Système et Réseau Linux

Red Hat Enterprise Linux
Community Enterprise Operating System

Sébastien ROHAUT

Table des matières

1 Installation....6
1.1 Présentation....6
1.2 Contraintes....6
1.2.1 matériel....6
1.2.2 Paramétrage réseau....7
1.2.3 Plan de partitionnement....7
1.2.4 Consoles virtuelles....7
1.3 Etapes d'installation....7
1.3.1 boot....7
1.3.2 Support d'installation....8
1.3.3 Langue, clavier, souris....8
1.3.4 Partitionnement....8
1.3.5 Chargeur d'amorçage....9
1.3.6 Configuration réseau et firewall....9
1.3.7 Autres langues, fuseaux....9
1.3.8 Mot de passe root....9
1.3.9 Sélection des groupes de paquetages....9
1.3.10 Premier démarrage....9
2 Processus de démarrage....10
2.1 Le bios....10
2.2 Le chargeur de démarrage....10
2.3 GRUB....10
2.3.1 Configuration....10
2.3.2 Démarrage et édition....11
2.4 Initialisation du noyau....12
2.5 init....12
2.5.1 Niveaux d'exécution....12
2.5.2 rc.sysinit....13
2.5.3 rc....13
2.5.4 Niveaux d'exécution System V....13
2.5.5 rc.local....14
2.5.6 Contrôle des services....14
2.5.7 consoles virtuelles....15
3 Le système de fichiers....16
3.1 Gestion des partitions....16
3.2 Créer les systèmes de fichiers....16
3.2.1 mkfs....16
3.2.2 ext2 et ext3....17
3.2.3 Reiserfs....17
3.3 Monter les systèmes de fichiers....17
3.3.1 mount....17
3.3.2 /etc/fstab....18
3.4 ACL....19
4 Redhat Package Manager....20
4.1 Notion de package....20
4.2 Le gestionnaire RPM....20

4.3 Installation, mise à jour et suppression....21
4.4 Cas du noyau....22
4.5 Requêtes RPM....22
4.6 Vérification des packages....23
4.7 Les dépendances....24
4.8 Mises à jour automatisées....24
5 Administration des utilisateurs....25
5.1 Ajout....25
5.2 modification....25
5.3 Suppression....25
5.4 Politique de sécurité....26
5.5 Les groupes....26
5.6 Groupes privés et setgid....26
5.7 L'environnement utilisateur....27
5.8 PAM....27
5.9 Les quotas disques....29
5.9.1 Définitions....29
5.9.2 Mise en place....30
5.10 Client NIS....31
6 Outils d'administration....33
6.1 Le réseau....33
6.2 L'impression....33
6.2.1 Principe....33
6.2.2 System V....33
6.2.3 BSD....34
6.2.4 CUPS....34
6.3 Automatisation avec cron....35
6.3.1 Formalisme....35
6.3.2 crontab système....36
6.4 Journal système....36
7 Services et modules noyau....38
7.1 Présentation....38
7.2 Gestion des modules....38
7.3 /proc et /sys....40
8 Partitionnement avancé RAID....42
8.1 Définitions....42
8.2 Précautions et considérations d'usage....42
8.2.1 Disque de secours....42
8.2.2 Disque défectueux....43
8.2.3 boot....43
8.2.4 Swap....43
8.2.5 Périphériques....43
8.2.6 IDE....43
8.2.7 Hot Swap....44
8.3 RAID avec mdadm....44
8.3.1 Préparation....44
8.3.2 Création....44
8.3.2.1 RAID-0....44
8.3.2.2 RAID-1....45
8.3.2.3 RAID-0+1....45

8.3.2.4 RAID 5 ... 45
8.3.3 Sauver la configuration ... 45
8.3.4 Etat du RAID ... 45
8.3.5 Simuler une panne ... 46
8.3.6 Remplacer un disque ... 47
8.3.7 Arrêt / Relance manuels ... 47
9 TCP/IP ... 48
9.1 Bases ... 48
9.2 adressage ... 48
9.2.1 classes ... 48
9.2.2 Sous-réseaux ... 49
9.2.3 Routage ... 50
9.3 Configuration ... 50
9.3.1 Outils de RHEL ... 50
9.3.2 Interfaces réseaux ... 50
9.3.3 Paramètres généraux ... 52
9.3.4 Routage ... 52
9.4 /etc/resolv.conf ... 52
9.5 /etc/hosts et /etc/networks ... 53
10 Services réseaux xinetd ... 53
10.1 Présentation ... 53
10.2 Configuration ... 53
10.3 Démarrage et arrêt des services ... 54
11 OpenSSH ... 56
11.1 Présentation ... 56
11.2 Configuration ... 56
11.3 Utilisation ... 56
12 Monter un serveur DHCP ... 57
12.1 Présentation ... 57
12.2 Serveur dhcpd ... 57
12.2.1 Informations de base ... 57
12.3 Côté client ... 58
13 Serveur DNS ... 59
13.1 Présentation ... 59
13.2 Lancement ... 59
13.3 Configuration de Bind ... 60
13.3.1 Configuration générale ... 60
13.3.2 Section globale ... 60
13.3.3 Section de zones ... 61
13.3.3.1 Zone de résolution ... 61
13.3.3.2 Zone de résolution inverse ... 61
13.3.3.3 Exemple ... 62
13.3.3.4 Zones spéciales ... 62
13.3.4 Fichiers de zones ... 63
13.3.4.1 Définitions ... 63
13.3.4.2 Zone ... 63
13.3.4.3 Zone de résolution inverse ... 65
13.3.5 Round-robin ... 66
13.3.6 Diagnostic des problèmes ... 67
13.3.7 Interrogation dig et host ... 67

14 Courrier électronique....69
14.1 Principe....69
14.2 Postfix....70
14.3 POP et IMAP....71
15 Service HTTP Apache....72
15.1 Présentation....72
15.2 Arrêt/Relance....72
15.3 Configuration....72
15.4 Directives générales....72
15.5 Gestion des performances....73
15.6 Les répertoires, alias et emplacements....74
15.7 Hôtes virtuels....74
16 Partage de fichiers....76
16.1 NFS....76
16.1.1 Lancement....76
16.1.2 Partage côté serveur....76
16.1.3 Montage côté client....77
16.2 FTP....78
16.3 Partages Windows avec Samba....79
16.3.1 Présentation....79
16.3.2 Configuration....79
16.3.2.1 Outils graphiques....79
16.3.2.2 structure....79
16.3.2.3 Partage de fichiers....80
16.3.2.4 Partage des imprimantes....81
16.3.2.5 Méthodes d'authentification....81
16.3.2.6 Correspondance des noms et mots de passe....81
16.3.3 Clients SAMBA....82
16.3.3.1 En ligne....82
16.3.3.2 Montage....82
17 Bases de sécurité services et réseau....83
17.1 Les tcp_wrappers....83
17.2 Netfilter....84
17.2.1 Présentation....84
17.2.2 Vie d'un paquet....84
17.2.3 Principe des règles....85
17.2.4 Cibles de règles....85
17.2.5 Premier exemple....86
17.2.6 Opérations de base....86
17.2.7 Critères de correspondance....86
17.2.7.1 Général....86
17.2.7.2 TCP, UDP et ICMP....87
17.2.7.3 Arguments des critères....87
17.2.8 Sauver ses réglages....87

1 Installation

1.1 Présentation

RHEL (*Red Hat Enterprise Linux*) est une distribution du système d'exploitation Linux fortement orientée serveur. Diffusée en plusieurs versions (poste de travail, ES Enterprise Server, AS Advanced Server) et sur plusieurs architectures (x86, x86_64, Itanium, Alpha, s390) RHEL est officiellement supportée par de nombreux éditeurs (Oracle par exemple) et constructeurs (HP par exemple). Les seules différences techniques entre une version ES et AS sont une modification du fichier **/etc/issue** (qu'on peut éditer) deux ou trois paramètres du noyau (que l'on peut de toute façon recompiler), quelques packages supplémentaires (dont les sources sont disponibles), et le support des architectures IBM. Par contre, le support technique n'est absolument pas le même.

RHEL est constituée de produits entièrement libres et ses sources sont entièrement disponibles. Vous pouvez récupérer une distribution RHEL librement cependant l'accès aux mises à jour de Red Hat n'est possible que pour les clients de Red Hat disposant d'un numéro de licence.

RHEL étant libre et ses sources accessibles, d'autres utilisateurs ont décidé de reconstruire une version ne souffrant pas de cette limitation depuis les sources. **CentOS** (*Community Enterprise Operating System*) est l'une de ces distributions :

- basée sur RHEL 4 release 2
- Construite sur une version AS
- Mises à jour disponibles généralement 72 heures après celles de RH
- Tous les copyrights et logos Red Hat ont été supprimés

Tous les outils, tous les logiciels sont identiques à ceux de Red Hat. Toute la configuration est 100% identique. Seul le RHN (Redhat Network) est désactivé par défaut.

1.2 Contraintes

1.2.1 matériel

La RHEL/CentOS s'installe sur une majorité des architectures matérielles 32 ou 64 bits, Sparc, ou s390. Comme il s'agit d'un outil serveur, la plupart des cartes réseaux et disques (et leurs bus) sont supportés. Les RHEL/CentOS sont basées sur des versions stabilisées et professionnelles des Fedora. Cette version est basée sur une Fedora Core 3. Le noyau Linux est un 2.6.9.

- mémoire : 128 Mo minimum, 256 ou plus conseillé (si graphique)
- disque : 4 Go ou +
- Carte réseau : toute carte réseau ethernet 10/100/1000. Wifi : fonctionne mais compilation et paramétrage manuels.
- Carte graphique : sans importance si pas d'utilisation de X, sinon simple carte SVGA
- moniteur : idem (1024x768 sinon)
- carte son : modèles courants supportés
- lecteur cd/dvd : conseillé mais pas vital pour les installations réseaux

1.2.2 Paramétrage réseau

Il est important de définir par avance le mode de fonctionnement du réseau. Quel est le sous-réseau à utiliser ? L'attribution est-elle statique ou dynamique ? Quel est le nom du domaine ? Quel est le nom de l'hôte ?

Dans le cas d'un démarrage via PXE (bootp/tftp), un serveur DHCP déjà actif fournira l'ensemble des données.

- Domaine : esgi (xxx.esgi)
- Protocole : DHCP (plage dynamique : 172.16.17.100->199)
- Plage IP fixe : 172.16.17.11->99
- Masque de sous-réseau : 255.255.255.0 (sous-réseau de classe C)

1.2.3 Plan de partitionnement

- Prévoir 5 Go pour la racine /
- Prévoir 150 Mo pour /boot (s'il est utilisé)
- Prévoir 500 Mo pour le /home
- Prévoir une à deux fois la taille de la mémoire pour le swap (1 Go)
- prévoir 400 Mo pour /var
- Ne pas utiliser l'ensemble de l'espace car nous allons tester le RAID.

1.2.4 Consoles virtuelles

Des consoles virtuelles sont disponibles durant l'installation (Alt+Ctrl+Fx) :

- 1 : installation texte
- 2 : invite du shell
- 3 : journal de l'installation
- 4 : messages du système
- 5 : autres messages
- 7 : installation graphique

1.3 Etapes d'installation

1.3.1 boot

RHEL/Centos s'installe en mode graphique ou en mode texte. Le programme d'installation s'appelle Anaconda. On appuie sur Entrée. On peut passer en mode texte en tapant au démarrage

```
linux text
```

Les machines de la salle 17 n'arrivent pas à démarrer l'installation en mode graphique. L'installation se fera par défaut en mode texte. De même, elles ne démarrent pas en mode PXE (raison inconnue, probablement un firmware absent sur la carte réseau). Aussi, le démarrage se fera depuis un CD de boot qui ne contient que le nécessaire pour démarrer sur le réseau.

1.3.2 Support d'installation

L'installation peut se faire via un support de type CD/DVD, un disque dur, NFS, FTP, HTTP. Dans tous les cas, une image de démarrage est nécessaire. Celle-ci est présente sur les cds/dvds d'installation. Elle peut être sur des disquettes, des clés USB, et sur le réseau via PXE (Bootp/tftp).

Lors de l'installation, on choisira une installation de type réseau en HTTP ou FTP. La première chose qui sera demandée dans ce cas sont les réglages TCP/IP. On choisira dans un premier temps DHCP.

- Le nom du site HTTP est : **`ftp.esgi`**
- Le répertoire CentOS est : **`centos/4.2/os/i386`**
- Si FTP : FTP en anonyme.

1.3.3 Langue, clavier, souris

On choisit la langue et le clavier français (Français Latin1). Si le modèle de souris est demandé, choisissez un modèle qui correspond au votre.

1.3.4 Partitionnement

L'installateur va vous laisser le choix entre un partitionnement automatique et un partitionnement manuel avec un outil appelé **`Disk Druid`**. Pour appliquer notre configuration ci-dessus, il faut passer en mode manuel.

Arrivé sous Disk Druid, si des partitions existent déjà supprimez-les : allez sur le disque (**`/dev/hda`**) et faites « **`Supprimer`** » (ou sélectionnez les partitions une par une).

La nomenclature des disques est assez simple

- **`hda`** : premier disque IDE0, maître
- **`hdb`** : second disque IDE0, esclave
- **`hdc`** : premier disque IDE1, maître
- **`hdd`** : second disque IDE1, esclave
- **`sda`** : premier disque SCSI / SATA de la chaîne
- **`sdb`** : second disque SCSI / SATA de la chaîne
- **`hda1`** : première partition primaire
- **`hda2`** : seconde partition primaire
- **`hda4`** : généralement partition logique (qui contient n autres partitions)
- **`hda5`** : première partition logique
- **`hda6`** : seconde partition logique
- ...

Ajoutez les partitions avec « **`Ajouter`** ». Dans la boîte de dialogue d'ajout, choisissez le point de montage (**`/`**, **`/boot`**, **`/home`**, **`/var`**, etc). Un **`swap`** n'a pas de point de montage. Sauf pour le **`swap`**, le type de système de fichiers doit être « **`ext3`** ». La taille est exprimée en Mo.

1.3.5 Chargeur d'amorçage

Laissez les options par défaut.

1.3.6 Configuration réseau et firewall

Laissez la configuration par défaut **`DHCP`** pour le réseau. Désactivez **`SELinux`** et n'activez pas le **`pare-feu`**.

1.3.7 Autres langues, fuseaux

Laissez le français (éventuellement, cochez l'anglais). Prenez **`Europe/Paris`** comme fuseau horaire.

1.3.8 Mot de passe root

Dans le cadre du cours, choisissez le même mot de passe : **`adm_esgi`**. Bien entendu ne faites pas ça chez vous.

1.3.9 Sélection des groupes de paquetages

Vous pouvez laisser le choix par défaut ou choisir les composants que vous souhaitez (voire tous). Ce n'est pas important : nous pourrons en réinstaller par la suite. A la validation l'installation commence.

1.3.10 Premier démarrage

Au premier démarrage, RHEL/CentOS démarre un assistant appelé « **`Agent de paramétrage`** ». Cet assistant permet de régler l'affichage, de créer des utilisateurs, de mettre à l'heure, d'ajouter des sources de logiciels et d'effectuer les mises à jour.

Pour les machines de la salle 17, il va falloir modifier le fichier `/etc/X11/Xorg.conf`. Recherchez la ligne « `Driver` » et placez « `vesa` » comme driver.

2 Processus de démarrage

2.1 Le bios

Le **Bios** : *Basic Input Output System*, est l'interface entre le matériel et le logiciel à un niveau très basique. Il fournit l'ensemble des instructions de base utilisé par le système d'exploitation. Il fournit le niveau d'interface le plus bas aux pilotes et périphériques.

Le Bios effectue un auto-test de l'allumage (POST) puis recherche les périphériques notamment ceux utilisés pour démarrer. Les informations sur le matériel sont stockées de manière permanente dans une petite mémoire CMOS alimentée par une batterie. A la fin du processus, le périphérique de démarrage est sélectionné.

Le bios lit et exécute le premier secteur physique du média de démarrage. Il s'agit généralement des 512 premiers octets du disque dur.

2.2 Le chargeur de démarrage

Le Bios charge le chargeur de programme initial (*Initial Program Loader IPL*) à partir des premiers 512 octets du support de démarrage. Sur Linux, le chargeur est décomposé en deux parties. Le chargeur initial des 512 octets ne contient pas assez de code pour proposer des menus et lancer le chargement d'un système d'exploitation. Il charge la seconde phase, basée sur un fichier de configuration.

La seconde phase fournit une interface pour lancer un système d'exploitation parmi un choix donnée. On peut aussi de là passer des paramètres au noyau Linux et au processus **init**.

Le chargeur par défaut sur la plupart des distributions Linux s'appelle **GRUB** (*Grand Unified Bootloader*). Il est hautement paramétrable notamment en acceptant une protection par mot de passe crypté, un interpréteur de commandes ou encore des graphiques. Il est basé sur un fichier de configuration texte et il n'y a pas besoin de réinstaller Grub à chaque modification.

2.3 GRUB

2.3.1 Configuration

La configuration de **GRUB** est dans **/boot/grub/grub.conf** ou **/boot/grub/menu.lst** (l'un est un lien sur l'autre). GRUB peut s'installer sur un **MBR** (*Master Boot Record*, les 512 premiers octets d'un disque) ou un **PBR** (*Partition Boot Record*, les 512 premiers octets d'une partition).

Pour installer ou réinstaller GRUB en cas de MBR corrompu, par exemple sur /dev/hda on utilise la commande « **grub-install** » :

```
/sbin/grub-install /dev/hda
```

Voici un exemple de configuration.

```
timeout=10
default=0
title Red Hat
        root (hd0,0)
        kernel /vmlinuz-2.6.12-15 ro root=LABEL=/
        initrd /initrd-2.6.12-15.img
title Windows XP
        rootnoverify (hd0,1)
```

```
chainloader +1
```

- **timeout** : secondes avant le démarrage par défaut
- **default** : image/os/partition par défaut (stanza 0=1er title, 1=2eme title, etc)
- **title** : étiquette pour le stanza
- **root** : tous les accès fichiers spécifiés dessous le seront à partir de cette partition (cf signification plus bas). Ici hd0,0 représente la 1ere partition du 1er disque détecté par le BIOS. C'est la partition /boot.
- **kernel** : image du noyau (boot/...) auquel on passe des paramètres (read only, partition root).
- **initrd** : initial ramdisk. Le noyau va charger ce fichier comme disque en mémoire pour y trouver une configuration et des pilotes initiaux.
- **rootnoverify** : La racine spécifiée, à ne pas monter par grub (il ne supporte pas NTFS).
- **chainloader +1** : Démarrer le premier secteur de la racine spécifiée ci-dessus.

Voici la signification des noms de périphériques sous GRUB.

- **(fd0)** : Premier lecteur de disquettes détecté par le BIOS (/dev/fd0 sous Linux)
- **(hd0,0)** : Première partition sur le premier disque dur détecté par le BIOS que ce soit IDE ou SCSI (/dev/hda1 ou /dev/sda1 suivant le cas)
- **(hd1,4)** : Cinquième partition sur le second disque dur détecté par le BIOS (/dev/hdb5 ou /dev/sda5)

2.3.2 Démarrage et édition

Au démarrage de **GRUB**, un menu s'affiche. Il peut être graphique ou texte, selon la configuration. On peut choisir une image de démarrage avec les flèches de direction. En appuyant sur <entrée> on démarre sur l'image sélectionnée.

On peut éditer les menus en direct pour modifier par exemple les paramètres passés au noyau Linux ou **init**. Dans ce cas, allez sur une entrée de menu et appuyez sur la touche **<e>** (edit). Ici, toutes les lignes du ztanza sont affichées. On peut appuyer sur :

- **e** : pour éditer la ligne (la compléter)
- **d** : pour supprimer la ligne
- **o** : pour ajouter une ligne
- **b** : pour démarrer l'image (booter).

Par exemple, pour démarrer en mode urgence (emergency) :

- Aller sur la ligne Linux ou Redhat et appuyer sur <e>.
- Aller sur la ligne kernel et appuyer sur <e>.
- A la fin de la ligne rajouter <1> et appuyer sur <entrée>. (essayer avec init=/bin/sh !)
- Appuyer sur <b>.

On peut aussi accéder à un interpréteur de commandes en appuyant sur <echap>. Attention seules les commandes GRUB sont reconnues.

2.4 Initialisation du noyau

Au chargement du noyau une multitude d'informations défile sur l'écran. On ne peut pas figer ces informations à l'instant même. Par contre juste après le passage à l'étape suivante (init) toutes les traces du noyau sont placées dans le fichier **/var/log/dmesg**.

- Le matériel est détecté et initialisé
- **initrd** est chargé, les modules présents éventuellement chargés
- Le noyau monte le système de fichiers racine en lecture seule
- Le premier processus est lancé (normalement **init**).

2.5 init

Le processus **init** est le père de tous les processus. Il a toujours le PID 1. Sa configuration est présente dans le fichier **/etc/inittab**. Si ce fichier est corrompu et inutilisable, il faudra démarrer en mode single (S, s, 1, Single) et le réparer. C'est un fichier central du système d'exploitation.

2.5.1 Niveaux d'exécution

Un niveau d'exécution (**runlevel**) est un état dans lequel se trouve Unix/Linux. Cet état est contrôlé par **init.** Chaque état dispose de sa configuration (soit par **inittab**, soit par les initscripts). Un niveau d'exécution peut par exemple être utilisé pour lancer Unix en mono-utilisateur, en multi-utilisateurs, avec ou sans réseau, avec ou sans mode graphique. Ces niveaux sont généralement définis comme ceci.

Niveau	*Effet*
0	Halte : stoppe le système d'exploitation / la machine
1,S	Mode mono-utilisateur simple utilisé pour la maintenance. Console
2	Multi-utilisateurs, sans réseau, console.
3	Multi-utilisateurs, avec réseau, mode console.
4	Idem par défaut que le 3, mais configurable par l'utilisateur (admin)
5	Idem 3 mais lancement de X Window
6	Redémarrage (reboot) de la machine.

Les niveau 7 à 9 sont valides mais pas utilisés par défaut. On choisit le niveau d'exécution par défaut dans **/etc/inittab** par la ligne **initdefault**.

```
Id:5:initdefault:
```

On remplace 5 par le niveau souhaité.

Note : NE PAS METTRE 0 OU 6 PAR DEFAUT.

On peut aussi changer de niveau à la volée après le démarrage de la machine avec la commande **/sbin/init** ou **/sbin/telinit**.

```
telinit 3
```

On peut voir le niveau actuel avec la commande **/sbin/runlevel**.

2.5.2 rc.sysinit

Quelque soit le niveau d'exécution choisi, le script **/etc/rc.d/rc.sysinit** est toujours exécuté. Ses tâches sont de :

- Configurer les paramètres du noyau présents dans **/etc/sysctl.conf** (ex : IP Forwarding).
- Configurer l'horloge du système.
- Charger les tables de caractères du clavier.
- Activer les partitions d'échange SWAP.
- Définir le nom d'hôte.
- Contrôler et monter le système de fichiers racine (en lecture-écriture cette fois).
- Ajouter les périphériques RAID.
- Activer les quotas de disque.
- Contrôler et monter les autres systèmes de fichiers.
- Nettoyer les verrous (stale locks) et fichiers PID en cas d'arrêt brusque.

Il est possible de passer en mode interactif. Au début du script, on vous demande de taper éventuellement sur la lettre **<i>** puis de répondre par oui ou par non aux différentes actions.

2.5.3 rc

Le script **/etc/rc.d/rc** prend comme paramètre le niveau d'exécution selon la ligne **initdefault** de /**etc/inittab** ou de la comme **init/telinit**. Le script **rc** initialise le niveau d'exécution voulu et est responsable du démarrage et de l'arrêt des services associés quand le niveau d'exécution change.

```
id:3:initdefault:
l0:0:wait:/etc/rc.d/rc 0
l1:1:wait:/etc/rc.d/rc 1
l1:2:wait:/etc/rc.d/rc 2
l1:3:wait:/etc/rc.d/rc 3
l1:4:wait:/etc/rc.d/rc 4
l1:5:wait:/etc/rc.d/rc 5
l1:6:wait:/etc/rc.d/rc 6
```

2.5.4 Niveaux d'exécution System V

- Le niveau d'exécution définit les services à démarrer.
- Tous les scripts des services sont dans **/etc/rc.d/init.d**.
- Chaque niveau d'exécution correspond à un répertoire **/etc/rc.d/rc*X*.d** où X est le niveau d'exécution.
- Les liens symboliques dans les répertoires ***rcX.d*** appellent les scripts présents dans **init.d** avec les arguments **start** et **stop**.
- Le préfixe du nom de chaque lien définit son ordre de lancement ou son ordre d'arrêt : **[SK]XXservice** :
    - **S** : start
    - **K** : kill (stop)

- **XX** : ordre numérique de démarrage ou d'arrêt. (00=premier, 99=dernier).
- **service** : nom du service
- ex : **S10network**.

```
[admprd@s64p17bib27 admprd]$ ls -l /etc/rc.d/init.d
total 153
-rwxr-xr-x    1 root     root          934 jun 25  2001 anacron
-rwxr-xr-x    1 root     root         1458 jui 24  2001 apmd
-rwxr-xr-x    1 root     root         1041 mai 11  2004 arpwatch
-rwxr-xr-x    1 root     root         1176 jun 28  2004 atd
...

[admprd@s64p17bib27 admprd]$ ls -l /etc/rc.d/rc3.d
total 0
lrwxrwxrwx    1 root     root           15 jun 27  2005 K15httpd -> ../init.d/httpd
lrwxrwxrwx    1 root     root           13 jun 27  2005 K20nfs -> ../init.d/nfs
lrwxrwxrwx    1 root     root           16 jun 27  2005 K20rstatd -> ../init.d/rstatd
lrwxrwxrwx    1 root     root           17 jun 27  2005 K20rusersd -> ../init.d/rusersd
lrwxrwxrwx    1 root     root           16 jun 27  2005 K20rwalld -> ../init.d/rwalld
lrwxrwxrwx    1 root     root           15 jun 27  2005 K20rwhod -> ../init.d/rwhod
lrwxrwxrwx    1 root     root           15 jun 27  2005 K25squid -> ../init.d/squid
```

2.5.5 rc.local

Le script **/etc/rc.d/rc.local** est exécuté après les scripts du niveau d'exécution. C'est le script d'initialisation finale de System V et l'administrateur peut y placer les commandes qu'il souhaite. Il est exécuté après chaque changement de niveau d'exécution.

2.5.6 Contrôle des services

Pour contrôler la configuration des services de System V lancés par init, il est déconseillé de tout faire à la main mais d'utiliser les outils du système.

- **ntsysv** : en mode console interactive (ncurses, avec menus)
- **chkconfig** : en mode ligne de commande qui fonctionne avec les scripts. Gère aussi directement les services xinetd.
- **serviceconf** (propre à redhat) en mode graphique
- **service** : arrête ou démarre immédiatement un service

/sbin/chkconfig est très pratique pour configurer les services notamment parce qu'il sait gérer tant les services System V que les services xinetd (cf réseau).

- **--list** : liste de l'ensemble de la configuration
- **--list service** : la configuration d'un service donné
- **--add service** : ajoute le service indiqué dans la configuration System V
- **--del service** : supprime le service de la configuration System V
- **--level xxx service on/off** : active ou désactive le service pour les niveaux d'exécution indiqués. Ex : **--level 345 smb on**

```
[admprd@s64p17bib27 admprd]$ /sbin/chkconfig --list
rwhod           0:arrêt  1:arrêt  2:arrêt  3:arrêt  4:arrêt  5:arrêt  6:arrêt
atd             0:arrêt  1:arrêt  2:arrêt  3:marche 4:marche 5:marche 6:arrêt
snmpd           0:arrêt  1:arrêt  2:marche 3:marche 4:marche 5:marche 6:arrêt
ntpd            0:arrêt  1:arrêt  2:marche 3:marche 4:marche 5:marche 6:arrêt
keytable        0:arrêt  1:marche 2:marche 3:marche 4:marche 5:marche 6:arrêt
syslog          0:arrêt  1:arrêt  2:marche 3:marche 4:marche 5:marche 6:arrêt
...
```

```
[admprd@s64p17bib27 admprd]$ /sbin/chkconfig --list smb
smb             0:arrêt 1:arrêt 2:arrêt 3:arrêt 4:arrêt 5:arrêt 6:arrêt

[admprd@s64p17bib27 admprd]$ /sbin/chkconfig --level 35 smb on

[admprd@s64p17bib27 admprd]$ /sbin/chkconfig --add httpd
```

Note : Sauf pour xinetd, chkconfig ne lance aucun service. Il ne fait que paramétrer les niveaux d'exécution. Pour lancer un service, on utilisera le script associé où la commande **service**.

La commande **/sbin/service** permet de gérer les services immédiatement. Elle transmet au service les paramètres **start**, **stop**, **restart**, **reload**, **condrestart** et **status.**

```
$ /sbin/service smb start
```

2.5.7 consoles virtuelles

Les consoles virtuelles permettent d'obtenir des terminaux virtuels sur une machine. Elles sont définies dans **/etc/inittab**. Elles sont disponibles via les périphériques **/dev/ttyn**.

```
# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

- On passe d'une console à l'autre avec la séquence de touches **<Alt-Fn>** (ex : Alt-F2) depuis la console ou **<Control-Alt-Fn>** depuis X Window.
- On peut aussi utiliser les touches **<Alt-Fleche_Droite>** et **<Alt-Fleche_Gauche>**.
- **/dev/ttyn** représente la console virtuelle n.
- **/dev/tty0** représente la console courante.
- Comme il y a 12 touches Fn, on peut avoir par défaut 12 terminaux virtuels.
- Cependant 6 sont activés par défaut.
- X Window se lance par défaut sur la première console disponible, généralement la 7.
- On peut router des infos vers une console : `tail -f /var/log/messages > /dev/tty12 &`

3 Le système de fichiers

3.1 Gestion des partitions

Les outils **fdisk**, **cfdisk**, **sfdisk** ou encore **parted** permettent de manipuler les partitions, sans compter les outils graphiques disponibles durant l'installation ou dans les panneaux de configuration.

- **fdisk** est le plus ancien et le plus utilisé des outils de partitionnement. Il n'a aucun rapport avec le fdisk de Microsoft. Il est à base de menus. Pour créer une partition, on appuie sur <n>, on choisit primaire ou logique, la taille et le type de partition. On sauve ensuite en appuyant sur <w>. Il est disponible avec tous les Linux.
- **cfdisk** est un peu plus « visuel » et s'utilise avec les flèches directionnelles. Il permet les mêmes opérations que fdisk mais de manière plus conviviale.
- **sfdisk** fonctionne en interactif ou non, est assez compliqué mais plus précis.
- **parted** permet des opérations très avancées sur les partitions comme par exemple leur redimensionnement. Il est soit interactif (c'est un interpréteur de commandes) soit scriptable. Il existe des interfaces graphiques comme **qtparted** ou **gparted**.

Les types de partitions sont les suivants :

- **83** : Partition de type Linux (données)
- **82** : Partition de type swap
- **fd** : Partition de type RAID

En principe après avoir créé pour modifié la table des partitions, il faut redémarrer la machine. Cependant quelques commandes permettent de passer outre, notamment en mettant à jour le système de fichiers /proc ou /sys. C'est le cas de **blockdev**. La commande **partprobe** de parted fait la même chose.

```
$ /sbin/blockdev --rereadpt
```

3.2 Créer les systèmes de fichiers

3.2.1 mkfs

Les commandes de « formatage » telles que sous Microsoft n'existent pas. Un formatage de type Microsoft est en fait la création et la vérification d'un système de fichiers sur une partition. La commande pour créer un système de fichiers est **mkfs**. **mkfs** appelle en fait d'autres programmes en fonction du type de système de fichiers sélectionné. C'est un frontend.

```
mkfs -t typefs périphérique
```

C'est typefs qui détermine le type de système de fichiers et donc le programme appelé.

- **ext2** (mkfs.ext2):Second extended filesystem, a longtemps été le système de fichiers par défaut de Linux. Il est rapide, simple et ne craint pas la fragmentation. Il n'est pas journalisé.
- **ext3** (mkfs.ext3) : Third extended filesystem, est en fait une évolution de ext2 (avec lequel il est compatible) mais journalisé. Très performant sur les gros fichiers. Il est possible de monter un système de fichiers ext3 en ext2 mais il faudra alors reconstruire le journal ensuite.

- **reiserfs** (mkfs.reiserfs) : du nom de son inventeur, système de fichiers journalisé. Très performant sur les petits fichiers, car il gère mieux les espaces vides.
- **vfat** (mkfs.vfat) : le système de fichiers vfat (32 bits, noms longs) de Microsoft.
- **msdos** (mkfs.msdos) : le système de fichiers fat (12 ou 16 bits).

3.2.2 ext2 et ext3

Voici les options courantes à ext2/3.

- **-b** : taille des blocs en octet, multiple de 512. Si la taille n'est pas précisée, elle sera déterminée par la taille de la partition. Tout fichier créé sur le disque occupe au moins un bloc et donc si on manipule un grand nombre de petits fichiers il faut mettre une valeur basse (ex : 1024).
- **-c** : vérifie les mauvais blocs avant de créer le système de fichiers. On peut aussi utiliser la commande « **badblocks** ».
- **-i** : ratio octets/inode. La taille de la table des inodes est calculée en fonction de la taille totale du système de fichiers. Un inode occupe 128 octets. En mettre moins limite le nombre de fichiers possibles mais permet de gagner de la place. -i 4096 : un inode pour chaque 4 ko.
- **-m** : pourcentage réservé au super-utilisateur, par défaut 5%. La mettre à zéro permet de gagner de la place et root pourra tout de même y travailler.
- **-L** : étiquette (nom) du système de fichiers, utile pour le montage.
- **-j** : crée un journal ext3.

```
mkfs -t ext2 -b 2048 -i 4096 -c -L HOME /dev/hda5
```

Ext3 est un système de fichiers ext2 auquel on a rajouté un journal. On peut convertir ext2 en ext3 en utilisant **tune2fs**.

```
tune2fs -j /dev/hda5
```

On peut afficher et changer le label du système de fichier en tapant e2label.

```
$ e2label /dev/hda5
HOME

$ e2label /dev/hda5 OLDHOME
```

Il ne faut pas oublier de modifier les options de montage en conséquence.

3.2.3 Reiserfs

On peut modifier un label **reiserfs** avec la commande **reiserfstune**.

```
reiserfstune -l HOME /dev/hda6
```

Il ne faut pas oublier de modifier les options de montage en conséquence.

3.3 Monter les systèmes de fichiers

3.3.1 mount

Tout d'abord, on peut souligner l'intérêt pratique d'utiliser des labels pour ses systèmes de fichiers. En cas de réorganisation des disques (déplacement dans une chaîne SCSI par exemple), l'ordonnancement des périphériques est modifié. L'utilisation des noms de périphériques oblige dans

ce cas à modifier le fichier **/etc/fstab** à chaque modification. Ce n'est pas le cas avec les labels.

La commande **mount** permet d'accéder aux périphériques de type blocs (les partitions) sur lesquels un système de fichier existe. La commande **mount** attache le répertoire racine du système de fichiers à un répertoire pré-existant appelé point de montage (mountpoint).

```
mount -t typefs -o options périphérique point_de_montage
```

Par exemple.

```
$ mount -t ext3 /dev/hda5 /home
```

Le système de fichiers contenu dans /dev/hda5 va être rattaché au répertoire /home existant. La commande **umount** détache le système de fichiers du point de montage.

```
$ umount /home
```

Si un ou plusieurs fichiers du système de fichiers à démonter sont encore en cours d'utilisation, alors **umount** ne marchera pas. Dans ce cas la commande **fuser** permettra de déterminer et éventuellement de tuer les processus les utilisant :

```
$ fuser -km /home
```

Si on modifie une option de montage du système de fichiers (via le paramètre -o) on peut aussi passer l'option remount pour que la modification soit prise tout de suite en compte.

```
$ mount -o rw,remount /home
```

3.3.2 /etc/fstab

Le fichier **/etc/fstab** contient une configuration statique des différents montages des systèmes de fichiers. Il est appelé à chaque démarrage du système car c'est ici qu'on indique les périphériques et leur point de montage. Il contient six champs.

```
périphérique  point_de_montage typefs options dump fsck
```

- **périphérique** : le périphérique à monter. Il peut être spécifié en tant que chemin de périphérique (**/dev/hda1** par exemple) ou label de système de fichiers s'il existe (ex **LABEL=/home**).
- **point_de_montage** : le chemin d'accès au système de fichiers
- **typefs** : le type de système de fichiers
- **options** : les options de montage séparés par des virgules
- **dump** : fréquence de dump. 1=quotidien, 2=tous les deux jours, etc. 0=jamais
- **fsck** : Fréquence de vérification du système de fichiers.0=ignorer. 1=en premier, 2 en second, etc. Les systèmes ayant le même numéro sont vérifiés en parallèle.

Au démarrage de la machine **rc.sysinit** balaie ce fichier. Tous les systèmes de fichiers ne possédant pas noauto comme option sont automatiquement montés. Le premier à l'être est le système de fichiers racine « / ». Puis viennent ensuite le swap et les autres systèmes de fichiers s'ils sont spécifiés (ex : /home, /usr, etc) ainsi que les systèmes de fichiers virtuels /proc, /sys , /dev/pts, etc.

```
LABEL=/         /               ext3    defaults        1 1
LABEL=/boot     /boot           ext3    defaults        1 2
LABEL=/home     /home           ext3    defaults        1 2
/dev/sda5       swap            swap    defaults        0 0
```

L'option particulière bind peut être très pratique pour faire apparaître une partie de système de fichiers à plusieurs points de montages. Dans l'exemple suivant le système de fichiers ayant le label **u01** est rattaché au répertoire **/u01**. Puis on rattache **/u01/applis** au répertoire **/applis**.

```
LABEL=/u01      /u01            ext3    defaults 1 2
/u01/applis        /applis         none    bind
```

Le contenu de **/etc/fstab** peut être utilisé après l'initialisation du système pour monter et démonter ponctuellement les systèmes de fichiers qui n'ont pas par exemple l'option noauto, ou les supports de masse comme les lecteurs cd/dvd. Dans ce cas on peut utiliser très simplement les labels, les points de montage ou le périphérique.

```
mount /home
mount -L /u01
mount LABEL=/boot
mount /dev/hda5
```

3.4 ACL

Les systèmes de fichiers ext3 et reiserfs supportent les listes de contrôles d'accès **ACL** (*Access Control List*). Pour les utiliser, il faut rajouter l'option **acl** soit à **mount**, soit dans **/etc/fstab**. Les ACL permettent un contrôle plus fin des droits que ceux présents par défaut sur Unix. Il devient possible d'appliquer un droit précis à une personne ou un groupe précis.

On applique un ACL avec la commande **setfacl**.

```
setfacl -m u:titi_05:rx script.sh
setfacl -m g:grp_classe1:rwx rep1
setfacl -b script.sh
setfacl -x g:grp_classe1 rep1
```

- **-m** : modifie les ACL
- **[ug]:nom :** s'applique sur le user (u) ou le groupe nommé
- **rwx** : les droits unix classiques
- **-b** : supprime tous les ACL sur le fichier donné
- **-x** : supprime les ACL pour le user ou le groupe donné sur un fichier donné

Quand un ACL est appliqué, un signe « + » apparaît à côté des droits Unix classiques.

On vérifie les ACL avec la commande **getfacl**.

```
getfacl script.sh
# file : script.sh
# owner : toto
# group : users
user::rwx
user:titi_05:r-x
group::r--
mask::rw-
other::r--
```

4 Redhat Package Manager

4.1 Notion de package

Contrairement à d'autres systèmes d'exploitation, il n'est pas courant sur Linux et Unix en général de disposer de logiciels fournis avec un programme l'installation interactif (pas de install.exe). Certains éditeurs proposent des scripts d'installation et bien souvent ceux-ci se contentent de décompresser et de désarchiver quelques fichiers.

Avec Linux il est très classique de disposer des divers produits, outils, mises à jour, etc sous forme de paquetages (packages). Un package est un fichier (parfois gros) qui contient le produit à installer et des règles. Ces règles peuvent être multiples :

- gestion des dépendances : le produit ne pourra être installé que si les produits qu'il utilise lui-même sont déjà présents.
- Pré-installation : des actions sont à prévoir avant de pouvoir installer le produit (changer des droits, créer des répertoires, etc)
- Post-installation : des actions sont à prévoir après l'installation du produit (paramétrage d'un fichier de configuration, compilation annexe, etc).

Sur Red Hat, Fedora, SuSE, Mandriva et quelques autres distributions le format de package par défaut est le **`RPM`** (Redhat Package Manager). Sous Debian, Knoppix, Kaella, Ubuntu, c'est le format **`DPKG`** (Debian Package). Outre le format, ce sont surtout les outils qui les différencient.

Le fait de disposer des informations de dépendances permet d'obtenir des outils performants qui peuvent seuls les résoudre en cascade. En installant un package, l'outil pourra installer toutes les dépendances nécessaires. On peut parfois spécifier plusieurs emplacements (repositories) pour ces packages, soit locaux (disque dur, cdrom, dvd, etc) soit distants (http, ftp, etc).

Il faut toujours utiliser un package prévu pour sa distribution quand il existe. Si ce n'est pas le cas, il est parfois possible d'utiliser un package d'un produit concurrent ou de recompiler le produit soit-même.

Les mises à jour d'un système Linux utilisant un système de packaging est très simplifiée. Pour passer d'une version d'un produit à un autre, il suffit de récupérer le package du produit en version supérieure et de l'installer. Toutes les mises à jour sont sous cette forme. Depuis peu, il existe un format de « **`delta-rpm`** » qui ne fournit en package que les différences d'une version à une autre. Mais il est toujours possible d'utiliser un package complet.

4.2 Le gestionnaire RPM

`RPM` est un gestionnaire de packages inventé par Red Hat puis utilisé massivement par de nombreuses autres distributions. Il simplifie fortement la distribution, l'installation, la mise à jour et la suppression des logiciels. Il se base sur des commandes (ex : **`rpm`**), une base de données locale et des packages au format rpm (extension rpm).

La base de données est située dans /var/lib/rpm. Toutes les informations concernant les logiciels installés, leurs versions, leurs fichiers et droits, et leurs dépendances y sont précisés. Sauf gros problème, il ne faut JAMAIS modifier cette base à la main. Il faut utiliser les outils RPM.

Chaque logiciel est fourni sous forme de package au format RPM. Le rpm répond à une nomenclature précise.

```
nom-version-edition.architecture.rpm
```
par exemple :
```
php-4.1.2-2.1.8.i586.rpm
```

L'édition est un identifiant de version du package RPM propre à l'éditeur. Ici c'est la version 2.1.8 du package PHP version 4.1.2. L'architecture est i586 (Intel Pentium). On peut aussi trouver i386, x86_64 (64 bits), ppc64, s390x ou noarch. Un package noarch no contient pas de programmes ou bibliothèques binaires mais du code indépendant comme des scripts, de la documentation, des images, du son, de la video, etc.

4.3 Installation, mise à jour et suppression

La distribution RHEL/CentOS propose l'outil **`system-config-packages`** permettant l'installation de packages regroupés par domaine (développement, serveur, etc).

On installe un package rpm avec le paramètre « `-i` ».

```
rpm -i php-4.1.2-2.1.8.i586.rpm
```

Comme il est possible d'utiliser des caractères de substitution (**`rpm -i *.rpm`**) ont peut afficher le nom du package en cours d'installation avec le paramètre « **`-v`** ». Le paramètre « **`-h`** » affiche des caractères « **`#`** » pour indiquer la progression de l'installation. L'installation ne fonctionnera pas si les dépendances ne sont pas résolues.

La mise à jour d'un produit vers une version supérieure depuis un package se fait avec le paramètre « **`-U`** ». Dans ce cas tous les fichiers sont mis à jours par ceux de la nouvelle version : les anciens sont supprimés et remplacés par les nouveaux. Les anciens fichiers de configuration sont sauvés avec l'extension « **`.rpmsave`** ». Si le package n'était pas installé, la mise à jour joue le rôle d'installation.

```
rpm -Uvh php-4.1.3-1.i586.rpm
```

La mise à jour est aussi possible avec « **`-F`** ». Mais si le package n'était pas installé, il ne le sera pas non plus lors de la mise à jour contrairement à « **`-U`** ». Ainsi si on dispose de tous les packages de mise à jour du système et qu'on ne souhaite mettre à jour que ceux qui sont réellement installés, alors on peut taper

```
rpm -Fvh *.rpm
```

La suppression s'effectue avec le paramètre « **`-e`** ». Attention cependant c'est le nom du package isntallé qui doit être passé en paramètre et pas le nom du fichier de package.

```
rpm -e php
```

Plusieurs options supplémentaires sont possibles.

- **`--force`** : en cas de conflit avec un autre package (le cas le plus courant est celui où deux packages proposent le même fichier au même endroit), cette option force tout de même l'installation.
- **`--nodeps`** : Si le package refuse de s'installer à cause d'un problème de dépendances, cette option forcera l'installation. Il arrive parfois que cette erreur se produise quand la dépendance en question a été installée autrement que depuis un package rpm (ex : compilation, binaire copié à la main).

4.4 Cas du noyau

L'installation ou la mise à jour d'un noyau est un cas particulier. En effet, la mise à jour supprime l'ancienne version. Le noyau est un composant très critique du système. S'il devait être avéré que le système ne fonctionne plus (ou mal) avec le noyau mis à jour, il faudrait donc réinstaller un ancien noyau depuis le support d'installation. Aussi la procédure est la suivante :

- Installation du nouveau noyau avec le paramètre « **-i** », il sera rajouté au système
- Redémarrage et test de vos logiciels et périphériques avec le nouveau noyau
- S'il fonctionne correctement, suppression éventuelle de l'ancien noyau avec « **-e** ».
- Edition de **/boot/grub/grub.conf** et modification de la ligne « **default** » pour démarrer par défaut sur le nouveau noyau.

4.5 Requêtes RPM

La base de données RPM peut être interrogée facilement avec le paramètre « **-q** » suivi de plusieurs options.

- **-a** : liste de tous les packages installés
- **-i** : informations générales (le résumé) du package
- **-l** : liste des fichiers installés
- **-f nom** : trouve le package qui contient le fichier donné
- **-p nom** : la recherche s'effectue dans le fichier de package donné.
- **--requires** : dépendances du package
- **--provides** : ce que fournit le package
- **--scripts** : scripts exécutés à l'installation et la suppression
- **--changelog** : l'historique du package

```
$ rpm -qilp libjpeg-6.2.0-738.i586.rpm
Name        : libjpeg                      Relocations: (not relocateable)
Version     : 6.2.0                             Vendor: SUSE LINUX Products GmbH, Nuernberg, Germany
Release     : 738                           Build Date: Sat Mar 19 20:07:55 2005
Install date: (not installed)               Build Host: d121.suse.de
Group       : System/Libraries              Source RPM: jpeg-6b-738.src.rpm
Size        : 125804                           License: BSD, Other License(s), see package
Signature   : DSA/SHA1, Sat Mar 19 20:12:06 2005, Key ID a84edae89c800aca
Packager    : http://www.suse.de/feedback
URL         : http://www.ijg.org/
Summary     : JPEG libraries
Description :
The libraries (static and dynamic) for the jpeg-graphics format. The
sources are contained in the jpeg source package.



Authors:
--------
    Rob Hooft <hooft@EMBL-Heidelberg.DE>
    Michael Mauldin <mlm@cs.cmu.edu>
/usr/lib/libjpeg.so.62
/usr/lib/libjpeg.so.62.0.0

$ rpm -qp libjpeg-6.2.0-738.i586.rpm --requires
/sbin/ldconfig
/sbin/ldconfig
rpmlib(PayloadFilesHavePrefix) <= 4.0-1
rpmlib(CompressedFileNames) <= 3.0.4-1
```

```
libc.so.6
libc.so.6(GLIBC_2.0)
libc.so.6(GLIBC_2.1.3)
rpmlib(PayloadIsBzip2) <= 3.0.5-1
```

```
$ rpm -qi php
Name        : php                           Relocations: (not relocateable)
Version     : 4.1.2                              Vendor: Red Hat, Inc.
Release     : 2.1.8                          Build Date: mer 14 jui 2004 11:24:16 CEST
Install date: lun 27 jun 2005 19:36:32 CEST      Build Host: porky.build.redhat.com
Group       : Development/Languages         Source RPM: php-4.1.2-2.1.8.src.rpm
Size        : 3843552                          License: The PHP License, version 2.02
Packager    : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
URL         : http://www.php.net/
Summary     : The PHP HTML-embedded scripting language. (PHP: Hypertext Preprocessor)
Description :
PHP is an HTML-embedded scripting language. PHP attempts to make it
easy for developers to write dynamically generated webpages. PHP also
offers built-in database integration for several commercial and
non-commercial database management systems, so writing a
database-enabled webpage with PHP is fairly simple. The most common
use of PHP coding is probably as a replacement for CGI scripts. The
mod_php module enables the Apache Web server to understand and process
the embedded PHP language in Web pages.
```

```
$ rpm -qa | grep php
php-ldap-4.1.2-2.1.8
php-imap-4.1.2-2.1.8
asp2php-0.75.17-1
php-4.1.2-2.1.8
```

```
$ rpm -qf /usr/bin/passwd
passwd-0.68-1.2.1
```

4.6 Vérification des packages

Il est possible qu'après l'installation d'un package, un ou plusieurs des fichiers installés aient été altérés (changement de droit, de propriétaire, édition, suppression, etc). Comme la base RPM contient toutes les informations nécessaires, on peut demander une vérification avec le paramètre « -V ».

```
$ rpm -V php
S.5....T c /etc/php.ini
```

Un point signifie qu'une étape de vérification est OK. Sinon :

- **S** : La taille du fichier a été modifiée
- **5** : la somme MD5 ne correspond plus
- **T** : la date de modification n'est plus la même
- **U** : le propriétaire a été modifié
- **G** : le groupe a été modifié
- **L** : le lien symbolique a été modifié
- **M** : les permission ou le type du fichier ont été modifiés
- **D** : le périphérique a été modifié (major/minor)

Le « **c** » indique qu'il s'agit d'un fichier de configuration.

Les fichiers de packages RPM sont très souvent signés par l'éditeur de la distribution de manière à en garantir l'intégrité. On peut vérifier l'intégrité d'un package avec une clé publique GPG mais il faut par avance avoir déjà chargé cette clé publique sur le système.

```
gpg --import RPM-GPG-KEY
```

```
rpm --import RPM-GPG-KEY
rpm --checksig libjpeg-6.2.0-738.i586.rpm
```

4.7 Les dépendances

Si vous utilisez les outils graphiques fournis par votre distribution, ceux-ci tenteront de résoudre les dépendances à votre place. La commande RPM seule ne le fait pas par défaut. Des outils complémentaires « frontend » comme **yast**, **apt** ou **yum** le font à sa place. La distribution Red Hat fournit un outil appelé **rpmdb-redhat** pour installer automatiquement les dépendances via rpm. Cela implique notamment le fait que tous les packages de la distribution doivent se trouver au même endroit (dans le même répertoire) et le système ne fonctionne qu'avec les packages officiels de Red Hat. On emploie le paramètre « --aid ».

```
$ rpm -ivh --aid libjpeg-6.2.0-738.i586.rpm
```

4.8 Mises à jour automatisées

Chaque distribution fournit maintenant un outil de mise à jour interactif ou automatisé. La SuSE propose **YOU** (Yast Online Update), la Red Hat propose **up2date**. La version RHEL de Red Hat étant payante, l'accès aux mises à jour dépend d'un numéro de licence et d'une inscription à RHN (Red Hat Network). Les versions dérivées comme CentOS ne peuvent pas se mettre à jour via RHN mais proposent leur propre site distant de mise à jour.

Il faut penser à ajouter la clé GPG de sécurité :

```
rpm --import http://mirror.centos.org/centos/RPM-GPG-KEY-CentOS-4
```

5 Administration des utilisateurs

5.1 Ajout

La création d'un utilisateur pourrait être entièrement effectuée à la main :

- Rajout d'une ligne dans **/etc/passwd**
- Rajout d'une ligne dans **/etc/shadow**
- Rajout éventuel des informations dans **/etc/group**
- Créer le répertoire personnel et mettre à jour son contenu avec **/etc/skel**
- Changer les permissions et le propriétaire du répertoire personnel
- Changer le mot de passe (encodé **MD5**)

Tout ceci peut être effectué avec la commande **useradd**. Elle ajoute un nouveau compte et effectue les principales opérations :

- création de l'utilisateur et remplissage des fichiers
- création d'un groupe privé d'utilisateur (de même nom que celui-ci)
- création du répertoire personnel, remplissage et modification des droits

```
useradd robert
```

La commande ne créé pas de mot de passe. Il faut le faire à la main.

```
passwd robert
```

5.2 modification

On utilise la commande **usermod** pour modifier un compte.

- **-c** : Modifie le commentaire (nom complet de l'utilisateur)
- **-d** : change le répertoire personnel
- **-e** : date d'expiration du compte
- **-g** : groupe initial de l'utilisateur
- **-G** : liste de groupes additionnels séparés par des virgules
- **-l** : change le login
- **-s** : modifie le shell de connexion de l'utilisateur
- **-u** : définit un nouveau UID
- **-p** : change le mot de passe. Attention aucun encodage n'est effectué !
- **-L** : verrouille le compte
- **-U** : déverrouille le compte

5.3 Suppression

On supprime un utilisateur avec **userdel**. On peut passer le paramètre « **-r** » qui va aussi

supprimer le répertoire personnel de l'utilisateur.

```
userdel -r robert
```

5.4 Politique de sécurité

Par défaut les mots de passe n'expirent jamais. Il reste possible si rien n'est fait de garder le même mot de passe « à vie ». La commande **chage** permet de définir une politique de modification des mots de passe.

- **-m** : nombre minimum de jours entre les changements de mot de passe
- **-M** : nombre maximum de jours entre les changements de mot de passe
- **-I** : délai en jours d'inactivité du compte après l'expiration du mot de avant son verrouillage
- **-E date** : le mot de passe expire à cette date. YYYY-MM-JJ.
- **-W** : nombre de jours avant la date d'expiration du mot de passe où l'utilisateur sera prévenu.

```
chage -m 40 -M 45 -I 0 -W 5 robert
```

Ces informations sont placées dans le fichier **/etc/shadow**.

5.5 Les groupes

On peut créer un groupe directement dans le fichier **/etc/group** ou passer par les commandes :

- **groupadd** : créer un groupe

  ```
  groupadd classe1
  ```

- **groupdel** : supprimer un groupe

  ```
  groupdel classe1
  ```

- **groupmod** : modifier un groupe

  ```
  groupmod -n nouveaunom anciennom
  ```

5.6 Groupes privés et setgid

La politique de Red Hat pour la sécurité des utilisateurs et de systématiquement leur créer un groupe privé et de leur attribuer un masque 002. De ce fait, les fichiers sont mieux protégés par défaut puisqu'aucun groupe ne peut accéder aux fichiers par défaut : les fichiers créés n'appartiennent qu'à un groupe qui n'a qu'un membre.

Comme un utilisateur peut faire partie de plusieurs groupes, il peut en théorie étendre l'accès à ses fichiers et répertoires avec la commande **chgrp** qui change le groupe d'un fichier. De même il peut changer temporairement de groupe principal avec la commande **newgrp**.

Cependant l'utilisateur préfère souvent ne pas se « casser la tête » et donner tous les droits avec un **chmod**.

Dans ce cas, la solution est d'utiliser le bit **setgid** sur un répertoire.

1) On définit un groupe commun à tous les utilisateurs devant pouvoir accéder à un répertoire et à ses fichiers.
2) On créé un répertoire dont le groupe propriétaire est le groupe commun.

3) On donne au répertoire les droits avec l'umask 002 (`rwxrwxr-x`) ou autre, mais avec tous les droits sur le groupe.
4) On ajoute au répertoire le droit setgid (s) sur le groupe.

```
$ mkdir rep
$ chgrp grpcommun rep
$ chmod 2770 rep
```

Lorsque le setgid est positionné sur un répertoire, tous les fichiers créés dans ce répertoire appartiennent au groupe du répertoire et non pas au groupe de l'utilisateur. Ainsi tous les membres du groupe commun pourront accéder aux données. Le fichier continue cependant à appartenir à son créateur.

Associé au Sticky-Bit, on obtient un bon niveau de protection et d'accès aux données.

5.7 L'environnement utilisateur

A la création d'un utilisateur et de son répertoire personnel, l'environnement de l'utilisateur est mis en place. L'environnement contient par exemple les variables d'environnement et les alias. Il est contenu dans des fichiers chargés au démarrage de l'interpréteur de commandes (shell). Les fichiers sont copiés de **`/etc/skel`** (skeleton) vers le répertoire personnel. Si on souhaite modifier les environnements de façon globale AVANT la création des utilisateurs,

A la connexion d'un utilisateur, les scripts suivants sont exécutés dans cet ordre :

- **`/etc/profile`** : définit les variables d'environnement importantes comme **`PATH`**, **`LOGNAME`**, **`USER`**, **`HOSTNAME`**, **`HISTSIZE`**, **`MAIL`** et **`INPUTRC`**.
- **`/etc/profile.d/*`** : **`/etc/profile`** appelle tous les scripts présents dans ce répertoire. Ces scripts peuvent compléter la configuration globale en ajoutant par exemple la configuration des paramètres linguistiques, des alias globaux, etc.
- **`~/.bash_profile`** : si le shell est bash, c'est le script suivant à être exécuté. Il est dans le répertoire utilisateur et appelle un autre script : **`~/.bashrc`** qui appelle lui-même **`/etc/bashrc`**. On peut définir dans .bash_profile des variables supplémentaires, alors qu'on aura tendance à définir dans **`~/.bashrc`** des alias et des fonctions. Il n'y a pas de règles strictes.
- **`/etc/bashrc`** est utilisé pour définir les fonctions et alias pour tout le système et tous les utilisateurs sous bash.

5.8 PAM

`PAM` « *Pluggable Authentification Modules* » est un ensemble de modules et une bibliothèque permettant de mettre en place des mécanismes d'authentification avancés pour tous les outils nécessitant une sécurité accrue. L'authentification est basée sur des modules. Chaque module peut utiliser des mécanismes différents pour tenter d'authentifier un utilisateur. L'un se basera sur une authentification Unix classique, un autre sur du LDAP, un troisième sur de l'Active Directory, et un dernier sur la reconnaissance d'une emprunte digitale. Ce mécanisme permet aussi la vérification via un dongle USB...

Le principe est assez simple, la configuration plus compliquée. Un outil appelle la bibliothèque **`libpam.so`** pour un besoin d'authentification. La bibliothèque lit un fichier de configuration ou l'appel à plusieurs modules peut être demandées. Chaque module appelé retourne vrai ou faux. Suivant la configuration, vrai peut être un pré-requis pour continuer avec un autre module ou être

suffisant pour se connecter.

Les fichiers de configuration sont placés dans **/etc/pam.d**. Il en existe généralement un par application utilisant PAM et il porte le même nom. Si le fichier est manquant c'est « **other** » qui est utilisé. La syntaxe est la suivante.

```
Type_module contrôle module arguments
```

Type_module :

- **auth** : module d'authentification (par exemple demande de login et de mot de passe)
- **account** : autorisation, gestion de comptes (vérification de l'utilisateur pour le service donné. Est-il autorisé ?)
- **password** : vérification et mise à jour des informations de sécurité (ex : le mot de passe est-il encore valable et si non, demande de nouveau mot de passe)
- **session** : modification de l'environnement de l'utilisateur

contrôle :

- **required** : réussite requise. En cas d'échec, les modules restants sont tout de même appelés mais quoi qu'il arrive au final PAM retournera un échec.
- **requisite** : un échec termine immédiatement l'authentification. La réussite continue
- **sufficient** : une réussite contourne les autres modules. Autrement dit PAM retourne ok quoi qu'il arrive en cas de réussite ici.
- **Optional** : le résultat est ignoré.

Module :

- **pam_unix.so** : authentification standard via la fonction C getpw() (généralement **/etc/passwd**, mais dépend de **/etc/nsswitch.conf**).
- **pam_env.so** : définition des variables d'environnement
- **pam_securetty.so** : Interdit une connexion super-utilisateur (root) depuis un terminal non sécurisé. La liste des terminaux autorisés est placée dans **/etc/securetty**.
- **pam_stack.so** : appelle un autre service PAM pour le chargement de modules supplémentaires.
- **pam_nologin.so** : interdit la connexion d'utilisateurs si le fichier **/etc/nologin**. Dans ce cas sont contenu est affiché.
- **pam_deny.so** : retourne toujours un échec.
- **pam_console.so** : donne des permissions supplémentaires à un utilisateur local.

Les paramètres dépendent de chaque module.

Exemple dans le cas classique d'une ouverture de session par la console. Dans ce cas, le shell de connexion appelle la commande **login**. Attention ce n'est qu'un exemple !

```
$ cat login
#%PAM-1.0
auth       requisite    /lib/security/pam_securetty.so
auth       required     /lib/security/pam_env.so
auth       sufficient   /lib/security/pam_unix.so
auth        required     /lib/security/pam_deny
auth       required     /lib/security/pam_nologin.so
```

- La première ligne vérifie si root tente de se connecter depuis un terminal non sécurisé (ex : telnet, rlogin, rsh, etc). Si c'est la cas, PAM retourne directement faux et l'authentification échoue directement.
- La seconde ligne charge l'environnement de l'utilisateur. Un échec ici n'empêche pas l'exécution des lignes suivantes, mais au bout du compte PAM retournera faux.
- La troisième ligne tente une authentification via les mécanismes Unix classiques (fichier des mots de passe, NIS, etc). La réussite ici stoppe la suite : l'utilisateur est directement authentifié. Autrement, on passe à la ligne suivante.
- La quatrième ligne retourne toujours faux. Les modules d'authentification précédents ayant échoué, la connexion de l'utilisateur échoue. La ligne suivante est quand même exécutée.
- Si le fichier **/etc/nologin** existe, il est affiché.

Il est possible d'interdire l'accès à une liste d'utilisateurs donnés. Placez les noms des utilisateurs interdits dans **/etc/nologinusers** et ajoutez la ligne suivante dans le fichier de configuration PAM. Dans notre exemple, il faudrait l'ajouter entre la deuxième et la troisième ligne.

```
auth required /lib/security/pam_listfile.so onerr=suceed item=user sense=deny file=/etc/nologinusers
```

RHEL est configuré de telle manière que le fichier **/etc/security/system-auth** soit appelé dans toutes les configurations PAM ou presque. Exemple du fichier de configuration login :

```
auth       required     /lib/security/pam_securetty.so
auth       required     /lib/security/pam_stack.so service=system-auth
auth       required     /lib/security/pam_nologin.so
...
```

Dans ce cas, nous aurions mis notre ligne en-dessous de celle de pam_nologin.so. Cependant il faut garder à l'esprit que le fichier de configuration **/etc/pam.d/login** n'est utilisé que pour l'authentification depuis la commande **login** et donc une connexion depuis un terminal (console texte). Depuis une fenêtre de connexion graphique (X-Window, x/k/gdm) nous n'aurons pas l'effet souhaité. Dans ce cas, la modification aurait dû être effectuée dans **/etc/pam.d/system-auth**.

```
auth        required      /lib/security/pam_env.so
auth        sufficient    /lib/security/pam_unix.so likeauth nullok
auth        required      /lib/security/pam_deny.so
...
```

Attention encore une fois à placer la ligne au bon endroit. Après ce bloc, la ligne n'aura aucun effet à cause de la ligne **sufficient** si l'utilisateur à un login et un mot de passe valides. On placera donc la ligne au début des lignes **auth**.

Bonus : si vous possédez un modèle IBM avec reconnaissance des empreintes digitales, rendez-vous sur http://www.qrivy.net/~michael/blua/.

5.9 Les quotas disques

5.9.1 Définitions

Les **quotas** permettent de poser des limites à l'utilisation de systèmes de fichiers. Ce limites sont de deux types :

- **inodes** : limite le nombre de fichiers
- **blocs** : limite la taille disque

Les quotas sont implémentés par système de fichier individuel et pas pour l'ensemble des systèmes de fichiers. Chaque utilisateur peut être géré de manière totalement indépendante. Il en est de même pour les groupes. Pour chaque utilisation (inode, bloc), on peut mettre en place deux limites :

- **Limite dure** (hard) : quantité maximale d'inodes ou de blocs utilisés que l'utilisateur ou le groupe ne peuvent absolument pas dépasser. Dans ce cas, plus rien ne sera possible (création de fichier ou fichier dont la taille dépasse la limite).
- **Limite douce** (soft) : quantité maximale d'inodes ou de blocs utilisés que l'utilisateur ou le groupe peuvent temporairement dépasser. Dans ce cas, les créations et modifications seront possibles jusqu'à un certain point.
- **Un délai de grâce** est mis en place. Durant ce temps, l'utilisateur peut continuer à travailler sur le système de fichiers. Le but est qu'il revienne à terme sous la limite douce. Le délai dépassé, la limite douce devient la limite dure. Quoi qu'il arrive, l'utilisateur ne pourra jamais dépasser la limite dure.

Les quotas sont implémentés dans le noyau Linux. Pour les utiliser, les outils du package RPM **quota** doivent être installés.

5.9.2 Mise en place

Nous allons mettre en place les quotas sur la partition /home.

1) **Modifier les options de partitions dans /etc/fstab**. On rajoute dans les options **usrquota** (utilisateur) ou **grpquota** (groupe).

```
LABEL=/home     /home           ext3    defaults,usrquota        1 2
```

2) **Remonter le système de fichiers.**

```
mount -o remount /home
```

3) **Créer les fichiers contenant les informations de quota** (base de données de quotas).

```
cd /home
touch aquota.user aquota.group
```

4) **Mettre à jour la base de données** avec la commande **quotacheck**.

```
quotacheck -c /home
```

5) **Démarrer les quotas** (ou les arrêter). Cette opération n'est pas nécessaire après un redémarrage de Linux car la mise en place des quotas est comprise dans **/etc/rc.d/rc.sysinit**. La commande **quotaon** démarre les quotas pour le système de fichiers indiqué (-a pour tous). La commande **quotaoff** stoppe les quotas.

```
quotaon /home
```

6) **Editer les quotas** pour les utilisateurs ou les groupes. La commande **edquota** est utilisée. En pratique, si tous les utilisateurs doivent avoir les mêmes quotas ou quelques variantes, on créé un utilisateur lambda dont on recopiera les propriétés.

   Etablir les quotas pour roger :

```
edquota roger # = edquota -u roger
```

   Les quotas de arthur sont identiques à ceux de roger :

```
edquota -p roger arthur
```

7) **Etablir le délai de grâce**. Le délai accepte les unités « seconds », « minutes », « hours », « days », « weeks », « monthes ».

```
edquota -t
```

8) **Vérifier les quotas**. Les utilisateurs peuvent vérifier l'état de leurs quotas avec la commande **quota**. L'administrateur peu générer un rapport avec **repquota**. Enfin la commande **warnquota** qui peut être exécutée via cron peut envoyer un mail aux utilisateurs pour les prévenir en cas de dépassement.

L'édition des quotas se fait avec l'éditeur par défaut du système qui est généralement VI (le comportement peut être modifié via les variables EDITOR et VISUAL). **Les blocs de quotas sont des blocks de 1 Ko**.

```
# edquota roger
Disk quotas for user roger (uid 502):
Filesystem     blocks  soft          hard          inodes  soft          hard
/dev/hda5      1695256 2500000 3000000 12345         0             0
```

Avec l'éditeur, on peut modifier les valeurs soft et hard qui correspondent aux limites douces et dures pour le nombre de blocs et le nombre d'inodes. Ci-dessus, nous avons établi une limite douce à environ 2,4 Go (2500000 ko) et dure à environ 2,9 Go (3000000 ko) d'occupation du système de fichiers pour roger. Nous n'avons pas placé de quotas sur le nombre d'inodes (valeur à 0).

Note : le contenu des champs blocks et inodes est dynamique.

```
# edquota -t
Filesystem            Block grace period     Inode grace period
/dev/hda3                     7days                  7days

# repquota /home
*** Report for user quotas on device /dev/hda5
Block grace time: 7 days; Inode grace time: 7 days
                        Block limits                File limits
User            Used     soft      hard    grace     used     soft     hard     grace
----------------------------------------------------------------------------------
root      --   12345        0         0                5        0        0
roger     -- 1695256  2500000   3000000             12345        0        0
```

Une dernière nécessité est d'utiliser régulièrement la commande **quotacheck** pour maintenir la cohérence des informations de quotas des systèmes de fichiers. En effet, en cas arrêt des quotas ou de problème (arrêt violent par exemple), il peut parfois être nécessaire de vérifier et de réactualiser les informations.

```
# quotacheck -avug
```

5.10 Client NIS

Un serveur **NIS** (*Network Information Services*) permet de diffuser des informations contenues dans des fichiers vers des clients appartenant à son domaine NIS. Ces informations sont appelées des « **maps** ». Dans les maps, on retrouve par exemple le fichier **/etc/passwd**, le fichier **/etc/shadow**, le fichier **/etc/hosts**, etc. Il est souvent utilisé en relation avec les partages NIS. En pratique, un client NIS récupère toutes les informations du serveur NIS et il n'y a plus besoin de créer de comptes locaux : tout est centralisé sur le serveur.

Le client NIS est un programme exploitant les RPC. Il a besoin du portmapper **portmap** et du programme **ypbind**. Il faut aussi définir le nom du domaine NIS dans **/etc/sysconfig/network** et le serveur dans **/etc/yp.conf**. Enfin il faut aussi modifier le fichier **/etc/nsswitch.conf** pour rajouter la recherche des informations dans NIS et aussi les modules PAM en conséquence.

RHEL propose l'outil **`authconfig`** pour paramétrer un client NIS simplement.

6 Outils d'administration

6.1 Le réseau

On se reportera au chapitre TCP/IP pour les outils d'administration réseau.

6.2 L'impression

6.2.1 Principe

Il existe trois standards d'impression sous Unix, un sous System V, un autre sous BSD et un dernier fédérateur.

Quelque soit le standard, le principe est le même. A chaque imprimante déclarée (généralement dans **`/etc/printcap`**) correspond une file d'attente (**queue**). L'ensemble de ces files d'attente est géré par un service indépendant. Ces deux principes permettent une impression multi-utilisateur (les jobs d'impression sont en file d'attente, **job queues**), et en réseau (le service peut être accédé depuis une autre machine distante).

En règle générale toutes les imprimantes savent directement imprimer du texte brut ASCII en 80 colonnes. Pour imprimer des documents formatés, des images, on peut utiliser un pilote. On parle en fait de filtre d'impression. Le filtre peut être un script ou un binaire qui récupère le flux entrant (texte, image, document, postscript, ...), l'identifie et à l'aide de traitements associés le transforme en langage compréhensible par l'imprimante (Postscript, PCL, Canon, Epson, WPS, ...).

6.2.2 System V

- **`lp [-dImprimante] [-nChiffre] fic1`** : imprime le contenu du fichier fic1. L'option -d permet de choisir l'imprimante, -n le nombre d'exemplaires.
- **`lpstat [-d] [-s] [-t] [-p]`** : informations sur l'impression. L'option -d affiche le nom de l'imprimante par défaut, -s un état résumé des imprimantes, -t la totalité des informations (statut, jobs, ...), -p [liste] uniquement les informations sur les imprimantes incluses dans la liste.
- **`cancel [ids] [printers] [-a] [-e]`** : supprime les tâches d'impression ids des imprimantes printers. L'option -a supprime tous les jobs de l'utilisateur, -e tous les jobs (seulement l'administrateur).
- Sur certains systèmes comme SCO, on peut trouver les commandes **`enable`** et **`disable`** qui prennent comme paramètre le nom de la file d'attente, permettant d'en activer ou désactiver l'accès.
- Le démon (ou daemon) s'appelle généralement **`lpd`** (line printer daemon) ou **`lpsched`** (line printer scheduler).
- **`lpadmin`** permet d'administrer les services d'impression comme les files d'attentes associées à une imprimante et la file d'attente par défaut. Ex : **`lpadmin -p queue1 -v imprimante-m modèle`**.
  - **`lpadmin -x file`** : suppression de la file d'attente
  - **`lpadmin -d file`** : définir la file d'attente par défaut

- **lpadmin -p file -u allow:liste** : autorisation d'imprimer pour les utilisateurs précisés
- **lpadmin -p file -u deny:liste** : interdiction d'imprimer pour les utilisateurs précisés

- **lpshut** arrête le service d'impression. Au redémarrage du démon, les impressions en cours au moment de l'arrêt sont reprises.
- **accept** et **reject** permettent de valider une file d'attente pour l'impression ou de la fermer.
- **lpmove** permet de transférer des requêtes d'impression d'une file d'attente vers une autre.

6.2.3 BSD

- **lpr [-Pimprimante] [-#copies] fic1** : imprime le contenu du fichier fic1. L'option -P permet de spécifier l'imprimante, -# le nombre de copies.
- **lpq [-Pimprimante]** : indique l'état et la liste des jobs pour l'imprimante éventuellement spécifiée par l'option -P.
- **lprm [-Pimprimante] [-] [ids]** : permet de supprimer un job de l'imprimante spécifiée par l'option -P, l'option – supprime tous les jobs de l'utilisateur, ids représente une liste de jobs à supprimer.
- Sur certains Unix on peut trouver la commande **lpc**, qui est une sorte de petit shell permettant de contrôler les imprimantes et les jobs.
- Le service s'appelle généralement **lpd.**

6.2.4 CUPS

CUPS (*Common Unix Printing System*) est un système d'impression Unix, orienté réseau :

- Basé sur le protocole **IPP** (*Internet Printing Protocol*) basé lui-même sur le protocole **HTTP/1.1**.
- Simple d'utilisation, notamment une configuration et administration centralisée depuis une interface HTTP, des règles de conversion basées sur les types MIME et des fichiers de description d'imprimante standards (**PPD**, *PostScript Printer Description*).
- CUPS reprend les commandes System V et BSD déjà abordées pour plus de simplicité.
- Les traces des impressions sont disponibles au format **CLF** (*Common Log Format*) de serveur Web et exploitables par les mêmes outils.
- CUPS est capable d'interagir avec les serveurs d'impression LPD pour garder une compatibilité ascendante.
- CUPS dispose de sa propre API permettant de créer des interfaces utilisateur pouvant s'intégrer dans des environnement graphiques ou des interfaces d'administration.
- Les pools d'impression permettent la redirection automatique des tâches.
- L'authentification est possible par utilisateur, hôte ou certificat numérique.
- Le service d'impression se nomme **cupsd**.

Il n'y a pas besoin d'outils graphiques pour administrer un serveur CUPS. CUPS propose une

interface d'administration WEB directement accessible depuis le port 631 du serveur. L'interface fonctionne avec n'importe quel navigateur HTTP.

```
http://localhost:631
```

Le fichier de configuration est **`/etc/cups/cupsd.conf`**. L'équivalent à /etc/printcap est présent dans **`/etc/cups/printers.conf`**.

On peut aussi utiliser l'outil graphique **`system-config-printer`**.

6.3 Automatisation avec cron

6.3.1 Formalisme

Le démon **`cron`** permet la programmation d'événements à répétition. Il fonctionne à l'aide d'une table, appelée une **`crontab`**. C'est un fichier texte simple, édité avec un simple éditeur, par défaut vi. Pour modifier sa cron table personnelle on utilise la commande **`crontab`** pour éditer la table, avec le paramètre « **`-e`** ». Le format est le suivant :

Minutes	*Heures*	*Jour du mois*	*Mois*	*Jour de la semaine*	*Commande*
1	**2**	**3**	**4**	**5**	**6**

On utilise le format suivant pour les valeurs périodiques :

- Une valeur pour indiquer quand il faut exécuter la commande. Ex : la valeur 15 dans le champ minute signifie la quinzième minute
- Une liste de valeurs séparées par des virgules. Ex : 1,4,7,10 dans le champ Mois pour Janvier, Avril, Juillet, Octobre
- Une intervalle de valeurs. Ex : 1-5 dans le champ Jour de la Semaine indique du lundi (1) au vendredi (5). Le 0 est le dimanche et le 6 le samedi.
- Le caractère * pour toutes les valeurs possibles. Ex : * dans le champ Jour du mois indique tous les jours du ou des mois.

Exemples :

Exécution de df tous les jours, toute l'année, tous les quarts d'heure :

```
0,15,30,45 * * * * df > /tmp/libre
```

Exécution d'une commande tous les jours ouvrables à 17 heures :

```
0 17 * * 1-5 fin_travail.sh
```

- Pour lister les crontabs actives :

  ```
  crontab -l
  ```

- Pour supprimer la crontab active :

  ```
  crontab -r
  ```

- Pour éditer la crontab d'un utilisateur particulier :

  ```
  crontab -u user
  ```

Les fichiers crontabs sont sauvés dans **/var/spool/cron**.

6.3.2 crontab système

La configuration crontab générale pour le système est dans **/etc/crontab**.

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run-parts
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

On voit par exemple que tous les jours à 4h02 du matin, « **run-parts /etc/cron.daily** » est exécuté. Le script « **run-parts** » prend comme paramètre un répertoire et exécute tous les programmes présents dans ce répertoire.

```
$ ls cron.daily/
00-logwatch  0anacron   makewhatis.cron  slocate.cron
00webalizer  logrotate  rpm              tmpwatch
```

Parmi les programmes exécutés, on remarque **logrotate** qui permet de faire des sauvegardes et renommages des fichiers logs et des journaux du système afin que ceux-ci ne deviennent pas inexploitables à cause de leur taille. Le programme **tmpwatch** est chargé de nettoyer le système des fichiers inutilisés (dans /tmp par exemple).

Enfin, le répertoire **/etc/cron.d** contient des crontabs supplémentaires.

6.4 Journal système

Les services **syslogd** et **klogd** permettent de recevoir, traiter et stocker des messages d'erreurs émis par le noyau ou certains démons. **Syslogd** traite le journal système et **klogd** les événements du noyau. Un fichier de configuration **/etc/syslog.conf** permet de définir l'origine, l'importance et la destination de chaque message, sous forme de deux champs.

1) L'origine définit en fait un ensemble de **systèmes** et de **sous-systèmes** (noyau, services). La liste, extensible, est composée à l'origine des éléments suivants. L'étoile « * » définit l'ensemble des sous-systèmes.

Sous-système	*Signification*
auth	service de sécurité et d'authentification
cron	service cron
daemon	les démons du système
kern	le noyau
lpr	le service d'impression
mail	la messagerie
news	le réseau

Sous-système	*Signification*
syslog	syslog lui-même
user	messages des processus utilisateurs
uucp	Unix to Unix CoPy

2) L'importance ou « **niveau** » définit le niveau de criticité du message. L'étoile « * » définit l'ensemble de tous les niveaux.

Niveau	*Signification*
emerg	Le système est inutilisable
alert	Une intervention immédiate est indispensable
crit	Erreur critique pour le sous-système
err	Erreur de fonctionnement
warning	Avertissement
notice	Évènement normal méritant d'être signalé
info	Pour information seulement
debug	Message envoyé pour la mise au point
none	Ignorer les messages

3) La destination ou « **action** » peut être un fichier, un mail à un utilisateur, la console, une liste d'utilisateurs, ...

Les messages syslog sont inscrits dans les fichiers **`/var/log/messages`** et **`/var/log/syslog`** ou dans tout autre fichier paramétré dans **`/etc/syslog.conf`**.

On peut envoyer des messages à **`syslogd`** par la commande « **`logger`** ».

7 Services et modules noyau

7.1 Présentation

Le noyau est le cœur du système d'exploitation Linux. Linux en tant que tel est uniquement le nom du noyau développé à l'origine par Linus Torvalds. Le système d'exploitation Linux est composé du noyau et des outils d'exploitation de base.

Le noyau de Linux est libre. Ses sources sont disponibles. Il est donc possible de le recompiler pour l'adapter finement à ses besoins, de le modifier, d'y rajouter des extensions.

Le noyau Linux fait partie de la famille des noyaux monolithiques. C'est à dire que toutes ses fonctionnalités et composants sont regroupés dans un programme unique. Cependant depuis la version 2.0 le noyau est modulaire.

Le noyau est appelé **`kernel`**. Il est présent dans **`/boot`** et son nom par convention commence souvent par « **`vmlinuz-X.Y.Z.p-Vtxt`** ».

On obtient la version du noyau avec la commande **`uname`**.

```
$ uname -r
2.4.9-e.57smp
```

Les lettres ont une signification particulière.

- **`X`** : version majeure du noyau. Entre la version 1 et la version 2, le passage au modulaire a été déterminant, ainsi que la réimplémentation de la couche réseau.
- **`Y`** : Une valeur paire représente une branche stable du noyau. Une version impaire représente une branche de développement (attention !). Chaque incrément pair (0,2,4,6) représente une évolution importante du noyau. Note : la version 2.6 ne dispose pas encore de branche de développement car il évolue trop vite. Les développeurs ont décidé d'implémenter leurs nouveautés directement dans la version stable.
- **`Z`** : Version mineure du noyau. Quand un lot de modifications par rapport à une version précédente nécessite la diffusion d'un nouveau noyau, alors on incrémente ce chiffre. Par exemple, un lot regroupant une modification du système son (Alsa qui passe de 1.0.8 à 1.0.9), du système de fichier (ajout de ReiserFS 4) et ainsi de suite...
- **`p`** : version corrigée ou intermédiaire présente depuis la version 2.6. Quand le noyau nécessite une mise à jour mineure (correction d'un ou deux bugs, etc) mais pas ou peu d'ajouts de fonctionnalités, on incrémente cette valeur.
- **`V`** : comme pour les packages, version propre à l'éditeur de la distribution.
- **`txt`** : on rajoute parfois un texte pour donner des précisions sur le noyau. Par exemple, smp indique un noyau multiprocesseurs.

7.2 Gestion des modules

Les composants de base (scheduler, gestion de la mémoire, des processus, API, etc) sont toujours présents au sein d'un programme unique. Mais certains pilotes de périphériques, systèmes de fichiers, extensions, protocoles réseaux, etc peuvent être présents sous forme de modules. Les modules communiquent avec le noyau via une API commune. Ils s'exécutent dans l'espace du noyau. Ils sont paramétrables. Ils peuvent être chargés et déchargés à la demande évitant ainsi un redémarrage de la machine. L'ajout d'un nouveau module (depuis ses sources par exemple) ne

nécessite pas de redémarrage.

Les modules sont présents dans **/lib/modules/`uname -r`**.

```
$ cd /lib/modules/`uname -r`
$ pwd
/lib/modules/2.4.9-e.57smp
```

Le contrôle des modules se fait avec les commandes :

- **lsmod** : liste les modules actuellement chargés, avec leurs dépendances éventuelles
- **depmod** : mise à jour des dépendances entre les modules en modifiant le fichier **modules.dep**. (commande **depmod -a**).
- **insmod** : charge le module donné, sans gérer les dépendances
- **rmmod** : décharge le module donné
- **modprobe** : charge le module donné ainsi que toutes ses dépendances et des paramètres contenus dans **/etc/modprobe.conf**. Le paramètre « **-r** » permet de décharger un modules et ceux qui en dépendent (s'ils ne sont pas utilisés).
- **modinfo** : informations détaillées sur le module, ainsi que les paramètres qu'il accepte.

```
$ /sbin/lsmod
Module                  Size  Used by    Not tainted
autofs                 13796   0 (autoclean) (unused)
tulip                  43456   1
eepro100               21968   1
ext3                   71264  11
jbd                    55700  11 [ext3]
sym53c8xx              67940  12
sd_mod                 14048  12
scsi_mod              128156   2 [sym53c8xx sd_mod]

# modprobe quickcam

# rmmod quickcam

# modprobe -r jdb
```

La configuration des modules est placée dans **/etc/modprobe.conf**. Dans ce fichier on peut définir des alias de modules (très pratique pour les cartes réseau), passer des options aux modules, ajouter des actions au chargement et déchargement d'un module.

Exemple : Nous avons deux cartes réseaux. Le pilote de notre première carte est contenu dans le module « **e1000** ». Nous voulons qu'il soit chargé en tant que « **eth0** ». Le pilote de la seconde carte est dans le module « **airo** » en tant que « **eth1** ». nous allons aussi passer des paramètres au module « **quickcam** » dont nous avons récupéré les informations avec **modinfo**.

```
# cat /etc/modprobe.conf
...
alias eth0 e1000
alias eth1 airo
options quickcam compress=1 skip=0
...

# modprobe eth0
```

Certains modules peuvent être nécessaires au démarrage de la machine, notamment pour monter un système de fichier. Comment monter la partition racine en ext3 alors que le module gérant ce type de système de fichier est sur cette partition (et pas en dur dans le noyau) ? Ces modules sont placés dans une « *image de disque mémoire initiale* » ou « **initrd** ». Ces fichiers compressés sont chargés au démarrage en mémoire et sont des ramdisks. Ils contiennent des instructions et modules qui sont chargés au démarrage.

```
$ ls /boot/init*
/boot/initrd-2.4.9-e.57.img

$ file /boot/initrd-2.4.9-e.57.img
/boot/initrd-2.4.9-e.57.img: gzip compressed data, from Unix, max compression
```

7.3 /proc et /sys

/proc et **/sys** sont des systèmes de fichiers virtuels contenant des informations sur le noyau en cours d'exécution. La version 2.4 du noyau ne connaît que **/proc** où toutes les informations sont regroupées. La version 2.6 du noyau a modifié la fonction de **/proc** pour en déléguer une partie à **/sys**. Par souci de compatibilité (mesure temporaire) le contenu de **/proc/sys** reflète le contenu de **/sys**.

S'agissant de systèmes de fichiers virtuels, ils ne prennent aucune place ni en mémoire, ni sur un disque quelconque. Il ne faut pas se laisser berner par la taille des pseudo-fichiers contenus dedans. Ne tentez pas de supprimer **/proc/kcore** pour gagner de la place ! Tous ces fichiers (ou presque) peuvent être lus et affichés directement.

Les fichiers de **/proc** vous donneront énormément d'informations sur le système :

- **interrupts** : les paramètres IRQ
- **cpuinfo** : détails sur vos processeurs
- **dma** : les paramètres DMA
- **ioports** : les ports mémoires E/S
- **devices** : les périphériques présents
- **meminfo** : l'état global de la mémoire
- **loadavg** : la charge du système
- **uptime** : uptime du système, attente
- **version** : détails de la version de Linux
- **modules** : identique au résultat de lsmod
- **swaps** : liste et état des partitions d'échange
- **partitions** : liste et état des partitions connues du système
- **mounts** : montages des systèmes de fichiers
- **pci** : détails du bus PCI

```
$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 8
model name      : Pentium III (Coppermine)
stepping        : 6
cpu MHz         : 996.895
cache size      : 256 KB
physical id     : 1360587528
siblings        : 1
fdiv_bug        : no
hlt_bug         : no
f00f_bug        : no
coma_bug        : no
fpu             : yes
fpu_exception   : yes
```

```
cpuid level     : 2
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 mmx fxsr
sse
bogomips        : 1985.74

$ cat /proc/version
Linux version 2.4.9-e.57smp (bhcompile@tweety.build.redhat.com) (gcc version 2.96 20000731 (Red Hat
Linux 7.2 2.96-129.7.2)) #1 SMP Thu Dec 2 20:51:12 EST 2004
```

/proc contient des sous-répertoires qui regroupent des informations par thème.

- **/proc/scsi** : informations sur le bus SCSI
- **/proc/ide** : informations sur le bus IDE
- **/proc/net** : informations sur le réseau
- **/proc/sys** : paramètres et configuration dynamique du noyau
- **/proc/<PID>** : informations sur le processus PID.

Le répertoire **/proc/sys** (**/sys** depuis le noyau 2.6) est différent des autres car son contenu peut être modifié et les modifications sont prises en compte directement par le noyau avoir à redémarrer la machine. Par exemple nous allons activer le forwarding IP et passer le nombre de handles de fichiers de 8192 à 16384.

```
echo "1" > /proc/sys/net/ipv4/ip_forward
echo "16384" > /proc/sys/fs/file-max
```

Les valeurs modifiées ne sont pas enregistrées. En cas de redémarrage il faut recommencer. Le fichier **rc.sysinit** appelle la commande **sysctl** qui agit sur ces paramètres. Pour que les valeurs restent permanentes (remises en place à chaque démarrage) il faut modifier le fichier **/etc/sysctl.conf**. On peut appeler les modifications à la main.

```
# sysctl -e -p /etc/sysctl.conf

# sysctl -a
...
dev.raid.speed_limit_max = 100000
dev.raid.speed_limit_min = 100
net.token-ring.rif_timeout = 60000
net.ipv4.conf.eth1.arp_filter = 0
net.ipv4.conf.eth1.tag = 0
net.ipv4.conf.eth1.log_martians = 0
net.ipv4.conf.eth1.bootp_relay = 0
net.ipv4.conf.eth1.proxy_arp = 0
net.ipv4.conf.eth1.accept_source_route = 1
net.ipv4.conf.eth1.send_redirects = 1
net.ipv4.conf.eth1.rp_filter = 1
net.ipv4.conf.eth1.shared_media = 1
net.ipv4.conf.eth1.secure_redirects = 1
...
```

8 Partitionnement avancé RAID

8.1 Définitions

Le **RAID** (*Redundant Array of Inexpansive Disks*) a été défini par l'université de Berkeley en 1987 dans le double but de réduire les coûts et d'augmenter la fiabilité du stockage des données. Le but est de combiner plusieurs petits disques physiques indépendants en une **matrice** (array : tableau, ensemble, rangée, matrice) de disques dont la capacité dépasse celle du **SLED** (*Single Large Expensive Drive*). Une matrice apparaît comme une unité logique de stockage unique.

Le **MTBF** (*Mean Time Between Failure*) de l'ensemble est égal au MTBF d'un disque individuel divisé par le nombre de disques dans l'ensemble et donc théoriquement une solution RAID peut être inadaptée pour des tâches critiques. Heureusement le RAID peut être tolérant aux fautes en stockant de manière redondante ses informations selon plusieurs méthodes :

- **RAID-0** : Appelé « **stripe mode** ». Deux disques au moins forment un seul unique volume. Les deux disques en en principe la même taille. Chaque opération de lecture/écriture sera divisée sur chaque disque. Par exemple, 4 ko seront écrits sur le disque 0, 4 ko sur le disque 1, 4 ko sur le disque 2, puis 4 ko sur le disque 0, etc. Ainsi les performances sont accrues puisque les lectures et écritures sont effectuées en parallèle sur les disques. Si N est le nombre de disques et P la vitesse de transfert, in est en principe proche de **N*P mbps**. Le RAID-0 n'a aucune redondance. En cas de panne d'un des disques, il est probable que l'ensemble des données soient perdues.
- **RAID-1** : Appelé « **mirroring** ». Premier mode redondant. Il peut être utilisé à partir de deux disques ou plus avec d'éventuels disques de secours (**Spare Disk**). Chaque information écrite sur un disque est dupliquée sur les autres. Si N-1 disques du RAID viennent à tomber les données restent intactes. Si un disque de secours est présent en cas de panne, il est automatiquement reconstruit et prend la place du disque défaillant. Les performances en écriture peuvent être mauvaises : écriture sur N disques en même temps, risquant de saturer le contrôleur disque et le bus. Les performances en lecture sont bonnes, car RAID emploie un algorithme qui peut lire les données sur chaque disque (puisqu'ils sont identiques).
- **RAID-5** : RAID avec bande de parité redistribuée. C'est le mode le plus utilisé car c'est celui qui offre le meilleur compromis entre le nombre de disques, l'espace disponible et la redondance. Il faut au moins trois disques avec d'éventuels disques de secours. La parité est présente sur chacun des disques. La taille finale est celles de N-1 disques. Le RAID-5 survit à une panne de disque. Dans ce cas, si un disque de secours est présent il sera automatiquement reconstruit. Les performances sont bonnes en lecture et écriture (lecture : comme du RAID-0, écriture, ça dépend) suivant l'algorithme employé et la mémoire de la machine.

8.2 Précautions et considérations d'usage

8.2.1 Disque de secours

Un disque de secours (**Spare Disk**) ne fait pas partie intégrante d'une matrice RAID tant qu'un disque ne tombe pas en panne. Si ça arrive, le disque est marqué défectueux et le premier disque Spare prend le relais. Quoi qu'il arrive, il faut tout de même le plus vite possible changer le disque défaillant et reconstruire le RAID.

8.2.2 Disque défectueux

Un disque défectueux (**`Faulty Disk`**) est un disque qui a été reconnu avoir eu une défaillance ou en panne par RAID. Dans ce cas il est indiqué défectueux, RAID utilise le premier disque Spare pour reconstruire sa matrice. Les disques Faulty appartiennent toujours à la matrice mais sont désactivés.

8.2.3 boot

La partition de boot (celle qui contient le noyau, la configuration du bootloader, les fichiers images de disques) ne doit pas être placée dans une matrice RAID : le bootloader est incapable de monter des partitions RAID.

8.2.4 Swap

On peut installer un swap sur du RAID mais ce n'est en principe pas utile dans les cas courants. En effet Linux est capable d'équilibrer l'utilisation du swaps sur plusieurs disques/partitions seuls. Dans ce cas on déclare n swaps dans /etc/fstab avec la même priorité.

```
/dev/sda2       swap            swap    defaults,pri=1    0 0
/dev/sdb2       swap            swap    defaults,pri=1    0 0
/dev/sdc2       swap            swap    defaults,pri=1    0 0
```

Cependant en cas de besoin de haute disponibilité, le swap sur le RAID est possible.

8.2.5 Périphériques

Une matrice RAID est reconnue par le système comme un périphérique de type bloc, comme n'importe quel disque physique. Ainsi, un RAID peut être constitué avec des disques, des partitions (généralement, on créé une unique partition sur chaque disque). On se fiche du bus : on peut construire une matrice RAID avec des disques SCSI et IDE mélangés. De même on peut construire du RAID sur d'autres matrices RAID, par exemple du RAID-0+1 (2x2 disques en RAID-1, les deux matrices résultantes en formant une nouvelle en RAID-0). Les périphériques RAID sont sous la forme :

```
/dev/md0
/dev/md1
...
```

8.2.6 IDE

Si les disques IDE ont longtemps été le SCSI du pauvre (matériel de moins bonne qualité, lenteur, fiabilité) ce n'est plus vraiment le cas où les derniers disques ont la même mécanique et où seul le bus change. On peut donc monter pour pas trop cher des configurations RAID en IDE. Cependant une règle est à retenir :

UN SEUL DISQUE IDE PAR BUS IDE

En pratique, cela correspond à un disque par câble, sans rien d'autre. En effet, en principe un bus IDE survit à la déficience d'un disque mais il arrive régulièrement que le bus IDE devienne lui-même défectueux, entraînant la perte du second disque présent sur le bus et donc la perte de la matrice RAID. L'achat de cartes IDE supplémentaires (bas prix) permet de compenser le problème de fiabilité (deux disques par carte).

8.2.7 Hot Swap

- **IDE** : **NE JAMAIS DEBRANCHER A CHAUD UN DISQUE IDE** ! C'est le meilleur moyen de 1) détruire le disque, si ce n'était pas encore le cas 2) détruire le contrôleur IDE (et donc éventuellement la carte mère ou additionnelle). L'IDE n'est pas prévu pour.
- **SCSI** : Les contrôleurs SCSI ne sont pas prévus pour le Hot Swap mais devraient en théorie tout de même fonctionner, si le disque est identique physiquement et logiquement.
- **SATA** : Le SATA est reconnu comme du SCSI. La spécification SATA supporte le Hot Swap. Seulement, la plupart des contrôleurs actuels implémentent mal ou pas du tout cette possibilité, d'où les risques de plantages ou de griller son contrôleur.
- **SCA** : Ce sont des disques SCSI spécifiques. Voir le document « Software RAID Howto ».

8.3 RAID avec mdadm

8.3.1 Préparation

L'outil **mdadm** remplace les outils **raidtools** des anciennes distributions Linux. Cet outil unique est plus simple et permet d'effectuer l'ensemble des opérations. Son fichier de configuration est **/etc/mdadm.conf**.

Afin de créer des matrices RAID, il faut que les partitions qui vont servir à créer la matrice soient de type **0xFD** (**Linux RAID autodetect**). Les partitions doivent être logiquement sur des disques différents, mais pour des tests le support RAID autorise des partitions sur le même disque (ce qui est bien pratique pour nos tests). Dans ce cas on veillera à ce que les partitions disposent de la même taille.

8.3.2 Création

8.3.2.1 RAID-0

Soient deux partitions **/dev/sdb1** et **/dev/sdc1**. On veut créer une partition RAID-0, assemblage de ces deux partitions.

```
mdadm --create /dev/md0 --level=raid0 --raid-devices=2 /dev/sdb1 /dev/sdc1
```

- **--create** : nous voulons créer un RAID
- **/dev/md0** : nom du fichier périphérique de type bloc représentant la matrice RAID
- **--level** : type de RAID à créer. **0**, **raid0** et **stripe** pour du raid0. Note : linear n'est pas du raid0 (remplissage au fur et à mesure).
- **--raid-devices** : nombre de partitions utilisées pour créer la matrice.
- **/dev/sdb1**, **/dev/sdc1** : partitions constituant la matrice, autant que indiqués dans --raid-devices.

Il ne reste plus qu'à formater le disque RAID :

```
mkfs -t ext3 /dev/md0
```

8.3.2.2 RAID-1

C'est le même principe. Nous allons cette fois rajouter une partition de secours /dev/sdd1.

```
mdadm --create /dev/md0 --level=raid0 --raid-devices=2 /dev/sdb1 /dev/sdc1 /
--spare-devices=1 /dev/sdd1
```

- **`--level :1`**, **`mirror`** ou **`raid1`** sont de bonnes valeurs pour un raid5.
- **`--spare-devices`** : nombre de disques de secours à utiliser
- **`/dev/sdd1`** : partitions constituant les disques de secours, autant que indiqués dans –spare-devices.

Puis on formate:

```
mkfs -t ext3 /dev/md2
```

8.3.2.3 RAID-0+1

Il nous faut au moins quatre partitions. Nous devons créer deux matrices RAID-1 que nous allons regrouper en une matrice RAID-0.

```
mdadm --create /dev/md0 --level=raid1 --raid-devices=2 /dev/sdb1 /dev/sdc1
mdadm --create /dev/md1 --level=raid1 --raid-devices=2 /dev/sdd1 /dev/sde1
mdadm --create /dev/md2 --level=raid0 --raid-devices=2 /dev/md0 /dev/md1
```

Puis on formate:

```
mkfs -t ext3 /dev/md2
```

8.3.2.4 RAID 5

Nous allons utiliser trois disques de données /dev/sdb1 /dev/sdc1 et /dev/sdd1, et un disque de secours /dev/sde1.

```
mdadm --create /dev/md0 --level=raid5 --raid-devices=3 /dev/sdb1 /dev/sdc1 /dev/sdd1
--spare-devices=1 /dev/sde1
```

Puis on formate:

```
mkfs -t ext3 /dev/md2
```

8.3.3 Sauver la configuration

Pour faciliter la tâche de l'outil **`mdadm`**, on créé le fichier de configuration **`/etc/mdadm.conf`**. Ce fichier peut être créé à la main mais l'outil **`mdadm`** sait le générer. Il est préférable de le créer APRES la création des matrices RAID.

```
echo "DEVICE partitions" >> /etc/mdadm.conf
mdadm --detail --scan >> /etc/mdadm.conf
```

8.3.4 Etat du RAID

Le fichier virtuel **`/proc/mdstat`** contient des informations sur le RAID. C'est ici qu'on peut voir le détail d'un RAID, notamment si un des volumes de la matrice est **`Faulty`**.

```
Personalities : [raid1]
md0 : active raid1 hda10[2] hda9[1] hda8[0]
      104320 blocks [2/2] [UU]
```

La commande **`watch`** permet de vérifier un état en continu :

```
watch cat /proc/mdstat
```

On peut aussi utiliser **mdadm** avec le paramètre **--detail** :

```
# mdadm --detail /dev/md0
/dev/md0:
        Version : 00.90.01
  Creation Time : Mon Jan 23 22:10:20 2006
     Raid Level : raid1
     Array Size : 104320 (101.88 MiB 106.82 MB)
    Device Size : 104320 (101.88 MiB 106.82 MB)
   Raid Devices : 2
  Total Devices : 3
Preferred Minor : 1
    Persistence : Superblock is persistent

    Update Time : Mon Jan 23 22:13:06 2006
          State : clean
 Active Devices : 2
Working Devices : 3
 Failed Devices : 0
  Spare Devices : 1


    Number   Major   Minor   RaidDevice State
       0       3        8        0      active sync   /dev/hda8
       1       3        9        1      active sync   /dev/hda9
       2       3       10       -1      spare   /dev/hda10
           UUID : 90e838b5:936f18c7:39f665d3:d9dad1a9
         Events : 0.4
```

On remarque avec cette dernière commande plus de détails notamment quels sont les disques **spare** et **faulty**.

8.3.5 Simuler une panne

Nous allons simuler une panne sur **/dev/hda8** :

```
# mdadm /dev/md0 -f /dev/hda8
mdadm: set /dev/hda8 faulty in /dev/md0
```

Regardons l'état dans **/proc/mdstat** durant l'exécution :

```
md0 : active raid1 hda10[2] hda9[1] hda8[0](F)
      104320 blocks [2/1] [U_]
      [=>..................]  recovery =  8.8% (9216/104320) finish=0.1min speed=9216K/sec
```

On remarque que **hda8** à un « **(F)** » qui est apparu indiquant un disque **Faulty**. On voit aussi que sur les deux disques un est en panne. On remarque aussi que le RAID reconstruit sa matrice avec le **spare disk**. Après l'exécution :

```
md0 : active raid1 hda10[1] hda9[0] hda8[2](F)
      104320 blocks [2/2] [UU]
```

Le RAID est reconstruit et fonctionne à merveille.

```
# mdadm --detail /dev/md0
...
          State : clean
 Active Devices : 2
Working Devices : 2
 Failed Devices : 1
  Spare Devices : 0


    Number   Major   Minor   RaidDevice State
       0       3        9        0      active sync   /dev/hda9
       1       3       10        1      active sync   /dev/hda10
       2       3        8       -1      faulty   /dev/hda8
...
```

Le disque **Faulty** est bien **/dev/hda8**. **/dev/hda10** a pris sa place en tant que disque de

secours. Ainsi, le disque de secours devient un disque RAID de la matrice.

8.3.6 Remplacer un disque

Puisque **/dev/hda8** est en panne nous allons le remplacer. Retirons-le avec **-r** (**--remove** fonctionne aussi) :

```
# mdadm /dev/md0 -r /dev/hda8
mdadm: hot removed /dev/hda8

# cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 hda10[1] hda9[0]
      104320 blocks [2/2] [UU]
```

On constate que **hda8** a disparu. A ce moment, on éteint la machine, on remplace le disque défaillant. On rallume la machine, puis on repartitionne le disque correctement. Il n'y a plus qu'à rajouter le disque réparé dans la matrice RAID avec **-a** (**--add**) :

```
# mdadm /dev/md0 -a /dev/hda8
mdadm: hot added /dev/hda8

# cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 hda8[2] hda10[1] hda9[0]
      104320 blocks [2/2] [UU]
```

Le disque **hda8** est revenu. Voyons le détail :

```
# mdadm --detail /dev/md0
...
         State : clean
 Active Devices : 2
Working Devices : 3
 Failed Devices : 0
  Spare Devices : 1


    Number   Major   Minor   RaidDevice State
       0       3        9        0      active sync   /dev/hda9
       1       3       10        1      active sync   /dev/hda10
       2       3        8       -1      spare   /dev/hda8
...
```

Le disque **/dev/hda8** a été remis, et est devenu le nouveau disque de secours !

8.3.7 Arrêt / Relance manuels

On peut arrêter ponctuellement une matrice RAID avec **-S** (**--stop**) APRES avoir démonté le périphérique :

```
mdadm --stop /dev/md0
```

On peut redémarrer une matrice RAID avec **-As** (**--assemble –scan**). Cela implique que le fichier /**etc/mdadm.conf** est correctement renseigné (**--scan** recherche les informations dedans).

```
mdadm --assemble --scan /dev/md0
```

Si le RAID ne redémarre pas, vous pouvez tenter avec -R (--run) : il est probable qu'il manque un disque ou qu'une reconstruction en cours n'est pas terminée :

```
mdadm --run /dev/md0
```

9 TCP/IP

9.1 Bases

L'origine de **TCP/IP** provient des recherches du **DARPA** « *Defense Advanced Research Project Agency* » qui débutent en 1970 et débouchent sur **ARPANET**. Dans les faits, le DARPA a financé l'**université de Berkeley** qui a intégré les protocoles de base de TCP/IP au sein de son système **UNIX BSD 4**.

TCP/IP s'est popularisé grâce à son interface générique de programmation d'échanges de données entre les machines d'un réseau, les primitives « **sockets** », et l'intégration de protocoles applicatifs. Les protocoles de TCP/IP sont supervisés par **l'IAB** « Internet Activities Board », lui-même supervisant deux autres organismes :

- **L'IRTF** « *Internet Reseach Task Force* » qui est responsable du développement des protocoles.
- **L'IETF** « *Internet Engineering Task Force* », qui est responsable du réseau Internet.

Les adresses réseau sont distribuées par le **NIC** « *Network Information Center* », en France **l'INRIA.** L'ensemble des protocoles de TCP/IP est décrit dans les documents **RFC** « *Request For Comments* ». (voir le RFC 793).

- La couche inférieure est **IP** : **Internet Protocol**
- La couche de transport est **TCP** : **Transmission Control Protocol**, ou **UDP** : **User Datagram Protocol**.
- Les couches supérieures sont les couches des protocoles applicatif, par exemple :
  - **NFS** : « *Network File System* », partage de fichiers à distance
  - **DNS** : « *Domain Name System* », association hôte<->IP
  - **FTP** : « *File Transfert Protocol* », transfert de fichiers
  - **TELNET** : émulation d'un terminal de type texte
  - ...

La version du protocole IP représenté est le la V4. Le futur, déjà présent, est le protocole IPV6. Compatible IPV4, il propose un adressage 64 bits (8 octets) permettant d'étendre les capacités du réseau notamment en matière de taille et d'adressage.

9.2 adressage

9.2.1 classes

Il est important de savoir avant l'installation dans quel type de réseau doit s'intégrer le nouveau serveur, TCP-IP bien sur, mais il faut déjà lui réserver une **adresse** IP, un **hostname** (nom de machine réseau), connaître les diverses passerelles, le **nom de domaine**, la **classe** utilisée et le masque de sous-réseau **netmask** ...

Un petit rappel sur les classes IP. Une adresse IP est définie sur 32 bits et représentée par quatre nombres séparés par des points : **n1.n2.n3.n4**. Cette adresse est constituée de deux parties qui définissent l'adresse réseau et l'hôte dans le réseau.

On distingue, suivant les cas, quatre ou cinq classes d'adresses : A, B, C, D et E, mais seules les trois premières nous intéressent.

Légende : N et h sont des bits, N identifiant du réseau h identifiant de la machine

```
Classe A : 0NNNNNNN hhhhhhhh hhhhhhhh hhhhhhhh soit 1.x.x.x à 126.x.x.x
n1 est compris entre 1 et 126
16777214 hôtes, 127 réseaux

Classe B : 10NNNNNN NNNNNNNN hhhhhhhh hhhhhhhh soit de 128.0.x.x à 191.255.x.x
n1 est compris entre 128 et 191
65534 hôtes, 16382 réseaux

Classe C : 110NNNNN NNNNNNNN NNNNNNNN hhhhhhhh soit de 192.0.0.x à 223.255.255.x
n1 est compris entre 192 et 223
254 hôtes, 2097150 réseaux

Classe D : Commence par 1110, pour la multidiffusion IP
Classe E : Commence par 1111, pour expérimentation
```

On remarque qu'il existe des adresses d'hôtes qui ne peuvent pas être exploités. Par exemple dans la classe C on ne peut avoir que 254 hôtes, alors que l'identifiant de la machine est codé sur 8 bits (donc 25). C'est que l'adresse 0 représente l'adresse du réseau, et l'adresse 255 celle du **broadcast** (multi-diffusion).

9.2.2 Sous-réseaux

De plus, il est possible de découper ces réseaux et sous-réseaux à l'aide de masques permettant un découpage plus fin des adresses. Un **`netmask`** est un masque binaire qui permet de séparer immédiatement l'adresse du réseau et du sous-réseau de l'adresse de l'hôte dan l'adresse IP globale. Les masques prédéfinis sont :

- **`Classe A : 255.0.0.0`**
- **`Classe B : 255.255.0.0`**
- **`Classe C : 255.255.255.0`**

Pour communiquer directement entre eux les hôtes doivent appartenir à un même réseau ou sous-réseau. Calculer un sous-réseau est assez simple. Voici un exemple pour un réseau de classe C.

- **`Réseau : 192.168.1.0`**
- **`Adresse de réseau : 192.168.1.255`**
- **`Masque de réseau : 255.255.255.0`**

1. Pour calculer le masque de sous-réseau, nous devons tout d'abord déterminer combien de machines nous souhaitons intégrer dans celui-ci. Un réseau de classe C permet d'intégrer 254 machines (0 et 255 étant réservés). Imaginons que nous souhaitons créer des réseaux contenant 60 machines. Ajoutons 2 à cette valeur pour les ardresses réservées (adresse du sous-réseau et adresse de broadcast) ce qui donne 62.
2. Une fois le nombre de machines, trouvons la puissance de deux exacte ou juste supérieure au nombre trouvé. 2 puissance 6 donne 64.
3. Ecrivons le masque en binaire Plaçons tous les bits du masque de réseau de classe C à 1, et plaçons à 0 les 6 premiers bits du masque correspondant à la partie machine : 11111111 11111111 11111111 11000000
4. Convertissons ce masque en décimal : 255.255.255.192, et calculons l'ensemble des sous-réseaux possibles. Comme nous sommes dans un réseau de classe C, nous pouvons encore faire varier les deux derniers bits de la partie machine :

- 00xxxxxx : 255.255.255.0
- 01xxxxxxx : 255.255.255.64
- 10xxxxxx : 255.255.255.128
- 11xxxxxx : 255.255.255.192

5. Au final, on obtient quatre sous réseaux de 62 machines, soit 248 machines. On tombe bien sur 256 si on rajoute les quatre adresses de broadcast et les quatre adresses de réseau.

9.2.3 Routage

Le masque de réseau permet de déterminer si une machine destinataire est sur le même réseau que vous ou non. Il faut indiquer le chemin que doivent prendre les paquets IP pour rejoindre leur destination. Si notre machine est un poste client disposant d'une seule carte réseau et que ce réseau ne comporte qu'un seul routeur (cas classique d'une connexion vers Internet) alors on doit créer deux routes. La première est celle indiquant quelle carte réseau doivent emprunter les paquets pour accéder au reste du réseau (au sous-réseau), la seconde quelle route doivent emprunter les paquets pour sortir du réseau. Généralement, on parle de route par défaut quand un seul routeur est présent.

- Vers réseau1 -> utiliser interface réseau gauche
- Vers réseau2 -> utiliser interface réseau droite
- Vers autres -> utiliser interface réseau droite vers routeur1

Exemple : Réseau1 de classe C 192.168.1.0 sur eth0, Réseau2 de classe B 172.16.0.0 sur eth1, adresse de routeur 192.168.1.254.

```
Réseau          Masque          Interface      Passerelle
192.168.1.0     255.255.255.0   eth0           eth0
172.16.0.0      255.255.0.0     eth1           eth1
0.0.0.0         0.0.0.0         eth0           192.168.1.254
```

Tous les paquets réseaux vers 192.168.1.0 transiteront par eth0. Tous les paquets à destination de 172.16.0.0 transiteront par eth1. Par défaut, tous les autres paquets pour les réseaux non spécifiés transiteront par eth0 et seront traités par la passerelle 192.168.1.254 qui routera les paquets.

9.3 Configuration

9.3.1 Outils de RHEL

Red Hat propose des outils pour configurer le réseau de base sans passer par la manipulation des fichiers de configuration. Le programme **`netconfig`** est en mode texte et permet de mettre en place la configuration TCP/IP de base (IP statique ou dynamique, routeur, nom d'hôte, serveur de noms).

```
# netconfig -device eth0
```

La commande graphique **`system-config-network`** ou **`neat`** lance une interface plus complète.

Note : l'utilisation du WIFI est possible avec une RHEL mais nécessitera une configuration manuelle.

9.3.2 Interfaces réseaux

La configuration de base d'une interface réseau se fait à l'aide de la commande **`ifconfig`**. Cependant la plupart des distributions utilisent des scripts d'administration et des fichiers de configuration qui simplifient énormément les choses car ils configurent à la fois l'interface réseau et

les routes.

Configuration de eth0 pour l'adresse de classe C 192.168.1.2

```
ifconfig eth0 inet 192.168.1.2 netmask 255.255.255.0
ifconfig eth0 192.168.1.2
```

Activation de l'interface réseau eth0

```
ifconfig eth0 up
```

Arrêt de l'interface réseau eth0

```
ifconfig eth0 down
```

Affichage des informations de eth0

```
ifconfig eth0
```

Affichage de toutes les interfaces réseaux activées

```
ifconfig
```

Affichage de toutes les interfaces réseaux activées ou non

```
ifconfig -a
```

Le mieux reste d'utiliser les scripts **ifup** et **ifdown**. Ceux-ci se basent sur les fichiers présents dans **/etc/sysconfig/network-scripts/**. Ces fichiers de configuration d'interface se nomment **ifcfg-xxx** où xxx est le nom de l'interface réseau comme eth0. Exemple de contenu :

```
DEVICE=eth0
IPADDR=192.168.1.2
NETMASK=255.255.255.0
NETWORK=192.168.1.0
BROADCAST=192.168.1.255
ONBOOT=yes
BOOTPROTO=static
```

Les paramètres parlent d'eux-mêmes. Les valeurs **NETWORK** et **BROADCAST** sont optionnelles si **IPADDR** et **NETMASK** sont renseignés (dans ce cas le calcul est automatique) ou si **DHCP** est utilisé.

BOOTPROTO indique comment monter l'interface, soit '**static**', soit '**dhcp**'. La valeur '**bootp**' peut aussi être utilisée. Dans le cas du DHCP, le fichier peut ressembler à ceci :

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
```

Dans le cas d'une configuration statique, **IPADDR** et **NETMASK** sont obligatoires :

```
DEVICE=eth0
IPADDR=192.168.1.2
NETMASK=255.255.255.0
ONBOOT=yes
BOOTPROTO=static
```

ONBOOT détermine si l'interface doit être automatiquement activée au démarrage de la machine. Une fois le fichier correctement renseigné, on utilise les commandes ifup/ifdown :

Activation de l'interface eth0

```
ifup eth0
```

Arrêt de l'interface eth0

```
ifdown eth0
```

9.3.3 Paramètres généraux

Le fichier **/etc/sysconfig/network** contient les paramètres généraux du réseau.

```
NETWORKING=yes
HOSTNAME=poste1.monreseau.org # nom complet
GATEWAY=0.0.0.0 # passerelle par défaut
NISDOMAIN= # nom du domaine NIS
```

- **NETWORKING** : activation ou non du réseau
- **HOSTNAME** : nom de domaine complet FQDN
- **GATEWAY** : adresse IP de la passerelle
- **GATEWAYDEV** : interface réseau permettant d'accéder à la passerelle
- **NISDOMAIN** : cas d'un domaine NIS

9.3.4 Routage

Avec l'utilisation des fichiers et commandes précédentes, il n'y a pas besoin de créer un routage spécifique car la passerelle par défaut est déjà présente (cf paramètres généraux) et les routes pour les interfaces réseaux sont automatiquement mises en place par **ifup**. Cependant on peut utiliser la commande **route**.

Affiche les routes actuelles

```
route
```

Ajout de l'entrée loopback

```
route add -net 127.0.0.0
```

Ajoute la route vers le réseau 192.168.1.0 passant par eth0. Netmask peut être omis.

```
route add -net 192.168.1.0 netmask 255.255.255.0 eth0
```

Ajoute la passerelle par défaut vers le routeur

```
route add default gw 192.168.1.254
```

Supprime la route vers le réseau 172.16.0.0

```
route del -net 172.16.0.0 eth0
```

9.4 /etc/resolv.conf

Le fichier **/etc/resolv.conf** est utilisé pour indiquer au système quels serveurs de noms et domaines à interroger pour résoudre les requêtes DNS clientes. Les API sont incluses dans la bibliothèque et les API standards de Linux (il n'y a pas besoin d'ajouter des outils supplémentaires). On appelle cette bibliothèque le **resolver**.

Note : En configuration DHCP ce fichier est en principe mis automatiquement à jour et ne devrait pas être modifié sauf si vous avez interdit la configuration DNS sur votre client.

```
$ cat /etc/resolv.conf
domain mondomaine.org
search mondomaine.org
nameserver 192.168.1.1
nameserver 192.168.1.2
```

- **domain** : nom du domaine local. La plupart des requêtes sont généralement réduites à des raccourcis relatifs au domaine local. Si l'entrée est absente, le resolver tente de récupérer la valeur par la fonction C « **gethostname** ».

- **search** : liste des domaines de recherche. Par défaut en utilisation de raccourcis (noms d'hôtes courts) le resolver lance une recherche sur le domaine défini ci-dessus, mais on peut spécifier ici une liste de domaines séparés par des espaces ou des virgules.
- **nameserver** : Adresse IP du serveur de noms (le serveur DNS). On peut en placer trois. Le resolver tente le premier. En cas d'échec (timeout) il passe au second, et ainsi de suite.

9.5 /etc/hosts et /etc/networks

Sans même utiliser de serveur de noms, on peut établir une correspondance entre les adresses IP et les noms des machines au sein du fichier **/etc/hosts**.

```
192.168.1.1    server1 www1 ftp
192.168.1.11   poste1
192.168.1.12   poste2
```

On peut faire de même pour nommer les réseaux (ce qui peut être utile pour les tcp_wrappers ou la commande route) dans le fichier **/etc/networks**.

```
loopnet    127.0.0.0
localnet   192.168.1.0
```

10 Services réseaux xinetd

10.1 Présentation

Le démon **xinetd** es un « super-service » permettant de contrôler l'accès à un ensemble de services dont **FTP** ou **Telnet**. Des options de configuration spécifiques peuvent être appliquées pour chaque service géré.

Lorsqu'un hôte client se connecte à un service réseau contrôlé par **xinetd**, **xinetd** reçoit la requête et vérifie tout d'abord les autorisations d'accès TCP (voir **tcp_wrappers**) puis les règles définies pour ce service (autorisations spécifiques, ressources allouées, etc). Une instance du service est alors démarrée et lui cède la connexion. A partir de ce moment, **xinetd** n'interfère plus dans la connexion entre le client et le serveur.

10.2 Configuration

Les fichiers de configuration sont :

- **/etc/xinetd.conf** : configuration globale
- **/etc/xinetd.d/*** : répertoire contenant les fichiers spécifiques aux services

Contenu de xinetd.conf :

```
defaults
{
        instances               = 60
        log_type                = SYSLOG authpriv
        log_on_success          = HOST PID
        log_on_failure          = HOST
        cps                     = 25 30
}

includedir /etc/xinetd.d
```

- **instances** : nombre maximal de requêtes qu'un service xinetd peut gérer à un instant donné.
- **log_type** : dans notre cas, les traces sont gérées par le démon **syslog** via **authpriv** et les

traces sont placées dans **/var/log/secure.** « **FILE /var/log/xinetd** » aurait placé les traces dans **/var/log/xinetd**.

- **log_on_success** : xinetd va journaliser l'événement si la connexion au service réussit. Les informations tracées sont l'hôte (**HOST**) et le **PID** du processus serveur traitant la connexion.
- **log_on_failure** : idem mais pour les échecs. Il devient simple de savoir quels hôtes ont tenté de se connecter si par exemple la connexion n'est pas autorisée.
- **cps** : xinetd n'autorise que 25 connexions par secondes à un service. Si la limite est atteinte, xinetd attendra 30 secondes avant d'autoriser à nouveau les connexions.
- **includedir** : inclut les options des fichiers présents dans le répertoire indiqué.

Exemple /etc/xinetd.d/telnet :

```
# default: on
# description: The telnet server serves telnet sessions; it uses \
#       unencrypted username/password pairs for authentication.
service telnet
{
        disable = no
        flags           = REUSE
        socket_type     = stream
        wait            = no
        user            = root
        server          = /usr/sbin/in.telnetd
        log_on_failure  += USERID
}
```

La première ligne en commentaire **default** a une importance particulière. Elle n'est pas interprétée par xinetd mais par ntsysv ou chkconfig pour déterminer si le service est actif (voir arrêt/relance).

- **service** : nom du service qui correspond à un service défini dans /etc/services
- **flags** : attributs pour la connexion. REUSE indique que la socket sera réutilisée pour une connexion telnet
- **socket_type** : spécifie le type de socket. Généralement **stream** (tcp) ou **dgram** (udp). Une connexion directe IP se fait par **raw**.
- **wait** : indique si le serveur est single-threaded (yes) ou multi-threaded (no)
- **user** : sous quel compte utilisateur le service sera lancé
- **server** : chemin de l'exécutable devant être lancé
- **log_on_failure** : le += indique qu'on rajoute l'option associée au fichier de trace en plus de celles par défaut. Ici : le login
- **disable** : indique si le service est actif ou non.

Certaines options peuvent améliorer les conditions d'accès et la sécurité :

- only_from : permet l'accès uniquement aux hôtes spécifiés
- no_access : empêche l'accès aux hôtes spécifiés (ex : 172.16.17.0/24)
- access_times : autorise l'accès uniquement sur une plage horaire donnée (ex :09:00-18:30)

10.3 Démarrage et arrêt des services

On distingue deux cas.

Premier cas, le service **xinetd** est un service comme un autre dont le démarrage ou l'arrêt peut s'effectuer avec la commande **service**.

```
service xinetd start
```

Dans ce cas, on utilise la commande **chkconfig** pour autoriser ou non le lancement du service au démarrage pour chaque niveau d'exécution (runlevel).

```
chkconfig --level 345 xinetd on
```

Second cas, comme **xinetd** gère plusieurs services, l'arrêt de xinetd arrête tous les services associés, et le démarrage de xinetd lance tous les services associés. Il n'est pas possible de choisir quels services de xinetd seront lancés dans tel ou tel niveau d'exécution. Mais on peut choisir d'activer ou de désactiver simplement un service avec **chkconfig**.

```
chkconfig telnet on
```

11 OpenSSH

11.1 Présentation

OpenSSH est un protocole de shell sécurisé, un mécanisme qui permet l'authentification sécurisée, l'exécution à distance et la connexion à distance. Il permet aussi le transport sécurisé du protocole X Window. En fait, il est capable d'encapsuler des protocoles non sécurisés en redirigeant les ports.

Les packages à utiliser pour un serveur sont **openssh**, **openssl** et **openssh-clients**. Pour X on rajoute les packages **openssh-askpass*** (il peut y en avoir plusieurs suivant l'environnement de bureau).

L'utilisation la plus commune reste l'accès distant sécurisé à une machine via le client ssh.

11.2 Configuration

La configuration est **/etc/ssh/sshd_config**. Quelques options sont éventuellement à modifier :

- **Port** : le numéro de port, par défaut 22
- **Protocol** : fixé à 2,1 il autorise SSH1 et SSH2. On préférera SSH2 et donc on laissera la valeur 2 seule
- **ListenAddress** : par défaut ssh écoute sur toutes les IP du serveur. On peut autoriser uniquement l'écoute sur une interface donnée
- **PermitRootLogin** : ssh autorise les connexions de root. On peut placer la valeur à « **no** ». Dans ce cas, il faudra se connecter en simple utilisateur et passer par **su**.
- **Banner** : chemin d'un fichier dont le contenu sera affiché aux utilisateurs lors de la connexion.

Ssh est un service System V qu'on lance avec **service**.

```
service sshd start
```

11.3 Utilisation

La commande **ssh** permet d'établir une connexion.

```
ssh -l login host
ssh login@host
```

L'option **-X** permet d'activer le forwarding de X Window.

```
ssh -X login@host
```

12 Monter un serveur DHCP

12.1 Présentation

DHCP : *Dynamic Host Configuration Protocol* : Protocole de configuration dynamique des hôtes. DHCP permet aux hôtes d'un réseau de demander et recevoir des informations de configuration (adresse, routage, DNS, etc). Il y a en général un seul serveur DHCP par segment de réseau même si plusieurs sont possibles. Si le serveur est sur un autre segment, on peut utiliser un agent de retransmission DHCP.

Autrement dit, un client DHCP recherche tout seul un serveur DHCP qui lui communiquera son adresse IP. L'adresse IP est assignée soit dynamiquement à partir de plages d'adresses prédéfinies, soit statiquement en fonction de l'adesse MAC du demandeur. Les informations sont valables un laps de temps donné et configurable (un bail) qui peut être renouvelé.

DHCP est un sur-ensemble de BOOTP (Bootstrap Protocol). Quand le client cherche à contacter un serveur, c'est BOOTP qui fournit les informations d'adressage. DHCP gère les renouvellements. BOOTP se base sur le protocole de transport UDP.

Un hôte n'a aucune information réseau de disponible au démarrage. Il doit trouver seul un serveur DHCP. Pour ça, BOOTP effectue un broadcast sur l'IP 255.255.255.255 avec une trame contenant ses informations (comme son adresse MAC) et les informations souhaitées (type de requête, ici DHCPDISCOVER, port de connexion, etc). Le broadcast est envoyé par définition à tous les hôtes du réseau local. Quand le serveur DHCP détecte la trame, il effectue lui aussi un broadcast (le hôte client n'a pas encore d'IP) avec les informations de base souhaitées par l'hôte (DHCPOFFER, premiers paramêtres). L'hôte établit une première configuration puis demande confirmation de l'IP (DHCPREQUEST). Le serveur DHCP confirme (DHCPHACK). Le bail est confirmé et le client dispose dès lors de toutes les informations valides.

12.2 Serveur dhcpd

Le serveur **dhcpd** est un service (daemon) lancé à l'aide d'un script (**/etc/init.d/dhcpd**). Il est configuré à l'aide du fichier **/etc/dhcpd.conf**. Les adresses IP allouées sont placées dans **/var/lib/dhcp/dhcpd.leases**.

```
service dhcpd start
```

12.2.1 Informations de base

```
ddns-update-style none; # pas de mise à jour du DNS par DHCP
option domain-name "toto.fr"; # nom de domaine transmis au client
option domain-name-servers 192.168.1.254; # liste des DNS séparés par des virgules
default-lease-time 21600; # durée du bail par défaut en secondes sans demande explicite
max-lease-time 43200; # durée max du bail si la demande du client est plus élevée
```

Comme dhcpd peut gérer plusieurs sous-réseaux, on doit lui préciser les règles à appliquer pour chaque sous-réseau. Généralement dans le cadre d'un petit réseau un seul bloc sera présent mais tous les cas sont permis. Si vous êtes certains de n'avoir d'un seul réseau, vous pouver omettre la déclaration du subnet.

```
# Gestion du sous-réseau 192.168.1.0
subnet 192.168.1.0 netmask 255.255.255.0
{
  option routers 192.168.1.254; # passerelle pour ce réseau
  option subnet-mask 255.255.255.0; # masque de sous-réseau
  range 192.168.1.2 192.168.1.250; # Configuration de l'intervalle DHCP
```

```
# Cas d'attributions d'IP statiques
host station1
{
  hardware ethernet 00:A0:ad:41:5c:b1; # Adresse MAC
  fixed-address 192.168.1.1; # cette machine aura l'IP 192.168.1.1
}
```

Note : Certains clients DHCP ignorent totalement le fait qu'un serveur DHCP peut dynamiquement allouer un nom (hostname) à l'hôte. Dans l'exemple précédent, notre machine avec l'IP 192.168.1.1 devrait obtenir le nom station1. Voici un exemple :

```
# les hôtes se verront attribués les noms des host déclarés
use-host-decl-names on;
host station1
{
  hardware ethernet 00:A0:ad:41:5c:b1; # Adresse MAC
  fixed-address 192.168.1.1; # cette machine aura l'IP 192.168.1.1 et le nom station1
}
```

On peut aussi faire du cas par cas :

```
host station2
{
  hardware ethernet 00:A0:ad:41:5c:b2; # Adresse MAC
  fixed-address 192.168.1.251; # ce host aura l'IP 192.168.1.251
  option host-name "station2"; # ce host aura comme nom station2
}
```

12.3 Côté client

Sous Windows, cochez l'option associée. Sous Linux, modifiez le fichier **`/etc/sysconfig/network-script/ifcfg-xxx`** en plaçant **`BOOTPROTO`** à « **`dhcp`** »

13 Serveur DNS

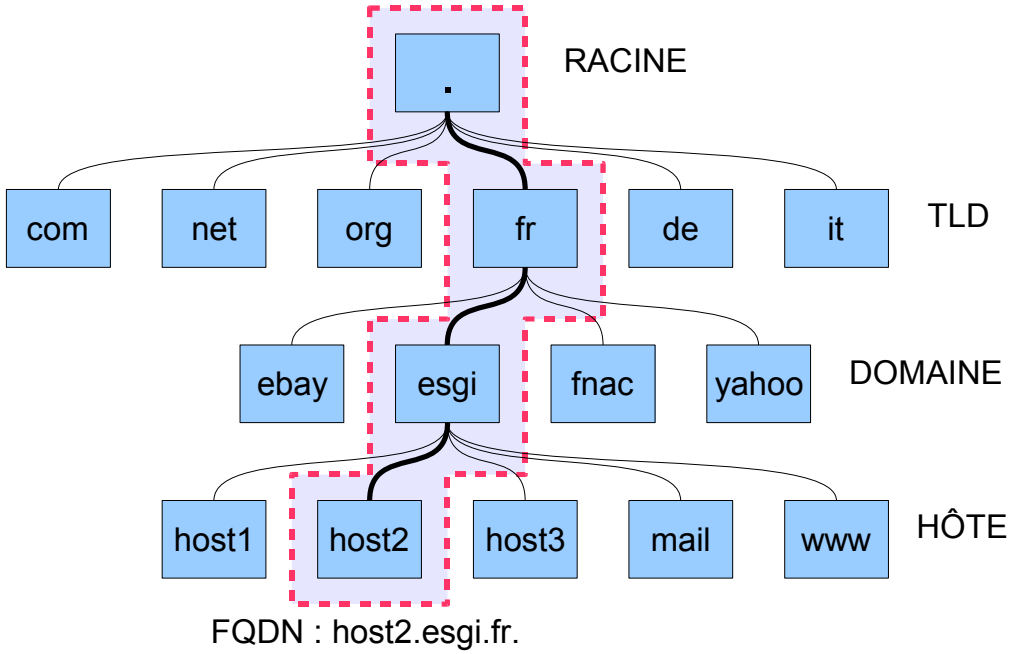
13.1 Présentation

Le Système de Noms de Domaines **DNS** (Domain Name System) transforme les noms d'hôtes en adresses IP : c'est la **résolution de nom**. Il transforme les adresses IP en noms d'hôtes : c'est la **résolution inverse**. Il permet de regrouper les machines par domaines de nom. Il fournit des informations de routage et de courrier électronique.

Le DNS permet de faire référence à des systèmes basés sur IP (les *hôtes*) à l'aide de noms conviviaux (les *noms de domaines*). L'intérêt d'un DNS est évident. Les noms de domaine sont plus simples à retenir et si son adresse IP change, l'utilisateur ne s'en rend même pas compte. On comprend que le DNS est un service clé critique pour Internet.

Les noms de domaine sont séparés par des points, chaque élément pouvant être composé de 63 caractères, il ne peut y avoir qu'un maximum de 127 éléments et le nom complet ne devant pas dépasser 255 caractères. Le nom complet non abrégé est appelé **FQDN** (*Fully Qualified Domain Name*). Dans un FQDN, l'élément le plus à droite est appelé **TLD** (*Top Level Domain*), celui le plus à gauche représente l'hôte et donc l'adresse IP.

Le DNS contient une configuration spéciale pour les routeurs de courrier électronique (définitions MX) permettant une résolution inverse, un facteur de priorité et une tolérance de panne.



Une zone est une partie d'un domaine gérée par un serveur particulier. Une zone peut gérer un ou plusieurs sous-domaines, et un sous-domaine peut être éclaté en plusieurs zones. Une zone représente l'unité d'administration dont une personne peut être responsable.

13.2 Lancement

Le service s'appelle **named**.

```
service named start
```

13.3 Configuration de Bind

Bind (*Berkeley Internet Name Daemon)* est le serveur de noms le plus utilisé sur Internet. Bind 9 supporte l'IPv6, les noms de domaine unicode, le multithread et de nombreuses améliorations de sécurité.

13.3.1 Configuration générale

La configuration globale de Bind est placée dans le fichier **/etc/named.conf.** La configuration détaillée des zones est placée dans **/var/lib/named**. **/etc/named.conf** est composé de deux parties. La première concerne la configuration globale des options de Bind. La seconde est la déclaration des zones pour les domaines individuels. Les commentaires commencent par un # ou //.

Attention il arrive parfois (notamment sur RHEL 4.x) que la configuration de Bind soit « chrootée ». Sur Centos et RHEL 4.x **named.conf** est dans **/var/named/chroot/etc/**.On peut modifier ce mode en modifiant le fichier de configuration **/etc/sysconfig/named**.

```
# cat /etc/sysconfig/named
...
CHROOT=/var/named/chroot
...
```

Dans ce cas, tous les fichiers de configuration, y compris les zones, sont relatifs à ce chemin.

Voici un fichier **named.conf** de base.

```
options {
        directory "/var/lib/named";
        forwarders { 10.0.0.1; };
        notify no;
};
zone "localhost" in {
        type master;
        file "localhost.zone";
};
zone "0.0.127.in-addr.arpa" in {
        type master;
        file "127.0.0.zone";
};
zone "." in {
        type hint;
        file "root.hint";
};
```

13.3.2 Section globale

La configuration globale est placée dans la section « **options** ». Voici un détail de quelques options importantes :

- **directory "filename";**
  - Emplacement des fichiers contenant les données des zones
- **forwarders { adresse-ip; };**
  - Si le serveur bind ne peut résoudre lui-même la requête, elle est renvoyée à un serveur DNS extérieur, par exemple celui du fournisseur d'accès.
- **listen-on-port 53 {127.0.0.1; adresse-ip; };**
  - Port d'écoute du DNS suivi des adresses d'écoute. On indique ici les adresses IP des interfaces réseau de la machine. Il ne faut pas oublier 127.0.0.1.
- **allow-query { 127.0.0.1; réseau; };**
  - Machine(s) ou réseau(x) autorisés à utiliser le service DNS. Par exemple 192.168.1/24. Si la directive est absente, tout est autorisé.

- **`allow-transfer { 192.168.1.2; } ;`**
  - Machine(s) ou réseau(x) autorisés à copier la base de données dans le cas d'une relation maître et esclave. Par défaut aucune copie n'est autorisée.
- **`notify no;`**
  - On notifie ou non les autres serveurs DNS d'un changement dans les zones ou d'un redémarrage du serveur

13.3.3 Section de zones

Pour chaque domaine ou sous-domaine, on définit deux sections « **`zone`** ». La première contient les informations de résolution de nom (nom->IP) et la seconde les informations de résolution inverse (IP->Nom). Dans chacun des cas, la zone peut être maître (**`Master`**) ou esclave (**`Slave`**) :

- **`Master`** : Le serveur contient la totalité des enregistrements de la zone dans ses fichiers de zone. Lorsqu'il reçoit une requête, il cherche dans ces fichiers (ou dans son cache) la résolution de celle-ci.
- **`Slave`** : Le serveur ne contient par défaut aucun enregistrement sur fichier. Il se synchronise avec un serveur maître d'où il récupère toutes les informations de zone. Ces informations peuvent être placées dans un fichier. Dans ce cas l'esclave stocke une copie locale de la base. Lors de la synchronisation, le numéro de série de cette copie est comparée à celle du maître. Si les numéros sont différents, une nouvelle copie a lieu, sinon la précédente copie continue à être utilisée.

13.3.3.1 Zone de résolution

Elle est généralement appelée « **`zone`** ». Pour chaque domaine ou sous-domaine, elle indique dans quel fichier sont placées les informations de la zone (c'est à dire et entre autre les adresses IP associées à chaque hôte), son type (maître ou esclave), si on autorise ou non la notification, l'IP du serveur DNS maître dans le cas d'un esclave, etc.

Le nom de la zone est très important puisque c'est elle qui détermine le domaine de recherche. Quand le DNS reçoit une requête, il recherche dans toutes les zones une correspondance.

```
zone "domaine.org" {
       type    "master";
       file    "domaine.org.zone";
};
```

- **`type`** : master ou slave
- **`file`** : nom du fichier qui contient les informations de la zone. Il n'y a pas de rêgles précises de nommage mais pour des raisons de lisibilité il est conseillé de lui donner le même nom tant slave que master. Pour un master, c'est l'original éventuellement rempli par vos soins. Pour un slave, ce n'est pas obligatoire. S'il est présent, ce sera une copie du master, synchronisée.
- En cas de Master : on peut rajouter **`allow-transfer`** (serveurs autorisés à dupliquer la zone) et **`notify yes`** (indique une MAJ ou une relance pour les slaves).
- En cas de Slave : on rajoute la directive **`masters`** pour indiquer sur quel serveur Master se dupliquer.

13.3.3.2 Zone de résolution inverse

Pour chaque réseau ou sous-réseau IP (ou plage d'adresse) on définit une zone de résolution inverse dont le fichier contient une association IP->Nom de machine. C'est en fait presque la même chose

que la zone de résolution sauf qu'on doit respecter une convention de nommage :

- Le nom de la zone se termine toujours par une domaine spécial « **.in-addr.arpa** ».
- On doit tout d'abord déterminer quel réseau la zone doit couvrir (cas des sous-réseaux). Pour nous : un réseau de classe C 192.168.1.0 soit **192.168.1/24**
- On inverse l'ordre des octets dans l'adresse : **1.168.192.**
- On ajoute **in-addr.arpa**. Notre nom de zone sera donc **1.168.192.in-addr.arpa**
- Pour le reste, les mêmes remarques s'appliquent que pour la zone de résolution.

```
Zone "1.168.192.in-addr.arpa" {
        type            master;
        file            "192.168.1.zone";
};
```

13.3.3.3 Exemple

Ex : soit un domaine « **domaine.org** » sur un réseau de classe C 192.168.1.0. Soit deux serveurs DNS 192.168.1.1 Master et 192.168.1.2 Slave.

Sur le Master :

```
zone "domaine.org" {
        type    master;
        file    "domaine.org.zone";
        allow-transfer { 192.168.1.2; } ;
        notify yes;
};
zone "1.168.192.in-addr.darpa" {
        type    master;
        file    "192.168.1.zone";
        allow-transfer { 192.168.1.2; } ;
        notify yes;
};
```

Sur le Slave :

```
zone "domaine.org" {
        type            slave;
        file            "domaine.org.zone";
        masters         { 192.168.1.1; };
};
zone "1.168.192.in-addr.darpa" {
        type            slave;
        file            "192.168.1.zone";
        masters         { 192.168.1.1; } ;
};
```

13.3.3.4 Zones spéciales

La zone racine « **.** » permet de spécifier les serveurs racine. Quand aucune des zones n'arrive à résoudre une requête, c'est la zone racine qui est celle par défaut et qui renvoie sur les serveurs racine.

La zone de loopback n'est pas nécessaire pas bien utile. Elle fait office de DNS cache. Quand une requête arrive sur le serveur et qu'il ne possède pas l'information de résolution il va la demander aux serveurs DNS racines qui redescendront l'information. Celle-ci est alors placée en cache. Du coup les accès suivants sont bien plus rapides !

13.3.4 Fichiers de zones

13.3.4.1 Définitions

Les fichiers de zones utilisent plusieurs termes, caractères et abréviations spécifiques.

- **`RR`** : *Ressource Record* : Nom d'un enregistrement DNS (les données du DNS).
- **`SOA`** : *Star Of Authority* : permet de décrire la zone.
- **`IN`** : *the Internet* : définit une classe d'enregistrement qui correspond aux données Internet (IP). C'est celle par défaut si elle n'est pas précisée pour les enregistrements.
- **`A`** : *Address* : Permet d'associer une adresse IP à un nom d'hôte. Pour Ipv6 c'est AAAA.
- **`NS`** : *Name Server* : Désigne un serveur DNS de la zone.
- **`MX`** : *Mail eXchanger* : Désigne un serveur de courrier électronique, avec un indicateur de priorité. Plus la valeur est faible, plus la priorité est élevée.
- **`CNAME`** : *Canonical Name* : Permet de rajouter des alias : donner un nom à un autre nom. On peut créer des alias sur des nom d'hôte et aussi sur des alias.
- **`PTR`** : *Pointer* : Dans une zone de résolution inverse, fait pointer une IP sur un nom d'hôte.
- **`TTL`** : *Time To Live* : durée de vie des enregistrements de la zone.
- `@` : Dans les déclarations de la zone, c'est un alias (caractère de remplacement) pour le nom de la zone déclarée dans /etc/named.conf. Ainsi si la zone s'appelle domaine.org, @ vaut domaine.org. Dans la déclaration de l'administrateur de la SOA, il remplace ponctuellement le point dans l'adresse de courrier électronique.
- Le point « **`.`** »: Il on omet le point en fin de déclaration d'hôte, le nom de la zone est concaténée à la fin du nom. Par exemple pour la zone domaine.org, si on écrit « **`poste1`** », cela équivaut à « **`poste1.domaine.org`** ». Si on écrit « **`poste1.domaine.org`** » (sans le point à la fin) alors on a comme résultat « **`poste1.domaine.org.domaine.org`** » ! Pour éviter ça, on écrira « **`poste1.domaine.org.`** » (notez le point à la fin).
- Certains enregistrements demandent une une notion de durée, qui est généralement exprimée en secondes mais aussi parfois avec des abréviations :
  - **`1M`** : une minute, soit 60 secondes (1M, 10M, 30M, etc)
  - **`1H`** : une heure, 3600 secondes
  - **`1D`** : un jour, 86400 secondes
  - **`1W`** : une semaine, 604800 secondes
  - **`365D`** : un an, 31536000 secondes

Attention, et ceci est très important : dans les fichiers de zones, IL NE FAUT JAMAIS COMMENCER UNE LIGNE PAR DES ESPACES OU TABULATIONS. Ça ne marche absolument pas.

13.3.4.2 Zone

On commence tout d'abord par une directive **`TTL`** qui indique le temps de vie de la zone en secondes. Cela signifie que chaque enregistrement de la zone sera valable durant le temps indiqué

par **TTL** (note : il est possible de modifier cette valeur pour chaque enregistrement). Durant ce temps, les données peuvent être placées en cache par les autres serveurs de noms distants. Une valeur élevée permet de réduire le nombre de requêtes effectuées et de rallonger les délais de synchronisation.

```
$TTL 86400
```

Après les directives, on place un enregistrement de ressources **SOA** :

```
<domain>        IN      SOA     <primary-name-server> <hostmaster-email> (
                                <serial-number>
                                <time-to-refresh>
                                <time-to-retry>
                                <time-to-expire>
                                <minimum-TTL> )
```

- **domain** : c'est le nom de la zone, le même nom que celui utilisé dans **/etc/named.conf**. On peut le remplacer par @ sinon il ne -faut pas oublier de le terminer par un point (pour éviter une concaténation).
- **primary-name-server** : le nom sur serveur DNS maître sur cette zone. Il ne faudra pas oublier de le déclarer dans la liste des hôtes (enregistrements **PTR** ou **A**).
- **hostmaster-email** : adresse de courrier électronique de l'administrateur du serveur de nom. Le caractère **@** étant déjà réservé à un autre usage, on utilise un point pour le remplacer. Ainsi « admin@domaine.org » devra s'écrire « **admin.domaine.org.** » .
- **serial-number** : c'est un numéro de série qu'on doit incrémenter manuellement à chaque modification du fichier zone pour que le serveur de nom sache qu'il doit recharger cette zone. Elle est utilisée pour la synchronisation avec les serveurs esclaves. Si le numéro de série est le même qu'à la dernier synchronisation les données ne sont pas rafraîchies. Par convention on place **YYYYMMDDNN** (année-mois-jour-numéro) sur dix chiffres.
- **time-to-refresh** : elle indique à tout serveur esclave combien de temps il doit attendre avant de demander au serveur de noms maître si des changements ont été effectués dans la zone.
- **time-to-retry** : elle indique au serveur esclave combien de temps attendre avant d'émettre à nouveau une demande de rafraîchissement si le serveur maître n'a pas répondu. La demande se fera toutes les time-to-retry secondes.
- **time-to-expire** : Si malgré les tentatives de contacts toutes les time-to-retry secondes le serveur n'a pas répondu au bout de la durée indiquée dans time-to-expire le serveur esclave cesse de répondre aux requêtes pour cette zone.
- **Minimum-TTL** : le serveur de nom demande aux autres serveurs de noms de mettre en cache les informations pour cette zone pendant au moins la durée indiquée.

```
@       IN      SOA     dns1.domaine.org.     hostmaster.domaine.org. (
                        2005122701      ; serial
                        21600           ; refresh de 6 heures
                        3600            ; tenter toutes les 1 heures
                        604800  ; tentatives expirent après une semaine
                        86400   )       ; TTL mini d'un jour
```

On passe ensuite au enregistrements **NS** (*Name Server*) où on spécifie les serveurs de noms de cette zone.

```
        IN      NS      dns1
        IN      NS      dns2
```

Note : Quand on ne spécifie pas en début de ligne un nom d'hôte ou de zone (complet ou @), cela veut dire qu'on utilise le même que la ligne du dessus. Tant qu'on n'en précise pas de nouveau, c'est le dernier indiqué qui est utilisé. Ainsi ci-dessus les lignes pourraient être :

```
@       IN      NS      dns1
@       IN      NS      dns2
```

ou

```
domaine.org.    IN      NS      dns1
domaine.org.    IN      NS      dns2
```

Notez qu'on a pas mis de point après le nom de l'hôte et donc domaine.org est concaténé dns1.domaine.org.

```
        IN      NS      dns1
```

équivaut à

```
        IN      NS      dns1.domaine.org.
```

On passe ensuite à l'énumération des serveurs de courrier électronique de la zone. La valeur numérique située après MX indique la priorité. Plus la valeur est basse plus le serveur est prioritaire et sera contacté en premier. Si les valeurs sont identiques, le courrier est redistribué de manière homogène entre les serveurs. Si un serveur ne répond pas (chargé, panne) la bascule vers un autre est automatique.

```
        IN      MX      10      mail
        IN      MX      15      mail2
```

Si on souhaite qu'une machine réponde en passant par le FQDN domaine.org sans préciser d'hôte (par exemple http://domaine.org sans utiliser http://www.domaine.org) alors on peut maintenant déclarer une adresse IP pour ce serveur. Ainsi un « **`ping domaine.org`** » répondra 192.168.1.3 !

```
        IN      A       192.168.1.3
```

On peut maintenant déclarer les autres hôtes dont nos serveurs de noms, mails, postes, etc.

```
dns1            IN      A       192.168.1.1
dns2            IN      A       192.168.1.2
server1 IN      A       192.168.1.3
server2 IN      A       192.168.1.4
poste1  IN      A       192.168.1.11
poste2  IN      A       192.168.1.12
poste3  IN      A       192.168.1.13
```

On remarque que nos serveurs mail et mail2 ne sont pas déclarés, et qu'on a pas indiqué de serveur web et ftp. Nous allons utiliser les alias, en faisant pointer ces noms d'hôtes sur d'autres hôtes.

```
mail            IN      CNAME   server1
mail2           IN      CNAME   server2
www             IN      CNAME   server1
ftp             IN      CNAME   server1
```

La configuration de la zone est terminée, il faut maintenant s'occuper de la zone de résolution inverse.

13.3.4.3 Zone de résolution inverse

La zone de révolution inverse est presque identique à la première, sauf que les enregistrements A sont remplacés par des enregistrements PTR pour traduire une IP en hôte. Le TTL et la déclaration SOA doivent être si possible identiques (sauf le nom de la zone). On placera aussi les enregistrements NS.

```
        IN      NS      dns1.domaine.org.
        IN      NS      dns2.domaine.org.
```

On est pas obligé de placer dans la zone de résolution inverse la traduction des IP du DNS, étant donné que c'est le DNS lui-même qui résout son propre nom ! Cependant le faire peut accélérer la

démarche, le DNS n'ayant pas à faire une requête sur lui-même. On passe aux enregistrements PTR traduisant l'IP pour chaque hôte.

```
1       IN      PTR     dns1.domaine.org.
2       IN      PTR     dns2.domaine.org.
3       IN      PTR     server1.domaine.org.
4       IN      PTR     server2.domaine.org.
11      IN      PTR     poste1.domaine.org.
12      IN      PTR     poste2.domaine.org.
13      IN      PTR     poste3.domaine.org.
```

Il est théoriquement possible pour la même IP d'attribuer plusieurs hôtes, les RFC ne sont pas très explicites sur cette possibilité qui au final peut créer des problèmes.

13.3.5 Round-robin

A l'aide de l'outil **`host`**, nous allons voir un point assez particulier du DNS.

```
$ host news.free.fr
news.free.fr is an alias for news.proxad.net.
news.proxad.net has address 212.27.42.71
news.proxad.net has address 212.27.42.72
news.proxad.net has address 212.27.42.73
news.proxad.net has address 212.27.42.74
news.proxad.net has address 212.27.42.75
news.proxad.net has address 212.27.42.76
news.proxad.net has address 212.27.42.77
news.proxad.net has address 212.27.42.78
news.proxad.net has address 212.27.42.79
news.proxad.net has address 212.27.42.80
news.proxad.net has address 212.27.42.129
news.proxad.net has address 212.27.42.130
news.proxad.net has address 212.27.42.131
news.proxad.net has address 212.27.42.132
news.proxad.net has address 212.27.42.133
news.proxad.net has address 212.27.42.134
news.proxad.net has address 212.27.42.135
news.proxad.net has address 212.27.42.136
news.proxad.net has address 212.27.42.137
news.proxad.net has address 212.27.42.138
news.proxad.net has address 212.27.42.65
news.proxad.net has address 212.27.42.66
news.proxad.net has address 212.27.42.67
news.proxad.net has address 212.27.42.68
news.proxad.net has address 212.27.42.69
news.proxad.net has address 212.27.42.70
```

Le premier enseignement de cette commande est qu'il est possible de donner le même nom d'hôte à plusieurs adresses IP. Relançons la commande une seconde fois.

```
news.proxad.net has address 212.27.42.72
news.proxad.net has address 212.27.42.73
news.proxad.net has address 212.27.42.74
news.proxad.net has address 212.27.42.75
news.proxad.net has address 212.27.42.76
news.proxad.net has address 212.27.42.77
news.proxad.net has address 212.27.42.78
news.proxad.net has address 212.27.42.79
news.proxad.net has address 212.27.42.80
news.proxad.net has address 212.27.42.129
news.proxad.net has address 212.27.42.130
news.proxad.net has address 212.27.42.131
news.proxad.net has address 212.27.42.132
news.proxad.net has address 212.27.42.133
news.proxad.net has address 212.27.42.134
news.proxad.net has address 212.27.42.135
news.proxad.net has address 212.27.42.136
news.proxad.net has address 212.27.42.137
news.proxad.net has address 212.27.42.138
news.proxad.net has address 212.27.42.65
news.proxad.net has address 212.27.42.66
news.proxad.net has address 212.27.42.67
```

```
news.proxad.net has address 212.27.42.68
news.proxad.net has address 212.27.42.69
news.proxad.net has address 212.27.42.70
news.proxad.net has address 212.27.42.71
```

On remarque que les réponses ont tourné ! Ainsi, à chaque nouvelle requête sur ces serveurs, l'adresse IP n'est pas la même. C'est ce qu'on appelle le round-robin qui peut être vu, comme pour MX, une balance de charge. Le serveur DNS fait tourner automatiquement les adresses. Il va de soi que comme on souhaite accéder aux mêmes données, le contenu de chacun des serveurs doit être identique. La configuration est simple. Dans le fichier zone, on donne n adresses IP au même nom d'hôte.

```
www     IN      A       192.168.1.3
www     IN      A       192.168.1.20
www     IN      A       192.168.1.21
```

ou

```
www     IN      A       192.168.1.3
        IN      A       192.168.1.20
        IN      A       192.168.1.21
```

On peut changer l'ordre de sortie des adresses IP en modifiant le fichier **`/etc/named.conf`** avec la directive **`rrset-order`**. L'ordre peut être « **`cyclic`** », « **`random`** » ou « **`fixed`** ». Ce dernier mode n'est pas supporté par Bind 9. par défaut l'ordre est cyclique.

```
rrset-order {
        order cyclic;
};
```

Il ne s'agit pas d'une réelle balance de charge : le serveur de noms ne tient ni compte de la charge de l'hôte, ni des ports utilisés.

13.3.6 Diagnostic des problèmes

La commande **`named-checkconf`** vérifie la syntaxe du fichier **`named.conf`**. On lui passe en paramètre le fichier. Par défaut, sur RHEL/CentOS, il cherche le fichier chrooté. La sortie donnera les lignes posant problème.

La commande **`named-checkzone`** vérifie la syntaxe d'un fichier de zone (y compris reverse). On lui passe en paramètre le nom du fichier zone.

13.3.7 Interrogation dig et host

Le programme **`dig`** est un outil d'interrogation avancé de serveur de noms, capable de restituer toutes les informations des zones.

```
$ dig sncf.fr
; <<>> DiG 9.2.3 <<>> sncf.fr
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;sncf.fr.                       IN      A

;; ANSWER SECTION:
sncf.fr.                3600    IN      A       10.17.30.29

;; Query time: 0 msec
;; SERVER: 10.17.30.3#53(10.17.30.3)
;; WHEN: Fri Jan 06 15:12:43 2006
;; MSG SIZE  rcvd: 41
```

Par défaut dig ne restitue que l'adresse de l'hôte passé en paramètre. En cas de réussite, le statut est à **NOERROR**, le nombre de réponses à côté de **ANSWER** et la réponse se situe en-dessous de la section **ANSWER**. Pour obtenir une résolution inverse il y a deux solutions.

```
$ dig 29.30.17.10.in-addr.arpa ptr
```

ou plus simplement

```
$ dig -x 10.17.30.29
...
;; ANSWER SECTION:
29.30.17.10.in-addr.arpa. 3600  IN      PTR     camelia.sncf.fr.
29.30.17.10.in-addr.arpa. 3600  IN      PTR     sitp17bib67.inet.sncf.fr.
...
```

Dans la première syntaxe, on remarque qu'on peut rajouter un paramètre d'interrogation. Voici les principaux.

- **a** : uniquement l'adresse
- **any** : toutes les informations concernant le domaine
- **mx** : les serveurs de messagerie
- **ns** : les serveurs de noms
- **soa** : la zone Start of Authority
- **hinfo** : Infos sur l'hôte
- **txt** : texte de description
- **ptr** : zone reverse de l'hôte
- **axfr** : liste de tous les hosts de la zone

```
$ dig any sncf.fr
sncf.fr.                3600    IN      A       10.17.30.29
sncf.fr.                3600    IN      NS      ns5.sncf.fr.
sncf.fr.                3600    IN      NS      ns2.sncf.fr.
sncf.fr.                3600    IN      NS      ns1.sncf.fr.
sncf.fr.                3600    IN      NS      ns3.sncf.fr.
sncf.fr.                3600    IN      NS      ns6.sncf.fr.
sncf.fr.                3600    IN      SOA     ns3.sncf.fr. dnsmaster.sncf.fr.
257454 300 600 691200 3600
sncf.fr.                3600    IN      MX      10 msginternet1.sncf.fr.
sncf.fr.                3600    IN      MX      10 msginternet2.sncf.fr.
sncf.fr.                3600    IN      TXT     "Domaine SNCF Intranet (Gestionn
aire DSIT/XI/E tel:301818)"

;; ADDITIONAL SECTION:
ns5.sncf.fr.            3600    IN      A       10.151.156.83
ns2.sncf.fr.            3600    IN      A       10.17.30.3
ns1.sncf.fr.            3600    IN      A       10.17.30.30
ns3.sncf.fr.            3600    IN      A       10.17.30.29
ns6.sncf.fr.            3600    IN      A       10.151.156.84
```

L'outil **host** fait la même chose de manière peut-être un peu plus simple.

```
$ host sncf.fr
sncf.fr has address 10.17.30.29
$ host -v sncf.fr
...
;; ANSWER SECTION:
sncf.fr.                3600    IN      A       10.17.30.29
...

$ host -t any sncf.fr
sncf.fr has address 10.17.30.29
sncf.fr name server ns5.sncf.fr.
sncf.fr name server ns2.sncf.fr.
sncf.fr name server ns1.sncf.fr.
```

```
sncf.fr name server ns3.sncf.fr.
sncf.fr name server ns6.sncf.fr.
sncf.fr SOA ns3.sncf.fr. dnsmaster.sncf.fr. 257454 300 600 691200 3600
sncf.fr mail is handled by 10 msginternet1.sncf.fr.
sncf.fr mail is handled by 10 msginternet2.sncf.fr.
sncf.fr text "Domaine SNCF Intranet (Gestionnaire DSIT/XI/E tel:301818)"

$ host -a sncf.fr
...
;; ANSWER SECTION:
sncf.fr.                3600    IN      A       10.17.30.29
sncf.fr.                3600    IN      NS      ns5.sncf.fr.
sncf.fr.                3600    IN      NS      ns2.sncf.fr.
sncf.fr.                3600    IN      NS      ns1.sncf.fr.
sncf.fr.                3600    IN      NS      ns3.sncf.fr.
sncf.fr.                3600    IN      NS      ns6.sncf.fr.
sncf.fr.                3600    IN      SOA     ns3.sncf.fr. dnsmaster.sncf.fr.
257454 300 600 691200 3600
sncf.fr.                3600    IN      MX      10 msginternet1.sncf.fr.
sncf.fr.                3600    IN      MX      10 msginternet2.sncf.fr.
sncf.fr.                3600    IN      TXT     "Domaine SNCF Intranet (Gestionn
aire DSIT/XI/E tel:301818)"

;; ADDITIONAL SECTION:
ns5.sncf.fr.            3600    IN      A       10.151.156.83
ns2.sncf.fr.            3600    IN      A       10.17.30.3
ns1.sncf.fr.            3600    IN      A       10.17.30.30
ns3.sncf.fr.            3600    IN      A       10.17.30.29
ns6.sncf.fr.            3600    IN      A       10.151.156.84
...
```

14 Courrier électronique

14.1 Principe

- Quand un client (un utilisateur) envoie un message, il utilise un **MUA** (*Mail User Agent*), par exemple Outlook Express, Thunderbird, Evolution, Kmail, Mutt, etc.
- Le **MUA** envoie le message au **MTA** (*Mail Transport Agent*). Le MTA étudie l'adresse électronique pour pour isoler le l'utilisateur et le domaine de destination. Puis il vérifie les informations DNS de type **MX** (*Mail exchanger*) pour le domaine choisi pour savoir à quel serveur transmettre le courrier. Si aucun MTA n'est disponible, le message est placé en file d'attente et relance la distribution plus tard.
- Le MX peut être soit un autre MTA qui jouera le rôle de routeur (cas d'une redirection vers un sous-domaine par exemple), soit un **MDA** (*Mail Delivery Agent*). Le MDA place le message dans un fichier temporaire, peut le filtrer, etc.
- A ce niveau, le destinataire reçoit le message : soit il le récupère en lisant directement le fichier temporaire (cas de la commande mail par exemple) soit il passe par un protocole de type **POP** ou **IMAP**.
- Le protocole de transport de messages est le **SMTP** (*Simple Mail Transfert Protocol*) sur le port 25.
- Les protocoles de réception de messages est soit **POP** (*Post Office Protocol*) sur le port 110 (POP3) ou 109 (POP2), soir **IMAP** (*Internet Message Access Protocol*)

Deux suites de courrier électronique se partagent l'essentiel du marché sur Unix : **sendmail** et **postfix**.

La suite libre **sendmail** est la plus connue et la plus utilisée. Il a été créé en 1981 par Eric Allman et a été intégré à BSD 4.2 en 1983. On estimait en 2000 son utilisation dans plus de 100 millions de

serveurs de courrier électronique. Tant qu'on a pas à modifier sa configuration de base sendmail est idéal. Si on souhaite aller plus loin, l'achat d'un livre (chez O'Reilly ou Eyrolles) de 60 euros et de 1000 pages est extrêmement conseillé. De même la configuration de sendmail est si complexe qu'un langage de macros appelé **m4** a été inventé rien que pour lui. Aussi, on édite pas (ou très rarement) le fichier de configuration de sendmail : on édite le fichier source des macros et on le « recompile » : m4 va créer le fichier de configuration de sendmail. Sendmail est devenu un monstre de puissance et de configuration.

Le produit **postfix** tend à être de plus en plus utilisé non pas par les déçus de sendmail mais par ceux qui craignent de devoir le configurer. C'est une alternative à sendmail. Les buts des développeurs (dont certains sont ceux de Sendmail) sont la :

- compatibilité avec Sendmail
- Rapidité (plus de un million de messages par jour sur un simple PIV)
- Simplicité d'administration (fichier de configuration simple et lisible)
- Sécurité (peut être chrooté)
- Modularité (décomposition des traitements)

On retiendra la simplicité. En effet on peut configurer postfix avec une seule commande sans avoir besoin d'éditer de fichier. C'est ce serveur que nous allons utiliser.

14.2 Postfix

La configuration de **postfix** est située dans **/etc/postfix/main.cf**. On modifie ses valeurs soit à la main, soit à l'aide de la commande **postconf**.

Postfix lance tout d'abord un service maître **master** qui sera chargé des processus secondaires **smtpd**, **pickup** et **nqmgr**.

Nous devons tout d'abord indiquer à RHEL/CentOS d'utiliser Postfix à la place de Sendmail avec la commande **alternatives**.

```
alternatives --set mta /usr/sbin/sendmail.postfix
```

Puis nous allons appliquer une configuration de base avec la commande **postconf**.

```
postconf -e "myorigin = mondomaine.org"              # domaine d'origine des messages
postconf -e "mydestination = mondomaine.org"         # de quels domaines recevoir le courrier
postconf -e "mynetworks = 192.168.1.0/24, 127.0.0.1" # de quels clients relayer le courrier
postconf -e "inet_interface = all"                   # sur quelle interface réseau écouter
```

Puis nous lançons le service.

```
service postfix start
```

On peut placer dans le fichier **/etc/aliases** des alias pour les utilisateurs locaux. Par exemple, si les messages de webmaster, admin et root doivent être redirigés vers regis :

```
regis: webmaster, admin, root
```

Les traces sont placées dans **/var/log/maillog**. On peut tester le serveur de cette manière :

```
mail -s `echo $USER` root@server1 < /etc/redhat-release # on centos-release
```

Si tout fonctionne, on aura des traces :

```
Fri 26 11:38:18 station1 postfix/pickup[12357] : F145040154: uid=0 from <root>
Fri 26 11:38:18 station1 postfix/cleanup[12318] : F145040154: message-
id=<20060126113017.F145040154:@station1.mondomaine.org>
Fri 26 11:38:26 station1 postfix/nqmgr[3469] : F145040154: from=<root@station1.example.com>,
```

```
size=314, nrcpt=1 (queue active)
Fri 26 11:38:32 station1 postfix/smtp[12468] : F145040154: to=<root@server1>,
relay=server1.mondomaine.org[192.168.1.1], delay=17, status=sent (250 ok dirdel)
```

14.3 POP et IMAP

RHEL / CentOS fournit deux packages différents pour gérer le POP et l'IMAP. Le premier se nomme **cyrus-imap** et est en principe réservé aux grosses structures et aux serveurs 100% dédiés au courrier, c'est à dire où les utilisateurs ne se connectent pas.

Le second se nomme **dovecot**. Après sont installation il suffit juste de le placer en service SYSTEM V (**chkconfig**) et de le lancer (**service dovecot start**) pour que tout fonctionne, ou presque. Nous allons utiliser ce dernier.

Il va falloir éditer le fichier **/etc/dovecot.conf** pour vérifier les protocoles supportés.

```
protocols =  imap pop3
```

Puis on lance le service :

```
service dovecot start
```

Testez en envoyant un message. Configurez un client de messagerie pour vérifier si le serveur POP fonctionne :

```
# telnet localhost 110
trying 127.0.0.1...
Connected to localhost.localdomain.
Escape character is '^]'.
+OK POP3 localhost.localdomain server ready
USER regis
+OK user name accepted, password please
PASS password
+OK Mailbox open, 1 messages
STAT
+OK 1 384
TOP 1 99999
<message ici>
...
DELE 1
+OK Message deleted
QUIT
+OK bye
```

15 Service HTTP Apache

15.1 Présentation

Apache 2 est le serveur HTTP le plus utilisé actuellement sur les serveurs Web. Sa configuration et sa flexibilité en font un serveur incontournable.

Lorsqu'un serveur Apache reçoit des requêtes, il peut les redistribuer à des processus fils. La configuration permet de lancer des processus de manière anticipée et d'adapter dynamiquement ce nombre en fonction de la charge.

Apache est modulaire. Chaque module permet d'ajouter des fonctionnalités au serveur. Le module le plus connu est probablement celui gérant le langage PHP, « mod_php ». Chaque module s'ajoute via les fichiers de configuration, et il n'y a même pas besoin de relancer le serveur Apache : on lui donne juste l'ordre de relire sa configuration.

Apache peut gérer plusieurs sites web en même temps, ayant chacun leur nom, à l'aide des hôtes virtuels.

15.2 Arrêt/Relance

- **`service httpd start`** : démarre
- **`service httpd stop`** : stoppe
- **`service httpd restart`** : redémarre
- **`service httpd reload`** : demande à Apache de relire sa configuration sans redémarrer.

15.3 Configuration

La configuration principale est stockée dans **`/etc/httpd/conf/httpd.conf`**. Elle contrôle les paramètres généraux du serveur web, les hôtes virtuels et les accès. La configuration des différents modules est placée dans **`/etc/httpd/conf.d`**. Les modules sont présents dans **`/etc/httpd/modules/`**. Par défaut la racine du serveur, celle où sont placées les pages du site, est dans **`/var/www/html/`**.

15.4 Directives générales

Il est n'est pas possible de lister toutes les directives du fichier httpd.conf mais quelques unes sont importantes.

- **`ServerRoot`** : répertoire contenant les fichiers du serveur (configuration et modules). C'est généralement **`/etc/httpd`**.
- **`Listen`** : Ports sur lesquels le serveur Apache écoute. Par défaut 80 (443 en https). On peut en spécifier plusieurs avec plusieurs directive Listen. Si le serveur dispose de plusieurs adresses IP, on peut rajouter l'IP au port associé : **`Listen 192.168.1.3:80`**
- **`User`** : utilisateur des processus Apache. On n'utilise jamais root, mais un compte créé pour l'occasion, généralement Apache.
- **`Group`** : idem mais pour le groupe.

- **ServerAdmin** : adresse de courrier électronique de l'administrateur.
- **ServerName** : nom d'hôte (et port) du serveur. Il ne correspond pas forcément au nom d'hôte de la machine. Par contre il doit être valide. **ServerName www.mondomaine.org**
- **UseCanonicalName** : si c'est à **on**, Apache va répondre en utilisant les informations de ServerName et Port et pas les informations envoyées par le client. Par exemple, un http//192.168.1.3 se transforme en http://www.mondomaine.org
- **UserDir** : Nom d'un sous-répertoire ou chaque utilisateur peut placer ses fichiers HTML personnels. Généralement à public_html. On y accède avec **http://www.mondomaine.org/~login/page.html**
- **ErrorLog** : fichier où sont placées les logs d'erreur du serveur. **/var/log/httpd/error_log**.
- **CustomLog** : fichier journal de Apache. **/var/log/httpd/access_log**.
- **Timeout** : durée durant laquelle le serveur attend des émissions/réceptions pendant une communication. Elle est réglée sur 300 secondes.
- **KeepAlive** : définit si le serveur peut exécuter plus d'une requête par connexion. C'est à **off** par défaut mais si on passe à on Apache peut générer rapidement des processus enfants pour le soulager s'il est très chargé.
- **MaxKeepAliveRequests** : nombre maximum de requêtes par connexion persistante. Une valeur élevée peut augmenter les performances du serveur. 100 par défaut.
- **KeepAliveTimeout** : durée durant laquelle le serveur (généralement un processus fils) attend après avoir servi une requête. Par défaut 15 secondes. Après les 15 secondes, la demande sera réceptionnée par le serveur avec un T**imeout.**
- **StartServers** : nombre de serveurs créés au démarrage. Par défaut 8. Apache gérant ensuite dynamiquement le nombre de serveurs fils, ce paramètre a peu d'importance car le nombre va baisser au augmenter rapidement.

15.5 Gestion des performances

- **MaxRequestPerChild** : nombre de requêtes pouvant être exécutés par un processus fils avant de s'arrêter. Par défaut 4000. Ca permet une occupation mémoire plus réduite en la libérant plus rapidement.
- **MaxClients** : Limite du nombre total de requêtes pouvant être traitées simultanément. Par défaut 150. La valeur limite le risque de saturation du serveur.
- **MinSpareServers** / **MaxSpareServers** : Suivant la charge de la machine, Apache peut lancer d'autres processus serveurs pour s'adapter à la charge actuelle. Par défaut elles sont de 5 et 20. Ces deux valeurs déterminent les nombres limite autorisés. Si un serveur est peu chargé avec 15 processus, Apache en supprimera mais en gardera toujours au moins 5. Si le serveur devient chargé avec 10 processus, Apache en créera mais au maximum 20.
- **MinSpareThreads** / **MaxSpareThreads** : chaque serveur fils peut accepter un certain nombre de requêtes simultanément. Pour ça il utilise les threads. Ces deux valeurs, fixées par défaut à 20 et 75 et le mécanisme fonctionne comme ci-dessus.
- **ThreadsPerChild** : nombre de threads par défaut au lancement d'un serveur fils. Par défaut 25.

15.6 Les répertoires, alias et emplacements

Les balises **<Directory chemin>** et **</Directory>** permettent de regrouper des directives qui ne s'appliqueront qu'au chemin (et à ses sous-répertoires) données. La directive **Options** est fortement conseillée.

```
<Directory /var/www/html/images>
        Options +Indexes +FollowSymLinks
        DirectoryIndex index.php index.html
        Order allow, deny
        Allow from All
</Directory>

<Directory /var/www/html/cgi-bin>
        Options +ExecCGI
</Directory>
```

La directive **Options** accepte les valeurs suivantes précédées de + ou – et séparées par des espaces :

- **All** : toutes les options sauf MultiViews
- **Indexes** : Si jamais le répertoire ne contient pas de fichier HTML par défaut (cf DirectoryIndex), le contenu du répertoire est affiché sous forme de listing
- **ExecCGI** : l'exécution de scripts CGI est autorisée
- **FollowSymLinks** : le serveur suit les liens symboliques

La directive **DirectoryIndex** précise les fichiers html ou cgi par défaut lors du chargement d'une URL.

```
DirectoryIndex index.php index.html
```

Au chargement sans préciser le nom du fichier html, le serveur tentera de charger index.php et s'il est absent index.html. Dans le cas contraire, c'est l'option Indexes qui détermine si le contenu doit être visible sous forme de répertoire.

La directive **Allow** indique quels clients seront autorisés à accéder au répertoire. Ce peut être **all**, un domaine, une IP, un morceau d'IP (sous-réseau), une paire réseau/sous-réseau, etc. La directive **Deny** interdit l'accès et s'utilise de la même manière. L'ordre est déterminé par la directive **Order.**

La directive Alias permet de créer un raccourci entre l'arborescence logique du site web et un chemin du système de fichiers.

```
Alias /help "/usr/share/doc/html"
```

Dans ce cas, l'url **http://www.monsite.org/help** ne cherchera pas dans le répertoire **/var/www/html/help** mais dans **/usr/share/doc/html**.

Contrairement aux balises **<Directory>**, les balises **<Location>** et **</Location>** permettent d'appliquer des directives basées sur l'URL (et pas les répertoires).

```
<Location /help>
        Options +All -FollowSymLinks
        Order deny, allow
        Deny from all
        Allow from .mondomaine.org
</Location>
```

15.7 Hôtes virtuels

Note pour RHEL/CentOS : Pour pouvoir utiliser les hôtes virtuels, éditez le fichier /etc/httpd/conf.d/welcome.conf et supprimez les commentaires de toutes les lignes.

Un serveur Apache est capable de gérer plusieurs sites web sur un même serveur. Il existe plusieurs méthodes. La première se base sur les noms (plusieurs sites web pour un serveur) l'autre sur les ip (une adresse IP pour chaque site web). Nous allons voir la première version.

La directive **NameVirtualHost** spécifie l'IP sur laquelle le serveur va recevoir les requêtes d'accès aux hôtes virtuels.

Les balises **<VirtualHost>** et **</Virtualhost>** permettent de définir un hôte virtuel.

```
NameVirtualHost 192.168.1.3

<VirtualHost 192.168.1.3>
        ServerName              www2.mondomaine.org
        ServerAdmin             webmaster@www2.mondomaine.org
        DocumentRoot            /var/www/www2.mondomaine.org/
        ErrorLog                logs/www2_error_log
        CustomLog               logs/www2_access_log
</VirtualHost>
```

On relance Apache. Avec un navigateur (ex : Firefox) on vérifie si notre hôte virtuel répond avec l'URL http://www2.mondomaine.org (cette URL doit être déclarée dans /etc/hosts ou connue du serveur de noms et pointer sur le bon serveur).

On remarque cependant que si on passe par http://www.mondomaine.org on n'obtient plus le site par défaut ! En effet quand on déclare des hôtes virtuels, le premier de la liste devient l'hôte par défaut et prioritaire.

Quand on accède au serveur, Apache recherche d'abord une correspondance entre le nom d'hôte passé par l'URL et chaque ServerName des hôtes virtuels. Si aucune correspondance exacte n'est trouvée, c'est le premier hôte virtuel qui est pris par défaut, faisant abstraction des paramètres globaux.

Dans ce cas on rajoute un hôte virtuel pour le site principal.

```
<VirtualHost 192.168.1.3>
        ServerName              www.mondomaine.org
        ServerAdmin             webmaster@www.mondomaine.org
        DocumentRoot            /var/www/html
        ErrorLog                logs/error_log
        CustomLog               logs/access_log
</VirtualHost>
```

Attention, la règle ci-dessus s'applique aussi avec un nom court. Si on écrit http://www ou http://www2 ont tombera toujours sur l'hôte virtuel par défaut. Il faut écrire http://www.mondomaine.org et http://www2.mondomaine.org.

On peut placer dans un hôte virtuel toutes les directives souhaitées (ou presque).

16 Partage de fichiers

16.1 NFS

16.1.1 Lancement

Le partage de fichier **NFS** (**Network File System**) ou système de fichiers réseau permet de partager tout ou partie de son système de fichiers à destination de clients NFS, bien souvent d'autres Unix. Dans sa version de base c'est un système simple et efficace. Nous allons étudier la version 2.

NFS s'appuie sur le **portmapper** (**portmap**), le support **nfs** du noyau et les services **rpc.nfsd** et **rpc.mountd**. Sous RHEL/CentOS, les packages **portmap** et **nfs-utils** doivent être installés.

Pour lancer le service NFS, portmap et nfs doivent être lancés (vérifiez le statut avant).

```
service portmap status # ou rpcinfo -p
service nfs status
service portmap start
service nfs start
service nfslock start
```

Pour savoir si le service est disponible sur un hôte distant :

```
rpcinfo -p hote
```

16.1.2 Partage côté serveur

La liste des systèmes de fichiers à exporter se trouve dans **/etc/exports**. Il contient un partage par ligne.

```
# Rep exportes  Autorisations d'acces
/               poste1(rw) poste2(rw,no_root_squash)
/projects       *.mondomaine.org(rw)
/home/joe       poste*.mondomaine.org(rw)
/pub            192.168.1.0/255.255.255.0(ro)
```

Chaque ligne est composée de deux parties. La première est le chemin du répertoire exporté. La seconde contient les autorisations d'accès.

L'autorisation d'accès est composée de paires hôtes/permissions selon le format suivant :

```
host(permissions)
```

Si l'hôte n'est pas défini, c'est tout le réseau (portée dite mondiale) qui sera concerné par les permissions. Si les permissions ne sont pas définies, l'export sera en lecture seule. Il ne faut surtout pas mettre d'espaces entre l'hôte et les permissions. L'hôte peut être :

- un nom d'hôte unique
- un domaine
- un réseau ou sous-réseau
- une combinaison de l'ensemble avec des caractères de substitution (*, ?).

Les permissions peuvent être :

- **ro** : lecture seule
- **rw** : lecture écriture

- **no_root_squash** : le root distant équivaut au root local
- **root_squash** : si root se connecte au partage, son uid sera remplacé par celui d'un utilisateur anonyme. Ainsi il n'y a pas de risques que l'utilisateur root d'un poste local puisse être root sur un partage distant
- **all_squash** : étend la règle précédente à tous les utilisateurs
- **anonuid / anongid** : uid et gid pour l'utilisateur anonyme

Pour une gestion correcte des droits et des permissions, **les utilisateurs de même nom (login) doivent avoir les mêmes UID et GID sur le serveur et le client**. NFS se base en effet sur ces valeurs pour la sécurité des données du partage. Le nom de login seul ne suffit pas. Dans l'exemple ci-dessus, l'utilisateur « **joe** » est autorisé à accéder au partage **/home/joe** (on suppose que c'est son répertoire personnel) sur tous les postes du domaine. L'utilisateur joe doit être déclaré de la même manière (mêmes Ids) sur le serveur et sur tous les postes. C'est pour ça qu'on utilise souvent NIS avec NFS.

La commande **exportfs** permet de contrôler les partages.

- **exportfs -r** : rafraîchit la liste des partages après modification de /etc/exports
- **exportfs -v** : liste des partages
- **exportfs -a** : exporte (ou recharge) tous les partages de /etc/exports ou un partage donné
- **exportfs -u** : stoppe le partage donné. -a : tous

La commande **showmount** montre les partages d'un hôte donné.

```
showmount -e host
```

16.1.3 Montage côté client

Le support NFS est inclus sous forme de module du noyau. Il est automatiquement chargé à l'utilisation d'un accès NFS.

Dans **/etc/fstab** on note des modifications :

```
server1:/pub   /mnt/pub       nfs     defaults       0  0
```

Le périphérique est remplacé par le chemin du partage sous la forme **serveur:chemin**. Le système de fichiers est **nfs**. C'est pareil avec la commande mount :

```
mount -t nfs serveur1:/pub /mnt/pub
```

Si les montages NFS sont définis dans **/etc/fstab**, le service « **/etc/rc.d/init.d/netfs** » le montera automatiquement au démarrage (vérifier avec la commande **chckconfig**).

NFS dispose d'options de montage spécifiques :

- **nolock** : option de compatibilité avec d'anciens serveurs NFS
- **intr** : interrompt une requête NFS si le serveur ne répond pas
- **hard** : bloque les processus qui tentent d'accéder à un partage inaccessible
- **soft** : un processus retournera une erreur en cas d'accès infructeux.
- **rsize=8192**, **wsize=8192** : taille des blocs de lecture / écriture sur le serveur. Une écriture de 8ko est plus rapide que 8 écritures de 1ko.

16.2 FTP

Le serveur **FTP** (*File Transfert Protocol*) par défaut sur RHEL/CentOS est **vsftpd** (*Very Secure FTP Daemon*). Il a l'avantage d'être très petit, performant et rapide tout en étant tout de même très configurable (moins toutefois que Proftpd ou d'autres). Il convient dans la quasi-totalité des situations. C'est un service **xinetd** et se lance avec **chkconfig**.

Deux niveaux de sécurité sont possibles :

- **Anonyme** : tout le monde peut se connecter au serveur FTP en tant que utilisateur **ftp** ou **anonymous**. L'environnement FTP est chrooté.
- **Utilisateur** : Les utilisateurs qui existent sur le serveur peuvent se connecter avec leur mot de passe et ont un accès complet à leurs données dans leur répertoire personnel.

Le fichier de configuration est présent dans **/etc/vsftpd/vsftpd.conf**.

La racine du ftp par défaut est dans **/var/ftp**.

Le script de lancement est **/etc/rc.d/init.d/vsftpd** (**service vsftpd start**).

Pour activer ou non l'accès anonyme on modifie. Dans ce cas, l'utilisateur peut se connecter en tant que anonymous ou ftp. Dans tous les cas, il sera reconnu comme utilisateur « ftp » du serveur une fois connecté :

```
anonymous_enable=YES/NO
```

Pour activer ou non l'upload de fichiers sur le serveur par des anonymes. Dans ce cas, l'autorisation d'écriture dans un répertoire est fonction des droits du répertoire sur le serveur (notamment si l'utilisateur ftp a le droit d'écrire ou non dans un répertoire) :

```
anon_upload_enable=YES/NO
```

On peut interdire à des utilisateurs de se connecter en plaçant leur noms dans **/etc/vsftpd.ftpusers**.

On peut ajouter des utilisateurs dans **/etc/vsftpd.user_list** si **userlist_enable=YES**. Dans ce cas, c'est la valeur de **userlist_deny** (**YES**/**NO**) qui déterminera si le fichier contient les utilisateurs interdits ou autorisés.

On peut créer dans chaque répertoire du serveur un fichier **.message**. Dans ce cas, son contenu sera affiché lors de l'accès au répertoire.

Ex de connexion anonyme avec upload autorisé limité dans le répertoire /var/ftp/upload :

```
cd /var/pub
mkdir upload
chown root.ftp upload
chmod 730 upload
```

Contenu (en partie) de vsftpd.conf :

```
anonymous_enable=YES
anon_upload_enable=YES
chown_upload=YES
chown_username=blacknight
anon_umask=077
```

Résultat :

Tous les anonymes peuvent rentrer dans « upload » et y créer des fichiers car ils font partie du groupe « ftp » et disposent des droits w et x. Cependant, le fichier uploadé change de nom d'utilisateur (blacknight) et les droits du fichiers sont masqués en 600. Un anonyme ne peut pas lire

le contenu du répertoire, et le serveur lui interdira le téléchargement depuis ce répertoire.

16.3 Partages Windows avec Samba

16.3.1 Présentation

Samba est un ensemble de serveurs implémentant les protocoles SMB/CIFS et NetBIOS/WINS pour Unix. Son utilisation la plus connue est le partages de ressources entre Windows et Unix, mais il fonctionne parfaitement bien entre deux Unix. Un **partage** est aussi appelé **service**. Samba est composé de deux services :

- **smbd** : serveur SMB/CIFS
  - Authentification et autorisation
  - Partages de fichiers et d'imprimantes
- **nmbd** : serveur de noms NetBIOS
  - Parcours des ressoures
  - Serveur WINS

Un troisième service **winbindd** permet d'utiliser les comptes utilisateurs d'un domaine Microsoft. Les dernières versions de Samba (3 et +) permettent aussi de se raccorder à un **Active Directory**.

Les fonctions principales de Samba sont :

- Authentification des utilisateurs
- Partage de fichiers et d'imprimantes
- Parcours des ressources partagées du réseau
- Résolution de noms (indépendante de DNS) Nom Netbios<->IP

16.3.2 Configuration

16.3.2.1 Outils graphiques

RHEL/CentOS fournit un outil de configuration graphique pour Samba **system-config-samba**. Il est simple et performant pour des tâches simple : ajout de partages et ajout de nouveaux utilisateurs de base.

On lance Samba en deux fois :

```
service smb start
service nmb start
```

16.3.2.2 structure

La configuration de Samba se trouve dans **/etc/samba/smb.conf**. On peut tester sa syntaxe avec l'outil **testparm**.

Le fichier **smb.conf** reprend la syntaxe des fichiers de configuration de Windows de type « **ini** » avec des sections délimitées par des crochets **[]**.

Par défaut trois sections sont présentes :

- **[global]** : réglages génériques et globaux du serveur : nom, commentaires, méthode d'authentification, réglages par défaut, etc
- **[homes]** : partage des répertoires personnels des utilisateurs
- **[printers]** : partage des imprimantes

Les paramètres sont de la forme

```
nom = valeur
```

Les commentaires commencent par un point-virgule « **;** » ou un dièse « **#** ».

```
[global]
workgroup = MYGROUP
netbios name = posteN
security = share
```

- **workgroup** : nom de groupe / domaine de travail
- **netbios name** : nom netbios de la machine
- **security** : méthode d'authentification (cf plus bas).

16.3.2.3 Partage de fichiers

Chaque partage doit disposer de sa propre section dans **smb.conf**.

```
[partage1]
comment = Répertoire partagé 1
path = /opt/partage1
browseable = yes
public = no
writable = yes
printable = no
group = partage
```

- **partage1** : le nom du partage tel qu'il apparaît dans le « voisinage »
- **comment** : Le commentaire tel qu'il apparaît à côté du nom de partage.
- **path** : le chemin du partage
- **public** : le partage est accessible à l'utilisateur par défaut (guest)
- **browseable** : le partage apparaît dans le « voisinage »
- **writable** : le partage est accessible en lecture et écriture
- **printable** : le partage est une imprimante
- **group** : nom du groupe par défaut pour la connexion
- **valid users** : nom des utilisateurs autorisés à accéder à ce partage
- **read only** : le partage est en lecture seule pour tout le monde
- **guest ok** : aucun mot de passe n'est nécessaire pour accéder au partage. Dans ce cas le compte invité par défaut sera utilisé.
- **guest only** : le partage est accessible uniquement aux invités. Doit aêtre

L'accès aux partages se fait par défaut (voir partie sur les méthodes d'authentification) en fonction des droits Unix. A l'accès au partage, un nom d'utilisateur et un mot de passe sont demandés. Les

droits du répertoire partagé et de son contenu déterminent les droits de l'utilisateur.

16.3.2.4 Partage des imprimantes

En plus de la section **[printers]** on peut ajouter des sections pour des imprimantes spécifiques. Le paramètre « **printing =** » de la section **[global]** permet de modifier le sous-système d'impression basé par défaut sur **CUPS**.

```
[Bureau150]
comment = Lexmark Optra 630 bureau 150
printer = op630_150
valid users = riri fifi loulou
path = /var/spool/op630_150
public = no
writable = no
printable = yes
browseable = yes
```

- **printer** : nom de l'imprimante sous Linux
- **valid users** : nom des utilisateurs autorisés à accéder à ce partage. Un **@** devant le nom indique un groupe d'utilisateurs.
- **path** : chemin du spool d'impression

16.3.2.5 Méthodes d'authentification

Samba propose plusieurs méthodes d'authentification définies dans la section **[global]** :

- **user** : méthode par défaut : l'accès à l'ensemble des partages d'un serveur se fait par la validation d'un nom d'utilisateur et d'un mot de passe uniques.
- **share** : méthode de validation des identifiants partage par partage. Dans ce cas, tous les accès aux partages, même publics, nécessitent des identifiants.
- **domain** : utilisation d'un groupe de travail avec authentification
- **ads** : utilisation de **Active Directoty**

D'autres types d'authentification sont possibles comme le LDAP.

```
[global]
        workgroup = MYGROUP
        netbios name = posteN
        security = share
```

16.3.2.6 Correspondance des noms et mots de passe

Les mots de passe du protocole SMB n'ont pas la même forme que les mots de passe Unix/Linux. Il faut recréer les mots de passe pour chaque utilisateur devant utiliser SMB avec la commande **smbpasswd**. Les utilisateurs doivent déjà exister sous Unix.

```
smbpasswd -a toto
```

Les utilisateurs SMB sont présents dans **/etc/samba/smbpasswd**. La commande **mksmbpasswd** peut faire ça en batch :

```
cat /etc/passwd | mksmbpasswd > /etc/samba/smbpasswd
```

De même, on peut établir une table de correspondance entre les noms d'utilisateurs Windows et ceux de Unix dans **/etc/samba/smbusers**.

```
# Unix_name = SMB_name1 SMB_name2 ...
```

```
root = administrator admin administrateur
```

16.3.3 Clients SAMBA

16.3.3.1 En ligne

Toute machine sous Microsoft Windows peut accéder aux partages Samba. Les navigateurs des environnements de bureau **KDE** et **GNOME** acceptent la navigation dans les partages grâce au protocole « **smb:/** » dans les URL. KDE propose même l'équivalent d'un voisinage réseau.

L'outil **smbclient** est une sorte de FTP pour le protocole SMB. Les chemins d'accès aux ressources sont de la forme **//machine/partage**.

```
smbclient //machine/service -U login%passwd #connexion
smbclient -L hostname -U login%passwd # liste des ressources
```

16.3.3.2 Montage

On peut monter un système de fichiers SMB avec la commande **smbmount**.

```
smbmount //machine/partage /mnt/mountpoint -o username=login
```

On peut aussi faire la même chose dans **/etc/fstab** :

```
//machine/partage /mnt/mountpoint smbfs defaults,username=nobody 0 0
```

17 Bases de sécurité services et réseau

17.1 Les tcp_wrappers

Les **enveloppeurs TCP** ou tout simplement **tcp_wrappers** permettent la vérification des accès à uns service réseau donné (service, xinetd, portmapper). Chaque programme utilisant les tcp_wrappers est compilé avec la bibliothèque **libwrap** de manière statique (la commande **ldd** ne permet pas de voir la bibliothèque).

Pour savoir si un service réseau est compilé avec libwrap, on tape la commande suivante :

```
strings -f <binaire> | grep host_access
```

Si aucune ligne n'est retournée, le programme n'utilise pas les tcp_wrappers.

Parmi les services utilisant les tcp_wrappers, on trouve :

- **sendmail** (y compris postfix)
- **sshd** (ssh)
- **xinetd** (et donc indirectement tous les services associés)
- **vsftpd** (ftp)
- **portmap** (et donc nis, nfs)
- **in.telnetd** (telnet) ainsi que la plupart des services supportés par xinetd
- **dovecot** (imap, pop)

La vérification d'accès à un service enveloppé TCP se fait en trois étapes :

- l'accès est-il explicitement autorisé ?
- Sinon, l'accès est-il explicitement interdit ?
- Sinon, par défaut, l'accès est autorisé.

Les fichiers de configuration sont **/etc/hosts.allow** et **/etc/hosts.deny**. La syntaxe est commune :

```
daemon_list : client_list [:options]
```

- **daemon_list** : liste des **exécutables** (**PAS DES SERVICES**) séparés par des virgules. On peut mettre **ALL** pour spécifier tous les services. Si on dispose de plusieurs interfaces réseau on peut utiliser la syntaxe avec @ : service@ip.

```
in.telnetd: ...
sshd, portmap: ...
sshd@192.168.1.7 : ...
```

- **client_list** : clients autorisés ou interdits pour ce service. On peut spécifier l'adresse IP, le nom, le masque de réseau, le nom du réseau, etc.

```
... : 192.168.1.7, 192.168.1.8
... : 192.168.1.
... : poste1, poste2
... : 192.168.1.0/255.255.255.0
... : .mondomaine.org
```

La liste des clients admet une syntaxe avancée :

- **ALL** : correspondance systématique

- **LOCAL** : tous les hôtes dont le nom ne contient pas de point (poste1, poste2, etc)
- **UNKNOWN** : hôtes dont le nom ne peut pas être résolu
- **KNOWN** : hôtes dont le nom peut être résolu
- **PARANOID** : hôtes dont le nom ne peut être résolu ou dont l'IP n'a pas de résolution inverse.
- **EXCEPT** : permet d'exclure certains hôtes

```
ALL EXCEPT poste10
```

/etc/hosts.allow est lu en premier, puis **/etc/hosts.deny**. La recherche s'arrête à la première correspondance. Une ligne dans hosts.allow autorise la connexion. Une ligne dans hosts.deny interdit la connexion. Si l'accès n'est pas explicitement refusé, la connexion est autorisé : la requête ne correspond à aucun critère.

```
# /etc/hosts.allow
vsftpd:               192.168.1.
in.telnetd, portmap:  poste1, poste2

# /etc/hosts.deny
ALL : .baddomaine.org except trusted.baddomaine.org
vsftpd,,portmap : ALL
dovecot : 192.168.0. EXCEPT 192.168.1.5
```

17.2 Netfilter

17.2.1 Présentation

Netfilter est une architecture de filtrage des paquets pour les noyaux Linux 2.4 et 2.6. Le filtrage se fait au sein même du noyau au niveau des couches 2, 3 et 4 du modèle OSI c'est à dire les liaisons données, réseau et transport. C'est à dire qu'il est par exemple capable d'agir à bas niveau sur les interfaces ethernet (2), sur la pile IP (3) et sur les protocoles de transport comme TCP (4). Le filtrage est **stateless** : comme Netfilter n'inspecte que les entêtes des paquets, il est extrêmement rapide et n'entraîne pas de temps de latence.

L'inspection des contenus des paquets (protocoles applicatifs) peut se faire à l'aide d'extensions mais devrait cependant être réservé à des outils en espace utilisateur.

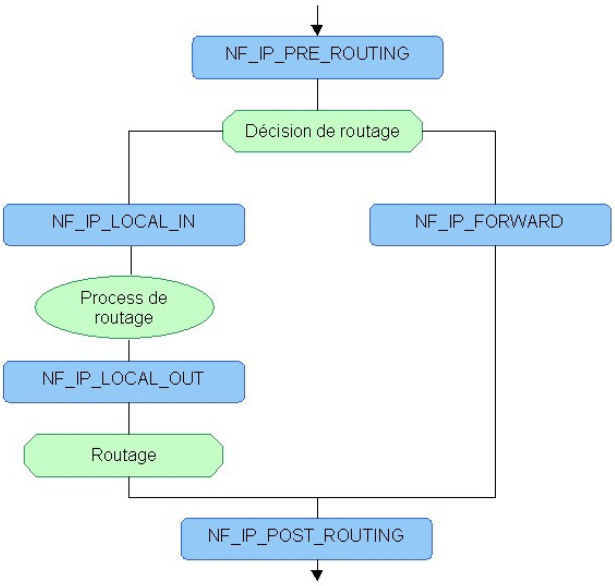
Autrement dit, Netfilter est un firewall au niveau protocole.

Le programme utilisateur permettant d'agir sur les règles de filtrage est **iptables**.

L'implémentation au niveau du noyau se fait par des modules.

17.2.2 Vie d'un paquet

Plutôt qu'un long discours, voici un schéma. Les paquets arrivent par le haut et ressortent par le bas. Entre les deux, il passe par différents états au niveau de netfilter.



Chaque état (carré bleu) correspond à un point de filtrage possible par la commande **`iptables`**.

- **`PREROUTING`** : Traite les paquets à leur arrivée. Si le paquet est à destination du système local, les paquets seront traités par un processus local (INPUT, OUTPUT). Sinon, et si le forwarding est activé, les règles FORWARD et POST_ROUTING seront appliquées.
- **`FORWARD`** : les paquets ne font que traverser le système local. Traite les paquets routés à travers le système local.
- **`INPUT`** : traite les paquets destinés au système local, en entrée (après le routage)
- **`OUTPUT`** : traite les paquets quittant le système local, avant POSTROUTING
- **`POSTROUTING`** : traite les paquets immédiatement avant leur sortie du système.

17.2.3 Principe des règles

Lorsqu'un paquet est traité par netfilter, il l'est par rapport à un certain nombre de règles qui déterminent ce qu'il faut en faire.

- Les règles sont ordonnées : la position d'une règle dans une liste représente quand et si la règle sera utilisée.
- Les paquets sont testés avec chacune de règles, l'une après l'autre.
- Netfilter fonctionne selon le mécanisme de la première correspondance. Si une règle correspond les autres règles sont négligées, la vérification s'arrête.
- Une règle peut spécifier plusieurs critères.
- Pour qu'une règle corresponde à un paquet, tous les critères doivent correspondre.
- Si malgré toutes les règles, le paquet passe, une règle par défaut peut être appliquée.

17.2.4 Cibles de règles

Une cible de règle détermine quelle est l'action à prendre lorsqu'un paquet correspond aux critères d'une règle. On utilise l'option « **`-j`** » de **`iptables`** pour spécifier la cible.

Les deux cibles de base sont **`DROP`** et **`ACCEPT`**. Des extensions de netfilter ajoutent d'autres règles

comme **LOG** ou **REJECT**.

- **DROP** : le paquet est rejeté. Aucune notification n'est envoyée à la source.
- **REJECT** : le paquet est rejeté, retournant une erreur à la source.
- **ACCEPT** : le paquet est accepté.
- **LOG** : une information est envoyée à syslog pour les traces.

On peut créer des règles sans cible. Dans ce cas, la règle incrémentera un compteur de paquets et un compteurs d'octets associés à la règle afin d'établir une collecte de statistiques.

17.2.5 Premier exemple

Voici une règle simple qui interdit tous les paquets en provenance de 192.168.1.11.

```
iptables -A INPUT -s 192.168.1.11 -j DROP
```

- **-A** : Point de filtrage (INPUT, OUTPUT, FORWARD, PREROUTING, POSTROUTING) qu'on appelle aussi **chaîne**.
- **-s** : Source, peut être une IP, un hôte, un réseau, etc.
- **-j** : jump, cible de la règle (ACCEPT, DROP, REJECT, ...)

Résultat : on interdit en entrée les paquets dont la source est 192.168.1.11.

17.2.6 Opérations de base

Les règles sont numérotées à partir de 1. On ajoute une règle avec « **-A** ». On insère avec un « **-I** » à la position souhaitée.

```
iptables -A INPUT -s 192.168.1.2 -j DROP
iptables -I OUTPOUT -d 192.168.1.25 -j DROP 3 # inserer à la 3eme position
```

On supprime un règle avec le « **-D** ».

```
iptables -D INPUT 1 # supprime la règle 1
```

On liste les règles avec le « **-L** ».

```
iptables -L OUTPUT
iptables -L
```

On utilise « **-F** » pour supprimer l'ensemble des règles.

```
iptables -F
```

17.2.7 Critères de correspondance

17.2.7.1 Général

Les critères de correspondance déterminent la validité d'une règle. Tous les critères doivent être vérifiés pour qu'un paquet soit bloqué. Les critères de base sont :

- **-i** : interface entrante (filtrage couche 2)
- **-o** : interface sortante (filtrage couche 2)
- **-p** : protocole de couche 4. Pour les noms, voir le fichier **/etc/protocols**

- **-s** : Adresse IP de la source (ou réseau)
- **-d** Adresse IP de destination (ou réseau)

Interdire les entrées par eth0

```
iptables -A INPUT -i eth0 -j DROP
```

Interdire le forward entre eth1 et eth2

```
iptables -A FORWARD -i eth1 -o eth0 -j DROP
```

Interdire le protocole ICMP en entrée (le ping !)

```
iptables -A input -p icmp -j DROP
```

17.2.7.2 TCP, UDP et ICMP

Suivant le protocole (couche 4) certaines options sont possibles. C'est le cas de tcp, udp ou icmp (notamment utilisé par ping). Le filtrage au niveau des protocoles est généralement effectué par des extensions à netfilter.

- **-p** : protocole (tcp, udp, icmp, etc)
- **--sport** : port source
- **--dport** : port destination

Si nous souhaitons par exemple interdire les connexion entrantes à destination du le port 80 (serveur httpd) nous ferions ainsi

```
iptables -A INPUT -p tcp –dport 80 -j DROP
```

17.2.7.3 Arguments des critères

Pour les adresses, on peut spécifier :

- un hôte par son nom ou son IP
- un réseau par son nom ou son masque (192.168.1.0/24, 192.168.1.0/255.255.255.0)

Pour les ports on peut spécifier :

- un numéro
- un nom (voir **/etc/services**)
- une gamme de ports : **123:1024**

Dans tous les cas, on peut utiliser la négation avec « **!** ».

```
iptables -A INPUT -s ! 10.0.0.1 -j DROP
```

17.2.8 Sauver ses réglages

Les règles définies avec iptables sont chargées en mémoire et transmises dynamiquement au noyau. En redémarrant la machine, elles sont perdues. RHEL/CentOS permet de sauver l'ensemble des règles de manière à les rendre persistantes.

```
service iptables save
```

Les règles sont sauvées dans le fichier **`/etc/sysconfig/iptables`**.

Enfin, il faut savoir que les règles iptables sont chargées AVANT le réseau. Ainsi il n'y a aucun risque de sécurité car les règles seront directement valides à l'activation du réseau.

Index lexical

accept....34
ACL....19
aliases....70
alternatives....70
Apache....72
aquota.group....30
aquota.user....30
authconfig....32
bashrc....27
Bind....60
blockdev....16
broadcast....49
cancel....33
cfdisk....16
chage....26
chgrp....26
chkconfig....14, 55
chmod....26
CIFS....79
cron....35
cron.d....36
crontab....35
CUPS....34
cupsd....34
cupsd.conf....35
cyrus-imap....71
delta-rpm....20
depmod....39
DHCP....57
dhcpd....57
dhcpd.conf....57
dhcpd.leases....57
dig....67
Disk Druid....8
dmesg....12
DNS....59
dovecot....71
dovecot.conf....71
DPKG....20
edquota....30
exportfs....77
exports....76
ext2....16
ext3....16
Faulty Disk....43
fdisk....16
FQDN....59
fstab....18, 30, 77, 82
FTP....78
fuser....18
getfacl....19
group....25
groupadd....26
groupdel....26
groupmod....26
grpquota....30
grub-install....10
grub.conf....10
host....66, 68
hostname....48
hosts....53
hosts.allow....83
hosts.deny....83
httpd....72
httpd.conf....72
ifcfg-xxx....51
ifconfig....50
ifdown....51
ifup....51
IMAP....69, 71
init....12
init.d....13
initrd....12
inittab....12, 15
insmod....39
IPP....34
iptables....84, 88
kernel....38
klogd....36
ldd....83
libwrap....83
logrotate....36
lp....33
lpadmin....33
lpc....34
lpd....33
lpq....34
lpr....34
lprm....34
lpshut....34
lpstat....33
lsmod....39
maillog....70
main.cf....70
MDA....69
mdadm....44
mdadm.conf....44
mkfs....16
mod_php....72
modinfo....39
modprobe....39
modprobe.conf....39
modules.dep....39
mount....18
msdos....17
MTA....69
MTBF....42
MUA....69
MX....69
named....59
named-checkconf....67
named-checkzone....67
named.conf....60
neat....50
NetBIOS....79
netconfig....50
Netfilter....84
netfs....77
netmask....48, 49
network....31, 52
networks....53
newgrp....26
NFS....76
nmbd....79
nologin....29
nologinusers....29
nsswitch.conf....31
ntsysv....14
OpenSSH....56
PAM....27
pam.d....28
parted....16
partprobe....16
passwd....25
POP....69, 71
portmap....31
portmapper....76
postconf....70

postfix....70
PPD....34
printcap....33
printers.conf....35
proc....40
profile....27
profile.d....27
protocols....86
quota....31
quotacheck....30, 31
quotaoff....30
quotaon....30
RAID....42
raidtools....44
rc....13
rc.local....14
rc.sysinit....13
reiserfs....17
reiserfstune....17
reject....34
repquota....31
resolv.conf....52
resolver....52
rmmod....39
route....52
rpc.mountd....76
rpc.nfsd....76
rpcinfo....76
RPM....20
rpmdb-redhat....24
run-parts....36
runlevel....12
Samba....79
SELinux....9
sendmail....69
service....14, 55
serviceconf....14
services....87
setfacl....19
setgid....26
sfdisk....16
shadow....25
showmount....77
skel....25
SLED....42
SMB....79
smb.conf....79
smbd....79
smbmount....82
SMTP....69
Spare Disk....42
ssh....56
sshd_config....56
swap....8
sys....40
sysctl....41
sysctl.conf....13
syslog.conf....36
syslogd....36
system-auth....29
system-config-network....50
system-config-packages....21
system-config-printer....35
system-config-samba....79
tcp_wrappers....83
telinit....12
TLD....59
tmpwatch....36
tune2fs....17
umount....18
uname....38
up2date....24
useradd....25
userdel....25
usermod....25
usrquota....30
vfat....17
vsftpd....78
vsftpd.conf....78
vsftpd.ftpusers....78
vsftpd.user_list....78
warnquota....31
winbindd....79
WINS....79
xinetd....53, 78
xinetd.conf....53
xinetd.d....53
yp.conf....31
ypbind....31
zone....61
.bash_profile....27
.bashrc....27