Quelques bases d'administration système et réseau sous Linux

Michel Billaud <billaud@info.iuta.u-bordeaux.fr> Avril 2002

Ce document explique quelques notions de base sur l'administration de machines Unix isolées ou en réseau. Ces notions sont montrées à travers Linux. ($Revision: 1.10 $, $Date: 2002/04/28 21:54:05 $)

Table des matières

1 Objectifs ?

De nos jours l'installation d'un système Linux (ou Unix en général) se fait simplement en insérant un CDROM et en répondant à quelques questions. Et l'administration de ce système ne semble guère plus compliquée : on clique sur quelques icônes, et on remplit des boîtes de dialogues. C'est très pratique.

Mais il y a quand même un inconvénient : on ne comprend pas vraiment ce qu'on fait, ni pourquoi, et on n'est pas sûr que ce qu'on fait correspond à ce que qu'on voudrait. C'est normal : la "façade ergonomique" d'administration est censé proposer une vision "intuitive", des choses. Malheureusement les choses en question ne sont pas forcément fondamentalement intuitives, et on travaille donc nécessairement avec une vision erronée.[1]

Pour une grande partie des utilisateurs, ce n'est pas très grave. En cliquant au hasard, on arrive à faire marcher à peu près. Pour ce qui ne marche pas, on s'en passe. Et pour ce dont on ne peut pas se passer, on lit la rubrique trucs et astuces de son journal favori, ou on consulte un gourou UNIX.

Ce document est destiné au lecteur qui ne veut pas en rester là, qui veut essayer de comprendre ce qui se passe. On en revient donc aux bonnes vieilles lignes de commandes, et à l'emploi d'un éditeur de textes pour modifier les fichiers de configuration.

Le plan général : on regarde d'abord ce qui se passe au démarrage d'un PC sous Linux. Cela permet d'introduire quelques concepts et de la terminologie (systèmes de fichiers, montages, etc.). Cela permet aussi de comprendre ce qui doit être fait pendant l'installation, pour mettre en place un système capable de démarrer.

On en arrive ensuite aux différentes tâches usuelles d'administration d'une machine isolée (enregistrer des utilisateurs, configurer le réseau, faire des sauvegardes), puis on en arrive à la mise en réseau de machines : partages de données, gestion centralisée des comptes.

A l'issue de ce cours, le lecteur devrait être capable de configurer un réseau sous Linux avec administration centralisée.

Matériel nécessaire. Pour tirer le meilleur bénéfice de ce cours, il convient de disposer de matériel réservé à cet effet[2]. Une machine de faible puissance (486 ou Pentium) suffit largement, munie d'une version légère de Linux (Recommandations : Slackware avec séries A et N, ou Debian minimaliste).

Pour les manipulations en réseau, il faut évidemment deux machines.

[1] Ceux qui affirment que le bon sens suffit pour administrer correctement un système informatique montrent qu'ils n'en ont pas vraiment. Il faut du bon sens **et** des connaissances sur le fonctionnement de ce système.

[2] On a l'esprit moins libre en bricolant un système sur lequel il y a des données ou des applications auxquelles on tient.

2 Pour commencer...

2.1 Que se passe-t-il au démarrage ?

Quand on allume une machine, il se passe une suite d'évènements qu'on peut décomposer comme suit

- Un programme contenu en mémoire morte est lancé. Ce programme fait un certain nombre de vérifications (par exemple test de la mémoire disponible, présence du lecteur de disquette, des disques etc.) en fonction des paramètres qui ont été déclarés dans le fameux *setup du BIOS* qui est enregistré dans une mémoire non volatile.
- Ce programme interroge ensuite successivement les dispositifs déclarés "bootables" dans le *setup*. Habituellement c'est la disquette puis le lecteur de CD et enfin le disque dur. Le premier qui répond (supposons que c'est le disque dur) transmet son premier secteur (le *Master Boot Record*, ou MBR), qui est chargé en mémoire, puis exécuté.
- Le MBR d'un disque contient habituellement un petit programme qui consulte la *table des partitions* du disque dur, puis va chercher le premier secteur (*Boot Record*) de la partition déclarée *active*, et l'exécute.
- Si la partition active est une partition DOS ou Windows, le secteur de boot contient une séquence d'instructions pour démarrer le système d'exploitation.
- Si le partition active est une partition Linux, le secteur de boot contient un programme *chargeur*, qui sous contrôle d'un menu permet de lancer un *noyau linux* avec un certain nombre de paramètres.
- Le noyau linux est un fichier compressé qui est chargé en mémoire, décompressé et exécuté. Sa première tâche est de détecter la présence de matériel (type de processeur, quantité mémoire, cartes installées, contrôleurs de périphériques etc.).
- Ensuite le noyau *monte* l'arborescence des fichiers et répertoires qui sera utilisée par la suite. Cette arborescence est contenue dans un *système de fichiers* enregistré sur une partition du disque. La partition à utiliser comme *racine* est indiquée en paramètre par le chargeur, ce n'est pas forcément la partition où se trouve le chargeur.
- Enfin, le noyau exécute le programme `/sbin/init`, qui s'occupera de lancer tous les scripts de démarrage, qui se chargeront d'activer les services et applications qui ont été installés, en fonction du *niveau* de fonctionnement demandé (encore un paramètre donné par le chargeur).

2.2 Un peu de terminologie

- Une *arborescence* est composée de fichiers et de répertoires (ou dossiers, classeurs, ...) qui contiennent eux-mêmes (éventuellement) des fichiers, des répertoires etc.
- La *racine* d'une arborescence est son répertoire principal, qui contient (directement ou indirectement) tous les autres éléments de l'arborescence.
- Un *système de fichiers* est le codage d'une arborescence sous forme d'une suite d'octets. [3] Les systèmes de fichiers sont généralement enregistrés sur des partitions de disques, ou sur des disquettes, des CDs, etc.
- Une *partition* est un morceau de disque. [4]
- Un *montage* est une opération qui permet d'accrocher une arborescence à un endroit choisi (le *point de montage*) de l'arborescence déjà présente. Le *montage d'une partition* consiste à accrocher l'arbores-

[3] Il y a plusieurs manières de coder, et donc il y a plusieurs types de systèmes de fichiers. Des dizaines, en fait. Le plus utilisé pour Linux est le type `ext2` à qui succèdera peut-être `ext3`. Il y a des systèmes de fichiers journalisés (comme `reiserfs` ou `jfs` d'IBM) qui résistent aux coupures de courant brutales. Linux permet aussi d'utiliser facilement les types `msdos` et `vfat` (compatibles avec MSDOS et Windows) et `iso9660` (les CD).

[4] À l'origine les disque étaient conçus pour supporter jusqu'à 4 partitions *primaires* : leur premier secteur contient donc une *table de partitions à 4 éléments*, qui indique les positions de début et de fin de la partition, son type et des indicateurs (en particulier pour savoir si elle est active). Rapidement le découpage en 4 s'est révélé insuffisant : on peut donner le type spécial "*partition étendue*" à l'une des partitions primaires, qui sera alors redécoupée en *partitions logiques*.

cence contenue dans une partition[5] sur un point de montage. Par exemple “monter la disquette” intégrera l’arborescence contenue sur le périphérique `/dev/fd0` sous le point de montage `/mnt/floppy`.

2.3 Le démarrage : `/etc/inittab`

On a vu qu’au démarrage le noyau démarrait un processus `init` qui lance les actions définies par `/etc/inittab`.

Ce qui se passe dépend de l’organisation des fichiers de démarrage. Il y a deux approches : System V et BSD.

2.3.1 Le démarrage type “System V”

Extrait d’un fichier `/etc/inittab` (distribution Debian) :

```
# The default runlevel.
id:2:initdefault:

# Boot-time system configuration/initialization script.
# This is run first except when booting in emergency (-b) mode.
si::sysinit:/etc/init.d/rcS

# What to do in single-user mode.
~~:S:wait:/sbin/sulogin

# /etc/init.d executes the S and K scripts upon change
# of runlevel.
#
# Runlevel 0 is halt.
# Runlevel 1 is single-user.
# Runlevels 2-5 are multi-user.
# Runlevel 6 is reboot.

l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6

# What to do when CTRL-ALT-DEL is pressed.
ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -h now

# /sbin/getty invocations for the runlevels.
1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
3:23:respawn:/sbin/getty 38400 tty3
...
```

Commentaires : le noyau démarre, par défaut, au niveau 2. Le processus `init` lance donc successivement
- le script `/etc/init.d/rc.S`
- la commande `/etc/init.d/rc 2`, qui démarre tous les scripts du répertoire `/etc/rc2.d` dont le nom commence par un `S` majuscule :

[5]plus précisément, l’aborescence contenue dans le système de fichiers enregistré sur la partition

```
ls /etc/rc2.d/S*
/etc/rc2.d/S10sysklogd    /etc/rc2.d/S20lpd
/etc/rc2.d/S11pcmcia      /etc/rc2.d/S20makedev
/etc/rc2.d/S12kerneld     /etc/rc2.d/S20nfs-kernel-server
/etc/rc2.d/S14ppp         /etc/rc2.d/S20openldapd
/etc/rc2.d/S19bind        /etc/rc2.d/S20postgresql
/etc/rc2.d/S19nfs-common  /etc/rc2.d/S20sendmail
/etc/rc2.d/S19nis         /etc/rc2.d/S20ssh
/etc/rc2.d/S19nscd        /etc/rc2.d/S25nfs-server
/etc/rc2.d/S20apmd        /etc/rc2.d/S89atd
/etc/rc2.d/S20autofs      /etc/rc2.d/S89cron
/etc/rc2.d/S20exim        /etc/rc2.d/S91apache
/etc/rc2.d/S20gpm         /etc/rc2.d/S99gdm
/etc/rc2.d/S20inetd       /etc/rc2.d/S99rmnologin
/etc/rc2.d/S20irmanager   /etc/rc2.d/S99wdm
/etc/rc2.d/S20logoutd     /etc/rc2.d/S99xdm
```

ces scripts sont lancés avec le paramètre `start`

– puis il démarre des consoles virtuelles sur `tty1`, `tty2`, etc.

En réalité, le répertoire `/etc/rc2.d` ne contient pas des scripts, mais des *liens symboliques* :

```
$ ls -l /etc/rc2.d/S20lpd
lrwxrwxrwx  1 root root 13 Mar 10  2001 /etc/rc2.d/S20lpd -> ../init.d/lpd
```

Chacun de ces scripts se charge de gérer un service. Il reçoit un paramètre `start`, `stop`, `restart`. Exemple, `/etc.init.d/lpd` :

```
#!/bin/sh

PATH=/bin:/usr/bin:/sbin:/usr/sbin
DAEMON=/usr/sbin/lpd
PIDFILE=/var/spool/lpd/lpd.lock

test -x $DAEMON -a -f /usr/sbin/pac || exit 0

case "$1" in
  start)
        echo -n "Starting printer spooler: lpd"
        if start-stop-daemon --quiet --stop --signal 0 \
            --pidfile $PIDFILE --name lpd
        then
                echo " already running."
                exit
        fi
        /sbin/start-stop-daemon --start --quiet --exec $DAEMON
        echo "."
        ;;
  stop)
        echo -n "Stopping printer spooler: lpd"
        if start-stop-daemon --quiet --stop --signal 0 \
                --pidfile $PIDFILE --name lpd
        then
                PID=`cat $PIDFILE`
                start-stop-daemon --quiet --stop --exec $DAEMON \
                        --pidfile $PIDFILE --name lpd
                # Now we wait for it to die
                while kill -0 $PID 2>/dev/null; do sleep 1; done
                echo "."
```

```
        else
                echo " not running.";
        fi
        ;;
  force-reload|restart)
        $0 stop
        $0 start
        ;;
  *)
        echo "Usage: /etc/init.d/lpd {start|stop|restart|force-reload}"
        exit 1
esac

exit 0
```

2.3.2 Le démarrage type BSD

Le fichier `/etc/inittab` lance quelques scripts, qui regroupent le démarrage de plusieurs services. Voir par exemple le démarrage d'un système de type Slackware : `/etc/rc.d/rc.modules` lance le chargement des *modules*, `/etc/rc.d/rc.inet1` lance les les services réseau de bas niveau (démarrage des interfaces, routes, ...), `/etc/rc.d/rc.inet.2` démarre les services de haut niveau : imprimantes, connexions, etc.

Pour modifier le démarrage BSD - par exemple pour ajouter un service- il faut éditer les scripts existants, alors qu'avec le démarrage à la System V on ajoute un lien dans un répertoire sans toucher aux scripts existants.

2.4 Documentation : La hiérarchie des répertoires

Voici comment sont organisés les principaux répertoires de la hiérarchie standard :

- `/` Répertoire racine
- `/bin` Executables nécessaires pour le démarrage et le fontionnement en mode "single user"
- `/boot` Contains static files for the boot loader.
- `/dev` Périphériques et fichiers spéciaux
- `/etc` Fichiers de configuration de la machine. contient des sous-répertoires pour les "gros" logiciels (X11, mail, ...)
- `/etc/skel` Fichiers initiaux pour la création d'un compte
- `/home` Emplacement habituel des répertoires des utilisateurs.
- `/lib` Bibliothèques nécessaires au démarrage du système et aux commandes de base
- `/mnt` répertoire pour les montages temporaires
- `/proc` Point de montage pour `proc` qui fournit des infos sur les processus qui tournent et le noyau
- `/sbin` comme `bin`, mais pour les commandes d'administration.
- `/tmp` Fichiers temporaires. Peuvent être effacés sans préavis.
- `/usr` Habituellement sur une partition séparée, partagable, en lecture seulement.
- `/usr/X11R6` The X-Window system, version 11 release 6.
- `/usr/doc` Documentation.
- `/usr/etc` Fichiers de configuration.
- `/usr/include` Pour le compilateur C
- `/usr/lib` Bibliothèques, fichiers de configuration
- `/usr/local` Fichiers locaux du site. Contient `bin lib doc etc info man sbin src sbin`
- `/usr/sbin` Commandes d'administration pas indispensables pour le fonctionnement de base.
- `/usr/share` Répertoires partagés

- /usr/src Fichiers sources.
- /usr/src/linux Sources du noyau
- /var fichiers qui peuvent changer de taille
- /var/cache/man/cat[1-9] Pages de manuel préformattées
- /var/lock Fichiers verrous
- /var/log Journaux divers
- /var/run Fichiers contenant les numéros des processus des services démarrés.
- /var/spool Fichiers en file d'attente
- /var/spool/at commandes différées
- /var/spool/cron commandes programmées
- /var/spool/lpd Files d'attente des imprimantes
- /var/spool/mail Boites aux lettres

3 L'installation

3.1 Les étapes

- On démarre un système d'installation (habituellement boot sur cdrom)
- Ce système propose quelques choix pour l'installation : langue, méthode d'installation, etc.
- L'administrateur se voit proposer de procéder au (re)découpage du disque (ou des disques) en partitions, et une affection des partitions à un usage particulier (swap, divers morceaux de l'arborescence unix, ...)
- Les nouvelles partitions sont initialiséees (un système de fichiers vide y est créé).
- le système d'installation propose un choix de listes de logiciels candidats à l'installation, plus ou moins détaillé.
- Les logiciels choisis sont transférés, une préconfiguration est éventuellement effectuée.
- Un secteur de démarrage est installé.

3.2 Le partitionnement

Se fait avec un utilitaire comme `fdisk`. Fixe la position des partitions, leur type (linux, swap, etc), et l'indicateur d'activation.

3.3 L'installation du secteur de démarrage (LILO)

Encore une fois il y a plusieurs outils pour installer un secteur de démarrage. Le plus connu est *LILO* (*Linux Loader*).

Exemple de fichier `/etc/lilo.conf`, pour une machine ayant un système Microsoft sur la première partition, et deux installations de distributions Linux. Le choix se fait au démarrage, par défaut c'est la première (debian) qui démarre au bout de 50 dixièmes de seconde :

```
boot = /dev/hda2
delay = 50

image = /vmlinuz
  root = /dev/hda2
  label = debian
  read-only

image = /vmlinuz-2.2.18a
  root = /dev/hda3
```

```
    label = mandrake
    read-only

  other = /dev/hda1
    label = dos
    table = /dev/hda
```

4 Gérer les utilisateurs

4.1 Les utilisateurs sont dans /etc/passwd

Le fichier `/etc/passwd` contient la liste des comptes déclarés localement sur une machine. En voici quelques lignes

```
root:x:0:0:Super Utilisateur:/root:/bin/bash
dupont:x:2000:100:Marcel Dupont:/home/dupont:/bin/bash
sdupont:x:0:0:Super Utilisateur Dupont:/root:/bin/bash
```

La seconde ligne décrit le compte d'utilisateur `dupont`, qui a le numéro d'utilisateur 2000, qui appartient au groupe 100, qui a son répertoire d'accueil dans `/home/dupont` et dont le *shell* par défaut est `/bin/bash`.

La valeur `x` dans le second champ indique que son mot de passe crypé est stocké dans le fichier `/etc/shadow`[6]. Le cinquième champ donne son “vrai” nom.

On remarque que plusieurs comptes peuvent se rapporter au même numéro d'utilisateur. C'est une pratique courante quand plusieurs personnes doivent administrer une même machine : chaque administrateur déclaré peut se connecter avec son propre mot de passe, et le changer sans avoir à prévenir ses collègues.

Les groupes d'utilisateurs sont répertoriés dans `/etc/group`, dont voici un extrait :

```
root:x:0:
...
games:x:60:
users:x:100:
```

Il y a aussi un fichier `/etc/gshadow`. Voir le manuel pour les détails.

4.2 Enregistrer des utilisateurs *à la main*

Nous allons d'abord voir comment enregistrer des utilisateurs “à la main” pour comprendre ce qui se passe, puis deux commandes qui permettent de le faire de façon plus confortable.

On peut procéder ainsi :

- Ajouter une ligne dans `/etc/passwd`, du genre

  ```
  durand:!:1234:100:Marcel Durand:/home/durand:/bin/bash
  ```

  Le point d'exclamation indique que le compte est bloqué (provisoirement).
- Créer le répertoire de durand à partir des fichiers modèles qui sont dans `/etc/skel`, et lui en concéder la propriété

  ```
  # cp -a /etc/skel /home/durand
  # chown -R durand.users /home/durand
  ```

[6]Autrefois le fichier `/etc/passwd` contenait le mot de passe crypté de tous les utilisateurs. Pour améliorer la sécurité, les mots de passe cryptés ont été déplacés dans `/etc/shadow` qui n'est pas lisible par tout le monde, et qui contient également des informations administratives sur la date de validité du mot de passe.

– et lui donner un mot de passe initial, ce qui débloquera le compte :

```
# passwd durand
Enter new UNIX password:
Retype new UNIX password:
```

4.3 Enregistrement rapide par adduser

La commande interactive `adduser` permet d'enregistrer rapidement un nouvel utilisateur. Selon la version utilisée, l'utilisateur est enregistré systématiquemant dans le groupe commun `users`, ou dans son propre groupe créé pour l'occasion (c'est le cas ci-dessous). L'enregistrement est rapide, il suffit de donner le nom d'utilisateur et deux fois le mot de passe, les autres réponses peuvent rester en blanc.

```
# adduser
Enter a username to add: martin
Adding user martin...
Adding new group martin (1001).
Adding new user martin (1001) with group martin.
Creating home directory /home/martin.
Copying files from /etc/skel
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for martin
Enter the new value, or press return for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [y/n] y
```

Cette commande convient bien pour des machines qui n'ont que quelques utilisateurs, regroupés dans un seul répertoire.

4.4 useradd, enregistrement par ligne de commande

Lorsqu'il y a beaucoup d'utilisateurs à enregistrer, on préfèrera sans doute `useradd`.

Soit par exemple à ajouter dix utilisateurs dans le groupe (à créer) `stagiaires` qui sera logé dans le sous-répertoire `/home/stagiaires2002` (à créer également).

– On crée le groupe

```
# groupadd stagiaires
```

– On crée le groupe et le répertoire

```
# groupadd stagiaires
# mkdir /home/stagiaires2002
```

– On donne les paramètres communs par défaut :

```
# useradd  -D  -g stagiaires  -b /home/stagiaires2002  -s /bin/bash
```

– et on procède à l'enregistrement des comptes :

```
# useradd -c "Paul MacC." -m paul
# useradd -c "John L."    -m john
...
```

Les comptes sont créés, mais restent bloqués tant qu'un mot de passe n'est pas défini.

Note : l'option `-m` demande la création effective du répertoire d'accueil de l'utilisateur à partir des fichiers modèles.

4.5 Enregistrer un administrateur supplémentaire

On peut procéder comme suit :

```
# useradd -u 0 -o  -g 0  -c "Super Durand" -d /root sdurant
```

Les options `-u 0` et `-g 0` précisent le numéro d'utilisateur et le groupe. On indique `-o` parce qu'il y a déjà un utilisateur avec le même numéro. Le répertoire d'accueil pour `sdurant` sera celui de `root`.

4.6 Changer les mots de passe

L'attribution de mots de passe par la commande `passwd` est envisageable pour une ouverture de quelques comptes, mais pas pour des besoins plus massifs, à cause du travail pénible de frappe des mots de passe (en double !)

Pour attribuer/changer des mots de passe en bloc, on commence par constituer un fichier `mots-de-passe` de paires "nom :mot de passe"

```
john:gui23tare
paul:bas67se
...
```

que l'on fait prendre en compte par :

```
# chpasswd < mots-de-passe
```

Ce fichier, une fois imprimé, peut être découpé en bandelettes de papier qui seront remises en mains propres aux titulaires respectifs des comptes.

5 Faire marcher le réseau

L'objectif ici est de pouvoir relier la machine au réseau local, et de la rendre capable de communiquer avec le reste du monde.

5.1 Faire reconnaitre la carte réseau

On suppose qu'il s'agit d'une carte Ethernet[7].

Pour savoir si votre carte est déjà reconnue par le noyau, tentez ceci :

```
# ifconfig eth0
```

Si elle n'est pas reconnue, vous aurez le message `"Périphérique non trouvé"` qui signifie que la carte n'est pas encore reconnue par le noyau qui a été lancé au démarrage, ni par les pilotes de cartes réseau qui ont éventuellement été lancés par la suite. Maintenant, que faire ?

Première étape, identifier le type de carte réseau, en regardant la boite, ou sur la carte elle-même. A tout hasard, lancez la commande

```
# lspci
```

ou

[7] Ça ne serait pas très différent avec des cartes Token Ring ou autres, mais inutile de compliquer les explications

```
# cat /proc/pci
```

qui vous dira ce que le noyau pense des cartes (et autres dispositifs) PCI présents dans l'ordinateur. Normalement - si c'est une carte PCI - vous devriez trouver une ligne qui parle d'Ethernet. Notez-la.

Seconde étape, cherchez un pilote pour la carte réseau parmi les modules disponibles. Les pilotes se trouvent dans `/lib/modules/xx.yy.zz/net`, où `xx.yy.zz` est la version du noyau, que vous pouvez déterminer par la commande `uname -r`.

Voila la liste des modules d'un système typique :

```
$ ls /lib/modules/$(uname -r)/net
3c501.o            comx-hw-locomx.o   hostess_sv11.o  sb1000.o
3c503.o            comx-hw-mixcom.o   hp-plus.o       sbni.o
3c505.o            comx-proto-fr.o    hp.o            scc.o
3c507.o            comx-proto-lapb.o  hp100.o         sdla.o
3c509.o            comx-proto-ppp.o   ibmtr.o         sdladrv.o
3c515.o            comx.o             ipddp.o         sealevel.o
3c523.o            cops.o             ircomm.o        shaper.o
3c527.o            cosa.o             irda.o          sis900.o
3c59x.o            cs89x0.o           irlan.o         sk98lin.o
6pack.o            de4x5.o            lance.o         sk_mca.o
8139too.o          de600.o            lanstreamer.o   sktr.o
82596.o            de620.o            lne390.o        slhc.o
8390.o             depca.o            ltpc.o          slip.o
ac3200.o           dgrs.o             mkiss.o         smc-mca.o
acenic.o           dlci.o             n2.o            smc-ultra.o
arc-rimi.o         dmascc.o           ne.o            smc-ultra32.o
arcnet.o           dmfe.o             ne2.o           smc9194.o
arlan-proc.o       dummy.o            ne2k-pci.o      soundmodem.o
arlan.o            e2100.o            ne3210.o        starfire.o
at1700.o           eepro.o            ng_tulip.o      strip.o
baycom_epp.o       eepro100.o         ni5010.o        syncppp.o
baycom_par.o       eexpress.o         ni52.o          tlan.o
baycom_ser_fdx.o   epic100.o          ni65.o          tulip.o
baycom_ser_hdx.o   eql.o              old_tulip.o     via-rhine.o
bonding.o          es3210.o           olympic.o       wanpipe.o
bpqether.o         eth16i.o           pc300.o         wanxl.o
bsd_comp.o         ethertap.o         pcnet32.o       wavelan.o
c101.o             ewrk3.o            plip.o          wd.o
com20020.o         fmv18x.o           ppp.o           x25_asy.o
com90io.o          hamachi.o          ppp_deflate.o   yam.o
com90xx.o          hdlc.o             rcpci.o         yellowfin.o
comx-hw-comx.o     hdlcdrv.o          rtl8139.o       z85230.o
```

Si vous trouvez votre bonheur dans la liste (par exemple `via-rhine.o`), il ne vous reste plus qu'à charger le module, par

```
# modprobe via-rhine
```

Si ça s'est bien passé, il n'y a pas de message à la console, et en faisant `ifconfig eth0` vous devez voir des informations à propos de la carte. Si ça ne marche pas, voyez ce qu'a dit la commande `dmesg` à propos du chargement du module. Eventuellement essayez d'autre modules.

À tout hasard, une "méthode de sauvage" pour essayer de déterminer le module qui va convenir :

```
# cd /lib/modules/$(uname -r)/net
# for m in *.o ; do modprobe $(basename $m .o) ; done
```

et ensuite regarder ce qu'en dit `dmesg`, consulter la liste des modules chargés (`lsmod`), ...

Ensuite, il ne reste plus qu'à demander le chargement du module à chaque démarrage : selon les distributions de linux, vous pourrez
– éditer le script de chargement des modules `/etc/rc.d/rc.modules` (distribution SlackWare),
– mettre le nom du module dans `/etc/modules` (Debian)
– éditer `/etc/modules.conf` (...)

Remarques :
– Il y a des modules réseau qui ne correspondent pas à des cartes, par exemple `bonding`, `dummy`, `ppp`, `slip`...
– Certaines cartes sont reconnues par plusieurs modules, plus ou moins bien.
– Pour des cartes ISA, il faut parfois préciser le numéro d'interruption, l'adresse des ports utilisés, du genre

```
# modprobe 3c509 io=0x300 irq=10
```

Et pour Plug'n'Play[8], c'est pire.

5.2 Configurer l'interface réseau et les routes

On suppose qu'on veut donner l'adresse `10.11.12.13` à notre machine, sachant que les autres machine du réseau local portent des numéros commençant par `10.11.12`, et que la passerelle de sortie est `10.11.12.254`.

La commande `ifconfig` permet de configurer l'interface réseau.

```
# ifconfig eth0 10.11.12.13  netmask 255.255.255.0  up
```

Si un "`ping 10.11.12.254`" répond, tout va bien.[9]

Sinon
– Vérifiez vos paramètres par "`ifconfig`"
– La machine passerelle `10.11.12.254` est-elle vraiment en fonctionnement ? Branchée sur le même réseau ?
– Un problème de câble ?
– Il y a un pilote pour la carte réseu, mais est-ce le bon ?
– ...

Enfin, la route par défaut est installée par

```
# route add default gw 10.11.12.254
```

Normalement, la machine peut maintenant communiquer avec le reste du monde.

En cas de doute, on peut consulter la route par `route -n` ou `netstat -rn`.

Pour survivre au redémarrage, la configuration doit être enregistrée dans des fichiers qui dépendent - comme d'habitude - de la distribution
– Slackware : le script `/etc/rc.d/rc.inet1` contient les commandes `ifconfig`, `route`, etc. On peut reconfigurer ce fichier par l'utilitaire `netconfig`.
– Debian : le fichier `network/interfaces` contient les paramètres des différentes interfaces. Extrait :

```
iface eth0 inet static
    address 10.11.12.13
    netmask 255.255.255.0
    broadcast 10.11.12.255
    gateway 10.11.12.254
```

– Autres : ...

[8] heureusement en voie de disparition

[9] Un "`ping 10.11.12.13`" qui répond n'est pas vraiment significatif. Dans ce cas la couche réseau de notre machine n'a absolument pas besoin de faire appel à la couche physique pour communiquer avec elle-même. Par contre si ça ne répond pas, c'est révélateur d'un problème (adresse mal configurée, pas de support réseau dans noyau, ...)

5.3 Utiliser un serveur de noms

Facile à configurer : il faut placer le numéro IP du serveur de noms dans le fichier `/etc/resolv.conf`, qui ressemble à ceci :

```
search stage.org
nameserver 10.11.12.1
```

La directive `search` indique un domaine par défaut, ce qui permet d'abréger un nom complet comme `pc112.stage.org` en `pc112`.

6 Les montages (locaux)

6.1 Faire un montage provisoire : la commande `mount`

La commande `mount` permet d'effectuer un *montage*, c'est-à-dire d'accrocher dans l'arborescence courante un bout d'arborescence contenu sur un systèle de fichiers.

Par exemple si on a un CD dans le lecteur `/dev/hdc` (dispositif maître sur la seconde nappe IDE), on peut intégrer son contenu sous le point de montage `/tmp/cederom` en faisant

```
# mkdir /tmp/cederom
# mount /dev/hdc /tmp/cederom
```

Après usage, il faudra libérer le disque par

```
# umount /tmp/cederom
```

En principe le type de système de fichiers est détecté automatiquement, mais on peut le préciser. C'est parfois utile pour les disquettes provenant de Windows, qui peuvent être de type `msdos` ou `vfat` (avec les noms longs). Exemple :

```
# mount -t vfat /dev/fd0 /tmp/disquette
```

Il y a aussi des options, par exemple pour un montage en lecture seule :

```
# mount -t vfat /dev/fd0 /tmp/disquette  -o ro
```

6.2 Montages au démarrage, le fichier `/etc/fstab`

Le fichier `/etc/fstab` indique principalement les montages à effectuer au démarrage.

```
# /etc/fstab: static file system information.
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
/dev/hda1       /       ext2    defaults,errors=remount-ro 1    1
/dev/hda2       /opt    ext2    defaults,errors=remount-ro 1    1
/dev/hda3       /home   ext2    rw                         1    2

/dev/hda4       none    swap    sw                         0    0
proc            /proc   proc    defaults                   0    0

/dev/fd0        /floppy auto    defaults,user,noauto       0    0
/dev/cdrom      /cdrom  iso9660 defaults,ro,user,noauto    0    0
```

On y trouve également une indication de la partition utilisée pour le *swap*. On y voit aussi des montages qui ne sont pas faits automatiquement au démarrage (`noauto`), qu'on déclenchera par `mount /floppy` ou par `mount /cdrom`. Ici ils peuvent être lancés par n'importe quel utilisateur (`user`).

7 NFS - le partage de fichiers à travers le réseau

NFS (*Networked File System*) est le protocole le plus utilisé dans le monde Unix pour partager des fichiers sur un réseau local.

C'est encore un protocole client-serveur : un serveur exporte un de ses répertoires, auxquels les clients accèdent. Sur le serveur, l'exportation fonctionne sous contrôle de deux démons `nfsd` et `mountd` qui sont lancés au démarrage. Le client utilise `mount`.

7.1 Exporter un répertoire

On peut exporter provisoirement un répertoire grâce à la commande `exportfs`. Exemple :

```
# exportfs  -o rw   pc132:/export/donnees
# exportfs  -o ro   *:/cdrom
```

Le fichier `/etc/exports` contient la liste des exportations à faire au démarrage. Exemple

```
/home/stage/invites/marcel pc112.stage(rw)
/usr/doc                              (ro)
/tftpboot/10.1.1.102       pc102.stage(rw,no_root_squash)
```

Pour faire prendre en compte immmédiatement les modifications de ce fichier, faire :

```
# exportfs -r
```

`exportfs` permet également
- de voir la liste des exports en cours : `exportfs -v`
- de cesser un export

```
# exportfs -u pc103:/floppy
```

7.2 NFS - du coté du client

Le client peut faire des montages provisoires par `mount` :

```
# mount  pc102:/cdrom  /mnt/net
```

Les performances sont en général bien meilleures (sur un réseau local Ethernet)avec les options qui suivent :

```
# mount pc102:/cdrom /mnt/net   -o rsize=8192,wsize=8192
```

Les montages à effectuer à chaque démarrage ont leur place dans `/etc/fstab` :

```
...
server:/usr/local/pub    /pub   nfs    rsize=8192,wsize=8192,timeo=14,intr
...
```

Principales options pour les montages NFS
- `rsize=8192,wsize=8192` Taille des lectures/écritures
- `timeo` délai initial avant retransmission
- `bg` mise en arrière-plan si échec initial
- `hard` En cas d'échec NFS, le système effectuec d'autres tentatives
- `soft` En cas d'échec NFS, le demandeur est interrompu
- `intr` possibilité d'interrompre un processus qui attend une opération NFS

7.3 L'automontage

Montage *au dernier moment* : quand on tente d'accéder à un élément du répertoire *sous* le point de montage.

Permet d'éviter les problèmes de démarrage des machines ayant des *montages croisés*.

On peut[10] utiliser l'automontage pour s'éviter de faire des `mount` et `umount` pour les lecteurs de disquette ou de CDROM.

Exemple
- La "carte d'automontage" `/etc/auto.misc` contient :

```
cd      -fstype=iso9660,ro,nodev,nosuid              :/dev/cdrom
floppy  -fstype=auto,sync,nodev,nosuid               :/dev/floppy
doc     -ro,soft,intr,rsize=8192,wsize=8192   serveur:/pub/doc
```

- L'automonteur est lancé par la commande

```
# automount  /magic  file /etc/auto.misc
```

- Pour lire le CD, il suffit de consulter `/magic/cd` !
- Démontage automatique 5 minutes après la dernière utilisation (délai réglable).

Au démarrage on peut lancer plusieurs `automount`, avec des points d'automontage différents. La liste est dans `/etc/auto.master`

```
/var/autofs/misc        /etc/auto.misc
```

Les cartes d'automontage peuvent être diffusées par les Pages Jaunes, voir section 10.4 (NIS et Automontage).

8 Faire les sauvegardes

8.1 Copie de répertoire sur disque, disquette, ...

Supposons que l'on veuille faire une copie du répertoire `/home` sur une partition inoccupée d'un disque amovible (par exemple `/dev/hdb2`. On pourra procéder comme suit.

```
# mke2fs /dev/hdb2
# mkdir /tmp/svgd
# mount /dev/hdb2 /tmp/svgd
# cp -a /home /tmp/svgd
# umount /tmp/svgd
```

L'option `-a` de la commande `cp` réalise une copie récursive, en repectant les indications de propriétaire, de date de création etc.

Pour une disquette, utiliser `/dev/fd0` à la place de `/dev/hdb2`.

8.2 Archivage sur bande, cartouche, ... par `tar`

Une archive est un "gros fichier" qui regroupe différents fichiers et répertoires, qui sont accessibles séquentiellement. Il y a différents formats d'archive, et différents outils, les plus connus étant `tar`, `cpio` et `dump/restore`. Ici on regarde `tar`.

Exemple : utilisation d'un lecteur de bandes IDE de type Colorado.

Avant tout, on charge les modules qui permettront de faire passer le périphérique IDE pour un dispositif SCSI, ainsi que le module pour les dispositifs à bandes (SCSI Tape) :

```
# modprobe ide-scsi
# modprobe st
```

Pour sauvegarder le répertoire `/home` :

[10] ce n'est pas parce qu'on peut le faire qu'on doit le faire

```
# tar cvf /dev/st0  /home
```

Pour consulter la liste des fichiers d'une cartouche :

```
# tar tf /dev/st0
```

Pour restaurer les répertoires `/home/dupont` et `/home/martin` dans `/tmp/recup` :

```
# mkdir /tmp/recup
# tar xvf /dev/st0 -C /tmp/recup  /home/martin  /home/dupont
```

les répertoires récupérés se trouveront dans `/tmp/recup/home/martin` et `/tmp/recup/home/dupont`.

8.3 Sauvegardes différées/programmée

La commande `at` permet de différer le lancement de commandes. Pour lancer une sauvegarde dimanche prochain à 3 heures du matin :

```
# at 3am sunday
tar cvf /dev/st0  /home
^D
```

Variante : mettre la commande de sauvegarde dans un script (elle pourrait resservir) `/usr/local/sbin/script-sauvegarde` et lancer

```
# at -f /usr/local/bin/script-sauvegarde 3am sunday
```

Pour programmer une sauvegarde tous les dimanches, lancer la commande "`crontab -e`" pour éditer le fichier des tâches programmées. Y ajouter une ligne de la forme

```
0 3 * * 0  /usr/local/sbin/script-sauvegarde
```

après avoir mis les commandes de sauvegardes dans un script `/usr/local/sbin/script-sauvegarde`.

9 Les impressions

Il existe plusieurs systèmes de gestion des imprimantes : LPR (BSD), System V, CUPS, LPRnG, etc. On présente ici le système traditionnel LPR de BSD.

9.1 Le système d'impression LPR (BSD)

Le système d'impression fonctionne de la façon suivante (en simplifiant)
- Il reçoit des *requêtes d'impression*. Ces requêtes peuvent provenir de la machine locales, ou arriver par le réseau.
- Les travaux sont stockés dans des files d'attente.
- Les fichiers à imprimer peuvent faire l'objet d'un traitement : mise en forme des textes bruts, conversions pour le type d'imprimante d'utilisée, etc. Voir plus loin "filtres d'impression".
- Les travaux d'impression peuvent également être transmis à une autre machine ou à une imprimante réseau.

Ce système est décrit par le fichier `/etc/printcap`.

9.2 Déclarer une imprimante locale

Voici un extrait du fichier `/etc/printcap`, concernant une imprimante nommée `lp`[11] reliée au port parallèle `/dev/lp0`.

[11] `lp` est le nom conventionnel de l'imprimante par défaut

```
lp|Imprimante sur port parallèle:\
        :lp=/dev/lp0:\
        :sd=/var/spool/lpd/lp:\
        :lf=/var/log/lp-errs:\
        :mx=0:\
        :sh:
```

Après le nom de l'imprimante et sa description, on trouve différents champs. Ici :
- `lp` : le périphérique
- `sd` : *spool directory*, le répertoire qui contient la file d'attente
- `lf` : *le log file* qui enregistre les messages d'erreurs concernant cette imprimante
- `mx` : *maximum size*, taille maximum des fichiers à imprimer (ici, non limitée).
- `sh` : *suppress header*, indicateur optionnel qui évite d'imprimer une page de garde pour chaque travail.

9.3 Le fonctionnement : `lpd` et `lpc`

Pour imprimer l'utilisateur emploie la commande `lpr`, ou `a2ps`, ou `enscript`, etc.

Le système d'impression est lancé par le démon `lpd` qui gère les files d'attentes, et lance les traitements quand les périphériques sont disponibles.

La commande `lpc` permet de contrôler le fonctionnement du système d'impression :
- "`lpc enable` *file*" autorise l'ajout de nouveaux travaux dans une file d'impression
- "`lpc disable` *file*" l'interdit
- "`lpc stop` *file*" arrête les impressions d'une file d'attente, mais n'interdit pas d'y ajouter des travaux
- "`lpc start` *file*" relance les impressions d'une file d'attente.
- "`lpc abort` *file*" arrête l'impression en cours, et bloque les impressions (comme pour `stop`).

9.4 Filtres d'impression

De nos jours la plupart des logiciels utilisent une version de PostScript[12] ou une autre comme langage pour les sorties sur imprimantes.

Les imprimantes courantes (et bon marché) ne reconnaissent pas ce langage, mais seulement des codes de bas niveau comme PCL (HP), ESC (epson) etc.

On utilise donc des programmes de conversion, appelés *filtres d'impressions* pour traduire les fichiers PostScript en codes spécifiques de l'imprimante. En intégrant ceci dans le système d'impression, on transformera donc n'importe quelle imprimante (ou presque) en imprimante PostScript virtuelle.

Ce n'est pas très compliqué à faire : il suffit d'ajouter un champ `if` qui précise l'emplacement du filtre. Exemple :

```
bureau302ps|Imprimante PostScriptifiée Bureau302:\
        :lp=/dev/lp0:\
        :sd=/var/spool/lpd/bureau302ps:\
        :if=/usr/local/bin/filtre-postscript:\
        :lf=/var/log/lp-errs:\
        :sh:
```

et bien sûr il faut réaliser (ou se procurer) le programme de filtrage.

Exemple : un filtre simpliste pour LaserJet 4

```
#!/bin/bash
```

[12] Marque déposée d'Adobe

```
gs -sDEVICE=$MODE -r300x300 \
       -q -dNOPAUSE -dBATCH -dSAFER \
       -sOutputFile=-
```

En réalité, on préfère plutôt faire des *filtres magiques* qui commencent par détecter le type du document reçu (texte ordinaire, PostScript, DVI, HTML, image, etc.) et lancent un programme de conversion/formattage adapté à la situation.

9.5 Utiliser une imprimante distante

Pour imprimer sur une imprimante qui est reliée à une autre machine, on indique deux clauses
- `rm` (*remote machine*), le nom de la machine qui support l'imprimante
- `rp` (*remote printer*), le nom de l'imprimante sur cette machine.

```
bureau302|Imprimante distante:\
        :rm=pc302a:\
        :rp=bureau302ps:\
        :sd=/var/spool/lpd/bureau302:
```

Pour que ça marche, il faut que la machine qui possède l'imprimante autorise les impressions venant de telle autre machine. Il suffit de donner les noms dans `/etc/lpd.conf`.

10 Gestion centralisée par NIS

Gérer des comptes d'utilisateurs séparément sur chaque machine d'un réseau poserait rapidement des problèmes insurmontables. On utilise donc une gestion centralisée : les utilisateurs sont déclarés sur un serveur d'informations administratives, et ces informations sont diffusées auprès des machines du réseau.

NIS (*network information system*[13]) est un système qui permet de faire ce genre de choses.

10.1 Configuration d'un client

- Le client doit d'abord positionner son *nom de domaine NIS*,

  ```
  # nisdomainname nis-stage
  ```

- puis lancer le *daemon* `ypbind` pour se connecter à un serveur

  ```
  # ypbind
  ```

- `ypbind` est configuré par `/etc/yp.conf`

  ```
  #
  domain nis-stage server 10.1.1.47
  ```

- Vérification par

  ```
  # ypwhich
  ```

- Le client peut alors consulter les informations ("*maps*") diffusées sur le domaine

  ```
  # ypcat passwd
  # ypcat group
  ...
  ```

- Pour utiliser effectivement ces informations pour l'authentification des utilisateurs, renseigner `/etc/nsswitch.conf`

[13]Ex-Yellow Pages, développé par la société SUN. Le nom a été changé pour cause de marque déposée.

```
passwd:         files nis
group:          files nis
shadow:         files nis
...
```

10.2 Configuration d'un serveur

Attention : le serveur NIS fournis dans le package `nis.tgz` de la distribution Slackware 8.0 ne fonctionne pas. Le remplacer ou utiliser une autre distribution pour un serveur NIS.

Cas le plus simple : le serveur diffuse ses propres fichiers

– Sélectionner le domain NIS

```
# nisdomainname nis-stage
```

– Lancer le *daemon* `ypserv`
– Initialiser le serveur, construire les *maps* et les donner à diffuser au serveur

```
# /usr/lib/yp/ypinit -m
```

Eventuellement, éditer le fichier `/var/yp/Makefile` pour éviter la diffusion d'informations inutiles.

Note. L'accès des clients NIS est contrôlé par deux mécanismes
– `/etc/securenets`
– *TCP wrapper*

10.3 Gestion des comptes sur le serveur

– Les comptes sont créés, détruits etc. sur le serveur
– Les mots de passe peuvent être changés depuis les clients

```
# yppasswd
```

si on fait tourner le *daemon* `yppasswdd` sur le serveur.
– Après toute modificationn faire

```
# ypmake
```

dans le répertoire `/var/yp` pour mettre à jour les *maps*.

10.4 NIS et Automontage

Les *cartes d'automontage* peuvent être diffusées par NIS. C'est utile quand chaque machine doit monter plusieurs systèmes de fichiers NFS

Le fichier `/var/yp/Makefile` prévoit déjà la diffusion d'une map `auto.home`, constituée à partir du fichier `/etc/auto.home`. Extrait de `/var/yp/Makefile` :

```
...
YPSRCDIR = /etc
...
AUTO_HOME   = $(YPSRCDIR)/auto.home
...

auto.home: $(AUTO_HOME) $(YPDIR)/Makefile
	@echo "Updating $@..."
	-@sed -e "/^#/d" -e s/#.*$$// $(AUTO_HOME) | $(DBLOAD) \
		-i $(AUTO_HOME) -o $(YPMAPDIR)/$@ - $@
	-@$(NOPUSH) || $(YPPUSH) -d $(DOMAIN) $@
```

Dans le fichier `/etc/auto.master` de chaque client, ajouter

```
/net    auto.home      rsize=8192,wsize=8192,soft,intr
```

11 Installer un serveur de noms (DNS)

11.1 A quoi ça sert ?

Les serveurs de noms (DNS, *Domain Name Serveur*) ont deux buts principaux :
- établir une correspondance entre des noms de machines, et des adresses IP
- désigner les serveurs de courrier

Les serveurs de noms sont reliés entre eux, et l'ensemble fonctionne selon un principe de *délégation de sous-zones*.

Exemple : pour trouver l'adresse IP de `pc43.mondomaine.fr`,
- Le client s'adresse à un serveur de noms *racine*, qui lui fournit l'adresse d'un serveur de noms connaissant la zone `fr`
- Le serveur de la zone `fr` donne l'adresse du serveur qui s'occupe de `mondomaine.fr`
- ce serveur fournit l'adresse de `pc43.mondomaine.fr`

En réalité, c'est un petit peu plus compliqué que cela :
- Pour une zone, il y a au moins un *serveur primaire* (ou *serveur maître*), mais il y a aussi des serveurs *secondaires* (*esclaves*) auxquels on s'adresse quand le serveur primaire n'est pas joignable.
- Des caches permettent de réduire le trafic : les réponses sont conservées un certain temps.

11.2 Configuration des clients (rappel)

On suppose qu'on a configuré un serveur de noms pour le domaine `stage`. L'adresse de ce serveur est `10.1.1.2`.

Dans `/etc/resolv.conf`, on met par exemple :

```
search stage
nameserver 10.1.1.2
```

la directive "`search`" indique des suffixes par défaut ici `pc110` sera synonyme de `pc110.stage`,

11.3 Configuration d'un serveur maître

Le logiciel *named* (Name Daemon) et/ou *bind* (Berkeley Internet Name Daemon ?). Il est lancé par la commande `named`, et le fichier de configuration principal est dans `/etc/bind/named.conf`

11.3.1 Le fichier de configuration `/etc/bind/named.conf`

Ce fichier les zones gérées par le serveur, celles dont il est serveur secondaire, les serveurs racine, etc. Extrait :

```
options {
        directory "/var/cache/bind";
};

// Les serveurs racine :

zone "." {
        type hint;
```

```
        file "/etc/bind/db.root";
};

// la zone localhost
zone "localhost" {
        type master;
        file "/etc/bind/db.local";
};

// pour les adresses 127.*.*.* et 0.*.*.*

zone "127.in-addr.arpa" {
        type master;
        file "/etc/bind/db.127";
};

zone "0.in-addr.arpa" {
        type master;
        file "/etc/bind/db.0";
};

zone "255.in-addr.arpa" {
        type master;
        file "/etc/bind/db.255";
};

// la zone stage, et les adresses 10.1.1.*

zone "stage" {
        type master;
        file "/etc/bind/db.stage";
};

zone "1.1.10.in-addr.arpa" {
        type master;
        file "/etc/bind/db.10.1.1";
};
```

11.3.2 Déclaration d'une zone : /etc/bind/db.stage

```
;
; Fichier de données BIND pour zone stage
;
$TTL    604800
@       IN      SOA     dns.stage. root.dns.stage. (
                         10             ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
                         IN     NS      dns.stage.

courrier  IN A    10.1.1.3
          IN MX 10   courrier.stage.
```

```
dns     IN A 10.1.1.2
        IN MX   10   courrier.stage.

pc101   IN A 10.1.1.101
        IN MX   10   courrier.stage.

pc102   IN A 10.1.1.102
        IN MX   10   courrier.stage.
```

L'entête contient diverses indications sur la durée de validité des informations, c'est à dire la durée de conservation dans les caches. C'est utile pour ne pas réinterroger trop fréquemment le même serveur. Le numéro de série est normalement changé chaque fois que le contenu est modifié : de cette façon on ne recharge pas le contenu d'une zone dont le numéro de saisie n'a pas bougé.

Les lignes "`IN A`" (Internet Address) indiquent l'adresse qui correspond à un nom, "`IN MX`" (Internet Mail eXchanger) donnent le nom du serveur de courrier associé.

On peut également introduire des *surnoms*, exemple :

```
mail   IN CNAME    courrier.stage.
```

Littéralement, cette déclaration fournit le vrai nom (*canonical name*) correspondant à un surnom.

11.3.3 Déclaration de zone inverse : `/etc/bind/db.10.1.1`

Avant tout, explication des requêtes inverses : ce sont les interrogations du type : "Quel est le nom de la machine numéro 11.122.33.44 ?".

Pour une requête de ce type, l'adresse `11.22.33.44` est d'abord retournée pour la tranformer en `44.33.22.11.in-addr.arpa`. Le traitement se fait ensuite par délégation ; on cherche le serveur qui s'occupe des numéros IP de la forme `11.*.*.*` (c'est-à-dire la sous-zone `11.in-addr.arpa`), puis celui qui gère `11.22.*.*` `22.11.in-addr.arpa`, etc.

Mais revenons à notre zone `10.1.1.*`. Le fichier de cette zone contient :

```
;
; BIND reverse data file for 10.1.1
;
$TTL    604800
@       IN      SOA     dns.stage. root.dns.stage. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
        IN      NS      dns.stage.

2       IN PTR          dns.stage.
101     IN PTR          pc101.stage.
102     IN PTR          pc102.stage.
```

Voir par exemple le "*DNS-Howto*" pour plus d'explications.

12 Courrier

12.1 La chaîne : envoi, transmission, livraison, consultation

– L'utilisateur *expédie* son courrier par un MUA (Mail User Agent)

– Le courrier est *acheminé* par des MTA (Mail Transfer Agent)
– Il est *livré* dans une *boîte aux lettres* par un MDA (Mail Delivery Agent)

Une fois livré, le courrier peut être consulté
– Sur le serveur lui-même (fichier dans `/var/spool/mail`)
– Sur une machine qui partage `/var/spool/mail` par NFS
– Par l'intermédiaire des protocoles POP et IMAP

12.2 Structure d'un serveur de courrier

En entrée :
– Commandes pour l'expédition de courrier depuis le serveur
– réception du courrier par le réseau (serveur SMTP)
En sortie :
– Envoi du courrier vers un serveur (client SMTP)
– Livraison dans les boites pour utilisateurs locaux
Stockage :
– File d'entrée non encore traitée
– File des messages à expédier

12.3 Décisions d'acheminement

Pour un serveur simple, les décisions sont basées sur la provenance et la destination du courrier.

Provenance	Destination	Action
local ou distant	local	livraison dans boite aux lettres,ouredirection
local	distant	**envoi par SMTP** à l'échangeur de courrierdu destinataire (consultation DNS)
distant	distant	**relaying** (interdit en principe)

12.4 Situation fréquentes

12.4.1 Client bête

Refile tout le courrier à expédier à un *smart host*, qui s'en occupera.

Cas des portables, accès chez fournisseurs, etc.

12.4.2 Smart Host

Relaie le courrier pour ses clients. En général, change les entêtes pour faire croire qu'il en est l'expéditeur (masquerading), pour que les retours de courriers se fassent bien.

12.4.3 Serveur de courrier pour plusieurs domaines

Vu de l'extérieur, est l'échangeur de courrier pour plusieurs domaines.

En interne, redirige le courrier vers d'autres serveurs

12.5 La configuration d'un logiciel de courrier

Les logiciels les plus connus : Sendmail, Postfix, Exim, Smail, ...

- Le service offert est assez complexe
- Les besoins sont très variables
- Fichiers de configuration assez longs

Un fichier `sendmail.cf` standard fait un bon millier de lignes. C'est normal, parce qu'il y a théoriquement des quantités de paramètres à régler.[14]

D'où la réputation de complexité des logiciels de courrier comme *sendmail*.

En réalité :
- Il y a des outils de configuration rapide, qui fabriquent les fichiers de configuration pour les situations-types les plus fréquentes.
- pour `sendmail`, il existe un *kit de configuration* basé sur le macroprocesseur *m4*

12.6 Exemple : client stupide

Ci-dessous le fichier de configuration `sendmail.mc` pour un poste de travail qui se contente de transmettre tout son courrier à un échangeur de courrier (`mail.boutique.fr`) :

```
divert(-1)
#
# client stupide pour domaine boutique.fr
# Adapté de /usr/share/sendmail/sendmail.cf/cf/clientproto.mc
#
divert(0)dnl
VERSIONID(@(#)stupide.mc')

OSTYPE(debian)
FEATURE(nullclient,mail.boutique.fr)
```

12.7 Exemple : Serveur permanent indépendant sur Internet

Cette machine expédie son courrier directement.

```
divert(0)dnl
VERSIONID(@(#)tcpproto.mc      8.10 (Berkeley) 5/19/1998')
OSTYPE(debian)
FEATURE(nouucp)
MAILER(local)
MAILER(smtp)
```

12.8 Macros utiles

Documentation : fichier `cf.README` livré avec *sendmail*.

Pour savoir si le courrier doit être livré sur place ou non :
- `FEATURE(`bestmx_is_local')` indique le courrier destiné à une certaine adresse doit être livré sur place si le DNS indique que nous sommes son serveur de courrier principal

[14] La complexité vient également des formats d'adresses qu'il convient de traiter. De nos jours, on n'emploie quasiment plus que les adresses de la forme `nom@machine.domaine`, mais il y en a beaucoup d'autres.

– le fichier /etc/mail/local-host-names[15] contient également une liste de sites pour qui nous livrons le courrier sur place.

Pour le déguisement de provenance (masquerading) :

– MASQUERADE_AS(host.domain) fait que le courrier envoyé d'ici semblera venir du site host.domain (par modification des entêtes de messages). C'est utile pour "normaliser" l'adresse de provenance[16]
– MASQUERADE_DOMAIN(otherhost.domain) indique le courrier provenant de l'adresse otherhost.domain (que nous relayons) sera aussi déguisé.
– FEATURE('masquerade_entire_domain') : si cette option est selectionnée, MASQUERADE_DOMAIN permet de désigner des domaines entiers, pas seulement des machines individuelles. Exemple :

```
MASQUERADE_AS(courrier.institut-formation.fr)
MASQUERADE_DOMAIN(institut-formation.fr)
MASQUERADE_DOMAIN(stage)
FEATURE(‘masquerade_entire_domain’)
```

  Les courriers émis depuis le réseau de test "stage" seront retransmis à l'extérieur, et sembleront provenir de courrier.institut-formation.fr, comme les autres machines du réseau institut-formation.fr.
– FEATURE('masquerade_envelope') déguisera les "données d'enveloppe" en plus des entêtes.
– FEATURE('all_masquerade') modifie également les adresses du destinataire

Des macros similaires s'appliquent au contrôle du *relaying* : RELAY_DOMAIN(host.domain), FEATURE('relay_entire_domain'), FEATURE('relay_based_on_MX') ...

Voir documentation pour les détails.

[15]Ici pour OSTYPE(debian). Peut dépendre des configurations.

[16]En particulier dans le cas où l'on relie un réseau privé à Internet : seule la machine de courrier possède un nom "correct" vis-à-vis des DNS du monde extérieur.