

---

# Administration système UNIX

version 7.0

## Partie 2

2004 – 2005

Thierry Besançon – Philippe Weill



Formation Permanente  
Université Pierre et Marie Curie - PARIS 6  
Tour centrale 13<sup>e</sup> étage  
4, place Jussieu  
75252 Paris Cedex 05  
Tél. : 01-44-27-58-49 ou 01-44-27-58-50  
01-44-27-38-19 ou 01-44-2738-25  
Fax : 01-44-27-27-15

e-mail : [formation.permanente@admp6.jussieu.fr](mailto:formation.permanente@admp6.jussieu.fr)

---

Les animateurs de ce cours peuvent être joints aux adresses suivantes :

Thierry.Besancon@formation.jussieu.fr

Philippe.Weill@formation.jussieu.fr

Ce cours est disponible au format PDF sur le web à l'URL :

<http://www.formation.jussieu.fr/ars/2004-2005/UNIX/cours/>

Si vous améliorez ce cours, merci de m'envoyez vos modifications ! :-)

Copyright (c) 1997-2005 by Thierry.Besancon@formation.jussieu.fr

This material may be distributed only subject to the terms and conditions set forth in the Open Publication Licence, v1.0 or later (the latest version is available at <http://www.opencontent.org/openpub/>).

"...the number of UNIX installations has grown to 10, with more expected..."

- Dennis Ritchie and Ken Thompson, June 1972

## Table des matières

N° de transparent

Chapitre 15	Contenu d'une arborescence type d'un système Unix	1
§1	Les difficultés pratiques	1
§2	Tentative de normalisation FHS Linux	4
§3	Panorama d'un système Unix moderne type	5
§4	Directory /sbin	7
§5	Directory /bin	8
§6	Directory /usr/sbin	9
§7	Directory /usr/bin	10
§8	Directory /usr/local/sbin	11
§9	Directory /usr/local/bin	12
§10	Directory /lib	13
§11	Directory /usr/lib	14
§12	Directory /usr/local/lib	15
§13	Directory /usr/include	16
§14	Directory /usr/local/include	17
§15	Directory /etc	18
§16	Directory /usr/local/etc	19
§17	Directory /dev	20
§18	Trou noir pour redirection : /dev/null	23
§19	(Windows :: NUL)	24
§20	Trou noir pour redirection : /dev/console	25
§21	Directory /proc	26
§22	Directory /tmp	33
§23	Directory /var/log	34
§24	Directory /var/mail	35
§25	Directory /var/run	36
§26	Directory /var/spool	37
§27	Directory /var/tmp	38
§28	Directory /mnt	39
§29	Directory /opt	40
§30	Directory /usr/src	41
§31	Directory /sys	42
Chapitre 16	Démarrage d'une station Unix	43
§1	Etape 1 du boot : le moniteur	44
§2	Etape 2 du boot : le chargeur primaire	48
§3	Etape 3 du boot : le chargeur secondaire	51
§4	Etape 4 du boot : le chargement du noyau Unix	52
§5	Etape 5 du boot : init et ses scripts de démarrage	56
§6	Scripts de démarrage System-V	60
§7	Gestion des scripts de démarrage LINUX : chkconfig	73
§8	Scripts de démarrage BSD	80
§9	Boot en single user/mono utilisateur	81
Chapitre 17	Arrêt d'une station Unix	87
§1	Mécanisme APM	88
§2	Mécanisme ACPI	90
§3	Commande d'arrêt : halt	92
§4	Commande d'arrêt : fasthalt	93
§5	Commande d'arrêt : poweroff	94
§6	Commande de redémarrage : reboot	95
§7	Commande de redémarrage : fastboot	96
§8	Commande d'arrêt et de redémarrage : shutdown	97
§9	Commande d'arrêt et de redémarrage : kill -TERM 1	101

§ 10	Informations de connexions : <code>last</code> . . . . .	102
§ 11	(Windows : : commandes d'arrêt et de redémarrage) . . . . .	103
Chapitre 18	Le noyau Unix et les périphériques . . . . .	106
§ 1	Assurer la validité du noyau . . . . .	108
§ 2	Configurer le noyau . . . . .	109
§ 3	Modifier le noyau . . . . .	110
§ 4	Compilation du noyau . . . . .	113
§ 5	Compilation du noyau LINUX . . . . .	115
§ 6	Compilation du noyau FreeBSD . . . . .	117
§ 7	Compilation du noyau SOLARIS . . . . .	118
§ 8	Noyaux modulaires . . . . .	119
§ 9	Modifications dynamiques du noyau LINUX : <code>sysctl, /etc/sysctl.conf</code> . . . . .	122
§ 10	Modifications dynamiques du noyau FreeBSD : <code>sysctl, /etc/sysctl.conf</code> . . . . .	124
§ 11	Modifications dynamiques du noyau SOLARIS : <code>nndd, /etc/system</code> . . . . .	126
§ 12	Messages du noyau : <code>dmesg</code> . . . . .	128
§ 13	Le noyau Unix et les périphériques . . . . .	129
§ 14	Types d'objets Unix . . . . .	130
§ 15	Répertoire <code>/dev</code> . . . . .	131
§ 16	Mode bloc . . . . .	134
§ 17	Mode caractère . . . . .	136
§ 18	Device drivers/Pilotes . . . . .	138
§ 19	Minor number / major number . . . . .	140
§ 20	Création des fichiers spéciaux : <code>mknod</code> . . . . .	141
§ 21	Copie de devices : <code>dd</code> . . . . .	146
Chapitre 19	Systèmes de fichiers . . . . .	149
§ 1	Technologies de disques durs . . . . .	150
§ 2	Disques durs IDE . . . . .	155
§ 3	Disques durs SCSI . . . . .	156
§ 4	Duplicateurs de disques durs . . . . .	158
§ 5	Tests de disques durs . . . . .	159
§ 6	Technologie NAS : Network Attached Storage . . . . .	160
§ 7	Technologie SAN : Storage Array Network . . . . .	162
§ 8	Technologie RAID . . . . .	163
§ 9	Filesystem . . . . .	172
§ 10	Partitionnement de disques durs, <code>format, fdisk</code> . . . . .	173
§ 11	Liste des partitions SOLARIS : <code>prtvtoc</code> . . . . .	176
§ 12	Liste des partitions LINUX : <code>fdisk</code> . . . . .	177
§ 13	Nommage des partitions . . . . .	178
§ 14	Formatage (1) : <code>mkfs</code> . . . . .	183
§ 15	Formatage (2) : <code>newfs</code> . . . . .	184
§ 16	Montage de filesystems : <code>mount</code> . . . . .	186
§ 17	Démontage de filesystems : <code>umount</code> . . . . .	188
§ 18	Remontage à chaud de filesystems . . . . .	189
§ 19	Identification des causes des filesystems busy : <code>fuser</code> . . . . .	190
§ 20	Identification des causes des filesystems busy : <code>lsof</code> . . . . .	191
§ 21	Liste des partitions montées : <code>df</code> . . . . .	192
§ 22	Liste des partitions montées (2) : <code>mount</code> . . . . .	197
§ 23	Montage automatique au boot : <code>/etc/fstab, /etc/vfstab</code> . . . . .	199
§ 24	Gestion des quotas : <code>quotaon, quota, edquota</code> . . . . .	201
§ 25	Structure interne associée à un objet : <code>inode</code> . . . . .	205
§ 26	Informations sur les inodes : <code>df</code> . . . . .	210
§ 27	Inodes 0, 1, 2 et 3 . . . . .	213
§ 28	Répertoire <code>lost+found</code> . . . . .	214
§ 29	Cohérence des disques : <code>sync</code> . . . . .	215

§ 30	Cohérence des disques : <code>update</code> . . . . .	217
§ 31	Vérification de la cohérence des disques : <code>fsck</code> . . . . .	218
§ 32	Stratégie d'allocation des blocs, <code>MINFREE</code> , <code>tunefs</code> . . . . .	224
§ 33	Vitesse de rotation du disque . . . . .	225
§ 34	Snapshots . . . . .	226
§ 35	(Windows : : Snapshots) . . . . .	231
§ 36	Logical Volume Managers, LVM . . . . .	238
§ 37	Filesystem journalisé . . . . .	239
§ 38	Gravure de CD/DVD (1) : image ISO 9660 . . . . .	241
§ 39	Gravure de CD/DVD (2) : manipulation d'une image ISO 9660 . . . . .	246
§ 40	Gravure de CD/DVD (3) : gravure d'une image ISO 9660 . . . . .	253
§ 41	Droits étendus : Access Control Lists (ACL) . . . . .	266
§ 42	ACL sous LINUX . . . . .	267
§ 43	ACL sous SOLARIS . . . . .	274
§ 44	ACL sous FREEBSD . . . . .	284
Chapitre 20	Mécanismes de sauvegarde . . . . .	285
§ 1	Technologies de lecteur de bandes . . . . .	287
§ 2	Périphériques UNIX de lecture de bande magnétique . . . . .	292
§ 3	Utilitaires <code>dump</code> , <code>restore</code> . . . . .	293
§ 4	Utilitaire <code>tar</code> . . . . .	300
§ 5	(Windows : : NtBackup) . . . . .	308
§ 6	Quelques règles . . . . .	309
Chapitre 21	Mémoire virtuelle . . . . .	318
§ 1	Principe de la mémoire virtuelle . . . . .	318
§ 2	Affichage de la taille du swap LINUX : <code>free</code> . . . . .	321
§ 3	Affichage de la taille du swap SOLARIS : <code>swap</code> . . . . .	322
§ 4	Activation de partitions de swap LINUX : <code>swapon</code> . . . . .	323
§ 5	Activation de partitions de swap SOLARIS : <code>swap</code> . . . . .	324
§ 6	Activation de partitions de swap au boot LINUX : <code>/etc/fstab</code> . . . . .	325
§ 7	Activation de partitions de swap au boot SOLARIS : <code>/etc/vfstab</code> . . . . .	326
§ 8	Taille du swap . . . . .	327
§ 9	(Windows : : taille du swap) . . . . .	328
§ 10	Fichiers de swap . . . . .	330
§ 11	(Windows : : Fichiers de swap) . . . . .	331
§ 12	Création de fichiers de swap : <code>mkfile</code> . . . . .	332
§ 13	(Windows : : <code>creatfil</code> ) . . . . .	333
§ 14	Création de fichiers de swap (2) : <code>dd</code> . . . . .	334
Chapitre 22	Processus et exécutable . . . . .	335
§ 1	Liste des processus : <code>ps</code> version BSD . . . . .	337
§ 2	Liste des processus : <code>ps</code> version System-V . . . . .	340
§ 3	Liste des processus : <code>top</code> . . . . .	342
§ 4	Contrôle des processus : <code>kill</code> . . . . .	344
§ 5	Priorité des processus : <code>nice</code> / <code>renice</code> . . . . .	348
§ 6	(Windows : : processus) . . . . .	351
§ 7	Temps d'exécution d'un processus : <code>time</code> . . . . .	357
§ 8	Charge du système : <code>uptime</code> . . . . .	359
§ 9	Processus non tuables, processus zombies . . . . .	361
§ 10	Processus zombies (2) . . . . .	362
§ 11	Attribut de fichier exécutable : bit <code>setuid</code> . . . . .	365
§ 12	Attribut de fichier exécutable : bit <code>setgid</code> . . . . .	369
§ 13	Directory <code>/proc</code> . . . . .	373
§ 14	Principes des processus Unix . . . . .	380
§ 15	Principes des threads Unix . . . . .	385
§ 16	Visualisation des threads Unix . . . . .	388

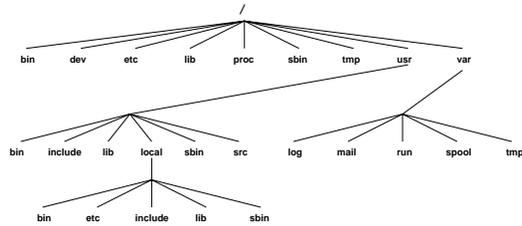
§ 17	Compilation avec threads sous Unix	390
§ 18	Chaîne de compilation sous Unix	391
§ 19	Edition de liens, link : ld	394
§ 20	Edition de liens statique : bibliothèques statiques	398
§ 21	Edition de liens statique : file	399
§ 22	Edition de liens statique : ar	400
§ 23	Edition de liens statique : ranlib	401
§ 24	Edition de liens statique : /sbin	402
§ 25	Edition de liens dynamique : bibliothèques dynamiques	403
§ 26	Edition de liens dynamique : file	404
§ 27	Edition de liens dynamique : ldd	405
§ 28	(Windows : : listdlls)	408
§ 29	Edition de liens dynamique : ldconfig	410
§ 30	Edition de liens dynamique : effacement de librairies dynamiques	412
§ 31	Dynamic Loading	413
§ 32	Représentation interne des symboles : Mangling, nm	420
§ 33	Table de symboles d'un exécutable : strip	422
§ 34	Chaînes de caractères : strings	424
§ 35	Debugging post mortem : core	426
§ 36	Debugging live d'un runtime : trace, strace, truss	433
§ 37	Exécution en environnement restreint non sécurisé : chroot	435
Chapitre 23	Horloges Unix	441
§ 1	Consultation de l'horloge : date	443
§ 2	Commande date améliorée : GNU date	445
§ 3	Réglage de l'horloge : date	446
§ 4	Heure d'été – Heure d'hiver	447
§ 5	Synchronisation d'horloges : NTP (Network Time Protocol)	448
§ 6	(Windows : : NTP)	450
§ 7	Faux ami : commande time	452
Chapitre 24	Programmation de tâches périodiques	454
§ 1	Utilitaire cron	454
§ 2	Fichiers crontab	455
§ 3	Exécution des fichiers crontab	457
§ 4	Edition des fichiers crontab	459
§ 5	Autorisation à utiliser cron, cron.allow, cron.deny	460
§ 6	(Windows : : cron)	461
Chapitre 25	Traces/Journaux système : SYSLOG	467
§ 1	Principes	467
§ 2	Exemples de messages SYSLOG	468
§ 3	Format des messages SYSLOG	469
§ 4	Commande shell logger	471
§ 5	Fichier de configuration : /etc/syslog.conf	472
§ 6	Exemple de fichier /etc/syslog.conf	474
§ 7	Conseils pour /etc/syslog.conf	475
§ 8	Reconfiguration, SIGHUP	477
§ 9	Surveillance des logs	478
§ 10	Autres implémentations de SYSLOG	479
§ 11	(Windows : : SYSLOG)	481
§ 12	Avenir de SYSLOG	483
Chapitre 26	Base de données ORACLE	484
Chapitre 27	Base de données MYSQL	486
§ 1	Utilisateur Unix dédié à MYSQL : mysql	487
§ 2	Arborescence MYSQL	488
§ 3	Création des bases initiales MYSQL : mysql_install_db	489

§ 4	lancement/arrêt de MYSQL . . . . .	492
§ 5	Création de bases MYSQL (1) : <code>mysqladmin</code> . . . . .	493
§ 6	Création de bases MYSQL (2) : commandes SQL . . . . .	494
§ 7	Dialogue interactif avec MYSQL en mode CLI : <code>mysql</code> . . . . .	495
§ 8	Notion d'utilisateurs MYSQL . . . . .	499
§ 9	Fichier de configuration : <code>my.cnf</code> . . . . .	501
§ 10	Sauvegarde : <code>mysqldump</code> . . . . .	502
§ 11	Rechargement d'une sauvegarde . . . . .	505
§ 12	Interface de gestion de MYSQL : <code>phpmyadmin</code> . . . . .	506
Chapitre 28	Base de données POSTGRESQL . . . . .	507
§ 1	Utilisateur Unix dédié à PGSQL : <code>pgsql</code> . . . . .	509
§ 2	Arborescence PGSQL . . . . .	510
§ 3	Création des bases initiales PGSQL : <code>initdb</code> . . . . .	511
§ 4	Lancement/arrêt de PGSQL : <code>pg_ctl</code> . . . . .	512
§ 5	Création de bases PGSQL : <code>createdb</code> . . . . .	513
§ 6	Destruction de bases PGSQL : <code>dropdb</code> . . . . .	514
§ 7	Dialogue interactif avec PGSQL en mode CLI : <code>psql</code> . . . . .	515
§ 8	Création de bases PGSQL : commandes SQL . . . . .	518
§ 9	Destruction de bases PGSQL : commandes SQL . . . . .	519
§ 10	Notion d'utilisateurs PGSQL . . . . .	520
§ 11	Gestion d'utilisateurs PGSQL (1) : <code>createuser</code> . . . . .	522
§ 12	Gestion d'utilisateurs PGSQL (2) : <code>dropuser</code> . . . . .	524
§ 13	Fichier de configuration : <code>postgresql.conf</code> . . . . .	526
§ 14	Fichier de configuration : <code>pg_hba.conf</code> . . . . .	527
§ 15	Sauvegarde : <code>pg_dump</code> , <code>pg_dumpall</code> . . . . .	529
§ 16	Rechargement d'une sauvegarde . . . . .	532
§ 17	Interface de gestion de PGSQL : <code>phppgadmin</code> . . . . .	533
Chapitre 29	Langage SQL . . . . .	534
§ 1	Définition des exemples . . . . .	535
§ 2	Description des données . . . . .	537
§ 3	Manipulation des données . . . . .	539
§ 4	Contrôle des accès . . . . .	540
§ 5	Projection, Restriction . . . . .	541
§ 6	Les requêtes imbriquées . . . . .	549
§ 7	La jointure . . . . .	551
§ 8	L'union . . . . .	553
§ 9	L'intersection . . . . .	554
§ 10	La différence . . . . .	555
§ 11	La division . . . . .	556
§ 12	Group by . . . . .	558
§ 13	Group by . . . . . HAVING . . . . .	559
§ 14	Conclusion . . . . .	560
§ 15	Correction des exercices . . . . .	561
Chapitre 30	Consolidation d'OS . . . . .	573
§ 1	VMWARE . . . . .	574

## Chapitre 15 : Contenu d'une arborescence type d'un système Unix

### § 15.1 Les difficultés pratiques

Exemple :



Pourquoi ce chapitre sur l'arborescence ?

1. Pour connaître le contenu des principaux directories
2. Pour connaître l'utilité des principaux directories
3. Pour savoir configurer les PATH (utilisateurs + root)
4. Pour surveiller les directories système pouvant grossir en raison de fichiers log.

Longtemps la hiérarchie a été non standard et très dépendante de chaque OS.

⇒ Difficulté de portage de shell scripts et autres applications.

⇒ Difficulté de gestion en environnement hétérogène.

Exemple :

FreeBSD 2.2.x	/usr/sbin/chown
HP-UX 9.07	/bin/chown
SunOS 4.1.4	/usr/etc/chown

Une certaine convergence apparaît actuellement (effet des différents standards Posix and co ? ou raréfaction des Unix ?).

**Solution du lien symbolique**

Ne pas oublier la commande "**ln -s fichier1 fichier2**".

IRIX 5.3	/usr/bsd
SunOS 4.1.4	/usr/ucb

On fera par exemple sur la machine SGI :

```
% ln -s /usr/bsd /usr/ucb
% ls -l /usr
drwxr-xr-x  2 root    sys      512 Aug 28  1996 bsd/
lrwxr-xr-x  1 root    sys        3 Aug 22  1996 ucb -> bsd/
% echo $PATH
/usr/local/bin:/bin:/usr/ucb:/usr/bin:/usr/sbin:/etc:/usr/etc
% type rlogin
rlogin is /usr/ucb/rlogin
```

**§ 15.2 Tentative de normalisation FHS Linux**

Se reporter au site <http://www.pathname.com/fhs/>

Document FHS version 2.3 : <http://www.pathname.com/fhs/pub/fhs-2.3.pdf>

Cf Annexe du volume 2. **Lecture indispensable !**

### § 15.3 Panorama d'un système Unix moderne type

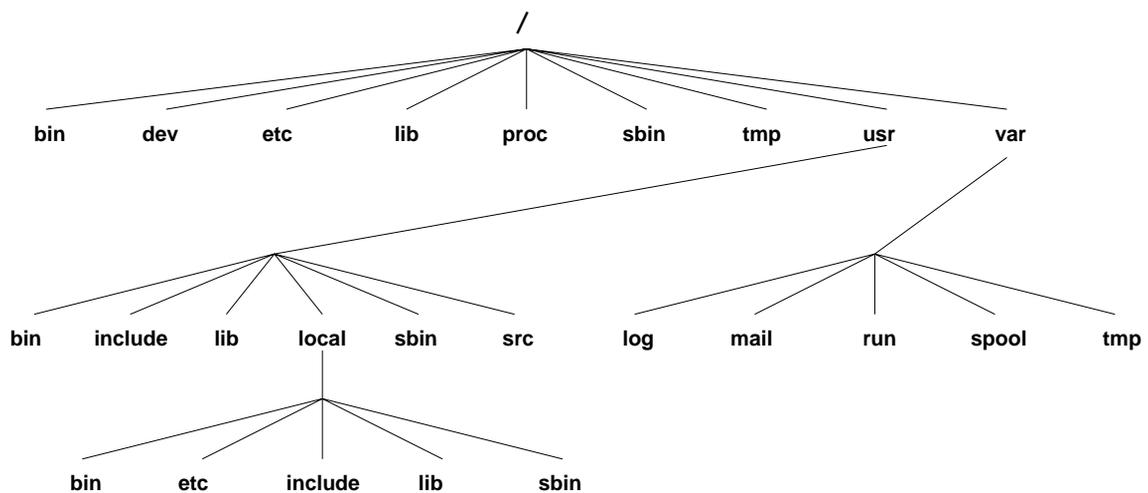
Quelques notions importantes :

- Répertoires système / Répertoires utilisateurs
  - ⇒ différences de PATH
- Répertoires partageables / Répertoires non partageables
  - ⇒ stations diskless
- Répertoires figés / Répertoires évoluant dans le temps
  - ⇒ partitions en read-only, partitions sur CDROM

Possibilité de mixer ces caractéristiques.

Exemple :

	Répertoires partageables	Répertoires non partageables
Répertoires figés	/usr, /opt	/etc, /boot
Répertoires évolutifs	/var/mail, /var/spool/lpd	/var/run, /var/lock



## § 15.4 Directory /sbin

/sbin  $\equiv$  **system binaries**

Il contient les binaires système primordiaux utilisés pendant le boot de la machine.

(historiquement, ces binaires étaient dans /etc)

Il réside dans la même partition que / ce qui garantit sa disponibilité à ce moment-là où d'autres partitions ne sont pas encore disponibles.

Exemple non exhaustif de commandes :

arp, dump, fsck, halt, ifconfig, init, ldconfig, mkfs, restore, shutdown, swapon, etc.

## § 15.5 Directory /bin

/bin  $\equiv$  **binaries**

Il contient les binaires primordiaux utilisés pendant le boot de la machine.

Il réside dans la même partition que / ce qui garantit sa disponibilité à ce moment-là où d'autres partitions ne sont pas encore disponibles.

Exemple non exhaustif de commandes :

bash, chmod, chown, cp, date, dd, df, echo, hostname, kill, mkdir, mknod, more, mv, ps, pwd, rm, rmdir, sed, umount, etc.

**§ 15.6 Directory /usr/sbin**

/usr/sbin ≡ **user level system binaries**

Il contient des binaires système (d'où le 's' dans son nom) non primordiaux et plus utilisés pour la gestion au jour le jour de la machine.

Il réside dans une même partition pouvant ne pas être disponible pendant les premières étapes du boot.

Exemple non exhaustif de commandes :

adduser, in.rlogind, inetd, lpd, lpc, pwck, sa, showmount, traceroute, vipw, etc.

**§ 15.7 Directory /usr/bin**

/usr/bin ≡ **user level binaries**

Il contient la plupart des commandes Unix pour utilisateurs (plusieurs centaines en général).

Exemple non exhaustif de commandes :

date, grep, ln, perl, sed, vi, who, etc.

### § 15.8 Directory /usr/local/sbin

/usr/local/sbin  $\equiv$  **local user level system binaries**

Il contient les commandes Unix d'administration propres au site ou propres à la machine.

### § 15.9 Directory /usr/local/bin

/usr/local/bin  $\equiv$  **local user level binaries**

Il contient la plupart des commandes Unix utilisateurs installées localement.

**§ 15.10 Directory /lib**

/lib ≡ **libraries**

Il contient en général les librairies dynamiques primordiales pour le fonctionnement du système.

**ATTENTION : ne pas effacer les librairies dynamiques de ce répertoire, sinon AIE !**

En général, on n'y trouve pas de librairie statique .a.

Exemple non exhaustif de fichiers :

ld.so, libc.so.5, libm.so.5, etc.

**§ 15.11 Directory /usr/lib**

/usr/lib ≡ **user level libraries**

Il contient les librairies dynamiques non primordiales au fonctionnement du système ainsi que les librairies statiques '.a' de programmation en langage C.

Linker avec -ltoto revient à linker avec /usr/lib/libtoto.a  
(sauf option spécifique à cc)

Exemple non exhaustif de fichiers :

libm.a, libm.so, libndbm.a, libtermcap.a, etc.

**§ 15.12 Directory `/usr/local/lib`**

`/usr/local/lib`  $\equiv$  **local user level libraries**

Il contient les librairies dynamiques ou statiques installées localement.

**§ 15.13 Directory `/usr/include`**

`/usr/include`  $\equiv$  **user level include files**

Il contient les fichiers '.h' de programmation en langage C.

`<foo.h>` fait référence à « `/usr/include/foo.h` »  
(sauf option de compilation explicite à `cc`)

**§ 15.14 Directory /usr/local/include**

/usr/local/include ≡ **local user level include files**

Il contient les fichiers '.h' installés localement.

<foo.h> fera référence à « /usr/local/include/foo.h » via une option à cc du genre « -I/usr/local/include ».

**§ 15.15 Directory /etc**

/etc ≡ **system level config files and etc.**

Il contient les fichiers de configuration des divers utilitaires primordiaux assurant le fonctionnement du système.

Exemple non exhaustif de fichiers :

group, passwd, motd, hosts, hosts.allow, hosts.deny, fstab, inittab, printcap, resolv.conf, syslog.conf, etc.

**§ 15.16 Directory /usr/local/etc**

/usr/local/etc ≡ **local system level config files and etc.**

Il contient les fichiers de configuration de divers utilitaires non nécessaires au fonctionnement du système.

**§ 15.17 Directory /dev**

/dev ≡ **system devices**

Il contient les fichiers spéciaux du système permettant d'accéder aux ressources physiques de la machine comme les disques durs, la mémoire RAM etc.

2 approches historiques :

- ancienne approche : les fichiers spéciaux sont générés manuellement via les scripts /dev/MAKEDEV et /dev/MAKEDEV.local
- approche moderne : les fichiers spéciaux sont générés automatiquement (faire en plus boot -r sur SOLARIS).

Pas de /usr/dev ni /usr/local/dev!

**Les droits d'accès aux fichiers spéciaux dans /dev sont très importants.**

Une précaution : conserver une trace écrite d'un "ls -l" de /dev

Quelle est l'organisation du répertoire /dev ?

- organisation en vrac (FreeBSD, LINUX)
- organisation structurée (SOLARIS, HP-UX, etc.) en sous répertoires

Exemple non exhaustif de SOLARIS :

Répertoire	Contenu
/dev/rdisk	disques en mode caractère
/dev/dsk	disques en mode bloc
/dev/rmt	bandes magnétiques
/dev/pty	pseudo-terminaux esclaves
/dev/ptym	pseudo-terminaux maîtres

Plusieurs fichiers communs à tous les Unix :

/dev/audio	le device de la carte audio de la machine
/dev/console	le device de la console texte de la machine
/dev/kmem	mémoire du noyau Unix
/dev/log	canal des messages système
/dev/mem	mémoire de la machine
/dev/null	trou noir
/dev/tty	le device virtuel de votre terminal texte
/dev/zero	générateur de caractères de code ASCII 0

**§ 15.18 Trou noir pour redirection : /dev/null**

On peut vouloir se débarrasser d'une partie de l'affichage.

Solution inefficace :

```
% application > /tmp/toto
...
% rm /tmp/toto
```

La solution est de rediriger vers /dev/null :

```
% application > /dev/null
...
```

C'est un fichier spécial (logique car dans /dev...) :

```
crw-rw-rw-  1 root    sys      13,  2 Oct 19 01:13 /dev/null
```

**/dev/null est indispensable dans la vie de l'administrateur système.**

**§ 15.19 (Windows : : NUL)**

Exemple :

```
C:\DOCUME~1\besancon\MESDOC~1>application.exe 2>NUL
...
```

## § 15.20 Trou noir pour redirection : `/dev/console`

La console texte de la machine est accessible via le device `/dev/console`



Classiquement, les messages importants s'affichent sur la console.

(sur un poste LINUX en mode X Windows, faire `Ctrl Alt F1` pour passer sur la console en mode texte)

## § 15.21 Directory `/proc`

`/proc`  $\equiv$  **system process**

Il contient une représentation sous forme de fichiers virtuels et directories virtuels d'informations de la machine Unix.

Utilité ?

$\Rightarrow$  pour le programmeur système qui utilisera les appels standard (et non pas des appels spécifiques comme la `libkvm` ou des `ioctl()` spécifiques)

Implémentation de `/proc` propre à chaque Unix.

Représentation virtuelle de :

- informations sur les processus
- informations sur le hardware
- informations sur le noyau

◇ Exemple : informations sur les processus

```
% echo $$
23247
% cd /proc/23247
% ls -l
total 0
-r--r--r--  1 besancon sysadmin    0 Sep 22 14:30 cmdline
lrwx-----  1 besancon sysadmin   64 Sep 22 15:45 cwd -> [0001]:1523515394
-r-----   1 besancon sysadmin    0 Sep 22 15:45 environ
lrwx-----  1 besancon sysadmin   64 Sep 22 15:45 exe -> [0801]:10251
dr-x-----  1 besancon sysadmin    0 Sep 22 15:45 fd
pr--r--r--  1 besancon sysadmin    0 Sep 22 15:45 maps
-rw-----  1 besancon sysadmin    0 Sep 22 15:45 mem
lrwx-----  1 besancon sysadmin   64 Sep 22 15:45 root -> [0801]:2
-r--r--r--  1 besancon sysadmin    0 Sep 22 14:29 stat
-r--r--r--  1 besancon sysadmin    0 Sep 22 15:45 statm
-r--r--r--  1 besancon sysadmin    0 Sep 22 15:45 status
```

◇ Exemple : informations sur le hardware

```
% ls -l /proc/pci
-r--r--r--  1 root      root          0 Nov  2 18:52 /proc/pci

% cat pci
PCI devices found:
  Bus 0, device 0, function 0:
    Host bridge: VIA Technologies, Inc. VT82C693A/694x [Apollo PRO133x] (rev 196).
    Prefetchable 32 bit memory at 0xd0000000 [0xd3ffffff].
  Bus 0, device 1, function 0:
    PCI bridge: VIA Technologies, Inc. VT82C598/694x [Apollo MVP3/Pro133x AGP] (rev 0).
    Master Capable. No bursts. Min Gnt=12.
...

```

```
% cat /proc/cpuinfo
processor       : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 8
model name    : Pentium III (Coppermine)
stepping      : 10
cpu MHz       : 1002.214
cache size    : 256 KB
Physical processor ID : 0
Number of siblings : 1
fdiv_bug     : no
hlt_bug      : no
f00f_bug     : no
coma_bug     : no
fpu          : yes
fpu_exception : yes
cpuid level  : 2
wp           : yes
flags        : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 mmx fxsr sse
bogomips     : 1989.22
...
```

#### ◇ Exemple : informations sur le noyau et sa version

```
% cat /proc/version
Linux version 2.4.18-27.7.xsmp (bhcompile@stripples.devel.redhat.com)
(gcc version 2.96 20000731 (Red Hat Linux 7.3 2.96-112)) #1 SMP Fri
Mar 14 05:52:30 EST 2003
```

#### ◇ Exemple : informations sur le uptime du noyau

```
% cat /proc/uptime
810311.03 800120.37
```

#### ◇ Exemple : informations sur des statistiques du noyau sur l'emploi de la machine

```
% cat /proc/stat
cpu 1276780 2094 1821586 158965456
cpu0 681135 1171 903015 79447636
cpu1 595645 923 918571 79517818
page 19636553 18178406
swap 3965 4959
intr 460335474 414888745 32 0 4 4 3635073 6 0 1 0 35855460 1382047 897324 0 11 3676767 0 0 0 0 0 0 0 0 0
disk_io: (3,0):(3,3,24,0,0) (22,2):(695266,373120,8363128,322146,6352160) (22,3):(2514420,999712,3090945
ctxt 233821067
btime 1066985101
processes 201147
```

◇ Exemple : informations sur la table ARP du noyau

```
% cat /proc/net/arp
IP address      HW type    Flags      HW address    Mask        Device
134.157.46.254  0x1       0x2       00:02:7E:21:F7:9C  *          eth0
192.168.4.18   0x1       0x2       00:48:54:6B:E5:B0  *          eth3
```

◇ Exemple : informations sur les sockets unix du noyau

```
% cat /proc/net/unix
Num      RefCount Protocol Flags      Type St Inode Path
db7f5580: 00000002 00000000 00010000 0001 01  9782 /tmp/.font-unix/fs7100
e9fcc580: 00000002 00000000 00010000 0001 01  9940 /tmp/.X11-unix/X0
d3e23a80: 00000002 00000000 00010000 0001 01  8308 /var/lib/mysql/mysql.sock
ce1dd080: 00000002 00000000 00010000 0001 01  9600 /dev/gpmctl
d75f5080: 00000014 00000000 00000000 0002 01  5050 /dev/log
cb91e080: 00000002 00000000 00000000 0002 01 1088373
d4f46080: 00000002 00000000 00000000 0002 01 1087696
f679a580: 00000002 00000000 00000000 0002 01 1085719
f1204080: 00000002 00000000 00000000 0002 01 1057186
```

◇ Exemple : informations sur les utilisations de la mémoire par le noyau

```
% cat /proc/meminfo
      total:      used:      free:  shared: buffers:  cached:
Mem:  1055293440 1040719872 14573568          0 107249664 692142080
Swap: 4294901760 37588992 4257312768
MemTotal:      1030560 kB
MemFree:        14232 kB
MemShared:         0 kB
Buffers:        104736 kB
Cached:         654592 kB
SwapCached:     21328 kB
Active:         547860 kB
Inact_dirty:    197696 kB
Inact_clean:    119996 kB
Inact_target:   173108 kB
HighTotal:     131008 kB
HighFree:       1024 kB
LowTotal:      899552 kB
LowFree:       13208 kB
SwapTotal:     4194240 kB
SwapFree:      4157532 kB
Committed_AS:  281672 kB
```

**§ 15.22 Directory /tmp**

(en anglais *temporary*)

Le répertoire /tmp sert à stocker des fichiers temporaires.

C'est l'équivalent de C:\windows\temp sur Microsoft Windows.

Ses droits d'accès :

```
drwxrwxrwt 12 root sys 2648 Sep 28 13:02 /tmp/
```

c'est-à-dire 1777 exprimé en octal :

- signification de 777 : tout le monde sur la machine peut créer, modifier, effacer des fichiers
- signification de 1000 : un utilisateur ne peut effacer que les fichiers qui lui appartiennent

En principe, /tmp est vidé au boot de la machine Unix.

**§ 15.23 Directory /var/log**

(en anglais *variable log files*)

Il stocke différents fichiers de bord du système.

Par exemple :

```
total 1900
-rw----- 1 root sys 0 Oct 29 2002 authlog
-rw-r--r-- 1 root other 294141 Nov 2 18:50 ppp.log
-rw-r--r-- 1 root root 844 Aug 28 13:06 skipd.log
-rw-r--r-- 1 root root 844 Aug 28 12:55 skipd.log.0
-rw-r--r-- 1 root root 844 Aug 28 12:52 skipd.log.1
-rw-r--r-- 1 root root 844 Aug 28 12:45 skipd.log.2
-rw-r--r-- 1 root root 844 Aug 28 12:41 skipd.log.3
-rw-r--r-- 1 root other 254 Oct 29 2002 sysidconfig.log
-rw-r--r-- 1 root sys 57297 Nov 2 18:50 syslog
-rw-r--r-- 1 root sys 74519 Oct 31 03:10 syslog.0
-rw-r--r-- 1 root sys 85749 Oct 24 03:10 syslog.1
-rw-r--r-- 1 root sys 79963 Oct 17 03:10 syslog.2
-rw-r--r-- 1 root sys 69391 Oct 10 03:10 syslog.3
-rw-r--r-- 1 root sys 85748 Oct 3 03:10 syslog.4
-rw-r--r-- 1 root sys 69120 Sep 26 03:10 syslog.5
-rw-r--r-- 1 root sys 74101 Sep 19 03:10 syslog.6
-rw-r--r-- 1 root sys 63515 Sep 12 03:10 syslog.7
```

**§ 15.24 Directory /var/mail**

(en anglais *variable mailbox files*)

Il stocke les boîtes aux lettres de courrier électronique.

Par exemple :

```
total 2
-rw-rw----  1 besancon mail          0 Dec  4 2002 besancon
```

Parfois, on rencontre encore /var/spool/mail.

**§ 15.25 Directory /var/run**

(en anglais *variable run time data*)

Il contient des informations sur l'état du système après le dernier boot.

⇒ nettoyer au moment du boot les anciens fichiers

En général, on y trouve des fichiers de la forme « /var/run/démon.pid » qui stocke le PID de l'application « démon ».

Intérêt ?

Plutôt que faire :

```
% ps -ax | grep démon
% kill -TERM PID-de-démon-trouvé-ci-dessus
```

on fait :

```
% kill -TERM `cat /var/run/démon.pid`
```

**§ 15.26 Directory /var/spool**

(en anglais *variable spool data*)

Il stocke des sous répertoires dédiés à des applications consommatrices de nombreux fichiers temporaires comme par exemple le système d'impression ou la messagerie électronique.

Pour le système d'impression, c'est « /var/spool/lpd ».

Pour la messagerie électronique, c'est « /var/spool/mqueue ».

**§ 15.27 Directory /var/tmp**

(en anglais *variable temporary*)

Le répertoire /var/tmp sert à stocker des fichiers temporaires.

Différence avec /tmp : /var/tmp n'est pas vidé au boot de la machine Unix.

Ses droits d'accès :

```
drwxrwxrwt 12 root sys 2648 Sep 28 13:02 /var/tmp
```

c'est-à-dire 1777 exprimé en octal :

- signification de 777 : tout le monde sur la machine peut créer, modifier, effacer des fichiers
- signification de 1000 : un utilisateur ne peut effacer que les fichiers qui lui appartiennent

**§ 15.28 Directory /mnt**

/mnt  $\equiv$  **mount point**

Point de montage pour rendre disponible provisoirement une partition (locale ou réseau)

**/mnt est un répertoire.**

Tendance LINUX :

- /mnt/floppy
- /mnt/cdrom
- etc.

**§ 15.29 Directory /opt**

/usr/local  $\equiv$  **local add-ons**

/opt  $\equiv$  **local add-on applications software**

/usr/local contient les ajouts locaux de commandes.

/opt contient les ajouts locaux de « paquets » complets de commandes (càd nombreuses commandes exécutables et fichiers annexes)

/opt existe par exemple sur SOLARIS

Le logiciel *application* s'installera dans /opt/application qui contiendra :

- /opt/application/bin
- /opt/application/include
- /opt/application/lib
- /opt/application/man

Les fichiers de config vont en principe dans /etc/opt/.

**§ 15.30 Directory /usr/src**

Quand il existe, ce directory contient en principe les sources du système Unix sur lequel on le trouve.  
Utile au développeur système.

Présent sur les unix libres.

Non présent sur les Unix commerciaux du type SOLARIS, AIX, HP-UX, etc.

**§ 15.31 Directory /sys**

Plus généralement, on trouve /usr/sys. Ce directory contient les fichiers modules '.o' nécessaires à recompiler uniquement un noyau Unix.

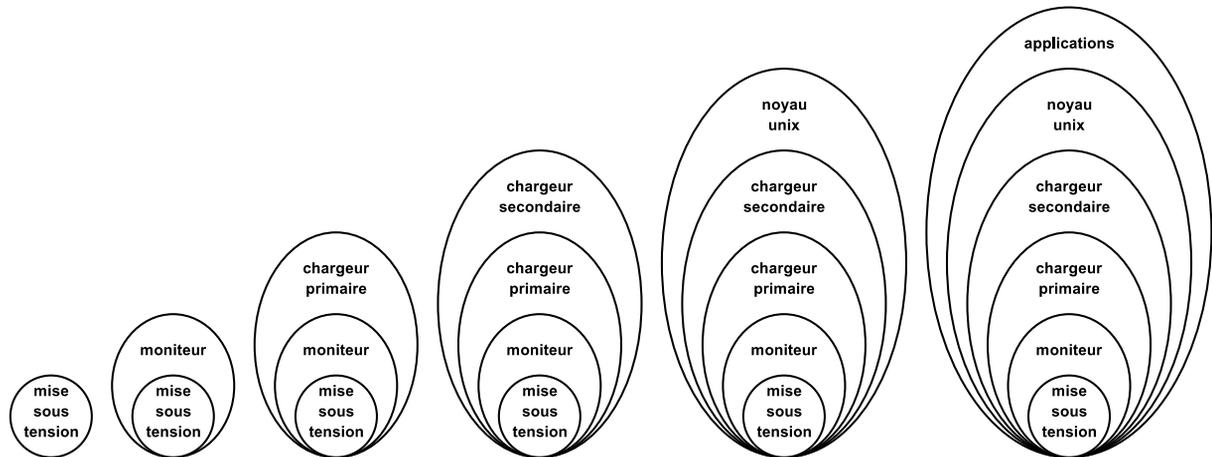
Présent sur les Unix commerciaux type SOLARIS, AIX, HP-UX, etc.

Non présent sur les unix libres.

## Chapitre 16 : Démarrage d'une station Unix

Objectif : arriver à un système 100% fonctionnel à partir de rien

Méthode : par étapes successives faisant boule de neige



### § 16.1 Etape 1 du boot : le moniteur

Le moniteur peut être :

- en ROM
- en PROM
- en EEPROM ⇒ flashable à volonté (par exemple SUN, HP, BIOS des PC)



Sauvegarde de la configuration du moniteur dans une NVRAM (*Non Volatile RAM*) ou EEPROM.

Rôles du moniteur :

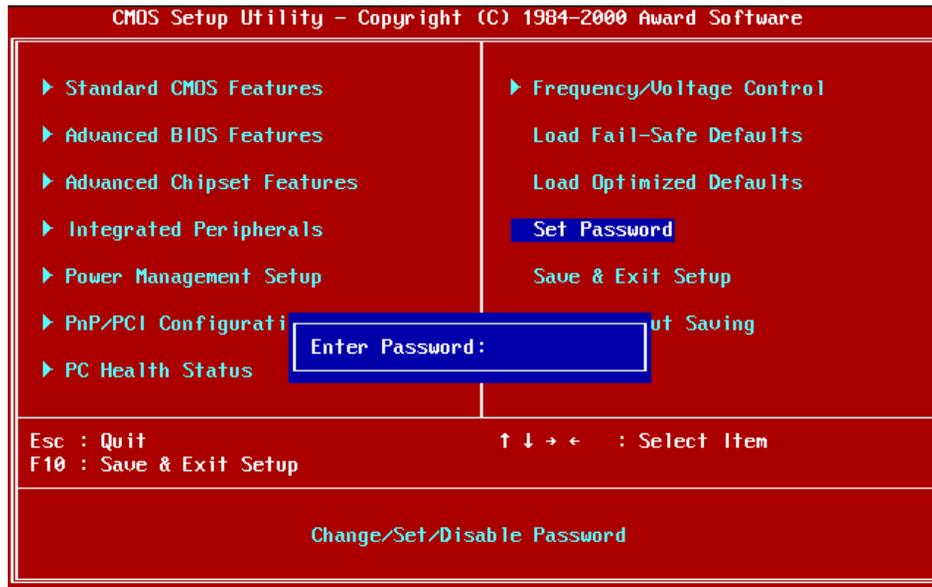
- inspecter l'état matériel de la machine
- permettre à l'utilisateur d'interagir si besoin pour divers réglages :
  - précision du disque de boot
  - consultation d'adresses mémoire
  - modification de valeurs dans EEPROM ou NVRAM
  - recherche des périphériques sur le bus SCSI
  - options de boot (single ou multi)
  - commandes de tests hardware
- **charger en mémoire vive le secteur de boot d'un certain périphérique et exécuter ce programme.**

Problèmes possibles :

- si le moniteur ne fonctionne pas ⇒ panne matérielle
- le moniteur donne l'accès à la sélection du disque de boot
  - ⇒ certains constructeurs permettent de protéger le moniteur par mot de passe
- si perte du mot de passe du moniteur ⇒ retour usine, à moins d'un super mot de passe secret connu du constructeur
- le fonctionnement du moniteur est particulier à chaque constructeur ; plus ou moins riche selon chaque constructeur

Pour le programme moniteur des stations SUN, se reporter à

[http://sunsolve.sun.com/handbook\\_pub/General/OBP.html](http://sunsolve.sun.com/handbook_pub/General/OBP.html)



BIOS ≡ moniteur des plateformes PC

Illustration de la possibilité de mettre un mot de passe

## § 16.2 Etape 2 du boot : le chargeur primaire

(*boot block* en anglais)

Le chargeur primaire est constitué en général du secteur de boot d'un périphérique (en général un disque dur).

Il est chargé en mémoire vive par le moniteur puis exécuté.

Il va lui même charger et exécuter un programme plus gros qui poursuivra le démarrage de la machine.

Ce secteur de boot rend le disque stratégique.

A noter : Le chargement en mémoire vive du secteur de boot se fait en faisant appel uniquement aux firmwares des périphériques. Aucun pilote logiciel (*device driver*) n'entre en jeu.

La création de ce secteur de boot est un mécanisme propre à chaque constructeur.

Par exemple sur Solaris :

```
# installboot /usr/platform/`uname -i`/lib/fs/ufs/bootblk /dev/rdsk/c0t3d0s0
```

Complexité sur LINUX : le chargeur primaire et le chargeur secondaire sont assurés par le même programme.

On parle de MBR (*Master Boot Record*).

Deux programmes principalement :

- LILO (*Linux Loader*)
- GRUB (*Grand Unified Boot Loader*)

Copie d'écran de GRUB :



### § 16.3 Etape 3 du boot : le chargeur secondaire

Le chargeur secondaire est le programme chargé depuis le secteur de boot puis exécuté.

Il a une connaissance partielle de la structure d'un disque Unix de façon à y trouver certains fichiers.

**Il charge en mémoire vive le noyau Unix** et plus si nécessaire (par exemple sur Solaris, il chargera `/kernel/unix` puis les modules de `/kernel` jusqu'à ce que le noyau soit capable de monter par lui même des filesystems) puis l'exécute.

A noter : Le chargement en mémoire vive du chargeur secondaire se fait en faisant appel uniquement aux firmwares des périphériques. Aucun pilote logiciel (*device driver*) n'entre en jeu.

### § 16.4 Etape 4 du boot : le chargement du noyau Unix

Le noyau est chargé puis exécuté par le chargeur secondaire.

Pas de convention en ce qui concerne le nom du noyau :

Système	Nom du noyau
FreeBSD	<code>/kernel</code>
Linux	<code>/boot/bzImage</code>
SunOS	<code>/vmunix</code>
Solaris 5.0 à 5.4	<code>/kernel/unix</code>
Solaris $\geq$ 5.5	<code>/kernel/genunix</code> combiné à <code>/platform/'uname -i'/kernel/unix</code>

Le noyau Unix permet la configuration du système :

- prise en compte des périphériques
- configuration logicielle des interfaces réseau
- création de toutes les structures mémoire nécessaires
- lancement des deux premiers processus de la machine, PID 0 et 1. Ce ne sont pas des processus Unix "traditionnels" puisqu'ils sont construits à partir de rien (par opposition aux processus ultérieurs qui auront tous un parent).

Les messages du noyau sont consultables via la commande `dmesg` :

```
% dmesg
Intel MultiProcessor Specification v1.4
  Virtual Wire compatibility mode.
OEM ID: OEM00000 Product ID: PROD00000000 APIC at: 0xFEE00000
Processor #0 Pentium(tm) Pro APIC version 17
Processor #1 Pentium(tm) Pro APIC version 17
I/O APIC #2 Version 17 at 0xFEC00000.
Enabling APIC mode: Flat.          Using 1 I/O APICs
Processors: 2
Kernel command line: ro root=/dev/hdd3
Initializing CPU#0
Detected 1002.276 MHz processor.
Console: colour VGA+ 80x25
Calibrating delay loop... 1998.84 BogoMIPS
Memory: 1027216k/1048512k available (1294k kernel code, 17716k reserved,
1080k data, 144k init, 131008k highmem)
...
hda: QUANTUM FIREBALLP AS20.5, ATA DISK drive
hdb: HITACHI DVD-ROM GD-8000, ATAPI CD/DVD-ROM drive
...
eth0: Digital DS21143 Tulip rev 65 at 0xf8a96000, 00:80:C8:C9:83:F9, IRQ 5.
...
```

Processus initiaux sur LINUX :

```
% ps -ax
  PID TT  STAT   TIME COMMAND
    0 ?   D      0:52 swapper
    1 ?   IW     0:00 /sbin/init -
    2 ?   D      0:00 pagedaemon
...

```

Processus initiaux sur SOLARIS :

```
% ps -edf
  UID  PID  PPID  C   STIME TTY      TIME CMD
  root   0    0  0   Oct 10 ?       0:18 sched
  root   1    0  0   Oct 10 ?       0:31 /etc/init -
  root   2    0  0   Oct 10 ?       0:01 pageout
  root   3    0  0   Oct 10 ?      29:51 fsflush
...

```

## § 16.5 Etape 5 du boot : init et ses scripts de démarrage

Le programme `init` a traditionnellement le PID 1.

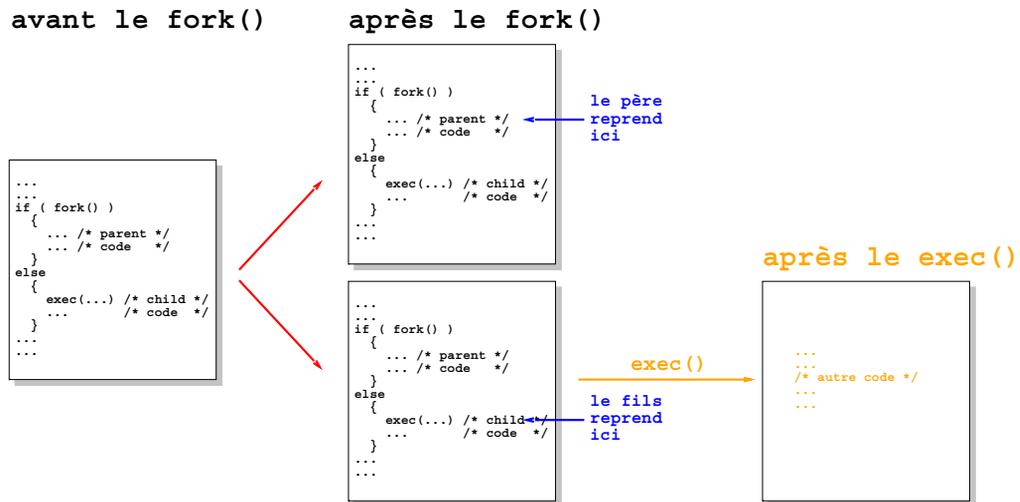
Le programme `init` assure le reste du démarrage de la machine Unix via le lancement de scripts de démarrage personnalisés et la surveillance des ports série.

Deux questions se posent :

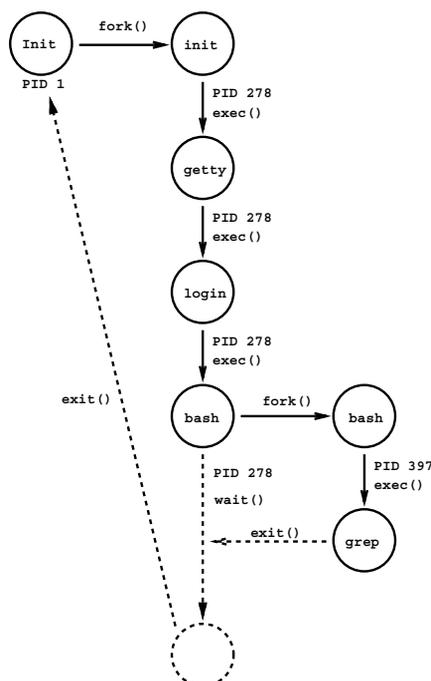
- Comment créer des processus supplémentaires ?
- Comment organiser les scripts de démarrage ?

◇ Comment créer des processus supplémentaires ?

Tout processus Unix est créé par un processus parent exécutant un `fork()`, c'est-à-dire une copie de lui même (code + data), suivi d'un `exec()` pour remplacer le contenu de cette copie par un nouveau programme.



Mécanisme de `fork()` et `exec()` appliqué à `init` :



◇ Comment organiser les scripts de démarrage ?

Il y a deux méthodes pour lancer les scripts de démarrage :

- la méthode System-V
- la méthode BSD

**§ 16.6 Scripts de démarrage System-V**

Par exemple AIX, DEC OSF1, HP-UX, IRIX, SOLARIS, LINUX, FreeBSD 5.x

**Ce mécanisme est reconnaissable à la présence du fichier `/etc/inittab`.**

Exemple de fichier `/etc/inittab` SOLARIS (fichier partiel) :

```
fs::sysinit:/etc/rcS >/dev/console 2<>/dev/console </dev/console
is:3:initdefault:
s0:0:wait:/etc/rc0 >/dev/console 2<>/dev/console </dev/console
s2:23:wait:/etc/rc2 >/dev/console 2<>/dev/console </dev/console
s3:3:wait:/etc/rc3 >/dev/console 2<>/dev/console </dev/console
s5:5:wait:/etc/rc5 >/dev/console 2<>/dev/console </dev/console
s6:6:wait:/etc/rc6 >/dev/console 2<>/dev/console </dev/console
```

◇ Syntaxe de /etc/inittab

Ensemble de lignes au format :

```
label : niveaux : action : commande
```

Le label est une étiquette quelconque. En général un terme adapté à ce que l'on fait.

◇ Niveaux

On distingue des niveaux ou *run-levels*.

Pour connaître le niveau dans lequel on se trouve :

– sur LINUX :

```
% /sbin/runlevel
```

```
N 5
```

– sur SOLARIS (ici le 7 Mars est la date du dernier boot) :

```
% who -r
```

```
.          run-level 3  Mar  7 22:29      3      0  S
```

Signification des niveaux sur SOLARIS :

SVR4 Run States	
S	Single-user (leaves filesystems mounted)
0	Power off
1	Single-user/System-admin (leaves only / mounted)
2	Multi-user, network disabled
3	Multi-user, network enabled
4	(not used)
5	PROM Monitor level
6	Halt & reboot to default state

#### ◇ Principales actions

##### – respawn

le programme init devra relancer la commande du champ commande chaque fois qu'elle se terminera.

##### – wait

Lorsque init rentre dans le niveau considéré, il doit lancer la commande du champ commande et en attendre la fin avant de passer à la ligne suivante de inittab.

##### – once

Si la commande n'a pas été lancée au préalable, init la lance et passe à la ligne suivante ; en aucun cas, cette commande ne sera relancée.

##### – initdefault

Cette ligne précise le run-level dans lequel init se place au début de son invocation. Il ne peut y avoir qu'une ligne de ce type.

◇ Structure classique de `/etc/inittab`

La structure classique (LINUX, SOLARIS, etc.) est la suivante (exemple extrait de SOLARIS) :

```
fs::sysinit:/etc/rcS >/dev/console 2<>/dev/console </dev/console
is:3:initdefault:
s0:0:wait:/etc/rc0 >/dev/console 2<>/dev/console </dev/console
s2:23:wait:/etc/rc2 >/dev/console 2<>/dev/console </dev/console
s3:3:wait:/etc/rc3 >/dev/console 2<>/dev/console </dev/console
s5:5:wait:/etc/rc5 >/dev/console 2<>/dev/console </dev/console
s6:6:wait:/etc/rc6 >/dev/console 2<>/dev/console </dev/console
```

On trouve parfois aussi les fichiers `/sbin/rc0`, `/sbin/rc1`, `/sbin/rc2`, `/sbin/rc3`, etc.

Ce sont des scripts. Tous ces scripts sont bâtis sur le même modèle.

Prenons le cas du script `/etc/rc0`. Sa structure est la suivante :

```
#!/sbin/sh
if [ -d /etc/rc0.d ]
then
    for f in /etc/rc0.d/K*
    {
        if [ -s ${f} ]
        then
            /sbin/sh ${f} stop
        fi
    }

    for f in /etc/rc0.d/S*
    {
        if [ -s ${f} ]
        then
            /sbin/sh ${f} start
        fi
    }
fi
```

Le principe :

1. Quand on entre dans le runlevel N on lance le script rcN.
2. Le script rcN lance des programmes dont le nom commence par la lettre K et situés dans /etc/rcN.d.

L'enchaînement de ces programmes se fait dans l'ordre alphabétique.

Un argument est passé à chaque programme : stop (cf le K pour Kill)

3. Le script rcN lance des programmes dont le nom commence par la lettre S et situés dans /etc/rcN.d.

L'enchaînement de ces programmes se fait dans l'ordre alphabétique.

Un argument est passé à chaque programme : start (cf le S pour Start)

Exemple de script S :

```
#!/sbin/sh

state=$1
set `who -r`
if [ $8 != "0" ]
then
    exit
fi
case $state in
'start')
    if [ $9 = "2" -o $9 = "3" ]
    then
        exit
    fi
    echo "Starting process accounting"
    /usr/lib/acct/startup
    ;;
'stop')
    echo "Stopping process accounting"
    /usr/lib/acct/shutacct
    ;;
esac
```

Exemple de script K :

```
#!/sbin/sh

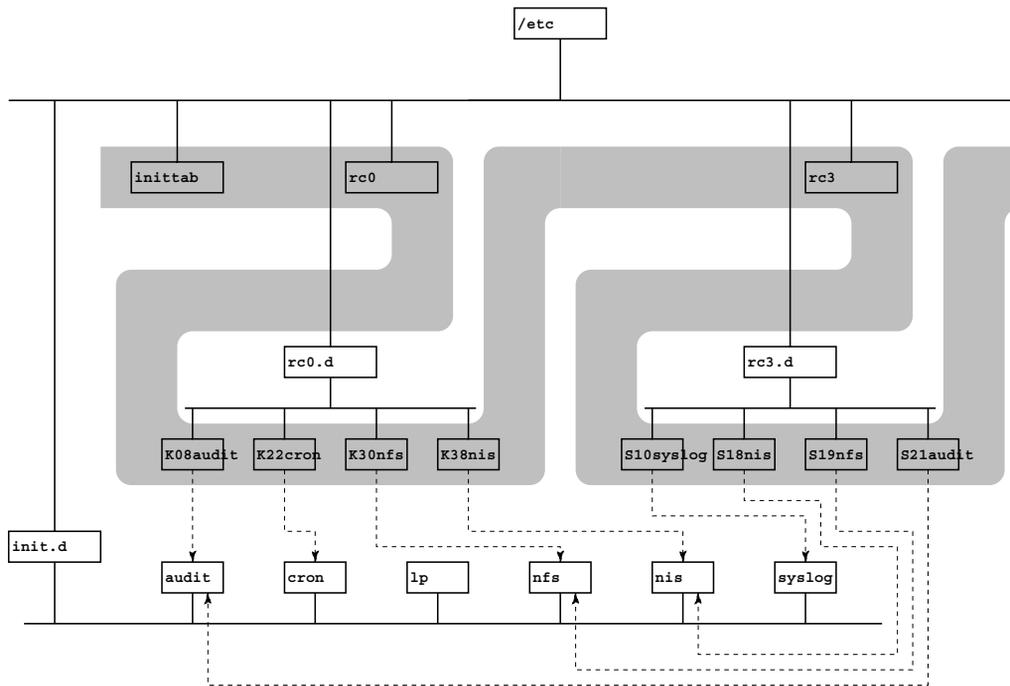
state=$1
set `who -r`
if [ $8 != "0" ]
then
    exit
fi
case $state in
'start')
    if [ $9 = "2" -o $9 = "3" ]
    then
        exit
    fi
    echo "Starting process accounting"
    /usr/lib/acct/startup
    ;;
'stop')
    echo "Stopping process accounting"
    /usr/lib/acct/shutacct
    ;;
esac
```

Moralité : les fichiers contiennent la même chose ; ils sont en fait identiques !

```
% ls -l *spc
ls -l /etc/rc2.d/*spc
lrwxrwxrwx 1 besancon      13 Nov 12 00:44 /etc/rc2.d/K20spc -> ../init.d/spc
lrwxrwxrwx 1 besancon      13 Nov 12 00:44 /etc/rc2.d/S80spc -> ../init.d/spc
```

**Via le mécanisme des liens symboliques (ou de liens hard), on regroupe au sein d'un même fichier la partie démarrage et la partie arrêt.**

Sur linux, on a affaire à `/etc/rc.d/init.d` et aux répertoires `/etc/rc.d/rc[0-9].d` mais le principe décrit précédemment reste le même.



### ◇ Comment personnaliser le démarrage de la machine ?

La meilleure façon de faire :

1. écrire un script qui sera placé dans `/etc/init.d`

Par exemple :

```
-rwxr-xr-x 1 root sys 355 Jul 25 1996 /etc/init.d/ars*
```

2. mettre dans le répertoire du bon niveau un lien symbolique vers le script précédent

Par exemple :

```
lrwxr-xr-x 1 root sys 13 Jul 25 1996 /etc/rc2.d/S99ars -> ../init.d/ars*
```

La moins bonne façon de faire :

1. écrire un script placé quelque part

Par exemple

```
-rwxr-xr-x 1 root sys 355 Jul 25 1996 /etc/rc.local
```

2. ajouter dans `/etc/inittab` une ligne du type Par exemple :

```
local:2:wait:sh /etc/rc.local > /dev/console 2>&1
```

**§ 16.7 Gestion des scripts de démarrage LINUX : *chkconfig***

On peut gérer à la main les scripts de démarrage vus précédemment :

- détermination des runlevels
- mise en place des liens symboliques
- mise à jour des liens symboliques

C'est une méthode assez simple.

Sous LINUX, on peut aussi gérer les scripts via la commande *chkconfig* (inspirée de la commande *chkconfig* de l'Unix IRIX de SGI).

◇ Structure du shell script nécessaire à *chkconfig*

Le script de démarrage comprend des lignes supplémentaires de commentaires :

```
#!/bin/bash
#
# syslog          Starts syslogd/klogd.
#
# chkconfig: 2345 12 88
# description: Syslog is the facility by which many daemons use to log \
# messages to various system log files.  It is a good idea to always \
# run syslog.
...
```

Lecture :

- « 2345 » : script activé dans les runlevels 2, 3, 4 et 5
- « 12 » : démarré en « 12 ième » position ; on parle de priorité de démarrage
- « 88 » : arrêté en « 88 ième » position ; on parle de priorité d'arrêt

Tous les exemples qui suivent reposent sur le script ci-dessus pour *syslog*.

◇ Runlevels et liens symboliques

La chaîne « 2345 » est interprétée correctement :

```
# ls -l /etc/rc[0-9].d/*syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc0.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc1.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:13 /etc/rc2.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:13 /etc/rc3.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:13 /etc/rc4.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:13 /etc/rc5.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc6.d/K88syslog -> ../init.d/syslog

# chkconfig --list syslog
syslog          0:off 1:off 2:on 3:on 4:on 5:on 6:off
```

◇ Désactivation d'un script dans un certain runlevel

Syntaxe : `chkconfig --level N name off`

```
# ls -l /etc/rc[0-9].d/*syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc0.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc1.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc2.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc3.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc4.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc5.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc6.d/K88syslog -> ../init.d/syslog

# chkconfig --level 2 syslog off

# ls -l /etc/rc[0-9].d/*syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc0.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc1.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc2.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc3.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc4.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc5.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc6.d/K88syslog -> ../init.d/syslog
```

◇ Désactivation d'un script dans tous les runlevels

Syntaxe : `chkconfig name off`

```
# ls -l /etc/rc[0-9].d/*syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc0.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc1.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc2.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc3.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc4.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc5.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc6.d/K88syslog -> ../init.d/syslog
```

```
# chkconfig syslog off
```

```
# ls -l /etc/rc[0-9].d/*syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc0.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc1.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc2.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc3.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc4.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:16 /etc/rc5.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Feb 13 09:57 /etc/rc6.d/K88syslog -> ../init.d/syslog
```

◇ Installation initiale/Réinstallation des liens symboliques

Syntaxe : `chkconfig name reset`

```
# rm /etc/rc[0-9].d/*syslog
# ls -l /etc/rc[0-9].d/*syslog
ls: /etc/rc[0-9].d/*syslog: No such file or directory
```

```
# chkconfig syslog reset
```

```
# ls -l /etc/rc[0-9].d/*syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:57 /etc/rc0.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:57 /etc/rc1.d/K88syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:57 /etc/rc2.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:57 /etc/rc3.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:57 /etc/rc4.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:57 /etc/rc5.d/S12syslog -> ../init.d/syslog
lrwxrwxrwx 1 root root 16 Jul 22 13:57 /etc/rc6.d/K88syslog -> ../init.d/syslog
```

◇ chkconfig et processus

**La commande `chkconfig` n'agit pas sur des processus qui tourneraient déjà. Elle agit sur les scripts de démarrage de fonctionnalités.**

```
# ps ax | grep syslog
 884 ?          S          0:39 syslogd -m 0

# chkconfig syslog off

# ps ax | grep syslog
 884 ?          S          0:39 syslogd -m 0
```

**§ 16.8 Scripts de démarrage BSD**

(par exemple DEC ULTRIX, SunOS, FreeBSD, etc)

`init` n'a pas de notion de niveaux, d'états.

Le shell script `/etc/rc` à **caractère général** est appelé par le programme `init`.

Les particularités de la station sont regroupées dans le script `/etc/rc.local` appelé par `/etc/rc`.

Ce mécanisme tombe en désuétude car trop fouillis.

FreeBSD 5.x.y est en voie d'adoption du mécanisme des scripts de démarrage de System-V.

## § 16.9 Boot en single user/mono utilisateur

C'est un mode particulier de démarrage de la machine Unix :

- la machine Unix ne fonctionne pas encore en mode multi utilisateurs
- les fonctionnalités réseau ne sont pas encore actives

Utilité : intervenir sur la machine Unix soit pour réparer un problème soit pour installer une partie logicielle qu'il est impossible de faire ultérieurement (changement de librairie dynamique par exemple).

Pour booter en single user, il faut le préciser au niveau du moniteur de la machine.

On passe du mode mono utilisateur au mode multi utilisateur via `exit` qui fait se terminer le shell root du mode single user.

Exemple : boot en single user sur LINUX avec GRUB



```
GRUB version 0.90 (637K lower / 64512K upper memory)

root (hd0,0)
kernel /boot/vmlinuz-2.4.18-17.7.x ro root=/dev/hda1
```

Use the ↑ and ↓ keys to select which entry is highlighted.  
Press 'b' to boot, 'e' to edit the selected command in the  
boot sequence, 'c' for a command-line, 'o' to open a new line  
after ('O' for before) the selected line, 'd' to remove the  
selected line, or escape to go back to the main menu.



redhat.

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

grub edit> kernel /boot/vmlinuz-2.4.18-17.7.x ro root=/dev/hda1 single
```



redhat.



Exemple : boot en single user sur SOLARIS

Taper «boot -s»

Il faudra entrer le mot de passe de l'utilisateur root pour entrer en mode mono utilisateur sinon on restera en mode boot multi utilisateur.

Sur Sun, il faudra certainement régler le type de terminal (pour pouvoir faire «vi» and co) :

```
TERM=sun
```

```
export TERM
```

## Chapitre 17 : Arrêt d'une station Unix

Théorie : il faut suivre une procédure d'arrêt du fait de l'aspect multiutilisateur :

1. Prévenir les utilisateurs
2. Signaler aux processus de se terminer proprement
3. Passer en single user
4. Flusher les disques avant d'éteindre (via la commande `sync`).

La pratique pourra être différente. . .

### § 17.1 Mécanisme APM

(en anglais *Advanced Power Management*)

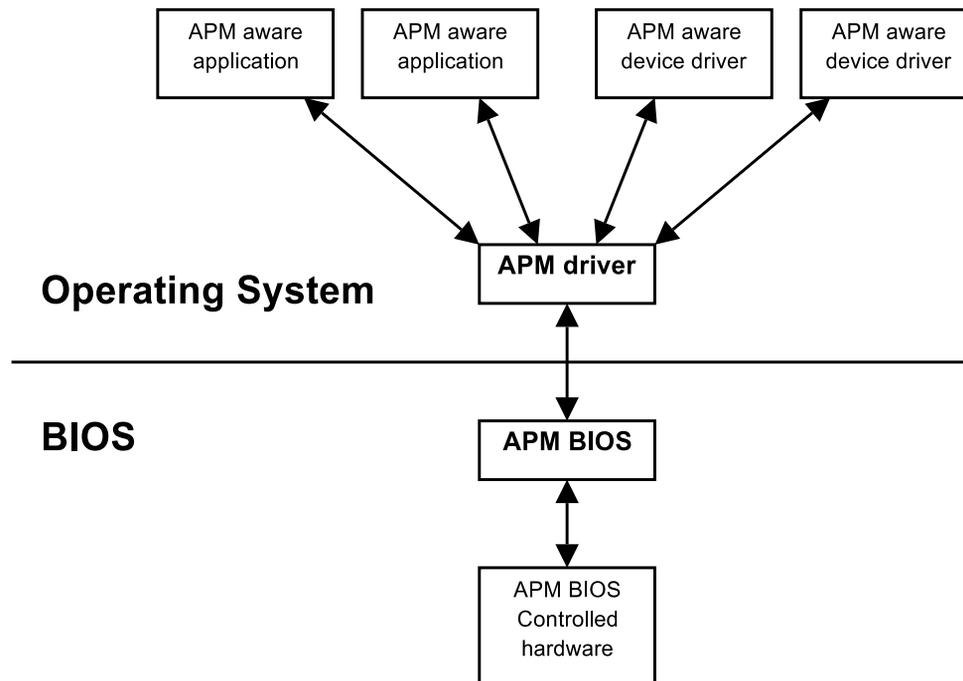
Date de 1996 ; version 1.2 ; consortium Intel, Microsoft

Principalement pour hardware de type PC

Obsolète ; remplacé par ACPI, cf chapitre suivant

Un Unix intégrant APM pourra s'éteindre.

Principe :



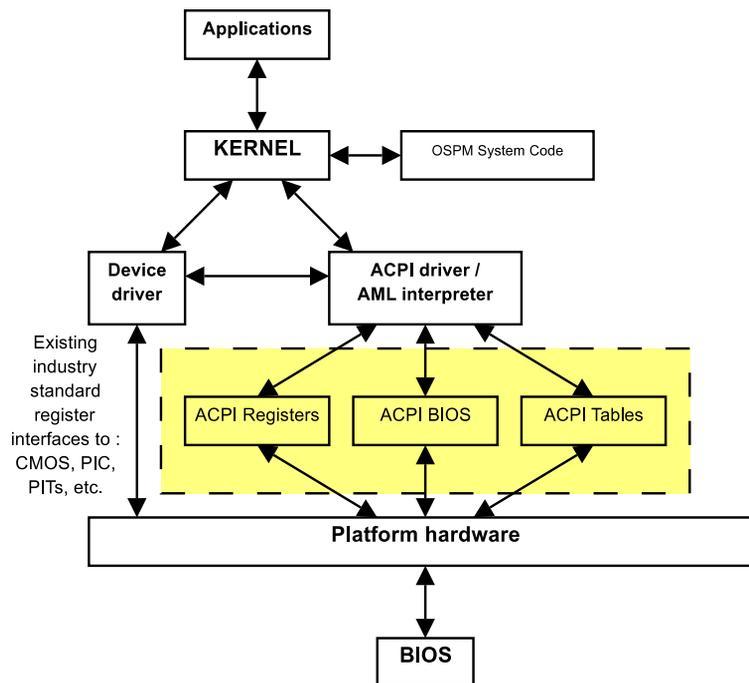
## § 17.2 Mécanisme ACPI

(en anglais *Advanced Configuration and Power Interface*)

Date de 1999 ; version 2.0c ; consortium Intel, Toshiba, Microsoft

Un Unix intégrant ACPI pourra s'éteindre.

Principe radicalement différent de APM : **tout se passe au niveau du système d'exploitation**



### § 17.3 Commande d'arrêt : halt

Réalise un `sync` avant d'arrêter les processeurs et de rendre la main au mode moniteur en général.

```

*** FINAL System shutdown message

System going down IMMEDIATELY

System shutdown time has arrived

sync'ing disks (14 buffers to flush): 13 1
0 buffers not flushed
0 buffers still dirty

System has halted
OK to turn off power or reset system
UNLESS "WAIT for UPS to turn off power" mes
  
```

**§ 17.4 Commande d'arrêt : *fasthalt***

Disponible sur SUN SOLARIS et les Unix BSD.

La commande crée un fichier `/fastboot` évitant de réaliser un `fsck` au démarrage de la machine.

```
#!/bin/sh
# @(#)fasthalt.sh 1.6 88/02/07 SMI; from UCB 4.2
PATH=/bin:/usr/bin:/usr/etc:$PATH
export PATH
cp /dev/null /fastboot
halt $*
```

**§ 17.5 Commande d'arrêt : *poweroff***

Disponible sur SUN SOLARIS.

Commande de plus en plus répandue.

Arrêt électrique de la machine.

**§ 17.6 Commande de redémarrage : *reboot***

Provoque un redémarrage du système.

**§ 17.7 Commande de redémarrage : *fastboot***

Disponible sur SUN SOLARIS et les Unix BSD.

La commande crée un fichier `/fastboot` évitant de réaliser un `fsck` au démarrage de la machine.

```
#!/bin/sh
# @(#)fastboot.sh 1.7 88/02/07 SMI; from UCB 4.2
PATH=/bin:/usr/bin:/usr/etc:$PATH
export PATH
cp /dev/null /fastboot
reboot "$@"
```

**§ 17.8 Commande d'arrêt et de redémarrage : *shutdown***

2 syntaxes selon l'architecture :

1. AT&T : `shutdown -g<délai> -i<état>`
2. BSD : `shutdown [-h|-r] +<délai en secondes> <message>`

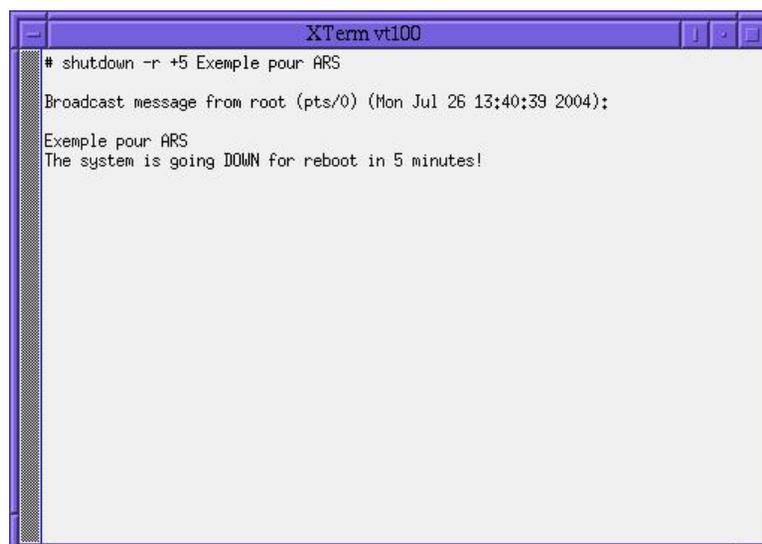
Formes les plus utilisées :

Pour redémarrer la machine : `shutdown -r now` (option `-r` comme *reboot*)

Pour arrêter la machine : `shutdown -h now` (option `-h` comme *halt*)

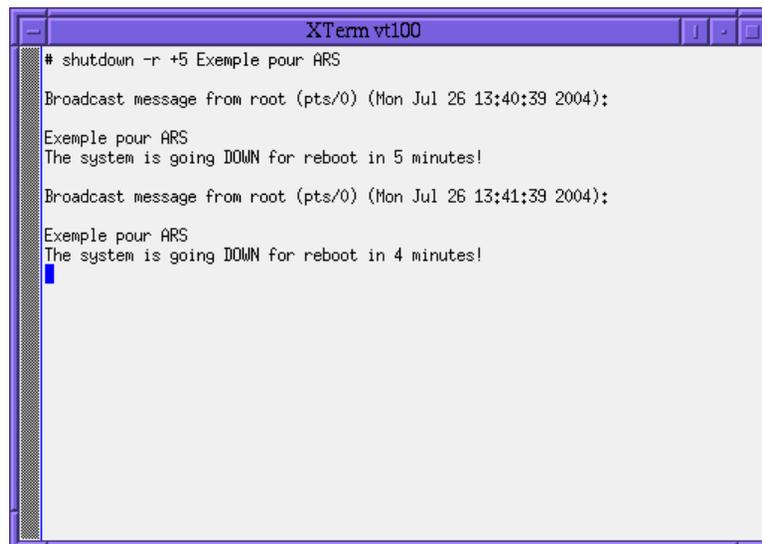
Pour arrêter la machine : `shutdown -p now` (option `-p` comme *poweroff*)

Commande de reboot dans 5 minutes :



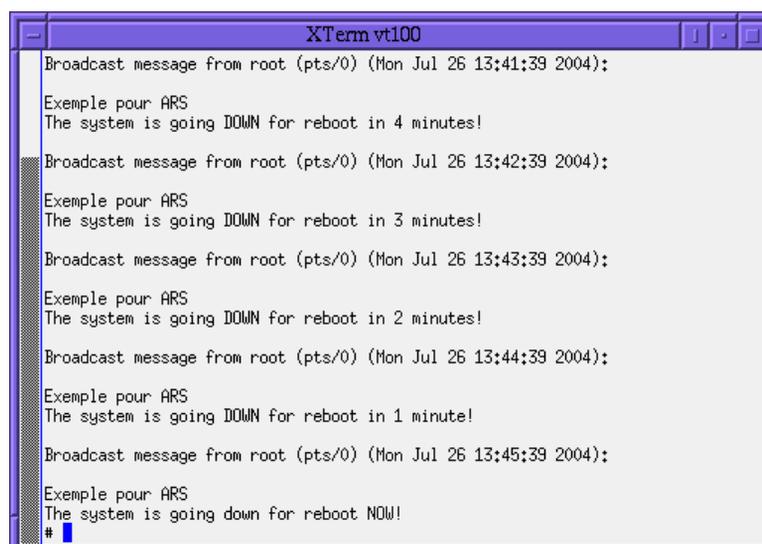
```
XTerm vt100
# shutdown -r +5 Exemple pour ARS
Broadcast message from root (pts/0) (Mon Jul 26 13:40:39 2004):
Exemple pour ARS
The system is going DOWN for reboot in 5 minutes!
```

Annonce régulièrement envoyée pour rappeler le reboot :



```
XTerm vt100
# shutdown -r +5 Exemple pour ARS
Broadcast message from root (pts/0) (Mon Jul 26 13:40:39 2004):
Exemple pour ARS
The system is going DOWN for reboot in 5 minutes!
Broadcast message from root (pts/0) (Mon Jul 26 13:41:39 2004):
Exemple pour ARS
The system is going DOWN for reboot in 4 minutes!
█
```

Moment du reboot arrivé : la machine ne sera plus joignable le temps de son redémarrage :



```
XTerm vt100
Broadcast message from root (pts/0) (Mon Jul 26 13:41:39 2004):
Exemple pour ARS
The system is going DOWN for reboot in 4 minutes!
Broadcast message from root (pts/0) (Mon Jul 26 13:42:39 2004):
Exemple pour ARS
The system is going DOWN for reboot in 3 minutes!
Broadcast message from root (pts/0) (Mon Jul 26 13:43:39 2004):
Exemple pour ARS
The system is going DOWN for reboot in 2 minutes!
Broadcast message from root (pts/0) (Mon Jul 26 13:44:39 2004):
Exemple pour ARS
The system is going DOWN for reboot in 1 minute!
Broadcast message from root (pts/0) (Mon Jul 26 13:45:39 2004):
Exemple pour ARS
The system is going down for reboot NOW!
# █
```

**§ 17.9 Commande d'arrêt et de redémarrage : `kill -TERM 1`**

PID 1  $\equiv$  init

Passage du système de multi utilisateur en mono utilisateur

**§ 17.10 Informations de connexions : `last`**

Les opérations de redémarrage ou arrêt laissent des traces visibles par la commande `last` :

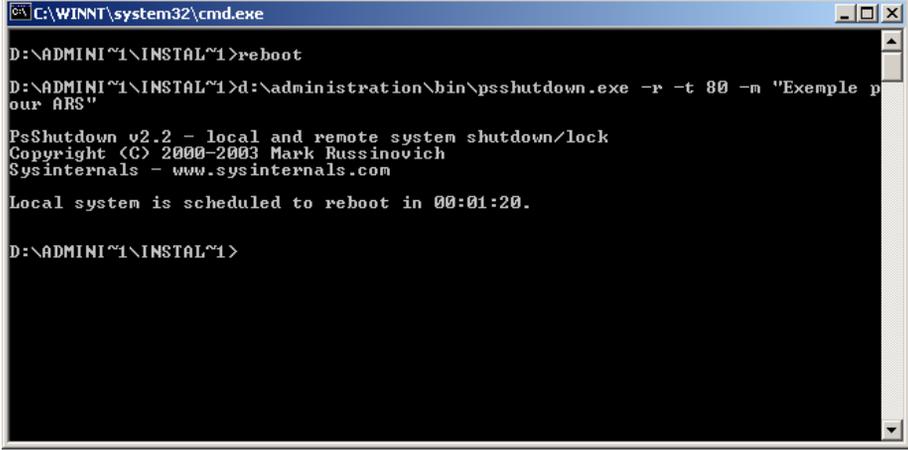
```
# last
...
besancon ftp          dell.example.org Sun Mar 14 14:54 - 14:56 (00:02)
besancon console                    Sun Mar  7 22:30  still logged in
reboot  system boot                    Sun Mar  7 22:29
besancon console                    Sun Mar  7 21:57 - down (00:31)
reboot  system boot                    Sun Mar  7 21:53
besancon ftp          dell.example.org Tue Mar  2 00:48 - 00:49 (00:00)
besancon console                    Sat Feb 28 23:24 - 21:43 (7+22:18)
root    console                    Sat Feb 28 23:23 - 23:24 (00:00)
reboot  system boot                    Sat Feb 28 23:22
besancon ftp          dell.example.org Tue Feb 24 23:20 - 23:21 (00:01)
besancon ftp          dell.example.org Mon Feb 23 00:28 - 00:28 (00:00)
...
```

**§ 17.11 (Windows : : commandes d'arrêt et de redémarrage)**

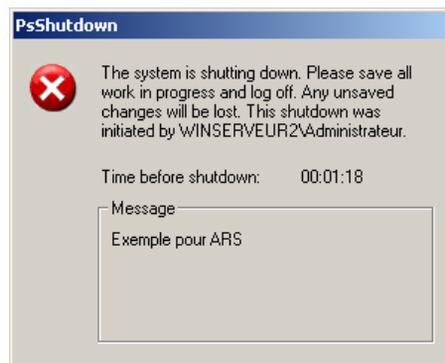
Comment programmer le redémarrage ou l'arrêt de machines Windows ?

Plusieurs programmes disponibles :

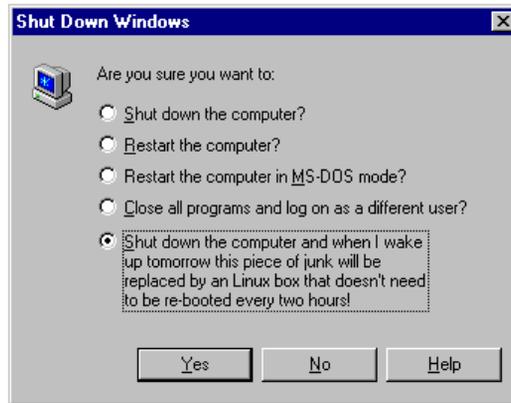
- programme shutdown.exe du Resource Kit Windows 2000
- programme psshutdown.exe de <http://www.sysinternals.com>



```
C:\WINNT\system32\cmd.exe
D:\ADMINI~1\INSTAL~1>reboot
D:\ADMINI~1\INSTAL~1>d:\administration\bin\psshutdown.exe -r -t 80 -m "Exemple p
our ARS"
PsShutdown v2.2 - local and remote system shutdown/lock
Copyright (C) 2000-2003 Mark Russinovich
Sysinternals - www.sysinternals.com
Local system is scheduled to reboot in 00:01:20.
D:\ADMINI~1\INSTAL~1>
```



Humour :



## Chapitre 18 : Le noyau Unix et les périphériques

Le noyau :

- un ensemble de fonctions, de procédures, de structures de données réalisant un certain nombre de tâches
- un exécutable lancé pendant le boot
- un exécutable contenu dans / (en général)

Les responsabilités du noyau :

- fournir une interface d'utilisation et de gestion efficace
- gérer le CPU
- savoir piloter les différents types de périphériques
- gérer les processus (création des processus, ordonnancement, accounting)
- gérer la mémoire (allocation aux processus, swap, paging)
- gérer le système de fichiers (accès locaux, accès réseau, quotas)
- gérer les autres ressources (entrées/sorties, réseau etc.)

Les tâches de l'administrateur concernant le noyau :

- assurer que le noyau du système fonctionne correctement
- assurer que le noyau est configuré pour représenter correctement le système
- modifier le noyau
- compiler le noyau

## § 18.1 Assurer la validité du noyau

Le boot de la machine Unix se compose des étapes :

1. Mode moniteur de la machine
2. Chargeur primaire
3. Chargeur secondaire
4. Chargement du noyau – création du processus `init`
5. Scripts de démarrage lancés par `init`

Si le noyau est incorrect (raisons diverses), le boot ne peut pas se réaliser.

⇒ **Toujours avoir en réserve un noyau que l'on sait être correct.**

**En pratique, avoir au moins le noyau de départ fourni par le constructeur.**

## § 18.2 Configurer le noyau

But : refléter l'état actuel du système.

En particulier :

- De quels drivers a-t-on besoin ?  
Éliminer les drivers non nécessaires ⇒ gain de place mémoire pour d'autres choses.
- Configurer les tailles de diverses structures mémoires (appel à `smi t` sur AIX, `sam` sur HP-UX, `ndd` sur Solaris).

La configuration du noyau est propre à chaque constructeur.

Chez certains constructeurs, c'est plus ou moins interactif (LINUX, DigitalUnix).

**§ 18.3 Modifier le noyau**

On peut modifier le noyau pour :

- incorporer des correctifs (alias "patches");
- incorporer de nouvelles fonctionnalités (par exemple ajouter un pilote de graveur de CD-ROM) ;  
Ne pas oublier de générer d'éventuels fichiers spéciaux après incorporation dans un noyau d'un nouveau driver.
- simplement l'informer de la présence de nouveaux périphériques.

Exemple de patch constructeur pour SunOS 4.1.4 :

Patch Installation Instructions:

```
Become superuser, cp tcx.o to /usr/sys/sun4m/OBJ after first saving  
the old version aside as a precaution.
```

```
% su  
# mv /usr/sys/sun4m/OBJ/tcx.o /usr/sys/sun4m/OBJ/tcx.o.FCS  
# cp sun4m/tcx.o /usr/sys/sun4m/OBJ
```

```
Rebuild the kernel. Refer to the Systems and Network administrations  
manual on how to configure a kernel.
```

```
# cd /usr/sys/sun4m/conf  
# /etc/config GENERIC  
# cd ../GENERIC  
# make
```

```
Preserve the original kernel, copy the new kernel to / and reboot.
```

```
# mv /vmunix /vmunix-  
# cp vmunix /  
# reboot
```

**Exemple de patch constructeur pour Solaris 2.x :**

Patch Installation Instructions:

-----

For Solaris 2.0-2.6 releases, refer to the Install.info file and/or the README within the patch for instructions on using the generic 'installpatch' and 'backoutpatch' scripts provided with each patch.

For Solaris 7-8 releases, refer to the man pages for instructions on using 'patchadd' and 'patchrm' scripts provided with Solaris. Any other special or non-generic installation instructions should be described below as special instructions. The following example installs a patch to a standalone machine:

```
example# patchadd /var/spool/patch/104945-02
```

The following example removes a patch from a standalone system:

```
example# patchrm 104945-02
```

For additional examples please see the appropriate man pages.

C'est de plus en plus automatisé...

**§ 18.4 Compilation du noyau**

C'est un mécanisme propre à chaque constructeur.

Bien sûr on n'oubliera pas la sauvegarde du noyau actuel pour pouvoir faire marche arrière.

A noter que la mise en place d'un nouveau noyau sans rebooter peut conduire à l'arrêt du fonctionnement de certains programmes, par exemple "ps" :

```
# ps -ax
ps: /dev/mem: error on ktextseg: Bad address
ps: could not read kernel VM
```

Différences entre les systèmes à sources publiques et les systèmes constructeurs :

- la compilation d'un noyau implique la compilation de tous ses fichiers sources. La compilation d'un noyau linux est donc une procédure longue (de l'ordre de 6 minutes sur un bi-pentium Pro 200 MHz pour Linux).
- les Unix constructeurs ne fournissent pas les fichiers sources mais par contre ils fournissent les fichier '.o' résultats de la compilation séparée intermédiaire.

La compilation d'un noyau Unix constructeur revient à :

- compilation de quelques fichiers '.c' contenant l'indication des pilotes installés, des tailles mémoire de certaines structures
- édition de liens des '.o' constructeurs et des '.o' locaux

## § 18.5 Compilation du noyau LINUX

1. `cd /usr/src/linux-x.y.z`
2. `make mrproper` pour faire le ménage complet
3. `make config` ou `make menuconfig` ou `make xconfig`
4. `make checkconfig` facultativement
5. `make dep` pour construire les fichiers de dépendance pour la compilation C
6. `make clean` (à faire quasi impérativement)
7. `make bzImage` pour construire une image compressée
8. `make modules` compilera les modules dynamiques du noyau
9. `cp bzImage /boot`
10. `cp System.map /boot`
11. `make modules_install`

12. Editer `/etc/lilo.conf` pour refléter l'ajout d'un nouveau noyau et relancer la commande `lilo` (si utilisation de LILO et pas de GRUB)
13. Rebooter.

## § 18.6 Compilation du noyau FreeBSD

(exemple pour plateforme i386)

1. `cd /usr/src/sys/i386/conf` pour se placer à l'endroit des configurations des noyaux, y compris la configuration par défaut
2. `cp GENERIC MYKERNEL` pour partir de la configuration par défaut
3. `vi MYKERNEL` pour éditer le fichier `MYKERNEL` pour y configurer les aspects utiles du noyau
4. `/usr/bin/config MYKERNEL` pour générer les fichiers C résultats de la configuration
5. `cd ../compile/MYKERNEL` pour se placer dans le répertoire de compilation des sources
6. `make depend`
7. `make` pour compiler le noyau
8. `make install` pour installer le noyau

## § 18.7 Compilation du noyau SOLARIS

Inutile de recompiler le noyau SOLARIS car :

- noyau modulaire à 100%
- variables du noyau modifiables via la commande `ndd` ou le fichier de configuration `/etc/system`

(l'idée à l'origine est la constatation que seul 10% des clients de SUN sont des informaticiens ⇒ simplifier les choses)

## § 18.8 Noyaux modulaires

Le noyau Unix traditionnel est monolithique. Il est malaisé d'y ajouter des drivers, d'y modifier des variables.

La mode est aux modules chargeables dynamiquement dans le noyau. Ils ajoutent alors les nouvelles fonctionnalités au noyau pendant qu'il tourne sans nécessiter d'arrêt.

Exemple sur Linux :

```
% lsmod
Module                Size  Used by
soundcore              2628  0 (autoclean) (unused)
vfat                   9276  0 (autoclean) (unused)
fat                   30400  0 (autoclean) [vfat]
nfs                   28896  3 (autoclean)
lockd                 31496  1 (autoclean) [nfs]
sunrpc                53028  1 (autoclean) [nfs lockd]
ne                     6668  1 (autoclean)
8390                   6072  0 (autoclean) [ne]
```

Exemple sur Solaris où le noyau est complètement modulaire :

```
% modinfo
Id Loadaddr  Size Info Rev Module Name
 5 600ba000  3b30  1  1 specfs (filesystem for specfs)
 7 600e1000  2bc8  1  1 TS (time sharing sched class)
 8 600b0a88   4a4  -  1 TS_DPTBL (Time sharing dispatch table)
 9 600f0000 23538  2  1 ufs (filesystem for ufs)
10 6012a000  dc4b 226  1 rpcmod (RPC syscall)
10 6012a000  dc4b  1  1 rpcmod (rpc interface str mod)
11 60144000 2752b  0  1 ip (IP Streams module)
11 60144000 2752b  3  1 ip (IP Streams device)
12 600e5550  11f3  1  1 rootnex (sun4u root nexus)
13 60037a48   188  57  1 options (options driver)
14 600e4010  14e8  12  1 sad (Streams Administrative driver's)
15 600b0548   538  2  1 pseudo (nexus driver for 'pseudo')
16 6018c000  f059  32  1 sd (SCSI Disk Driver 1.255)
17 601a2000  5f26  -  1 scsi (SCSI Bus Utility Routines)
18 601b8000  ca68  61  1 esp (ESP SCSI HBA Driver 1.250)
...
```

## § 18.9 Modifications dynamiques du noyau LINUX : `sysctl, /etc/sysctl.conf`

Sur LINUX, certaines variables du noyau peuvent être modifiées au boot de la machine ou après sans avoir à recompiler le noyau pour faire cela.

Pour cela, utiliser la commande `sysctl`

Pour connaître les valeurs de certains réglages du noyau, faire `sysctl -a` :

```
% /sbin/sysctl -a | grep ip_forward
net.ipv4.ip_forward = 0
```

Pour régler une variable, utiliser `sysctl -w variable=valeur` :

```
# sysctl -w net.ipv4.ip_forward="1"
net.ipv4.ip_forward = 1
```

Le fichier `/etc/sysctl.conf` peut contenir des réglages à faire au démarrage de la machine. Syntaxe du type `variable=valeur`.

Par exemple sur `serveur.formation.jussieu.fr` :

```
# Enables packet forwarding
net.ipv4.ip_forward = 1
# Enables source route verification
net.ipv4.conf.default.rp_filter = 1
# Disables the magic-sysrq key
kernel.sysrq = 0
```

## § 18.10 Modifications dynamiques du noyau FreeBSD : `sysctl, /etc/sysctl.conf`

Sur FreeBSD, certaines variables du noyau peuvent être modifiées au boot de la machine ou après sans avoir à recompiler le noyau pour faire cela.

Pour cela, utiliser la commande `/sbin/sysctl` :

Pour connaître les valeurs de certains réglages du noyau, faire `sysctl -a` :

```
% sysctl -a | grep forwarding
net.inet.ip.forwarding: 0
net.inet.ip.fastforwarding: 0
net.inet6.ip6.forwarding: 0
```

Pour régler une variable, utiliser `sysctl variable=valeur` :

```
# sysctl net.inet.ip.forwarding="1"
net.inet.ip.forwarding: 0 -> 1
```

Le fichier `/etc/sysctl.conf` peut contenir des réglages à faire au démarrage de la machine. Syntaxe du type `variable=valeur`.

Par exemple :

```
net.inet.ip.forwarding=1
```

## § 18.11 Modifications dynamiques du noyau SOLARIS : *ndd, /etc/system*

Sur SOLARIS, certaines variables du noyau peuvent être modifiées au boot de la machine ou après sans avoir à recompiler le noyau pour faire cela.

Pas de commande `sysctl` à la LINUX ou FreeBSD, par contre...

Pour les paramètres réseau, utiliser la commande `/usr/sbin/ndd`.

Par exemple :

```
# ndd /dev/ip \? | grep forwarding
ip_forwarding          (read and write)
ip6_forwarding         (read and write)
lo0:ip_forwarding     (read and write)
eri0:ip_forwarding    (read and write)
```

```
# ndd /dev/ip ip_forwarding
0
```

Pour régler un certain paramètre, on procède ainsi :

```
# ndd -set /dev/ip ip_forwarding 0
```

Au niveau du fichier `/etc/system`, on peut régler certains paramètres du noyau qui seront pris en compte au démarrage.

Par exemple pour rendre les piles des processus non exécutables :

```
set noexec_user_stack = 1
set noexec_user_stack_log = 1
```

Pour connaître comment paramétrer certains aspects du noyau SOLARIS, se reporter au manuel

*Solaris Tunable Parameters Reference Manual* :

<http://docs-pdf.sun.com/817-1759/817-1759.pdf>

## § 18.12 Messages du noyau : dmesg

(en anglais *Diagnostic Messages*)

Syntaxe : dmesg

```
% dmesg
Intel MultiProcessor Specification v1.4
  Virtual Wire compatibility mode.
OEM ID: OEM00000 Product ID: PROD00000000 APIC at: 0xFEE00000
Processor #0 Pentium(tm) Pro APIC version 17
Processor #1 Pentium(tm) Pro APIC version 17
I/O APIC #2 Version 17 at 0xFEC00000.
Enabling APIC mode: Flat. Using 1 I/O APICs
Processors: 2
Kernel command line: ro root=/dev/hdd3
Initializing CPU#0
Detected 1002.276 MHz processor.
Console: colour VGA+ 80x25
Calibrating delay loop... 1998.84 BogoMIPS
Memory: 1027216k/1048512k available (1294k kernel code, 17716k reserved,
1080k data, 144k init, 131008k highmem)
...
```

## § 18.13 Le noyau Unix et les périphériques

Point clé d'Unix : **abstraction d'un périphérique sous la forme de fichier**

⇒ tout se ramène à des fonctions du langage C d'accès à des fichiers

⇒ interface unifiée de programmation

### § 18.14 Types d'objets Unix

- fichiers réguliers (test -f toto)
 

```
-rw-r--r-- 1 besancon software 940 Sep 21 14:23 cours.tex
```
- directories (test -d toto)
 

```
drwxr-sr-x 17 besancon software 1536 Sep 21 14:33 .
```
- objets périphériques blocs (test -b toto) ou objets périphériques caractères (test -c toto) : ils font référence à des périphériques
 

```
crw-r----- 1 root operator 13, 0 Jun 10 13:27 /dev/rhd0a
brw-r----- 1 root operator 4, 0 Jun 10 13:27 /dev/sd0a
```
- pipes nommés ou *FIFO* (test -p toto)
 

```
prw-r--r-- 1 besancon software 0 Sep 21 14:38 essai
```
- sockets (test -S toto)
 

```
srwxrwxrwx 1 root bin 0 Sep 4 17:21 /tmp/.X11-unix/X0
```
- liens symboliques (test -L toto)
 

```
lrwxrwxrwx 1 root wheel 23 Jun 10 13:24 /etc/termcap@ -> \
/usr/share/misc/termcap
```

### § 18.15 Répertoire /dev

(en anglais *device*)

/dev est le lieu de dépôt des fichiers spéciaux.

Traditionnellement, le script /dev/MAKEDEV crée les fichiers spéciaux nécessaires par défaut au système (boot -r sur Solaris).

**Les droits d'accès aux fichiers spéciaux dans /dev sont très importants.**

Une précaution : conserver une trace écrite d'un "ls -l" de /dev

Quel est le contenu du répertoire /dev ?

Dans le cas de machines de type System-V (comme Solaris), /dev contient des sous répertoires :

Répertoire	Contenu
/dev/rdsk	disques en mode caractère
/dev/dsk	disques en mode bloc
/dev/rmt	bandes magnétiques
/dev/pty	pseudo-terminaux esclaves
/dev/ptym	pseudo-terminaux maîtres

Les machines BSD mettent tout en vrac.

Les machines LINUX ont une position intermédiaire :

- moitié organisée
- moitié en vrac

```
# du -k /dev
0    /dev/pts
4    /dev/ataraid
32   /dev/cciss
4    /dev/compaq
4    /dev/cpu/0
4    /dev/cpu/1
...
4    /dev/cpu/9
68   /dev/cpu
4    /dev/dri
32   /dev/i2o
32   /dev/ida
4    /dev/inet
4    /dev/input
12   /dev/logicalco
4    /dev/net
4    /dev/raw
68   /dev/rd
4    /dev/scramdisk
0    /dev/shm
4    /dev/usb
4    /dev/video
4    /dev/video1394
4    /dev/snd
4    /dev/cpqhealth
436  /dev
```

## § 18.16 Mode bloc

Reconnaissable lors d'un `ls -l` :

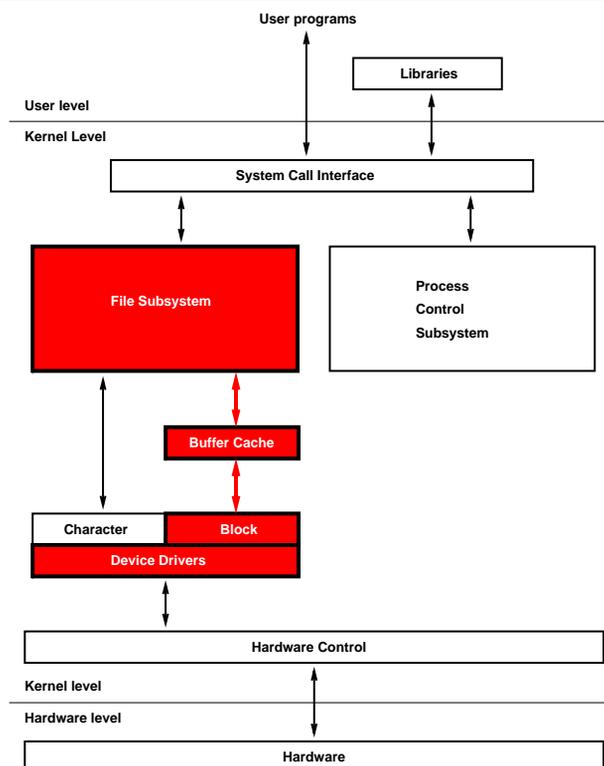
```
brw-r----- 1 root operator  4, 0 Jun 10 13:27 /dev/sd0a
brw-r----- 1 root sys       32, 8 Jul  9 1999 /dev/dsk/c0t1d0s0
```

Ce mode concerne surtout les disques durs :

- la lecture d'un octet de données se traduit par la lecture d'un bloc complet conservé alors dans un peu de mémoire du noyau, le **buffer cache**.
- la lecture de données supplémentaires passe par la consultation du cache d'abord
- une écriture est asynchrone : les données sont écrites dans le buffer cache
- les données écrites s'ajoutent au buffer cache
- en cas de remplissage du cache, les données sont écrites sur disque et le cache se vide

Finalité du mode bloc : **minimaliser les entrées / sorties**

**Les transferts se font toujours par blocs de taille fixée.**



## § 18.17 Mode caractère

Reconnaisable lors d'un `ls -l` :

```
crw-r----- 1 root operator 13, 0 Jun 10 13:27 /dev/rsd0a
crw-r----- 1 root sys      32, 8 Jan  6 2000 /dev/rdisk/c0t1d0s0
```

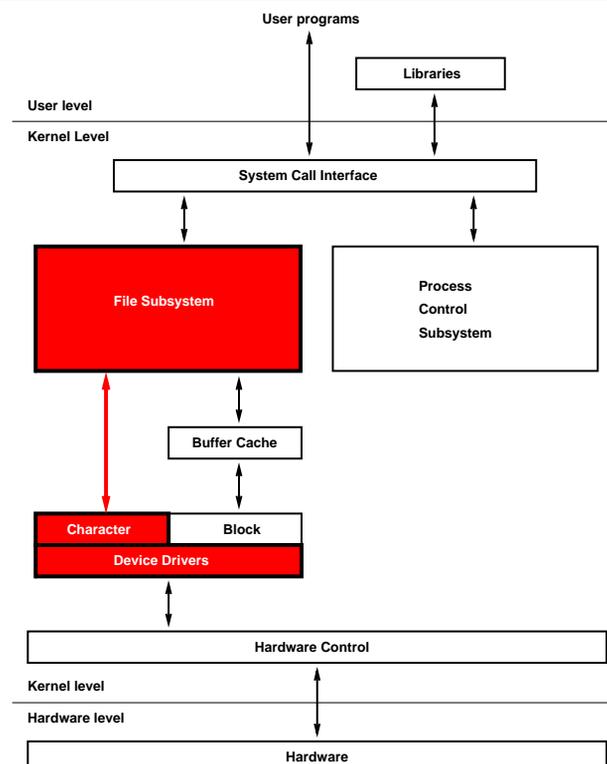
Un périphérique en mode caractère accepte un transfert d'une quantité quelconque d'octets.

**Pas de transit par une zone cache intermédiaire.** ⇒ accès rapide

D'autres opérations sont permises selon le périphérique en mode caractère :

- formatage
- réglage de paramètres de liaison série
- autres opérations de bas niveau
- ...

Les périphériques en mode block ont souvent aussi une interface en mode caractères.

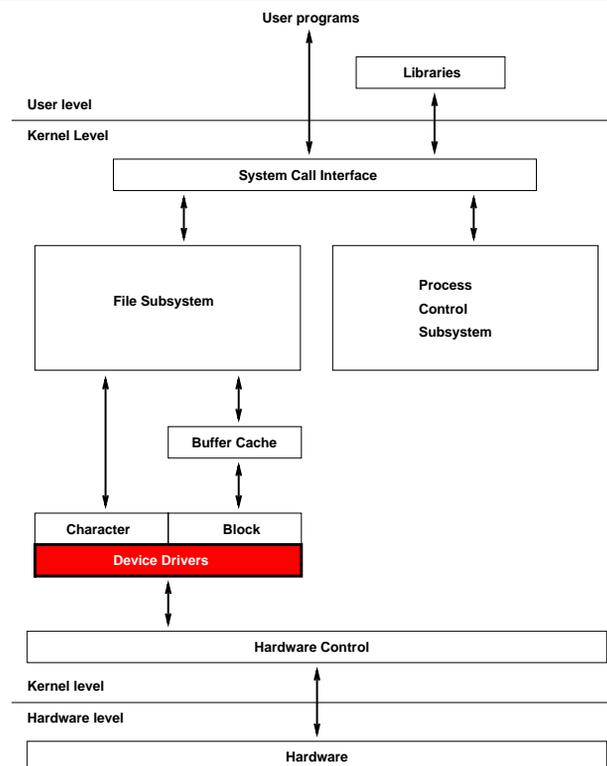


## § 18.18 Device drivers/Pilotes

L'utilisation d'une pièce de hardware nécessite un pilote spécial (dit "device driver").

⇒ un device driver par périphérique de type différent.

⇒ installation au niveau du noyau Unix du device driver (donc reconfiguration en cas d'ajout de device) ; l'installation est spécifique à chaque Unix.



**§ 18.19 Minor number / major number**

L'association entre le périphérique et le noyau est assurée par un couple : (minor number, major number)

- major number  
Il identifie le pilote dans le noyau.
- minor number  
Il identifie le périphérique pour un certain driver.

**§ 18.20 Création des fichiers spéciaux : `mknod`**

(en anglais *make node*)

La commande pour créer les fichiers spéciaux est **mknod**.

Syntaxe : `mknod nom [c|b] major minor`

Qui connaît les minor et les major numbers à mettre en paramètres ?

Ils ne s'inventent pas ! La réponse est dans le système !

Actuellement on distingue 2 approches différentes pour les générer :

- automatique
- manuelle

◇ Approche création dynamique des devices

Le système gère dynamiquement les devices au fur et à mesure qu'il rencontre des périphériques.

Par exemple sur FreeBSD 5.x :

1. insertion d'une clef USB ⇒ le système détecte la clef comme le montrent les lignes de `dmesg` suivantes :

```
Jul 26 13:27:50 alpha kernel: umass0: M-Systems DiskOnKey, rev 1.10/2.00, addr 3
Jul 26 13:27:50 alpha kernel: umass0: Get Max Lun not supported (SHORT_XFER)
Jul 26 13:27:50 alpha kernel: da0 at umass-sim0 bus 0 target 0 lun 0
Jul 26 13:27:50 alpha kernel: da0: <M-Sys DiskOnKey 2.51> Removable Direct Access SCSI-C
Jul 26 13:27:50 alpha kernel: da0: 1.000MB/s transfers
```

2. le système a créé les devices :

```
% ls -l /dev/*da0*
crw-rw----  1 root  operator    4,  34 Jul 20 10:16 /dev/da0
```

3. retrait de la clef USB : le système détecte que la clef n'est plus là :

```
Jul 26 13:29:49 alpha kernel: umass0: at uhub1 port 2 (addr 3) disconnected
Jul 26 13:29:49 alpha kernel: (da0:umass-sim0:0:0:0): lost device
Jul 26 13:29:49 alpha kernel: (da0:umass-sim0:0:0:0): removing device entry
Jul 26 13:29:49 alpha kernel: umass0: detached
```

4. suppression des devices associés à la clef

```
% ls -l /dev/*da0*
ls: /dev/*da0*: No such file or directory
```

Sur Solaris, booter la machine en faisant `boot -r` se charge de générer les devices qu'il faut.

#### ◇ Approche création statique des devices

On a une liste exhaustive des minors et majors pour les drivers de base. Sur ces systèmes, le programme `/dev/MAKEDEV` suit cette liste et génère convenable le périphérique qu'on lui désigne comme par exemple *le disque SCSI 1*.

C'est l'approche de Linux.

```
# cd /dev
# ls
MAKEDEV          MAKEDEV.local
# ./MAKEDEV std
# ls
MAKEDEV          fd          lkm          random       tty
MAKEDEV.local   io          mem           stderr       urandom
console         klog        null          stdin        zero
drum            kmem        pci           stdout
# ./MAKEDEV all
MAKEDEV          perfmon     rsd0s4       rwd3g        ttypb
MAKEDEV.local   psm0       rsd1          rwd3h        ttypc
apm             ptyp0      rsd1.ct1     rwd3s1       ttypd
bpf0           ptyp1      rsd1a        rwd3s2       ttype
...
```

**§ 18.21 Copie de devices : dd**

(en anglais *device to device*)

Syntaxe (non habituelle, attention) :

```
dd if=device-ou-fichier of=device-ou-fichier [bs=taille-bloc] [count=N]
```

◇ Exemple 1

Générer un fichier de 50 Mo :

```
# dd if=/dev/zero of=grosfichier bs=1M count=50
50+0 records in
50+0 records out
52428800 bytes transferred in 3.533640 secs (14837052 bytes/sec)
# ls -l grosfichier
-rw----- 1 besancon 52428800 Nov 16 01:27 grosfichier
```

◇ Exemple 2

Constituer un fichier image d'une disquette :

```
# dd if=/dev/fd0 of=/tmp/image.raw bs=16k
90+0 records in
90+0 records out
# ls -l /tmp/image.raw
-rw-r--r-- 1 besancon adm 1474560 Jan 25 22:36 /tmp/image.raw
```

## ◇ Exemple 2 bis

Recopier l'image sur une autre disquette :

```
# dd if=/tmp/image.raw of=/dev/fd0 bs=16k
90+0 records in
90+0 records out
# mdir a:
Volume in drive A has no label
Volume Serial Number is 2186-280C
Directory for A:/

z      tar      378880 10-26-2003 18:45 z.tar
      1 file                378 880 bytes
                        1 078 784 bytes free
```

## Chapitre 19 : Systèmes de fichiers

Exemple récent :

Source : *LIBÉRATION* vendredi 18 juin 2004 (Reuters - 08 :32) – par Lucas van Grinsven

LES SITES DE STOCKAGE DE PHOTOS NUMÉRIQUES SE MOBILISENT

AMSTERDAM - Trouver un stockage gratuit pour leurs données en ligne est devenu une préoccupation de millions d'internautes, que ce soit pour leur courrier électronique ou leurs photographies numériques.

Deux des principaux sites de photographie numérique, Ofoto ([www.ofoto.com](http://www.ofoto.com)), contrôlé par Kodak, et Fotango ([www.photos.fotango.com](http://www.photos.fotango.com)), détenu par Canon, proposent une capacité illimitée de sauvegarde.

Ofoto revendique plus de 13 millions d'utilisateurs pour 450 millions de photos stockées sur ses serveurs. **Chaque semaine, le site ajoute cinq téraoctets de mémoire** pour accueillir cinq millions de nouvelles images.

Comment cela marche-t-il ?

### § 19.1 Technologies de disques durs

Plusieurs technologies de disques durs : ST-506, ESDI, SMD (*Storage Module Device*), ESMD, IPI, SCSI (1, 2, 3, Fibre Channel), IDE (ATA100, ATA133), Firewire, Disk On Flash, etc.



(source <http://www.almaden.ibm.com/sst/>)

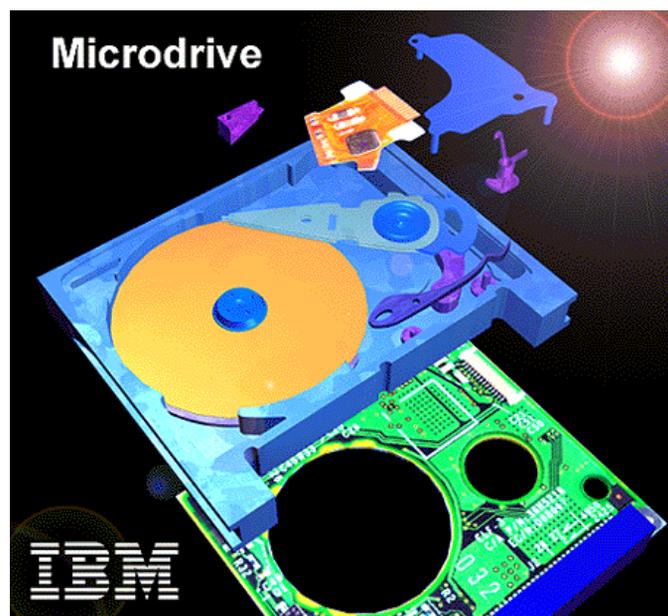
Perspectives :

IBM HDD Design Projections					
	1999	2000	2001	2002	2004
<b>Server</b> 3.5 Inch HDD 25.4 mm high 	36 GB 10000 RPM 4.9 ms Tseek	73 GB 10000 RPM 4.9 ms Tseek	36 GB 15000 RPM 4.0 ms Tseek	147 GB 10000 RPM 4.9 ms Tseek	147 GB 15000 RPM 3.5 ms Tseek
<b>Desktop/Entry-Server</b> 3.5 Inch HDD 25.4 mm high 	37 GB 7200 RPM 8.5 ms Tseek	75 GB 7200 RPM 8.5 ms Tseek	120 GB 7200 RPM 8.5 ms Tseek	200 GB 7200 RPM 8.0 ms Tseek	300 GB 10000 RPM 7.0 ms Tseek
<b>Mobile/Desktop</b> 2.5 Inch HDD 9.5/12.5mm high 	25 GB 5400 RPM 12 ms Tseek	36 GB 5400 RPM 12 ms Tseek	60 GB 5400 RPM 12 ms Tseek	120 GB 7200 RPM 10 ms Tseek	200 GB 7200 RPM 8 ms Tseek
<b>Consumer</b> 1.0 Inch HDD 5.0 mm high 	0.34 GB 4500 RPM 15 ms Tseek	1.0 GB 3600 RPM 15 ms Tseek		2.0 GB 4500 RPM 12 ms Tseek	4.0 GB 4500 RPM 12 ms Tseek

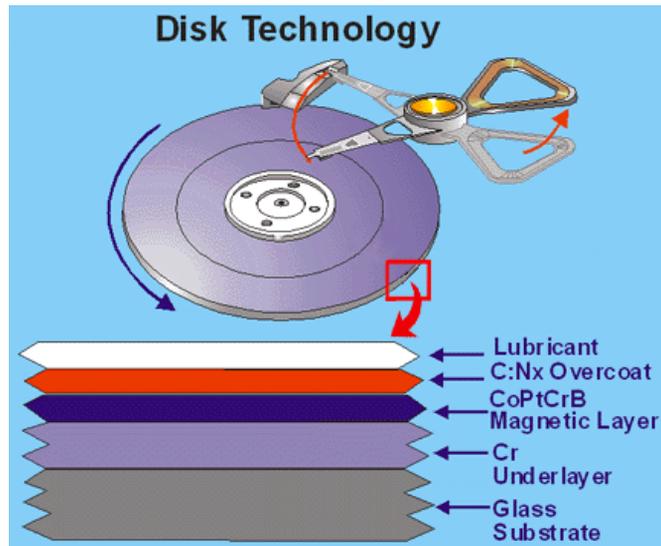
ED GROCHOWSKI at ALMADEN

(source <http://www.almaden.ibm.com/sst/>)

Structure d'un disque dur (quelle que soit sa technologie) :

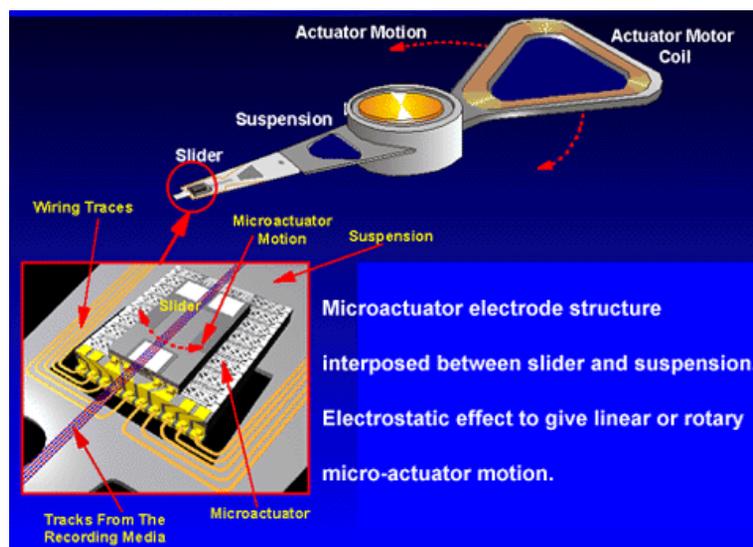
(source <http://www.almaden.ibm.com/sst/>)

La tête d'un disque dur plane au dessus du plateau magnétique :



(source <http://www.almaden.ibm.com/sst/>)

Une tête de lecture d'un disque dur est un objet très sophistiquée, très fragile en cas de choc :



(source <http://www.almaden.ibm.com/sst/>)

## § 19.2 Disques durs IDE

Apparus avec les PC.

Différences avec un disque SCSI :

- prix inférieur
- carte contrôleur différente (la mécanique est donc identique)

Evolution ?

- plus grosse capacité
  - vitesse d'accès, de rotation
- ⇒ cela devrait profiter aussi à SCSI

## § 19.3 Disques durs SCSI

(en anglais *Small Computer System Interface*)

SCSI : pas uniquement des disques

1986 : SCSI 1

1990 : SCSI 2

? : SCSI 3

Limitations de SCSI 2 :

- limitation à 16 périphériques par bus SCSI
- longueur de câble limitée
- l'ajout de périphérique diminue la longueur de câble autorisé
- c'est du **parallèle SCSI**

## Améliorations : SCSI 3

- passage d'un câble de cuivre à du **fibre chanel** dont les trames peuvent transiter sur du câble de cuivre ou sur de la fibre optique
- nombre de périphériques illimité (limite à 16 millions)
- décomposition de la norme SCSI 3 en plusieurs couches (la norme SCSI 3  $\equiv$  20 documents contre un seul pour SCSI 2)
  - ⇒ séparation entre les couches physiques de SCSI 3 et les couches hautes
- c'est du **serial SCSI**

**§ 19.4 Duplicateurs de disques durs**

Par exemple :



Se reporter par exemple à :

- <http://www.pgsoftware.net> pour un revendeur français de deux marques d'appareils de ce type
- <http://www.aberdeeninc.com> pour un fabricant

**§ 19.5 Tests de disques durs**

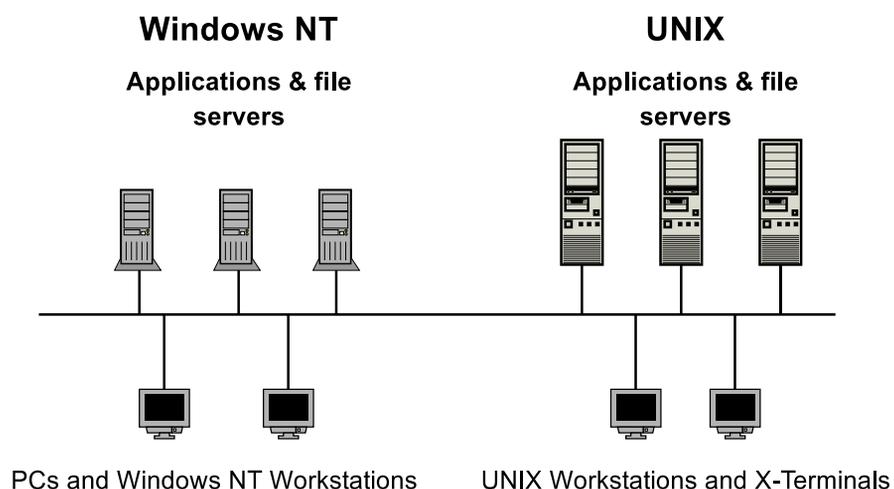
Des applications de tests de disques durs sont disponibles sur les sites suivants :

- applications pour disques de marque Western Digital :  
`http://support.wdc.com/download/index.asp#diagutils`
- applications pour disques de marque Seagate :  
`http://www.seagate.com/support/seatools/`
- applications pour disques de marque Hitachi :  
`http://www.hitachigst.com/hdd/support/download.htm`
- applications pour disques de marque Fujitsu :  
`http://www.fcpa.com/support/hard-drives/software_utilities.html`
- applications pour disques de marque Maxtor :  
`http://www.maxtor.com/en/support/downloads/`
- applications de la société OnTrack pour marques diverses de disques :  
`http://www.ontrack.fr/`

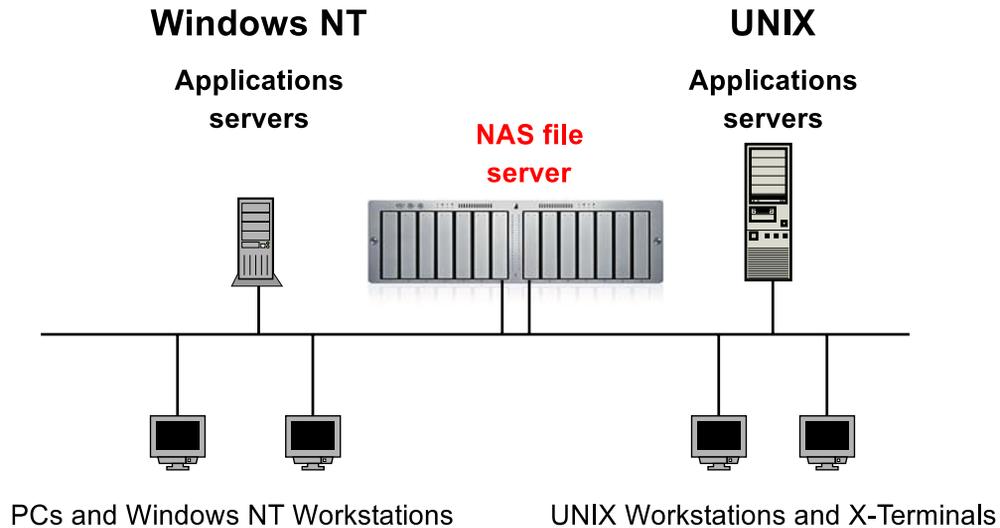
**§ 19.6 Technologie NAS : Network Attached Storage**

Cf le cours de Marc Cheminaud pour plus de détails.

Problème : les multiples serveurs de fichiers du réseau



Solution du NAS : une machine spécialisée équipée de disques RAID et qui sert ces fichiers via le réseau IP en protocoles NFS, CIFS en général.



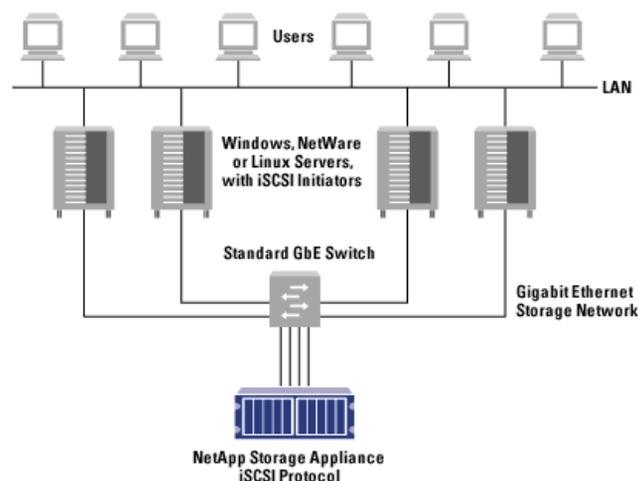
La marque la plus connue : Network Appliance, <http://www.netapp.com>.

### § 19.7 Technologie SAN : Storage Array Network

Cf le cours de Marc Cheminaud pour plus de détails.

Principe : chaque machine cliente de fichiers est équipée d'une carte adaptateur l'interconnectant à un réseau Fibre Channel composé d'unités de stockage (RAID, TAPE, etc.) mises en commun

**SAN est un réseau qui utilise le protocole serial SCSI.**



**§ 19.8 Technologie RAID**

RAID  $\equiv$  Redundancy Array of Inexpensive Disks

Papier historique : «A Case for Redundant Arrays of Inexpensive Disks (RAID)» par Patterson, Gibson, et Katz de l'Université de Berkeley Californie.

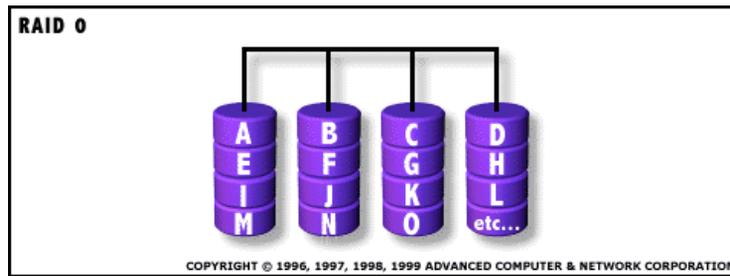
<http://sunsite.berkeley.edu:80/Dienst/Repository/2.0/Body/ncstr1.ucb/CSD-87-391/pdf>

Cf le cours de Marc Cheminaud.

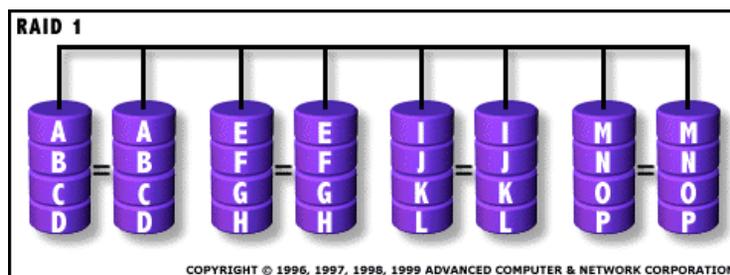
«Principe à la louche» : on n'utilise plus de disques durs indépendants mais un ensemble de disques durs (en général rackable) gérés par des contrôleurs disques spécialisés.



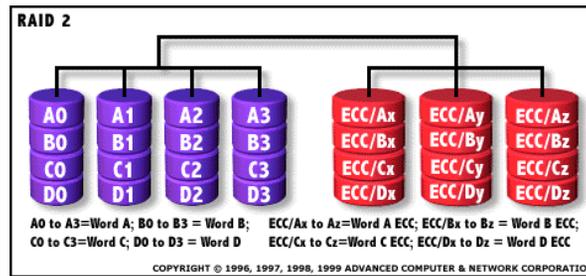
L'ensemble est censé encaisser la panne d'au moins un disque dur. En cas de panne d'un disque dur, le système continue de travailler. Un disque en remplacement sera rempli par des données reconstruites à partir des données sur le reste des disques de l'ensemble.



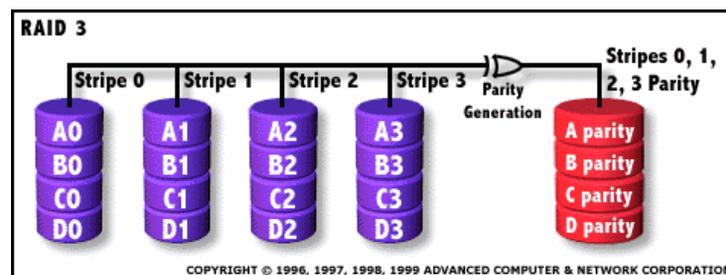
- Principe du RAID 0 : **striping**
  - minimum de 2 disques
  - les données sont découpées en blocs (A, B, C, etc.) écrits sur des disques distincts
- Avantages :
  - I/O en parallèle
  - simple à réaliser
  - maximum de performance
- Inconvénients :
  - non tolérant aux pannes (un disque en panne  $\Rightarrow$  tout est perdu)



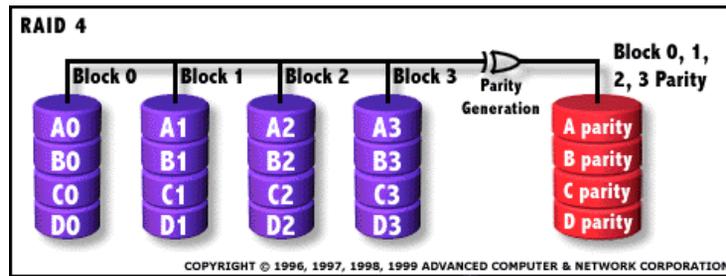
- Principe du RAID 1 : **mirroring et duplexing**
  - minimum de 2 disques
  - les données sont découpées en blocs (A, B, C, etc.) écrits sur des disques jumeaux
- Avantages :
  - lecture sur n'importe lequel des disques jumeaux  $\Rightarrow$  lecture deux fois plus performante
  - simple à réaliser
- Inconvénients :
  - pas très efficace (100% de disques en plus)
  - charge le CPU en général (car réalisation en logiciel et pas en hardware)



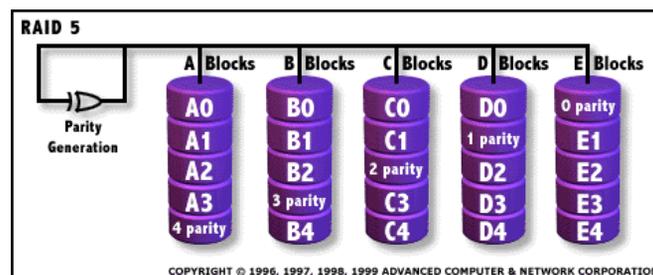
- Principe du RAID 2 : **code correcteur d'erreur ECC**
  - la donnée est l'octet
  - on découpe les données et on en calcule le code ECC que l'on écrit sur des disques dédiés
  - une lecture est vérifiée par le code ECC associé à la donnée ; on corrige ainsi une panne de disque
- Avantages :
  - grand taux de transfert de données
- Inconvénients :
  - inefficace
  - cher ; peu d'offres commerciales



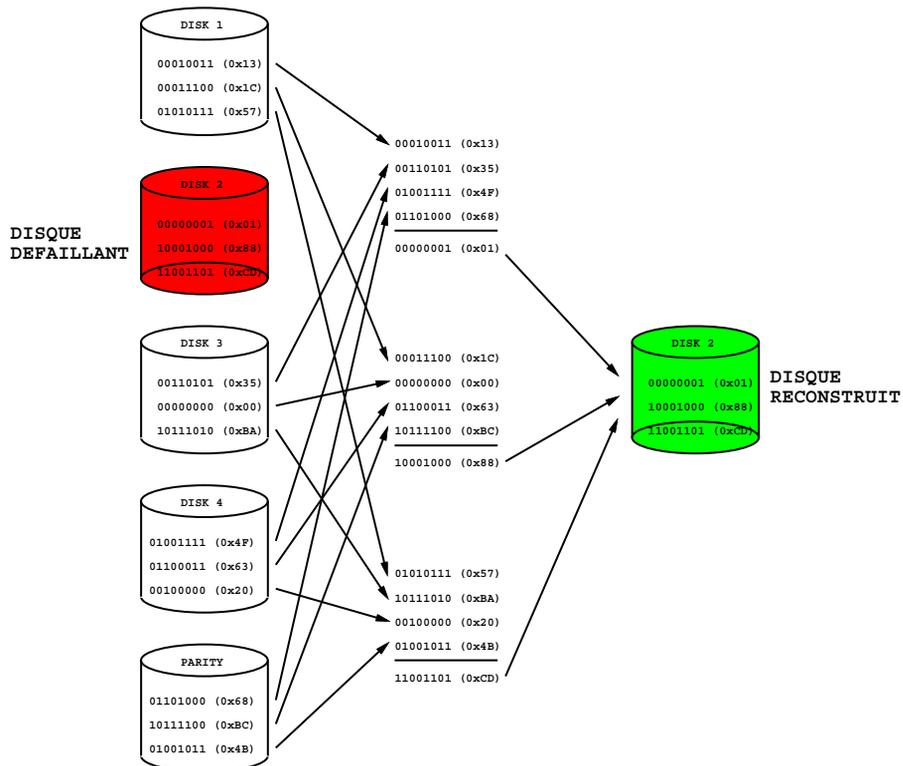
- Principe du RAID 3 (version simplifiée du RAID 2 avec un seul disque de parité) :
  - minimum de 3 disques
  - on découpe les octets en stripes écrits sur les disques
  - on calcule la parité du stripe que l'on écrit sur le disque dédié de parité
  - on vérifie la parité du stripe lors des lectures
- Avantages :
  - taux élevé de lectures ou d'écritures
  - panne disque relativement insensible
- Inconvénients :
  - contrôleur disque complexe



- Principe du RAID 4 :
  - minimum de 3 disques
  - on travaille par bloc d'octets que l'on disperse sur les disques durs
  - on calcule la parité de ces blocs que l'on écrit sur le disque dédié de parité
- Avantages :
  - les disques durs peuvent agir indépendamment
- Inconvénients :
  - reconstruction d'un disque complexe
  - pour écrire un bloc, on doit lire le bloc et sa parité ⇒ globalement plus lent
  - peu d'offres commerciales



- Principe du RAID 5 :
  - minimum de 3 disques
  - chaque bloc d'octets est écrit sur un disque
  - la parité est dispersée sur tous les disques évitant ainsi un goulet d'étranglement du RAID 4.
- Avantages :
  - Fort taux de transfert en lecture, taux moyen en écriture
  - Solution populaire
- Inconvénients :
  - Contrôleur complexe
  - Reconstruction délicate d'un disque



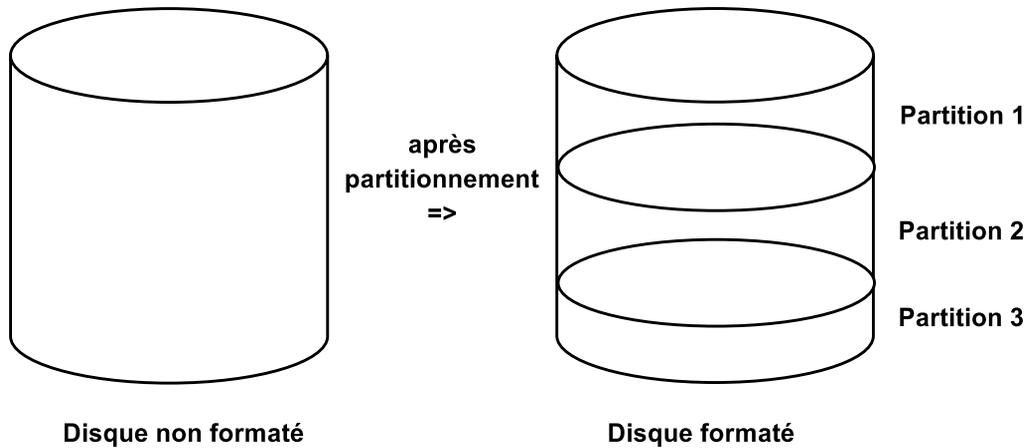
## § 19.9 Filesystem

Filesystem  $\equiv$  structure performante de gestion des fichiers

Exemple : *Fast File System* dit FFS, EXT2 ou EXT3 de LINUX, etc.

Brèvement, en voici quelques caractéristiques :

1. Notion d'inode
2. Notion d'écriture géographiquement proche
3. Optimisation de la gestion de la liste des blocs libres
4. Stratégie d'allocation des blocs

**§ 19.10 Partitionnement de disques durs, format, fdisk**

Pourquoi partitionner un disque dur en filesystems ?

- pour cloisonner les utilisateurs c'est-à-dire les problèmes
- pour faciliter les sauvegardes
- pour limiter l'expansion de certains utilisateurs au détriment des autres
- pour faciliter l'utilisation de NFS (Network File System)

Problèmes courants avec les partitions :

- Une partition est pleine alors qu'une autre, surdimensionnée est quasi vide.
- Les différents constructeurs divergent sur les implémentations des partitions.

Une partition peut contenir :

- un filesystem
- une zone de swap, sans structure de filesystem plaquée dessus
- rien ; la partition est réservée pour un usage ultérieur

Les informations concernant les partitions sont stockées dans une zone du disque dite **disk label** (sur SunOS dans le premier secteur de la première partition).

Le disk label contient :

- caractéristiques physiques du disque
- paramètres de partitionnement
- secteurs défectueux

Les commandes pour déposer un disk label sont propres à chaque constructeur (`disklabel`, `format ...`):

- Sur Solaris : `format`
- Sur Linux : `fdisk`

## § 19.11 Liste des partitions SOLARIS : prtvtoc

Syntaxe : `prtvtoc raw-device`

```
# prtvtoc /dev/rdisk/c1t1d0s2
* /dev/rdisk/c1t1d0s2 partition map
*
* Dimensions:
*   512 bytes/sector
*   417 sectors/track
*   6 tracks/cylinder
*   2502 sectors/cylinder
*   14342 cylinders
*   14340 accessible cylinders
*
* Flags:
*   1: unmountable
*   10: read-only
*
*
* Partition  Tag  Flags      First      Sector      Last
*           Tag  Flags      Sector     Count       Sector  Mount Directory
*   0         8   00         0  35878680  35878679
*   2         5   01         0  35878680  35878679  /mnt/ufs/1
```

## § 19.12 Liste des partitions LINUX : *fdisk*

Syntaxe : *fdisk -l*

```
# /sbin/fdisk -l /dev/hdc
```

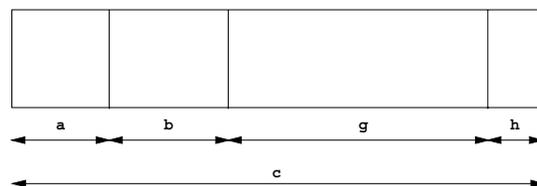
```
Disk /dev/hdc: 16 heads, 63 sectors, 148945 cylinders
Units = cylinders of 1008 * 512 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/hdc1		1	4162	2097616+	82	Linux swap
/dev/hdc2		4163	75000	35702352	83	Linux
/dev/hdc3		75001	148945	37268280	83	Linux

## § 19.13 Nommage des partitions

En général, on a 8 partitions par disque au plus.

Leur géographie est conventionnellement quelque chose du genre :



◇ Partitions System-V

Les noms des partitions sont de la forme `/dev/dsk/cn1dn2sn3`, où :

- $n_1$  est le numéro du contrôleur ;
- $n_2$  est le numéro du disque ;
- $n_3$  est le numéro de la section sur le disque.

```
% df -F ufs -k
Filesystem          1024-blocks  Used Available Capacity Mounted on
/dev/dsk/c201d6s0   745942     636366    34981     95%  /
```

◇ Partitions LINUX

Il y aura une différence selon que les disques utiliseront des contrôleurs autres que les contrôleurs IDE de base.

Exemple avec un contrôleur IDE :

```
% df -k -t ext3
Filesystem          1k-blocks      Used Available Use% Mounted on
/dev/hdd3            1032056     244640    734976   25% /
/dev/hdd1              63413       23066     37073   39% /boot
/dev/hdd8            2064168      62636   1896680    4% /tmp
/dev/hdd6            4128400   1686152   2232536   44% /var
/dev/hdd2            4128432   3069260    849460   79% /usr
/dev/hdd5            4128400   276480    3642208    8% /home
```

Exemple avec un contrôleur RAID SCSI :

```
# df -k
Filesystem          1K-blocks      Used Available Use% Mounted on
/dev/cciss/c0d0p2   32890776  24691344   6528672   80% /
/dev/cciss/c0d0p1    98747      10991     82657   12% /boot
/dev/cciss/c0d1p1   35002928  12628172  20596696   39% /share/vmware2
```

◇ Partitions SOLARIS

`/dev/dsk/cn1tn2dn3sn4` (contrôleur, target, disk, slice) :

```
% df -F ufs -k
Filesystem      kbytes   used   avail capacity Mounted on
/dev/dsk/c0t0d0s0 246167  60541 161010    28%   /
/dev/dsk/c0t0d0s6 962582 419332 485496    47%  /usr
/dev/dsk/c0t1d0s0 1015695 822607 132147    87% /net/serveur/home
/dev/dsk/c0t1d0s1 7717573 3463222 4177176    46%  /usr/local
/dev/dsk/c0t0d0s5 183687  13017 152302     8%   /opt
```

◇ Partitions BSD

Les noms typiques sont de la forme `/dev/ddnp`, où :

- *dd* est un code mnémonique du driver ;
- *n* est le numéro du disque ;
- *p* est une lettre entre a et h identifiant la partition ;

```
% df
Filesystem 1K-blocks   Used   Avail Capacity Mounted on
/dev/sd0a      95359   42623   45108    49%   /
```

**§ 19.14 Formatage (1) : `mkfs`**

(en anglais *make file system*)

Commande : `mkfs [options] fichier-spécial`

Par exemple sur Solaris :

```
# mkfs -F ufs /dev/rdiskette
size not specified
ufs usage: mkfs [-F FSType] [-V] [-m] [-o options] special size(sectors) \
[nsect ntrack bsize fragsize cpg free rps nbpi opt apc gap nrpos maxcontig]
-m : dump fs cmd line used to make this partition
-V :print this command line and return
-o :ufs options: :nsect=32,ntrack=16,bsize=8192,fragsize=1024
-o :ufs options: :cgsiz=0,free=0,rps=60,nbpi=2048,opt=t
-o :ufs options: :apc=0,gap=0,nrpos=0,maxcontig=0
NOTE that all -o suboptions: must be separated only by commas so as to
be parsed as a single argument
```

⇒ **la commande nécessite des paramètres complexes**

**§ 19.15 Formatage (2) : `newfs`**

(en anglais *new file system*)

La commande `newfs` offre des options simples et appelle en fait derrière `mkfs` avec ce qu'il faut.

Syntaxe la plus répandue : `newfs [options] fichier-spécial`

Options intéressantes :

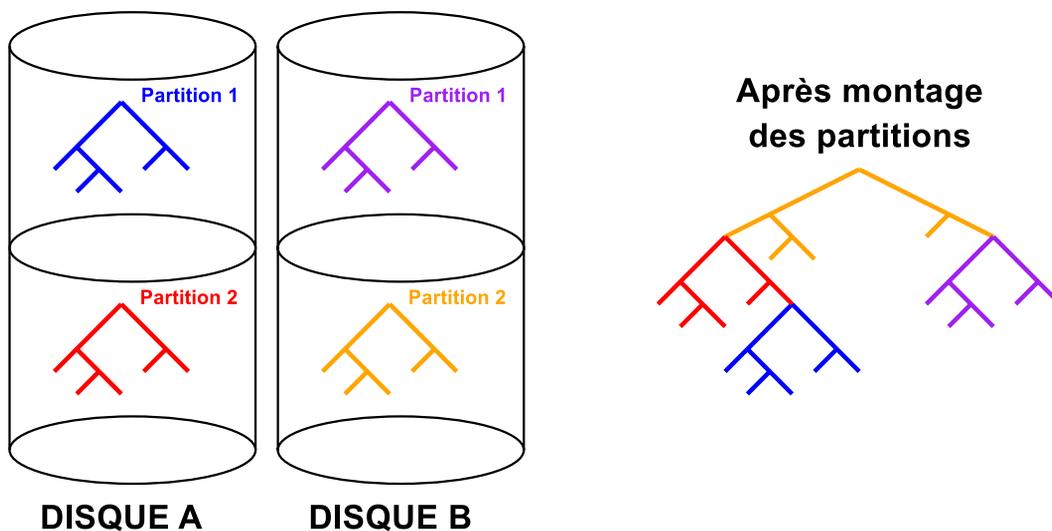
- option `-m` : précise la taille de `minfree`.
- option `-i` : précise le nombre d'octets de données pour un inode (2048 octets pour un inode par défaut).
- option `-v` : affiche la commande `mkfs` qui serait lancée

Par exemple sur Solaris :

```
# newfs -v /dev/rdiskette
newfs: construct a new file system /dev/rdiskette: (y/n)? y
mkfs -F ufs /dev/rdiskette 2880 18 2 8192 1024 16 10 5 2048 t 0 -1 8 16
/dev/rdiskette: 2880 sectors in 80 cylinders of 2 tracks, 18 sectors
      1.4MB in 5 cyl groups (16 c/g, 0.28MB/g, 128 i/g)
super-block backups (for fsck -F ufs -o b=#) at:
  32, 640, 1184, 1792, 2336,
```

### § 19.16 Montage de filesystems : mount

La greffe d'une partition d'un disque sur l'arborescence s'appelle le **montage** d'un filesystem.



Syntaxe : `mount [options] fichier-spécial point-de-montage`

Exemples :

```
# mount /dev/sd0g /mnt
# mount -t cdfs -o rrip,ro /dev/rz4c /cdrom
```

Attention :

1. Le point de montage doit exister avant de lancer la commande.
2. Le contenu du point de montage devient invisible après le montage.
3. On ne peut monter un filesystem que si celui-ci a une structure interne marquée comme cohérente (cf la suite sur fsck) :

```
# mount /users
WARNING: R/W mount of /users denied. Filesystem is not clean - run fsck.
```

## § 19.17 Démontage de filesystems : umount

Syntaxe : `umount point-de-montage`

Attention : on ne peut pas démonter une partition

1. si une commande s'exécute dans la partition
2. si des fichiers sont ouverts dans la partition
3. si l'on a un répertoire courant dans un répertoire de la partition

(ce qui se ramène à avoir un file descripteur ouvert sur la partition dans tous les cas)

Message d'erreur classique :

```
# cd /mnt
# umount /mnt
umount: /mnt: Device busy
```

**§ 19.18 Remontage à chaud de filesystems**

Rappel : Traditionnellement un bot en single user monte la partition « / » uniquement et elle est en read-only.

Contexte :

- On boote en single user.
- On veut écrire sur « / » (par exemple pour corriger quelque chose)

Comment faire ? car on ne peut pas démonter « / » car la partition est busy (seule partition montée et on est dedans).

⇒ existence d'une option de « remontage à chaud » de partition :

```
# mount -o remount,rw /
```

**A verifier...**

**§ 19.19 Identification des causes des filesystems busy : *fuser***

(en anglais *file user*)

Contexte : on veut démonter une partition mais elle est busy !

Une solution : commande *fuser* (à peu près standard)

```
# cd /mnt
# umount /mnt
umount: /mnt: Device busy
# fuser /mnt
/mnt: 17853c 10237c
# echo $$
17853
```

## § 19.20 Identification des causes des filesystems busy : `lsof`

(en anglais *list of open files*)

Contexte : on veut démonter une partition mais elle est busy !

Une solution : commande `lsof`

`ftp://vic.cc.purdue.edu/pub/tools/unix/lsof/`

```
# cd /mnt
# umount /mnt
umount: /mnt: Device busy
# lsof /mnt
COMMAND  PID USER  FD   TYPE DEVICE SIZE/OFF NODE NAME
bash     21372 root   cwd   VDIR  36,2    512    2 /mnt
lsof     21376 root   cwd   VDIR  36,2    512    2 /mnt
lsof     21377 root   cwd   VDIR  36,2    512    2 /mnt
# echo $$
21372
```

## § 19.21 Liste des partitions montées : `df`

(en anglais *display free*)

La commande `df` permet de voir la liste des montages :

```
% df -k
Filesystem          kbytes  used  avail capacity  Mounted on
/dev/dsk/c0t0d0s0    123231  39164  71744    36%      /
/dev/dsk/c0t0d0s3   1523574  699814  762818    48%     /usr
/dev/dsk/c0t0d0s4    492422   27835  415345     7%     /var
/dev/dsk/c0t0d0s5   3095966 1371773 1662274    46%     /opt
serveur.example.com:/users
                    4375366 2949034 1382579    69%     /users
serveur.example.com:/logiciels
                    650847  343011  242756    59%     /logiciels
```

**Attention** : certains Unix affichent les informations en blocs de 512 octets.

⇒ utiliser l'option `-k` pour préciser d'afficher en kilo-octets.

On peut limiter l'affichage à certains types de filesystems :

- Sur LINUX et la plupart des Unix traditionnels :

```
df -t type-du-filesystem
```

(par exemple sur LINUX : types `ext2`, `ext3`, `iso9660`, `proc`, `nfs`)

- Sur SUN Solaris :

```
df -F type-du-filesystem
```

(par exemple : types `ufs`, `hsfs`, `proc`, `lofs`, `nfs`)

◇ Exemples sur LINUX :

On peut limiter l'affichage aux disques locaux (ie présents sur la machine même) :

```
% df -k -t ext2
```

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
/dev/sda3	15330272	6080340	8471184	42%	/
/dev/sda1	256667	34981	208434	15%	/boot

## ◇ Exemples sur SUN Solaris :

On peut limiter l'affichage aux disques locaux (ie présents sur la machine même) :

```
% df -k -F ufs
Filesystem          kbytes    used   avail capacity  Mounted on
/dev/dsk/c0t0d0s0   123231    39164   71744    36%      /
/dev/dsk/c0t0d0s3   1523574  699814  762818    48%     /usr
/dev/dsk/c0t0d0s4    492422    27835  415345     7%     /var
/dev/dsk/c0t0d0s5   3095966 1371773 1662274    46%     /opt
```

On peut limiter l'affichage aux disques distants :

```
% df -k -F nfs
Filesystem          kbytes    used   avail capacity  Mounted on
serveur.example.com:/users
                    4375366 2949034 1382579    69%     /users
serveur.example.com:/logiciels
                    650847  343011  242756    59%     /logiciels
```

**Attention** : `df` peut bloquer sur un montage défaillant (disque presque en panne, serveur réseau injoignable).

⇒ Lancer "`df &`" en cas de blocage possible.

**Attention** : ne pas confondre :

- la commande `df` renvoie la liste des partitions montées
- la commande `fdisk -L` renvoie la liste des partitions d'un disque, que les partitions soient montées ou pas

## § 19.22 Liste des partitions montées (2) : mount

La commande `mount` permet aussi d'avoir la liste des filesystems montés :

```
% mount
/ on /dev/dsk/c0t0d0s0 read/write/setuid/intr/largefiles/logging/onerror=panic/dev=2200000 \
on Tue Oct 10 17:49:13 2000
/usr on /dev/dsk/c0t0d0s3 read/write/setuid/intr/largefiles/logging/onerror=panic/dev=2200003 \
on Tue Oct 10 17:49:13 2000
/proc on /proc read/write/setuid/dev=2d80000 on Tue Oct 10 17:49:11 2000
/dev/fd on fd read/write/setuid/dev=2e40000 on Tue Oct 10 17:49:14 2000
/etc/mnttab on mnttab read/write/setuid/dev=2f40000 on Tue Oct 10 17:49:16 2000
/var on /dev/dsk/c0t0d0s4 read/write/setuid/intr/largefiles/logging/onerror=panic/dev=2200004 \
on Tue Oct 10 17:49:16 2000
/var/run on swap read/write/setuid/dev=1 on Tue Oct 10 17:49:16 2000
/tmp on swap read/write/setuid/dev=2 on Tue Oct 10 17:49:18 2000
/opt on /dev/dsk/c0t0d0s5 read/write/setuid/intr/largefiles/logging/onerror=panic/dev=2200005 \
on Tue Oct 10 17:49:19 2000
/infosystems on /dev/dsk/c0t0d0s6 read/write/setuid/intr/largefiles/logging/onerror=panic/dev=2200006 \
on Tue Oct 10 17:49:19 2000
/users on serveur.example.com:/users remote/read/write/setuid/dev=2fc0001 \
on Tue Oct 10 17:49:24 2000
/logiciels on serveur.example.com:/logiciels remote/read/write/setuid/dev=2fc0002 \
on Wed Oct 11 10:17:21 2000
```

**Attention** : `mount` peut bloquer sur un montage défaillant (disque presque en panne, serveur réseau injoignable).

⇒ Lancer "`mount &`" en cas de blocage possible.

**Attention** : ne pas confondre :

- la commande `mount` renvoie la liste des partitions montées
- la commande `fdisk -L` renvoie la liste des partitions d'un disque, que les partitions soient montées ou pas

## § 19.23 Montage automatique au boot : /etc/fstab, /etc/vfstab

(en anglais *file system table*)

Au démarrage, les disques sont montés automatiquement par la commande "mount -a".

La commande "mount -a" utilise le contenu du fichier /etc/fstab et monte tous les volumes mentionnés dedans.

Par exemple sur Linux, le fichier /etc/fstab contient quelque chose comme :

```
/dev/hda1      /                ext2           defaults       1 1
/dev/cdrom     /mnt/cdrom       iso9660        noauto,owner,ro 0 0
/dev/fd0       /mnt/floppy      auto           noauto,owner   0 0
none          /proc            proc           defaults       0 0
none          /dev/pts         devpts         gid=5,mode=620 0 0
/dev/hda5     swap             swap           defaults       0 0
```

Sur Solaris, le fichier équivalent à /etc/fstab s'appelle /etc/vfstab et suit une syntaxe équivalente.

Le format de /etc/fstab (ou équivalent) :

- champ 1 : nom du fichier spécial
- champ 2 : point de montage
- champ 3 : type de flesystem
- champ 4 : options de montage
- champ 5 : utilisé par la commande "dump"
- champ 6 : numéro d'ordre du disque lors d'une vérification de cohérence des disques par la commande **fsck**

Une fois renseigné le fichier /etc/fstab, on peut :

- monter la totalité des partitions : `mount -a`
- monter une certaine partition : `mount partition`  
comme par exemple :  
# `mount /var`

## § 19.24 Gestion des quotas : *quotaon, quota, edquota*

Le mécanisme des quotas impose à l'utilisateur des limites sur :

1. le nombre de blocs disque utilisables **par filesystem**
2. le nombre d'inodes utilisables **par filesystem**

Une limite est en fait composée d'un couple de valeurs :

– **limite soft** :

Aucune alarme n'est activée tant qu'on reste en dessous de cette limite. Si l'on dépasse ce seuil, il y a une alarme qui est activée. Des messages d'erreur apparaissent. L'utilisateur a 7 jours pour revenir en dessous de ce seuil. Au delà de 7 jours, toute consommation de ressources supplémentaires est bloquée.

– **limite hard** :

C'est un seuil supérieur indépassable. L'utilisateur s'il dépasse la limite soft peut continuer de consommer des ressources à concurrence de la limite hard.

Par exemple :

```
% quota -v besancon
Disk quotas for besancon (uid 4332):
Filesystem  usage  quota  limit  timeleft  files  quota  limit  timeleft
/adm        98696 80000 100000 7.0 days  4959  4000  5000  7.0 days
```

L'utilisateur `besancon` dépasse son quota soft et a 7 sept jours pour revenir en dessous des 2 limites soft.

Les commandes de gestion des quotas sont :

Commande	Description
<code>quota -v utilisateur</code>	Affiche l'utilisation disque de l'utilisateur et son quota
<code>quotacheck</code>	Recalcule les quotas utilisés
<code>quotaon</code>	Active les quotas
<code>quotaoff</code>	Désactive les quotas
<code>repquota</code>	Rapport de situation des quotas
<code>edquota</code>	Modification du quota d'un utilisateur

```
# repquota -v -a
/dev/sd3g (/adm):
                Block limits
User           used soft  hard timeleft  used soft  hard timeleft
besancon  ++  98696 80000 100000 7.0 days  4959 4000 5000 7.0 days
```

L'activation du système de quotas nécessite les points suivants :

1. Le noyau doit supporter le mécanisme des quotas. En effet l'aspect comptabilité incombe au noyau.
2. Le filesystem doit supporter les quotas.
3. Les quotas doivent être activés lors du montage de la partition :
  - (a) Préciser que le filesystem est monté avec les quotas activés au niveau de `/etc/fstab` ou équivalent :
 

```
/dev/sd3g /adm                4.2 rw,quota 1 4
```
  - (b) Lancer `quotaon`. De préférence pendant le boot de la station. Une opération `quotacheck` doit être lancée au préalable afin d'avoir un état cohérent des quotas. Cela peut ralentir le boot.
4. Les quotas doivent être entrés pour chaque utilisateur. C'est une opération automatisable au moyen d'un squelette type et de la commande `edquota`.

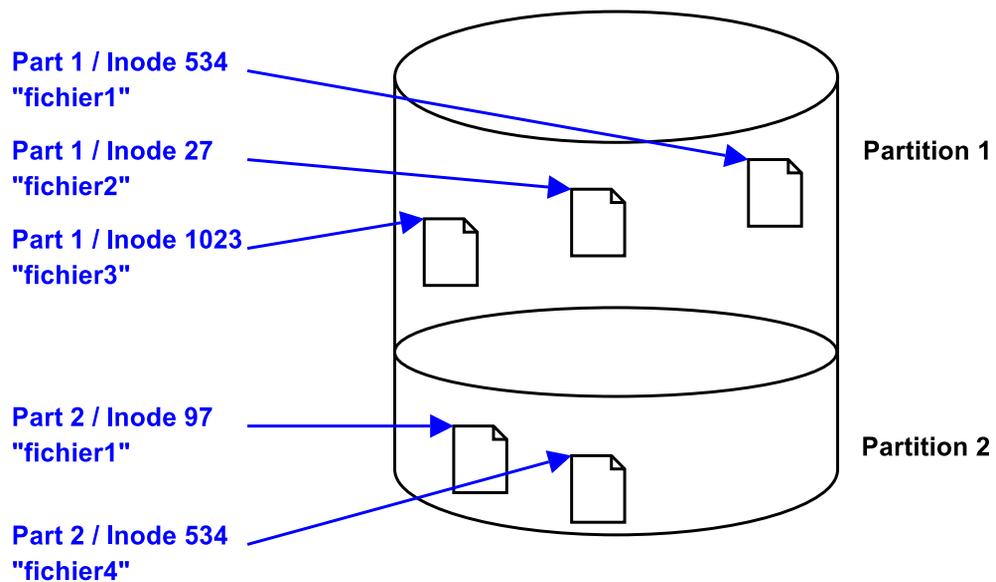
**§ 19.25 Structure interne associée à un objet : inode**

inode  $\equiv$  *Index Node*

**Un objet est identifié sur une partition de disque dur par son numéro d'inode (index node) et non pas par son nom.**

L'inode n'est unique qu'au sein d'une même partition.

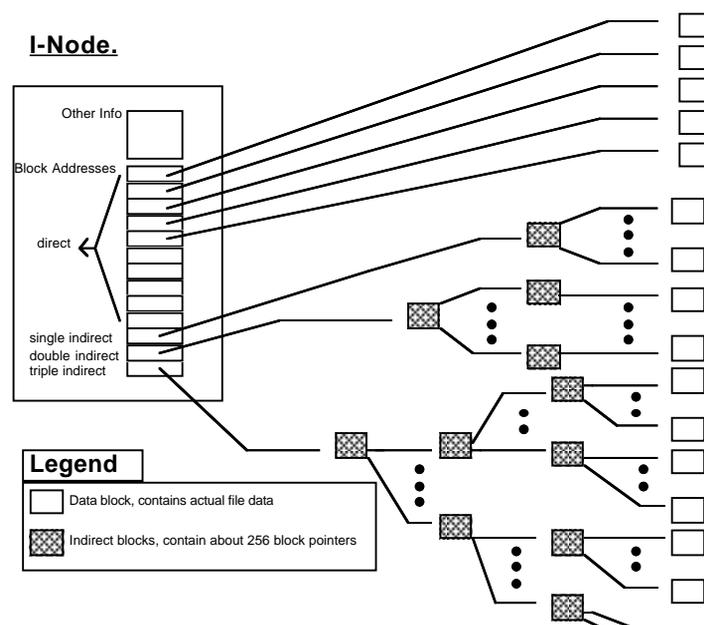
Le système manipule en fait les objets via un couple (*disque, inode*).



La structure inode (cf `<sys/ufs/dinode.h>`) décrit un objet de la façon suivante :

- type de l'objet
- droits d'accès
- nombre de liens ; quand il passe à 0 et que l'objet n'est plus utilisé, l'objet est détruit et la place est récupérée.
- UID, GID
- taille
- dates de dernière modification, consultation etc.
- localisation sur le disque des blocs de l'objet

La structure inode (~128 octets) ne peut pas contenir la liste complète de tous les blocs utilisés. On utilise donc le principe de **liste indirecte** :



Un bloc a pour taille 4 ko (4096 octets). Une adresse de bloc occupe 4 octets.

L'inode contient 13 numéros de blocs :

- les 12 premiers sont les numéros des 12 premiers blocs de données du fichier ;
- le suivant indique le numéro d'un bloc indirect : il contient les numéros des blocs de données à partir du onzième. Il adresse donc 1024 blocs.
- le douzième numéro est le numéro d'un bloc. Celui-ci contient 1024 numéros de blocs indirects, c'est-à-dire que chacun de ces 1024 blocs adressent un bloc de données ;
- le treizième numéro suit cette logique avec une triple indirection.

Ce schéma donne une taille limite de fichiers de :

$$(12 + 1024 + 1024^2 + 1024^3) \times 4096 \text{ octets} = 4\,402\,345\,721\,856 \text{ octets}$$

Le troisième niveau d'indirection est quasi inutilisé car le remplissage des blocs indirects et indirects de niveau 1 et 2 assure déjà une taille de fichiers > à 4 Go.

## § 19.26 Informations sur les inodes : *df*

(en anglais *display free*)

La commande **df** permet de voir le nombre d'inodes utilisées ou libres.

Syntaxe :

Sur LINUX et la plupart des Unix traditionnels : *df -i*

Sur SUN Solaris : *df -e*

## ◇ Exemple sur LINUX :

```
% df -i
Filesystem          Inodes    IUsed    IFree  IUse% Mounted on
/dev/hda3           2280320  318523 1961797   14% /
/dev/hda1            66264     50    66214    1% /boot
none                193904     1   193903    1% /dev/shm
mailhost.example.com:/var/mail
                    7654398  17808 7636590    1% /var/mail
```

## ◇ Exemple sur SUN Solaris :

```
% df -e
Filesystem          ifree
/dev/dsk/c0t0d0s0   442832
/proc               9788
mnttab              0
/dev/dsk/c0t0d0s3   498452
swap                59968
...
```

### § 19.27 Inodes 0, 1, 2 et 3

Points importants dans la logique du fonctionnement du filesystem :

#### Certains directories ont des numéros d'inodes bien précis

```
% /bin/ls -lai /
  2 drwxr-sr-x 22 root          512 Sep 11 15:11 .
  2 drwxr-sr-x 22 root          512 Sep 11 15:11 ..
  3 drwxr-xr-x  2 root        8192 Aug  8 17:38 lost+found
```

Le numéro d'inode 0 sert à marquer les inodes inutilisés.

Le numéro d'inode 1 est réservé.

### § 19.28 Répertoire `lost+found`

(en anglais `lost+found`  $\equiv$  *bureau des objets trouvés*)

Points importants dans la logique du fonctionnement du filesystem :

#### Le directory `lost+found` (inode 3) est créé automatiquement par la commande `newfs`

La taille du directory (vide) est suffisamment grande pour y stocker des numéros d'inodes deconnectés dans des conditions précaires :

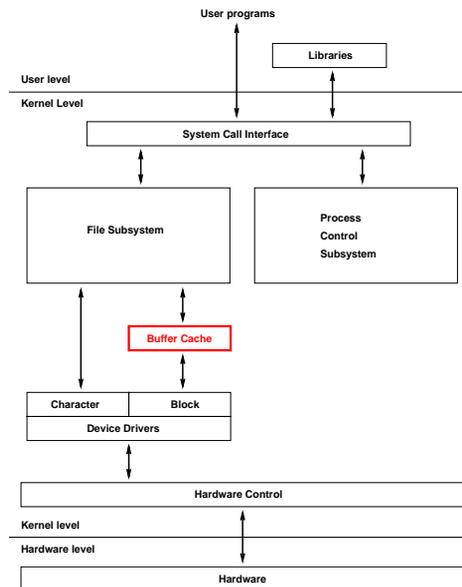
```
% ls -l /lost+found
-rw-----  1 root      sys          0 Mar  1 1996 #155782
drwxr-xr-x  2 mailman  101        512 Jun  2 2000 #091584
```

Il ne faut pas détruire ce directory ni essayer de le reconstruire manuellement par `mkdir`.

Selon le constructeur, il existe ou pas une commande spéciale pour reconstruire ce directory.

## § 19.29 Cohérence des disques : *sync*

Rappel :



Quelques constatations :

1. Les filesystems sont constamment modifiés.
2. Il peut arriver que des informations se perdent en cas de dysfonctionnement du matériel.
3. Il y a le cache buffer.

Le buffer cache conserve des données dont on retarde l'écriture sur disques (exception faite pour les répertoires, les superblocks, etc.).

⇒ besoin de provoquer l'écriture sur disque du contenu du buffer cache.

⇒ commande manuelle **sync** pour cela.

Lors de l'écriture manuelle sur disque du buffer cache, le filesystem est marqué FSCLEAN.

**§ 19.30 Cohérence des disques : `update`**

La commande manuelle **sync** provoque l'écriture sur disque du contenu du buffer cache.

Le système Unix exécute ce mécanisme d'écriture forcée du contenu du buffer cache **toutes les 30 secondes**.

C'est le programme `update` qui réalise cela.

Autres noms possibles : `syncher`

Lors de l'écriture chronique sur disque du buffer cache, le filesystem est marqué FSCLEAN.

**§ 19.31 Vérification de la cohérence des disques : `fsck`**

La perte du contenu du buffer cache en cas de dysfonctionnement peut conduire à effectuer des réparations par la commande **fsck (File System Check Consistency)**.

Lors du boot d'une machine, si un filesystem est marqué FSCLEAN, `fsck` ne vérifie pas ce filesystem. On se trouve dans ces conditions si la machine a été arrêtée ou redémarrée en utilisant des commandes comme `fastboot`, `fasthalt` etc.

Plusieurs vérifications sont effectuées par *fsck* :

- le super-block contient le bon nombre de blocs libres ;
- le super-block contient le bon nombre d'inodes libres ;
- le super-block contient la bonne taille de la partition ;
- un bloc de données appartient à un et un seul fichier, ou à la liste des blocs libres ;
- un inode, s'il est référencé  $n$  fois dans des répertoires, a bien  $n$  liens physiques ;
- un inode est soit utilisé, soit dans la liste des inodes libres ;
- la taille du fichier correspond au nombre de blocs alloués ;
- tout répertoire contient une entrée pour `.` et `..` ;
- une entrée dans un répertoire doit référencer un inode non libre.

Exemple quand tout se passe bien :

```
# fsck /dev/rsd2g
** /dev/rsd2g
** Last Mounted on /users
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
CLEAN FLAG NOT SET IN SUPERBLOCK
FIX? [yn] y

23423 files, 1174684 used, 2946897 free (6641 frags, 367532 blocks,
0.2% fragmentation)

***** FILE SYSTEM WAS MODIFIED *****
```

Exemple en cas de problème :

```
Automatic reboot in progress...
/dev/rsd0a: clean, 234610 free (9626 frags, 28123 blocks, 2.6% fragmentation)
/dev/rsd10c: clean, 842274 free (386 frags, 105236 blocks, 0.0% fragmentation)
/dev/rsd12c: clean, 470238 free (822 frags, 58677 blocks, 0.1% fragmentation)
/dev/rsd11c: clean, 155358 free (1406 frags, 19244 blocks, 0.1% fragmentation)
/dev/rccd0c: clean, 3642368 free (15392 frags, 453372 blocks, 0.2% fragmentation)
cannot alloc 8340482 bytes for lncntp
/dev/rccd1c: CAN'T CHECK FILE SYSTEM.
/dev/rccd1c: UNEXPECTED INCONSISTENCY; RUN fsck MANUALLY.
/dev/rccd2c: clean, 1759345 free (10417 frags, 218616 blocks, 0.1% fragmentation)
cannot alloc 10629122 bytes for lncntp
/dev/rsd5c: CAN'T CHECK FILE SYSTEM.
/dev/rsd5c: UNEXPECTED INCONSISTENCY; RUN fsck MANUALLY.
THE FOLLOWING FILE SYSTEMS HAD AN UNEXPECTED INCONSISTENCY:
    /dev/rccd1c (/people), /dev/rsd5c (/hold)
Automatic file system check failed... help!
Enter root password, or ^D to go multi-user
```

Syntaxe :

```
fsck [options] [fichier-spécial]
```

Options intéressantes :

- option `-y` : Répond "yes" à toutes les questions que posent `fsck`
- option `-n` : Répond "no" à toutes les questions que posent `fsck`
- option `-p` : Répond "yes" à toutes les questions anodines que posent `fsck`. C'est ce qui est utilisé au moment du boot.

Quelques règles :

- Ne jamais lancer `fsck` sur une partition active, montée car cela peut conduire à un état instable du filesystem.
- Si possible, lancer `fsck` en étant en single user.
- En cas de `fsck` sur `/`, redémarrer juste après sans sauver le buffer cache.

Présence d'un fsck là où l'on ne s'y attend pas toujours : démarrage d'un switch CISCO récent :

```
C2950 Boot Loader (C2950-HBOOT-M) Version 12.1(11r)EA1, RELEASE SOFTWARE (fc1)
Compiled Mon 22-Jul-02 17:18 by antonino
WS-C2950-24 starting...
Base ethernet MAC Address: 00:0b:46:eb:9b:c0
Xmodem file system is available.
Initializing Flash...
flashfs[0]: 17 files, 2 directories
flashfs[0]: 0 orphaned files, 0 orphaned directories
flashfs[0]: Total bytes: 7741440
flashfs[0]: Bytes used: 3961856
flashfs[0]: Bytes available: 3779584
flashfs[0]: flashfs fsck took 6 seconds.
...done initializing flash.
Boot Sector Filesystem (bs:) installed, fsid: 3
Parameter Block Filesystem (pb:) installed, fsid: 4
Loading "flash:/c2950-i6q4l2-mz.121-11.EA1.bin" ..#####
#####
#####
#####
#####
#####
#####
#####

File "flash:/c2950-i6q4l2-mz.121-11.EA1.bin" uncompressed and installed, entry
point: 0x80010000
executing...
```

## § 19.32 Stratégie d'allocation des blocs, MINFREE, *tunefs*

La stratégie d'allocation des blocs au sein d'un filesystem ne fonctionne que si le disque contient un espace libre raisonnable. Cet espace est appelé **minfree**. La réserve est de 10% d'ordinaire :

```
% df -t ufs /tmp
Filesystem 1K-blocks      Used    Avail Capacity  Mounted on
/dev/sd0s1f  127151        22843   94136     20%    /tmp
```

Ceci explique que l'on puisse remplir une partition à plus de 100% :

```
# df -t ufs /mnt
Filesystem 1K-blocks      Used    Avail Capacity  Mounted on
/dev/sd0s1d  1702685    1638403   -71932    105%    /mnt
```

La commande **tunefs** permet de régler ce pourcentage :

```
tunefs -m minfree raw-filesystem
(démonter le filesystem avant le tunefs puis le remonter après)
```

**§ 19.33 Vitesse de rotation du disque**

Humour :

Newsgroup: rec.humor.funny  
From: pete@mudhead.uottawa.ca  
Subject: Why do USENET articles disappear  
Date: Wed, 9 Aug 95 4:30:06 EDT  
In article <3u6moi\$sn2@newsbf02.news.aol.com>, foo <bar@aol.com> wrote :

*I have noticed that some postings are disappearing faster than others. Why is that and who decides ?*

Its the size of the article. You see, the articles are stored on disk. An article is encoded in things known as 'bits' which are written on the disk. A disk is a rotating platter. As anyone knows centrifugal force will force anything off of a rotating surface.

As time goes on, the article moves closer and closer to the edge of the disk, and finally, it flies right off. Of course, the larger articles (more bits == more weight) tend to fly off faster.

Unix systems have something known as a "sticky bit" which can help articles remain longer if it is set. Remember, USENET was originally set up by Unix people, and they knew what they were doing.

On the other hand, some systems have their own rules, and you might get a better (or at least more correct) answer by asking the people who administer your machine.

**§ 19.34 Snapshots**

Snapshot ≡ cliché instantané du filesystem (terme de photographie)

Le snapshot montre un filesystem figé alors que le filesystem continue de se modifier.

Avantages :

- snapshot rapide à construire
- backup d'une image stable d'un filesystem
- récupération facile et simple d'un fichier

Disponible à partir de Solaris 8 (commande `fs_snap`)

Mais ne subsiste pas après un reboot. . .

Disponible à partir de FreeBSD 5 (option de `/etc/fstab` : `snapshot`)

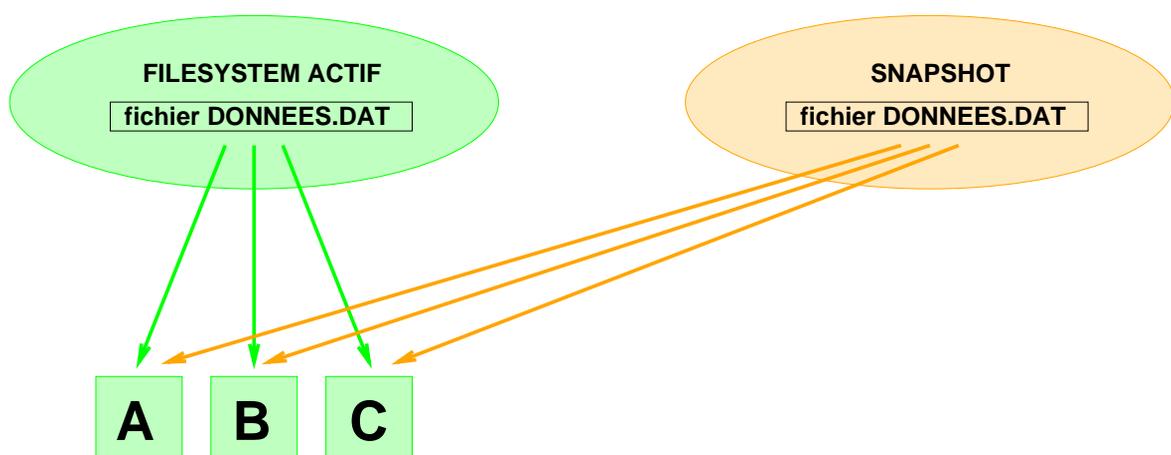
Mais plante le noyau d'une machine multiprocesseur en 5.0. . .

Disponible à partir de Linux si on utilise un Logical Volume Manager.

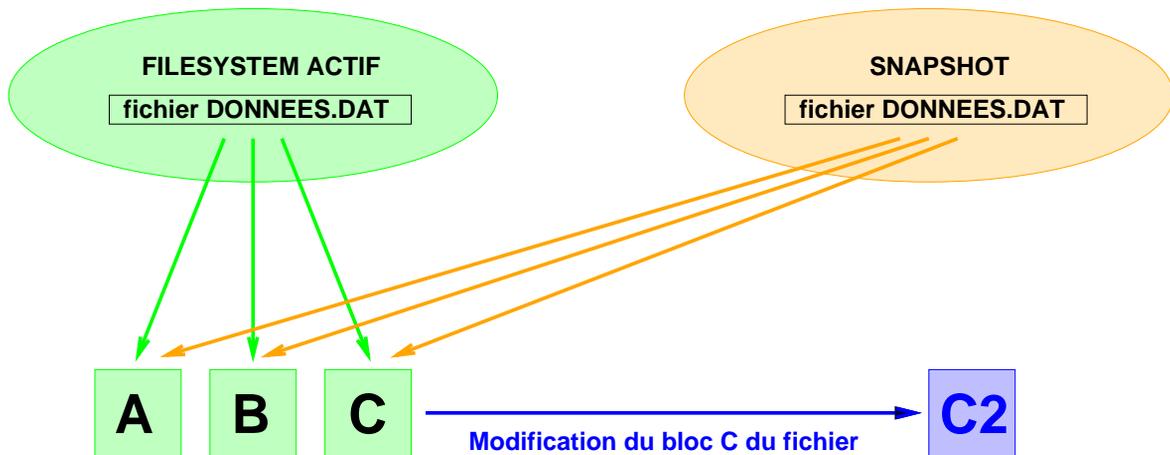
Disponible sur les machines spécialisées en stockage de la marque Network Appliance ;

<http://www.netapp.com>

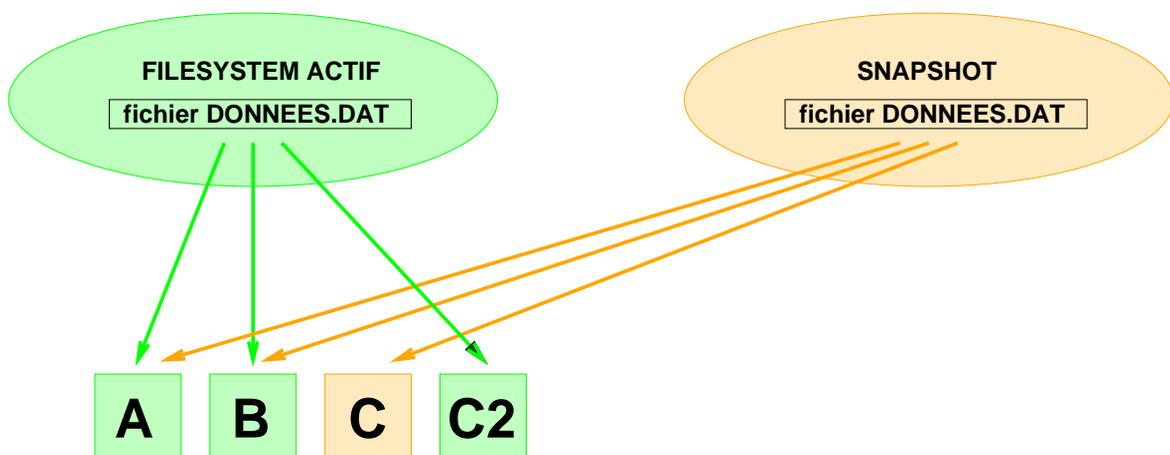
Principe : on construit le snapshot en notant les blocs disques utilisés.



Le filesystem peut continuer à évoluer. Un fichier modifie ainsi ses blocs.



Le filesystem est conçu pour travailler conjointement avec les snapshots. Il va dupliquer le bloc que l'on veut modifier car le filesystem sait que ce bloc est associé à un snapshot. Il modifie la copie du bloc. Le snapshot pointe toujours sur les mêmes blocs.



**Nécessité de gérer le nombre et la durée de vie des snapshots !**

### § 19.35 (Windows : : Snapshots)

Disponible en Windows 2003 server : *Volume Shadow-Copy Service* dit VSS

VSS :

- serveur de fichiers Windows 2003 server
- clients Windows XP ou Windows 2003 server
- logiciel Shadow-Copy sur les clients

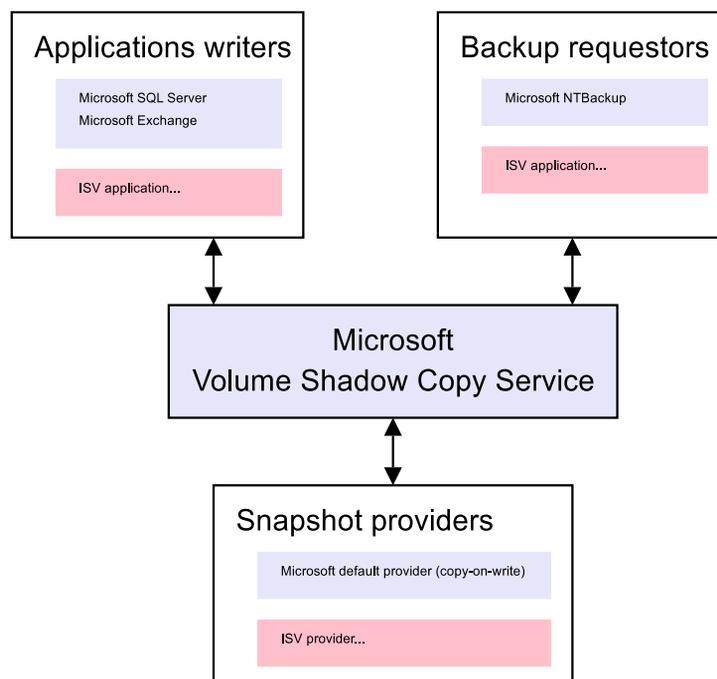
cf

[http://www.laboratoire-microsoft.org/articles/win/wolume\\_shadow\\_copy/0/](http://www.laboratoire-microsoft.org/articles/win/wolume_shadow_copy/0/)

Approche différente des systèmes Unix :

- sur Unix : le snapshot est fait indépendamment de l'application ; un snapshot d'une base de données s'il est invalide va conduire la base de données à faire une réparation du snapshot ⇒ délai de retour à la normale...
- sur Windows : le snapshot sera toujours « sain » ; une application pourra se fier à un snapshot et l'utiliser comme point de reprise immédiat.

Schéma de principe de VSS :



◇ Application Writers

Les applications sont les utilisatrices des snapshots car elles savent mieux que le filesystem les fichiers et l'importance des fichiers qu'elles utilisent.

Il faut donc rendre une application compatible avec VSS moyennant un kit de développement Microsoft, « Microsoft snapshot SDK » (disponible en NDA).

Les applications ainsi modifiées sont des *Snapshot Writers* :

L'application sera responsable de retenir l'activité en écriture le temps de tenter la création du snapshot.

- Si l'application n'a pas pu retenir le flux d'écriture, alors le snapshot n'est pas sain mais l'application le sait.
- Si le flux d'écriture a pu être retenu, le snapshot est sain et l'application le sait.

◇ Backup requestors

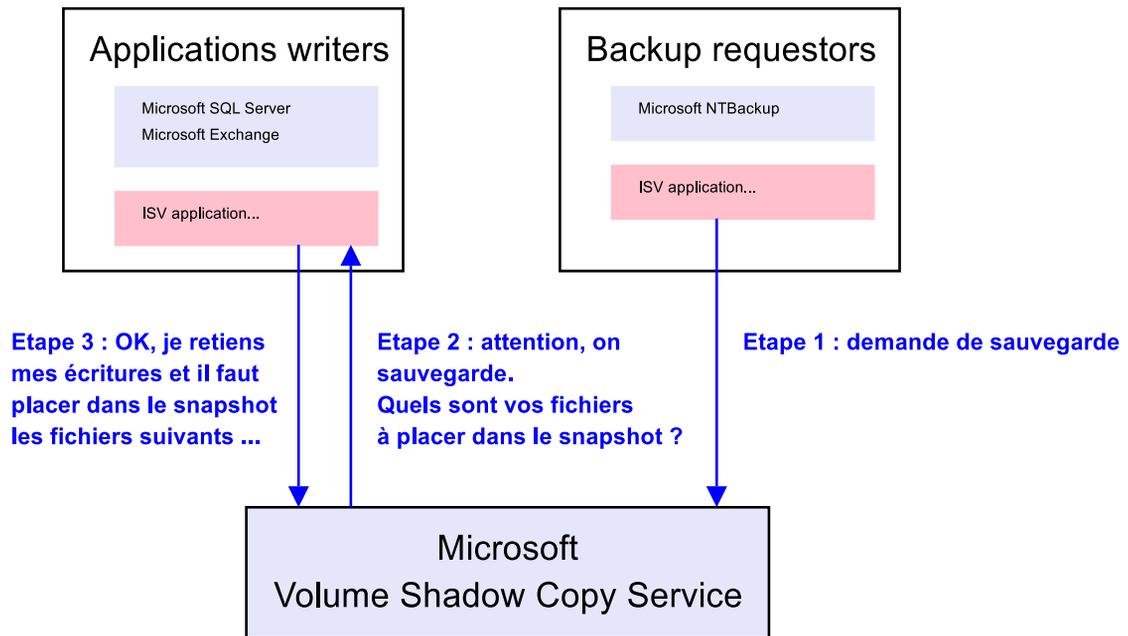
Des programmes externes comme NTBackup . exe peuvent demander la sauvegarde ou la restauration de la machine.

Ces programmes de « backup » doivent être modifiés au moyen du « Microsoft snapshot SDK » pour être rendus compatibles avec VSS.

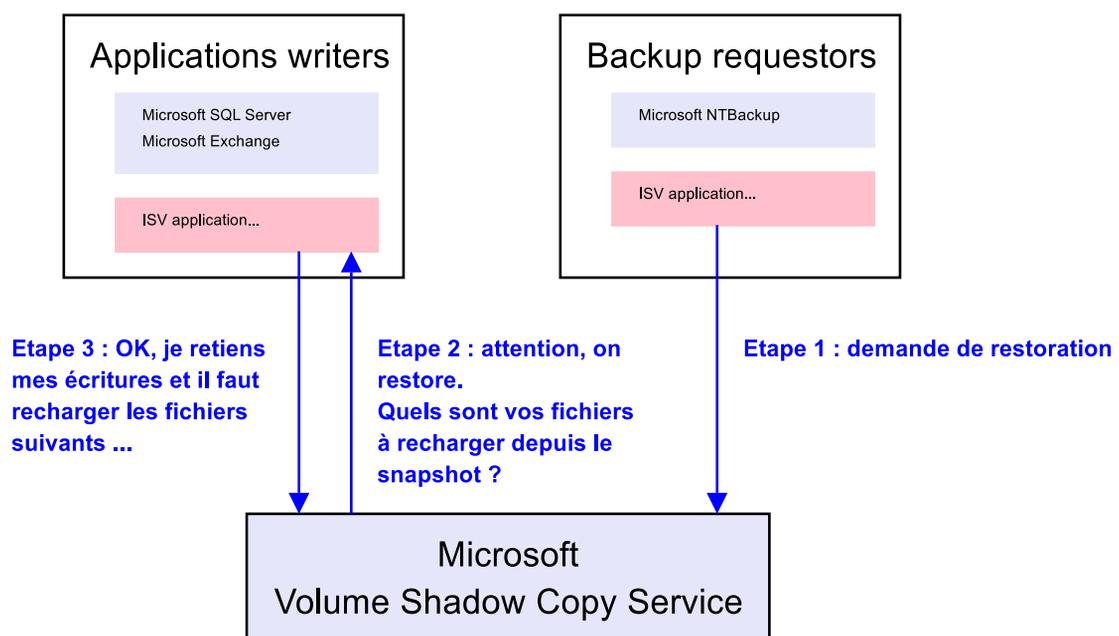
Principe de la sauvegarde : voir pages suivantes

Principe de la restauration : voir pages suivantes

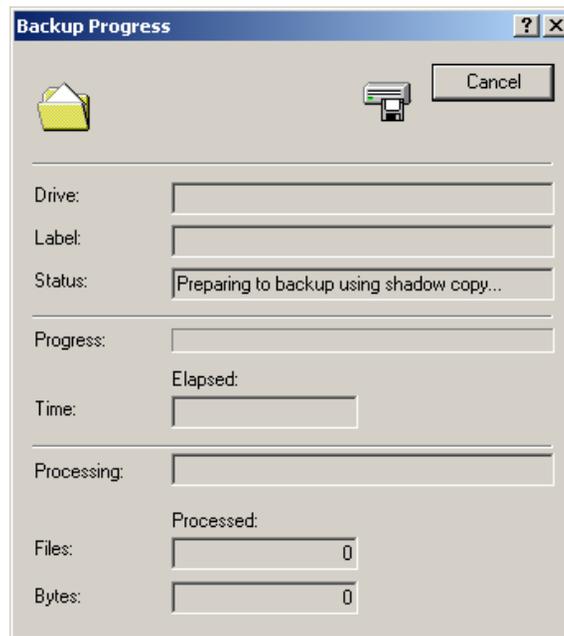
Principe de la sauvegarde :



Principe de la restauration :



Mention de VSS dans une fenêtre transitoire de NTBackup . exe de Windows XP :



### § 19.36 Logical Volume Managers, LVM

Par exemple sur les Unix IBM AIX, SUN Solaris, COMPAQ Tru64 Unix

LVM  $\equiv$  gestion dynamique des filesystems :

- création dynamique de filesystems
- supporte davantage que les 8 partitions classiques d'un seul disque dur
- redimensionnement dynamique de filesystems actifs
- suppression de filesystems
- ajout de disques online

Un LVM est un avantage.

Un LVM est délicat à manipuler.

Un LVM est spécifique à chaque Unix.

### § 19.37 Filesystem journalisé

La commande `fsck` peut se révéler non pratique :

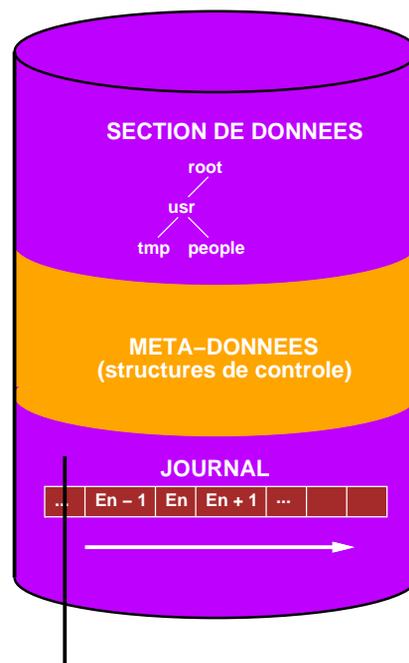
- lorsque l'on a beaucoup de disques
- lorsque les disques sont gros

⇒ solution du filesystem journalisé :

- garantit la fiabilité des données
- grande rapidité de recouvrement lors d'un crash machine grâce à la lecture complète des opérations stockées dans un journal

Le journal correspond à une zone disque réservée dans laquelle sont rangées séquentiellement toutes les mises à jour de données, et ce, avant leur mise à jour effective dans le système de fichiers.

Lors d'un crash machine, la relecture du journal permet de reconstituer la totalité des systèmes de fichiers en quelques secondes. Il n'y a plus besoin de recourir à `fsck`.



**Point de sauvegarde**  
**Point de départ pour reexecuter les actions effectuees avant le crash**

## § 19.38 Gravure de CD/DVD (1) : image ISO 9660

Un CDR ou CDRW peut être mono-session ou multi-session.

Il ne sera envisagé ici que le cas de gravure mono-session.

Un graveur CDR dit 1X grave la capacité de 650 Mo en 1 heure.

Appliquer une règle de trois pour un lecteur N X.

La gravure de CD/DVD se fait en deux étapes sous Unix :

1. construction d'une image de l'arborescence à graver ; le format est dit ISO 9660 avec les extensions Rock Ridge pour gérer les noms longs des fichiers et des répertoires ainsi que des choses spéciales comme les liens symboliques
2. gravure de l'image iso

La commande la plus connue pour faire cela est « `mkisofs` », disponible à l'URL

`ftp://ftp.berlios.de/pub/cdrecord/` dans le package « `cdrtools` ».

### ◇ Exemple 1

On veut une image ISO utilisable sur plateforme Unix et PC Windows.

⇒ employer les options « `-r` » et « `-J` » :

```
% mkisofs -r -J -o image.iso arborescence
Using UNIX_000.EPS;1 for /unix-fork.eps (unix-fork-exec.eps)
Using VANGO000.;1 for arborescence/poubelle/vangogh-ps-ef (vangogh-ps-e)
Using TASKM000.EPS;1 for arborescence/windows/taskmgr-07.eps (taskmgr-06.eps)
Using TASKM001.EPS;1 for arborescence/windows/taskmgr-06.eps (taskmgr-05.eps)
Using TASKM002.EPS;1 for arborescence/windows/taskmgr-05.eps (taskmgr-03.eps)
Using TASKM003.EPS;1 for arborescence/windows/taskmgr-03.eps (taskmgr-02.eps)
Using TASKM004.EPS;1 for arborescence/windows/taskmgr-02.eps (taskmgr-01.eps)
Using TASKM000.GIF;1 for arborescence/windows/taskmgr-07.gif (taskmgr-06.gif)
Using TASKM001.GIF;1 for arborescence/windows/taskmgr-06.gif (taskmgr-05.gif)
Using TASKM002.GIF;1 for arborescence/windows/taskmgr-05.gif (taskmgr-03.gif)
Using TASKM003.GIF;1 for arborescence/windows/taskmgr-03.gif (taskmgr-02.gif)
Using TASKM004.GIF;1 for arborescence/windows/taskmgr-02.gif (taskmgr-01.gif)
Using CHROO000.DIA;1 for arborescence/chroot/chroot.dia~ (chroot.dia)
 61.28% done, estimate finish Sat Oct 23 13:03:57 2004
Total translation table size: 0
Total rockridge attributes bytes: 4284
Total directory bytes: 10240
Path table size(bytes): 64
Max brk space used 10000
8184 extents written (15 Mb)
```

## ◇ Exemple 2

On veut une image ISO à partir d'un CDROM que l'on a (CDROM de fichiers informatiques, pas CD musical).

⇒ employer la commande « dd » :

```
% dd if=/dev/cdrom of=/tmp/image.iso bs=128k
```

## ◇ Exemple 3

**ATTENTION : pour des backups système, ne pas utiliser l'option -r mais lui préférer l'option -R. Sinon les droits enregistrés seront incorrects et ne refléteront pas les vrais droits.**

Soit des fichiers protégés :

```
% ls -l
drwxr-xr-x  2 besancon adm          512 Sep 27 19:18 backup/
drwx----- 6 pgsq1    pgsq1      512 Oct 23 11:20 data/
```

Si l'on fait :

```
# mkisofs -r -J -o /tmp/image.iso .
```

on obtient après gravure un CDROM où les droits et les propriétaires des fichiers sont :

```
% ls -l /mnt/cdrom
dr-xr-xr-x  2 root      root          2048 Sep 27 19:18 backup
dr-xr-xr-x  6 root      root          2048 Oct 23 11:20 data
```

Tous les fichiers dans data sont lisibles!!!

Tous les fichiers sont propriété de root!!!

⇒ Ce n'est pas une sauvegarde « système » réutilisable.

Si l'on fait :

```
# mkisofs -R -J -o /tmp/image.iso .
```

on obtient après gravure un CDROM où les droits et les propriétaires des fichiers sont :

```
% ls -l /mnt/cdrom
```

```
drwxr-xr-x  2 besancon adm          2048 Sep 27 19:18 backup
drwx----- 6 pgsq1    pgsq1       2048 Oct 23 11:20 data
```

Tous les fichiers ont conservé leurs droits originaux. La sauvegarde « système » est utilisable.

### § 19.39 Gravure de CD/DVD (2) : manipulation d'une image ISO 9660

On peut vouloir vérifier le contenu d'une image ISO avant de la graver.

Certains systèmes Unix permettent de « monter une image ISO 9660 ».

Avantages :

- Permet de tester une image avant de la graver peut-être pour rien
- Gain de temps si l'on peut monter l'image ISO car on évite la phase de gravure sur le support CDROM, support altérable (rayures, cassabilité, etc.)
- L'image ISO n'est pas rayable, cassable

Défauts :

- L'image ISO peut-être corrompue lors d'un transfert réseau par exemple. Nécessité de faire un checksum de l'image et de la vérifier.

◇ Montage d'une image ISO 9660 sous LINUX

```
# mount -o loop -t iso9660 /var/tmp/image.iso /mnt/iso
# ls /mnt/iso
20041013          factures          pgsql
__THIS_IS_SAE5_DATA__  html             rr_moved
cisco            pamo             shares
# umount /mnt/iso
```

◇ Montage d'une image ISO 9660 sous SOLARIS

Sur SOLARIS, il faut :

1. associer l'image ISO avec un device de type block ⇒ commande `lofiadm`
2. monter ensuite le device de type block ⇒ commande `mount` traditionnelle
3. démonter le device de type block après utilisation ⇒ commande `umount` traditionnelle
4. dissocier le device de type block ⇒ commande `lofiadm`

Exemple d'une session complète page suivante.

Exemple d'une session complète :

```
# lofiadm -a /var/tmp/image.iso
/dev/lofi/1
# mount -r -F hsfs /dev/lofi/1 /mnt/iso
# ls /mnt/iso
20041013          factures          pgsql
__THIS_IS_SAE5_DATA__  html             rr_moved
cisco            pamo             shares
# umount /mnt/iso
# lofiadm -d /dev/lofi/1
```

#### ◇ Montage d'une image ISO 9660 sous FREEBSD

Sur FREEBSD, il faut :

1. associer l'image ISO avec un device de type block ⇒ commande `mdconfig` (anciennement `vnconfig`)
2. monter ensuite le device de type block ⇒ commande `mount` traditionnelle
3. démonter le device de type block après utilisation ⇒ commande `umount` traditionnelle
4. dissocier le device de type block ⇒ commande `mdconfig` (anciennement `vnconfig`)

Exemple d'une session complète page suivante.

Exemple d'une session complète :

```
# mdconfig -a -t vnode -f /var/tmp/image.iso
md0
# mount -t cd9660 /dev/md0 /mnt/iso
# df -k /mnt/iso
Filesystem 1K-blocks  Used Avail Capacity  Mounted on
/dev/md0      354240 354240    0   100%   /mnt/iso
# ls /mnt/iso
20041013          factures          pgsql
__THIS_IS_SAE5_DATA__  html             rr_moved
cisco            pamo             shares
# umount /mnt/iso
# mdconfig -d -u md0
```

#### ◇ Montage d'une image ISO 9660 sous WINDOWS

On peut monter une image ISO 9660 via le logiciel gratuit DAEMONTOOLS disponible à l'URL <http://www.daemon-tools.cc>.

**§ 19.40 Gravure de CD/DVD (3) : gravure d'une image ISO 9660**

La gravure d'une image ISO 9660 est une opération spécifique à chaque OS.

◇ Gravure d'une image ISO sous LINUX

La gravure d'une image ISO sous LINUX se fait par la commande suivante :

- pour graveur IDE : commande non standard : `cdrecord` disponible à l'URL  
`http://www.fokus.fhg.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html`.
- logiciel avec interface graphique : K3B, site officiel `http://www.k3b.org`

◇ Gravure d'une image ISO sous SOLARIS

La gravure d'une image ISO sous LINUX se fait par les commandes suivantes :

- pour graveur CDRW IDE : commande fournie par SOLARIS : `cdwr`
- pour graveur DVD+-RW IDE : commande fournie par SOLARIS : `cdwr` (selon niveau de patch)
- pour graveur CDRW SCSI (de plus en plus rare à trouver) : commande non fournie par SOLARIS :  
`cdrecord` disponible à l'URL  
`http://www.fokus.fhg.de/research/cc/glone/employees/joerg.schilling/private/cdrecord.html`.

Liste des graveurs IDE et SCSI : `cdrw -l`

Exemple :

```
# cdrw -l
Looking for CD devices...
      Node                               Connected Device                               Device type
-----+-----+-----+-----+-----+-----+-----+-----+-----+
/dev/rdsk/c0t1d0s2 | _NEC      DVD_RW ND-2500A  1.06 | CD Reader/Writer
/dev/rdsk/c1t5d0s2 | HP        CD-Writer+ 9200  1.0e | CD Reader/Writer
```

Liste des graveurs SCSI : `cdrecord -scanbus`

Exemple :

```
# cdrecord -scanbus
Cdrecord 1.9 (sparc-sun-solaris2.9) Copyright (C) 1995-2000 Jörg Schilling
Using libscg version 'schily-0.1'
scsibus0:
    0,0,0    0) *
    0,1,0    1) *
    0,2,0    2) *
    0,3,0    3) 'SEAGATE ' 'ST318436LW      ' '0010' Disk
    0,4,0    4) *
    0,5,0    5) 'HP          ' 'CD-Writer+ 9200 ' '1.0e' Removable CD-ROM
    0,6,0    6) *
    0,7,0    7) HOST ADAPTOR
```

Effacement d'un CDRW sur un graveur IDE : `cdwr -b all`

Exemple :

```
# cdwr -b -v -d /dev/rdisk/c0t1d0s2 session
Initializing device...done.
Blanking the media (Can take several minutes)...done.

# cdwr -b -v -d /dev/rdisk/c0t1d0s2 all
Blanking the media (Can take several minutes)...done.
```

Effacement d'un CDRW sur un graveur SCSI : `cdrecord blank=fast`

Exemple :

```
# cdrecord blank=fast
Cdrecord 1.9 (sparc-sun-solaris2.9) Copyright (C) 1995-2000 Jörg Schilling
scsidev: '0,5,0'
scsibus: 0 target: 5 lun: 0
Using libscg version 'schily-0.1'
Device type      : Removable CD-ROM
Version          : 4
Response Format   : 2
Capabilities     : SYNC
Vendor_info      : 'HP      '
Identifikation   : 'CD-Writer+ 9200 '
Revision         : '1.0e'
Device seems to be: Generic mmc CD-RW.
Using generic SCSI-3/mmc CD-R driver (mmc_cdr).
Driver flags     : SWABAUDIO
cdrecord: Drive needs to reload the media to return to proper status.
Starting to write CD/DVD at speed 4 in write mode for single session.
Last chance to quit, starting real write in 1 seconds.
```

Exemple de gravure d'une image ISO sur un graveur IDE :

```
# cdrw -i -v -d /dev/rdsk/c0t1d0s2 image.iso
Initializing device...done.
Writing track 1...12 %
Writing track 1...done.
Finalizing (Can take several minutes)...done.
```

Exemple de gravure d'une image ISO sur un graveur SCSI :

```
# cdrecord image.iso
Cdrecord 1.9 (sparc-sun-solaris2.9) Copyright (C) 1995-2000 Jörg Schilling
scsidev: '0,5,0'
scsibus: 0 target: 5 lun: 0
Using libscg version 'schily-0.1'
Device type      : Removable CD-ROM
Version         : 4
Response Format  : 2
Capabilities    : SYNC
Vendor_info     : 'HP      '
Identifikation  : 'CD-Writer+ 9200 '
Revision       : '1.0e'
Device seems to be: Generic mmc CD-RW.
Using generic SCSI-3/mmc CD-R driver (mmc_cdr).
Driver flags    : SWABAUDIO
Starting to write CD/DVD at speed 8 in write mode for single session.
Last chance to quit, starting real write in 1 seconds.
Track 01: Total bytes read/written: 660957184/660957184 (322733 sectors).
```

◇ Gravure d'une image ISO sous FREEBSD

La gravure d'une image ISO sous LINUX se fait par la commande suivante :

- pour graveur CDRW IDE : commande fournie par FREEBSD : `burncd`
- pour graveur DVD+-RW IDE : commande fournie par FREEBSD : `burncd`

Effacement d'un CDRW : `burncd blank`

Exemple :

```
# burncd blank
blanking CD - 100 % done
```

Gravure d'une image ISO 9660 en mono-session : `burncd data image.iso fixate`

Exemple :

```
% burncd data z.iso fixate
next writeable LBA 0
writing from file z.iso size 354240 KB
written this track 354240 KB (100%) total 354240 KB
fixating CD, please wait..
```

#### ◇ Gravure d'une image ISO sous WINDOWS

La gravure d'une image ISO sous WINDOWS se fait par les commandes suivantes :

- pour graveur CDRW IDE : commande fournie par MICROSOFT dans le resource kit Windows 2003 : `cdburn.exe`
- pour graveur DVD+-RW IDE : commande fournie par MICROSOFT dans le resource kit Windows 2003 : `dvdburn.exe`
- pour graveur CDRW IDE : commande non standard : BURNCDCC disponible sur <http://www.terabyteunlimited.com/downloads/burncdcc.zip> (version 1.03)
- pour graveur DVD+-RW IDE : commande non standard : BURNCDCC disponible sur <http://www.terabyteunlimited.com/downloads/burncdcc.zip> (version 1.03)

Gravure d'une image ISO 9660 en mono-session :

```
cdburn.exe drive -erase [image [options]] ou  
cdburn.exe drive image [-speed max]
```

Exemple :

```
C:\Documents and Settings\root\Desktop>cdburn e: -erase  
Erasing media before burning  
Number of blocks in ISO image is ffffffff  
Erasing target media  
Media erased
```

```
C:\Documents and Settings\root\Desktop>cdburn e: cd040818.iso -speed max  
Requesting burn at maximum speed  
Number of blocks in ISO image is 613  
Finished Writing  
Synchronizing Cache: Error ejecting/reinserting disc
```

### § 19.41 Droits étendus : Access Control Lists (ACL)

Utilité :

- 5 formations
- 2 projets par formation
- 20 élèves par formation
- travail en binôme

Au total :  $5 * 2 * 20 / 2 = 100$  groupes Unix à gérer que seul l'administrateur peut gérer et que le modèle des droits Unix rendra très difficile à mettre en place.

⇒ création d'un autre mécanisme connu sous le nom de *Access Control Lists* ou ACL.

Disponible sur LINUX, SOLARIS, FREEBSD, WINDOWS

Attention aux différentes implémentations non compatibles des ACL. L'unification des différentes implémentations n'a pas encore eu lieu en pratique.

**§ 19.42 ACL sous LINUX**

Le mécanisme des ACL sous LINUX nécessite :

- un noyau 2.4 plus des patches
- un noyau 2.6 standard
- un filesystem supportant les ACL (ext2, ext3, jfs, xfs)

◇ Activation du mécanisme des ACL

Il faut indiquer l'option « `acl` » au niveau du fichier « `/etc/fstab` » ou de la commande « `mount` » :

```
# mount -o acl /dev/hda2 /mnt/exemple
# mount
```

**A completer...**

◇ Commande pour lister les ACL : `getfacl`

(en anglais *get file ACL*)

Syntaxe : `getfacl [options] objets`

Exemple :

XXXXX

**A completer...**

◇ Commande pour manipuler les ACL : `setfacl`

(en anglais *set file ACL*)

Syntaxe : `setfacl [options] objets`

Exemple :

XXXXX

**A completer...**

◇ Attention

**Il faut vérifier que les commandes de manipulation des fichiers fonctionnent correctement avec les ACL.**

mv, cp, rm, vi, ...

Patches sur <http://acl.bestbits.at>

◇ A propos de tar

A priori le tar de LINUX ne supporte pas les ACL.

2 alternatives :

– lors d'un backup faire :

– « getfacl -R partition > /tmp/acl.txt »

– « tar cvf archive.tar partition »

et lors de la restauration faire :

– « tar xvf archive.tar »

– « setfacl --restore /tmp/acl.txt »

– utiliser la commande star disponible à l'adresse

<http://www.fokus.gmd.de/research/cc/glone/employees/joerg.schilling/private/star.html>

◇ ACL et partage réseau NFS

- patch pour NFS v3 sur <http://www.bestbits.at>
- NFS v4 OKr <http://www.bestbits.at>

**§ 19.43 ACL sous SOLARIS**◇ Activation du mécanisme des ACL

Le mécanisme des ACL sous SOLARIS est toujours activé.

◇ Commande pour lister les ACL : getfacl

(en anglais *get file ACL*)

Syntaxe : `getfacl [options] objets`

Exemple : fichier sans ACL

Rien de visible au niveau de `ls` :

```
% /bin/ls -l
total 2
-rw-r--r--  1 besancon adm          49 Oct 31 14:43 exemple.txt
```

Au niveau de `getfacl` :

```
% getfacl exemple.txt

# file: exemple.txt
# owner: besancon
# group: adm
user::rw-
group::r--          #effective:r--
mask:r--
other:r--
```

Exemple : fichier avec ACL

Au niveau de `ls` : le symbole « + » indique la présence d'ACL :

```
% ls -l
total 4
-rw-r--r--+  1 besancon adm          49 Oct 31 14:43 exemple.txt
-rw-r--r--  1 besancon adm          449 Oct 31 14:45 log
```

Au niveau de `getfacl` :

```
% getfacl exemple.txt

# file: exemple.txt
# owner: besancon
# group: adm
user::rw-
user:mysql:rw-      #effective:rw-
group::r--          #effective:r--
mask:rw-
other:r--
```

◇ Commande pour manipuler les ACL : setfacl(en anglais *set file ACL*)Syntaxe : `setfacl [options] objets`

Exemple :

```
% setfacl -m user:mysql:rw- exemple.txt
% getfacl exemple.txt

# file: exemple.txt
# owner: besancon
# group: adm
user::rw-
user:mysql:rw-          #effective:r--
group::r--              #effective:r--
mask:r--
other:r--
```

A cause du mask par défaut, l'utilisateur `mysql` n'aura pas son droit en écriture même si on l'a mémorisé dans l'ACL. Cf l'indication « `#effective:r--` » de la ligne « `user:mysql:rw-` ».  
⇒ modification du mask à faire pour autoriser les droits donnés individuellement via les ACL.

Exemple avec modification du mask à faire pour autoriser les droits donnés individuellement via les ACL.

```
% setfacl -m mask:rw- exemple.txt
% setfacl -m user:mysql:rw- exemple.txt
% getfacl exemple.txt

# file: exemple.txt
# owner: besancon
# group: adm
user::rw-
user:mysql:rw-          #effective:rw-
group::r--              #effective:r--
mask:rw-
other:r--
```

◇ Attention

Sur LINUX, il faut vérifier que les commandes de manipulation des fichiers fonctionnent correctement avec les ACL.

*Commentaire personnel : pas terrible car manifestement les ACL ne sont pas intégrés au système.*

Sur SOLARIS, les commandes fournies par le système intègrent le support des ACL.

Exemple : commande `cp` compatible avec les ACL

```
% /bin/ls -l
total 2
-rw-r--r--+ 1 besancon adm          49 Oct 31 14:43 exemple.txt
% cp exemple.txt exemple2.txt
% ls -l
total 4
-rw-r--r--+ 1 besancon adm          49 Oct 31 14:43 exemple.txt
-rw-r--r--+ 1 besancon adm          49 Oct 31 15:13 exemple2.txt
```

Exemple : commande `mv` compatible avec les ACL

```
% /bin/ls -l
total 2
-rw-r--r--+ 1 besancon adm          49 Oct 31 14:43 exemple.txt
% mv exemple2.txt exemple3.txt
% ls -l
total 4
-rw-r--r--+ 1 besancon adm          49 Oct 31 15:13 exemple3.txt
```

Exemple : démonstration de l'effet de l'ACL

```
% id
uid=6000(mysql) gid=6000(mysql)
% cat /etc/hosts > exemple.txt
% ls -l
total 8
-rw-r--r--+ 1 besancon adm          2408 Oct 31 15:17 exemple.txt
```

◇ A propos de tar

La commande `tar` de SOLARIS supporte les ACL via l'option « `-p` ».

Sauvegarde des fichiers avec leurs ACL dans une archive TAR :

```
% tar cvfp /tmp/archive.tar exemple.txt
a exemple.txt 1K
```

On retrouve bien dans l'archive TAR la présence d'ACL :

```
% strings /tmp/archive.tar | grep mysql
user::rw-,user:mysql:rw-,group::r--,mask:rw-,other:r--
```

Restoration des fichiers avec leurs ACL à partir de l'archive TAR :

```
% cd /chemin/vers/ailleurs
% tar xvpf /tmp/archive.tar
tar: blocksize = 6
x exemple.txt, 49 bytes, 1 tape blocks

% ls -l
total 2
-rw-r--r--+ 1 besancon adm          49 Oct 31 14:43 exemple.txt

% getfacl exemple.txt

# file: exemple.txt
# owner: besancon
# group: adm
user::rw-
user:mysql:rw-          #effective:rw-
group::r--              #effective:r--
mask:rw-
other:r--
```

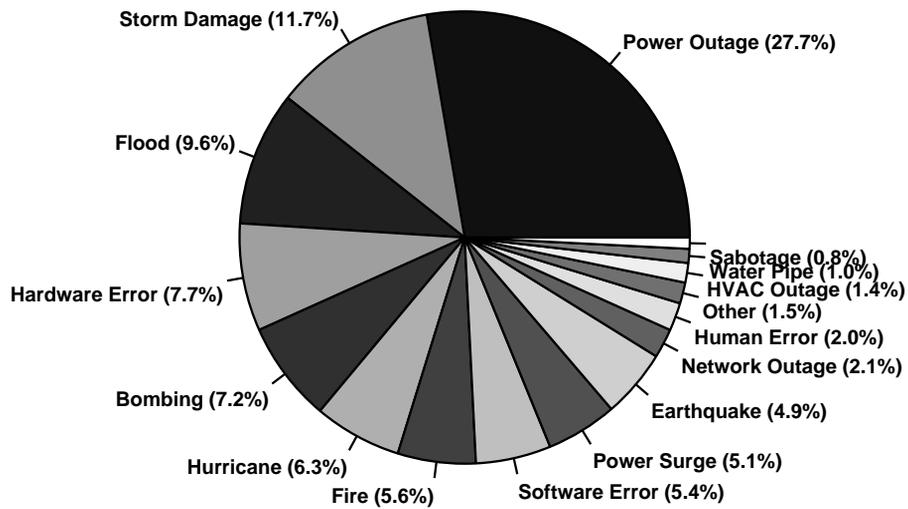
◇ ACL et partage réseau NFS

**A verifier...**

**§ 19.44 ACL sous FREEBSD**

**A completer...**

## Chapitre 20 : Mécanismes de sauvegarde



*There are two types of computer users in the world. . .  
those that have lost data, and those that are going to.*

Quand un fichier est détruit, il n'y a aucun moyen de le récupérer sous Unix.

⇒ Seule prévention : faire des sauvegardes

Les sauvegardes sont donc **indispensables**.

Unix offre une vue unifiée des fichiers ou des périphériques. On peut donc sauvegarder :

- sur n'importe quel périphérique en mode caractère
- dans un fichier
- sur un autre disque
- sur une autre machine

## § 20.1 Technologies de lecteur de bandes

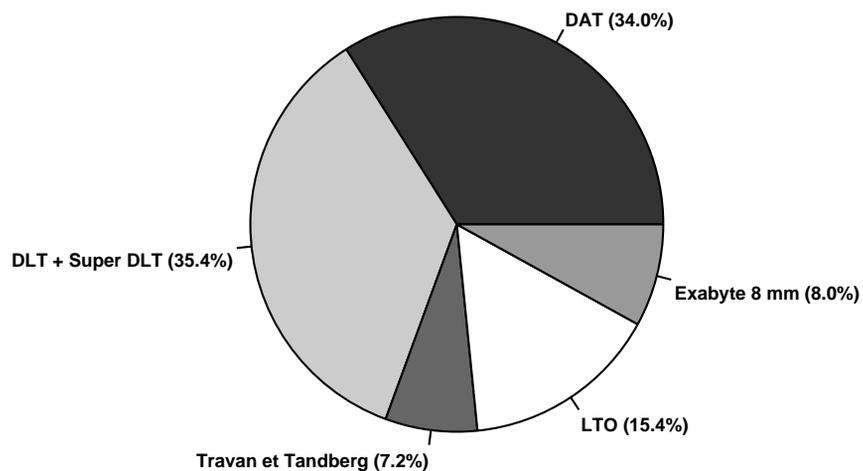
Il y a plusieurs technologies de lecteur de bandes magnétiques :

- technologie de la bande ronde : obsolète
- technologie QIC : QIC 60 (obsolète), QIC 150 (obsolète)
- technologie Exabyte (bande 8mm) : obsolète
- technologie DDS (bande 4mm) : DDS1 (obsolète), DDS2 (obsolète), DDS3, DDS4, DDS5  
apparemment ?
- technologie DLT : en cours d'obsolescence
- technologie Super DLT (alias SDLT) (Quantum, IBM, HP, Seagate) : en vogue
- technologie AIT (Sony) : essaye de percer
- technologie Super AIT (Sony en 2003) : 500 Go/bande, 30 Mo/s
- technologie Ultrium LTO : en vogue ; 100 Go/bande, 15 Mo/s

Type	Capacité (Go)	Vitesse (Mo/s)
AIT 1	25	4
AIT 2	50	6
DDS 2	4	0.4 / 0.7
DDS 3	12	1.2
DDS 4	20	2.75
DDS 5 ?	36	
DLT 4000	20	1.5
DLT 7000	35	5
DLT 8000	40	6
Ultrium LTO 215	100	7.5
Ultrium LTO 230	100	15
Ultrium LTO 460	200	30

Quelques prix :

Technologie	Type	Type	Capacité native	Prix
Ultrium LTO	Lecteur	SCSI externe	100 Go	1990 EUR (nov 2003)
Exabyte	Lecteur	SCSI 3	60 Go	1069 EUR (nov 2003)
Tandberg SLR 100	Lecteur	SCSI ?	50 Go	1668 EUR (nov 2003)
DLT 8000	Lecteur	SCSI interne	40 Go	1135 EUR (mai 2004)
DDS3	Lecteur	SCSI, interne	12 Go	569 EUR (mai 2004)
DDS4	Lecteur	SCSI, interne	20 Go	595 EUR (mai 2004)
DDS5	Lecteur	SCSI, interne	36 Go	679 EUR (mai 2004)



Expérience personnelle :

- un lecteur de bande DDS meurt en 3 ans (vérifié par 5 collègues)
- pas de soucis avec les lecteurs DLT
- les cartouches SDLT, Ultrium LTO sont chères

Le lecteur peut être :

- isolé (interne ou externe)
- monté dans une library ≡ ensemble lecteur + robot de manipulation de cartouches stockées dans un carroussel

**Figure manquante...**

## § 20.2 Périphériques UNIX de lecture de bande magnétique

Il y a plusieurs déclinaisons possibles :

- compression hardware activée ou pas
- rembobinage en fin d'écriture automatique ou non

Les périphériques par défaut :

- Sur LINUX : `/dev/st0`
- Sur SOLARIS : `/dev/rmt/0`
- Sur DIGITAL UNIX : `/dev/tape/0`
- Sur FreeBSD : `/dev/st0`

A noter que les lecteurs de bande magnétique sont en SCSI.

**§ 20.3 Utilitaires *dump*, *restore***

La particularité de **dump** est d'assurer la sauvegarde d'un disque en consultant la structure interne même du filesystem via le fichier spécial en mode caractère de la partition.

Syntaxe conseillée :

```
dump 0f archive partition
```

(sur Solaris, utiliser `/sbin/ufsdump`)

Points forts de *dump* :

- Possibilité de sauvegardes incrémentales. Un niveau est associé à la sauvegarde. Le niveau 0 correspond à une sauvegarde complète. Le niveau 1 correspond à la sauvegarde des seuls fichiers modifiés depuis le niveau 0. etc.
- Sauvegarde des fichiers spéciaux en tant que fichiers spéciaux.
- Sauvegarde des fichiers à trous à l'identique (sauvegarde des seuls blocs utilisés)

Un dump se compose de 4 étapes :

```
[root@localhost d]# dump 0f /tmp/archive /etc
DUMP: Date of this level 0 dump: Sat Jan 27 16:48:52 2001
DUMP: Date of last level 0 dump: the epoch
DUMP: Dumping /dev/hda2 (/ (dir etc)) to /tmp/archive
DUMP: Label: none
DUMP: mapping (Pass I) [regular files]
DUMP: mapping (Pass II) [directories]
DUMP: estimated 3514 tape blocks on 0.09 tape(s).
DUMP: Volume 1 started at: Sat Jan 27 16:48:52 2001
DUMP: dumping (Pass III) [directories]
DUMP: dumping (Pass IV) [regular files]
DUMP: Closing /tmp/archive
DUMP: Volume 1 completed at: Sat Jan 27 16:48:54 2001
DUMP: Volume 1 took 0:00:02
DUMP: Volume 1 transfer rate: 2535 KB/s
DUMP: DUMP: 5071 tape blocks on 1 volumes(s)
DUMP: finished in less than a second
DUMP: DUMP: Date of this level 0 dump: Sat Jan 27 16:48:52 2001
DUMP: DUMP: Date this dump completed: Sat Jan 27 16:48:54 2001
DUMP: DUMP: Average transfer rate: 2535 KB/s
DUMP: DUMP IS DONE
```

La commande pour relire une sauvegarde réalisée par dump est **restore** (sur Solaris, utiliser /sbin/ufsrestore).

Possibilité de restauration totale automatique :

```
cd partition-vidé
restore -r -f archive
```

On peut utiliser dump + restore pour dupliquer une arborescence. Par exemple :

```
dump 0f - /usr/local | ( cd /mnt ; restore rf - )
```

Possibilité de restauration en mode interactif avec sélection de ce que l'on restorer :

restore -f archive -i puis utiliser les commandes add fichier/dir, cd dir, delete fichier/dir, ls et la commande extract :

```
[root@localhost d]# restore -f /tmp/archive -i
...
restore > ls
.:
etc/

restore > cd etc
restore > ls
./etc:
...
im_palette-small.pal  passwd.OLD          yp.conf

restore > add yp.conf
restore > ls
./etc:
  im_palette-small.pal  passwd.OLD          *yp.conf
```

```
restore > extract
You have not read any tapes yet.
Unless you know which volume your file(s) are on you should start
with the last volume and work towards the first.
Specify next volume #: 1
set owner/mode for './?' [yn] y
restore > quit
[root@localhost d]# ls -Rl
.:
total 8
drwxr-xr-x  2 root  root          4096 Jan 27 14:47 etc

./etc:
total 4
-rw-r--r--  1 root  root           361 Jan 26 14:51 yp.conf
```

La commande pour connaître la table des matières d'une sauvegarde réalisée par *dump* est aussi **restore** (sur Solaris, utiliser `/sbin/ufsrestore`).

```
# ufsdump 0f /tmp/archive /opt3
DUMP: Date of this level 0 dump: Sat Jan 24 22:27:28 2004
DUMP: Date of last level 0 dump: the epoch
DUMP: Dumping /dev/rdsk/c0t0d0s6 (furax.unixiens.org:/opt3) to /tmp/archive.
DUMP: Mapping (Pass I) [regular files]
DUMP: Mapping (Pass II) [directories]
DUMP: Writing 32 Kilobyte records
DUMP: Estimated 350 blocks (175KB).
DUMP: Dumping (Pass III) [directories]
DUMP: Dumping (Pass IV) [regular files]
DUMP: 318 blocks (159KB) on 1 volume at 2650 KB/sec
DUMP: DUMP IS DONE

# ufsrestore tf /tmp/archive
      2      .
      3      ./lost+found
      4      ./foo
```

## § 20.4 Utilitaire *tar*

**tar** utilitaire assure une sauvegarde en accédant normalement au filesystem Unix.

Cette commande a un immense avantage : elle est disponible sur la totalité des plateformes Unix.

Contrairement à *dump*, elle permet la sauvegarde d'une partie seulement d'une partition. En fait, on peut sauvegarder des fichiers ou directories sans aucun rapport entre eux.

Syntaxe :

```
tar cvf archive fichiers
```

De préférence, utiliser le GNU tar :

```
ftp://ftp.lip6.fr/pub/gnu/tar/tar-1.14.tar.gz
```

associé au GNU zip :

```
ftp://ftp.lip6.fr/pub/gnu/gzip/gzip-1.2.4a.tar
```

Avantages :

- le GNU tar ne sauve pas les fichiers avec un '/' initial ;
- il offre la possibilité de se cantonner au sein d'un filesystem.

Problèmes classiques avec tar :

- Il ne sait pas sauver convenablement les major et minor numbers des fichiers de /dev. Pour sauver /dev, on préférera plutôt recréer les devices à partir de /dev/MAKEDEV.
- Par défaut, il ne sait pas sauver un fichier à trous (les blocs intermédiaires non alloués sont sauvés sous forme de caractère de code ASCII zéro).  
⇒ effacer par exemple les fichiers core avant de sauvegarder par cron, sinon remplissage inutile de la bande

Exemple de sauvegarde par tar – option "c" :

```
# tar cvf archive /etc
[root@localhost /tmp]# tar cvf archive /etc
tar: Removing leading '/' from member names
etc/
etc/csh.cshrc
etc/csh.login
etc/exports
etc/filesystems
etc/group
etc/host.conf
etc/hosts.allow
etc/hosts.deny
...
```

Exemple de consultation de la table de contenu de l'archive (option "t") :

```
# tar tvf archive
[root@localhost /tmp]# tar tvf archive
drwxr-xr-x root/root          0 2001-01-27 14:47:46 etc/
-rw-r--r-- root/root        220 2000-01-13 00:18:52 etc/csh.cshrc
-rw-r--r-- root/root        674 2000-01-13 17:24:18 etc/csh.login
-rw-r--r-- root/root          0 2000-01-13 00:18:52 etc/exports
-rw-r--r-- root/root         43 2000-02-17 22:42:07 etc/filesystems
-rw-r--r-- root/root        419 2001-01-26 14:51:13 etc/group
-rw-r--r-- root/root         26 2000-01-13 00:18:52 etc/host.conf
-rw-r--r-- root/root        161 2000-01-13 00:18:52 etc/hosts.allow
-rw-r--r-- root/root        347 2000-01-13 00:18:52 etc/hosts.deny
...
drwxr-xr-x root/root          0 2001-01-26 14:48:49 etc/profile.d/
...
```

Extraction d'un fichier – option "x" :

```
[root@localhost /tmp]# ls -l etc/passwd
ls: etc/passwd: No such file or directory
[root@localhost /tmp]# tar xvf archive etc/passwd
etc/passwd
[root@localhost /tmp]# ls -l etc/passwd
-rw-r--r--  1 root  root      597 Jan 27 14:47 etc/passwd
```

Sur Internet, il y a souvent des packages avec ".tar.gz" ou ".tgz" comme suffixes.

Comment en extraire le contenu ?

```
% gunzip foo.tar.gz
% tar xvf foo.tar
...
```

Cela nécessite le GNU zip.

Si l'on a le GNU tar, on peut combiner les 2 opérations en une seule :

```
% gtar xvzf foo.tar.gz
...
```

Exemple d'emploi de tar pour dupliquer une arborescence :

```
( cd /usr/local ; tar cf - . ) | ( cd /mnt ; tar xf - )
```

**§ 20.5 (Windows : : NtBackup)**

**Pour ARS 2005-2006...**

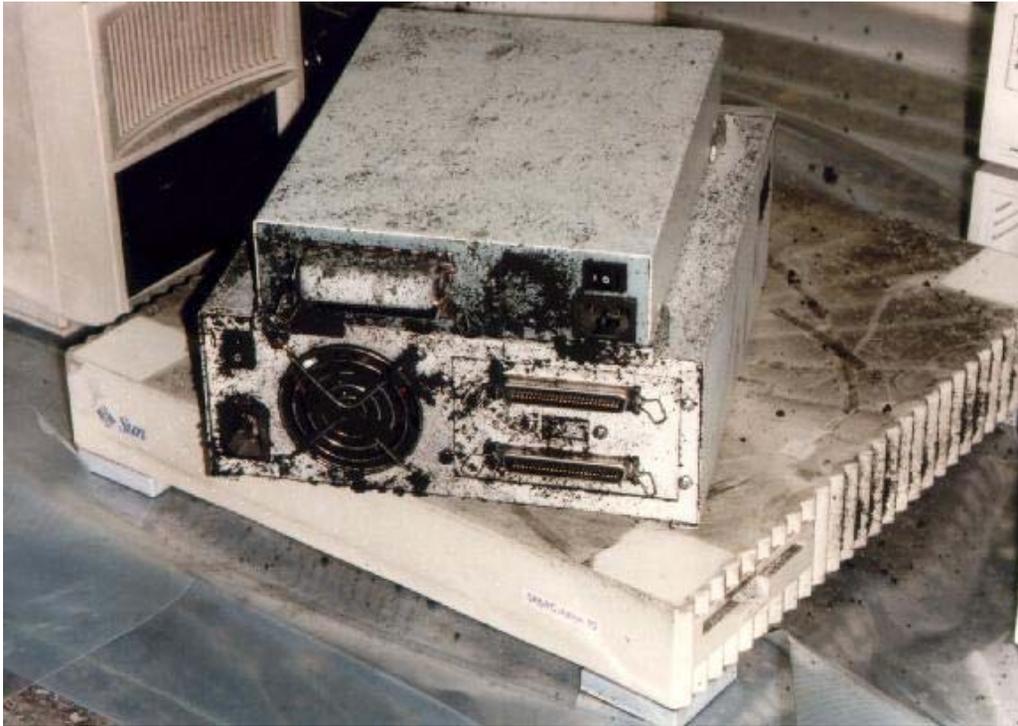
KB Microsoft

**§ 20.6 Quelques règles**

1. Sauvegarder
2. Sauvegarder lorsque le filesystem est le moins actif :
  - en mode mono-utilisateur
  - pendant les périodes creuses d'utilisation (la nuit par exemple)
3. Ne jamais restaurer de fichiers dans le homedir d'un utilisateur de peur d'y effacer des fichiers plus récents que ceux de la sauvegarde.
4. Vérifiez périodiquement les sauvegardes.
5. Conserver un jeu de sauvegardes ok à part des autres en cas de destruction des autres par un sinistre.



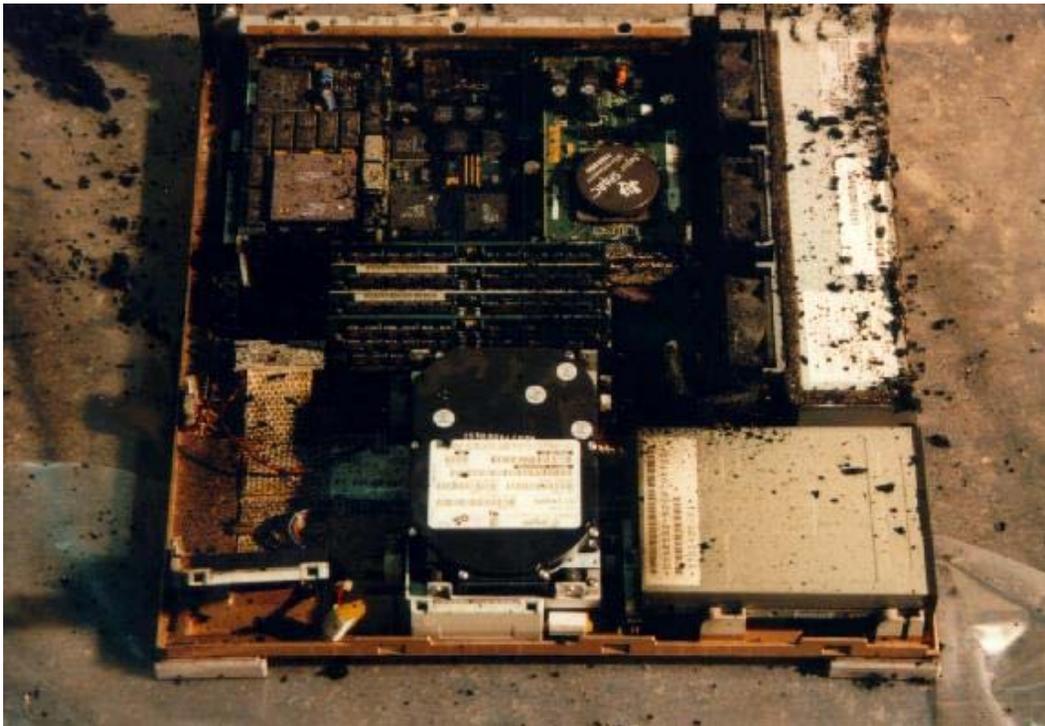
Matériels encrassés suite à un incendie (1/4).



Matériels encrassés suite à un incendie (2/4).



Matériels encrassés suite à un incendie (3/4).



Matériels encrassés suite à un incendie (4/4).



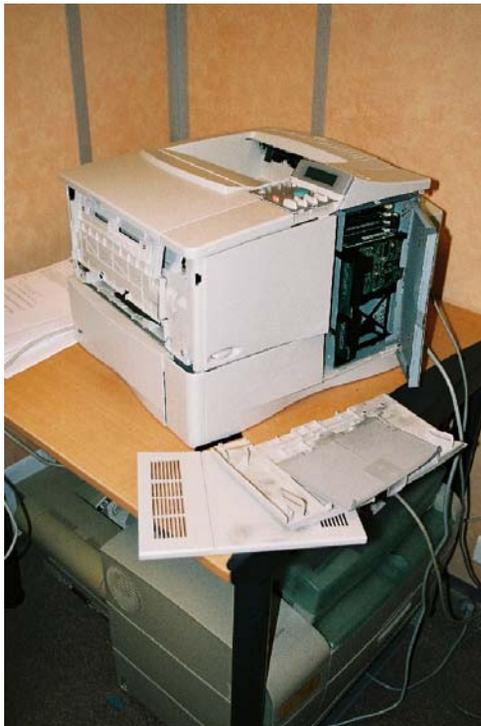
Faux plafond effondré suite à une inondation.



Locaux inondés suite à la tempête de décembre 1999.



Matériel cassé suite à une crise de folie d'un employé (1/2)



Matériel cassé suite à une crise de folie d'un employé (2/2)

## Chapitre 21 : Mémoire virtuelle

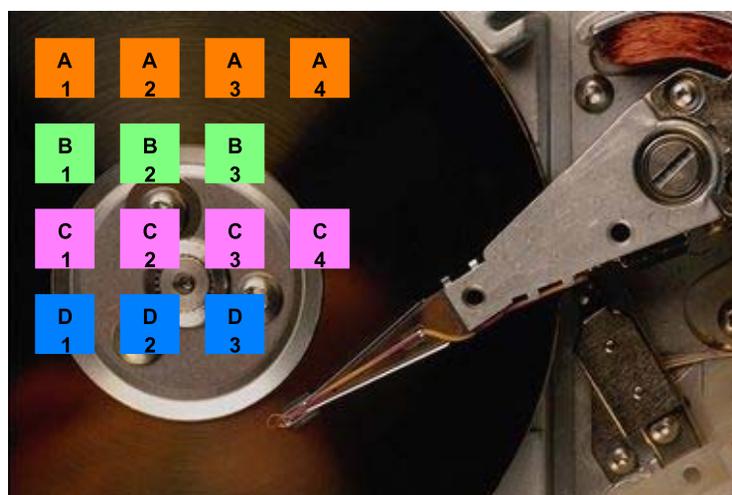
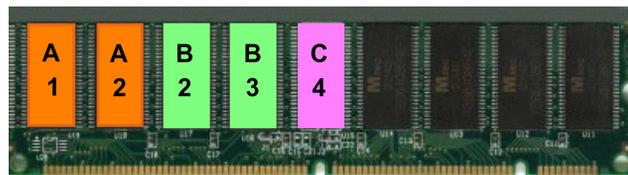
### § 21.1 Principe de la mémoire virtuelle

Unix  $\equiv$  système multi-utilisateurs + multi-tâche

$\Rightarrow$  gestion sophistiquée de la mémoire de tous les processus

Principe approximatif :

- Un processus a l'impression d'avoir un espace mémoire  $>$  RAM. C'est la **mémoire virtuelle**.
- L'espace mémoire est découpée en segments de taille donnée, les pages. On charge les pages d'un processus en RAM au fur et à mesure des besoins. C'est la **pagination**.
- Quand la RAM est "pleine", et que l'on doit charger des pages, on copie sur disque de vieilles pages et on les élimine de la RAM pour les remplacer par les nouvelles pages. C'est ce que l'on appelle **swapper**.



Le fonctionnement d'une machine Unix standard nécessitera donc une partition disque pour le swap.

La partition de swap est créée comme les autres partitions régulières.

Par contre, on n'a pas besoin de faire « `newfs` » sur la partition de swap.

## § 21.2 Affichage de la taille du swap LINUX : `free`

Comment connaître la taille du swap ?

Sur Linux : utiliser `free`

```
# free
      total        used         free       shared    buffers
Mem:   129488      121716         7772         4136     104824
-/+ buffers:           16892       112596
Swap:   259768           0       259768
```

### § 21.3 Affichage de la taille du swap SOLARIS : `swap`

Comment connaître la taille du swap ?

Sur SOLARIS, 2 possibilités

Utiliser `swap -l`

```
# /usr/sbin/swap -l
swapfile          dev  swaplo blocks   free
swapfs            -      0 139408 123280
/dev/dsk/c0t3d0s1 32,25 8 98488 98488
```

Utiliser `swap -s`

```
# /usr/sbin/swap -s
total: 8264k bytes allocated + 3520k reserved = 11784k used, \
52632k available
```

### § 21.4 Activation de partitions de swap LINUX : `swapon`

Suite de commandes à donner :

```
# free
              total          used          free   shared  buffers   cached
Mem:          512720       198032       314688        1168    13428    89168
-/+ buffers/cache:    95436       417284
Swap:         1052216           0       1052216
```

```
# mkswap -f /dev/hda5
```

```
# swapon /dev/hda5
```

```
# free
              total          used          free   shared  buffers   cached
Mem:          512720       198128       314592        1168    13428    89180
-/+ buffers/cache:    95520       417200
Swap:         1068208           0       1068208
```

**§ 21.5 Activation de partitions de swap SOLARIS : swap**

Syntaxe : `swap -a partition`

```
# swap -l
swapfile          dev  swaplo blocks  free
/dev/dsk/c0t0d0s1 136,1      16 4198304 3455392
# swap -s
total: 601128k bytes allocated + 230824k reserved = 831952k used, \
1686832k available
# swap -a /dev/dsk/c0t0d0s6
# swap -l
swapfile          dev  swaplo blocks  free
/dev/dsk/c0t0d0s1 136,1      16 4198304 3455392
/dev/dsk/c0t0d0s6 136,6      16 1440224 1440224}
# swap -s
total: 601128k bytes allocated + 230824k reserved = 831952k used, \
2406896k available
```

**§ 21.6 Activation de partitions de swap au boot LINUX : /etc/fstab**

Les scripts de démarrage activent automatiquement les partitions de swap au boot.

Les partitions sont indiquées dans le fichier `/etc/fstab` :

```
/dev/hda2          none  swap  sw          0 0
```

**§ 21.7 Activation de partitions de swap au boot SOLARIS : /etc/vfstab**

Les scripts de démarrage activent automatiquement les partitions de swap au boot.

Les partitions sont indiquées dans le fichier /etc/vfstab :

```
#device          device mount  FS      fsck    mount  mount
#to mount        to fsck point  type    pass    at boot options
#
/dev/dsk/c0t0d0s1 -      -      swap    -      no     -
```

**§ 21.8 Taille du swap**

Quelle taille donner à une partition de swap ?

Par tradition on met trois fois la RAM.

En cas de taille de swap insuffisante, aïe, problèmes :

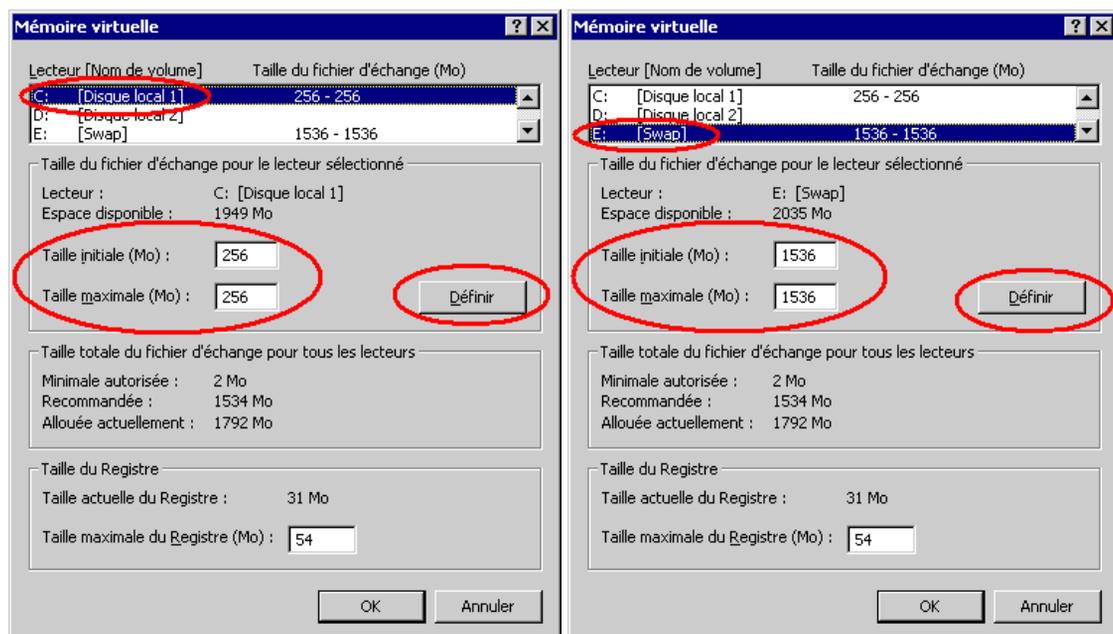
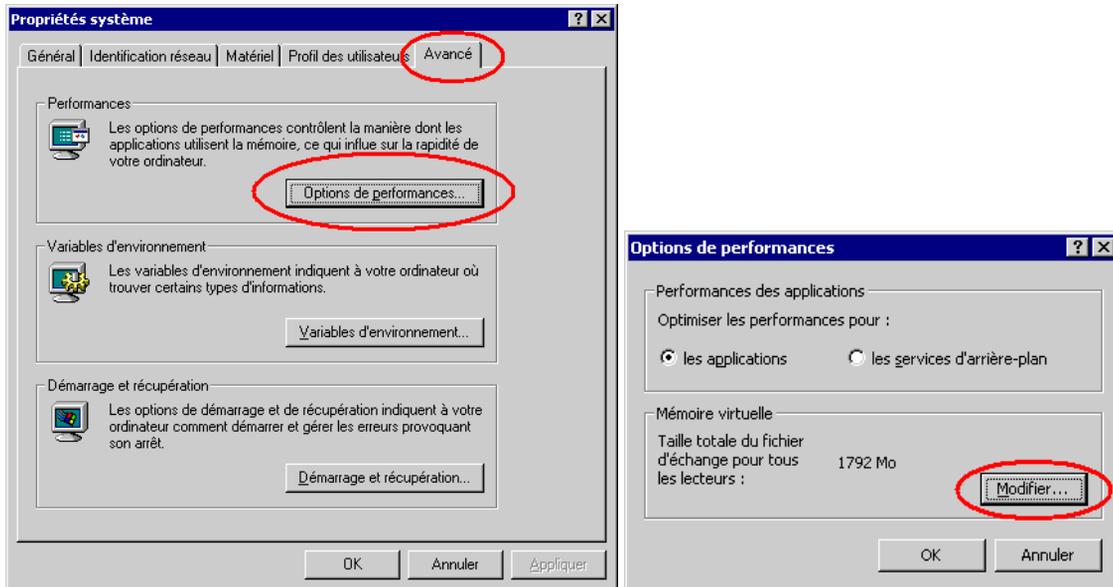
```
% ./programme-gourmand
No swap space available
```

```
Sep 15 17:07:43 solaris.example.com /kernel: pid 335 (z), uid 0, was killed:
out of swap space
```

## § 21.9 (Windows : : taille du swap)

Le swap sous Windows est stocké dans un fichier à la taille variable.

Il faut régler la taille de ce fichier pour qu'elle ne varie pas.



## § 21.10 Fichiers de swap

On peut utiliser des fichiers pour swapper au lieu de partitions.

Tous les systèmes Unix ne le permettent pas.

Avantages :

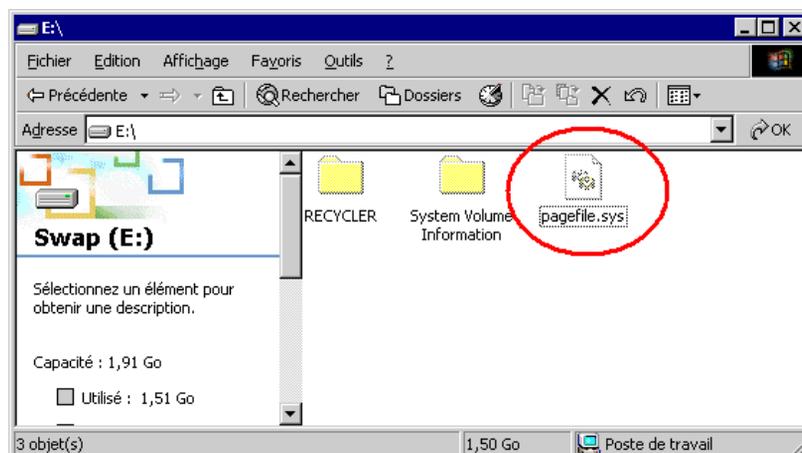
- en cas de swap insuffisant, il est simple de créer un fichier de swap.

Inconvénients :

- une fois ajouté au swap, il est souvent impossible de faire machine arrière ; pour supprimer le fichier de swap, il faut rebooter.
- swap via le filesystem et non via en dessous  $\Rightarrow$  solution a priori lente/peu performante

## § 21.11 (Windows : : Fichiers de swap)

Sous Windows 2000, il n'y a que des fichiers de swap :



**§ 21.12 Création de fichiers de swap : `mkfile`**

Comment générer des fichiers de swap ?

Via la commande `mkfile` si elle est disponible sur l'Unix :

```
# mkfile 50m grosfichier
# chmod 600 grosfichier
# ls -l grosfichier
-rw----- 1 besancon 52428800 Nov 16 01:27 grosfichier
```

Faire ensuite :

```
# swapon /chemin/vers/grosfichier
```

Les droits du fichier doivent être 600. Pourquoi ?

**§ 21.13 (Windows :: `creatfil`)**

Commande `creatfil` du Resource Kit Windows 2000.

Syntaxe : `creatfil fichier nombre-de-blocs`

(la taille d'un bloc est de 1024 octets)

```
c:\>creatfil %TEMP%\grosfichier.dat 1024
c:\>dir %TEMP%\grosfichier.dat
Volume in drive C is Windows XP
Volume serial Number is 0C5C-E708
```

Directory of C:\DOCUME~1\besancon\LOCALS~1\Temp

```
18/07/2004 12:35          1 048 576 grosfichier.dat
                1 File(s)          1 048 576 bytes
                0 Dir(s)          1 173 176 320 bytes free
```

**§ 21.14 Création de fichiers de swap (2) : dd**

Comment générer des fichiers de swap ?

Via la commande dd (en anglais *device to device*) :

```
# dd if=/dev/zero of=grosfichier bs=1M count=50
50+0 records in
50+0 records out
52428800 bytes transferred in 3.533640 secs (14837052 bytes/sec)
# chmod 600 grosfichier
# ls -l grosfichier
-rw----- 1 besancon 52428800 Nov 16 01:27 grosfichier
```

Faire ensuite :

```
# swapon /chemin/vers/grosfichier
```

Les droits du fichier doivent être 600. Pourquoi ?

## Chapitre 22 : Processus et exécutable

Unix est synonyme de multitâches. L'aspect multitâche se voit par la commande `ps`

```
% ps -aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.1  1120    52 ?        S    Oct23   0:06  init
root         2  0.0  0.0     0     0 ?        SW   Oct23   0:00  [kflushd]
root         3  0.0  0.0     0     0 ?        SW   Oct23   0:01  [kupdate]
...
nobody     476  0.0  0.1  1300    44 ?        S    Oct23   0:01  [identd]
daemon     490  0.0  0.0  1144     0 ?        SW   Oct23   0:00  [atd]
xfs        636  0.0  0.3  2820   120 ?        S    Oct23   0:18  xfs -droppriv -da
root     14703  0.0  0.0  2256     0 tty1     SW   Oct25   0:00  [login]
root     9813  0.0  0.0  6912     0 ?        SW   Oct31   0:09  [kdm]
idiri   20810  0.0  0.0  6552     0 ?        SW   15:13   0:01  [kwm]
idiri   20837  0.0  0.0  2080     0 ?        SW   15:13   0:00  [tcsh]
idiri   20863  0.0  0.0  1996     0 pts/0    SW   15:13   0:00  [tcsh]
besancon 21785  0.0  1.3  1732   416 pts/1    S    15:25   0:00  -bash
idiri   23600  0.0  0.0  1844     0 tty1     SW   16:26   0:00  [vi]
idiri   23660  0.2  1.5  1860   472 tty2     S    16:39   0:01  vi probleme6.c
...
```

Comment bien utiliser "ps" ?

Une bonne utilisation de "ps" repose sur la connaissance des différentes options employables.

La difficulté est que les options sont différentes selon que le système est de type BSD ou AT&T. On retiendra :

(BSD)    `ps -aux`    ≡    `ps -edf`    (AT&T)

## § 22.1 Liste des processus : ps version BSD

- **ps** : les processus associés à son terminal

```
% ps
  PID  TT  STAT      TIME COMMAND
24578  p7  Ss      0:00.33 -bash (bash-2.01)
24600  p7  S       0:00.52 xdvi.bin partie-03.dvi
24602  p7  I       0:00.99 gs -sDEVICE=x11 -dNOPAUSE -dSAFER -q - (gs-5.10)
24876  p7  R+      0:00.00 ps
```

- **ps -x** : tous ses processus

```
% ps -x
  PID  TT  STAT      TIME COMMAND
24578  p7  Ss      0:00.34 -bash (bash-2.01)
24600  p7  S       0:00.52 xdvi.bin partie-03.dvi
24602  p7  I       0:00.99 gs -sDEVICE=x11 -dNOPAUSE -dSAFER -q - (gs-5.10)
24899  p7  R+      0:00.00 ps -x
24881  p8  Ss+     0:00.27 -bash (bash-2.01)
```

- **ps -ax** : tous les processus de la machine

```
% ps -ax
  PID  TT  STAT      TIME COMMAND
    0  ??  DLs      0:00.59 (swapper)
    1  ??  Ss      0:01.18 /sbin/init --
    2  ??  DL      1:25.50 (pagedaemon)
    3  ??  DL      0:00.00 (vmdaemon)
    4  ??  DL      2:35.03 (update)
...
24496  p6  I+      0:00.01 rlogin alpha
24578  p7  Ss      0:00.35 -bash (bash-2.01)
24600  p7  S       0:00.55 xdvi.bin partie-03.dvi
24602  p7  I       0:00.99 gs -sDEVICE=x11 -dNOPAUSE -dSAFER -q - (gs-5.10)
24935  p7  R+      0:00.00 ps -ax
24881  p8  Is+     0:00.27 -bash (bash-2.01)
 3869  p9  Is      0:00.16 -tcsh (tcsh-6.07)
...

```

- **ps -aux** : tous les processus de la machine avec les noms de login associés

```
% ps -aux
USER      PID %CPU %MEM    VSZ   RSS  TT  STAT  STARTED      TIME COMMAND
satish    24499  3.7  5.8  9320  7504  p5  SN    4:31PM    0:39.34 /usr/local/net
root      1  0.0  0.1   484    76  ??  Is    Wed04PM    0:01.18 /sbin/init --
root      2  0.0  0.0     0    12  ??  DL    Wed04PM    1:25.50 (pagedaemon)
root      3  0.0  0.0     0    12  ??  DL    Wed04PM    0:00.00 (vmdaemon)
...
besancon 24578  0.0  0.7   784   944  p7  Ss    4:44PM    0:00.36 -bash (bash-2.
besancon 24600  0.0  1.9  1108  2456  p7  S    4:44PM    0:00.60 xdvi.bin parti
```

- **ps -w** : Parfois la commande est tronquée par "ps -ax" :

```
% ps -ax
  PID  TT  STAT      TIME COMMAND
}
  368  ??  Is      0:04.36 sendmail: accepting connections on port 25 (sendmail-
```

On utilise alors "ps -axw" :

```
% ps -axw
...
  368  ??  Is      0:04.36 sendmail: accepting connections on port 25 (sendmail-8.8.6)
```

**ps -ww** : Encore plus de caractères affichés par rapport à "ps -w".

## § 22.2 Liste des processus : ps version System-V

- **ps -e** : tous les processus de la machine

```
% ps -e
  PID  TTY      TIME  CMD
    0  ?        0:01  sched
    1  ?        0:02  init
    2  ?        0:00  pageout
    3  ?        0:52  fsflush
  181  ?        0:06  automoun
  306  ?        0:00  sac
   49  ?        0:00  devfseve
   51  ?        0:00  devfsadm
  123  ?        0:01  rpcbind
  137  ?        0:01  rpc.nisd
...

```

- **ps -l** : affichage sous forme longue

```
% ps -l
 F S   UID   PID  PPID  C  PRI  NI           ADDR           SZ        WCHAN  TTY          TIME CMD
 8 S   999 16463 16461  0  40  20           ?             172             ? pts/0    0:00 csh
```

- **ps -f** : full listing

```
% ps -edf
      UID   PID  PPID  C    STIME TTY          TIME CMD
      root     0     0  0 09:09:47 ?           0:01 sched
      root     1     0  0 09:09:47 ?           0:02 /etc/init -
      root     2     0  0 09:09:47 ?           0:00 pageout
      root     3     0  0 09:09:47 ?           0:52 fsflush
      root    181     1  0 09:12:07 ?           0:06 /usr/lib/autofs/automountd
      ...
      daemon  283     1  0 09:12:12 ?           0:11 /usr/sbin/lpd
      root    291     1  0 09:12:13 ?           0:00 /usr/local/apache/bin/httpd
      root    296     1  0 09:12:14 ?           0:00 /usr/local/admin/lib/idled
      nobody 15130   291  0 23:30:56 ?           0:00 /usr/local/apache/bin/httpd
      besancon 16463 16461  0 00:12:26 pts/0    0:00 -csh
      ...
```

## § 22.3 Liste des processus : *top*

Inconvénient de *ps* : c'est la liste des processus à un instant *t*.

On ne pourra jamais sous Unix avoir la liste des processus en cours : le temps de chercher les processus et de faire le rapport à l'écran, certains processus peuvent avoir disparu.

Amélioration de *ps* : la commande "top" qui n'est cependant pas standard sur tous les Unix.

Son intérêt : elle affiche une liste des processus toutes les *n* secondes

URL : <ftp://lsof.itap.purdue.edu/pub/tools/unix/lsof/>

```

XTerm vt100
last pid: 21509; load averages: 0.02, 0.03, 0.04          14:52:22
71 processes: 69 sleeping, 1 running, 1 on cpu
CPU states: 98,2% idle, 0,6% user, 0,6% kernel, 0,6% iowait, 0,0% swap
Memory: 128M real, 141M swap in use, 445M swap free

  PID USERNAME THR PRI NICE  SIZE  RES STATE  TIME  CPU COMMAND
21507 thb      1  58   0 2544K 1592K cpu    0:00  0,73% top
   326 thb      1  48   0   28M   17M run    77:44  0,52% Xsun
21486 thb      1  48   0 2576K 1904K sleep  0:00  0,24% bash
21485 thb      1  58   0 4120K 3128K sleep  0:00  0,14% xterm
21509 thb      1  58   0 2920K 1616K sleep  0:00  0,14% xwd
18503 thb      1  58   0 4128K 2760K sleep  0:06  0,11% xterm
   327 thb      1  58   0 2704K 1432K sleep  1:29  0,05% fvwm
18504 thb      1  48   0 2592K 1736K sleep  0:02  0,03% bash
   539 thb      1  59   0   48M   14M sleep  24:41  0,01% emacs-20.4
   362 thb      1  13  15 3360K 1560K sleep  2:04  0,00% xbuffy
20528 thb      1  59   0   47M   25M sleep  5:46  0,00% netscape
   215 root     10  51   0 3040K 1880K sleep  0:50  0,00% nscd
     1 root      1  58   0   776K  144K sleep  0:31  0,00% init
   275 root      1  58   0 1896K  552K sleep  0:22  0,00% ssfd1
   133 root      1  58   0 1856K  696K sleep  0:10  0,00% in.ndpd

```

## § 22.4 Contrôle des processus : *kill*

La commande **kill** sert à communiquer avec des processus :

- arrêt de processus
- demande au processus de se reconfigurer
- passage en mode verbeux du processus
- etc.

La commande **kill** existe sur tous les Unix et il n'y a pas de différence de fonctionnement selon les Unix.

2 syntaxes possibles :

- syntaxe numérique : `kill -9 2878`
- syntaxe symbolique : `kill -KILL 2878`

Nom en langage C	Nom pour la commande <i>kill</i>	Valeur	Comportement	Sens
SIGHUP	HUP	1	Exit	Hangup
SIGINT	INT	2	Exit	Interrupt
SIGQUIT	QUIT	3	Core	Quit
SIGILL	ILL	4	Core	Illegal Instruction
SIGTRAP	TRAP	5	Core	Trace or Breakpoint Trap
SIGABRT	ABRT	6	Core	Abort
SIGEMT	EMT	7	Core	Emulation Trap
SIGFPE	FPE	8	Core	Arithmetic Exception
SIGKILL	KILL	9	Exit	Killed
SIGBUS	BUS	10	Core	Bus Error
SIGSEGV	SEGV	11	Core	Segmentation Fault
SIGSYS	SYS	12	Core	Bad System Call
SIGPIPE	PIPE	13	Exit	Broken Pipe
SIGALRM	ALRM	14	Exit	Alarm Clock
SIGTERM	TERM	15	Exit	Terminated
SIGUSR1	USR1	16	Exit	User Signal 1

Nom en langage C	Nom pour la commande <i>kill</i>	Valeur	Comportement	Sens
SIGUSR2	USR2	17	Exit	User Signal 2
SIGCHLD	CHLD	18	Ignore	Child Status Changed
SIGPWR	PWR	19	Ignore	Power Fail or Restart
SIGWINCH	WINCH	20	Ignore	Window Size Change
SIGURG	URG	21	Ignore	Urgent Socket Condition
SIGPOLL	POLL	22	Exit	Pollable Event
SIGSTOP	STOP	23	Stop	Stopped (signal)
SIGTSTP	TSTP	24	Stop	Stopped (user)
SIGCONT	CONT	25	Ignore	Continued
SIGTTIN	TTIN	26	Stop	Stopped (tty input)
SIGTTOU	TTOU	27	Stop	Stopped (tty output)
SIGVTALRM	VTALRM	28	Exit	Virtual Timer Expired
SIGPROF	PROF	29	Exit	Profiling Timer Expired
SIGXCPU	XCPU	30	Core	CPU time limit exceeded
SIGXFSZ	XFSZ	31	Core	File size limit exceeded
SIGWAITING	WAITING	32	Ignore	Concurrency signal reserved

Les signaux les plus utiles sont :

– SIGHUP

Cela envoie l'équivalent du `Ctrl` `C` du clavier.

– SIGKILL

Cela envoie un signal que le processus est obligé de suivre et qui se traduira inélectablement par la mort du processus.

## § 22.5 Priorité des processus : *nice* / *renice*

Un processus Unix possède une priorité d'exécution : son «nice number».

On a :  $-20 \leq \text{nice number} \leq 20$ .

-20 correspond à la priorité maximale, 20 correspond à la priorité minimale.

Un utilisateur peut descendre la priorité de ses processus.

Seul l'administrateur peut augmenter la priorité de n'importe quel processus.

Pour lancer un processus à la priorité N (N pouvant être négatif) : `nice [-N]` commande

Une fois un processus démarré, on peut changer sa priorité par la commande :

`renice` priorité PID

Exemple de la commande `nice` lancée par un utilisateur normal :

```
% nice -10 ./z
% ps -aux
USER          PID %CPU %MEM    SZ   RSS TT   STAT CPU START   TIME COMMAND
besancon 17641 94.6  0.1   16   144 p1    R N  0 23:42   0:07 ./z
...
% ps -xa
  PID TT   STAT CPU TIME COMMAND
...
17641 p1    R N  2  0:47 ./z

% nice --10 ./z
nice: cannot set priority: Permission denied
```

Exemple de la commande `nice` lancée par l'administrateur :

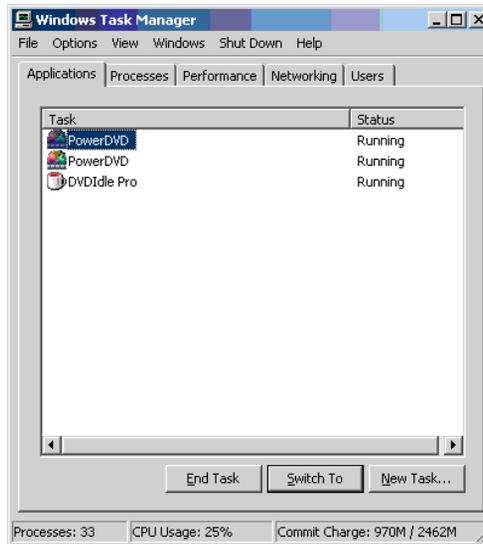
```
# nice --10 ./z
# ps -xa
  PID TT   STAT CPU TIME COMMAND
...
17663 p1    R <  0  0:04 ./z
```

Exemple de la commande `renice` :

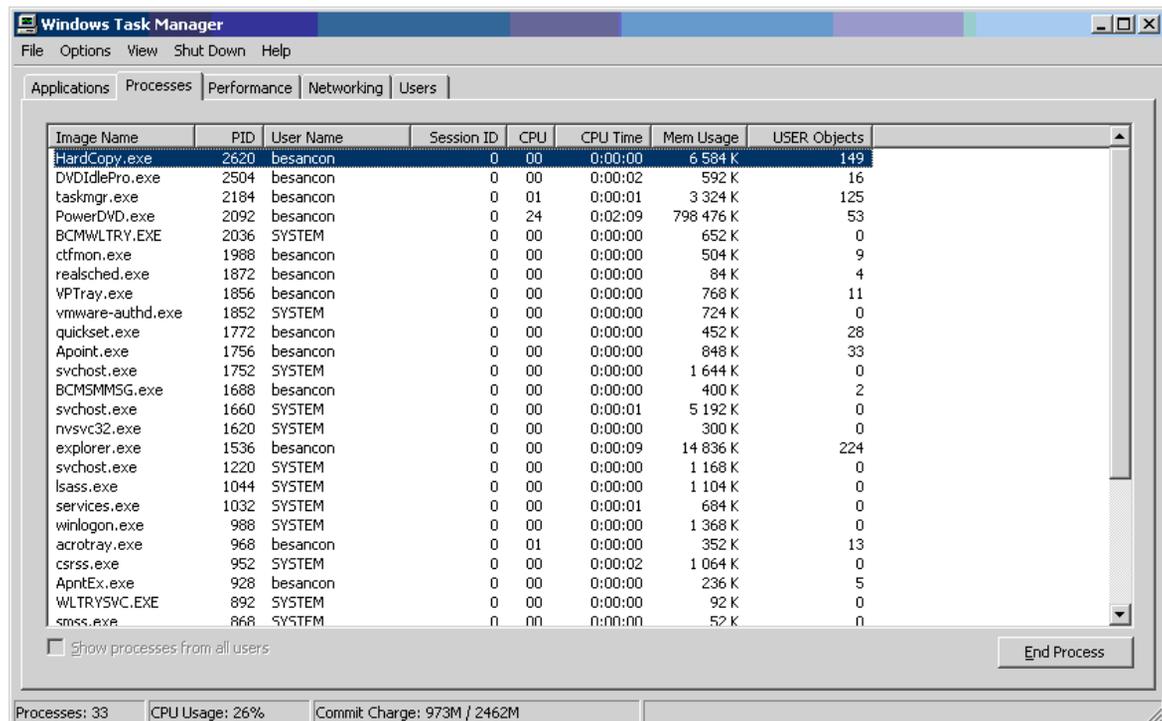
```
% nice -10 ./z
% ps -al
  F  UID    PID  PPID CP  PRI  NI   SZ   RSS WCHAN   STAT CPU TT   TIME COMMAND
...
20008001 4332 17665   224211  97  10   16   144          R N  2 p1  0:29 ./z
...
% renice 15 17665
17665: old priority 10, new priority 15
% ps -al
  F  UID    PID  PPID CP  PRI  NI   SZ   RSS WCHAN   STAT CPU TT   TIME COMMAND
...
20008001 4332 17665   224255 102  15   16   144          R N  0 p1  1:44 ./z
...
```

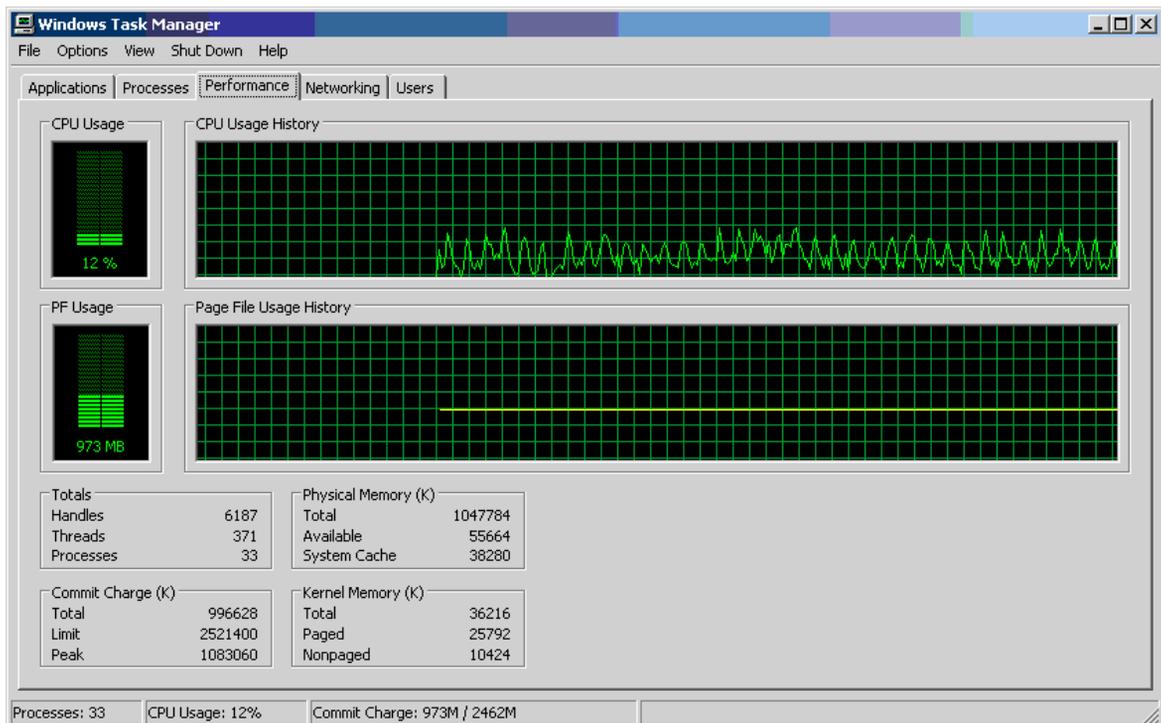
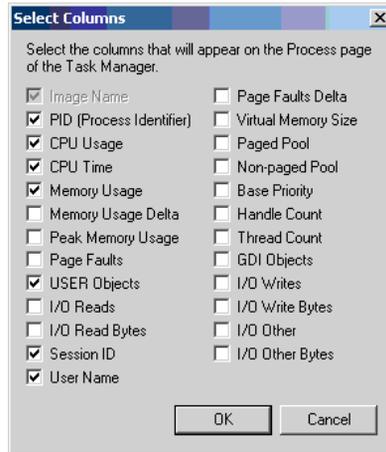
## § 22.6 (Windows : : processus)

Utilitaire taskmgr.exe de chez Microsoft :

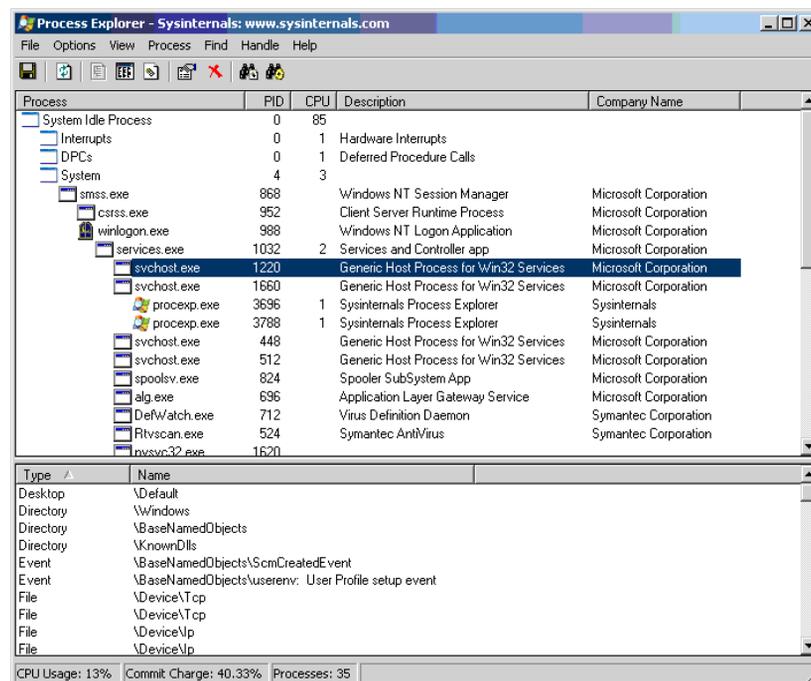


C'est le programme appelé via Ctrl - Alt - Del





Utilitaire procexp de chez <http://www.sysinternals.com>:



Utilitaires pstools de chez <http://www.sysinternals.com>:

- PsExec - execute processes remotely
- PsFile - shows files opened remotely
- PsGetSid - display the SID of a computer or a user
- PsKill - kill processes by name or process ID
- PsInfo - list information about a system
- PsList - list detailed information about processes
- PsLoggedOn - see who's logged on locally and via resource sharing
- PsLogList - dump event log records
- PsService - view and control services
- PsShutdown - shuts down and optionally reboots a computer
- PsSuspend - suspends processes
- PsUptime - shows you how long a system has been running since its last reboot

## § 22.7 Temps d'exécution d'un processus : `time`

A un programme sont associés :

- le temps d'exécution de code de bas niveau du système (par exemple écrire sur disque)
- le temps d'exécution de code de haut niveau de l'utilisateur (par exemple calculer une fonction mathématique)

La commande « `time` » fournit ces temps.

Syntaxe : `time commande parametres`

Par exemple :

```
% time gzip -v archive.tar
archive.tar:                26.8% -- replaced with archive.tar.gz

real    0m0.485s
user    0m0.390s
sys     0m0.070s
```

Attention, subtilité :

- il existe un builtin du shell appelé « `time` »
- il existe une commande « `time` » : « `/bin/time` »

```
% echo $SHELL
/bin/bash
% type time
time is a shell keyword
% time gzip -v archive.tar
archive.tar:                26.8% -- replaced with w.tar.gz

real    0m0.485s
user    0m0.390s
sys     0m0.070s

% /bin/time gzip -v archive.tar
archive.tar:                26.8% -- replaced with w.tar.gz

real    0.4
user    0.3
sys     0.0
```

## § 22.8 Charge du système : *uptime*

Pour obtenir la charge du système, utiliser la commande *uptime* :

```
% uptime
11:15pm up 4:44, 4 users, load average: 0.12, 0.02, 0.00
```

Quels sont les renseignements renvoyés :

- **le temps de fonctionnement du système**
- **le nombre d'utilisateurs connectés**
- **la charge du système** : 3 nombres représentant :
  - la moyenne de processus actifs durant la minute précédente
  - la moyenne de processus actifs durant les 5 dernières minutes
  - la moyenne de processus actifs durant les 15 dernières minutes

Difficulté de définir une valeur «normale» :

- de nombreux processus en attente d'une entrée sortie donnent un fort indice de charge alors que le système est en parfaite mesure de répondre efficacement
- un seul processus sollicitant fortement la mémoire virtuelle de la machine en swappant sans cesse peut à lui seul empêcher la machine de répondre alors que l'indice de charge est bas
- on traite indifféremment les processus renicés et ceux qui ne le sont pas

On prendra comme valeur «normale» une valeur de 3.

On préférera les renseignements renvoyés par *ps* pour se faire une idée de la charge réelle du système.

## § 22.9 Processus non tuables, processus zombies

Il arrive occasionnellement qu'un processus soit non tuable par la commande `kill` même avec le signal `SIGKILL` (-9).

Ces processus tombent dans 3 catégories :

- Processus en attente d'écriture à travers NFS
- Processus en attente d'opération physique sur un périphérique
- Processus zombie, marqués par un **Z** ou **<defunct>** lorsque l'on fait `ps`, généré lorsqu'un processus se termine et ne reçoit pas à ce moment là d'acknowledgement de la part de son parent (par exemple parce que le processus père est déjà mort).

## § 22.10 Processus zombies (2)

Les processus zombies ne sont pas gênant : ils ne consomment plus de ressources. Ils disparaîtront au prochain reboot de la machine.

La seule gêne est la consommation d'une entrée processus dans le tableau interne du noyau de gestion des processus.

Programme C de création de processus zombie (auteur : Pierre.David@crc.u-strasbg.fr)

```
main (int argc, char *argv [])
{
    switch (fork ())
    {
        case -1 :      /* erreur */
            perror ("fork") ;
            exit (1) ;
        case 0 :      /* fils */
            exit (0) ; /* => zombification */
        default :     /* père */
            pause () ; /* => père indigne : il se fiche de la mort du fils */
    }
}
```

L'exécution du programme donne :

```
% ./zombie &
[1] 16274
% ps -edf | grep 16274
besancon 16274 18533 0 17:31:44 pts/3    0:00 ./zombie
besancon 16275 16274 0                0:00 <defunct>
```

Le processus 16275 est un processus zombie.

Commande SOLARIS pour tuer les processus zombies :

```
% ps -edf
      UID  PID  PPID  C   STIME TTY      TIME CMD
...
besancon 19763 13971 0 08:32:31 pts/7    0:00 ./zombie
besancon 19764 19763 0                0:00 <defunct>
...

% ls -l /usr/bin/preap
-r-xr-xr-x  42 root      bin           5416 Apr  7  2002 /usr/bin/preap

% preap 19764
19764: exited with status 0

% ps -edf | grep 1976
      UID  PID  PPID  C   STIME TTY      TIME CMD
...
besancon 19763 13971 0 08:32:31 pts/7    0:00 ./zombie
```

## § 22.11 Attribut de fichier exécutable : bit setuid

Il existe un attribut spécial de fichier réservé à la gestion du système : le bit **setuid** (4000 en octal).

Avec ce bit positionné, le programme est exécuté avec les droits de l'utilisateur propriétaire.

```
% ls -lgF a b
-rwxr-xr-x  1 besancon  software      249 Sep 20 22:43 a
-rwxr-xr-x  1 besancon  software      249 Sep 20 22:43 b
% chmod u+s a
% chmod 4711 b
% ls -lgF a b
-rwsr-xr-x  1 besancon  software      249 Sep 20 22:43 a
-rws--x--x  1 besancon  software      249 Sep 20 22:43 b
```

### ◇ Exemple : commande `passwd`

Les fichiers contenant les informations sur les comptes utilisateurs sont :

```
-rw-r--r--  1 root      sys           946 Sep 21 22:36 /etc/passwd
-rw-----  1 root      sys           488 Jun 21 22:43 /etc/shadow
```

Notez leurs droits !

Pourtant un simple utilisateur change son mot de passe et modifie donc ces fichiers !

A PRIORI, IMPOSSIBLE !!! et pourtant si...

Explication : bit setuid de la commande `passwd` :

```
-r-sr-sr-x  1 root      sys          21964 Apr  7 2002 /bin/passwd
```

Attention à l'affichage du bit setuid !

Classiquement :

```
% gcc exemple.c -o exemple.exe
% ls -l exemple.exe
-rwxr-xr-x  1 besancon adm          6204 Jan 24 20:22 exemple.exe
% chmod u+s exemple.exe
% ls -l exemple.exe
-rwsr-xr-x  1 besancon adm          6204 Jan 24 20:22 exemple.exe
```

Moins classiquement :

```
% touch exemple.txt
% ls -l exemple.txt
-rw-r--r--  1 besancon adm          0 Jan 24 20:21 exemple.txt
% chmod u+s exemple.txt
chmod: WARNING: exemple.txt: Execute permission required for set-ID on execution
% ls -l exemple.txt
-rw-r--r--  1 besancon adm          0 Jan 24 20:21 exemple.txt
% chmod 4644 exemple.txt
% ls -l exemple.txt
-rwsr--r--  1 besancon adm          0 Jan 24 20:21 exemple.txt
```

Bref :

- affichage « S » ≡ « bit 04000 seul »
- affichage « s » ≡ « bit x + bit S »

## § 22.12 Attribut de fichier exécutable : bit setgid

Il existe un attribut spécial de fichier réservé à la gestion du système : le bit **setgid** (2000 en octal).

Avec ce bit positionné, le programme est exécuté avec les droits du groupe propriétaire

```
% ls -lgF a b
-rwxr-xr-x  1 besancon  software      249 Sep 20 22:43 a
-rwxr-xr-x  1 besancon  software      249 Sep 20 22:43 b
% chmod g+s a
% chmod 2711 b
% ls -lgF a b
-rwxr-sr-x  1 besancon  software      249 Sep 20 22:43 a
-rwx--s--x  1 besancon  software      249 Sep 20 22:43 b
```

### ◇ Exemple : commande `write`

La console d'une machine Unix appartient à l'utilisateur qui se connecte dessus :

```
crw--w----  1 besancon tty          0,  0 Aug 28 13:06 /dev/console
```

Notez leurs droits !

Pourtant un simple utilisateur peut écrire sur la console via la commande `write` (si l'on est en `mesg y`)!

A PRIORI, IMPOSSIBLE!!! et pourtant si...

Explication : bit setgid de la commande `write` :

```
-r-xr-sr-x  1 root      tty          11484 Apr  7 2002 /usr/bin/write
```

Attention à l'affichage du bit setgid !

Classiquement :

```
% gcc exemple.c -o exemple.exe
% ls -l exemple.exe
-rwxr-xr-x  1 besancon adm          6204 Jan 24 20:22 exemple.exe
% chmod g+s exemple.exe
% ls -l exemple.exe
-rwxr-sr-x  1 besancon adm          6204 Jan 24 20:22 exemple.exe
```

Moins classiquement :

```
% touch exemple.txt
% ls -l exemple.txt
-rw-r--r--  1 besancon adm          0 Jan 24 20:21 exemple.txt
% chmod g+s exemple.txt
chmod: WARNING: exemple.txt: Execute permission required for set-ID on execution
% ls -l exemple.txt
-rw-r--r--  1 besancon adm          0 Jan 24 20:21 exemple.txt
% chmod 2644 exemple.txt
% ls -l exemple.txt
-rw-r-lr--  1 besancon adm          0 Jan 24 20:21 exemple.txt
```

Bref :

- affichage « l » ≡ « bit 02000 seul »
- affichage « s » ≡ « bit x + bit l »

## § 22.13 Directory /proc

/proc  $\equiv$  **system process**

Il contient une représentation sous forme de fichiers virtuels et directories virtuels d'informations de la machine Unix.

Utilité ?

$\Rightarrow$  pour le programmeur système qui utilisera les appels standard (et non pas des appels spécifiques comme la `libkvm` ou des `ioctl()` spécifiques)

Implémentation de /proc propre à chaque Unix.

Représentation virtuelle de :

- informations sur les processus
- informations sur le hardware
- informations sur le noyau

### ◇ Exemple : informations sur les processus

```
% echo $$
23247
% cd /proc/23247
% ls -l
total 0
-r--r--r--  1 besancon sysadmin    0 Sep 22 14:30 cmdline
lrwx-----  1 besancon sysadmin   64 Sep 22 15:45 cwd -> [0001]:1523515394
-r-----  1 besancon sysadmin    0 Sep 22 15:45 environ
lrwx-----  1 besancon sysadmin   64 Sep 22 15:45 exe -> [0801]:10251
dr-x-----  1 besancon sysadmin    0 Sep 22 15:45 fd
pr--r--r--  1 besancon sysadmin    0 Sep 22 15:45 maps
-rw-----  1 besancon sysadmin    0 Sep 22 15:45 mem
lrwx-----  1 besancon sysadmin   64 Sep 22 15:45 root -> [0801]:2
-r--r--r--  1 besancon sysadmin    0 Sep 22 14:29 stat
-r--r--r--  1 besancon sysadmin    0 Sep 22 15:45 statm
-r--r--r--  1 besancon sysadmin    0 Sep 22 15:45 status
```

◇ Exemple : informations sur le hardware

```
% ls -l /proc/pci
-r--r--r--    1 root      root           0 Nov  2 18:52 /proc/pci

% cat pci
PCI devices found:
  Bus 0, device 0, function 0:
    Host bridge: VIA Technologies, Inc. VT82C693A/694x [Apollo PRO133x] (rev 196).
    Prefetchable 32 bit memory at 0xd0000000 [0xd3ffffff].
  Bus 0, device 1, function 0:
    PCI bridge: VIA Technologies, Inc. VT82C598/694x [Apollo MVP3/Pro133x AGP] (rev 0).
    Master Capable.  No bursts.  Min Gnt=12.
...

```

```
% cat /proc/cpuinfo
processor       : 0
vendor_id     : GenuineIntel
cpu family    : 6
model        : 8
model name    : Pentium III (Coppermine)
stepping     : 10
cpu MHz      : 1002.214
cache size   : 256 KB
Physical processor ID : 0
Number of siblings : 1
fdiv_bug     : no
hlt_bug      : no
f00f_bug     : no
coma_bug     : no
fpu          : yes
fpu_exception : yes
cpuid level  : 2
wp           : yes
flags        : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 mmx fxsr sse
bogomips    : 1989.22
...

```

◇ Exemple : informations sur le noyau et sa version

```
% cat /proc/version
Linux version 2.4.18-27.7.xsmp (bhcompile@stripples.devel.redhat.com)
(gcc version 2.96 20000731 (Red Hat Linux 7.3 2.96-112)) #1 SMP Fri
Mar 14 05:52:30 EST 2003
```

◇ Exemple : informations sur le uptime du noyau

```
% cat /proc/uptime
810311.03 800120.37
```

◇ Exemple : informations sur des statistiques du noyau sur l'emploi de la machine

```
% cat /proc/stat
cpu 1276780 2094 1821586 158965456
cpu0 681135 1171 903015 79447636
cpu1 595645 923 918571 79517818
page 19636553 18178406
swap 3965 4959
intr 460335474 414888745 32 0 4 4 3635073 6 0 1 0 35855460 1382047 897324 0 11 3676767 0 0 0 0 0 0 0 0
disk_io: (3,0):(3,3,24,0,0) (22,2):(695266,373120,8363128,322146,6352160) (22,3):(2514420,999712,3090945)
ctxt 233821067
btime 1066985101
processes 201147
```

◇ Exemple : informations sur la table ARP du noyau

```
% cat /proc/net/arp
IP address      HW type        Flags           HW address     Mask           Device
134.157.46.254  0x1            0x2            00:02:7E:21:F7:9C  *             eth0
192.168.4.18    0x1            0x2            00:48:54:6B:E5:B0  *             eth3
```

◇ Exemple : informations sur les sockets unix du noyau

```
% cat /proc/net/unix
Num      RefCount Protocol Flags      Type St Inode Path
db7f5580: 00000002 00000000 00010000 0001 01  9782 /tmp/.font-unix/fs7100
e9fcc580: 00000002 00000000 00010000 0001 01  9940 /tmp/.X11-unix/X0
d3e23a80: 00000002 00000000 00010000 0001 01  8308 /var/lib/mysql/mysql.sock
ce1dd080: 00000002 00000000 00010000 0001 01  9600 /dev/gpmctl
d75f5080: 00000014 00000000 00000000 0002 01  5050 /dev/log
cb91e080: 00000002 00000000 00000000 0002 01 1088373
d4f46080: 00000002 00000000 00000000 0002 01 1087696
f679a580: 00000002 00000000 00000000 0002 01 1085719
f1204080: 00000002 00000000 00000000 0002 01 1057186
```

## ◇ Exemple : informations sur les utilisations de la mémoire par le noyau

```
% cat /proc/meminfo
      total:      used:      free:  shared: buffers:  cached:
Mem:  1055293440 1040719872 14573568          0 107249664 692142080
Swap: 4294901760 37588992 4257312768
MemTotal:      1030560 kB
MemFree:       14232 kB
MemShared:     0 kB
Buffers:       104736 kB
Cached:        654592 kB
SwapCached:    21328 kB
Active:        547860 kB
Inact_dirty:   197696 kB
Inact_clean:   119996 kB
Inact_target: 173108 kB
HighTotal:     131008 kB
HighFree:      1024 kB
LowTotal:      899552 kB
LowFree:       13208 kB
SwapTotal:     4194240 kB
SwapFree:      4157532 kB
Committed_AS: 281672 kB
```

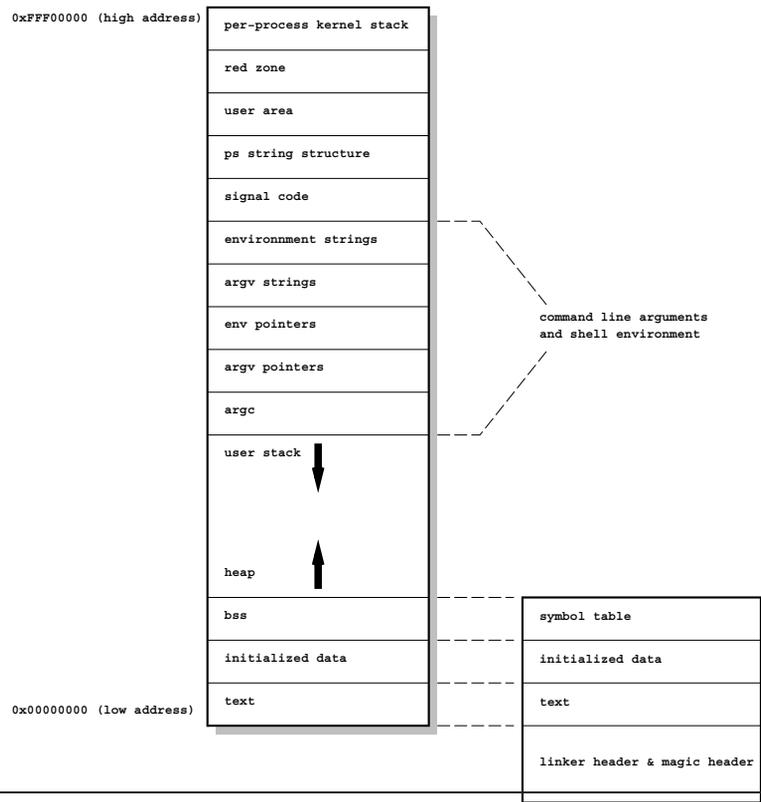
**§ 22.14 Principes des processus Unix**

Un processus Unix est une tâche définie par :

- le code machine à exécuter (*text*)
- une pile d'exécution (*user stack*)
- différentes zones mémoire stockant les structures manipulées par le code machine du programme (*initialized data, bss, heap*)
- différentes zones mémoire stockant les informations système sur le processus (*argc, argv, user area, red zone, kernel stack*)

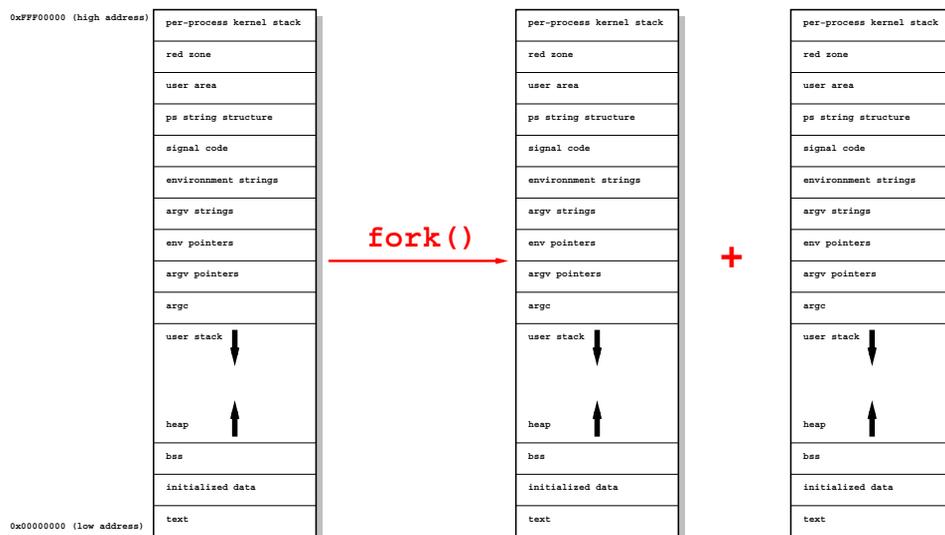
(cf `<sys/proc.h>`, `<sys/user.h>`)

ce qui donne le schéma de la page suivante :



Créer de nouveaux processus passe par la fonction C `fork()`

Créer un nouveau processus, c'est dupliquer toutes ces structures mémoire.



Un `fork()` est un mécanisme coûteux.

Au niveau de l'exécution du code :

avant le fork()

```
...
...
if ( fork() )
{
  ... /* parent */
  ... /* code */
}
else
{
  ... /* child */
  ... /* code */
}
...
...
```

après le fork()

```
...
...
if ( fork() )
{
  ... /* parent */ ← le père reprend ici
  ... /* code */
}
else
{
  ... /* child */
  ... /* code */
}
...
...
```

le père  
reprend  
ici

```
...
...
if ( fork() )
{
  ... /* parent */
  ... /* code */
}
else
{
  ... /* child */ ← le fils reprend ici
  ... /* code */
}
...
...
```

le fils  
reprend  
ici

Principale utilisation : fork () + exec () :

avant le fork()

```
...
...
if ( fork() )
{
  ... /* parent */
  ... /* code */
}
else
{
  exec(...) /* child */
  ... /* code */
}
...
...
```

après le fork()

```
...
...
if ( fork() )
{
  ... /* parent */ ← le père reprend ici
  ... /* code */
}
else
{
  exec(...) /* child */
  ... /* code */
}
...
...
```

le père  
reprend  
ici

```
...
...
if ( fork() )
{
  ... /* parent */
  ... /* code */
}
else
{
  exec(...) /* child */ ← le fils reprend ici
  ... /* code */
}
...
...
```

exec ()

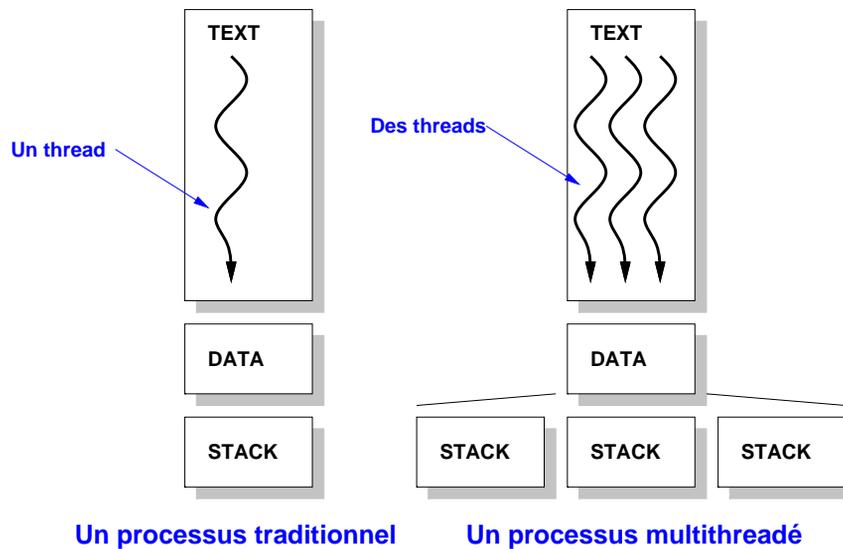
le fils  
reprend  
ici

après le exec ()

```
...
... /* autre code */
...
...
```

## § 22.15 Principes des threads Unix

Un thread  $\equiv$  *Lightweight process*  $\equiv$  un programm counter, une stack, un jeu de registres, **tout le reste est partagé avec le processus**



Développement commencé sous Unix dans les années 1980.

Les threads sont un héritage du système **MACH** qui y empruntent la terminologie : **task**, **thread**

Un processus Unix  $\equiv$  une task + un seul thread

Standards Posix : P1003.1b final draft 14 (realtime) et P1003.1c final draft 10 (multithreading)

Avantages des threads :

- création *light* des threads (60 % plus rapide de créer un thread qu'un processus)
- possibilité de créer des applications concurrentes sur des machines mono ou multi processeurs en étant complètement transparentes pour l'utilisateur
- possibilité de faire des applications non bloquantes (le plus souvent en termes d'I/O)
  - ⇒ amélioration du temps de réponse
 exemple : serveur web Apache 2.x pour traiter plus de connexions en parallèle  
 cf `http://httpd.apache.org`

Inconvénients des threads : **penser en profondeur le code à programmer** :

- attention aux *race conditions* (les threads sont ordonnancés et exécutés au hasard par l'OS)
- attention à la non réentrance du code (variables statiques ou globales a priori bannies)
  - ⇒ le code doit être *thread safe*
- utilisation de `mutex` ou de *POSIX condition variable* pour synchroniser les threads entre eux, avec risque de *deadlocks* malgré tout

## § 22.16 Visualisation des threads Unix

Sur Solaris, utiliser l'option `-L` de `ps` ou les options d'affichage `lwp` et `nlwp` :

```
% ps -e -o pid,user,lwp,nlwp,comm
  PID      USER      LWP  NLWP  COMMAND
...
  361      root        1     8  /usr/sbin/nscd
  361      root        2     8  /usr/sbin/nscd
  361      root        3     8  /usr/sbin/nscd
  361      root        4     8  /usr/sbin/nscd
  361      root        5     8  /usr/sbin/nscd
  361      root        6     8  /usr/sbin/nscd
  361      root        7     8  /usr/sbin/nscd
  361      root        8     8  /usr/sbin/nscd
...
```

Ici on voit que le processus Unix 361 comporte en fait 8 threads.

Sur LINUX, utiliser l'option -m :

```
% ./mkthread2 &
[1] 17049
Thread number 16386 / pid 17051
Thread number 32771 / pid 17052
Thread number 49156 / pid 17053
Thread number 65541 / pid 17054

% ps ax | grep mkt
17049 pts/32  S      0:00 ./mkthread2
17062 pts/32  S      0:00 grep mkthread2

% ps axm | grep mkt
17049 pts/32  S      0:00 ./mkthread2
17050 pts/32  S      0:00 ./mkthread2
17051 pts/32  S      0:00 ./mkthread2
17052 pts/32  S      0:00 ./mkthread2
17053 pts/32  S      0:00 ./mkthread2
17054 pts/32  S      0:00 ./mkthread2
```

## § 22.17 Compilation avec threads sous Unix

La librairie pour le link s'appelle traditionnellement par «-lpthread» (pthread  $\equiv$  Posix Thread)

Si votre système n'a pas de thread intégrés, se reporter aux GNU Portable Threads ou <http://www.gnu.org/software/pth/> (download sur <ftp://ftp.lip6.fr/pub/gnu/pth/>)

Il y a des threads non Posix.

Par exemple les threads Solaris, cf «-lthread».

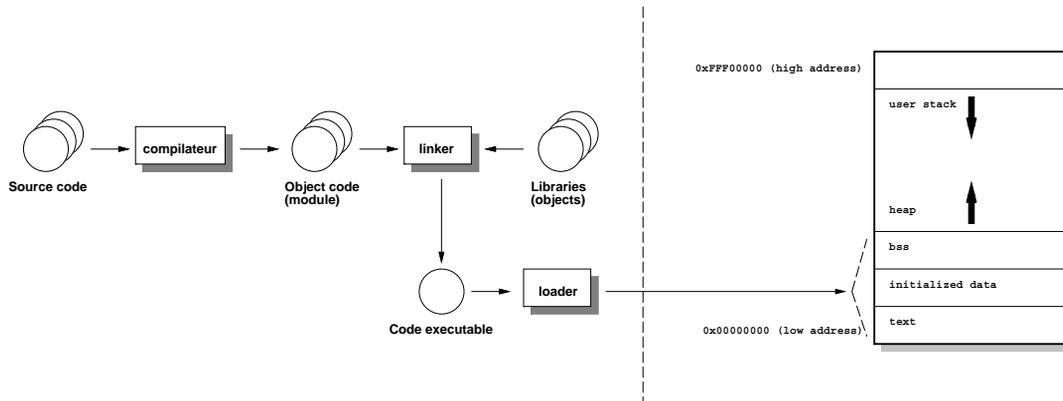
Pour un manuel de programmation, se reporter par exemple au manuel SUN intitulé *Multithread Programming Guide* d'URL <ftp://192.18.99.138/802-5938/802-5938.pdf>

## § 22.18 Chaîne de compilation sous Unix

Apparemment la compilation se résume à

```
% cc toto.c
% ls -lF a.out
-rwxr-xr-x  1 besancon software    24576 Sep 28 23:25 a.out*
```

En fait, c'est beaucoup plus complexe :



Le même exemple de compilation mais en ne masquant plus rien (exemple sur SunOS) :

```
% cc -v toto.c
/lib/cpp -undef -Dunix -Dsun -Dsparc toto.c >/tmp/cpp.13192.0.i
/lib/ccom - </tmp/cpp.13192.0.i >/tmp/ccom.13192.1.s
rm /tmp/cpp.13192.0.i
/bin/as -o toto.o -Q /tmp/ccom.13192.1.s
rm /tmp/ccom.13192.1.s
/bin/ld -dc -dp -e start -X -o a.out /usr/lib/crt0.o toto.o -lc
rm toto.o
% ls -lF a.out
-rwxr-xr-x  1 besancon software    24576 Sep 28 23:26 a.out*
```

On distingue donc plusieurs passes formant la chaîne de compilation :

1. Préprocesseur : commande `cpp`

```
/lib/cpp -undef -Dunix -Dsun -Dsparc toto.c >/tmp/cpp.13192.0.i
```

2. Compilateur C générant le code assembleur : commande `ccom`, `cc1`, ou autre

```
/lib/ccom - </tmp/cpp.13192.0.i >/tmp/ccom.13192.1.s  
rm /tmp/cpp.13192.0.i
```

3. Assembleur générant le code machine : commande `as`

```
/bin/as -o toto.o -Q /tmp/ccom.13192.1.s  
rm /tmp/ccom.13192.1.s
```

4. Edition de liens (assemblage du code machine avec les bibliothèques système) : commande `ld`

```
/bin/ld -dc -dp -e start -X -o a.out /usr/lib/crt0.o toto.o -lc  
rm toto.o
```

On peut bien sûr lancer chaque étape manuellement.

## § 22.19 Edition de liens, link : ld

Le langage C consiste en :

– des types atomiques (int, long, etc.)

– une syntaxe ; par exemple :

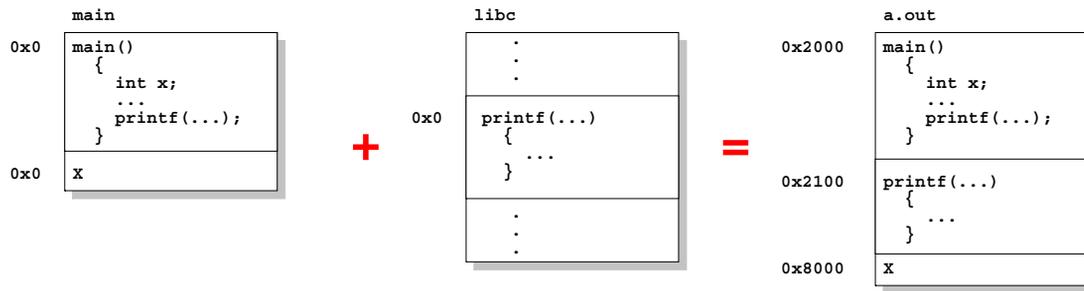
```
• a = b + c ;  
• i++ ;  
• int foo(int a)  
  {  
    return( a / 3 );  
  }
```

Ne font pas partie du langage C :

– les directives du préprocesseur : `#include`, `#ifdef`, etc.

– les fonctions! : `printf()`, `scanf()`, etc.

L'édition de liens consiste en la complémentation du code utilisateur par le code précompilé des fonctions système utilisées. ⇒ emploi de **bibliothèques**.



Comment précise-t-on une bibliothèque à la compilation ?

```
% cc toto.c -lm
```

C'est toujours de la forme **-lbibliothèque**.

Où chercher les bibliothèques de fonctions système ?

Traditionnellement dans : /lib, /usr/lib, /usr/local/lib

C'est précisable au niveau de la ligne de compilation :

```
% cc -Lrépertoire toto.c -lfoo
```

Quels fichiers stockent-ils les bibliothèques ?

Par exemple :

```
% cc prog.c -lrpcsvc -lm
```

utilise les bibliothèques

```
/usr/lib/librpcsvc.a
```

```
/usr/lib/libm.a
```

**Attention : cela dépend en fait du type d'édition de liens :**

- édition de liens statique
- édition de liens dynamique

## § 22.20 Edition de liens statique : bibliothèques statiques

Edition de liens statique : Le code de toutes les fonctions système utilisées est chargé dans l'exécutable.

Inconvénient : l'exécutable est figé, monolithique, auto-suffisant, gros. En cas de problème avec une fonction système (incorrectement codée par exemple), si une nouvelle bibliothèque est disponible auprès du constructeur, il faudra recompiler.

```
% cc -static foo.c -lrpcsvc
```

```
% ls -l a.out
```

```
-rwxr-xr-x 1 besancon software 32768 Oct 8 21:42 a.out
```

⇒ emploi de /usr/lib/librpcsvc.a

**§ 22.21 Edition de liens statique : file**

Comment connaître le type d'édition de liens d'un exécutable a posteriori ?

Utiliser la commande **file** :

```
% gcc -static toto.c
% file a.out
a.out: FreeBSD/i386 compact demand paged executable not stripped
```

Par défaut les compilateurs travaillent en mode dynamic.

**§ 22.22 Edition de liens statique : ar**

Une bibliothèque statique est une collection de modules objet '.o' avec un index.

La commande Unix pour assembler les modules objet et créer la table d'index est : **ar**

```
% ar cq libtoto.a module1.o module2.o ...
```

ou plus généralement

```
% ar [option] bibliotheque module1.o module2.o ...
```

### § 22.23 Edition de liens statique : ranlib

Selon les systèmes Unix, il faut utiliser la commande **ranlib** pour mettre à jour la table des matières (problème de date) :

```
# touch /usr/lib/libm.a
% cc -Bstatic foo.c -lm
ld: /lib/libm.a: warning: table of contents for archive is out \
of date; rerun ranlib(1)
```

Parfois, l'édition de liens échoue uniquement par absence d'un ranlib non fait.

La solution :

```
# ranlib /usr/lib/libm.a
```

### § 22.24 Edition de liens statique : /sbin

Les binaires de `/sbin` sont compilés en static en théorie :

```
% file /sbin/*
/sbin/dump:      FreeBSD/i386 compact demand paged executable
/sbin/fastboot:  FreeBSD/i386 compact demand paged executable
/sbin/fasthalt:  FreeBSD/i386 compact demand paged executable
/sbin/fdisk:     FreeBSD/i386 compact demand paged executable
/sbin/fsck:      FreeBSD/i386 compact demand paged executable
...
/sbin/ifconfig:  FreeBSD/i386 compact demand paged executable
/sbin/init:      FreeBSD/i386 compact demand paged executable
...
/sbin/mknod:     FreeBSD/i386 compact demand paged executable
...
/sbin/mount:     FreeBSD/i386 compact demand paged executable
...
```

Raison : on fait appel à ces binaires à un moment dans la vie de la station Unix où le mécanisme des bibliothèques partagées n'est pas encore activé.

## § 22.25 Edition de liens dynamique : bibliothèques dynamiques

Edition de liens dynamique : On ne chargera **qu'à l'exécution** le code des fonctions système utilisées.

Avantage : l'exécutable n'a pas besoin d'être remis à jour en cas de changement de bibliothèques système.

Inconvénients :

1. l'exécutable est **dépendant** vis-à-vis des bibliothèques système
2. pseudo ralentissement au moment du lancement de l'exécutable ?

```
% cc -dynamic foo.c -lrpcsvc
% ls -lF a.out
-rwxr-xr-x 1 besancon software 24576 Oct 8 21:42 a.out*
```

⇒ emploi de

```
/usr/lib/libc.so.2.0
/usr/lib/librpcsvc.so.3.1
```

## § 22.26 Edition de liens dynamique : file

Comment connaître le type d'édition de liens d'un exécutable a posteriori ?

Utiliser la commande **file** :

```
% gcc toto.c
% file a.out
a.out: FreeBSD/i386 compact demand paged dynamically linked executable \
not stripped
```

Par défaut les compilateurs travaillent en mode dynamic.

**§ 22.27 Edition de liens dynamique : ldd**

Comment connaître la liste des bibliothèques partagées dont a besoin un exécutable ?

⇒ Utiliser la commande **ldd**

```
% gcc toto.c
% ldd a.out
a.out:
    -lc.3 => /usr/lib/libc.so.3.1 (0x2001e000)

% ldd /usr/X11R6/bin/xterm
/usr/X11R6/bin/xterm:
    -lXaw.6 => /usr/X11R6/lib/libXaw.so.6.1 (0x2003e000)
    -lXmu.6 => /usr/X11R6/lib/libXmu.so.6.0 (0x20071000)
    -lXt.6 => /usr/X11R6/lib/libXt.so.6.0 (0x20080000)
    -lX11.6 => /usr/X11R6/lib/libX11.so.6.1 (0x200bc000)
    -lSM.6 => /usr/X11R6/lib/libSM.so.6.0 (0x2014c000)
    -lICE.6 => /usr/X11R6/lib/libICE.so.6.3 (0x20154000)
    -lXext.6 => /usr/X11R6/lib/libXext.so.6.3 (0x20165000)
    -ltermcap.2 => /usr/lib/libtermcap.so.2.1 (0x2016e000)
    -lc.3 => /usr/lib/libc.so.3.1 (0x20172000)
```

Bien sûr :

```
% ldd /sbin/fsck
ldd: /sbin/fsck: not a dynamic executable
```

Une bibliothèque dynamique a un numéro de version :

```
% ldd /usr/X11R6/bin/xterm
/usr/X11R6/bin/xterm:
...
-lSM.6 => /usr/X11R6/lib/libSM.so.6.0 (0x2014c000)
-lICE.6 => /usr/X11R6/lib/libICE.so.6.3 (0x20154000)
-lXext.6 => /usr/X11R6/lib/libXext.so.6.3 (0x20165000)
-ltermcap.2 => /usr/lib/libtermcap.so.2.1 (0x2016e000)
-lc.3 => /usr/lib/libc.so.3.1 (0x20172000)
```

Exemple de message en cas de mismatch de versions d'une bibliothèque partagée :

```
% ./a.out
/usr/libexec/ld.so: warning: /usr/lib/libc.so.3.0: minor version 0 \
older than expected 1, using it anyway
```

## § 22.28 (Windows :: listdlls)

Commande Windows proche chez <http://www.sysinternals.com>:

« listdlls.exe »

```
ListDLLs V2.23 - DLL lister for Win9x/NT
Copyright (C) 1997-2000 Mark Russinovich
http://www.sysinternals.com
```

```
-----
System pid: 4
```

```
Command line: <no command line>
-----
```

```
smss.exe pid: 1304
```

```
Command line: \SystemRoot\System32\smss.exe
```

Base	Size	Version	Path
0x48580000	0xe000		\SystemRoot\System32\smss.exe
0x77f50000	0xa7000	5.01.2600.1217	C:\WINDOWS\System32\ntdll.dll

```
...
```

...

```
-----
cmd.exe pid: 780
Command line: "C:\WINDOWS\System32\cmd.exe"
```

Base	Size	Version	Path
0x4ad00000	0x5e000	5.01.2600.0000	C:\WINDOWS\System32\cmd.exe
0x77f50000	0xa7000	5.01.2600.1217	C:\WINDOWS\System32\ntdll.dll
0x77e60000	0xe6000	5.01.2600.1106	C:\WINDOWS\system32\kernel32.dll
0x77c10000	0x53000	7.00.2600.1106	C:\WINDOWS\system32\msvcrt.dll
0x77d40000	0x8c000	5.01.2600.1255	C:\WINDOWS\system32\USER32.dll
0x7e090000	0x41000	5.01.2600.1346	C:\WINDOWS\system32\GDI32.dll
0x77dd0000	0x8d000	5.01.2600.1106	C:\WINDOWS\system32\ADVAPI32.dll
0x78000000	0x87000	5.01.2600.1361	C:\WINDOWS\system32\RPCRT4.dll
0x75f40000	0x1f000	5.01.2600.1106	C:\WINDOWS\system32\Apphelp.dll

...

## § 22.29 Edition de liens dynamique : ldconfig

Exemple de message en cas d'absence d'une bibliothèque partagée :

```
% ./a.out
ld.so failed: Can't find shared library "libpico.so.1.3"
% ldd a.out
a.out:
/usr/libexec/ld.so: warning: /usr/lib/libc.so.3.0: minor version 0 \
older than expected 1, using it anyway
        -lpico.1 => not found (0x0)
        -lc.3 => /usr/lib/libc.so.3.0 (0x1001d000)
```

⇒ Comment ajouter des bibliothèques partagées au système ?

Méthode d'ajout de bibliothèques partagées :

1. Copier la bibliothèque partagée dans un répertoire système prévu pour cela
2. Faire prendre connaissance au système (en fait au programme **ld.so**) de la présence d'une nouvelle bibliothèque partagée.  
⇒ Utiliser la commande **ldconfig**

## § 22.30 Edition de liens dynamique : effacement de librairies dynamiques

En cas d'effacement de bibliothèques partagées. . .

1. AïE
2. Essayer d'utiliser les binaires static pour réparer
3. Redémarrer sur CDROM
4. Avoir pensé à exporter par NFS les répertoires de bibliothèques partagées de façon à recopier les bibliothèques manquantes par NFS.
5. Avoir compilé en static les outils nécessaires à réparer. C'est possible en utilisant des sources d'Unix du domaine public.

Dans tous les cas, c'est difficile et pas gagné d'avance. . .

## § 22.31 Dynamic Loading

Dynamic linking : le système Unix complète lors de l'exécution d'une application le code binaire de l'application par le code binaire de bibliothèques.

Dynamic loading : l'application complète elle-même son code binaire lors de son exécution par le code binaire de modules.

Avantages du dynamic loading :

- extensions possibles d'une application
- possibilité de publier une API pour écrire des modules sans pour autant dévoiler le code source de l'application ⇒ intéressant pour des ISV

### ◇ Application du dynamic loading : Modules, Plugins

#### Modules Apache

Modules de filtrage, de réécriture de documents HTML, etc.

Cf <http://modules.apache.org/>

#### Modules PAM

Modules d'ajout de procédures d'authentification dans des programmes système (programmes du genre « su », « login », « telnetd », etc.)

Cf <http://www.kernel.org/pub/linux/libs/pam/>

Cf <http://www.solaris.com/solaris/pam/>

#### Modules PostgreSQL

Modules définissant de nouvelles fonctions, de nouveaux opérateurs, de nouveaux types SQL, etc.

Cf <http://gborg.postgresql.org/>

etc.

## ◇ Exemple : application faisant du dynamic loading

```
#include <stdio.h>
#include <dlfcn.h>
#include <link.h>
#include <limits.h>
#include <unistd.h>

main()
{
    void *module_handle;
    int module_mode = RTLD_NOW | RTLD_GLOBAL;
    void (*sym_handle)();
    int dlinfo ;

    printf("Nous sommes dans le fichier %s\n", __FILE__);
    printf("Nous sommes le processus %d\n", getpid());
    printf("sleep(20)\n");
    sleep(20);
    printf("Chargement du module...\n");
    module_handle = dlopen("./module.so.1", module_mode);
```

Suite à la page suivante...

```
if ( module_handle )
{
    printf("Module chargé en = 0x%x\n", module_handle);

    /* Recuperation de l'adresse du symbole */
    sym_handle = dlsym(module_handle, "fonction");
    printf("Symbole chargé en = 0x%x\n", sym_handle);

    /* Appel du symbole */
    (void) (*sym_handle)();

    /* On en a fini avec le module dynamique */
    printf("Déchargement du module...\n");
    dlclose(module_handle);
    printf("sleep(20)\n");
    sleep(20);
    exit(0);
}
else
{
    /* Module non charge */
    printf("Module non chargé.\n");
    exit(1);
}
}
```

◇ Exemple : module pour l'application

```
#include<stdio.h>

void fonction()
{
    printf("Nous sommes dans le fichier %s dans %s()+%d\n",
        __FILE__, __FUNCTION__, __LINE__);
    printf("Nous sommes toujours le processus %d\n", getpid());
    printf("sleep(20)\n");
    sleep(20);
    printf("Fin d'exécution de %s()\n", __FUNCTION__);
}
```

◇ Exemple : lancement de l'application faisant du dynamic loading

```
% ./prog
Nous sommes dans le fichier prog.c
Nous sommes le processus 3021
sleep(20)
Chargement du module...
Module chargé en = 0xff3a1dcc
Symbole chargé en = 0xff2502d0
Nous sommes dans le fichier module.c dans fonction()+6
Nous sommes toujours le processus 3021
sleep(20)
Fin d'exécution de fonction()
Déchargement du module...
sleep(20)
```

Plus exactement :

```

% ./prog
Nous sommes dans le fichier prog.c
Nous sommes le processus 3021
sleep(20)
Chargement du module...
Module chargé en = 0xff3a1dcc
Symbole chargé en = 0xff2502d0
Nous sommes dans le fichier module.c dans fonction()+6
Nous sommes toujours le processus 3021
sleep(20)
Fin d'exécution de fonction()
Déchargement du module...
sleep(20)

% pldd 3021
3021:  ./prog
      /usr/lib/libdl.so.1
      /usr/lib/libc.so.1
      /usr/platform/sun4u/lib/libc_psr.so.1

% pldd 3021
3021:  ./prog
      /usr/lib/libdl.so.1
      /usr/lib/libc.so.1
      /usr/platform/sun4u/lib/libc_psr.so.1
      /var/tmp/dl/module.so.1
      /usr/lib/libld.so.2
      /usr/lib/libelf.so.1

% pldd 3021
3021:  ./prog
      /usr/lib/libdl.so.1
      /usr/lib/libc.so.1
      /usr/platform/sun4u/lib/libc_psr.so.1
      /usr/lib/libld.so.2
      /usr/lib/libelf.so.1

```

## § 22.32 Représentation interne des symboles : Mangling, nm

Le mangling est le passage du nom de fonction exprimé dans le langage de programmation au symbole interne pour la compilation.

Par exemple :

Langage	Système Unix	Fonction	Symbole manglé
Fortran	HP-UX	foo	foo
Fortran	SunOS	foo	_foo_
Fortran	AIX	foo	.foo
C	HP-UX	foo	foo
C	FreeBSD	foo	_foo
C++	FreeBSD	foo	_foo__Fv

Le problème de l'administrateur (lors du portage d'un soft par exemple) :

```
% cc foo.c
/var/tmp/cc1134891.o: Undefined symbol `__printf' referenced from \
text segment
```

Il faut donc :

- savoir identifier la fonction en cause dans le source C.  
La fonction en cause existe-t-elle sur cet Unix ?
- Si la fonction est implémentée, dans quelle bibliothèque la trouve-t-on ?

La commande **nm** répond au point 2.

```
% nm /usr/lib/libm.a
i387_s_tan.o:
00000038 t __arch_select_tan
00000090 D __arch_tan
          U __generic_tan
          U __get_hw_float
0000005c T __i387_tan
00000054 T _tan
...
```

## § 22.33 Table de symboles d'un exécutable : strip

Pour diverses raisons, il y a au sein d'un exécutable une table interne d'informations qui ne servent pas la plupart du temps : **la table des symboles**

Inconvénient : l'exécutable est plus gros que nécessaire à cause de la table des symboles.

Pour supprimer cette table, utiliser la commande **strip**.

```
% gcc foo.c -lm
% file a.out
a.out:      sparc demand paged dynamically linked executable not stripped
% ls -l a.out
-rwxr-xr-x  1 besancon software    32768 Oct  8 22:21 a.out
% strip a.out
% file a.out
a.out:      sparc demand paged dynamically linked executable
% ls -l a.out
-rwxr-xr-x  1 besancon software    24576 Oct  8 22:21 a.out
```

Bien sûr :

```
% file a.out
a.out:      sparc demand paged dynamically linked executable not stripped
% strip a.out
% strip a.out
strip: a.out already stripped
```

## § 22.34 Chaînes de caractères : *strings*

Soit un programme C :

```
#include <stdio.h>
main()
{
    int i = 3;
    printf ("Hello world %d !\n", i);
}
```

Alors :

```
% cc foo.c
% ./a.out
Hello world 3 !
% strings a.out
Hello world %d !
```

Intérêts de la commande **strings** :

- elle permet de trouver certaines informations dans un exécutable sans avoir recours aux sources de l'exécutable
- elle s'applique à tout fichier

La commande sera particulièrement utilisée sur des programmes récupérés sur Internet, sur des applications suspectes. Exemple sur un programme suspect, en fait un mouchard de mots de passe déposé par un pirate informatique :

```
# /usr/ucb/gcore 26704
/usr/ucb/gcore: core.26704 dumped
# strings core.26704
...
log.log
...
```

⇒ révélation du nom du fichier de log; plus qu'à faire un `find`

### § 22.35 Debugging post mortem : **core**

Quand une application se termine anormalement, elle génère une image de son espace mémoire sous la forme d'un fichier **core** :

```
% cat buggy.c
#include <string.h>
main()
{
    char *p = NULL;
    strcpy(p, "boum !");
}

% cc buggy.c
% ./a.out
Segmentation fault (core dumped)
```

A noter que selon les systèmes Unix, le fichier core prend des noms différents :

Hypothèse : le programme en question s'appelle « prog »

Unix	Nom de l'image
SOLARIS	core
FreeBSD	prog.core
LINUX	core.PID

De par sa nature (image mémoire), le fichier core est plein de trous :

```
% df .
Filesystem          1024-blocks  Used Available Capacity Mounted on
/dev/sd2g           1791382 1519809    92435     94%  /users/adm

% gcc big-buggy.c
% ./a.out 10
Allocation de 10 Mo...
So far so good :-)
Bye bye...
Segmentation fault (core dumped)

% ls -ls core
10336 -rw-r--r--  1 besancon software 18911664 Oct  8 23:49 core
% df .
Filesystem          1024-blocks  Used Available Capacity Mounted on
/dev/sd2g           1791382 1530081    82163     95%  /users/adm
```

On constate que  $92435 - 18468 \neq 82163!$   $\Rightarrow$  le fichier contient des trous.

La taille du fichier ne correspond pas à ce qui est utilisé en réalité.

Raison : les trous de l'espace mémoire n'allouent pas de bloc disque. On parle de **fichier sparse**.

Comment créer soi-même un fichier sparse ?

```
#include <unistd.h>
#include <string.h>
#include <fcntl.h>
#include <stdio.h>

main()
{
    int    fd;
    off_t  offset;
    char   *buf = "foo";

    /*
     * Attention ce programme est quick & dirty car on ne verifie
     * aucune valeur de retour des fonctions.
     */
    fd = open("/tmp/sparse", O_WRONLY | O_CREAT, 0644);
    offset = 16 * 1024 * 1024 ;
    lseek(fd, offset, SEEK_SET);
    write(fd, buf, strlen(buf));
    close(fd);
}
```

Après avoir fait tourner le programme précédent, on voit que les blocs intermédiaires ne sont pas alloués :

```
% ls -ls /tmp/sparse
16 -rw-r--r--  1 besancon  bin  16777219 Sep 21 19:17 /tmp/sparse
```

Le caractère "sparse" n'est pas conservé par recopie :

```
% cp /tmp/sparse /tmp/sparse2
% ls -ls /tmp/sparse /tmp/sparse2
  16 -rw-r--r--  1 besancon  bin  16777219 Sep 21 19:17 /tmp/sparse
16400 -rw-r--r--  1 besancon  bin  16777219 Sep 21 19:17 /tmp/sparse2
```

◇ Empêcher la création de fichiers core

Plusieurs solutions :

– créer un répertoire « core » dans le répertoire où un core peut être créé

– commande « ulimit » (selon le shell) :

```
% echo $SHELL
/bin/bash
% ulimit -c 128
% ulimit -a
core file size (blocks)      128
data seg size (kbytes)      unlimited
file size (blocks)          unlimited
open files                   256
pipe size (512 bytes)       10
stack size (kbytes)         8192
cpu time (seconds)          unlimited
max user processes          9845
virtual memory (kbytes)     unlimited
```

– commande « limit » (selon le shell) :

```
% echo $SHELL
/bin/csh
% limit coredumpsize 128
% limit
cputime           unlimited
filesize          unlimited
datasize          unlimited
stacksize         8192 kbytes
coredumpsize     128 kbytes
descriptors       256
memorysize        unlimited
```

## § 22.36 Debugging live d'un runtime : trace, strace, truss

Pour debugger une application qui ne marche pas bien, il peut être intéressant de lancer un suivi du runtime : analyse des appels système (fonctions de la section 2 du manuel) faits par le programme.

En pratique :

- programme trace
- programme truss sur SOLARIS
- programme ktrace et kdump sur FreeBSD
- autres ?

Disponibilité de la commande sur la quasi totalité des systèmes soit de base soit via <http://www.liacs.nl/~wichert/strace/>

Exemple 1 : Connaître le type d'actions de base réalisées par un programme

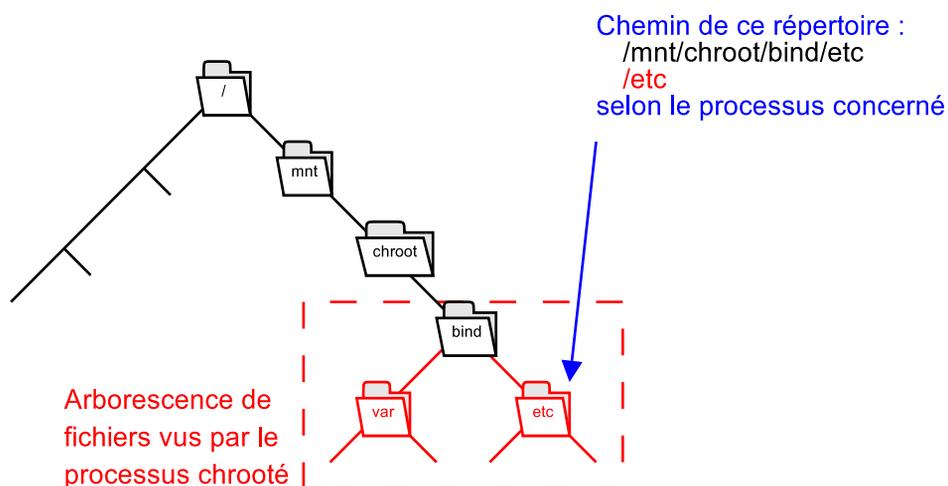
```
close(3) = 0
open("/usr/lib/libc.so.1.9.1", O_RDONLY) = 3
read(3, "\201\3\1\v\0\7\240\0\0\0@\0\0\000"... , 32) = 32
```

Exemple 2 : Pouvoir comprendre les dysfonctionnements de programmes

```
25487 xmgr-3.01p17.LIN RET   execve 0
25487 xmgr-3.01p17.LIN CALL  getitimer(0x1e6d7e)
25487 xmgr-3.01p17.LIN NAMI  "/compat/linux/lib/ld.so"
25487 xmgr-3.01p17.LIN NAMI  "/lib/ld.so"
25487 xmgr-3.01p17.LIN RET   getitimer JUSTRETURN
25487 xmgr-3.01p17.LIN CALL  write(0x2,0xefbfdd9c,0x14)
25487 xmgr-3.01p17.LIN GIO   fd 2 wrote 20 bytes "./xmgr-3.01p17.LINUX"
25487 xmgr-3.01p17.LIN CALL  write(0x2,0xefbfdd20,0x2a)
25487 xmgr-3.01p17.LIN GIO   fd 2 wrote 42 bytes ": can't load dynamic linker '/lib/ld.so' \0"
25487 xmgr-3.01p17.LIN lin{\u}N5{\^I} RET   write 42/0x2a
25487 xmgr-3.01p17.LIN lin{\u}N5{\^I} CALL  exit(0x80)
```

**§ 22.37 Exécution en environnement restreint non sécurisé : *chroot***

« Chrooter » un processus consiste à faire tourner un processus avec une origine de l'arborescence des fichiers différente de « / ».



« chroot » est un appel système : « chroot(2) »

Pour le programme chrooté, les fichiers visibles se limitent à ceux contenus dans la « cage » du chrootage.

Principaux programmes chrootés : des programmes réseau exposés à Internet vers lesquels viennent des requêtes hostiles.

Dans le cadre de chroot, on distingue deux catégories de processus qui rendent les choses plus ou moins complexes à configurer :

- processus nés hors de la cage puis vivant et mourant dans la cage
- processus nés, vivant et mourant dans la cage.

◇ Processus nés hors de la cage puis vivant et mourant dans la cage

Par exemple le logiciel BIND (cf <http://www.isc.org/products/BIND/>)

Le principe général de la programmation en pseudo langage de ce type de processus :

1. vérification que l'on tourne sous l'identité de root
2. faire un « chroot ( ) » dans la cage
3. lecture des fichiers de configuration dans la cage
4. mise en écoute sur certains ports réseau < 1024
5. abandon des privilèges système via le passage sous l'identité d'un utilisateur dédié (pour BIND, il s'agit de l'utilisateur « bind » du groupe « bind »)

Le binaire lancé se limite à accéder à des fichiers de données (en lecture ou en écriture) dans la cage lors de son fonctionnement en mode régime de croisière (c'est-à-dire que l'on passe sous silence la phase d'initialisation de l'application peut-être avec des fichiers hors de la cage mais cette étape est très limitée dans le temps).

◇ Processus nés, vivant et mourant dans la cage

Par exemple un serveur FTPD qui lancerait un binaire « ls » pour répondre à la commande FTP « dir ».

Ce programme « ls » naît, vit et meurt dans la cage.

**La difficulté : les bibliothèques dynamiques**

Rappel :

```
% ldd /bin/ls
      libc.so.1 => /usr/lib/libc.so.1
      libdl.so.1 => /usr/lib/libdl.so.1
      /usr/platform/SUNW,Sun-Blade-100/lib/libc_psr.so.1
```

Implication :

- **Toutes les bibliothèques dynamiques utilisées par des programmes naissant dans une cage doivent exister au sein de la cage.**
- **Tous les devices utilisés par des programmes naissant dans une cage doivent exister au sein de la cage.**

◇ Aspects de sécurité**CHROOT n'a pas été conçu pour de la sécurité !**

En théorie, un processus chrooté ayant les privilèges système peut faire :

```
# mount /dev/hda1 /mnt
```

où /mnt est un répertoire dans la cage

où /dev/hda1 désigne la partition du vrai « / » de la machine.

Le problème de CHROOT : il y a partage de la définition des devices entre tous les processus, chrootés ou non chrootés.

⇒ CHROOT n'a pas été conçu pour de la sécurité. Le renforcement de la sécurité via CHROOT est un pur effet de bord de son fonctionnement.

Autres solutions apportant de la sécurité :

- JAIL sur FREEBSD
- ZONE sur SOLARIS 10

◇ Processus chrootés contre processus non chrootés

Au niveau de la commande « ps », rien ne distingue un processus chrooté d'un processus non chrooté.

Un processus chrooté peut obtenir la liste de tous les processus de la machine. Donc problème en cas de processus chrooté compromis tournant au nom de root car il peut connaître les processus hors cages.

Un processus chrooté peut obtenir la liste de tous les processus de la machine et leur envoyer des signaux. Donc problème en cas de processus chrooté compromis tournant au nom de root car il peut tuer des processus hors cage.

## Chapitre 23 : Horloges Unix

Pour Unix, le début du temps est le 1 janvier 1970.

La fin du temps : le mardi 19 janvier 2038 à 4 h 14 min 7s (heure GMT) (c'est-à-dire 2147483647 secondes  $\equiv$  0x7fffffff)

L'horloge sert à dater les fichiers, lancer des applications à des moments précis voulus (cf cron), dater des événements, etc.

Problèmes classiques : mise à l'heure d'une horloge, consultation d'une horloge, an 2000, etc.

Exemple de problème :

```
% date
Thu Jan 29 11:18:49 CET 2004

% tar xvf z.t
lspci.log
tar: lspci.log: time stamp Jan 29 11:59 2004 is 2434 s in the future
lspci-n.log
tar: lspci-n.log: time stamp Jan 29 12:04 2004 is 2724 s in the future
lspci-t.log
tar: lspci-t.log: time stamp Jan 29 12:03 2004 is 2697 s in the future
lspci-v.log
tar: lspci-v.log: time stamp Jan 29 12:04 2004 is 2734 s in the future
lspci-vv.log
tar: lspci-vv.log: time stamp Jan 29 12:03 2004 is 2689 s in the future
proc-pci.log
tar: proc-pci.log: time stamp Jan 29 12:05 2004 is 2816 s in the future
```

## § 23.1 Consultation de l'horloge : `date`

Syntaxe : `date [+format]`

`% date`

Thu Oct 22 22:00:51 CEST 1998

`date +format` utilise un format à la `printf` avec des `%`.

Cf la page de manuel de la fonction `strftime`.

Les bigrammes les plus utiles :

<code>%H</code>	heure (00..23)
<code>%M</code>	minute (00..59)
<code>%S</code>	seconde (00..61)
<code>%T</code>	temps (hh :mm :ss)
<code>%m</code>	mois, (01..12)
<code>%d</code>	jour, (01..31)
<code>%y</code>	derniers chiffres de l'année (00..99)
<code>%Y</code>	année (1970..)

Exemples d'utilisation :

– Avoir la date sous la forme YYYY/MM/DD :

`% date +%Y/%m/%d`

1998/02/22

– Avoir le jour de la semaine :

`% date +%w`

4

– Avoir le numéro de la semaine :

`% date +%W`

42

Pour des informations mathématiques sur les dates (années bissextiles, calcul du jour de la semaine etc.), on pourra se reporter à :

<ftp://ftp.lip6.fr/pub/doc/faqs/sci-math-faq/dayWeek.gz>

## § 23.2 Commande date améliorée : GNU date

Il existe une commande plus sophistiquée : le GNU date d'URL

`ftp://ftp.lip6.fr/pub/gnu/sh-utils/sh-utils-2.0.tar.gz`

On peut grâce à cette commande calculer des choses impossibles à calculer autrement :

```
% date
Wed Aug  5 11:09:06 MET DST 1998

% date '+%Y/%m/%d'
1998/08/05
% date '+%Y/%m/%d' --date "10 days"
1998/08/15
% date '+%Y/%m/%d' --date "-10 days"
1998/07/26
% date '+%Y/%m/%d' --date "1 years 2 months 3 days"
1999/10/08
% date '+%Y/%m/%d' --date "-1 years -2 months -3 days"
1997/06/02
% date '+%Y/%m/%d' --date "1 fortnight"
1998/08/19
```

## § 23.3 Réglage de l'horloge : date

La commande date permet de régler l'horloge de la machine.

Syntaxes :

```
date [options] [mmd] HHMM
date [options] mmdHHMM[cc]yy[.SS]
```

Par exemple

```
# date
Fri Sep 11 11:40:33 CEST 1998

# date 09110938
Fri Sep 11 09:38:00 CEST 1998

# date
Fri Sep 11 09:38:04 CEST 1998
```

Attention aux commandes date non compatibles an 2000.

### § 23.4 Heure d'été – Heure d'hiver

Le réglage de l'heure d'été, heure d'hiver se fait au niveau des fichiers suivants :

AIX	/etc/environment
DEC	/etc/zoneinfo/localtime
FreeBSD	/etc/localtime
HP-UX 10	/etc/TIMEZONE
IRIX	/etc/TIMEZONE
Linux	/usr/lib/zoneinfo/localtime
SunOS	/usr/share/lib/zoneinfo/localtime
Solaris	/etc/TIMEZONE

Pour la France on prend **MET-1DST** (Middle European Time Daylight Saving Time)

La variable d'environnement **TZ** permet de changer de fuseau horaire lors de l'affichage de `date`.

### § 23.5 Synchronisation d'horloges : NTP (Network Time Protocol)

Autre solution : Network Time Protocol

Cf :

- <http://www.eecis.udel.edu/~ntp/>
- <http://www.cru.fr/NTP/>
- <http://www.sun.com/blueprints/0701/NTP.pdf>
- <http://www.sun.com/blueprints/0801/NTPpt2.pdf>
- <http://www.sun.com/blueprints/0901/NTPpt3.pdf>

La synchronisation se fait en considérant l'heure de plusieurs machines de références (équipées par exemple d'horloges atomiques) et en tenant compte des délais de transmission via le réseau.

Pour une liste de serveurs de temps en France, se reporter à

[http://www.cru.fr/NTP/serveurs\\_francais.html](http://www.cru.fr/NTP/serveurs_francais.html)

En pratique :

1. on choisit des serveurs de temps
2. on désigne une machine du service comme serveur d'heure pour le service
3. on y installe le package NTP ce qui se traduit par lancer le démon `xntpd`
4. sur les machines du service, faire "`ntpdate serveur-ntp`" à intervalle régulier

Garde fou : on ne met pas la machine à l'heure si la différence d'heures est trop grande :

```
Sep 11 02:06:47 example.com xntpd[93]: system event 4 status c621
Sep 11 02:06:47 example.com xntpd[93]: time error -7390.467777 is way too large (set clock manually)
Sep 11 02:06:47 example.com xntpd[93]: system event 2 status c634
```

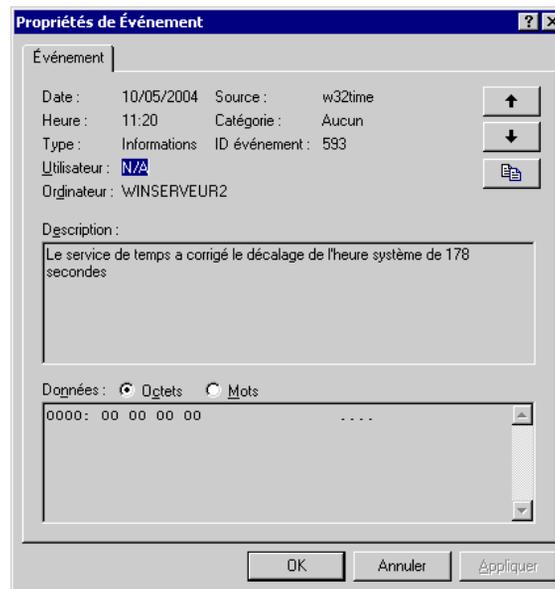
### § 23.6 (Windows :: NTP)

Windows 2000 comporte des outils de synchronisation d'horloges (à cause de Active Directory)

Mettre à jour se fait de la façon suivante :

```
set time /setsntp:ntp.example.com
w32tm -once
```

Trace d'une mise à l'heure :



## § 23.7 Faux ami : commande `time`

A un programme sont associés :

- le temps d'exécution de code de bas niveau du système (par exemple écrire sur disque)
- le temps d'exécution de code de haut niveau de l'utilisateur (par exemple calculer une fonction mathématique)

La commande « `time` » fournit ces temps.

Syntaxe : `time commande parametres`

Par exemple :

```
% time gzip -v archive.tar
```

```
archive.tar:                26.8% -- replaced with archive.tar.gz
```

```
real    0m0.485s
```

```
user    0m0.390s
```

```
sys     0m0.070s
```

Attention, subtilité :

- il existe un builtin du shell appelé « time »
- il existe une commande « time » : « /bin/time »

```
% echo $SHELL
/bin/bash
% type time
time is a shell keyword
% time gzip -v archive.tar
archive.tar:                26.8% -- replaced with w.tar.gz

real    0m0.485s
user    0m0.390s
sys     0m0.070s

% /bin/time gzip -v archive.tar
archive.tar:                26.8% -- replaced with w.tar.gz

real    0.4
user    0.3
sys     0.0
```

## Chapitre 24 : Programmation de tâches périodiques

### § 24.1 Utilitaire cron

L'administrateur doit lancer périodiquement des actions (nettoyage de /tmp, remise à l'heure par rapport à une horloge de référence, etc.). Il existe un système automatisant ces lancement : **cron**

Le démon **cron** est lancé par les scripts de démarrage du boot :

Sur Linux :

```
% ls -l /etc/rc.d/init.d/crond
-rwxr-xr-x 1 root root 1031 Feb 3 2000 /etc/rc.d/init.d/crond
```

Sur Solaris :

```
% ls -l /etc/init.d/crond
-rwxr--r-- 4 root sys 513 Jul 16 1997 /etc/init.d/cron
```

### § 24.2 Fichiers crontab

Les actions à lancer périodiquement sont indiquées dans un fichier texte au format :

```
minutes heures jour-du-mois mois jour-de-semaine commande
```

- champ 1 : minutes, de 00 à 59.
- champ 2 : heures, de 00 à 23.
- champ 3 : jour du mois, de 1 à 31.
- champ 4 : mois, de 1 à 12.
- champ 5 : jour de la semaine, de 0 (dimanche) à 6 (samedi).
- champ 6 : commande à exécuter.

Le caractère **\*** sert de caractère joker pour n'importe quel des champs 1 à 5.

Le caractère **#** introduit un commentaire.

Par exemple :

```
15 * * * * /etc.local/cron/scripts/ntpdate
*/2 * * * * /etc.local/cron/scripts/check-devnull
00 21 * * 1 /etc.local/cron/scripts/tartare
00 21 * * 3 /etc.local/cron/scripts/tartare
00 21 * * 5 /etc.local/cron/scripts/tartare
```



Après son exécution, on trouve dans sa boîte aux lettres :

```
From besancon Sat Aug 10 23:20:00 2002
Return-Path: <besancon>
Received: (from besancon@localhost)
    by solaris.example.com (8.11.6+Sun/8.11.6) id g7ALK0W23530
    for besancon; Sat, 10 Aug 2002 23:20:00 +0200 (MEST)
Date: Sat, 10 Aug 2002 23:20:00 +0200 (MEST)
From: Thierry Besancon <besancon>
Message-Id: <200208102120.g7ALK0W23530@solaris.example.com>
To: besancon
Subject: Output from "cron" command
Content-Type: text
Content-Length: 118
```

```
Your "cron" job on solaris.example.com
/tmp/demo
```

produced the following output:

```
blabla bla bla
erreuuurrrrrrrrrrrrrrrrrrr
```

## § 24.4 Edition des fichiers `crontab`

Les fichiers `crontab` ont beau être au format texte, il ne faut pas les éditer manuellement. Il faut passer par l'intermédiaire de la commande `crontab`. Il y a deux façons de faire selon les versions de la commande :

1. `crontab -e`

Cela lance l'éditeur de texte indiqué par la variable d'environnement `EDITOR` ou bien `vi` par défaut.

2. `crontab -l > myfile`

```
vi myfile
```

```
crontab myfile
```

```
rm myfile
```

**§ 24.5 Autorisation à utiliser cron, cron.allow, cron.deny**

Les fichiers cron.allow et cron.deny contrôlent l'utilisation de cron.

Système	Program
Linux	/etc/cron.allow
	/etc/cron.deny
Solaris	/etc/cron.d/cron.allow
	/etc/cron.d/cron.deny

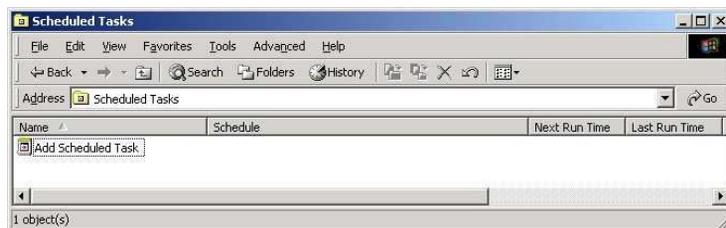
Ces 2 fichiers ont la même syntaxe : ils contiennent des noms de login.

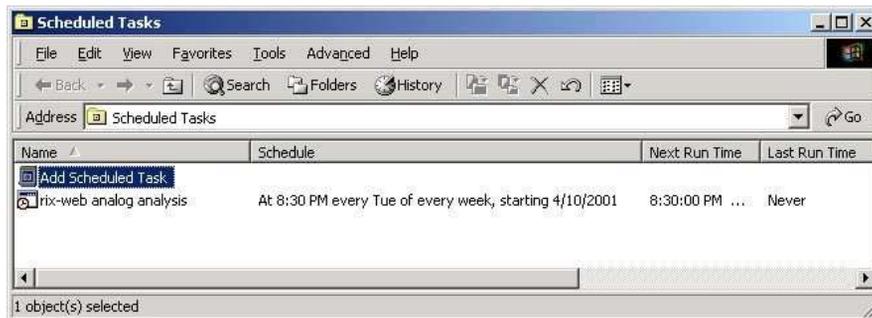
cron.allow	cron.deny	Utilisateurs autorisés
Présent	Présent	Ceux explicitement dans cron.allow
Présent	-	Ceux explicitement dans cron.allow
-	Présent	Tous sauf ceux dans cron.deny
-	-	Uniquement root

Il est conseillé de ne laisser l'accès à cron qu'à l'utilisateur root.

**§ 24.6 (Windows : : cron)**

Méthode de base fournie dans Windows 2000 : programmation de tâches



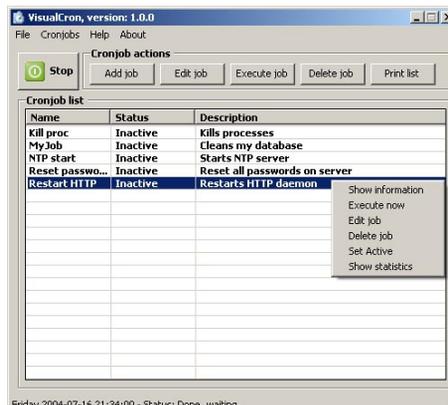


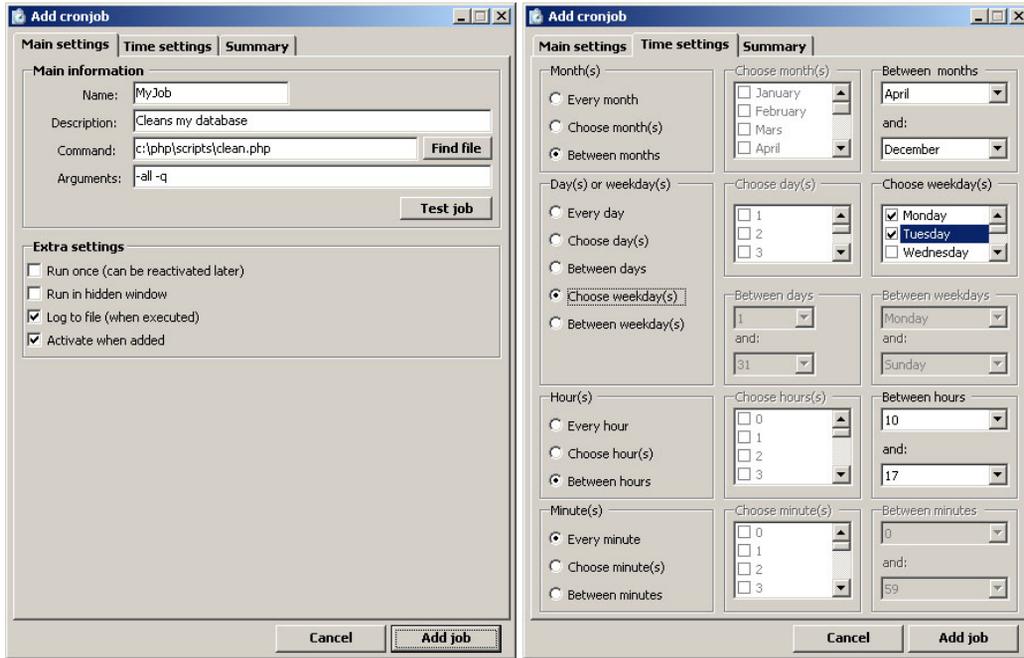
Sinon plusieurs utilitaires à la Unix.

Utilitaire visual-cron, <http://visual-cron.com>

Avis :

- un peu encore windows-like
- pour l'administrateur uniquement sauf erreur de ma part
- manque action au boot, au shutdown (mais comme sous Unix)

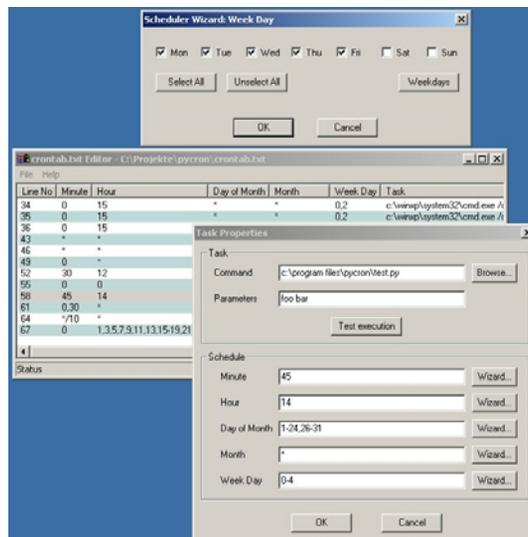




Utilitaire pycron, <http://www.kalab.com/freeware/pycron/pycron.htm>

Avis :

- pour l'administrateur uniquement sauf erreur de ma part
- manque action au boot, au shutdown (mais comme sous Unix)



Utilitaire nncron-lite, <http://www.nncron.ru/>

**Pour ARS 2005-2006...**

**Chapitre 25 : Traces/Journaux système : SYSLOG****§ 25.1 Principes**

SYSLOG est le mécanisme standard de distribution des messages envoyés par le noyau et les programmes (systèmes ou autres) :

- Il fonctionne au dessus de UDP (port 514), donc en mode non connecté. Cf RFC3164.
- Il agit comme un routeur des messages.
- Il n'utilise aucune authentification des clients SYSLOG.
- Le programme associé s'appelle `syslogd`
- D'origine BSD. Améliorations sur certains LINUX.

**§ 25.2 Exemples de messages SYSLOG**

```
Nov 11 02:00:02 example.org pppd: pppd 2.4.0b1 started by besancon, uid 1000
Nov 11 02:00:21 example.org pppd: Serial connection established.
Nov 11 02:00:21 example.org pppd: Using interface sPPP0
Nov 11 02:00:21 example.org pppd: Connect: sPPP0 <--> /dev/ttya
Nov 11 02:00:22 example.org pppd: LCP: Rcvd Code-Reject for Identification id 13
Nov 11 02:00:22 example.org pppd: Remote message: Login ok
Nov 11 02:00:22 example.org pppd: local IP address 192.168.0.30
Nov 11 02:00:22 example.org pppd: remote IP address 192.168.0.100
Nov 11 02:05:07 example.org sPPPasyn: sPPP0: bad fcs (len=352)
Nov 11 02:05:59 example.org pppd: Terminating on signal 15.
Nov 11 02:05:59 example.org pppd: Connection terminated.
Nov 11 02:05:59 example.org pppd: Connect time 5.7 minutes.
Nov 11 02:05:59 example.org pppd: Sent 60962 bytes (4074 packets), received 1086491 bytes
(4142 packets).
Nov 11 02:06:00 example.org pppd: Exit.

Nov 11 11:05:41 example.org usba: /pci@1f,0/usb@c,3 (ohci0): getting device status failed
Nov 11 11:05:41 example.org usba: USB-device: storage@3, scsa2usb0 at bus address 5
Nov 11 11:05:41 example.org usba: M-Systems DiskOnKey 0415A20E29003852
Nov 11 11:05:41 example.org genunix: scsa2usb0 is /pci@1f,0/usb@c,3/storage@3
Nov 11 11:05:41 example.org genunix: /pci@1f,0/usb@c,3/storage@3 (scsa2usb0) online
Nov 11 11:05:41 example.org scsi: sd16 at scsa2usb0: target 0 lun 0
Nov 11 11:05:41 example.org genunix: sd16 is /pci@1f,0/usb@c,3/storage@3/disk@0,0
```

### § 25.3 Format des messages SYSLOG

Pour router les messages SYSLOG, ces messages sont transmis au niveau réseau avec une étiquette.

L'étiquette est composée :

- de la date
- du nom de la classe générique du programme émetteur : c'est le **facility**  
Principaux noms rencontrés : auth, authpriv, cron, daemon, ftp, kern, lpr, mail, mark, news, ntp, syslog, user, uucp, local0 à local7  
\* est un caractère joker.
- du nom du niveau de la priorité du message : c'est le **level**  
Principaux noms rencontrés : emerg, alert, crit, err, warning, notice, info, debug  
\* est un caractère joker (non disponible sur Solaris)

Cf <syslogd.h>

Par exemple :

```
#include <syslog.h>

main(int argc, char *argv[])
{
    openlog("ftpd", LOG_PID, LOG_DAEMON);
    syslog(LOG_INFO, "Connection from host %s", "333");
    syslog(LOG_INFO|LOG_LOCAL2, "error: %m");
    closelog();
}
```

ce qui donne les messages :

```
Nov 11 11:33:09 example.org ftpd[10658]: Connection from host 333
Nov 11 11:33:09 example.org ftpd[10658]: error: Bad file number
```

## § 25.4 Commande shell logger

On peut programmer en langage C :

```
#include <syslog.h>

main(int argc, char *argv[])
{
    openlog("ftpd", LOG_PID, LOG_LOCAL3);
    syslog(LOG_NOTICE, "Essai en langage C") ;
    closelog();
}
```

mais non pratique si on veut l'utiliser au niveau du shell ⇒ commande logger au niveau du shell.

Syntaxe: logger [-p facility.level] message

```
% logger -p local3.notice Essai de la commande logger
```

```
% grep Essai /var/log/messages
```

```
Nov 16 14:56:18 example.com ftpd[1098]: Essai en langage C
```

```
Nov 16 14:58:03 example.com besancon: Essai de la commande logger
```

## § 25.5 Fichier de configuration : /etc/syslog.conf

SYSLOG agit comme un routeur des messages :

- Les messages sont émis par les programmes et reçus sur chaque machine par le démon syslogd.
- syslogd décide quoi faire des messages sur la base de son fichier de configuration, /etc/syslog.conf.

Syntaxe du fichier /etc/syslog.conf (se reporter à man syslog.conf):

```
facility.level[;facility.level]*\taction
```

\t indique le caractère tabulation. Notation du printf du langage C.

**La tabulation est indispensable !**

Par "action", on désigne les possibilités de distribution des messages :

- distribution des messages dans des fichiers  
Indiquer pour cela un nom de fichier via son path absolu (commençant donc par un /);
- aiguillage des messages vers le démon d'une autre machine  
Indiquer pour cela un nom de machine ou son adresse IP, précédé de @;
- affichage des messages sur un terminal particulier  
Indiquer pour cela le path absolu du terminal;
- affichage des messages sur les terminaux de certains utilisateurs énumérés  
Indiquer pour cela une liste des logins des utilisateurs (séparés par une virgule); par exemple  
root, operator  
utiliser \* pour atteindre tous les utilisateurs connectés

## § 25.6 Exemple de fichier /etc/syslog.conf

Exemple de fichier /etc/syslog.conf Solaris :

```
#
# syslog configuration file.
#

*.err;kern.notice;auth.notice           /dev/sysmsg
*.err;kern.debug;daemon.notice;mail.crit /var/adm/messages

*.alert;kern.err;daemon.err             operator
*.alert                                   root
*.emerg                                   *

mail.debug                               @sysloghost.example.com
```

## § 25.7 Conseils pour /etc/syslog.conf

Personnellement, le fichier de ma machine contient :

```
*.* @sysloghost.example.com
```

ou version Solaris :

```
*.emerg @sysloghost.example.com
*.alert @sysloghost.example.com
*.crit @sysloghost.example.com
*.err @sysloghost.example.com
*.warning @sysloghost.example.com
*.notice @sysloghost.example.com
*.info @sysloghost.example.com
*.debug @sysloghost.example.com
```

Il redirige tout message vers la machine appelée `sysloghost.example.com` qui centralise et stocke ce message.

**Il vaut mieux avoir trop de messages que pas du tout ou pas ceux qui sont intéressants.**

Sur la machine appelée `sysloghost.example.com`, on stockera par exemple les messages de la façon suivante :

```
*.* /var/log/syslog
```

ou version Solaris :

```
*.emerg /var/log/syslog
*.alert /var/log/syslog
*.crit /var/log/syslog
*.err /var/log/syslog
*.warning /var/log/syslog
*.notice /var/log/syslog
*.info /var/log/syslog
*.debug /var/log/syslog
```

## § 25.8 Reconfiguration, SIGHUP

syslogd est lancé par les scripts de démarrage.

Si l'on modifie le fichier `/etc/syslog.conf`, la modification est prise en compte en envoyant le signal SIGHUP au processus syslogd :

```
# ps -edf | grep syslogd
  root   339      1  0   Aug 04 ?           0:18 /usr/sbin/syslogd
# kill -HUP 339
```

ce qui se traduit par l'émission du message SYSLOG :

```
Aug 19 15:02:34 example.com syslogd: configuration restart
```

Le signal SIGTERM arrête le processus ce qui se traduit par l'émission du message SYSLOG :

```
Aug 19 15:02:54 example.com syslogd: going down on signal 15
```

## § 25.9 Surveillance des logs

Rien ne sert de configurer syslogd si l'on n'inspecte pas les logs recueillis.

Plusieurs méthodes :

- méthode artisanale via `tail -f fichier-de-log` pour afficher en temps réel l'arrivée des messages
  
- méthode plus sophistiquée via certains programmes dédiés à cela ; liste non exhaustive :
  - logiciel `xlogmaster` ; cf <http://www.gnu.org/software/xlogmaster/>
  - logiciel `awacs` ; cf <http://www.gnu.org/software/awacs/>
  - logiciel `swatch` ; cf <ftp://ftp.stanford.edu/general/security-tools/swatch/>

Pour résoudre le problème des logs qui grossissent : `trimlog`

<http://online.securityfocus.com/tools/118>

## § 25.10 Autres implémentations de SYSLOG

Il y a plusieurs autres implémentations de démons SYSLOG.

- Le démon `syslogd` de Digital Unix introduit un mot clef spécial `syslog.dated` :

```
kern.debug          /var/adm/syslog.dated/kern.log
mail.debug          /var/adm/syslog.dated/mail.log
daemon.debug       /var/adm/syslog.dated/daemon.log
```

Le mot `syslog.dated` sera remplacé par la date du jour ce qui donnera :

```
/var/adm/syslog.dated/13-Sep-11:57/*.log
/var/adm/syslog.dated/14-Sep-11:57/*.log
/var/adm/syslog.dated/15-Sep-11:58/*.log
```

- `nssyslogd` disponible à l'URL

<http://coombs.anu.edu.au/~avalon/nssyslog.html>

- Syslog-ng tries to fill the gaps original `syslogd`'s were lacking :
  - powerful configurability
  - filtering based on message content
  - message integrity, message encryption (not yet implemented in 1.4.x)
  - portability
  - better network forwarding

<http://www.balabit.hu/products/syslog-ng/>

**§ 25.11 (Windows : : SYSLOG)**

Windows NT, 2000, XP utilisent des mécanismes de journaux et non pas SYSLOG. Les événements sont des données en binaire.

Mais il existe des programmes apportant SYSLOG à Windows.

## ◇ BackLog



On voit bien sur Unix :

```
Nov 11 21:53:08 dell.example.org Application 14 Tue Nov 11 21:53:06 2003 \
23      Norton AntiVirus          N/A      N/A      Information      VAGABOND \
      Protection temps réel Symantec AntiVirus chargée.
```

Gratuit... Cf

<http://www.intersectalliance.com/projects/BackLogNT/setup.exe>

## § 25.12 Avenir de SYSLOG

Possibilités à venir ? :

- couplage de `syslogd` à une base de données genre SQL pour stocker les messages et permettre des recherches dedans
- mécanisme de chiffrement pour répondre à la non falsification  
<http://www.corest.com/products/corewisdom/CW01.php>

## Chapitre 26 : Base de données ORACLE

Plusieurs moteurs de bases de données disponibles :

- Oracle
- Microsoft SQL Server
- Microsoft MSDE (version light et gratuite de SQL Server)
- IBM DB2
- Sybase
- MYSQL
- POSTGRESQL

ORACLE est la référence en bases de données.

Version de ORACLE disponible au moment de l'écriture de ce document : 10g

Site officiel : <http://www.oracle.com>

A noter qu'Oracle diffuse maintenant des versions gratuites 100% fonctionnelles pour LINUX, pour SOLARIS, pour WINDOWS.

A noter le livre «*Oracle 9i sous Linux*» par Gilles Briard, éditeur Eyrolles, prix 50 euros, livré avec la distribution Oracle 9i complète pour Linux. (je n'ai pas lu le livre)

**Mettre a jour...**

## Chapitre 27 : Base de données MYSQL

Plusieurs moteurs de bases de données disponibles :

- Oracle
- Microsoft SQL Server
- Microsoft MSDE (version light et gratuite de SQL Server)
- IBM DB2
- Sybase
- MYSQL
- POSTGRESQL

Version de MYSQL disponible au moment de l'écriture de ce document : 4.1.7 (2004-10-30)

Site officiel : <http://www.mysql.com>

Changement de la licence de MYSQL dans la version 4. Mal reçu par la communauté des développeurs ⇒ Par exemple dans PHP 5, il n'y a plus de support d'origine pour MYSQL.

### § 27.1 Utilisateur Unix dédié à MYSQL : `mysql`

Etape préliminaire à l'installation mais indispensable.

Le bon fonctionnement de MYSQL nécessite la définition d'un utilisateur au sens Unix qui s'appelle « `mysql` ».

On créera aussi un groupe Unix « `mysql` ».

## § 27.2 Arborescence MySQL

Par défaut, les sources de MySQL s'installeront dans `/usr/local/mysql`.

On trouvera alors dans `/usr/local/mysql` :

- sous-répertoire `bin` : binaires de l'application
- sous-répertoire `include/mysql` : headers de programmation C
- sous-répertoire `info` : fichiers de documentation
- sous-répertoire `lib/mysql` : bibliothèques partagées
- sous-répertoire `libexec` : binaires « système » de l'application
- sous-répertoire `share/mysql` : fichiers des textes des erreurs
- sous-répertoire `sql-bench` : fichiers de tests
- sous-répertoire `var` : fichiers de données et de log

A noter le fichier « `var/my.cnf` » qui définit la configuration de MySQL.

**L'installation de MySQL doit se compléter de la création des bases initiales de MySQL.**

## § 27.3 Création des bases initiales MySQL : `mysql_install_db`

(en anglais *MySQL install databases*)

Syntaxe : `mysql_install_db`

Par défaut, c'est le répertoire « `var` » dans l'arborescence MySQL (par défaut donc « `/usr/local/mysql/var` »).

La création des bases initiales se fera sous l'identité de l'utilisateur « `mysql` ».

```
% id
uid=6000(mysql) gid=6000(mysql)
```

```
% mysql_install_db
Installing all prepared tables
Fill help tables
```

To start mysqld at boot time you have to copy support-files/mysql.server to the right place for your system

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !  
To do so, start the server, then issue the following commands:  
/usr/local/mysql/bin/mysqladmin -u root password 'new-password'  
/usr/local/mysql/bin/mysqladmin -u root -h furax.unixiens.org password 'new-password'  
See the manual for more instructions.

You can start the MySQL daemon with:  
cd /usr/local/mysql ; /usr/local/mysql/bin/mysqld\_safe &

You can test the MySQL daemon with the benchmarks in the 'sql-bench' directory:  
cd sql-bench ; perl run-all-tests

Please report any problems with the /usr/local/mysql/bin/mysqlbug script!

The latest information about MySQL is available on the web at  
<http://www.mysql.com>

~~Support MySQL by buying support/licenses at <https://order.mysql.com>~~  
Formation permanente – ARS 7.0

490

ATTENTION : Finaliser l'installation par la mise en place d'un mot de passe pour le compte SQL de nom « root » :

```
% /usr/local/mysql/bin/mysqladmin -u root password 'XXXXXXXX'
```

**§ 27.4 lancement/arrêt de MySQL**

Lancer sous l'identité de mysql le programme « bin/mysqld\_safe » sous l'identité Unix « mysql ».

**§ 27.5 Création de bases MySQL (1) : mysqladmin**

La création de bases normales MySQL se fait par la commande « mysqladmin ».

Syntaxe : `mysqladmin [options] create base`

◇ Exemple 1 : création d'une base

```
% mysqladmin -u root -p create ars
Enter password: XXXXXXXX
```

◇ Exemple 2 : suppression d'une base

```
% mysqladmin -u root -p drop ars
Enter password: XXXXXXXX
Dropping the database is potentially a very bad thing to do.
Any data stored in the database will be destroyed.
```

```
Do you really want to drop the 'base' database [y/N] y
Database "base" dropped
```

**§ 27.6 Création de bases MySQL (2) : commandes SQL**

```
% mysql -u root -pXXXXXXXXX
mysql> create database ars ;
Query OK, 1 row affected (0.01 sec)

mysql> show databases ;
+-----+
| Database |
+-----+
| ars      |
| mysql    |
| test     |
+-----+
3 rows in set (0.01 sec)

mysql> drop database ars ;
Query OK, 0 rows affected (0.02 sec)

mysql> show databases ;
+-----+
| Database |
+-----+
| mysql    |
| test     |
+-----+
2 rows in set (0.01 sec)
```

**§ 27.7 Dialogue interactif avec MySQL en mode CLI : `mysql`**

La commande « `mysql` » permet d'interagir avec MySQL et de lui demander d'exécuter des commandes SQL. Cela se présente comme un shell spécialisé en SQL.

Syntaxe : `mysql [options] base`

De nombreuses commandes sont ensuite disponibles.

**◇ Exemple 1 : connexion et déconnexion**

```
% mysql -u root -pXXXXXXXXX ars
...
mysql> quit ;
Bye
```

◇ Exemple 2 : liste des bases

```
% mysql -u root -pXXXXXXXXX
...
mysql> show databases ;
+-----+
| Database |
+-----+
| ars      |
| mysql    |
| test     |
+-----+
3 rows in set (0.01 sec)

mysql> quit ;
Bye
```

◇ Exemple 3 : liste des tables systèmes

```
% mysql -u root -pXXXXXXXXX mysql
...
mysql> show tables ;
+-----+
| Tables_in_mysql |
+-----+
| columns_priv    |
| db              |
| func            |
| help_category   |
| help_keyword    |
| help_relation   |
| help_topic      |
| host            |
| tables_priv     |
| time_zone       |
| time_zone_leap_second |
| time_zone_name  |
| time_zone_transition |
| time_zone_transition_type |
| user            |
+-----+
15 rows in set (0.00 sec)
```

## ◇ Exemple 4 : liste des utilisateurs

```
% mysql -u root -pXXXXXXXXX mysql
...
mysql> select User from user order by User asc ;
+-----+
| User |
+-----+
|      |
|      |
| root |
| root |
+-----+
4 rows in set (0.03 sec)
```

**§ 27.8 Notion d'utilisateurs MySQL**

Une requête SQL est émise au nom d'un utilisateur fictif interne à la base SQL.

Un utilisateur SQL n'est pas nécessairement un vrai utilisateur Unix.

La notion d'utilisateurs SQL est utilisée dans le contrôle des requêtes SQL :

```
GRANT privilège[,privilège...]
  ON {table | * | *.* | base.*}
  TO utilisateur [IDENTIFIED BY [PASSWORD] 'PASSWORD']
  [, utilisateur [IDENTIFIED BY [PASSWORD] 'PASSWORD']] ...
```

où :

- les privilèges sont parmi les mots clef « SELECT », « INSERT », « UPDATE », « DELETE », « RULE », « ALL »
- utilisateur désignera un utilisateur bien précis qui obtient les privilèges indiqués

◇ Exemple : ajout de droits pour un utilisateur

```
% mysql -u root -pXXXXXXXXX mysql
...
mysql> grant all on ars.* to besancon@localhost identified by 'XXXXXXXXX' ;
Query OK, 0 rows affected (0.02 sec)
```

◇ Exemple : révocation de droits pour un utilisateur

```
% mysql -u root -pXXXXXXXXX mysql
...
mysql> revoke all privileges on ars.* from besancon@localhost ;
Query OK, 0 rows affected (0.02 sec)
```

**§ 27.9 Fichier de configuration : my.cnf****A completer...**

**§ 27.10 Sauvegarde : *mysqldump***

MySQL fournit un outil pour sauvegarder au format SQL sous forme de commandes SQL une ou plusieurs bases.

Pour sauver une base, utiliser la commande « *mysqldump* ».

Syntaxe : *mysqldump* -u root -p [options] base

Pour sauver toutes les bases, utiliser la commande « *mysqldump --all-databases* ».

Syntaxe : *mysqldumpall* -u root -p [options]

**Attention : joindre à la sauvegarde le fichier *my.cnf* qui n'est pas sauvegardé par l'outil SQL *mysqldump*.**

**◇ Exemple 1 : sauvegarde d'une seule base**

```
% mysqldump -u root -pXXXXXXXXX base > /tmp/base.sql

% ls -l /tmp/base.sql
-rw-r--r--  1 besancon adm      15054229 Oct 23 12:12 /tmp/base.sql

% head /tmp/base.sql
-- MySQL dump 10.8
--
-- Host: localhost      Database: ars
--
-----
-- Server version      4.1.7-log
...
DROP TABLE IF EXISTS `t1`;
CREATE TABLE `t1` (
...

```

## ◇ Exemple 2 : sauvegarde de toutes les bases

```
% mysqldump -u root -pXXXXXXXX --all-databases > /tmp/bases.sql

% ls -l /tmp/bases.sql
-rw-r--r--  1 besancon adm      15064263 Oct 23 12:18 /tmp/bases.sql

% head /tmp/base.sql
-- MySQL dump 10.8
--
-- Host: localhost      Database:
--
-----
-- Server version      4.1.7-log
...
CREATE DATABASE /*!32312 IF NOT EXISTS*/ `ars`;

USE `ars`;
...
```

**§ 27.11 Rechargement d'une sauvegarde**

S'il s'agit de recharger la sauvegarde d'une base, faire sous l'identité de l'utilisateur « *mysql* » :

```
% mysql -u root -pXXXXXXXX < /tmp/base.sql
```

S'il s'agit de recharger la sauvegarde de la totalité des bases, faire sous l'identité de l'utilisateur « *mysql* » :

```
% mysql -u root -pXXXXXXXX < /tmp/bases.sql
```

## § 27.12 Interface de gestion de MySQL : *phpmyadmin*

A noter un interface WWW de configuration et d'administration : *phpmyadmin*

Cf <http://www.phpmyadmin.net/>

**Figure manquante...**

## Chapitre 28 : Base de données POSTGRESQL

(On rencontre souvent le mot PGSQL pour POSTGRESQL)

Plusieurs moteurs de bases de données disponibles :

- Oracle
- Microsoft SQL Server
- Microsoft MSDE (version light et gratuite de SQL Server)
- IBM DB2
- Sybase
- MYSQL
- POSTGRESQL

Version de PGSQL disponible au moment de l'écriture de ce document : 7.4.6 (2004-10-29)

Site officiel : <http://www.postgresql.org>

Système de réplication possible mais cela semble un produit jeune.

**A compléter...**

vue comparative avec d'autres SGBD ?

## § 28.1 Utilisateur Unix dédié à PGSQL : `pgsql`

Etape préliminaire à l'installation mais indispensable.

Le bon fonctionnement de PGSQL nécessite la définition d'un utilisateur au sens Unix qui s'appelle « `pgsql` ».

On créera aussi un groupe Unix « `pgsql` ».

## § 28.2 Arborescence PGSQL

Par défaut, les sources de PGSQL s'installeront dans `/usr/local/pgsql`.

On trouvera alors dans `/usr/local/pgsql` :

- sous-répertoire `bin` : répertoire des binaires de l'application
- sous-répertoire `data` : répertoire des fichiers de configuration au premier niveau ; ce répertoire devra être la propriété de l'utilisateur « `pgsql` ».
- sous-répertoire `data/base` : répertoire des fichiers de stockage de la base de données
- sous-répertoire `doc` : répertoire de documentation (format HTML)
- sous-répertoire `include` : répertoire des headers de programmation C
- sous-répertoire `lib` : répertoire des bibliothèques partagées des binaires PGSQL
- sous-répertoire `man` : répertoire des pages de manuel
- sous-répertoire `share` : répertoire de fichiers d'exemples

A noter le fichier « `data/pg_hba.conf` » qui définit les accès autorisés.

**L'installation de PGSQL doit se compléter de la création des bases initiales de PGSQL.**

**§ 28.3 Création des bases initiales PGSQL : *initdb***

(en anglais *initialize database*)

Syntaxe : `initdb -D répertoire-de-données`

Par défaut, on prend le répertoire « data » dans l'arborescence PGSQL (par défaut donc « /usr/local/pgsql/data »).

La création des bases initiales se fera sous l'identité de l'utilisateur « `pgsql` ».

**§ 28.4 Lancement/arrêt de PGSQL : *pg\_ctl***

Deux processus sont en jeu :

- `postmaster` : accepte les requêtes de connexion
- `postgres` : exécute les requêtes SQL et renvoie les résultats

Un script de lancement et d'arrêt est fourni avec PGSQL : `pg_ctl`

Attention : les processus doivent être lancés au nom de l'utilisateur « `pgsql` ».

Syntaxes :

- démarrage de PGSQL : « `pg_ctl start` » :  
# `su psql -c "pg_ctl -D /chemin/vers/pgsql/data start"`  
postmaster successfully started
- arrêt de PGSQL : « `pg_ctl stop` » :  
# `su psql -c "pg_ctl -D /chemin/vers/pgsql/data stop"`  
waiting for postmaster to shut down.....done  
postmaster successfully shut down
- redémarrage de PGSQL : « `pg_ctl restart` »

**§ 28.5 Création de bases PGSQL : *createdb***

La création de bases normales PGSQL se fait par la commande « *createdb* ».

Syntaxe : *createdb* [options] base

La création de toute nouvelle base se fait par duplication de la base initiale de nom « *template1* ».

◇ Exemple

```
% createdb -e test
CREATE DATABASE base;      <--- commande interne lancée
CREATE DATABASE            <--- affichage du résultat de cette commande
```

L'option « -e » sert à afficher les commandes internes SQL en fait réalisées par le programme.

**§ 28.6 Destruction de bases PGSQL : *dropdb***

La destruction de bases normales PGSQL se fait par la commande « *dropdb* ».

Syntaxe : *dropdb* [options] base

◇ Exemple

```
% dropdb -e test
DROP DATABASE base;      <--- commande interne lancée
DROP DATABASE            <--- affichage du résultat de cette commande
```

L'option « -e » sert à afficher les commandes internes SQL en fait réalisées par le programme.

**§ 28.7 Dialogue interactif avec PGSQL en mode CLI : `psql`**

La commande « `psql` » permet d'interagir avec PGSQL et de lui demander d'exécuter des commandes SQL. Cela se présente comme un shell spécialisé en SQL.

Syntaxe : `psql base`

De nombreuses commandes sont ensuite disponibles.

◇ Exemple 1 : connexion et déconnexion

```
% psql template1
Welcome to psql 7.4.2, the PostgreSQL interactive terminal.

Type: \copyright for distribution terms
      \h for help with SQL commands
      \? for help on internal slash commands
      \g or terminate with semicolon to execute query
      \q to quit

template1=# \q
```

◇ Exemple 2 : liste des bases

```
% psql template1
...
template1=# \l
          List of databases
  Name      | Owner  | Encoding
-----+-----+-----
 template0 | pgsql  | SQL_ASCII
 template1 | pgsql  | SQL_ASCII
(2 rows)

template1=# select * from pg_database ;
 datname | datdba | encoding | datistemplate | dataallowconn | datlastsysoid | datvacuumxid | datfrozen
-----+-----+-----+-----+-----+-----+-----+-----
 template1 |      1 |          0 | t              | t              |              17140 |              464 |
 template0 |      1 |          0 | t              | f              |              17140 |              464 |
(2 rows)
```

## ◇ Exemple 3 : liste des tables systèmes

```
% psql template1
...
template1=# \ds
                List of relations
 Schema |          Name          | Type  | Owner
-----+-----+-----+-----
...
 pg_catalog | pg_database          | table | postgres
...
 pg_catalog | pg_user              | view  | postgres
...
(54 rows)
```

## ◇ Exemple 4 : liste des utilisateurs

```
% psql template1
...
template1=# select * from pg_user ;
 username | usesysid | usecreatedb | usesuper | usecatupd | passwd | valuntil | useconfig
-----+-----+-----+-----+-----+-----+-----+-----
 postgres |          1 | t           | t        | t         | ***** |          |
 apache   |        1002 | f           | f        | f         | ***** |          |
 besancon |        1003 | t           | t        | t         | ***** |          |
(3 rows)
```

**§ 28.8 Création de bases PGSQL : commandes SQL**

```
% psql template1
Welcome to psql 7.4.2, the PostgreSQL interactive terminal.
```

```
Type: \copyright for distribution terms
       \h for help with SQL commands
       \? for help on internal slash commands
       \g or terminate with semicolon to execute query
       \q to quit
```

```
template1=# create database exemple ;
CREATE DATABASE
```

```
template1=# \l
                List of databases
 Name      | Owner  | Encoding
-----+-----+-----
 exemple   | besancon | SQL_ASCII
 template0 | postgres | SQL_ASCII
 template1 | postgres | SQL_ASCII
(3 rows)
```

**§ 28.9 Destruction de bases PGSQL : commandes SQL**

```
% psql template1
Welcome to psql 7.4.2, the PostgreSQL interactive terminal.

Type: \copyright for distribution terms
      \h for help with SQL commands
      \? for help on internal slash commands
      \g or terminate with semicolon to execute query
      \q to quit

template1=# drop database exemple ;
DROP DATABASE

template1=# \l
          List of databases
-----+-----+-----
 Name      | Owner  | Encoding
-----+-----+-----
 template0 | pgsql  | SQL_ASCII
 template1 | pgsql  | SQL_ASCII
(2 rows)
```

**§ 28.10 Notion d'utilisateurs PGSQL**

Une requête SQL est émise au nom d'un utilisateur fictif interne à la base SQL.

Un utilisateur SQL n'est pas nécessairement un vrai utilisateur Unix.

La notion d'utilisateurs SQL est utilisée dans le contrôle des requêtes SQL :

```
GRANT privilège[,...] ON objet[,...] TO {PUBLIC | utilisateur} ;
```

où :

- les privilèges sont parmi les mots clef « SELECT », « INSERT », « UPDATE », « DELETE », « RULE », « ALL »
- un objet désigne une table, une vue ou une séquence
- PUBLIC indiquera que tous les utilisateurs SQL peuvent obtenir les privilèges indiqués
- utilisateur désignera un utilisateur bien précis qui obtient les privilèges indiqués

## ◇ Exemple

```
% psql base
...
base=# \d
          List of relations
 Schema | Name   | Type  | Owner
-----+-----+-----+-----
 public | exemple | table | besancon
(1 row)

t=# \z
      Access privileges for database "base"
 Schema | Table | Access privileges
-----+-----+-----
 public | exemple |
(1 row)

t=# grant all on exemple to public ;
GRANT
t=# \z
          Access privileges for database "base"
 Schema | Table | Access privileges
-----+-----+-----
 public | exemple | {besancon=a*r*w*d*R*x*t*/besancon,=arwdRxt/besancon}
(1 row)
...
```

## § 28.11 Gestion d'utilisateurs PGSQL (1) : createuser

Pour créer un utilisateur SQL, employer la commande « createuser » et répondre aux diverses questions posées.

Syntaxe: `createuser [options] [utilisateur-SQL]`

-a	-A	Création de nouveaux utilisateurs ou pas
-d	-D	Création de nouvelles databses ou pas

◇ Exemple 1 : création toute interactive

```
# createuser
Enter name of user to add: ars
Shall the new user be allowed to create databases? (y/n) n
Shall the new user be allowed to create more new users? (y/n) n
CREATE USER
```

◇ Exemple 2 : création non interactive

```
# createuser -D -A ars
CREATE USER
```

◇ Exemple 3 : création non interactive et avec affichage des instructions SQL internes

```
# createuser -D -A -e ars
createuser -D -A -e ars
SET autocommit TO 'on';CREATE USER "ars" NOCREATEDB NOCREATEUSER
CREATE USER
```

**§ 28.12 Gestion d'utilisateurs PGSQL (2) : dropuser**

Pour supprimer un utilisateur SQL, employer la commande « dropuser » et répondre aux diverses questions posées.

Syntaxe : dropuser [options] [utilisateur-SQL]

◇ Exemple 1 : suppression toute interactive

```
# dropuser
Enter name of user to drop: ars
DROP USER
```

◇ Exemple 2 : suppression non interactive

```
# dropuser ars
DROP USER
```

◇ Exemple 3 : suppression interactive avec garde fou et avec affichage des instructions SQL internes

```
# dropuser -e -i ars
User "ars" will be permanently removed.
Are you sure? (y/n) y
DROP USER ars;
DROP USER
```

**§ 28.13 Fichier de configuration : postgresql.conf**

Le fichier « data/postgresql.conf » est créé après la phase du « initdb ».

Il contiendra les paramètres de gestion mémoire pour PGSQL.

**Le paramètre tcpip\_socket indiquera si PGSQL accepte les connexions réseau.**

## § 28.14 Fichier de configuration : `pg_hba.conf`

Le fichier « `data/pg_hba.conf` » est créé après la phase du « `initdb` ».

Ce fichier sert à régler les contrôles d'accès.

Syntaxe :

```
local      DATABASE  USER  METHOD  [OPTION]
host       DATABASE  USER  IP-ADDRESS  IP-MASK  METHOD  [OPTION]
hostssl    DATABASE  USER  IP-ADDRESS  IP-MASK  METHOD  [OPTION]
hostnossl  DATABASE  USER  IP-ADDRESS  IP-MASK  METHOD  [OPTION]
host       DATABASE  USER  IP-ADDRESS/CIDR-MASK  METHOD  [OPTION]
hostssl    DATABASE  USER  IP-ADDRESS/CIDR-MASK  METHOD  [OPTION]
hostnossl  DATABASE  USER  IP-ADDRESS/CIDR-MASK  METHOD  [OPTION]
```

avec

The first field is the connection type : "local" is a Unix-domain socket, "host" is either a plain or SSL-encrypted TCP/IP socket, "hostssl" is an SSL-encrypted TCP/IP socket, and "hostnossl" is a plain TCP/IP socket.

DATABASE can be "all", "sameuser", "samegroup", a database name (or a comma-separated list thereof), or a file name prefixed with "@".

USER can be "all", an actual user name or a group name prefixed with "+" or a list containing either.

IP-ADDRESS and IP-MASK specify the set of hosts the record matches. CIDR-MASK is an integer between 0 and 32 (IPv6) or 128(IPv6) inclusive, that specifies the number of significant bits in the mask, so an IPv4 CIDR-MASK of 8 is equivalent to an IP-MASK of 255.0.0.0, and an IPv6 CIDR-MASK of 64 is equivalent to an IP-MASK of ffff:ffff:ffff:ffff:..

METHOD can be "trust", "reject", "md5", "crypt", "password", "krb4", "krb5", "ident", or "pam". Note that "password" uses clear-text passwords ; "md5" is preferred for encrypted passwords.

OPTION is the ident map or the name of the PAM service.

**§ 28.15 Sauvegarde : `pg_dump`, `pg_dumpall`**

PGSQL fournit un outil pour sauvegarder au format SQL sous forme de commandes SQL une ou plusieurs bases.

Pour sauver une base, utiliser la commande « `pg_dump` » sous l'identité de l'utilisateur « `pgsql` ».

Syntaxe : `pg_dump [options] base`

Pour sauver toutes les bases, utiliser la commande « `pg_dumpall` » sous l'identité de l'utilisateur « `pgsql` ».

Syntaxe : `pg_dumpall [options]`

**Attention : joindre à la sauvegarde le fichier `pg_hba.conf` qui contrôle les accès et qui n'est pas sauvegardé par les 2 outil SQL `pg_dump` et `pg_dumpall`.**

**◇ Exemple 1 : sauvegarde d'une seule base**

```
% pg_dump base > /tmp/base.sql

% ls -l /tmp/base.sql
-rw-r--r--  1 besancon adm      15054229 Oct 23 12:12 /tmp/base.sql

% head /tmp/base.sql
--
-- PostgreSQL database dump
--

SET client_encoding = 'SQL_ASCII';
SET check_function_bodies = false;
...
```

## ◇ Exemple 2 : sauvegarde de toutes les bases

```

% pg_dumpall > /tmp/bases.sql

% ls -l /tmp/bases.sql
-rw-r--r--  1 besancon adm      15064263 Oct 23 12:18 /tmp/bases.sql
% head /tmp/base.sql
--
-- PostgreSQL database cluster dump
--

\connect "template1"

--
-- Users
--

DELETE FROM pg_shadow WHERE usesysid <> (SELECT datdba FROM pg_database WHERE datname = 'template0')

CREATE USER apache WITH SYSID 1002 NOCREATEDB NOCREATEUSER;
CREATE USER besancon WITH SYSID 1003 CREATEDB CREATEUSER;
...

```

**§ 28.16 Rechargement d'une sauvegarde**

S'il s'agit de recharger la sauvegarde d'une base, faire sous l'identité de l'utilisateur « *pgsql* » :

```

% dropdb base
% psql < /tmp/base.sql

```

S'il s'agit de recharger la sauvegarde de la totalité des bases, faire sous l'identité de l'utilisateur « *pgsql* » :

```

% initdb -D /chemin/vers/pgsql/data/base
% psql < /tmp/bases.sql

```

## § 28.17 Interface de gestion de PGSQL : *phpPgAdmin*

A noter un interface WWW de configuration et d'administration : *phpPgAdmin*

Cf <http://sourceforge.net/projects/phpPgAdmin/>

**Figure manquante...**

## Chapitre 29 : Langage SQL

SQL  $\equiv$  Structured Query Language

SQL permet :

- la définition et la modification du schéma de la base de données
- la manipulation des données
- l'interrogation des données
- le contrôle des accès à la base

### § 29.1 Définition des exemples

Voici les relations qui seront utilisées dans ce cours :

Table AVION			
Numav	Capacite	Type	Entrepot
14	25	A400	Garches
345	75	B200	Maubeuge

Table PILOTE				
Matricule	Nom	Ville	Age	Salaire
1	Figue	Cannes	45	28004
2	Lavande	Touquet	24	11758

Table PASSAGER	
Numtab	Nomab
1	Nifance
2	Tearice

Table VOL				
Numvol	Heure_depart	Heure_arrivee	Ville_depart	Ville_arrivee
AL12	08-18	09-12	Paris	Lilles
AF8	11-20	23-54	Vaux	Rio

Table DEPART			
Numvol	Date_dep	Numav	Matricule
AL12	31-DEC-95	14	1
AL12	19-DEC-95	342	2

Table RESERVATION		
Numab	Numvol	Date_sep
1	AL12	31-DEC-95
2	AL12	31-DEC-95

## § 29.2 Description des données

### ◇ Création d'une relation

```
CREATE TABLE Pilote (
    pl#      Number(4),
    plnom    char(20),
    adresse  char(30)
) ;
```

### ◇ Création d'une relation avec contraintes d'intégrité

```
CREATE TABLE Pilote (
    pl#      Number(4) not null,
    plnom    char(20),
    adresse  char(30)
) ;
```

◇ Création d'index

```
CREATE UNIQUE INDEX IPL#  
ON Pilote(pl#);
```

◇ Suppression d'une relation

```
DROP TABLE Pilote ;
```

◇ Modification du schéma d'une relation

```
ALTER TABLE Pilote ADD ( age number(2) );  
ALTER TABLE Pilote MODIFY ( adresse char(50) );  
ALTER TABLE Pilote MODIFY ( pl# not null );
```

◇ Création d'une vue

```
CREATE VIEW VoldeParis AS (  
    SELECT vol#, va, hd, ha  
    FROM Vol  
    WHERE vd='Paris'  
) ;
```

**§ 29.3 Manipulation des données**◇ Insertion des données

```
INSERT INTO Pilote values (1, 'Antoine', 'Annecy') ;  
INSERT INTO Pilote values (2, 'Arthur', 'Adge') ;  
INSERT INTO Pilote values (3, 'Jean', null) ;
```

◇ Suppression des données

```
DELETE FROM Pilote WHERE pl# = 1 ;
```

◇ Modification des données

```
UPDATE Pilote  
SET adresse = 'Aurillac'  
WHERE pl#=2 ;
```

**§ 29.4 Contrôle des accès**◇ Fin normale (validation) d'une transaction

```
COMMIT ;
```

L'instruction commit rend effectifs les changements effectués sur la base. Ils deviennent visibles aux autres utilisateurs.

◇ Annulation d'une transaction

```
ROLLBACK ;
```

L'instruction rollback défait toutes les modifications effectuées sur la base depuis le début de la transaction.

**§ 29.5 Projection, Restriction**◇ Principe

```
SELECT AVION.numav  
FROM AVION  
WHERE AVION.capacite > 200 ;
```

Projection seule :

```
SELECT AVION.numav  
FROM AVION ;
```

Restriction seule :

```
SELECT AVION.*  
FROM AVION  
WHERE AVION.capacite > 200 ;
```

Exercice 1 : Quel est l'âge du pilote de matricule 15 ?

Exercice 2 : Quels sont les pilotes de cette compagnie aérienne ? (leur matricule, leur nom, leur ville, leur âge et leur salaire)

◇ Tri

Tri croissant : ORDER BY... ASC

Tri décroissant : ORDER BY... DESC

```
SELECT AVION.numav
FROM AVION
WHERE AVION.capacite > 200
ORDER BY AVION.numav DESC ;
```

◇ Suppression des doubles

```
SELECT DISTINCT AVION.numav
FROM AVION
WHERE AVION.capacite > 200 ;
```

◇ Omission du préfixe

```
SELECT numav
FROM AVION
WHERE capacite > 200 ;
```

◇ Emploi de variables

```
SELECT A.numav
FROM AVION A
WHERE A.capacite > 200 ;
```

Exercice 3 : Quels sont les types d'avions entreposés à Tarascon, avec leur capacité, par ordre de capacités croissantes ?

◇ Prédicats dans le critère de sélection

Les prédicats disponibles :

=, !=, >, <, >=, <=

OR, AND, LIKE, NOT NULL, IN, NOT, BETWEEN X AND Y

```
SELECT numav
FROM AVION
WHERE capacite > 200
AND ( (ENTREPOT = 'Saint nom la Breteche')
      OR (ENTREPOT = 'Mantes la jolie')
    )
AND numav != 300 ;
```

```
SELECT numav
FROM AVION
WHERE entrepot in ('Saint nom la Breteche', 'Mantes la jolie')
AND capacite BETWEEN 200 AND 210
AND TYPE LIKE 'B_EING%';
```

Exercice 4 : Quelles sont les villes ayant un nom commençant par 'L' et où un pilote de moins de 20 ans gagne plus de 20000 F ?

Exercice 5 : Quels sont les numéros d'avions, avec leur entrepôt, ne résidant pas à Marolles-en-Hurepoix, ayant une capacité supérieure à 200 ? (donnés par ordre de numéros décroissants)

#### ◇ Les fonctions

Quelques fonctions disponibles :

- fonction lower (a)
- fonction soundex (a)
- fonction abs (a)
- fonction sqrt (a)
- fonction length (a)
- etc.

```
SELECT numav
FROM AVION
WHERE LOWER(ENTREPOT) = 'le vesiney';
```

```
SELECT MATRICULE, NOM
FROM PILOTE
WHERE SOUNDEX(NOM) = SOUNDEX('SINTEXUPERI') ;
```

```
SELECT UPPER(NOM)
FROM PILOTE
WHERE ADRESSE = 'Cordillère des Andes' ;
```

Exercice 6 : Quelles sont les villes d'où part un vol pour Ajaccio ? (donner les résultats en majuscule)

#### ◇ Les agrégats

Quelques agrégats disponibles :

- agrégat `avg()`
- agrégat `count()`
- agrégat `max()`
- agrégat `min()`
- agrégat `sum()`
- agrégat `stddev()`
- agrégat `variance()`
- etc.

Exemple : Quel est le nombre d'avions ayant une capacité supérieure à 200 ?

```
SELECT COUNT(NUMAV)
FROM AVION
WHERE CAPACITE > 200 ;
```

```
SELECT COUNT(DISTINCT NUMAV)
FROM AVION
WHERE CAPACITE > 200 ;
```

Exercice 7 : Quel est le nombre d'avions en service (ie dans la relation DEPART), le 19 décembre 1995 ?

Exercice 8 : Quelle est la capacité moyenne des avions entreposés à Pogo Togo ?

## § 29.6 Les requêtes imbriquées

### ◇ Prédicat d'existence

```
SELECT
FROM
WHERE EXISTS ( SELECT
                FROM
                WHERE
            ) ;
```

Exemple : Quels sont les pilotes qui habitent dans une ville où n'est basé aucun avion ?

```
SELECT PILOTE.NOM
FROM PILOTE
WHERE NOT EXISTS ( SELECT AVION.*
                   FROM AVION
                   WHERE AVION.ENTREPOT = PILOTE.VILLE
                 ) ;
```

◇ Requête à l'intérieur du IN

Exemple : Quels sont les noms des pilotes qui habitent dans une ville où sont localisés des avions de capacité supérieure à 250 ?

```
SELECT PILOTE.NOM
FROM PILOTE
WHERE VILLE IN ( SELECT ENTREPOT
                 FROM AVION
                 WHERE CAPACITE > 250
               ) ;
```

Exercice 9 : Exprimer la requête suivante avec un EXISTS : Quels sont les noms des pilotes qui habitent dans une ville où sont localisés des avions de capacité supérieure à 250 ?

Exercice 10 : Exprimer la requête suivante avec un IN : Quels sont les noms des passagers ayant réservé pour le 1er avril 1996 ?

**§ 29.7 La jointure**

2 méthodes principales :

- prédicative
- requêtes imbriquées

Exemple : Quels sont les noms des pilotes qui assurent un vol le vendredi 13 octobre 1996 ?

Prédicative :

```
SELECT PILOTE.NOM
FROM PILOTE, DEPART
WHERE PILOTE.MATRICULE = DEPART.MATRICULE
AND DEPART.DAPE_SEP = '13-10-95' ;
```

Imbrication :

```
SELECT NOM
FROM PILOTE
WHERE MATRICULE IN ( SELECT DEPART.MATRICULE
                     FROM DEPART
                     WHERE DATE_SEP = '13-10-95'
                   ) ;
```

Il y a de nombreuses autres façons d'exprimer une jointure.

Par exemple :

```
SELECT PILOTE.NOM
FROM PILOTE
WHERE EXISTS ( SELECT *
                FROM DEPART
                WHERE
                PILOTE.MATRICULE = DEPART.MATRICULE
                AND DEPART.DATE_DEP = '13-10-95'
              ) ;
```

Exercice 11 : Quels sont les noms des pilotes qui assurent un vol le vendredi 13 octobre 1995, sur un avion de capacité supérieure à 250 ?

- de manière prédicative
- à l'aide de requêtes imbriquées

Exercice 12 : Quelles sont les paires de pilotes (matricules) habitant dans la même ville ?

Exercice 13 : Quels sont les noms des pilotes qui conduisent l'avion de numéro 666 ?

Exercice 14 : Quels sont les noms des pilotes qui habitent dans la ville où est entreposé l'avion 666 ?

## § 29.8 L'union

Quels sont les pilotes qui conduisent l'avion de numéro 666 OU qui habitent dans la ville où est entreposé l'avion 666 ?

```
SELECT NOM
FROM PILOTE, DEPART
WHERE PILOTE.MATRICULE = DEPART.MATRICULE
AND DEPART.NUMAV = 666
UNION
SELECT NOM
FROM PILOTE, AVION
WHERE PILOTE.VILLE = AVION.ENTREPOT
AND AVION.NUMAV = 666 ;
```

**§ 29.9 L'intersection**

Quels sont les pilotes qui conduisent l'avion de numéro 666 ET qui habitent dans la ville où est entreposé l'avion 666 ?

```
SELECT NOM
FROM PILOTE, DEPART
WHERE PILOTE.MATRICULE = DEPART.MATRICULE
AND DEPART.NUMAV = 666
INTERSECT
SELECT NOM
FROM PILOTE, AVION
WHERE PILOTE.VILLE = AVION.ENTREPOT
AND AVION.NUMAV = 666 ;
```

**§ 29.10 La différence**

Quels sont les pilotes qui conduisent l'avion de numéro 666 SAUF ceux qui habitent dans la ville où est entreposé l'avion 666 ?

```
SELECT NOM
FROM PILOTE, DEPART
WHERE PILOTE.MATRICULE = DEPART.MATRICULE
AND DEPART.NUMAV = 666
MINUS
SELECT NOM
FROM PILOTE, AVION
WHERE PILOTE.VILLE = AVION.ENTREPOT
AND AVION.NUMAV = 666 ;
```

**§ 29.11 La division**

Quels sont les pilotes qui conduisent tous les avions ?

≡ Quels sont les pilotes tels que, quel que soit l'avion, ils le conduisent ?

≡ Quels sont les pilotes tels que, quel que soit l'avion, il existe un départ assuré par ce pilote sur cet avion ?

≡ Quels sont les pilotes tels qu'il n'existe pas d'avion tel qu'il n'existe pas départ de ce pilote sur cet avion ?

```
SELECT MATRICULE
FROM PILOTE
WHERE NOT EXISTS ( SELECT *
                   FROM AVION
                   WHERE NOT EXISTS ( SELECT *
                                     FROM DEPART
                                     WHERE DEPART.MATRICULE = PILOTE.MATRICULE
                                     AND DEPART.NUMAV = AVION.NUMAV
                                   )
                 ) ;
```

Exercice 15 : Quels sont les passagers qui ont réservé sur tous les vols ?

Exercice 16 : Quel est le nombre de pilotes qui conduisent des avions en service ? (ie dans la relation DEPART)

**§ 29.12 Group by**

L'opérateur GROUP BY partitionne la table.

Exemple : Quel est le nombre de pilotes qui conduisent des avions en service, par avion ?

```
SELECT NUMAV, COUNT(DISTINCT DEPART.MATRICULE)
FROM DEPART
GROUP BY NUMAV ;
```

Exercice 17 : Quelle est la capacité moyenne des avions, par entrepôt et part type ?

**§ 29.13 Group by ... HAVING**

L'opérateur HAVING permet d'introduire une qualification sur les sous-tables issues du GROUP BY.

Le HAVING est au GROUP BY ce que le WHERE est au FROM.

Exemple : quels sont les avions qui assurent plus de 3 départ ?

```
SELECT DEPART.NUMAV
FROM DEPART
GROUP BY NUMAV
HAVING COUNT(*) > 3 ;
```

Exercice 18 : Quelles sont les villes où sont entreposés plus de 2 avions de type ICBM ?

**§ 29.14 Conclusion**

SQL est un langage d'interrogation très lisible, par tous les utilisateurs.

SQL est un langage dans lequel il est relativement facile d'écrire, pour un programmeur, à condition de bien visualiser l'ordre d'application des instructions :

1. From
2. Where
3. Group By
4. Having
5. Select
6. Order By

**§ 29.15 Correction des exercices**◇ Exercice 1

Quel est l'âge du pilote de matricule 15 ?

```
SELECT PILOTE.AGE  
FROM PILOTE  
WHERE MATRICULE = 15 ;
```

◇ Exercice 2

Quels sont les pilotes de cette compagnie aérienne (leur matricule, leur nom leur ville, leur âge et leur salaire)

```
SELECT *  
FROM PILOTE ;
```

◇ Exercice 3

Quels sont les types d'avions entreposés à Tarascon, avec leur capacité, par ordre de capacité croissantes ?

```
SELECT DISTINCT TYPE, CAPACITE
FROM AVION
WHERE ENTREPOT = 'Tarascon'
ORDER BY CAPACITE ASC ;
```

◇ Exercice 4

Quelles sont les villes ayant un nom commençant par L et où un pilote de moins de 20 ans gagne plus de 200000 F ?

```
SELECT VILLE
FROM PILOTE
WHERE AGE < 20
AND SALAIRE > 200000
AND VILLE LIKE 'L%';
```

◇ Exercice 5

Quels sont les numéros d'avions, avec leur entrepôt, ne résidant pas à Marolles-en-Hurepoix, ayant une capacité supérieure à 200 ? (donnés par ordre de numéros décroissants)

```
SELECT NUMAV, ENTREPOT
FROM AVION
WHERE ENTREPOT != 'Marolles-en-Hurepoix'
AND CAPACITE > 200
ORDER BY NUMAV DESC ;
```

◇ Exercice 6

Quelles sont les villes d'où part un vol pour Ajaccio ? (donner les résultats en majuscules)

```
SELECT UPPER(VILLE_DEPART)
FROM VOL
WHERE VILLE_ARRIVEE = 'Ajaccio';
```

◇ Exercice 7

Quel est le nombre d'avions en service (ie dans la relation DEPART), le 19 décembre 1995 ?

```
SELECT COUNT(DISTINCT NUMAV)
FROM DEPART
WHERE DATE_DEP = '19-12-95' ;
```

◇ Exercice 8

Quelle est la capacité moyenne des avions entreposés à Pogo Togo ?

```
SELECT AVG(ALL CAPACITE)
FROM AVION
WHERE AVION.ENTREPOT = 'Pogo Togo' ;
```

◇ Exercice 9

Exprimer la requête suivante avec un EXISTS :

Quels sont les noms des pilotes qui habitent dans une ville où sont localisés des avions de capacité supérieure à 250 ?

```
SELECT PILOTE.NOM
FROM PILOTE
WHERE EXISTS ( SELECT *
                FROM AVION
                WHERE CAPACITE > 250
                AND AVION.ENTREPOT = PILOTE.VILLE
              ) ;
```

◇ Exercice 10

Quels sont les noms des passagers ayant réservé pour le 1er avril 1996 ? (avec un IN)

```
SELECT PASSAGER.NOMAB
FROM PASSAGER
WHERE NUMAB IN ( SELECT RESERVATION.NUMAB
                  FROM RESERVATION
                  WHERE RESERVATION.DATE_DEP = '01-04-96'
                ) ;
```

◇ Exercice 11 a

Quels sont les noms des pilotes qui assurent un vol le vendredi 13 octobre 1995, sur un avion de capacité supérieure à 250 ?

De manière prédicative :

```
SELECT PILOTE.NOM
FROM PILOTE, DEPART, AVION
WHERE PILOTE.MATRICULE = DEPART.MATRICULE
AND DEPART.DATE_DEP = '13-10-95'
AND DEPART.NUMAV = AVION.NUMAV
AND AVION.CAPACITE > 250 ;
```

◇ Exercice 11 b

Quels sont les noms des pilotes qui assurent un vol le vendredi 13 octobre 1995, sur un avion de capacité supérieure à 250 ?

A l'aide de requêtes imbriquées :

```
SELECT NOM
FROM PILOTE
WHERE MATRICULE IN ( SELECT DEPART.MATRICULE
                     FROM DEPART
                     WHERE DATE_DEP = '13-10-95'
                     AND NUMAV IN ( SELECT AVION.NUMAV
                                   FROM AVION
                                   WHERE CAPACITE > 250
                                   )
                     ) ;
```

◇ Exercice 12

Quelles sont les paires de pilotes (matricules) habitant dans la même ville ?

```
SELECT P1.MATRICULE, P2.MATRICULE
FROM PILOTE P1, PILOTE P2
WHERE P1.VILLE = P2.VILLE
AND P1.MATRICULE > P2.MATRICULE ;
```

(il s'agit d'une auto jointure)

◇ Exercice 13

Quels sont les noms des pilotes qui conduisent l'avion de numéro 666 ?

```
SELECT NOM
FROM PILOTE, DEPART
WHERE PILOTE.MATRICULE = DEPART.MATRICULE
AND DEPART.NUMAV = '666' ;
```

◇ Exercice 14

Quels sont les noms des pilotes qui habitent dans la ville où est entreposé l'avion 666 ?

```
SELECT NOM
FROM PILOTE, AVION
WHERE PILOTE.VILLE = AVION.ENTREPOT
AND AVION.NUMAV = '666' ;
```

◇ Exercice 15

Quels sont les passagers qui ont réservé sur tous les vols ?

```
SELECT NUMAB
FROM PASSAGER
WHERE NOT EXISTS ( SELECT *
                   FROM VOL
                   WHERE NOT EXISTS ( SELECT *
                                     FROM RESERVATION
                                     WHERE RESERVATION.NUMAB = PASSAGER.NUMAB
                                     AND RESERVATION.NUMVOL = VOL.NUMVOL
                                     )
                   ) ;
```

◇ Exercice 16

Quel est le nombre de pilotes qui conduisent des avions en service ? (ie dans la relation DEPART)

```
SELECT COUNT(DISTINCT DEPART.MATRICULE)
FROM DEPART ;
```

◇ Exercice 17

Quelle est la capacité moyenne des avions, par entrepôt et par type ?

```
SELECT ENTREPOT, TYPE, AVG(CAPACITE)
FROM AVION
GROUP BY ENTREPOT, TYPE ;
```

◇ Exercice 18

Quelles sont les villes où sont entreposés plus de 2 avions de type ICBM ?

```
SELECT ENTREPOT  
FROM AVION  
WHERE TYPE = 'ICBM'  
GROUP BY ENTREPOT  
HAVING COUNT(*) > 2 ;
```

## Chapitre 30 : Consolidation d'OS

### § 30.1 VMWARE



<http://www.vmware.com/>

VMWARE ne tourne que sur plateforme à base de processeur Intel ou compatible.

Ce logiciel offre sur la machine hôte, des machines virtuelles dans lesquelles peuvent tourner des OS comme Windows, Linux, FreeBSD.

Il s'agit donc de **machine virtuelle** et non pas de prise en main comme avec VNC.

Architecture :

