Mise à jour de la Slackware vers une nouvelle version

Mise à niveau ou bien installation depuis zéro

Si vous utilisez une Slackware plutôt ancienne et si vous envisagez un saut de plusieurs numéros de version, la meilleure solution pour vous sera d'installer une Slackware récente à partir de zéro. Bien trop de changements intrusifs auront été faits à la distribution si votre Slackware date de quelques années. Cela rendrait la mise à jour à la main difficile et le succès n'en serait pas garanti.
Dans ces cas là, il vaut mieux faire une sauvegarde de la liste de vos paquets (“`ls -lart /var/log/packages`”), sauvegarder également votre répertoire “`/etc`” ainsi que (mais vous l'avez déjà fait) vos données personnelles. Après avoir formaté votre disque dur et fait l'installation à partir d'un support bootable, dans environ une heure vous pourrez reprendre vos activités.

Si vous voulez passer à la version suivante de Slackware, vous pouvez le faire à la main en suivant les instructions données dans le fichier “`UPGRADE.TXT`” que vous trouverez à la racine du DVD/CD1 Slackware . Des instructions avancées et bon nombre de conseils sont disponibles à “`CHANGES_AND_HINTS.TXT`” au même endroit. On peut aussi faire ce type de mise à niveau en utilisant slackpkg. Il allègera grandement votre tâche.
Je fais (alienbob) *toujours* appel à slackpkg (avec soin) pour passer d'une version stable de mes systèmes Slackware à la version suivante. J'appelle cette action une “*mise à jour du système*”.
Vous pouvez employer le même procédé pour migrer vers slackware-current, maintenir à jour un système slackware-current, ou bien franchir le pas depuis une slackware-pas-si-current vers la version stable qui vient de sortir.

Considérations sur le noyau

Se contenter de lancer slackpkg en espérant que tout aille pour le mieux, ça ne marche pas. Il faut prendre en compte plusieurs données importantes. À retenir absolument:

Ne faites jamais la mise à jour du noyau que vous utilisez.

Pourquoi donc? C'est simple - vous feriez la mise à jour de centaines de paquets et il vous faudrait être préparé à la situation peu probable de votre ordinateur ne fonctionnant plus après une mise à niveau du système. Ce à quoi vous ne voulez pas être confronté, c'est votre système qui ne booterait plus du tout. Il se peut qu'une nouvelle version de slackware installe un noyau qui refuse de démarrer votre ordinateur (ce risque est faible mais toutefois...soyez prévenu). C'est pourquoi vous devez conserver votre “ancien” noyau actif installé et garder une section pour lui dans votre fichier `/etc/lilo.conf`. De cette façon, si le nouveau noyau ne réussit pas à booter, vous pouvez vous rabattre sur l'ancien et commencer à chercher ce qui n'allait pas.
Fondamentalement, vous devez prendre les mêmes précautions quand vous compilez un nouveau noyau vous-même.

Considérations sur les pilotes vidéo

Si votre ordinateur est équipé d'une carte vidéo à processeur Nvidia ou Ati et si vous avez installé les pilotes d'accélération graphique de ces sociétés (avec seulement les binaires aux sources privatives), ne tentez pas de démarrer une session X après la mise à niveau vers la version plus récente de Slackware.
Ces pilotes dépendent de la version du noyau, de la version de Mesa et du serveur X-org. Il vous faut réinstaller le binaire du pilote avant de démarrer le mode graphique. De plus, les paquets Slackware de Mesa et Xorg-server écrasent de toute façon des fichiers essentiels de ces pilotes propriétaires qui permettent l'accélération graphique.

Si vous voulez savoir comment gérer ces pilotes binaires, nous donnons des indications plus détaillées dans l'article “Proprietary Graphics Drivers” du présent Wiki.

Considérations sur slackpkg

Si vous mettez à niveau Slackware (voir ci-dessous la démarche), une des premières étapes sera de mettre à jour slackpkg. La commande `upgradepkg` installera un fichier `/etc/slackpkg/mirrors.new`. C'est le fichier qui contient les URLs des miroirs qui diffusent la nouvelle version de Slackware. Vous comparerez cette liste et sa version plus ancienne afin d'en fusionner les contenus.

Veillez bien à ne décommenter qu' ***une seule ligne*** *qui pointe vers un miroir pour la bonne édition de Slackware avec la version et l'architecture de votre choix.*

Mise à jour du système avec slackpkg

Les actions suivantes devraient convenir dans tous les cas:

- Blacklister ces paquets relatifs au noyau, dans “`/etc/slackpkg/blacklist`”:

  ```
  kernel-generic
  kernel-generic-smp
  kernel-huge
  kernel-huge-smp
  kernel-modules
  kernel-modules-smp
  ```

  Pour éviter une mise à niveau accidentelle du noyau que vous utilisez.

- Blacklister des paquets venant de dépôts tiers en ajoutant les lignes qui conviennent pour leurs *repo tags* (“identifiants de dépôts”). Voici par exemple comment blacklister SlackBuilds.org, AlienBOB et multilib:

  ```
  [0-9]+_SBo
  [0-9]+alien
  ```

```
[0-9]+compat32
```

- Si un ou plusieurs noyau(x) ont été ajoutés à l'édition de Slackware vers laquelle vous mettez à niveau, alors utilisez "`installpkg`" pour installer ces nouveaux paquets du noyau (n'utilisez pas "`upgradepkg`" parce que cela effacerait votre noyau actif).
  Il vous faudra installer au moins un noyau (kernel-generic, kernel-generic-smp, kernel-huge, ou kernel-huge-smp) et le paquet correspondant des modules du noyau (kernel-modules ou kernel-modules-smp).
  Vous ne pouvez pas employer `slackpkg` pour cette étape.

- Maintenant que nous disposons du ou des nouveau(x) noyau(x) avec les modules en place, nous pouvons commencer la mise à niveau des autres paquets. D'abord réactualisons la base de données de paquets de `slackpkg`:

```
# slackpkg update
```

- Quand slackpkg a mis à jour sa base de données interne, la première chose à faire est de mettre à niveau `slackpkg` lui-même vers la version la plus récente (y compris les URL's pour la nouvelle version de Slackware et chacune des règles de mise à jour qui s'appliquent à la nouvelle version). Si vous ne réussissez pas à effectuer cette étape, `slackpkg` va se mettre à jour lui-même en pleine *mise à jour du système*, et s'arrêter après ceci...

```
# slackpkg upgrade slackpkg
# slackpkg new-config
```

  Cette dernière commande `new-config` est là pour que vous puissiez voir la différence entre les anciens et les nouveaux fichiers de configuration de `slackpkg`, tout particulièrement `/etc/slackpkg/mirrors` et `/etc/slackpkg/blacklist` sont des fichiers que vous devez vérifier. Il est généralement recommandé d'écraser `/etc/slackpkg/slackpkg.conf`.

- Habituellement, une nouvelle édition de Slackware a une nouvelle version des bibliothèques de GNU C. Les nouveaux paquets sont compilés en liaison avec cette nouvelle version de `glibc`. Pour éviter l'échec de la mise à niveau, il vous faut mettre à jour le paquet `glibc-solibs` à la main, tout de suite après la mise à jour de `slackpkg`:

```
# slackpkg upgrade glibc-solibs
```

  Permettez moi de donner un exemple d'une telle défaillance potentielle: quand `slackpkg install-new` installe le paquet `libusb-compat`, votre commande `gpg` s'arrête parce qu'elle est liée dynamiquement à libusb.so dans la version ancienne qui va être écrasée par le nouveau paquet `libusb-compat`. La nouvelle bibliothèque a besoin du nouveau paquet `glibc`, `gpg` s'arrête à cause de l'erreur de lien aux bibliothèques et `slackpkg` stoppera la mise à jour du système parce qu'il lui faut vérifier la signature gpg de chaque paquet avant de le mettre à niveau. Par la mise à niveau du paquet `glibc-solibs` on évite les erreurs de lien aux bibliothèques en donnant les symboles corrects de "`GLIBC`".

- `Slackpkg` va mettre à jour l'ordinateur vers la nouvelle version de Slackware:

```
# slackpkg install-new
# slackpkg upgrade-all
```

```
# slackpkg clean-system
```

  - La première de ces trois commandes (`slackpkg install-new`) installera chaque paquet repéré par la chaîne “`Added.`” trouvée dans le fichier Slackware ChangeLog.txt. Cette commande **n'installera aucun** autre paquet qui ne soit déjà installé.
    Par exemple si vous n'aviez pas KDE précédemment installé, la commande “`slackpkg install-new`” n'ajoutera pas les paquets KDE à votre ordinateur tout à coup.
  - La seconde commande (`slackpkg upgrade-all`) va comparer chacun des paquets Slackware officiels qui sont actuellemnt installés dans votre système avec la liste de paquets de votre miroir Slackware. Si une version différente est disponible, cette version sera (téléchargée et) mise à niveau. [1)]
  - La troisième commande (`slackpkg clean-system`) vous présentera une vue d'ensemble de tous les paquets qui sont actuellement installés dans votre système mais qu'on ne retrouve pas dans l'édition de Slackware Linux vers laquelle vous faites une mise à niveau. Ce qui signifie que la liste obtenue vous indiquera tous les paquets qui ont été retirés de la Slackware. Pour Slackware 14 , c'est le cas de `kdeaccessibility`, `kdebase`, ... Un autre exemple de paquet qu'on ne retrouve plus dans Slackware 14 est `ntfsprogs`. En fait ce paquet n'a pas été *enlevé*, mais *renommé*... pour Slackware cela équivaut à une *suppression* de paquet suivie d'un *ajout*.
    La commande vous montrera également les paquets provenant de dépôts tiers (autres que slackware.com)! Utilisez cette commande avec prudence: vous devez désélectionner chaque paquet que vous voulez garder (i.e. tous les paquets issus de dépôts extérieurs), puis cliquez “OK” pour que `slackpkg` enlève tous les paquets obsolètes.

- Une mise à niveau de cette importance aura installé plusieurs fichiers “`.new`”. Dans certains paquets il y a des fichiers de configuration qui ont été renommés (à l'intérieur du paquet) par une extension “`.new`” pour qu'un fichier de configuration existant (contenant vos réglages personnels) ne soit pas écrasé imprudemment pendant la mise à niveau. Slackpkg va rechercher la présence de ces fichiers à extension “`.new`” dès la fin d'une mise à niveau ou d'une installation et il vous demande comment vous souhaitez en disposer.
  Il vous est conseillé de mettre à niveau vers les nouvelles versions des fichiers de configuration quand c'est possible, parce que, souvent elles amélioreront votre configuration logicielle. Slackpkg vous permet de voir les différences entre les anciens et les nouveaux fichiers et même, de fusionner les deux fichiers. Vous pouvez aussi décider de garder le fichier “`.new`”, en laissant l'ancien fichier en place, pour juger plus tard si les changements ne sont pas trop intrusifs.
  À tout moment vous pouvez déclencher une vérification sur les fichiers “`.new`” en lançant la commande

```
  * # slackpkg new-config
```

  vous pourrez alors utiliser l'interface commode de slackpkg pour fusionner les modifications.

- Vous devriez alors probablement décider d'installer un noyau générique, particulièrement si vous employez LVM ou RAID, ou si vous avez installé Slackware sur un disque chiffré par LUKS. C'est aussi le conseil donné dans le README qui est sur le DVD/CD Slackware. Par contre - si votre configuration système est simple et votre matériel très récent, vous pouvez vous en tenir au noyau ***énorme*** (“huge” kernel).
  *N'oubliez pas qu'on ne peut pas utiliser un 'initial ramdisk' avec un noyau 'archi-complet' (“huge”), mais il est **impératif** de créer un nouvel 'initial ramdisk ' quand on va utiliser un noyau générique!*

Si vous ne savez pas très bien comment faire pour cette création, vérifiez bien si vous démarrez ou non sur un noyau “huge” qui ne nécessite pas un ramdisk de chargement de modules au démarrage, dit “initial ramdisk”.
Cependant, le script “`mkinitrd_command_generator.sh`” vous aidera à créer un “initial ramdisk” si nécessaire. Exécutez ce script avec la version du *nouveau* noyau en paramètre et il vous donnera un exemple de commande “`mkinitrd`” qui fonctionnera pour votre configuration système et vos réglages matériels particuliers:

```
# /usr/share/mkinitrd/mkinitrd_command_generator.sh -k 3.2.29
```

et donnera en sortie les précisions ci-dessous (la version de noyau 3.2.29 correspond à la Slackware 14)

```
#
# mkinitrd_command_generator.sh revision 1.45
#
# This script will now make a recommendation about the command to use
# in case you require an initrd image to boot a kernel that does not
# have support for your storage or root filesystem built in
# (such as the Slackware 'generic' kernels').
# A suitable 'mkinitrd' command will be:

mkinitrd -c -k 3.2.29 -f ext4 -r /dev/sdb2 -m usb-
storage:pcmcia_core:pcmcia:mmc_core:ssb:modprobe:usbhid:ehci-hcd:ohci-
hcd:mbcache:jbd2:ext4 -u -o /boot/initrd.gz
```

Vous pouvez copier et coller cette ligne de commande dans votre console, et lui faire créer un “initial ramdisk” pour vous.

Si vous utilisiez déjà un noyau générique avec par conséquent son initrd, nous vous recommandons vivement de créer un **nouvel** initrd nommé de façon **nouvelle** et unique! Par exemple, vous pouvez copier l'exemple ci-dessus et modifier le nom du fichier initrd comme ceci:

```
mkinitrd -c -k 3.2.29 -f ext4 -r /dev/sdb2 -m usb-
storage:pcmcia_core:pcmcia:mmc_core:ssb:modprobe:usbhid:ehci-
hcd:ohci-hcd:mbcache:jbd2:ext4 -u -o /boot/initrd_3.2.29.gz
```

- Après avoir choisi le noyau que vous allez utiliser et dès que vous aurez créé un 'initial ramdisk', vous devrez mettre à jour votre fichier “`/etc/lilo.conf`” en indiquant une section pour le nouveau noyau (*n'enlevez pas votre noyau actif!*). Le script “`mkinitrd_command_generator.sh`” peut vous aider à trouver le bon bloc à ajouter à `/etc/lilo.conf`. Par exemple, la commande:

```
# /usr/share/mkinitrd/mkinitrd_command_generator.sh -l /boot/vmlinuz-
generic-3.2.29
```

produira en sortie les indications suivantes que vous pouvez copier/coller:

```
# Linux bootable partition config begins
# initrd created with 'mkinitrd -c -k 3.2.29 -f ext4 -r /dev/sdb2 -m
usb-storage:pcmcia_core:pcmcia:mmc_core:ssb:modprobe:usbhid:ehci-
hcd:ohci-hcd:mbcache:jbd2:ext4 -u -o /boot/initrd.gz'
image = /boot/vmlinuz-generic-3.2.29
  initrd = /boot/initrd.gz
  root = /dev/sdb2
  label = 3.2.29
  read-only
# Linux bootable partition config ends
```

Notez que cette commande ajoute une ligne “initrd” à la section du noyau. Si `mkinitrd` attribue un nom unique à votre ramdisk, écrivez bien ce nom là dans la section de /etc/lilo.conf qui convient.
Vous n'avez pas besoin de la ligne “initrd” dans le cas où vous utilisez un noyau “huge” ('énorme' car il inclut déjà un maximum de modules).

- Enfin, lancez la commande “`lilo`” pour que le changement soit permanent et que le nouveau noyau soit ajouté au menu de démarrage de lilo.
  Retenez bien, vous devez toujours pouvoir recourir au démarrage de votre système (le boot) grâce à un noyau précédent au cas où le nouveau noyau Slackware vous créerait des difficultés.

Note de bas de page: ayez confiance en slackpkg pour réussir une mise à niveau du système, votre intelligence et vos soins attentifs interviendront aussi.

Considérations sur multilib

Si vous mettez à niveau un ordinateur tournant sous Slackware 64-bit multilib, alors il y a plusieurs autres points à considérer.

Une installation multilib, cela signifie que vous avez remplacé les paquets `gcc` et `glibc` de la Slackware 64-bit par leurs versions multilib (ie.e. qui peuvent traiter aussi bien les binaires 32-bit que les 64-bit). Et par ailleurs, vous avez installé un ensemble de paquets Slackware 32-bit “convertis” pour votre Slackware 64-bit multilib. Ces modifications sont toutes nécessaires pour pouvoir utiliser et compiler des logiciels 32-bit.

En mettant à jour un tel système, vous devez bien sûr mettre à jour les paquets standard de Slackware, mais il vous faut mettre à jour séparément les paquets spécifiques multilib grâce aux nouvelles versions qui sont disponibles à http://slackware.com/~alien/multilib/

- Tout d'abord (si vous utilisez slackpkg) blacklistez tous ces fichiers pour leur éviter d'être remplacés ou supprimés par accident lors d'une mise à jour du système. Si vous ne les blacklistez pas, vous devrez faire bon nombre de désélections manuelles dans “slackpkg clean-system”. Depuis la version de `compat32-tools` pour Slackware 14.0 vous pouvez simplement ajouter deux lignes au fichier `/etc/slackpkg/blacklist`:

```
[0-9]+alien
```

```
[0-9]+compat32
```

Ensuite il vous faudra télécharger et mettre à niveau à la main les paquets multilib. Dans l'exemple suivant je prends la Slackware 14.0 comme version vers laquelle vous allez mettre à niveau.

- Téléchargez les paquets multilib qui conviennent à votre nouvelle version de Slackware depuis un miroir, comme ceci:

```
# rsync -av
rsync://taper.alienbase.nl/mirrors/people/alien/multilib/14.0/
multilib-14.0/
```

Cette commande va créer un nouveau sous-répertoire "`multilib-14.0`" dans votre répertoire courant, avec tous les paquets à l'intérieur.

- installez/mettez à niveau les paquets `gcc` et `glibc` existants, ainsi que les outils compat32 (`compat32-tools`):

```
# cd multilib-14.0
# upgradepkg --install-new *.t?z
```

- Mettez à niveau l'ensemble des paquets 32-bit Slackware convertis (rendus compatibles avec l'environnement multilib, souvent appelés paquets "*compat32*"):

```
# upgradepkg --install-new slackware64-compat32/*-compat32/*.t?z
```

Sinon vous pouvez lancer le script "`massconvert32.sh`" qui a été installé en tant qu'élément du paquet `compat32-tools`. En lui passant un répertoire de paquets Slackware 32-bit (ou l'URL d'un miroir Slackware 32-bit) en paramètre, il créera un ensemble de paquets convertis "`compat32`" que vous pourrez alors installer. Vous n'aurez à procéder ainsi que si vous pensez que le contenu du répertoire "`slackware64-compat32`" n'est pas à jour.

Considérations sur Java

Slackware installait habituellement un binaire *Java Run-time Engine* dans les distributions avant la Slackware 14.0 (les binaires de JRE provenaient dans les premiers temps de Sun et plus tard de chez Oracle qui avait acheté la société Sun).
Mais Oracle modifia la licence de redistribution de telle sorte que Slackware (exactement comme toutes les autres distributions) n'était plus autorisée à fournir ces binaires Java dans la distribution.
Quand vous procédez à une *mise à niveau du système* vers Slackware 14.0, une ancienne version du *JRE* reste dans votre arborescence. Cette version "6u25" présente plusieurs failles de sécurité critiques, il vous faut donc l'enlever manuellement de votre ordinateur aussi vite que possible par l'appel de la commande:

```
removepkg jre
```

Si vous avez besoin de Java, jetez donc un coup d'œil dans le répertoire "`/extra/source/java`" de la version Slackware 14 de notre distribution. Vous y trouverez un script qui permet de créer un

paquet Slackware à partir du logiciel le plus récent de Java fourni par l'éditeur Oracle, que vous installerez ensuite en utilisant la commande “`installpkg`”. Voyez aussi notre article de Wiki “Java in Slackware”

Sources

- Originally written by Eric Hameleers

howtos, slackpkg, author alienbob, translator pierreaverseng
[1)]

La gestion des paquets dans Slackware n'est pas fondée sur la notion de “version inférieure” ni “version supérieure”. Les outils pour gérer les paquets tiennent uniquement compte de “*version différente*” et de ce fait “rétrograder” des paquets (revenir à une version antérieure) est tout aussi aisé que mettre à niveau vers une version plus récente!