Introduction aux systèmes GNU/Linux

Séance 6

inetdoc.net

UPSSITECH
ÉCOLE D'INGÉNIEURS

Philippe Latu / Université Toulouse 3 – Paul Sabatier

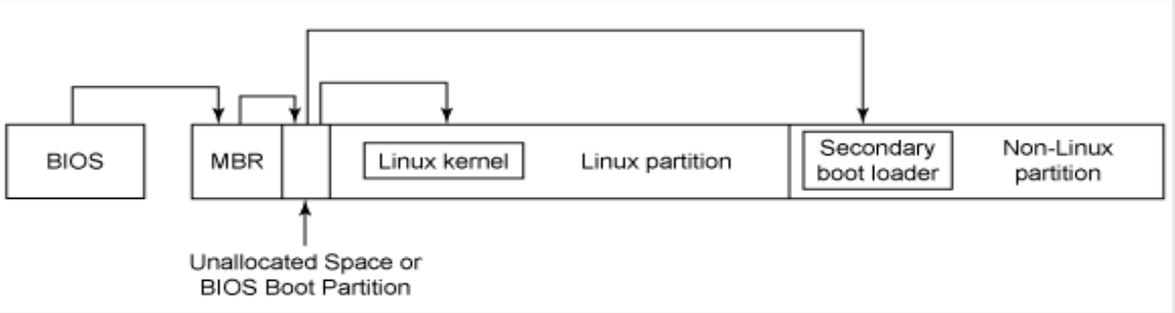
Plan séance 6

- Séance 6 - Initialisation du Système GNU/Linux
  - Identifier les étapes de sélection d'un système d'exploitation
  - Analyser le fonctionnement des niveaux de démarrage
    - runlevels
  - Caractériser le découpage de l'espace mémoire du système
    - kernelspace & userspace
  - Analyser la gestion des modules pilotes de périphériques
    - sysfs & udev

- Manipulations réalisables sur machines virtuelles
  - Étudier la liste des services lancés au démarrage
  - Différencier les modules utilisés entre système hôte et système virtuel

Initialisation du système

- POST + BIOS
  - POST → Power On Self Test
  - BIOS → Basic Input Output System
    - Premiers programmes appelés par le processeur
    - Analyse de la configuration matérielle
- Recherche d'un système d'exploitation
  - Ordre de scrutation défini dans les paramètres du BIOS
    - Pour chaque périphérique désigné → lecture du Master Boot Record (MBR)
    - Si le code Boot Loader est présent dans un MBR → initialisation du système



Initialisation du système

- Master Boot Record
  - Contient le code Boot Loader
  - Désigne la partition d'amorçage
  - Accède au gestionnaire d'amorce
- Gestionnaire d'amorce → 2 exemples
  - LILO → Linux Loader
    - Accès direct aux adresses disque
    - Écriture du code MBR à chaque changement de configuration
  - GRUB2 → Grand Unified Boot Loader
    - Code modulaire à deux «étages» → Boot Loader + Shell
    - Support GPT → Grand Partition Table
    - Support systèmes de fichiers + RAID + LVM
    - Fichier de configuration lu à chaque initialisation

Initialisation du système

- Partition ou répertoire /boot
  - Fichiers noyau + configuration gestionnaire d'amorce

```
$ mount | grep boot
/dev/vda2 on /boot type ext2 (rw,relatime,errors=continue)
```

  - Contenu du répertoire

```
$ ls -1X /boot
grub
lost+found
config-3.2.0-3-amd64
initrd.img-3.2.0-3-amd64
System.map-3.2.0-3-amd64
vmlinuz-3.2.0-3-amd64
```

Gestionnaire d'amorce

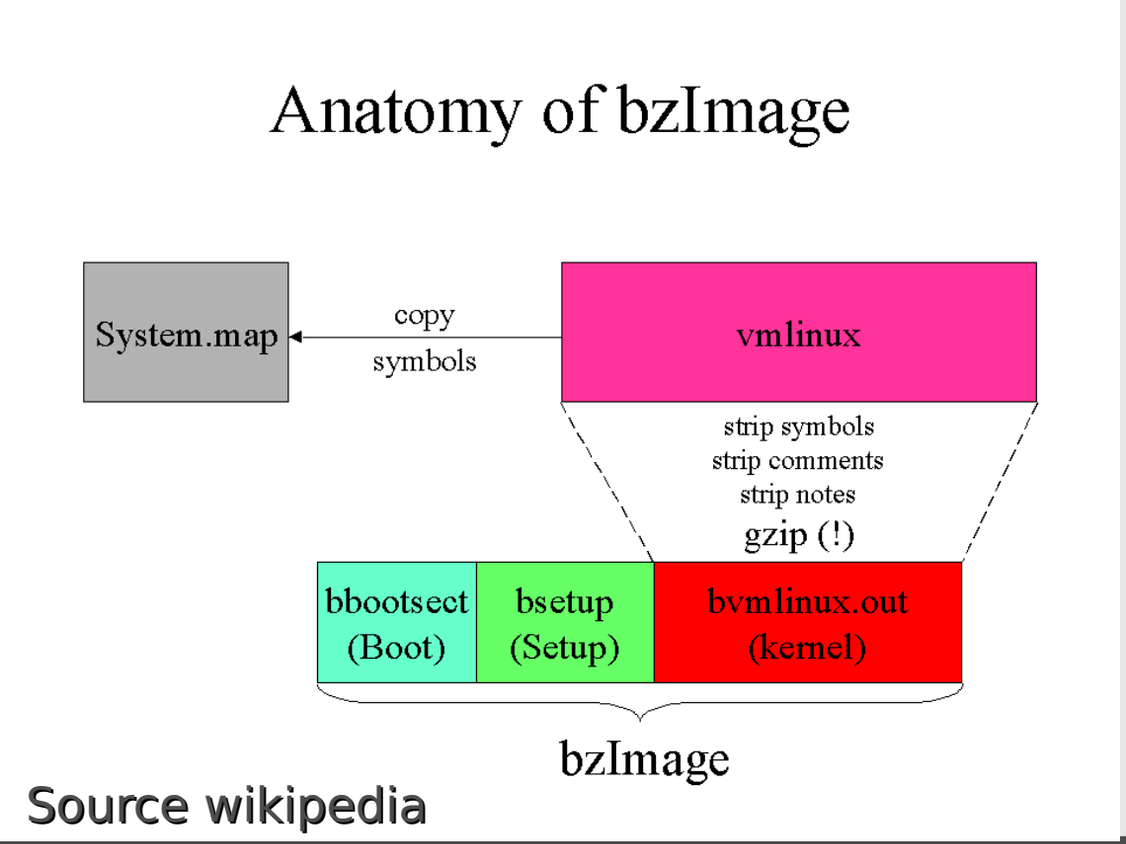
Partie monolithique du noyau

```
$ ls -1X /boot/grub/part*
/boot/grub/partmap.lst
/boot/grub/parttool.lst
/boot/grub/part_acorn.mod
/boot/grub/part_amiga.mod
/boot/grub/part_apple.mod
/boot/grub/part_bsd.mod
/boot/grub/part_gpt.mod
/boot/grub/part_msdos.mod
/boot/grub/part_sun.mod
/boot/grub/part_sunpc.mod
/boot/grub/parttool.mod
```

Modules de gestion des types de partitions

Initialisation du système

- Partie monolithique du noyau
  - Fichier bzimage → image compressée du noyau
  - Initialisation → éclatement en plusieurs zones mémoire discontigues



Source wikipedia

Initialisation du système

- Disque RAM initial
  - Shell + boîte à outils busybox → chargement des modules utiles

```
$ mkdir initrd && cd initrd
$ gzip -dc /boot/initrd.img-3.2.0-3-amd64 | cpio -idv
$ tree -L 1
.
├── bin
├── conf
├── etc
├── init
├── lib
├── lib64
├── run
├── sbin
└── scripts

8 directories, 1 file
```

Processus parent busybox

```
$ tree -L 2 lib
lib
├── firmware
│   ├── cis
│   └── isci
├── klibc-zwaqV1ZlhQoBCjGimVZufRwsjdg.so
├── modprobe.d
│   └── aliases.conf
├── modules
│   └── 3.2.0-3-amd64
├── udev
│   ├── ata_id
│   ├── edd_id
│   ├── firmware.agent
…
```

Arborescence des modules du noyau

Compilation d'un nouveau noyau

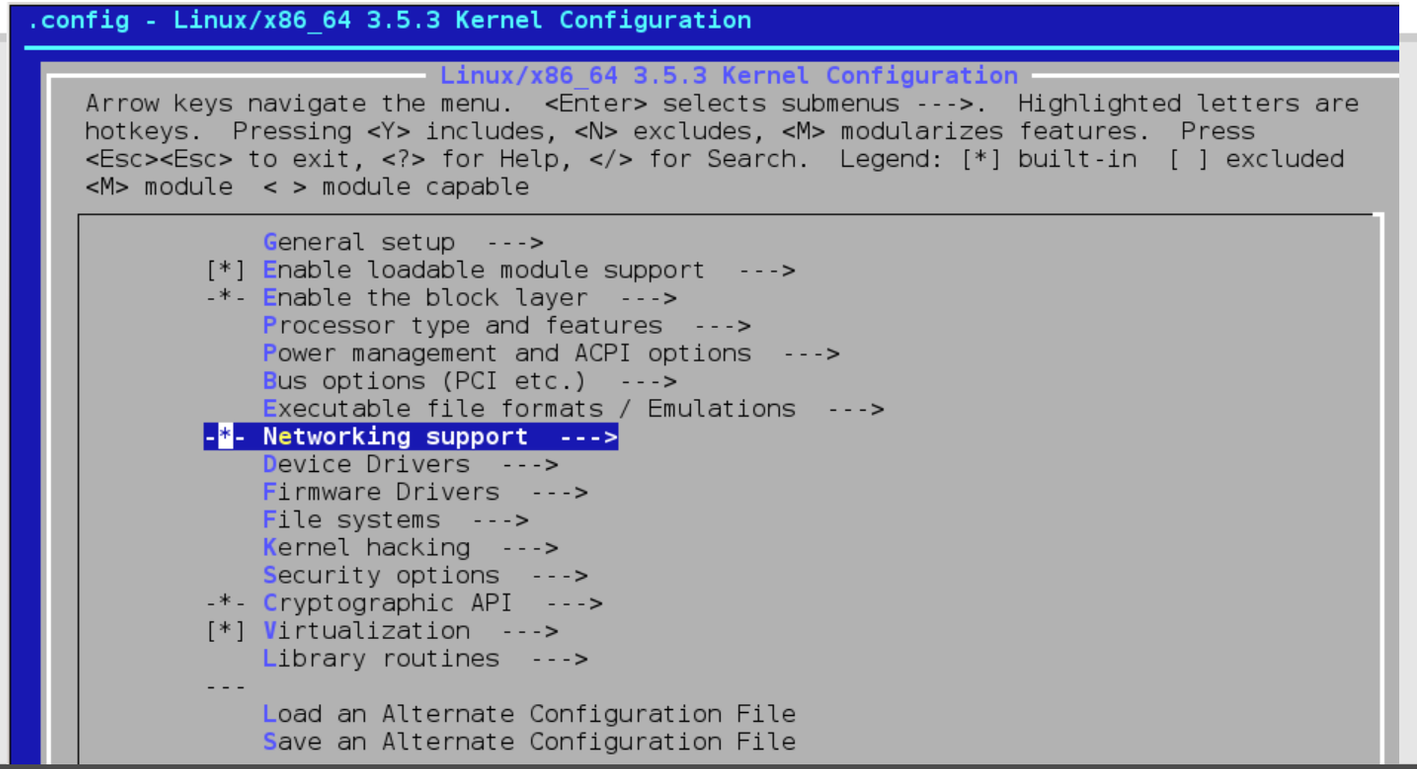
- Conditions initiales
  - L'utilisateur etu appartient au groupe système src
  - L'arborescence /usr/src appartient au groupe src
  - Les paquets suivants doivent être installés
    - kernel-package fakeroot libncurses-dev

```
$ cd /usr/src && wget http://www.kernel.org/pub/linux/kernel/v3.0/linux-3.5.3.tar.bz2
$ tar xf linux-3.5.3.tar.bz2
$ ln -s linux-3.5.3 linux && cd linux
$ cp /boot/config-3.2.0-3-amd64 .config
$ make menuconfig
$ make-kpkg --rootcmd fakeroot --initrd kernel_image
$ cd .. ; su
# dpkg -i linux-image-3.5.3_3.5.3-10.00.Custom_amd64.deb
```

Reprise de la configuration du noyau de la distribution

Compilation d'un nouveau noyau

- Interface de configuration
  - Arborescence assez complexe et difficile à assimiler
  - Noyau de la distribution → configuration déjà fonctionnelle
  - Pour débuter → procéder par modifications successives



Gestionnaire d'amorce

- Configuration déclenchée
  - À chaque nouvelle installation de noyau
  - À chaque nouvelle version des outils
- Script update-grub & personnalisation

```
# update-grub
Generating grub.cfg ...
Found background image: /usr/share/images/desktop-base/desktop-grub.png
Found linux image: /boot/vmlinuz-3.5.3
Found initrd image: /boot/initrd.img-3.5.3
Found linux image: /boot/vmlinuz-3.2.0-3-amd64
Found initrd image: /boot/initrd.img-3.2.0-3-amd64
done
```

nouveau noyau

noyau de la distribution

```
$ ls -1X /etc/grub.d/
00_header
05_debian_theme
10_linux
20_linux_xen
30_os-prober
40_custom
41_custom
README
```

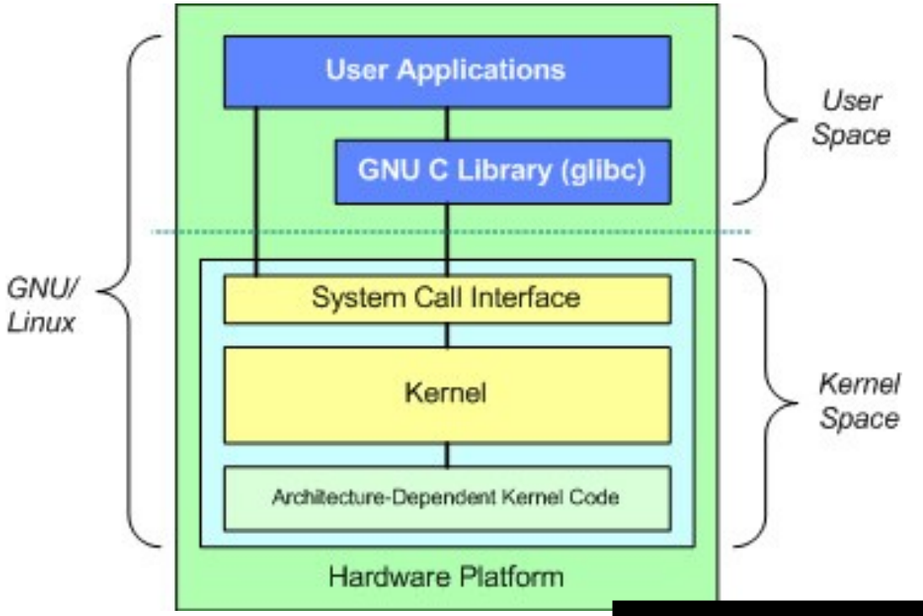
personnalisation du menu

Initialisation du noyau

- Architecture & processeur
  - Exemple des processeurs Intel → mode protégé, 32 ou 64 bits
  - Séquence d'initialisation
    - Architecture
    - Mémoire virtuelle
    - Ordonnanceur → horloges et interruptions
  - Paramètres de la ligne de commande
    - Systèmes embarqués
  - Lancement de la boîte à outils busybox
    - Ouverture du disque RAM initial
    - Chargement des modules propres au système
- Distinction entre noyau monolithique & modulaire
  - Noyau monolithique → architecture figée

Représentation des périphériques

- Contexte historique Unix
  - Tout est fichier
  - Arborescence /dev
  - Création manuelle des entrées à la demande
    - Commande /sbin/MAKEDEV fournie avec le paquet makedev
- Contexte contemporain
  - Répartition entre espaces mémoire
    - Kernelspace
    - userspace



Représentation des périphériques

- Kernelspace → espace noyau
  - Arborescences /sys et /proc
  - sysfs
    - Système de fichiers virtuel
    - Exportation des informations du noyau vers l'espace utilisateur
  - procfs
    - Système de fichiers spécial
    - Informations sur les processus
    - Paramètres de réglage des sous-systèmes du noyau
- Userspace → espace utilisateur
  - Démon udev
    - Actions déclenchées par les informations sysfs
    - Règles de configuration dans /etc/udev

Représentation des périphériques

- Démon udev et unité de disque
  - Informations sur le matériel

```
$ lspci | grep -i storage
00:06.0 SCSI storage controller: Red Hat, Inc Virtio block device
```

  - Informations collectées par udev

```
# udevadm info --query=all --name=/dev/vda
P: /devices/pci0000:00/0000:00:06.0/virtio2/block/vda
N: vda
S: disk/by-path/pci-0000:00:06.0-virtio-pci-virtio2
E: DEVLINKS=/dev/disk/by-path/pci-0000:00:06.0-virtio-pci-virtio2
E: DEVNAME=/dev/vda
E: DEVPATH=/devices/pci0000:00/0000:00:06.0/virtio2/block/vda
E: DEVTYPE=disk
E: ID_PART_TABLE_TYPE=dos
E: ID_PATH=pci-0000:00:06.0-virtio-pci-virtio2
E: ID_PATH_TAG=pci-0000_00_06_0-virtio-pci-virtio2
...
```

Représentation des périphériques

- Démon udev et interface réseau
  - Informations sur le matériel

```
$ lspci | grep -i ethernet
00:03.0 Ethernet controller: Red Hat, Inc Virtio network device
```

  - Informations collectées par udev

```
# udevadm info --query=all --path /sys/class/net/eth0
P: /devices/pci0000:00/0000:00:03.0/virtio0/net/eth0
E: DEVPATH=/devices/pci0000:00/0000:00:03.0/virtio0/net/eth0
E: ID_BUS=pci
E: ID_MODEL_ID=/sys/devices/pci0000:00/0000:00:03.0/virtio0
E: ID_VENDOR_ID=0x1af4
E: IFINDEX=2
E: INTERFACE=eth0
E: MATCHDEVID=0x0
E: MATCHIFTYPE=1
E: SUBSYSTEM=net
...
```

Accès aux autres attributs avec la commande
udevadm info --attribute-walk --path=/sys/class/net/eth0

Représentation des périphériques

- Démon udev
  - Exemple de règles sur les interfaces réseau

```
$ cat /etc/udev/rules.d/70-persistent-net.rules
# This file was automatically generated by the /lib/udev/write_net_rules
# program, run by the persistent-net-generator.rules rules file.
#
# You can modify it, as long as you keep each rule on a single
# line, and change only the value of the NAME= key.

# PCI device 0x8086:0x1502 (e1000e)
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="e4:11:5b:25:28:f5",
ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*", NAME="eth0"

# PCI device 0x8086:/sys/devices/pci0000:00/0000:00:1c.3/0000:25:00.0
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="08:11:96:b4:33:10",
ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="wlan*", NAME="wlan0"
```

Identification du composant matériel

Adresse MAC

Nom d'interface référencé dans /etc/network/interfaces

Manipulations sur les modules

- Correspondance entre matériel et nom de module

```
# lspci -k
...
00:19.0 Ethernet controller: Intel Corporation 82579LM Gigabit Network Connection (rev 04)
        Subsystem: Hewlett-Packard Company Device 1630
        Kernel driver in use: e1000e
```

- Paquet kmod
  - Ensemble des outils
    - lsmod → liste des modules chargés en mémoire
    - modprobe → (dé)chargement d'un module et de ses dépendances en mémoire
- Fichier /etc/modules
  - Liste des modules à charger obligatoirement

Manipulations sur les modules

- Paramétrage des modules
  - Fichiers du répertoire /etc/modprobe.d/
  - Exemple pour une interface wifi

```
# cat /etc/modprobe.d/ieee80211.conf
options cfg80211 ieee80211_regdom=EU
# lsmod | grep cfg80211
cfg80211              137140  2 mac80211,iwlwifi
rfkill                 19012  5 bluetooth,cfg80211,hp_wmi
# grep cfg80211 /var/log/kern.log
…
cfg80211: Calling CRDA for country: EU
cfg80211: Regulatory domain changed to country: EU
cfg80211:     (start_freq - end_freq @ bandwidth), (max_antenna_gain, max_eirp)
cfg80211:     (2402000 KHz - 2482000 KHz @ 40000 KHz), (N/A, 2000 mBm)
cfg80211:     (5170000 KHz - 5250000 KHz @ 40000 KHz), (N/A, 2000 mBm)
cfg80211:     (5250000 KHz - 5330000 KHz @ 40000 KHz), (N/A, 2000 mBm)
cfg80211:     (5490000 KHz - 5710000 KHz @ 40000 KHz), (N/A, 2700 mBm)
```

Manipulations sur les modules

- Applications
  - Gestion des capteurs de carte mère
    - Bus I2C non hotplug
    - Exécution du script sensors-detect du paquet lm-sensors
    - Scrutation du bus I2C
    - Liste des composants reconnus → modules à charger manuellement
- Recherches sur les modules
  - Comment retrouver un module dans l'arborescence système ?
  - Comment charger un module en mémoire ?
  - Comment vérifier qu'un module est chargé ?
  - Quelles sont les informations udev sur l'interface réseau ?

Dynamic Kernel Module Support (dkms)

- Environnement de mise à jour autonome de modules
  - Compilation
  - Construction de paquets
  - (dés)installation
- Gestion des modules analogue à celle des paquets
  - Application d'un processus d'assurance qualité
  - Possibilité de publier des modules «propriétaires» indépendants
- Exemple pratique
  - Installer le paquet xtables-addons-dkms

Initialisation des processus

- 2 processus initiaux
  - Processus idle
    - Occupe le processeur pendant les «temps morts»
  - Processus init
    - Assure le séquencement du démarrage du système
    - Contrôle l'accès à la partition racine
    - Exécute le programme /sbin/init

```
ps aux --sort=pid
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  10636   796 ?        Ss   09:58   0:00 init [2]
root         2  0.0  0.0      0     0 ?        S    09:58   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    09:58   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S    09:58   0:00 [kworker/u:0]
root         6  0.0  0.0      0     0 ?        S    09:58   0:00 [migration/0]
root         7  0.0  0.0      0     0 ?        S    09:58   0:00 [watchdog/0]
root         8  0.0  0.0      0     0 ?        S    09:58   0:00 [migration/1]
```

Niveaux de démarrage - runlevels

- 7 niveaux

  http://refspecs.linuxfoundation.org/lsb.shtml

```
0 halt
1 single user mode
2 multiuser with no network services exported
3 normal/full multiuser
4 reserved for local use, default is normal/full multiuser
5 multiuser with a display manager or equivalent
6 reboot
```

- Niveaux définis dans /etc/inittab
  - Debian → niveau 2 par défaut → id:2:initdefault:
- 1 niveau → un ensemble de scripts à exécuter

Niveaux de démarrage - runlevels

- Scripts d'initialisation des services
  - Répertoire /etc/init.d
  - Fonctions usuelles
    - start → lancement du service
    - stop → arrêt du service
    - reload → rechargement de la configuration
    - restart → redémarrage complet du service
  - Appel direct du script ou commande service

```
# /etc/init.d/apache2 reload
[....] Reloading web server config: apache2
. ok
# service apache2 reload
[....] Reloading web server config: apache2
. ok
```

Niveaux de démarrage - runlevels

- Scripts d'initialisation des services
  - Exemple de script → /etc/init.d/atd

```
#! /bin/sh
### BEGIN INIT INFO
# Provides:          atd
# Required-Start:    $syslog $time $remote_fs
# Required-Stop:     $syslog $time $remote_fs
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Deferred execution scheduler
# Description:       Debian init script for the atd deferred
#                    executions  scheduler
### END INIT INFO
#
# Author:       Ryan Murray <rmurray@debian.org>
#

PATH=/bin:/usr/bin:/sbin:/usr/sbin
DAEMON=/usr/sbin/atd
PIDFILE=/var/run/atd.pid
```

description des dépendances

```
test -x $DAEMON || exit 0

. /lib/lsb/init-functions

case "$1" in
  start)
        log_daemon_msg "Starting deferred execution scheduler" "atd"
        start_daemon -p $PIDFILE $DAEMON
        log_end_msg $?
    ;;
  stop)
        log_daemon_msg "Stopping deferred execution scheduler" "atd"
        killproc -p $PIDFILE $DAEMON
        log_end_msg $?
    ;;
  force-reload|restart)
    $0 stop
    $0 start
```

instructions pour chaque commande

Niveaux de démarrage - runlevels

- Répertoires /etc/rc*.d
  - Liens symboliques vers les scripts du répertoire /etc/init.d
  - Un même script → exécution à des niveaux différents
  - Lien débutant par 'S' → commande start
  - Lien débutant par 'K' → commande stop
  - Numéro d'ordre d'exécution après la première lettre
  - Exemple du service atd

```
$ find /etc/rc*.d -type l -iname \*atd
/etc/rc0.d/K01atd
/etc/rc1.d/K01atd
/etc/rc2.d/S18atd
/etc/rc3.d/S18atd
/etc/rc4.d/S18atd
/etc/rc5.d/S18atd
/etc/rc6.d/K01atd
```

Bilan séance 6

- Initialisation du système
  - Gestionnaire d'amorce
    - GRUB2
  - Noyau Linux
    - Contrôle de l'empreinte mémoire
- Modules du noyau
  - 2 espaces mémoire
    - kernelspace → sysfs
    - userspace → udev
- Démons & Services
  - Gestion autonome de chaque processus

BIOS	Basic Input Ouput System → recherche MBR
MBR	Master Boot Record → recherche GRUB
GRUB	Grand Unified Bootloader → recherche noyau
kernel	Noyau Linux → exécution /sbin/init
runlevels	Niveaux de démarrage → exécution des services