Travaux pratiques

1. Schéma de partitionnement

Ce TP a pour but de déterminer le meilleur schéma de partitionnement possible, ce qui représente la plus grande difficulté pour un débutant. Le TP convient pour toutes les distributions. Vous disposez sur votre PC personnel d'un disque de 160 Go dont 40 sont déjà occupés par un autre système. Votre machine dispose de 2 Go de mémoire vive. Il vous reste 120 Go d'espace disque. Comment pouvez-vous les répartir, sachant que vous voulez séparer vos données du système ?

1. Quelle doit être la taille de la partition d'échange SWAP ?
   La partition d'échange est utilisée lorsque Linux ne dispose plus d'assez de place en mémoire vive pour traiter toutes ses données. Les données sont déplacées en mémoire virtuelle sur cette partition d'échange pour libérer plus ou moins temporairement de la mémoire pour d'autres données.
   Votre système disposant de 2 Go de mémoire, vous pouvez ne prévoir que 2 Go de swap. Il reste alors 118 Go d'espace disque pour la suite.

2. Quelle place réserver au système / ?
   Même si vous deviez installer tous les produits présents sur un DVD d'installation, le total n'atteindrait pas 10 Go. Mais deux choses doivent attirer votre attention : vous pouvez rajouter des produits issus d'autres sources (nouveaux dépôts, installation manuelle, etc.) par la suite, et les répertoires /var et /tmp peuvent être amenés à grossir. Disposant d'assez de place, pourquoi ne pas réserver 20 ou 30 Go à la racine ? Partez par exemple sur 20 Go.
   Il reste 98 Go.

3. Quelle place réserver au /home ?
   La partition qui contient /home est celle qui contient vos données, ou celle des autres utilisateurs. C'est elle qui occupe le plus de place, entre les photos, la musique, les films, les documents de travail, etc. Réservez les 98 Go restants. Le disque est entièrement partitionné.

4. Est-il utile de créer une partition étendue ?
   Vous avez trois partitions à créer sur le disque, en plus de celle qui existe déjà soit en tout quatre partitions. C'est le nombre exact pour prévoir quatre partitions primaires. Mais pensez que vous pouvez avoir besoin de réduire, supprimer ou recréer des partitions. Dans ce cas, la limite est déjà atteinte. Soyez prévoyant et créez une partition étendue où créer des partitions logiques.

5. Quel est le schéma final du disque ?
   Partition primaire 1 : l'OS déjà présent, 40 Go.
   Partition étendue :
   - Partition logique 1 : /, 20 Go.
   - Partition logique 2 : /home, 98 Go.
   - Partition logique 3 : swap, 2 Go.

   Les 160 Go sont tous occupés.

2. Gestion des RPM

Le but de ce TP est de travailler sur la base RPM des packages déjà installés sur votre poste et d'en installer de nouveaux. Le poste, ou une machine virtuelle, doit disposer d'une distribution basée sur RPM (Red Hat, Fedora, Mandriva, openSUSE, etc.).

1. Déterminez le nombre de packages RPM actuellement installés sur votre poste de travail.
   Réponse :
   Lancez la commande suivante :

```
$ rpm -qa | wc -l
```

Sur le poste de l'auteur elle retourne : 1684 !

2. Vérifiez que le package coreutils est bien présent sur votre système, puis déterminez à quoi il sert à l'aide de sa description. Pouvez-vous faire en sorte de n'obtenir que la description et rien d'autre ? Lisez la page du manuel pour en savoir plus.
Dans un premier temps interrogez la base RPM sur ce package pour en obtenir les informations :

```
$ rpm -qi coreutils
```

En cas d'erreur, le package n'est sûrement pas installé. Sinon, lisez le contenu du champ Description.
Dans un second temps, lisez la section du manuel de rpm consacrée au format de sortie. Le paramètre `-q` accepte un format de sortie que vous pouvez formater avec `--queryformat`. Le format se spécifie ainsi : %{CHAMP} avec le champ en majuscules :

```
$ rpm -q --queryformat=%{DESCRIPTION}
```

3. Essayez de supprimer le package coreutils. Pouvez-vous fournir la liste des dépendances qui vous en empêche ?
Tentez en tant que root :

```
# rpm -e coreutils
```

Vous obtenez la liste de tous les packages qui empêchent sa désinstallation : plusieurs centaines ! Notez l'existence du paramètre `-R` qui affiche de quoi dépend le package lui-même, et le --provides qui fournit le nom des éléments fournis par le package.

```
$ rpm -q --provides coreutils
fileutils
sh-utils
stat
textutils
coreutils = 6.9-43
```

4. Téléchargez le package RPM du JRE (*Java Runtime Environment*) de Java présent sur http://www.java.com/en/download/manual.jsp (prenez la version correspondant à votre architecture). Installez ce package en indiquant le nom du package et une barre de progression.
Installez le package avec les paramètres `-i`, `-v` et `-h` :

```
# rpm -ivh jre-7u60-linux-x64.rpm
```

5. Si le package était déjà installé, comment auriez-vous pu le mettre à jour ? Sachant qu'il est déjà installé maintenant, tentez de mettre à jour ce package de manière inconditionnelle. Dans quel cas cela peut-il être nécessaire ? Enfin, supprimez-le.
Vous pouvez mettre à jour le package avec les paramètres `-U` ou `-F`. Notez que vous auriez pu installer le package directement avec `-U` :

```
# rpm -Uvh jre-7u60-linux-x64.rpm
```

Simplement si le package est déjà installé dans la même version cela ne marche pas. Vous pourriez avoir besoin de le faire si des fichiers de ce package ont été supprimés : leur suppression, même complète, ne supprime pas le rpm de la base locale. Indiquez l'option `--force`.

```
# rpm -Uvh --force jre-7u60-linux-x64.rpm
```

3. Gestion de DPKG et APT

Le but de ce TP est de travailler sur la base dpkg des packages déjà installés sur votre poste, d'en installer de nouveaux et d'utiliser APT. Le poste, ou une machine virtuelle, doit disposer d'une distribution de type Debian ou Ubuntu.

1. Répondez aux questions 1 à 4 du TP précédent, mais avec les commandes et packages DPKG équivalents : coreutils est présent sous le même nom. Par contre il n'y a pas de package Java, vous allez utiliser le package de Skype que vous récupérez ainsi :

```
wget -O skype-install.deb http://www.skype.com/go/getskype-linux-deb
```

a - La liste des packages installés doit être filtrée. Par défaut dpkg fournit la liste de tous les paquets connus, dont ceux installés. ils commencent par « ii » :

```
$ dpkg -l| grep ^ii | wc -l
```

Sur la machine de test de l'auteur il y a 695 packages installés.

b - L'option -l de dpkg peut prendre un filtre comme paramètre :

```
$ dpkg -l "*coreutils*"
```

Il est possible que vous trouviez deux packages de ce nom, aussi vous devrez soit lire les résultats, soit rechercher une correspondance exacte :

```
$ dpkg -l coreutils
```

Pour obtenir les détails du package déjà installé, il vous faut aller dans le manuel qui vous informe qu'il est possible d'utiliser la commande **dpkg-query** et le paramètre `-W` :

```
$ dpkg-query -W coreutils
```

Mais il manque la description. Le manuel de dpkg-query fournit une information supplémentaire : vous pouvez modifier le format de sortie avec le `-f` :

```
$ dpkg-query -W -f='${Description}' coreutils
```

c - Pour supprimer un package Debian, utilisez l'option `-r` :

```
# dpkg -r coreutils
```

Vous allez obtenir des erreurs :

```
dpkg : erreur de traitement de coreutils (--remove) :
C'est un paquet indispensable - il ne doit pas être supprimé.
Des erreurs ont été rencontrées pendant l'exécution :
coreutils
```

Pour savoir ce que fournit coreutils, utilisez à nouveau dpkg-query :

```
#  dpkg-query -W -f='${Provides}' coreutils
textutils, shellutils, fileutils
```

d - Pour installer un package Debian, utilisez le paramètre `-i` :

```
# dpkg -i skype-install.deb
```

Comme indiqué dans ce chapitre il n'y a pas de méthode directe équivalente à rpm pour la mise à jour d'un package. Si le package est déjà installé le -i va le mettre à jour. C'est à vous de vérifier avant si celui-ci est vraiment installé (voyez pour cela la réponse à la première question).

2. APT est un gestionnaire de meta-packages : il gère les dépendances à votre place et travaille sur des dépôts et non plus sur des packages individuels. Supertuxkart (un jeu de kart en 3D) est présent dans le dépôt des backports. Le but est de gérer ce dépôt et les installations associées.

a - Rajoutez la ligne « deb http://http.debian.net/debian wheezy-backports main contrib non-free » dans le fichier des dépôts. Quelle est l'URL du dépôt main pour une architecture amd64 ?

b - Mettez à jour la base de données locale de APT.

c - Installez uniquement Supertuxkart.

d - Mettez à jour l'intégralité de votre distribution avec les éventuels nouveaux packages disponibles au sein de vos dépôts.

Réponse :

a - Vous devez modifier le fichier /etc/apt/sources.list et rajouter dedans la ligne « deb http://http.debian.net/debian wheezy-backports main contrib non-free ». L'url complète du dépôt contrib pour les i386 est http://http.debian.net/debian/dists/wheezy-backports/contrib/binary-amd64/.

b - Mettez à jour la base locale APT avec la commande suivante :

```
# apt-get update
```

c - installez uniquement Supertuxkart ainsi :

```
# apt-get install supertuxkart
```

Remarquez que contrairement à la première question, APT gère les dépendances et va installer tuxpaint ainsi que les dépendances associées.

d - Mettez à jour votre système avec :

```
# apt-get upgrade
```

4. Les sources

Ce TP vous propose de compiler les sources d'un gestionnaire de fichiers KDE appelé Beesoft Commander disponible sur http://kde-apps.org/content/show.php/Beesoft+Commander?content=37435

1. Téléchargez les sources au format tar.gz dans votre répertoire personnel, décompressez l'archive résultante. Renommez le répertoire des sources en bsc_sources et rentrez dans le répertoire des sources.

Le lien de téléchargement pointe sur le site beesoft. Placez le fichier dans votre répertoire personnel. Pour décompresser les sources tapez :

```
$ tar xvzf bsc_4.1.0_src.tar.gz
```

Renommez le répertoire résultant de la décompression :

```
$ mv bsc bsc_sources
```

Entrez dans le répertoire :

```
$ cd bsc_source
```

2. La compilation de ce produit se lance par un script appelé ./install.pl. Configurez les sources pour une installation dans /usr/local et lancez la compilation.
   Éditez le fichier install.pl. Au début remplacez les chemins /usr par /usr/local. Sauvez. Puis exécutez la commande :

   ```
   $ ./install.pl
   ```

   Après la compilation, install.pl tente d'installer le produit dans le chemin indiqué. Mais ça ne marche pas car seul root en a le droit. Il va donc installer le produit dans $HOME/bsc, pour votre utilisateur. Pour l'installer pour tous, il aurait fallu compiler bsc avec les droits root, ce qui n'est pas normal.

3. Si bsc est correctement compilé, il doit se lancer seul. Dans le cas inverse, qu'a-t-il pu se passer ?
   BSC dépend des bibliothèques de développement QT4. Sont-elles bien installées sur votre machine ? Si non, utilisez votre gestionnaire de package pour le faire, et relancez la compilation.

5. Bibliothèques partagées

Dans ce TP, vous allez gérer les bibliothèques partagées liées au programme compilé dans le TP précédent.

1. Regardez à quelles bibliothèques partagées est lié le programme bsc. Trouvez-vous une bibliothèque appelée libQT4Gui.so.4 ?
   La commande **ldd** permet de voir toutes les bibliothèques partagées utilisées par un programme :

   ```
   $ ldd bsc
   ```

   Isolez la ligne recherchée :

   ```
   $ ldd bsc | grep -i qtgui
   ```

   Une seule ligne doit être retournée. Sur la machine de test la bibliothèque est présente dans /usr/lib.

2. Partant du principe qu'aucun programme n'utilise la bibliothèque partagée libQT4Gui.so.4, en tant que root allez dans son emplacement et déplacez tous les fichiers associés dans /tmp/lib. Exécutez ensuite la commande **ldconfig**. Pouvez-vous lancer bsc ? Expliquez.
   Créez le répertoire /tmp/lib :

   ```
   # mkdir /tmp/lib
   ```

   Déplacez-vous dans /usr/lib (ou le chemin de la bibliothèque) :

   ```
   # cd /usr/lib
   ```

   Déplacez les fichiers :

   ```
   # mv libQtGui.so* /tmp/lib
   ```

   Exécutez bsc. Il ne se lance pas :

   ```
   # ./bsc
   ./bsc: error while loading shared libraries: libQtGui.so.4: cannot
   open shared object file: No such file or directory
   ```

   La commande **ldd** vous informe de la disparition de la bibliothèque déplacée :

   ```
   libQtGui.so.4 => not found
   ```

3. Éditez le fichier de configuration /etc/ld.so.conf et rajoutez-y le chemin /tmp/lib. Exécutez à nouveau ldconfig puis relancez bsc. Que se passe-t-il ?

En rajoutant le chemin /tmp/lib puis en mettant à jour le cache du chargeur dynamique avec ldconfig, la bibliothèque est de nouveau accessible.

```
# vi /etc/ld.so.conf
# ldconfig
# ldd ./bsc
libQtGui.so.4 => /tmp/lib/libQtGui.so.4 (0xb77bc000)
```

Rajoutez /tmp/lib dans une nouvelle ligne et sauvez le fichier.

Le programme fonctionne à nouveau.

4. Remettez tout dans l'état initial.

Déplacez la bibliothèque vers sa position d'origine, supprimez /tmp/lib de ld.so.conf et relancez ldconfig. Supprimer /tmp/lib.