

[inetdoc.LINUX]

<http://www.linux-france.org/prj/inetdoc>

Exploration GNU/Linux - Séance 6

Initialisation système & noyau Linux

Niveaux de démarrage

Chargement des pilotes de périphériques



Philippe Latu

philippe.latu@linux-france.org

IUT 'A' Paul Sabatier - STRI

Objectifs

- Identifier les étapes de sélection du système
- Analyser les niveaux de démarrage
 - Runlevels
- Caractériser l'occupation mémoire du noyau
 - Kernelpspace / Userspace
- Analyser la gestion des pilotes de périphériques
 - Sysfs / Udev / Hotplug

Initialisation du système

- BIOS (Basic Input-Output System)
 - Premier programme appelé par le processeur
 - Analyse de la configuration matérielle
 - ▶ Test RAM
 - ▶ Identification processeur
 - ▶ Identification unités de stockage
- Recherche du système d'exploitation
 - Ordre défini par la configuration BIOS
 - Pour chaque unité désignée
 - ▶ Lecture du Master Boot Record (MBR)
 - Lancement du système
 - ▶ Si un code Boot Loader est présent

Initialisation du système

- Master Boot Record

- Contient un code appelé Boot Loader
- Désigne la partition d'amorçage
- Accède au gestionnaire de démarrage

- Gestionnaire de démarrage

- LILO

- ▶ Nouvelle écriture du MBR à chaque changement de configuration
- ▶ Accès direct au disque via le BIOS

- GRUB

- ▶ Fichier de configuration lu à chaque démarrage
- ▶ Accède aux partitions ext(2|3)

- Partition /boot

- Contient les fichiers compressés des noyaux
- Contient les paramètres du gestionnaire de démarrage

Initialisation du système

● Partition /boot

```
# mount | grep oot
```

```
/dev/mapper/Casper-root on / type ext3 (rw,errors=remount-ro)
```

```
/dev/sda1 on /boot type ext3 (rw) # partition formatée exclue du gestionnaire LVM
```

● Répertoire /boot

```
# ls -lX /boot/
```

```
grub # répertoire configuration GRUB
```

```
lost+found
```

```
config-2.6.32.2
```

```
initrd.img-2.6.32.2 # disque RAM initialisé au démarrage du système
```

```
System.map-2.6.32.2 # carte des appels de fonctions
```

```
vmlinuz-2.6.32.2 # partie monolithique du noyau
```

```
config-2.6.30-2-amd64 # configuration du noyau distribuée avec le paquet
```

```
initrd.img-2.6.30-2-amd64
```

```
System.map-2.6.30-2-amd64
```

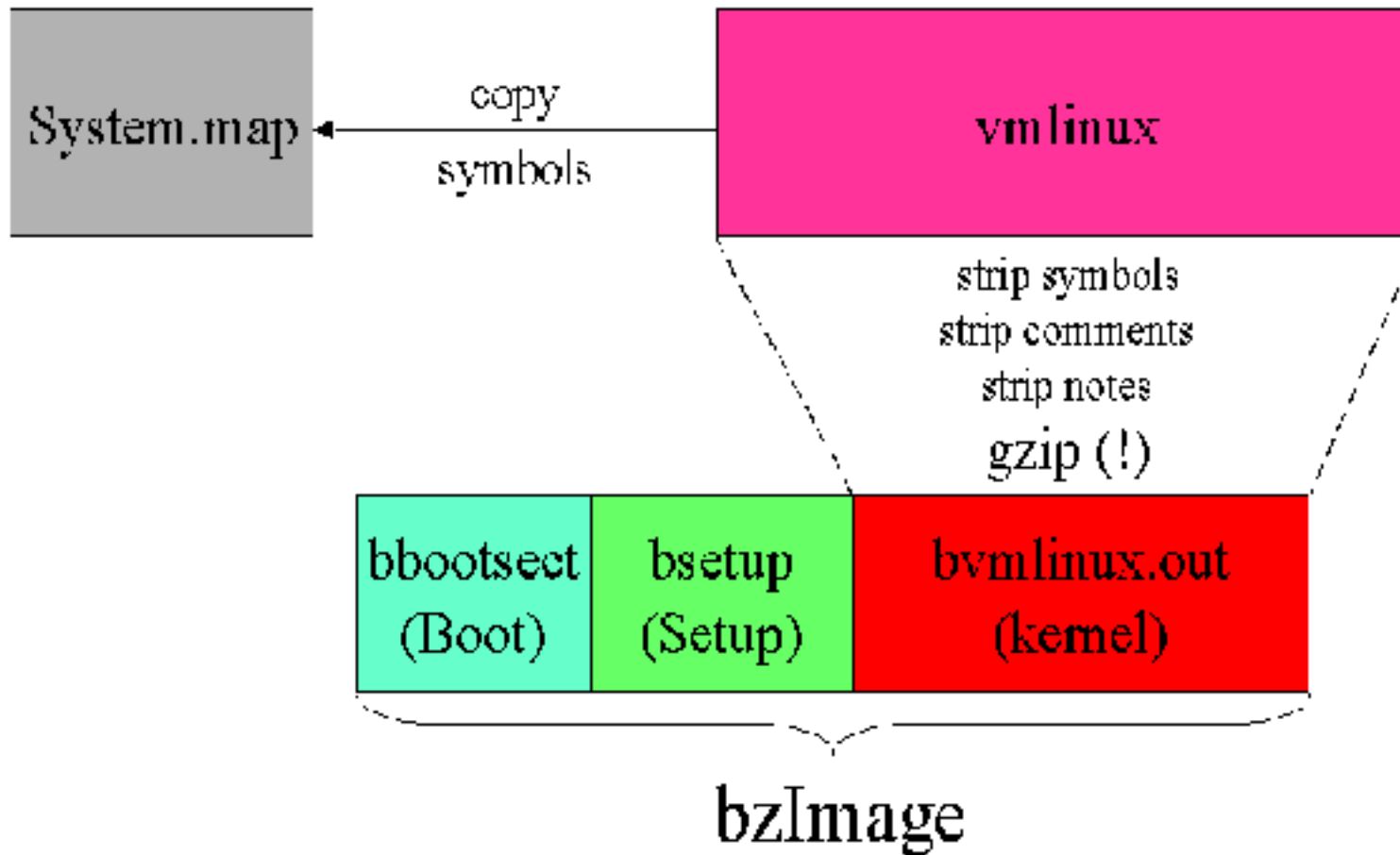
```
vmlinuz-2.6.30-2-amd64
```

Initialisation du système

- Partie monolithique vmlinux-2.6.xx

- Fichier bzImage

- ▶ Image compressée du noyau
- ▶ Éclatée en plusieurs zones mémoire discontinues



Initialisation du système

- Disque RAM initial
 - Ouvert après le chargement de la partie monolithique
 - Shell minimal + Outils BusyBox
 - Arborescence des modules

```
$ mkdir initrd ; cd initrd
```

```
$ gzip -dc /boot/initrd.img-2.6.xx | cpio -idv
```

```
$ tree -L 1
```

```
.  
|-- bin  
|-- conf  
|-- etc  
|-- init  
|-- lib -----> modules  
|-- lib64      '-- 2.6.xx  
|-- sbin  
'-- scripts
```

Initialisation du système

- Ajout d'un nouveau noyau
 - Étapes rituelles à la mode Debian
 - ▶ L'utilisateur normal etu appartient au groupe src
 - ▶ Paquets installés : kernel-package fakeroot libncurses-dev

```
$ wget http://www.eu.kernel.org/pub/linux/\
    kernel/v2.6/linux-2.6.xx.tar.bz2
$ mv linux-2.6.xx.tar.bz2 /usr/src/
$ cd /usr/src/
$ tar xf linux-2.6.xx.tar.bz2
$ ln -s linux-2.6.xx linux
$ cd linux
$ cp /boot/config-2.6.xx .config
$ make menuconfig # menus de configuration des fonctions du noyau
$ make-kpkg --rootcmd fakeroot --initrd kernel_image
$ cd .. ; su
# dpkg -i linux-image-2.6.xx-10.00.Custom_amd64.deb
```

Initialisation du système

- Ajout d'un nouveau noyau
 - Menus de configuration

```
.config - Linux Kernel v2.6.23.14 Configuration

Linux Kernel Configuration

Arrow keys navigate the menu. <Enter> selects submenus --->. Highl
excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> f
excluded <M> module < > module capable

General setup --->
[*] Enable loadable module support --->
--- Enable the block layer --->
Processor type and features --->
Power management options --->
Bus options (PCI etc.) --->
Executable file formats / Emulations
Networking --->
Device Drivers --->
Firmware Drivers --->
File systems --->
Instrumentation Support --->
Kernel hacking --->
Security options --->
--- Cryptographic API --->
Library routines --->
---
Load an Alternate Configuration File
Save an Alternate Configuration File
```

Initialisation du système

- Ajout d'un nouveau noyau
 - Menus de configuration
 - ▶ Arborescence étendue et complexe
 - ▶ Difficile à prendre en main
 - ▶ Partir d'une configuration fonctionnelle
 - Configuration existante
 - ▶ Configuration distribuée avec le paquet
 - ▶ Exemple du paquet linux-image-2.6-amd64
 - ▶ Fichier /boot/config-2.6.22-3-amd64
 - Options de configuration
 - ▶ Points essentiels d'un sous-système
 - ▶ Équivaut à la table des matières du cours

Initialisation du système

- Ajout d'un nouveau noyau

- Configuration GRUB

- ▶ Script update-grub fourni avec le paquet grub
- ▶ Mise à jour automatisée du menu du gestionnaire de démarrage

```
# update-grub
```

```
Generating grub.cfg ...
```

```
Found linux image: /boot/vmlinuz-2.6.32.2 # noyau ajouté
```

```
Found initrd image: /boot/initrd.img-2.6.32.2
```

```
Found linux image: /boot/vmlinuz-2.6.30-2-amd64 # noyau fourni par la distribution
```

```
Found initrd image: /boot/initrd.img-2.6.30-2-amd64
```

```
done
```

- Personnalisation de GRUB

```
# ll /etc/grub.d/
```

```
total 18K
```

```
-rwxr-xr-x 1 root root 3,4K janv.  1 00:49 00_header
```

```
-rwxr-xr-x 1 root root 1,4K déc.  10 00:47 05_debian_theme
```

```
-rwxr-xr-x 1 root root 3,6K déc.  14 15:05 10_linux
```

```
-rwxr-xr-x 1 root root 5,5K janv.  1 00:49 30_os-prober
```

```
-rwxr-xr-x 1 root root 214 sept. 12 15:47 40_custom
```

```
-rw-r--r-- 1 root root 483 mars  17 2009 README
```

Initialisation du noyau

- Processeurs Intel
 - Passage en mode protégé
 - Passage en mode 32 bits
- Séquence d'initialisation
 - Architecture
 - Mémoire virtuelle
 - Ordonnanceur / Scheduler
 - ▶ Interruptions et horloge
- Paramètres de la ligne de commande
- Chargement des modules
 - 2 possibilités
 - ▶ Montage de la partition racine en lecture seule
 - ▶ Montage d'une racine en RAM = initrd

Initialisation des processus

● 2 Processus initiaux

■ Processus idle

- ▶ Occupe le processeur pendant les «temps morts»

■ Processus init

- ▶ Assure le séquençement du démarrage du système
- ▶ Contrôle l'accès à la partition racine
- ▶ Exécute le programme init
- ▶ Fichier /sbin/init pour Debian

■ Processus init

- ▶ Processus parent de tous les autres excepté idle
- ▶ Documentation

\$ man init

Niveaux de démarrage - runlevels

- 7 niveaux appelés runlevels

- ▶ <http://www.linuxbase.org/spec/lbreview.html>

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 xdm or equivalent

6 reboot

- Niveaux définis dans le fichier /etc/inittab

- Debian = id:2:initdefault:

- Conditions de réinitialisation

What to do when CTRL-ALT-DEL is pressed.

ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now

- Un niveau = un ensemble de scripts à exécuter

Niveaux de démarrage - runlevels

- Répertoire /etc/init.d
 - Code source des scripts
 - Commandes conventionnelles
 - ▶ start = démarrage du service
 - ▶ stop = arrêt du service
 - ▶ reload = rechargement de la configuration
 - ▶ restart = redémarrage du service
 - Exemple du redémarrage des connexions réseau
 - ▶ /etc/init.d/networking restart

force-reload|restart)

```
process_options
```

```
log_action_begin_msg "Reconfiguring network interfaces"
```

```
ifdown -a --exclude=lo || true
```

```
if ifup -a --exclude=lo; then
```

```
    log_action_end_msg $?
```

```
else
```

```
    log_action_end_msg $?
```

```
fi
```

```
::;
```

Niveaux de démarrage - runlevels

- Répertoire /etc/init.d

- Exemple de code

- ▶ Extrait du script /etc/init.d/atd

```
case "$1" in
start)
    log_daemon_msg "Starting deferred execution scheduler" "atd"
    start_daemon $DAEMON
    log_end_msg $?
    ;;
stop)
    log_daemon_msg "Stopping deferred execution scheduler" "atd"
    killproc $DAEMON
    log_end_msg $?
    ;;
force-reload|restart)
    $0 stop
    $0 start
    ;;
*)
    echo "Usage: /etc/init.d/atd {start|stop|restart|force-reload}"
    exit 1
    ;;
esac
```

Niveaux de démarrage - runlevels

- Répertoire /etc/rcS.d
 - Liens symboliques
 - Scripts à exécuter indépendamment du niveau
 - Exemple du jeu de caractères du clavier
 - ▶ S05keymap.sh -> ../init.d/keymap.sh
 - /etc/rcS.d remplace /etc/rc.boot obsolète
- Répertoires '/etc/rc0.d' à '/etc/rc6.d'
 - Liens symboliques
 - Scripts à exécuter en fonction du niveau choisi
 - ▶ Liens commençant par 'S??' = commande 'start'
 - ▶ Liens commençant par 'K??' = commande 'stop'
 - Exemple du service acpid

```
$ find /etc/rc?.d -type l -name "*acpid" -printf "%p "
```

/etc/rc0.d/K20acpid /etc/rc1.d/K20acpid \
/etc/rc2.d/S20acpid /etc/rc3.d/S20acpid /etc/rc4.d/S20acpid /etc/rc5.d/S20acpid \
/etc/rc6.d/K20acpid

Niveaux de démarrage - runlevels

- Éditeur Sys V KDE
 - Banque de script à gauche
 - Runlevels de 0 à 6

Services disponibles	Niveau d'exécution 0		Niveau d'exécution 1		Niveau d'exécution 2		Niveau d'exécution 3		Niveau d'exécution 4		Niveau d'exécution 5		Niveau d'exécution 6	
	Num.	Nom	Num.	Nom	Num.	Nom	Num.	Nom	Num.	Nom	Num.	Nom	Num.	Nom
acpid	20	@ sendmail	30	@ killprocs	05	@ loadcpufreq	05	@ loadcpufreq	05	@ loadcpufreq	05	@ loadcpufreq	05	@ loadcpufreq
acpi-suppo	30	@ urandom	00	@ single	05	@ vbesave	05	@ vbesave	05	@ vbesave	05	@ vbesave	05	@ vbesave
alsa-utils	31	@ umountfs			10	@ syslogd	10	@ syslogd	10	@ syslogd	10	@ syslogd	10	@ syslogd
anacron	35	@ network			11	@ klogd	11	@ klogd	11	@ klogd	11	@ klogd	11	@ klogd
apache2					15	@ bind9	15	@ bind9	15	@ bind9	15	@ bind9	15	@ bind9
atd														
avahi-daemon	00	@ DMF	00	@ DMF	00	@ DMF	00	@ DMF	00	@ DMF	00	@ DMF	00	@ DMF
bind9	01	@ kdm	01	@ cdm	08	@ vmware					08	@ vmware	01	@ kdm
bluetooth	01	@ timidity	01	@ timidity									01	@ timidity
brctl	08	@ vmware	09	@ apache2									08	@ vmware
brctl	09	@ apache2	11	@ anacron									09	@ apache2

Chargement du noyau

- 2 structures de base
 - Monolithique
 - ▶ Structure compacte
 - ▶ Toutes les ressources sont intégrées au noyau
 - Modulaire
 - ▶ Structure minimaliste
 - ▶ Ajout|Retrait de ressources en cours d'exécution
- Choix d'intégration = configuration du noyau
 - Structure modulaire
 - ▶ Utilisable sur plusieurs configurations matérielles
 - ▶ Un noyau de distribution est «massivement» modulaire
 - Structure monolithique
 - ▶ Consomme moins de mémoire
 - ▶ Accès direct aux pilotes sans tableaux d'adressage
 - ▶ Distinction kernelspace/userspace

Chargement des modules

- Paquet module-init-tools

- Commandes de manipulation des modules

\$ apt-cache show module-init-tools

- Répertoires /lib/modules/2.6.xx/

\$ ls -lA /lib/modules/

- Commandes usuelles

\$ lsmod # liste les modules chargés en mémoire

modprobe -v <nom_du_module> # chargement d'un module + dépendances

modprobe -rv <nom_du_module> # déchargement d'un module

Représentation des périphériques

- Contexte historique Unix

- Tout est système de fichiers
- Arborescence des périphériques
 - ▶ /dev
- Création manuelle des objets à la demande
 - ▶ MAKEDEV
- Linux 2.4 - *BSD - Solaris
 - ▶ Représentation dans l'espace noyau
 - ▶ Système de fichiers devfs
 - ▶ Code difficile à maintenir

- Contexte contemporain

- Répartition des fonctions entre espaces mémoire
 - ▶ Noyau = inventaire du matériel et communications
 - ▶ Utilisateur = chargement automatisé des pilotes

Représentation des périphériques

- Espace noyau - kernelspace

- Fourniture de service

- ▶ sysfs = système de fichier dédié
- ▶ /proc et /sys
- ▶ Unix domain sockets
- ▶ Fonctions réseau

- Espace utilisateur - userspace

- Construction de l'arborescence /dev

- ▶ Démon udev
- ▶ Démarrage au niveau /etc/rcS.d/
- ▶ Réaction aux évènements matériels
- ▶ Connexions à chaud - hotplug = Unités de stockage, RAM, CPUs

- Paquet Debian udev

- ▶ Configuration dans /etc/udev
- ▶ Jeux de règles d'initialisation des périphériques

Représentation des périphériques

- Démon udev et stockage

- Informations matérielles

```
# lspci | grep 02:0e.0
```

```
02:0e.0 RAID bus controller: Dell PowerEdge Expandable RAID controller 5
```

- Informations reconnues par le noyau

```
# less /var/log/kern.log
```

```
scsi0 : LSI SAS based MegaRAID driver
```

```
scsi 0:0:0:0: Direct-Access SEAGATE ST3500620SS MS07 PQ: 0 ANSI: 5
```

```
scsi 0:0:1:0: Direct-Access SEAGATE ST3500620SS MS07 PQ: 0 ANSI: 5
```

```
scsi 0:0:2:0: Direct-Access SEAGATE ST3500620SS MS07 PQ: 0 ANSI: 5
```

```
scsi 0:0:3:0: Direct-Access SEAGATE ST3500620SS MS07 PQ: 0 ANSI: 5
```

```
scsi 0:0:4:0: Direct-Access SEAGATE ST3500620SS MS07 PQ: 0 ANSI: 5
```

```
scsi 0:2:0:0: Direct-Access DELL PERC 5/i 1.03 PQ: 0 ANSI: 5
```

```
sd 0:2:0:0: [sda] 2927099904 512-byte hardware sectors: (1.49 TB/1.36 TiB)
```

- Informations collectées

```
# udevadm info --query=all --name=/dev/sda
```

```
P: /devices/pci0000:00/0000:00:05.0/0000:01:00.0/0000:02:0e.0/host0/target0:2:0/0:2:0:0/block/sda
```

```
N: sda
```

```
W: 31
```

```
S: block/8:0
```

```
S: disk/by-id/scsi-360022190c6377400118c892518e63397
```

```
S: disk/by-id/wwn-0x60022190c6377400
```

```
S: disk/by_path/pci-0000:02:0e.0-scsi-0:2:0:0
```

Représentation des périphériques

- Démon udev et interfaces réseau

- Informations matérielles

```
# lspci | grep 0f:00.0
```

```
0f:00.0 Ethernet controller: Intel Corporation 82575GB Gigabit Network Connection (rev 02)
```

- Informations reconnues par le noyau

```
# grep igb /var/log/kern.log
```

```
igb 0000:0f:00.0: Intel(R) Gigabit Ethernet Network Connection
```

```
igb 0000:0f:00.0: eth1: (PCIe:2.5Gb/s:Width x4) 00:1b:21:36:35:1c
```

```
igb 0000:0f:00.0: eth1: PBA No: d96950-006
```

```
igb 0000:0f:00.0: Using MSI-X interrupts. 4 rx queue(s), 4 tx queue(s)
```

- Informations collectées

```
# udevadm info -a -p /sys/class/net/eth1
```

```
...
```

```
looking at device '/devices/pci0000:00/0000:00:04.0/0000:0c:00.0/0000:0d:04.0/0000:0f:00.0/net/eth1':
```

```
  KERNEL=="eth1"
```

```
  SUBSYSTEM=="net"
```

```
  DRIVER=="
```

```
  ATTR{addr_len}=="6"
```

```
...
```

```
  ATTR{link_mode}=="0"
```

```
  ATTR{address}=="00:1b:21:36:35:1c"
```

```
  ATTR{broadcast}=="ff:ff:ff:ff:ff:ff"
```

Représentation des périphériques

- Démon udev

- 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 0x8086:0x10d6 (igb) <----- Identification composant
```

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:1b:21:36:35:18", \
    ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*", NAME="eth2"
```

```
# PCI device 0x8086:0x10d6 (igb) Adresse MAC --.
```

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:1b:21:36:35:1c", \
    ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*", NAME="eth1"
```

```
# PCI device 0x8086:0x10d6 (igb)
```

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="00:1b:21:36:35:1d", \
    ATTR{dev_id}=="0x0", ATTR{type}=="1", KERNEL=="eth*", NAME="eth0"
```

^

Nom d'interface -----'

Chargement des modules

- Script `/etc/init.d/module-init-tools`
 - Gestion du chargement des modules au
- Fichier `/etc/modules`
 - Liste des modules à charger sans règles udev
- Répertoire `/etc/modprobe.d/`
 - Paramétrage manuel des modules
 - ▶ Exemple des paramètres de réglage Wifi

```
$ cat /etc/modprobe.d/ieee80211
```

```
options cfg80211 ieee80211_regdom=EU
```

```
$ lsmod |grep cfg80211
```

```
cfg80211          37856  2 iwl3945,mac80211
```

```
$ dmesg | grep cfg80211
```

```
cfg80211: Using static regulatory domain info
```

```
cfg80211: Regulatory domain: EU
```

Chargement des modules

- Chargement manuel des modules
 - 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
- Application
 - Recherche des modules
 - ▶ Comment retrouver un module dans l'arborescence du noyau ?
 - ▶ Comment charger un module en mémoire ?
 - ▶ Comment vérifier qu'un module est chargé ?
 - Exploitation des capteurs
 - ▶ Configurer les plasmoides d'affichage des informations sur les capteurs

Synthèse

- Initialisation système

- Noyau Linux

- ▶ Contrôle précis de l'empreinte mémoire
- ▶ Chargement en mémoire des modules correspondant aux composants

- Démons et services indépendants et autonomes

- ▶ runlevels
- ▶ /etc/init.d/<nom_service> restart
- ▶ Éviter Ctrl+Alt+Suppr sur les serveurs en production ;))

- Gestion des périphériques

- Distinction claire entre espaces mémoire

- ▶ kernlespace = sysfs
- ▶ userspace = udev