Système de fichiers réseaux et installation d'applications

But du TP

Durant cette manipulation nous allons étudier et configurer des systèmes de fichiers réseau `nfs` et samba puis étudier les principales commandes pour installer des paquets Debian.

Introduction

Nous allons obligatoirement utiliser une machine virtuelle neuve obtenue avec le disque Debian7-32b.vdi en mode partagé. Il y a déjà pas mal de logiciel déjà installé sur ce disque.

I. Réglage du réseau

1. Quelques règles pour bien fonctionner

Le logiciel `network-manager` a été retiré de la machine. Nous allons revoir et étendre un peu les connaissances acquises en M1101.

Nous prendrons comme règles :

- Le réglage statique se fera au moyen du fichier `/etc/network/interfaces`, l'activation et l'arrêt de l'interface se faisant par `ifup eth`**`N`**`, ifdown eth`**`N`**.
- Le réglage dynamique sera fait exclusivement par la commande `ip`.

2. On examine le réglage actuel

Tout le TP se passe bien évidemment sur la machine virtuelle, et les questions posées s'y rapportent.

En scrutant le compte rendu du démarrage de la machine (cf. TP2) quelles sont vos cartes réseau et sur quelles interfaces ?

Sur quelle interface physique `eth`**`N`** est connectée l'interface virtuelle `eth`**`M`** ?

Une façon simple sur une machine physique de connaître l'état des interfaces est de regarder le fichier de log (d'historique) du système :

```
# tail -f /var/log/syslog
```

Lorsque l'on branche et débranche le câble il apparaît un message dans `syslog` et un autre quand on rebranche.

Tester cet effet sur la machine virtuelle, en débranchant et rebranchant le câble virtuel (depuis l'interface virtualbox) à chaud. Quels messages apparaissent dans le syslog ?

3. Passage en statique

Même si cela ne présente aucun intérêt pour la suite du TP, nous allons passer l'interface `eth1` de `dhcp` en adressage statique. Pour cela nous allons éditer le fichier `/etc/network/interfaces` (inspirez-vous de l'exemple ci-dessous).

Le paquet « `resolvconf` » ayant été installé, il va utiliser les lignes `dns-` pour configurer le fichier `/etc/resolv.conf` en indiquant les serveurs de noms et les domaines de recherche des noms d'hôtes incomplets.

Faites les modifications nécessaires dans le fichier et indiquez les dans votre rapport.

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
#allow-hotplug eth0
#iface eth0 inet dhcp

auto eth1
iface eth1 inet dhcp

#iface eth0 inet static
#       address 10...
#       netmask 255.255.255.0
#       gateway 10...254
#       dns-nameservers 10.4.110.250
#       dns-search unice.fr gtr.tp salle410.neticien.tp
```

Listing 1: Fichier interfaces d'origine

Il faut maintenant tester (car actuellement, votre interface est toujours en dhcp).

On arrête l'interface eth1 :

```
# ifdown eth1
```

On la réactive :

```
# ifup eth1
```

Vérifiez que l'adresse, le masque, le broadcast, et la passerelle sont les bons.

```
# ip addr
# ip route
```

Vérifiez que vous pouvez accéder aux serveurs de noms locaux et distants :

```
# host centrex
# host www.google.com
```

Remettez votre interface en dhcp

4. Utilisation commande `ip`

Tapez la commande :

```
# ip addr add 192.168.1.N/24 broadcast 192.168.1.255 dev eth1
```

où **N** est le numéro de votre machine.

Que voyez-vous dans votre table de routage ?

Comparez ce que voit ifconfig et ce que voit ip lorsque vous listez les interfaces ?

Testez que vous pouvez contacter d'autres machines dans ce nouveau sous-réseau.

On enlève cette adresse:

```
# ip addr del 192.168.1.N/24 dev eth1
```

II. Les systèmes de fichiers réseau

1. NFS

NFS est le système de fichiers réseau natif de tous les systèmes `unix`. NFS permet donc de se monter un système de fichier (une partition), comme fait au TP précédent, mais au travers d'un réseau. Cette partition s'intègrera à votre arbre global et lorsque vous vous déplacez de répertoire en répertoire, vous ne verrez pas de différence entre être dans une partition ext3 ou une partition NFS.

a) Sur le serveur

Il y a bien sûr un aspect client et un aspect serveur (les deux machines virtuelles de votre binome). Côté client, il suffit d'un simple « `mount` » que nous reverrons donc quand le serveur sera opérationnel.

Le serveur NFS est constitué d'un démon (=un service) principal `nfsd` qui « `fork` » c'est-à-dire qui se subdivise en plusieurs sous-démons qui servent les requêtes. Par défaut il travaille en UDP mais on peut aussi travailler en TCP. Il est impératif d'avoir aussi démarré le démon `portmap` qui va gérer les requêtes `rpc (remote procedure call)`.

Le mieux pour tout démarrer correctement c'est d'utiliser la procédure normale.

```
# /etc/init.d/nfs-kernel-server restart
```

Mais pour l'instant le serveur ne sert rien. Il faut indiquer ce qui est exporté et vers qui. Il existe plusieurs syntaxes d'exportation. Tout se trouve dans le fichier « `/etc/exports` ».

Mettre dans ce fichier la ligne (adaptez l'adresse à votre réseau si besoin- attention, il n'y a pas d'espace entre l'adresse réseau et les options)

```
/usr/local 10.4.105.0/24(rw,sync,no_subtree_check)
```

Sauvegarder le fichier, puis taper dans un terminal

```
# /etc/init.d/nfs-kernel-server reload
```

Ce qui rend l'exportation effective.

On peut vérifier cette exportation par un

```
# showmount -e
```

Que signifie la ligne mise dans le fichier "`/etc/exports`" ?

b) Sur le client

On a constaté que l'exportation n'est contrôlée que par le numéro IP du client. **Rien ne prouve qu'il s'agit de la bonne machine.** Nous allons passer outre ses problèmes de sécurité et nous concentrer sur la mise en œuvre de ce service. Les versions plus récentes de NFS permettent de résoudre ces problèmes.

On démarre le client c'est-à-dire les démons `nfsiod`.

```
# /etc/init.d/nfs-common restart
```

Normalement le montage `nfs` système est fait au démarrage et les requêtes sont paramétrées dans `/etc/fstab`.

Montons à la main, la partition exportée précédemment

```
# mount nomDuServeur:/usr/local /mnt/serv1
```

Remarque : il faut que le répertoire local `/mnt/serv1` ait été créé précédemment

Copier des fichiers entre les 2 machines sur `/mnt/serv1` . Est-ce que cela fonctionne avec le login rt?

Est-ce que cela fonctionne en étant root sur le client ?

Maintenant, modifiez le fichier d'exportation sur le serveur en demandant que `root` soit exporté en tant que `root`. Cela se fait en ajoutant l'option `no_root_squash`

Est-ce que cela fonctionne maintenant en rt ? Même question en root ?

NFS peut être utilisé pour stocker sur un serveur central les données des comptes utilisateurs.

Créer un utilisateur nfs_client sur la machine avec la commande useradd sur le client (il n'y aura pas de homedir pour le client) et adduser sur le serveur (il va y avoir création d'un homedir)

Partagez le homedir (répertoire de base) de nfs_client en rw sur le serveur. Faites le sans `no_root_squash.`

Sur le client, montez le répertoire /home/nfs_client en nfs (il faut être root) sur le répertoire de base local de nfs_client à savoir /home/nfs_client.

Vérifiez que cela fonctionne en écrivant un fichier toto.txt. Est-ce que cela fonctionne en root ?

Que vous indique la commande suivante ?

```
df .
```

2. Partages de fichiers avec Samba

Samba est un projet libre d'exportation de services fichiers et d'imprimantes qui fonctionne sur la plupart des Unix. Il met à disposition de ces systèmes, les protocole `smb` et `cifs`, qui relient les machines Windows entre elles. Il est possible d'utiliser Samba comme Contrôleur Principal de Domaine au sens NT4 (on verra ces notions dans les TPs Windows Servers).

a) Création du serveur

La configuration du serveur samba est faite dans le fichier `/etc/samba/smb.conf`. On y trouve plusieurs zones : `[global]` pour la configuration générale, `[homes]` pour la configuration des services liés à l'exportation du répertoire de `login` d'un client, `[public]` pour les accès ouverts à tous, `[printers]` pour le partage d'imprimantes.

Le nombre d'options est très important (d'où la taille du fichier `smb.conf`), nous n'indiquons ici que ce qui nous semble nécessaire.

b) Exportation du système de fichiers et droits

Avant de pouvoir utiliser samba il faut l'installer.

```
# apt-get install samba smbclient cifs-utils
```

Modifier dans `smb.conf` (sur le serveur seulement bien sûr) le paramètre `workgroup` en faisant apparaître votre nom.

~~Avec `findsmb` visualisez les serveurs présents sur le réseau.~~

```
# findsmb
```

Avec `smbclient` visualisez les exportations

```
# smbclient -L nomDuServeur -N
```

Quel est le rôle de l'option -N ici?

- ***Authentification cryptée***

Dans la partie Authentication de smb.conf, mettre `encrypt password = no` en `yes`. Vous devez constater que `rt` ne peut plus se loger. Il faut lui créer un mot de passe crypté samba en local sur le serveur.

Redémarrer le serveur.

```
# /etc/init.d/samba restart
```

Puis créez le mot de passe samba pour rt :

```
# smbpasswd -a rt
```

et mettre le bon mot de de passe lors du login (pas celui de Linux si ils sont différents). Celui-ci est stocké ailleurs que dans la base Unix.

Vérifiez le bon fonctionement avec

```
# smbclient -L nomDuServeur -U rt
```

- ***Montages samba***

Nous allons permettre différents types d'accès à différents répertoires. Nous voulons :

- un accès de chaque utilisateur à son propre répertoire home ;
- un répertoire public dans lequel on pourra se loger en anonyme (sans mot de passe).
- Permettre aux machines windows de visualiser les répertoires home et le répertoire public.

Pour cela, il faut modifier la partie « Share Definition » du fichier smb.conf. Astuce : dans l'éditeur de texte nano, utilisez la séquence Crtl+W pour faire une recherche par mot-clef.

Voici quelques informations à connaître pour configurer correctement cette partie :

- les commentaires de ce fichier sont les lignes qui commencent soient par des' #', soit par des ';'
- la partie [homes] correspond à l'ensemble des répertoires homes des utilisateurs.
- On peut définir d'autres « shares » (zones partagées) simplement en faisant une nouvelle rubrique, par exemple en ajoutant les lignes :

```
[music2]
path = /media/musicbind
read only = yes
public = yes
```

- la directive browsable permet à des machines Windows de browser les répertoires du serveur.
- La directive path indique le chemin sur le serveur du « share » concerné
- guest ok = yes permet de se loger en anonyme

Une fois que vous avez modifié les paramètres du fichier, il faut dire au service de relire son fichier de configuration, ce que l'on fait classiquement avec un :
`/etc/init.d/samba reload`

Faites les modifications suivantes :

* Mettez le répertoire tmp en public et lecture-écriture

* Mettez les homedirs en « browsable » et en lecture-écriure

Montez le répertoire de l'utilisateur «rt» sur la machine cliente.

```
# smbclient //nomDuServeur/rt -U rt
```

Executez les commandes suivantes

```
# put /etc/hosts hosts
# ls
```

On peut faire un montage classique :

```
sudo mount -t cifs //nomDuServeur/rt /mnt/ -o user=rt
```

Suspendez le client, créez une machine Windows avec createvm et regarder l'exportation des répertoires partagés en allant dans un explorateur > outil > connexion lecteur réseau puis parcourir.

Arrêtez la machine windows et redémarrez le client.

III. Technique de mise à jour DEBIAN

Fréquemment durant les manipulations vous allez être obligés de mettre à jour certaines applications voire de les installer, pour ajouter des fonctionnalités. Nous allons voir les principales commandes nécessaires au quotidien.

1. Les principes

On doit considérer qu'un logiciel compilé a besoin pour fonctionner d'un certain nombre de bibliothèques qui lui apportent leurs fonctionnalités qui n'ont pas été redéveloppées pour chacun des logiciels séparément. On introduit donc la notion de dépendance. Cependant chacune des composantes des logiciels évolue et gagne ou perd des fonctionnalités au cours des changements de version. On obtient ainsi un arbre de dépendances qui prend en compte les différents éléments logiciels et leur version.

La mise à jour d'un petit logiciel peut entraîner la mise à jour de plusieurs gros logiciels. De même le retrait d'une bibliothèque de base (`qt`, `Xlib`, …) devrait entraîner le retrait de tous les logiciels qui en dépendent car ils ne pourront plus fonctionner correctement.

On a aussi 2 stratégies bien distinctes : la distribution compilée et la distribution des sources qui seront compilées par l'utilisateur final. Cela conduit à 2 arbres de dépendances distincts. Vous noterez aussi que chaque fois que l'on installe un logiciel, on modifie l'arbre existant localement ce qui conduit à devoir recalculer l'arbre de manière dynamique.

2. Les commandes globales

Bien qu'elles soient nombreuses, nous utilisons la plupart du temps toujours les mêmes commandes car elles sont très hiérarchisées afin d'en simplifier l'usage.

a) apt-get

La commande reine est `apt-get` : c'est elle qui gère l'arbre de dépendance et qui va retrouver tous les paquets nécessaires à l'installation. Elle est la plupart du temps utilisée pour gérer les paquets compilés les fameux `.deb`.

Cette commande admet plusieurs sous-commandes:

- `update` pour mettre à jour la description des paquets,
- `install` pour (aller je le dis) installer les nouveaux paquets (ou les mettre à jour),
- `upgrade` pour mettre à jour des paquets,
- `remove` pour désinstaller le paquet et tous les fichiers qui lui appartiennent
  - pour effacer aussi les fichiers de configuration il faut ajouter `--purge`

Toute séquence d'installation doit commencer par un `apt-get update.`

Pourquoi l'apt-get update est-il fortement recommendé?

b) `Le fichier sources.list`

Afin de pouvoir retrouver sur l'ensemble des miroirs de la distribution debian les dernières versions disponibles, le fichiers `/etc/apt/sources.list` décrit ce que l'on veut pouvoir intégrer dans son système.

```
# La partie due à l'installation; c'est le cdrom
# Commentée

#deb cdrom:[Debian GNU/Linux wheezy-DI-b2 _Wheezy_ - Official Snapshot
amd64 NETINST Binary-1 20120905-23:05]/ wheezy main

deb http://kheops.unice.fr/debian/ wheezy main non-free contrib
deb-src http://kheops.unice.fr/debian/ wheezy main non-free contrib

deb http://security.debian.org/ wheezy/updates main contrib non-free
deb-src http://security.debian.org/ wheezy/updates main contrib non-free
```

Listing 2: Le contenu du fichier `sources.list`

Il peut y avoir autant de lignes qu'on le souhaite. Ici seules 4 sont actives, les autres sont toutes commentées pour favoriser la mise à jour locale. Les problèmes de sécurité ne sont pas critiques.

Examinons les différentes lignes « `deb` »

- La première c'est celle pour le cdrom de l'installation.
- La deuxième indique le serveur local de l'IUT : qu'il faut aller la distribution « `Wheezy` » (presque stable)
  - Il y a deux types de lignes
    - `deb` pour la distribution binaires et
    - `deb-src` pour la distribution à partir des sources
  - Il y a en Debian, 3 distributions :
    - la stable (actuellement «squeeze»), qui devrait être utilisée par tous les serveurs
    - la testing (actuellement «Wheezy») et que l'on peut utiliser à ses propres risques, cependant cette distribution contient les mises à jour les plus récentes mais la cohérence de la distribution n'est pas garantie
    - l'instable (« sid ») dans laquelle les développeurs font leurs modifications
  - Ensuite il y a plusieurs sous répertoires
    - `main` : l'arbre principal qui suit les règles des logiciels « libres »
    - `non-free` : paquets qui sont librement distribuables mais ne suivent pas la pratique « libre » comme par exemple, une licence interdisant la commercialisation,…
    - `contrib` : suit les règles libres mais dépend d'un paquet « non-libre »
- Ensuite on trouve les deux lignes qui indiquent où aller chercher les paquets pour les mises à jour de sécurité.

Il peut y avoir plusieurs lignes pour plusieurs serveurs

Nous allons ajouter des dépôts pour un logiciel spécifique : virtualbox. On peut voir sur la page de virtualbox comment procéder : https://www.virtualbox.org/wiki/Linux_Downloads rubrique Debian.

Il faut ajouter les dépots dans le fichier sources.list.

```
deb http://download.virtualbox.org/virtualbox/debian wheezy
  contrib
```

puis installer une clef d'identification :

```
wget -q
  http://download.virtualbox.org/virtualbox/debian/oracle_vbox.as
  c -O- | sudo apt-key add -
```

Faites un apt-get update à la suite. Combien de nouveaux paquets sont apparus (ils n'ont pas été téléchargés ni installés bien sûr) ?

c) `apt-cache`

Pour retrouver le bon nom de paquet qui nous intéresse il faut scruter dans la liste complète des paquets. C'est le rôle de

```
# apt-cache search nomAppochéDuPaquet
```

Combien de paquets concernent samba?

3. Ce qu'il y a dans un paquet

Un paquet, c'est un ensemble de fichiers (source ou binaire) à installer sur votre machine. Un paquet est donc une archive. Mais un paquet, c'est aussi un ensemble de scripts qui vont s'exécuter durant l'installation. C'est aussi des informations sur l'arbre de dépendance : de quel(s) paquet(s) dépend un paquet donné. Ce sont ses parents. C'est aussi un ensemble d'enfants : qui dépend de ce paquet. Mettons en pratique avec un paquet simple d'un logiciel réseau, tcpdump.

Commençons par voir la description de ce paquet

```
apt-cache show tcpdump
```

Listons les parents avec apt-cache depends tcpdump

```
apt-cache depends tcpdump
```

Combien y en a-t-il ?

Combien d'enfants à tcpdump ?

```
apt-cache rdepends tcpdump
```

Enfin, on peut lister les fichiers qui seront installés si on installe tcpdump

```
dpkg -L tcpdump
```

Enfin, pour voir que différents programmes peuvent s'exécuter durant l'installation d'un paquet prenons un exemple d'un module noyau :

Nous allons installer un nouveau module pour gérer un modem sl-modem sur la machine. On installe les sources du pilote et quelques outils.

```
# apt-get install sl-modem-source
```

Que se passe-t-il ?

Le programme Debian (`dkms`) de gestion des modules, vient de récupérer les sources, compiler, faire l'édition des liens, installer le module et le charger en mémoire.

Vérifier que le module a été chargé (cf. TP2) avec `dmesg` et `lsmod`.

4. Les commandes de deuxième niveau

Avec `apt-get`, on gère globalement au niveau de l'ensemble de la distribution des logiciels. Si l'on veut n'agir que sur un paquet, on utilisera plutôt la commande `dpkg` et toutes ses options et déclinaisons.

Pour connaître

- la liste des paquets installés dont le nom ressemble à une chaîne (ici qui finissent par `utils`)

```
# dpkg -l *utils
```

- le paquet qui contient un fichier dont on connaît un bout du nom

```
# dpkg -S mailing
```