Atelier3 - Gestion des packages et des bibliothèques partagées

Exercice1 : Le gestionnaire de packages dpkg

dpkg est l'utilitaire de base sous Debian, pour installer, supprimer, configurer ou obtenir des informations sur des paquets (.deb).

- Pour lister tous les packages Debian connus par le système, lancer la commande : **dpkg –l**
- Les lignes commençant par ii correspondent à des packages déjà installés. Pour afficher uniquement les packages déjà installés, donc uniquement les lignes commençant par **ii**, exécuter la commande : **dpkg -l | grep "^ ii"**
- Pour compter le nombre de ces lignes, lancer la commande : **dpkg -l | grep "^ ii" | wc -l**

1. Déterminer le nombre de packages déjà installés ?

Considérons à titre d'exemple le package ***wget***. Pour afficher la liste de tous les packages dont le nom contient ***wget***, taper la commande : **dpkg -l "* wget *"**

Lancer maintenant la commande : **dpkg -l wget**

2. Le package est-il installé ? Noter sa version et la description

Pour lister tous les fichiers qui ont été installés depuis le packages **wget**, en entrant la commande : **dpkg –L wget**

3. Lister maintenant tous les fichiers qui ont été installés depuis le packages **coreutils**. A partir des résultats obtenus, donner le rôle de ce paquet.

Pour supprimer un package Debian, utilisez la commande **dpkg** avec l'option **–r** en tant que **root**.

4. Supprimer le paquet **coreutils** et vérifier résultats obtenus.

Soit à installer le package **openssh-server**.

5. Télécharger le package **openssh-server** Sous forme **deb** avec la commande :
   wget http://ftp.fr.debian.org/debian/pool/main/o/openssh/openssh-server_7.3p1-2_i386.deb
6. Ajouter les droits d'exécution au fichier deb : **chmod u+x openssh-server_7.3p1-2_i386.deb**
7. Installer-le avec la commande : **sudo dpkg –i openssh-server_7.3p1-2_i386.deb** et vérifier résultats obtenus.

dpkg est un outil qui ne gère pas les **dépendances** : il sait ce qui est installé sur le système et ce qu'on lui indique en ligne de commande, mais, n'ayant aucune connaissance de tous les autres paquets disponibles, il échouera si une dépendance n'est pas satisfaite.

Exercice2 : L'outil APT

1. Éditer le fichier **/etc/apt/sources.list**, mettre en commentaire (un commentaire est une ligne commençant par #) les lignes déjà présentes, et ajouter les lignes suivantes :

```
deb http://fr.archive.ubuntu.com/ubuntu/ trusty main restricted
deb http://security.ubuntu.com/ubuntu trusty-security main restricted
deb http://fr.archive.ubuntu.com/ubuntu/ trusty-updates main restricted
```

Les lignes ci-dessus décrivent tous les dépôts de packages binaires, officiellement supportés par Debian pour sa version **trusty**.

2. Pour mettre à jour la base de données locale de APT (l'ensemble des packages connus avec leurs versions actuelles), lancer la commande : **sudo apt-get update**

Soit à installer le package **iftop** permettant de visualiser le trafic réseau sur une interface donnée.

3. Lancer la commande : **sudo apt-get install iftop**

Remarquer que contrairement à **dpkg**, **APT** gère les dépendances : il installe **iftop** ainsi que les dépendances associées.

4. Lancer **iftop** sur l'interface eth0 avec la commande : **sudo iftop -i eth0**
5. Pour mettre à jour tous les packages installés sur le système, lancer la commande : **sudo apt-get upgrade**

Exercice3 : Gestion des bibliothèques partagées

Lors du développement d'un logiciel (libre), il est souvent inutile d'utiliser des algorithmes de gestion mémoire, ou pour utiliser les méthodes d'accès aux fichiers. Ces algorithmes et méthodes sont souvent regroupés sous forme de bibliothèques que d'autres programmes peuvent utiliser.

Dans le cas de la compilation d'un programme, il est utile de connaître les bibliothèques partagées nécessaires à ce programme. Sous GNU/Linux il existe une commande: **ldd**.

1. Déterminer à quelles bibliothèques partagées est lié le programme apt-get : **ldd /usr/bin/apt-get**

Notons que le programme dépend du fichier de bibliothèque **/usr/lib/i386-linux-gnu/libapt-pkg.so.4.12** qui est en fait un lien symbolique vers le fichier **/usr/lib/i386-linux-gnu/libapt-pkg.so.4.12.0**

2. Créer le répertoire /tmp/lib par la commande : **mkdir /tmp/lib**
3. Déplacer /usr/lib/i386-linux-gnu/libapt-pkg.so.4.12.0 dans /tmp/lib/ de sorte qu'il ne puisse pas être trouvé dans son répertoire **/usr/lib/i386-linux-gnu/** par la commande :
   sudo mv /usr/lib/i386-linux-gnu/libapt-pkg.so.4.12.0 /tmp/lib/
4. Lancer maintenant le programme **apt-get** par la commande suivante et noter le message d'erreur : **sudo apt-get update**

Une partie du code du programme **apt-get**, contenue dans la bibliothèque partagée **libapt-pkg.so.4.12.0** est manquante. Le programme **apt-get** ne peut donc pas être exécuté.

5. Éditer le fichier **/etc/ld.so.conf** et ajouter la ligne **/tmp/lib**
6. Pour tenir compte de cette nouvelle configuration, régénérer le cache du chargeur dynamique par la commande : **sudo ldconfig**
7. Relancer le programme **apt-get** et noter qu'il fonctionne : **sudo apt-get update**
8. Remettre tout dans l'état initial en procédant comme suit :
   a. Remettre le fichier de bibliothèque **libapt-pkg.so.4.12.0** dans son répertoire original :
      sudo mv /tmp/lib/libapt-pkg.so.4.12.0 /usr/lib/i386-linux-gnu/
   b. Supprimer la ligne **/tmp/lib** du fichier **/etc/ld.so.conf**
   c. Reconstruire le cache du chargeur dynamique par la commande : **sudo ldconfig**