeur de télécharger des logiciels porteurs de virus depuis un dépôt non sécurisé.

À cet effet, les paquets d'un dépôt sont signés. La signature est basée sur la cryptographie asymétrique. Cette technique utilise clé privée et clé publique. Tout paquet téléchargé est vérifié via sa signature électronique avant d'être installé. Bien entendu, l'utilisateur qui référence un dépôt devra porter une attention toute particulière à la validité de ce dépôt.

Le principe de la signature électronique est décrit dans Wikipedia **[1]**. Nous verrons comment ajouter un dépôt intéressant et configurer la vérification de signature de ce dépôt.

Versions debian

Le système debian diffère de nombre de ses héritiers par ses *versions* **[2]**. Plaçons-nous dans la peau d'un développeur de logiciels. Lorsqu'une nouvelle version de son logiciel est fonctionnelle, qu'elle a été testée, elle est intégrée dans les paquets debian dans une version dite *unstable* (instable en français). Cette version *unstable* porte toujours le même nom : *sid*. Cependant, cette version n'a – à priori – pas été testée dans les conditions réelles et elle comporte probablement des bugs pouvant être gênants pour les utilisateurs finaux. Il leur est donc déconseillé de travailler avec cette version sauf s'ils souhaitent contribuer au processus de test **[3]** de la version *unstable* de debian.

Après quelque temps passé dans la version *unstable*, le paquet est considéré comme ayant été suffisamment testé et ne plus être affecté de gros bugs. Il est alors transféré dans une version dite *testing*. À l'heure où j'écris ces lignes, la version *testing* porte le nom *trixie*. Cette version est considérée comme n'étant plus affectée que de bugs mineurs. Son principal avantage est de permettre à l'utilisateur de disposer de versions récentes des logiciels. Il faudra cependant demander régulièrement la mise à jour de debian pour une actualisation.

Tous les 2 ou 3 ans, la version *testing* est bloquée et une chasse aux bugs résiduels est lancée. Après corrections de ces bugs, la version *testing* devient la nouvelle version dite *stable*. À l'heure où j'écris ces lignes, la version stable porte le nom *bookworm* et a comme numéro de version 12. Depuis sa publication, *bookworm* a fait l'objet de mises à jour. La dernière version est numérotée 12.5. La date de sortie d'une nouvelle version stable n'est jamais publiée par debian. debian considère qu'il faut prendre le temps nécessaire pour que la nouvelle version stable soit aussi parfaite que possible. Évidemment, il reste toujours des bugs ou des mises à jour de sécurité importantes à appliquer et de nouvelles versions stables correctives sont régulièrement publiées.

Nous verrons dans le chapitre *Utilisation* comment gérer les versions de manière pratique.

Ces 3 versions sont les principales. D'autres sources de paquets existent et concernent sécurité, fréquence de mise à jour **[4]**.

sections dans un dépôt

Chaque version debian peut être plus ou moins étendue en précisant quelles sections **[5]** de logiciels intégrer.

Les logiciels qui se conforment aux DSFG (directives debian pour le logiciel libre) **[6]** sont listés dans la section *main*.

À l'opposé, les logiciels qui ne s'y conforment pas figurent dans la section *non-free*.

Certains paquets peuvent se conformer aux DSFG mais référencer des dépendances de *non-free*. Ces paquets sont situés dans une section intermédiaire nommée *contrib*.

Une section un peu spéciale référence les pilotes de matériel (par exemple les pilotes de cartes réseau wifi) : *non-free-firmware*.

Types de paquets

Revenons sur les types de paquets rapidement évoqués dans l'introduction. Il est important de comprendre qu'une installation ne comprend pas uniquement des logiciels. Voyons quelques types de paquets avec des exemples :

- paquets logiciels :

  Les dépôts référencent un grand nombre de logiciels sur des domaines variés.

- Librairies logicielles :

  Lorsqu'un développeur écrit un logiciel, il va avoir besoin de « *sous-éléments* » comme des outils pour gérer ou afficher des images, des outils de traitement mathématique… Il n'est pas utile de ***re-***développer ces outils, car il en existe très probablement déjà. Ces outils sont regroupés par thèmes dans des librairies logicielles. Par exemple, si le développeur souhaite gérer des images au format tiff dans son logiciel, il pourra utiliser la librairie *libtiff5*. Les paquets de librairies logicielles sont préfixés par « *lib* ».

  De plus, coté utilisation, un utilisateur lancera plusieurs logiciels. Si ces logiciels utilisent une même librairie, celle-ci ne sera chargée qu'une seule fois par le système d'exploitation et le code de la librairie sera partagé par les logiciels, assurant ainsi une meilleure utilisation de la mémoire. Bien entendu, les données gérées par la librairie seront différentes selon le logiciel utilisé et ne seront pas partagées. Cette gestion n'est pas l'apanage de Linux. Les systèmes d'exploitation modernes le font tous.

- Traductions

  Pour être utilisé dans différents pays, un logiciel doit proposer les traductions dans un maximum de langues. À l'installation du logiciel, il n'est pas utile d'intégrer toutes les traductions. Seule la traduction dans sa propre langue peut suffire. C'est pourquoi les paquets de traduction sont distincts des paquets logiciels. Ils sont souvent suffixés par une balise linguistique. Exemple : le paquet de la traduction française du logiciel de courriel *thunderbird* se nomme *thunderbird-l10n-fr*. Le premier suffixe (*l10n*) indique un paquet de localisation **[7]**, le deuxième suffixe (*fr*) est une balise linguistique **[7]**.

- Données

  Certains logiciels utilisent des données. Par exemple, un simulateur de vol aura besoin de cartes. Prenons le cas du jeu de stratégie de guerres anciennes *0ad*. Le logiciel est dans le paquet *0ad* et les données du jeu dans le paquet *0ad-data*. Un lien de dépendance relie les 2 paquets, car, pour être pleinement fonctionnel, le jeu nécessite ses données. Dans la suite de cet article, nous reviendrons rapidement sur les dépendances.

- Documentations

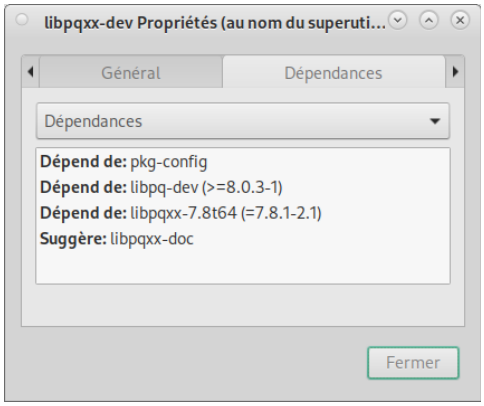
  Les dépôts de paquets intègrent des documentations. Ces documentations peuvent être diverses. Les plus utilisées sont bien entendues les documentations utilisateur. D'autres types de documentation existent. Nous venons par exemple de citer les *librairies logicielles*. Celles-ci intègrent souvent leur documentation.

  Un utilisateur utilise un logiciel, lequel gère les données dans la base de données postgresql. Ce logiciel aura besoin de la librairie logicielle *libpqxx* permettant de se connecter à la base de données. Le logiciel utilisera la librairie logicielle situé dans le paquet *libpqxx-6.4*. Nous verrons qu'il existe un lien de dépendance entre le logiciel et la librairie. L'utilisateur sera très intéressé d'installer la documentation de son logiciel (celle que nous venons de citer).

  Quittons la casquette de l'utilisateur, prenons celle du développeur du logiciel. Il aura besoin de fichiers complémentaires lors du processus de développement. Il trouvera ces fichiers dans le paquet correspondant nommé *libpqxx-dev* ; Enfin, pour utiliser pleinement la librairie, il aura besoin de la documentation correspondante située, elle, dans le paquet *libpqxx-doc*. Cette documentation est bien entendu différente de celle de l'utilisateur. Elle ne s'adresse qu'au développeur.

Dépendances

L'exemple que nous venons de voir montre divers niveaux de dépendances. Citons en cinq : dépendance pure, dépendance recommandée, dépendance suggérée, dépendance cassante et les conflits. Chaque dépendance peut cibler des versions de paquets (ex : ≥ 11 ou < 0.8.10)



L'exemple des dépendances de *libpqxx-dev* montre une dépendance pure entre le logiciel et la librairie libpq-dev et une dépendance suggérée avec *libpqxx-doc*. L'installation du logiciel implique l'installation de la librairie libpq-dev et suggère d'installer la documentation associée. Nous reviendrons sur les documentations dans un prochain chapitre.

Le logiciel de courrier électronique thunderbird est intéressant pour montrer les différents types de dépendances.

Ce logiciel télécharge des courriels que vous enverront vos correspondants. Ces courriels peuvent contenir des fichiers multimédia (musiques, images, vidéo …). Il existe donc des dépendances pures vers des librairies capables de gérer musique, image, vidéo.

thunderbird intègre des outils de cryptographie pour gérer des courriels chiffrés. Ces outils étaient auparavant gérés avec le paquet *enigmail*. Cette situation a provoqué une incompatibilité entre

thunderbird et enigmail. L'installation de thunderbird **casse** donc le paquet *enigmail*.

L'écriture de courriels avec thunderbird permet la vérification orthographique. Cependant, les utilisateurs ne parlent pas tous la même langue et chacun utilisera un dictionnaire adapté à sa langue. Le paquet thunderbird **recommande** donc le paquet logiciel dictionnaire anglais myspell-en-us. Un Français préférera probablement le dictionnaire français du paquet *hunspell-fr*. La recommandation ne sera donc pas totalement suivie mais l'attention de l'utilisateur linux éveillée en voyant la recommandation de thunderbird…

À un niveau moindre, thunderbird **suggère** l'installation de paquets. Le paquet *fonts-lyx* (polices de caractères TrueType) est par exemple suggéré. L'utilisateur pourra installer les paquets suggérés s'il le souhaite.

UTILISATION

sources.list et dépôt debian

Commençons par le haut de la pyramide. À l'installation de debian, il est demandé à l'utilisateur de sélectionner un miroir dans une liste. Ce miroir référence l'adresse internet du dépôt. Cette adresse est copiée dans un fichier */etc/apt/sources.list*

Examinons les 5 premières lignes de ce fichier *sources.list* :

```
jcc@neptune:~$ cat /etc/apt/sources.list | head -5
# deb cdrom:[Official debian GNU/Linux Live 11.1.0 mate 2021-10-09T12:35]/ bullseye main
# deb cdrom:[Official debian GNU/Linux Live 11.1.0 mate 2021-10-09T12:35]/ bullseye main

deb http://ftp.fr.debian.org/debian/ testing main contrib non-free
deb-src http://ftp.fr.debian.org/debian/ testing non-free contrib main
```

Les deux premières commencent par '**#**'. Ce sont des commentaires ! Nous voyons sur ces 2 lignes, le mot 'cdrom'. À l'installation, le miroir était la clé USB d'installation notée 'cdrom'. La première ligne a été préfixée par le caractère '**#**'. La 2e ligne a été oubliée provoquant un bug sur le système Linux debian *testing* installé. Si vous êtes victime du même problème, soit vous commentez cette 2e ligne, soit vous la supprimez, car elle est identique à la 1re. Il faut être administrateur (root) pour pouvoir modifier ce fichier.

Les 4e et 5e lignes référencent successivement :

- Le mot clé deb indique que le dépôt contient des paquets binaires
  Sur la 5e ligne, le mot clé deb-src précise l'adresse des paquets sources.
- L'adresse internet du dépôt : http://ftp.fr.debian.org/debian/
- La version référencée, ici *testing*

  Notons qu'à l'installation, la version n'était pas *testing* mais son nom (*trixie* par exemple). Cependant, il est intéressant de modifier le nom de la version par son code *immuable*. Par exemple *stable* ou *testing* voire *unstable* si vous souhaitez basculer vers *unstable*. Supposons que vous installiez la version *trixie*, correspondant à *testing* au moment de l'installation. L'installateur debian indiquera *trixie* dans notre fichier *sources.list* et le jour où *trixie* devient *stable*, nous changerons subrepticement de version.
- Enfin, nos lignes se terminent par *main contrib non-free*, référençant les trois sections correspondantes.

dépôt http://ftp.fr.debian.org/debian/

Jouons les curieux et glissons quelques regards dans le dépôt. Tapons l'adresse ci-dessus dans un navigateur (ici, le navigateur *lynx* en mode console) :

```
jcc@neptune:~$ lynx http://ftp.fr.debian.org/debian/
Index of /debian/

../
dists/                                             10-Sep-2022 10:42                   -
doc/                                               25-Oct-2022 07:52                   -
indices/                                           25-Oct-2022 08:17                   -
pool/                                              05-Oct-2022 17:09                   -
project/                                           17-Nov-2008 23:05                   -
tools/                                             10-Oct-2012 16:29                   -
zzz-dists/                                         14-Aug-2021 13:55                   -
README                                             10-Sep-2022 10:42                1328
README.CD-manufacture                              26-Jun-2010 09:52                1290
README.html                                        10-Sep-2022 10:42                3211
README.mirrors.html                                04-Mar-2017 20:08                 291
README.mirrors.txt                                 04-Mar-2017 20:08                  86
```

```
extrafiles                                 25-Oct-2022 08:24              233662
ls-lR.gz                                   25-Oct-2022 08:17            17049311
```

Intéressons-nous plus particulièrement aux dossiers *dists/* puis *pool/*.

Le dossier dists/

Un affichage partiel et partial nous montre que dists/ contient des dossiers vers différentes versions de debian (*stable*, *testing* …). Ce que confirme le fichier README : « *Chaque distribution est accessible par nom ou par état à partir d'ici.* »

```
jcc@neptune:~$ lynx http://ftp.fr.debian.org/debian/dists/
Index of /debian/dists/

../
bookworm/                                  24-Oct-2022 10:14                  -
bullseye/                                  10-Sep-2022 09:42                  -
sid/                                       19-Jul-2022 02:00                  -
stable/                                    10-Sep-2022 09:42                  -
testing/                                   24-Oct-2022 10:14                  -
testing-backports/                         19-Jul-2022 02:00                  -
testing-proposed-updates/                  19-Jul-2022 02:00                  -
testing-updates/                           19-Jul-2022 02:01                  -
unstable/                                  19-Jul-2022 02:00                  -
README                                     10-Sep-2022 10:42               1054
```

Succinctement, nous constatons que ces dossiers contiennent des fichiers dont les noms indiquent des contenus et référencent des types de processeurs (32 ou 64 bits, mips, arm…). Nous pouvons en déduire qu'un dossier *dists* permet de définir le sommaire des paquets du dépôt liés à la version debian et au type de processeur.

Le dossier pool/

Le dossier *pool* contient les paquets. Ces paquets sont organisés dans une hiérarchie de sous-dossiers prenant en compte la section debian (*main*, *contrib*, non-free …) et l'ordre alphabétique. Ainsi, les paquets du logiciel gimp sont situés dans le dossier http://ftp.fr.debian.org/debian/pool/main/g/gimp/. Leur suffixe est **.deb**.

Revenons aux paquets du logiciel gimp. Citons quelques noms de paquets et voyons à quoi ils correspondent :

- gimp_2.10.22-4_amd64.deb : Ce paquet correspond à la version 2.10.22-4 de gimp et est valide pour un processeur amd64 (intel ou amd 64 bits).
- gimp_2.10.22-4_i386.deb : Ce paquet correspond à la version 2.10.22-4 de gimp et est valide pour un processeur i386 (intel ou amd 32 bits).
- gimp_2.10.8-2_amd64.deb : Ce paquet correspond à la version 2.10.8-2 de gimp et est valide pour un processeur amd64 (intel ou amd 64 bits).

Compte tenu du nombre important de paquets pour un même logiciel, nous voyons l'importance du sommaire. Le sommaire de la version debian testing (dans le dossier *dist*) référencera par exemple le paquet de la version V2 alors que le sommaire de la version debian *stable* référencera le paquet de la version V1.

Nous constatons l'importance du nommage des paquets qui tient compte de sa version et du processeur de l'ordinateur sur lequel sera installé le paquet. En fonction de ces éléments, apt – le gestionnaire de paquets – téléchargera le paquet approprié et l'installera.

Rechargement du sommaire

Ces considérations nous amènent à définir une sorte de plan d'actions pour installer ou mettre à jour les paquets installés.

En premier lieu, il est important de synchroniser les sommaires des dépôts référencés. Rappelons que la sécurité impose que ce soit l'administrateur de l'ordinateur qui réalise cette opération et les suivantes. Les dépôts sont signés et le système *apt* vérifiera cette signature avant une éventuelle installation. Après rechargement du sommaire, le gestionnaire de paquets pourra comparer la version du sommaire avec la version installée, repérant alors les paquets à mettre à jour.

Sous l'application *synaptic*, le sommaire est rechargé via le bouton **Recharger** de la barre d'outils :

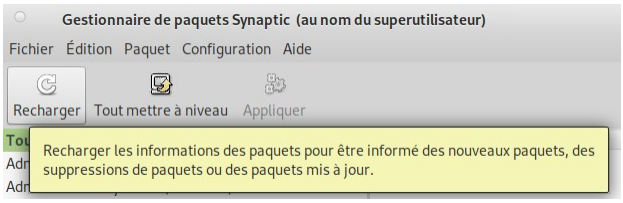


Figure 1: Rechargement du sommaire sous synaptic

En mode commande, il faut être administrateur (root) et lancer la commande *apt update*

```
jcc@neptune:~$ su # commande pour se connecter administrateur
Mot de passe : # indiquez ici le mot de passe root
# Nous passons de l'utilisateur jcc (prompt='$') à l'utilisateur root (prompt='#')
root@neptune:/home/jcc# apt update
Atteint :1 http://ftp.fr.debian.org/debian testing InRelease
Atteint :2 http://security.debian.org/debian-security testing-security InRelease
Atteint :3 http://ftp.fr.debian.org/debian testing-updates InRelease
Atteint :4 https://ftp.debian.org/debian trixie-proposed-updates InRelease
Atteint :5 http://ftp.debian.org/debian testing-backports InRelease
Atteint :6 https://debian.qgis.org/debian unstable InRelease
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
133 paquets peuvent être mis à jour. Exécutez « apt list --upgradable » pour les voir.
```

Figure 2: Rechargement du sommaire avec la commande apt update

La commande *apt update* nous informe de la possibilité d'afficher les mises à jour possibles. Affichons-en quelques-unes :

```
root@neptune:/home/jcc# apt list --upgradable

busybox/testing 1:1.35.0-4 amd64 [pouvant être mis à jour depuis : 1:1.35.0-2]
dh-elpa-helper/testing 2.0.15 all [pouvant être mis à jour depuis : 2.0.14]
gir1.2-harfbuzz-0.0/testing 5.2.0-2+b1 amd64 [pouvant être mis à jour depuis : 5.2.0-2]
gir1.2-javascriptcoregtk-4.0/testing 2.38.2-1+b1 amd64 [pouvant être mis à jour depuis : 2.38.2-1]
gir1.2-javascriptcoregtk-4.1/testing 2.38.2-1+b1 amd64 [pouvant être mis à jour depuis : 2.38.2-1]
gir1.2-webkit2-4.0/testing 2.38.2-1+b1 amd64 [pouvant être mis à jour depuis : 2.38.2-1]
gir1.2-webkit2-4.1/testing 2.38.2-1+b1 amd64 [pouvant être mis à jour depuis : 2.38.2-1]
```

Figure 3: Affichage des paquets pouvant être mis à jour avec la commande apt list –upgradable

Installer les mises à jour

En deuxième lieu, le sommaire étant synchronisé avec celui du dépôt sur internet, il est possible d'installer les mises à jour. Sous la version *stable*, celles-ci seront probablement peu nombreuses. Par contre, sous une version *testing* ou une version *unstable*, des mises à jour sont publiées très fréquemment.

Sous l'application synaptic, les mises à jour sont sélectionnées et installées via les boutons **Tout mettre à niveau** pour sélectionner les mises à jour puis **Appliquer** de la barre d'outils :

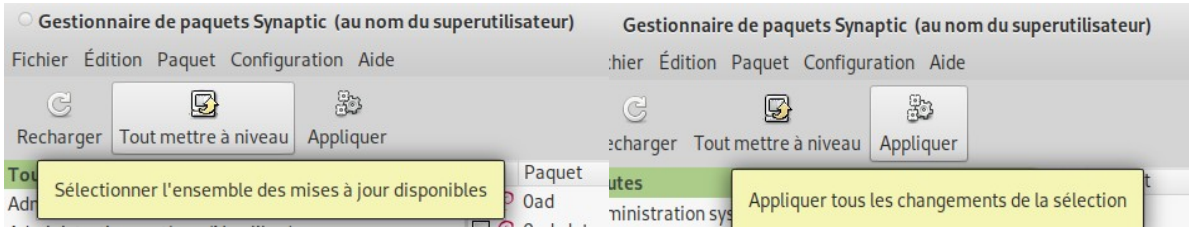


Figure 4 : Sélection puis installation des mises à jour.

En mode commande, il faut être administrateur (root) et lancer la commande *apt upgrade*. Je laisse le lecteur prendre connaissance du manuel en tapant la commande *man apt* dans un terminal.

Voici un exemple partiel du lancement et de l'affichage de la commande *apt upgrade*.

```
root@neptune:/home/jcc# apt upgrade
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Calcul de la mise à jour... Fait
Les NOUVEAUX paquets suivants seront installés :
  libbpf1 libpython3.11 libpython3.11-minimal libpython3.11-stdlib
Les paquets suivants seront mis à jour :
  apparmor caja-rename gir1.2-ibus-1.0 hwdata i2c-tools ibus ibus-data ibus-gtk ibus-gtk3 ibus-gtk4
64 mis à jour, 4 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 59,6 Mo dans les archives.
Après cette opération, 76,6 Mo d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n]
Réception de :1 http://ftp.fr.debian.org/debian testing/main amd64 libcap-ng0 amd64 0.8.3-1+b2 [17,1 kB]
Réception de :2 http://ftp.fr.debian.org/debian testing/main amd64 libseccomp2 amd64 2.5.4-1+b2 [46,6 kB]
Réception de :3 http://ftp.fr.debian.org/debian testing/main amd64 libselinux1 amd64 3.4-1+b3 [73,8 kB]
```

```
Réception de :4 http://ftp.fr.debian.org/debian testing/main amd64 libbpf1 amd64 1:1.0.1-2 [142 kB]
Réception de :5 http://ftp.fr.debian.org/debian testing/main amd64 iproute2 amd64 6.0.0-1+b1 [1 040 kB]
Réception de :6 http://ftp.fr.debian.org/debian testing/main amd64 libnewt0.52 amd64 0.52.21-6+b1 [58,8 kB]
Réception de :7 http://ftp.fr.debian.org/debian testing/main amd64 whiptail amd64 0.52.21-6+b1 [23,7 kB]
Réception de :8 http://ftp.fr.debian.org/debian testing/main amd64 apparmor amd64 3.0.7-1+b2 [616 kB]
59,6 Mo réceptionnés en 2s (34,7 Mo/s)
Préconfiguration des paquets...
(Lecture de la base de données... 831437 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../libcap-ng0_0.8.3-1+b2_amd64.deb ...
Dépaquetage de libcap-ng0:amd64 (0.8.3-1+b2) sur (0.8.3-1+b1) ...
Paramétrage de libcap-ng0:amd64 (0.8.3-1+b2) ...
(Lecture de la base de données... 831437 fichiers et répertoires déjà installés.)
```

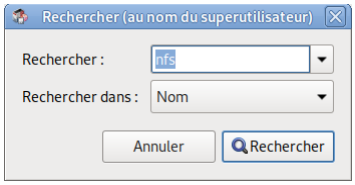
Figure 5: Installation des mises à jour avec la commande apt upgrade

Important : Les paquets téléchargés devront régulièrement être supprimés avant de trop encombrer le disque dur (nous y reviendrons).

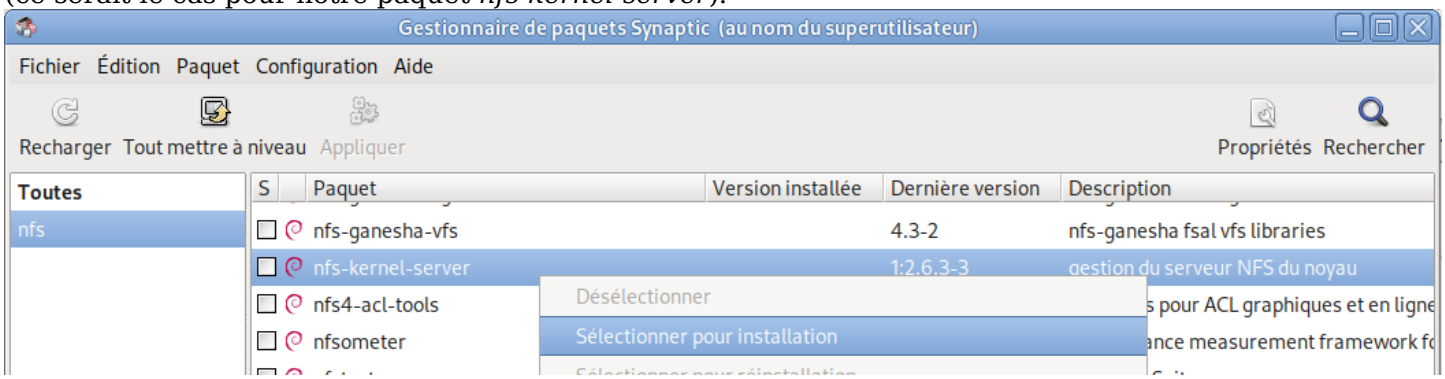
Rechercher un paquet

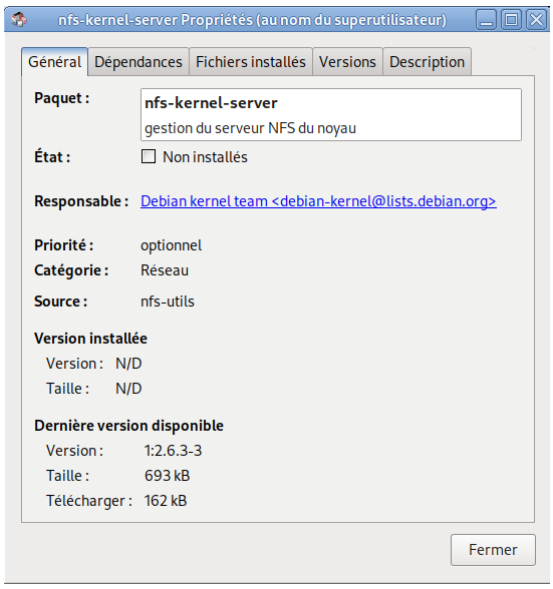
Bien souvent, nous souhaitons installer un logiciel par son nom. Nous avons repéré ce logiciel via un site web à moins que le copain du club informatique voisin nous l'ait conseillé. Parmi la multitude de paquets du dépôt, nous allons donc rechercher ce logiciel.

Supposons que nous souhaitions préparer le serveur de notre club informatique pour délivrer un service de distribution de fichiers par réseau (*network file system*) nommé **nfs**. En mode interface graphique, via le bouton **Rechercher**, nous recherchons **nfs** par son nom dans les dépôts référencés :



L'interface graphique nous présente l'ensemble des paquets dont le nom contient nfs. Nous pointons le paquet qui nous intéresse (ici, *nfs-kernel-server*) et par un simple clic-droit, nous sélectionnons notre paquet pour installation. Nous pouvons itérer le processus pour compléter la sélection. Il nous reste à Appliquer (3ème bouton de la barre d'outil) pour procéder à l'installation. Nous serons informés d'éventuelles dépendances à installer avant que l'outil ne télécharge le nécessaire et procède ensuite à l'installation. Cette installation ne préjuge pas d'éventuelles configurations à paramétrer plus finement (ce serait le cas pour notre paquet *nfs-kernel-server*).





Le paquet étant installé, il est possible d'obtenir des métadonnées sur le paquet. Dans l'interface graphique, ces informations sont classées dans les onglets *Général*, *Dépendances*, *Fichiers installés*, *Versions* et *Description*.

En mode console, une première commande permet d'afficher le résultat d'une recherche : *apt search*

```
root@neptune:/root# apt search nfs # Ci dessous, affichage très partiel du résultat de la commande
En train de trier... Fait
Recherche en texte intégral... Fait
4pane/testing 8.0-1+b2 amd64
  gestionnaire de fichiers détaillé en quatre panneaux

nfs-kernel-server/testing 1:2.6.3-3 amd64
  gestion du serveur NFS du noyau
Package: nfs-kernel-server
Version: 1:2.6.3-3
```

Le gros avantage de la recherche en ligne de commande concerne la possibilité de combiner la commande avec des tubes pour – par exemple – filtrer la sortie d'*apt-search* selon des critères qui vous appartiennent. Quelques exemples du chapitre suivant utilisent cette possibilité en filtrant avec la commande *grep*.

Pour obtenir les métadonnées d'un paquet en mode console, nous passons par la commande *apt show*

```
root@neptune:/root# apt show nfs-kernel-server # Ci dessous, affichage très partiel du résultat de la commande
Nous y retrouvons :
Recommends: # les paquets recommandés
Suggests: procps # les paquets suggérés
Conflicts: knfs, nfs-server # les paquets en conflit
Replaces: knfs, nfs-server : # les paquets à remplacer
...
```

Installer un logiciel, une traduction, une documentation …

Bien souvent, un logiciel est livré avec les traductions et une documentation relative à ce logiciel.

Les fichiers de traduction peuvent être très nombreux et installer une application donnée avec l'ensemble des traductions est consommateur de ressources. Vous ne souhaitez probablement pas installer la version chinoise d'une traduction (pour vous, le chinois c'est de l'hébreu …). Certains logiciels délivrent donc les fichiers de traduction dans des paquets séparés et vous installerez la ou les traductions qui vous intéressent après avoir installé le logiciel maître.

C'est par exemple le cas des logiciels Thunderbird (courrier électronique) et Firefox (navigateur internet) de la fondation Mozilla. Prenons le cas de Thunderbird :

Les paquets de traductions sont souvent référencés avec le code **l10n** dans leur nom. Ils sont suivis du code du pays de traduction.

Un affichage partiel des paquets de traductions de Thunderbird donnent le résultat suivant :

```
jcc@neptune:~$ apt search "l10n-" | grep thunderbird-l10n # Ci dessous, affichage très partiel d'une recherche
thunderbird-l10n-fi/testing 1:115.12.0-1 all
thunderbird-l10n-fr/testing,now 1:115.12.0-1 all  [installé]
thunderbird-l10n-fy-nl/testing 1:115.12.0-1 all
```

Nous remarquons les traductions pour la Finlande (thunderbird-l10n-fi), pour la France (thunderbird-l10n-fr), pour la langue frisonne occidentale parlée aux Pays-Bas (thunderbird-l10n-fy-nl). Pour avoir Thunderbird en français, vous installerez donc le paquet **thunderbird-l10n-fr**.

Portons maintenant le regard sur les documentations.

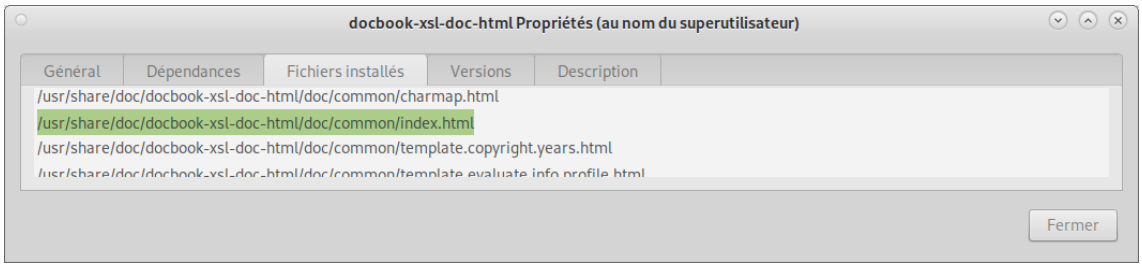
Vous porterez votre attention sur une documentation précise. Notre exemple portera sur la documentation docbool-xsl-doc pour deux raisons : d'une part, pour montrer qu'il peut exister plusieurs documentations pour un même logiciel, d'autre part pour montrer comment utiliser ensuite cette documentation.

Une recherche (filtrée) sur les paquets docbook-xsl-doc nous montre qu'il existe trois paquets concernant la documentation de ce logiciel :

```
jcc@neptune:~$ apt search docbook-xsl-doc | grep docbook-xsl-doc
docbook-xsl-doc-html/testing,now 1.79.1-2 all  [installé]
docbook-xsl-doc-pdf/testing 1.79.1-2 all
docbook-xsl-doc-text/testing 1.79.1-2 all
```

Nous constatons qu'il existe des documentations aux formats html, pdf et texte. Pour les besoins de cet article et comme le montre notre recherche, le paquet *docbook-xsl-doc-html* a été installé.

Les fichiers installés par ce paquet (le paquet doit être installé pour voir ce qui fut installé) montre par exemple le fichier */usr/share/doc/docbook-xsl-doc-html/doc/html/index.html*. Vous pouvez copier ce lien et l'ouvrir dans un navigateur pour visualiser la documentation.



Le conseil que je vous donne est de privilégier les documentations au format html beaucoup plus lisibles. Dans notre exemple, la documentation au format pdf est une feuille de style quasiment incompréhensible pour qui ne sait l'utiliser… Dans certains cas, vous pourrez combiner traduction et documentation pour obtenir la documentation dans votre langue préférée (exemple : le paquet *debian-edu-doc-fr*).

Changer de version

Considérons le cas où les gestionnaires du projet debian promulgue une nouvelle version du système. Dans ce cas, il est fortement probable que vous rencontriez une erreur car la signature du dépôt aura été modifiée. Dans un tel cas, cette erreur sera signalée au rechargement du sommaire par un long message qu'un débutant, voire même un utilisateur aguerri, ne comprendra pas. Ce message vous signalera en substance que la clé du dépôt n'est pas valide. C'est logique dans la mesure où la clé de la nouvelle version vient d'être créée et ne correspond plus à la clé de votre installation. Pour remédier à cette situation (je ne connais pas la commande de l'interface graphique permettant la mise à jour correcte, ni s'il en existe une ?), voici les deux commandes à lancer dans un terminal (en mode administrateur) pour synchroniser votre installation avec votre dépôt debian :

```
root@neptune:/root# apt upgrade --without-new-pkgs
root@neptune:/root# apt full-upgrade
```

Un peu de ménage

Les paquets téléchargés, par exemple lors d'une mise à jour sont stockés dans le dossier */var/cache/apt/archives*. Compte tenu de leur nombre et de leur taille, ils encombrent rapidement l'espace disque. Pour ne pas être rapidement submergé, il est important de faire le ménage et supprimer régulièrement ces paquets. Ceci est d'autant plus vrai lorsque vous utilisez une version *testing* ou *sid* compte tenu des fréquences élevées de mises à jour.

Si vous passez par l'interface graphique, vous passerez par votre navigateur de fichier préféré (par exemple *nemo*), vous basculez en mode administrateur (votre navigateur de fichier doit le permettre !) puis vous supprimez les fichiers suffixés *.deb* de ce dossier. **Une opération de mise à jour ou d'installation ne doit pas être en cours**.

En mode console (commande *apt clean* en mode administrateur), l'opération est mieux sécurisé car une vérification est assurée : n'y a t'il pas une installation en cours pendant que vous tentez de supprimer des paquets comme dans l'exemple ci-dessous ?

```
root@neptune:/home/jcc# apt clean
E: Could not get lock /var/cache/apt/archives/lock. It is held by process 26306 (synaptic)
N: Be aware that removing the lock file is not a solution and may break your system.
E: Impossible de verrouiller le répertoire /var/cache/apt/archives/
```

Ajouter un dépôt

Diverses organisations de logiciels libres proposent d'intégrer leurs propres dépôts dans votre système

debian. Citons par exemple qgis (système d'information géographique), postgresql (base de données). Ces dépôts ont l'avantage de proposer des paquets plus récents, mis à jour régulièrement, corrigés de bugs gênants …

Mais, référencer ces dépôts demande de savoir où les trouver, quelles versions référencer, assurer la sécurité avec la clé de chaque dépôt, se conformer au système de gestion de gestion de paquets *apt*.

Le principe est le suivant :

1. Installer la clé du dépôt pour sécuriser ce dépôt,
2. Créer et définir un fichier suffixé *.sources* pour référencer le dépôt et sa clé

Premier exemple : ajouter le dépôt postgresql

Une recherche sur internet nous donne les références du dépôt postgresql : https://apt.postgresql.org/pub/repos/apt/.

Avec un navigateur internet, consultons ce dépôt. Nous y trouvons les fichiers *README, ACCC4CF8.asc* et les dossiers dists et pool. Nous avons déjà vu l'utilité des dossiers dists et pool et nous les utiliserons pour référencer correctement le dépôt dans un fichier *postgresql.sources*. Le fichier *ACCC4CF8.asc* est la clé du dépôt et nous devons installer cette clé dans le dossier */etc/apt/keyrings*. Si l'on consulte les propriétés de ce dossier, nous constatons que le propriétaire est root, le groupe est root et que tous autres utilisateurs et groupes n'ont pas la faculté d'ajouter des fichiers dans cet espace ! La sécurité du système impose donc que ce soit l'administrateur qui référence de nouveaux dépôts.

Les commandes suivantes exécutées sous root permettent cet import de la clé :

```
jcc@neptune:~$ su # pour basculer en mode 'root'
Mot de passe :
root@neptune:/home/jcc# cd /etc/apt/keyrings/ # se déplacer dans le dossier /etc/apt/keyrings/
root@neptune:/etc/apt/keyrings# wget https://www.postgresql.org/media/keys/ACCC4CF8.asc
```

Suite à ces commandes, le fichier *ACCC4CF8.asc* a été aspiré dans le dossier */etc/apt/keyrings*. Il ne nous reste qu'à opérer le référencement du dépôt dans un fichier */etc/apt/sources.list.d/postgresql.sources*

Les fichiers référençant les dépôts sont obligatoirement dans le dossier */etc/apt/sources.list.d/*. Vous le nommer comme bon vous semble mais ils doivent cependant être terminés par *.sources*. Moyennant ces deux conditions, ils seront automatiquement pris en compte en synchronisant les sommaires. Antérieurement à la version 11, cette gestion était réalisée via des fichiers *.list*. Nous ne détaillerons pas les fichiers *.list* dans cet article. Notre fichier *postgresql.sources* sera constitué comme suit :

```
Types: deb deb-src
URIs:https://apt.postgresql.org/pub/repos/apt/
Suites: trixie-pgdg-testing
Components: main
Architectures: amd64
Signed-By: /etc/apt/keyrings/postgresql.asc
```

Détaillons chaque ligne :

- **Types: deb deb-src**
  Nous spécifions ici *deb* pour référencer les binaires et *deb-src* si nous voulons récupérer les sources.
- **URIs:https://apt.postgresql.org/pub/repos/apt/**
  Nous reportons ici l'adresse du dépôt trouvée sur internet.
- **Suites: trixie-pgdg-testing**
  Nous consultons le dossier dists/ pour connaître la version à installer et prenons celle correspondant au mieux au système Linux installé. Sur mon ordinateur où est installé la version testing (nom de code *trixie*), c'est la version *trixie-pgdg-testing.*
- **Components: main**
  La liste des sections possibles pourra être complétée en consultant quels sous-dossiers existent dans https://apt.postgresql.org/pub/repos/apt/*pool/*.
- **Architectures: amd64**
  L'architecture correspondant à votre processeur. Le plus souvent avec les ordinateurs 64 bits actuels, ce sera amd64.
- **Signed-By: /etc/apt/keyrings/postgresql.asc**
  et enfin, une référence vers la clé du dépôt précédemment aspirée vers le dossier /etc/apt/keyrings/.

Deuxième exemple : ajouter le dépôt qgis

```
Types: deb deb-src
URIs: https://qgis.org/debian
Suites: noble
Components: main
```

```
Architectures: amd64
Signed-By: /etc/apt/keyrings/qgis-2022.gpg.key
```

Une recherche sur internet nous donne les références du dépôt qgis : https://qgis.org/debian nous permettant de définir la section *URIs*:

Ceux qui souhaitent consulter les sources pourront ajouter *deb-src* à *deb* dans la section *Types:*

Pour connaître la version qui correspond le mieux à notre installation et définir le paramètre *Suites:*, il suffit de parcourir le sous dossier https://qgis.org/debian/dists et choisir la version ad-hoc. Personnellement, ma version étant debian testing (trixie), je constate qu'elle n'est pas mentionnée. Aussi ai-je référencé la version ubuntu *noble* qui me semble la mieux adaptée.

Le dossier https://qgis.org/debian/pool ne référence que *main* que nous reportons dans la section *Components:*

Avec un navigateur internet, consultons ce dépôt. Parmi quelques fichiers et dossiers, nous y trouvons les dossiers dists et pool que nous utiliserons pour référencer correctement le dépôt dans un fichier qgis.*sources*.

La clé du dépôt semble être dans le fichier *qgis-2021.gpg.key*. Cependant, en cherchant mieux, nous pouvons trouver un fichier **qgis-2022.gpg.key** à l'adresse https://download.qgis.org/downloads/. Il nous restera à tester quelle clé convient le mieux à notre configuration… Nous pouvons télécharger ces deux clés vers le dossier /etc/apt/keyrings/ et tester successivement ces deux versions en jouant sur la section *Signed-By:* dans le fichier qgis.*sources*.

Synchroniser les dépôts

Les nouveaux dépôts étant maintenant paramétrer, il s'agit maintenant de les prendre en compte. Pour cela, rien de plus simple : rechargeons tout bêtement les sommaires ! Vous connaissez la commande : bouton « **Recharger** » ou commande « **apt update** ». Le système consulte systématiquement le dossier /etc/apt/sources.list.d et intègre les fichiers suffixés .sources dans le référentiel.

Si vous avez une erreur, vérifier les paramètres référencés et vérifiez que la clé du dépôt est bien prise en compte. Il est préférable de paramétrer vos dépôts un par un et les référencer avant de passer au suivant.

CONCLUSION

apt est un système présentant des caractéristiques intéressantes :

- Il est **évolutif** car il permet d'ajouter des dépôts tiers comme nous avons pu le voir avec postgresql et qgis,
- Il est **intelligent** car il connaît les dépendances obligatoires et en suggère d'autres,
- Il est **pratique** car l'installation d'un paquet et les mises à jour globale sont plutôt simples et très commodes,
- Il est **sûr** car il intègre des mécanismes de signature par clé publiques. Vous devez toutefois prêter une attention tout particulière aux dépôts que vous référencez.
- Il est **centralisé** car, dès la prise en compte de vos dépôts, vous avez tout sous la main.

Quand vous aurez goûté ces caractéristiques, vous aurez toute chance d'être conquis !

RÉFÉRENCES

[1] Signature : https://fr.wikipedia.org/wiki/Signature_num%C3%A9rique

[2] Versions debian : *https://www.debian.org/releases/*

[3] Aider debian : *https://www.debian.org/intro/help*

[4] *https://wiki.debian.org/fr/SourcesList#Sources_de_paquets_couramment_utilis.2BAOk-es*

[5] Sections : *https://wiki.debian.org/fr/SourcesList#Composants*

[6] DSFG : *https://www.debian.org/social_contract#guidelines*

[7] Localisation : *https://fr.wikipedia.org/wiki/Localisation_%28linguistique%29*