

# **Administration**

# **G N U - L I N U X**

# PLAN

1. Introduction
2. Installation du système
3. FHS
4. Démarrage et arrêt
5. Installation et dés-installation des paquets
6. Gestion des utilisateurs
7. Impression sous Linux
8. Gestion des Systèmes de Fichiers
9. Noyau : Compilation et Installation
10. Journalisation et Observation
11. Configuration réseau

# Introduction

- Termes à définir
  - UNIX
  - GNU/LINUX
  - Distributions GNU/LINUX

# Unix

- UNIX™ est le nom d'un système d'exploitation multitâche et multiutilisateur créé en 1969, à usage principalement professionnel, conceptuellement ouvert et fondé sur une approche par laquelle il offre de nombreux petits outils chacun dotés d'une mission spécifique.
- Il a donné naissance à une famille de systèmes, dont les plus populaires sont Linux, Mac OS X et Solaris.
- On nomme famille Unix l'ensemble de ces systèmes. On dit encore qu'ils sont de type Unix
- Il existe aujourd'hui un ensemble de standards réunis sous la norme POSIX qui vise à unifier certains aspects de leur fonctionnement.

**Wikipédia**

# Unix : caractéristiques 1

- Écrit à 98% en langage C, portable.
- Énormément de versions (HP/UX, XENIX, AIX, SOLARIS, IRIX, LINUX) se rapprochant des 2 branches SysV & BSD.
- Multi-tâches
- Multi-utilisateurs
- multi-plateformes : Intel, Motorola (Apple MAC et Amiga), Sun Sparc, DEC Alpha, ...
- Plusieurs interfaces graphiques, aisées; plusieurs systèmes d'interpréteurs de commandes (*shell*).

# Unix : caractéristiques 2

- Gestion hiérarchique des fichiers.
- Sécurité par fichier (fonctions & appartenance), info temporelles.
- Indépendance des périphériques (son succès!).  
Notion de *device file*.
- Exécution en tâche de fond possible.
- Redirection des I/O.

# Unix : avantages & inconvénients

- Avantages multiples :
  - Très grande fiabilité.
  - Multi plateformes (tout processeur),
  - Patrimoine applicatif très riche (FTP, news, WWW, messagerie, compilo, outils d'admin & sécurité, etc.).
  - Enormément de logiciels free/shareware.
- Inconvénients :
  - Trop de standards : portage difficile.
  - Administration pour les avertis

# GNU

- Acronyme récursif qui signifie en anglais "Gnu's Not Unix"
- Système libre conçu pour être compatible avec Unix mais complètement nouveau (par l'absence de tout code source d'origine Unix).
- Le projet GNU est lancé par Richard Stallman en 1984 afin de créer un système d'exploitation libre et complet.
- En 1985, Stallman crée la Free Software Foundation (FSF), structure logistique, légale et financière du projet GNU



# GNU - Linux

- À partir de 1990, le système GNU dispose d'un ensemble important d'applications: éditeurs, compilateurs, bibliothèques système...etc. Le principal composant encore manquant étant le noyau.
- Le projet GNU avait prévu le développement du noyau Hurd pour compléter le système, mais au début des années 1990, Hurd ne fonctionnait pas encore et son développement rencontrait encore des difficultés
- L'arrivée du noyau Linux (fin 1991) compléta le projet GNU
- En janvier 2004, l'Unesco a inscrit comme "Trésor du monde" le projet GNU.

# Linux

- Au sens strict, Linux est le nom du noyau de système d'exploitation libre, multitâche, multiplate-forme et multi-utilisateur de type UNIX créé par Linus Torvalds.
- Par extension, Linux désigne couramment le système d'exploitation libre combinant le noyau et un ensemble d'utilitaires GNU (emacs, gcc, ...) et d'applications libres (Xwindow, LaTeX, ...)
- Pour désigner cet ensemble, la Free Software Foundation (FSF) soutient la désignation GNU/Linux afin de rappeler que le noyau Linux est généralement distribué avec de nombreux logiciels ainsi que l'infrastructure du projet GNU.

# Distribution Linux

- Pour l'utilisateur final, Linux se présente sous la forme d'une distribution Linux.
- Une distribution Linux (ou distribution GNU/ Linux) est un ensemble cohérent de logiciels rassemblant un noyau Linux, des logiciels issus du projet GNU, et des logiciels supplémentaires, le plus souvent libres.
- Les distributions comprennent le plus souvent un logiciel d'installation et des outils de configuration.
- Il existe de nombreuses distributions, chacune ayant ses particularités : certaines sont dédiées à un usage spécifique (pare-feu, routeur, grappe de calcul...), d'autres à un matériel spécifique, par contre les grandes distributions restent à usage générale.

# Distributions Linux

Liste non exhaustive des distributions :

- RedHat – Fedora - CentOS
- Mandriva
- Debian
- Suse
- Ubuntu
- Slackware
- Caldera
- Gentoo
- ...

+ mini-distributions: Trinux, DosLinux, ...

# GNU / Linux : le présent

- Support de plusieurs architectures (x86, ppc, ...)
- Multiples interfaces graphiques utilisateur,
- Gestion multiprocesseurs
- Facilité (modularité) d'installation,
- Réactivité et adaptation au matériel récent et aux besoins spéciaux (temps réel, sécurité, etc),
- Existence de centres de services
- Outils dédiés à ce système.

# **Installation Redhat - Fedora**

# Matériel requis

- Le minimum requis dépend principalement de la destination et de sa version :
- Pour Fedora 16 (version x86)
  - Processeur Pentium 3
  - Mémoire : min 768 Mo, recommandé 1152 Mo
  - Disque :
    - 9 Go pour une installation complète + l'espace swap
    - + 200Mo nécessaire au moment de l'installation
    - + espace de travail des utilisateurs.

# Préparatifs

- Pour disposer d'un système en « dual-boot » il est préférable (avant de démarrer l'installation) de revoir les partitions existantes afin de réserver l'espace nécessaire à l'installation du nouveau système
- Outils (Défragmentation + Partitionnement)
  - Parted, gparted,
  - fdisk,
  - Partition Magic,
  - fips,
  - ....



# Démarrage

- Pour installer le système on peut démarrer à partir d'un :
  - Support (CD, DVD ou clé USB) contenant une version LIVE
  - Support (DVD ou clé USB) d'installation
  - Support (CD, DVD ou clé USB) de démarrage minimal
  - Serveur d'installation via PXE boot

# Média d'installation

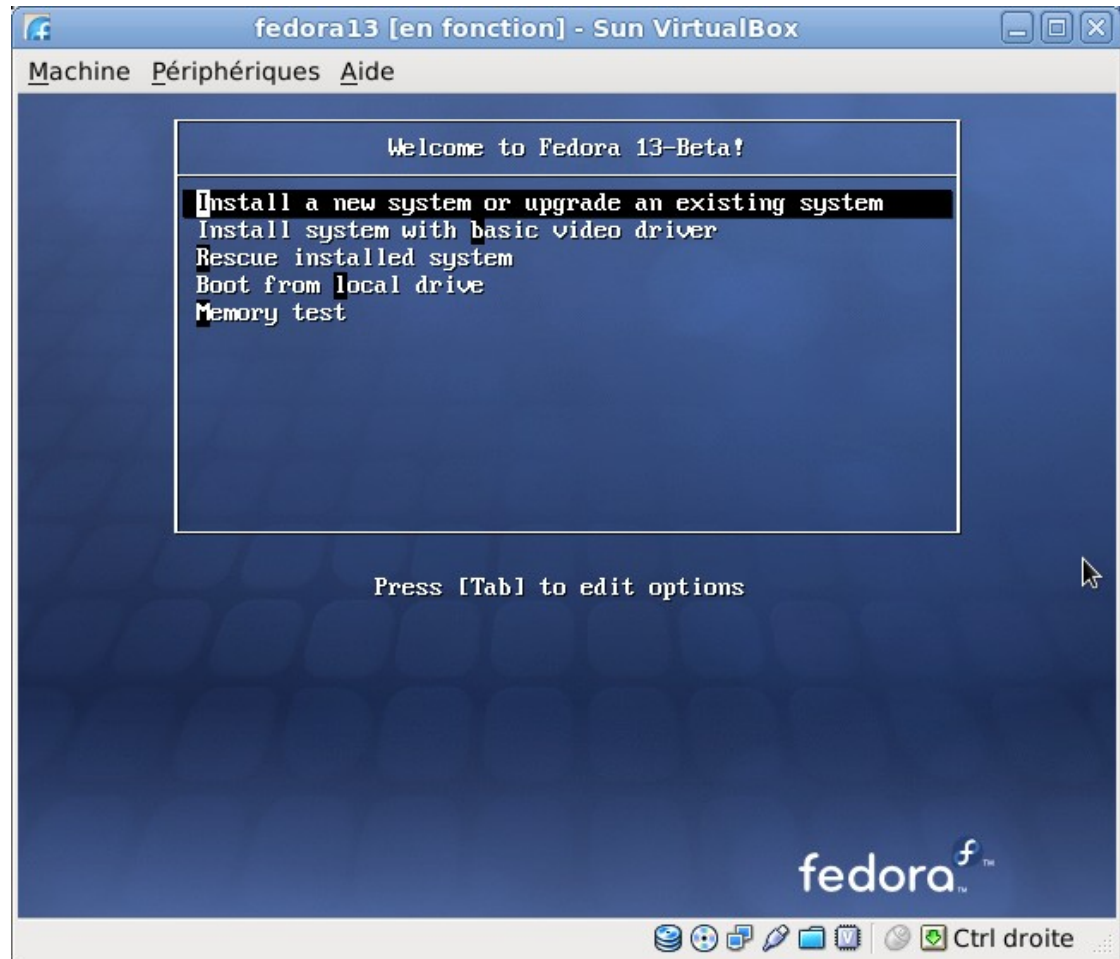
- Après le démarrage l'installation peut continuer à partir :
  - Du support de démarrage lui même s'il contient une version complète d'installation ou une version *Live*.
  - Du disque dur
  - D'un serveur FTP ou HTTP
  - D'un partage NFS

# Plan d'installation

- Démarrer le système à partir d'une unité "bootable" : DVD, CD, ou clé USB
- Le système chargé en mémoire crée un Ramdisk et démarre l'installation qui vous demande de :
  - Choisir le langage et le type du clavier
  - Configurer le time\_zone
  - Choisir le mot de passe pour le compte de l'administrateur.
  - Créer les partitions
  - Installer et configurer le chargeur GRUB
  - Choisir les paquetages à installer

# Choix du mode d'installation

Ci-contre la  
première image  
qui s'affiche  
lorsque  
l'ordinateur  
démarré sur le  
CD/DVD  
d'installation  
de Fedora



# Choix du mode d'installation

- Le premier écran d'installation offre plusieurs possibilités :
  - Installer ou mettre à jour Fedora mode graphique ;
  - Installer ou mettre à jour Fedora mode graphique dégradée ;
  - Dépanner une installation existante ;
  - Booter depuis le disque dur
  - Test de la mémoire physique (RAM).
- Pour régler certains problèmes d'affichage lors de l'installation ou passer outre la détection automatique du matériel, plusieurs options sont possibles. Pour pouvoir les entrer, il suffit d'appuyer sur la touche [ECHAP] du clavier une fois le menu apparaît .

# Options

Quelques options disponibles:

- `linux vesa` : pour utiliser les pilotes Vesa si on a des problèmes d'affichage
- `Linux 1024*768` : pour configurer la résolution de l'écran si elle est mal détectée
- `linux noapic` : APIC = Advanced Programmable Interrupt Controller
- `linux acpi=off` : ACPI = Advanced Configuration and Power Interface (pour désactiver la gestion de l'énergie)
- `linux clocksource=acpi_pm` : Utilisé si l'installation se bloque à `/sbin/loader`
- `linux pci=noms` : Utilisé pour la reconnaissance de certains disques en SATA ;
- `linux vnc` : Utilisé pour l'installation avec vnc.

Remarque : pour enchaîner les options, les ajouter à la suite les une des autres:  
`linux vesa pci=noms`

# Vérification du support

L'écran suivant vous demande si vous voulez vérifier l'intégrité de vos/votre CD/DVD. Il est recommandé d'effectuer cette vérification, cela évitera des mauvaises surprises lors de l'installation.



# Langue & clavier

- Après la vérification de l'intégrité du support, le système vous demande de choisir votre langue puis la disposition de votre clavier.
- Pour un clavier « azerty » français vous avez le choix entre le Latin-1 et le Latin-9, ce deuxième présente plus de caractère pour la langue française (notamment le signe €), il est donc fortement conseillé de le choisir.
- Le clavier proposé par l'outil d'installation (Anaconda) est fonction de la langue choisie est non suite à une détection automatique. Veuillez alors le modifier si nécessaire.



# Type d'installation

Nouveau dans Fedora 13, Anaconda vous propose de choisir entre une installation sur un disque local (SATA, SCSI, USB, ...) ou sur un baie de stockage externe (SAN, iSCSI...).




# périphériques de stockage



L'écran de sélection des périphériques de stockage affiche tous les périphériques de stockage auxquels anaconda a accès. Sélectionner un disque permet de rendre disponible durant l'installation

Please select the drives you'd like to install the operating system on, as well as any drives you'd like to automatically mount to your system, below:

<input type="checkbox"/>	Model	Capacity	Vendor	Interconnect	Serial Number	
<input type="checkbox"/>	ATA HARDDISK	8192 MB		ATA	7dda9c2d-de12e9b2	
<input type="checkbox"/>	ATA HARDDISK	20480 MB		ATA	e7204743-23e78ca8	

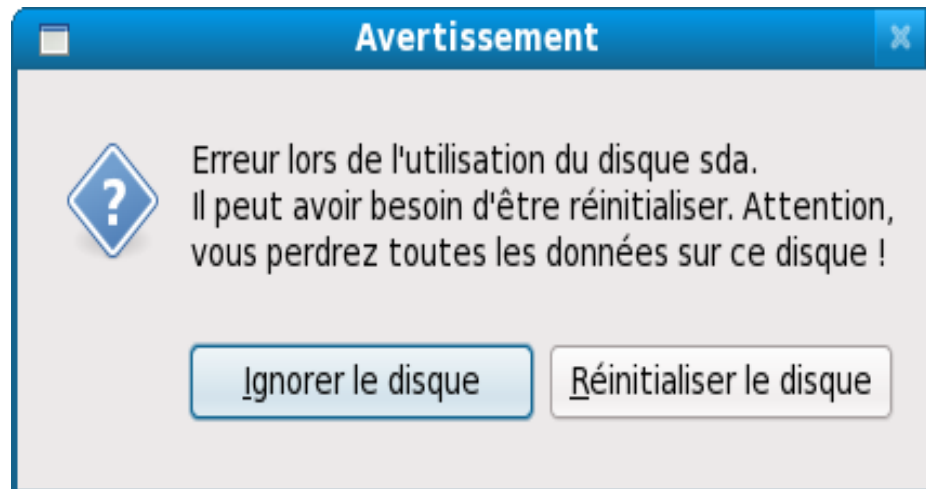
**0 device(s) (0 MB) selected** out of 2 device(s) (28672 MB) total.

 **Tip:** Selecting a drive on this screen does not necessarily mean it will be wiped by the installation process. Also, note that post-installation you may mount drives you did not select here by modifying your `/etc/fstab` file.

# Initialisation du disque






Si aucune table de partition lisible n'est trouvée sur les disques durs, le programme d'installation vous demande l'autorisation pour initialiser le disque dur. Cette opération rend illisible toutes les données présentes antérieurement sur le disque.



# Partitionnement

À partir de cet écran, vous pouvez choisir de créer le schéma de partitionnement par défaut de quatre manières différentes, ou de choisir le partitionnement manuel pour créer un schéma personnalisé.

Which type of installation would you like?

- ☐  **Use All Space**  
Removes all partitions on the selected device(s). This includes partitions created by other operating systems.  
**Tip:** This option will remove data from the selected device(s). Make sure you have backups.
- ☒  **Replace Existing Linux System(s)**  
Removes only Linux partitions (created from a previous Linux installation). This does not remove other partitions you may have on your storage device(s) (such as VFAT or FAT32).  
**Tip:** This option will remove data from the selected device(s). Make sure you have backups.
- ☐  **Shrink Current System**  
Shrinks existing partitions to create free space for the default layout.
- ☐  **Use Free Space**  
Retains your current data and partitions and uses only the unpartitioned space on the selected device(s), assuming you have enough free space available.
- ☐  **Create Custom Layout**  
Manually create your own custom layout on the selected device(s) using our partitioning tool.

☐ Encrypt system  
☐ Review and modify partitioning layout

[< Back](#) [Next >](#)

# Partitionnement

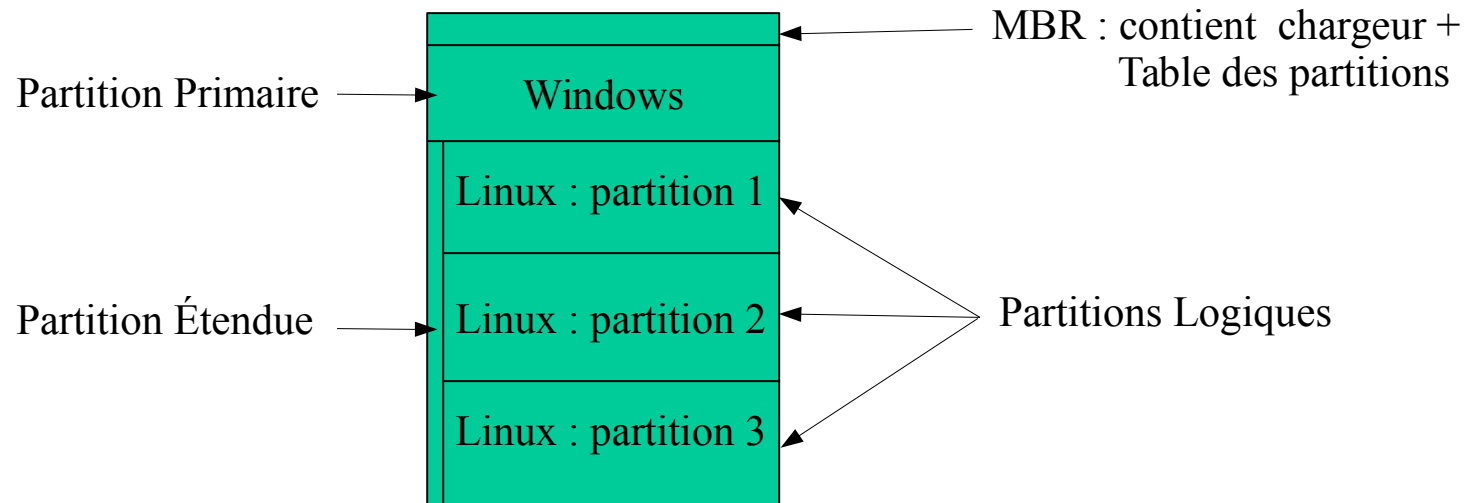
Les quatre premières options vous permettent d'effectuer une installation automatique sans avoir à partitionner manuellement vos disques.

Ces options sont :

- Utiliser tout l'espace : Si vous sélectionnez cette option, toutes les données sur les disques sélectionnés seront supprimées.
- Remplacer le(s) système(s) Linux existant(s) : Sélectionner cette option pour ne supprimer que les partitions créées par une installation précédente de Linux (toutes distributions confondues).
- Réduire une partition existante : cette option permet de récupérer un espace libre en réduisant une partition existante.
- Utiliser l'espace libre : pour installer Fedora sur l'espace inutilisé.
- Créer un partitionnement personnalisé : cette option vous permet de partitionner les périphériques de stockage manuellement.

# Partitionnement personnalisé

- Dans ce cas il faut d'abord commencer par récupérer un espace libre en supprimant/réduisant les partitions existantes puis ensuite créer les partitions que vous souhaitez.
- En générale :
  - On peut créer au maximum 4 partitions primaires.
  - L'une d'elle peut être une partition étendue
  - Une partition étendue peut abriter jusqu'à 64 partitions logiques



# Partitions requises (1)

Pour l'installation de Gnu/Linux, il est vivement conseillé de créer au minimum les trois partitions suivantes :

- /boot** d'une taille de 200Mo à 500Mo, elle abritera tout ce qui est nécessaire au démarrage de la machine: le(s) chargeur(s) et le(s) noyau(x) Linux. Pour les machine relativement anciennes elle doit résider en deçà du cylindre 1023 du disque.
- /** à partir de 500Mo : elle contient le système de fichiers racine de Linux « / ». Dans cette configuration, tous les fichiers (sauf ceux stockés dans /boot/) se trouvent sur cette partition

# Partitions requises (2)

**swap** (d'au moins 256 Mo) — les partitions swap sont utilisées pour prendre en charge la mémoire virtuelle. En d'autres termes, les données transférées sur une partition swap lorsqu'il n'y a pas assez de RAM pour continuer à exécuter les applications.

L'espace swap devrait être égal à deux fois la quantité de RAM physique jusqu'à 2 Go de RAM physique et une fois et demi à une fois la quantité de RAM physique pour toute quantité au-dessus de 2 Go.

La quantité d'espace swap varie selon les critères suivants :

- Les applications devront tourner sur le système.
- La quantité de RAM physique installée sur la machine.
- La version du noyau.



# Autres Partitions (1)

Les principales autres partitions qu'on peut ajouter pour une installation professionnelle sont :

- /usr** contient la partie applicatif du système. On y trouve la plupart des commandes et services pour administrateur et utilisateurs.
- /home** sa taille dépend du nombre d'utilisateurs et de leur consommation d'espace disque.
- /var** à partir de 100Mo : Elle contient les fichiers et répertoires variables. En particulier les files d'attente (spool) et fichiers de journalisation (logs).

# Autres Partitions (2)

- /tmp** sa taille dépend du nombre de services installés. On peut l'estimer à environ 50Mo. Elle contient des fichiers temporaires créés par ces services. La création spécifique d'une partition pour /tmp évite la saturation de la partition racine.
- /usr/src** sa taille est d'au moins 600 Mo. Elle contient les sources du noyau et des paquetages. Ce répertoire est indispensable pour régénérer un noyau personnalisé.

# Partitionnement

- Les deux partitions « / » et « swap » sont indispensables pour toute installation de Gnu/Linux
- En fonction d'usage et d'espace disque disponible, Il est parfois souhaitable de créer les autres partitions : /boot, /home, ...
- Avantage :
  - Structuration
  - Fiabilité
  - Partage
  - Sécurité
- L'utilitaire de partitionnement utilisé est: Disk Druid
- Pour chaque partition à créer on doit spécifier :
  - Le type (swap, ext\*, ....)
  - Le répertoire de montage (pour les partitions non *swap*)
  - La taille

# Installation du chargeur

Dans cet écran Vous devez choisir l'emplacement d'installation de GRUB.

fedora

☒ Le programme d'installation GRUB sera installé sur `/dev/sda`.

☐ Aucun chargeur de démarrage ne sera installé.

Vous pouvez configurer le chargeur de démarrage pour démarrer d'autres systèmes d'exploitation. Il vous permettra de sélectionner un système d'exploitation à démarrer parmi ceux de la liste. Pour ajouter des systèmes d'exploitation qui ne sont pas détectés automatiquement, cliquez sur « Ajouter ». Pour modifier le système d'exploitation démarré par défaut, sélectionnez « Par défaut » à côté du système d'exploitation désiré.

Par défaut	Étiquette	Périphérique
<input checked="" type="checkbox"/>	Fedora	/dev/sda2

Ajouter

Éditer

Supprimer

Le mot de passe du chargeur de démarrage empêche que les utilisateurs envoient des options arbitraires au noyau. L'utilisation du mot de passe est recommandée pour une sécurité optimale.

☐ Utiliser un mot de passe pour le chargeur de démarrage [Modifier le mot de passe](#)

☐ Configuration des options avancées du chargeur de démarrage

Notes de mise à jour

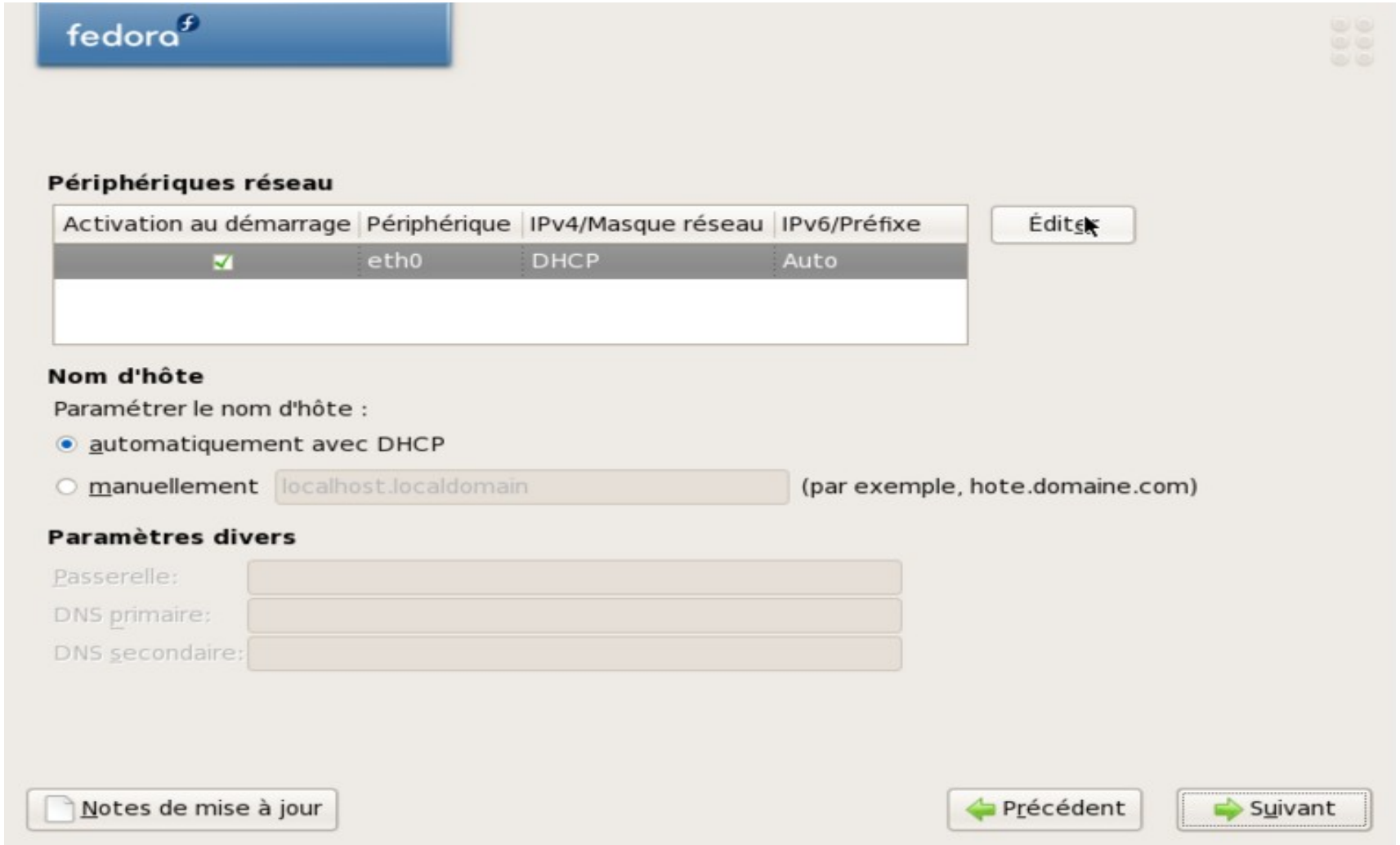
← Précédent

Suivant →

# GRUB

- GRUB (Grant Unified Boot loader) est ce qu'on appelle un chargeur de démarrage, en d'autres termes un programme qui vous permet de choisir et lancer un système d'exploitation.
- Si vous n'avez qu'un seul disque, a priori, les options par défauts conviennent.
- Si vous avez plusieurs disques, faites attention au disque sur lequel GRUB va s'installer.
- Si vous avez d'autres systèmes d'exploitation déjà installés, Anaconda essaye de les détecter automatiquement et configure GRUB pour les démarrer. Vous pouvez configurer manuellement tout système d'exploitation non détecté.
- Le bouton « Par défaut » à côté des systèmes permet de choisir celui qui s'amorce par défaut.

# Configuration réseau



The screenshot shows the 'fedora' logo in the top left corner. The main window is titled 'Périphériques réseau' (Network Devices). It contains a table with four columns: 'Activation au démarrage' (Start-up activation), 'Périphérique' (Device), 'IPv4/Masque réseau' (IPv4/Network mask), and 'IPv6/Préfixe' (IPv6/Prefix). The first row shows a checked box for activation, the device 'eth0', 'DHCP' for IPv4, and 'Auto' for IPv6. An 'Éditer' (Edit) button is to the right of the table. Below the table, the 'Nom d'hôte' (Hostname) section is visible, with a label 'Paramétrer le nom d'hôte :'. It has two radio buttons: 'automatiquement avec DHCP' (selected) and 'manuellement' (manual). The manual option has a text input field containing 'localhost.localdomain' and a hint '(par exemple, hote.domaine.com)'. Below this is the 'Paramètres divers' (Miscellaneous parameters) section, which includes three text input fields for 'Passerelle:' (Gateway), 'DNS primaire:' (Primary DNS), and 'DNS secondaire:' (Secondary DNS). At the bottom left is a button 'Notes de mise à jour' (Update notes), and at the bottom right are two buttons: 'Précédent' (Previous) with a left arrow and 'Suivant' (Next) with a right arrow.

**Périphériques réseau**

Activation au démarrage	Périphérique	IPv4/Masque réseau	IPv6/Préfixe
<input checked="" type="checkbox"/>	eth0	DHCP	Auto

**Nom d'hôte**  
Paramétrer le nom d'hôte :

☒ automatiquement avec DHCP


☐ manuellement  (par exemple, hote.domaine.com)



**Paramètres divers**

Passerelle:

DNS primaire:

DNS secondaire:

 **Notes de mise à jour**

 **Précédent**  **Suivant**

# Configuration réseau

- Demandé lorsque le processus d'installation détecte la présence d'une interface réseau
- On peut choisir entre :
  - Une configuration automatique via DHCP (cela suppose avoir un serveur DHCP)
  - Entrer manuellement les paramètres réseaux :
    - Adresse IP et masque
    - Hostname
    - Adresse de la passerelle par défaut
    - Les adresses des serveurs DNS utilisés


# Sélection des paquets


The default installation of Fedora includes a set of software applicable for general internet usage. You can optionally select a different set of software now.

☒ Graphical Desktop  
☐ Software Development  
☐ Web Server  
☐ Minimal

Please select any additional repositories that you want to use for software installation.


☒ Installation Repo  
☐ Fedora 13-Beta - i386  
☐ Fedora 13-Beta - i386 - Test Updates

 [Add additional software repositories](#)

 [Modify repository](#)

You can further customize the software selection now, or after install via the software management application.

☒ [Customize later](#)    ☐ [Customize now](#)

 [Back](#)

 [Next](#)

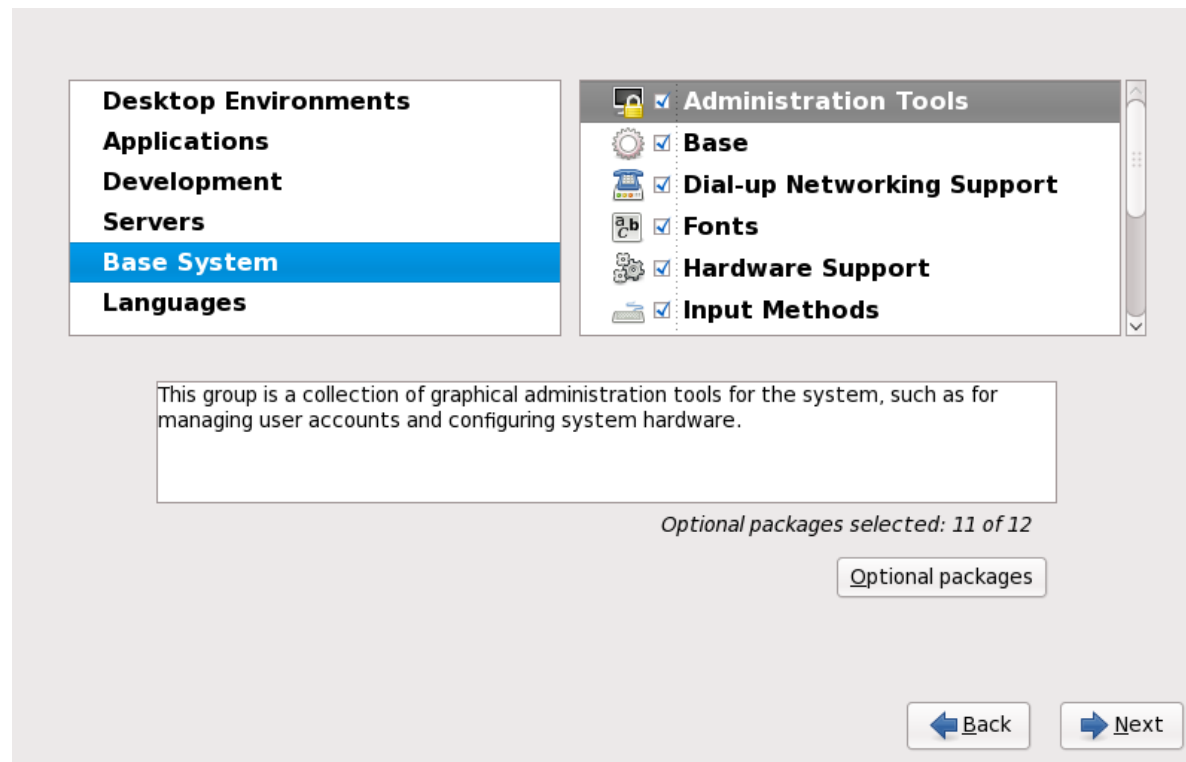


# Sélection des paquets

- Lors de cette étape, Anaconda vous propose trois catégories de programmes que vous pouvez sélectionner en fonction de l'usage que vous voulez faire de votre ordinateur.
- Si vous possédez une connexion Internet fiable et rapide, vous pouvez envisager d'installer à partir des dépôts officiels de la distribution
- Vous pouvez aussi décider d'augmenter le nombre de programmes disponibles en ajoutant des dépôts externes.
- Si vous en avez un usage plus particulier ou que vous désirez installer des programmes spécifiques vous avez la possibilité de les choisir en personnalisant l'installation.

# Sélection des paquets

L'écran suivant apparaît si vous choisissez de personnaliser la liste des paquets à installer



# Sélection des paquets

- Tous les paquets disponibles sont rassemblés par groupe (X Window, Éditeurs, serveur FTP, ...)
- Les groupes de paquets sont rassemblés par catégorie (Environnements de Bureau, Applications, Développement, ...)
- La sélection des paquets peut s'effectuer par groupe ou individuellement.
- Le bouton *détail* devant chaque groupe peut aider à raffiner la sélection dans le groupe en question
- La fenêtre de contrôle de dépendance apparaît lorsqu'un paquet est sélectionné. Elle vous permet de choisir entre les actions suivantes:
  - Sélectionner les paquets manquants
  - Ne pas installer les paquets sources du problème
  - Ignorer le système de dépendance

# Post Installation

L'écran ci-dessous apparaît après le premier redémarrage du système



# Post Installation

- Vous demande de :
  - Lire et valider la licence
  - Déclarer les utilisateurs : en effet, seul l'administrateur (root) est créé lors de l'installation
  - Configurer la date et l'heure
  - Envoyer votre profile matériel aux responsables de la distribution.

# Hiérarchie des répertoires L I N U X

# F H S

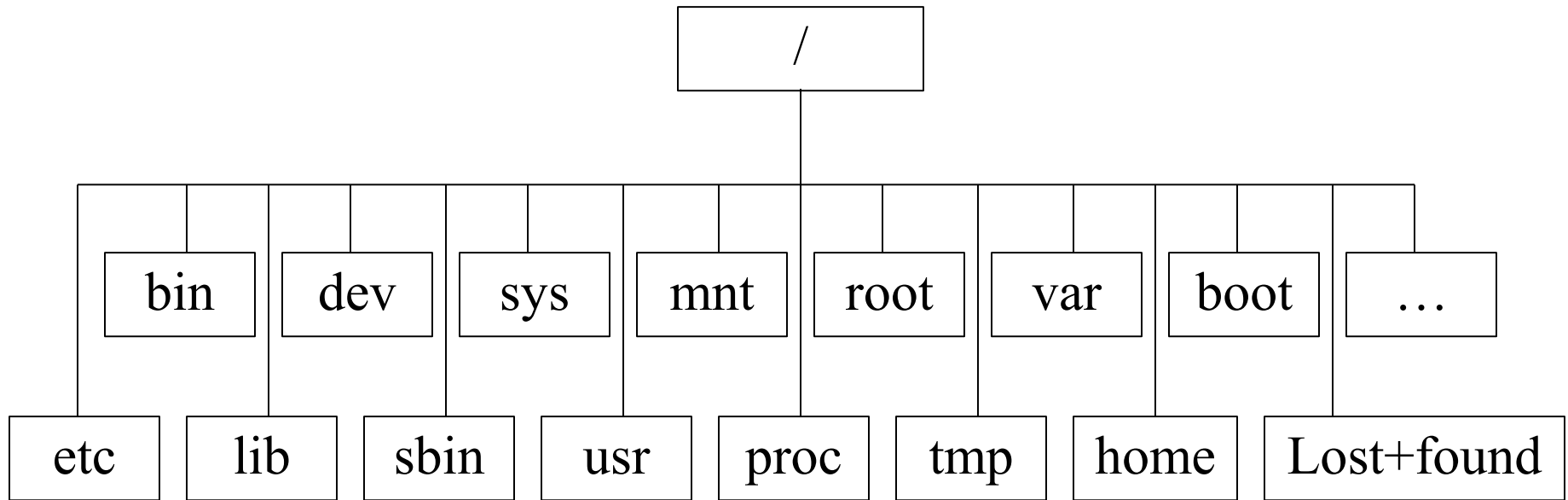
- Filesystem Hierarchy Standard (« norme de la hiérarchie des systèmes de fichiers », abrégé en FHS) définit l'arborescence et le contenu des principaux répertoires des systèmes de fichiers des systèmes d'exploitation GNU/Linux et de la plupart des systèmes Unix.
- La version actuelle est la 2.3, publiée en janvier 2004.

# F H S

- La standardisation de la hiérarchie de système de fichiers a commencé en août 1993 par le FSSTND (Filesystem Standard) dont la première version remonte au 14 février 1994.
- En 1996, la définition de ce standard a été généralisée aux différents Unix. Ce standard est alors renommé en Filesystem Hierarchy Standard.
- Le FHS est maintenu par le Free Standards Group qui compte parmi ses membres Hewlett-Packard, Red Hat, IBM, Dell, etc.
- La vaste majorité des distributions GNU/Linux ne respectent pas strictement le standard (répertoire /srv par exemple)



# Hiérarchie des répertoires (FHS)



# **/bin , /sbin , /lib**

- /bin, /sbin et /lib contient des outils indispensables qui doivent être disponibles dans les pires conditions
- Il doivent toujours logger dans le système de fichier racine.

**/bin**     exécutable pour tous les utilisateurs (ls, cp, mv, vi, bash, ...)

**/sbin**    exécutable pour administration (shutdown, ifconfig, arp, dump, fsck, ...)

**/lib**     contient les bibliothèques partagées (shared libraries) utilisés par la quasi-totalité des exécutables système

# /dev

- **/dev** contient des fichiers spéciaux (device files) correspondant aux périphériques
- La plupart des périphériques sont représentés par des fichiers spéciaux se trouvant dans le répertoire **/dev**
- les fichiers spéciaux ne prennent quasiment pas de place sur le disque, et sont utilisés pour dialoguer avec le système.

⇒ tout accès à un fichier spécial se traduit par un accès physique au périphérique correspondant :

`ls -l > ~/lsfile` le résultat de la commande est écrit dans un fichier ordinaire.

`ls -l > /dev/lp0` le résultat de la commande est redirigé vers le port parallèle.

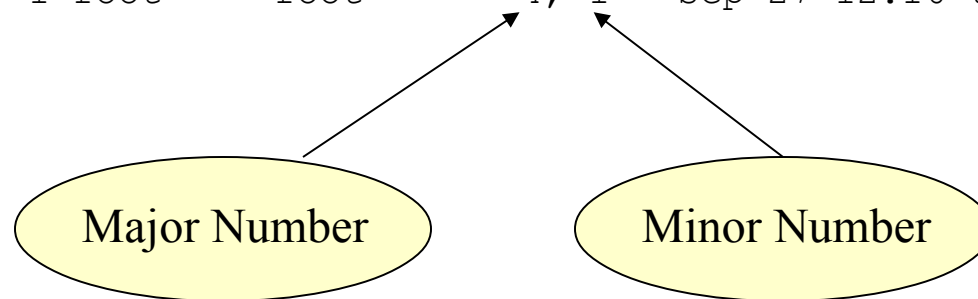
# /dev

- L'accès aux *device files* est généralement réservé à l'administrateur
- Deux type de fichiers spéciaux :
  - Block device files
  - Character device files :
- Les fichiers en mode bloc sont des périphériques comme des disques (où les données sont accessibles à travers un numéro de bloc, et où il est intéressant d'avoir une mémoire cache). Tous les autres périphériques sont en mode caractère.
- avec la commande : `ls -l /dev`  
les lettre **b** et **c** débutent respectivement les lignes correspondant aux *Block device files* et *Character device files*.

# /dev

```
# ls -l /dev
```

```
brw-rw---- 1 root floppy 2, 0 Aug 30 2002 fd0
brw-rw---- 1 root disk 3, 0 Aug 30 2002 hda
brw-rw---- 1 root disk 3, 1 Aug 30 2002 hda1
brw-rw---- 1 root disk 22, 0 Aug 30 2002 hdc
crw-rw---- 1 root lp 6, 0 Aug 30 2002 lp0
brw----- 1 root disk 8, 0 Aug 30 2002 sda
crw--w---- 1 root root 4, 0 Aug 30 2002 tty0
crw----- 1 root root 4, 1 Sep 27 12:16 tty1
```



# /dev

- Le numéro majeur d'un fichier spécial sert à identifier le pilote correspondant
- Le numéro mineur identifie un des périphériques parmi ceux gérés par le même pilote ou une autre manière de le considérer (densité, rembobinage, ...)
- Traditionnellement le contenu de */dev* est créé à l'installation du système grâce au script MAKEDEV.
- MAKEDEV utilisait la commande Unix **mknod** pour créer tous les fichiers spéciaux correspondant à tous les périphériques possibles

Syntaxe: `mknod [-m mode] nom {bc} majeur mineur`

# devfs

- La création statique à l'installation de l'ensemble des fichiers spéciaux possibles a comme :
  - Avantage : la simplicité
  - Inconvénients : exhaustivité difficile, consomme beaucoup de ressources (environ 18 000 fichiers sont créés dans Fedora Core 1)
- À partir du noyau 2.3.46 le répertoire */dev* est remplacé par le système de fichier **devfs** dont le but est de créer dynamiquement seuls les fichiers spéciaux correspondant aux périphériques détectés.
  - Avantage : consomme moins de ressources
  - Inconvénients : pas de possibilité de configurer le système de nommage des fichiers spéciaux créés

# *udev*

- Avec le développement du noyau 2.5, un nouveau système de fichiers virtuel appelé *sysfs* est arrivé.
- Le travail de *sysfs* est d'exporter une vue de la configuration matérielle du système vers les processus en espace utilisateur « *user space* ».
- *devfs* est alors remplacé par un autre système : *udev* tournant dans l'espace utilisateur et permettant :
  - La création / destruction à la volé des fichiers /dev correspondant aux périphériques insérés / retirés.
  - La personnalisation des noms attribués aux nouveaux périphériques détectés.



# Quelques fichiers /dev

- Quelques fichiers spéciaux intéressants :
  - /dev/hda le premier disque non SCSI
  - /dev/hda1, /dev/hda5 respectivement la première et la cinquième partition du premier disque du premier contrôleur non SCSI
  - /dev/fd0 disquette
  - /dev/cdrom généralement un lien vers le CD\_ROM
  - /dev/lp0 la première sortie parallèle (lpt1)
  - /dev/ftape lecteur de bandes non SCSI
  - /dev/sda le premier disque SCSI
  - /dev/sda1 la 1<sup>ière</sup> partition du 1<sup>ier</sup> disque SCSI
  - /dev/st0 le 1<sup>ier</sup> lecteur de bandes SCSI
  - /dev/tty1 le 1<sup>ier</sup> terminal virtuel. Accessible via la touche [ctl] [alt] [F1]
  - /dev/pts/? sont créés dynamiquement pour représenter les fenêtres X et les connexions à distance
  - /dev/ttyS0 la première sortie série (COM1)

# Quelques fichiers /dev

- Quelques fichiers spéciaux correspondant à des pseudo-périphériques :
  - /dev/null      la poubelle, tout ce qui écrit dans /dev/null est complètement ignoré
  - /dev/zero      un générateur de zéros
  - /dev/tty      le terminal de contrôle du programme en cours d'exécution
  - /dev/mem      la mémoire physique. Utilisé généralement par les outils de débogage
  - /dev/random    générateur de nombres aléatoires

# /etc

- **/etc** contient les fichiers et scripts de configuration des différents services du système.
- Doit se situer obligatoirement dans la partition racine
- Contient les répertoires suivants :
  - **/etc/X11** les fichiers de configuration de Xwindow
  - **/etc/rc.d** les scripts de démarrage du système
  - **/etc/cron** les tâches à effectuer à la périodicité donnée (daily, hourly, monthly, weekly)
  - **/etc/skel** les fichiers à recopier dans le répertoire d'un nouvel utilisateur
  - **/etc/sysconfig** les fichiers de configuration des périphériques

# /usr

- **Unix System Resources**
- Contient les programmes, utilitaires et librairies non indispensables au fonctionnement du système
- Généralement mis dans une partition séparée.
  - Peut être accessible en lecture seule (read only)
  - Peut être monté en NFS

# /usr

**contient :**

<b>bin, sbin et lib</b>	les équivalents de /bin, /sbin et /lib
<b>etc</b>	les fichiers de configuration des applications. Ce répertoire est très rarement utilisé, en effet, la plupart des applications installe leurs fichiers de configuration directement sous <i>/etc</i>
<b>include</b>	les fichiers (.h) pour le compilateur C.
<b>local</b>	arborescence des fichiers propres à la machine.
<b>share</b>	contient les fichiers indépendants de l'architecture : manuel, docs, images, etc...
<b>src</b>	est un emplacement contenant les sources
<b>game</b>	contient les données relatives aux jeux installés

# /var

Le répertoire **/var** contient :

- les fichiers dont la taille peut croître considérablement (log files)
- les fichiers de verrouillage des ressources (lock files)
- les répertoires dont le contenu varie considérablement
  - Les boîtes aux lettres
  - Les spools d'impression
  - ...
- Les fichiers temporaires sauvegardés plus longtemps.
- /var doit avoir de préférence sa propre partition.

# /var

## Quelques sous-répertoires

<b>/var/lock</b>	contient les "fichiers de verrouillage". Généralement des fichiers vides, leur simple présence permet de verrouiller l'accès aux ressources correspondantes
<b>/var/catman</b>	les fichiers d'aide mis en forme
<b>/var/log</b>	est utilisé pour stocker les divers journaux du système.
<b>/var/spool</b>	contient les files d'attentes (cron, lpd, mail,...)
<b>/var/run</b>	les fichiers contenant les "pid" des processus des différents services système

# /proc

- **/proc** est un pseudo-système de fichiers utilisé comme interface avec les structures de données du noyau.
- L'objectif principal de /proc est d'exporter quelques informations gérées par le noyau vers le « *user space* »
- Beaucoup de commandes GNU/Linux (ps, top, vmstat, etc.) tirent leurs informations de ce système de fichiers.
- Les fichiers et répertoires de /proc sont virtuels parce que les données ne sont pas réellement enregistrées sur le disque ; ils sont créés dynamiquement en mémoire.



# /proc : Informations sur les processus (1)

Chaque processus qui tourne dans le système est représenté par un répertoire sous /proc dont le nom n'est rien autre que le pid correspondant et qui contient les fichiers et répertoires suivants:

<b>cmdline</b>	La ligne de commande du processus. Les arguments sont séparés par le caractère <i>null</i>
<b>cwd</b>	Un lien sur le répertoire de travail courant
<b>environ</b>	Contient l'environnement du processus. Liste (variable, valeur)
<b>exe</b>	Un pointeur sur le fichier binaire exécuté,
<b>fd</b>	Un sous-répertoire contenant un lien pour chaque fichier ouvert.
<b>maps</b>	Un fichier contenant les régions mémoire actuellement Projetées et leurs autorisations d'accès.
<b>mem</b>	L'espace mémoire du processus
<b>root</b>	Racine du système de fichier du processus, configurable ( <i>chroot</i> )
<b>stat</b>	Informations sur l'état du processus.

# /proc : Informations sur les processus (2)

En plus des répertoires représentant les processus, /proc contient :

<b>cpuinfo</b>	informations dépendantes de l'architecture et du processeur.
<b>devices</b>	Liste littérale des groupes de périphériques et des numéros majeurs.
<b>dma</b>	Il s'agit d'une liste des canaux DMA en cours d'utilisation.
<b>filesystems</b>	Liste des systèmes de fichiers utilisés par le noyau.
<b>interrupts</b>	Il s'agit du nombre d'interruptions reçues pour chaque IRQ.
<b>ioports</b>	Liste des régions d'entrée-sortie en cours d'utilisation.
<b>kcore</b>	l'espace mémoire du kernel.
<b>kmsg</b>	peut être utilisé à la place de l'appel <i>syslog</i>
<b>ksyms</b>	Symboles exportés par le noyau et utilisés pour assurer l'édition dynamique des liens des modules chargeables.
<b>loadavg</b>	Ce fichier indique les charges système : <i>uptime</i>

# /proc : Informations sur le système

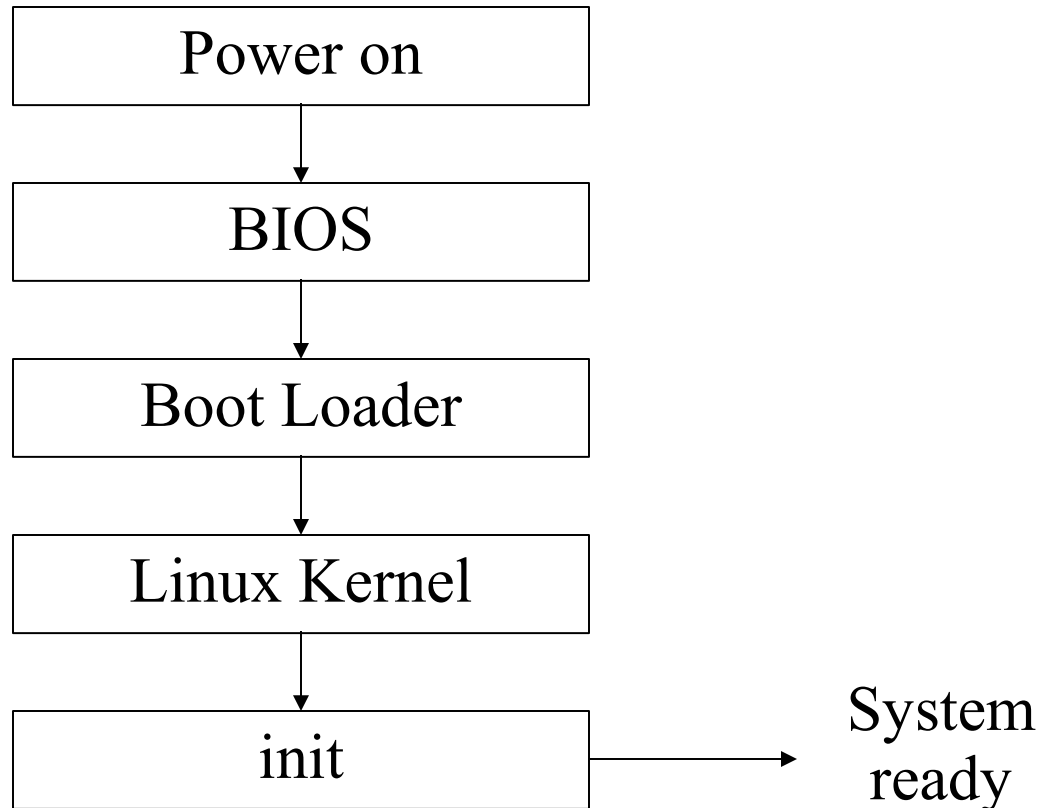
<b>locks</b>	Ce fichier montre les verrouillages actuels des fichiers.
<b>meminfo</b>	Indique les quantités de mémoires (physique et swap) libres et utilisées
<b>modules</b>	Une liste littérale des modules qui ont été chargés.
<b>net</b>	Regroupe divers pseudo-fichiers relatifs aux fonctionnalités réseau regroupées par couche (arp, rarp, raw, tcp, ...)
<b>pci</b>	Liste de tous les périphériques PCI détectés pendant l'initialisation ainsi que leurs configurations.
<b>self</b>	Lien vers le répertoire /proc du processus accédant au /proc.
<b>sys</b>	Ce répertoire contient un ensemble de fichiers et de sous-répertoires correspondant à des variables internes du noyau.
<b>uptime</b>	Contient deux valeurs : la durée de fonctionnement (en sec), et le temps écoulé à ne rien faire (idle), en secondes
<b>version</b>	Cette chaîne identifie la version du noyau en cours d'exécution.

# Autres répertoires

<b>/boot</b>	Contient les fichiers utiles pour le chargeur (les chargeurs eux mêmes + noyaux Linux)
<b>/home</b>	Les espaces privés des utilisateurs.
<b>/mnt</b>	Contient des répertoires utilisés comme points de montage des partitions externes au système
<b>/media</b>	Contient les points de montage des unités amovibles : disquette, CD_ROM, mémoire flash...
<b>/tmp</b>	Contient les fichiers temporaires.
<b>/root</b>	L'espace de travail privé de l'administrateur (root)
<b>/lost+found</b>	Utilisé par <i>fsck</i> pour y mettre les fichiers perdus et récupérés.

# Démarrage et Arrêt du système

# Processus de démarrage



# Démarrage : Bios

- analyse la configuration matérielle de l'ordinateur:  
Recensement des périphériques + Test de certains périphériques :
  - Test de la mémoire,
  - Test de la présence de clavier, ...
  - Test de la présence des disques durs, lecteurs de CDROM IDE
  - Assignment d'une IRQ pour le contrôleur SCSI
  - ...
- Localisation de l'OS

# Démarrage : Bios

- Lancement de l'OS
  - lit les 512 premiers octets du périphérique d'amorçage : soit le premier secteur (MBR = *Master Boot Record*) sur un disque dur ; soit le secteur d'amorçage sur disquette ou CD.
  - Ce secteur contient
    - un boot Loader (LILO, GRUB,...) et
    - la table des partitions



# Démarrage : LILO

- LILO est scindé en 2 parties : la première faisant moins de 512 octets étant chargée de lancer la deuxième partie qui chargera le noyau.
- Une fois totalement chargé, la première chose que fait LILO est de regarder si une des touches [Shift], [Ctrl] ou [Alt] est enfoncée. Si tel est le cas (ou bien si l'option « prompt » est spécifiée dans `/etc/lilo.conf`) celui-ci affichera le menu et attend une action de la part de l'utilisateur.

# Démarrage : LILO

- L'utilisateur peut activer le mode texte en appuyant sur la touche « Echap » ou « ctrl x »
- La liste des différents noyaux LINUX ou autres systèmes d'exploitation amorçables par LILO est obtenue en tapant sur [Tab].
- A la suite du nom du noyau, l'utilisateur peut également entrer des paramètres supplémentaires pour modifier le comportement du système.

# Démarrage : LILO

Voici quelques paramètres utilisés :

- **boot: Nom-Noyau root=/dev/hdaX** : changer la partition racine
- **boot: Nom-Noyau mem=128M** : forcer l'utilisation d'une certaine quantité de mémoire.
- **boot: Nom-Noyau 3** : Pour spécifier directement le niveau d'exécution à transmettre au programme **init**
- **boot: Nom-Noyau single** : Pour démarrer en single-user

# Démarrage : LILO

- LILO lit dans **/boot/map** pour avoir les adresses des blocs disque qui contiennent le noyau à charger.  
Pendant ce temps s'affiche :  
**"Loading Linux..."**
- Il y a décompression de l'image auto-extractible : **/boot/vmlinuz-XXX**. On voit le message  
**"Uncompressing Linux"**
- Le noyau LINUX démarre :  
**"OK, booting the kernel"**

# LILO : Installation

Si LILO n'as pas été mis en place au moment de l'installation du système :

- Vérifiez si le paquetage LILO (la dernière version) est installé
- Vérifiez l'existence et le contenu du fichier `/etc/lilo.conf`
- Mettez en place le chargeur en exécutant la commande:  
`/sbin/lilo -vv`  
ceci installe chargeur LILO au MBR.
- d'autres options de la commande `/sbin/lilo` permettent de :
  - Désinstaller LILO
  - Sauvegarder l'ancien chargeur dans un fichier
  - ...
- La commande `/sbin/lilo` lit le fichier `/etc/lilo.conf` pour fixer les options d'installation

# **/etc/lilo.conf : Exemple**

```
boot=/dev/hda
map=/boot/map
install=/boot/boot.b
prompt
timeout=50
message=/boot/message
default=linux
lba32

image=/boot/vmlinuz-2.6.5-1.358
    label=linux
    initrd=/boot/initrd-2.6.5-1.358.img
    read-only
    root=/dev/hda5

other=/dev/hda1
    label=Windows
```

# /etc/lilo.conf

<b>boot=/dev/hda</b>	Indique à LILO de s'installer dans le MBR du premier disque IDE rencontré.
<b>map=/boot/map</b>	Localise le fichier map. A ne pas modifier dans le cas normal d'utilisation.
<b>install=/boot/boot.b</b>	Indique à LILO d'installer le fichier spécifier dans le secteur de démarrage.
<b>prompt</b>	Sans cette option LILO ne permet pas aux utilisateurs de choisir le système à amorcer. A moins d'appuyer sur la touche [Shift] avant que LILO ne lance le système par défaut.
<b>timeout=50</b>	Spécifie la durée d'attente (en 10 <sup>ième</sup> de sec) avant le chargement du système par défaut.
<b>message=/boot/message</b>	— Spécifie l'écran à afficher indiquant aux utilisateurs de sélectionner le système à amorcer.

# /etc/lilo.conf

<b>lba32</b>	Indique à LILO la géométrie du disque. (même que : <b>linear</b> ).
<b>default=linux</b>	Spécifie le système à démarrer par défaut
<b>image=/boot/vmlinuz-2.4.20</b>	— l'emplacement du noyau Linux à démarrer en utilisant les options spécifiées dans les lignes suivantes.
<b>label=linux</b>	Nom choisi pour le système/noyau que LILO affiche dans le menu
<b>initrd=/boot/initrd-2.4.20.img</b>	— Spécifie l' <i>initial ram disk</i> à utiliser au démarrage du noyau en question. initrd est une collection de modules nécessaires au chargement du système (pilotes SCSI, ...).
<b>read-only</b>	La partition racine doit être montée en lecture seule, pour éviter son altération au cours du démarrage.
<b>root=/dev/hda5</b>	Spécifie la partition racine.
<b>other=/dev/hda1</b>	Spécifie la partition contenant un autre système.



# GRUB

- ***GRand Unified Bootloader*** est actuellement le chargeur par défaut de la plupart des distributions Linux
- Supporte le chargement de plusieurs noyaux 32 ou 64 bits : Linux, Hurd, FreeBSD, NetBSD, OpenBSD ...
- En plus de l'interface menu habituelle Grub possède une interface en ligne de commande offrant un jeu de commande pre-OS relativement riche et flexible
- Supporte plusieurs systèmes de fichiers de façon transparente comme les ext2fs, BSD FFS, DOS FAT16 et 32, Minix fs, ReiserFS, JFS, XFS ...

# GRUB

- Le fichier de configuration */boot/grub/grub.conf* ou *menu.lst* est lu à chaque démarrage de la machine. Plus besoin de réinstaller après chaque modification de la configuration
- Supporte le mode LBA (*Logical Block Addressing*). Permet d'accéder aux blocs disque se trouvant au delà du cylindre 1023
- Est indépendant des traductions de géométrie du disque
- Supporte le démarrage depuis un réseau : il est tout à fait possible de charger des images d'OS sur un réseau en utilisant le protocole TFTP.
- Supporte le contrôle de démarrage à partir des terminaux distants pour les ordinateurs sans console.

# GRUB : Structure

Comme LILO, GRUB est scindé en 2 parties :

- la première réside dans le MBR (fait moins de 512 octets). Elle est chargée de lancer la partie 1.5 ou directement la partie n°2 qui chargera le noyau.
- La partie 1.5: sollicitée si l'accès à la deuxième partie (se trouvant dans un système de fichiers) nécessite un intermédiaire.
- La deuxième partie de GRUB lit le fichier de configuration : */boot/grub/grub.conf* et l'interprète pour afficher le menu contenant les systèmes opératoires (et les différents noyaux Linux) amorçables.

# Démarrage : GRUB

Si le fichier de configuration est absent ou corrompu; GRUB affiche un message d'erreur et passe en mode texte pour permettre à l'utilisateur de saisir manuellement les commandes nécessaires pour achever le processus de démarrage :

```
GNU GRUB  version 0.95  (640K lower / 3072K  
upper memory)
```

```
[ Minimal BASH-like line editing is supported.  
For the first word, TAB lists possible command  
completions.  Anywhere else TAB lists the  
possible completions of a device/filename.]
```

```
grub>
```

# Démarrage : GRUB

La commande help affiche la liste d'une cinquantaine de commandes.

```
Grub> help --all
blocklist FILE
chainloader [--force] FILE
clear
color NORMAL [HIGHLIGHT]
displaymem
help [--all] [PATTERN...]
md5crypt
quit
rootnoverify [DEVICE [HDBIAS]] lock
install [--stage2=S2_FILE] [--forc ioprobe DRIVE
kernel [--no-mem-option] [--type=TYPE]
...
boot
cat FILE
cmp FILE1 FILE2
configfile FILE
find FILENAME
initrd FILE [ARG ...]
password [--md5] PASSWD
root [DEVICE [HDBIAS]]
```

# Démarrage : GRUB

- Ces commandes permettent de :
  - Lire, comparer et rechercher des fichiers (ou blocks) sur disque ou support externe: `cat`, `cmp`, `find`, ...
  - Fixer le partition ou le support de démarrage: `root`, `rootnoverify`
  - Choisir le système à démarrer ainsi que ces paramètres de démarrage : `kernel`, `initrd`, `chainloader`, ...
  - ...
- d'autres commandes (`color`, `default`, `timeout`, ...) ne sont possibles que dans le fichier de configuration.

# GRUB : terminologie

- Sont utilisés dans le fichier de configuration et dans les commandes passées à GRUB en mode texte :
  - **Disques** : sont représentés par :  
(*<type-of-device><bios-device-number>,<partition-number>*)  
(hd0,0) dénote la première partition du premier disque.
    - hd pour les disques SCSI, IDE, ...
    - fd pour les disquettes
  - **Fichiers** : sont notés par:  
(*<type-of-device><bios-device-number>,<partition-number>*) /path/to/file
  - **GRUB's Root File System** : le système de fichier racine de GRUB. Généralement est le système de fichiers /boot de Linux.

# GRUB : commandes

L'interface commande est accessible en appuyant sur la touche **c** à partir du menu

- **boot** — démarre le système /noyau précédemment chargé
- **initrd <file-name>** — permet de spécifier le fichier RAM disk à utiliser au démarrage. **initrd** est indispensable quand le kernel nécessite certains modules pour son démarrage. Par exemple lorsque le système de fichiers racine est formaté en **ext3**.
- **kernel <kernel-file-name> <option-1> ... <option-N>** — Spécifie le fichier image à charger à partir du système de fichier de GRUB.  
Ex :        **kernel /vmlinuz root=/dev/hda5**
- **root <device-and-partition>** — Configure la partition contenant le système de fichier racine de GRUB.



# GRUB : commandes

Autres commandes :

- Displaymem — affiche la taille mémoire (information tenue du BIOS).
- install *<stage-1>* *<install-disk>* *<stage-2>* p *<config-file>*  
installe GRUB à partir des fichiers *<stage-1>* et *<stage-2>* en considérant le fichier de configuration *<config-file>*  
Ex: pour installer GRUB dans le MBR.  
install (hd0,0)/grub/stage1 (hd0) (hd0,0)/grub/stage2 p (hd0,0)/grub/grub.conf
- chainloader *<file-name>* — charge le fichier ou bloc spécifié pour lancement en chaîne. Utilisez +1 comme nom de fichier pour charger le premier secteur de la partition désignée.

# GRUB : Fichier de configuration

- **/boot/grub/grub.conf** : structuré en sections, chacune décrivant un système ou noyau linux amorçable par GRUB
- Quelques commandes en plus de ceux citées précédemment :
  - `color <normal-color> <selected-color>` — Spécifie les couleurs de visualisation du menu. Ex :  
`color red/black green/blue`
  - `default <title-name>` — Spécifie le système à amorcer par défaut
  - `fallback <title-name>` — le système à amorcer si la première tentative échoue.

# GRUB : Fichier de configuration

Autres commandes :

- `hiddenmenu` — Pour ne pas afficher le menu. La touche [Esc] peut forcer l’affichage
- `password <password>` — fixe un mot de passe pour une entrée.
- `timeout` — fixe l’intervalle (en deci\_secondes) avant le chargement du système par défaut
- `splashimage` — Spécifie l’image à afficher au moment du démarrage.
- `title` — débute une section. Elle spécifie le nom du système qui sera visualisé dans le menu.

# **/boot/grub/grub.conf : Exemple**

```
verbose=0
default=0
timeout=5
splashimage=(hd0,4)/grub/fedora.xpm.gz
hiddenmenu
```

## **title Fedora (2.6.34.fc13.x86\_64)**

```
    root (hd0,4)
    kernel /vmlinuz-2.6.34.fc13.x86_64 ro
        root=UUID=8734147e-50bd-4223-8fcf-4631ac807aa6
        LANG=fr_FR.UTF-8 rhgb quiet vga=0x318 nomodeset
    initrd /initramfs-2.6.34.7-61.fc13.x86_64.img
```

## **title Windowz Vista**

```
    rootnoverify (hd0,0)
    chainloader +1
```

# GRUB : Installation

- Si GRUB n'as pas été mis en place au moment de l'installation du système :
  - Vérifiez si le paquetage GRUB (la dernière version) est installé
  - Mettez en place le chargeur en exécutant la commande:

```
/sbin/grub-install <location>
```

où *<location>* est l'endroit où l'on désire mettre la première partie du chargeur GRUB. Ex:

```
/sbin/grub-install /dev/sda
```

# GRUB 2

- Le code de Grub est, d'après ses développeurs, difficile à maintenir, trop complexe, souffre d'erreurs de conception limitant son développement.
- Ce code a été complètement réécrit donnant naissance à une nouvelle version : GRUB 2. : L'ancienne version est désormais connu sous le nom de GRUB Legacy.
- Grub Legacy est toujours maintenu, mais aucune nouvelle fonctionnalité ne sera rajoutée.
- GRUB2 a pour but d'être un chargeur d'amorçage plus modulaire et portable que son prédécesseur. Il est devenu le chargeur par défaut de la plupart des distributions

# GRUB2 vs GRUB legacy (1)

- les fichiers de GRUB 2 sont situés à trois endroits:  
**/boot/grub/grub.cfg**, **/etc/grub.d** et **/etc/default/grub**
- le fichier de configuration a été renommé en **grub.cfg** (anciennement grub.conf). Il possède une nouvelle syntaxe ainsi que de nouvelles commandes.
- **grub.cfg** est maintenant un véritable script: variables, conditions et boucles sont maintenant possibles.
- **grub.cfg** est automatiquement généré par **grub2-mkconfig** ce qui simplifie les mises à jour du noyau.
- les numéros des partitions commencent à 1 et non plus à 0

# GRUB2 vs GRUB legacy (2)

- Grub2 offre un petit espace de stockage pour sauvegarder (via la commande *save\_env*) quelques paramètres de démarrage afin de les récupérer (via *load\_env*) dans les prochains redémarrages.
- GRUB 2 possède des moyens plus fiables pour retrouver les périphériques (partitions, CD, ...) contenant les fichiers à charger. Ainsi la recherche de périphériques peut se faire par nom de fichier (chemin complet), par label de système fichiers ou par Universally Unique Identifiers (UUIDs).
- plusieurs systèmes de fichiers sont supportés comme l' ext4, HFS+, NTFS, ...
- GRUB2 est valable pour plusieurs autres systèmes. En plus du BIOS PC supporté par GRUB Legacy, il supporte PC EFI, PC coreboot, PowerPC, SPARC, et MIPS Lemote Yeeloong.



# GRUB2 vs GRUB legacy (3)

- GRUB2 peut accéder directement aux fichiers se trouvant dans les périphériques LVM et RAID.
- Un terminal graphique ainsi qu' un menu système graphique sont maintenant disponibles.
- L'interface de GRUB2 peut être traduit en plusieurs langues.
- Côté architecture, GRUB2 a été complètement réorganisé. Les étapes (stage) 1, 1.5 et 2 n'existent plus, ils sont remplacés par deux fichiers images *boot.img* et *core.img* ainsi qu'un ensemble de modules *\*.mod* chargés en fonction du contexte
- GRUB2 gère plus facilement les modules dynamiques.

# GRUB2 autres caractéristiques

- GRUB2 est conçu pour démarrer en natif les noyaux conformes à la spécification Multiboot
- Comme GRUB Legacy, GRUB2 prend en charge en natif le démarrage de quelques noyaux Non Multiboot, comme : Linux, OpenBSD, FreeBSD et NetBSD. D'autres noyaux (Windows, ...) sont aussi supportés via le *Chain-loading*
- Supporte pleinement les fonctionnalités de démarrage Multiboot en particulier le chargement de modules
- Le Langage de commande est plus flexible et plus complet
- Détecte et utilise le mode LBA
- Supporte le démarrage réseau ainsi que les terminaux distants

# GRUB2 : règles de nommage

La convention de nommage de GRUB2 est quelque peu différente de l'ancienne version :

- Comme GRUB Legacy, GRUB2 numérote les noms des disques et lecteurs de disquettes à partir de 0. Ainsi
  - (fd0) designe *floppy disk n°0* : le **premier** lecteur de disquettes.
  - (hd1) est le *hard disk n°1* c'est à dire le **deuxième** disque.
- Les partitions sont numérotées à partir de 1 : (hd0,2) désigne la **deuxième** partition du premier disque.
- Dans les systèmes contenant à la fois plusieurs tables de partitions, un mot clé (msdos, gpt, ...) doit être renseigné avant le numéro de la partition pour indiquer la table en question. Ainsi
  - (hd0,msdos1) désigne la **première** partition de la table MBR
  - (hd1,gpt5) est la cinquième partition de la table GPT du disque n°2

# GRUB2 : Mise en place

- La plupart des distributions récentes installe automatiquement GRUB2.
- Cette étape n'est alors nécessaire que pour réinstaller GRUB2 ou migrer de Gurb Legacy vers Grub2 dans le cas d'une ancienne distribution.
- La procédure de mise en place comprend :
  - L'installation ou mise à jour des paquetages
  - L'installation du chargeur
  - Configuration

# GRUB2 : Mise en place

- La plupart des distributions récentes installe automatiquement GRUB2.
- Cette étape n'est alors nécessaire que pour réinstaller GRUB2 ou migrer de Gurb Legacy vers Grub2 dans le cas d'une ancienne distribution.
- La procédure de mise en place comprend :
  - L'installation ou mise à jour des paquetages
  - L'installation du chargeur
  - Configuration

# GRUB2 : installation

- L'installation des paquetages peut se faire à l'aide de la commande :

```
yum -y install grub2 os-prober
```

ou

```
yum -y update grub2 os-prober
```

pour la mise à jour

- Le paquetage *os-prober* est utilisé particulièrement par *grub2-mkconfig* pour chercher les systèmes installés et générer les entrées correspondantes dans le fichier *grub.cfg*
- L'installation du chargeur peut se faire à l'aide de la commande :  

```
grub2-install /dev/sda
```

# Génération du fichier de Configuration

- La commande suivante permet de générer le fichier de configuration `grub.cfg` :

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

- La commande **grub2-mkconfig** utilise les scripts suivants :
  - */etc/default/grub* : qui contient les valeurs de quelques paramètres, comme la résolution de l'écran, le Timeout, le système par défaut, ...
  - */etc/grub.d/\** : un ensemble de scripts se servant des variables initialisées dans */etc/default/grub* pour générer le fichier *grub.cfg*
- q

# Génération du fichier de Configuration

- `grub2-mkconfig` est sollicité automatiquement à chaque mise à jour du noyau pour intégrer ce dernier dans le menu de démarrage.
- La personnalisation des paramètres de démarrage ne doit pas alors s'effectuer directement dans *grub.cfg* au risque de la perdre après la mise à jour du noyau
- Les personnalisations doivent être portées dans */etc/default/grub* et dans les scripts se trouvant sous */etc/grub.d*
- Pour mettre le *timeout* à 0 (pour un démarrage directe et sans délai) par exemple on met :

**GRUB\_TIMEOUT=0**

Dans le fichier */etc/default/grub* puis on relance la commande *grub2-mkconfig*



# Restaurer Grub 2

- Sous Fedora 17, La procédure suivante permet de réinstaller GRUB dans le MBR, par exemple lorsqu'il a été effacé suite à l'installation d'un autre système d'exploitation (Microsoft® Windows® en l'occurrence) :
  - Démarrer à partir du DVD d'installation ou CD Live ou n'importe quel autre média d'installation
  - Sélectionner l'option « *Troubleshoot* » du menu de démarrage puis « *Rescue installed system* »
  - Après le démarrage du système, lancer les commandes suivante dans le terminale :
    - `chroot /mnt/sysimage`
    - `grub2-install /dev/sda`
    - `grub2-mkconfig -o /boot/grub2/grub.cfg`
  - Redémarrer

# Configuration

- Pour grub “legacy”, la configuration consistait à retoucher les entrées du fichier *menu.lst* ou *grub.conf* directement.
- Avec grub2 on n'édite pas directement le fichier *grub.cfg*, mais on intervient sur :
  - */etc/default/grub* : qui contient les valeurs de quelques paramètres, comme la résolution de l'écran, le Timeout, le système par défaut, ...
  - */etc/grub.d/\** : un ensemble de scripts permettant de générer le fichier *grub.cfg*. Et dans lesquels on peut rajouter par exemple le code permettant de mettre en place un mot de passe de démarrage.
- Après modification, la commande `grub2-mkconfig` doit être appelée pour régénérer un nouveau *grub.conf*

# /etc/default/grub

- Contient les valeurs associées aux paramètres de démarrage
- Chacune de ses lignes est sous forme : VARIABLE=valeur
- Exemple :

```
GRUB_TIMEOUT=5
```

```
GRUB_DISTRIBUTOR="Fedora"
```

```
GRUB_DEFAULT=saved
```

```
GRUB_CMDLINE_LINUX="rd.md=0 rd.lvm=0 rd.dm=0 quiet LANG=fr_FR.UTF-8
```

```
SYSFONT=latacyrheb-sun16 rhgb rd.luks=0 KEYTABLE=fr-latin9"
```

# /etc/default/grub

Quelque paramètres :

- GRUB\_TIMEOUT : durée en seconde à attendre avant le démarrage du système par défaut
- GRUB\_HIDDEN\_TIMEOUT : Le nombre de secondes à attendre avant l'affichage du menu si aucune touche n'est pressée. Le menu s'affiche ensuite durant le nombre de secondes spécifié dans GRUB\_TIMEOUT avant de démarrer l'entrée par défaut. La plupart des gens qui utilisent GRUB\_HIDDEN\_TIMEOUT mettent GRUB\_TIMEOUT à 0
- GRUB\_HIDDEN\_TIMEOUT\_QUIET : la valeur *true* dans cette variable permet de supprimer l'affichage du compte à rebours.
- GRUB\_DEFAULT : désigne le système à démarrer par défaut, sa valeur peut être l'ordre du système dans le menu (0 est le premier) son nom ou le mot clé *saved* pour désigner le dernier système activé.

# /etc/default/grub

- GRUB\_CMDLINE\_LINUX : les arguments à passer au noyau Linux
- GRUB\_CMDLINE\_LINUX\_DEFAULT : À moins que «GRUB\_DISABLE\_RECOVERY» soit mis à *true*, deux entrées de menu sont générées pour chaque noyau Linux: une « par défaut » et l'autre pour le « *recovery mode* ». Cette option répertorie les arguments à ajouter uniquement à l'entrée « par défaut », après ceux énumérés dans « GRUB\_CMDLINE\_LINUX ».
- GRUB\_DISABLE\_RECOVERY : la valeur *true* de ce paramètre permet de désactiver la génération de l'entrée du menu relative au mode *Recovery*.
- GRUB\_DISABLE\_OS\_PROBER : la valeur *true* de ce paramètre désactive l'utilisation de os-prober pour détecter les OS installés

# **/etc/default/grub**

- **GRUB\_GFXMODE** : spécifie la résolution à utiliser. Seules les résolutions supportées via le « VESA BIOS Extensions » (VBE) sont disponibles.
- **GRUB\_BACKGROUND** : Définir une image d'arrière-plan. La valeur de cette option doit être un fichier lisible par GRUB au démarrage. Les formats supportés sont : png, tga, jpg et jpeg. L'image sera réduite si nécessaire pour s'adapter à l'écran.
- **GRUB\_THEME** : Définir un thème à utiliser dans le terminal graphique.
- **GRUB\_GFXPAYLOAD\_LINUX** : Réglez-le sur «texte» pour forcer le noyau Linux à démarrer en mode texte normal

# /etc/grub.d

- Les autres fichiers de configuration se trouvent dans le répertoire /etc/grub.d/, on y trouve sur Fedora :
  - 00\_header
  - 10\_linux
  - 20\_linux\_xen
  - 30\_os-prober
  - 40\_custom
  - 41\_custom
- Le préfixe numérique dans les noms des scripts sert à déterminer l'ordre dans lequel ces scripts seront lus par “*grub2-mkconfig*”.

# /etc/grub.d

- **00\_header** : script permettant la génération de l'en-tête du fichier grub.cfg. Cet en-tête est généré principalement à l'aide des informations paramétrées dans le fichier /etc/default/grub.
- **10\_linux** : script permettant de générer les entrées correspondant au système GNU/Linux hôte.
- **30\_os-prober** : script permettant de détecter les autres systèmes installés et générer les entrées correspondantes. Il est normalement capable de détecter les systèmes avec un noyau Linux, un noyau Hurd, les systèmes Windows et les systèmes Mac OS X.
- **40\_custom** : script permettant de générer des entrées introduites manuellement dans ce fichier.



# Démarrage du noyau

Après son chargement, le noyau :

- Met en place de la gestion des interruptions
- Initialise la mémoire virtuelle
- Initialise le vecteur d'interruptions, le planificateur (le "scheduler") et l'horloge
- Traite les arguments de la ligne de commande
- Détecte le hardware et charge des modules nécessaires
- Passe au mode multi\_user si sollicité

# Processus PID 1

- A ce stade, le noyau libère l'espace utilisé précédemment pour la détection et la configuration des périphériques :  
**Freeing unused kernel memory...**
- Ensuite, il lance le premier processus (PID 1)
- Sur un système UNIX, le tout premier processus (PID 1) a un rôle particulier :
  - c'est le seul qui est lancé par le noyau ; il doit donc assurer toute l'initialisation du système, le lancement des différents services ;
  - lorsqu'un processus devient orphelin, la norme et l'usage veut qu'il soit rattaché au processus de PID 1. Dans ce cas lorsque le processus se termine, c'est au PID 1 qui revient de lire le code de retour du processus zombie, avant que le système ne puisse libérer certaines ressources associées.

# Les services

- Le processus PID 1 est donc le processus d'initialisation du système et de gestion des services
- Différents système de gestion des services sont utilisés :
  - SysVinit (init System V).
  - UpStart.
  - systemd.
- SysVinit est le classique système de démarrage et de gestion des services qui a été utilisé jusqu'à Fedora 8. UpStart a remplacé SysVinit dans Fedora 9, jusqu'à Fedora 14. systemd prend place dans Fedora à partir de la version 15.
- La commande : *ps tree* permet d'afficher l'arbre des processus. Celui au sommet est le gestionnaire de services utilisé.

# *sysVinit*

- Est le système d'initialisation hérité d'UNIX Système V
- Ce système est représenté par le processus **/sbin/init**
- Le processus **/sbin/init** a le **pid = 1**
- Il est le parent de tous les processus présents sur le système (sauf du singulier **idle**).
- Le processus **init** effectue un contrôle des partitions puis procède au montage du système de fichier principal sous **/**
- Après, **init** lance les différents services suivant le contenu du fichier **/etc/inittab** en faisant la distinction entre les différents niveaux d'exécution (**runlevels**)

# Niveau d'exécution

- Un niveau d'exécution (Run Level) est une configuration qui définit l'ensemble des services en exécution
- Les niveaux d'exécution prédéfinis dans Linux Redhat/Fedora sont:
  - 0 — Halt
  - 1 — Single-user text mode
  - 2 — Not used (user-definable)
  - 3 — Full multi-user text mode
  - 4 — Not used (user-definable)
  - 5 — Full multi-user graphical mode (with an X-based login screen)
  - 6 — Reboot

# Niveau d'exécution

- La commande :

`init N`

ou

`telinit N`

Permet de faire passer le système au niveau passé en argument

- La commande

`runlevel`

ou

`who -r`

- Permet d'afficher le niveau courant ainsi que le précédent

# Le fichier /etc/inittab

```
id:5:initdefault:
si::sysinit:/etc/rc.d/rc.sysinit
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l2:2:wait:/etc/rc.d/rc 2
...
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure"
# Run xdm in runlevel 5
x:5:once:/etc/X11/prefdm -nodaemon
```

# Le fichier /etc/inittab

Ensemble de lignes ayant la syntaxe suivante :

**<code> : <niveau> : <action> : <commande>**

**code** est le label de la ligne.

**niveau** correspond au(x) niveau(x) où doit être exécutée la commande.

**action** désigne le mode de lancement de la commande

**commande** la commande à exécuter



# Le fichier /etc/inittab (suite)

## Les actions possibles sont:

<b>respawn</b>	relance automatiquement le processus une fois terminé.
<b>wait</b>	<i>init</i> attend la fin du processus avant de poursuivre.
<b>once</b>	pour une exécution unique du processus à l'arrivée dans le niveau spécifié.
<b>boot</b>	qui définit que le processus doit être lancé au moment du boot système (le niveau est ignoré).
<b>bootwait</b>	reprend les actions de wait et de boot cumulées.
<b>off</b>	désactive l'entrée
<b>ondemand</b>	pour lancer les processus lors d'un pseudo passage en ondemand runlevel.

# Le fichier /etc/inittab (suite)

- initdefault**     pour définir le niveau par défaut
- sysinit**        définit un processus à lancer pendant le démarrage. Il sera exécuté avant toutes les lignes boot ou bootwait
- powerwait**     définit le processus à lancer quand init reçoit le signal SIGPWR. init attendra que le processus soit terminé.
- powerfail**     idem que powerwait mais init n'attend pas que le processus soit terminé.
- powerokwait**   définit le processus à lancer si un SIGPWR est reçu. Le fichier /etc/powerstatus est contrôlé. Si celui-ci contient OK, le problème d'alimentation est résolu.
- ctrlaltdel**     définit le processus à lancer si la séquence de touches CTRL + ALT + SUPP est initiée.

# Le scripte **rc.sysinit**

- init exécute tout d'abord **/etc/rc.d/rc.sysinit**.
- **rc.sysinit**
  - termine la configuration de base du système :
    - mettre l'horloge du système à l'heure du CMOS,
    - charge la configuration clavier adéquate,
    - active la partition de swap,
    - configure le nom de la machine et la variable PATH
  - vérifie aussi que le système de fichier racine ne comporte pas d'erreur et le remonte en rw.
  - Et active les modules nécessaires en appelant **/etc/rc.d/rc.modules**.

# Le script rc

- **init** lance en suite le script **/etc/rc.d/rc** en lui passant comme paramètre le niveau d'exécution souhaité.
- À chaque niveau d'exécution (***Run Level***) correspond un ensemble de services lancés automatiquement lorsque le système entre le niveau en question.
- La configuration de chaque ***runlevel*** est sauvegardée dans le répertoire **/etc/rc.d/rcX.d/** où **X** est le ***runlevel*** en question
- Les différents niveaux d'exécution peuvent être gérés via une interface : **ntsysv**, **tksysv** ou **chkconfig**

# Le répertoire `/etc/rc.d`

Ce répertoire contient les scripts utilisés pour l'initialisation du système.

- ***rc.sysinit*** exécuté une fois au démarrage pour initialiser le système
- ***rc*** script de gestion du niveau d'exécution. Il le reçoit en paramètre.
- ***rc.local*** script utilisé pour les initialisations particulières à la machine
- ***init.d*** répertoire contenant les scripts d'initialisation des sous-systèmes
- ***rc0.d*, ..., *rc6.d*** répertoires contenant des liens sur les scripts du répertoire ***init.d*** devant être lancés à un niveau d'exécution particulier

# Le répertoire `/etc/rc.d/init.d`

- Contient l'ensemble des scripts de démarrage des différents services installés
- Tous les scripts ont pratiquement la même syntaxe d'activation

**`/etc/init.d/script [start | stop | restart | status]`**

**ou**

**`service nom-script [start | stop | restart | status]`**

- l'ensemble de ces scripts ont été mis en place au moment de l'installation des paquetages correspondants.

# Les répertoires `/etc/rc.d/rcN.d`

- Contient des liens symboliques vers les services du répertoire **init.d**
- Les liens des répertoires `rcN.d` reprennent les noms des scripts d'origine précédés par la lettre S ou K (respectivement pour Start et Kill) et d'un nombre à deux chiffres.
  - Le lien **S55sshd** du répertoire `/etc/rc.d/rc3.d` indique le lancement du service `sshd` au niveau d'exécution 3
  - Le lien **K20nfs** du `/etc/rc.d/rc2.d` indique l'exécution
- La valeur numérique permet de spécifier l'ordre d'arrêt ou de démarrage du service.
- La plupart des démons du système enregistrent leur numéro de processus dans un fichier du répertoire `/var/run`.

# Configuration des RL

- Manuellement en maniant directement les liens :

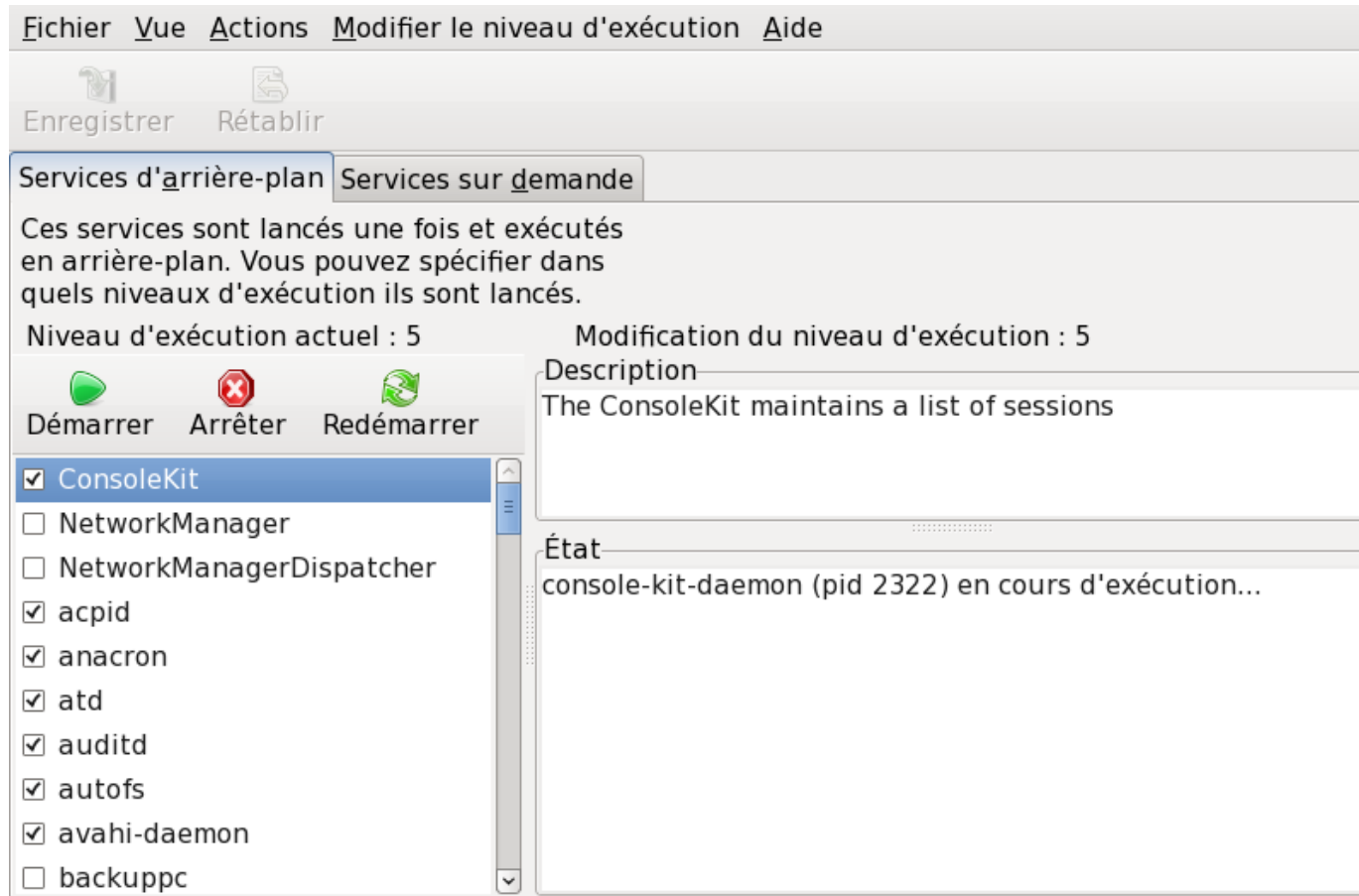
```
# ln -sf /etc/rc.d/init.d/httpd /etc/rc.d/rc5.d/S85httpd
```

problème:

- Il faut connaître exactement les priorités de lancement et d'arrêt. Ceux-ci sont visibles dans les fichiers scripts correspondants
  - Lancer une commande pour chaque service et chaque niveau d'exécution
- Par la commande : *chkconfig*
- Graphiquement à l'aide de l'application GUI disponible comme : ***system-config-services*** pour Fedora, visible aussi sous le menu: *paramètres système → paramètres de serveur → services*



# GUI Configuration des RL



# La commande *chkconfig*

## Syntaxe:

```
chkconfig --list [name]
chkconfig --add name
chkconfig --del name
chkconfig [--level levels] name <on|off|reset>
```

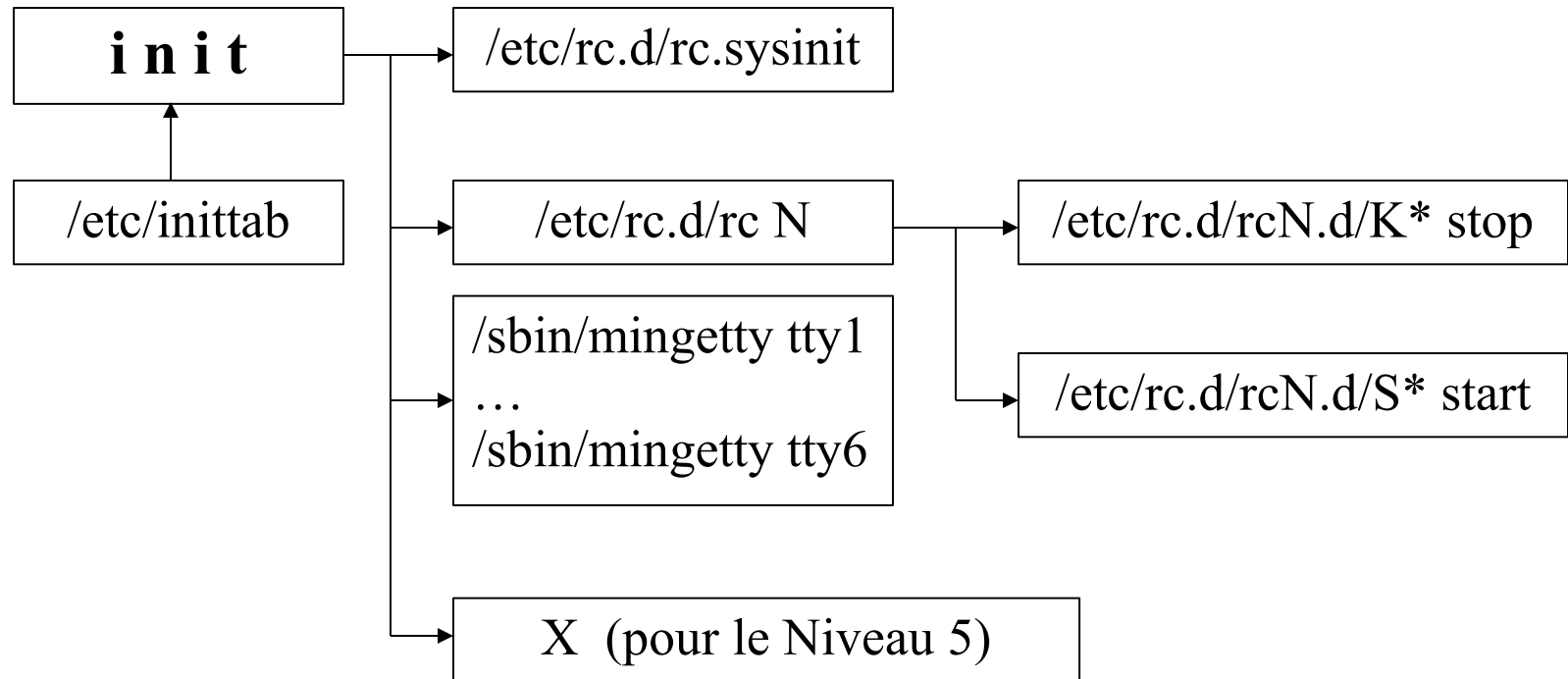
## Permet de:

- visualiser la situation
- ajouter les liens nécessaires pour démarrer/arrêter les services nouvellement installés.
- supprimer un service (supprime toutes les indications dem/arr)
- modifier la configuration des run level

Exemples :

```
# chkconfig --list httpd
# chkconfig --level 35 httpd on
```

# *i n i t* : récapitulatif



# UpStart

- Malheureusement, SysVinit reste très basique, peu performant et sujet à de nombreux problèmes. Il n'est pas très adapté à toutes les évolutions modernes que connaît Linux.
- Ces principaux défauts :
  - Le démarrage des services est séquentiel.
  - La prise en charge des dépendances entre services est très basique
  - Tout ou presque est sous forme de scripts Shell ; ceci rend le processus de démarrage lent et fastidieux.
  - ...
- Dans le but de paralléliser le démarrage des services, *UpStart* a été proposé comme remplaçant à sysVinit.

# *UpStart* : Avantages

- Par rapport à *sysVinit*, *UpStart* présente les avantages suivants :
  - Lancement et arrêt des services par événement
  - Gestion des dépendances pour permettre le lancement parallèle des services au démarrage
  - Possibilité de définir des services utilisateurs lancés et arrêtés par ces derniers
  - ...

# *systemd*

- *systemd* est un gestionnaire du système et des services. Il permet entre autre de lancer/arrêter les services de gérer les points de montage, les sessions, les points de communication socket et DBUS, etc.
- Par rapport à ses prédécesseurs, *systemd* :
  - Parallélise efficacement le démarrage des services
  - Gère les points de communication sockets et D-Bus
  - Permet le lancement à la demande des services
  - Les dépendance entre services sont traitées d'une manière transactionnelle
  - Supporte le « snapshotting and restoring »
  - Maintient les points de montages

# ***Systemd : Commandes***

- systemd fournit un large panel de commande qui permet d'avoir des informations ou de modifier l'état du système.
- Sans être exhaustif, voici les plus importantes :
  - **systemctl** : contrôle systemd et gère les unités.
  - **journalctl** : consultation du journal de systemd.
  - **loginctl** : contrôle des sessions utilisateurs (systemd-logind).

# ***Systemctl : pour l'arrêt et redémarrage du système***

- Les actions sont envoyées en utilisant la commande `systemctl` :
- Redémarrer ou arrêter :  
# `systemctl reboot`  
# `systemctl poweroff`
- Les commandes `reboot` et `poweroff` sont désormais des liens vers ***systemctl***
- Mettre en veille ou en hibernation :  
# `systemctl suspend`  
# `systemctl hibernate`



# *systemd : Unité*

- Une unité représente un fichier de configuration. Entre autres, une unité peut être un service (\*.service), un target (\*.target), un montage (\*.mount), un socket (\*.socket)...
- Liste les unités:  
# systemctl  
# systemctl list-units
- Démarrer, arrêter, redémarrer ou recharger une unité:  
# systemctl start <unit>  
# systemctl stop <unit>  
# systemctl restart <unit>  
# systemctl reload <unit>

# ***systemd : Unité***

- Voir son statut:  
`# systemctl status <unit>`
- Activer, désactiver une unité au démarrage:  
`# systemctl enable <unit>`  
`# systemctl disable <unit>`
- Lister les dépendances d'une unité:  
`# systemctl list-dependencies [<unit>]`
- Note: Il faut utiliser le nom du fichier d'une unité en entier, exemple:  
`# systemctl restart avahi-daemon.service`

# *systemd : Unité*

- Néanmoins, certains raccourcis d'écriture sont disponibles: sans suffixe, *systemctl* présume qu'il s'agit d'un .service. Ainsi, dbus et dbus.service sont équivalents:  

```
# systemctl status dbus
```
- un point de montage est automatiquement retranscrit en l'unité .mount appropriée. Par exemple /home est équivalent à home.mount:  

```
# systemctl status /home
```
- de la même manière, un périphérique est retranscrit en l'unité .device appropriée. Ainsi, /dev/sda2 est équivalent à dev-sda2.device:  

```
# systemctl status /dev/sda2
```

# *systemd : Unité*

- Recharger la configuration des services (après modification d'une unité):

```
# systemctl daemon-reload
```

- Les unités peuvent correspondre à des instances d'un fichier template, ceci permet d'avoir un fichier de configuration pour plusieurs unités. Ces unités sont reconnaissables par le @ inclus dans leur nom. Un exemple concret est le service dhcpd@.service. Ce dernier permet d'activer le DHCP sur une interface :

```
# systemctl start dhcpd@eth0.service
```

- Pour activer le service au démarrage :

```
# systemctl enable dhcpd@eth0.service
```

# *systemd : services*

- Un service est une unité ayant comme suffixe .service. La commande :

```
# systemctl list-unit-files -t service
```

permet d'afficher la liste de tous les services installés avec leur état d'activation

# ***systemd : target***

- Un target est une unité particulière, elle permet de regrouper d'autres unités. Son nom de fichier prend le suffixe .target
- Les targets permettent de fournir l'équivalent des niveaux d'exécution (runlevel) de sysvinit :

# *systemd : target*

SystemV init Runlevel	Systemd Target	Notes
0	runlevel0.target, poweroff.target	Arrête le système
1, s, single	runlevel1.target, rescue.target	Mode single user.
2, 4	runlevel2.target, runlevel4.target, multi-user.target	Mode défini par l'utilisateur, identique au 3 par défaut.
3	runlevel3.target, multi-user.target	Multi-utilisateur, non grap
5	runlevel5.target, graphical.target	Multi-utilisateur, en mode
6	runlevel6.target, reboot.target	Redémarre
emergency	emergency.target	Shell d'urgence

# *systemd : target*

- Pourz voir ce que regroupe un target :

```
# systemctl show -p Wants -p Requires <target>
```

- Par exemple, on peut voir que graphical ne fait que rajouter un gestionnaire de connexions en plus de multi-user :

```
# systemctl --no-pager show -p Wants -p Requires graphical.target  
Requires=multi-user.target
```

```
Wants=display-manager.service systemd-update-utmp-runlevel.service
```

- Pour changer de target, par exemple pour passer au multi-user, lancez l'une de ces commandes:

```
# systemctl isolate multi-user.target  
# systemctl isolate runlevel3.target  
# telinit 3
```



# ***systemd : target***

- Le target par défaut à l'installation est graphical :

```
# readlink /usr/lib/systemd/system/default.target  
graphical.target
```

- Ou

```
# systemctl get-default
```

- Pour spécifier un autre niveau par défaut, par exemple le multi-user :

```
# ln -sf /lib/systemd/system/multi-user.target  
/etc/systemd/system/default.target
```

- Ou

```
# systemctl set-default multi-user.target
```

# L'arrêt du système

- La commande *shutdown* permet, selon les options utilisées :
  - d'avertir les utilisateurs
  - de demander aux applications de s'arrêter correctement en fermant les connexions et fichiers ouverts (signal SIGTERM)
  - de vider les tampons mémoire du cache disque (sync)
  - de programmer l'heure de l'arrêt / redémarrage (*now*, *hh:mm*, *+minutes*)
  - de spécifier le mode d'arrêt (arrêt ou redémarrage avec ou non un fsck)
  - ...
- Deux autres commandes : *reboot* et *halt* (brutales)

# Gestion des Paquetages RPM

# Installation des paquets

- Au début de l'histoire de Linux, l'installation des applications :
  - s'effectuait à l'aide des fichiers tar.
  - Était manuelle (par lignes de commandes) et difficile : passe souvent par une compilation des sources.
- Actuellement :
  - Des formats spécifiques ont vu le jour (deb, rpm)
  - La procédure d'installation est prise en charge par l'interface graphique: sous Fedora, il suffit de cliquer sur un fichier *rpm* pour que l'IG demande l'autorisation de l'installer automatiquement.
  - Des outils graphiques d'installation/des-installation sont maintenant disponibles (synaptic, yumex, yast, ...)
  - La plupart des distributions mettent à disposition de la communauté des dépôts complets de paquets.

# Formats d'installation

- Sous Linux, les fichiers d'installation peuvent avoir être de différents formats :
  - tar : Le format le plus ancien et le plus répandu mais aussi le plus difficilement maniable. Les fichiers tar contiennent généralement des sources et peuvent contenir aussi des binaires. Ils s'installent sur toutes les distributions.
  - rpm : Format d'installation conçu par RedHat et utilisé particulièrement sous RedHat, Fedora, Mandriva et SuSe.
  - deb : Conçu à l'origine par Debian et utilisé par les distribution à base de Debian (Ubuntu, Mepis, Knoppix, ...)
  - On peut rencontrer aussi d'autres formats comme : zip, bin, jar, ...

# Installation des fichiers tar

- La procédure d'installation des fichiers tar dépend du type de contenu de ces fichiers (sources, binaires, ...) :
  - Extraire l'archive : par interface graphique ou par commande *tar*
  - Chercher et lire un fichier de l'archive décrivant la procédure d'installation (INSTALL, README, install.txt, ...)
- La procédure d'installation des sources passe généralement par les trois commandes:
  - `./configure` cherche les dépendances et génère le Makefile (fichier utilisé par *make* et contenant le schéma de compilation et d'installation)
  - `make` compile les sources
  - `make install` installe l'application (doit être lancé en administrateur)

# Paquetage RPM

- Paquetage (*package*) RPM = Fichier archive contenant un ou plusieurs programmes prêt à s'installer sur la machine par une simple commande ou un clic de souris.
- Un paquetage RPM contient :
  - Executables et librairies
  - Fichiers de configuration
  - Documentation (généralement des fichiers man)
  - Informations et scripts pour installation / désinstallation
  - Informations concernant les dépendances.
  - Informations complémentaires (nom, version, description, auteurs, dates, ...)

# L'outil RPM

- RPM (Redhat Package Manager) = outil puissant de gestion des paquetages, conçu par RedHat Labs et inspiré de dpkg (gestionnaire de paquetage de Debian).
- Placé sous licence GPL. Il est utilisé par les 3 grosses distributions : RedHat, Mandrake et SuSe.
- Permet de construire, installer, interroger, vérifier, mettre à jour, et désinstaller des paquetages de logiciels tout en incluant la notion de dépendance.
- Maintient une base de donnée (/var/lib/rpm) des paquetages installés.
- Utilise PGP/GPG pour la vérification des signatures



# RPM : Installation, Désinstallation, Mise à jour

- Installer un nouveau paquetage : `rpm -i fichier_paquetage`
- Désinstaller un paquetage existant: `rpm -e nom_du_paquetage`
- Mettre à jour ou installer  
s'il ne l'est pas encore : `rpm -U fichier_paquetage`
- Rafrâchir  
(c-a-d : mettre à jour uniquement  
si le paquetage est déjà installé) `rpm -F fichier_paquetage`

# RPM : Installation, Désinstallation, Mise à jour

- Les noms des fichiers paquetages ont tous la forme suivante :

`nom_du_paquetage-version.plateforme.rpm`

Ex:                `gcc-4.4.4-10.fc13.x86_64.rpm`  
                  `gzip-1.3.13-4.fc13.i386.rpm`

- En cas d'installation ou de mise à jour Le fichier paquetage peut être spécifié par une URL *ftp* ou *http*, auquel cas le paquetage sera téléchargé avant d'être installé.

la forme générale une URL *ftp* est :

`ftp://<user>:<password>@site:<port>/chemin/vers/paquetage.rpm`

# RPM : Installation, Désinstallation, Mise à jour

## Quelques autres options

<code>-v</code>	verbose
<code>-vv</code>	Afficher un tas d'horribles informations de débogage.
<code>--quiet</code>	Seuls les messages d'erreur seront affichés.
<code>--prefix &lt;path&gt;</code>	Si possible utiliser <path> comme chemin d'installation
<code>--dbpath &lt;chemin&gt;</code>	Spécifier le chemin de la base de données RPM
<code>--excludedocs</code>	Ne pas installer les fichiers de documentation.
<code>--force</code>	Forcer l'opération en dépit des problèmes rencontrés
<code>-h, --hash</code>	Afficher 50 marques de hachage quand l'archive est déballée. A utiliser avec <code>-v</code> pour un bel affichage.
<code>--nodeps</code>	Ne pas effectuer de vérification de dépendances.
<code>--test</code>	Ne pas installer, rapporter les conflits potentiels.

# RPM : Interrogation

- La forme générale d'une commande de requête rpm est :

**rpm -q [options\_de\_requête]**

- Exemples :

Pour rechercher la version installée d'un package

**rpm -qa | grep package** OU **rpm -q package**

Pour obtenir des informations sur un package :

**rpm -qpi nom\_du\_package.rpm**

**rpm -qi package** s'il est installé

Pour savoir le nom du package auquel appartient un fichier :

**rpm -qf nom\_du\_fichier**

Pour avoir la liste des fichiers installés par un package

**rpm -ql package** s'il est installé

**rpm -qlp package.i386.rpm** s'il ne l'est pas encore

# RPM : Interrogation

- En générale Il y a deux sous-ensembles d'option pour les requêtes : la sélection de paquetages et la sélection d'information.
- Quelques options de sélection de paquetages :

<code>&lt;nom_package&gt;</code>	Interroger le paquetage intallé nommé <code>&lt;nom_package&gt;</code>
<code>-a, --all</code>	Afficher tous les paquetages installés.
<code>--whatrequires &lt;capacité&gt;</code>	Afficher tous les paquetages qui ont besoin de <code>&lt;capacité&gt;</code> pour leur fonctionnement propre.
<code>--whatprovides &lt; capacité&gt;</code>	Interroger tous les paquetages qui fournissent la capacité <code>&lt; capacité&gt;</code> .
<code>-f &lt;fichier&gt;</code>	Interroger le paquetage possédant <code>&lt;fichier&gt;</code> .
<code>-p &lt;fichier_package&gt;</code>	Interroger le fichier paquetage spécifié.

# RPM : Interrogation

Quelques options de sélection d'information :

<code>-i</code>	Afficher l'information du paquetage, incluant son nom, sa version, et sa description.
<code>-R, --requires</code>	Lister les capacités dont dépend ce paquetage.
<code>--provides</code>	Lister les capacités fournis par ce paquetage.
<code>-l, --list</code>	Lister les fichiers inclus dans le paquetage.
<code>-s, --state</code>	Afficher les états des fichiers du paquetage. L'état de chaque fichier est soit normal, non installé, ou remplacé.
<code>-d, --docfiles</code>	Lister uniquement les fichiers de documentation
<code>-c, --configfiles</code>	Lister uniquement les fichiers de configuration
<code>--scripts</code>	Lister les scripts post / pré installation / désinstallation.
<code>--dump</code>	Pour chaque fichier lister l'ensemble d'informations associées: taille, droit d'accès, somme MD5, proprio, ...

# RPM : vérification

- La forme générale d'une commande de vérification *rpm* est

**`rpm -V | --verify [verify-options]`**

- Compare les informations sur les fichiers installés avec celles obtenues à partir du paquetage original et conservées dans la base de données *rpm*.
- Entre autres choses, la vérification compare la taille, la somme MD5, les permissions, le propriétaire et les groupes de chaque fichier.
- Les options de spécification de paquetage sont les mêmes que pour l'interrogation.
- Les fichiers qui n'étaient pas installés en provenance du paquetage, par exemple les fichiers de documentation exclus à l'installation en utilisant l'option "`--excludedocs`", sont ignorés.

# RPM : vérification

- Options pouvant être utilisées en mode de vérification :
  - nodeps      Ne pas vérifier les dépendances.
  - nomd5      Ne pas vérifier la somme MD5 des fichiers.
  - nofiles      Ne pas vérifier les attributs des fichiers (dépendance seule).
- Le format de sortie est constitué d'une chaîne de 8 caractères suivie d'un "c" éventuel dénotant un fichier de configuration, et ensuite le nom du fichier.
- Chacun des 8 caractères dénote le résultat d'une comparaison d'un attribut du fichier avec la valeur de cet attribut enregistré dans la base de données rpm.
- Un simple "." (point) signifie que le test s'est bien passé, alors qu'un "?" seul indique que le test n'a pas pu être effectué (p.ex. quand les permissions d'accès aux fichiers empêchent la lecture).



# RPM : vérification

- Les caractères suivants dénotent l'échec à certains tests :

<b>S</b>	la taille (Size) du fichier diffère
<b>M</b>	le Mode diffère (inclut les permissions et le type du fichier)
<b>5</b>	la somme MD5 diffère
<b>D</b>	le numéro de périphérique (Device) majeur/mineur diffère
<b>L</b>	le chemin renvoyé par readLink(2) diffère
<b>U</b>	le propriétaire (User) diffère
<b>G</b>	le Groupe diffère
<b>T</b>	la date de dernière modification (mTime) diffère

- **Exemple:**

```
[shems]$ rpm -V httpd
SM5....T  c      /etc/httpd/conf/httpd.conf
..?.....  /usr/sbin/suexec
```

# RPM : signatures

- Les paquetages *rpm* peuvent être signés par leurs distributeurs.

```
[Shems]$ rpm -qip aairsnort-0.2.7e-5.fc4.i386.rpm |  
grep -i signature
```

```
Signature      : DSA/SHA1, Sat Aug 13 03:36:29 2005,  
Key ID 82ed95041ac70ce6
```

- l'option `-K` ou `--checksig` de la commande *rpm* permet de vérifier l'intégrité et l'authenticité du paquetage en se basant sur la signature elle même et la clé public du signataire.

```
[Shems]$ rpm -K aairsnort-0.2.7e-5.fc4.i386.rpm  
aairsnort-...: (sha1) dsa sha1 md5 gpg OK
```

# RPM : signatures

- En plus de la taille, 4 différents contrôles sont testés :

```
[Shems]$ rpm -Kv airtsnort-0.2.7e-5.fc4.i386.rpm
airtsnort-0.2.7e-5.fc4.i386.rpm:
  Header V3 DSA signature: OK, key ID 1ac70ce6
  Header SHA1 digest: OK (bcf7bdd5b5c2403deae3)
  MD5 digest: OK (2a3b2a50ec82a1d01033571bb2)
  V3 DSA signature: OK, key ID 1ac70ce6
```

- Si le paquetage n'est pas signé, seuls les hachages SHA1 de l'entête et MD5 du corps (en plus de la taille) sont vérifiés :

```
[Shems]$ rpm -qip skype.rpm |grep -i signature
Signature      : (none)
[Shems]$ rpm -Kv skype-1.2.0.17-fc3.i586.rpm
skype.rpm:
  Header SHA1 digest: OK (c06b173cbc3e57a68b0e4)
  MD5 digest: OK (cd51f228559b0d0d0ea2aba8e6f1)
```

# RPM : signatures

- En cas d'absence de la clé public de l'organisme signataire :

```
[Shems]$ rpm -K bandwidth-0.9.4-rc1.i386.rpm
bandwidth-0.9.4-rc1.i386.rpm: (SHA1) DSA sha1 md5
(GPG) NOT OK (MISSING KEYS: GPG#ba560df3)
```

1. Obtenir la clé public du distributeur à partir de :

- CD-ROM
- Internet
- Serveur de clés

2. Ajouter la clé en question au porte-clés (*keyring*)

```
[Shems]# rpm --import url
```

3. en suite, vérifier le paquetage

```
[Shems]# rpm -K bandwidth-0.9.4-rc1.i386.rpm
bandwidth...rpm (sha1) dsa sha1 md5 gpg OK
```

# Dépôts de paquets

- La plupart des distributions maintiennent des sites web/ftp (appelés dépôts) contenant des paquets.
- Pour Fedora les adresses de ces dépôts sont mise dans des fichiers (ayant l'extension `.repo`) se trouvant sous `/etc/yum.repos.d/`
- Ces dépôts peuvent être consultés manuellement par n'importe quel navigateur web
- Leur principal intérêt est qu'ils sont consultables automatiquement par les interfaces graphiques d'installation/des-installation ou par la commande ***yum***

# Dépôts de paquetages

- À l'installation de Fedora seuls les dépôts officiels suivant sont reconnus:
  - Fedora dépôt de base contenant l'ensemble des paquetages sélectionnés par RedHat
  - Update dépôt des mises à jour
- D'autres dépôts peuvent être rajoutés, en particulier : *rpmfusion*  
Le détail concernant ce dépôt ainsi que la procédure d'ajout se trouve à l'adresse : <http://www.rpmfusion.org/>
- La liste des dépôts pour Fedora est disponible en particulier dans le *wiki* français de la distribution :  
<http://doc.fedora-fr.org/wiki/Accueil>

# Interfaces graphiques

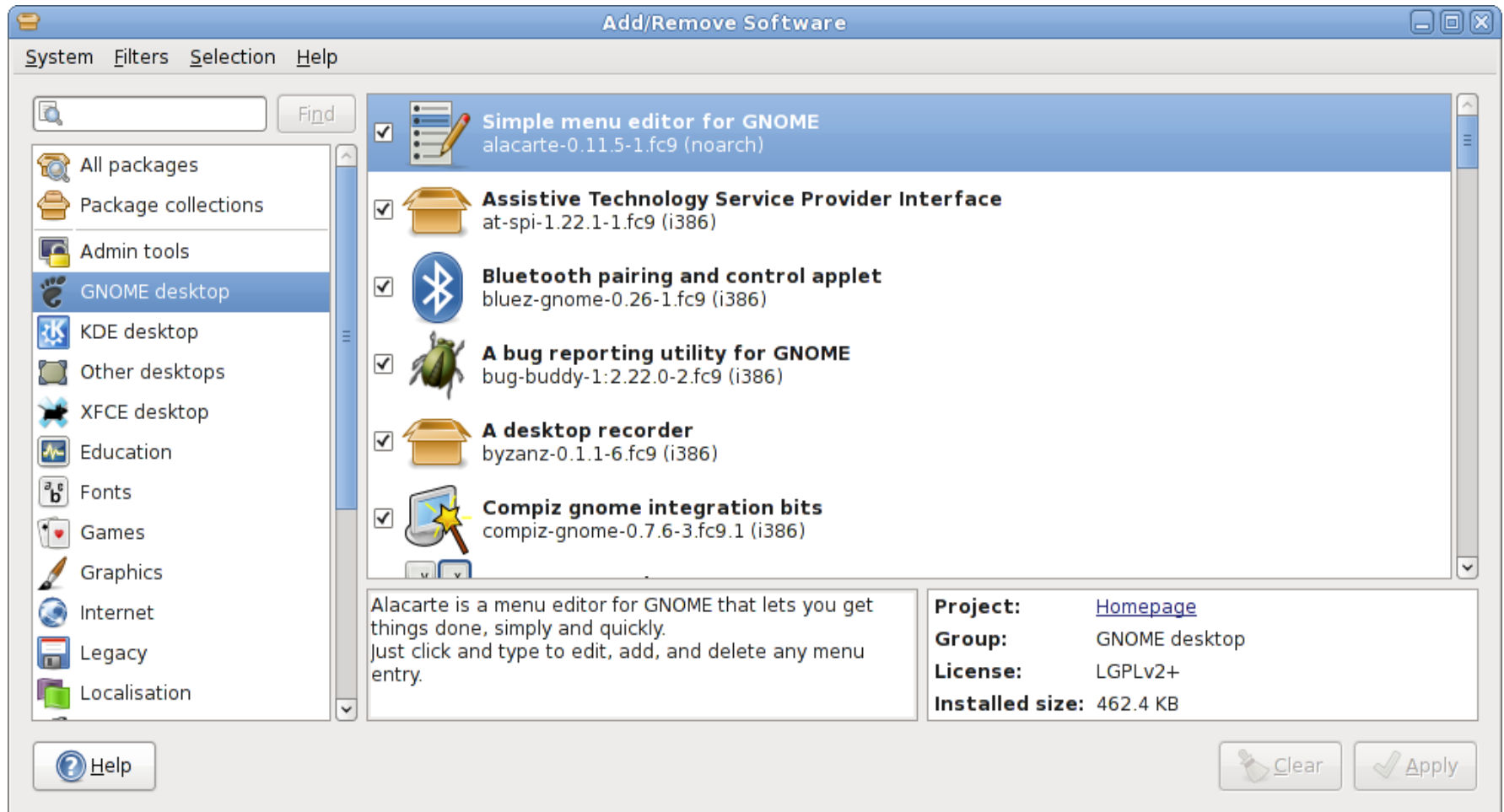
- Sous Fedora , Trois applications graphiques étaient disponibles par défaut :
  - *pirut* : se charge des installations/dés-installations en utilisant les dépôts. Elle est accessible via le menu :  
*Applications → Ajouter/Enlever des logiciels*
  - *Pup* : permet de lancer uniquement les opérations de mise à jour. Il est accessible via le menu :  
*Applications → Outils Système → Mise à jour des logiciel*
  - *Yum-updatesd* : un service de notification des mise à jour.
- D'autre interfaces graphiques étaient également disponibles, *yumex* en l'occurrence.

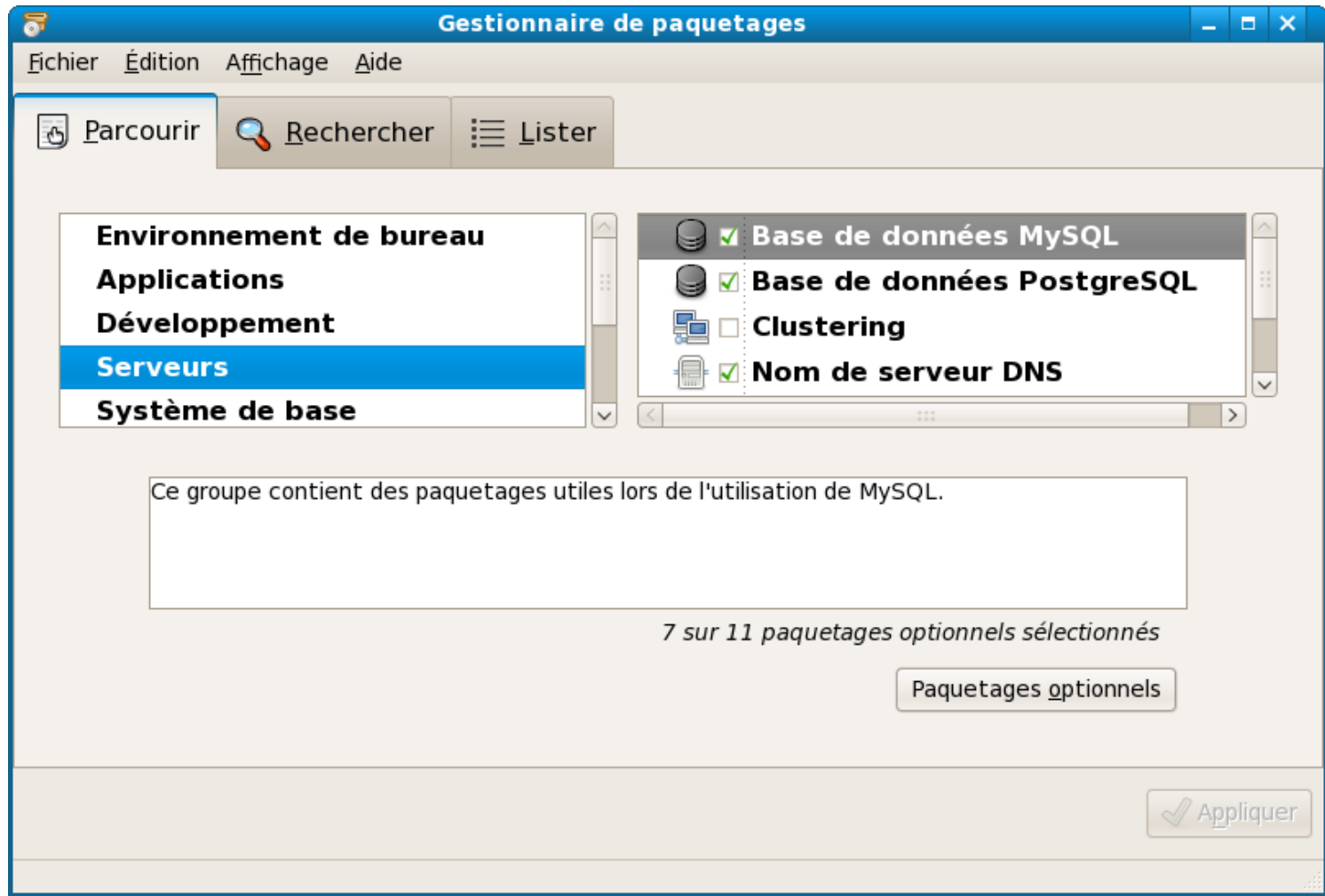
# Interfaces graphiques

- Avec Fedora 9, les trois outils de la précédente version sont remplacés par un seul: PackageKit. Avantage :
  - Indépendance vis à vis les gestionnaires de paquetages utilisés: yum, apt, ... PackageKit tente ainsi d'unifier les interfaces des gestionnaires de paquetages
  - intégration avec PolicyKit : un mot de passe en moins à saisir pour installer les paquets
  - La transparence d'utilisation : avec l'icône dans la zone de notification, pouvant aller jusqu'à travailler tout seul dans son coin sans avoir de fenêtre ouverte sur le bureau
- *yumex* est toujours disponible pour assurer la continuité

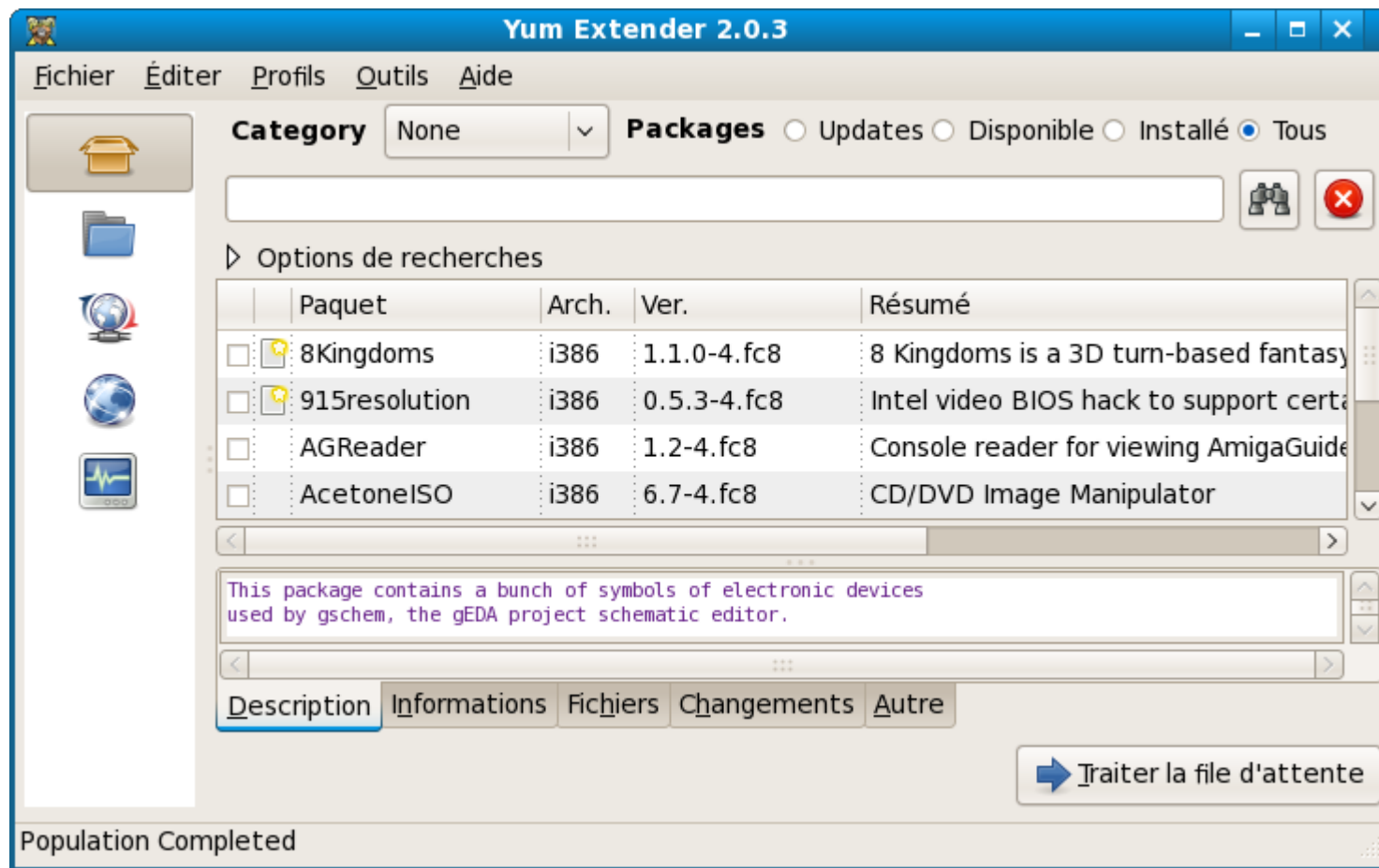


# PackageKit





# yumex



# Commande yum

- Utilise les dépôts pour installer, des-installer et mettre à jour des paquets

- Usage :

<code>yum list all</code>	liste tous les paquets disponibles dans les dépôts
<code>yum info p1</code>	affiche les informations concernant les paquets passés en argument
<code>Yum search mot_clé</code>	recherche dans les dépôts les paquets concernant le mot_clé cité
<code>yum install p1</code>	installe les paquets spécifiés
<code>yum check-update</code>	affiche la liste des mises à jour disponibles
<code>yum update [p1...]</code>	met à jour les paquets spécifiés (à défaut, applique toutes les mises à jour disponibles)
<code>yum remove p1 ...</code>	dés-installe les paquets passés en argument
<code>yum repolist</code>	liste les dépôts disponibles
<code>yum clean all</code>	vide les caches

# **Observation et Analyse du système**

# who

- Lister les connexions
  - **who** permet de voir la liste des utilisateur connectés
  - **who** lit par défaut le fichier /var/run/utmp. Un fichier de remplacement peut être fourni en argument (/var/log/wtmp par exemple)
  - **who** possède plusieurs option permettant de raffiner la liste affichée
    - Une autre commande **w** affiche plus d'information pour chaque utilisateur connecté
    - La commande **finger** offre à peu près les mêmes fonctionnalités. Elle possède cependant une fonctionnalité réseau réputée dangereuse

# last

- historique des connexions: **last**

affiche la liste des utilisateurs dernièrement connectés (login, terminal, date...). Elle puise dans **/var/log/wtmp** ou dans le fichier indiqué par l'option -f

Des noms d'utilisateurs ou de terminaux peuvent être spécifiés, afin que last ne montre que les connexions/déconnexions correspondant à ces arguments.

Le pseudo-utilisateur **reboot** est enregistré à chaque redémarrage du système

**lastb** se comporte comme last, mais il utilise le fichier **/var/log/btmp** qui journalise toutes les tentatives infructueuses de connexion.

les options -d et -x sont particulièrement intéressante pour afficher respectivement les connexions non-locales, et les arrêts / modifications de run level.

# lastlog

- historique des connexions : **lastlog**  
exploite le binaire `/var/log/lastlog` , pour afficher le login, le port et la durée de la dernière connexion pour l'ensemble des utilisateurs déclarés

## SYNOPSIS

**lastlog [-u login-name] [-t days]**

L'option `-u` permet de restreindre l'affichage pour l'utilisateur spécifié

`lastlog -t 5` affiche les utilisateurs qui se sont connectés les 5 derniers jours.



# vmstat

## Statistiques d'utilisation des ressources

- La commande **vmstat** permet d'afficher les statistiques sur l'utilisation de la mémoire virtuelle.
- Elle offre entre autres des informations sur les processus, la pagination, la mémoire, les entrées/sorties disques, les interruptions et l'activité du processeur.

```
[root@khayyam]# vmstat -n 5 2
```

procs			memory				swap		io		system			cpu		
r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa	
0	1	94284	5384	3360	94992	10	18	267	54	422	388	13	2	74	12	
0	0	94284	5324	3368	94992	0	0	0	14	315	118	1	0	97	1	
0	0	94284	5324	3368	94992	0	0	0	0	314	114	1	1	98	0	

# vmstat

Les informations affichées par *vmstat* sont en partie :

- Le nombre de processus actifs (r) et dans l'état *uninterruptible sleep* (b)
- Quantité de mémoire virtuelle utilisée (swpd), mémoire physique libre (free) et la quantité de mémoire cache (cache)
- Quantité de mémoire paginé lue et écrite en ko/s (si, so)
- Blocs lus et écrits par seconde (bi, bo)
- Nombre d'interruptions par seconde (in), et le nombre de changement de contextes (cs)
- La répartition du temps CPU en pourcentages pour exécuter les tâches utilisateurs (us), les tâches noyaux (sy) ainsi que le temps inutilisé (id)

# free

## Statistiques d'utilisation des ressources: **free**

-**free** affiche les quantités totales de mémoire et de zone de swap libres et utilisées dans le système, ainsi que la mémoire partagée et les buffers utilisés par le noyau.

SYNOPSIS :            **free** [-b | -k | -m] [-o] [-s nb\_sec ] [-t] [-V]

[root@khayyam]# **free -t**

	total	used	free	shared	buffers	cached
Mem:	126112	122492	3620	0	5420	84024
-/+ buffers/cache:		33048	93064			
Swap:	184708	17460	167248			
Total:	310820	139952	170868			

# top

## Statistiques d'utilisation des ressources: **top**

- La commande **top** affiche une page d'information, périodiquement mise à jour (taper q pour quitter), pour gérer les processus et être informé de la charge de travail du CPU et de l'utilisation mémoire.
- **top** est interactif supportant des commandes permettant de sélectionner les processus à afficher, de définir les champs à visualiser, de trier la liste, de tuer les processus ou de changer les priorités.
- L'équivalent graphique existe sous KDE, et GNOME sous le nom de **Ktop**, kpm, ksysguard ou simplement System Monitor

# sysstat

Sysstat est une suite d'outils de contrôle des ressources

**iostat** Affiche une vue d'ensemble de l'utilisation CPU ainsi que des statistiques d'E/S pour un ou plusieurs disque(s) dur(s).

**mpstat** Affiche des statistiques plus détaillées sur le CPU.

**sadc** est le collecteur de données sur l'activité du système, sadc recueille des informations relatives à l'utilisation des ressources système et les enregistre dans un fichier.

**sar** Génère des rapports à partir des fichiers créés par sadc ; ces rapports sar peuvent être produits de manière interactive ou écrits dans un fichier qui fera l'objet d'une analyse plus approfondie.

# iostat

- iostat dans son utilisation la plus élémentaire fournit une vue d'ensemble des statistiques du CPU et des E/S de disque.

```
[root@Shems ~]# iostat
```

```
Linux 2.6.15-1.1830_FC4 (Shems)      09.02.2006
```

```
cpu-moy: %user  %nice  %sys %iowait  %idle
          9,52   0,43   1,27   6,66   82,12
```

Device:	tps	Blk_lus/s	Blk_écrits/s	Blk_lus
hda	12,44	293,48	64,53	2987715
hdc	0,00	0,03	0,00	292

# mpstat

- Offre des informations de charge par CPU

```
[root@Shems ~]# mpstat
```

```
Linux 2.6.15-1.1830_FC4 (Shems)      09.02.2006
```

```
00:34:56 CPU %user %nice %sys %iowait %irq %soft %idle  intr/s
00:34:56  all  9,38   0,40   1,00   6,26   0,26   0,00  82,70
          447,00
```

# sadc

- sadc recueille des données sur l'utilisation du système et les enregistre ensuite dans un fichier pour une analyse ultérieure.
- Par défaut, les données sont écrites dans des fichiers faisant partie du répertoire `/var/log/sa/`. Ces derniers se nomment `sa<dd>` où `<dd>` correspond au jour actuel.
- sadc est normalement exécutée par le script `sa1`.
- `sa1` est invoqué périodiquement par cron par le biais du fichier `sysstat` qui se trouve dans `/etc/cron.d/`. Le script `sa1` invoque `sadc` pour un seul intervalle d'évaluation durant une seconde. Par défaut, cron exécute `sa1` toutes les 10 minutes et ajoute les données recueillies lors de chaque intervalle au fichier courant nommé `/var/log/sa/sa<dd>`.



# sar

- sar crée des rapports sur l'utilisation du système en fonction des données recueillies par sadc.
- sar est automatiquement exécutée pour traiter les fichiers recueillis automatiquement par sadc. Les fichiers du rapport qui sont écrits dans /var/log/sa/ se nomment sar<dd> où <dd> correspond au jour
- La commande sar est normalement exécutée par le script sa2. Ce dernier est invoqué périodiquement par cron par le biais du fichier sysstat qui se trouve dans /etc/cron.d/. Par défaut, cron exécute sa2 une fois par jour à 23:53, ce qui lui permet de créer un rapport prenant en compte les données recueillies au cours de la journée entière.

# du

- **Utilisation des disques : du.**

**du** affiche la quantité d'espace disque utilisée par chacun des arguments, et pour chaque sous-répertoire des répertoires indiqués en argument. L'espace est mesure en blocs d'un KO

Les options les plus fréquemment utilisées :

- a Afficher les statistiques pour tous les fichiers, pas seulement les répertoires.
- s Afficher seulement le total pour chaque argument.
- x Ignorer les répertoires situés sur un système de fichiers différent de celui de l'argument étudié.

Exemples:

**du -sx /home/\* 2>/dev/null**

**du -sch .**

# df

- **Utilisation des disques : df**

- df** donne l'espace disque disponible sur tous les systèmes de fichiers montés (par défaut la taille est indiquée en nombre de blocs de 1024 octets), ainsi que les points de montage
- df -h** affiche la taille en multiples du Ko (Mo, Go)
- df -T** rajoute dans l'affichage le type de système de fichier (ext2, vfat,...)
- df -t ext2** affiche les informations concernant uniquement les partitions du type précisé (ici ext2)
- df -x ext2** exclut les partitions du type précisé
- df -i** affiche les informations sur l'utilisation des i-noeuds plutôt que les blocs.
- df .** pour connaître l'espace restant dans la partition contenant le répertoire courant

# syslog

- UNIX/LINUX étant un système multi-tâches, il est normal que plusieurs processus se partagent les ressources de la machine.
- Il est nécessaire que chaque démon puisse signaler les problèmes qu'il rencontre ou, simplement, délivrer des messages statistiques de temps à autre.
- Pour remédier à ce problème, les concepteurs d'UNIX nous ont concocté un autre démon, dédié à la gestion des messages des démons (y compris lui-même !) : **syslog**.

# syslog : Principe

- Le principe est simple : au lieu de consigner directement son message dans un fichier, chaque démon va l'envoyer à **syslog** au moyen d'un appel système.
- Ce dernier, en fonction de ses paramètres de configuration, décidera si le message doit être affiché sur la console de la machine, consigné dans un fichier ou renvoyé sur le réseau vers un autre **syslog**.
- Chaque message contient, en plus du texte proprement dit :
  - le nom du programme émetteur et son numéro de processus ;
  - un code de catégorie, identifiant très grossièrement le rôle du programme ;
  - un niveau de sévérité.

# syslog : Liste des catégories

<b>auth :</b>	Messages d'authentification et de sécurité
<b>authpriv :</b>	Messages messages, mais à caractère plus confidentiel
<b>cron :</b>	Messages des planificateurs de tâches (cron et at)
<b>daemon :</b>	Messages générés par les autres daemons systèmes.
<b>ftp :</b>	Messages du serveur ftp
<b>kern :</b>	Messages du noyau (en provenance de klogd)
<b>lpr :</b>	Messages du serveur d'impression
<b>mail :</b>	Messages du système de messagerie
<b>news :</b>	Messages du serveur de newsgroups
<b>syslog :</b>	Messages internes de syslog
<b>local0 à local7 :</b>	Services définis localement par l'administrateur système.
<b>user :</b>	Messages en provenance des utilisateurs.
<b>uucp :</b>	Messages générés par le système <i>UUCP</i>

# syslog : Liste des sévérités

Les sévérités sont classées de la plus grave à la plus bénigne :

- emerg** : (emergency) : situation d'urgence rendant le système inutilisable ou risque de le devenir à très court terme.
- alert** : (alerte) : Message alertant l'administrateur système qu'une action de sa part est requise.
- crit** : (critique) : situation critique
- err** : (erreur) : condition d'erreur
- warning** : simple avertissement
- notice** : message d'information
- info** : message d'information à caractère moins important que « notice »
- debug** : message de debuggage

# syslog : Configuration

Chaque ligne du fichier de configuration **/etc/syslog.conf** est une règle comportant deux champs (séparés par des espaces ou tabs):

sélecteur	action
-----------	--------

Le **sélecteur** détermine quels seront les messages concernés.

Le deuxième champ concerne **l'action** à effectuer lorsqu'un message correspondant doit être traité.

Le champ **sélecteur** a la forme suivante : **service.sévérité**

Exemple:

<b>ftp.err</b>	<b>/var/log/ftp-err</b>
----------------	-------------------------

consigne les messages d'erreur du serveur ftp dans le fichier **/var/log/ftp-err** .



# syslog : Configuration

En fait la règle :

**ftp.err**

**/var/log/ftp-err**

consignera les messages du serveur ftp de sévérité « err » ou supérieure, donc aussi les messages « crit », « alert » et « emerg ».

Pour remédier aux problèmes :

- préfixer la sévérité d'un signe égal (=) ne prend en compte que les messages indiqués
- préfixer la sévérité d'un point d'exclamation (!) permet d'exclure les messages de cette sévérité ou supérieure.
- préfixer la sévérité d'un != ne permet d'exclure que la sévérité indiquée.

# syslog : Configuration

Exemple 2 :

<b>ftp.crit</b>	<b>/var/log/ftp-critique</b>
<b>ftp.=err</b>	<b>/var/log/ftp-err</b>
<b>ftp.!err</b>	<b>/var/log/ftp</b>

Indique que les messages de sévérité « crit » ou supérieure seront consignés dans un premier fichier, ceux de sévérité « err » uniquement dans un deuxième fichier et ceux de sévérité moins élevée, dans le troisième fichier.

Pour regrouper des logs de plusieurs démons :

<b>ftp.err;mail.err</b>	<b>/var/log/ftp-mail-err</b>
<b>ftp,mail.err</b>	<b>/var/log/ftp-mail-err</b>

# syslog : Configuration

Vous l'aurez compris :

**\*.err**                      **/var/log/all-err**

consigne tous les messages d'erreur (et sup) dans un seul fichier.

De la même manière, il est possible d'utiliser l'étoile en guise de sévérité, pour désigner tous les messages d'un démon :

**ftp.\***                      **/var/log/ftp**

Enfin, la priorité **none** signifie qu'aucun message ne doit être journalisé pour le service correspondant.

**\*.=debug ; mail,news.none**    indique que l'on journalise tous les messages de débogage, sauf ceux venant des services mail et news.

# syslog : Actions

## Fichier :

L'action la plus courante consiste simplement à ajouter le message à la fin d'un fichier. Pour cela, il suffit de mentionner le nom de fichier (le chemin complet) comme seconde partie de la règle.

```
kern.*      /var/log/kernel.lo
```

Par défaut, le fichier est écrit immédiatement sur disque après l'ajout de chaque message. Il est possible de délayer cette écriture en préfixant le nom du fichier d'un caractère '-'.

## Console

Dans le cas où le fichier mentionné serait une console virtuelle (/dev/console, /dev/tty0, etc), les messages seront écrits à l'écran directement.

# syslog : Actions

## Machine distante

syslogd permet d'envoyer les messages sur une autre machine qui utilise également syslogd. Il suffit pour cela de donner le nom de la machine distante prefixée par un caractère @.

Pour que syslogd accepte de recevoir des messages d'une machine distante, on doit le lancer avec le paramètre -r.

L'option -h autorise la retransmission d'un message reçu d'une autre machine.

Les paramètres -l et -s peuvent également être utilisés pour spécifier (respectivement) une liste de noms de machines ou de noms de domaines pour lesquels on accepte les messages.

**Exemple:**

```
*.emerg;kern.crit @alharith
```

# syslog : Actions

## Liste d'utilisateurs

Dans le cas où un message critique est émis, il est important que l'administrateur ou les autres utilisateurs soient avertis immédiatement. syslogd permet d'avertir une liste d'utilisateurs s'ils sont connectés sur la machine via la commande *write (1)*.

Pour cela, il suffit de donner la liste des logins des utilisateurs concernés.

```
*.crit;*.alert root,kmaster,ali
```

Si l'on spécifie une étoile comme action, le message sera envoyé à tous les utilisateurs via la commande *wall (1)* :

```
*.emerg      *
```

# syslog : Actions

## Tubes nommés

Une dernière possibilité consiste à envoyer les messages dans un tube nommé (fifo). Pour cela il faut préfixer le nom du tube par '|'. Ceci sert principalement à fournir des sorties syslog à un autre programme pour traitement.

### Exemple:

On crée le tube :

```
mkfifo /dev/xconsole
```

Dans /etc/syslog.conf On ajoute :

```
*.* | /dev/xconsole
```

On relance syslogd :

```
killall -HUP syslogd
```

puis on lance l'application xconsole: **xconsole -file /dev/xconsole**

A partir de là, on doit voir tous les messages système s'afficher dans la fenêtre au fur et à mesure qu'ils surviennent.

# syslog : test de configuration

Une fois votre configuration optimale, vous pouvez la tester en envoyant des messages bidons à **syslogd** via l'utilitaire **logger**

Exemple :

```
# logger -p mail.info "Message d'info mail envoyé par logger"
```

syslogd reçoit le message indiqué comme s'il provient du service mail avec une sévérité info

Logger a d'autres options plus au moins intéressantes

```
# man logger
```

Pour plus de détail



# Nettoyage des logs

L'utilitaire **logrotate** se charge d'archiver, de compresser, de supprimer et/ou d'envoyer par mail les fichiers logs à intervalles réguliers ou lorsque la taille dépasse une certaine limite .

Le fichier de configuration de logrotate (**/etc/logrotate.conf**) décrit les actions à effectuer pour chaque fichier log ainsi la périodicité ou la taille limite d'archivage

Les fichiers sont généralement archivés en appliquant une rotation. Par exemple si on décide d'archiver le fichier **/var/log/messages** par semaine avec une rotation 2, on aura dans notre répertoire **/var/log** trois fichiers :

<b>messages</b>	contient les messages de la semaine courante
<b>messages.1</b>	contient les messages de la semaine d'avant
<b>messages.2</b>	contient les messages datés il y'a deux semaines

# Nettoyage des logs

- Les archives peuvent être compressés (gzip par défaut)
- À chaque opération les fichiers sortant du jeu de rotation appliqué sont supprimés ou envoyés par mail
- **logrotate** est généralement lancé par le planificateur de tâche **cron** une fois par jour,
- **logrotate** reçoit son (ou ses) fichier(s) de configuration en argument. Généralement, il est évoqué avec **/etc/logrotate.conf**
- **/etc/logrotate.conf** contient les options globales d'archivage qui affectent l'ensemble des fichiers logs.
- Les services produisant des logs ont généralement leurs propres fichiers de configuration placés sous le répertoire **/etc/logrotate.d**
- L'instruction

**include /etc/logrotate.d**

Mise dans **/etc/logrotate.conf** permet de signaler leur présence