UNIX

ADMINISTRATION AVANCEE

Version 2.0

- -

1. Table des matières

- -

2. INITIALISATION ET ARRÊT DU SYSTÈME

2.1 CHARGEMENT DU SYSTÈME - PREMIERS PROCESSUS

RAPPELS SUR LES PROCESSUS

Toute demande d'exécution d'un programme (fichier ordinaire exécutable) provoque la création par le système d'un **processus** ("process" ou "tâche").

UN PROCESSUS CONSTITUE L'ENVIRONNEMENT D'EXÉCUTION D'UN PROGRAMME.

L'activité du système se traduit par la présence simultanée de nombreux processus dans une politique de **temps partagé** dans laquelle le noyau gère le partage des ressources (accès à la mémoire, au(x) processeur(s) etc...).

Mécanisme fork + exec

Une connexion interactive consiste en la présence d'un processus qui exécute le programme shell.

Que se passe-t-il (en terme de processus) lorsque l'utilisateur lance une commande ?

Processus initial (parent)
Exécute le shell --- *fork* ---> **Nouveau processus (child)**
(duplication du processus initial)

wait
(attente de la fin
du processus généré)

exec commande
(changement de programme dans
le cadre du même processus)

exit (code retour)
(fin du processus)

Déblocage, on peut enchaîner
sur une autre commande ...

Principaux atributs d'un processus

PID	Numéro identifiant le processus (unique à un instant donné)
PPID	Numéro du processus "parent"
UID	Propriétaire effectif Par défaut, il s'agit du "login" ayant lancé le programme. Si le programme possède la permission **SUID**, le propriétaire effectif du processus devient celui du programme.
GUID	Groupe du propriétaire Par défaut, il s'agit du groupe principal du "login". Si le programme possède la permission **SGID**, le groupe propriétaire du processus devient celui du programme.
TTY	Terminal de contrôle Il n'est pas toujours défini (processus de type "**daemon**").
NICE VALUE	Valeur de priorité de départ

La commande ps

Cette commande "ps" (process status) permet d'obtenir différentes informations sur les processus.
Ses options ainsi que le détail de l'affichage varient selon les versions.

Sans option, elle donne la liste des processus attachés à son propre terminal.
Les options **-l** ou **-f** permettent d'obtenir plus de détails.
Avec d'autres options, il est possible de sélectionner certaines familles de processus.

-e	Tous les processus
-t *tty*	Tous ceux contrôlés par le terminal de nom *tty*
-u *login*	Tous ceux d'un propriétaire donné

Exemple

```
solaris> ps  -ef
  UID   PID  PPID  C    STIME   TTY    TIME    COMD
 root     0     0 80  jan  26   ?      0:21    sched
 root     1     0 80  jan  26   ?      1:13    /etc/init -
 root     2     0 35  jan  26   ?      0:00    pageout
 root     3     0 80  jan  26   ?      5:26    fsflush
 root   170     1 13  jan  26   ?      0:00    /usr/lib/saf/sac -t 300
 root   139   131 16  jan  26   ?      0:00    lpNet
 root    66     1 80  jan  26   ?      0:01    /usr/sbin/rpcbind
 root    83     1 80  jan  26   ?      0:01    /usr/sbin/inetd -s
.....................................
.....................................
```

PROCÉDURE DE BOOT - PREMIERS PROCESSUS

Les procédures de chargement varient fortement d'une machine à une autre.
Cependant, sur la plupart des systèmes, il existe la possibilité d'opérer cette initialisation en *mode automatique* ou en *mode manuel* (ou "maintenance").

Dans le premier cas, le démarrage se fait directement à partir des informations situées sur le disque système et le fichier correspondant au **noyau** est **chargé en mémoire**.

En mode manuel, on peut indiquer quel sera le périphérique utilisé pour le chargement (disquette, bande, disque dur ...). Il est alors possible de choisir un autre fichier noyau .

Ce mode peut être invoqué dans plusieurs cas :

- La première fois, lors de l'*installation*,
- En cas de problème, pour une *restauration* ou pour *corriger un problème*,
- Pour l'essai d'un noyau personnalisé,
- Pour opérer des *diagnostics* ...

Une fois chargé, le système d'exploitation s'initialise lui-même et met en place toutes ses structures de données.

Le premier processus créé se nomme souvent **sched** ou **swapper** (*PID = 0*).

Il crée lui-même d'autres processus associés dont les PIDs démarrent au chiffre 2 .

Dans tous les cas, est créé ensuite le processus **init** (*PID = 1*) , ancêtre de tous les futurs processus du système.
Le paramétrage éventuel de l'initialisation du système se fait au niveau de ce programme "init".

2.2 LE PROGRAMME INIT

NIVEAUX D'EXÉCUTION

Un niveau d'exécution est une configuration du système qui n'autorise l'existence que d'un groupe donné de processus.
Chaque version Unix propose un certain nombre de *niveaux prédéfinis paramétrables*
et donne souvent la possibilité de créer des niveaux personnalisés.

Niveaux System V (Unix V.4, Sco, Hp/Ux 10 ...)

s ou S	Environnement mono-utilisateur de "maintenance" (On ne peut se connecter qu'à la console)
Niveau 0	État stable permettant de couper l'alimentation électrique
Niveau 1	Variante peu utilisée du niveau "s"
Niveau 2	Mode multi-utilisateurs classique
Niveau 3	Niveau 2 + certains processus liés au réseau et à X_Window
Niveau 4	Disponible pour une configuration personnalisée
Niveau 5	Arrêt du système et redémarrage
Niveau 6	Quasiment identique au niveau 5

NB: La définition exacte des niveaux est à vérifier dans la documentation car les variations sont nombreuses par rapport à ces définitions "historiques" du System V.

Niveaux AIX

s ou S ou m ou M	Niveau maintenance (mono-utilisateur)	
Niveaux 0 à 9	*0 , 1*	Réservés pour un usage futur
	2	Mode multi-utilisateurs (niveau par défaut)
	3 à 9	Disponibles pour une configuration personnalisée

Anciennes Versions Berkeley

Dans ces versions, il n'existe que deux niveaux "single-user" et "multi-user".

LE FICHIER INITTAB

Le programme "init" (PID = 1) est lancé, une première fois, lors du démarrage du système, par le noyau chargé en mémoire.
Il peut, par la suite, être relancé par l'administrateur (sauf dans les versions Berkeley).

Son objectif est de placer le système dans un niveau d'exécution donné ou de modifier certains aspects du niveau courant.

Pour effectuer les traitements souhaités, le programme consulte le fichier **/etc/inittab** .

init [sS0123456789qQ]

s , S	Passage au mode maintenance (single-user)
0 à 9	Activation d'un niveau
q ou Q	Relecture immédiate du fichier /etc/inittab (sans changer de niveau)

Le fichier /etc/inittab

Il est organisé en lignes constituées de *quatre champs séparés par le caractère :* .

Identificateur : Niveau : Action : Commande

Une entrée peut se continuer sur la ligne suivante par un \ .
Des commentaires peuvent apparaître (# en première colonne).
Le fichier ne doit pas comporter de lignes vides.

Identificateur	Identificateur **unique** de l'entrée
Niveau	Niveaux pour lesquels la ligne est valide Un champ vide signifie : "tous les niveaux"
Action	Comportement vis à vis du champ "Commande"
initdefault	Détermine le niveau d'exécution initial
sysinit	Entrée examinée avant d'accéder à la console Lancer le programme et attendre la fin (entrée examinée uniquement lors du démarrage du système)
boot	Lancer le programme sans attendre la fin (entrée examinée uniquement lors du démarrage du système)
bootwait	Lancer le programme et attendre la fin (entrée examinée uniquement lors du démarrage du système)

Le fichier /etc/inittab (suite)

wait	Lancer le programme associé et attendre la fin (entrée examinée uniquement lors d'un changement de niveau)
respawn	Lancer le programme associé sans attendre la fin Le relancer dès qu'il se termine Si le programme est déjà actif, ne rien faire
off	Ignorer la ligne Éliminer le programme si celui-ci est actif
once	Lancer, une seule fois, le programme sans attendre la fin
Commande	Programme associé Utiliser les **noms complets** et des **redirections explicites**

Extraits de fichier /etc/inittab (Solaris)

```
ap::sysinit:/sbin/autopush -f /etc/iu.ap
ap::sysinit:/sbin/soconfig -f /etc/sock2path
fs::sysinit:/sbin/rcS sysinit >/dev/msglog 2<>/dev/msglog </dev/console
is:3:initdefault:
sS:s:wait:/sbin/rcS  >/dev/msglog 2<>/dev/msglog </dev/console
s0:0:wait:/sbin/rc0  >/dev/msglog 2<>/dev/msglog </dev/console
s2:23:wait:/sbin/rc2 >/dev/msglog 2<>/dev/msglog </dev/console
s3:3:wait:/sbin/rc3  >/dev/msglog 2<>/dev/msglog </dev/console
s5:5:wait:/sbin/rc5  >/dev/msglog 2<>/dev/msglog </dev/console
s6:6:wait:/sbin/rc6  >/dev/msglog 2<>/dev/msglog </dev/console
fw:0:wait:/sbin/uadmin 2 0  >/dev/msglog 2<>/dev/msglog </dev/console
of:5:wait:/sbin/uadmin 2 6  >/dev/msglog 2<>/dev/msglog </dev/console
rb:6:wait:/sbin/uadmin 2 1  >/dev/msglog 2<>/dev/msglog </dev/console
sc:234:respawn:/usr/lib/saf/sac -t 300
```

Extraits de fichier /etc/inittab (Aix)

```
init:2:initdefault:
brc::sysinit:/sbin/rc.boot 3 >/dev/console 2>&1
rc:2:wait:/etc/rc 2>&1 | alog -tboot > /dev/console
fbcheck:2:wait:/usr/sbin/fbcheck 2>&1 | alog -tboot > /dev/console
srcmstr:2:respawn:/usr/sbin/srcmstr
rctcpip:2:wait:/etc/rc.tcpip > /dev/console 2>&1
```

2.3 PASSAGE AU MODE MULTI-UTILISATEURS

Suivant les versions, le mode "multi-utilisateurs" correspond au **niveau 2, 3 ou 4 .**
L'examen du fichier /etc/inittab permettra de détailler l'enchaînement des diverses étapes.

VERSIONS AU "PROTOCOLE" SYSTEM V (SOLARIS , SCO, HP/UX ...)

Premières initialisations

Entrées de type sysinit , boot , bootwait (liées au matériel)

Éventuels scripts ***bcheckrc*** *et/ou* ***brc***

Initialisation de certains fichiers système,
Vérification des systèmes de fichiers ...

Exécution du script ***rcN*** (N = Numéro du niveau concerné)

Entrée de type "wait"
rcN utilise le répertoire **/etc/rcN.d** ou **/sbin/rcN.d**
pour exécuter les procédures **K*** avec l'argument **stop**
puis les procédures **S*** avec l'argument **start** .
Ces procédures sont des liens sur des procédures
génériques situées dans **/etc/init.d** ou **/sbin/init.d** .

A l'aide de ces diverses procédures, on peut :

- Monter les systèmes de fichiers
- Lancer certains services :
  - Gestionnaire d'imprimantes (lpsched),
  - Planificateur de travaux (cron),
  - Service de statistiques (sa),
  - Service de comptabilité (acct) ...
- Démarrer les divers services et démons réseau
- Démarrer les aspects X_Window

etc...

Activation des terminaux asynchrones

Entrées de type "respawn" liées au programme "**getty**"
ou Lancement du service "**sac**" et des processus "**ttymon**"

Paramétrage

Un paramétrage se fera, a priori, au niveau des procédures des sous-répertoires de **/etc/rcN.d .**
L'esprit du mécanisme proposé est de ne pas intervenir dans le fichier "/etc/inittab" lui-même.

VERSION AIX

rc.boot (script shell)	Entrée de type *sysinit* Tests hardware Montage des filesystems de base Activation de l'espace de pagination
rc (script shell)	Entrée de type *wait* Activation des groupes de volumes applicatifs Activation des espaces de pagination complémentaires Montage des systèmes de fichiers Lancement optionnel de certains services ...
srcmstr (exécutable)	Entrée de type *respawn* Lancement du SRC (service spécifique AIX)
rc.tcpip (script shell)	Entrée de type *wait* Lancement des différents démons TCP/IP
rc.nfs (script shell)	Entrée de type *wait* Lancement des différents démons NFS et NIS
Entrées individuelles pour le lancement de **divers services**	"cron" (planificateur de travaux) "qdaemon" (gestionnaire d'imprimantes) etc...
rc.dt (script shell)	Entrée de type *wait* Lancement du CDE (Common Desktop Environment)
getty (exécutables)	Entrées de type *respawn* Activation des terminaux asynchrones

Paramétrage

Une méthode simple consiste à créer une nouvelle ligne dans /etc/inittab , par exemple :

rclocal:2:wait:/etc/rc.local # Programmes locaux

Le fichier **/etc/rc.local** sera un script shell réalisant le lancement souhaité des diverses commandes.

L'usage de la **commande su** sera peut-être nécessaire si ces commandes doivent être démarrées dans le contexte d'un login particulier et avec un environnement précis (par exemple, le lancement d'une base de données).

/usr/bin/su - nom_login -c "ligne-de-commande"

2.4 ARRÊT ET REDÉMARRAGE DU SYSTÈME

Tout comme pour l'aspect démarrage, l'examen de la documentation permettra de détailler les protocoles et les syntaxes précises des commandes d'arrêt.

Le comportement habituel de ces commandes est le suivant :

- Prévenir les utilisateurs connectés
- Marquer un "délai de grâce" (60 secondes par défaut)
- Éliminer les processus "démons" et utilisateurs
- Démonter les systèmes de fichiers
- Mettre à jour les fichiers sur disque (commande interne "**sync**").

Versions System V

/usr/sbin/shutdown -y -gdélai -iniveau

Options

-y	Commande non interactive (Réponse "yes" à toutes les questions)
-g	Changer le délai d'attente par défaut (nombre de secondes)
-i	Indiquer le niveau d'exécution souhaité
-i0	Arrêter le système
-i5 ou -i6	"Rebooter" (arrêt puis redémarrage)

Version AIX

/usr/sbin/shutdown [options] [Time [Message]]

Quelques options

-F	Arrêt immédiat
-h	Arrêt complet (option par défaut, peu utilisée)
-r	Redémarrage ("reboot")

Time

+nombre	Nombre de minutes
hh:mm	Heure absolue

Message Message supplémentaire pour les utilisateurs

Version HP/UX

/etc/shutdown [-h | -r] [-y] [grace]

Options

-h	Arrêt complet du système
-r	Redémarrage ("reboot")
-y	Commande non interactive (Réponse "yes" à toutes les questions)
grace	Délai en secondes (60 par défaut)

2.5 SMF (Service Management Facility) des versions Solaris 10

Le nouveau service baptisé **SMF** (*Service Management Facility*) constitue une des nouveautés les plus intéressantes de Solaris 10. Son objectif est de simplifier et d'améliorer la **gestion des services et des applications** (locales ou distantes).

Le processus **/lib/svc/bin/svc.startd** concrétise l'implémentation du SMF. Il est activé par une entrée de type *sysinit* dans le fichier **/etc/inittab** de Solaris 10.

Nous pouvons résumer les avantages de ce nouveau service SMF :

Administration simplifiée

Les services sont des objets qui peuvent être aisément gérés par un petit nombre de commandes simples.

Redémarrage automatique des services en erreur

Persistance de la configuration

Les définitions des services et les configurations sont permanentes, même après l'installation de mises à jour.

Gestion des dépendances

SMF permet de définir des relations de dépendance entre les services afin de ne pas démarrer un service tant que ceux dont il dépend ne sont pas en activité.

Amélioration des traces

Des fichiers journaux individuels facilitent la détermination des problèmes.

Accélération du démarrage du système

Quand cela est possible, les services sont démarrés en parallèle.

*La plupart des services qui étaient précédemment démarrés depuis les scripts **rcn** associés aux niveaux d'exécution, ne sont plus démarrés via ce protocole. Pour des raisons de compatibilité, ce genre de scripts continue à être pris en compte mais il est fortement recommandé d'opérer la **migration** vers le nouveau procédé SMF.*

Composantes d'un service SMF

Un service SMF est caractérisé par un certain nombre d'entités :

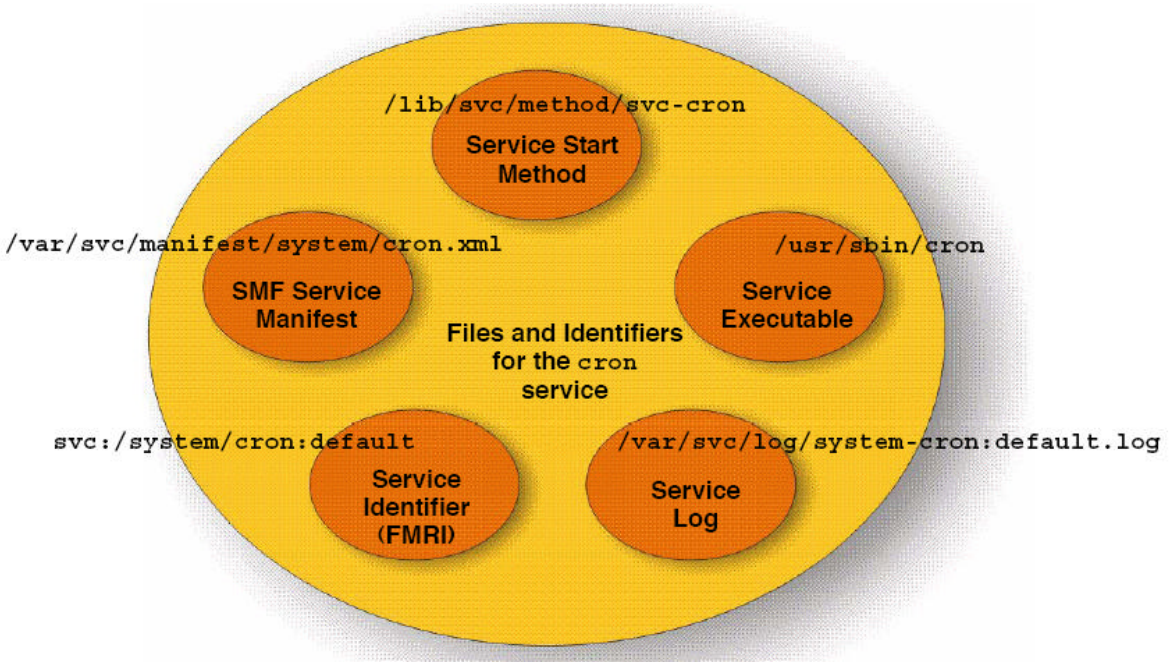
- un **SMF manifest** (propriétés par défaut du service).

- une ou plusieurs **méthodes** qui définissent les interactions avec le **service restarter** (service de redémarrage en cas d'erreur).

- un ou .plusieurs **exécutables** (démons).

- un fichier **journal** (*log*).

- un **FMRI** (*Fault Management Resource Identifier*) qui correspond à l'identification d'une instance du service.



Composantes d'un service SMF (exemple du service cron)

Une instance de service peut prendre plusieurs états, tels que :

- **online**

Le service doit être activé au démarrage du système et il a bien démarré.

- **disabled**

Le service ne doit pas être activé au démarrage du système et il n'est pas en activité.

- **offline**

Le service doit être activé au démarrage du système mais il est en attente de l'activation de services dont il dépend.

- **maintenance**

Le service est en erreur.

- **legacy_run**

Le programme est en activité mais ne peut pas être contrôlé par le *SMF*. Il s'agit sans doute d'un script *rcn* de l'ancien protocole.

Les outils du service SMF

SMF fournit un certain nombre de commandes de gestion.

svcs

Cette commande fournit des informations sur les instances de services.

svcs -a

Liste de tous les services, quel que soit leur état.

svcs -p [FMRI]

Liste des processus associés aux instances de service.

```
# svcs -p svc:/system/sac:default
STATE          STIME    FMRI
online         18:31:21 svc:/system/sac:default
               18:31:21      241 sac
               18:31:22      259 ttymon
#
```

svcs -l FMRI

Détails sur un service particulier.

```
# svcs -l svc:/network/ssh:default
fmri         svc:/network/ssh:default
name         SSH server
enabled      true
state        online
next_state   none
state_time   Wed Feb 24 18:31:33 2006
logfile      /var/svc/log/network-ssh:default.log
restarter    svc:/system/svc/restarter:default
contract_id  51
dependency   require_all/none svc:/system/filesystem/local (online)
dependency   optional_all/none svc:/system/filesystem/autofs
             (online)
dependency   require_all/none svc:/network/loopback (online)
dependency   require_all/none svc:/network/physical (online)
dependency   require_all/none svc:/system/cryptosvc (online)
dependency   require_all/none svc:/system/utmp (online)
dependency   require_all/restart
             file://localhost/etc/ssh/sshd_config (online)
```

```
#
```

svcs -x

Informations sur les services en erreur (*broken services*).

svcadm

Cette commande permet d'agir sur les services (activation, désactivation, redémarrage…).

```
# svcadm disable cron
# svcs -l cron
fmri         svc:/system/cron:default
name         clock daemon (cron)
enabled      false
state        disabled
next_state   none
state_time   Thu Mar 28 16:37:37 2006
logfile      /var/svc/log/system-cron:default.log
restarter    svc:/system/svc/restarter:default
contract_id
dependency   require_all/none svc:/system/filesystem/local (online)
dependency   require_all/none svc:/milestone/name-services (online)
# svcadm enable cron
# svcs -l cron
fmri         svc:/system/cron:default
name         clock daemon (cron)
enabled      true
state        online
next_state   none
state_time   Thu Mar 28 16:38:02 2006
logfile      /var/svc/log/system-cron:default.log
restarter    svc:/system/svc/restarter:default
contract_id  101
dependency   require_all/none svc:/system/filesystem/local (online)
dependency   require_all/none svc:/milestone/name-services (online)
#
```

svccfg

Cette commande permet d'effectuer des choix de configuration concernant les services. Elle peut être utilisée de manière interactive.

```
# svccfg
svc:> help
General commands:        help set repository end
Manifest commands:       inventory validate import export archive
Profile commands:        apply extract
Entity commands:         list select unselect add delete
Snapshot commands:       listsnap selectsnap revert
Property group commands: listpg addpg delpg
Property commands:       listprop setprop delprop editprop
Property value commands: addpropvalue delpropvalue setenv unsetenv
svc:>
```

svcprop

Cette commande permet d'extraire les propriétés associées aux services.

```
# svcprop -p start/exec system/cron
/lib/svc/method/svc-cron
#
```

Dans cet exemple, nous obtenons le nom de la méthode de démarrage pour le service *cron*.

inetadm

Cette commande permet de gérer les services réseau pilotés par le ***superdémon inetd***.

inetd est un processus permanent qui est en écoute simultanément sur plusieurs ports TCP/IP. On le qualifie quelquefois de *superdémon* car son rôle est d'assurer le lancement ponctuel de certains services réseau qui ne sont démarrés qu'à la demande, lorsqu'ils sont contactés par les clients correspondants.

SMF par rapport aux anciennes procédures

Nous pouvons proposer un tableau comparatif entre d'anciennes tâches usuelles et leur équivalent dans le contexte SMF.

Activation et désactivation d'un service

Dans l'ancien protocole, pour activer un service, il faut créer un lien dont le nom commence par **S** dans le répertoire ***/etc/rcn.d*** (n = niveau concerné). Pour le désactiver, il faut supprimer ce lien.

En contexte SMF, on utilise la commande **svcadm** avec l'argument **enable** ou **disable**.

Dans les versions précédentes de Solaris, en ce qui concerne un service réseau géré par le superdémon **inetd**, il faut ajouter une ligne adéquate dans le fichier ***/etc/inetd.conf*** et envoyer à *inetd* le signal 1 (**HUP**).

En contexte SMF, on utilise la commande **svcadm** avec l'argument **enable**.

Démarrage ou arrêt ponctuel d'un service

Dans l'ancien protocole, les services possèdent un script générique dans le répertoire ***/etc/init.d***. Ces scripts sont invoqués avec les arguments **stop** et **start**.

En contexte SMF, on utilise la commande **svcadm** avec l'option **-t**.

Exemples

```
/etc/init.d/sshd stop
/etc/init.d/sshd start
```

trouvent leur équivalent en :

```
svcadm disable -t ssh
svcadm enable -t ssh
```

3. Groupes et Utilisateurs

3.1 Quelques rappels indispensables

La gestion maîtrisée des comptes utilisateurs et des groupes constitue une tâche essentielle d'administration. Elle influe, en effet, sur la cohérence et la sécurité du système tant au niveau local qu'au niveau d'une collaboration de plusieurs machines sur le réseau.

Tout système Unix comporte un compte privilégié baptisé historiquement '**root**" (son numéro interne est égal à zéro). Ce compte possède les pleins pouvoirs sur le système et doit correspondre à une personne compétente, rigoureuse et irréprochable d'un point de vue déontologique.

Certains autres comptes ne correspondent pas à des personnes physiques. Il s'agit de comptes prédéfinis qui sont associés à des logiciels ou à des services. Ils jouent alors le rôle de propriétaire pour certains processus ou fichiers assurant ainsi modularité et sécurité. La connexion interactive est souvent impossible sur ces comptes (mot de passe verrouillé ou autre moyen) et on les qualifie ainsi de "**pseudo-logins**".

De façon similaire, il existe un certain nombre de groupes prédéfinis jouant également un rôle important au niveau de la bonne protection des fichiers du système.

La plupart des versions ne différencient pas le groupe principal et les groupes secondaires au niveau des droits d'accès aux fichiers. Un utilisateur peut donc appartenir simultanément à plusieurs groupes.

Les commandes **groups** ou **id** affichent la liste de tous les groupes d'un utilisateur. Sur d'anciennes versions, un utilisateur ne pouvait appartenir qu'à un seul groupe à la fois. La commande **newgrp** permettait de changer dynamiquement de groupe. Elle est toujours opérationnelle pour changer ponctuellement de groupe principal même si son utilisation est très peu répandue.

Enfin, certaines versions proposent une distinction entre les groupes définis pour un utilisateur et ceux réellement actifs à un instant donné. La commande **setgroups** AIX permet, par exemple, de gérer cet aspect.

3.2 Fichiers de configuration

3.2.1 Le fichier /etc/passwd

Ce fichier doit être accessible en lecture pour tout le monde (beaucoup de commandes ont besoin de pouvoir le lire). Chaque ligne du fichier décrit un compte utilisateur. Elle est constituée de sept champs séparés par le caractère : (deux points).

```
nom:mop:uid:gid:commentaire:répertoire:shell
```

nom

Certaines versions limitent le nom de l'utilisateur à huit caractères. Il est conseillé d'utiliser uniquement des lettres minuscules et des chiffres pour le constituer.

mop

La quasi-totalité des versions utilisent aujourd'hui un fichier spécifique sécurisé (accessible en lecture uniquement au compte "root") pour stocker les informations concernant le mot de passe. Ce champ historique ne donne plus aujourd'hui d'informations pertinentes et contient un caractère variable selon les versions (x ou ! ou *).

uid

Il s'agit du numéro interne associé au nom de l'utilisateur (User IDentification). Par convention, les petits numéros sont réservés aux comptes prédéfinis du système et on attribuera aux comptes ordinaires des numéros supérieurs à 200, voire 500 ou 1000 selon les versions. Il est intéressant de planifier des intervalles de numéros associés aux différents groupes et il est important de toujours attribuer le même numéro à un utilisateur ayant accès à plusieurs systèmes du réseau.

gid

Il s'agit du numéro interne associé au groupe principal de l'utilisateur (Group IDentification). Ce numéro désigne une entrée dans le fichier "/etc/group" présenté juste après.

commentaire

Ce champ peut contenir un commentaire associé à l'utilisateur, par exemple son nom complet, son poste téléphonique etc... Ce champ peut contenir des espaces et il peut également être vide.

répertoire

Il s'agit du nom complet du répertoire de connexion. Ce répertoire constitue le point de départ de la sous-arborescence privée de l'utilisateur (variable d'environnement HOME). Les répertoires "/home" ou encore "/export/home" accueillent le plus souvent ces répertoires de connexion qui portent d'habitude assez naturellement le nom de l'utilisateur.

shell

Il s'agit du nom complet du programme automatiquement démarré à la connexion. La liste des "shells" valides est, en général, fourni par le fichier "**/etc/shells**". Le "Korn shell" (ksh) est le standard de fait la plupart du temps tandis que le "Bourne-Again shell" (bash) est proposé par défaut dans les distributions Linux. Le "Bourne shell" (sh) est implicitement choisi quand le champ n'est pas renseigné. Dans le cas où la connexion interactive en mode texte n'est pas souhaitée (comptes de messagerie, "pseudo-logins" système...), une bonne protection consiste à indiquer un programme qui se termine immédiatement ("/bin/false" ou "/bin/nologin" ou encore "/dev/null" ...).

Exemple

```
michel:x:1001:1000:Michel Dupont:/home/michel:/usr/bin/ksh
```

L'utilisateur "michel" porte le numéro 1001. Son groupe principal porte le numéro 1000 (à voir dans le fichier "/etc/group"). Son nom complet est "Michel Dupont". Il se connecte en "Korn shell" dans le répertoire "/home/michel".

3.2.2 Le fichier /etc/group

Ce fichier doit également être accessible en lecture pour tout le monde. Chaque ligne du fichier décrit un groupe. Elle est constituée de quatre champs séparés par le caractère : (deux points).

```
nom:mop:gid:liste des membres
```

La présence explicite d'un groupe n'est pas requise dans ce fichier. L'apparition d'un nouveau "gid" dans le quatrième champ d'une ligne du fichier "/etc/passwd" est suffisante. Cependant, il est nettement préférable de recenser ici tous les groupes. Cela est même obligatoire dans les versions AIX.

nom

Certaines versions limitent le nom de groupe à huit caractères. Il est conseillé d'utiliser uniquement des lettres minuscules et des chiffres pour le constituer. Le nom de groupe peut être le même que celui du compte utilisateur, notamment si cet utilisateur en est le seul membre.

mop

Ce champ, anciennement destiné à stocker un mot de passe de groupe, est obsolète aujourd'hui et contient un caractère variable selon les versions (x ou ! ou *).

gid

Il s'agit du numéro interne associé au groupe (Group IDentification). Ce numéro désignera le groupe principal d'un utilisateur s'il correspond au quatrième champ de sa ligne dans le fichier "/etc/passwd". Par convention, les petits numéros sont réservés aux groupes prédéfinis du système et on attribuera aux groupes ordinaires des numéros supérieurs à 200, voire 500 ou 1000 selon les versions.

liste des membres

Ce quatrième champ peut être vide. Dans le cas contraire, il contient une liste de comptes (on sépare les noms par des virgules) pour lesquels le groupe est un groupe supplémentaire. Même si cela ne constitue pas une erreur, il est inutile de mentionner ici un compte pour lequel le groupe est déjà le groupe principal.

Exemple

```
stage:!:1000:
ecole:!:2000:stage1,stage2,stage3
```

Le groupe "stage" porte le numéro 1000 et est le groupe principal des utilisateurs ayant la valeur 1000 comme quatrième champ du fichier "/etc/passwd". Le groupe "ecole" porte le numéro 2000 et est le groupe principal des utilisateurs ayant la valeur 2000 comme quatrième champ du fichier "/etc/passwd". De plus, il comprend les utilisateurs "stage1", "stage2" et "stage3" qui possèdent donc aussi les permissions du groupe sur un fichier de groupe propriétaire "ecole".

3.2.3 Les fichiers spécifiques sécurisés

En complément des deux fichiers précédents (présents sur toutes les versions et accessibles en lecture aux comptes ordinaires), chaque version propose un ou plusieurs fichiers spécifiques qui ne sont accessibles, même en lecture, qu'au seul compte administrateur. Le rôle de ces fichiers spécifiques est de stocker les mots de passe cryptés et certaines informations associées.

3.2.3.a Solaris, Linux : le fichier /etc/shadow

Chaque ligne du fichier correspond à un compte utilisateur du fichier "/etc/passwd". Elle est constituée de neuf champs séparés par le caractère : (deux points). Certains champs peuvent être vides. Cela indique alors que la fonctionnalité associée n'est pas mise en oeuvre.

```
nom:mop:lastchg:min:max:warning:inactive:expire:unused
```

nom

Nom de l'utilisateur.

mop

Ce champ stocke le cryptage du mot de passe. Une chaîne cryptée de treize caractères exactement correspond à un mot de passe Unix traditionnel. Le mot de passe en clair est lui-même limité à huit caractères significatifs. Des mots de passe plus longs (mots de passe MD5) sont aujourd'hui disponibles sur certaines versions dont Linux. La chaîne cryptée résultat dépasse alors les treize caractères. Un mot de passe impossible ("pseudo-logins") ou verrouillé correspondra à une chaîne cryptée, soit inférieure à treize caractères, soit commençant par un caractère conventionnel comme le ! (point d'exclamation). Enfin, un champ vide signifie que le compte ne possède pas de mot de passe. Ceci est, bien sur, fortement déconseillé et il est toujours possible de rendre, par ailleurs, le mot de passe obligatoire.

lastchg

Ce champ indique la date du dernier changement de mot de passe. Cette date est stockée sous la forme d'un nombre de jours écoulés depuis le 1er janvier 1970.

min

Il s'agit du nombre de jours minimum exigé entre deux changements de mot de passe.

max

Il s'agit du nombre de jours maximum de validité du mot de passe.

warning

Il s'agit d'un nombre de jours. Ceci permet de déclencher un avertissement vers l'utilisateur avant l'expiration de son mot de passe.

inactive

Il s'agit du nombre de jours de non-utilisation du compte provoquant une désactivation de celui-ci.

expire

Ce champ indique la date d'expiration du compte. Cette date est stockée sous la forme d'un nombre de jours écoulés depuis le 1er janvier 1970.

unused

Ce champ n'est pas utilisé et réservé à un usage futur.

Exemple

```
michel:JipbAw3I1umKU:12122::14:28:60:12417:
```

L'utilisateur "michel" a un mot de passe (chaîne cryptée de 13 caractères). Le mot de passe doit être changé toutes les quatre semaines et il doit s'écouler deux semaines entre deux changements. Le compte devient inactif au bout de 60 jours de non-utilisation et possède également une date absolue d'expiration (12417 jours depuis le 1/1/1970, soit le 31/12/2003).

3.2.3.b AIX : de nombreux fichiers supplémentaires

Les versions AIX comportent plus de fichiers spécifiques sécurisés que les autres versions Unix. La plupart de ces fichiers sont organisés en strophes ("stanzas"). Nous résumons ici les plus importants.

/etc/security/passwd

Ce fichier stocke les mots de passe. Chaque strophe comporte trois attributs :

- **password** Codage du mot de passe
- **lastupdate** Date de dernier changement du mot de passe (nombre de secondes écoulées depuis le 1er janvier 1970)
- **flags**

ADMIN	Seul le compte "root" peut modifier le mot de passe
ADMCHG	Changement de mot de passe obligatoire à la prochaine connexion
NOCHECK	Les contraintes sur les mots de passe ne sont pas prises en compte.

Exemple de strophe

```
michel:
        password =
        lastupdate = 916391075
        flags = ADMCHG
```

L'utilisateur "michel" n'a pas encore de mot de passe mais il devra en choisir un lors de sa prochaine connexion.

/etc/security/user

Ce fichier permet d'indiquer de nombreux attributs complémentaires concernant les comptes avec notamment des propriétés liées à la gestion des mots de passe. Le fichier comporte une strophe "default" regroupant des valeurs génériques pour tous les comptes et une strophe par utilisateur dans laquelle on pourra mentionner les valeurs individuelles modifiées.

Quelques attributs (liste non exhaustive)

login	L'utilisateur peut-il se connecter en local ?
rlogin	Le compte est-il accessible par des connexions distantes ?
ttys	Liste de terminaux autorisés
su	Le compte est-il accessible via la commande "su" ?
expires	Date d'expiration du compte
umask	Valeur de "umask" (droits par défaut des fichiers en création)
logintimes	Heures autorisées pour la connexion
loginretries	Nombre maximum de connexions infructueuses consécutives
minage	Durée minimum de validité du mot de passe (en semaines)
maxage	Durée maximum de validité du mot de passe (en semaines)
minalpha	Nombre minimum de caractères alphabétiques dans un mot de passe
minother	Nombre minimum de caractères non-alphabétiques dans un mot de passe
minlen	Nombre minimum de caractères dans un mot de passe
mindiff	Nombre minimum de caractères non présents dans l'ancien mot de passe
maxrepeats	Nombre maximum d'apparitions d'un caractère dans un mot de passe
histexpire	Nombre de semaines pendant lequel on ne peut réutiliser un mot de passe
histsize	Nombre de mots de passe précédents non réutilisables
dictionlist	Liste de fichiers contenant des mots de passe interdits

Exemple

Certains attributs sont des booléens (valeur "true" ou "false"). Pour les autres, par convention et selon le cas, les valeurs "0" ou "-1" indiquent une valeur infinie ou signifient que la fonctionnalité n'est pas activée.

```
default:
        login = true
        su = true
        rlogin = true
        ttys = ALL
        umask = 022
        expires = 0
```

```
        loginretries = 0
        histexpire = 0
        histsize = 0
        minage = 0
        maxage = 0
        minalpha = 0
        minother = 0
        minlen = 0
        mindiff = 0
        maxrepeats = 8
        dictionlist =
        ...................
root:
        ...................
        ttys = /dev/console
        rlogin = false
```

On fait ici le choix que l'utilisateur "root" ne puisse se connecter directement qu'à la console. Il reste cependant accessible via la commande "su" (l'attribut correspondant a la valeur "true" dans la strophe "default").

/etc/security/limits

Ce fichier permet d'indiquer des limitations de ressources. Le fichier comporte une strophe "default" regroupant des valeurs génériques et une éventuelle strophe par utilisateur dans laquelle on pourra mentionner les valeurs individuelles modifiées.

Une limitation de type "soft" peut être modifiée par un utilisateur (via la commande "ulimit") mais ne peut, en aucun cas, atteindre la limitation de type "hard". Une limitation de type "hard" ne peut être modifiée que par le compte "root".

Quelques attributs (liste non exhaustive)

fsize	Taille maximum ("soft") d'un fichier
core	Taille maximum ("soft") d'un fichier "core"
cpu	Temps CPU maximum ("soft") pour un processus
nofiles	Nombre maximum ("soft") de fichiers ouverts simultanément
fsize_hard	Taille maximum ("hard") d'un fichier
core_hard	Taille maximum ("hard") d'un fichier "core"
cpu_hard	Temps CPU maximum ("hard") pour un processus
nofiles_hard	Nombre maximum ("hard") de fichiers ouverts simultanément

Exemple

Le fichier est abondamment auto-commenté. Les tailles sont données en nombre de blocs de 512 octets. Les durées sont exprimées en secondes. Une valeur "-1" signifie "illimité".

```
default:
        fsize   = 2097151
        core    = 2097151
        cpu     = -1
        nofiles = 2000
        ..................
michel:
        cpu_hard = 3600
        fsize_hard = 20480
```

On fait ici le choix que l'utilisateur "michel" ne puisse pas lancer un processus qui consommerait plus d'une heure de temps CPU. De plus, il ne pourra pas créer de fichier dont la taille atteindrait 20480 blocs de 512 octets (soit 10 Mo octets).

Autres fichiers

/etc/security/group	Attributs complémentaires pour les groupes
/etc/security/login.cfg	Attributs liés aux méthodes de connexion
/etc/security/environ	Attributs d'environnement
/usr/lib/security/mkuser.default	Attributs par défaut en création

3.2.3.c HP-UX : Trusted Computing Base (TCB)

Certains systèmes, dont HP-UX, peuvent être configurés pour répondre à la norme de sécurité C2 en implémentant un ensemble de fichiers qui constituent une partie de ce que l'on désigne sous le nom de "Trusted Computing Base". Il s'agit alors d'un ensemble de fichiers situés dans la sous-arborescence "/tcb/files".

Sous HP-UX, il y a un répertoire "auth" dans lequel chaque compte utilisateur correspond à un fichier situé dans un sous-répertoire correspondant à la première lettre de son nom.

Ainsi l'utilisateur "michel" sera décrit dans le fichier "/tcb/files/auth/m/michel". Le fichier comporte une strophe dotée d'attributs de type booléen, numérique ou alphanumérique. Des valeurs par défaut pour certains attributs sont stockées dans le fichier "/tcb/files/auth/system/default".

Exemple

```
michel:u_name=michel:u_id#1001:\
       :u_pwd=JipbAw3I1umKU:u_lock@:
       ..............................
```

"u_name" et "u_pwd" sont des attributs alphanumériques (nom et mot de passe crypté). "u_id" (numéro d'utilisateur) est un attribut numérique. "u_lock" est un attribut booléen (le @ final indique une valeur fausse : le compte n'est pas verrouillé).

3.3 Gestion des groupes

La définition des groupes est prise en charge par le fichier standard "/etc/group" déjà évoqué. La création d'un groupe doit précéder celle des comptes qui en seront membres.

Solaris, Linux, HP-UX : groupadd, groupmod, groupdel

Plutôt que d'éditer directement le fichier "/etc/group", il est préférable d'utiliser les commandes officielles de gestion qui sont d'ailleurs d'une grande simplicité.

Exemples

```
# groupadd  -g 2000  stage
# tail  -1  /etc/group
stage::2000:
#
```

Création d'un groupe "stage" de numéro 2000

```
# groupmod  -n unixstage  stage
# tail  -1  /etc/group
unixstage::2000:
#
```

Le groupe "stage" est renommé en "unixstage".

```
# groupdel  unixstage
#
```

Le groupe "unixstage" est supprimé.

AIX : mkgroup, chgroup, rmgroup

Les versions AIX utilisent un jeu de commandes différent ainsi qu'un fichier complémentaire "/etc/security/group" qui permet une éventuelle délégation de pouvoirs en désignant des administrateurs de groupes.

Exemple

```
# mkgroup  id='2000'  stage
# tail  -1  /etc/group
stage:!:2000:
# tail  -3  /etc/security/group
stage:
     adms = root
     admin = false
#
```

3.4 Gestion des comptes utilisateurs

La création d'un compte utilisateur met en jeu divers aspects qui permettent de dégager une démarche "multi-versions" qu'il s'agira d'adapter à chaque mise en oeuvre effective.

Planification

Il faut mettre en place une saine organisation des groupes par rapport aux besoins de partage de fichiers entre les utilisateurs. D'autre part, un choix explicite des numéros internes des comptes et des groupes s'impose pour garantir une cohérence dans l'utilisation de certains services réseau. Une création centralisée des comptes sur des serveurs NIS ou LDAP (services d'annuaire qui dépassent le cadre de cet ouvrage) sera sans doute souhaitable sur un site où les utilisateurs doivent accéder à plusieurs systèmes de façon transparente et sécurisée.

Maîtrise des fichiers de configuration

L'administrateur doit prendre le temps d'étudier, dans la documentation du système, le détail des divers fichiers de configuration spécifiques à la version concernée. La maîtrise complète de ces fichiers lui permet de connaître l'étendue des possibilités offertes par la version, notamment en ce qui concerne une gestion sécurisée des mots de passe.

Identification des commandes et utilitaires

Bien que d'anciennes commandes ou habitudes le permettent encore souvent, les fichiers de configuration ne devraient pas être édités directement. Chaque version propose aujourd'hui des commandes officielles de gestion ainsi que des utilitaires interactifs dans le cadre des menus d'administration. Comme dans le point précédent, l'administrateur prendra le temps d'étudier la documentation de référence pour être capable d'effectuer les opérations via la ligne de commande traditionnelle. Si ce travail d'étude a été effectué, les menus interactifs n'en seront que plus clairs et l'administrateur pourra constater que certains paramétrages avancés ne sont pas toujours disponibles dans le cadre de ces menus.

Valeurs par défaut

En liaison avec les points précédents, il convient d'identifier où sont indiquées ou bien encore comment sont déterminées les valeurs par défaut de certains attributs. Ceci permettra d'alléger la syntaxe des commandes et, éventuellement, de modifier ces valeurs génériques pour les adapter à ses besoins particuliers.

Fichiers prototypes

Lors de la création d'un compte, il est souhaitable de copier automatiquement un certain nombre de fichiers d'initialisation dans le répertoire du nouvel utilisateur. Chaque version Unix prévoit cette possibilité. Il faudra donc identifier l'endroit par défaut où se trouvent ces fichiers que l'on appelle "prototypes" et il sera sans doute nécessaire d'améliorer le contenu de ces fichiers de base, voire d'en rajouter certains.

Gestion du mot de passe

La politique de gestion des mots de passe constitue, sous Unix comme ailleurs, un point clé de la sécurité des systèmes. La commande "passwd", quand elle est utilisée par le compte "root", est dotée de nombreuses options permettant de choisir sa stratégie lors de la création d'un compte : pas de mot de passe, mot de passe imposé ou mot de passe choisi par l'utilisateur. Si la version le permet, divers paramétrages auront pu être activés dans les phases précédentes (mot de passe obligatoire, durées de validité, contraintes syntaxiques, longueur minimale, dictionnaire de mots de passe interdits etc...).

3.4.1. Création de compte (Solaris, Linux, HP-UX)

Sur les versions ci-dessus, la création complète d'un compte nécessite d'enchaîner la commande **useradd** (définition des attributs du compte et création du répertoire de connexion) et la commande **passwd** (positionnement du mot de passe). La commande "useradd" permet également de gérer certaines valeurs par défaut.

Exemple Solaris

```
# useradd  -u 2001  -g stage  -m  -s /usr/bin/ksh  michel
#
```

Création du compte "michel" de numéro 2001. Son groupe principal est le groupe "stage", son shell de connexion est le Korn-shell. Grâce à l'option "-m", il y a également création du répertoire de connexion (nom par défaut : "/home/michel") avec recopie des fichiers prototypes situés dans le répertoire par défaut "/etc/skel".

```
# tail -1 /etc/passwd
michel:x:2001:2000::/home/michel:/usr/bin/ksh
# tail -1 /etc/shadow
michel:*LK*:::::::
#
```

Le compte est bien défini dans les fichiers de configuration mais le champ "mot de passe" du fichier "/etc/shadow" montre que le compte est pour l'instant verrouillé. La commande "passwd" permet de terminer la mise en oeuvre, soit en supprimant le mot de passe (option "-d"), soit en le positionnant d'autorité, soit en le laissant choisir par l'utilisateur lors de sa première connexion (option "-f").

```
# passwd -d michel
ou bien
# passwd -f michel
ou bien
# passwd michel
New password:
Re-enter new password:
passwd (SYSTEM): passwd successfully changed for michel
#
```

3.4.2. Création de compte (AIX)

Dans les versions AIX, la création complète d'un compte nécessite d'enchaîner la commande **mkuser** et la commande **passwd**.

Exemple

```
# mkuser  id='2001'  pgrp='stage'  michel
#
```

Création du compte "michel" de numéro 2001 et de groupe principal "stage". Le shell de connexion par défaut est le Korn-shell (fichier "/usr/lib/security/mkuser.default"). Il y a également création du répertoire de connexion (script "/usr/lib/security/mkuser.sys" appelé par la commande "mkuser"). Le fichier prototype "/etc/security/.profile" est, de plus, copié dans le répertoire de connexion.

```
# tail  -1  /etc/passwd
michel:*:2001:2000::/home/michel:/usr/bin/ksh
#
```

Le champ numéro 2 comporte une "". Cela signifie qu'il n'y a pas encore de strophe dans le fichier des mots de passe "/etc/security/passwd". La œmmande "passwd" permet de terminer la mise en oeuvre, par exemple, en laissant choisir le mot de passe par l'utilisateur lors de sa première connexion.*

```
# passwd  michel
Changing password for "michel"
michel's New password:                    Saisie vide
Re-enter michel's new password:           Saisie vide
#
```

Il y a création d'une strophe dans le fichier "/etc/security/passwd" avec l'indicateur "ADMCHG" qui signifie que le mot de passe sera choisi à la première connexion. De plus, le caractère "!" remplace le caractère "" dans le fichier "/etc/passwd" pour indiquer que les informations sur le mot de passe sont maintenant valides.*

```
# tail  -4  /etc/security/passwd
michel:
    password =
    lastupdate = 949166174
    flags = ADMCHG
# tail  -1  /etc/passwd
michel:!:2001:2000::/home/michel:/usr/bin/ksh
#
```

3.4.3. Désactivation et suppression de comptes

Un compte peut être facilement désactivé (il n'est pas supprimé mais la connexion devient impossible) :

Solaris, Linux	**passwd -l** *compte*
AIX	**chuser account_locked=yes** *compte*
HP-UX	paramètre "**u_lock**" dans les fichiers "TCB"

Un compte peut être également supprimé :

Solaris, Linux, HP-UX	**userdel -r** *compte*

L'option "-r" permet la suppression du répertoire de connexion.

AIX	**rmuser -p** *compte*

L'option "-p" permet la suppression complète dans tous les fichiers de configuration. La suppression du répertoire de connexion n'est pas prise en charge par la commande.

Si un compte doit être complètement supprimé, la commande dédiée doit s'accompagner d'un certain nombre d'opérations complémentaires :

Archivage éventuel et suppression des fichiers de l'utilisateur

La commande de suppression de l'utilisateur dispose souvent d'une option permettant de supprimer également le répertoire de connexion. Une utilisation préalable de la commande "find" permettra de vérifier que l'utilisateur n'avait pas réussi à créer des fichiers à d'autres endroits du système et inversement que son arborescence ne comportait pas de fichiers d'autres utilisateurs.

Suppression d'éventuels courriers et alias de courriers

Suppression de requêtes d'impression en attente

Annulation de requêtes "cron" ou "at" (voir chapitre 8)

Modification de mots de passe complémentaires connus de l'utilisateur

Éventuelles actions spécifiques au site

4. Disques et Systèmes de fichiers

4.1 Quelques rappels indispensables

Le noyau Unix n'intègre pas de sémantique sur les noms des fichiers et la notion d'extension n'existe pas.

Il n'y a pas de caractères interdits pour constituer un nom (sauf le / qui sert de séparateur dans les chemins). On se contente tout naturellement des lettres, des chiffres et du caractère _ (tiret bas) pour constituer des "noms composés". Les minuscules et les majuscules sont différentes. Il est habituel de réserver l'utilisation des majuscules à des fichiers particuliers que l'on veut placer en tête de liste dans un affichage ("README", "LISEZMOI" ...).

Par tradition, certains fichiers d'initialisation ou de paramétrage ont des noms qui commencent par un point (".profile", ".bash_profile", ".exrc" ...).

Sur de très anciennes versions, les noms étaient limités à 14 caractères. On dispose aujourd'hui des noms longs (pouvant aller jusqu'à 255 caractères).

Pour l'utilisateur, le système de fichiers Unix apparaît comme une **arborescence** de répertoires. On y distingue quatre types principaux de fichiers :

Fichier ordinaire

Le noyau Unix ne distingue pas les types de contenu (texte, binaire). Il faut recourir à la commande "file" qui lit le début du fichier pour identifier la nature de ce contenu.

Répertoire

De par le concept même d'arborescence, tout fichier, quel que soit son type, appartient à un répertoire.

Fichier spécial

Il s'agit du nom d'un périphérique. Le fait de désigner un fichier spécial dans une commande appropriée active implicitement le programme de gestion ("driver") de ce périphérique.

Lien symbolique

Un lien symbolique contient le nom d'un autre fichier et joue le rôle de pointeur vers cet autre emplacement. Les liens symboliques sont apparus notamment pour gérer les évolutions de l'arborescence Unix en préservant la compatibilité avec les arborescences antérieures.

Les répertoires contiennent des couples "nom de fichier - numéro d'inode". Le numéro d'inode correspond à une entrée dans une table des **inodes** du "**filesystem**" concerné (la notion de "filesystem" est présentée plus loin dans ce chapitre).

Un "inode" contient notamment les informations suivantes :
- Type du fichier
- Taille (en octets)
- Propriétaire et groupe
- Droits d'accès
- Adresses des blocs disque alloués au fichier
- Trois dates système

(dernière modification, dernière consultation, dernière modification de l'inode)

4.1.1 Les principaux répertoires

Les liens symboliques permettent de rendre transparents les changements de noms de fichiers par rapport aux versions antérieures du système.

/bin Commandes publiques

"/bin" est le plus souvent un lien symbolique vers "/usr/bin".

/dev Fichiers spéciaux

Les noms de périphériques sont propres à chaque version ou matériel. Ce répertoire est de plus en plus réorganisé en sous-répertoires.

/etc Répertoire administratif

Il contient des fichiers de configuration (non exécutables). Les commandes sont le plus souvent des liens symboliques vers "/sbin" et "/usr/sbin".

/home Répertoire par défaut pour y créer les répertoires des utilisateurs

/lib Bibliothèques, programmes des compilateurs, commandes diverses...

"/lib" est le plus souvent un lien symbolique vers "/usr/lib".

/sbin Commandes administratives utilisées au démarrage du système

Les fichiers situés sous "/usr" ne sont pas toujours accessibles dans les toutes premières étapes du démarrage du système.

/unix Noyau (exécutable chargé en mémoire au démarrage)

Ce fichier est souvent un lien symbolique vers un autre emplacement physique. Son nom varie avec les versions.

/tmp Répertoire public de travail pour fichiers temporaires

/usr Ce répertoire contient beaucoup d'autres répertoires correspondant aux diverses composantes du système en terme de commandes et d'exécutables.

/var Ce répertoire contient les répertoires associés aux aspects variables d'un système (files d'attente, fichiers de statistiques et d'historique...).

Les arborescences actuelles peuvent se résumer par une classification en quatre catégories correspondant à des coutumes de fait d'utilisation :

Fichiers non partagés
Ils se trouveront dans le répertoire "/etc" (fichiers de configuration du système local).

Fichiers partageables entre machines de même "hardware"
Il s'agit principalement des exécutables et des bibliothèques. Ils se trouveront sous le répertoire "/usr" dont le contenu est, a priori, figé au quotidien. *Le contenu de ce répertoire ne devrait évoluer que lors de l'installation de nouveaux paquets logiciels.*

Fichiers partageables par toute machine
Il s'agit de fichiers "texte" ou de format universel. Ils se trouveront sous "/usr/share".

Fichiers de travail du système (fichiers d'historique, files d'attente...)
Ils se trouveront sous le répertoire "/var".

4.1.2 Droits d'accès aux fichiers

4.1.2.a Permissions de base

Un fichier Unix, quel que soit son type, possède neuf permissions de base. En effet, il y a trois catégories d'utilisateurs (propriétaire, groupe, autres) et trois permissions associées à chaque catégorie (lecture (r), écriture (w), exécution (x)).

Ces permissions sont visualisées via l'option "-l" de la commande "ls".

```
$ ls  -l
-rwxr-x---   1   michel   stage   3053   Dec 13 14:43   fic
$
```

Le fichier ordinaire "fic" appartient au compte "michel" et au groupe "stage".

Son propriétaire possède les droits de lecture, d'écriture et d'exécution.

Les membres du groupe "stage" possèdent les droits de lecture et d'exécution.

Les comptes n'appartenant pas au groupe "stage" ne possèdent aucun droit.

La sémantique des permissions de base dépend du type de fichier :

Fichier spécial

Lecture et écriture ont leur signification naturelle et le droit d'exécution n'est pas utilisé.

Fichier ordinaire

r On peut consulter ou copier le fichier.

w On peut modifier son contenu.

x Le fichier est une commande, le "shell" essaye de l'exécuter. Si ce fichier n'est pas un script, le droit de lecture n'est pas nécessaire à son exécution.

Répertoire

r On peut lister les noms des fichiers du répertoire avec la commande "ls".

w On peut créer ou supprimer des fichiers dans ce répertoire. Il est à noter que l'on peut détruire un fichier sans posséder le droit d'écriture sur ce fichier individuel.

x On peut se positionner dans le répertoire ou le mentionner dans un chemin. L'absence de cette permission rend le répertoire et toute sa sous-arborescence complètement inaccessible.

4.1.2.b Permissions supplémentaires

Quelques permissions supplémentaires peuvent être associées à certains types de fichiers.

En interne, 12 bits (et non pas seulement 9) de l'inode sont utilisés pour représenter les permissions. Cependant, la commande "ls" visualise les droits sur 9 positions grâce à des conventions de représentation dans le cas où des droits supplémentaires sont positionnés.

Fichiers ordinaires exécutables (commandes)

bit SUID

Un processus Unix possède deux propriétaires : l'**effectif** qui déterminera les droits et le **réel** qui sera considéré dans divers aspects de statistiques, de comptabilité ou de communication inter-processus. Par défaut, le réel et l'effectif correspondent naturellement au compte ayant lancé le programme. Mais, si le programme possède la permission "SUID", le propriétaire du programme devient le propriétaire effectif du processus. Quand le propriétaire du programme est le compte "root" (cas le plus fréquent), ceci permet d'envisager des commandes publiques pouvant accomplir des tâches privilégiées.

La présence de cette permission est visualisée par la lettre "s" à la place du "x" dans la partie propriétaire.

bit SGID

De la même façon, un processus possède deux groupes propriétaires (réel et effectif). Par défaut, le réel et l'effectif correspondent au groupe principal du compte ayant lancé le programme. Mais, si le programme possède la permission "SGID", le groupe propriétaire du programme devient le groupe propriétaire effectif du processus.

La présence de cette permission est visualisée par la lettre "s" à la place du "x" dans la partie groupe.

Exemple

```
# ls -l /usr/bin/passwd
-r-sr-sr-x   3   root   sys   73748   jui 27 2001   /usr/bin/passwd
#
```

On constate ici que, grâce au bit "SUID", tout utilisateur qui fait appel à la commande "passwd" génère un processus de propriétaire effectif "root". Ceci lui permet d'avoir les droits nécessaires à la réussite cette commande. On remarque que le bit "SGID" est également positionné, ce qui modifie aussi le groupe propriétaire effectif du processus.

Sticky Bit

Cette permission indique la volonté de conserver le processus en mémoire ou dans l'espace de pagination pour un chargement, a priori, plus rapide. Cette fonctionnalité historique des premières années d'Unix peut être considérée comme largement obsolète aujourd'hui.

Répertoires

bit SGID ou "Inheritance flag"

Tous les fichiers créés dans le répertoire prennent comme groupe propriétaire celui du répertoire. Cette permission est héritée par les sous-répertoires. Cela permet d'obtenir une arborescence où tous les fichiers sont placés dans le même groupe quels que soient les propriétaires concernés.

La présence de cette permission est visualisée par la lettre "s" à la place du "x" dans la partie groupe.

Sticky Bit

Cette permission complète le droit d'écriture. Elle indique que la suppression d'un fichier du répertoire sera réservée au seul propriétaire de ce fichier.

La présence de cette permission est visualisée par la lettre "t" à la place du "x" dans la partie "autres".

Exemple

```
# ls -ld /tmp
drwxrwxrwt   6   root   sys   272   mar 13 09:56   /tmp
#
```

Le répertoire "/tmp" est accessible en écriture à tous les utilisateurs qui peuvent donc y créer des fichiers. Cette seule écriture leur permettrait aussi de supprimer n'importe quel fichier dans le répertoire. La permission "sticky bit" supplémentaire fait que un utilisateur ne peut pas supprimer des fichiers qui ne lui appartiennent pas.

4.1.2.c ACLs (Access Control Lists)

La plupart des versions proposent aujourd'hui des permissions étendues baptisées "ACLs". L'objectif est de pouvoir associer à un fichier des droits particuliers pour un utilisateur ou pour un groupe en complément ou remplacement des permissions de base.

L'implémentation actuelle de ces "ACLs" n'est pas normalisée et le jeu de commandes et de fonctionnalités varie selon les versions :

AIX	aclget, aclput, acledit
HP-UX	lsacl, chacl
Solaris, Linux	getfacl, setfacl

Exemple AIX

```
$ ls -l toto
-rw-rw-rw-   1   root   system   29 Dec 13 16:12   toto
$ aclget  toto
attributes:
base permissions
    owner(root):  rw-
    group(system):  rw-
    others:  rw-
extended permissions
    enabled
    specify  r--     u:michel
$
$ id
uid=1001(michel) gid=1000(stage)
$ cp  /etc/profile  toto
cp: toto: Permission denied
$
```

Le fichier "toto" est accessible en écriture pour tout le monde d'après ses permissions de base. Cependant, l'utilisateur "michel" ne parvient pas à modifier le fichier car les "ACLs" associés à ce fichier ne lui permettent que la lecture.

4.1.2.d Commandes de gestion des permissions

La commande umask

Au sein de l'arborescence Unix, l'utilisateur est responsable de la protection de ses propres fichiers. Pour faciliter cette gestion, il dispose de la commande "umask" qui lui permet d'indiquer quels seront les droits de départ associés aux nouveaux fichiers ordinaires et aux nouveaux répertoires.

Les neuf permissions de base doivent être exprimées par une valeur octale de trois chiffres avec les associations suivantes :

r (lecture)	vaut	4
w (écriture)	vaut	2
x (exécution)	vaut	1

ce qui donne, par exemple : 750 pour la combinaison `rwx r-x ---`

644 pour la combinaison `rw- r-- r--`

Sans aucun appel à la commande, les permissions associées aux nouveaux fichiers seraient :

`rw- rw- rw-`	pour les fichiers ordinaires (666 en notation octale)
`rwx rwx rwx`	pour les répertoires (777 en notation octale)

Ces droits sont trop permissifs et ils doivent donc être modifiés via un appel à la commande "umask" qui supprime, dans les masques de droits par défaut, les bits spécifiés en argument. Une valeur générale est positionnée par l'administrateur (par exemple dans le fichier "/etc/profile") mais l'utilisateur peut faire un choix personnel en appelant la commande dans ses fichiers de paramétrage du "shell" (par exemple, son fichier ".profile").

Exemples

umask 022	Fichiers ordinaires en	rw- r-- r--	(644)
	Répertoires en	rwx r-x r-x	(755)
umask 002	Fichiers ordinaires en	rw- rw- r--	(664)
	Répertoires en	rwx rwx r-x	(775)
umask 027	Fichiers ordinaires en	rw- r-- ---	(640)
	Répertoires en	rwx r-x ---	(750)
umask 007	Fichiers ordinaires en	rw- rw- ---	(660)
	Répertoires en	rwx rwx ---	(770)

La commande chmod

La commande "chmod" permet de gérer les droits des fichiers dont on est propriétaire.

chmod [-R] mode fichier...
chmod [-R] symbolique fichier...

```
-R              Traitement récursif sur les répertoires
mode            Nombre octal sur 3 ou 4 positions
symbolique      Combinaison: qui opération accès
           [ugoa] +|-|= [rwxstugo]
qui        u    propriétaire
           g    groupe
           o    autres
           a    tous
opération  +    ajout
           -    suppression
           =    affectation
accès      r    lecture
           w    écriture
           x    exécution
           s    user ou group set-ID
           t    sticky bit
           u    permissions de l'utilisateur
           g    permissions du groupe
           o    permissions des autres
fichier...      Fichier(s) concerné(s)
```

Exemple

```
$ ls  -ld  .
drwxr-xr-x   6   michel   stage   512   jan 23   12:42   .
$ chmod  g+w,o+t  .
$ ls  -ld  .
drwxrwxr-t   6   michel   stage   512   jan 23   12:42   .
$
```

Dans le répertoire courant, les membres du groupe "stage" auront la possibilité de créer des fichiers (permission "w") mais ils ne pourront supprimer que ceux dont ils sont propriétaires ("sticky bit" sur répertoire).

4.2 Organisation de l'espace disque

Même si, pour l'utilisateur, le système de fichiers Unix apparaît comme une seule arborescence globale, celle-ci est, en fait, constituée très fréquemment de plusieurs sous-arborescences rassemblées, lors du démarrage du système, par une opération dite de montage. Ces différentes sous-arborescences sont baptisées "**filesystems**" (nous garderons, dans la suite, ce terme anglo-saxon qui est bien ancré dans le vocabulaire des "unixiens" et plus précis qu'une traduction du style "système de fichiers". Nous déciderons même de ne plus l'encadrer de guillemets.).

Un filesystem correspond à une structure physique précise qui est installée sur une partie de l'espace disque. Plusieurs types de filesystems sont disponibles selon les versions. La suite de ce chapitre détaille complètement ces notions fondamentales.

Dans un premier temps, nous allons évoquer les différentes possibilités d'organisation de l'espace disque permettant la création des filesystems.

4.2.1. Organisation classique en partitions

Lors de son intégration (dès l'installation du système lui-même ou lors d'un ajout ultérieur), un disque est divisé en partitions baptisées aussi "sections" ou encore "slices".

Une partition est une région contiguë d'un seul disque. Sa taille n'est pas modifiable par la suite. Un découpage judicieux demande, de la part de l'installateur, planification et expérience afin d'effectuer des choix cohérents au niveau des tailles des différents filesystems qui seront créés sur ces partitions.

En complément de celles destinées à accueillir des filesystems, il convient de prévoir une partition destinée à la gestion de la mémoire virtuelle. Cet espace se nomme traditionnellement "**zone de swap**" ou encore "**espace de pagination**". Il s'agit d'un espace disque particulier directement géré par le noyau.

4.2.1.a Mode bloc et mode "raw bloc"

Les entrées/sorties disques peuvent s'effectuer de deux façons : le mode bloc et le mode "raw" (ou caractère). Le mode bloc correspond à l'utilisation normale des disques par un service de tampons ("buffers") gérés par le noyau qui pratique ainsi des écritures différées et des lectures anticipées.

Il est néanmoins possible d'effectuer des entrées/sorties directes sans utiliser les services classiques du noyau. Pour ce faire, les partitions possèdent deux noms différents en tant que fichier spécial.

Exemple de notations HP-UX

```
# cd  /dev
# ls  -l  *dsk
dsk:
brw-r-----   1   bin   sys   31 0x004000   Nov 15 14:38   c0t4d0
brw-r-----   1   bin   sys   31 0x005000   Nov 15 14:39   c0t5d0
.........................
rdsk:
crw-r-----   1   bin   sys   188 0x004000   Nov 15 14:38   c0t4d0
crw-r-----   1   bin   sys   188 0x005000   Nov 15 14:39   c0t5d0
.........................
#
```

L'entrée en mode bloc correspond aux commandes usuelles et à l'utilisation en tant que filesystem. L'entrée en mode "raw bloc" (type "c" dans l'affichage de la commande "ls -l") est disponible pour une utilisation en tant que partition dédiée à un logiciel. Dans ce cas de figure, elle ne contient pas de filesystem et sa gestion est faite sous la responsabilité de ce logiciel (par exemple, un S.G.B.D. qui gère directement l'espace disque correspondant à une base de données). On parle alors communément, dans le jargon des administrateurs, d'une "raw partition". Il faut noter enfin que, même si la partition contient un filesystem, certaines commandes administratives, de par leur fonctionnement interne, utiliseront aussi l'entrée en mode "raw bloc".

*Les versions **Linux** ne conservent pas cette distinction historique entre les deux types de fichiers spéciaux pour une même partition. L'entrée en mode bloc est la seule disponible. La commande "**raw**" permet cependant d'effectuer une association avec une entrée caractère (fichier spécial sous "/dev/raw") pour permettre l'utilisation en tant que partition dédiée.*

4.2.1.b Visualisation des partitions (Solaris, Linux)

Les versions Unix concernées possèdent une commande spécifique pour visualiser ou modifier l'organisation d'un disque en partitions

Solaris

La commande interactive **format** (issue des anciennes versions SunOS qui ont précédé la famille Solaris) est un utilitaire complet de maintenance des disques et comprend évidemment une rubrique dédiée aux partitions.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0d0 <DEFAULT cyl 19845 alt 2 hd 16 sec 63>
          /pci@0,0/pci-ide@1f,1/ide@0/cmdk@0,0
Specify disk (enter its number): 0
selecting c0d0
................................
FORMAT MENU:
        disk       - select a disk
        type       - select (define) a disk type
        partition  - select (define) a partition table
................................
        quit
format> partition
PARTITION MENU:
        0      - change `0' partition
        1      - change `1' partition
................................
        print  - display the current table
................................
partition> print
Current partition table (original):
Total disk cylinders available: 19845 + 2 (reserved cylinders)
Part   Tag   Flag   Cylinders          Size            Blocks
  0   root     wm    1044 - 5107      1.95GB    (4064/0/0)   4096512
  1   swap     wu       3 - 1043    512.37MB    (1041/0/0)   1049328
  2   backup   wm       0 - 19845     9.54GB    (19846/0/0) 20004768
```

```
................................
  5   home     wm   11204 - 19330     3.91GB    (8127/0/0)   8192016
................................
partition> quit
................................
```

Tout comme les partitions individuelles, le disque complet est désigné via un fichier spécial avec une convention propre à la version. ***Sous Solaris, le disque complet correspond à une partition virtuelle qui porte le numéro deux****. Ainsi, dans l'extrait d'affichage ci-dessus, on remarque que le disque utilisé a une capacité d'environ 10 Go (9,54 GB). Les autres partitions correspondent sans doute à des filesystems hormis la partition numéro un qui représente la "zone de swap".*

L'affichage des partitions peut être mis en correspondance avec la liste des filesystems montés pour en visualiser directement l'utilisation réelle. Il est normal que la partition de "swap" n'apparaisse pas dans cette liste car il ne s'agit pas d'un filesystem.

```
# mount
/  on  /dev/dsk/c0d0s0  read/write/setuid/intr on mar 17 10:15
.............................................................
/home  on  /dev/dsk/c0d0s5  read/write/setuid/intr on mar 17 10:15
.............................................................
#
```

Solaris implémente aussi la commande **prtvtoc** (commande de la famille "Unix System V"). Cette commande attend en argument le nom du disque complet en mode "raw". L'affichage obtenu est moins intuitif car les tailles (colonne "sector count") sont exprimées en blocs de 512 octets :

```
# prtvtoc  /dev/rdsk/c0d0s2
.............................................................
                        First      Sector   Last
Partition  Tag  Flags   Sector     Count    Sector   Mount Directory
      0      2    00   1052352    4096512   5148863    /
      1      3    01      3024    1049328   1052351
      2      5    00         0   20004768  20004767
.............................................................
      5      8    00  11293632    8192016  19485647    /home
.............................................................
#
```

Linux

La commande **fdisk** est un utilitaire de gestion des partitions et comprend évidemment une possibilité d'affichage. Il existe aussi une commande **sfdisk** ainsi qu'une commande **cfdisk** plus conviviale (interface de menus déroulants).

```
# fdisk /dev/hda
.........................
Command (m for help): p
Disk /dev/hda: 16 heads, 63 sectors, 6304 cylinders
Units = cylinders of 1008 * 512 bytes
   Device Boot    Start       End    Blocks   Id  System
/dev/hda1   *         1      2000   1007968+   b  Win95 FAT32
/dev/hda2          2001      6304   2169216    5  Extended
/dev/hda5          2001      5657   1843096+  83  Linux
/dev/hda6          5658      6267    307408+  82  Linux swap
/dev/hda7          6268      6304     18616+  83  Linux
Command (m for help): q
#
```

Pour les disques de type IDE, les noms des partitions Linux sont de la forme "/dev/hdxy" où "x" est une lettre comprise entre a et h qui désigne le disque et "y" un numéro qui désigne la partition. Pour les disques de type SCSI, les noms des partitions sont de la forme "/dev/sdxy".

Les matériels de type PC sont limités à quatre partitions physiques. Il est possible de créer trois partitions primaires et une partition étendue au sein de laquelle on pourra créer autant de partitions logiques supplémentaires que nécessaire. Les quatre premiers chiffres sont associés aux partitions physiques. Les partitions numérotées à partir de 5 correspondent aux partitions logiques créées au sein de la partition physique étendue.

Dans cet exemple, le système Linux est installé derrière un système "Microsoft Windows". Il utilise une partition étendue "/dev/hda2" au sein de laquelle ont été générées trois partitions logiques dont la numérotation commencent donc au chiffre 5.

4.2.2 Organisation en volumes logiques

La plupart des versions permettent aujourd'hui une organisation en volumes logiques (LVM: "Logical Volume Manager") et les administrateurs disposent alors de beaucoup plus de souplesse. Dans une première approche, on peut considérer que les volumes logiques lèvent les limitations des partitions classiques. Un volume logique sera une région non nécessairement contiguë, pouvant s'étendre sur un ou plusieurs disques. Son agrandissement ultérieur sera possible. La gestion des disques par volumes logiques est obligatoire dans les versions AIX. HP-UX propose également cette démarche de préférence aux partitions. Linux et Solaris 9 disposent aussi d'une implémentation. La mise en oeuvre se fait par des commandes spécifiques qui mettent en jeu une terminologie commune.

4.2.2.a Terminologie du LVM (Logical Volume Manager)

Donnons quelques premières définitions résumées :

VG (Volume Group) Groupe de volumes (un ensemble de disques)
PV (Physical Volume) Volume physique (un disque)

LV (Logical Volume) Volume logique
Il s'agit d'un espace disque destiné, la plupart du temps, à contenir un filesystem. Ce sera donc l'équivalent, plus souple, d'une partition classique.

PP (Physical Partition) Partition physique (terminologie AIX)
PE (Physical Extent) Synonyme de PP (terminologie HP-UX)
Il s'agit d'unités d'allocation d'espace disque (4 ou 8 Mo par exemple) utilisées pour constituer des volumes logiques.

LP (Logical Partition) Partition logique (terminologie AIX)
LE (Logical Extent) Synonyme de LP (terminologie HP-UX)
Une partition logique correspond par défaut à une partition physique. Il est cependant possible d'indiquer qu'une partition logique sera associée à deux, voire trois, partitions physiques pour mettre ainsi implicitement en oeuvre une fonctionnalité de copies redondantes de l'information ("mirroring").

Il est important de noter dès ces premières définitions résumées que le terme de "partition" prend ici un sens complètement différent de celui des organisations classiques. L'équivalent fonctionnel (amélioré) de la partition classique est bien le volume logique. La confusion pourrait peut-être apparaître dans certaines implémentations telles que Linux où une partition classique peut être utilisée comme un "PV" (disque physique) afin d'être intégrée à un groupe de volumes.

Précisons, à l'aide des définitions résumées, les principes d'organisation.

Un disque ne peut pas être utilisé tant qu'il n'a pas été intégré à un groupe de volumes. Le groupe de volumes est donc une entité obligatoire qui englobe à la fois des disques physiques et les volumes logiques qui seront ensuite créés sur ces disques.

Le système comporte donc au moins un groupe de volumes généré pendant l'installation. D'éventuels disques supplémentaires, non associés à ce groupe de volumes initial, pourront y être ajoutés ultérieurement. Ils pourront aussi être intégrés à d'autres groupes de volumes qu'il s'agira de créer. Précisons qu'un disque ne peut appartenir qu'à un seul groupe de volumes.

La création d'un groupe de volumes s'accompagne du choix important de la taille des partitions ("PPs" ou "PEs"). Cette taille est souvent de quatre ou huit méga-octets par défaut et correspond à l'unité minimum d'allocation lors de la création ou de l'agrandissement d'un volume logique.

Un volume logique est un ensemble de partitions logiques ("LPs" ou "LEs"). Il est créé à l'intérieur d'un groupe de volumes pour accueillir, la plupart du temps, une structure de filesystem. Certaines limitations fonctionnelles des partitions Unix classiques sont levées :

Il est possible de répartir l'espace disque de manière non forcément contiguë et éventuellement sur plusieurs disques, notamment si on souhaite profiter de la fonctionnalité intégrée de "mirroring".

L'agrandissement dynamique devient possible en puisant dans le stock de partitions libres. Sauf dans le cas de configurations bien connues et calibrées, il est donc habituel et recommandé de trouver, dans un volume logique, des partitions non encore allouées. Cela constitue la condition bien évidente pour permettre l'agrandissement immédiat d'un volume logique existant. L'intégration d'un nouveau disque dans le groupe de volumes reste toujours possible pour bénéficier de nouvelles partitions libres et ne pas remettre en question l'organisation courante.

Les volumes logiques correspondent à des fichiers spéciaux et respectent la sémantique Unix traditionnelle (mode bloc et mode "raw bloc"). Les équivalents de la partition "zone de swap" et des partitions dédiées ("raw partitions") s'obtiennent tout naturellement via les volumes logiques.

4.2.2.b Visualisation de l'organisation (implémentation AIX)

La procédure d'installation AIX crée le groupe de volumes "**rootvg**" (fichier spécial "/dev/rootvg") avec une organisation imposée de volumes logiques et de filesystems. Si le système comporte plusieurs disques, l'installateur doit préciser le ou les disques qui feront partie de ce groupe de volumes initial. Les groupes de volumes, volumes logiques et filesystems complémentaires devront être créés après l'installation. De plus, la taille de certains filesystems devra être adaptée immédiatement aux besoins de l'exploitation.

La commande **lsvg** permet de visualiser, de façon plus ou moins détaillée, les groupes de volumes existants :

```
# lsvg  rootvg
VOLUME GROUP:   rootvg          VG IDENTIFIER:  00b35be6a8a0a194
VG STATE:       active          PP SIZE:        8 megabyte(s)
VG PERMISSION:  read/write      TOTAL PPs:      537 (4296 megabytes)
MAX LVs:        256             FREE PPs:       463 (3704 megabytes)
LVs:            8               USED PPs:       74 (592 megabytes)
OPEN LVs:       7               QUORUM:         2
TOTAL PVs:      1               VG DESCRIPTORS: 2
STALE PVs:      0               STALE PPs:      0
ACTIVE PVs:     1               AUTO ON:        yes
# lsvg  -l  rootvg
rootvg:
LV NAME    TYPE       LPs   PPs   PVs  LV STATE      MOUNT POINT
hd6        paging     18    18    1    open/syncd    N/A
hd5        boot       1     1     1    closed/syncd  N/A
hd8        jfslog     1     1     1    open/syncd    N/A
hd4        jfs        1     1     1    open/syncd    /
hd2        jfs        34    34    1    open/syncd    /usr
hd9var     jfs        3     3     1    open/syncd    /var
.....................................................
#
```

La première commande donne les caractéristiques générales du groupe de volumes. Sans détailler ici certains aspects spécifiques ou avancés, on remarque facilement la taille des partitions, le nombre de disques et le nombre de volumes logiques.

La deuxième forme ("option -l") détaille la liste des volumes logiques. Ils sont désignés par des fichiers spéciaux situés directement dans le répertoire "/dev". Le nom d'un volume logique est donc global au système. A notre niveau de discours, le type "jfs" indique la présence d'un filesystem et le type "paging" désigne l'espace de pagination ("swap"). Les tailles, en nombre de "LPs" et "PPs", sont identiques car il n'est pas fait usage de la possibilité de disque miroir.

4.2.2.c Visualisation de l'organisation (implémentation HP-UX)

La procédure d'installation HP-UX crée le groupe de volumes "**vg00**" (sous-répertoire de "/dev"). L'organisation en volumes logiques et filesystems peut être faite pendant ou après l'installation.

La commande **vgdisplay** permet de visualiser, de façon plus ou moins détaillée, les groupes de volumes existants.

```
# vgdisplay  vg00
--- Volume groups ---
VG Name                     /dev/vg00
VG Write Access             read/write
VG Status                   available
Max LV                      255
Cur LV                      8
Open LV                     8
Max PV                      16
Cur PV                      1
Act PV                      1
Max PE per PV               2500
VGDA                        2
PE Size (Mbytes)            4
Total PE                    511
Alloc PE                    422
Free PE                     89
Total PVG                   0
Total Spare PVs             0
Total Spare PVs in use      0
#
```

Cette commande sans option donne les caractéristiques générales du groupe de volumes. On remarque que le terme "PE" remplace le terme "PP" par rapport à la terminologie AIX.

```
# vgdisplay  -v  vg00
.....................................
   --- Logical volumes ---
   LV Name                     /dev/vg00/lvol1
   LV Status                   available/syncd
   LV Size (Mbytes)            68
```

```
   Current LE                17
   Allocated PE              17
   Used PV                   1

   LV Name                   /dev/vg00/lvol2
   LV Status                 available/syncd
   LV Size (Mbytes)          288
   Current LE                72
   Allocated PE              72
   Used PV                   1
.......................................
```

Cette deuxième forme ("option -v") détaille notamment les volumes logiques appartenant à ce groupe de volumes. Les volumes logiques sont désignés par des fichiers spéciaux situés dans le sous-répertoire portant le nom du groupe de volumes.

4.2.2.d Création d'un groupe de volumes (implémentation AIX)

Un disque physique correctement connecté est reconnu automatiquement par le système. Il correspond à un fichier spécial de nom "/dev/hdiskn" (où "n" représente le numéro d'ordre de configuration).

La création d'un groupe de volumes est prise en charge par la commande **mkvg** :

```
# mkvg  -s 8  -y vgstage  hdisk1  hdisk2
#
```

*Cette commande permet la création d'un groupe de volumes baptisé "vgstage" à partir de deux disques "hdisk1" et "hdisk2". La taille des partitions physiques sera de 8 Mo. L'activation du groupe de volumes est réalisée par un appel sous-jacent à la commande "**varyonvg**".*

4.2.2.e Création d'un groupe de volumes (implémentation HP-UX)

Un disque dur doit être au préalable initialisé en tant que "PV" (Physical Volume) pour pouvoir être intégré ensuite dans un groupe de volumes. Cette opération s'effectue via la commande **pvcreate** qui a pour effet de réserver un certain nombre d'emplacements adéquats sur le disque :

```
# pvcreate  /dev/rdsk/c0t5d0
Physical volume "/dev/rdsk/c0t5d0" has been successfully created.
#
```

Il convient ensuite de créer le sous-répertoire correspondant au futur groupe de volumes, d'y créer un fichier spécial baptisé "group" et enfin d'utiliser la commande dédiée **vgcreate**.

Le détail de ces opérations est donné par la documentation de référence "man lvm". L'usage du menu HP-UX "sam" peut, bien entendu, faciliter la mise en oeuvre.

```
# mkdir /dev/vgstage
# mknod  /dev/vgstage/group  c  64  0x010000
# vgcreate  -s 8  /dev/vgstage  /dev/dsk/c0t5d0
#
```

Cette commande permet la création d'un groupe de volumes baptisé "vgstage" à partir du disque "c0t5d0". La taille des "extents physiques" sera de 8 Mo. Le fichier spécial "group" correspond toujours au majeur 64. Le mineur prend la valeur "Ox0n0000" où "n" doit être différent pour chaque groupe de volumes.

4.2.2.f Autres implémentations

Linux

Le support des volumes logiques apparaît avec les versions 2.4.x du noyau. Les commandes de gestion sont assez proches de celles des versions HP-UX.

Une partition classique peut être utilisée comme "PV" (Physical Volume) si on positionne son type à la valeur "0x8e" (Linux LVM). Cette opération est réalisée via la commande **fdisk**. Un disque complet peut bien sur être utilisé comme "PV" mais il faut, au préalable, effacer son premier secteur. Cette opération peut être réalisée via la commande "dd".

Exemple

```
# dd  if=/dev/zero  of=/dev/hdb  bs=512  count=1
#
```

La première utilisation du gestionnaire de volumes logiques nécessite un appel à la commande **vgscan**. Cette commande crée notamment un fichier de description "/etc/lvmtab".

Pour terminer, la démarche est similaire à celle des versions HP-UX avec une utilisation nécessaire de la commande "pvcreate" pour préparer les disques à leur utilisation dans le contexte des volumes logiques :

```
# pvcreate  /dev/hda5  /dev/hda6
.................................
pvcreate -- physical volume "/dev/hda5" successfully created
pvcreate -- physical volume "/dev/hda6" successfully created
# vgcreate  vgstage  /dev/hda5  /dev/hda6
vgcreate -- INFO: using default physical extent size 4 MB
vgcreate -- INFO: maximum logical volume size is 255.99 Gigabyte
vgcreate -- doing automatic backup of volume group "vgstage"
vgcreate -- volume group "vgstage" successfully created and activated
#
```

Cette dernière commande permet la création d'un groupe de volumes baptisé "vgstage" à partir de deux partitions classiques utilisées en tant que "PVs". La taille des "extents physiques" sera de 4 Mo (taille par défaut).

Solaris 9

Le support des volumes logiques est intégré au système Solaris à partir de la version 9 (Solaris Volume Manager). Il fallait auparavant recourir à un produit optionnel.

Les commandes **metadb** et **metainit** permettent l'essentiel des manipulations.

4.3 Filesystems

Pour que les partitions ou les volumes logiques puissent être utilisés comme des arborescences, il est nécessaire d'y créer une structure de filesystem.

Comme évoqué plus tôt dans ce chapitre, nous avons choisi de conserver ce terme de filesystem, bien traditionnel dans le vocabulaire Unix, sans lui donner à tout prix une traduction.

La structure physique d'un filesystem peut se résumer ainsi :

des **blocs de gestion**

une **table des inodes**

On sait que chaque fichier Unix est associé, dans le répertoire où il se trouve, à un numéro d'inode (voir paragraphe A). La dimension de la table des inodes correspondra au nombre maximum de fichiers du filesystem.

la **zone des fichiers**

L'espace disque est alloué par blocs. Les blocs disque occupés par un fichier sont retrouvés grâce aux informations d'adressage situées dans son inode.

Le détail de cette structure physique et les mécanismes d'allocation de l'espace disque se sont perfectionnés en terme de sécurité et de performance au fur et à mesure des années.

Dans un ordre chronologique, on peut distinguer trois types de filesystems :

type **System V**

Cette première famille est quasiment obsolète aujourd'hui.

type **Berkeley**

Par rapport à la génération précédente, on dispose de meilleures performances (blocs disque plus grands, meilleurs algorithmes d'allocation...) et d'une plus grande fiabilité (redondance des blocs de gestion). Beaucoup de versions utilisent encore aujourd'hui ce genre de filesystem.

type **journalisé**

Il s'agit de la dernière génération qui tend à devenir le type par défaut sur la majorité des versions. Les performances en écriture sont meilleures et le mécanisme de journalisation accélère grandement les opérations de vérification lors du démarrage du système.

Dans la suite du chapitre, nous détaillerons les deux dernières familles fonctionnelles qui correspondent à des types aux noms variés selon les versions :

AIX	**jfs** (journalisé) et **jfs2** (évolution de jfs)
HP-UX	**hfs** (Berkeley) et **vxfs** (journalisé)
Solaris	**ufs** (Berkeley et journalisé dans les implémentations récentes)
Linux	**ext2** (Berkeley) et **ext3** ou **reiserfs** (journalisés)

4.3.1 Filesystems de type Berkeley

La structure physique de ces filesystems consiste en la répétition d'une même entité baptisée "groupe de cylindres". Chaque groupe de cylindres permet une redondance des blocs de gestion (bien synchronisés) et une répartition de la table des inodes ainsi que de la zone des fichiers.

Certains paramètres intéressants peuvent être calibrés via des options de la commande de création :

Taille des **blocs**

La taille par défaut peut être de 4K ou 8K octets et peut parfois aller jusqu'à 64K octets.

Taille des **fragments**

Si nécessaire, un bloc peut être partagé par plusieurs fichiers via la notion de fragment. Un bloc peut ainsi être divisé en un, deux, quatre ou huit fragments. Un fragment est occupé par un seul fichier.

Dimension de la table des inodes

Cette dimension correspond au nombre maximum de fichiers dans le filesystem. Le paramètre "**nbpi**" (number of bytes per inode) est proposé pour éventuellement modifier le choix par défaut.

Pourcentage minimum d'espace disque libre ("**free**")

Quand la place libre devient inférieure à ce pourcentage (5 ou 10% par défaut), seul le compte "root" peut encore consommer de l'espace disque et le filesystem apparaît plein pour les utilisateurs.

Critère d'optimisation ("**opt**")

Par défaut, le filesystem a une politique d'optimisation des performances. Il est possible de préférer une optimisation de l'espace disque.

Seuls les deux derniers paramètres ("free" et "opt") peuvent être modifiés après création.

4.3.1.a Création d'un filesystem "ufs" Solaris

La création de filesystem est, assez souvent, une opération interactive lors de l'installation du système. Elle est précédée de la création des partitions ou des volumes logiques. Cependant, lors de l'ajout ou de la réorganisation d'un disque ou encore lors d'une opération de restauration, l'administrateur peut être confronté à l'utilisation directe des commandes de création. Il est bien clair que l'opération de création de filesystem réinitialise l'espace disque associé par une structure vierge correspondant à une arborescence vide. Comme on le découvrira plus loin dans le chapitre, cette arborescence devra ensuite être montée pour pouvoir être accédée.

La commande **newfs** permet de créer un filesystem de type "ufs" pour Solaris. Il s'agit d'une commande de confort qui appelle la commande de plus bas niveau **mkfs**. Celle-ci peut néanmoins être invoquée directement (le détail de la syntaxe est donné par "man mkfs_ufs"). On doit bien sur disposer d'une partition qui a pu être définie via la commande "format".

Exemple

Dans ce premier exemple, on utilise l'option "-N" qui permet de simuler la création d'un filesystem sur la partition dont on donne le nom sous la forme "raw bloc". La commande affiche notamment le détail de la commande "mkfs" sous-jacente.

```
# newfs  -Nv  /dev/rdsk/c0d0s4
mkfs -F ufs -o N /dev/rdsk/c0d0s3 3072384 63 16 8192 1024 32 4 60 4096
t 0 -1 8 7
/dev/rdsk/c0d0s3: 3072384 sectors in 3048 cylinders of 16 tracks, 63
sectors
        1500,2MB in 96 cyl groups (32 c/g, 15,75MB/g, 3904 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 32352, 64672, 96992, 129312, 161632, 193952, 226272, 258592,
290912,
 323232, 355552, 387872, 420192, 452512, 484832, 516128, 548448,
580768,
.....................................
#
```

Dans ce deuxième exemple, on décide de créer réellement le filesystem sur la partition concernée en indiquant deux options modifiées : 4K octets pour la taille des blocs et 10% d'espace disque réservé au compte "root". La commande affiche le détail de la commande "mkfs" sous-jacente, le nombre de groupes de cylindres générés ainsi que les numéros des blocs de sauvegarde du "superbloc" (bloc de gestion indispensable).

```
# newfs  -v  -b 4096  -m 10  /dev/rdsk/c0d0s3
newfs: construct a new file system /dev/rdsk/c0d0s3: (y/n)? y
mkfs -F ufs /dev/rdsk/c0d0s3 3072384 63 16 4096 1024 32 10 60 4096 t 0
-1 8 14
/dev/rdsk/c0d0s3: 3072384 sectors in 3048 cylinders of 16 tracks, 63
sectors
        1500,2MB in 96 cyl groups (32 c/g, 15,75MB/g, 3904 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
 32, 32352, 64672, 96992, 129312, 161632, 193952, 226272, 258592,
290912,
 323232, 355552, 387872, 420192, 452512, 484832, 516128, 548448,
580768,
.....................................
#
```

4.3.1.b Création d'un filesystem "hfs" HP-UX

La commande **newfs** permet de créer un filesystem de type "hfs" pour HP-UX. Il s'agit d'une commande de confort qui appelle la commande de plus bas niveau **mkfs**. L'option fondamentale "-F" permet de choisir le type de filesystem concerné et le détail de syntaxe s'obtient naturellement par "man newfs_hfs".

On doit disposer d'un volume logique qu'il faut créer au préalable via la commande **lvcreate**.

Exemple

On crée tout d'abord un volume logique baptisé "monlv" de 24 Mo. Ce volume logique sera membre du groupe de volumes "stage". Le volume logique ainsi créé reproduit les conventions de nommage des partitions Unix classiques. Il y a donc une entrée en mode bloc et une autre en mode "raw bloc".

```
# lvcreate -L 24 -n monlv /dev/stage
Logical volume "/dev/stage/monlv" has been successfully created with
character device "/dev/stage/rmonlv".
Logical volume "/dev/stage/monlv" has been successfully extended.
Volume Group configuration for /dev/stage has been saved in
/etc/lvmconf/stage.conf
# ls -l /dev/stage
total 0
crw-r--r--   1   root   sys   64 0x010000   Mar 27 16:05   group
brw-r-----   1   root   sys   64 0x010001   Mar 28 10:33   monlv
crw-r-----   1   root   sys   64 0x010001   Mar 28 10:33   rmonlv
#
```

On crée ensuite le filesystem sur le volume logique "monlv". On donne le nom du volume logique sous la forme "raw bloc". L'option "-L" autorise les noms longs pour les fichiers (255 caractères). D'autre part, on choisit 4K octets pour la taille des blocs et 5% d'espace disque réservé au compte "root".

```
# newfs -F hfs -L -v -b 4096 -m 5 /dev/stage/rmonlv
/usr/sbin/mkfs -F hfs -L /dev/stage/rmonlv 24576 22 7 4096 1024 16 5 60
6144
mkfs (hfs): /dev/stage/rmonlv - 24576 sectors in 160 cylinders of 7
tracks, 22 sectors
25.2Mb in 10 cyl groups (16 c/g, 2.52Mb/g, 384 i/g)
Super block backups (for fsck -b) at:
    16,   2504,   4992,   7480,   9968,  12456,  14944,  17432,  19728,
22216
#
```

4.3.1.c Création d'un filesystem "ext2" Linux

La commande **mkfs** permet de créer un filesystem de type "ext2" pour Linux. L'option fondamentale "-t" permet de choisir le type de filesystem concerné mais il existe aussi une commande directe **mke2fs** (également appelée **mkfs.ext2**).

On doit disposer, soit d'une partition définie via la commande "fdisk", soit d'un volume logique qu'il faut créer au préalable via la commande "lvcreate" (très similaire à celle des versions HP-UX).

Exemple

On décide ici de créer un filesystem sur la partition "/dev/hda7" en indiquant 3% d'espace disque réservé au compte "root".

```
# mke2fs  -m 3  /dev/hda7
mke2fs 1.27 (8-Mar-2002)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
4656 inodes, 18616 blocks
558 blocks (3.00%) reserved for the super user
First data block=1
3 block groups
8192 blocks per group, 8192 fragments per group
1552 inodes per group
Superblock backups stored on blocks:
        8193
Writing inode tables: done
Writing superblocks and filesystem accounting information: done
This filesystem will be automatically checked every 33 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
#
```

4.3.2 Filesystems de type journalisé

Cette nouvelle génération de filesystems s'impose aujourd'hui comme standard de fait en améliorant encore performances et fiabilité. Les opérations en cours sur le filesystem sont conservées, sur disque, dans un journal des transactions qui fait partie de la structure même du filesystem (implémentations "vxfs" et "ext3") ou qui se trouve dans un volume logique spécialisé (implémentation "jfs"). Une transaction est un ensemble d'opérations élémentaires qui doivent obligatoirement être effectuées en totalité pour garantir la cohérence de la structure du filesystem. En cas d'arrêt anormal du système, les procédures de reprise consistent à examiner le contenu du journal afin de traiter d'éventuelles transactions non terminées. Ce mécanisme s'avère beaucoup plus rapide qu'une véritable vérification de la structure physique et constitue, de ce fait, un avantage intéressant par rapport à la génération Berkeley. Il faut cependant bien préciser que la journalisation ne garantit aucunement le contenu des fichiers, son seul objectif est bien la cohérence de la structure.

Lorsqu'une version propose plusieurs types de filesystems, il convient, a priori, de choisir le plus récent (à savoir le type journalisé). Les filesystems de type Berkeley continuent, bien entendu, à être proposés à titre de compatibilité. Il se pourrait qu'un logiciel nécessite ce type particulier (ce cas est tout de même rare) ou bien encore que des utilitaires ou services administratifs (quotas, scripts de sauvegardes...) ne soient pas adaptés à cette dernière génération. Il faut noter également que la notion de "free" (espace disque réservé au compte "root") n'existe pas pour les filesystems journalisés.

4.3.2.a Création d'un filesystem "jfs" AIX

Dans l'implémentation AIX, le journal des transactions consiste en un volume logique spécialisé qui se trouve en dehors du filesystem proprement dit. Il y a, dans chaque groupe de volumes, au moins un volume logique spécialisé qui gère la journalisation de plusieurs filesystems simultanément.

```
# lsvg  -l  rootvg
rootvg:
LV NAME       TYPE       LPs   PPs   PVs  LV STATE      MOUNT POINT
hd6           paging     37    37    1    open/syncd    N/A
hd5           boot       1     1     1    closed/syncd  N/A
hd8           jfslog     1     1     1    open/syncd    N/A
hd4           jfs        3     3     1    open/syncd    /
.....................................................
```

Les versions AIX 5L introduisent l'implémentation "jfs2" ("enhanced journaled filesystem"). Il s'agit d'un filesystem adapté au noyau 64 bits et amélioré en terme de performance et de tailles maximales autorisées. De plus, la gestion des inodes devient complètement dynamique.

La commande **crfs** permet de créer un filesystem de type "jfs" ou "jfs2" pour AIX. Deux méthodes sont possibles :

- Créer un volume logique au préalable via la commande **mklv** puis créer le filesystem associé
- Créer le filesystem en demandant la création implicite du volume logique associé.

La première méthode permet de contrôler complètement les caractéristiques du volume logique en terme de positionnement précis sur le disque ou encore de mise en oeuvre du "mirroring". La seconde méthode est plus simple mais ne permet pas de maîtriser finement l'allocation de l'espace disque.

Exemple (Première méthode)

On crée tout d'abord un volume logique baptisé "monlv" dont la taille est de 5 "LPs" ("logical partitions"). Si la taille des "LPs" est de 4 Mo, cela correspondra bien sur à 20 Mo. Ce volume logique sera membre du groupe de volumes "rootvg". Les volumes logiques AIX sont des fichiers spéciaux situés directement dans le répertoire "/dev".

```
# mklv  -y monlv  rootvg  5
# ls -l /dev/*monlv
brw-rw----   1 root   system   10,9   Jan 28 14:28   /dev/monlv
crw-rw----   1 root   system   10,9   Jan 28 14:28   /dev/rmonlv
#
```

On crée ensuite le filesystem sur le volume logique "monlv". L'option "-v" indique le type de filesystem souhaité. D'autre part, on donne des informations anticipées sur l'activité de montage (nom du répertoire et indicateur de montage automatique au démarrage du système). La commande affiche les tailles en nombre de blocs de 512 octets. La commande "lsfs" permet ensuite de visualiser les caractéristiques résumées d'un filesystem en donnant le nom du répertoire de montage.

```
# crfs  -v jfs  -d 'monlv'  -m '/rep'  -A 'yes'
Based on the parameters chosen, the new /rep JFS file system
is limited to a maximum size of 134217728 (512 byte blocks)
New File System size is 40960
# lsfs  /rep
Name        Nodename   Mount Pt   VFS   Size    Options   Auto
/dev/monlv  --         /rep       jfs   40960   --        yes
#
```

Exemple (Seconde méthode)

La création du filesystem et du volume logique associé peut aussi se faire en une seule étape. On indique le nom du groupe de volumes concerné. La taille doit être exprimée en nombre de blocs de 512 octets. Il y a création implicite du volume logique et la taille est arrondie au nombre de "LPs" nécessaires.

```
# crfs  -v jfs  -g 'rootvg'  -a size=40000  -m '/rep2'  -A 'yes'
Based on the parameters chosen, the new /rep2 JFS file system
is limited to a maximum size of 134217728 (512 byte blocks)
New File System size is 40960
#
```

4.3.2.b Création d'un filesystem "vxfs" HP-UX

Dans l'implémentation "vxfs" (Veritas Filesystem), le journal des transactions fait partie de la structure du filesystem lui-même. Le nombre d'inodes est géré dynamiquement.

La commande **newfs** permet de créer un filesystem de type "vxfs" pour HP-UX. L'option fondamentale "-F" permet en effet de choisir le type de filesystem concerné et le détail de syntaxe s'obtient naturellement par "man newfs_vxfs".

Comme évoqué précédemment pour les filesystems de type "hfs", on doit, bien sur, disposer d'un volume logique qu'il faut créer au préalable via la commande **lvcreate**.

```
# newfs  -F vxfs  /dev/stage/rmonlv
    version 3 layout
    24576 sectors, 24576 blocks of size 1024, log size 1024 blocks
    unlimited inodes, 24576 data blocks, 23480 free data blocks
    1 allocation units of 32768 blocks, 32768 data blocks
    last allocation unit has 24576 data blocks
    first allocation unit starts at block 0
    overhead per allocation unit is 0 blocks
#
```

On crée le filesystem sur le volume logique "monlv" dont on donne le nom sous la forme "raw bloc". L'affichage fait apparaître la taille du journal des transactions ("log size") et le fait que le nombre d'inodes sera géré dynamiquement. On peut noter également que les groupes de cylindres d'un filesystem de type Berkeley trouvent ici leur équivalent fonctionnel avec les unités d'allocation ("allocation unit").

4.3.2.c Création d'un filesystem "ext3" Linux

L'implémentation Linux "ext3" s'appuie complètement sur la génération précédente "ext2" en y ajoutant un journal des transactions. La même commande **mkfs** permet de créer un filesystem de type "ext3" via l'option fondamentale "-t". La commande directe **mke2fs** peut aussi être utilisée avec, cette fois, l'option "-j".

On doit bien sur disposer, soit d'une partition définie via la commande "fdisk", soit d'un volume logique qu'il faut créer au préalable via la commande "lvcreate".

Exemple

On crée un filesystem de type "ext3" sur la partition "/dev/hda7". L'affichage de la commande mentionne la création du journal.

```
# mke2fs  -j  /dev/hda7
mke2fs 1.27 (8-Mar-2002)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
4656 inodes, 18616 blocks
930 blocks (5.00%) reserved for the super user
First data block=1
3 block groups
8192 blocks per group, 8192 fragments per group
1552 inodes per group
Superblock backups stored on blocks:
        8193
Writing inode tables: done
Creating journal (1024 blocks): done
Writing superblocks and filesystem accounting information: done
This filesystem will be automatically checked every 24 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
#
```

*Une autre implémentation de filesystem journalisé "reiserfs" est disponible dans les environnements Linux. La commande **mkreiserfs** permet de créer ce type de filesystem.*

4.4 Montage et démontage

L'opération de montage est nécessaire pour accéder aux fichiers d'un filesystem par les commandes Unix classiques.

Elle consiste à accrocher l'arborescence dans un répertoire vide d'un filesystem lui-même déjà monté. Ce répertoire s'appelle le point de montage ("mount point").

Le filesystem contenant la racine Unix et les répertoires de premier niveau est baptisé "**filesystem root**". Il est monté dès le début de la phase de démarrage du système. Cela permet ensuite la constitution, via d'autres montages successifs, de l'arborescence complète vue par les utilisateurs.

Un fichier de configuration, spécifique à chaque version, permet de décrire les montages automatiques au démarrage et de bénéficier d'abrégés dans les commandes ultérieures de gestion. Ces fichiers sont détaillés plus loin dans ce paragraphe.

La commande **mount** réalise l'opération de montage.

Dans sa syntaxe complète, elle nécessite deux arguments :

- le nom de la partition ou du volume logique contenant le filesystem (exprimé sous la forme du fichier spécial en mode bloc)
- le répertoire de montage

Des options spécifiques au type de filesystem peuvent être ajoutées. L'option la plus intuitive permet d'effectuer le montage en lecture seule. Dans ce cas de figure, aucune modification, même par le compte "root", ne peut avoir lieu dans l'arborescence.

La commande "mount" sans arguments donne la liste des filesystems montés. Les utilisateurs ordinaires ont accès à cette seule forme de la commande.

Dès ces premières informations, on comprend certains intérêts majeurs d'une organisation en filesystems :

- Mise à disposition maîtrisée (montage ou démontage)
- Séparation fonctionnelle du système proprement dit et des données ou logiciels
- Sécurité complète de certaines arborescences (montage en lecture seulement)

4.4.1 Illustration par l'exemple

```
# mount
/      on /dev/vg00/lvol3   log on Mar 15 18:39:38
/stand on /dev/vg00/lvol1   defaults on Mar 15 18:39:40
/var   on /dev/vg00/lvol8   delaylog,nodatainlog on Mar 15 18:39:48
/usr   on /dev/vg00/lvol7   delaylog,nodatainlog on Mar 15 18:39:48
/tmp   on /dev/vg00/lvol4   delaylog,nodatainlog on Mar 15 18:39:48
/opt   on /dev/vg00/lvol6   delaylog,nodatainlog on Mar 15 18:39:49
/home  on /dev/vg00/lvol5   delaylog,nodatainlog on Mar 15 18:39:49
#
```

Cette commande donne la liste des filesystems montés. Par exemple, la dernière ligne signifie que le filesystem situé sur le volume logique "/dev/vg00/lvol5" (il s'agit ici d'un exemple Hp-UX) est actuellement monté et donc accessible via le répertoire "/home" qui est bien un répertoire du "filesystem root" (décrit sur la première ligne de l'affichage).

```
# ls -al /home2
total 2
drwxr-xr-x   2 root       sys          96   Mar 28 11:51  .
drwxr-xr-x  18 root       root       1024   Mar 28 11:51  ..
# mount  -F vxfs  /dev/stage/monlv  /home2
# mount
.....................................................................
/home2  on /dev/stage/monlv   log,nodatainlog on Mar 28 11:55:12
#
```

On dispose ici d'un répertoire "/home2" (répertoire vide du "filesystem root") et on y réalise le montage d'un filesystem (de type "vxfs") situé sur le volume logique "/dev/stage/monlv". Après cette opération, le nouveau montage apparaît dans la liste.

```
# cd  /home2
# date  >  toto
# ls  -al  /home2
total 4
drwxr-xr-x   3 root       root         96   Mar 28 11:58  .
drwxr-xr-x  18 root       root       1024   Mar 28 11:51  ..
drwxr-xr-x   2 root       root         96   Mar 28 11:38  lost+found
-rw-r--r--   1 root       sys          29   Mar 28 11:58  toto
#
```

*On crée ici un fichier "toto" dans le répertoire "/home2". L'important est de comprendre que ce fichier est bien créé sur l'espace disque correspondant au volume logique "/dev/stage/monlv" et que cette appartenance physique est transparente pour l'utilisateur de l'arborescence Unix globale. On remarque également la présence du sous-répertoire "**lost+found**". Ce répertoire a été généré par la commande de création du filesystem. Il joue un rôle dans des opérations éventuelles de réparation qui seront développées plus loin dans ce chapitre.*

4.4.2 Quelques remarques

Un répertoire de montage est, a priori, dédié à cet usage. Il ne contient donc aucun fichier. Si c'était le cas, ceux-ci seraient masqués et inaccessibles pendant toute la durée du montage.

Le répertoire de montage est d'ailleurs lui-même masqué par la racine du filesystem monté. Il est donc très important de **veiller au bon positionnement du propriétaire, du groupe et des permissions de cette racine, lors du premier montage**, avant de commencer à créer des fichiers dans le filesystem.

Un exemple AIX

```
# mount
  /dev/hd4      /       jfs   Jul 18 09:19  rw,log=/dev/hd8
  /dev/hd2      /usr    jfs   Jul 18 09:19  rw,log=/dev/hd8
  /dev/hd9var   /var    jfs   Jul 18 09:19  rw,log=/dev/hd8
  /dev/hd3      /tmp    jfs   Jul 18 09:19  rw,log=/dev/hd8
  /dev/hd1      /home   jfs   Jul 18 09:21  rw,log=/dev/hd8
# ls  -ld  /ecole
drwxr-xr-x   2  root   system    512   Jul 17 17:43   /ecole
# mount  /dev/ecole  /ecole
# mount
  /dev/hd4      /       jfs   Jul 18 09:19  rw,log=/dev/hd8
  /dev/hd2      /usr    jfs   Jul 18 09:19  rw,log=/dev/hd8
  /dev/hd9var   /var    jfs   Jul 18 09:19  rw,log=/dev/hd8
  /dev/hd3      /tmp    jfs   Jul 18 09:19  rw,log=/dev/hd8
  /dev/hd1      /home   jfs   Jul 18 09:21  rw,log=/dev/hd8
  /dev/ecole    /ecole  jfs   Jul 18 10:27  rw,log=/dev/hd8
# ls  -ld  /ecole
drwxr-sr-x   2  sys   sys        512   Jul 17 18:00   /ecole
#
```

Dans cet exemple (issu d'une version AIX 4.3), on constate que la racine du filesystem appartient au compte "sys" et au groupe de même nom. De plus, le répertoire est doté de la permission "sgid", ce qui signifie que tous les fichiers créés dans le filesystem seront attribués au groupe "sys". Il serait donc très important, via les commandes respectives "chown", "chgrp" et "chmod", de modifier ces divers aspects pour être tout de suite en phase avec l'utilisation concrète de cette arborescence.

En ce qui concerne le répertoire "/ecole" proprement dit, il est de propriétaire et groupe "root" et ses permissions "755" semblent tout à fait adaptées pour une bonne sécurité dans le cas où le filesystem ne serait pas monté.

Excepté dans les versions Linux, un filesystem ne peut être monté qu'à un seul endroit à la fois.

Dans cet exemple Solaris, il est impossible de monter un filesystem à deux endroits simultanément.

```
# mount  /dev/dsk/c0d0s3  /manip
# mount  /dev/dsk/c0d0s3  /mnt
mount: /dev/dsk/c0d0s3 is already mounted, /mnt is busy,
        or the allowable number of mount points has been exceeded
#
```

Dans cet exemple Linux, la manipulation est autorisée.

```
# mount  /dev/hda7  /manip
# mount  /dev/hda7  /mnt
# mount
.....................................
/dev/hda7  on  /manip  type ext2 (rw)
/dev/hda7  on  /mnt    type ext2 (rw)
#
```

Le montage en lecture seule apporte une sécurité complète pour des arborescences figées car aucune modification n'y est possible, y compris pour le compte "root" :

```
# mount  -r  /dev/dsk/c0d0s3  /manip1
# cp  /etc/group  /manip1/zorro
cp: cannot create /manip1/zorro: Read-only file system
#
```

4.4.3 Démontage

L'opération de démontage correspond à la commande **umount** à laquelle on donne en argument le nom du répertoire de montage ou encore le nom (en mode bloc) de la partition ou du volume logique concerné. Le répertoire de montage est souvent plus court au niveau de la frappe et est, de fait, plus dans les habitudes.

S'il existe au moins un processus pour lequel le répertoire courant ou des fichiers ouverts font partie du filesystem, la commande échoue en indiquant le fait que le filesystem est occupé. La commande **fuser** est utile pour identifier les processus qui empêchent le démontage.

Exemple Linux

```
# mount
.................................
/dev/hda7 on /manip type ext2 (rw)
# cd  /manip
# umount  /manip
umount: /manip: device is busy
# fuser  -u  /dev/hda7
# fuser  -mu  /dev/hda7
/manip:       1466c(root)
# ps  -ef  |  awk  '$2 == 1466 { print $0}'
root  1466   1465   0   13:09   pts/0   00:00:00   bash
#
```

Dans cet exemple, le démontage du filesystem est impossible car le répertoire courant du processus shell (PID 1466) du compte "root" appartient au filesystem. On donne en argument le nom de la partition concernée de préférence au nom du répertoire de montage. En effet, cela assure une recherche récursive à partir du point de montage sans se limiter à celui-ci. Il faut noter que l'option "-m" est alors nécessaire sous Linux. De plus, l'option "-u" est recommandée car elle affiche les noms des propriétaires des processus.

```
# cd  /
# umount  /manip
# ls -al /manip
total 8
drwxrwx---    2 root   root   4096   Dec 30 15:36 .
drwxr-xr-x   20 root   root   4096   Mar 31 08:42 ..
#
```

Le fait de quitter le filesystem en se positionnant par exemple à la racine Unix (celle du "filesystem root" toujours occupé) suffit pour permettre le démontage. Le répertoire "/manip" apparaît alors à nouveau vide suite à cette opération.

4.4.4 Fichier de description des filesystems

Toutes les versions Unix intègrent un fichier de description des filesystems dont le rôle essentiel est de décrire les montages automatiques au moment du démarrage du système.

Ce fichier permet également de bénéficier d'abrégés dans les commandes de gestion. Par exemple, la commande "mount" ne nécessitera qu'un seul argument (souvent, le point de montage, plus court en terme de frappe) car elle saura trouver les autres informations associées dans le fichier.

Le nom et la syntaxe interne du fichier varient selon les versions.

Fichier /etc/vfstab (Solaris)

Chaque ligne du fichier est constituée de sept champs :
- Nom de la partition en mode bloc
- Nom de la partition en mode "raw bloc"
- Nom du répertoire de montage
- Type de filesystem
- Indicateur de vérification (lié à la commande "fsck" décrite plus loin dans ce chapitre)
- Indicateur de montage au démarrage (La ligne du "filesystem root" comporte la valeur "no" car celui-ci est déjà monté lorsque le fichier est examiné.)
- Options de montage (Le signe - remplace toutes les options par défaut. L'option "logging" permet à un filesystem "ufs" de se comporter comme un filesystem journalisé via l'ajout d'un journal des transactions.)

Exemples de lignes

```
/dev/dsk/c0d0s0  /dev/rdsk/c0d0s0  /      ufs  1  no   -
/dev/dsk/c0d0s5  /dev/rdsk/c0d0s5  /home  ufs  2  yes  rw,logging
```

Fichier /etc/fstab (HP-UX, Linux)

Chaque ligne du fichier est constituée de six champs :
- Nom de la partition ou du volume logique en mode bloc
- Nom du répertoire de montage
- Type de filesystem
- Options de montage (Le terme "defaults" remplace toutes les options par défaut.)
- Indicateur de sauvegarde (lié à la commande de sauvegarde "dump" et peu souvent utilisé)
- Indicateur de vérification (lié à la commande "fsck" décrite plus loin)

Exemples de lignes

HP-UX

```
/dev/vg00/lvol1   /       vxfs   delaylog      0   1
/dev/vg00/lvol3   /home   hfs    rw,quota      0   2
```

Linux

```
/dev/hda5         /       ext3   defaults      1   1
/dev/hda7         /home   ext2   rw,usrquota   1   2
```

Fichier /etc/filesystems (AIX)

Le fichier est organisé en strophes portant le nom du répertoire de montage.

Exemple de strophe

```
/home:
     dev        =  /dev/hd1
     vfs        =  jfs
     log        =  /dev/hd8
     mount      =  true
     options    =  rw
     account    =  false
```

Quelques attributs

dev	Nom du volume logique
vfs	Type de filesystem
log	Volume logique de journalisation associé au filesystem
mount	Indicateur de montage au démarrage
options	Options de montage
account	Indicateur d'activation du service de comptabilité ("accounting")

Inventaire des principales options de montage

rw	Lecture/Écriture
ro	Lecture seule
nosuid	Désactivation de la permission "suid"
nodev	Interdiction de l'accès aux fichiers spéciaux (AIX, Linux)
noauto	Désactivation du montage automatique au démarrage (Linux)
user	Autorisation de montage par les utilisateurs ordinaires (Linux)
largefiles	Autorisation de fichiers de plus de 2Go (Solaris, HP-UX)
logging	Activation d'un journal des transactions (Solaris "ufs")
delaylog	Gestion du journal orientée vers l'amélioration des performances (HP-UX "vxfs")
quota	Activation des quotas (HP-UX, Solaris)
usrquota	Activation des quotas pour les utilisateurs (Linux)
grpquota	Activation des quotas pour les groupes (Linux)

(L'activation des quotas AIX nécessite l'ajout d'une ligne "quota = userquota,groupquota" dans la strophe de "/etc/filesystems")

Les quotas sont présentés dans la suite de ce chapitre.

4.5 Agrandissement et suppression de filesystem

L'opération d'agrandissement de filesystem est disponible dans les organisations en volumes logiques. Le détail de mise en oeuvre se décline par versions.

4.5.1 Agrandissement d'un filesystem sous HP-UX

Il est nécessaire de démonter le filesystem et d'opérer en deux étapes :

Agrandir le volume logique via la commande **lvextend**

Aligner ensuite la taille du filesystem sur celle du volume logique via la commande **extendfs**

Exemple

Affichage de la taille actuelle (24 Mo)

```
# bdf  /home2
Filesystem       kbytes   used   avail    %used  Mounted on
/dev/stage/monlv   24576    1109   22008    5%    /home2
#
```

Démontage obligatoire avant l'agrandissement du volume logique (32 Mo)

```
# umount  /home2
# lvextend  -L 32  /dev/stage/monlv
Logical volume "/dev/stage/monlv" has been successfully extended.
Volume Group configuration for /dev/stage has been saved in
/etc/lvmconf/stage.conf
#
```

Le filesystem n'a pas enregistré l'agrandissement, il faut aligner sa taille sur celle du volume logique. Après montage, on constate que l'agrandissement a réussi.

```
# bdf  /dev/stage/monlv
Filesystem       kbytes   used   avail    %used  Mounted on
/dev/stage/monlv   24576    1109   22008    5%
# extendfs  -F vxfs  /dev/stage/monlv
# mount  -F vxfs  /dev/stage/monlv  /home2
# bdf  /home2
Filesystem       kbytes   used   avail    %used  Mounted on
/dev/stage/monlv   32768    1109   29688    4%     /home2
#
```

4.5.2 Agrandissement d'un filesystem sous Linux

Dans un contexte éventuel de volumes logiques sous Linux, on dispose de la commande **e2fsadm** qui permet l'agrandissement d'un filesystem de type "ext2" ou "ext3". Le filesystem doit être démonté pour effectuer l'opération.

Cette commande effectue tout d'abord une vérification de la cohérence du filesystem (**e2fsck**) puis un agrandissement du volume logique (**lvextend**) puis enfin un alignement de la taille du filesystem (**resize2fs**).

Exemple

```
# e2fsadm  -L+4M  /dev/stage/monlv
e2fsck 1.27 (8-Mar-2002)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/stage/monlv: 11/2048 files (0.0% non-contiguous), 274/8192 blocks
lvextend -- extending logical volume "/dev/stage/monlv" to 12 MB
lvextend -- doing automatic backup of volume group "stage"
lvextend -- logical volume "/dev/stage/monlv" successfully extended

resize2fs 1.27 (8-Mar-2002)
Begin pass 1 (max = 1)
Extending the inode table     XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Begin pass 3 (max = 1)
Scanning inode table          XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
The filesystem on /dev/stage/monlv is now 12288 blocks long.

e2fsadm -- ext2fs in logical volume /dev/stage/monlv successfully
extended to 12 MB
#
```

4.5.3 Agrandissement d'un filesystem sous AIX

En restant homogène avec la démarche de création, deux méthodes d'agrandissement sont possibles :

- Agrandir le volume logique au préalable via la commande **extendlv** puis aligner la taille du filesystem associé (commande **chfs**) car celui-ci n'a pas eu connaissance de la nouvelle taille de volume logique.
- Agrandir directement le filesystem (même commande **chfs**) en provoquant un agrandissement implicite du volume logique associé.

La première méthode permet de continuer à contrôler complètement les caractéristiques du volume logique en terme de positionnement précis sur le disque.

L'opération ne nécessite pas de démonter le filesystem.

Exemple (deuxième méthode)

```
# chfs  -a size=96000  /rep
#
```

Le filesystem concerné est désigné par son point de montage habituel (nom de la strophe dans le fichier "/etc/filesystems"). La taille est exprimée en nombre de blocs de 512 octets. Il suffit de faire un calcul approximatif (nombre de Mo multiplié par 2000) puisque ce nombre sera aligné exactement sur un multiple de la taille des "PPs" du groupe de volumes. Ici, pour une taille souhaitée de 48 Mo, on indique 96000 blocs. Il serait possible d'indiquer une taille relative en la faisant précéder du signe + (par exemple, pour une augmentation de 24 Mo, on indiquerait +48000).

4.5.4 Suppression de filesystem

Lors d'une éventuelle réorganisation des disques, l'administrateur peut souhaiter supprimer un filesystem.

Dans une organisation en partitions classiques, il n'y a pas de commande de suppression. Pour écraser un filesystem, il suffit d'en recréer un sur la partition concernée.

Dans une organisation en volumes logiques, il existe une commande officielle qui permet de supprimer aussi le volume logique associé. Les "PPs" (AIX) ou "PEs" (HP-UX, Linux) correspondantes sont ainsi réintégrées dans le stock du groupe de volumes concerné.

Sous Linux et HP-UX, la commande **lvremove** permet de supprimer un volume logique et, par conséquent, le filesystem correspondant.

Par contre sous AIX, la commande de suppression de volume logique (**rmlv**) ne supprime pas le filesystem associé (les données sont bien sur perdues mais le ménage n'est pas fait correctement dans les fichiers de configuration). Il convient d'utiliser la commande **rmfs** qui supprime le filesystem et le volume logique en une seule opération. Elle est dotée de l'option "-r" pour supprimer aussi le répertoire de montage.

Exemple HP-UX

```
# lvremove  /dev/stage/monlv
The logical volume "/dev/stage/monlv" is not empty;
do you really want to delete the logical volume (y/n) : y
lvremove: Couldn't delete logical volume "/dev/stage/monlv":
The specified logical volume is open, or
a sparing operation is in progress.
Volume Group configuration for /dev/stage has been saved in
/etc/lvmconf/stage.conf
# umount  /home2
# lvremove  -f  /dev/stage/monlv
Logical volume "/dev/stage/monlv" has been successfully removed.
Volume Group configuration for /dev/stage has been saved in
/etc/lvmconf/stage.conf
#
```

La suppression ne peut se faire que si le filesystem est démonté, ce qui est bien normal. L'option "-f" permet de sauter la question de confirmation.

4.6 Vérification et paramétrages de filesystem

4.6.1 Vérification et réparation

Un filesystem peut se trouver incohérent dans le cas d'un arrêt anormal du système. Ceci est du aux écritures physiques différées qu'impliquent les entrées/sorties en mode bloc.

Lors du démarrage du système, un contrôle d'intégrité de certains types de filesystems est réalisé par la commande **fsck**. Sur la plupart des versions, l'ordre et l'éventuelle simultanéité de ce contrôle sont demandés par un champ du fichier des filesystems ("/etc/vfstab" pour Solaris, "/etc/fstab" pour Linux et HP-UX).

Le temps d'exécution de la commande sur un filesystem non journalisé peut être long car celle-ci parcourt le disque à la recherche de diverses incohérences avec un algorithme en plusieurs passes.

De plus en plus, la commande n'est véritablement lancée que si le filesystem n'a pas été démonté correctement (un indicateur dans le "superbloc" permet de le savoir).

Dans certaines rares circonstances (doute sur l'intégrité physique du disque, vérification avant sauvegarde...), l'administrateur peut souhaiter lancer lui-même cette commande.

Il convient alors de **démonter le filesystem** au préalable. De plus, le nom du filesystem (partition ou volume logique) doit être donné sous sa forme "raw bloc".

Si le filesystem ne peut pas être démonté ("filesystem root" par exemple), il est important qu'il n'y ait pas d'autres activités simultanées. Un redémarrage peut d'ailleurs être nécessaire ou recommandé après ou à la place de cette manipulation.

Si aucune erreur n'est détectée, la commande affiche un compte-rendu (nombre de fichiers, taille, espace libre, taux de fragmentation...).

Dans le cas où des incohérences seraient détectées, la commande devient interactive en posant des questions. Une réponse "n" (no) correspond à une simple consultation. Une réponse "y" (yes) correspond le plus souvent au meilleur compromis pour tenter une réparation.

Deux options majeures "-n" et '-y" permettent d'ailleurs de lancer la commande de façon non interactive en prévoyant ces deux réponses respectives automatiques. Les programmes de démarrage utilisent souvent ce traitement automatisé en cas de problème.

Certains fichiers alloués mais non référencés peuvent parfois être récupérés dans le répertoire **lost+found** qui doit être présent à la racine de chaque filesystem. Si ce répertoire n'est pas généré par la commande de création ou bien s'il a été supprimé par erreur, il peut être créé par une commande spécialisée dont le nom usuel est **mklost+found**.

Il est à noter cependant que certaines implémentations ne créent le répertoire que lorsqu'il devient nécessaire. La documentation système permet de vérifier ce point et de savoir si l'absence du répertoire "lost+found" constitue ou non un problème potentiel. L'examen du contenu du répertoire est, de toute façon, recommandé après toute opération de réparation via la commande "fsck". Les éventuels fichiers présents pourront être identifiés en terme de contenu par la commande "file" puis récupérés ou simplement supprimés pour réintégrer correctement le stock de blocs libres.

En ce qui concerne notamment les filesystems journalisés, un véritable parcours du disque par "fsck" doit être forcé par une option afin de passer outre le mécanisme normal du journal des transactions.

On devra également souvent forcer le contrôle par cette même option dans le cas de filesystems "Berkeley" notés comme corrects dans les informations du "superbloc".

Exemples

```
# fsck  /home
** Checking /dev/home (/home) MOUNTED FILE SYSTEM;
WRITING SUPPRESSED;
Checking a mounted file system does not produce dependable results.
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Inode Map
** Phase 6 - Check Block Map
1430 files 19600 blocks 185200 free
#
```

Dans cet exemple AIX, on désigne le filesystem sous son nom abrégé (nom de la strophe dans "/etc/filesystems"). Celui-ci est monté et la commande indique qu'il ne sera pas autorisé de le modifier. La commande ne détecte cependant pas d'erreurs car la commande "fsck" est la seule en activité sur le filesystem.

```
# fsck  /dev/vg00/lvol3
fsck: /dev/vg00/lvol3: mounted file system
continue (y/n)? n
# umount  /home
# fsck  /dev/vg00/rlvol3
** /dev/vg00/rlvol3
** Last Mounted on /home
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
2 files, 0 icont, 9 used, 19852 free (12 frags, 2480 blocks)
#
```

Dans cet exemple HP-UX, on constate aussi que le filesystem est monté. On décide alors de le démonter pour opérer la vérification. On remarque que le nom du volume logique concerné est donné sous sa forme "raw bloc".

```
# fsck  -F vxfs  /dev/rdsk/c0t0d0s4
file system is clean - log replay is not required
# fsck  -F vxfs  -o full  /dev/rdsk/c0t0d0s4
log replay in progress
pass1 - checking inode sanity and blocks
pass2 - checking directory linkage
pass3 - checking reference counts
pass4 - checking resource maps
OK to clear log? (ynq) y
set state to CLEAN? (ynq) y
#
```

Dans cet exemple de vérification d'un filesystem journalisé "vxfs", l'option spécifique "-o full" est utilisée pour réaliser un contrôle complet après application du journal des transactions.

```
# fsck  /dev/stage/monlv
fsck 1.27 (8-Mar-2002)
e2fsck 1.27 (8-Mar-2002)
/dev/stage/monlv: clean, 11/4096 files, 534/12288 blocks
# e2fsck  -f  /dev/stage/monlv
e2fsck 1.27 (8-Mar-2002)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/stage/monlv: 11/4096 files, 534/12288 blocks
#
```

*Dans cet exemple de vérification d'un filesystem "ext2" Linux, l'option spécifique "-f" est utilisée pour réaliser un contrôle complet malgré l'information "clean" du "superbloc" . On remarque également que la commande "fsck" appelle, en fait, la commande **e2fsck** et que celle-ci peut donc être invoquée directement.*

4.6.2 Paramétrages

Quelques caractéristiques peuvent être modifiées après la création d'un filesystem. La commande **tunefs** permet notamment de gérer le paramètre "free" (espace disque réservé au compte "root") des filesystems de type "Berkeley".

Exemple Solaris

Dans un premier temps, l'option "-m" de la commande de bas niveau "mkfs" permet de visualiser les caractéristiques :

```
# mkfs  -F ufs  -m  /dev/dsk/c0d0s3
mkfs -F ufs -o nsect=63,ntrack=16,bsize=8192,fragsize=1024,cgsize=32
,free=10,rps=60,nbpi=4156,opt=t,apc=0,gap=0,nrpos=8,maxcontig=7
/dev/dsk/c0d0s3 3072384
#
```

On décide alors de modifier le paramètre "free" en le réduisant ici à 5%.

```
# tunefs  -m  5  /dev/dsk/c0d0s3
minimum percentage of free space changes from 10% to 5%
#
```

Dans les implémentations Linux, on dispose de la commande **tune2fs** qui permet un peu plus de choses que la commande "tunefs" des autres versions.

Exemples

```
# tune2fs  -l  /dev/stage/monlv
.............................
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              4096
Block count:              12288
Reserved block count:     491
Free blocks:              11754
Free inodes:              4085
First block:              1
Block size:               1024
Fragment size:            1024
.............................
Check interval:           15552000 (6 months)
#
```

L'option "-l" permet de visualiser les caractéristiques du filesystem.

```
# tune2fs  -m 5  /dev/stage/monlv
tune2fs 1.27 (8-Mar-2002)
Setting reserved blocks percentage to 5 (610 blocks)
#
```

On décide ici de modifier le paramètre "free" en lui donnant une valeur de 5%.

```
# tune2fs  -j  /dev/stage/monlv
tune2fs 1.27 (8-Mar-2002)
Creating journal inode: done
This filesystem will be automatically checked every 20 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
#
```

L'option "-j" permet de transformer un filesystem "ext2" en un filesystem "ext3" (journalisé). En effet, l'implémentation "ext3" se veut compatible avec l'ancienne génération en permettant cette transformation rapide sans avoir à sauvegarder les données, recréer le filesystem et le restaurer.

4.7 Quotas

Le principe des quotas est d'instaurer, pour certains utilisateurs ou certains groupes, des limitations de consommation dans quelques filesystems. L'administrateur peut agir, soit sur le nombre de blocs disques, soit sur le nombre de fichiers (inodes), soit sur les deux aspects simultanément.

Deux limites sont prévues :

"soft limit"

Il s'agit d'une valeur qu'il est autorisé de dépasser mais uniquement pendant un délai de grâce (souvent sept jours) au terme duquel toute consommation supplémentaire devient impossible.

"hard limit"

Il s'agit d'une limite toujours infranchissable.

Les conséquences concrètes, pour l'utilisateur qui atteint sa limite de quotas, sont complètement liées aux commandes ou logiciels utilisés pour lesquels le disque apparaît comme plein. Cela peut se traduire bien évidemment par une perte de données due à l'impossibilité de sauvegarder un fichier. Il est donc important de bien communiquer avec les utilisateurs dans le cas d'activation de ce service.

Comme d'habitude, la mise en oeuvre varie, dans les détails, selon les versions mais elle consiste toujours à enchaîner les étapes suivantes :

1) Modification du fichier des filesystems

L'utilisation des quotas doit être demandée dans ce fichier pour chacun des filesystems concernés :

Solaris	option "rq" ou "rw,quota" dans "/etc/vfstab"
HP-UX	option "quota" dans "/etc/fstab"
Linux	option "usrquota,grpquota" dans "/etc/fstab"
AIX	une ligne "quota = usrquota,groupquota" dans la strophe concernée

S'il s'agit de la première mise en oeuvre, le filesystem devra être démonté puis remonté avec cette nouvelle option.

2) Création d'un fichier "quotas"

Ce fichier doit être créé à la racine du filesystem (c'est à dire dans le répertoire de montage) lors de la première mise en oeuvre. Il doit appartenir au compte "root" et il est conseillé de lui attribuer la valeur "600" comme masque de permissions.

Sous Linux, il faut créer deux fichiers "quota.user" et "quota.group".

3) Choix des valeurs attribuées à chaque utilisateur ou groupe

La commande **edquota** reçoit en argument un nom d'utilisateur (ou de groupe pour les versions qui implémente cet aspect) et édite un fichier temporaire (sous "vi" par défaut) pour permettre de positionner les limites "soft" et "hard" pour les blocs disques et les inodes respectivement. Une option "-p" permet ensuite d'associer plus facilement les mêmes limitations à plusieurs utilisateurs ou groupes.

4) Modification éventuelle du délai de grâce (dépassement de la "soft limit")

La même commande **edquota** (via l'option "-t") permet de changer ce délai qui, par défaut, est souvent de sept jours.

5) Activation du service

La commande **quotaon** permet d'activer les quotas sur un filesystem particulier ou, via l'option "-a", sur tous ceux qui ont les bonnes caractéristiques dans le fichier des filesystems. Il convient de s'assurer que cette commande sera activée automatiquement lors du prochain redémarrage du système. La commande symétrique **quotaoff** permet la désactivation du service.

6) Vérification de cohérence

La commande **quotacheck** vérifie la cohérence des informations du fichier "quotas" par rapport à la consommation effective des ressources. Tout comme la commande "quotaon", elle peut être activée sur un filesystem particulier ou sur tous (via la même option "-a"). Il faudra également s'assurer de son activation au démarrage du système.

Par la suite, les commandes **repquota** et **quota** permettent de suivre le détail des consommations des utilisateurs.

La première commande affiche les consommations de tous les utilisateurs du filesystem donné en argument.

La seconde est destinée aux utilisateurs ordinaires pour leur permettre de visualiser leur propre consommation. On peut conseiller de l'intégrer aux fichiers d'initialisation du "shell" afin d'être sensibilisé à ces limitations à chaque connexion.

Exemple de mise en oeuvre sur un filesystem "ufs" Solaris

On décide d'activer les quotas sur le filesystem "/home" (option de montage "rq")

```
# grep  /home  /etc/vfstab
/dev/dsk/c0d0s5 /dev/rdsk/c0d0s5   /home   ufs     2   yes   rq
#
```

On crée dans le répertoire de montage un fichier "quotas", de taille 0, bien protégé.

```
# cd  /home
# >  quotas
# chmod  600  quotas
#
```

On positionne ensuite des quotas pour l'utilisateur "michel". Il pourra consommer jusqu'à 8000 blocs (environ 4 Mo) avec une tolérance de dépassement pendant 7 jours. En aucun cas, il ne pourra consommer plus de 10000 blocs (environ 5 Mo).

La commande "edquota" consiste à éditer (sous "vi") un fichier temporaire. Les valeurs "0" associées aux inodes montrent que les quotas ne concerneront que l'espace disque et non pas le nombre de fichiers.

```
# edquota  michel
fs /home blocks (soft = 8000, hard = 10000) inodes (soft = 0, hard = 0)
...................
#
```

On active le service (enchaînement des commandes "quotacheck" et "quotaon").

```
# quotaon  /home
# quotacheck  /home
#
```

On visualise les consommations actuelles des utilisateurs dans ce filesystem. Dans cet exemple, l'utilisateur "michel" est la seule victime des quotas.

```
# repquota  -v  /home
/dev/dsk/c0d0s5 (/home):
                  Block limits                File limits
User        used  soft hard   timeleft  used  soft  hard   timeleft
michel  --  7902  8000 100000           5     0     0
#
```

Connexion sous le compte "michel" et visualisation des consommations

```
$ id
uid=2001(michel) gid=2000(stage)
$ pwd
/home/michel
$ quota  -v
Disk quotas for michel (uid 2001):
Filesystem usage quota  limit timeleft files quota limit timeleft
/home      7902  8000   10000          5     0     0
$
```

Quelques temps après, lors du dépassement de la "soft limit", un message d'avertissement est affiché mais la consommation d'espace disque n'a pas été refusée. La commande "quota -v" permet de voir que le délai de grâce de sept jours a cependant commencé.

```
$ cp  fichier  autrefichier
quota_ufs:
Warning: over disk limit (pid 925, uid 2001, inum 49549, fs /home)
$ quota  -v
Disk quotas for michel (uid 2001):
Filesystem usage quota  limit timeleft files quota limit timeleft
/home      9030  8000   10000 7.0 days 6     0     0
$
```

Par contre, lors d'une tentative de dépassement de la "hard limit", un message d'erreur indique que cette consommation d'espace disque est refusée. La commande concernée échoue complètement.

```
$ cp  fichier  fichierbis
cp: fichierbis: Disc quota exceeded
$
```

4.8 Sauvegardes et restaurations

De nombreuses commandes de sauvegarde sont proposées aux administrateurs Unix. Certaines **commandes "standards"** ("tar", cpio", "dd") sont présentes sur toutes les versions mais d'autres **commandes spécifiques** sont souvent disponibles. Après les avoir identifiées et maîtrisées, il est possible, à partir de ces commandes, d'écrire des "scripts" adaptés à sa propre stratégie de sauvegarde. De plus, le service "cron" de planification de travaux (voir chapitre 8) permettra de lancer automatiquement ces utilitaires personnalisés.

Le choix de matériels spécialisés ainsi que l'hétérogénéité de nombreux sites de production font assez souvent choisir une solution commerciale plutôt que l'utilisation directe des commandes du système.

Le site Internet *"http://www.storagemountain.com"* constitue, par exemple, une excellente source d'informations pour toutes ces solutions de sauvegarde et de stockage, parmi lesquelles on peut notamment trouver certains logiciels libres.

Donnons quelques principes importants qui sont mis en jeu dans la quasi-totalité des commandes Unix :

Une commande de sauvegarde fabrique **un fichier résultat (archive)** qui peut être écrit, soit sur un fichier spécial (lecteur de disquette ou de bande), soit sur un fichier ordinaire (avec création ou écrasement), soit encore sur la sortie standard (pour une collaboration, via un "pipeline", avec une autre commande).

Lors d'une sauvegarde, il faut souvent être attentif aux éventuels liens symboliques mis en jeu dans l'arborescence concernée. Il est nécessaire de ne pas oublier d'éventuels fichiers pointés ou inversement de recenser d'éventuels liens symboliques qui pointeraient vers l'arborescence à sauvegarder.

Une restauration consiste à recréer les fichiers avec le nom qu'ils ont sur l'archive. Ceci incite à **préférer les sauvegardes en noms relatifs** pour permettre une restauration à n'importe quel endroit de l'arborescence. En contrepartie, il faut s'assurer que les filesystems sont bien montés et que l'on se trouve bien au bon endroit avant de lancer une restauration effective.

Lors d'une restauration, il faut maîtriser le comportement de la commande utilisée. Sera-t-on satisfait du comportement par défaut ou devra-t-on recourir à certaines options ? On peut se demander, par exemple, si un fichier disque de même nom que sur l'archive sera écrasé ou pas, si la date d'origine du fichier sera conservée ou si c'est la date système qui sera utilisée etc...

Dans le cas de récupération d'une archive provenant d'un autre système, il faudra être attentif aux propriétaires et groupes des fichiers de cette archive. Si ces "uids" ou "gids" ne correspondent pas à ceux du système cible, il conviendra de les changer de façon adéquate via les commandes "chgrp" et "chown".

4.8.1 La commande tar

```
tar  clés  [ fichier | -C rep ]...
```

clés Il s'agit d'un ensemble d'options précédées ou non du signe "-"

Quelques options (les trois premières sont exclusives)

c	Sauvegarde
t	Liste de l'archive
x	Restauration
f arch	"arch" est le nom d'une archive qui peut être : un fichier spécial ou un fichier ordinaire ou le signe "-" (entrée ou sortie standard)
h	Suivre les liens symboliques
m	Restauration avec la date courante (date d'origine par défaut)
v	Mode bavard

fichier... Fichier(s) concerné(s)
Dans le cas de répertoires, la commande traite la sous-arborescence.

-C rep... Ceci permet un positionnement préalable dans des répertoires pour effectuer un traitement sur les noms relatifs en divers endroits de l'arborescence

*Les versions Linux proposent une version améliorée de la commande (**GNU tar**).*

Quelques exemples de syntaxe

Sauvegarde du répertoire courant vers la bande "/dev/rmt0"

```
tar  cvf /dev/rmt0  .
```

Sauvegarde d'une liste de fichiers préparée dans le fichier "liste"

```
tar  cvf /dev/rmt0  `cat liste`
```

Liste des fichiers de l'archive située sur la bande "/dev/rmt0"

```
tar  tvf /dev/rmt0
```

Restauration de toute l'archive (être au bon endroit pour cette opération)

```
tar  xvf /dev/rmt0
```

Restauration du seul fichier "toto" (la notation "./toto" est nécessaire pour respecter exactement le nom du fichier sur l'archive)

```
tar  xvf /dev/rmt0  ./toto
```

Recopie fidèle d'arborescence

```
cd  /rep1  ;  tar  cf -  .  |  ( cd  /rep2  ;  tar  xf - )
```

4.8.2 La commande cpio

```
cpio  -o [ options ]  >  sortie
cpio  -i [ options ]  <  entrée
cpio  -p [ options ]  répertoire
```

La syntaxe de la commande "cpio" paraît plus compliquée au premier abord. En effet, la commande ne reçoit pas d'arguments classiques car elle lit sur son entrée standard les noms des fichiers à sauvegarder. Elle fabrique alors une archive qu'elle écrit sur sa sortie standard. Un appel préliminaire à une commande qui génère la liste est donc nécessaire dans le cadre d'un "pipeline" qui se terminera par une redirection vers le fichier résultat (fichier spécial ou ordinaire). Les options sont nombreuses et peuvent varier selon les versions. Certaines options essentielles apparaissent dans les exemples ci-dessous.

La commande "find" est la plus adaptée pour générer la liste des fichiers à sauvegarder. La commande "cpio" réalise donc facilement des sauvegardes sélectives puisqu'elle profite, à ce niveau, des nombreuses possibilités de la commande "find".

Quelques exemples de syntaxe

Sauvegarde du répertoire courant vers la bande "/dev/rmt0"

L'option "-o" indique qu'il s'agit de la fabrication d'une archive sur la sortie standard. L'option complémentaire "-v" correspond simplement au mode bavard.

```
find . -print | cpio -ov > /dev/rmt0
```

Liste des fichiers de l'archive située sur la bande "/dev/rmt0"

L'option "-i" précise à la commande que l'archive est à lire sur l'entrée standard.

```
cpio -itv < /dev/rmt0
```

Restauration de toute l'archive (être au bon endroit pour cette opération)

L'absence de l'option "-t" indique qu'il s'agit d'une restauration effective et non pas seulement d'une liste. L'option "-m" est nécessaire pour restaurer la date d'origine des fichiers (par défaut, la date courante est utilisée). L'option "-u" permet l'écrasement des fichiers disque de même nom que sur l'archive.

```
cpio -ivmu < /dev/rmt0
```

Recopie d'arborescence

```
cd /rep1 ; find . -print | cpio -pmdv /rep2
```

L'option "-d" permet de recréer les sous-répertoires.

4.8.3 La commande dd

```
dd  [ option=valeur ]...
```

La commande "dd" effectue une copie physique d'un fichier d'entrée sur un fichier de sortie en effectuant une éventuelle conversion. Les fichiers d'entrée et de sortie sont la plupart du temps des fichiers spéciaux. Ainsi, cette commande constitue un utilitaire intéressant pour dupliquer des disquettes ou des bandes, recopier ou sauvegarder des partitions, convertir des archives etc...

Quelques options

if=fic	Fichier d'entrée (entrée standard par défaut)
of=fic	Fichier de sortie (sortie standard par défaut)
ibs=n	Taille des blocs en entrée
obs=n	Taille des blocs en sortie
bs=n	Taille des blocs en entrée et en sortie
count=n	Nombre de blocs à traiter
conv=type	Type de conversion

Quelques types de conversion

ascii	code "ebcdic" vers code "ascii"
ebcdic	code "ascii" vers code "ebcdic"
lcase	transformation des majuscules en minuscules
ucase	transformation des minuscules en majuscules
swab	permutation des paires d'octets

(parfois utile lors de transferts de bandes entre matériels différents)

Quelques exemples

Vérification de l'intégrité physique d'une disquette

On réussit la lecture complète de la disquette vers le périphérique virtuel "/dev/null". Le nombre d'enregistrements lus est identique au nombre d'enregistrements écrits et correspond bien à la taille du support. On peut donc considérer que la disquette est physiquement correcte.

```
# dd  if=/dev/fd0  of=/dev/null
2880+0 records in
2880+0 records out
#
```

Fabrication d'une "bande tar" tout en permutant les paires d'octets en vue d'une relecture vers un matériel cible différent

```
tar  cf -  .  |  dd  of=/dev/rmt0  conv=swab
```

4.8.4 Sauvegardes de filesystem

La plupart des versions proposent des commandes pour effectuer la sauvegarde d'un **filesystem éventuellement démonté**. Cette possibilité est intéressante en assurant la stabilité des fichiers au moment de leur sauvegarde.

De plus, ces commandes permettent une gestion de **sauvegardes incrémentales** via la notion de niveau de sauvegarde. Par convention, le niveau 0 représente une sauvegarde complète. Par la suite, on peut planifier des sauvegardes de niveau 1 à 9. Dans ce cas, seuls les fichiers modifiés depuis la sauvegarde de niveau immédiatement inférieur sont pris en compte.

Les commandes sont spécifiques mais elles sont très proches en terme de fonctionnement et de syntaxe.

Contrairement aux commandes "tar" ou "cpio", il y a deux commandes symétriques pour respectivement sauvegarder et restaurer. On peut citer par exemple :

AIX	**backup, restore** (gestion de filesystems "jfs")
Solaris	**ufsdump, ufsrestore** (gestion de filesystems "ufs")
HP-UX	**fbackup, frecover** (gestion de filesystems)
	vxdump, vxrestore (gestion de filesystems "vxfs")
Linux	**dump, restore** (gestion de filesystems "ext2" ou "ext3")

Exemple Solaris : ufsdump et ufsrestore

```
ufsdump  [ options ]  [-f device]  filesystem
```

Options

niveau		Niveau de sauvegarde (0 à 9 , 9 par défaut)
	0	Sauvegarde complète
	> 0	Sauvegarde incrémentale
-u		La date et le niveau de sauvegarde sont notés dans le fichier "/etc/dumpdates" pour permettre la gestion ultérieure des sauvegardes incrémentales
-c		Sauvegarde sur cartouche
-f device		Choix du périphérique de sauvegarde (cela peut être un fichier disque)
filesystem		Nom de la partition ou du répertoire de montage

Sauvegarde du filesystem "/home" vers la bande "/dev/rmt/0" après l'avoir démonté

```
# umount  /home
```

```
# ufsdump  -0uf /dev/rmt/0  /home
.....................................
  DUMP: Date of this level 0 dump: mar 01 avr 2003 18:42:35 WEST
  DUMP: Dumping /dev/rdsk/c0d0s5 (solar8:/home) to /dev/rmt/0.
.....................................
  DUMP: DUMP IS DONE
  DUMP: Level 0 dump on mar 01 avr 2003 18:48:20 WEST
#
```

L'option "-u" a permis de noter la date et le niveau de sauvegarde dans le fichier "/etc/dumpdates" (ceci permettra de gérer des sauvegardes incrémentales).

```
# cat  /etc/dumpdates
  DUMP: Level 0 dump on mar 01 avr 2003 18:48:20 WEST
#
```

ufsrestore [options] [-f device] [fichiers]

Options

	-t	Liste du contenu de l'archive
	-r	Restauration complète du filesystem
	-x	Restauration de certains fichiers seulement
	-v	Mode bavard
-f device		Choix du nom de l'archive
fichiers		Nom des fichiers à restaurer (restauration partielle avec l'option "-x")

Liste du contenu de l'archive située sur la bande "/dev/rmt/0"

On constate que les fichiers sont sauvés en relatif. La commande affiche aussi le numéro d'inode et indique si le fichier est ou non un répertoire ("dir" ou "leaf").

```
# ufsrestore  -tvf /dev/rmt/0
.....................................
Dump   date: Tue Apr 01 18:48:20 2003
Level 0 dump of /home on solar8:/dev/dsk/c0d0s5
.....................................
dir          2  .
dir          3  ./lost+found
dir       5504  ./stage1
leaf      5505  ./stage1/.profile
.....................................
```

Restauration complète du filesystem

S'il s'agit d'une récupération de données suite à un incident important, la manipulation de restauration a pu être précédée d'une recréation du filesystem. Dans tous les cas, il faut veiller à monter celui-ci et à se positionner dans le répertoire de montage puisque les fichiers ont été sauvegardés avec un chemin relatif et seront donc recréés dans le répertoire courant.

```
# mount  /home
# cd  /home
# ufsrestore  -rvf /dev/rmt/0
.....................................
Dump   date: Tue Apr 01 18:48:20 2003
Level 0 dump of /home on solar8:/dev/dsk/c0d0s5
.....................................
Begin level 0 restore
Initialize symbol table.
Extract directories from tape
Calculate extraction list.
Extract new leaves.
Check pointing the restore
extract file ./stage1/.profile
.....................................
.....................................
Add links
Set directory mode, owner, and times.
Check the symbol table.
Check pointing the restore
#
```

A l'issue de la restauration, un fichier "restoresymtable" est présent sur le disque. Ce fichier est nécessaire entre chaque restauration incrémentale pour en assurer la cohérence. Dans le cas d'une restauration complète, il peut être supprimé.

```
# ls  -l  restoresymtable
-rw-------  1  root  other  415292  avr 1 18:58  restoresymtable
# rm  restoresymtable
#
```

4.8.5 Autres commandes

En complément des commandes standards et des commandes spécifiques de sauvegardes de filesystem, certaines versions proposent encore d'autres possibilités.

AIX

Les commandes **backup** et **restore** ont deux formes syntaxiques. La sauvegarde de filesystem est baptisée "sauvegarde par inodes". Une sauvegarde classique de fichiers est également possible et est baptisée alors "sauvegarde par noms". L'utilisation de ces commandes est donc fortement recommandée pour effectuer des sauvegardes AIX.

AIX propose également les commandes **savevg** et **restvg** pour effectuer une sauvegarde complète de groupe de volumes. Tous les filesystems montés au moment de la sauvegarde sont pris en compte dans cette archive globale.

Enfin, AIX implémente la commande **mksysb** qui permet de créer, sur une bande magnétique, une sauvegarde du groupe de volumes "rootvg". Cette bande pourra être utilisée pour redémarrer le système afin de résoudre un problème mais surtout de réinstaller rapidement celui-ci en ayant sauvegardé la configuration. Il est possible également d'installer un autre système avec des caractéristiques similaires (opération de clonage). Une commande plus récente **mkcd** effectue sur disque des images "mksysb" qui peuvent ensuite être gravées sur CD ou DVD.

HP-UX

Dans le cadre du produit "Ignite-UX", l'utilitaire **make_recovery** permet d'effectuer, sur bande magnétique, une sauvegarde de récupération du système.

Sauvegardes compressées

Sauf exception, les commandes de sauvegarde n'ont pas d'options de compression car elles peuvent collaborer facilement dans un "pipeline" avec les commandes dédiées.

Pour mémoire, les commandes de compression de fichiers sont, dans un ordre croissant de performance :

pack et **unpack** (fichiers ".z")
compress et **uncompress** (fichiers ".Z")
gzip et **gunzip** (fichiers ".gz")

Exemples

Quand les commandes de compression n'ont pas d'arguments, elles traitent tout naturellement leur entrée standard et envoient leur résultat sur la sortie standard.

Sauvegarde du répertoire courant en format "tar compressé"

Rappelons que le signe "-" comme nom d'archive signifie ici "sortie standard", ce qui permet d'envoyer le résultat dans le "pipeline" pour être traité par la commande "compress". Si la sauvegarde a lieu vers un fichier disque, une bonne habitude consiste à le nommer avec une pseudo-extension qui renseigne immédiatement sur la nature de l'archive (".tar.Z" fait immédiatement penser à un "tar compressé").

```
tar cf - . | compress > /tmp/monarchive.tar.Z
```

Récupération de l'archive

Il est important de rediriger l'entrée standard de la commande "uncompress" pour qu'elle ne tente pas de créer un fichier mais envoie bien son résultat sur la sortie standard et, par conséquent, dans le "pipeline". Le signe "-" comme nom d'archive signifie ici "entrée standard".

```
uncompress < /tmp/monarchive.tar.Z | tar xvf -
```

Sauvegarde du répertoire courant en format "cpio compressé"

```
find . -print | cpio -o | compress > /tmp/monarchive.cpio.Z
```

Récupération de l'archive

```
uncompress < /tmp/monarchive.cpio.Z | cpio -ivu
```

4.9 Mémoire virtuelle

Un système Unix doit disposer d'au mois un **espace de pagination** (appelé aussi "**zone de swap**") pour réaliser les inévitables échanges de données avec la mémoire physique. Cet espace de pagination consiste en une partition ou un volume logique d'un type particulier créé lors de la procédure d'installation. D'autres espaces complémentaires pourront être créés ensuite, si nécessaire. S'il s'agit d'un volume logique, l'espace existant pourra être agrandi facilement. Enfin, certaines versions permettent d'utiliser des fichiers ordinaires comme espaces de pagination complémentaires. Ceci peut constituer une solution de dépannage lors d'une montée en charge exceptionnelle.

Le choix judicieux de la taille de cet espace n'est pas très évident car chaque situation est particulière et évidemment liée à la véritable utilisation du système (nombre d'utilisateurs simultanés, types de logiciels, interfaces graphiques...). Une habitude très répandue consiste à choisir une taille égale à deux fois celle de la mémoire physique. Ce choix par défaut peut constituer un compromis acceptable. Cependant, il sera peut-être surdimensionné pour une station de travail individuelle ou nettement insuffisant pour un serveur fortement sollicité. Lorsque l'espace de pagination est trop petit, cela se traduit par des messages d'erreur très explicites à la console du système, accompagnés sans doute d'une nette dégradation des temps de réponse, voire d'un "crash" complet. L'ajustement nécessaire devra alors être pratiqué jusqu'à parvenir à une situation correcte de production stable.

4.9.1 Visualisation

Les commandes de visualisation sont spécifiques à chaque version :

Solaris	**swap**
HP-UX	**swapinfo**
AIX	**lsps**
Linux	**free** ou **swapon**

Exemple Solaris

```
# swap  -l
swapfile             dev    swaplo    blocks    free
/dev/dsk/c0d0s1     102,1        8   1049320  1049320
# swap  -s
total: 15768k bytes allocated + 5572k reserved = 21340k used,
587544k available
#
```

Exemple HP-UX

```
# swapinfo  -atm
            Mb     Mb     Mb   PCT  START/      Mb
TYPE     AVAIL  USED  FREE  USED  LIMIT  RESERVE  PRI  NAME
dev        288     0   288    0%      0        -    1  /dev/vg00/lvol2
reserve      -    59   -59
memory     103    39    64   38%
total      391    98   293   25%      -        0    -
#
```

Exemple AIX

```
# lsps  -a
Page Space   Phys.Volume  Volume Group  Size  %Used  Active  Auto
hd6          hdisk0       rootvg        256   64     yes     yes
paging00     hdisk1       stage           128   13     yes     yes
#
```

Exemple Linux

```
# free  -mo
       total       used       free     shared    buffers     cached
Mem:      29         26          2          0          2         13
Swap:     300         0        300
# swapon  -s
Filename       Type          Size      Used    Priority
/dev/hda6      partition     307400    0       -1
#
```

4.9.2 Activation

L'activation des espaces de pagination est quelquefois assurée par la présence d'entrées particulières dans le fichier des filesystems.

Une commande spécifique peut aussi remplir ce rôle.

Sous HP-UX et Linux, les "zones de swap" sont décrites par des entrées dans "/etc/fstab" :

```
/dev/hda6      swap      swap      defaults      0      0
```

Sous Solaris, le fichier "/etc/vfstab" est utilisé :

```
/dev/dsk/c0d0s1    -    -    swap    -    no    -
```

Sous AIX, les "zones de swap" sont décrites dans le fichier "/etc/swapspaces".

HP-UX, Linux et AIX utilisent une commande **swapon** pour activer les espaces de pagination lors du démarrage du système alors que Solaris utilise un "script shell" **swapadd** pour effectuer le même traitement. Ces commandes peuvent aussi être utilisées de façon interactive (commande "**swap -a**" pour Solaris).

HP-UX et Linux permettent d'associer une priorité d'utilisation entre plusieurs espaces de pagination.

Une commande symétrique de désactivation **swapoff** est disponible sous Linux et AIX (à partir de la version 5). Solaris utilise la forme "swap -d".

La suppression d'un espace de pagination peut consister à supprimer son entrée dans le fichier des filesystems (la suppression sera ainsi effective au prochain redémarrage) ou être proposée via une commande (**rmps** sous AIX).

La création d'un espace de pagination complémentaire est également très spécifique à la version. Certaines d'entre elles permettent l'utilisation d'un fichier ordinaire.

Exemple Solaris

La commande "mkfile" crée un espace de 30 Mo via le fichier "/monswap" et la commande "swap -a" permet de l'activer immédiatement.

```
# mkfile  30m  /monswap
# swap  -a  /monswap
# swap -l
swapfile             dev    swaplo    blocks    free
/dev/dsk/c0d0s1     102,1        8   1049320  1049320
/monswap                -        8     61432    61432
#
```

Exemple Linux

La commande "dd" préliminaire sert à créer un fichier "/monswap" de 30 Mo. L'utilisation du pseudo-périphérique "/dev/zero" est une méthode Unix classique pour cette opération. La commande "mkswap" crée ensuite un espace via ce fichier et la commande "swapon" permet de l'activer immédiatement.

```
# dd  if=/dev/zero  of=/monswap  bs=1024  count=30720
30720+0 records in
30720+0 records out
# mkswap  /monswap
Setting up swapspace version 1, size = 30712K
# chmod  600  /monswap
# swapon  -v  /monswap
swapon on /monswap
# swapon  -s
Filename        Type          Size      Used    Priority
/dev/hda6       partition     307400    0       -1
/monswap        file          30712     0       -4
#
```

Exemple AIX

Les espaces de pagination sont des volumes logiques d'un type particulier (paging). L'espace initial porte le nom "/dev/hd6" et les éventuels espaces complémentaires seront baptisés "/dev/pagingxx" ("xx" démarre à la valeur "00").

La commande "mkps" permet de créer un espace complémentaire.

L'option "-a" demande l'activation au démarrage du système. L'option "-n" provoque l'activation immédiate. L'option "-s" permet d'indiquer la taille en nombre de "LPs". Enfin, l'argument "rootvg" désigne, bien sur, le nom du groupe de volumes concerné.

```
# mkps  -a  -n  -s 8  rootvg
#
```

La commande "chps" permet d'agrandir un espace existant. L'option "-s" indique alors le nombre de "LPs" supplémentaires.

```
# chps  -s 4  paging00
#
```

4.10 Compléments sur les groupes de volumes

Un groupe de volumes est, comme on l'a vu précédemment dans ce chapitre, une entité obligatoire dans les organisations en volumes logiques. Le groupe de volumes contient à la fois des disques physiques et des volumes logiques installés sur ces disques. Cette notion pourrait apparaître plutôt comme une contrainte mais elle apporte, en fait, certaines fonctionnalités supplémentaires intéressantes.

Tout d'abord, comme on l'a déjà évoqué rapidement, les versions AIX proposent des commandes de sauvegarde globale d'un groupe de volumes. On peut ainsi bénéficier d'une approche simplifiée dans la stratégie de sauvegarde.

Deuxièmement, les groupes de volumes peuvent être activés ou désactivés. A l'image des filesystems qui sont disponibles ou non selon qu'ils sont montés ou démontés, on bénéficie également d'une gestion simple pour la mise à disposition ou non des disques.

Enfin, l'intérêt majeur de la notion de groupe de volumes réside sans doute dans la fonctionnalité dite d'**export-import**. L'objectif de cette opération est de pouvoir déplacer un ensemble de disques d'une machine vers une autre en conservant tous les filesystems présents sur ces disques.

Une bonne organisation consiste à séparer le système d'exploitation (correspondant au groupe de volumes créé par l'installation) des autres filesystems (données, répertoires des utilisateurs, logiciels...). L'utilisation de plusieurs disques, judicieusement répartis dans plusieurs groupes de volumes, permet d'envisager facilement la fonction d'export-import. Il est ainsi fréquent de trouver des groupes de volumes mono-disque car l'export-import ne peut pas s'effectuer au niveau du disque seul. Un groupe de volumes ne contiendra donc plusieurs disques que si, bien entendu, la taille du seul disque est insuffisante ou bien si on souhaite activer les possibilités de "striping" ou de "mirroring".

Exemple d'export-import sur HP-UX

L'enchaînement des opérations doit être le suivant :

1) Désactivation de tous les volumes logiques (démontage des filesystems)

2) Désactivation du groupe de volumes (**vgchange**)

3) Exportation du groupe de volumes (**vgexport**)

4) Transfert des disques sur la machine cible

5) Importation du groupe de volumes (**vgimport**)

6) Activation du groupe de volumes (**vgchange**)

7) Montage des filesystems

Désactiver le groupe de volumes (on a démonté tous les filesystems au préalable)

```
# vgchange  -a n  /dev/stage
Volume group "/dev/stage" has been successfully changed.
```

Exporter le groupe de volumes

```
# vgexport  -v  /dev/stage
Volume group "/dev/stage" has been successfully removed.
```

Le groupe de volumes a bien été supprimé du système

```
# vgdisplay  stage
vgdisplay: Volume group "/dev/stage" does not exist in the
"/etc/lvmtab" file
vgdisplay: Cannot display volume group "stage".
#
```

Avant l'importation, il faut procéder comme pour une création par "vgcreate", on crée le répertoire correspondant au groupe de volumes et le fichier spécial "group".

```
# mkdir  /dev/stage
# mknod  /dev/stage/group  c  64  0x010000
```

Importer le groupe de volumes (choix du nom et du disque associé)

```
# vgimport  -v /dev/stage  /dev/dsk/c0t5d0
Beginning the import process on Volume Group "/dev/stage".
Logical volume "/dev/stage/lvol1" has been successfully created
with lv number 1.
Volume group "/dev/stage" has been successfully created.
Warning: A backup of this volume group may not exist on this machine.
Please remember to take a backup using the vgcfgbackup command after
activating the volume group.
```

Activation du groupe de volumes

```
# vgchange  -a y  /dev/stage
Activated volume group
Volume group "/dev/stage" has been successfully changed.
```

On peut monter directement les filesystems qui viennent d'être importés. Il faudra également compléter le fichier des filesystems "/etc/fstab".

```
# mount  /dev/stage/lvol1  /home2
#
```

La mise en oeuvre **AIX** est très similaire avec les commandes **varyonvg**, **varyoffvg**, **exportvg** et **importvg**.

5. Services d'impression

5.1 Considérations générales

Le système Unix est doté d'un service d'impression permettant l'accès à des **imprimantes locales** (sur port série ou parallèle), des **imprimantes distantes** (connectées à une autre machine jouant le rôle de serveur), des **imprimantes réseau** (imprimantes directement connectées au réseau).

Les deux derniers cas correspondent à une configuration TCP/IP baptisée "lpr/lpd". Le terme "lpr" désigne l'aspect client ("lp request") tandis que le terme "lpd" désigne l'aspect serveur ("lp daemon").

Pour l'utilisateur, une imprimante est connue par un nom logique attribué par l'administrateur. Une imprimante par défaut (générale ou individuelle via une variable d'environnement) sera prévue pour les requêtes d'impression effectuées sans option de destination.

Trois services d'impression coexistent dans le monde Unix :
- le spouleur "System V" (démon **lpsched**) utilisé sur Solaris et HP-UX
- le spouleur "Berkeley" (démon **lpd**) utilisé sur Linux
- le spouleur AIX (démon **qdaemon**)

Les utilisateurs disposent d'au moins trois commandes :

	System V	Berkeley	AIX
Requêtes d'impression	**lp**	**lpr**	**qprt**
État des files d'attente	**lpstat**	**lpq**	**qchk**
Annulation de requêtes	**cancel**	**lprm**	**qcan**

Les versions AIX émulent les commandes des autres spouleurs.

Dans le spouleur "Berkeley", le fichier à imprimer est recopié dans un répertoire de travail. Il sera donc imprimé dans l'état où il était au moment de la requête. Dans les deux autres services, cette recopie n'a pas lieu et le fichier sera imprimé dans l'état où il se trouve au moment de la sortie physique sur l'imprimante. Les commandes sont dotées d'une option pour modifier ce comportement.

5.2. Service System V (Solaris - HP-UX)

5.2.1 Commandes de l'utilisateur

Les trois commandes de base de l'utilisateur du service sont :

lp	Requête d'impression
lpstat	État du service
cancel	Annulation d'une requête

Exemples

```
$ lp  .profile  toto      Requête vers l'imprimante par défaut
request id is impr-242 (2 files)
$ lp  -n2  /etc/passwd    Deux exemplaires
request id is impr-243 (1 file)
$ lp  -d impr2  main.c    Requête vers l'imprimante "impr2"
request id is impr2-244 (1 file)
$ lpstat                  Liste de ses propres requêtes
impr-242               stage1             727   Jan  3 09:52
impr-243               stage1            2394   Jan  3 09:52
$ lpstat  -o              Liste de toutes les requêtes
impr-242               stage1             727   Jan  3 09:52
impr-243               stage1            2394   Jan  3 09:52
impr2-244              stage2         2052580   Jan  3 09:56
$ cancel  impr2-244       Annulation d'une requête
Request "impr2-244" canceled.
$ cancel  -u stage1       Annulation des requêtes d'un utilisateur
Request "impr-242" canceled.
Request "impr-243" canceled.
$
```

5.2.2 Ajout d'une imprimante locale

La configuration des imprimantes met en jeu la commande **lpadmin** ainsi que les couples de commandes : **accept** - **reject** et **enable** - **disable**.

*HP-UX nécessite l'arrêt préalable du service via la commande **lpshut**.*

Avant d'utiliser les commandes, il faut préciser la **terminologie** utilisée :

imprimante ("printer")	Nom logique d'imprimante
"device"	Nom du périphérique local (port série ou parallèle)

classe	Ensemble ordonné d'imprimantes
destination	Imprimante ou classe

Si les utilisateurs envoient une requête vers une classe, cette requête s'exécutera sur la première imprimante disponible. Ceci peut être intéressant dans le cas où l'on disposerait de plusieurs imprimantes de même type, situées dans le même local.

modèle	Fichier prototype à associer à une imprimante

Il peut s'agir de scripts (pour des imprimantes génériques) ou de programmes associés à des imprimantes officiellement supportées. Les modèles sont situés dans le répertoire "/usr/lib/lp/model" sous Solaris ou "/etc/lp/model" sous HP-UX.

interface	Modèle associé à une imprimante

Quand un modèle est associé à une imprimante, il prend le nom d'interface.

La syntaxe générale de la commande de configuration est la suivante :

```
          lpadmin   -p printer  [ options de configuration ]
ou bien
          lpadmin   -x destination
ou encore
          lpadmin   -d [ destination ]
```

-p printer	Création ou configuration avec les options complémentaires associées
-x dest	Suppression d'une destination (imprimante ou classe)
-d dest	Choix de la destination par défaut (variable "LPDEST" pour avoir des destinations par défaut individuelles)
-d	Suppression de la destination par défaut

Principales options de configuration

Ces options s'utilisent en complément de l'option préliminaire "-p printer".

-v device	Association du périphérique physique (port série ou parallèle)
-c class	Insertion dans une classe (création éventuelle de cette classe)
-r class	Retrait d'imprimante d'une classe (suppression de la classe si celle-ci devient vide)
-m model	Choix d'un modèle comme interface
-e printer	Reprise de l'interface d'une autre imprimante

Commandes complémentaires de gestion

```
accept   destinations
reject   [ -r raison ]   destinations
```

Ces deux commandes permettent d'accepter ou de refuser les requêtes. Lors de la déclaration d'une nouvelle imprimante, un premier appel à la commande "accept" sera nécessaire. Par la suite, en cas de problème prolongé, la commande "reject" permettra d'éviter l'accumulation de requêtes vers une imprimante. L'option "-r" est utile pour associer un texte qui expliquera aux utilisateurs la cause du rejet.

```
enable   imprimantes
disable   [ options ]   imprimantes
     -r raison    Message explicatif
     -W           Attente de la fin de la sortie en cours
     -c           Désactivation immédiate
```

Ces deux commandes permettent d'activer ou de désactiver la sortie physique sur les imprimantes. Lors de la déclaration d'une nouvelle imprimante, un premier appel à la commande "enable" sera nécessaire. Par la suite, en cas de problème ponctuel, la commande "disable" permettra d'éviter des erreurs d'accès physiques. Quelquefois, le spouleur décide lui-même d'une désactivation en cas de problème de communication avec le port concerné (imprimante éteinte ou "hors ligne"). La commande "enable" est accessible aux utilisateurs ordinaires qui peuvent ainsi réactiver une imprimante sans recourir à l'administrateur.

```
          lpmove   liste_de_requêtes   nouvelle_destination
ou        lpmove   destination_1   destination_2
```

Cette commande permet de rediriger certaines requêtes (ou l'ensemble des requêtes d'une destination donnée) vers une autre destination. Ceci peut être utile, en cas de défaillance d'une imprimante, pour ne pas abandonner des travaux en attente.

Exemple de configuration

Sous Solaris, il n'est pas nécessaire d'arrêter le service et, de plus, les options et leurs arguments peuvent être séparés par un espace. HP-UX nécessite l'arrêt du service et n'accepte pas d'espace entre les options et leurs arguments.

```
# lpstat  -t
scheduler is running
no system default destination
#
```

```
# lpadmin  -p impr  -v /dev/lp  -m standard
# accept  impr
UX:accept: INFO: destination "impr" now accepting requests
# enable  impr
UX:enable: INFO: printer "impr" now enabled
#
```

Les trois commandes ci-dessus sont nécessaires pour la définition d'une nouvelle imprimante. Dans cet exemple, on nomme l'imprimante "impr". Elle est connectée au port parallèle "/dev/lp" et on lui associe le modèle "standard" (modèle générique pour une imprimante de base).

```
# lpadmin  -d impr
# lpstat  -t
scheduler is running
system default destination: impr
device for impr: /dev/lp
impr accepting requests since Dec 31 18:33:17
printer impr is idle. enabled since Dec 31 18:33:21 available.
#
```

On décide que cette imprimante sera la destination par défaut et on visualise l'état du spouleur.

Dans l'autre exemple ci-dessous, on définit une deuxième imprimante "impr2" puis on déclare une classe "ascii" qui contiendra, dans l'ordre, les deux imprimantes "impr" et "impr2". Pour une classe, seule la commande "accept" est significative.

```
# lpadmin  -p impr2  -v /dev/tty01  -m standard
# accept  impr2
UX:accept: INFO: destination "impr2" now accepting requests
# enable  impr2
UX:enable: INFO: printer "impr2" now enabled
#
# lpadmin  -p impr  -c ascii
# lpadmin  -p impr2  -c ascii
# accept  ascii
UX:accept: INFO: destination "ascii" now accepting requests
# lpstat  -t
scheduler is running
system default destination: impr
```

```
members of class ascii:
        impr
        impr2
device for impr: /dev/lp
device for impr2: /dev/tty01
ascii accepting requests since Dec 31 18:36:45
impr accepting requests since Dec 31 18:33:17
impr2 accepting requests since Dec 31 18:35:29
printer impr is idle. enabled since Dec 31 18:33:21  available.
printer impr2 is idle. enabled since Dec 31 18:35:35  available.
#
```

5.2.3 Imprimantes distantes

Les impressions distantes sont apparues tardivement dans le spouleur "System V" et la mise en oeuvre diffère entre HP-UX et Solaris.

5.2.3.a Mise en oeuvre HP-UX

En ce qui concerne l'aspect client, HP-UX utilise le modèle **rmodel** pour accéder à des serveurs d'impression "System V" ou "Berkeley".

La commande "lpadmin" doit être utilisée en spécifiant "/dev/null" comme périphérique local et les options "-orm" et "-orp" pour désigner respectivement le système serveur et le nom de l'imprimante sur ce serveur.

```
lpadmin   -pimpr   -v/dev/null   -mrmodel   -ormserveur   -orplaser
```

Dans cet exemple, l'imprimante "impr" permet d'accéder à l'imprimante "laser" de la machine "serveur". Rappelons que HP-UX n'accepte pas d'espace entre les options et leur argument.

En ce qui concerne l'aspect serveur, HP-UX fournit le démon **rlpdaemon**.

5.2.3.b Mise en oeuvre Solaris

En ce qui concerne l'aspect client, la commande "lpadmin" doit être utilisée avec l'option "-s" qui permet d'indiquer (séparés par le caractère "!") le nom du serveur et le nom de l'imprimante sur ce serveur.

```
lpadmin   -p impr   -s serveur!laser
```

Si les imprimantes ont le même nom sur le client et sur le serveur, la deuxième partie de l'option "-s" n'est pas nécessaire.

```
lpadmin   -p impr   -s serveur
```

Solaris peut aussi utiliser le modèle **netstandard** pour accéder directement à une imprimante réseau dont on donne le nom par l'option "-o dest".

```
lpadmin  -p impr  -v /dev/null  -m netstandard  -o dest=remimpr
```

En ce qui concerne l'aspect serveur, Solaris fournit le démon **in.lpd**.

5.3 Service Berkeley (Linux)

5.3.1 Commandes de l'utilisateur

Les trois commandes de base de l'utilisateur du service sont :

lpr Requête d'impression
lpq État du service
lprm Annulation d'une requête

Exemples

```
$ lpr  /etc/group          Requête vers l'imprimante par défaut
request id is 64
$ lpq                      Liste de ses propres requêtes
Rank   Owner      Job   Files             Total Size
1st    root       64    /etc/group        188 bytes
$ lprm  64                 Annulation d'une requête
dfA064arc dequeued
cfA064arc dequeued
$
```

5.3.2 Ajout d'une imprimante

Le fichier **/etc/printcap** est utilisé pour déclarer les imprimantes. Chaque imprimante sera décrite par une strophe comprenant un certain nombre d'attributs. Il y a beaucoup de valeurs par défaut, ce qui explique que ces strophes sont souvent assez peu renseignées.

Quelques attributs

br	Vitesse (pour les imprimantes sur port série)
pl	Longueur de page (66 lignes par défaut)
pw	Largeur de page (132 colonnes par défaut)
lp	Nom du périphérique physique (fichier spécial)
rm	Nom d'un serveur d'impression (l'attribut "lp" est alors vide)
rp	Nom de l'imprimante sur le serveur d'impression
sd	Répertoire de "spool" (les fichiers y sont recopiés avant impression)
lf	Fichier des erreurs ("/dev/console" par défaut)
if	Filtre d'impression (l'équivalent de l'interface dans le service "System V")

Exemples de strophes

```
lp|impr1:\
        :lp=/dev/lp0:\
        :lf=/var/adm/lpd-errs:\
        :if=/usr/share/printconf/util/mf_wrapper:

impr2|remote printer:\
        :lp=:rm=serveur:rq=laser:lf=/usr/adm/lpd-errs:
```

La première strophe décrit une imprimante locale "impr1" connectée au port parallèle "/dev/lp0". Les erreurs éventuelles sont envoyées dans le fichier "/var/adm/lpd-errs". L'attribut "if" indique le nom du filtre associé.

La seconde strophe décrit une imprimante distante baptisée "impr2" (l'attribut "lp" est vide). Il s'agit de l'imprimante "laser" connectée sur la machine de nom "serveur".

En ce qui concerne l'aspect serveur, le fichier **/etc/hosts.lpd** doit contenir les noms des machines clientes dont on souhaite autoriser l'accès..

L'**imprimante par défaut** correspond à la strophe de nom "lp". La variable d'environnement "PRINTER" permet également d'avoir des destinations par défaut individuelles.

La commande lpc

Il s'agit d'une commande interactive de gestion du service.

Les actions essentielles sont :

Acceptation des requêtes	*enable*
Rejet des requêtes	*disable*
Activation physique de l'imprimante	*start*
Désactivation de l'imprimante	*stop*
Désactivation immédiate de l'imprimante	*abort*
Commandes groupées	*up (enable + start)*
	down (disable + stop)
Modification de l'ordre des requêtes	*topq*
État du service	*status*
Sortie de la commande	*quit*

```
# lpc
lpc> status
 Printer        Printing  Spooling  Jobs  Server  Subserver
impr@localhost  enabled   enabled   0     none    none
lpc> disable  impr
Printer: impr@localhost
impr@localhost.localdomain: disabled
lpc> status
 Printer        Printing  Spooling  Jobs  Server  Subserver
impr@localhost  enabled   disabled  0     none    none
lpc> stop  impr
Printer: impr@localhost
impr@localhost.localdomain: stopped
lpc> status
 Printer        Printing  Spooling  Jobs  Server  Subserver
impr@localhost  disabled  disabled  0     none    none
lpc> up  impr
Printer: impr@localhost
impr@localhost.localdomain: enabled and started
lpc> status
 Printer        Printing  Spooling  Jobs  Server  Subserver
impr@localhost  enabled   enabled   0     none    none
lpc>quit
#
```

5.4 Service AIX

5.4.1 Commandes de l'utilisateur

Les commandes de base de l'utilisateur du service sont :

qprt	Requête d'impression
qchk	État du service
qcan	Annulation d'une requête
qpri	Changement de priorité d'une requête existante

Exemples

```
$ qprt  /etc/group          Requête vers l'imprimante par défaut
$ qchk                      Liste des requêtes
Queue   Dev   Status    Job   Files       User  PP %   Blks  Cp  Rnk
lp0     lp0   DOWN
              QUEUED    543   /etc/group  root          1     1   1
$ qcan  -x 543              Annulation d'une requête
Message from qdaemon:
Job number 543 has been deleted from the queue.<EOT>
$
```

5.4.2 Principes de fonctionnement et terminologie

AIX propose un service général de mise en file d'attente. La fonctionnalité effective d'une file d'attente est déterminée par un programme exécutif ("backend program"). Le programme **piobe** correspond à des files d'attente d'imprimantes locales et le programme **rembak** correspond à des files d'attente d'imprimantes distantes.

Depuis la version 5, AIX implémente aussi le spouleur "System V".

Processus d'impression

1) Commande utilisateur
2) Mise en file d'attente via une commande générique **enq**
   Cette commande constitue l'interface entre les commandes usuelles et le démon "qdaemon", elle pourrait être invoquée directement.
3) Le démon **qdaemon** se charge du lancement du programme exécutif.

Terminologie du service (on conserve les termes en anglais)

printer Imprimante physique ("/dev/lp*n*")
queue File d'attente
device Sous-élément de la file d'attente
print queue Le couple "**queue:queue_device**"

Il s'agit d'une imprimante virtuelle qu'il est nécessaire d'associer à l'imprimante physique pour pouvoir l'utiliser via le spouleur. Une imprimante virtuelle sera associée à chaque mode possible d'utilisation (formats de fichier).

Quand une même imprimante physique supporte plusieurs formats de fichiers, il faut lui associer des noms de "queues" différents (avec chacune un seul "device").

Une "queue" ne comportera plusieurs "devices" que si ceux-ci sont associés à des imprimantes physiques différentes. Cela reproduit le concept de classe d'imprimantes du service "System V". En effet, une requête adressée à la "queue" sans mentionner de "device" explicite sera dirigée vers le premier disponible.

Comme le nom de "device" n'est quasiment jamais mentionné explicitement dans les commandes, on prend rarement la peine de le renseigner lors de la configuration. Il prend alors le nom de l'imprimante physique.

Le fichier **/etc/qconfig** décrit la configuration des files d'attente. Ce fichier reflète une situation décrite dans un fichier binaire. Il est donc préférable de ne pas éditer ce fichier et d'utiliser systématiquement les menus **smit** de configuration.

Le fichier est constitué de couples de strophes correspondant aux "print queues" définies.

Exemple

```
ascii:
        device = lp0
lp0:
        file = /dev/lp0
        header = group
        trailer = never
        access = both
        backend = /usr/lib/lpd/piobe
```

file Fichier associé (imprimante physique)
header Choix pour la page d'en-tête (par fichier, par requête ou jamais)
trailer Choix pour la page de fin
backend Nom du programme exécutif

5.4.3 Ajout d'une imprimante locale

La configuration se fait via les menus de l'utilitaire **smit**.

smit spooler

```
  Print Spooling
  Start a Print Job
  Manage Print Jobs
  List All Print Queues
  Manage Print Queues
  Add a Print Queue
  Add an Additional Printer to an Existing Print Queue
  Change / Show Print Queue Characteristics
  Change / Show Printer Connection Characteristics
  Remove a Print Queue
  Manage Print Server
  Programming Tools
```

La création des imprimantes virtuelles englobe celle de l'imprimante physique mais la création de cette imprimante physique peut être faite préalablement dans les menus "Devices" de "smit".

Add a Print Queue

local	*Printer Attached to Local Host*
remote	Printer Attached to Remote Host
xstation	Printer Attached to Xstation
ascii	Printer Attached to ASCII Terminal
hpJetDirect	Network Printer (HP JetDirect)
file	File (in /dev directory)
ibmNetPrinter	IBM Network Printer
ibmNetColor	IBM Network Color Printer
other	User Defined Backend

Printer Type

Bull
Canon
Dataproducts
Hewlett-Packard

```
IBM
OKI
Printronix
QMS
Texas Instruments
Other (Select this if your printer type is not listed above)

     Printer Type
hplj-2    Hewlett-Packard LaserJet II
hplj-3    Hewlett-Packard LaserJet III
hplj-3si  Hewlett-Packard LaserJet IIISi
hplj-4    Hewlett-Packard LaserJet 4,4M
Other (Select this if your printer type is not listed above)

     Printer Interface
parallel
rs232
rs422

     Add a Print Queue
Description               Hewlett-Packard LaserJ>
Names of NEW print queues to add
PCL                                  [hppcl]
PostScript                           [hpps]
HP-GL/2                              []
Printer connection characteristics
*  PORT number                       [p]                +
Type of PARALLEL INTERFACE                [standard]          +
Printer TIME OUT period (seconds)    [600]              +#
STATE to be configured at boot time  available          +
```

Dans cet écran final, le système propose de renseigner trois imprimantes virtuelles pour chacun des trois formats de fichier supportés par l'imprimante que l'on a sélectionnée (PCL, PostScript, HP-GL/2). Il suffit, comme expliqué précédemment, de donner le nom de "queue" (le nom du "device" sera ainsi le même que celui de l'imprimante physique). Cette imprimante physique est simultanément définie dans la deuxième partie de l'écran. Enfin, il suffit de ne pas renseigner les "print queues" correspondant aux formats non utilisés.

5.4.4 Imprimantes distantes

La configuration se fait également via les menus de l'utilitaire **smit**.

```
Manage Print Server              Aspect serveur
     List all Remote Clients with Print Access
     Add Print Access for a Remote Client
     Remove Print Access for a Remote Client
     Start the Print Server Subsystem (lpd daemon)
     Stop the Print Server Subsystem
     Show Status of the Print Server Subsystem
```

Ces menus permettent d'autoriser des machines clientes dans le fichier **/etc/hosts.lpd** et de lancer le démon supplémentaire **lpd**.

```
Add a Print Queue                Aspect client
     local          Printer Attached to Local Host
     remote         Printer Attached to Remote Host
     ..................................................

          Type of Remote Printing
     Standard processing
     Standard with NFS access to server print queue attributes
     Local filtering before sending to print server

          Add a Standard Remote Print Queue
     * Name of QUEUE to add                    []
     * HOSTNAME of remote server               []
     * Name of QUEUE on remote server               []
       TYPE of print spooler on remote server       AIX       +
       DESCRIPTION of printer on remote server  []
```

Ce dernier écran permet de renseigner le nom de "queue" côté client et côté serveur, le nom du système serveur et le type de service sur ce serveur (AIX, System V ou Berkeley).

6. Rappels sur la configuration TCP/IP de base

6.1 Format des adresses IP

Des adresses logiques doivent être associées à chaque interface réseau (**host**). Si une machine est dotée, par exemple, de deux cartes réseau, il faudra attribuer une adresse à chacune des deux interfaces.

Adresses IPv6

Au début des années 1990, la croissance rapide du réseau Internet a fait entrevoir l'arrivée de divers problèmes ou besoins. L'**IETF** (*Internet Engineering Task Force*) a alors démarré l'élaboration d'une nouvelle version du protocole IP, baptisée **IPv6** ou **IPng** (*next generation*).

Actuellement, avec une transition très progressive, on peut dire qu'il y a un peu d'IPv6 dans un monde IPv4. Les adresses IPv6 sont codées sur **128 bits**. Elles sont exprimées sous la forme de huit valeurs hexadécimales séparées par le caractère ":". L'adresse peut être déduite de l'adresse physique et peut ainsi être attribuée de façon quasi permanente en auto-configuration. Il est prévu des adresses spéciales de compatibilité avec IPv4 sous la forme "**::FFFF:n1.n2.n3.n4**" (la notation "::" remplace ici une suite contiguë de champs à zéro).

Adresses IPv4

Les adresses IPv4 sont des adresses logiques codées sur **32 bits**. Elles sont exprimées sous la forme de quatre valeurs décimales **n1.n2.n3.n4** où chaque nombre peut prendre la valeur d'un octet [0,255].

```
ADRESSE IP  =  ADRESSE RÉSEAU  +  ADRESSE HOST
```

Deux *hosts* ayant la même adresse de réseau communiqueront directement. Dans le cas contraire, une activité de **routage** sera nécessaire.

Les premiers bits (poids fort) de la partie *adresse réseau* déterminent une **classe** d'adresses. Les concepteurs ont choisi trois classes principales : A, B, C.

D'autre part, compte tenu de certaines adresses particulières ou réservées, l'intervalle complet [0,255] n'est pas toujours utilisable pour chacun des quatre octets.

- **Classe A**

Si le premier bit du premier octet est à 0, la partie *adresse réseau* correspond alors au premier octet. Cet octet peut prendre des valeurs dans l'intervalle **[1,126]** car les adresses 0 et 127 sont réservées. La partie *adresse host* correspond aux trois derniers octets, ce qui permettrait de coder plus de 16 millions de valeurs.

- **Classe B**

Si les deux premiers bits du premier octet prennent les valeurs "10", la partie *adresse réseau* correspond alors aux deux premiers octets. Ceci permet environ 16000 combinaisons possibles. Le premier octet de l'adresse peut prendre ici des valeurs dans l'intervalle **[128,191]**. La partie *adresse host* correspond aux deux derniers octets qui permettent de coder environ 65000 valeurs.

- **Classe C**

Si les trois premiers bits du premier octet prennent les valeurs "110", la partie *adresse réseau* correspond alors aux trois premiers octets. Ceci permet plus de deux millions de combinaisons possibles. Le premier octet de l'adresse peut prendre ici des valeurs dans l'intervalle **[192,223]**. La partie *adresse host* correspond au dernier octet qui permet de coder 254 valeurs car les valeurs 0 et 255 sont réservées.

Quelques adresses particulières

- **127.0.0.1**

Il s'agit de l'adresse de bouclage (**loopback**) permettant au système de s'auto-adresser sans solliciter la carte réseau.

- **plein 0** dans la partie *adresse host*

Ce format d'adresse permet la désignation du réseau lui-même. Par exemple, l'adresse 172.16.0.0 désigne tout un réseau de classe B.

- **plein 1** dans la partie *adresse host*

Ce format correspond à une adresse de diffusion (**broadcast**) permettant l'émission d'une trame vers tous les hosts du réseau (exemple : 172.16.255.255).

- **Classe D** (*multicast*)

Si les 4 premiers bits du premier octet de l'adresse prennent les valeurs "1110", cette adresse identifie un groupe d'ordinateurs partageant une application commune de type *multicast*. Ces adresses ne comportent pas de partie *adresse réseau* et le premier octet est dans l'intervalle **[224,239]**.

Réseaux non connectés à l'Internet

Des plages d'adresses sont réservées pour constituer des **réseaux privés**. Elles ne peuvent pas correspondre à une adresse officielle du réseau mondial et sont systématiquement rejetées par les routeurs de l'Internet. Ces adresses constituent donc les bons choix pour les systèmes non accessibles directement depuis le monde extérieur.

- **Classe A**

Une adresse privée est réservée : 10.0.0.0.

- **Classe B**

16 adresses sont prévues : 172.16.0.0 à 172.31.0.0.

- **Classe C**

256 adresses sont disponibles : 192.168.0.0 à 192.168.255.0.

Sous-réseaux

La structure d'une adresse IP peut être modifiée pour définir des sous-réseaux au sein du réseau principal. Celui-ci reste unique vis-à-vis du monde extérieur.

L'adresse IP est associée à un masque de sous-réseau (**subnet mask**). Si un bit est à 1 dans le masque, le bit correspondant dans la partie *host* sera interprété comme étant un bit d'adresse réseau. Nous pouvons ainsi déplacer la séparation entre la partie *réseau* et la partie *host* de l'adresse complète.

Le masque de sous-réseau existe toujours. Il prend une valeur par défaut correspondant à la frontière exacte de la classe d'adresses. La conservation de cette valeur par défaut lors de la configuration signifie que nous ne souhaitons pas mettre en œuvre des sous-réseaux.

Par exemple, si nous considérons le réseau 172.16.0.0, il s'agit d'une adresse de classe B et le *subnet mask* par défaut est donc égal à 255.255.0.0. Si nous choisissons comme masque la valeur 255.255.255.0, nous indiquons que le troisième octet de l'adresse IP devra être interprété comme un sous-réseau.

Plutôt que d'indiquer la valeur complète du masque, celui-ci est souvent précisé plus simplement par son nombre de bits. Par exemple, la notation *172.16.1.1/24* désigne un *host* d'adresse *172.16.1.1* associée à un masque de 24 bits. Ceci correspond alors au fait que le troisième octet doit être interprété comme un sous-réseau. La notation *172.16.1.1/16* indiquerait, au contraire, la conservation du masque par défaut de la classe d'adresses.

Récapitulatif des adresses IPv4

- Les trois classes d'adresses IPv4

	Adresse IP	**Subnet mask**	**Taille**
Class A	1.0.0.0 - 126.255.255.255	255.0.0.0	= /8
Class B	128.0.0.0 - 191.255.255.255	255.255.0.0	= /16
Class C	192.0.0.0 - 223.255.255.255	255.255.255.0	= /24

- Les réseaux d'adresses privées

	Adresse Réseau	**Taille du subnet mask**	**Nombre**
Class A	10.x.x.x	/8	1
Class B	172.16.x.x - 172.31.x.x	/16	16
Class C	192.168.0.x - 192.168.255.x	/24	256

6.2 Interfaces physiques

Les commandes de base pour la configuration des interfaces réseau sont les programmes **ifconfig** et **route**.

La commande **ifconfig** permet de rendre disponible une carte réseau pour son utilisation via les protocoles TCP/IP.

Les informations attendues sont :

- le nom de l'interface,

- l'adresse IP attribuée à cette interface,

- le masque de sous-réseau associé,

- l'adresse de diffusion (*broadcast*).

Par exemple, sous Linux, l'interface de bouclage (adresse 127.0.0.1) porte le nom **lo**. La première carte Ethernet, qui est souvent la seule, est baptisée **eth0**.

Sans argument, la commande *ifconfig* donne la description des interfaces existantes.

```
# ifconfig
eth0 Lien encap:Ethernet  HWaddr 00:0F:B0:D7:78:00
     inet adr:192.168.0.3  Bcast:192.168.0.255  Masque:255.255.255.0
     adr inet6: fe80::20f:b0ff:fed7:7800/64 Scope:Lien
     UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
     RX packets:1408 errors:0 dropped:0 overruns:0 frame:0
     TX packets:1045 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 lg file transmission:1000
     RX bytes:123534 (120.6 KiB)  TX bytes:236122 (230.5 KiB)
     Interruption:58 Adresse de base:0x6000

lo   Lien encap:Boucle locale
     inet adr:127.0.0.1  Masque:255.0.0.0
     adr inet6: ::1/128 Scope:Hôte
     UP LOOPBACK RUNNING  MTU:16436  Metric:1
     RX packets:10 errors:0 dropped:0 overruns:0 frame:0
     TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 lg file transmission:0
     RX bytes:660 (660.0 b)  TX bytes:660 (660.0 b)
#
```

La configuration TCP/IP peut être obtenue à l'aide d'un serveur **DHCP** (matériel spécialisé ou système Unix remplissant ce rôle). En ce qui concerne un système jouant le rôle de serveur, il est clair que sa configuration doit être mise en oeuvre de manière statique avec un nom d'hôte et une adresse bien déterminée.

Le choix de la configuration TCP/IP se fait pendant l'installation du système mais peut être modifiée a posteriori, soit en modifiant les fichiers de texte adéquats, soit via un menu graphique spécifique à la version Unix ou Linux.

6.3 Résolution des noms

Un système Unix possède un **nom de machine**, maintenu par le noyau.

Un script d'initialisation définit le nom de machine au démarrage (en utilisant la commande **hostname**).

Pour afficher le nom de machine actuel, on exécute la commande *hostname* sans argument.

Dans leurs manipulations, les utilisateurs mentionnent les noms en clair des machines. Bien évidemment, ces noms doivent être mis en correspondance avec les adresses IP.

Les noms d'hôtes ne peuvent contenir que des caractères alphanumériques, le caractère - (signe moins) et des points. Ils doivent commencer par un caractère alphabétique et se terminer par un caractère alphanumérique.

Ces correspondances peuvent être résolues de plusieurs manières :

- grâce au fichier **/etc/hosts**, fichier local ou centralisé via un service d'annuaire tel que **NIS** (*Network Information Service*).

- par une organisation en domaines **DNS**.

6.3.1 Fichier /etc/hosts

Dans des réseaux locaux regroupant un petit nombre de machines, les noms consistent en de simples chaînes de caractères et la correspondance avec les adresses peut être indiquée via un fichier local */etc/hosts*.

La syntaxe des lignes du fichier est la suivante :

```
Adresse IP    Nom officiel (ou canonique)    Liste d'alias...
```

Le deuxième champ représente le nom officiel du système (nom complet si on se trouve en contexte DNS). Il peut être suivi d'éventuels alias. Des commentaires, introduits par le caractère # (dièse), peuvent apparaître sur une ligne complète ou en fin de ligne.

En pratique, même si le contexte DNS est présent, la plupart de systèmes peuvent conserver un petit fichier contenant le nom et l'adresse des hôtes importants sur le réseau local. Ceci est utile lorsque le DNS n'est pas actif, notamment lors de la mise en route du système. Dans tous les cas, il est conseillé de laisser au moins présente la ligne correspondant à sa propre adresse IP.

Exemple de fichier /etc/hosts

```
127.0.0.1         localhost
192.168.1.10      toto.mondomaine.org    toto
192.168.1.13      titi.mondomaine.org    titi
216.234.231.5     master.debian.org      master
205.230.163.103   www.opensource.org
```

6.3.2 Aspect client DNS

Le DNS (*Domain Name System*) est le système de correspondance entre adresses et noms de machines sur l'Internet. Il permet de considérer le réseau comme une arborescence de domaines. Le **nom complet** d'une machine prend en compte la hiérarchie de ces domaines.

Exemple de noms complets

```
www.debian.org
pc1.gestion.mondomaine.com
```

Une machine Unix devient client DNS dès la présence du fichier **/etc/resolv.conf** qui mentionne le nom du domaine auquel appartient le système ainsi que la liste des serveurs DNS à contacter pour effectuer les résolutions.

Exemple de fichier /etc/resolv.conf

```
search  mondomaine.com
nameserver  80.10.246.1
nameserver  80.10.246.132
```

Les principales lignes du fichier */etc/resolv.conf* sont :

- **search** (ou *domain*)

Cette entrée définit le nom de domaine qui sera ajouté aux noms de machine ne se terminant pas par un point. Une entrée *search* est souvent préférée à une entrée *domain* car elle permet d'indiquer une liste explicite de noms de domaines de recherche (par exemple, *masociete.fr* puis *masociete.com*).

- **nameserver**

Ce type d'entrée indique l'adresse IP d'un serveur DNS. Ces lignes (au maximum 3) seront placées dans un ordre préférentiel d'interrogation.

La présence du fichier */etc/resolv.conf* est nécessaire mais n'est pas suffisante.

Il convient d'indiquer quel est l'ordre de recherche, pour la résolution de noms, entre le fichier local */etc/hosts* et d'autres sources d'informations centralisées telles que les serveurs DNS.

Dans les versions récentes des bibliothèques internes, l'ordre est déterminé dans le fichier complémentaire **/etc/nsswitch.conf**. Ce fichier est utilisé dans le cadre de bien d'autres services mais, en ce qui concerne la résolution des noms, il s'agit de renseigner correctement la ligne qui porte le nom *hosts*.

Exemple

```
hosts: files dns
```

Telle quelle, cette ligne indique que la résolution de noms doit se faire d'abord avec l'aide du fichier local */etc/hosts* et n'utiliser ensuite le service DNS qu'en cas d'échec.

Les commandes **nslookup** ou **dig** ou encore **host** permettent d'interroger les serveurs DNS :

6.4 Routage

Le routage est l'activité permettant d'acheminer des informations à travers plusieurs réseaux ou sous-réseaux. Un routeur doit disposer d'une interface sur chacun des réseaux qu'il doit interconnecter. Il s'agit, en général, de matériels spécialisés même si une machine Linux dotée de plusieurs cartes réseau peut remplir ce rôle. Dans le contexte TCP/IP, il est à noter que les termes routeur et passerelle (*gateway*) sont confondus.

On peut distinguer deux types de routage :

- le routage **statique**

Une table de routage est générée, dans les scripts de démarrage, par des appels à la commande **route**. Toutes les informations nécessaires sont disponibles dès le départ et n'ont plus besoin d'évoluer par la suite. Ceci est bien adapté à un réseau comportant un petit nombre de passerelles et ne subissant pas ou peu de modifications. La **passerelle par défaut** (celle qui est essayée quand aucune route explicite n'est disponible) constitue très souvent l'information nécessaire et suffisante dans ce type de configuration.

- le routage **dynamique**

Une table de routage dynamique est générée à partir des informations échangées par des protocoles de routage. Avec des objectifs de redondance ou de performance, ceci permet d'utiliser, par exemple, plusieurs routes vers une même destination.

La commande **route** (ou **netstat -r**) permet de visualiser la table de routage. L'option **-n** n'effectue pas de résolutions DNS et affiche directement les adresses IP. La notation *0.0.0.0*, dans la colonne *Destination*, est alors synonyme de *default*.

```
# netstat -nr
Destination  Passerelle   Genmask       Indic  Metric Ref Use Iface
192.168.0.0  *            255.255.255.0 U      0      0   0   eth0
0.0.0.0      192.168.0.1  0.0.0.0       UG     0      0   0   eth0
```

#

6.5 Super-Démon inetd

L'exécution des divers services TCP/IP se traduira par l'activation de nombreux démons, soit permanents, soit lancés ponctuellement via un processus qualifié de *super-démon* (ou *super-serveur*).

Le super-démon est, bien entendu, un processus permanent qui consulte le fichier de configuration **/etc/inetd.conf** pour connaître l'ensemble des programmes serveurs qu'il doit piloter et notamment les ports sur lesquels il doit écouter.

Les différents champs de chaque ligne décrivent respectivement :

- le service (nom d'une entrée dans le fichier */etc/services* permettant de connaître le numéro de port réservé),

- le type de service (*stream* : mode connecté, *dgram* : mode non connecté),

- le protocole de transport (nom d'une entrée dans le fichier */etc/protocols* permettant de connaître le numéro de port réservé),

- l'indicateur *wait* ou *nowait*,

- le propriétaire du programme à démarrer (très souvent le compte *root*),

- le nom complet du programme,

- les éventuels arguments de ce programme (pour des raisons d'implémentation, le nom du programme lui-même doit être répété en tant que premier argument.).

Pour une prise en compte immédiate, toute modification de ce fichier (pour ajouter, enlever ou modifier un service) doit s'accompagner de l'envoi, vers le démon, du signal *SIGHUP*.

Le logiciel **TCP Wrapper** (démon **tcpd**) est souvent mis en oeuvre pour améliorer la sécurité. On peut alors gérer des autorisations à l'aide de deux fichiers **/etc/hosts.allow** et **/etc/hosts.deny**.

Dans le fichier */etc/inetd.conf*, le démon *tcpd* se place en tant que couche intermédiaire vis-à-vis du super-démon.

Ainsi, la ligne

```
login  stream  tcp  nowait  root  /usr/sbin/rlogind  rlogind
```

devient

```
login  stream  tcp  nowait  root  /usr/sbin/tcpd  /usr/sbin/rlogind
```

7. Serveur DHCP

7.1 Un serveur DHCP pour quoi faire ?

DHCP (Dynamic Host Configuration Protocol) permet à un équipement (PC, serveur, imprimante, téléphone IP, …) d'obtenir sa configuration réseau sans « rien faire ni trop connaître ». Il s'agit aussi d'offrir un service de configuration délicat à mettre en place manuellement sur certains équipements proposants une interface limitée ou inexsitante.

Les besoins en « informations de configuration réseau » d'un client DHCP peuvent se limiter au minimum (adresse IP, masque de sous-réseau), être classiques (@IP, masque, passerelle, DNS) ou être beaucoup plus fins et variés avec d'autres informations comme les adresses de serveur de NEWS, SIP, SMTP, le nom de notre domaine, etc…

L'attribution d'adresse IP peut donc se faire avec le service DHCP. Ce mécanisme est intéressant lorsque le nombre d'adresse IP attribuable est faible, lorsque les machine considérées sont souvent déplacées d'un réseau à l'autre et bien sûr lorsque la topologie du réseau est inconnue.

L'adresse attribuée n'est pas forcément connue de toutes les autres machines, ce qui rend impossible l'utilisation de DHCP (client) sur un serveur.

On parle également aussi de **BOOTP** (Bootstrap Protocol) comme protocole de configuration. Il est plus ancien et moins riche que DHCP. Il était destiné à configurer les stations de travail sans disque avec des capacités de démarrage limitées (imprimantes par exemple). Le service offre une configuration IP, un ou plusieurs noms de fichier de démarrage, et l'adresse du serveur TFTP à contacter. DHCP offre également ce service.

Le protocole DHCP étant indépendant de la plateforme (UNIX, Windows, Mac, Mainframe, …) la nature des informations peut ne pas concerner l'environnement du client mais c'est aussi l'intérêt de ne pas être dédié à un système d'exploitation. Afin de normaliser les échanges chaque « information» DHCP porte un numéro que l'on retrouve d'un serveur DHCP à un autre.

7.2 Vocabulaire de DHCP

Le terme DHCP désigne à la fois le **protocole** et le **service** lui-même (comme ftp).

Il fonctionne selon un modèle client/serveur, on parlera donc de **client DHCP** et de **serveur DHCP**.

Les informations délivrés par le serveur DHCP sont valables un certain temps. On parle de **bail** DHCP.

Le client DHCP effectue un broadcast pour essayer de « trouver » un serveur DHCP. Les broadcats ne traversant pas les routeurs on devra éventuellement installer des **relais DHCP** sur ces derniers pour « prolonger » la requête.

7.3 Principe de fonctionnement

➔DECOUVERTE

Le client fait une requête (broadcast) DHCPDISCOVER afin de trouver un serveur DHCP ou BOOTP (`dhcpd` ou `in.dhcpd`). Il utilise le port udp client 68 et cherche à contacter le port UDP 67 du serveur DHCP.

➔REPONSE(S)

Plusieurs serveurs peuvent répondre par un DHCPOFFER afin de proposer leurs «services» au client. La réponse se fait également en broadcast car le client ne dispose pas encore d'adresse IP.

➔CHOIX DU SERVEUR DHCP

Le client accède par DHCPREQUEST au serveur choisi

➔ACCEPTATION

Le serveur retourne les informations demandées par DHCPACK à son client pour lui signifier qu'il l'accepte.

Les requêtes DHCPDISCOVERER étant faites en broadcast, elles ne peuvent donc pas franchir les routeurs. Pour cette raison il peut être nécessaire de mettre en place des serveurs intermédiaires sur chaque sous-réseau (RELAY). Les serveurs RELAY ne faisant que contacter les vrais serveurs DHCP (PRIMARY).

Un serveur trouve les informations nécessaires à une cliente par rapport à la provenance de la requête en terme de *netid* : fichier `/var/dhcp/`*`netid`*`_dhcp` (Solaris) ou directement dans le fichier de configuration principal `/etc/dhcpd.conf` (Linux). Le renouvellement du bail dirigé vers le serveur DHCP avant la fin du bail (à sa moitié) permet d'éviter les 4 étapes par diffusion d'une demande compète

```
OLD-BROADCAST -> BROADCAST    DHCP/BOOTP DHCPDISCOVER
mais2 -> (broadcast)  ARP C Who is 150.9.6.43, 150.9.6.43 ?
mais2 -> (broadcast)  ARP C Who is 150.9.6.43, 150.9.6.43 ?
mais2 -> BROADCAST    DHCP/BOOTP DHCPOFFER
mais2 -> (broadcast)  ARP C Who is 150.9.6.43, 150.9.6.43 ?
OLD-BROADCAST -> BROADCAST    DHCP/BOOTP DHCPREQUEST
mais2 -> BROADCAST    DHCP/BOOTP DHCPACK
OLD-BROADCAST -> (broadcast)  ARP C Who is 150.9.6.43, 150.9.6.43 ?
150.9.6.43 -> (broadcast)  ARP C Who is 150.9.6.43, 150.9.6.43 ?
```

Visualisation d'une requête DHCP

7.4 Mise en œuvre de DHCP

La mise en place de DHCP consiste d'abord à créer un ou plusieurs serveurs (PRIMARY, RELAY) pour ensuite s'occuper des machines clientes.

- PRIMARY : Serveur d'adresse IP
- RELAY : Serveur «relai» chargé de contacter un PRIMARY suite à une requête cliente lorsque celle-ci est sur un sous réseau différent du PRIMARY

Quel que soit l'Unix utilisé, le principe du serveur DHCP reste le même :

- Configuration du service DHCP
- Lancement d'un daemon à l'écoute des requêtes DHCP des clientes

	Linux	Solaris8	AIX
daemon	dhcpd	in.dhcpd	dhcpd
Fichiers des symboles	man dhcpd	/etc/dhcp : inittab	
Fichiers de configuration	/etc/dhcpd.conf	/var/dhcp : netid_dhcp dhcptab	/etc/dhcpsd.cnf db_file (database)

Principaux fichiers de DHCP

Les fichiers de configuration d'un serveur DHCP contiennent la liste des sous-réseaux auxquels appartiennent les clientes potentielles. Pour chaque sous-réseau il suffit donc de préciser les différentes informations dont peuvent avoir besoin les clientes (adresse IP, serveur DNS, Bail etc.).

Au sein de chaque sous-réseau, afin d'éviter d'avoir à décrire individuellement chaque cliente, il est préférable de les regrouper en fonction de leurs points communs : même serveur DNS, même Time serveur etc.

Pour cette raison, la première chose à faire lors de la configuration consiste à déterminer précisément les sous-réseaux à gérer et mettre à jour en conséquence les fichiers correspondants.

Toute la configuration d'un serveur DHCP peut être réalisée à l'aide de Webmin.

7.5 Configuration d'un serveur DHCP sur Solaris

Solaris travaille avec plusieurs fichiers de configuration. Un par sous-réseau et un général. Le général (dhcptab) permet de créer des **macros** qui contiennent des définitions utilisables séparément. Chaque macro pouvant elle-même référencer une autre macro. Il existe déjà des définition internationales utilisables dans les macros (DNSdomain, Timeserver, Hostname etc.). Si cela ne suffit pas, l'administrateur peut créer ses propres définitions que l'on appelle des **symboles**. Ces symboles devenant immédiatement utilisables.

Une fois les macros et les symboles définis, il reste à configurer les caractéristiques des clientes en utilisant les macros et les symboles préalablement définis.

Trois étapes sont nécessaires pour cette configuration :

- Création du fichier des macros et symboles `dhcptab`
- Création du fichier de correspondance clientes, adresse IP (`netid_dhcp`) netid = numéro de réseau
- Lancement du serveur, le daemon `dhcpd` (`in.dhcpd`)

Toute cette configuration est réalisable par la commande **`dhcpconfig`** ou, depuis Solaris 8, avec l'outil graphique **`dhcpmgr`**. Les fichiers cités par la suite sont modifiables par les commandes **`pntadm`** (pour `netid_dhcp`) et **`dhtadm`** (`pour dhcptab`).

Le fichier *netid*_dhcp

```
IDclient   Flag   IPclient       IPserveur     Bail    Macro
00         00     150.9.170.41   150.9.170.2   0       res02
00         00     150.9.170.42   150.9.170.2   0       res02
00         00     150.9.170.43   150.9.170.2   0       res02
00         00     150.9.170.44   150.9.170.2   0       res02
00         00     150.9.170.45   150.9.170.2   0       res02
```

- IDclient
  00 : pas encore attribuée
  xxxx : adresse IPclient attribuée au client xxxx
- Flag
  00 : allocation dynamique de l'adresse IP
  02 : allocation manuelle de l'adresse IP
  03 : attribution permanente de l'adresse *IPclient* au client *IDclient*
  04 : adresse inutilisable car sans doute déjà utilisée sur le réseau
  08 : adresse IP attribuable uniquement à une cliente BOOTP
- IPclient : adresse IP attribuable à une cliente
- IPserveur : adresse du serveur DHCP dont dépend le client
- Bail
  0 : Non encore défini
  -1 : infini ou sans importance
  xxx : durée en seconde du bail
- Macro : nom de la macro associée, dans le fichier dhcptab

Le fichier dhcptab

Ce fichier contient des définitions de «symboles» ou de «macros» qui pourront être utilisées pour la configuration des clients.

Chaque ligne du fichier dhcptab est composée de plusieurs champs séparés par le symbole du pipe «|».

```
# SUNWfiles1_dhcptab
#
# Do NOT edit this file by hand -- use dhtadm(1M) or dhcpmgr(1M) instead
#
Locale|m|7279787322667696129|:UTCoffst=0:
res02|m|3371507271040237569|:Include=Locale:Timeserv=:LeaseTim=86400:Leas
eNeg:
150.9.170.0|m|7267683898669137921|:Broadcst=150.9.170.255:Subnet=255.255.
255.0:MTU=1500:
res03|m|3317052717402073965|:Include=Locale:Timeserv=:LeaseTim=3600:Lease
Neg:
150.9.171.0|m|7286768983196631297|:Broadcst=150.9.171.255:Subnet=255.255.
255.0:MTU=1500:
```

- Nom : nom de l'objet (64 caractères pour un macro et 8 pour un symbole)
- Type : m (macro) ou s(symbole)
  Une **macro** regroupe des informations complémentaires pour les clientes. Elle est composée de symboles internes standard et/ou de symboles propres au site ou à des fabricants.
  Un **symbole** permet de définir des spécificités pour le site ou pour les fabricants. Ces symboles ne font pas partie des symboles internationaux qui, eux, sont rappelés dans le fichier `/etc/dhcp/dhcptags` (`/etc/dhcp/inittab` en Solaris 8)
- Le contenu d'un champs valeur dépend du type :
  - **m**
    séries d'associations «étiquette=valeur» encadrées par «:».
  - **s**
    La valeur d'un symbole est organisée en champs séparés par des virgules.

Exemples de macro :

- **Locale** : avec le mot clef UTCoffst, cette macro définit le décalage du temps par rapport à GMT.
- **res02** : cette macro définit les caractéristiques des clients qui appartiendront au «groupe» res02.
  Include=Locale : Le groupe 02 utilise l'horloge GMT+0
  Timeserv= : Le «serveur de temps» est le client lui-même
  LeaseTim=86400 : Définit la durée (en secondes) du BAIL accordé au client (1 journée dans

cet exemple)
LeaseNeg : Booléen (TRUE si présent, False si absent) autorise un client à renégocier son bail avant expiration.

- **150.9.170.0** : cette macro définit les caractéristiques des clients qui appartiendront au «groupe» 150.9.170.0
Broadcst=150.9.170.255 : définit la valeur de Broadcast
Subnet=255.255.170.0 : définit le Netmask
MTU=1500 : Maximum Trame Unit à 1500 octets

Si un client «réclame» son hostname, le fichier hosts doit avoir été mis à jour avec toutes les valeurs d'adresses IP dynamiques possibles. Cette opération est automatiquement réalisable si l'on utilise `dchpconfig` ou `dhcpmgr`.

Si un client appartient à plusieurs «groupes», c'est la somme des informations de ces groupes qui lui sera affectée.

7.6 Configuration d'un serveur DHCP sur Linux

Linux ne dispose que d'un seul fichier de configuration : `/etc/dhcpd.conf` contenant toutes les informations nécessaires. Une fois ce fichier mis à jour, il suffit de lancer (ou) de relancer le daemon dhcpd.

```
ddns-update-style none;
subnet 150.9.0.0 netmask 255.255.0.0 {
     range 150.9.6.41 150.9.6.45;
     host lin04 {
          hardware ethernet 8:0:20:7d:de:7;
          option host-name "lin04";
          }
     host lin01 {
          hardware ethernet 0:50:da:38:a1:a7;
          option host-name "lin01";
          }
     }
```

Exemple de /etc/dhcpd.conf

7.6 Configuration d'un serveur DHCP sur AIX 5L

Sur AIX 5L le package correspondant au serveur DHCP est **bos.net.tcp.server**

Pour vérifier votre installation DHCP :

```
lslpp -w /etc/dhcpsd.cnf
```

Exemple de sortie :

```
File              file set             Type
-----------------------------------------
/etc/dhcpsd.cnf   bos.net.tcp.server   File
```

1 Mettre à jour le fichier de configuration de DHCP

```
cat /etc/dhcpsd.cnf
```

```
leaseTimeDefault       0xffffffff
leaseExpireInterval    1 year
supportBOOTP           yes
supportUnlistedClients yes
network 30.0.0.0 255.255.255.0
  { #Commentaires libres
  option 51   0xffffffff
  hostnamepolicy suggested
  subnet 30.0.0.0 30.0.0.51-30.0.0.150
  }
```

2 Configurer la carte réseau

```
chdev -l en1 -a netaddr='30.0.0.1' -a netmask='255.255.255.0' -a state='up'
```

3 Vérifier la configuration réseau

```
netstat -in
```

Exemple de sortie

```
en0  1500 link#2   0.1.22.6c.6d.c8 865867837   0 358343036  0  0
en0  1500 9.111.11 9.111.11.1      865867837   0 358343036  0  0
en1  1500 link#3   0.1.22.b9.36.e  5064631     0 1575938    0  0
en1  1500  30      30.0.0.1        5064631     0 1575938    0  0
lo0  16896 link#1                  17395186    0 18161794   0  0
lo0  16896 127     127.0.0.1       17395186    0 18161794   0  0
lo0  16896 ::1                     17395186    0 18161794   0  0
```

4 Démarrer le serveur DHCP

```
startsrc –s dhcpsd
```

5 Vérifier le démarrage du serveur DHCP

```
lssrc -ls dhcpsd | more
```

Exemple de sortie

```
Log File:                   /usr/tmp/dhcpsd.log
Log Level:                  0x806
Client Expire Interval:     3600
Reserve Expire Interval:    900
Bad Addr Reclaim Interval: 4294967295
Database Save Interval:     3600

IP Address  Status  Duration  Time  Stamp   Client ID
----------  ------  --------  ----- ------- ---------
30.0.0.51   Leased  Infinite  Apr 4  16:27  1-000d600b6297
30.0.0.52   Leased  Infinite  Apr 4  16:25  1-000d600b78f3
30.0.0.53   Leased  Infinite  Apr 4  15:30  1-a2e260017002
30.0.0.54   Leased  Infinite  Apr 4  13:08  1-a2e260004002
30.0.0.55   Leased  Infinite  Apr 4  12:06  1-a2e260005002
30.0.0.56   Leased  Infinite  Apr 4  12:04  1-a2e260006002
30.0.0.57   Free
30.0.0.58   Free
30.0.0.59       Free
```

7.7 Configuration d'un serveur DHCP sur HPUX 11

Extraits de http://docs.hp.com/fr/B2355-95153/apas04.html

Exemple de plage retenue : 15.1.48.50 – 15.1.48.80

```
dhcptools -h fip=15.1.48.50 no=30 sm=255.255.255.0 hn=devlab##
```

Cette commande crée un fichier /tmp/dhcphosts qui peut être inséré dans votre base de données /etc/hosts ou DNS/NIS.

Utilisez la procédure suivante sur le futur serveur DHCP :

1 Démarrez l'application interactive SAM en tapant sam. (Notez que vous devrez peut-être définir la variable DISPLAY pour utiliser la version graphique.)

2 Double-cliquez sur l'icône Networking and Communications.

3 Double-cliquez sur l'icône Bootable Devices.

4 Double-cliquez sur l'icône DHCP Device Groups Booting From this Server.

L'écran qui apparaît doit présenter une liste de tous les groupes déjà définis (cette liste peut être vide si DHCP n'a pas encore été configuré).

Pour ajouter le nouveau groupe d'adresses IP que vous venez d'allouer au début de la procédure cliquez sur l'option de menu Action et sélectionnez Add DHCP Group.

Cette commande doit faire apparaître un formulaire de paramétrage

7.8 Configuration des clients DHCP sur Solaris

- Créer le fichier vide suivant :
  - `/etc/dhcp.xxx` (xxx = nom de l'interface réseau elxl0, nei0 etc.)

Une fois terminé, rebooter le client.

7.9 Configuration des clients DHCP sur Linux

- Créer le fichier vide `/etc/dhclient.conf`
- modifier le fichier `/etc/sysconfig/network-scripts/ifcfg-eth0`

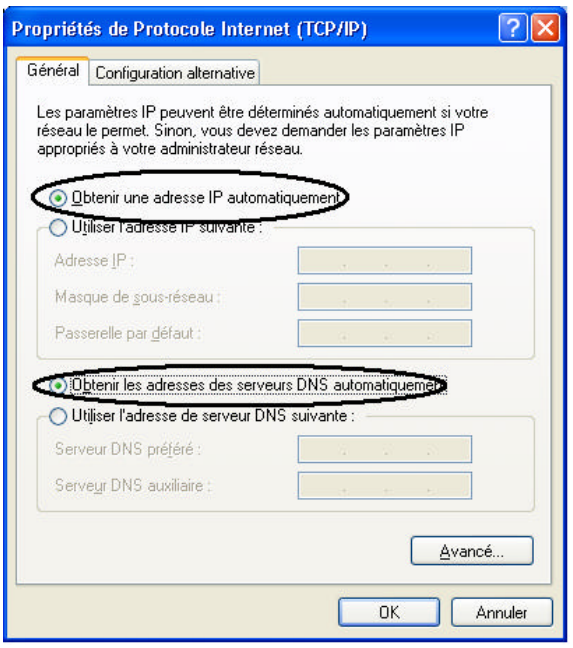
```
DEVICE=eth0
BOOTPROTO=dhcp
NETMASK=255.255.0.0
ONBOOT=yes
```

Une fois terminé, rebooter le client.

7.10 Configuration des clients DHCP sous Windows XP

Clic droit Favoris réseau / Propriétés. Clic droit sur la carte réseau / Propriétés / TCP/IP

Cocher « Obtenir une adresse IP automatiquement » et « Obtenir les adresses des serveurs DNS automatiquement » ….



Visualisation de la configuration d'un client DHCP sous Windows :

ipconfig /all

```
Configuration IP de Windows

        Nom de l'hôte . . . . . . . . . . . . : acer
        Suffixe DNS principal . . . . . . . :
        Type de noud . . . . . . . . . . . . : Inconnu
        Routage IP activé . . . . . . . . . . : Non
        Proxy WINS activé . . . . . . . . . . : Non
        Liste de recherche du suffixe DNS : mydomain.local

Carte Ethernet Connexion au réseau local:
        Suffixe DNS propre à la connexion : mydomain.local
        Description . . . . . . . . . . . . : NVIDIA nForce Networking Controller
        Adresse physique . . . . . . . . . .: 00-16-D3-50-EB-6F
        DHCP activé. . . . . . . . . . . . . : Oui
        Configuration automatique activée . . . . : Oui
        Adresse IP. . . . . . . . . . . . . . : 192.168.0.72
        Masque de sous-réseau . . . . . . . . : 255.255.255.0
        Passerelle par défaut . . . . . . . . : 192.168.0.254
        Serveur DHCP. . . . . . . . . . . . . : 192.168.0.253
        Serveurs DNS . . . . . . . . . . . .  : 62.4.16.70
                                                62.4.17.69
```

```
        Bail obtenu . . . . . . . . . . . : vendredi 21 mai 2010 15:51:20
        Bail expirant . . . . . . . . . . : samedi 29 mai 2010 15:51:20
```

Libération d'un bail DHCP sous Windows :

ipconfig /release

```
Carte Ethernet Connexion au réseau local:

        Suffixe DNS propre à la connexion :
        Adresse IP. . . . . . . . . . . . : 0.0.0.0
        Masque de sous-réseau . . . . . . : 0.0.0.0
        Passerelle par défaut . . . . . . :
```

Si un serveur DHCP ne répond pas le protocole APIPA (Automatic Private Internet Protocol Addressing) permet au système de s'attribuer automatiquement une adresse IP. Le daemon Avahi joue ce rôle sous UNIX.

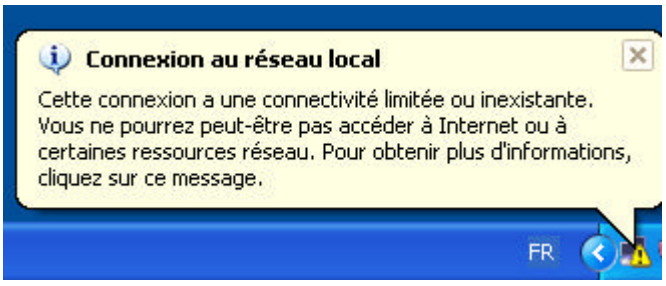
APIPA utilise la plage d'adresses IP de 169.254.0.0 à 169.254.255.255. Elle est réservée à cet usage auprès de l'IANA. Si un serveur DHCP est censé exister sur le réseau il n'est donc pas « normale » d'obtenir une adresse en 169.254.x.x (problème de câblage, relai DHCP, service arrêté, …).

Exemple d'obtention d'adresse IP par APIPA :

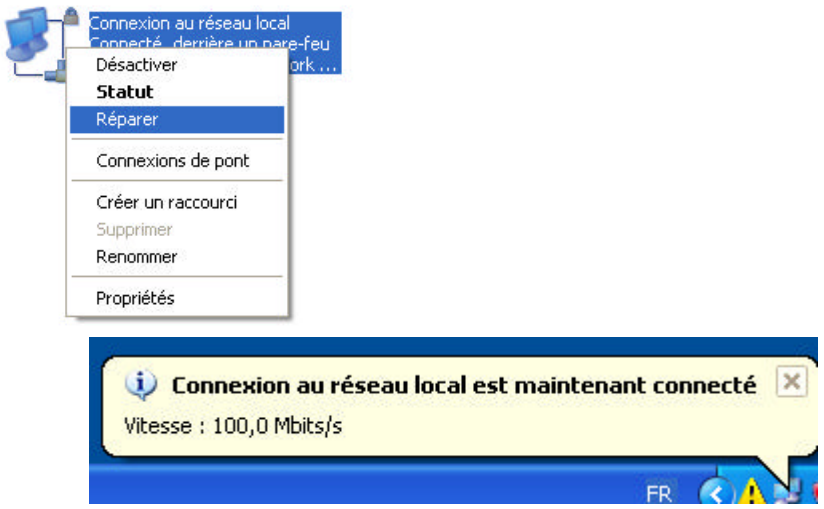
ipconfig

```
Carte Ethernet Connexion au réseau local:

        Suffixe DNS propre à la connexion :
        Autoconfiguration d'adresse IP. . : 169.254.205.60
        Masque de sous-réseau . . . . . . : 255.255.0.0
        Passerelle par défaut . . . . . . : 169.254.205.60
```

Observation de la non présence d'un serveur DHCP …



Lorsqu'il est à nouveau opérationnel on peut taper **ipconfig /renew** ou réparer « graphiquement » …


Connexion au réseau local
Désactiver
Statut
Réparer
Connexions de pont
Créer un raccourci
Supprimer
Renommer
Propriétés
Connexion au réseau local est maintenant connecté
Vitesse : 100,0 Mbits/s
FR

8. Mettre en oeuvre un serveur DNS

8.1 Présentation du DNS

Dans des réseaux locaux regroupant un petit nombre de machines, les noms consistent en de simples chaînes de caractères et la correspondance avec les adresses IP peut être indiquée via le fichier **/etc/hosts** sur chaque machine.

Le service **DNS** (*Domain Name System*) est un système hiérarchique distribué permettant la résolution des noms de machines en adresses IP et inversement.

Son utilisation est obligatoire sur le réseau Internet et s'avère également très intéressante pour un réseau d'entreprise.

Le DNS considère le réseau comme une arborescence de **domaines**.

Directement sous la racine, identifiée par le caractère **.** (point), se trouvent des domaines dits de premier niveau (**TLDs** : *Top Level Domains*).

C'est l'organisme **ICANN** (*Internet Corporation for Assigned Names and Numbers*) qui supervise l'attribution des noms de domaine de premier niveau (voir l'URL www.icann.org/tr/french.html).

Cet organisme a remplacé en 1998, l'ancien organisme historique **InterNIC**.

Certains domaines (à trois lettres) sont des domaines génériques historiques :

.com	Organisations commerciales
.edu	Écoles, universités ... (accréditées aux Etats Unis)
.gov	Organisations gouvernementales américaines
.mil	Organismes militaires américains
.net	Organismes spécialisés dans les réseaux
.int	Organisations gouvernementales internationales
.org	Organismes à but non lucratif

D'autres domaines de premier niveau, constitués de deux lettres, correspondent à des **domaines géographiques**, normalisés au niveau de l'organisme international ISO (codes **ISO 3166**, voir l'URL www.iso.org/iso/fr/country_codes).

D'autres domaines complémentaires ont vu le jour plus ou moins récemment, tels que :

.biz	Affaires (business)
.name	Individus (avec charte de nommage)
.eu	Union européenne
.info	Informations

etc…

Pour la France, l'**AFNIC** (*Association Française pour le Nommage Internet en Coopération*) est l'organisme en charge de la gestion administrative et technique des noms de domaine en **.fr** et **.re** (Réunion).

Dans le cadre d'un réseau local d'entreprise, nous pouvons créer arbitrairement nos domaines de premier niveau. On utilise fréquemment des noms tels que **.lan** ou **.local** ou encore **.intra**.

Le nom **.example** est recommandé pour une utilisation dans le cadre de documentations ou de préconfigurations de matériels ou de logiciels.

Dans les noms de domaines, les minuscules et majuscules sont confondues.

Le terme **zone** est souvent assimilé à celui de domaine, il ne faut cependant pas les confondre. Lorsque l'on est en charge d'un domaine donné, il est bien sûr possible de définir des **sous-domaines**.

Initialement, une zone correspond au seul nom de domaine DNS. Si d'autres domaines sont ajoutés en dessous du domaine utilisé pour créer la zone, ils peuvent faire partie de cette même zone ou appartenir à une autre zone. En effet, un sous-domaine ajouté à une zone peut être :

- inclus et géré dans les enregistrements de la zone d'origine ;

- délégué à une autre zone pour prendre en charge ce sous-domaine.

Le **nom complet d'une machine** prendra en compte la hiérarchie des domaines, par exemple : www.masociete.com, pc1.gestion.masociete.com , www.icann.org etc..

8.2 Le logiciel bind de l'ISC

Sous Unix/Linux, l'implémentation standard de fait du service DNS est celle de l'**ISC** (*Internet Software Consortium*) à travers le logiciel **Bind** en version 8.x ou 9.x.

Le consortium ISC (voir le site www.isc.org) développe par ailleurs un serveur DHCP, un serveur de news (*INN*) ainsi qu'un serveur de synchronisation temporelle (*NTP*).

Entre autres sites, on pourra conseiller l'excellent portail *www.bind9.net* pour accéder à des informations et des documentations concernant le logiciel BIND.

Le serveur DNS du logiciel *Bind* correspondra au démon **named**.

Ce démon utilisera un fichier de configuration dont le nom habituel est **named.conf**.

Ce fichier initial permet de spécifier le **rôle** et l'**emplacement** des autres fichiers de configuration. Son contenu indique également le type de serveur (maître, esclave, cache).

Compilation du serveur Bind depuis l'archive source

Si nous ne souhaitons pas utiliser les paquets logiciels pré-configurés, nous pouvons décider de compiler nous-même le logiciel Bind en partant de l'archive source disponible sur le site www.isc.org.

Nous allons suivre cette démarche pour une étude progressive de la mise en œuvre de différents types de serveurs DNS.

Dans un premier temps, nous avons téléchargé l'archive source stable la plus récente depuis le site officiel. Nous nous plaçons ensuite dans le répertoire de travail */usr/local/src* pour décompresser puis restaurer cette archive.

```
/usr/local/src# ls -l bind*
-rw-r--r-- 1 root root 6341409 2007-10-12 12:20 bind-9.4.1-P1.tar.gz
/usr/local/src# tar xzf bind-9.4.1-P1.tar.gz
/usr/local/src#
```

Nous allons ensuite nous positionner dans le sous-répertoire résultat de la restauration de l'archive.

```
/usr/local/src# cd bind-9.4.1-P1
/usr/local/src/bind-9.4.1-P1#
```

Le fichier **README** comporte les indications nécessaires à la suite des opérations.

Nous décidons d'installer le logiciel dans le répertoire */usr/local/bind9*.

```
/usr/local/src/bind-9.4.1-P1# mkdir /usr/local/bind9
/usr/local/src/bind-9.4.1-P1# ./configure --prefix=/usr/local/bind9
```

……

……

Après l'exécution de la commande *configure*, les fichiers **config.status** et **config.log** contiennent les traces des traitements effectués. Ils peuvent aider à résoudre d'éventuels problèmes. En cas de succès, le fichier **Makefile** est le fichier résultat, exploité ensuite par la commande *make*.

Nous pouvons opérer maintenant la compilation effective par la commande **make** puis, en cas de succès, nous pouvons opérer l'installation du logiciel dans le répertoire cible par la commande **make install**.

Nous pouvons ensuite constater le résultat de l'installation dans la sous-arborescence du répertoire /usr/local/bind9.

```
/usr/local/src/bind-9.4.1-P1# cd /usr/local/bind9
/usr/local/bind9# ls -l
total 28
drwxr-sr-x 2 root staff 4096 2007-10-12 12:51 bin
drwxr-sr-x 2 root staff 4096 2007-10-12 12:50 etc
drwxr-sr-x 9 root staff 4096 2007-10-12 12:50 include
drwxr-sr-x 2 root staff 4096 2007-10-12 12:51 lib
drwxr-sr-x 6 root staff 4096 2007-10-12 12:50 man
drwxr-sr-x 2 root staff 4096 2007-10-12 12:51 sbin
drwxr-sr-x 3 root staff 4096 2007-10-12 12:50 var
/usr/local/bind9#
```

Dans cette arborescence, les outils clients d'interrogation du serveur sont dans le répertoire *bin* tandis que le démon *named* et d'autres commandes administratives se trouvent dans le répertoire *sbin*. Il convient donc d'ajouter ces deux répertoires en tête de notre *PATH shell*, en faisant le nécessaire pour que cet ajout soit permanent.

Par exemple, la ligne suivante peut être ajoutée à la fin de notre fichier *.bashrc* :

```
PATH=/usr/local/bind9/bin:/usr/local/bind9/sbin:$PATH
```

Lors du lancement du démon *named*, il sera nécessaire que celui-ci s'exécute sous l'identité d'un compte créé à cet effet. Nous ferons donc le nécessaire pour la création d'un compte sécurisé et d'un groupe associé que nous appellerons, par exemple, *bind*.

Afin que ce compte *bind* dispose des bons droits d'accès, nous en ferons le propriétaire des sous-répertoires *etc* et *var*, utilisés dans la suite pour placer les fichiers de configuration et de travail.

Nous pourrons utiliser par exemple, la commande :

```
/usr/local/bind9# chown –R bind:bind etc var
/usr/local/bind9#
```

8.3 Types de serveurs DNS

Pour améliorer les performances, tous les serveurs DNS fonctionnent sur un principe de mémoire cache.

Il existe trois types de serveurs DNS :

- **serveur maître** (**master**)

Ce serveur possède les informations officielles sur le domaine et fait autorité sur la zone concernée. Il charge en mémoire cache ses informations à partir de fichiers disques maintenus par le gestionnaire.

- **serveurs esclaves** (**slave**)

Ces serveurs chargent en mémoire les informations à l'aide de copies des fichiers d'un serveur maître. Ils interrogent régulièrement celui-ci pour mettre à jour leurs informations. Ces serveurs sont aussi considérés comme fournissant des informations officielles et font également autorité sur la zone concernée.

- **serveurs caches** (**hint**)

Ces serveurs ne possèdent pas d'informations sur disque. Ils répondent aux requêtes des clients en interrogeant d'autres serveurs faisant autorité. Ils conservent les informations dans leur mémoire cache pour les requêtes ultérieures. On les qualifie de serveurs cache seulement (*caching only servers*). Ils ne sont pas considérés comme serveurs officiels et ne font pas autorité sur la zone concernée.

8.4 Configuration d'un serveur cache seulement (hint)

Nous allons procéder par l'exemple en mettant en œuvre le premier type de serveur DNS le plus immédiat à réaliser, à savoir un serveur cache seulement.

Ce type de serveur ne répond pas directement aux requêtes, il les transmet aux serveurs DNS officiels. Il retourne ensuite la réponse vers l'émetteur de la requête et met cette réponse en mémoire cache afin de ne pas recommencer le même traitement ultérieurement. On peut dire qu'un serveur cache apprend au fur et à mesure des questions qu'on lui soumet. Ses réponses sont ensuite très rapides mais pourraient être obsolètes en cas de changement au niveau des serveurs officiels. C'est la raison pour laquelle ce type de serveur ne fait pas autorité.

Nous considérons ici que notre serveur accède à l'Internet et qu'il va ainsi pouvoir réaliser des recherches sur le réseau mondial.

8.4.1 Fichier named.conf pour un serveur de type cache seulement

Le démon **named** utilise donc un fichier de configuration général **named.conf**. Nous placerons ce fichier dans le répertoire **/usr/local/bind9/etc**.

Ce fichier est composé de sections. Les lignes de commentaire commencent par les caractères // .

Il commence, en général, par une section **options** qui comporte au moins une directive **directory** qui précise le répertoire où se trouveront les autres fichiers de configuration. Nous choisissons de placer les fichiers dans le sous-répertoire *var* de notre répertoire d'installation.

Ensuite, nous trouverons principalement des sections **zone** qui définissent le rôle du serveur par rapport à une zone donnée (directive *type*), en indiquant principalement le nom du fichier de données associé (directive *file*).

```
//
//  Exemple de serveur cache seulement
//

options {
  // emplacement des autres fichiers
  directory "/usr/local/bind9/var";
};

// Serveur cache (hint) pour la racine Internet (.)
// le fichier named.root contient les coordonnees
// des serveurs officiels de la racine
zone "." IN {
  type hint;
  file "named.root";
};

// Serveur maitre pour la resolution de localhost en 127.0.0.1
```

```
zone "localhost" IN {
  type master;
  file "zone.localhost";
};

// Serveur maitre pour la resolution de 127.0.0.1 en localhost
zone "0.0.127.in-addr.arpa" IN {
  type master;
  file "rev.127.0.0";
};

// fin du fichier named.conf
```

8.4.2 Autres fichiers pour un serveur de type cache seulement

8.4.2.a Fichier named.root

Notre fichier **named.conf** est auto-commenté et comporte plusieurs strophes de type *zone*.

La première précise que nous serons serveur cache (type **hint**) par rapport aux serveurs officiels de la racine Internet.

Le fichier que nous avons baptisé **named.root** contient la liste officielle de ces serveurs. Ce fichier peut s'obtenir facilement comme indiqué dans les premiers commentaires.

Nous ne détaillons pas immédiatement la syntaxe des enregistrements, il nous suffit, pour l'instant, de comprendre qu'il s'agit des coordonnées des serveurs officiels de la racine Internet qui pourront être interrogés par notre serveur cache. Ces serveurs dédiés connaissent les coordonnées de tous les serveurs DNS de toutes les zones de l'Internet, ils servent d'intermédiaire pour aiguiller les requêtes vers les bons serveurs faisant autorité sur les zones recherchées.

Exemple de fichier named.root

```
;     This file holds the information on root name servers needed to
;     initialize cache of Internet domain name servers
;     (e.g. reference this file in the "cache  .  <file>"
;     configuration file of BIND domain name servers).
;
;     This file is made available by InterNIC
;     under anonymous FTP as
;          file                /domain/named.root
;          on server           FTP.INTERNIC.NET
;     -OR-                     RS.INTERNIC.NET
;
;       last update:    Jan 29, 2004
;       related version of root zone:   2004012900
;
;
; formerly NS.INTERNIC.NET
;
.                        3600000  IN  NS    A.ROOT-SERVERS.NET.
```

```
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
;
; formerly NS1.ISI.EDU
;
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     192.228.79.201
;
; formerly C.PSI.NET
;
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
;
; formerly TERP.UMD.EDU
;
.                        3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.      3600000      A     128.8.10.90
;
; formerly NS.NASA.GOV
;
.                        3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.      3600000      A     192.203.230.10
;
; formerly NS.ISC.ORG
;
.                        3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET.      3600000      A     192.5.5.241
;
; formerly NS.NIC.DDN.MIL
;
.                        3600000      NS    G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET.      3600000      A     192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
.                        3600000      NS    H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET.      3600000      A     128.63.2.53
;
; formerly NIC.NORDU.NET
;
.                        3600000      NS    I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET.      3600000      A     192.36.148.17
;
; operated by VeriSign, Inc.
;
.                        3600000      NS    J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET.      3600000      A     192.58.128.30
;
; operated by RIPE NCC
;
.                        3600000      NS    K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET.      3600000      A     193.0.14.129
;
; operated by ICANN
;
```

```
.                       3600000      NS    L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET.     3600000      A     198.32.64.12
;
; operated by WIDE
;
.                       3600000      NS    M.ROOT-SERVERS.NET.
M.ROOT-SERVERS.NET.     3600000      A     202.12.27.33
; End of File
```

8.4.2.b Fichier zone.localhost

La deuxième strophe de type zone du fichier *named.conf* mentionne un fichier que nous avons baptisé **zone.localhost**. Son rôle est de permettre la résolution du nom localhost en adresse 127.0.0.1. Elle précise que nous sommes serveur maître (type **master**) pour cette activité. Cette résolution est quasiment toujours faite grâce au fichier local */etc/hosts*. C'est simplement une sécurité de résoudre aussi cette correspondance au niveau du DNS.

Nous ne détaillons pas non plus immédiatement la syntaxe des enregistrements. Il nous suffit pour l'instant de comprendre le rôle du fichier.

Exemple de fichier zone.localhost

```
;
; loopback/localhost zone file
;
$TTL 1D
$ORIGIN localhost.
@              IN  SOA   @  root (
                         1   ; Serial
                         8H  ; Refresh
                         15M ; Retry
                         1W  ; Expire
                         1D) ; Minimum TTL
               IN   NS   @
               IN   A    127.0.0.1
```

8.4.2.c Fichier rev.127.0.0

Enfin, la troisième strophe de type zone du fichier *named.conf* mentionne un fichier que nous avons baptisé **rev.127.0.0**. Son rôle est de permettre la résolution inverse de l'adresse 127.0.0.1 en localhost. Elle précise que nous sommes serveur maître (type **master**) pour cette activité. Le nom de domaine virtuel **in-addr.arpa** est utilisé pour les résolutions de noms à partir des adresses IP. Nous verrons dans la suite de ce chapitre que l'on doit le faire précéder du début des adresses IP, écrites de la droite vers la gauche. Ceci implique la notation : *0.0.127.in-addr.arpa* comme nom de zone pour cette section.

Comme pour les deux fichiers précédents, nous ne détaillons pas encore la syntaxe des enregistrements. Il nous suffit pour l'instant de comprendre le rôle du fichier.

Exemple de fichier rev.127.0.0

```
;
; reverse pointers for localhost
;
$TTL 1D
$ORIGIN 0.0.127.in-addr.arpa.
@    IN   SOA  localhost. root.localhost. (
               1    ; serial
               8H   ; refresh
               15M  ; retry
               1W   ; expire
               1D ) ; minimum
     IN   NS   localhost.
1    IN   PTR  localhost.
```

8.4.3 Démarrage du serveur et premiers tests

Le démon *named* comporte deux options principales.

L'option **-u** permet d'indiquer l'identité sous laquelle doit s'exécuter le service et l'option **-c** permet d'indiquer l'emplacement du fichier de configuration.

Nous allons tenter le démarrage de notre serveur par la commande :

```
# named -u bind -c /usr/local/bind9/etc/named.conf
```

Il convient ensuite de vérifier que le démon *named* est bien en activité.

```
# ps -ef | grep named
bind  19018     1  0 13:45 ?      00:00:00  named -u bind
-c /usr/local/bind9/etc/named.conf
root  19053  3564  0 14:26 pts/0  00:00:00  grep named
#
```

Si ce n'était pas le cas, cela pourrait être du à une erreur de syntaxe dans le fichier *named.conf*. L'utilitaire **named-checkconf** permet de vérifier cette syntaxe.

Cela pourrait provenir également d'un problème de permissions pour le compte *bind* par rapport aux répertoires des fichiers de connexion.

Les outils clients d'interrogation

Les commandes **dig**, **nslookup** ou encore **host** permettent d'interroger les serveurs DNS.

La commande *dig* est l'outil de test officiel du logiciel *Bind*. Elle permet des interrogations plus détaillées et peut constituer aussi un outil de recherche d'erreurs.

Pour tester le bon fonctionnement de notre serveur avec ces commandes, nous devons devenir notre propre client.

Il suffit pour cela de placer la ligne **nameserver 127.0.0.1** en tête dans notre fichier /etc/resolv.conf.

Interrogation du serveur à l'aide des trois commandes de tests

```
# nslookup www.debian.org
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
Name:   www.debian.org
Address: 194.109.137.218

#


# dig www.debian.org

; <<>> DiG 9.4.1-P1 <<>> www.debian.org
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 41488
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 0

;; QUESTION SECTION:
;www.debian.org.                        IN      A

;; ANSWER SECTION:
www.debian.org.         678     IN      A       194.109.137.218

;; AUTHORITY SECTION:
debian.org.             678     IN      NS      raff.debian.org.
debian.org.             678     IN      NS      klecker.debian.org.
debian.org.             678     IN      NS      rietz.debian.org.

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Fri Oct 12 14:34:52 2007
;; MSG SIZE  rcvd: 109

#

# host www.debian.org
www.debian.org has address 194.109.137.218
www.debian.org mail is handled by 10 dummy.debian.org.
#
```

Nous constatons le bon fonctionnement de notre première configuration de serveur cache. L'affichage de la commande *nslookup* précise clairement que notre serveur ne fait pas autorité même si sa réponse est probablement correcte.

8.5 Pilotage du démon named, commande rndc

Le logiciel Bind en version 9 contient un utilitaire baptisé **rndc** (*remote named control*) qui permet d'utiliser des lignes de commande pour administrer le démon *named* à partir de l'hôte local ou d'un hôte distant. Cet utilitaire remplace la commande **ndc** des versions Bind 8.

Afin de prévenir les accès non autorisés au démon, le dialogue entre *rndc* et *named* doit se faire à l'aide d'une **clé secrète** partagée.

Une clé identique doit donc être décrite aussi bien dans le fichier *named.conf* que dans le fichier de configuration de rndc, à savoir **rndc.conf**.

L'utilitaire **rndc-confgen** permet de générer, sur sa sortie standard, les strophes adéquates à placer ensuite dans chacun des deux fichiers de configuration.

Exemple

```
# rndc-confgen > /tmp/buffer
# more /tmp/buffer
# Start of rndc.conf
key "rndc-key" {
        algorithm hmac-md5;
        secret "fxsjvAcMIj4HGpKxv/vX1g==";
};

options {
        default-key "rndc-key";
        default-server 127.0.0.1;
        default-port 953;
};
# End of rndc.conf

# Use with the following in named.conf, adjusting the allow list as
needed:
# key "rndc-key" {
#       algorithm hmac-md5;
#       secret "fxsjvAcMIj4HGpKxv/vX1g==";
# };
#
# controls {
#       inet 127.0.0.1 port 953
#               allow { 127.0.0.1; } keys { "rndc-key"; };
# };
# End of named.conf
#
```

Comme le montre assez clairement le résultat de la commande *rndc-confgen*, il faudra d'une part, compléter le fichier *named.conf* et, d'autre part, prévoir un fichier *rndc.conf* qui sera consulté lors de l'appel à la commande *rndc*.

Une déclaration **controls** doit être présente dans le fichier *named.conf* du serveur.

Dans l'exemple, La connexion avec rndc ne sera possible que depuis hôte local (directive **allow**). Nous constatons aussi que le port d'écoute est le port 953, uniquement sur 127.0.0.1 également (directives **inet** et **port**).

Enfin, la clause **keys** fait référence à une strophe de type *key* qui devra se trouver, complètement identique, dans le fichier rndc.conf.

Le lien entre le client *rndc* et le serveur *named* sera décrit par la strophe **options** du fichier rndc.conf.

La ligne de commande de rndc

Une commande **rndc** se présente sous le format suivant :

```
rndc <options> <commande> <options-commande>
```

Les options et commandes essentielles sont les suivantes :

Commande ou Option	**Effet**
halt	Arrêter le service *named*
querylog	Tracer toutes les requêtes
refresh	Rafraîchir la base de données
reload	Recharger les fichiers de zone en conservant toutes les réponses précédemment placées en cache
stats	Ecrire les statistiques courantes de *named* dans le fichier des statistiques (option *statistics-file*)
stop	Arrêter le serveur de manière propre, en enregistrant préalablement toute mise à jour ou transfert de zones.
status	Obtenir l'état du serveur.
-c fichier	Sélectionner le fichier de configuration à utiliser
-p numéro-port	Spécifier le numéro de port à utiliser
-s serveur	Envoyer les instructions à un serveur spécifique
-y nom-clé	Spécifier une clé autre que l'option *default-key*

	dans le fichier *rndc.conf*

Il n'est pas possible actuellement de redémarrer le service, la commande *restart* n'est pas encore implémentée.

Exemple d'interrogation du serveur par rndc (après l'avoir relancé)

```
# ps -ef | grep named
bind     19217     1  0 16:23 ?        00:00:00 named -u bind -c
/usr/local/bind9/etc/named.conf
root     19219  3564  0 16:23 pts/0    00:00:00 grep named
# kill 19217
# named -u bind -c /usr/local/bind9/etc/named.conf
# ps -ef | grep named
bind     19221     1  0 16:23 ?        00:00:00 named -u bind -c
/usr/local/bind9/etc/named.conf
root     19223  3564  0 16:23 pts/0    00:00:00 grep named
# rndc -c /usr/local/bind9/etc/rndc.conf status
number of zones: 13
debug level: 0
xfers running: 0
xfers deferred: 0
soa queries in progress: 0
query logging is OFF
recursive clients: 0/0/1000
tcp clients: 0/100
server is up and running
#
```

8.6 Configuration d'un serveur maître

Nous allons maintenant transformer notre serveur cache seulement en un serveur **maître** pour un domaine Intranet que nous allons baptiser **mondomaine.intra**.

Le serveur possède maintenant les informations officielles sur le domaine et fait autorité sur la zone concernée. Il charge en mémoire cache ses informations à partir de fichiers disques que nous devrons renseigner. Ce sera l'occasion de découvrir la syntaxe des enregistrements DNS.

Nous considérons toujours que notre serveur accède à l'Internet et qu'il va ainsi pouvoir réaliser des recherches sur le réseau mondial.

8.6.1 Fichier named.conf pour un serveur maître

Le fichier de configuration général conserve les strophes précédentes, nous allons simplement ajouter deux nouvelles strophes de type zone.

```
//
//  Exemple de serveur maitre pour le domaine "mondomaine.intra"
//  avec des machines dont les adresses commencent par 192.168.0
//

options {
  // emplacement des autres fichiers
  directory "/usr/local/bind9/var";
};

// Resolution des noms en adresses IP
zone "mondomaine.intra" IN {
  type master;
  file "zone.mondomaine.intra";
};

// Resolution inverse des adresses IP
zone "0.168.192.in-addr.arpa" IN {
  type master;
  file "rev.192.168.0";
};

// Serveur cache (hint) pour la racine Internet (.)
// le fichier named.root contient les coordonnees
// des serveurs officiels de la racine
zone "." IN {
  type hint;
  file "named.root";
};

// Serveur maitre pour la resolution de localhost en 127.0.0.1
zone "localhost" IN {
  type master;
  file "zone.localhost";
```

```
};

// Serveur maitre pour la resolution de 127.0.0.1 en localhost
zone "0.0.127.in-addr.arpa" IN {
  type master;
  file "rev.127.0.0";
};

// fin du fichier named.conf
```

8.6.2 Autres fichiers pour un serveur maître

Les trois fichiers associés aux trois strophes étudiées lors de la mise en œuvre du serveur cache seulement restent nécessaires au bon fonctionnement de notre futur serveur. Nous allons devoir mettre en place deux fichiers supplémentaires qui vont stocker les informations sur les machines du domaine dont nous allons devenir le serveur maître.

8.6.2.a Fichier zone.mondomaine.intra

La première strophe de type zone que nous avons ajoutée mentionne un fichier que nous avons baptisé **zone.mondomaine.intra**.

Son rôle est de permettre la résolution des noms en adresses IP. Elle précise que nous sommes serveur maître (type **master**) pour cette activité. Notre serveur fera autorité sur la zone et le fichier associé contiendra la liste centralisée de toutes les machines de notre domaine *mondomaine.intra*. Ce genre de fichier est souvent appelé *fichier de zone*.

Exemple de fichier zone.mondomaine.intra

```
;
; fichier de zone pour mondomaine.intra
;
$ORIGIN mondomaine.intra.
$TTL 1D
; any time you make a change to the domain, bump the
; "serial" setting below. the format is easy:
; YYYYMMDDI, with the I being an iterator in case you
; make more than one change during any one day
@     IN SOA   debian1 root (
                        200710191 ; serial
                        8H        ; refresh
                        4H        ; retry
                        4W        ; expire
                        1D )      ; minimum
; debian1.mondomaine.intra joue le rôle de NS et de MX
                NS      debian1
                MX      10  debian1
; quelques alias
www             CNAME   debian1
ftp             CNAME   debian1
```

```
smtp            CNAME   debian1
; noms des hosts du domaine stage1.intra
routeur         A       192.168.0.1
debian1         A       192.168.0.2
debian2         A       192.168.0.12
poste1          A       192.168.0.101
poste2          A       192.168.0.102
poste3          A       192.168.0.103
```

8.6.2.b Fichier rev.192.168.0

La deuxième strophe de type zone que nous avons ajoutée mentionne un fichier que nous avons baptisé **rev.192.168.0**.

Son rôle est de permettre la résolution inverse des adresses IP. Elle précise que nous sommes serveur maître (type **master**) pour cette activité. Notre serveur fera autorité sur la zone et le fichier associé contiendra les informations nécessaires pour retrouver les noms des machines dont l'adresse commence par 192.168.0. Il faut noter que si notre domaine comporte des machines dont l'adresse réseau ne commence pas par la même chose, il nous faudra créer un autre strophe de type zone avec un autre fichier associé. Ce genre de fichier est souvent appelé *fichier de résolution inverse*.

Exemple de fichier rev.192.168.0

```
;
; fichier de resolution inverse pour les adresses 192.168.0
;
$ORIGIN 0.168.192.in-addr.arpa.
$TTL 1D
; any time you make a change to the domain, bump the
; "serial" setting below. the format is easy:
; YYYYMMDDI, with the I being an iterator in case you
; make more than one change during any one day
@     IN SOA   debian1.mondomaine.intra. root.mondomaine.intra. (
                        200710191 ; serial
                        8H        ; refresh
                        4H        ; retry
                        4W        ; expire
                        1D )      ; minimum
                NS      debian1.mondomaine.intra.
; noms des hosts du domaine stage1.intra
1               PTR     routeur.mondomaine.intra.
2               PTR     debian1.mondomaine.intra.
12              PTR     debian2.mondomaine.intra.
101             PTR     poste1.mondomaine.intra.
102             PTR     poste2.mondomaine.intra.
103             PTR     poste3.mondomaine.intra.
```

8.6.3 Syntaxe des enregistrements RR (Resource Records)

8.6.3.a Syntaxe générale

Les fichiers de configuration comportent des enregistrements de ressources (**Resource Records : RRs**). Nous allons en donner ici une description résumée qui nous permettra de comprendre le contenu des fichiers précédents.

Le format général des *enregistrements RRs* est le suivant :

```
Name    TTL    Class    RecordType    RecordSpecificData
```

- **Name**

Nom du domaine DNS.

Si ce nom n'est pas renseigné, la valeur est celle héritée des lignes précédentes.

- **TTL**

Time To Live.

Il s'agit de la durée de vie de l'enregistrement dans la mémoire cache du serveur. Si ce champ n'est pas renseigné, on utilise une valeur par défaut définie en début de fichier par une ligne **$TTL**.

- **Class**

Famille de protocoles.

Ce champ est toujours égal à IN pour la famille de protocoles TCP/IP.

- **RecordType**

Type d'enregistrement.

Les principaux types d'enregistrement sont :

SOA	Start Of Authority (informations générales sur la zone)
NS	Name Server (serveur de noms officiel pour la zone)
MX	Mail Exchanger (serveur de messagerie)
A	Internet Address (pour la résolution nom->adresse IP)
CNAME	Canonical Name (alias)
PTR	Pointer (pour la résolution inverse : adresse IP -> nom)

- **RecordSpecificData**

Données associées au type d'enregistrement.

Le *fichier de zone* met en jeu des enregistrements **SOA**, **NS**, **MX**, **A** et **CNAME**.

Le *fichier de résolution inverse* met en jeu des enregistrements **SOA**, **NS** et **PTR**.

L'utilitaire **named-checkzone** permet une vérification de syntaxe pour ces fichiers.

8.6.3.b Enregistrement SOA

SOA Start Of Authority

Définition du début de la zone, premier enregistrement du fichier.

Ce type d'enregistrement est très important et doit débuter un fichier de zone ou un fichier de résolution inverse.

La syntaxe résumée est la suivante :

```
Zone      [ TTL ]   IN   SOA  Serveur   Contact  (
          Version
          Rafraîchissement
          Retentative
          Expiration
          Minimum
          )
```

Exemple

```
$ORIGIN mondomaine.intra.
$TTL 1D
@     IN  SOA   debian1   root  (
                        200710191 ; serial
                        8H        ; refresh
                        4H        ; retry
                        4W        ; expire
                        1D )      ; minimum
```

- **Zone**

Nom de la zone décrite par le fichier.

Le caractère @ désigne le nom de domaine courant. Si une variable **$ORIGIN** est présente en début de fichier, le caractère @ reprend cette valeur. En l'absence de la variable $ORIGIN, le caractère @ correspondra au nom de zone dans le fichier de configuration **named.conf**.

Dans tous les enregistrements RRs, un nom qui ne se termine pas par le caractère . (point) est systématiquement complété par le nom de domaine courant.

Par exemple, si @ contient la valeur *mondomaine.intra.* (avec le point final), nous pourrons désigner une machine par son nom simple tel que *debian1* (sans le point final). En effet, *debian1* sera correctement transformé en *debian1.mondomaine.intra.* (avec le point final). Par contre, si nous mentionnons le nom de machine *debian1.mondomaine.intra* (sans le point final), ce nom sera malheureusement complété en *debian1.mondomaine.intra.mondomaine.intra.* Il s'agit là d'une erreur fréquente dans les fichiers de ressources. Il convient d'être attentif à la valeur courante de @ ainsi qu'à la sémantique de l'enregistrement concerné pour savoir si l'on peut abréger ou non le nom d'une machine. On peut dire que si l'on décide de mentionner la machine sous son nom complet, il ne faut quasiment jamais omettre le point final.

Ensuite, le champ **TTL** est vide car la valeur par défaut est indiquée globalement par la variable $TTL. Le champ **IN** désigne la famille de protocoles TCP/IP, il peut être omis également.

Les données de l'enregistrement SOA sont :

- **Serveur**

Nom de la machine serveur.

- **Contact**

Adresse électronique du responsable de la zone.

Dans cette adresse, le caractère @ doit être remplacé par un point. Ainsi, la notation *root* va être, dans l'exemple ci-dessus, complétée en *root.mondomaine.intra.* et le point qui suit immédiatement le nom *root* signifiera @ comme il se doit dans dans une adresse e-mail.

- **Version** (serial)

Numéro de version.

Il s'agit d'un nombre entier qu'il faut incrémenter à chaque modification pour permettre les mises à jour d'éventuels serveurs esclaves. Ce nombre doit être constitué avec la syntaxe suivante : YYYYMMDDxx (YYYY = année, MM = mois, DD = jour, xx = numéro de version du jour)

- **Rafraîchissement** (refresh)

Périodicité avec laquelle les serveurs esclaves scrutent les mises à jour.

Les serveurs esclaves vérifient régulièrement si le numéro de version de la zone a été incrémenté. Dans l'affirmative, ils effectuent une mise à jour des données de la zone. Une durée de 8 heures semble constituer un bon choix générique.

- **Retentative** (retry)

Périodicité avec laquelle les serveurs esclaves retentent une mise à jour en cas d'échec. Une durée de 4 heures semble constituer un bon choix générique.

- **Expiration** (expire)

Durée de validité des informations pour les serveurs esclaves, indépendamment du numéro de version. Une durée de 2 à 4 semaines est souvent recommandée.

- **Minimum**

Durée de vie des réponses négatives.

Les réponses négatives provenant de serveurs faisant autorité sont conservés en mémoire cache pendant cette durée. Il est conseillé de ne pas choisir une valeur supérieure à une journée.

8.6.3.c Enregistrement NS

NS Name Server

Description d'un serveur faisant autorité (maître ou esclave).

Après l'enregistrement SOA, nous allons trouver un ou plusieurs enregistrements de type NS pour recenser les serveurs de noms officiels pour la zone concernée.

La syntaxe résumée est la suivante :

```
Zone        [ TTL ]    IN    NS    Serveur
```

Les données de l'enregistrement NS sont :

- **Serveur**

Nom de la machine serveur.

8.6.3.d Enregistrement A

A Address

Correspondance entre un nom et une adresse IP.

Chaque machine doit être décrite par un enregistrement de type A dans le *fichier de zone* qui recense de manière centralisée toutes les adresses des machines du domaine.

La syntaxe résumée est la suivante :

```
Nom         [ TTL ]    IN    A     Adresse
```

Les données de l'enregistrement A sont :

- **Nom**

Nom de la machine serveur.

- **Adresse**

Adresse IP correspondante.

8.6.3.e Enregistrement CNAME

CNAME Canonical Name

Définition d'alias.

Présents dans le *fichier de zone*, les enregistrements de type CNAME permettent de définir des alias associés à des noms officiels de machine. Des alias assez traditionnels sont *www* (serveur Web), *ftp* (serveur FTP) , *smtp* ou *mail* (serveur de messagerie), *pop* ou *imap* (serveur de remise de courrier) etc…

La syntaxe résumée est la suivante :

```
Alias      [ TTL ]    IN    CNAME      Nom
```

Les données de l'enregistrement NS sont :

- **Alias**

  Nom d'un alias.

- **Nom**

  Nom officiel d'une machine décrit par un enregistrement de type A.

8.6.3.f Enregistrement MX

MX Mail Exchanger

Association d'un nom de domaine à une machine serveur de messagerie.

Présents dans le *fichier de zone*, les enregistrements de type MX sont indispensables pour permettre aux utilisateurs de la messagerie de n'indiquer dans leur adresse e-mail que le nom d'un domaine, sans précision d'une machine particulière.

La syntaxe résumée est la suivante :

```
Zone       [ TTL ]    IN    MX    Priorité   Serveur
```

Les données de l'enregistrement MX sont :

- **Priorité**

  Priorité du serveur.

Il peut y avoir plusieurs enregistrements MX. Le serveur prioritaire est celui qui a la valeur la plus basse. Par tradition, on donne une succession de valeurs comme 10, 20, 30 etc… pour permettre d'intercaler facilement plus tard un nouveau serveur dans la liste. Si un serveur de messagerie ne peut être contacté, les messages sont envoyés au serveur suivant disponible, à charge pour celui-ci de réexpédier le courrier ensuite vers le serveur titulaire quand il redevient disponible.

- **Serveur**

  Nom officiel de la machine serveur de messagerie.

8.6.3.g Enregistrement PTR

PTR Domain Name Pointer

Résolution des traductions inverses : Adresses IP -> Noms

Présents dans le *fichier de résolution inverse*, les enregistrements de type PTR sont indispensables pour permettre la résolution inverse des adresses IPs.

La syntaxe résumée est la suivante :

```
Adresse    [ TTL ]    IN    PTR   Nom
```

Les données de l'enregistrement PTR sont :

- **Adresse**

    Fin de l'adresse IP inversée.

Cette fin d'adresse (dans le sens inverse : droite -> gauche) est concaténée à l'adresse réseau indiquée dans le nom de la zone courante, à savoir soit le contenu de la variable $ORIGIN, soit le nom de la strophe correspondante dans le fichier *named.conf*. Ces résolutions inverses s'appuient sur la zone virtuelle **in-addr.arpa**.

- **Nom**

    Nom complètement qualifié (avec le point final) de la machine correspondante.

8.6.4 Redémarrage du serveur et premiers tests

Nous allons demeurer notre propre client pour tester la nouvelle configuration de serveur maître.

Nous devons d'abord compléter notre fichier **/etc/resolv.conf** pour y indiquer (par une entrée *search*) que nous faisons maintenant partie du domaine *mondomaine.intra*. Ce nom de domaine sera ajouté aux noms de machines non complètement qualifiés. Ainsi, par exemple, la machine de nom *poste1* sera considérée comme étant le système de nom *poste1.mondomaine.intra*.

```
# cat /etc/resolv.conf
search mondomaine.intra
nameserver 127.0.0.1
#
```

Nous allons ensuite arrêter notre démon **named** (via la commande *rndc*) et le relancer pour qu'il tienne compte de la nouvelle configuration décrite dans le fichier *named.conf*.

```
# rndc -c /usr/local/bind9/etc/rndc.conf stop
# ps -ef | grep named
root  4401  3599  0 17:00 pts/0    00:00:00  grep named
# named -u bind -c /usr/local/bind9/etc/named.conf
# ps -ef | grep named
bind  4403  1     0 17:00 ?        00:00:00  named -u bind
-c /usr/local/bind9/etc/named.conf
root  4405  3599  0 17:01 pts/0    00:00:00  grep named
#
```

Les commandes **nslookup**, **dig** et **host** nous permettent d'interroger à nouveau le serveur DNS. Nous prendrons soin de tester aussi la résolution inverse.

```
# nslookup poste1
Server:         127.0.0.1
Address:        127.0.0.1#53

Name:   poste1.mondomaine.intra
Address: 192.168.0.101

# nslookup 192.168.0.101
Server:         127.0.0.1
Address:        127.0.0.1#53

101.0.168.192.in-addr.arpa      name = poste1.mondomaine.intra.

#
```

Nous constatons a priori le bon fonctionnement de notre configuration de serveur maître. L'affichage de la commande *nslookup* ne donne plus d'avertissement concernant l'autorité du serveur, ce qui signifie que la réponse obtenue est une réponse tout à fait officielle.

Pour la commande *dig*, il faut toujours indiquer un nom de machine complètement qualifié alors que c'est l'option *-x* qui demande une résolution inverse.

```
# dig debian2.mondomaine.intra

; <<>> DiG 9.4.1-P1 <<>> debian2.mondomaine.intra
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 1079
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;debian2.mondomaine.intra.      IN      A

;; ANSWER SECTION:
debian2.mondomaine.intra. 86400 IN      A       192.168.0.12

;; AUTHORITY SECTION:
mondomaine.intra.       86400   IN      NS      debian1.mondomaine.intra.

;; ADDITIONAL SECTION:
debian1.mondomaine.intra. 86400 IN      A       192.168.0.2

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Fri Oct 12 15:34:52 2007
;; MSG SIZE  rcvd: 96

# dig -x 192.168.0.12

; <<>> DiG 9.4.1-P1 <<>> -x 192.168.0.12
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46973
```

```
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 1

;; QUESTION SECTION:
;12.0.168.192.in-addr.arpa.     IN      PTR

;; ANSWER SECTION:
12.0.168.192.in-addr.arpa. 86400 IN     PTR     debian2.mondomaine.intra.

;; AUTHORITY SECTION:
0.168.192.in-addr.arpa. 86400   IN      NS      debian1.mondomaine.intra.

;; ADDITIONAL SECTION:
debian1.mondomaine.intra. 86400 IN      A       192.168.0.2

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Fri Oct 12 15:34:57 2007
;; MSG SIZE  rcvd: 119

#
```

La commande *host* nous donne aussi de bonnes informations.

```
# host routeur
routeur.mondomaine.intra has address 192.168.0.1
# host 192.168.0.1
1.0.168.192.in-addr.arpa domain name pointer routeur.mondomaine.intra.
#
```

Il est à noter que notre serveur maître du domaine *mondomaine.intra* continue à jouer le rôle de cache vis à vis des serveurs de la racine Internet puisque nous avons conservé la strophe nécessaire dans le fichier *named.conf*.

```
# nslookup www.debian.org
Server:         127.0.0.1
Address:        127.0.0.1#53

Non-authoritative answer:
Name:   www.debian.org
Address: 194.109.137.218

#
```

8.6.5 Ajout d'une machine dans le domaine

Une mise à jour dynamique pourrait s'envisager via un dialogue entre notre serveur DNS et un serveur DHCP qui lui annoncerait l'apparition de nouvelles machines membres du domaine.

Dans la pratique, cette mise à jour est peu ou pas pratiquée, en tout cas, en ce qui concerne des machines à la configuration statique, telles que des serveurs.

L'ajout d'une nouvelle machine dans le domaine consiste à enchaîner les étapes suivantes :

- ajouter un enregistrement de type A dans le fichier de zone,

- incrémenter le numéro de version dans l'enregistrement SOA de ce fichier de zone,

- ajouter un enregistrement de type PTR dans le fichier de résolution inverse correspondant,

- incrémenter le numéro de version dans l'enregistrement SOA de ce fichier de résolution inverse,

- recharger les informations en mémoire cache par un appel à la commande **rndc reload**.

8.7 Configuration d'un serveur esclave

Pour un domaine donné, il est important de configurer au moins un serveur de secours au cas où le serveur maître connaisse un gros problème. Ce serveur de secours est qualifié d'esclave dans le schéma fonctionnel DNS.

En complément des trois fichiers, déjà étudiés, qui existent toujours quel que soit le type de serveur, un serveur esclave stockera des fichiers disques contenant les informations sur les machines du domaine. Cependant, il faut bien comprendre que ces fichiers seront des copies de ceux de même rôle sur le serveur maître. Ils seront obtenus auprès de ce serveur maître, lors du démarrage de l'esclave et ils ne seront en aucun cas constitués manuellement.

8.7.1 Fichier named.conf pour un serveur esclave

Le fichier de configuration général d'un serveur esclave fait apparaître le type **slave** dans les strophes des zones concernées. Une directive **masters** fait apparaître l'adresse IP du serveur maître qu'il faudra contacter pour obtenir une copie du fichier de même rôle. Nous décidons de donner à ce fichier, obtenu auprès du maître, le préfixe *slave*.

```
//
//  Exemple de serveur esclave pour le domaine "mondomaine.intra"
//  avec des machines dont les adresses commencent par 192.168.0
//

options {
  // emplacement des autres fichiers
  directory "/usr/local/bind9/var";
};

// Resolution des noms en adresses IP
zone "mondomaine.intra" IN {
  type slave;
  masters { 192.168.0.2 ; } ;
  file "slave.zone.mondomaine.intra";
};

// Resolution inverse des adresses IP
zone "0.168.192.in-addr.arpa" IN {
  type slave;
  masters { 192.168.0.2 ; } ;
  file "slave.rev.192.168.0";
};

// Serveur cache (hint) pour la racine Internet (.)
// le fichier named.root contient les coordonnees
// des serveurs officiels de la racine
zone "." IN {
  type hint;
  file "named.root";
};
```

```
// Serveur maitre pour la resolution de localhost en 127.0.0.1
zone "localhost" IN {
  type master;
  file "zone.localhost";
};

// Serveur maitre pour la resolution de 127.0.0.1 en localhost
zone "0.0.127.in-addr.arpa" IN {
  type master;
  file "rev.127.0.0";
};

// fin du fichier named.conf
```

En plus de ce fichier de configuration sur le serveur esclave, il faudra opérer quelques légères modifications sur le serveur maître :

- ajouter un enregistrement de type **NS** qui mentionnera le nouveau serveur esclave, à la fois dans le fichier de zone et dans le fichier de résolution inverse,

- incrémenter le numéro de version de ces deux fichiers,

- autoriser le serveur esclave à recopier les informations de zone à l'aide de directive **notify** et **allow-transfer** dans les strophes du fichier *named.conf*.

Exemple (si l'esclave a l'adresse 192.168.0.12)

```
zone "mondomaine.intra" IN {
  type master;
  file "zone.mondomaine.intra";
  notify yes ;
  allow-transfer { 192.168.0.12 ; } ;
};


zone "0.168.192.in-addr.arpa" IN {
  type master;
  file "rev.192.168.0";
  notify yes ;
  allow-transfer { 192.168.0.12 ; } ;
};
```

9. Serveur FTP

9.1 Un serveur FTP pour quoi faire ?

FTP (File Transfer Protocol) permet de transférer des fichiers selon un modèle client/serveur.

Le transfert peut s'automatiser (fichier .netrc) et les commandes FTP peuvent se trouver dans un shellscript selon le principe du heredocument (balise <<). Il est ainsi possible de lancer des scripts de transferts de fichiers dans la crontab.

La variante sécurisée de FTP est FTPS et elle s'appuie sur SSL. A ne pas confondre avec le client sftp qui lui dialogue avec un serveur sshd (plugin sftp-open-server par exemple).

9.2 Vocabulaire et fonctionnement de FTP

FTP peut s'utiliser de deux façons différentes :

Mode actif (par rapport au client) :

C'est le client FTP qui détermine le port de connexion à utiliser pour permettre le transfert des données. Ainsi, pour que l'échange des données puisse se faire, le serveur FTP initialisera la connexion de son port de données (port 20) vers le port spécifié par le client. Le client devra alors configurer son pare-feu pour autoriser les nouvelles connexions entrantes afin que l'échange des données se fasse. De plus, il peut s'avérer problématique pour les utilisateurs essayant d'accéder à des serveurs FTP lorsqu'ils sont derrière une passerelle NAT. Étant donnée la façon dont fonctionne le NAT, le serveur FTP lance la connexion de données en se connectant à l'adresse externe de la passerelle NAT sur le port choisi. Certaines passerelles NAT n'ayant pas de correspondance pour le paquet reçu dans la table d'état, le paquet sera ignoré et ne sera pas délivré au client.

Mode passif (par rapport au client):

Le serveur FTP détermine lui-même le port de connexion à utiliser pour permettre le transfert des données (data connexion) et le communique au client. En cas de présence d'un pare-feu devant le serveur, celui-ci devra être configuré pour autoriser la connexion de données. L'avantage de ce mode, est que le serveur FTP n'initialise aucune connexion. Ce mode fonctionne sans problèmes avec une passerelle NAT. Dans les nouvelles implémentations, le client initialise et communique directement par le port 21 du serveur; cela permet de simplifier les configurations des pare-feu serveur.

Deux ports sont standardisés pour les connexions FTP : le port 21 pour les commandes et le port 20 pour les données.

Le mode de transfert

Deux modes de transfert peuvent être utilisés. Il est important de bien comprendre ce paramètre pour éviter de se retrouver avec des fichiers corrompus (classique entre windows et UNIX).

Mode Binaire :

Le fichier est transmis tel quel (il a donc exactement la même taille de chaque côté).

Mode ASCII :

uniquement destiné aux fichiers texte. Le fichier est examiné et des transformations apportées pour conserver un format correct. Par exemple, la fin de ligne est représentée par le caractère <LF> sur un système UNIX, et par la paire <CR><LF> sous Windows. Une machine Windows recevant un fichier texte par FTP récupère donc au final un fichier avec des <CR><LF> en mode ASCII et des <LF> en mode binaire. Ce mode a donc ses avantages, mais peut être source de corruption de fichiers (non texte) pendant le transfert si on utilise un client ancien / en ligne de commande, incapable de s'adapter au type de fichier. Il faut alors basculer de mode (en utilisant généralement la commande BIN) avant le transfert, afin de le conserver intact.

```
ftp> status
Connecté à n196.nerim-hosting.net.
Type : ascii ; …
ftp> bi
200 TYPE est maintenant 8-bit binary
ftp> status
Connecté à n196.nerim-hosting.net.
Type : binary ; …
```

Il faut donc penser à activer l'option ascii pour transférer un fichier texte entre deux plateformes utilisant un jeu de caractères différents (comme UNIX et Windows) et surtout pas pour un transfert non texte (.doc, .mp3, .pdf, …).

Sous UNIX la commande **od** permet de visualiser ce type de problème et la commande **dos2unix** de le « réparer ». La séquence CRLF (Carriage Return + LineFeed) n'est donc pas un caractère ASCII mais l'association de 2 caractères ASCII, le caractère 13 (0x0D) suivi du caractère 10 (0x0A).

```
od -c fichier
0000000   U   I   D                       P   I   D
0000020   P   P   I   D                   C   M   D
0000040   S   I   Z   E  \r  \n   r   o   o   t                   1
0000060                        0                                i   n
0000100   i   t                4   6   8  \r  \n   r   o   o   t
…
```

```
head fichier
PID     PPID    CMD     SIZE
root    1       0       init    468
root    388     1       sh      1228
…
```

```
grep 'SIZE$' fichier
```

Aucun résultat car la première ligne ne se termine pas par SIZE mais par SIZE **et** un caractère invisible valant Carriage Return (13 en ascii) !

Résumé des principales commandes FTP

help donne la liste des commandes disponibles

```
ftp> help ls
ls              Liste le contenu du dossier distant)
```

open permet de se connecter à un serveur ftp : open nom_ou_@IP

user si la machine ne vous demande pas de taper immédiatement un nom d'utilisateur (login) vous pouvez taper la commande : user login puis votre mot de passe ; ou user login motdepasse directement.

get pour récupérer un fichier : get source [destination]

put pour envoyer un fichier vers le serveur ftp : put source [destination]

ls pour lister tous les fichiers du répertoire courant **sur le serveur**

cd pour changer de répertoire sur le serveur

!cmd pour exécuter une commande sur votre machine **en local**

lcd pour changer de répertoire en local sur votre machine (équivalent de : !cd)

bin pour passer en mode binaire, indispensable avant d'envoyer (put) ou de récupérer (get) des images par exemple, vous devez avoir la réponse : 200 Type set to I.

ascii pour passer en mode ascii, vous devez avoir la réponse : 200 Type set to A.

prompt Pour passer du mode manuel au mode automatique et inversement (très pratique pour mget et mput pour éviter de confirmer à chaque fichier et donc obligatoire dans le cadre d'un script de transfert automatisé)

mget permet de récupérer plusieurs fichiers à la fois. exemples : mget *.htm , ou mget toto.htm toto1.htm (get ne gère pas les * contrairement à la version de get dans sftp)

mput permet de d'envoyer plusieurs fichiers à la fois. exemples : mput *.htm , ou mput toto.htm toto1.htm

close ferme la connexion (open permet d'en ouvrir une autre sans sortir de l'application FTP)

quit sort de l'application ftp

9.3 Utilisation du serveur FTP standard sous Solaris

Sous SOLARIS le daemon correspondant au service FTP fourni en standard est in.ftpd

Obtenir l'état u service ftp :
```
inetadm | grep ftp
```

Activer le serveur ftp :
```
inetadm -e ftp
```

Désactiver le serveur ftp :
```
inetadm -d ftp
```

Obtenir de l'aide :
```
man inetadm
man svcadm
```

9.4 Installation d'un serveur PROFTPD sous AIX

Au-delà de ces fonctionnalités, performances, et stabilité proftpd est un serveur FTP open source qui présente l'avantage de disposer d'un fichier de configuration identique à celui d'Apache (httpd.conf).

Le serveur FTP historique d'IBM est plus limité dans ses capacités de configuration notamment dans l'utilisation d'environnements chrootés ou encore le virtual hosting.

Le paramétrage de proftpd est commun à toutes les plateformes.

1 Téléchargez coreutils et proftpd depuis le site d'IBM
Dans notre exemple :
http://www-03.ibm.com/systems/p/os/aix/linux/toolbox/download.html
(ou récuperer le CD AIX Toolbox for Linux)

2 Installez coreutils-5.2.1-2.aix5.1.ppc.rpm et proftpd-1.2.8-1.aix5.1.ppc.rpm

```
rpm -Uvh coreutils-5.2.1-2.aix5.1.ppc.rpm
rpm -Uvh proftpd-1.2.8-1.aix5.1.ppc.rpm
```

3 Personnaliser le fichier de configuration /etc/proftpd.conf, les groupes/comptes utilisateurs, et l'environnement système (répertoires, droits)

4 modifiez le fichier /etc/inetd.conf en remplaçant ftp par proftp

```
#ftp     stream  tcp6    nowait  root    /usr/sbin/ftpd          ftpd
ftp     stream  tcp     nowait  root    /usr/sbin/proftpd        proftpd
```

5 Relancez le service inetd:

```
refresh -s inetd
```

9.5 Installation d'un serveur PROFTPD sous Solaris

A partir du package …

```
pkg-get install proftpd
```

(éventuellement faire un pkg-get upgrade au préalable)

Des nouveaux paquets seront installés, tels que OpenSSL pour le cryptage des transmissions entre le serveur FTP et les clients.

Message ontenu à la fin de l'installation :

```
[ vérification de la classe <none> en cours...]

L´installation de <CSWproftpd> a abouti.
```

Fichier de configuration : /opt/csw/etc/proftpd.conf.CSW

A partir des sources …

```
cd /tmp
wget ftp://ftp.proftpd.org/distrib/source/proftpd-1.3.0a.tar.gz
gunzip -c proftpd-1.3.0a.tar.gz ¦ tar xvf –
cd proftpd-1.3.0a
./configure      ;     make       ;     make install
```

Fichier de configuration : /usr/local/etc/proftpd.conf

Démarrage de proftpd

```
/usr/local/sbin/proftpd
```

9.6 Paramétrage de PROFTPD

Création (éventuelle) du répertoire FTP

```
mkdir /home/ftp
```

L'utilisateurt limité « nobody »

Par défaut le propriétaire du processus proftpd est root. Il est donc conseillé d'associer au service FTP un compte sans droit particulier. La plupart des systèmes utilisent le compte « nobody » du groupe « nogroup ». L'un et l'autre peuvent déjà exister sur votre système.
Rappels (fichiers des comptes = /etc/passwd et fichiers des groupes = /etc/group)

```
groupadd nogroup
useradd nobody -d / -s /bin/false
usermod -g nogroup nobody
```

usermod permet ici d'associer l'utilisateur "nobody" au group "nogroup" existant

Les autres utilisateurs

Beaucoup de scénarios sont possibles. Nous choisissons ici de créer un compte avec pouvoir pour lire et écrire sur le site (admftp) et un autre qui ne pourra que lire (userftp). Ni l'un ni l'autre ne pourront disposer d'un shell. Enfin un compte anonymous permettra de se connecter sans mot de passe.

L'utilisateur administrateur ayant les droits d'écriture:

```
groupadd ftp
useradd admftp -s /bin/false
usermod -g ftp admftp
passwd admftp
```

L'utilisateur qui aura un accès en lecture seulement:

```
useradd -g ftp -s /bin/false  userftp
passwd userftp
```

L'utilisateur anonyme qui aura un accès en lecture seulement, si il n'existe pas déjà:

```
useradd -g ftp -u 21 -s /bin/false -d /home/ftp ftp
```

Certains systems peuvent ne pas "apprécier" que le compte ftp ne corresponde pas à l'userid 21.

Mise en place des droits et de la sécurité générale

```
chgrp ftp /home/ftp -R
chmod 775 /home/ftp -R
ls -ld /home/ftp
drwxrwxr-x. 4 ftp ftp 4096 mai 24 13:56 /home/ftp
```

Le "shell" /bin/false correspond à un simple binaire qui retourne 1. Ceci afin d'éviter de se connecter avec un shell classique. Mais il faut tout de même que ce programme soit « validé »comme faisant partie des programmes de connection potentiels. C'est le rôle du fichier /etc/shells.

```
echo "/bin/false" >> /etc/shells
```

Le fichier /etc/ftpusers déclare tous les utilisateurs **dont on interdit l'accès** au service FTP. On peut le générer avec la commande :

```
cat /etc/passwd | cut -d":" -f1 > /etc/ftpusers (
```

puis le personnaliser)

Le fichier de configuration /etc/proftpd.conf

Comme toujours il est préférable d'en faire une copie avant toute modification.

`cp /etc/proftpd.conf /etc/proftpd.conf.origine`

```
#-----------------------------------------------------------------------
#DIRECTIVE SYSTEMES   => OBLIGATOIRES!
#-----------------------------------------------------------------------
ServerType              standalone
DefaultServer           on
AllowStoreRestart       on
Port                    21
User                    nobody
Group                   nogroup


#-------------------------------------------------------------------------------
#DIRECTIVE ADMINISTRATEUR        => FACULTATIVES
#-------------------------------------------------------------------------------
ServerName              "Serveur ProFTPD de formation "
LogFormat               format_1  "[%{%d/%m/%y %H:%M:%S}t]-[IP=%a]-[LOGIN=%U(%u)]-
[CDE_CLT=%r]-[REP_FTP=%s]-[DIR=%D]"
LogFormat               format_2  "[%{%d/%m/%y %H:%M:%S}t]-[IP=%a]-[LOGIN=%U]-
[CDE_CLT=%r]-[REP_FTP=%s]-[FICHIERS=%f]-[NB_OCT=%b]"
ExtendedLog             /var/log/proftpd/LOG_PRINCIPAL_FTP.log AUTH,INFO,DIRS
format_1
ExtendedLog             /var/log/proftpd/LOG_TRANSFERT_FTP.log READ,WRITE
format_2
UseFtpUsers             on
AllowOverwrite          on


#-----------------------------------------------------------------------------------------------
#DIRECTIVES SECURITAIRES/RESTRICTIVES     =>FACULTATIVES
#-----------------------------------------------------------------------------------------------
DefaultRoot             /home/ftp
MaxClients              30  « Nombre de connexions limites atteintes : '%m' "
MaxInstances            30  #Nombre de processus maximum
AccessDenyMsg           "Authentification est invalide !!"
AccessGrantMsg            "Bienvue \"%u\" sur le serveur FTP de …"
TimeoutIdle             300
<Directory /home/ftp>
   <Limit WRITE>
     AllowUser          admftp
     DenyAll
   </Limit>
</Directory>
<Limit LOGIN>
    Allow             192.168.0. 127.0.0.1
    DenyAll
</Limit>
```

```
#-----------------------------------------------------------------
#DIRECTIVES ANONYMES   =>FACULTATIVES
#-----------------------------------------------------------------
<Anonymous /home/ftp>
  User                ftp
  Group               ftp
  AnonRequirePassword  on
  RequireValidShell    off
  UserAlias           anonymous ftp
  UserAlias           personne ftp
  UserAlias           anonyme ftp
  <Limit WRITE>
    DenyAll
 </Limit>
</Anonymous>
```

Lancement du service

`/usr/sbin/proftpd start`

(si installation à partir des sources)

Tests et vérifications

Test de présence :

```
ps -ef | grep ftp
nobody    2705     1  0 14:07 ?        00:00:00 proftpd: (accepting connections)
```

Test du serveur FTP :

```
ftp localhost
```

Ecoute réseau :

```
netstat -natup | grep :21
```

Connections en cours :

```
tail -f /var/log/proftpd/LOG_PRINCIPAL_FTP.log
```

Transferts en cours : (pensez aussi à /proc même si plus brut)

```
tail -f /var/log/proftpd/LOG_TRANSFERT_FTP.log
```

9.7 Automatisation de transfert avec FTP

Le principe du “here document” consiste à “insérer” un autre langage que le shell dans un shellscript (ici des commandes FTP). Le shell s’y « retrouve » grâce au symbole << suivi d’un mot qui fait office de borne d’arrêt. Tout ce qu’il trouve entre le symbole << et ce mot sera interprété mais pas exécuté. On pourrait ainsi trouver des commandes SQL, mails, sed, …

Le mot TOT comme Termination Of Treatment est totalement libre.

```
cat transfert
HOST=$1
FILE=$2
echo DEBUT DU TRANSFERT SUR $1
echo TRANSFERT DU FICHIER $FILE
ftp $HOST << TOT
binary
prompt
runique
get $FILE
bye
TOT
echo FIN DU TRANSFERT

transfert webserv /racine/images/new.gif
```

Pour fonctionner correctement il faut créer un fichier .netrc (sous son home directory) pour automatiser la phase login/password. L’argument passé à ftp est donc une sorte de « goto » dans ce fichier.

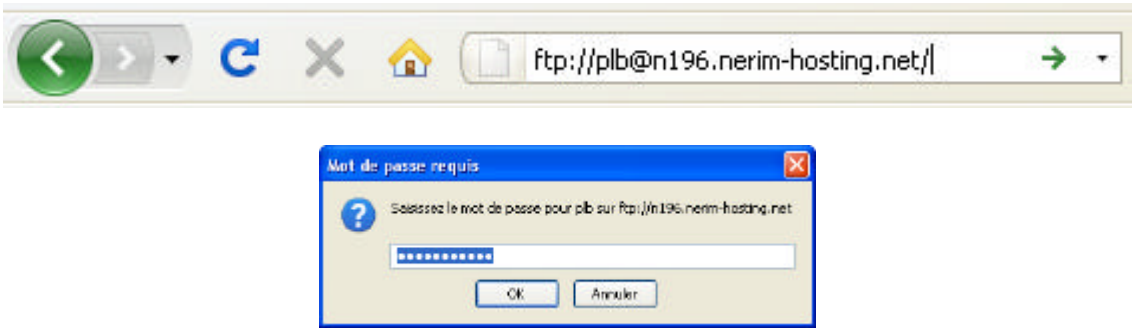
```
cat ~/.netrc
machine webserv         login laurent     password mot_de_passe_en_clair
machine www.site.fr     login michel      password mot_de_passe_en_clair
```

Si les droits sont trop ouverts la session ne s’ouvrira pas :

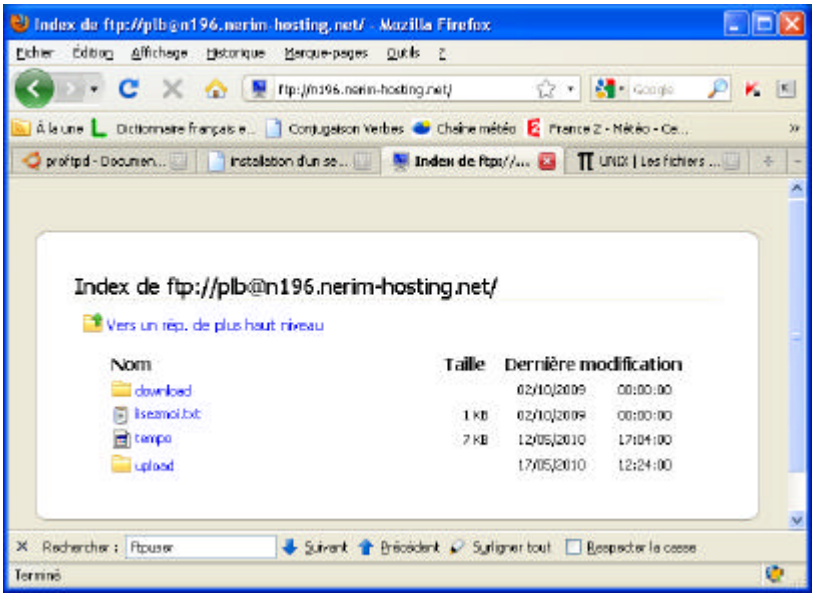
```
chmod 600 .netrc
```

9.8 Clients FTP inclus dans les navigateurs

Il est possible de lancer une session FTP depuis firefox, IE, …
Il suffit de le préciser dans l'URL comme ici :



On peut aussi saisir le mot de passe dans l'URL
`ftp://plb:password@n196.nerim-hosting.net/`



10. Le serveur NFS de partage de fichiers

NB: Nous prenons comme exemple, dans ce chapitre, l'implémentation de NFS dans les distributions Linux Debian.

10.1 Principe de fonctionnement

NFS (*Network File System*) est un service TCP/IP, conçu chez Sun, qui s'est imposé comme standard de fait pour le partage de fichiers entre systèmes Unix.

NFS fait partie intégrante d'un système Unix mais est également disponible, sous forme de logiciel, sur différents systèmes d'exploitation. Son utilisation est donc possible en environnement hétérogène avec, par exemple, un serveur Unix et des clients Microsoft Windows.

Cependant, on lui préfère souvent, dans ce cas, le logiciel libre **Samba** qui évite le coût d'acquisition de licences NFS pour les clients Microsoft. De plus, Samba permet de partager également des imprimantes, ce qui n'est pas le cas pour NFS.

NFS définit un modèle virtuel de système de fichiers qui est mis en correspondance avec la sémantique du système de fichiers local.

Concrètement, le **serveur** NFS donne l'autorisation d'accès à certaines arborescences. On parle de **partage** ou d'**exportation** de ressources.

Le **client** NFS accède à la ressource partagée via un **montage** dans son arborescence locale. Ainsi, l'accès aux fichiers du serveur est totalement transparent pour les utilisateurs.

Le service NFS s'appuie sur la bibliothèque dite **RPC** (*Remote Procedure Call*). Cette bibliothèque est d'ailleurs utilisée par beaucoup d'autres applications réseau.

A cause de cette implémentation, NFS ne peut fonctionner qu'en présence d'un processus démon indispensable baptisé **portmap**. Ce démon portmap a en charge la mise en correspondance des numéros de programmes RPCs avec des numéros de ports TCP/IP.

Le démon portmap est fourni via le paquet logiciel de même nom.

Le service portmap correspond au numéro de programme RPC 100000 et aux ports TCP et UDP 111. La commande **rpcinfo** permet d'interroger le démon pour connaître les services RPCs en activité.

Le serveur NFS est intégré au noyau Linux sous forme de module. Le paquet logiciel **nfs-kernel-server** fournit cette possibilité.

Le paquet **nfs-common** vient compléter l'implémentation en fournissant notamment quelques outils clients.

10.2 Serveur NFS

Le fichier **/etc/exports** doit décrire les partages proposés sur le serveur NFS.

Sa présence est détectée par le script de démarrage et provoque la mise en œuvre effective des partages par la commande **exportfs**.

Elle déclenche également le lancement des démons nécessaires parmi lesquels :

- *rpc.mountd* pour la gestion des requêtes de montages,

- *rpc.statd* pour la gestion des procédures de reprise,

- *rpc.lockd* pour la gestion des accès concurrents,

- *rpc.nfsd* en tant que démon principal (lancé en plusieurs exemplaires, sous forme de threads).

Rappelons que le lancement de ces démons nécessite la présence initiale du service *portmap*.

Un partage est associé à une liste de machines autorisées à y accéder.

Des options de partage permettent de fixer les règles d'accès.

Chaque ligne du fichier /etc/exports correspond à un répertoire partagé, suivi d'une liste de machines clientes autorisées à utiliser le partage. Les noms de machines seront séparés par des espaces.

Chaque client de la liste peut être suivi par une liste d'options de partage pour ce client. Ces options seront exprimées entre parenthèses, séparées par des virgules. Il ne doit pas y avoir d'espaces entre un nom de client et sa liste d'options.

Les lignes vides sont ignorées. Le caractère # (dièse) indique un commentaire qui s'étend jusqu'à la fin de la ligne.

Les entrées peuvent s'étendre sur plusieurs lignes en utilisant la barre oblique inverse (*antislash*).

Les clients NFS peuvent être exprimés de plusieurs façons :

- un nom DNS (abrégé ou complet) ou une adresse IP de machine individuelle,

- un nom de groupes de machines tel que gérés dans le service système *NIS*,

- un ensemble de noms de machines en utilisant les caractères jokers * et ?,

- une adresse de réseau ou de sous-réseau en notation *adresse IP/ masque*

La commande **exportfs** exploitera donc ce fichier, elle accepte un certain nombre d'options parmi lesquelles :

- **ro** (option par défaut)

Partage en lecture seule

- **rw**

Partage en lecture/écriture

- **secure** (option par défaut)

Connexion imposé depuis un port réservé à *root* (inférieur à 1024). L'option inverse **insecure** annule cet effet.

- **async**

Amélioration des performances en permettant au serveur NFS de répondre aux requêtes avant que tous les changements impliqués par la requête n'aient été validés sur le disque.

- **sync**

Le serveur NFS ne répond aux requêtes que lorsqu'elles ont pu être validées sur le disque. La valeur par défaut de l'option a évolué au cours des versions NFS, il est donc recommandé de la spécifier explicitement.

- **root_squash** (option par défaut)

Le compte *root* du client NFS perd son identité et se retrouve considéré comme un **utilisateur anonyme** (compte prédéfini *nobody*). L'option inverse **no_root_squash** annule cet effet et le compte *root* du client NFS conserve alors son identité.

Les options complémentaires *anonuid=uid* et *anongid=gid* permettent d'indiquer un utilisateur plus ciblé comme utilisateur anonyme.

L'option *all_squash* transforme tous les utilisateurs clients en utilisateurs anonymes.

Exemple de fichier /etc/exports

```
/              master(rw) trusty(rw,no_root_squash)
/projects      proj*.local.domain(rw)
/usr           *.local.domain(ro) @trusted(rw)
/home/joe      pc001(rw,all_squash,anonuid=150,anongid=100)
/pub           (ro,insecure,all_squash)
```

Dans cet exemple, la première ligne exporte l'ensemble du système de fichiers vers les machines *master* et *trusty*. En plus des droits d'écriture, toute transformation d'UID est désactivée pour l'hôte trusty.

Les deuxième et troisième lignes montrent des exemples de noms de machines exprimés avec des caractères jokers et des groupes de machines (@trusted).

La quatrième ligne montre une entrée pour un client NFS de type PC.

La dernière ligne partage un répertoire public pour toutes les machines dans le monde, en effectuant les requêtes sous le compte anonyme. L'option *insecure* permet l'accès aux clients dont l'implémentation NFS n'utilise pas un port réservé.

Exemples de commandes exportfs explicites

```
# exportfs *:/usr/share/man
# exportfs -v
/usr/share/man  <world>(ro,wdelay,root_squash,anonuid=65534,anongid=65534)
#
```

Dans cet exemple, nous exportons le répertoire */usr/share/man* pour toutes les machines dans le monde. Nous remarquons que le partage a lieu par défaut en lecture seule (*ro*) avec le comportement sous-jacent d'utilisateur anonyme (*root_squash*) associé au compte *nobody* (*anonuid* et *anongid* associé à l'UID 65534).

```
# exportfs -o rw,no_root_squash 192.168.0.0:/export/home
# exportfs -v
/export/home   192.168.0.0(rw,wdelay,no_root_squash,anonuid=65534,anongid=65534)
/usr/share/man  <world>(ro,wdelay,root_squash,anonuid=65534,anongid=65534)
#
```

Dans cet autre exemple, nous exportons le répertoire /export/home en lecture/écriture (*rw*) pour toutes les machines du réseau local 192.168.0. Le compte *root* des machines clientes gardera son identité vis à vis de la ressource partagée (*no_root_squash*).

10.3 Client NFS

La commande **mount** permet de réaliser l'aspect client NFS.

La sémantique du montage reste la même que pour un système de fichiers local. Il faut disposer d'un répertoire de montage vide dans lequel vient se placer l'arborescence distante.

Au lieu d'indiquer une partition ou volume logique comme premier argument, il faut mentionner ici le nom de la machine serveur suivi du nom du répertoire partagé. Les deux informations doivent être reliées par le caractère **:** (deux points).

Des options de montage pourront, bien entendu, être ajoutées.

Exemple

```
mount  -t nfs  -o rw debian1:/export/home  /export/home
```

Dans cet exemple, le contenu du répertoire */export/home* situé sur le serveur NFS de nom *debian1* devient accessible via le répertoire local */export/home*. Ce répertoire local est physiquement vide mais doit exister sur la machine cliente. Le montage a lieu en lecture/écriture, si le partage le permet.

Pour obtenir des **montages permanents**, il convient simplement de les intégrer au fichier administratif **/etc/fstab**.

Le fichier /etc/fstab donne la liste de tous les montages à effectuer au démarrage du système ou à exécuter manuellement (par exemple, pour certains périphériques amovibles).

Chaque point de montage y est détaillé sur une ligne par plusieurs champs séparés par des espaces.

Les montages NFS viendront s'ajouter aux montages locaux.

Exemple de ligne dans /etc/fstab pour un montage NFS

```
debian1:/export/home  /export/home  nfs  rw,nosuid  0  0
```

Les 4 premiers champs des lignes du fichier /etc/fstab désignent, dans l'ordre :

- le nom de la ressource partagée sur le serveur NFS,

- le répertoire de montage sur le client,

- le type de montage (nfs),

- les options de montage (liées pour certaines à la sémantique NFS).

Les deux derniers champs ne sont pas significatifs pour un montage NFS, on a l'habitude de leur donner la valeur 0.

Les options de montage NFS ainsi que la syntaxe des lignes correspondantes dans le fichier */etc/fstab* sont détaillées dans la page de manuel *man nfs*.

Parmi les nombreuses options de montage, nous pouvons citer :

- **rw**

Monter la ressource en lecture/écriture si le partage l'autorise. Il s'agit de l'option par défaut.

- **ro**

Monter la ressource en lecture seule.

- **hard**

Recommencer la requête de montage jusqu'au succès de l'opération. Le type *hard* est nécessaire dans le cas d'un montage en lecture/écriture. Il s'agit de l'option par défaut.

- **soft**

Abandonner la requête de montage en cas d'échec.

- **fg**

Les nouvelles tentatives de montage de type *hard* ont lieu en premier plan. Il s'agit de l'option par défaut.

- **bg**

Les nouvelles tentatives de montage de type *hard* ont lieu en arrière plan.

- **intr**

Autoriser les interruptions clavier pour mettre fin à une tentative de montage de type *hard*. Il s'agit de l'option par défaut.

- **nointr**

Ne pas autoriser les interruptions clavier pour mettre fin à une tentative de montage de type *hard*.

- ***nosuid***

La permission Unix *suid* n'est plus opérationnelle pour les programmes de la ressource montée.

Une combinaison appropriée des options doit provenir d'une réflexion par rapport au rôle de la ressource NFS dans le scénario du démarrage de la machine cliente. Par exemple, en partant de l'obligation du type *hard* (c'est à dire bloquant) pour tout montage en lecture/écriture, il faut se demander si un tel montage est indispensable au bon déroulement du démarrage. Si ce n'est pas le cas, une retentative en arrière plan (option *bg*) est recommandée pour pouvoir régler le problème plus tard sans avoir rendu la procédure de démarrage bloquante.

Dans tous les cas, des messages d'erreur répétitifs à la console d'une machine cliente NFS lors de son démarrage doivent rapidement faire penser que le problème se situe sur une autre machine, en l'occurence le serveur NFS. Assez souvent, il convient de s'assurer que celui-ci est bien démarré et que les démons nécessaires sont lancés. Quand tout redevient normal au niveau du serveur, les montages de la machine cliente se débloquent à leur tour.

10.4 Automontage

L'automontage est une amélioration de l'aspect client de NFS.

Parmi d'autres, le paquet logiciel **autofs** offre cette fonctionnalité.

Le montage de la ressource souhaitée aura lieu uniquement au moment où le répertoire de montage est sollicité dans une commande.

De plus, la ressource sera démontée lorsqu'elle n'est plus utilisée pendant un certain temps.

Pour obtenir ce traitement, on ne mentionne plus les ressources dans le fichier /etc/fstab mais on utilise, en remplacement, un démon *automounter* qui consulte ses propres tables.

Contrairement aux montages NFS classiques, le répertoire de montage n'existe pas sur la machine cliente. Il est créé dynamiquement par l'*automounter* et il disparaît lors du démontage automatique.

Une configuration Unix très classique consiste à centraliser sur des serveurs la définition des comptes par un service d'annuaire tel que NIS (*Network Information Service*). Dans le même temps, les répertoires de connexion sont également centralisés sur un serveurs NFS (sans doute le même serveur).

Une tentative de connexion sur une machine cliente provoquera le montage automatique du répertoire de l'utilisateur pour lequel toute cette organisation est complètement transparente. Les répertoires des comptes partagés sont toujours situés dans un autre répertoire que ceux des comptes locaux (souvent */export/home* et */home* respectivement).

A un instant donné, l'examen de la commande *mount* sur une machine cliente indique clairement les montages correspondant aux utilisateurs connectés ou déconnectés depuis moins de quelques minutes, selon le délai par défaut du démontage automatique.

11. Le serveur Samba

11.1 Présentation générale de Samba

Samba est une suite logicielle utilisée principalement pour le **partage de fichiers et d'imprimantes** entre des serveurs Unix ou Linux et des clients Microsoft Windows.

Samba permet aussi de réaliser des tâches complémentaires habituellement dévolues à des serveurs Microsoft, telles que l'**authentification** des utilisateurs, la gestion des droits d'accès, la **résolution de noms** ou encore le parcours du **voisinage réseau**.

Le produit intègre également des **outils clients Unix** permettant d'accéder, de manière symétrique, à des serveurs Microsoft ou, bien entendu, à des serveurs Samba.

De par son aspect *Open Source* et grâce à sa fiabilité et ses performances, Samba constitue une excellente alternative à diverses autres solutions de partage en milieu hétérogène. Le choix de Samba sera notamment dicté par la volonté de se substituer à des serveurs propriétaires et payants tels que Microsoft Windows ou Novell Netware.

Du point de vue du poste client, le serveur Samba se comportera exactement comme le serveur attendu (serveur Windows NT/2000 par exemple). Cette substitution (*masquerade*) est totalement transparente. Les postes clients Windows ne nécessitent aucun logiciel complémentaire pour accéder au serveur. Outre l'aspect technique, l'économie de licences payantes pour ces postes clients peut constituer un aspect très motivant dans le choix de serveurs Samba, a fortiori sur des systèmes gratuits tels que Linux.

Samba implémente le protocole **SMB** (*Server Message Block*) et son nom s'en inspire d'ailleurs très clairement. Le protocole **CIFS** (*Common Internet FileSystem*) est aujourd'hui le successeur de SMB et l'un ou l'autre des deux noms peut être utilisé indifféremment.

Pour accéder à un serveur Samba, le support du protocole **NetBIOS over TCP/IP** (*NBT*), sur lequel s'appuie SMB, est nécessaire. Cette condition est remplie par toutes les versions actuelles de systèmes Microsoft. D'autre part, il est bien évident qu'une configuration TCP/IP correcte est un préalable pour tous les systèmes concernés par une utilisation de Samba.

11.1.1 NetBIOS, NetBEUI, NBT, SMB/CIFS ?

Au début des années 1980, IBM propose une interface de programmation baptisée **NetBIOS** (*Network Basic Input/Output System*). Il s'agit d'une extension du BIOS (*Basic Input/Output System*) destinée à fournir des services réseau.

En 1985, IBM finalise un protocole associé à cette interface et le baptise **NetBEUI** (*NetBIOS Enhanced User Interface*). NetBEUI est un protocole de transport non routable qui convient donc uniquement aux petits réseaux constitués d'un seul segment physique.

Le terme générique protocole est bien souvent utilisé comme un raccourci pour désigner une famille de protocoles ou d'interfaces associés. Ainsi, sans être trop puriste, il est très habituel et acceptable de confondre les deux termes NetBIOS et NetBEUI.

Par la suite, pour permettre une utilisation plus large et supporter notamment l'activité de routage, NetBIOS fut implémenté au-dessus des protocoles IPX, DECNet et naturellement TCP/IP.

L'implémentation **NetBIOS over TCP/IP** (*NBT*) est celle utilisée par Samba.

De son côté, Microsoft a développé le protocole **SMB** (*Server Message Block*). Il s'agit d'un protocole de plus haut niveau qui s'appuie sur NBT pour offrir diverses fonctionnalités, dont les plus importantes sont :

- le parcours du voisinage réseau (**browsing**),

- la résolution de noms NetBIOS en adresses IP (service **WINS**),

- l'authentification centralisée (via la notion de **domaine**).

Microsoft a ensuite amélioré le protocole SMB et l'a rebaptisé **CIFS** (*Common Internet FileSystem*).

A partir de Windows 2000, l'usage de NBT (NetBIOS over TCP/IP) n'est plus indispensable. Le protocole SMB peut s'appuyer directement sur TCP/IP via le port 445. La version 3 de Samba peut être configurée pour tenir compte de cet aspect.

11.1.2 Liste résumée des fonctionnalités

Samba a donc été conçu principalement pour permettre à des serveurs Unix ou Linux d'utiliser le protocole SMB/CIFS et de se comporter ainsi comme des serveurs Microsoft dans diverses opérations.

Nous pouvons résumer ainsi les fonctionnalités principales du produit Samba :

- partage d'arborescences de fichiers,

- partage d'imprimantes connectées sur le serveur ou sur les clients,

- parcours du voisinage réseau (browsing),

- service d'authentification des clients Windows et gestion des autorisations et droits d'accès (Samba peut, entre autres, jouer le rôle d'un **PDC** (*Primary Domain Controller*) dans le cadre d'un domaine Windows),

- résolution de noms NetBIOS en adresses IP (Samba peut ainsi jouer le rôle d'un serveur Microsoft WINS),

- fourniture d'outils clients Unix orientés ligne de commande,

- fourniture d'un utilitaire de configuration baptisé **SWAT** (*Samba Web Administration Tool*) utilisable depuis un navigateur Web.

Par contre, malgré les efforts de l'équipe de développement, Samba comporte toujours quelques lacunes et ne peut pas tenir les rôles suivants :

- contrôleur secondaire de domaine (**BDC** : *Backup Domain Controller*) quand le PDC est un serveur Windows,

- contrôleur de domaine Active Directory,

- serveur WINS secondaire.

Depuis les versions **3.0.x**, Samba est capable de s'intégrer en tant que serveur membre dans un domaine *Active Directory*. Pour ce faire, l'authentification des utilisateurs peut s'appuyer sur *LDAP* et sur *Kerberos*. Le système d'authentification a d'ailleurs été complètement réécrit en interne et est extrêmement paramétrable.

Samba 3 peut maintenant établir des relations d'approbation avec des contrôleurs de domaines Windows NT.

11.1.3 Composantes de Samba

Nous donnons ici une liste résumée des programmes qui constituent la distribution Samba.

Samba s'articule autour de trois programmes démons : **smbd**, **nmbd** et **winbindd** dont le comportement est piloté par le contenu d'un important fichier de configuration baptisé **smb.conf**.

- **smbd**

Ce programme fondamental se charge de la gestion des ressources partagées (fichiers et imprimantes). Il fournit aussi les fonctionnalités d'authentification.

- **nmbd**

Cet autre programme fondamental participe à la fonctionnalité de parcours du voisinage réseau (browsing) et fournit l'équivalent d'un serveur Microsoft WINS (serveur de noms NetBIOS).

Le démon *smbd* est à l'écoute des demandes de connexion et génère un processus fils pour chaque connexion active. Il ne faudra donc pas s'étonner de l'éventuelle présence de nombreux processus *smbd*. Il n'y a, par contre, qu'une seule occurrence du démon *nmbd*, sauf si Samba joue le rôle de serveur WINS, ce qui génère alors une deuxième instance.

- **winbindd**

Ce démon permet d'obtenir des informations au sujet des utilisateurs définis sur des contrôleurs de domaine Windows NT/2000. Son objectif est de faciliter l'intégration d'un serveur Samba dans un domaine comprenant déjà un PDC (Primary Domain Controller), en évitant la duplication des comptes Windows sur le système Unix.

Outre les trois démons précédents, Samba comporte un certain nombre de commandes et utilitaires :

- **findsmb**

Obtention d'informations sur les systèmes supportant le protocole SMB.

- **net**

Commande similaire à la commande Windows de même nom.

- **nmblookup**

Interrogation d'un serveur de noms NetBIOS.

- **pdbedit**

Gestion de comptes stockés dans une base de données *SAM Windows*.

- **rpcclient**

Exécution de programmes administratifs sur des clients Windows.

- **smbcacls**

Gestion des ACLs.

- **smbclient**

Programme interactif multifonctions. Cette commande peut être utilisée sous forme de session interactive pour réaliser diverses opérations (transferts de fichiers, sauvegardes, impressions, tests…).

- **smbcontrol**

Interrogations simples auprès des démons.

- **smbmount**

Montage d'une ressource SMB.

- **smbpasswd**

Gestion des mots de passe.

- **smbspool**

Gestion des impressions.

- **smbstatus**

État des connexions.

- **smbtar**

Utilitaire de sauvegarde.

- **smbumount**

Démontage d'une ressource SMB.

- **swat**

Utilitaire de configuration accessible depuis un navigateur Web.

- **testparm**

Vérification du fichier de configuration.

- **testprns**

  Vérification des informations sur les imprimantes.

- **wbinfo**

  Interrogation du démon winbindd.

11.1.5 Ressources, démarrage

Le nom du site officiel de référence est **www.samba.org**.

De nombreux sites miroirs sont accessibles depuis ce nom générique.

Ces sites donnent accès au téléchargement et à la **documentation** fournie sous la forme d'un document très complet de type *HOWTO*.

Dans la plupart des implémentations sous forme de paquets logiciels, le script de démarrage **/etc/init.d/samba** permet de démarrer les démons **nmbd** et **smbd**.

```
/etc/init.d# ./samba start
Starting Samba daemons: nmbd smbd.
/etc/init.d# ps -ef | egrep "smbd|nmbd"
root      5895     1  0 18:13 ?        00:00:00 /usr/sbin/nmbd -D
root      5897     1  0 18:13 ?        00:00:00 /usr/sbin/smbd -D
root      5898  5897  0 18:13 ?        00:00:00 /usr/sbin/smbd -D
root      5904  5565  0 18:13 pts/0    00:00:00 grep -E smbd|nmbd
/etc/init.d#
```

De la même façon, le script de démarrage **/etc/init.d/winbind** permet de démarrer le démon **winbindd**.

```
/etc/init.d# ./winbind start
Starting the Winbind daemon: winbind.
/etc/init.d# ps -ef | grep "winbind"
root      5910     1  0 18:16 ?        00:00:00 /usr/sbin/winbindd
root      5911  5910  0 18:16 ?        00:00:00 /usr/sbin/winbindd
root      5917  5565  0 18:16 pts/0    00:00:00 grep winbind
/etc/init.d#
```

La version installée de Samba nous est donnée en interrogeant le démon *smbd* avec l'option **-V**.

```
# smbd -V
Version 3.0.24
#
```

11.2 Premier test, outils clients

11.2.1 Constitution d'un fichier de configuration minimum

La présence du fichier de configuration **/etc/samba/smb.conf**. est nécessaire pour permettre le lancement effectif des démons.

Le fichier comporte une section **[global]** dans laquelle l'attribut **workgroup** doit préciser le nom du groupe de travail (ou domaine) dont fait partie le système.

Pour un premier test minimal, nous allons renseigner cet attribut *workgroup* (avec la valeur *MONDOMAINE*) et nous allons ajouter à la fin du fichier une section **[test]** qui correspondra à un nom de partage.

L'attribut **path** y précise le nom d'un répertoire partagé (*/export/sambatest*) que nous allons créer. Il est accompagné d'un attribut **comment** qui donne un texte descriptif qui apparaîtra dans les affichages de diverses commandes.

Nous relancerons le script *samba* pour prendre en compte les modifications.

```
# mkdir –p /export/sambatest
# cat /etc/samba/smb.conf

……..

……..

#======================= Global Settings =======================

[global]

## Browsing/Identification ###

# Change this to the workgroup/NT-domain name your Samba server will
part of
   workgroup = MONDOMAINE

……..

……..

[test]
        path = /export/sambatest
        comment = repertoire de test
# /etc/init.d/samba restart
Stopping Samba daemons: nmbd smbd.
Starting Samba daemons: nmbd smbd.
#
```

11.2.2 La commande smbclient

La commande **smbclient** permet de tester et d'interroger les démons Samba.

Cette commande est un utilitaire très complet, développé à l'origine comme outil de test et de recherche d'erreurs. Elle intègre beaucoup de fonctionnalités mais nous allons l'utiliser ici dans une syntaxe simple qui demande à la machine locale (option *-L localhost*) la liste des services disponibles. L'option *-U%* évite une authentification avec mot de passe.

```
# smbclient -U% -L localhost
Domain=[MONDOMAINE] OS=[Unix] Server=[Samba 3.0.24]

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        test            Disk      repertoire de test
        IPC$            IPC       IPC Service (debian1 server)
Domain=[MONDOMAINE] OS=[Unix] Server=[Samba 3.0.24]

        Server               Comment
        ---------            -------
        DEBIAN1              debian1 server

        Workgroup            Master
        ---------            -------
        MONDOMAINE
```

Cet affichage nous confirme la présence d'un partage baptisé *test* dans le cadre du groupe de travail *MONDOMAINE* sur le serveur de nom *DEBIAN1* (version Samba 3.0.24).

11.2.3 SWAT (Samba Web Administration Tool)

SWAT est un outil de configuration accessible depuis un navigateur Web.

Il s'exécute en tant que démon ponctuel géré par le super démon *inetd*.

Le service *swat* doit être renseigné dans le fichier **/etc/services** (port 901) :

```
swat    901/tcp    # swat
```

Une ligne adéquate doit également être présente dans le fichier **/etc/inetd.conf** :

```
swat  stream  tcp  nowait.400  root  /usr/sbin/tcpd  /usr/sbin/swat
```

L'utilitaire sera ensuite accessible via l'URL **http://localhost:901**.

La page d'accueil pointe directement sur la documentation intégrée au produit et comporte un certain nombre d'icônes permettant d'accéder aux divers sous-menus.

L'icône **VIEW** permet de visualiser le contenu du fichier de configuration **smb.conf**.

L'icône **STATUS** permet de voir l'état des démons et de visualiser les éventuels partages en cours.

11.3 Le fichier de configuration smb.conf

Le comportement des démons *smbd* et *nmbd* est déterminé par les informations du fichier de configuration *smb.conf*.

L'utilitaire *SWAT*constitue l'interface privilégiée pour gérer ce fichier qui peut aussi, bien entendu, être édité directement.

11.3.1 Règles générales de syntaxe

Le fichier a une structure de lignes et est organisé en **sections** dont le nom est placé entre crochets. La section se termine dès l'apparition d'une autre section ou, bien sûr, en fin de fichier.

Le nom d'une section désigne un **service** (ou **partage** ou encore **ressource**) correspondant à un répertoire ou à une imprimante partagée.

Chaque section contient une liste d'**options** sous la forme :

```
option = valeur
```

Les noms de sections et d'options sont insensibles à la casse.

Les espaces ne sont pas significatifs dans le nom des options même s'il est conseillé de respecter la syntaxe proposée dans la documentation officielle.

Dans la définition d'une option, les espaces avant et après le signe = sont ignorés. Par contre, les espaces internes sont significatifs dans la valeur d'une option.

Pour plus de lisibilité, il est possible d'encadrer la valeur d'une option par des quotes même si cela est inutile.

Une ligne peut se continuer via l'utilisation du caractère **** en fin de ligne :

Exemple

```
     comment = mon texte peut se continuer \
sur la ligne suivante
```

Les lignes qui commencent indifféremment par un point-virgule ou par un dièse sont considérées comme des commentaires.

```
# cette ligne est un commentaire
; cette ligne est aussi un commentaire
```

Les lignes vides sont également ignorées et permettent d'aérer la présentation.

Les **valeurs** d'une option peuvent être des **chaînes** de caractères (la casse y est préservée), des valeurs **numériques** ou des **booléens**.

Les valeurs numériques peuvent être des valeurs entières, octales (précédées d'un zéro) ou hexadécimales (précédées de 0x).

En ce qui concerne les booléens, les valeurs acceptées sont **yes/no** ou bien **1/0** ou encore **true/false**. La casse n'est pas significative (*YES* est identique à *yes*).

Enfin, une valeur peut consister en une **liste**. Selon les cas, les valeurs individuelles sont alors séparées par des espaces ou des virgules.

Les modifications du fichier sont automatiquement détectées par Samba avec un délai maximum d'une minute. Cependant, il est possible de forcer une relecture immédiate via l'envoi du signal SIGHUP au processus concerné (*smbd* ou *nmbd*).

11.3.2 Variables

Chaque connexion d'un client génère une nouvelle occurrence du démon *smbd*. Samba positionne des **variables** qui sont utiles pour constituer la valeur de certaines options du fichier de configuration.

Les noms de ces variables sont précédés du caractère **%**.

Nous donnons ici la liste des principales variables reconnues par Samba.

Nom de variable	Définition
%a	Architecture du client *Samba* *WfWg* Windows for Workgroups *Win95* Windows 95, 98 ou Me *WinNT* Windows NT *Win2K* Windows 2000 ou XP *UNKNOWN* Autres systèmes
%I	Adresse IP du client
%M	Nom DNS du client
%m	Nom NetBIOS du client
%u	Identité de l'utilisateur pour le partage concerné
%U	Identité souhaitée par l'utilisateur du partage

	(pas toujours identique à %u)
%H	Répertoire de connexion de l'utilisateur %u
%g	Groupe principal de l'utilisateur %u
%G	Groupe principal de l'utilisateur %U
%S	Nom du partage
%P	Répertoire racine du partage concerné
%d	PID du processus courant
%h	Nom DNS du serveur Samba
%L	Nom NetBIOS du serveur Samba (possibilité d'envisager des serveurs virtuels via des noms différents donnés par les clients)
%v	Version de Samba
%T	Date et heure système
%$*var*	Valeur de la variable d'environnement *var*

11.3.3 Sections particulières

Certaines sections ne désignent pas directement un partage car elles jouent un rôle particulier.

[global]

Cette section contient des options qui concernent tous les services. Si une autre section du fichier reprend une option de cette section globale, la valeur particulière remplace la valeur générale. Certaines options ne peuvent apparaître qu'à l'intérieur de la section [global]. Il s'agit d'options qui décrivent le comportement général de Samba.

[homes]

Cette section correspond à la possibilité de partager les répertoires de connexion des utilisateurs, sans avoir à les énumérer individuellement en tant que services. En effet, si cette section [homes] est présente et si un utilisateur tente une connexion vers un partage qui n'apparaît pas explicitement dans le fichier de configuration, Samba considère ce nom comme le nom d'un utilisateur du système. Dans l'affirmative, Samba considère que l'utilisateur souhaite accéder à son répertoire de connexion.

[printers]

Cette section joue un rôle assez identique à la précédente pour l'accès aux imprimantes. En effet, si les deux sections *[homes]* et *[printers]* sont présentes et si un utilisateur tente une œonnexion vers un partage qui n'apparaît pas explicitement dans le fichier de configuration, Samba considère d'abord ce nom comme un nom d'utilisateur du système. Si ce n'est pas le cas, Samba considère que l'utilisateur souhaite accéder à une imprimante du système. Compte tenu de cet algorithme, il est clair qu'il faut éviter de donner à une imprimante le même nom qu'à un utilisateur. La section *[homes]*, consultée en premier, masquerait en effet l'accès à cette imprimante.

11.3.4 Options particulières

Certaines options permettent d'agir sur le fichier de configuration lui-même :

- **config file** (option globale)

Cette option permet d'indiquer le nom d'un autre fichier de configuration qui remplace alors totalement le fichier courant. Si le fichier n'existe pas, ceci ne provoque pas d'erreur, l'option est simplement ignorée. L'intérêt éventuel est de pouvoir charger dynamiquement une configuration en fonction de la valeur d'une variable telle que, par exemple, le nom de la machine cliente ou le nom d'un utilisateur.

Exemple

```
[global]
     config file = /home/samba/etc/smb.conf.%m
```

La variable *%m* contient le nom NetBIOS du client. Si le nom constitué à l'aide de cette variable correspond à un fichier existant, celui-ci remplacera le fichier de configuration courant. Dans le cas contraire, le fichier initial sera conservé.

- **include**

Cette option permet d'indiquer le nom d'un autre fichier de configuration qui s'insère dans le fichier courant. Il n'y a pas, cette fois-ci, de remplacement complet. Si le fichier n'existe pas, ceci ne provoque pas d'erreur, l'option est simplement ignorée. L'intérêt éventuel est également de pouvoir charger dynamiquement une configuration en fonction de la valeur d'une variable telle que le nom de la machine cliente.

Les variables *%u* (nom de l'utilisateur), *%P* (nom du répertoire partagé) et *%S* (nom du partage) ne sont pas utilisables dans le cadre de cette option.

Exemple

```
[global]
     include = /home/samba/etc/smb.conf.%m
```

La variable *%m* contient le nom NetBIOS du client. Si le nom constitué à l'aide de cette variable correspond à un fichier existant, son contenu sera inséré dans le fichier de configuration courant. Dans le cas contraire, l'option sera ignorée.

- **copy**

Cette option permet de recopier le contenu d'une section dans une autre. La section à copier doit précéder la section cible dans le fichier. L'intérêt est de pouvoir constituer des sections prototypes pour des combinaisons d'options répétitives.

Exemple

```
[proto1]
     guest ok = yes
     guest only = yes
     read only = yes
[data1]
     path = /export/data1
     copy = proto1
```

Le partage de nom *data1* correspond au répertoire */export/data1* et reprend les options de la section *proto1* décrite en amont dans le fichier.

11.4 Noms NetBIOS

11.4.1 Structure des noms NetBIOS

Les acteurs d'un réseau SMB sont identifiés par un **nom NetBIOS** qui doit être unique sur tout le réseau. Ces noms sont enregistrés de manière dynamique lors du démarrage des systèmes. En outre, il faut pouvoir associer une **adresse IP** à chaque nom NetBIOS.

Parmi d'autres solutions à base de requêtes de diffusion ***(broadcast***), la meilleure alternative est de mettre en œuvre des serveurs de noms NetBIOS (**NBNS**: *NetBios Name Server*) qui se chargent de l'enregistrement des machines et de la résolution des noms en adresses IP.

L'implémentation Microsoft du serveur de noms NetBIOS s'appelle **WINS** (*Windows Internet Name Service*). Parmi les systèmes Windows, seuls des serveurs NT ou 2000 peuvent faire office de serveurs WINS. Un serveur Samba peut alors également jouer avantageusement ce rôle.

Les systèmes Windows actuels fonctionnent, a priori, dans un mode hybride, baptisé *h-node*. Cela signifie que le système a recours à un serveur WINS pour l'enregistrement et la résolution de noms et n'utilise des requêtes de diffusion qu'en cas d'indisponibilité de ce serveur WINS.

Un nom NetBIOS occupe **16 octets**.

Les 15 premiers octets du nom sont définis par l'utilisateur. Même si certains autres caractères sont valides, il est conseillé de n'utiliser que les caractères alphanumériques (lettres et chiffres) pour constituer ce nom.

Le 16ème caractère est utilisé comme suffixe de nom pour fournir des informations sur les fonctionnalités du système, lors de l'enregistrement.

Un même nom de machine est donc enregistré plusieurs fois, pour chaque occurrence de type de ressource.

A titre d'exemple, nous donnons ici quelques valeurs possibles pour ces types de ressources (en gardant les termes anglo-saxons originels).

Type de ressource	Valeur hexadécimale
Standard Workstation	00
Messenger Service	03
RAS Server Service	06
RAS Client Service	21
Domain Master Browser Service	1B
Master Browser name	1D

Fileserver (including printer server)	20
Network Monitor Agent	BE
Network Monitor Utility	BF

Les noms NetBIOS peuvent aussi être enregistrés comme des noms de **groupes**.

Certains types de ressources peuvent être associés à ces groupes.

Type de ressource	**Valeur hexadécimale**
Standard Workstation group	00
Logon server	1C
Master Browser name	1D
Normal Group name	1E

11.4.2 Rôle de Samba dans la résolution de noms

Un client utilise le serveur *WINS* pour faire enregistrer son propre nom et son adresse IP. Cet enregistrement est renouvelé périodiquement. Lorsqu'il s'arrête normalement, le client en informe le serveur WINS qui peut ainsi libérer le nom NetBIOS.

Le client utilise aussi le serveur WINS pour la résolution des noms en adresses IP.

Le fichier **LMHOSTS** est quelquefois utilisé sur des postes Windows pour le cas où il n'y aurait pas de serveurs WINS ou en cas d'indisponibilité de celui-ci. La présence de ce fichier évite le recours aux requêtes de type diffusion.

Le contenu d'un tel fichier est très similaire à celui d'un fichier **hosts** pour TCP/IP.

L'enregistrement et la résolution des noms NetBIOS peut être géré de différentes manières par le démon **nmbd**.

11.4.2.a Choisir l'ordre de recherche

L'option globale **name resolve order** a pour valeur une liste qui permet de choisir un ordre de recherche pour la résolution des noms NetBIOS.

(Valeur par défaut: lmhosts host wins bcast)

Exemple

```
[global]
     name resolve order = wins lmhosts host bcast
```

Les quatre valeurs possibles de la liste ont les significations suivantes :

- lmhosts Utiliser le fichier local *lmhosts*,
- host Utiliser la méthode de résolution du système Unix,
- wins Utiliser un serveur WINS,
- bcast Effectuer une requête de diffusion (*broadcast*).

11.4.2.b Faire de Samba un serveur WINS

La seule option globale **wins support** est suffisante pour que Samba devienne un serveur WINS.

(Valeur par défaut: no)

Exemple

```
[global]
     wins support = yes
```

Samba peut donc jouer très facilement le rôle de serveur WINS primaire mais il ne sait pas se synchroniser avec d'autres serveurs WINS. Il est donc très important, sur les postes clients Windows, de ne pas indiquer d'autres serveurs WINS si un serveur Samba a été choisi comme serveur WINS principal.

11.4.2.c Utiliser un autre serveur WINS

L'option globale **wins server** permet d'indiquer le nom DNS ou l'adresse IP d'un unique serveur WINS vers lequel rediriger les requêtes. Cette option est, bien entendu, incompatible avec la précédente.

Exemple

```
[global]
     wins server = 192.168.0.100
```

Si nous disposons de plusieurs serveurs Samba, l'un d'entre eux peut être configuré comme serveur WINS via l'option *wins support* et les autres peuvent utiliser l'option *wins server* pour le désigner.

Si le serveur WINS se trouve sur un autre réseau, il est possible d'ajouter l'option globale **wins proxy**. Celle-ci permet de rediriger correctement les requêtes de clients qui n'utiliseraient qu'une recherche de type diffusion.

(Valeur par défaut: no)

Exemple

```
[global]
     wins server = 172.16.0.100
     wins proxy = yes
```

Il est également possible d'envisager une recherche de noms auprès d'un serveur DNS en cas d'échec dans la recherche des noms NetBIOS. Ceci est réalisée via l'option globale **dns proxy**.

(valeur par défaut: no)

Exemple

```
[global]
     dns proxy = yes
```

11.5 Types de réseaux Microsoft

11.5.1 Workgroups (groupes de travail)

Un groupe de travail (**workgroup**) consiste en un ensemble restreint de systèmes situés, a priori, sur un même réseau physique. Cette organisation en groupe de travail fonctionne bien pour une petite communauté d'utilisateurs pour lesquels un faible niveau de sécurité est suffisant.

Les ressources partagées peuvent être protégées par des mots de passe. L'association de mots de passe différents pour chaque ressource partagée s'avère rapidement très contraignante, voire ingérable tout en étant peu efficace au niveau de la sécurité.

Un système Samba peut tout naturellement s'insérer dans un groupe de travail et participer aux partages de fichiers et d'imprimantes.

11.5.2 Domaines

Microsoft a introduit la notion de **domaine** avec les versions NT. Un domaine Windows NT consiste en un groupe de travail auquel s'ajoute un **contrôleur de domaine**.

Ce contrôleur de domaine a pour rôle de maintenir des informations centralisées sur les utilisateurs et les groupes tout en permettant leur authentification sur le réseau et, de par le fait, le contrôle d'accès aux ressources partagées.

Pour un contrôleur de domaine, le service de gestion des informations sur les utilisateurs s'appelle **SAM** (*Security Account Manager*). Le modèle de sécurité Windows NT s'appuie sur des représentations d'objets baptisés **SIDs** (*Security IDentifiers*). Les SIDs permettent notamment la désignation des utilisateurs, des groupes et des ordinateurs membres du domaine. D'autre part, les contrôles d'accès aux objets sont gérés via des **ACLs** (*Access Control Lists*).

Un domaine contient obligatoirement un **PDC** (*Primary Domain Controller*) et peut comporter un ou plusieurs **BDCs** (*Backup Domain Controller*s) pour pallier une éventuelle indisponibilité du contrôleur primaire.

Les BDCs ont des copies (en lecture seule) de la base de données SAM du contrôleur primaire. Ces copies peuvent être mises à jour via des opérations de synchronisation.

Toutes les versions récentes de Windows (sauf Windows XP home) peuvent se connecter à un domaine en tant que client et accéder ainsi aux ressources partagées. Certains ordinateurs peuvent être considérés comme membres du domaine. Ils sont alors connus des contrôleurs de domaine via un compte d'ordinateurs dans la base SAM.

Samba peut jouer le rôle d'un **PDC** pour des clients Windows 95/98/Me ainsi que NT/2000/XP. Par contre, il ne peut pas remplir le rôle de **BDC**.si le PDC est un serveur Windows NT/2000. Samba peut également se comporter comme un **serveur membre** du domaine.

Les **relations d'approbation** constituent un autre aspect important des domaines Windows NT. Elles permettent à un utilisateur d'un domaine d'accéder aux ressources

partagées d'un autre domaine sans avoir à s'authentifier de nouveau. Samba 2.2 ne supportent pas les relations d'approbation mais elles sont possibles en version 3.

11.5.3 Active Directory

Avec Windows 2000, Microsoft a introduit l'organisation **Active Directory** qui constitue une étape complémentaire par rapport à l'organisation en domaines.

Avec Active Directory, le modèle d'authentification est centré sur les annuaires **LDAP** (*Lightweight Directory Access Protocol*) et le service de noms est fourni par **DNS** (noms TCP/IP). L'usage des noms NetBIOS (serveurs WINS) n'est plus nécessaire.

Les domaines sont alors organisés en une structure d'arbre hiérarchique et il n'y a plus de distinction entre contrôleurs primaires et secondaires.

Les systèmes Windows 2000/XP peuvent toujours être configurés comme des clients d'un domaine Windows NT, voire même d'un simple workgroup.

Les serveurs Windows 2000 sont destinés à une configuration Active Directory mais ils supportent les domaines Windows NT à titre de compatibilité. Dans ce cas, Samba fonctionne avec ces serveurs de la même façon qu'avec des serveurs Windows NT.

Quand un serveur Windows 2000 fonctionne en mode natif (Active Directory seulement), Samba peut opérer en tant que serveur en utilisant le mode d'émulation PDC. Cependant, il n'est pas possible que Samba joue le rôle d'un contrôleur de domaine en environnement purement Active Directory.

11.6 Voisinage réseau (browsing)

11.6.1 Terminologie et principes de fonctionnement

La fonctionnalité de parcours du voisinage réseau consiste à découvrir quels sont les systèmes disposés à partager leurs ressources (fichiers et imprimantes) et quelles sont précisément ces ressources.

Dans le cadre d'un groupe de travail, un ordinateur est désigné pour maintenir une liste actualisée des systèmes accessibles (*browse list*).

Cet ordinateur est baptisé **local master browser**. La mise à disposition de la liste par une machine bien identifiée évite que chaque système génère des interrogations coûteuses de type diffusion (*broadcast*).

Après avoir procédé à l'enregistrement de son nom NetBIOS, chaque ordinateur du groupe de travail Windows doit annoncer sa présence au *local master browser*. Il doit également théoriquement annoncer son départ du groupe lors de sa procédure d'arrêt. Cet aspect n'est pas toujours bien géré et il se peut que la *browse list* ne soit pas toujours en phase avec la réalité du moment.

Tous les systèmes Windows (95/98/Me/NT/2000/XP) peuvent jouer le rôle de *local master browser*. Selon des conventions Microsoft liées au nombre de systèmes présents sur le réseau, une ou plusieurs machines sont désignées pour jouer un rôle de secours (**local backup browser**) et pallier ainsi une éventuelle défaillance du *master browser*. Les systèmes concernés synchronisent régulièrement leur liste pour en assurer une cohérence optimale.

Dans le cadre d'un domaine, il peut y avoir plusieurs réseaux physiques. Dans ce cas, l'un des ordinateurs jouant le rôle de *local master browser* pour un réseau donné devra également remplir la fonction de **domain master browser** pour l'ensemble du domaine. La *browse list* sera synchronisée entre les *local master browsers* de chaque réseau physique.

Dans les organisations Microsoft, le rôle de **PDC** est indissociable de celui de **domain master browser**.

La désignation du *local master browser* est réalisée par une **élection**.

Sans rentrer dans les détails de l'algorithme, la désignation se base d'abord sur la version de système :

Système d'exploitation	Valeur pour l'élection
Windows NT/2000 jouant le rôle de PDC	32
Windows NT/2000/XP ne jouant pas le rôle de PDC	16
Windows 95/98/Me	1

Les valeurs ont été choisies pour qu'un PDC devienne systématiquement *local master browser*. En l'absence de PDC, les systèmes NT/2000/XP sont privilégiés. En cas d'égalité, sur un principe similaire de cotations, le rôle de l'ordinateur rentre en ligne de compte pour désigner le vainqueur.

Depuis un poste Windows, le voisinage réseau ou la commande **NET VIEW** permettent de visualiser la liste des systèmes disponibles ainsi que la liste des ressources pour chacun d'entre eux.

Depuis une machine Samba, il est possible d'obtenir cette information via la commande **nmblookup**.

11.6.2 Rôle de Samba dans l'activité de voisinage réseau

Samba peut jouer tous les rôles possibles dans l'activité de *browsing*.

Par défaut, Samba participe aux élections avec la valeur 20 en tant que système d'exploitation. Il l'emporte donc sur tous les systèmes Windows qui ne font pas office de PDC.

11.6.2.a Samba en tant que local master browser

Plusieurs options interdépendantes rentrent en jeu dans la désignation ou non de Samba en tant que **local master browser**.

local master Participation à l'élection du *local master browser*

Cette option peut apparaître uniquement dans la section [global].

(Valeur par défaut: yes)

os level Valeur du système dans les élections du voisinage réseau

Cette option peut apparaître uniquement dans la section [global].

(Valeur par défaut: 20)

preferred master Provoquer une élection du *local master browser*

prefered master Synonyme

Cette option peut apparaître uniquement dans la section [global].

Exemple

```
[global]
     local master = yes
     preferred master = yes
     os level = 255
```

Dans cet exemple, l'option *preferred master* provoque une élection au démarrage de Samba. L'option *local master* fait que le système est candidat à cette élection. La valeur

maximale de 255 (codage sur 8 bits) lui permet de l'emporter face à tous les systèmes Windows.

Il conviendrait de ne pas donner la même configuration à d'autres serveurs Samba, ce qui provoquerait des élections aux résultats aléatoires car liés à des critères tels que la date de démarrage. Il est donc important de donner une valeur inférieure à l'option *os level* pour les autres serveurs Samba et de ne pas y activer l'option *preferred master*.

11.6.2.b Samba en tant que domain master browser

domain master Devenir *domain master browser*

Cette option peut apparaître uniquement dans la section [global].

Exemple

```
[global]
     domain master = yes
     local master = yes
     preferred master = yes
     os level = 255
```

Dans cet exemple, la combinaison de toutes les options nous assure que le serveur Samba sera à la fois *local master browser* et *domain master browser*.

Si Samba est configuré pour jouer le rôle de PDC, il est important qu'il soit également *domain master browser*. Inversement, si Samba n'est pas un PDC, il ne faut pas lui faire jouer ce rôle de *domain master browser*.

11.6.2.c Options annexes

browsable Visualisation de la ressource dans le voisinage réseau

browseable Synonyme

(Valeur par défaut: yes)

Exemple

```
[homes]
     browsable = no
```

Cette option n'empêche pas l'accès à la ressource. Elle masque simplement son nom dans le voisinage réseau. Il est notamment possible d'y accéder via une notation dite UNC : *machine**ressource* .

auto services Forcer la visualisation d'une liste de ressources

dans le voisinage réseau

preload Synonyme

Cette option peut apparaître uniquement dans la section [global].

Exemple

```
[global]
     auto services = michel cath
```

Cette option permet de forcer la visualisation de certains partages gérés via la section spéciale *[homes]* et qui seraient, par ailleurs, invisibles dans la mesure où l'option précédente *browsable = no* est très souvent associée à la section *[homes]*. Dans l'exemple, les répertoires de connexion des deux utilisateurs *michel* et *cath* seront affichés dans le voisinage réseau.

load printers Obtenir la visualisation de toutes les imprimantes

dans le voisinage réseau

Cette option peut apparaître uniquement dans la section [global].

(Valeur par défaut: yes)

Exemple

```
[global]
     load printers = yes
```

Cette option permet la visualisation dans le voisinage réseau de toutes les imprimantes du système, gérées par ailleurs via la section spéciale *[printers]*.

default service Définir un partage par défaut

en cas d'échec de connexion à un service

default Synonyme

Cette option peut apparaître uniquement dans la section [global].

Exemple

```
[global]
     default service = pub
[pub]
     path = /home/samba/default/%S
```

La valeur de l'option doit être le nom d'un partage existant (sans les crochets). Un client qui tente une connexion vers un service inexistant ou dont l'accès lui est refusé, se verra connecté sur ce partage par défaut. La variable *%S* contient alors le nom du partage en échec et elle peut éventuellement être exploitée dans les options du service par défaut. L'usage de cette option n'est pas tellement recommandé.

11.7 Partages en contexte Workgroup

11.7.1 Configuration des partages sur le serveur Samba

11.7.1.a Options globales

Dès le début de la configuration, trois options de base apparaîtront dans la section [global].

workgroup Nom du groupe de travail auquel appartient le serveur

La chaîne *WORKGROUP* constitue souvent la valeur par défaut de cette option. Il convient, bien entendu, de ne pas conserver ce nom passe-partout.

netbios name Nom NetBIOS du serveur

La valeur par défaut est constituée de la première partie du nom DNS.

server string Description du serveur

La variable *%v* correspond à la version de Samba. Nous pouvons, par exemple, compléter le descriptif par la variable *%L* qui contient le nom NetBIOS du serveur.

Exemple

```
[global]
     workgroup = MONDOMAINE
     netbios name = DEBIAN1
     server string = Samba %v sur %L
```

Cet exemple décrit un serveur de nom *DEBIAN1*, membre du workgroup *MONDOMAINE*. Dans les listes du voisinage réseau, il est décrit par la version de Samba *(%v*) et par son nom NetBIOS *(%L*).

11.7.1.b Ajout d'un partage de répertoire

Nous allons ajouter un premier partage de disque très simple pour découvrir quelques options essentielles. Ces options seront placées dans une section portant le nom du partage.

path Nom complet du répertoire partagé

directory Synonyme

Le nom de répertoire peut utiliser les deux variables *%u* (nom de l'utilisateur) et *%m* (nom NetBIOS du client).

comment Description du partage dans les listes du voisinage réseau

guest ok Accès public (sans mot de passe) autorisé

public Synonyme

(Valeur par défaut: no)

L'interprétation exacte de cette option dépend du niveau de sécurité choisi. Le sous-chapitre suivant précisera les différents modes d'authentification supportés par Samba.

guest account Identité de l'utilisateur pour les ressources accessibles en mode public (option *guest ok*)

(Valeur par défaut: nobody)

Le compte Unix *nobody* correspond à un utilisateur prédéfini disposant de très peu de droits sur le système. Il peut être souhaitable de créer un compte particulier destiné au contexte Samba. Nous pourrons, en effet, choisir un utilisateur plus ciblé (par exemple : *guest* ou encore *lp* pour des imprimantes en accès public).

guest only Ressource accessible uniquement en mode public (nécessite l'option **guest ok = yes**)

only guest Synonyme

(Valeur par défaut: no)

Selon le niveau de sécurité choisi (voir sous-chapitre suivant), l'accès à une ressource munie de l'option *guest ok = yes* reste éventuellement possible par un utilisateur avec mot de passe. Cet utilisateur conserve alors son identité et les droits associés. Si ce comportement n'est pas souhaité, l'option complémentaire *guest only = yes* devra être ajoutée.

writable Accès au partage en lecture/écriture

writeable Synonyme

write ok Synonyme

(Valeur par défaut: no)

read only Accès en lecture seulement

(Valeur par défaut: yes)

Les options *writable* et *read only* correspondent à deux façons symétriques d'exprimer la même chose.

Exemple

```
[global]
     workgroup = MONDOMAINE
     netbios name = DEBIAN1
     server string = Samba %v sur %L
[test]
     path = /export/sambatest
     comment = repertoire de test
     guest ok = yes
     guest account = pc
     writable = yes
```

Cet exemple ajoute une section correspondant à un partage baptisé *test*. Il s'agit d'un accès en lecture/écriture (option *writable*) au répertoire */export/sambatest* (option *path*). Un accès public est possible (option *guest ok*). Dans ce cas, l'utilisateur prend l'identité et les droits du compte *pc* (option *guest account*).

11.7.2 Précisions sur les mots de passe

Les utilisateurs d'un partage Samba doivent être identifiés sur le poste client Windows avec un compte et un mot de passe connus également par le système Unix. Depuis Windows 98 et depuis le Service Pack 3 de Windows NT, les mots de passe sont implicitement transmis sous forme cryptée, lors des échanges sur le réseau.

L'option globale **encrypt passwords** est donc absolument nécessaire pour gérer correctement cet aspect.

Exemple

```
[global]
     encrypt passwords = yes
```

En complément de cette option, il convient de déclarer l'utilisateur via la commande **smbpasswd** qui gère un fichier des mots de passe Samba, complémentaire au fichier des mots de passe du système Unix lui-même.

Depuis les versions Samba 3, il est recommandé officiellement de ne plus stocker les mots de passe Samba dans un simple fichier texte. L'option **passdb backend** pilote ce choix qui peut être aujourd'hui *tdbsam* (base triviale au format binaire) ou *ldapsam* (utilisation d'un annuaire LDAP). Si nous souhaitons conserver malgré tout l'ancien fichier texte pour les mots de passe, nous choisirons la valeur *smbpasswd* pour cette option.

```
# If you are using encrypted passwords, Samba will need to know what
# password database type you are using.
;   passdb backend = tdbsam
   passdb backend = smbpasswd
```

Exemple de création d'un compte Samba

```
# smbpasswd -a michel
New SMB password:
Retype new SMB password:
Added user michel.
#
```

Dans cet exemple, l'option *-a* de la commande *smbpasswd* permet de créer l'utilisateur *michel* au niveau de Samba. Nous devons renseigner son mot de passe qui sera stocké dans le fichier **/etc/samba/smbpasswd** (à ne pas confondre avec la commande */usr/bin/smbpasswd*). L'utilisateur *michel* doit également correspondre à un compte Unix défini dans le fichier */etc/passwd*.

11.7.2.a Format du fichier smbpasswd

Le fichier **smbpasswd** est organisé en lignes. Chaque ligne représente un utilisateur Samba et est constituée de six champs séparés par le caractère : (deux points).

```
# grep michel /etc/samba/smbpasswd
michel:1000:BAC14D04669EE1D1AAD3B435B51404EE:
FBBF55D0EF0E34D39593F55C5F2CA5F2:[U          ]:LCT-471AEA65:
```

Le format de chaque ligne du fichier est le suivant :

```
username:uid:lm:ntlm:flag:lastchange
```

username Nom de l'utilisateur (issu de */etc/passwd*)

uid Numéro de l'utilisateur (issu de */etc/passwd*)

lm Chaîne hexadécimale de 32 caractères
(mot de passe crypté des clients Windows 95/98/Me)

ntlm Chaîne hexadécimale de 32 caractères
(mot de passe crypté des clients Windows NT/2000/XP)

flag Chaîne, entre crochets, de 11 caractères (espaces compris)

- U utilisateur ordinaire
- D utilisateur désactivé (les connexions sont refusées)
- N utilisateur sans mot de passe
- W compte d'ordinateur (utilisé quand Samba joue le rôle de PDC)

lastchange Date de dernier changement de mot de passe
(chaîne *LCT* suivie du nombre, en hexadécimal , de secondes écoulées depuis le 1er janvier 1970)

11.8 Partages en contexte Domaines, Samba en tant que PDC

11.8.1 Organisation à base de domaines

L'organisation en *workgroup* atteint rapidement ses limites, notamment en matière de sécurité. En pratique, il nous semble préférable d'adopter une organisation à base de **domaine**.

Un domaine consiste en un groupe de travail auquel s'ajoutent des **contrôleurs de domaine** dont le rôle majeur est d'assurer une **authentification** centralisée et sécurisée des utilisateurs.

Un des systèmes doit jouer le rôle de **PDC** (*Primary Domain Controller*) et il est possible de mettre en œuvre des contrôleurs secondaires baptisés **BDCs** (*Backup Domain Controllers*).

Du point de vue de l'utilisateur, l'intérêt principal de cette architecture réside en une authentification unique (**SSO** : *Single Sign On*) depuis n'importe quelle station de travail.

L'organisation en domaine permet également la gestion centralisée des scripts d'ouverture de session (*logon scripts*) et des profils d'utilisateurs (*roaming profiles*). L'objectif est encore de renforcer la possibilité de connexion depuis un poste de travail quelconque. Concrètement, la multiplicité des versions de clients Windows ne facilite pas les choses pour les administrateurs.

Depuis les versions 2..2 , un serveur Samba peut jouer le rôle d'un PDC et gérer des clients Windows quelle qu'en soit la version (95/98/Me ou NT/2000/XP).

Rappelons cependant que la version *XP Home* ne peut, en aucun cas, s'insérer dans un domaine. Pour envisager autre chose qu'un workgroup, il faut disposer obligatoirement de *XP Professional*.

Samba ne peut pas remplir la fonction de BDC (Backup Domain Controller) auprès de serveurs Windows NT/2000 car il ne sait pas se synchroniser avec eux. Par contre, une communauté Samba de contrôleurs de domaine peut être configurée de façon cohérente

Un serveur Samba peut également s'intégrer dans une organisation déjà existante où le PDC serait un système Windows NT/2000 ou un autre serveur Samba. Il se comporte alors comme **serveur membre** du domaine.

En ce qui concerne les organisations **Active Directory**, disponibles depuis Windows 2000, Samba ne peut pas actuellement y jouer le rôle de contrôleur de domaine.

La version 3 permet cependant de devenir serveur membre.

11.8.2 Niveaux de sécurité

Samba implémente plusieurs niveaux de sécurité concernant les tentatives de connexion à une ressource partagée. Le niveau de sécurité est transmis, par le protocole SMB, au client qui peut ainsi choisir une méthode d'authentification appropriée.

L'option globale **security** est fondamentale pour gérer ces niveaux de sécurité.

security Choix du mode d'authentification

(Valeur par défaut: user)

Les valeurs possibles sont **share**, **user**, **server**, **domain** et **ads**.

11.8.2.a security = share

Les mots de passe sont associés directement aux partages.

N'importe quel utilisateur peut accéder à une ressource dont il connaît le mot de passe. Une ressource peut même avoir plusieurs mots de passe, par exemple, un mot de passe permettant un accès en lecture seule et un autre pour un accès en lecture/écriture.

Ce niveau de sécurité est peu pratiqué car il s'adapte assez mal au comportement interne de Samba qui utilise toujours un couple "nom d'utilisateur/mot de passe" pour réaliser les authentifications.

Le traitement peut se compliquer à cause du fait que, dans ce mode, le client ne fournit pas toujours un nom d'utilisateur. Samba tente malgré tout d'associer une identité à ce client.

L'algorithme est le suivant :

Si le partage comprend l'option **guest only**, l'accès à la ressource est directement autorisé. L'identité et les droits seront alors ceux du compte défini par l'option **guest account**. Il faut se rappeler que l'option *guest only* nécessite l'option **guest ok**.

Dans le cas contraire, Samba ajoute le nom de l'utilisateur à une liste interne associée à la ressource et il effectue la vérification du mot de passe pour ce compte. En cas de succès, l'identité et les droits seront naturellement ceux du compte concerné. L'utilisateur n'aura plus besoin de s'authentifier lors des futurs accès au partage.

En cas d'utilisateur inconnu ou de mot de passe invalide, Samba tente de valider le mot de passe par rapport aux autres utilisateurs ayant déjà accompli des connexions à la ressource ainsi que ceux listés dans l'éventuelle option **username** (voir l'exemple). En cas de succès, l'identité et les droits seront naturellement ceux du compte concerné.

Si tous les tests précédents ont échoué mais si le partage comprend l'option *guest ok*, l'accès sera, malgré tout, autorisé. L'identité et les droits seront alors ceux du compte défini par l'option *guest account*.

Exemple

```
[global]
     security = share

[test]
path = /export/sambatest
guest ok = no
writable = yes
username = michel, cath, @gestion
```

Dans cet exemple, quand un utilisateur essaye de se connecter au partage *test*, Samba vérifie le mot de passe fourni, par rapport à sa liste interne d'utilisateurs associés à la ressource, puis, si nécessaire, par rapport à la liste donnée par l'option *username* (les comptes *michel* et *cath* ainsi que les membres du groupe Unix *gestion*). En cas de succès, l'accès est autorisé sous l'identité de l'utilisateur concerné. En cas d'échec, aucun accès n'est possible. On l'a précisé par *guest ok = no* même si cela est la valeur par défaut pour cette option.

11.8.2.b security = user

Les partages sont associés à des listes d'utilisateurs pouvant y accéder.

Il s'agit du niveau de sécurité recommandé dans le contexte Samba et cela correspond d'ailleurs à la valeur par défaut de l'option *security*. Dans ce mode, le client fournit toujours un nom d'utilisateur et un mot de passe associé.

L'éventuelle option **valid users** (ou son contraire **invalid users**) permet de préciser une liste d'utilisateurs ou de groupes autorisés (ou interdits) par rapport à une ressource donnée.

Samba réalise l'authentification des utilisateurs connus du système et tient compte de l'éventuelle présence des deux options précédentes.

En cas de succès, l'identité et les droits seront naturellement ceux du compte concerné. L'utilisateur n'aura plus besoin de s'authentifier lors des futurs accès à la ressource.

Comme précisé précédemment dans ce chapitre, nous rappelons que les mots de passe Windows sont implicitement transmis sous forme cryptée, lors des échanges sur le réseau. L'option globale *encrypt passwords = yes* est donc absolument nécessaire. En complément de cette option, il convient de créer l'utilisateur, au niveau de Samba, via la commande *smbpasswd*.

Exemple

```
[global]
     security = user
     encrypt passwords = yes
     invalid users = root bin daemon sys sync\
                     mail news uucp
[test]
path = /export/sambatest
writable = yes
valid users = michel, cath, @gestion
admin users = michel
```

Dans cet exemple, nous précisons qu'un certain nombre de comptes sensibles n'auront accès à aucun partage puisque l'option *invalid users* est placée dans la section *[global]*. En ce qui concerne le partage *[test]*, l'option *valid users* indique que seuls les comptes *michel*, *cath* et les membres du groupe *gestion* ont une possibilité d'accès. Par ailleurs, le

compte *michel* prend l'identité *root* par rapport à cette ressource, grâce à l'option *admin users*.

11.8.2.c security = server

Dans ce niveau de sécurité, notre serveur Samba utilise un autre serveur pour réaliser l'authentification des utilisateurs. Il s'agira sans doute d'un serveur Windows NT/2000 ou Samba tenant le rôle de PDC.

L'option globale **password server** doit permettre de désigner le serveur d'authentification vers lequel sont redirigées les validations de connexion.

Exemple

```
[global]
     security = server
     password server = serv1
```

password server Liste de serveurs d'authentification

Les serveurs doivent être nommés par leur nom NetBIOS ou par leur adresse IP.

Ce niveau de sécurité est une survivance de l'époque où Samba ne pouvait pas devenir serveur membre d'un domaine. Il est donc recommandé de ne plus utiliser ce niveau et de lui préférer le niveau *security = domain*.

11.8.2.d security = domain

Ce niveau de sécurité ressemble donc au précédent puisqu'il s'agit d'utiliser un autre serveur pour réaliser l'authentification préalable des utilisateurs. L'option **password server** devra toujours préciser le serveur concerné.

La différence réside dans le fait que notre serveur Samba devra être effectivement **membre d'un domaine** afin d'utiliser un des contrôleurs de ce domaine (PDC ou BDC) pour réaliser l'authentification. Celle-ci restera valide pour toutes les autres ressources du domaine. Les contrôleurs de domaine peuvent être des serveurs NT/2000 ou un serveur Samba jouant le rôle de PDC.

Il est à noter que les comptes utilisateurs doivent tout de même être définis sur notre serveur intermédiaire. Il conviendra de les désactiver en terme de connexion locale (avec un mot de passe ou un shell invalide) puisqu'il ne s'agit pas de comptes à usage direct. L'utilisation du service **winbind** permet de se libérer de cette contrainte de duplication des comptes.

11.8.2.e security = ads

Il s'agit d'une possibilité disponible depuis les versions Samba 3, tout à fait similaire, en terme de principe, au niveau *security* = *domain*.

L'authentification est réalisée auprès d'un serveur **Active Directory**. Notre serveur Samba doit aussi être effectivement membre du domaine Active Directory.

Une nouvelle option **ads server** permet de désigner le serveur d'authentification.

ads server Liste de serveurs d'authentification

Les serveurs doivent être nommés par leur nom DNS ou par leur adresse IP.

11.8.3 Adaptation du fichier smb.conf

Nous partons d'un fichier exemple pour présenter un fichier de configuration qui fera de notre serveur Samba un PDC.

Exemple

```
[global]
     workgroup = MONDOMAINE
     netbios name = DEBIAN1
     server string = Samba %v sur %L
     encrypt passwords = yes

     wins support = yes
     domain master = yes
     local master = yes
     preferred master = yes
     os level = 65

     security = user
     domain logons = yes

     logon path = \\%L\profiles\%u\%m
     logon script = logon.bat
     logon drive = H:
     logon home = \\%L\%u\winprofile\%m

[netlogon]
     path = /etc/samba/netlogon
     writable = no
     browsable = no

[profiles]
     path = /etc/samba/ntprof
     browsable = no
     writable = yes
     create mask = 0600
     directory mask = 0700

[homes]
     read only = no
```

```
browsable = no
```

11.8.3.a Options globales

L'option **workgroup** contient cette fois un nom de domaine et non plus celui d'un groupe de travail.

L'usage de mots de passe cryptés (**encrypt passwords = yes**) est obligatoire dans le contexte des domaines.

```
workgroup = MONDOMAINE
netbios name = DEBIAN1
server string = Samba %v sur %L
encrypt passwords = yes
```

L'ensemble des options suivantes, déjà évoquées, a pour objectif de faire de Samba le **serveur WINS** ainsi que le **domain master browser** dans l'activité de parcours du voisinage réseau, ce qui est indissociable du rôle de PDC.

```
wins support = yes
domain master = yes
local master = yes
preferred master = yes
os level = 65
```

L'option **security = user** définit le niveau de sécurité tel que présenté dans le paragraphe précédent. L'apparition explicite de l'option n'est pas absolument nécessaire puisque nous utilisons la valeur par défaut. Il n'est cependant pas inutile de la mentionner pour plus de clarté.

```
security = user
```

Les choix de notation des auteurs de Samba sont quelquefois trompeurs. Pour notre rôle de PDC, c'est bien le niveau *user* qu'il faut choisir et non pas les niveaux *server* ni *domain* qui signifient au contraire que les authentifications sont gérées par d'autres serveurs !

De nouvelles options sont nécessaires pour compléter la configuration. Certaines correspondent à la gestion des scripts d'ouverture de session (**logon scripts**) et des profils errants (**roaming profiles**).

domain logons Acceptation des connexions de type domaine

(Valeur par défaut: no)

```
domain logons = yes
```

Cette option doit obligatoirement prendre la valeur *yes* lorsque Samba joue le rôle d'un PDC. Elle signifie que le serveur accepte les connexions en provenance d'un domaine.

logon path Emplacement des profils errants (roaming profiles)

pour des clients Windows NT/200/XP

```
logon path = \\%L\profiles\%u\%m
```

Les profils ne sont pas gérés de la même manière sur Windows 95/98/Me et sur Windows NT/2000/XP. Il faut donc prévoir des emplacements de stockage différents. Dans notre exemple, le chemin indiqué est relatif à la racine d'un partage de nom **[profiles]** qu'il est obligatoire de prévoir. Les variables *%L*, *%u* et *%m* contiennent respectivement le nom du serveur, le nom de l'utilisateur et le nom du système client. Elles sont intéressantes pour constituer des noms paramétrés pour ces répertoires de stockage.

logon script Nom d'un script d'ouverture de session

```
logon script = logon.bat
```

Cette option précise le nom d'une commande à exécuter lors de la connexion au domaine. Le nom du fichier est relatif à la racine d'un partage de nom **[netlogon]** qu'il est obligatoire de prévoir. Des variables, telles que *%u* et *%m* pourraient aussi être utilisées dans le nom du script pour gérer divers paramétrages.

logon drive Spécifier l'unité qui sera attribuée au répertoire de connexion.

(Valeur par défaut: Z:)

```
logon drive = H:
```

logon home Emplacement des répertoires de connexion des clients

ainsi que des profils errants pour des clients Windows 95/98/Me

```
logon home = \\%L\%u\winprofile\%m
```

Cet emplacement est relatif au répertoire de connexion de l'utilisateur. Les variables *%L*, *%u* et *%m* contiennent respectivement le nom du serveur, le nom de l'utilisateur et le nom du système client. Elles sont intéressantes pour constituer des noms paramétrés pour ces répertoires.

11.8.3.b Sections complémentaires

[netlogon]

Cette section est obligatoire pour permettre les connexions de type domaine.

Son rôle est de décrire l'emplacement de stockage des scripts de connexion (*logon scripts*).

```
[netlogon]
     path = /etc/samba/netlogon
     writable = no
     browsable = no
```

Ce partage n'a pas besoin d'apparaître dans la liste du voisinage réseau. Nous lui associons donc l'option *browsable = no*. Il ne nécessite pas non plus le droit d'écriture. Nous le précisons via l'option *writable = no*.

[profiles]

Cette section décrit l'emplacement de stockage des profils (*roaming profiles*) pour les utilisateurs Windows NT/2000/XP.

```
[profiles]
     path = /etc/samba/ntprof
     browsable = no
     writable = yes
     create mask = 0600
     directory mask = 0700
```

Ce partage n'a pas besoin d'apparaître dans la liste du voisinage réseau. Nous lui associons donc l'option *browsable = no*. Il nécessite le droit d'écriture. Nous le précisons via l'option *writable = yes*.

Ce droit d'écriture est affiné avec deux nouvelles options.

create mask Gestion des permissions pour les nouveaux fichiers

create mode Synonyme

```
     create mask = 0600
```

Cette option adopte les notations octales des commandes Unix pour indiquer les droits par défaut associés aux nouveaux fichiers. Dans l'exemple, la valeur *0600* précise le masque de permission pour les nouveaux fichiers (lecture/écriture pour le propriétaire seulement).

directory mask Gestion des permissions pour les nouveaux répertoires

directory mode Synonyme

```
     directory mask = 0700
```

Cette option adopte les notations octales des commandes Unix pour indiquer les droits par défaut associés aux nouveaux répertoires. Dans l'exemple, la valeur *0700* précise le

masque de permission pour les nouveaux répertoires (lecture/écriture/exécution pour le propriétaire seulement).

[homes]

Cette section correspond à la possibilité de partager les répertoires de connexion des utilisateurs sans avoir à les énumérer individuellement dans le fichier *smb.conf*. Sa présence est nécessaire pour que les options *logon drive* et *logon home* soient opérationnelles.

```
[homes]
     read only = no
     browsable = no
```

Ce partage n'a pas besoin d'apparaître dans la liste du voisinage réseau. Nous lui associons donc l'option *browsable = no*. Il nécessite bien sûr le droit d'écriture. Nous le précisons ici via l'option *read only = no*.

11.8.3.c Test du fichier de configuration

Comme notre fichier de configuration devient plus fourni, il est temps de présenter l'outil **testparm** qui permet d'opérer un contrôle syntaxique du fichier *smb.conf*.

Il est conseillé de faire appel à cette commande après chaque modification du fichier pour détecter d'éventuelles erreurs ou incohérences.

```
# testparm
Load smb config files from /etc/samba/smb.conf
Processing section "[homes]"
Processing section "[printers]"
Processing section "[print$]"
Processing section "[test]"
Loaded services file OK.
Server role: ROLE_DOMAIN_PDC
Press enter to see a dump of your service definitions

……..
```

La commande comporte un certain nombre d'options (voir le guide de référence). L'option *-s* permet de ne pas avoir à saisir une entrée clavier pour visualiser les sections après vérification. L'option *-x* demande de n'afficher que les paramètres qui n'ont pas leur valeur par défaut..

Il faut penser à créer les répertoires mentionnés dans les sections *[netlogon]* et *[profiles]* car l'utilitaire *testparm* ne vérifie pas leur présence.

11.8.3.d Configuration des postes clients Windows

Un système Windows NT/2000/XP doit au préalable être défini comme membre du domaine. Cela nécessite la création d'un **compte d'ordinateur** au niveau du PDC.

Tout d'abord, du point de vue de Samba, le compte Windows **Administrateur de domaine** va correspondre au compte **root**. Ce dernier doit donc être ajouté, une fois pour toutes, au fichier des mots de passe Samba. Il est recommandé d'utiliser un autre mot de passe que le véritable mot de passe de l'administrateur Unix.

Ensuite, un compte d'ordinateur doit être ajouté au fichier *smbpasswd*. Ce traitement est réalisé automatiquement par le serveur Samba lors de l'ajout du compte d'ordinateur depuis le client Windows. Pour le différencier d'un compte utilisateur, un compte d'ordinateur est constitué du nom NetBIOS complété par le caractère **$** (dollar).

Une seconde étape consiste à créer l'entrée correspondante dans le fichier Unix */etc/passwd*. Ce traitement peut être manuel mais il est préférable de provoquer un déclenchement automatique lors de l'ajout du compte d'ordinateur depuis le client Windows. Ceci est le rôle de l'option globale **add user script**.

L'option globale **logon script** précise l'emplacement (relatif au répertoire racine du partage *[netlogon]*) des scripts d'ouverture de session. La section **[netlogon]** est obligatoire pour permettre les connexions de type domaine.

Ce partage n'a pas besoin d'apparaître dans la liste du voisinage réseau et il ne nécessite pas le droit d'écriture. En effet, le répertoire des scripts de connexion ne doit être modifiable que par les administrateurs.

Dans notre exemple, nous avons choisi de constituer un fichier **logon.bat**.

L'extension *.bat* est importante pour pouvoir désigner un fichier de commandes quelle que soit la version Windows. Il faudra veiller à ce que ce fichier soit bien au format texte Windows. Contrairement au format Unix où les lignes sont délimitées par le seul caractère *\n* (saut de ligne), le format Windows doit intégrer les deux caractères *\r* (retour chariot) et *\n* (saut de ligne).

Plusieurs solutions sont possibles pour arriver à ce résultat :

- constituer le fichier sous Windows et le recopier ensuite (par exemple via une session *ftp* en mode *binary*),

- constituer le fichier sous Unix et le transformer (via un ordre *Perl* ou un ordre *sed*).

```
     logon script = logon.bat

[netlogon]
     path = /etc/samba/netlogon
     writable = no
     browsable = no
```

Dans le contexte d'un domaine, un utilisateur peut se connecter sur plusieurs ordinateurs en souhaitant retrouver toujours le même environnement. Toutes les versions Windows permettent une configuration locale personnalisée, stockée à la fois dans la base de registres et dans des fichiers.

Ces préférences s'appellent des **profils locaux**. Les fichiers correspondants sont stockés à des endroits spécifiques à chaque version :

Windows 95/98/Me	C:\windows\profiles
Windows NT	C:\winnt\profiles
Windows 2000/XP	C:\Documents and Settings

Les profils stockés sur un serveur sont baptisés **profils errants** (*roaming profiles*) car l'utilisateur va les retrouver sur l'ordinateur depuis lequel il se connecte. Pour ce faire, le profil est téléchargé, pendant la connexion, depuis le contrôleur de domaine. Lorsque l'utilisateur quitte le domaine, le profil local est recopié sur le serveur et validé comme nouveau profil centralisé. Il n'y a pas de mises à jour avant la déconnexion de l'utilisateur et une certaine confusion peut se produire dans le cas où un utilisateur serait connecté simultanément sur plusieurs postes clients.

De nombreux autres problèmes liés aux profils errants peuvent perturber le bon fonctionnement de la connexion mais cela dépasse le cadre de notre exposé et nous considérerons que les aspects Microsoft sont correctement gérés.

Les profils ne sont pas gérés de la même façon sur les versions Windows 95/98/Me et sur les versions Windows NT/2000/XP. Au niveau de Samba, deux endroits différents doivent être prévus pour stocker les fichiers correspondants.

L'option globale **logon path** précise l'emplacement des profils pour des clients Windows NT/2000/XP. Cet emplacement est relatif au répertoire racine du partage *[profiles]*. Nous avons choisi, dans notre exemple, le répertoire */etc/samba/ntprof*. Il serait possible (et sans doute préférable) de choisir un sous-répertoire de */home* pour stocker ces fichiers dans la même arborescence que les répertoires de connexion et faciliter ainsi les sauvegardes. Nous choisissons, en outre, de placer les profils dans des sous-répertoires portant les noms des utilisateurs et de bien distinguer le nom du poste client.

```
      logon path = \\%L\profiles\%u\%m

[profiles]
      path = /etc/samba/ntprof
      browsable = no
      writable = yes
      create mask = 0600
      directory mask = 0700
```

L'option globale **logon home** précise l'emplacement des profils pour des clients Windows 95/98/Me. Cet emplacement est relatif au répertoire de connexion de l'utilisateur. Nous supposons, par ailleurs, qu'une section *[homes]* génère le partage adéquat.

La valeur doit commencer obligatoirement par la notation *\\%L\%u* pour permettre au client d'associer une unité à son répertoire de connexion via la commande :

```
NET USE  H: /home
```

La lettre choisie doit être celle indiquée par l'option **logon drive**.

Ensuite, nous choisissons de placer les profils dans un sous-répertoire dédié baptisé *winprofile* et de bien distinguer le nom du poste client.

Il est important que les profils soient stockés dans un répertoire dédié car leur mise à jour suppriment les fichiers qui n'en font plus partie. Tous les répertoires seront créés automatiquement à la première déconnexion du client. Il faut simplement veiller à bien créer le répertoire de stockage des profils.

```
      logon drive = H:
      logon home = \\%L\%u\winprofile\%m

[homes]
      read only = no
      browsable = no
```

12. Le serveur Web Apache

12.1 Le projet Apache

La fondation Apache est à l'origine du développement du serveur Web de même nom mais elle gère aujourd'hui de nombreux autres projets *Open Source*. Le site officiel **www.apache.org** recense et décrit ces différentes activités.

Le projet, rebaptisé aujourd'hui *HTTP Server* (site **httpd.apache.org**), correspond au serveur Web le plus utilisé aujourd'hui sur le réseau Internet. Apache maintient cette position dominante depuis de longues années.

À l'origine, le serveur Apache est une évolution du serveur HTTP du NCSA (*National Center for Supercomputing Applications*). Officiellement, le nom Apache a été choisi en hommage à la célèbre tribu indienne, réputée pour son habileté guerrière et son endurance. D'autres sources, plus officieuses, font référence au terme *A PAtCH* dans la mesure où le projet a consisté, au départ, à corriger de nombreux bogues sur le serveur historique du NCSA.

Apache consiste en un exécutable de base qui sera complété, à volonté, par un ensemble de **modules**. Un module peut être considéré comme un programme que l'on ajoute à l'exécutable de base pour bénéficier de fonctionnalités supplémentaires. Un module peut être ajouté de manière statique dès la compilation ou bien de manière dynamique lors du démarrage du serveur.

Les versions

Aujourd'hui, Apache est disponible en deux versions majeures.

Les versions **Apache 1.3** sont destinées aux systèmes Unix/Linux. Elles correspondent à un énorme parc de serveurs installés. Sur des systèmes autres qu'Unix, Apache 1.3 n'est pas officiellement recommandé pour les sites de production.

Les versions **Apache 2** (**2.0** puis aujourd'hui **2.2**) ont été disponibles à partir d'avril 2002. Elles concrétisaient une nouvelle orientation du produit dont l'objectif majeur était de ne plus se cantonner aux seules plates-formes Unix/Linux. Dans tous les cas, les versions Apache 2 apportent bon nombre d'améliorations internes, notamment en terme de performances.

La plus importante évolution concerne l'implémentation des modules avec notamment l'introduction de **modules multi-traitements** (*MPMs*), spécifiques aux systèmes d'exploitation. Ces modules particuliers utilisent directement les bibliothèques système plutôt que de recourir à des couches d'émulation. Sur des systèmes autres qu'Unix, la conséquence en est, d'une part, une meilleure stabilité et, d'autre part, de meilleures performances. Sur les plate-formes Windows particulièrement, Apache est une alternative très séduisante au logiciel *Microsoft IIS*.

Dans un contexte de production, le choix de continuer à utiliser les versions Apache 1.3 peut encore se justifier par la crainte d'éventuelles incompatibilités de logiciels existants avec la nouvelle implémentation interne des modules.

Dans tous les cas, y compris dans le contexte Unix/Linux, Apache 2 apporte de meilleures performances. Il est notamment possible de choisir une compilation débouchant sur un modèle d'exécution hybride, à la fois multi-processus et *multithreads*.

12.2 Le protocole HTTP

L'acronyme **HTTP** signifie *HyperText Transfer Protocol*. Ce protocole s'appuie sur TCP/IP et utilise le port 80 par défaut.

Initialement limité au transfert de texte (version **0.9**), les versions actuelles du protocole HTTP (**1.0** et **1.1**) permettent le transfert d'informations multimédia grâce à l'apport des types **MIME** (*Multipurpose Internet Mail Extension*). L'acronyme MIME désigne, à l'origine, une extension du protocole de messagerie SMTP (*Simple Mail Transfer Protocol*). Cette extension s'applique aujourd'hui à bien d'autres protocoles réseau, dont HTTP.

Lors de sa réponse au client navigateur, le serveur Web utilise les spécifications MIME pour préciser le format des données. Celles-ci peuvent alors être interprétées correctement par le navigateur.

HTTP 1.0

Dans cette version, chaque requête est indépendante des autres. Les requêtes peuvent être de plusieurs types, tels que GET, HEAD et POST… Une requête **GET** correspond à une demande de transfert d'un document. Une requête **HEAD** permet d'obtenir des informations sur une ressource. Une requête **POST** propage au serveur les données saisies dans un formulaire.

La première ligne d'une requête comporte son type, le chemin vers la ressource demandée ainsi que la version de protocole. D'autres lignes optionnelles peuvent suivre et une ligne vide doit clôturer la requête.

HTTP 1.1

Parmi d'autres améliorations, la version 1.1 permet de grouper plusieurs échanges "requête-réponse" dans une même connexion TCP (connexion persistante).

Le nom de l'hôte virtuel souhaité sur le serveur est indiqué dans la requête. Un seul serveur peut en effet héberger plusieurs arborescences Web correspondant à des noms DNS différents.

Quelques codes retour

Dans la première ligne de sa réponse, le serveur HTTP intègre un code retour. Nous donnons ici un tableau des codes les plus fréquents.

Code	Message	Description
200	OK	Requête réussie
201	CREATED	Création réussie d'un nouveau document
202	ACCEPTED	Requête acceptée
203	PARTIAL INFORMATION	Réponse incomplète
204	NO RESPONSE	Requête reçue mais réponse vide
301	MOVED	Document déplacé de façon permanente
302	FOUND	Document déplacé temporairement
304	NOT MODIFIED	Document non modifié (GET conditionnel)
400	BAD REQUEST	Requête incorrecte syntaxiquement
401	UNAUTHORIZED	Problème d'autorisation
403	FORBIDDEN	Accès complètement interdit
404	NOT FOUND	Ressource introuvable
500	INTERNAL ERROR	Erreur système sur le serveur
501	NOT IMPLEMENTED	Opération demandée non disponible

12.3 Les modules

Terminologie et principes

L'exécutable Apache n'intègre que les fonctionnalités essentielles. Les aspects complémentaires sont donc implémentés sous forme de **modules**. La distribution Apache fournit elle-même un nombre important de modules. Le site *modules.apache.org* en propose beaucoup d'autres, non inclus dans la distribution officielle.

À chaque module, sont associées des **directives** qu'il faut placer dans des fichiers de configuration.

Un certain nombre de ces modules peuvent être intégrés au serveur au moment de la compilation. Il s'agit alors de modules **statiques**.

L'option *-l* du futur démon *httpd* permettra d'en avoir la liste.

```
# httpd -l
Compiled in modules:
  core.c
  worker.c
  http_core.c
  mod_so.c
#
```

D'autres modules seront chargés dynamiquement lors du lancement du serveur Apache car ils auront été compilés en tant que *Dynamic Shared Object (DSO)*.

Le module statique **mod_so** permet cette gestion dynamique.

D'autres modules *DSO,* non présents dans la distribution, pourront être intégrés ultérieurement à l'aide de l'outil **apxs** (APache eXtenSion) fourni avec Apache. Ainsi, il ne sera pas nécessaire de recompiler le serveur pour lui ajouter certaines fonctionnalités.

Un module correspond à un fichier source et à un identifiant dont les noms apparaîtront dans les fichiers de configuration dans le cadre d'une directive **<IfModule>**.

D'autre part, son identifiant sera utilisé dans une autre directive **LoadModule** afin de désigner une librairie dynamique (fichier d'extension *.so*) correspondant à sa forme compilée.

Nous pouvons distinguer plusieurs types de modules :

- **MPM** (*Multi Processing Module*)

Ces modules permettent à Apache d'exploiter au mieux les caractéristiques du système d'exploitation. Une occurrence de serveur Apache ne peut intégrer qu'un seul module de ce type, compilé de manière statique. Dans le cas des systèmes Unix/Linux, deux modules MPMs sont proposés :

- **prefork**

Ce module est choisi par défaut pour la compilation d'Apache sous Unix/Linux. Il correspond à un processus unique de contrôle, responsable de la création

des processus fils chargés de traiter les demandes de connexion. Apache se charge du meilleur choix en ce qui concerne le nombre de processus actifs à un instant donné.

- **worker**

Il s'agit de l'alternative au module *prefork* pour les plates-formes Unix/Linux. Ce module implémente un modèle hybride de processus et de threads. En effet, les processus fils créés par le processus de contrôle génèrent un nombre fixe de threads afin de gérer plus efficacement un nombre important de connexions.

- **Base**

Un module de ce type est intégré par défaut au serveur Apache à moins d'un choix contraire lors de la compilation.

- **Extension**

Inversement, un module de ce type n'est pas intégré par défaut au serveur Apache. Il convient de le sélectionner explicitement par une directive de compilation et ceci même si l'on décide de le compiler en tant que module dynamique.

- **Expérimental**

Ce type de module est fourni dans la distribution Apache sans garantie de fonctionnement en environnement de production.

- **Externe**

Il s'agit de modules tiers non inclus dans la distribution Apache.

12.4 Installation

Apache est, bien entendu, facilement disponible sous la forme de paquet logiciel. Il est toujours possible de compiler et d'installer soi-même le logiciel à l'aide de la distribution source. Nous pouvons ainsi bénéficier de la version la plus récente possible, configurable à volonté.

12.4.1 Étapes de l'installation

Dans un premier temps, nous avons téléchargé l'archive source depuis le site officiel *httpd.apache.org*.

Nous nous plaçons ensuite dans un répertoire de travail (par exemple, */usr/local/src*) pour décompresser puis restaurer cette archive.

```
/usr/local/src# ls -l httpd-2.2.6.tar.gz
-rw-r--r-- 1 root root 6028951 2007-09-24 17:01 httpd-2.2.6.tar.gz
/usr/local/src# tar xzf httpd-2.2.6.tar.gz
```

Nous allons ensuite nous positionner dans le sous-répertoire *httpd-2.2.6*, résultat de la restauration de l'archive. Le fichier **INSTALL** comporte les indications nécessaires à la suite des opérations.

```
/usr/local/src# cd httpd-2.2.6
/usr/local/src/httpd-2.2.6# more INSTALL
APACHE INSTALLATION OVERVIEW

  Quick Start - Unix
  ------------------

  For complete installation documentation, see
  [ht]docs/manual/install.html or
  http://httpd.apache.org/docs-2.2/install.html

     $ ./configure --prefix=PREFIX
     $ make
     $ make install
     $ PREFIX/bin/apachectl start
.........................
.........................
.........................
  The easiest way to find all of the configuration flags
  for Apache 2.2 is to run ./configure --help.

.....................................................................
```

12.4.2 Mise en oeuvre

La première étape (*configure*) est très importante, comme toujours dans la démarche de compilation d'un logiciel Open Source. Elle nous permet de sélectionner les modules que nous souhaitons intégrer et de choisir leur mode de compilation (statique ou dynamique).

L'option **--help** donne le détail des opérations disponibles. La liste est longue et assez fastidieuse à consulter mais il s'agit bien là d'opérer les sélections importantes.

Il est à noter que le fichier **config.layout** de l'archive source permet de choisir des types de configuration déjà préparées en terme d'emplacement de répertoires. Ce fichier est, bien entendu, paramétrable à volonté.

```
# more config.layout
##
##  config.layout -- Pre-defined Installation Path Layouts
##
#   Classical Apache path layout.
<Layout Apache>
    prefix:        /usr/local/apache2
    exec_prefix:   ${prefix}
    bindir:        ${exec_prefix}/bin
    sbindir:       ${exec_prefix}/bin
    libdir:        ${exec_prefix}/lib
    libexecdir:    ${exec_prefix}/modules
    mandir:        ${prefix}/man
    sysconfdir:    ${prefix}/conf
    datadir:       ${prefix}
    installbuilddir: ${datadir}/build
    errordir:      ${datadir}/error
    iconsdir:      ${datadir}/icons
    htdocsdir:     ${datadir}/htdocs
    manualdir:     ${datadir}/manual
    cgidir:        ${datadir}/cgi-bin
    includedir:    ${prefix}/include
    localstatedir: ${prefix}
    runtimedir:    ${localstatedir}/logs
    logfiledir:    ${localstatedir}/logs
    proxycachedir: ${localstatedir}/proxy
</Layout>
```

……

Après le choix des répertoires d'installation, il faut s'intéresser à la sélection des modules.

Les modules non compilés par défaut sont identifiables par :

```
./configure --help | grep enable-
```

Inversement, les module inclus par défaut sont identifiables par :

```
./configure --help | grep disable-
  --disable-FEATURE       do not include FEATURE (same as --enable-FEATURE=no)
  --disable-authn-file    file-based authentication control
  --disable-authn-default authentication backstopper
  --disable-authz-host    host-based authorization control
  --disable-authz-groupfile
  --disable-authz-user    'require user' authorization control
  --disable-authz-default authorization control backstopper
  --disable-auth-basic    basic authentication
  --disable-include       Server Side Includes
  --disable-filter        Smart Filtering
```

```
  --disable-charset-lite  character set translation
  --disable-log-config    logging configuration
  --disable-env           clearing/setting of ENV vars
  --disable-setenvif      basing ENV vars on headers
  --disable-mime          mapping of file-extension to MIME
  --disable-status        process/thread monitoring
  --disable-autoindex     directory listing
  --disable-asis          as-is filetypes
  --disable-cgid          CGI scripts
  --disable-cgi           CGI scripts
  --disable-negotiation   content negotiation
  --disable-dir           directory request handling
  --disable-actions       Action triggering on requests
  --disable-userdir       mapping of requests to user-specific directories
  --disable-alias         mapping of requests to different filesystem parts
```

L'option **--with-mpm=worker** constitue une alternative au module par défaut *prefork* pour les plates-formes Unix/Linux. Ce module *worker* implémente un modèle hybride de processus et de threads. Les processus fils créés par le processus de contrôle génèrent un nombre fixe de threads afin de gérer plus efficacement un nombre important de connexions.

L'option **--enable-mods-shared** permet de préciser quels sont les modules à compiler en tant que *DSO* (Dynamic Shared Object). A priori, nous lui donnerons la valeur **all** afin d'opérer un choix global à ce niveau. Il est à noter que certains modules ne sont curieusement pas sélectionnés par le mot clé *all* et qu'ils doivent faire l'objet d'une sélection explicite. C'est la cas, par exemple, du module **mod_ssl** qui permettra la mise en œuvre de connexions sécurisées (URLs de type *https://*).

```
# ./configure --with-mpm=worker --enable-ssl --enable-mods-shared=all

……..
```

Après l'exécution de la commande *configure*, les fichiers **config.status** et **config.log** contiennent les traces des traitements effectués. Ils peuvent aider à résoudre d'éventuels problèmes. En cas de succès, le fichier **Makefile** est le fichier résultat, exploité ensuite par la commande **make**.

```
/usr/local/src/httpd-2.2.6# more config.log
This file contains any messages produced by compilers while
running configure, to aid debugging if configure makes a mistake.

It was created by configure, which was
generated by GNU Autoconf 2.60.  Invocation command line was

./configure --with-mpm=worker --enable-ssl --enable-mods-shared=all

## --------- ##
## Platform. ##
## --------- ##

hostname = debian1
```

```
uname -m = i686
uname -r = 2.6.18-4-686
uname -s = Linux
uname -v = #1 SMP Wed May 9 23:03:12 UTC 2007
```

……..

Nous allons opérer maintenant la compilation effective.

```
/usr/local/src/httpd-2.2.6# make
```

……..

En cas de succès, nous pouvons opérer l'installation du logiciel dans les répertoires cibles.

```
/usr/local/src/httpd-2.2.6# make install
 make install
Making install in srclib
make[1]: entrant dans le répertoire «/usr/local/src/httpd-2.2.6/srclib»
Making install in apr
make[2]: entrant dans le répertoire «/usr/local/src/httpd-2.2.6/srclib/apr»
.............................
.............................
mkdir /usr/local/apache2
mkdir /usr/local/apache2/lib
mkdir /usr/local/apache2/bin
mkdir /usr/local/apache2/build
mkdir /usr/local/apache2/lib/pkgconfig
mkdir /usr/local/apache2/include
.............................
.............................

Installing configuration files
mkdir /usr/local/apache2/conf
mkdir /usr/local/apache2/conf/extra
mkdir /usr/local/apache2/conf/original
mkdir /usr/local/apache2/conf/original/extra
Installing HTML documents
mkdir /usr/local/apache2/htdocs
Installing error documents
mkdir /usr/local/apache2/error
Installing icons
mkdir /usr/local/apache2/icons
mkdir /usr/local/apache2/logs
Installing CGIs
mkdir /usr/local/apache2/cgi-bin
Installing header files
Installing build system files
Installing man pages and online manual
mkdir /usr/local/apache2/man
mkdir /usr/local/apache2/man/man1
mkdir /usr/local/apache2/man/man8
mkdir /usr/local/apache2/manual
make[1]: quittant le répertoire « /usr/local/src/httpd-2.2.6 »
/usr/local/src/httpd-2.2.6#
```

Nous avons choisi de conserver l'emplacement par défaut pour Apache. Nous pouvons donc constater la création du répertoire **/usr/local/apache2** contenant lui-même toute la sous-arborescence du logiciel.

```
/usr/local/src/httpd-2.2.6# cd /usr/local/apache2
/usr/local/apache2# ls -l
total 52
drwxr-sr-x  2 root staff 4096 2007-09-24 18:44 bin
drwxr-sr-x  2 root staff 4096 2007-09-24 18:44 build
drwxr-sr-x  2 root staff 4096 2007-09-24 18:44 cgi-bin
drwxr-sr-x  4 root staff 4096 2007-09-24 18:44 conf
drwxr-sr-x  3 root staff 4096 2007-09-24 18:44 error
drwxr-sr-x  2 root staff 4096 2007-09-24 18:44 htdocs
drwxr-sr-x  3 root staff 4096 2007-09-24 18:44 icons
drwxr-sr-x  2 root staff 4096 2007-09-24 18:44 include
drwxr-sr-x  3 root staff 4096 2007-09-24 18:44 lib
drwxr-sr-x  2 root staff 4096 2007-09-24 18:44 logs
drwxr-sr-x  4 root staff 4096 2007-09-24 18:44 man
drwxr-sr-x 14 root staff 4096 2007-09-24 18:44 manual
drwxr-sr-x  2 root staff 4096 2007-09-24 18:44 modules
/usr/local/apache2#
```

Le serveur Apache correspondra au processus **httpd**. Ce processus sera démarré à l'aide d'un script baptisé **apachectl**. Le processus *httpd* s'exécutera sous l'identité de l'utilisateur **root** mais générera des processus fils qui devront, pour leur part, s'exécuter avec l'identité d'un compte et d'un groupe dédié au logiciel Apache. Ceci est fondamental en terme de sécurité de base.

Nous allons donc, dans un premier temps, sécuriser l'arborescence en donnant tous les fichiers au compte et au groupe *root*. Dans un second temps, nous allons créer un utilisateur et un groupe dédié au logiciel Apache. Nous décidons de les nommer a*pache2*.

```
/usr/local/apache2# chown -R root:root .

/usr/local/apache2# groupadd -g 198 apache2
/usr/local/apache2# useradd -u 198 -g apache2 apache2
/usr/local/apache2#
```

Nous avons invoqué la commande *useradd* sans enchaîner par la commande *passwd*. De ce fait, l'utilisateur *apache2* ne possède pas de mot de passe correct et il est impossible de l'atteindre par une connexion interactive. Ceci est une bonne chose puisqu'il ne sera invoqué que de manière sous-jacente, depuis un contexte *root* via la commande *su*.

Après l'installation, le serveur Apache est quasiment opérationnel. La configuration est obtenue par de nombreuses directives placées dans des fichiers de texte. Le fichier essentiel est le fichier **httpd.conf** du sous-répertoire *conf* de l'arborescence Apache.

Avant de faire un premier essai de démarrage du serveur, nous devons modifier ce fichier de configuration principal pour mentionner notre utilisateur et notre groupe créés précédemment;

```
/usr/local/apache2# cd conf
```

```
/usr/local/apache2/conf# vi httpd.conf
```

```
.....................................................................

#
# If you wish httpd to run as a different user or group, you must run
# httpd as root initially and it will switch.
#
# User/Group: The name (or #number) of the user/group to run httpd as.
# It is usually good practice to create a dedicated user and group for
# running httpd, as with most system services.
#
User daemon
Group daemon

.....................................................................
```

Nous allons modifier les deux lignes génériques précédentes qui deviennent donc :

```
User apache2
Group apache2
```

12.4.3. Premiers tests

Le script **apachectl** permet de gérer le lancement et l'arrêt du serveur Apache en adoptant la sémantique des scripts de démarrage des systèmes Unix/Linux. Si nous décidons d'officialiser notre installation, il nous suffira de placer ce script dans le répertoire */etc/init.d* et de choisir les niveaux d'exécution où nous souhaitons démarrer le serveur.

Pour l'instant, nous opérons un premier démarrage explicite.

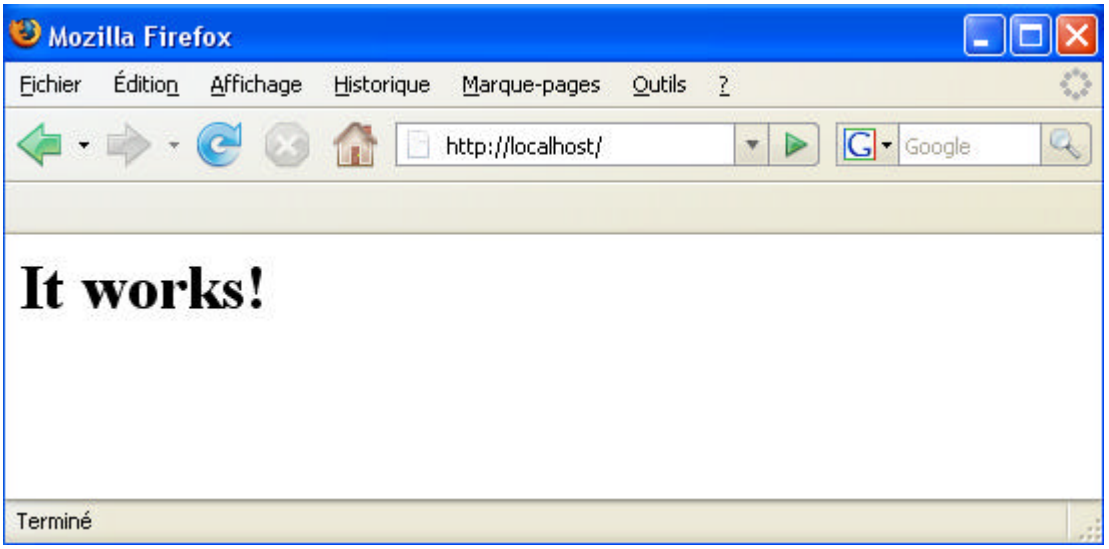
```
# /usr/local/apache2/bin/apachectl start
httpd: Could not reliably determine the server's fully qualified domain
name, using 127.0.1.1 for ServerName
#
```

Nous avons obtenu un message d'avertissement concernant le nom réseau de notre serveur car notre machine de test ne possède pas pour l'instant de nom DNS complètement qualifié. Ceci ne sera pas le cas, bien entendu, d'un serveur réellement en production. Nous corrigerons ce problème à l'aide de la directive *ServerName*, étudiée dans la suite de ce chapitre.

Apache a bien démarré malgré tout, comme la liste des processus vient nous le confirmer.

```
# ps -ef | grep httpd
root   3972 1    0 14:41  ? 00:00:00  /usr/local/apache2/bin/httpd -k start
apache2 3973 3972 0 14:41 ? 00:00:00  /usr/local/apache2/bin/httpd -k start
apache2 3974 3972 0 14:41 ? 00:00:00  /usr/local/apache2/bin/httpd -k start
apache2 3976 3972 0 14:41 ? 00:00:00  /usr/local/apache2/bin/httpd -k start
apache2 3978 3972 0 14:41 ? 00:00:00  /usr/local/apache2/bin/httpd -k start
#
```

Nous pouvons maintenant tenter une première connexion depuis notre client navigateur avec l'URL **http://localhost**. Nous devons obtenir la page d'accueil très explicite du serveur fraîchement installé.



Page d'accueil du serveur Apache 2.2 après installation

Il est possible d'invoquer directement le programme **httpd** pour obtenir quelques informations.

L'option **-l** donne la liste des modules compilés de manière statique. Notre liste est courte puisque nous avions choisi de compiler tous les modules en tant que modules dynamiques. Les modules **core** et **http_core** sont incontournables et correspondent aux fonctionnalités toujours disponibles. Le module **mod_so** est également obligatoire pour permettre la gestion des modules dynamiques. Le module **worker** est donc le module de type MPM (*Multi Processing Module*) que nous avons préféré au choix par défaut *prefork*. Il nous indique que Apache fonctionnera selon un modèle hybride de processus et de threads.

```
# httpd -l
Compiled in modules:
  core.c
  worker.c
  http_core.c
  mod_so.c
#
```

L'option **-M** (disponible depuis Apache 2.2) fait la liste complète des modules et nous indique aussi ceux compilés de manière dynamique.

```
# httpd -M
Loaded Modules:
 core_module (static)
 mpm_worker_module (static)
 http_module (static)
```

```
 so_module (static)
 authn_file_module (shared)
 authn_dbm_module (shared)
 authn_anon_module (shared)
 authn_dbd_module (shared)
 authn_default_module (shared)
 authz_host_module (shared)
 authz_groupfile_module (shared)
 authz_user_module (shared)
 authz_dbm_module (shared)
 authz_owner_module (shared)
 authz_default_module (shared)
 auth_basic_module (shared)
 auth_digest_module (shared)
.............................
.............................
 imagemap_module (shared)
 actions_module (shared)
 speling_module (shared)
 userdir_module (shared)
 alias_module (shared)
 rewrite_module (shared)
Syntax OK
#
```

L'option **-v** donne clairement la version du serveur ainsi que sa date de compilation.

```
# httpd -v
Server version: Apache/2.2.6 (Unix)
Server built:   Sep 24 2007 18:37:34
#
```

L'option **-V** donne plus de précisions sur la configuration du serveur.

```
# httpd -V
Server version: Apache/2.2.6 (Unix)
Server built:   Sep 24 2007 18:37:34
Server's Module Magic Number: 20051115:5
Server loaded:  APR 1.2.11, APR-Util 1.2.10
Compiled using: APR 1.2.11, APR-Util 1.2.10
Architecture:   32-bit
Server MPM:     Worker
  threaded:     yes (fixed thread count)
    forked:     yes (variable process count)
Server compiled with....
 -D APACHE_MPM_DIR="server/mpm/worker"
 -D APR_HAS_SENDFILE
 -D APR_HAS_MMAP
 -D APR_HAVE_IPV6 (IPv4-mapped addresses enabled)
 -D APR_USE_SYSVSEM_SERIALIZE
 -D APR_USE_PTHREAD_SERIALIZE
 -D SINGLE_LISTEN_UNSERIALIZED_ACCEPT
 -D APR_HAS_OTHER_CHILD
 -D AP_HAVE_RELIABLE_PIPED_LOGS
 -D DYNAMIC_MODULE_LIMIT=128
```

```
 -D HTTPD_ROOT="/usr/local/apache2"
 -D SUEXEC_BIN="/usr/local/apache2/bin/suexec"
 -D DEFAULT_SCOREBOARD="logs/apache_runtime_status"
 -D DEFAULT_ERRORLOG="logs/error_log"
 -D AP_TYPES_CONFIG_FILE="conf/mime.types"
 -D SERVER_CONFIG_FILE="conf/httpd.conf"
#
```

12.4.4 Accès à la documentation

L'URL **http://localhost/manual/** permet d'accéder en local à la documentation qui est donc fournie avec Apache. Il nous faut pour cela décommenter une ligne dans le fichier httpd.conf.

```
# Local access to the Apache HTTP Server Manual
Include conf/extra/httpd-manual.conf
```

L'équivalent actualisé de cette documentation locale est également accessible depuis le site *httpd.apache.org*.

Les rubriques **Modules** et **Résumé des Diretives** donnent accès à une documentation de référence extrêmement détaillée. Quelques tutoriels et guides d'utilisation peuvent aider à synthétiser les démarches de mise en œuvre par rapport à la seule information de référence.

12.5 Les fichiers de configuration

Comme nous l'avons déjà exprimé et constaté, le serveur Apache est quasiment opérationnel, directement après son installation. Sa configuration est obtenue à l'aide de **directives** placées dans des fichiers au format texte. Chaque directive est associée à un **module**. Les noms des directives son insensibles à la casse même s'il est plus élégant et lisible de les écrire tel que le propose la documentation officielle.

Le principal fichier de configuration se nomme **httpd.conf**. Il est situé dans le sous-répertoire *conf* de l'arborescence Apache.

Des fichiers baptisés **.htaccess** pourront éventuellement apparaître dans certains répertoires de l'arborescence Web. Leur objectif sera de proposer une configuration décentralisée.

Dans ces fichiers, les lignes qui commencent par le caractère # (dièse) correspondent à des commentaires. Le fichier *httpd.conf* est abondamment auto-commenté.

Une ligne non commentée correspond à une directive. Cette directive peut éventuellement s'étendre sur plusieurs lignes à l'aide du caractère \ (antislash).

Une directive est valable dans certains **contextes**. Cela signifie qu'elle peut apparaître ou non à différents endroits du fichier de configuration. Pour désigner le contexte d'une directive, nous avons choisi de conserver les termes anglo-saxons tels qu'ils apparaissent dans la documentation de référence.

- **server config**

La directive peut apparaître dans le fichier *httpd.conf* mais pas dans les fichiers décentralisés *.htaccess*. Elle ne peut pas se trouver non plus dans les paragraphes <*VirtualHost*> ou <*Directory*>.

- **virtual host**

La directive peut apparaître dans les paragraphes <*VirtualHost*>. Le serveur Apache peut héberger plusieurs arborescences de pages Web correspondant à ce que l'on nommera des hôtes virtuels. Il y a plusieurs manières d'implémenter ces hôtes virtuels, soit selon l'adresse IP, soit selon le nom DNS.

- **directory**

La directive peut apparaître dans des paragraphes <*Directory*>, <*Location*> ou <*Files*>.

- **.htaccess**

La directive peut apparaître dans un fichier décentralisé *.htaccess*.

Certains de ces termes font référence à des thèmes non encore étudiés mais qui vont se clarifier tout au long de ce chapitre.

Le fichier principal peut intégrer d'autres fichiers de configuration à l'aide de la directive *Include*.

Include

Description	Inclusion de fichiers de configuration
Syntaxe	Include *file-path*|*directory-path*
Contexte	server config, virtual host, directory
Module	Core

L'inclusion d'un répertoire permet d'indiquer facilement un ensemble de fichiers, y compris ceux d'éventuels sous-répertoires. Les caractères spéciaux sont autorisés dans la désignation des fichiers. Les noms de fichiers et de répertoires peuvent être des chemins absolus ou bien des chemins relatifs à l'emplacement du serveur.

Le fichier **httpd.conf** des versions Apache **2.2** comporte un certain nombre d'inclusions de fichiers situés dans le **sous-répertoire extra** du répertoire conf.

```
# Supplemental configuration
#
# The configuration files in the conf/extra/ directory can be
# included to add extra features or to modify the default configuration of
# the server, or you may simply copy their contents here and change as
# necessary.

# Server-pool management (MPM specific)
#Include conf/extra/httpd-mpm.conf

# Multi-language error messages
#Include conf/extra/httpd-multilang-errordoc.conf

# Fancy directory listings
#Include conf/extra/httpd-autoindex.conf

# Language settings
#Include conf/extra/httpd-languages.conf

# User home directories
#Include conf/extra/httpd-userdir.conf

# Real-time info on requests and configuration
#Include conf/extra/httpd-info.conf

# Virtual hosts
#Include conf/extra/httpd-vhosts.conf

# Local access to the Apache HTTP Server Manual
Include conf/extra/httpd-manual.conf

# Distributed authoring and versioning (WebDAV)
#Include conf/extra/httpd-dav.conf

# Various default settings
#Include conf/extra/httpd-default.conf

# Secure (SSL/TLS) connections
#Include conf/extra/httpd-ssl.conf
```

La syntaxe des fichiers de configuration peut être vérifiée par l'outil *apachectl*.

```
# apachectl configtest
Syntax OK
#
# apachectl configtest
Syntax error on line 29 of /usr/local/apache2/conf/httpd.conf:
Invalid command 'SeverRoot', perhaps mis-spelled or defined by a module
not included in the server configuration
#
```

Dans ce deuxième exemple, la directive *ServerRoot* est mal orthographiée (*SeverRoot*).

Un autre fichier important **mime.types** décrit l'ensemble des types de documents gérés par le serveur.

```
/usr/local/apache2/conf# more mime.types
# This file controls what Internet media types are sent to the client for
# given file extension(s).  Sending the correct media type to the client
# is important so they know how to handle the content of the file.
# Extra types can either be added here or by using an AddType directive
# in your config files. For more information about Internet media types,
# please read RFC 2045, 2046, 2047, 2048, and 2077.  The Internet media type
# registry is at <http://www.iana.org/assignments/media-types/>.

.....................................................................
```

Les lignes de ce fichier décrivent les types MIME et les extensions associées.

Exemples

```
application/zip    zip
image/jpeg         jpeg jpg jpe
text/html          html htm
text/plain         asc txt
```

Le nom *mime.types* correspond en fait au nom de fichier par défaut précisé par la directive *TypesConfig*.

Types Config

Description	Nom du fichier des types MIME
Syntaxe	TypesConfig *file-path*
Valeur par défaut	TypesConfig conf/mime.types
Contexte	server config
Module	mod_mime

La directive *AddType* est également disponible pour ajouter ou surcharger des définitions de types MIME.

AddType

Description Définition de type MIME

Syntaxe AddType *MIME-type extension* [*extension*] ...

Contexte server config, virtual host, directory, .htaccess

Module mod_mime

Exemple

```
# AddType allows you to add to or override the MIME configuration
# file mime.types for specific file types.
AddType  application/x-gzip  .tgz
```

12.6 Premières directives essentielles

Nous allons présenter maintenant quelques directives importantes pour le bon fonctionnement du serveur Apache.

12.6.1 ServerRoot

Description Répertoire de base du serveur

Syntaxe ServerRoot *directory-path*

Contexte server config

Module Core

Cette directive indique l'emplacement du serveur Apache. Les chemins ne commençant pas par le caractère / seront des chemins relatifs à cet emplacement, tels que conf/ et logs/. Il est possible de démarrer le processus *httpd* avec l'option *-d* pour indiquer un emplacement autre que celui mentionné par la directive.

Exemple

```
# ServerRoot: The top of the directory tree under which the server's
# configuration, error, and log files are kept.
# Do NOT add a slash at the end of the directory path.
ServerRoot  "/usr/local/apache2"
```

Le serveur Apache est démarré par l'utilisateur *root* pour ensuite s'exécuter sous l'identité d'un compte prévu à cet effet lors de l'installation. Il est nécessaire que le répertoire *ServerRoot* ainsi que les sous-répertoires bin, conf et logs ne comportent pas le droit d'écriture pour un utilisateur autre que *root*. Les fichiers ordinaires doivent suivre le mêmes contraintes.

12.6.2 Listen

Description Adresse IP et port d'écoute

Syntaxe Listen [*IP-address*:]*portnumber*

Contexte server config

Module Modules de type MPM

Cette directive permet de préciser les adresses IP et les ports sur lesquels le serveur doit écouter les requêtes. Si aucune adresse IP n'est précisée, Apache peut utiliser toutes les interfaces disponibles. Il est donc quelquefois utile de préciser une interface particulière.

Même si le port 80 est le port par défaut, la directive est obligatoire dans les versions Apache 2. En son absence, le démarrage du serveur serait impossible :

```
# apachectl start
```

```
no listening sockets available, shutting down
Unable to open logs
#
```

Plusieurs directives peuvent apparaître pour préciser plusieurs ports éventuels d'écoute.

Exemple

```
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default.
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses (0.0.0.0)
#
#Listen 12.34.56.78:80

Listen 80
Listen 8080
```

Dans cet exemple, on se donne la possibilité de contacter le serveur avec une URL telle que *http://www.mondomaine.com:8080*.

12.6.3 ServerName

Description Nom DNS du serveur et port utilisé par défaut

Syntaxe ServerName *fully-qualified-domain-name*[:*port*]

Contexte server config, virtual host

Module Core

Il est fortement recommandé de renseigner cette directive (nom + port). Elle précise le nom d'hôte et le port qu'utilise le serveur pour s'identifier lui-même. Ces informations sont utilisées notamment pour créer des URLs de redirection. En l'absence de la directive, le serveur tenterait de résoudre son nom à partir de sa propre adresse IP (résolution DNS inverse) et utiliserait le numéro de port précisé dans la requête.

Exemple

```
#
# ServerName gives the name and port that the server uses to identify itself.
# This can often be determined automatically, but we recommend you specify
# it explicitly to prevent problems during startup.
#
# If your host doesn't have a registered DNS name, enter its IP address here.
#
#ServerName www.example.com:80
ServerName www.mondomaine.com:80
```

12.6.4 User

Description Utilisateur propriétaire des processus Apache

Syntaxe	User *unix-userid*
Valeur par défaut	User #-1
Contexte	server config
Module	Modules de type MPM

La directive définit l'identité de l'utilisateur sous laquelle s'exécuteront les processus *httpd*. Le processus initial sera démarré en tant que *root* (par le script *apachectl*) pour ensuite lancer les autres processus sous cette nouvelle identité. L'utilisateur peut être désigné par son nom ou par son numéro précédé d'un # (dièse).

12.6.5 Group

Description	Groupe propriétaire des processus Apache
Syntaxe	Group *unix-group*
Valeur par défaut	Group #-1
Contexte	server config
Module	Modules de type MPM

La directive définit l'identité du groupe sous laquelle s'exécuteront les processus *httpd*. Le groupe peut être désigné par son nom ou par son numéro précédé d'un # (dièse).

Exemple

```
#
# If you wish httpd to run as a different user or group, you must run
# httpd as root initially and it will switch.
#
# User/Group: The name (or #number) of the user/group to run httpd as.
# It is usually good practice to create a dedicated user and group for
# running httpd, as with most system services.
#
#User daemon
#Group daemon
User apache2
Group apache2
```

12.6.6 ServerAdmin

Description	Adresse e-mail incluse dans les messages d'erreur
Syntaxe	ServerAdmin *email-address*
Valeur par défaut	ServerAdmin you@example.com
Contexte	server config, virtual host
Module	Core

La directive mentionne une adresse e-mail à faire apparaître dans certaines pages de messages d'erreur. Il est recommandé de créer une adresse dédiée à cette fonctionnalité.

Exemple

```
# ServerAdmin: Your address, where problems with the server should be
# e-mailed.  This address appears on some server-generated pages, such
# as error documents.  e.g. admin@your-domain.com
#
#ServerAdmin you@example.com
ServerAdmin  www-admin@mondomaine.com
```

12.6.7 DocumentRoot

Description	Répertoire des pages Web
Syntaxe	DocumentRoot *directory-path*
Valeur par défaut	DocumentRoot /usr/local/apache2/htdocs
Contexte	server config, virtual host
Module	Core

Cette directive indique l'emplacement de l'arborescence Web accessible par les clients. Le nom de répertoire ne doit pas comporter de caractère / final.

Exemple

```
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot  "/usr/local/apache2/htdocs"
```

Le répertoire *DocumentRoot* peut éventuellement être modifiable par des comptes autres que *root* selon la politique choisie pour la mise à jour des pages Web.

12.6.8 Alias

Description	Accès à des arborescences hors de *DocumentRoot*
Syntaxe	Alias *URL-path file-path*|*directory-path*
Contexte	server config, virtual host
Module	mod_alias

Cette directive permet de donner l'accès à des documents stockés en-dehors de l'arborescence définie par la directive *DocumentRoot*.

Exemple

```
Alias  /icons/  "/usr/local/apache2/icons/"
```

Dans cet exemple, si nous utilisons l'URL *http://www.monserveur.com/icons/*, le serveur nous donnera en fait accès aux fichiers situés dans le répertoire */usr/local/apache2/icons*.

Si la définition de l'alias se termine par le caractère /, celui-ci devra être fourni dans la requête du navigateur. En d'autres termes, si nous demandions l'URL *http://www.monserveur.com/icons* (sans le / final), cela provoquerait une erreur.

Il est tout à fait possible de définir des alias qui ne comportent pas ce caractère / final.
Exemple

```
Alias  /image /ftp/pub/image
```

12.6.9 Directives concernant les modules

LoadModule

Description	Chargement dynamique d'un module
Syntaxe	Loadmodule *module filename*
Contexte	server config
Module	mod_so

Grâce au module statique *mod_so*, cette directive permet le chargement dynamique d'un module de type DSO (Dynamic Shared Object). Elle effectue le lien entre l'identifiant du module et l'emplacement du fichier objet correspondant au module compilé.

Exemples

```
LoadModule  ssl_module   modules/mod_ssl.so
LoadModule  php5_module  modules/libphp5.so
```

Le nom du répertoire *modules*, mentionné dans ces exemples, ne commence pas par le caractère /. Il est donc relatif au répertoire désigné par la directive *ServerRoot*.

IfModule

Description	Configuration liée à la présence d'un module
Syntaxe	<IfModule [!]*module-name*> ... </IfModule>
Contexte	server config, virtual host, directory, .htaccess
Module	Core

Cette directive définit un bloc dans lequel les directives sont prises en compte selon la présence ou non d'un module. Le module est désigné par le nom du fichier source ou par son identifiant.

Exemple

```
<IfModule dir_module>
    DirectoryIndex index.html
</IfModule>
```

12.7 Sécurisation des répertoires

12.7.1 Directory

Description	Regroupement de directives pour un répertoire
Syntaxe	<Directory *directory-path*> ... </Directory>
Contexte	server config, virtual host
Module	Core

Cette directive permet d'associer un ensemble de directives à un répertoire et à tous ses sous-répertoires. Le nom de répertoire est un chemin complet. Ce chemin peut comporter des caractères spéciaux tels que ceux du shell Unix.

Exemples

```
<Directory "/usr/local/apache2/htdocs">
    Options Indexes FollowSymLinks
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>

<Directory /home/*/public_html>
    AllowOverride FileInfo AuthConfig Limit Indexes
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
</Directory>
```

Si plusieurs sections *<Directory>* désignent le répertoire d'un document ou l'un de ses répertoires parents, alors les directives seront tout naturellement appliquées les unes après les autres, selon la loi "du plus court chemin d'abord". Ces directives seront combinées à celles d'éventuels fichiers décentralisés *.htaccess* (voir plus loin dans ce chapitre).

Les directives *<Directory>* ne peuvent pas être imbriquées.

12.7.2 DirectoryMatch

Description Regroupement de directives pour un répertoire

Syntaxe <DirectoryMatch *regex*> ... </DirectoryMatch>

Contexte server config, virtual host

Module Core

Cette directive a le même rôle que la précédente mais son argument doit être une expression régulière.

Exemple

```
<DirectoryMatch "/var/www/[0-9]{3}">

</DirectoryMatch>
```

Cette notation permettrait de désigner les sous-répertoires de */var/www* dont le nom se compose de trois chiffres exactement.

La directive <*Directory*> accepte également les expressions régulières via le caractère ~.

```
<Directory ~ "/var/www/[0-9]{3}">

</Directory>
```

12.7.3 Files

Description Regroupement de directives pour un nom de fichier

Syntaxe <Files *nomfichier*> ... </Files>

Contexte server config, virtual host, directory, .htaccess

Module Core

Cette directive permet de contrôler l'accès à des fichiers. Elle est comparable à la directive <*Directory*>. Les sections <*Files*> sont traitées dans l'ordre où elles apparaissent dans le fichier de configuration. Le nom de fichier peut faire apparaître des caractères spéciaux tels que ceux du shell Unix ou des expressions régulières précédées du caractère ~.

Exemple

```
<Files *.html>

</Files>
```

Cette section désigne les fichiers d'extension *.html*.

12.7.4 FilesMatch

Description Regroupement de directives pour un nom de fichier

Syntaxe <FilesMatch *regex*> ... </FilesMatch>

Contexte server config, virtual host, directory, .htaccess

Module Core

Cette directive a le même rôle que la précédente mais son argument doit être une expression régulière.

Exemplse

```
<FilesMatch "\.(gif|jpe?g|png)$">

</FilesMatch>

<FilesMatch "^\.ht">

</FilesMatch>
```

Ces sections désignent respectivement des fichiers de type image (extensions *.gif*, *.jpg*, *.jpeg* ou *.png*) et les fichiers dont le nom commence par la chaîne de caractères *.ht* .

12.7.5 Location

Description Regroupement de directives pour une URL

Syntaxe <Location *URL-path|URL*> ... </Location>

Contexte server config, virtual host

Module Core

Cette directive a le même rôle que la directive <*Directory*> mais son argument désigne une URL (sans la partie *http://serveur*). Les sections <*Location*> sont traitées dans l'ordre où elles apparaissent dans le fichier de configuration, après les sections <*Directory*> et <*Files*>.

Exemple

```
# Allow server status reports generated by mod_status,
# with the URL of http://servername/server-status
# Change the ".example.com" to match your domain to enable.
#
<Location /server-status>
    SetHandler server-status
    Order deny,allow
    Deny from all
    Allow from .example.com
</Location>
```

Cet exemple fait apparaître la directive *SetHandler* qui est particulièrement utile quand elle est combinée à la directive <Location>.

12.7.6 SetHandler

Description Association d'un "handler" à des fichiers

Syntaxe SetHandler *handler-name*|None

Contexte server config, virtual host, directory, .htaccess

Module Core

Un **handler** désigne une action associée à un fichier autrement que par son type. Quelques *handlers* prédéfinis sont fournis avec Apache. Dans le paragraphe précédent, le handler *server-status* est associé à l'URL */server-status* pour déclencher l'affichage d'informations concernant l'état du serveur.

12.7.7 LocationMatch

Description Regroupement de directives pour une URL

Syntaxe <LocationMatch *regex*> ... </LocationMatch>

Contexte server config, virtual host

Module Core

Cette directive a le même rôle que la directive <*Location*> mais son argument doit être une expression régulière.

Exemple

```
<LocationMatch "(extra|special)/data">

</LocationMatch>
```

Cette notation permet de désigner les URLs qui comporteraient la chaîne */extra/data* ou */special/data*.

La directive <*Location*> accepte également les expressions régulières via le caractère ~.

12.7.8 Options

Description Gestion des options pour un répertoire

Syntaxe Options [+|-]*option* [[+|-]*option*] ...

Valeur par défaut Options ALL

Contexte server config, virtual host, directory, .htaccess

Module Core

Cette directive permet d'autoriser ou non certaines fonctionnalités à l'intérieur d'un répertoire. Les arguments disponibles sont :

- **All**

Toutes les options sont activées (sauf *Multiviews* qui doit être explicite). Il s'agit de la valeur par défaut.

- **None**

Aucune option n'est activée.

- **ExecCGI**

L'exécution des scripts CGI est autorisée.

- **FollowSymLinks**

Le serveur peut suivre les liens symboliques.

- **Includes**

On autorise l'utilisation des directives SSI (*Server Side Include*). Les directives SSI consistent en un jeu d'instructions permettant d'insérer dynamiquement du code HTML dans la page Web.

- **IncludesNOEXEC**

On autorise l'utilisation des directives SSI, sauf *#exec*.

- **Indexes**

Il sera possible de visualiser la liste des fichiers du répertoire (voir la directive *DirectoryIndex* un peu plus loin dans ce chapitre).

- **MultiViews**

La négociation de contenu est activée. Il s'agit de pouvoir proposer plusieurs versions d'une page donnée selon les préférences du client (la langue, la plupart du temps).

- **SymLinksIfOwnerMatch**

Le serveur peut suivre les liens symboliques uniquement si la cible est de même propriétaire que le lien lui-même.

Exemples

```
<Directory />
    Options FollowSymLinks
</Directory>

<Directory "/usr/local/apache2/htdocs">
    Options Indexes FollowSymLinks
</Directory>
```

```
<Directory /home/*/public_html>
    Options MultiViews Indexes SymLinksIfOwnerMatch IncludesNoExec
</Directory>

<Directory "/usr/local/apache2/cgi-bin">
    Options None
</Directory>
```

Les arguments de la directive Options peuvent être précédés du **signe +** ou du **signe -**. Dans ce cas, ils s'ajoutent ou se retranchent aux éventuels arguments déjà positionnés. Quand aucun de ces deux signes n'est présent devant les arguments, ceux-ci remplacent leurs éventuels prédécesseurs.

Exemples

```
<Directory "/var/www/rep">
    Options Indexes FollowSymLinks
</Directory>

<Directory "/var/www/rep/repbis">
    Options Includes Multiviews
</Directory>
```

Dans ce premier exemple, les options *Includes* et *Multiviews* remplacent les deux précédentes dans le cadre du sous-répertoire *repbis*.

```
<Directory "/var/www/rep">
    Options Indexes FollowSymLinks
</Directory>

<Directory "/var/www/rep/repbis">
    Options +Includes -Indexes
</Directory>
```

Dans ce deuxième exemple, l'option *Includes* est ajoutée, l'option *Indexes* est enlevée tandis que l'option *FollowSymlinks* est maintenue dans le cadre du sous-répertoire *repbis*.

Les options *FollowSymLinks* et *SymLinksIfOwnerMatch* ne sont autorisées que dans les strophes <Directory> ou dans les fichiers .htaccess.

La directive *Options* ne peut pas apparaître dans une strophe <Files>.

12.7.9 Contrôles d'accès

Allow

Description Contrôle d'accès aux ressources

Syntaxe Allow from all|*host*|env=*env-variable* ...

Contexte directory, .htaccess

Module mod_access

Cette directive permet de préciser quels sont les *hosts* clients qui peuvent accéder à une ressource donnée.

Il y a diverses façons de préciser les clients autorisés :

- **all** (tous les clients)

- une **adresse IP** (172.16.1.1)

- une **adresse de réseau** ou sous-réseau (172.16.1)

- une **adresse IP** et un **masque** de sous-réseau (172.16.1.1/255.255.255.0)

- la même association en notation **CIDR** (172.16.1.1/24)

- un **nom de domaine** DNS (mondomaine.com)

ou encore,

- une **variable d'environnement** (positionnée via la directive *SetEnvIf)*

Deny

Description	Contrôle d'accès aux ressources
Syntaxe	Deny from all|*host*|env=*env-variable* ...
Contexte	directory, .htaccess
Module	mod_access

Cette directive est identique à la précédente mais elle permet de préciser quels sont les *hosts* clients qui ne peuvent pas accéder à une ressource donnée.

Order

Description	Ordre d'évaluation des droits
Syntaxe	Order *ordering*
Valeur par défaut	Order deny,allow
Contexte	directory, .htaccess
Module	mod_access

Cette directive indique l'ordre d'évaluation entre les directives *Allow* et *Deny*. Les arguments possibles sont :

- **deny,allow**

Les accès sont autorisés par défaut. Les directives *Deny* sont examinées en premier. Tout client non mentionné dans une directive *Deny* ou bien apparaissant dans une directive *Allow* obtient le droit d'accès.

- **allow,deny**

Les accès sont interdits par défaut. Les directives *Allow* sont examinées en premier. Tout client non mentionné dans une directive *Allow* ou bien apparaissant dans une directive *Deny* se voit refuser le droit d'accès.

- **mutual-failure**

Synonyme déprécié de allow,deny.

Exemples

- Accès réservé aux seules machines du domaine *apache.org*

```
Order deny,allow
Deny from all
Allow from apache.org
```

- Accès réservé aux seules machines du domaine *apache.org* (sauf celles du sous-domaine *foo*)

```
Order allow,deny
Allow from apache.org
Deny from foo.apache.org
```

- Accès complètement interdit

```
Order allow,deny
```

- Accès réservé aux navigateurs *Knock* en versions 2.0

```
SetEnvIf  User-Agent ^Knock/2\.0  let_me_in
<Directory /docroot>
   Order Deny,Allow
   Deny from all
   Allow from env=let_me_in
</Directory>
```

12.7.10 Fichiers décentralisés .htaccess

Il est possible d'obtenir une gestion décentralisée des répertoires en y plaçant des fichiers dont le nom par défaut est **.htaccess**.

La directive **AccessFileName** permettrait de modifier le nom de ces fichiers.

Un certain nombre de directives, notamment des directives d'authentification, peuvent apparaître dans ces fichiers qui sont lus à chaque requête. Toute modification prend donc effet immédiatement sans avoir à redémarrer le serveur Apache. Ceci s'inscrit tout à fait dans la démarche de délégation de pouvoirs administratifs.

Comme il se doit, l'administrateur Apache peut contrôler l'usage de ces fichiers décentralisés à l'aide d'une directive **AllowOverride** (autorisée uniquement dans une

strophe <Directory>). Cette directive permet de préciser des catégories de directives autorisées dans les fichiers .htaccess.

Il est à noter que, à partir du moment où la directive *AllowOverride* l'autorise, chaque requête déclenche la recherche d'un fichier *.htaccess* dans tous les répertoires faisant partie du chemin de l'URL. Ceci peut avoir un impact négatif non négligeable sur les performances. Il faut donc limiter l'usage des fichiers décentralisés à des parties bien ciblées de l'arborescence Web, telles que d'éventuels sites personnels.

AllowOverride

Description	Contrôle de l'utilisation des fichiers .htaccess
Syntaxe	AllowOverride All|None|*directive-type* ...
Contexte	directory
Module	Core

Les arguments disponibles sont :

- **All**

Toutes les directives valides dans ce contexte sont autorisées.

- **None**

Aucune directive n'est autorisée. Les fichiers *.htaccess* ne sont pas opérationnels.

- **AuthConfig**

Il est possible d'utiliser des directives liées à une procédure d'authentification.

- **FileInfo**

Il est possible d'utiliser des directives liées aux types de documents.

- **Indexes**

Il est possible d'utiliser des directives liées à la visualisation des fichiers d'un répertoire.

- **Limit**

Il est possible d'utiliser des directives liées au contrôle d'accès.

- **Options**

Il est possible d'utiliser la directive *Options*.

12.7.11 Ordre d'évaluation

Les directives placées dans une strophe <Directory> concernent des répertoires du systèmes de fichiers ainsi que tous leurs sous-répertoires. D'éventuels fichiers décentralisés .htaccess permettent d'obtenir les mêmes fonctionnalités.

Les directives placées dans une strophe <Files> concernent des fichiers situés dans n'importe quel répertoire du système de fichiers à moins que la strophe <Files> ne soit placée elle-même dans une strophe <Directory> ou dans un fichier .htaccess.

Les variantes <DirectoryMatch> ou <FilesMatch> intègrent des expressions régulières pour désigner les noms de répertoires ou de fichiers.

Les directives placées dans une strophe <Location> concernent une partie de l'arborescence Web, c'est-à-dire une URL, sans la partie *http://serveur*.

L'ordre d'évaluation est le suivant :

1) Simultanément, les strophes <Directory> et les fichiers .htaccess qui peuvent donc prendre le pas sur les strophes <Directory>,
2) Les strophes <DirectoryMatch> (et <Directory ~ …>),
3) Les strophes <Files> et <FilesMatch>,
4) Les strophes <Location> et <LocationMatch>.

Après inclusion des fichiers désignés par les directives *Include*, les strophes sont examinées dans leur ordre d'apparition pour chacune des 4 catégories ci-dessus.

Les strophes <Directory> font exception à cet examen séquentiel. Elles sont examinées selon la règle "du plus court chemin d'abord".

Par exemple, la strophe <Directory /var/web/dir> sera examinée avant la strophe <Directory /var/web/dir/subdir>. Si plusieurs strophes <Directory> s'appliquaient au même répertoire, elles seraient traitées dans un ordre séquentiel.

Enfin, les strophes associées aux hôtes virtuels (à l'intérieur de strophes <VirtualHost>) seront examinées dans une dernière phase, ce qui permet que les configurations d'hôtes virtuels prennent le pas sur la configuration du serveur principal.

12.7.12 Visualisation des fichiers d'un répertoire

DirectoryIndex

Description	`Liste des fichiers par défaut d'un répertoire`
Syntaxe	`DirectoryIndex` *`local-url`* `[`*`local-url`*`] ...`
Valeur par défaut	`DirectoryIndex index.html`
Contexte	`server config, virtual host, directory, .htaccess`
Module	`mod_dir`

Quand le client spécifie une URL se terminant par le caractère /, le serveur recherche les fichiers précisés dans la directive *DirectoryIndex*. Si aucun de ces fichiers n'existe dans le répertoire concerné et si l'option *Indexes* est activée, le serveur générera la liste des fichiers du répertoire.

Exemple

```
DirectoryIndex  index.html  index.htm  index.php  /cgi-bin/index.pl
```

Dans cet exemple, le programme CGI */cgi-bin/index.pl* sera exécuté si aucun des trois premiers fichiers n'existe dans le répertoire concerné.

IndexOptions

Description Configuration de la liste d'un répertoire

Syntaxe IndexOptions [+|-]*option* [[+|-]*option*] ...

Contexte server config, virtual host, directory, .htaccess

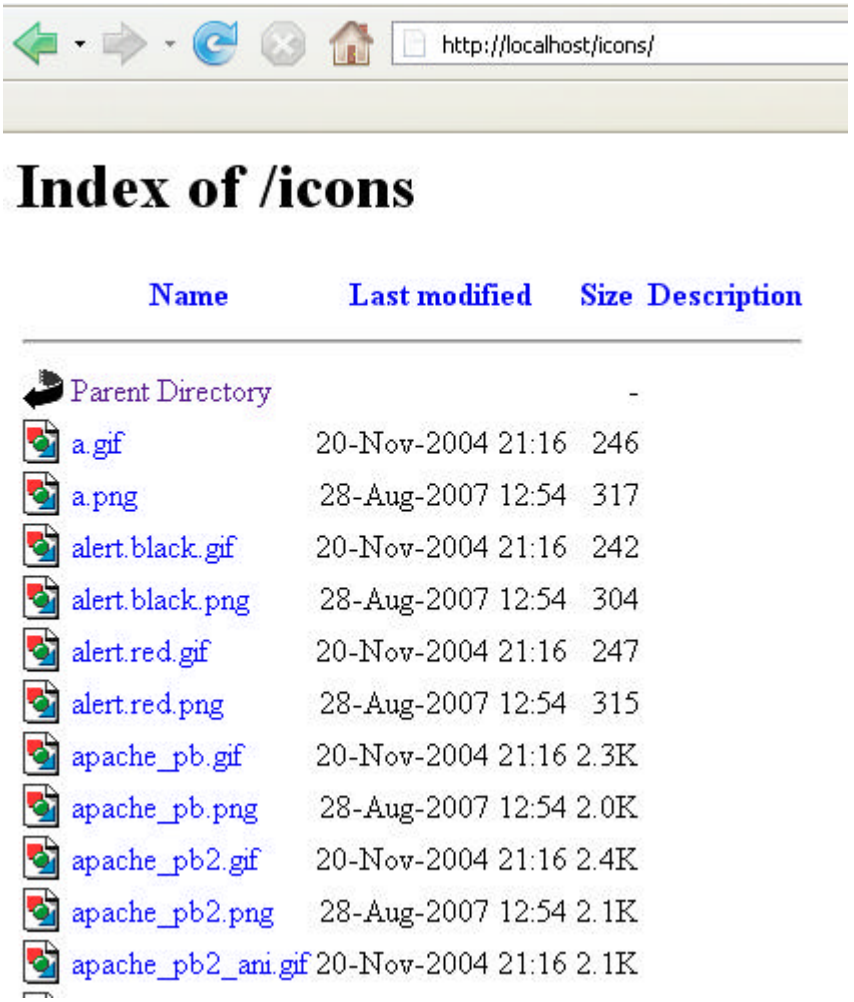
Module mod_autoindex

Cette directive permet de paramétrer l'aspect de la liste des fichiers d'un répertoire quand l'option *Indexes* le permet. Les possibilités sont nombreuses et assez élaborées. On se contentera de l'aspect par défaut ou bien il faudra se reporter à la documentation de référence de la directive.

Exemple

```
IndexOptions FancyIndexing HTMLTable VersionSort

Alias /icons/ "/usr/local/apache2/icons/"

<Directory "/usr/local/apache2/icons">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
```

Les directives de cet exemple nous rappelle l'existence de l'alias */icons/* qui va nous permettre d'accéder au répertoire */usr/local/apache2/icons/*. Ce répertoire ne contient aucun des fichiers listés dans la directive *DirectoryIndex*. La directive *Options* autorise l'argument *Indexes*. Par conséquent, si nous demandons l'URL *http://monserveur.mondomaine.com/icons/*, tous ces paramétrages cumulés vont nous donner accès à la liste des fichiers du répertoire cible. L'aspect de la liste est déterminé par les paramétrages de la directive *IndexOptions*.



Liste des fichiers du répertoire pointé par l'URL /icons/

D'autres directives se rajoutent à *IndexOptions* en ce qui concerne le paramétrage de la visualisation des fichiers d'un répertoire. Ces directives appartiennent au module **mod_autoindex**.

Extraits du fichier de configuration concernant ces directives

```
#
# AddIcon* directives tell the server which icon to show for different
# files or filename extensions.  These are only displayed for
# FancyIndexed directories.
#
AddIconByEncoding (CMP,/icons/compressed.gif) x-compress x-gzip

AddIconByType (TXT,/icons/text.gif) text/*
AddIconByType (IMG,/icons/image2.gif) image/*
AddIconByType (SND,/icons/sound2.gif) audio/*
AddIconByType (VID,/icons/movie.gif) video/*

AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/binhex.gif .hqx
```

```
AddIcon /icons/tar.gif .tar
AddIcon /icons/world2.gif .wrl .wrl.gz .vrml .vrm .iv
AddIcon /icons/compressed.gif .Z .z .tgz .gz .zip
AddIcon /icons/a.gif .ps .ai .eps
AddIcon /icons/layout.gif .html .shtml .htm .pdf
AddIcon /icons/text.gif .txt
AddIcon /icons/c.gif .c
AddIcon /icons/p.gif .pl .py
AddIcon /icons/f.gif .for
AddIcon /icons/dvi.gif .dvi
AddIcon /icons/uuencoded.gif .uu
AddIcon /icons/script.gif .conf .sh .shar .csh .ksh .tcl
AddIcon /icons/tex.gif .tex
AddIcon /icons/bomb.gif core
AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^

#
# DefaultIcon is which icon to show for files which do not have an icon
# explicitly set.
#
DefaultIcon /icons/unknown.gif

# AddDescription allows you to place a short description after a file in
# server-generated indexes.  These are only displayed for FancyIndexed
# directories.
# Format: AddDescription "description" filename
#
#AddDescription "GZIP compressed document" .gz
#AddDescription "tar archive" .tar
#AddDescription "GZIP compressed tar archive" .tgz
#
#
# ReadmeName is the name of the README file the server will look for by
# default, and append to directory listings.
#
# HeaderName is the name of a file which should be prepended to
# directory indexes.
ReadmeName README.html
HeaderName HEADER.html

#
# IndexIgnore is a set of filenames which directory indexing should ignore
# and not include in the listing.  Shell-style wildcarding is permitted.
#
IndexIgnore .??* *~ *# HEADER* README* RCS CVS *,v *,t
```

Il est possible notamment par les directives **HeaderName** et **ReadmeName** d'afficher, respectivement en début et fin de liste, le contenu de deux fichiers.

La directive **IndexIgnore** précise les noms de fichiers qui ne seront pas intégrés dans la liste.

12.8 Pages personnelles

La directive *UserDir* va permettre la déclaration d'arborescences personnelles pour les utilisateurs possédant un compte sur le serveur.

UserDir

Description	Emplacement des arborescences des utilisateurs
Syntaxe	UserDir *directory-filename*
Valeur par défaut	userDir public_html
Contexte	server config, virtual host
Module	mod_userdir

L'argument de la directive peut prendre beaucoup de valeurs précisées dans la documentation de référence et notamment la valeur *Disabled* si les arborescences Web des utilisateurs ne doivent pas être autorisées.

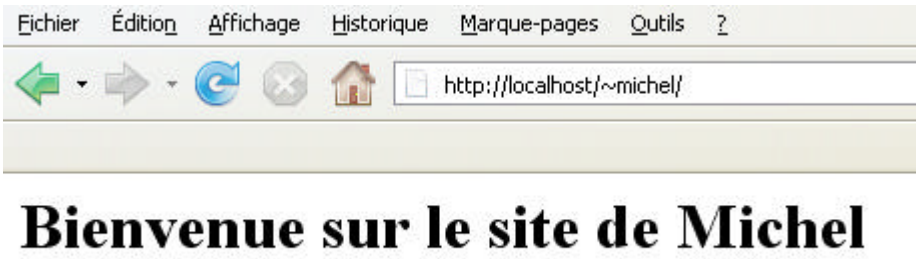
Lorsque la directive prend la valeur *public_html*, cela signifie que l'URL *http://serveur/~user/* donnera l'accès à l'arborescence Web située dans le répertoire de connexion de l'utilisateur, sans doute */home/user/public_html*.

Exemple de mise en oeuvre

```
#
# UserDir: The name of the directory that is appended onto
# a user's home directory if a ~user request is received.
#
UserDir public_html
#
# Control access to UserDir directories.  The following is an
#
<Directory /home/*/public_html>
   AllowOverride FileInfo AuthConfig Limit Indexes
   Options Indexes SymLinksIfOwnerMatch
</Directory>
```

Les directives ci-dessus vont permettre aux utilisateurs de proposer une arborescence Web à partir du sous-répertoire *public_html* de leur répertoire de connexion Linux. Dans notre exemple, l'utilisateur *michel* a préparé une page d'accueil *index.html* directement accessible depuis l'URL *http://serveur/~michel/*.

```
# cat /home/michel/public_html/index.html
<HTML>
<HEAD>
        <TITLE>Page d'accueil de Michel</TITLE>
</HEAD>
<BODY>
        <H1>Bienvenue sur le site de Michel</H1>
</BODY>
</HTML>
#
```



Page d'accueil obtenue via l'URL http://serveur/~michel/

12.9 Authentification des utilisateurs

Sans programmation, Apache permet la mise en œuvre d'un mécanisme d'authentification par nom de connexion et mot de passe sur tout ou partie d'un site. L'authentification est définie dans le protocole HTTP et la quasi-totalité des clients navigateurs l'implémentent.

Dans les versions Apache 2.2, les modules liés à l'authentification ont été assez largement réorganisés par rapport aux versions Apache 2.0. Les directives de l'ancien module de base *mod_auth* sont cependant conservées et elles permettent une authentification dite *basique* qui utilisera des fichiers de texte pour stocker les noms de connexion et les mots de passe associés.

On pourra sans doute se contenter de l'authentification basique. Si un nombre important d'utilisateurs doivent s'authentifier de manière très sécurisée, il est probable qu'il faudra faire l'effort de la programmation (par exemple en PHP, en stockant les comptes dans une base MySQL.).

12.9.1 Création des comptes

La commande **htpasswd**, fournie dans la distribution Apache, permet de gérer des fichiers contenant les noms des utilisateurs et leur mot de passe crypté. Un premier appel avec l'option *-c* permet de créer ou de réinitialiser le fichier. Les appels ultérieurs, sans cette option, permettront de créer les comptes ou de modifier les mots de passe.

Il est recommandé de faire commencer le nom du fichier par la chaîne de caractères **.ht** (par exemple, .htpasswd) car ces noms de fichiers sont, par défaut, protégés de tout accès par les clients Web, grâce à une strophe *<Filesmatch>* dans les fichiers de configuration d'Apache.

```
# The following lines prevent .htaccess and .htpasswd files from
# being viewed by Web clients.
#
<FilesMatch "^\.ht">
    Order allow,deny
    Deny from all
    Satisfy All
</FilesMatch>
```

Nous donnons ici un exemple d'utilisation de la commande *htpasswd*, depuis le compte ordinaire de l'utilisateur michel. Le premier appel crée le fichier .htpasswd (option *-c*) et définit un premier utilisateur. Le deuxième appel définit un autre compte Apache et le troisième appel permet de modifier le mot de passe d'un compte déjà existant. Nous constatons que le fichier est bien au format texte et qu'il contient les mots de passe sous forme cryptée. Nous avons utilisé l'option *-m* pour obtenir un cryptage MD5.

```
/home/michel/public_html$ id
uid=1000(michel) gid=1000(michel)
/home/michel/public_html$ htpasswd -c -m .htpasswd visiteur
```

```
New password:
Re-type new password:
Adding password for user visiteur
/home/michel/public_html$ htpasswd -m .htpasswd michel
New password:
Re-type new password:
Adding password for user michel
/home/michel/public_html$ cat .htpasswd
visiteur:$apr1$An0/X/..$8HE8Nehrfg9XXLMWMwPtJ1
michel:$apr1$0hJdk...$vfaFACXXS/PdoIPHPU3Tr0
/home/michel/public_html$ htpasswd -m .htpasswd michel
New password:
Re-type new password:
Updating password for user michel
/home/michel/public_html$ cat .htpasswd
visiteur:$apr1$An0/X/..$8HE8Nehrfg9XXLMWMwPtJ1
michel:$apr1$HgXah/..$vNZ/RRiluuf5KrLx4fORm0
/home/michel/public_html$
#
```

12.9.2 Directives de configuration

Les directives de configuration peuvent être placées dans une section <*Directory*> ou bien dans un fichier *.htaccess*. Dans ce deuxième cas, la directive *AllowOverride* du fichier général doit autoriser le paramètre *AuthConfig*.

Un certain nombre de directives doivent être positionnées simultanément.

AuthName

Description	Identification d'une ressource protégée (realm)
Syntaxe	AuthName *auth-domain*
Contexte	directory, .htaccess
Module	Core

Afin de pouvoir identifier plusieurs zones protégées par mot de passe sur le même serveur, le protocole HTTP prévoit le nommage de chacune de ces zones par un nom baptisé *realm.* La directive permet de spécifier le *realm* envoyé au client pour identifier la zone protégée. Ceci permet au navigateur de conserver le mot de passe en mémoire pour ne pas le redemander à la prochaine consultation d'un document de ce même *realm*.

AuthType

Description	Type d'authentification
Syntaxe	AuthType Basic|Digest
Contexte	directory, .htaccess

Module Core

L'authentification de type *Basic* correspond à la commande *htpasswd* initiée précédemment.

Require

Description Désignation des utilisateurs autorisés

Syntaxe Require *entity-name* [*entity-name*] ...

Contexte directory, .htaccess

Module Core

Cette directive recense les utilisateurs ou les groupes autorisés à accéder à la ressource protégée. Le mot clé **valid-user** désigne n'importe quel utilisateur connu dans le fichier des comptes créé via la commande *htpasswd*.

Satisfy

Description Interaction entre le contrôle d'accès (allow,deny)et l'authentification

Syntaxe Satisfy Any|All

Valeur par défaut Satisfy All

Contexte directory, .htaccess

Module Core

Si une ressource est gérée à la fois par un contrôle d'accès sur les adresses réseau (directives *Allow* et *Deny*) et par un mécanisme d'authentification, les deux conditions devront être satisfaites quand cette directive prend la valeur *All* (valeur par défaut). La valeur *Any* autorisera le client à ne satisfaire qu'une seule des deux conditions.

Exemple

```
Require valid-user
Allow from 172.16
Satisfy Any
```

Dans cet exemple, les clients qui ne viennent pas du réseau 172.16 doivent s'authentifier. Les clients qui viennent de ce réseau interne en sont dispensés.

AuthUserFile

Description Nom du fichier des comptes

Syntaxe AuthUserFile *file-path*

Contexte directory, .htaccess

Module mod_authn_file

Cette directive mentionne le nom du fichier des comptes, créé par la commande *htpasswd* (nom absolu ou relatif à *ServerRoot*). Le fichier peut être placé en dehors de l'arborescence Web. Dans le cas contraire, nous prendrons soin de faire commencer son nom par la chaîne de caractères *.ht* pour qu'il ne soit pas accessible par une requête HTTP.

AuthGroupFile

Description Nom du fichier des groupes

Syntaxe AuthGroupFile *file-path*

Contexte directory, .htaccess

Module mod_authz_groupfile

Cette directive mentionne le nom de l'éventuel fichier des groupes (nom absolu ou relatif à *ServerRoot*). La directive *Require*, (via la syntaxe *Require group groupname*) peut exploiter cette notion de groupe d'utilisateurs. Il n'y a pas de commande dédiée à la création de ce fichier qui devra donc être généré avec un éditeur de texte. Chaque ligne comportera un nom de groupe suivi du caractère ":" et d'une liste d'utilisateurs séparés par des espaces.

Exemple

```
groupe1: michel marcel roger
groupe2: toto zorro
```

Le fichier peut être placé en dehors de l'arborescence Web. Dans le cas contraire, nous prendrons soin de faire commencer son nom par la chaîne de caractères *.ht* pour qu'il ne soit pas accessible par une requête HTTP.

12.9.3 Exemple de mise en oeuvre

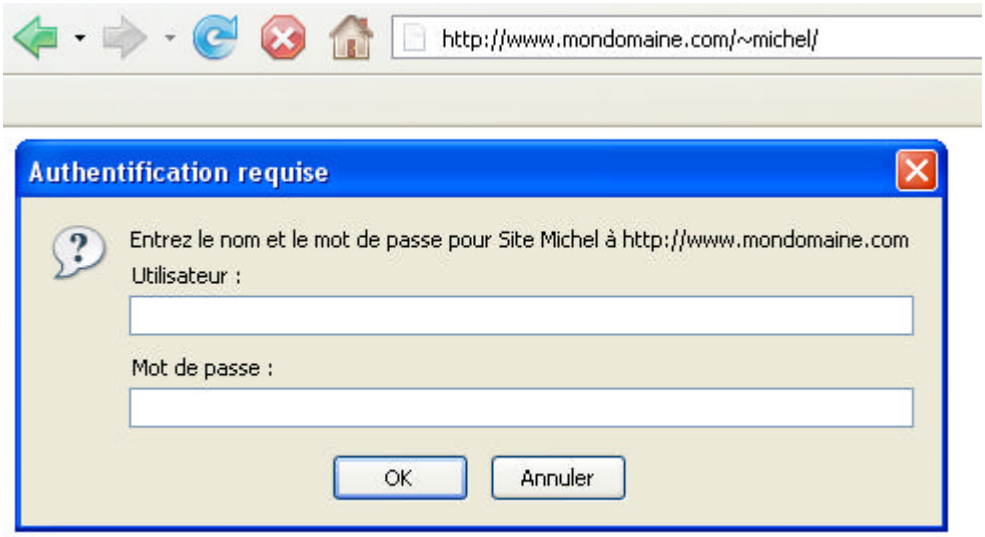
Nous allons préparer les directives nécessaires pour que le site personnel de l'utilisateur *michel* (voir le paragraphe précédent) nécessite une authentification. Bien que ce ne soit nullement obligatoire, nous allons utiliser le fichier décentralisé *.htaccess* et l'utilisateur *michel* aura lui-même géré au préalable son fichier des comptes, via la commande htpasswd.

L'utilisateur *michel* crée son fichier administratif *.htaccess* dans le répertoire racine de son site personnel. Ce fichier doit comporter les quatre directives nécessaires au bon fonctionnement de la procédure d'authentification, dite basique

```
/home/michel/public_html$ cat .htaccess
AuthName      "Site Michel"
AuthType      Basic
AuthUserFile  /home/michel/public_html/.htpasswd
Require       valid-user
/home/michel/public_html$
```

Comme il s'agit d'une administration décentralisée autorisée par le fichier général *httpd.conf*, ce fichier *.htaccess* est immédiatement opérationnel.

Une tentative de connexion au site provoque l'apparition d'un boîte de connexion dont le nom est bien celui indiqué dans la directive *AuthName*.



Authentification nécessaire pour accéder au site personnel de michel

12.10 Hôtes virtuels

Le serveur Apache est capable de gérer simultanément plusieurs arborescences Web grâce à la notion d'**hôtes virtuels** (*virtual hosts*).

Plusieurs méthodes sont possibles :

- Hôtes virtuels basés sur l'**adresse IP**

Le serveur est doté de plusieurs interfaces réseau ou bien l'administrateur a pu associer plusieurs adresses IP à une même carte réseau (*IP aliasing*). Les noyaux Linux permettent de mettre en œuvre facilement cette fonctionnalité.

- Hôtes virtuels basés sur le **numéro de port**

Plusieurs ports d'une même adresse IP peuvent être associés à des arborescences Web différentes. Il faudra, bien entendu, préciser ce port dans les URLs (exemple : *http://www.mondomaine.com:8080*).

- Hôtes virtuels basés sur le **nom**

Plusieurs noms DNS peuvent être associés à une seule adresse IP et correspondre aussi à des arborescences différentes.

La troisième méthode (basée sur le nom) n'utilise qu'une seule adresse IP et nécessite simplement une bonne configuration DNS. Il s'agit aujourd'hui de la méthode recommandée, sauf besoins particuliers de compatibilité avec certains logiciels clients.

12.10.1 Hôtes virtuels basés sur l'adresse IP

Nous allons, tout d'abord, associer une deuxième adresse à notre unique carte réseau. La commande *ifconfig* nous permet de créer cet *alias IP*.

```
# ifconfig  eth0:0  192.168.0.100
# ifconfig
eth0 Lien encap:Ethernet  HWaddr 00:0F:B0:D7:78:00
     inet adr:192.168.0.2  Bcast:192.168.0.255  Masque:255.255.255.0
     adr inet6: fe80::20f:b0ff:fed7:7800/64 Scope:Lien
     UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
     RX packets:261 errors:0 dropped:0 overruns:0 frame:0
     TX packets:158 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 lg file transmission:1000
     RX bytes:25594 (24.9 KiB)  TX bytes:20435 (19.9 KiB)
     Interruption:58 Adresse de base:0x6000

eth0:0 Lien encap:Ethernet  HWaddr 00:0F:B0:D7:78:00
   inet adr:192.168.0.100  Bcast:192.168.0.255  Masque:255.255.255.0
   UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
   Interruption:58 Adresse de base:0x6000
#
```

Pour faire notre essai de configuration, nous simulerons une vraie configuration DNS en renseignant notre fichier local */etc/hosts* avec deux noms de sites *www.site1.com* et *www.site2.com*, associés respectivement aux deux adresses maintenant disponibles.

```
# cat /etc/hosts
127.0.0.1       localhost
192.168.0.2     www.site1.com
192.168.0.100   www.site2.com
#
```

Le fichier de configuration principal prévoit l'inclusion d'un fichier décrivant les configurations d'hôtes virtuels :

```
# Virtual hosts
Include conf/extra/httpd-vhosts.conf
```

Nous devons prévoir, dans ce fichier, deux strophes <*VirtualHost*>.

Pour un test minimum, il nous suffit de renseigner les deux directives de base *DocumentRoot* et *ServerName*. Chaque hôte virtuel est donc tout naturellement caractérisé par son nom et son arborescence. Il faut prendre soin également de mettre en commentaire la directive *NameVirtualHost* qui ne doit être active que pour mettre en œuvre des hôtes virtuels basés sur le nom.

```
#NameVirtualHost *:80
<VirtualHost 192.168.0.2:80>
    DocumentRoot /usr/local/apache2/htdocs/site1.com
    ServerName www.site1.com
</VirtualHost>

<VirtualHost 192.168.0.100:80>
    DocumentRoot /usr/local/apache2/htdocs/site2.com
    ServerName www.site2.com
</VirtualHost>
```

Nous préparons ensuite deux pages d'accueil très simples pour nos deux sites et les plaçons dans les répertoires adéquats.

```
/usr/local/apache2/htdocs# cat site1.com/index.html
<HTML>
<HEAD>
        <TITLE>Page d'accueil de site1.com</TITLE>
</HEAD>
<BODY>
        <H1>Bienvenue sur SITE1.COM</H1>
</BODY>
</HTML>
/usr/local/apache2/htdocs# cat site2.com/index.html
<HTML>
<HEAD>
        <TITLE>Page d'accueil de site2.com</TITLE>
</HEAD>
<BODY>
```

```
        <H1>Bienvenue sur SITE2.COM</H1>
</BODY>
</HTML>
/usr/local/apache2/htdocs#
```

Nous redémarrons le serveur Apache.

```
# apachectl restart
#
```

Nous testons successivement les deux URLs *www.site1.com* et *www.site2.com* qui nous présentent bien leurs deux pages d'accueil distinctes.

Écran d'accueil de l'hôte virtuel www.site1.com

Écran d'accueil de l'hôte virtuel www.site2.com

12.10.2 Hôtes virtuels basés sur le numéro de port

Pour une adresse et un nom DNS donnés, il est possible de solliciter un autre port que le port 80 par défaut. Il suffit de constituer des strophes <*VirtualHost*> en précisant le port à ajouter dans les URLs.

Il ne faut pas oublier d'ajouter une directive *Listen* associé au nouveau port d'écoute.

```
<VirtualHost 192.168.0.2:80>
    DocumentRoot /usr/local/apache2/htdocs/site1.com
    ServerName www.site1.com
</VirtualHost>

<VirtualHost 192.168.0.2:8080>
```

```
    DocumentRoot /usr/local/apache2/htdocs/site1.com/8080
    ServerName www.site1.com
</VirtualHost>
Listen 8080
```

Dans cet exemple, l'URL *http://www.site1.com* nous emmènera dans le répertoire *site1.com* comme précédemment, alors que l'URL *http://www.site1.com:8080* nous conduira dans le répertoire *site1.com/8080*.

Écran d'accueil de l'hôte virtuel www.site1.com:8080

12.10.3 Hôtes virtuels basés sur le nom

Nous allons, pour cette troisième méthode, associer plusieurs noms DNS à une seule adresse IP. Pour faire notre essai de configuration, nous simulerons à nouveau une vraie configuration DNS en renseignant notre fichier local */etc/hosts*

```
# cat /etc/hosts
192.168.0.2     www.site1.com
192.168.0.2     www.site2.com
#
```

Nous devons maintenant activer la directive *NameVirtualHost*.

NameVirtualHost

Description	Adresse IP pour hôtes virtuels basés sur le nom
Syntaxe	NameVirtualHost *addr*[:*port*]
Contexte	server config
Module	Core

Dans cette directive, l'adresse IP correspond à celle sur laquelle le serveur acceptera uniquement les requêtes adressées aux hôtes virtuels.

L'argument des strophes *<VirtualHost>* doit être exactement le même que celui de la directive *NameVirtualHost*.

Notre configuration de test va donc être la suivante :

```
NameVirtualHost 192.168.0.2:80
#
<VirtualHost 192.168.0.2:80>
    DocumentRoot /usr/local/apache2/htdocs/site1.com
    ServerName www.site1.com
    ErrorLog logs/www.site1.com-error_log
    CustomLog logs/www.site1.com-access_log common
    ServerAdmin www-admin@site1.com
</VirtualHost>
<VirtualHost 192.168.0.2:80>
    DocumentRoot /usr/local/apache2/htdocs/site2.com
    ServerName www.site2.com
    ErrorLog logs/www.site2.com-error_log
    CustomLog logs/www.site2.com-access_log common
    ServerAdmin www-admin@site2.com
</VirtualHost>
```

Nous testons à nouveau successivement les deux URLs *www.site1.com* et *www.site2.com* qui nous présentent bien leurs deux pages d'accueil distinctes.

Écran d'accueil de l'hôte virtuel www.site1.com

Écran d'accueil de l'hôte virtuel www.site2.com

Nous avons complété notre configuration de base avec les directives *ServerAdmin*, *ErrorLog* et *CustomLog*.

Les deux dernières concernent les fichiers de trace (paragraphe à suivre) qu'il est bon de distinguer pour chaque hôte virtuel. Après le test de bon fonctionnement, nous pourrons constater l'existence de fichiers de trace bien distincts pour chaque hôte virtuel.

```
/usr/local/apache2/logs# ls -l
total 56
```

```
-rw-r--r-- 1 root    root 24211 2007-10-20 10:41 access_log
srwx------ 1 apache2 root     0 2007-10-20 10:48 cgisock.4129
-rw-r--r-- 1 root    root  9584 2007-10-20 10:48 error_log
-rw-r--r-- 1 root    root     5 2007-10-20 10:48 httpd.pid
-rw-r--r-- 1 root    root   287 2007-10-20 10:48 www.site1.com-access_log
-rw-r--r-- 1 root    root   239 2007-10-20 10:48 www.site1.com-error_log
-rw-r--r-- 1 root    root   212 2007-10-20 10:44 www.site2.com-access_log
-rw-r--r-- 1 root    root   118 2007-10-20 10:44 www.site2.com-error_log
/usr/local/apache2/logs##
```

En mentionnant l'adresse explicite de notre carte réseau dans la directive *NameVirtualHost*, nous conservons l'URL *http://localhost* opérationnelle. Elle reste associée au répertoire racine mentionné dans la directive *DocumentRoot* du fichier de configuration principal. En effet, l'adresse 127.0.0.1 n'est alors pas prise en compte pour les hôtes virtuels.

Par contre l'URL qui mentionne le serveur associé à la directive *ServerName* du fichier principal nous emmènera maintenant vers le premier hôte virtuel de la liste. Ceci est le comportement normal des configurations d'hôtes virtuels basés sur le nom quand un **ServerName** associé une adresse IP de la directive *NameVirtualHost* ne correspond pas à une strophe <*VirtualHost*> explicite.

Si nous conservions la notation *NameVirtualHost *:80,* proposée par défaut, qui désigne toutes les interfaces, l'URL *http://localhost* serait elle aussi associée au premier hôte virtuel de la liste puisque, cette fois-ci, l'adresse 127.0.0.1 serait prise en compte.

Une autre configuration pourrait donc être la suivante :

```
NameVirtualHost *:80
#
<VirtualHost *:80>
    DocumentRoot /usr/local/apache2/htdocs/site1.com
    ServerName www.site1.com
    ErrorLog logs/www.site1.com-error_log
    CustomLog logs/www.site1.com-access_log common
    ServerAdmin www-admin@site1.com
</VirtualHost>
<VirtualHost *:80>
    DocumentRoot /usr/local/apache2/htdocs/site2.com
    ServerName www.site2.com
    ErrorLog logs/www.site2.com-error_log
    CustomLog logs/www.site2.com-access_log common
    ServerAdmin www-admin@site2.com
</VirtualHost>
```

12.11 Fichiers de logs, gestion des erreurs

12.11.1 Archivage des accès au serveur

LogFormat

Description Description d'un format de fichier de "log"

Syntaxe LogFormat *format|nickname* [*nickname*]

Contexte server config, virtual host

Module mod_log_config

Le premier argument de la directive est une chaîne de caractères, comportant des variables et décrivant le format d'une ligne dans le fichier de trace des accès. La documentation officielle décrit les nombreuses possibilités. Le second argument est un nom de format qui sera utilisé par la directive *CustomLog*.

Exemples

```
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\""
combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common

<IfModule logio_module>
# You need to enable mod_logio.c to use %I and %O
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %I %O"
combinedio
</IfModule>
```

Ces formats sont ceux fournis d'origine dans le fichier de configuration et il est, bien entendu, possible d'en concevoir d'autres.

Le format baptisé *combined* est assez complet. Il mémorise, entre autres, l'adresse du client, la date d'accès, la requête, le code retour, le nombre d'octets transférés, le type de navigateur…

CustomLog

Description Choix du fichier et du format de "log"

Syntaxe CustomLog *file|pipe format|nickname*

Contexte server config, virtual host

Module mod_log_config

Cette directive permet de choisir la destination des traces d'accès ainsi que le format des informations.

La destination peut être un fichier ou bien une commande alimentée via un *pipeline*. La commande **rotatelogs**, fournie avec Apache, est intéressante pour archiver

automatiquement, d'après une taille ou une périodicité, les versions successives des fichiers de trace. La page de manuel de la commande donne le détail des syntaxes disponibles.

Exemples

```
CustomLog  logs/access_log  combined
```

Le fichier de trace *access_log* sera situé dans le sous-répertoire *logs* de *ServerRoot* (chemin relatif) et il utilisera le format *combined* défini par une directive *LogFormat*.

```
CustomLog "|bin/rotatelogs /var/logs/logfile 86400" combined
```

Les fichiers de trace */var/logs/logfile.nnnn* utiliseront le format *combined.* Un nouveau fichier sera créé chaque jour (86400 secondes).

TransferLog

Description	Choix du fichier de "log"
Syntaxe	TransferLog *file*|*pipe*
Contexte	server config, virtual host
Module	mod_log_config

Cette directive remplit le même rôle que la directive *CustomLog* mais elle ne permet pas de spécifier explicitement le format des informations. Ce format sera celui de la dernière directive *LogFormat* ne comportant pas de nom de format. En l'absence de ce type de directive, le format *common* sera utilisé.

12.11.2 Exploitation des fichiers de trace

Dans le cadre de la mise en oeuvre de statistiques sur les accès au serveur, la commande **logresolve**, fournie avec Apache, permet d'effectuer des résolutions différées de noms DNS pour les adresses IP mémorisées dans les fichiers de logs.

Il existe un certain nombre d'outils évolués de statistiques sachant exploiter les formats des fichiers de trace Apache.

Le logiciel Open Source **AWStats** (awstats.sourceforge.net) fait partie des plus performants.

12.11.3 Archivage des erreurs

ErrorLog

Description Choix du fichier des erreurs

Syntaxe ErrorLog *file-path*|syslog[:*facility*]

Contexte server config, virtual host

Module mod_log_config

Cette directive permet de choisir la destination des messages d'erreur du serveur.

Comme pour la directive *CustomLog*, la destination peut être un fichier ou bien une commande alimentée via un *pipeline*.

Il est également possible d'indiquer la valeur *syslog* afin d'utiliser le service système **syslogd**.

Exemple

```
ErrorLog logs/error_log
```

LogLevel

Description Choix du niveau de sévérité des messages d'erreur

Syntaxe LogLevel *level*

Valeur par défaut LogLevel warn

Contexte server config, virtual host

Module mod_log_config

Cette directive permet de choisir le niveau de sévérité des messages d'erreur.

Par ordre décroissant, les niveaux disponibles sont *emerg*, *alert*, *crit*, *error*, *warn*, *notice*, *info* et *debug*.

Tous les messages de niveau supérieur ou égal à celui demandé seront archivés. Cependant, les messages de niveau *notice* sont systématiquement mémorisés quand la destination est un fichier ordinaire.

Exemple de message

```
[Sat Oct 20 13:12:26 2007] [error] [client 192.168.0.3] File does not
exist: /usr/local/apache2/htdocs/truc.html
```

Ce message nous permet de connaître la date de l'erreur, son niveau de sévérité (*error*), l'adresse du client concerné ainsi que la cause du problème (fichier introuvable).

ErrorDocument

Description Paramétrage des pages retournées en cas d'erreur

Syntaxe ErrorDocument *error-code document*

Contexte server config, virtual host, directory, .htaccess

Module Core

Cette directive permet de paramétrer l'aspect des documents retournés au navigateur en cas d'erreur.

Elle associe à un code d'erreur HTTP, soit un texte brut, soit une page personnalisée, obtenue éventuellement par un scrip CGI.

Le fichier de configuration comporte des commentaires qui nous donnent des idées :

```
# Customizable error responses come in three flavors:
# 1) plain text 2) local redirects 3) external redirects
# Some examples:
#ErrorDocument  500  "The server made a boo boo."
#ErrorDocument  404  /missing.html
#ErrorDocument  404  "/cgi-bin/missing_handler.pl"
#ErrorDocument  402  http://www.example.com/subscription_info.html
#
```

Le fichier principal de configuration prévoit l'inclusion d'un fichier complémentaire qui permet d'obtenir immédiatement des pages d'erreur multi-langues pour les principaux codes d'erreur HTTP. Cela peut constituer un bon point de départ pour un paramétrage plus personnalisé de ces pages.

```
# Multi-language error messages
Include conf/extra/httpd-multilang-errordoc.conf
```

```
/usr/local/apache2/conf/extra# cat httpd-multilang-errordoc.conf
#
# The configuration below implements multi-language error documents through
# content-negotiation.
#
# Required modules: mod_alias, mod_include, mod_negotiation
#
# We use Alias to redirect any /error/HTTP_<error>.html.var response to
# our collection of by-error message multi-language collections.  We use
# includes to substitute the appropriate text.
#
# You can modify the messages' appearance without changing any of the
# default HTTP_<error>.html.var files by adding the line:
#
#   Alias /error/include/ "/your/include/path/"
#
# which allows you to create your own set of files by starting with the
# /usr/local/apache2/error/include/ files and copying them to
/your/include/path/,
# even on a per-VirtualHost basis.  The default include files will display
# your Apache version number and your ServerAdmin email address regardless
# of the setting of ServerSignature.
```

```
Alias /error/ "/usr/local/apache2/error/"

<Directory "/usr/local/apache2/error">
    AllowOverride None
    Options IncludesNoExec
    AddOutputFilter Includes html
    AddHandler type-map var
    Order allow,deny
    Allow from all
    LanguagePriority en cs de es fr it ja ko nl pl pt-br ro sv tr
    ForceLanguagePriority Prefer Fallback
</Directory>

ErrorDocument 400 /error/HTTP_BAD_REQUEST.html.var
ErrorDocument 401 /error/HTTP_UNAUTHORIZED.html.var
ErrorDocument 403 /error/HTTP_FORBIDDEN.html.var
ErrorDocument 404 /error/HTTP_NOT_FOUND.html.var
ErrorDocument 405 /error/HTTP_METHOD_NOT_ALLOWED.html.var
ErrorDocument 408 /error/HTTP_REQUEST_TIME_OUT.html.var
ErrorDocument 410 /error/HTTP_GONE.html.var
ErrorDocument 411 /error/HTTP_LENGTH_REQUIRED.html.var
ErrorDocument 412 /error/HTTP_PRECONDITION_FAILED.html.var
ErrorDocument 413 /error/HTTP_REQUEST_ENTITY_TOO_LARGE.html.var
ErrorDocument 414 /error/HTTP_REQUEST_URI_TOO_LARGE.html.var
ErrorDocument 415 /error/HTTP_UNSUPPORTED_MEDIA_TYPE.html.var
ErrorDocument 500 /error/HTTP_INTERNAL_SERVER_ERROR.html.var
ErrorDocument 501 /error/HTTP_NOT_IMPLEMENTED.html.var
ErrorDocument 502 /error/HTTP_BAD_GATEWAY.html.var
ErrorDocument 503 /error/HTTP_SERVICE_UNAVAILABLE.html.var
ErrorDocument 506 /error/HTTP_VARIANT_ALSO_VARIES.html.var
```

12.11.4 Le fichier favicon.ico

Les fichiers des accès et des erreurs font souvent mention d'un fichier **favicon.ico** manquant à la racine du site.

Il s'agit d'une icône qui doit être mise à disposition par un site pour enjoliver, dans les navigateurs, les endroits où le site est mentionné (barre d'adresses, barre de titre, favoris…). Ce concept a commencé à être utilisé avec le navigateur Internet Explorer de Microsoft et a été adopté par quasiment l'ensemble des navigateurs.

L'usage de cette icône revêt donc désormais une certaine importance dans la finition d'un site et dans l'image que l'on souhaite communiquer.

Il est possible pour chaque page Web de préciser sa propre icône dans les entêtes de la balise <HEAD> avec, par exemple, un code du type:

```
<link rel=″shortcut icon″ type=″image/gif″ href=″favicon.gif″ />
```

Cela permet entre autres de s'affranchir du format d'origine *.ico* de Windows. Le plus souvent, la taille de cette icône est de 16×16 ou 32×32 pixels en 256 couleurs.

12.12 Programmes CGI

Le terme **CGI** (*Common Gateway Interface*) correspond à un schéma fonctionnel permettant à un serveur Web d'exécuter des programmes écrits dans n'importe quel langage (assez souvent en Perl, mais éventuellement en C, voire en shell).

La directive *ScriptAlias* donne accès à des programmes CGI stockés en dehors de l'arborescence Web. Lorsque l'URL du client navigateur sollicite un tel alias, le serveur Web comprend qu'il ne s'agit pas d'une requête simple et déclenche alors l'exécution du programme CGI. Le programme doit fournir ses résultats sur sa sortie standard et le serveur Apache se chargera d'envoyer cette réponse vers le client navigateur.

ScriptAlias

Description	Emplacement de programmes CGI
Syntaxe	ScriptAlias *URL-path file-path|directory-path*
Contexte	server config, virtual host
Module	mod_alias

Exemple

```
ScriptAlias /cgi-bin/ "/usr/local/apache2/cgi-bin/"
<Directory "/usr/local/apache2/cgi-bin">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>
```

Cet exemple nous montre le *ScriptAlias* par défaut proposé dans le fichier de configuration. Si nous soumettons l'URL *http://serveur/cgi-bin/test-cgi*, cela provoquera l'exécution du programme *test-cgi* situé dans le répertoire */usr/local/apache2/cgi-bin*.

Il est à noter qu'une strophe <*Directory*> protège le répertoire cible en nous interdisant notamment d'obtenir la liste des fichiers (*Options None*). Il est, de toute façon, impossible pour le client de récupérer le source des programmes CGI puisque, d'une part, ils sont stockés en-dehors de l'arborescence Web et parce que, d'autre part, la fonctionnalité de *ScriptAlias* en déclenche toujours l'exécution. Le client obtiendra donc toujours le document résultat généré par un programme dont le code lui reste inaccessible.

Le programme CGI doit donc écrire ses résultats sur sa sortie standard. Cette sortie est composée d'un **en-tête** suivi des données correspondantes.

Le format de l'en-tête est le plus souvent minimal car il sera complété par le serveur Apache. Il doit comporter au moins les deux lignes suivantes :

```
Content-type:   type/sous-type
une ligne vide
```

La première ligne correspond au type MIME du résultat (*text/html* la plupart du temps). L'oubli de la seconde ligne vide serait une des causes possible d'échec d'un script CGI.

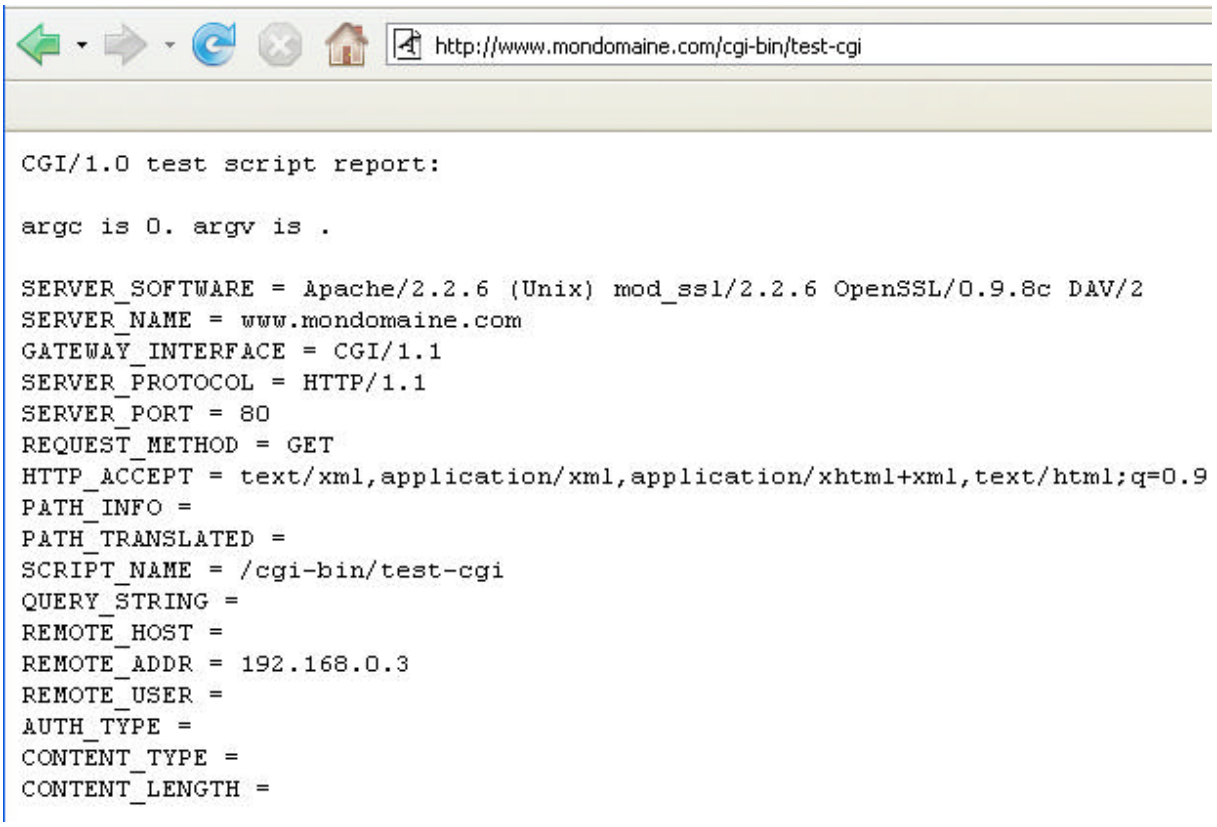
Exemple du script test-cgi (fourni dans la distribution Apache)

Ce script simple est écrit en shell (*#! /bin/sh*). Nous faisons apparaître *en italique* les commandes *echo* qui génèrent l'en-tête (type MIME *text/plain* suivi d'une ligne vide). Le reste du programme affiche un certain nombre de variables d'environnement gérées au niveau du protocole HTTP.

```
/usr/local/apache2/cgi-bin# cat test-cgi
#!/bin/sh

# disable filename globbing
set -f

echo "Content-type: text/plain; charset=iso-8859-1"
echo

echo CGI/1.0 test script report:
echo

echo argc is $#. argv is "$*".
echo

echo SERVER_SOFTWARE = $SERVER_SOFTWARE
echo SERVER_NAME = $SERVER_NAME
echo GATEWAY_INTERFACE = $GATEWAY_INTERFACE
echo SERVER_PROTOCOL = $SERVER_PROTOCOL
echo SERVER_PORT = $SERVER_PORT
echo REQUEST_METHOD = $REQUEST_METHOD
echo HTTP_ACCEPT = "$HTTP_ACCEPT"
echo PATH_INFO = "$PATH_INFO"
echo PATH_TRANSLATED = "$PATH_TRANSLATED"
echo SCRIPT_NAME = "$SCRIPT_NAME"
echo QUERY_STRING = "$QUERY_STRING"
echo REMOTE_HOST = $REMOTE_HOST
echo REMOTE_ADDR = $REMOTE_ADDR
echo REMOTE_USER = $REMOTE_USER
echo AUTH_TYPE = $AUTH_TYPE
echo CONTENT_TYPE = $CONTENT_TYPE
echo CONTENT_LENGTH = $CONTENT_LENGTH
/usr/local/apache2/cgi-bin#
```

Il est nécessaire que le serveur Apache possède les droits d'exécution sur ce programme. Si ce n'était pas le cas, cela provoquerait une erreur (*Internal Server Error*, code 500) lors de l'utilisation.

Quand toutes les bonnes conditions sont réunies, le navigateur reçoit donc le résultat du script :



Le ScriptAlias /cgi-bin/ nous permet d'exécuter le programme test-cgi.

Un programme CGI peut recevoir des données de différentes manières et, notamment, via des formulaires HTML. Le protocole HTTP définit plusieurs méthodes d'acquisition de données :

- méthode **GET**

Les données sont ajoutées à l'URL référençant le script CGI. Le caractère *?* marque le début d'une chaîne de caractères qui sera affectée dans la variable d'environnement *QUERY_STRING*.

- méthode **POST**

Il s'agit de la méthode la plus utilisée. Les données ne sont pas ajoutées à l'URL mais sont transmises sur l'entrée standard du programme, avec une taille en octets fournie par la variable d'environnement *CONTENT_LENGTH*.

- variables d'**environnement**

Le serveur Web fournit toujours un ensemble de variables d'environnement au programme CGI, telles que, par exemple :

- SERVER_SOFTWARE Nom et version du serveur

- SERVER_NAME Nom du serveur

- REMOTE_HOST Nom du client

- REMOTE_ADDR Adresse IP du client
- HTTP_USER_AGENT Informations sur le client (logiciel, version)

Il est possible d'autoriser l'exécution des programmes CGI sans recourir à la directive *ScriptAlias*.

Tout fichier d'extension **.cgi** pourra être considéré comme un programme CGI si nous lui associons le **handler** *cgi-script*. Nous pouvons rappeler qu'un *handler* désigne une action associée à un fichier autrement que par son type.

Une directive **AddHandler** est déjà prévue dans le fichier de configuration si nous souhaitons activer cette fonctionnalité :

```
AddHandler  cgi-script  .cgi
```

Cependant, cette seule directive n'est pas suffisante pour exécuter des scripts en-dehors d'un répertoire désigné par une directive *ScriptAlias*.

Il sera nécessaire que le répertoire concerné soit doté de l'option *ExecCGI*.

```
Options  +ExecCGI
```

12.13 Performances

12.13.1 Serveurs n'utilisant pas les threads (module prefork)

Le mode de fonctionnement baptisé **prefork** correspond au choix de compilation par défaut sur les plates-formes Unix/Linux.

Un processus de contrôle est responsable de l'écoute des requêtes des clients et du lancement de processus fils pour satisfaire ces requêtes. Les directives *User* et *Group* précisent les propriétaires de ces fils.

Le serveur ajuste la charge en augmentant ou diminuant le nombre de processus. Apache garde, si possible, plusieurs fils inactifs en réserve pour anticiper efficacement sur la création de ces processus.

Les directives **StartServers**, **MinSpareServers**, **MaxSpareServers** et **MaxClients** permettent de calibrer le nombre de processus.

Les valeurs par défaut sont a priori satisfaisantes. Les sites qui veulent pouvoir traiter beaucoup de requêtes simultanées peuvent éventuellement augmenter la valeur de *StartServers* et de *MaxClients*.

```
# prefork MPM
# StartServers: number of server processes to start
# MinSpareServers: minimum number of server processes which are kept spare
# MaxSpareServers: maximum number of server processes which are kept spare
# MaxClients: maximum number of server processes allowed to start
# MaxRequestsPerChild: maximum number of requests a server process serves
<IfModule mpm_prefork_module>
    StartServers          5
    MinSpareServers       5
    MaxSpareServers      10
    MaxClients          150
    MaxRequestsPerChild   0
</IfModule>
```

La directive **MaxRequestsPerChild** contrôle la durée de vie des processus fils en indiquant le nombre maximum de requêtes traitées par un même processus. La valeur 0 indique que le processus ne meure jamais.

12.13.2 Serveurs utilisant les threads (module worker)

Le mode de fonctionnement baptisé **worker** doit faire l'objet d'un choix explicite de compilation sur les plates-formes Unix/Linux. Il s'agit d'un modèle hybride qui utilise conjointement des processus classiques et des *threads*.

De la même façon que dans le modèle *prefork*, un processus de contrôle se charge du lancement des processus fils et ajuste la charge en augmentant ou diminuant le nombre de processus.

Ces processus fils comportent chacun un certain nombre de threads et la directive **ThreadsPerChild** précise le nombre de threads par processus. Chaque thread écoute les

requêtes des clients et se charge de les traiter. Apache garde, si possible, plusieurs threads inactifs en réserve pour anticiper efficacement.

Les directives **StartServers**, **MinSpareThreads**, **MaxSpareThreads** et **MaxClients** permettent de calibrer le nombre de processus et de threads. Les valeurs par défaut sont a priori satisfaisantes.

```
# worker MPM
# StartServers: initial number of server processes to start
# MaxClients: maximum number of simultaneous client connections
# MinSpareThreads: minimum number of worker threads which are kept spare
# MaxSpareThreads: maximum number of worker threads which are kept spare
# ThreadsPerChild: constant number of worker threads in each server process
# MaxRequestsPerChild: maximum number of requests a server process serves
<IfModule mpm_worker_module>
    StartServers          2
    MaxClients          150
    MinSpareThreads      25
    MaxSpareThreads      75
    ThreadsPerChild      25
    MaxRequestsPerChild   0
</IfModule>
```

12.13.3 Autres paramètres concernant les performances

La documentation officielle Apache comporte une rubrique assez détaillée sur le réglage des performances (*Apache Performance Tuning*). Nous encourageons le lecteur à s'y reporter.

En complément des paramètres directement liés au module de type MPM utilisé (*prefork* ou *worker*), un certain nombre d'éléments complémentaires sont à prendre en compte, parmi lesquels :

- la directive **HostNameLookups**

Il faut s'assurer a priori que cette directive prend bien la valeur *Off* pour éviter une résolution DNS inverse pour chaque requête, lors de sa mémorisation dans les fichiers de trace. La commande **logresolve** permettra d'effectuer cette résolution en différé lors du traitement ultérieur de ces fichiers de trace.

- la directive **AllowOverride**

Il faut veiller à être le plus souvent possible en *AllowOverride None* pour éviter la recherche des fichiers *.htaccess* lors de chaque requête.

- les **liens symboliques**

Pour la directive *Options*, les arguments *FollowSymLinks* et *SymLinksIfOwnerMatch* peuvent provoquer des recherches inutiles de liens symboliques lors de chaque requête.

Apache fournit un outil de test de performance par l'intermédiaire de la commande **ab** (*apache benchmark*) qui permet notamment de voir le nombre de requêtes par secondes que le serveur est capable de traiter.

De nombreuses options sont disponibles :

```
# ab –h
Usage: ab [options] [http[s]://]hostname[:port]/path
Options are:
    -n requests     Number of requests to perform
    -c concurrency  Number of multiple requests to make
    -t timelimit    Seconds to max. wait for responses
    -p postfile     File containing data to POST
    -T content-type Content-type header for POSTing
    -v verbosity    How much troubleshooting info to print
    -w              Print out results in HTML tables
    -i              Use HEAD instead of GET
    -x attributes   String to insert as table attributes
    -y attributes   String to insert as tr attributes
    -z attributes   String to insert as td or th attributes
    -C attribute    Add cookie, eg. 'Apache=1234. (repeatable)
    -H attribute    Add Arbitrary header line, eg. 'Accept-Encoding: gzip'
                    Inserted after all normal header lines. (repeatable)
    -A attribute    Add Basic WWW Authentication, the attributes
                    are a colon separated username and password.
    -P attribute    Add Basic Proxy Authentication, the attributes
                    are a colon separated username and password.
    -X proxy:port   Proxyserver and port number to use
    -V              Print version number and exit
    -k              Use HTTP KeepAlive feature
    -d              Do not show percentiles served table.
    -S              Do not show confidence estimators and warnings.
    -g filename     Output collected data to gnuplot format file.
    -e filename     Output CSV file with percentages served
    -h              Display usage information (this message)
    -Z ciphersuite  Specify SSL/TLS cipher suite (See openssl ciphers)
    -f protocol     Specify SSL/TLS protocol (SSL2, SSL3, TLS1, or ALL)
#
```

Exemple (100 fois 100 requêtes simultanées vers www.debian.org)

```
# ab -n 100 -c 100 -k http://www.debian.org/
This is ApacheBench, Version 2.0.40-dev <$Revision: 1.146 $> apache-2.0
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Copyright 2006 The Apache Software Foundation, http://www.apache.org/

Benchmarking www.debian.org (be patient).....done


Server Software:        Apache/2.0.54
Server Hostname:        www.debian.org
Server Port:            80

Document Path:          /
Document Length:        14370 bytes

Concurrency Level:      100
Time taken for tests:   2.398390 seconds
Complete requests:      100
```

```
Failed requests:        0
Write errors:           0
Keep-Alive requests:    100
Total transferred:      1551466 bytes
HTML transferred:       1498854 bytes
Requests per second:    41.69 [#/sec] (mean)
Time per request:       2398.390 [ms] (mean)
Time per request:       23.984 [ms] (mean, across all concurrent
requests)
Transfer rate:          631.67 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0   24  42.1      0     115
Processing:   138  541 416.1    361    2148
Waiting:       65  320 292.0    257    1499
Total:        138  565 448.1    433    2263

Percentage of the requests served within a certain time (ms)
  50%    433
  66%    614
  75%    638
  80%    806
  90%   1484
  95%   1564
  98%   1632
  99%   2263
 100%   2263 (longest request)
#
```

12.14 Connexions sécurisées avec SSL

12.14.1 Algorithmes de chiffrement

Avant d'expliquer plus avant le fonctionnement du protocole, nous allons présenter brièvement quelques notions préliminaires.

Les algorithmes dits de chiffrement ou de cryptographie sont indispensables dans les communications d'informations sensibles. Ils permettent d'atteindre plusieurs objectifs :

- l'**authentification**

Le chiffrement des informations fournies lors d'une authentification est nécessaire par rapport aux nombreuses possibilités d'écoute disponibles sur un réseau physique.

- la **confidentialité**

Il est également nécessaire de pouvoir coder le contenu d'un échange pour empêcher les lectures d'informations par des personnes non habilitées.

- l'**intégrité**

Le chiffrement peut être un moyen de s'assurer qu'un document n'a pas subi de modification.

- la **signature électronique**

La signature électronique permet d'identifier formellement les interlocuteurs concernés par un échange d'informations.

Chiffrement symétrique

Ce type d'algorithme s'appuie sur une **clé secrète unique**, utilisée à la fois pour coder et décoder les informations. Les deux interlocuteurs doivent se partager cette clé secrète.

L'intérêt majeur de ce type d'algorithme est la performance. On s'en servira donc lors des transferts de données. L'inconvénient assez évident réside dans le besoin de générer un clé secrète pour chaque couple d'interlocuteurs.

Parmi ces algorithmes, on peut citer DES, 3DES, RC4, RC5, AES…

Chiffrement asymétrique

Ce type d'algorithme s'appuie sur un couple de clés (**clé privée** et **clé publique**) générées ensemble et dépendantes via un algorithme mathématique.

La clé publique sera, par définition, diffusée à tous les interlocuteurs potentiels. Assez souvent, la clé publique sert à coder les informations et la clé secrète est utilisée pour effectuer le décodage. Tout le monde peut chiffrer un message avec la clé publique mais seul le propriétaire de la clé secrète sera capable de le déchiffrer.

Les deux principaux algorithmes utilisés sont **DSA** (*Digital Signature Algorithm*) et **RSA** (*Rivest, Shamir, Adleman*, les trois auteurs de l'algorithme). Ces algorithmes sont moins performants que les algorithmes symétriques. Ils seront donc utilisés dans les étapes d'authentification alors que les échanges effectifs seront ensuite réalisés via des algorithmes symétriques, à l'aide de clés de session générées dynamiquement..

Signature électronique d'un message

Les algorithmes asymétriques peuvent être utilisés pour effectuer des signatures électroniques de messages, permettant ainsi de garantir l'identité de l'expéditeur.

L'expéditeur va générer une empreinte de son message avec des fonctions dites de *hachage* (algorithme *MD5* par exemple). Cette empreinte va ensuite être codée à l'aide de la clé secrète de l'expéditeur et associée au message proprement dit.

La clé publique de l'expéditeur permettra de déchiffrer l'empreinte et de s'assurer que le message provient bien de l'interlocuteur prévu.

En combinant donc un message codé avec la clé publique du destinataire et une empreinte codée avec la clé privée de l'expéditeur, il est possible de s'assurer de la confidentialité du contenu ainsi que de l'identité de l'expéditeur.

12.14.2 Protocole SSL

SSL (*Secured Sockets Layer*) est un protocole proposé à l'origine par la société *Netscape*. Il s'est imposé comme standard de fait dans les activités de sécurisation des communications sur le Web.

Dans le modèle fonctionnel TCP/IP, SSL vient s'intercaler entre les protocoles applicatifs tels que HTTP et la couche transport TCP pour offrir des services de chiffrement entre clients et serveurs.

Dans le cadre des échanges sur le Web, les connexions sécurisées via SSL permettront de s'assurer de l'identité des serveurs lors, par exemple, d'un achat en ligne. Le serveur pourra également s'assurer de l'identité du client lors de l'envoi d'informations sensibles. Ces mêmes informations pourront être codées lors des échanges proprement dits afin d'assurer leur confidentialité.

Certificats

La notion de **certificat** est fondamentale pour la mise en œuvre des authentifications SSL. Elle permet de garantir l'identité d'un interlocuteur.

Un certificat rassemble un ensemble d'informations sur l'identité de son détenteur, accompagnées de sa clé publique, indispensable pour les échanges chiffrés.

Pour garantir son authenticité, le certificat est, de plus, signé électroniquement à l'aide de la clé privée d'un organisme officiel, qualifié d'**autorité de certification** (*CA : Certification Authority*). Les navigateurs Web pourront déchiffrer les certificats à l'aide des clés publiques , largement diffusées, des organismes officiels.

La structure d'un certificat est décrit par la norme *ISO X509* et contient notamment :

- la description de l'entité concernée,

- la clé publique de cette entité,

- le nom de l'autorité de certification,

- la période de validité,

- l'algorithme de chiffrement utilisé,

ainsi que,

- la signature des informations précédentes à l'aide de la clé privée de l'autorité de certification.

Des certificats *auto-signés* peuvent être fabriqués lors de la configuration du serveur Apache ou dans le cadre d'un réseau local ou d'un site Intranet. Il sera nécessaire de les remplacer par des certificats délivrés par des autorités de certification commerciales (par exemple, *www.verisign.com*) lors de la mise en service du site nécessitant l'usage des connexions sécurisées.

Déroulement d'une connexion SSL

Sans rentrer dans tous les détails, une connexion SSL peut se résumer en plusieurs étapes successives :

- Le client contacte le serveur en lui indiquant les systèmes de chiffrement supportés.

- Le serveur envoie au client son certificat et son choix en terme d'algorithme de chiffrement.

- Le client déchiffre le certificat à l'aide de la clé publique de l'autorité de certification concernée.

- Le client crée une clé secrète aléatoire (**clé de session**) et il l'envoie au serveur en la signant avec la clé publique de celui-ci.

- Le serveur est le seul capable de déchiffrer cette clé de session grâce à sa propre clé privée. La clé de session correspond à un algorithme de chiffrement symétrique, plus performant pour les échanges de données.

12.14.3 Mise en oeuvre de SSL pour Apache

Le module Apache **mod_ssl** s'appuie sur la librairie **OpenSSL** et permet la mise en oeuvre de connexions sécurisées via le protocole SSL (*Secured Sockets Layer*).

OpenSSL est une implémentation Open Source du protocole SSL, intégrée systématiquement dans les distributions Linux. Le site de référence est *www.openssl.org*.

12.14.3.a Création d'un certificat

Le fichier principal de configuration d'Apache prévoit l'inclusion d'un fichier destiné à la mise en œuvre de la configuration SSL :

```
# Secure (SSL/TLS) connections
Include conf/extra/httpd-ssl.conf
#
```

Une première étape va consister à générer un **certificat** pour notre serveur Web. Dans une phase de test, il pourra s'agir d'un certificat privé. Nous devrons plus tard obtenir un certificat officiel auprès d'une autorité de certification reconnue.

Pour ce faire, nous commençons par générer notre clé privée.

Nous nous positionnons dans le répertoire */usr/local/apache2/conf* car c'est l'emplacement prévu par défaut dans le fichier de configuration pour stocker la clé et le certificat.

Nous générons tout d'abord la clé privée de notre serveur (algorithme RSA). Nous décidons de ne pas associer d'empreinte (pas d'option *-des3*). Ainsi, il ne sera pas nécessaire de saisir un mot de passe à chaque utilisation et notamment au démarrage du serveur Apache.

```
/usr/local/apache2/conf# openssl genrsa -out server.key 1024
Generating RSA private key, 1024 bit long modulus
..............................++++++
..............................++++++
e is 65537 (0x10001)
/usr/local/apache2/conf# cat server.key
-----BEGIN RSA PRIVATE KEY-----
MIICWwIBAAKBgQDIYtxrD6i0xMCWopwEq/9jMqfUM2ySbkbaG0l35aCuEDvKUMG0
QWmAd53exk3lnj2N8OAva3finhVLtu6lPYT2Y4erNlPmqjuIBwFniPQXAPJDgG5H
4zHa3OElRxMF+w+o7TtYhOUGRWTk9CmuxaKPhWqoE9zzX/vNYyATqDY7VwIDAQAB
AoGASCdA0tQKnTyTVCotH4mHJgLhZ4k0uO04WeSlaaSnXMW/obO+a9d506Y+lC63
ccyuwWNQdgKuYk0lweOWvq0OKL2p5ZCSYXE4e91YhqS8uD88NcmsryC/UzbiU5f6
xxuTyWnx5C8Xm7NWRYRYEj+2JijvOp4lAwXVHsEeB7dzmdECQQD+fjs7vQhFO03C
e/iGE8lCM7ipVxjW/0IC6iPh/JkyICY4ML8k5KYyCJ6FZOdO8XomXxtmoYGvLHTA
KNfga4nzAkEAyZKc17kmQi2rqCB1Ae2uDratUU2iSQfKMAToayid4XySdkIvT4Oh
znk7JMmGOU6rHEp0CTVxcV35bbwwBsNeDQJAVWown0A2wmXBF4FcTEPfzKRnWg6t
e5t4oFnNaU/KaxR3P2y8+rjzPt+9tK/FIq3RfYd2Pt/+ErktBkGvEteFGQJAWOgL
ubJ2R3YxYkR+l00oGmFEbLnbPG7tJOTlRm5xxITXbMWgbGfYtGMQTRd91kT2f9Hv
ACjMUn5qh5YbcMA2+QJAULEzxiLgt6s9c/1KreZ8QubhLfxvL9iJWm7a1QIT9J7d
s+44zOihKMyhW/GbajfjV9Vu1nzmDpAUV7+2R9x8iA==
-----END RSA PRIVATE KEY-----
```

```
/usr/local/apache2/conf#
```

Nous pouvons ensuite préparer la demande de certificat.

```
/usr/local/apache2/conf# openssl req -new -key server.key -out monsite.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:France
Locality Name (eg, city) []:Paris
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Société Bidule
Organizational Unit Name (eg, section) []:
Common Name (eg, YOUR name) []:www.bidule.fr
Email Address []:webmaster@bidule.fr

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
/usr/local/apache2/conf# cat monsite.csr
-----BEGIN CERTIFICATE REQUEST-----
MIIBzTCCATYCAQAwgYwxCzAJBgNVBAYTAkZSMRYwFAYDVQQIEw1JbGUgZGUgRnJh
bmNlMQ4wDAYDVQQHEwVQYXJpczEXMBUGA1UEChQOU29jael06SBCaWR1bGUxFzAV
BgNVBAMTDnd3dy5iaWR1bGUuY29tMSMwIQYJKoZIhvcNAQkBFhR3ZWJtYXN0ZXJA
YmlkdWxlLmNvbTCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAljHlo0JRjICl
HLxSUzRNGXVJZShiWE398w0beMWjUaSSEiZV+q6rVpbbpz4d+41EC/qyQAsNUa8N
fo40OC9aVRZJgMvjoIEcnGp1IHYZ6eFviuMHM6vaThU6xF/rcIjLlYyp7EH4/lrG
pTRLPlg8YF+EYTyXXcT1ZdpXlrLCZbsCAwEAAaAAMA0GCSqGSIb3DQEBBAUAA4GB
AGxpJmd0D1k2cQ/L/LxlcsghNehCLbA5waAec8kd/+bsNpF2WTIXkg95DsmDsm9B
zfO0ektkEb3YeVsP3jgr+IllDhykfrq29YHlohDUAIN8bPYo/WXIK8CB+T59oAWV
K7zWq8Y6cQ2JX4Imo1WJoAKRxzzpefSc6pI6476EigT+
-----END CERTIFICATE REQUEST-----
#
```

Il nous faudrait donc ensuite envoyer le fichier d'extension *.csr* obtenu à une autorité de certification et attendre le retour du certificat officiel.

Le temps d'un test ou pour un simple usage interne, nous pouvons générer et signer nous-même notre propre certificat. Nous générons donc ce certificat final (fichier *server.crt*) à partir du fichier de requêtes fabriqué précédemment (fichier d'extension *.csr*).

```
# openssl x509 -req -in monsite.csr -signkey server.key -out server.crt
Signature ok
subject=/C=FR/ST=France/L=Paris/O=Soci\xC3\xA9t\xC3\xA9
Bidule/CN=www.bidule.fr/emailAddress=webmaster@bidule.fr
Getting Private key
#
# cat server.crt
-----BEGIN CERTIFICATE-----
MIICgzCCAewCCQCM1F7F3OODRzANBgkqhkiG9w0BAQUFADCBhTELMAkGA1UEBhMC
```

```
RlIxDzANBgNVBAgTBkZyYW5jZTEOMAwGA1UEBxMFUGFyaXMxGTAXBgNVBAoUEFNv
Y2nDqXTDqSBCaWR1bGUxFjAUBgNVBAMTDXd3dy5iaWR1bGUuZnIxIjAgBgkqhkiG
9w0BCQEWE3dlYm1hc3RlckBiaWR1bGUuZnIwHhcNMDcxMDIwMTMxMTM4WhcNMDcx
MTE5MTMxMTM4WjCBhTELMAkGA1UEBhMCRlIxDzANBgNVBAgTBkZyYW5jZTEOMAwG
A1UEBxMFUGFyaXMxGTAXBgNVBAoUEFNvY2nDqXTDqSBCaWR1bGUxFjAUBgNVBAMT
DXd3dy5iaWR1bGUuZnIxIjAgBgkqhkiG9w0BCQEWE3dlYm1hc3RlckBiaWR1bGUu
ZnIwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMhi3GsPqLTEwJainASr/2My
p9QzbJJuRtobSXfloK4QO8pQwbRBaYB3nd7GTeWePY3w4C9rd+KeFUu27qU9hPZj
h6s2U+aqO4gHAWeI9BcA8kOAbkfjMdrc4SVHEwX7D6jtO1iE5QZFZOT0Ka7Foo+F
aqgT3PNf+81jIBOoNjtXAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEALoXGyb3PXZIu
RYkhjHciw7rHcZaaApkZIuGRb6rV8LPrCvX7KHmEWwwrgRnmRK8+kk2E97pO4prF
84jc4zOwA6VbcGcQ2gK75/jk+8eOMIJb0OfqRaIPkwLkti5i0OYr7EpYwBGlBH1k
FrLNG/FnPmbkotMnq9bl2ug4vNJZqDc=
-----END CERTIFICATE-----
#
```

Pour la gestion des certificats, la distribution OpenSSL propose les utilitaires **CA.pl** (écrit en Perl) ou **CA.sh** (écrit en shell) qui ont pour objectif de simplifier l'utilisation sous-jacente de la commande directe *openssl*.

Ces scripts sont également fournis par le paquet logiciel Debian openssl, dans le répertoire /usr/lib/ssl/misc.

12.14.3.b Configuration Apache

Les connexions SSL (URLs de type *https://*) se feront sur le port 443 et seront gérées à l'aide d'un hôte virtuel prévu à cet effet. Les types MIME adéquats devront être associés aux certificats (fichiers d'extension *.crt*) et aux fichiers de révocation des certificats (fichiers d'extension *.crl*).

Après compilation d'Apache avec le module *mod_ssl*, on trouve les directives suivantes dans le fichier de configuration principal :

```
LoadModule ssl_module modules/mod_ssl.so

Include conf/extra/httpd-ssl.conf
```

Le fichier de configuration *httpd-ssl.conf* est quasiment opérationnel. Ce fichier est largement auto-commenté et permet d'approfondir le détail de chaque directive dans la documentation officielle. Nous faisons apparaître en gras les directives essentielles.

```
/usr/local/apache2/conf/extra# cat httpd-ssl.conf
# This is the Apache server configuration file providing SSL support.
# It contains the configuration directives to instruct the server how to
# serve pages over an https connection. For detailing information about these
# directives see <URL:http://httpd.apache.org/docs/2.2/mod/mod_ssl.html>
#

.....................................................
#
```

```
# When we also provide SSL we have to listen to the
# standard HTTP port (see above) and to the HTTPS port
#
# Note: Configurations that use IPv6 but not IPv4-mapped addresses need two
#       Listen directives: "Listen [::]:443" and "Listen 0.0.0.0:443"
#
Listen 443

##
##  SSL Global Context
##
##  All SSL configuration in this context applies both to
##  the main server and all SSL-enabled virtual hosts.
##

#
#   Some MIME-types for downloading Certificates and CRLs
#
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl    .crl

.....................................................
##
## SSL Virtual Host Context
##

<VirtualHost _default_:443>

#   General setup for the virtual host
DocumentRoot "/usr/local/apache2/htdocs"
ServerName www.example.com:443
ServerAdmin you@example.com
ErrorLog "/usr/local/apache2/logs/error_log"
TransferLog "/usr/local/apache2/logs/access_log"

#   SSL Engine Switch:
#   Enable/Disable SSL for this virtual host.
SSLEngine on

#   SSL Cipher Suite:
#   List the ciphers that the client is permitted to negotiate.
#   See the mod_ssl documentation for a complete list.
SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL

#   Server Certificate:
#   Point SSLCertificateFile at a PEM encoded certificate.  If
#   the certificate is encrypted, then you will be prompted for a
#   pass phrase.  Note that a kill -HUP will prompt again.  Keep
#   in mind that if you have both an RSA and a DSA certificate you
#   can configure both in parallel (to also allow the use of DSA
#   ciphers, etc.)
SSLCertificateFile "/usr/local/apache2/conf/server.crt"
#SSLCertificateFile "/usr/local/apache2/conf/server-dsa.crt"

#   Server Private Key:
#   If the key is not combined with the certificate, use this
#   directive to point at the key file.  Keep in mind that if
#   you've both a RSA and a DSA private key you can configure
```

```
#   both in parallel (to also allow the use of DSA ciphers, etc.)
SSLCertificateKeyFile "/usr/local/apache2/conf/server.key"

.....................................................
</VirtualHost>

.....................................................
#
```

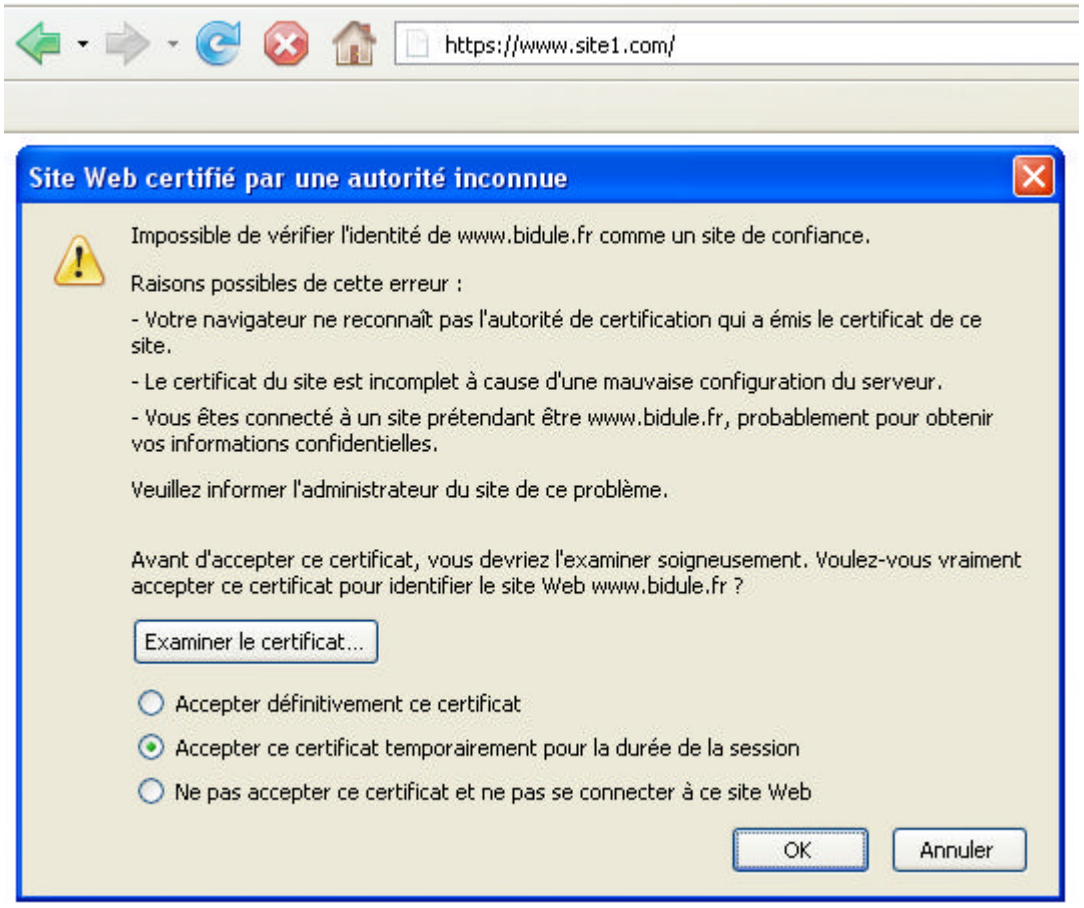
Nous n'avons que quelques lignes à modifier dans la strophe de l'hôte virtuel, par exemple :

```
DocumentRoot   "/usr/local/apache2/htdocs/site1.com/test_https"
ServerName     www.site1.com:443
ServerAdmin    www-admin@site1.com
ErrorLog       /usr/local/apache2/logs/site1.com_https-error_log
TransferLog    /usr/local/apache2/logs/site1.com_https-access_log
```

Par ailleurs, nous avions déjà placé la clé privée et le certificat aux bons endroits :

```
SSLCertificateFile     /usr/local/apache2/conf/server.crt
SSLCertificateKeyFile  /usr/local/apache2/conf/server.key
```

Après redémarrage, nous sollicitons le serveur via une URL de type **https://**. La connexion est opérationnelle même si le certificat ne peut pas, pour l'instant, être accepté en l'état par le navigateur.



Avertissement du navigateur par rapport au certificat auto-signé

Si nous acceptons, même temporairement, ce certificat, nous pouvons néanmoins tester le bon accès à la partie sécurisée de notre site via l'URL de type https.

Accès au site en https:// après acceptation du certificat auto-signé