Sommaire

TRAVAIL PRÉLIMINAIRE

Objectifs

Être capable de gérer les phases de démarrage et d'arrêt d'un système GNU/Linux.

Manipulations

On va travailler dans l'arborescence suivante :

```
$HOME
      |
      - tv
         |
         - TPOS15
```

Créer l'arborescence de répertoires :

```
$ mkdir -p $HOME/tv/TPOS15
```

Se déplacer dans l'arborescence de travail :

```
$ cd $HOME/tv/TPOS15
```

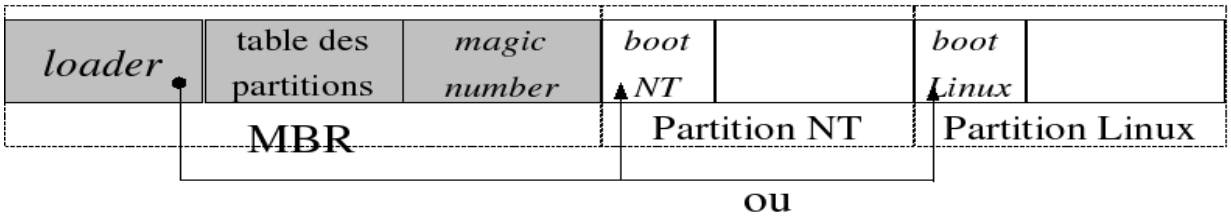
DÉMARRAGE ET ARRÊT

Démarrage de la machine

Le processus de démarrage d'un PC commence par l'exécution automatique d'un programme de chargement situé dans le BIOS (*Basic Input Output System*).

Ce programme charge, en fonction de sa configuration de démarrage (setup), et lance le programme de chargement du MBR (*Master Boot Record*).

Le programme de chargement (*loader*) du MBR charge à son tour en RAM le secteur de boot du SE de la partition active.



Démarrage de Linux

Un système Linux démarre grâce à un loader comme Lilo ou Grub.

La fonction première d'un *Linux Loader* est donc de démarrer un système Linux :

- démarrage automatique ou interactif (choix du noyau et définition de paramètres de démarrage).
- permet également le démarrage d'autres SE (Windows 9x, NT, 2000, OS/2, etc ...).

Les composants de LILO

La commande /sbin/lilo (*map installer*) : elle installe les programmes de démarrage. Elle utilise le fichier /etc/lilo.conf ou un fichier associé à l'option -C

Le fichier /boot/boot.b (*boot loader*) : il contient les fichiers de démarrage du noyau Linux, mais permet aussi le démarrage d'autre SE. Composé d'un boot primaire (stocké dans le secteur de boot, le MBR pour un DD) et d'un boot secondaire situé alors dans la partition Linux.

Le fichier /boot/map (*map file*) : ce fichier mémorise l'emplacement physique des blocs qui composent les programmes de démarrage.

Le fichier /etc/lilo.conf : c'est le fichier de configuration de la commande lilo. Il spécifie les différents noyaux Linux et éventuellement les autres SE. Outre la définition des options, il précise l'emplacement des programmes de démarrage : premier secteur d'une disquette, MBR du premier disque dur (boot primaire) ou le secteur de boot de la partition Linux (boot secondaire). Un boot secondaire devra être chargé et démarré par un boot manager.

Grub (GRand Unified Bootloader)

Sa différence principale avec lilo est la lecture de la configuration au démarrage. Pas besoin de réinstaller GRUB dans le secteur d'amorçage après un changement de configuration, contrairement à LILO.

Dans un système GNU/Linux, la commande grub-install est uniquement utilisée pour installer la Partie 1 de GRUB dans le MBR ou dans une partition. Les fichiers de configuration de GRUB doivent se trouver sur une partition utilisable, et dans le cas contraire, la Partie 1 exécute l'interpréteur de commandes automatiquement.

Le nom et l'emplacement de ce fichier varient d'un système à un autre. Par exemple, dans les distributions Debian et Mandriva GNU/Linux, ce fichier est situé dans /boot/grub/menu.lst.

Quand la partie 2 est chargée, elle affiche une interface à l'utilisateur et ce dernier peut choisir quel système d'exploitation charger. Une fois le système choisi, GRUB le charge et lui transfère le contrôle de la machine.

L'activation des processus (init)

Le système Linux offre plusieurs niveaux de fonctionnement (*runlevel*). A chacun d'eux correspond un certain nombre de services à démarrer ou à arrêter.

Le rôle d'`init` est d'activer tous les processus associés au niveau demandé (passé en argument). La commande init trouve la définition des commandes à exécuter pour un niveau donné dans le fichier /etc/inittab.

Par convention, les *runlevels* 0, 1 et 6 sont réservés de la manière suivante :

- 0 : pour arrêter le système
- 1 : pour démarrer en mode "single user"
- 6 : pour redémarrer le système

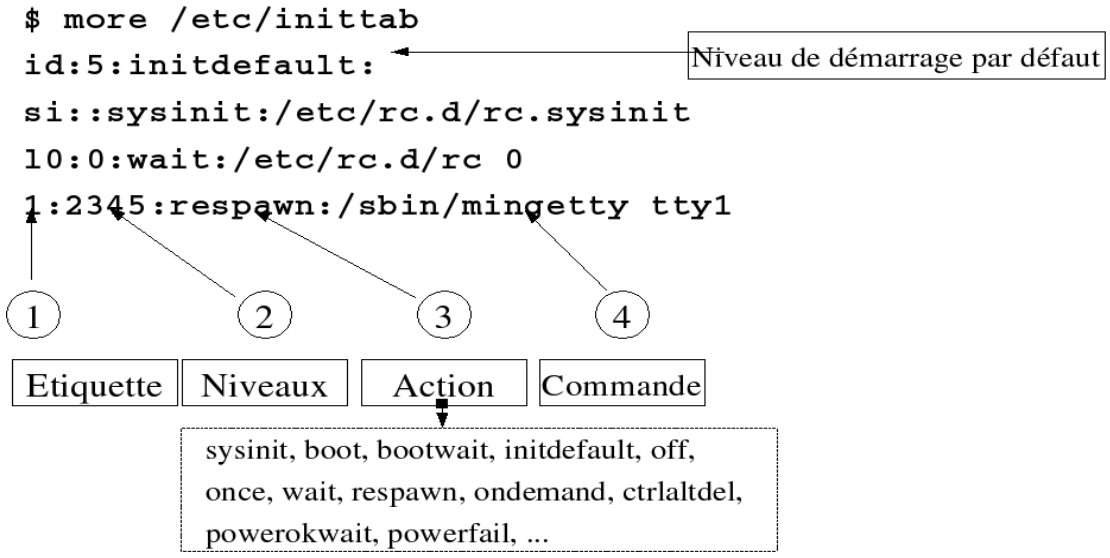
Les niveaux de démarrage habituels sont 3 (mode console) et 5 (mode graphique par l'intermédiaire des daemons xdm, kdm, gdm ou autres). On ne peut avoir qu'un seul niveau de fonctionnement actif à la fois.

Il est possible de passer, à l'invite de LILO, des paramètres au processus init et notamment le niveau de démarrage souhaité (sinon c'est celui indiqué dans inittab).

Exemple pour lilo :

```
LILO boot: linux 5
...
```

Le fichier inittab



Informations supplémentaires :

```
$ man inittab
```

Quelques commandes

Les commandes sont simples et peu nombreuses :

```
runlevel : affiche le niveau de fonctionnement courant
init : permet de changer de niveau (init 3 ou init 5 par exemple)
init q  : demande à init de relire le fichier /etc/inittab
telinit : même effet que la commande init
chkconfig : gère les informations des niveaux d'exécution pour les
services système
dmesg : affiche et contrôle le tampon des messages du noyau
service : permet de contrôler les services système
```

Les scripts de démarrage rc

La plupart des commandes exécutées à partir de inittab sont des scripts.

Le processus init commence par exécuter le script /etc/rc.d/rc.sysinit qui contient des commandes de contrôle et d'initialisation indispensables au fonctionnement de Linux. Puis, il exécute les scripts spécifiques au niveau de démarrage demandé.

Le dernier script exécuté par init est le script run commands /etc/rc.d/rc.local. Ce script est à la charge de l'administrateur pour y exécuter ses propres commandes.

Les principales opérations réalisées par le script rc.sysinit :

- initialisation de la variable PATH avec /bin:/usr/bin:/sbin:/usr/sbin
- initialisation des disques de swap
- détermination du nom de la machine (HOSTNAME)
- vérification du système de fichiers principal et montage du pseudo-système de fichiers /proc
- montage de tous les systèmes de fichiers (sauf NFS)
- chargement des pilotes dynamiques (les modules chargeables)
- création du fichier /var/log/dmesg qui contient les messages de démarrage (accessible avec la commande dmesg) ... etc ...

C'est le script /etc/rc.d/rc qui exécute les actions pour le niveau demandé.

Le fonctionnement du script rc est simple : il exécute deux boucles. La première déclenche l'exécution des scripts K* du répertoire /etc/rc.d/rcx.d et le seconde celle des scripts S* du même répertoire (x représente le niveau).

- Nommage des scripts : Kxxnom ou Sxxnom
  - K : Kill ou S : Start
  - xx : ordre d'exécution
  - nom : indique le contenu du script

Consulter l'annexe 1 pour voir le contenu « type » d'un script Kxxnom ou Sxxnom.

Démarrage et arrêt manuel d'un service par l'administrateur : utilisation des script du répertoire /etc/init.d/.

Exemple :

```
/etc/init.d/network {start|stop|restart|reload|status}   ou
service network {start|stop|restart|reload|status}
```

L'arrêt du système

L'arrêt du système est réalisé par la commande `shutdown` qui bascule le système dans le niveau de fonctionnement 0.

La frappe des touches Ctrl-Alt-Suppr provoque l'exécution de la commande `shutdown -r now` (voir le fichier /etc/inittab) et un redémarrage du système. De même pour la commande `reboot`.

La commande `halt` est équivalente à la commande `shutdown -h now` (arrêt immédiat).

MANIPULATIONS

1) Quel est le niveau de fonctionnement actuel pour init ?

2) A quoi correspondent les niveaux 3 et 5 d'init ? A partir d'une console root, tester les ces niveaux de démarrage avec la commande init.

3) Donner la commande qui permet de lister les scripts utilisés au niveau de fonctionnement 2.

4) Visualiser les messages affichés lors du dernier démarrage.

5) Lister les services systèmes. A quel niveau de fonctionnement est démarré le service cron (utiliser la commande chkconfig) ?

6) Provoquer l'arrêt complet du système dans trois minutes avec l'affichage du message "arrêt pour maintenance".

7) Dans le fichier inittab, ajouter une commande qui enregistre la date dans le fichier /tmp/date.log et qui est lancée dans le niveau par défaut. Tester.

Remarque : faire d'abord une copie de sauvegarde du fichier inittab :

```
# cp /etc/inittab /etc/inittab.old
```

8) Modifier l'affichage de lilo ou de grub en remplaçant "linux" en "GNU/Linux". Tester.

Annexe 1 : un script pour rc.d

```
#!/bin/sh
# Startup script for svn

# Source function library.
. /etc/rc.d/init.d/functions

[ -x /usr/bin/svnserve ] || exit 0

# See how we were called.
case "$1" in
  start)
    gprintf "Starting SVN Daemon: "
    daemon /usr/bin/svnserve \
    -r /var/lib/svn/repositories \
    -d
    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ] && touch /var/lock/subsys/svnserve
    ;;
  stop)
    gprintf "Stopping SVN Daemon: "
    killproc svnserve
    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/svnserve
    ;;
  status)
    status svnserve
    RETVAL=$?
    ;;
  restart|reload)
    $0 stop
    $0 start
    RETVAL=$?
    ;;
  *)
    gprintf "Usage: svnserve {start|stop|status|restart|reload}\n"
    exit 1
esac

exit $RETVAL
```