Table des matières

Règles de base pour le LVM

Règles de calcul

Sous AIX, on ne peut pas spécifier la taille en Mo à la création d'un LV. Il faut spécifier le nombre de « partitions logiques » attribuées au LV lors de sa création.

- Calcul du nombre de partitions logiques pour un LV :

Avant tout, il faut connaître la taille des *partitions physiques* du VG (PP SIZE) dans lequel le LV doit être créé. Pour cela, il faut utiliser la commande `lsvg <VG_NAME>` qui donne les infos suivantes :

```
lsvg cluster-apps

VOLUME GROUP:   cluster-apps        VG IDENTIFIER:    004007faa3d706a3
VG STATE:   active            PP SIZE:        8 megabyte(s)
VG PERMISSION:  read/write            TOTAL PPs:        11277 (90216 megabytes)
MAX LVs:    256           FREE PPs:        4559 (36472 megabytes)
LVs:        26            USED PPs:        6718 (53744 megabytes)
OPEN LVs:   24            QUORUM:            1
TOTAL PVs:  21            VG DESCRIPTORS:        21
STALE PVs:  0           STALE PPs:       0
ACTIVE PVs: 21            AUTO ON:        no
MAX PPs per PV: 1016              MAX PVs:        32
```

La taille des PP est ici de 8 Mo. A l'heure actuelle 8 Mo c'est assez peu. En effet avec ce pramètre on ne peut utiliser que *1018*6 = 8128 Mo* par disques. Or les disques actuels peuvent maintenant faire plusieurs centaines de Go. Du coup il vaut mieux ajuster ces paramètres (voir *-t factor* et *-s* du mkvg).

Le calcul est le suivant :

```
Nombre de LP = <taille_en_Mo> / <PP_SIZE>
```

Exemple : si on veut créer un LV de 500 Mo, le calcul est :

```
500 / 8 = 62,5 => 63 LP (on arrondit toujours au-dessus)
```

Attention : pour les Raw Devices Sybase, ajouter une LP.

- Calcul du nombre de blocs d'un FS :

AIX se base sur des blocs de 512 Ko. Le calcul du nombre de blocs pour un FS est :

```
<taille_en_Mo> * 1024 * 2
```

Exemple : Pour un FS de 500 Mo, le nombre de blocs est : 500 * 1024 * 2 = 1024000

Règles à respecter

- Trop souvent ignoré, il faut savoir que le lv de log (jfslog ou jfs2log) doit faire au moins 4LP pour 1000LP de FS dans le vg.
- Ne pas mirrorer sur le même disque ni sur le même site si possible.

Caractéristiques des VGs (AIX 5.x)

Type de VG	max PVs	max PPs per PV	max PPs per VG	max LVs per VG	max PP size
Normal VG	32	1016	32 512 (1016×32)	256	1 Go
Big VG	128	1016	130 048 (1016×128)	512	1 Go
Scalable VG	1024	4096	2 097 152	4096	128 Go

Normal VG		
Nb de PV	***Nombre max de PPs/PV***	***t factor***
2	16256	(mkvg) -t 5
4	8128	(mkvg) -t 4
8	4064	(mkvg) -t 3
16	2032	(mkvg) -t 2
32	1016	(mkvg)

Big VG		
Nb de PV	***Nombre max de PPs/PV***	***t factor***
2	65024	(mkvg) -B -t 7
4	32512	(mkvg) -B -t 6
8	16256	(mkvg) -B -t 5
16	8128	(mkvg) -B -t 4
32	4064	(mkvg) -B -t 3
64	2032	(mkvg) -B -t 2
128	1016	(mkvg) -B

LVM : Commandes de base

Quelques liens intéressants :

http://www.ahinc.com/aix/logicalvol.htm

http://www.linux-france.org/~mdecore/aix/memo-aix/html/

Créer un rawdevice

```
mklv -c 2 -s s -y lv_CRF3_uritylo -t raw vg_crf3 1 hdiskpower39 hdiskpower50
```

Créer un LV

On ne peut pas utiliser un *upperbound* de 2 si l'un des disques est mirroir de l'autre. On procède donc de cette façon :

```
mklv -y lv_oraRAIP1 -t jfs -c2 -ss -u1 -bn vg_raid 60 hdiskpower0 hdiskpower10
chlv -u2 lv_oraRAIP1
extendlv -ss lv_oraRAIP1 60 hdiskpower1 hdiskpower11
```

Concernant ce chapitre, je dirai qu'en fait l'upperbound correspond à un nombre de disque par copie (si tu as 2 copies pv1,pv2,pv3, donc, alors u1 suffira) si tu etend sur d'autres disques (pv4,5,6), alors faudra passer en u2 etc... **Merci Jean-Gab pour cette précision.**
mklv -ex (cf. man) **Merci Patrice**

On peut spécifier les droits du LV à la création (ou à postériori avec *chlv*). Il suffit d'utiliser *-U*, *-G* et *-P*. Cela est utile en cas d'import/exportvg de VGs pour des raws par exemple. Ceux-ci sont repositionnées à *root* par défaut et cela peut gêner Sybase ou Oracle.

Ensuite on peut créé le FS :

```
crfs -A no -a bf=true -a nbpi=16384 -v jfs -d lv_oraRAIP1 -m /apps/oradata/RAIFRPI1
```

Etendre un LV

```
extendlv lv_DRESU_21 5 hdiskpower60
```

Créer un FS

```
crfs -v jfs2 -d lv_sar -m /var/adm/sa -A yes
crfs -v jfs2 -m /apps/oracledata/VBAFRP02/data001 -d /dev/lv_VBAFRP02_dat -A no -p rw -a options=rw,cio
```


En JFS penser à changer les paramètres si on doit créer un FS dont la taille > 64 Go :

NBPI	Minimum AG Size	Fragment Size	Maximum Size (GB)
512	8	512, 1024, 2048, 4096	8
1024	8	512, 1024, 2048, 4096	16
2048	8	512, 1024, 2048, 4096	32
4096	8	512, 1024, 2048, 4096	64

NBPI	Minimum AG Size	Fragment Size	Maximum Size (GB)
8192	8	512, 1024, 2048, 4096	128
16384	8	1024, 2048, 4096	256
32768	16	2048, 4096	512
65536	32	4096	1024
131072	64	4096	1024

Mettre à jour la map

```
varyonvg -u -b vg_crf3 (noeud modifié)
importvg -L vg_crf3 hdiskpower18 (un disque connu du noeud non modifié -> il lit la VGDA)
varyonvg vg_crf3 (noeud modifié)
```

Déterminer la bonne config des I/O asynchrones

Avec `smitty aio` on visualise la conf. On doit avoir *nb de disques X nb CPUs / 2*.

Créer un RAMDISK

- Créer un ramdisk sous AIX

→ utiliser 'rbrw' (évite de cacher sur disque le ramdisk . . .)

```
#!/bin/ksh -x

# Creation de 3 ramdisks pour DB_PAR_LIV_SQL
mkramdisk 4613734
mkramdisk 4613734
mkramdisk 2306867

# Initialisation en jfs des ramdisks
mkfs -V jfs -o nbpi=131072,ag=64,bf=true /dev/ramdisk0 << !
yes
!

mkfs -V jfs -o nbpi=131072,ag=64,bf=true /dev/ramdisk1 << !
yes
!

mkfs -V jfs -o nbpi=131072,ag=64,bf=true /dev/ramdisk2 << !
yes
!

mkdir -p /sybase_data/DB_PAR_LIV_SQL/ram_files/ramdisk0
mkdir -p /sybase_data/DB_PAR_LIV_SQL/ram_files/ramdisk1
mkdir -p /sybase_data/DB_PAR_LIV_SQL/ram_files/ramdisk2

mount -V jfs -o nointegrity /dev/ramdisk0 /sybase_data/DB_PAR_LIV_SQL/ram_files/ramdisk0
mount -V jfs -o nointegrity /dev/ramdisk1 /sybase_data/DB_PAR_LIV_SQL/ram_files/ramdisk1
mount -V jfs -o nointegrity /dev/ramdisk2 /sybase_data/DB_PAR_LIV_SQL/ram_files/ramdisk2

# prioprietaire des ramfs a sybase
chown -R sybase:sybase /dev/ramdisk0 /dev/ramdisk1 /dev/ramdisk2 /sybase_data/DB_PAR_LIV_SQL/ram_files

touch /sybase_data/DB_PAR_LIV_SQL/ram_files/ramdisk0/tempdb01.dat
touch /sybase_data/DB_PAR_LIV_SQL/ram_files/ramdisk1/tempdb02.dat
touch /sybase_data/DB_PAR_LIV_SQL/ram_files/ramdisk2/tempdb03.dat

# prioprietaire des ramfs a sybase
chown -R sybase:sybase /dev/ramdisk0 /dev/ramdisk1 /dev/ramdisk2 /sybase_data/DB_PAR_LIV_SQL/ram_files
```

Mirrorer un LV

```
root@machine:/apps/sys/log/stats/NMON$ mklvcopy -u 32 -s y lv_nmon 2
root@machine:/apps/sys/log/stats/NMON$ varyonvg rootvg
```

Le `varyonvg` permet de faire passer le lv de `open/stale` à `open/syncd`. Si on a un VG concurrent, il faut utiliser *varyonvg -c* (à confirmer).

Déplacer un FS dans un nouveau VG

Soit le LV suivant : *lvglognat2 jfs 91 182 2 open/syncd /apps/goal116/logs/NAT2*

- Copie des lv vers vg_applis :

```
cplv -y lvglognat2c -v vg_applis lvglognat2
```

- Raccorder le fs au nouvel lv :

```
chfs -a dev=/dev/lvglognat2c -a log=/dev/loglv00 /apps/goal116/logs/NAT2
```

- On fait un fsck :

```
fsck -fp /dev/lvglognat2c
```

- Un petit test de montage et de démontage
- Renommage des LVs :

```
chlv -n lvglognat2o lvglognat2
chlv -n lvglognat2 lvglognat2c
```

Si on doit déplacer les données dans un nouveau VG qui ne comprend pas de FS, il faut auparavant créer un FS temporaire pour obtenir un jfslog.

Ne pas oublier de lancer un *logform* sur le device créé.

lsvg.AIX

Un petit script pour afficher plus clairement les VGs, LVs et disques associés.

- lsvg.AIX.ksh

Migrer une LP par PV d'un VG

⇒ Pré-requis au passage en BIG VG

- Aide pour y parvenir

```
inq -nodots >/tmp/INQ #commande lente, on pousse son resultat une bonne fois pour toute dans un fichier
listePV=`lspv |grep vg_mtshist|awk '{print $1}'` #On recupere les PV concernés par la migration en big VG

echo HDISKPOWER     BAIE
for pv in $listePV
do
    grep $pv /tmp/INQ|awk '{gsub("/dev/r","",$1);baie=substr($5,2,2);print $1,baie}'
done|sort -k2
hdiskpower110 21
hdiskpower111 21
hdiskpower112 21
hdiskpower113 21
hdiskpower114 21
hdiskpower19 21
hdiskpower25 21
hdiskpower26 21
hdiskpower33 21
hdiskpower79 21
hdiskpower84 21
hdiskpower88 21
hdiskpower91 21
hdiskpower95 21
hdiskpower98 21
hdiskpower99 21
hdiskpower104 84
hdiskpower105 84
hdiskpower106 84
hdiskpower107 84
hdiskpower108 84
hdiskpower18 84
hdiskpower22 84
hdiskpower23 84
hdiskpower28 84
hdiskpower51 84
```

```
hdiskpower56 84
hdiskpower60 84
hdiskpower63 84
hdiskpower67 84
hdiskpower70 84
hdiskpower71 84
```

- Chercher les disques sans PP à migrer

```
lsvg -p vg_mtshist
vg_mtshist:
PV_NAME           PV STATE          TOTAL PPs   FREE PPs    FREE DISTRIBUTION
hdiskpower104     active            871         1           00..01..00..00..00
hdiskpower110     active            871         1           00..01..00..00..00
hdiskpower105     active            871         1           00..01..00..00..00
hdiskpower111     active            871         1           00..01..00..00..00
hdiskpower51      active            871         1           01..00..00..00..00
hdiskpower56      active            435         0           00..00..00..00..00
hdiskpower84      active            435         0           00..00..00..00..00
hdiskpower60      active            435         0           00..00..00..00..00
hdiskpower79      active            871         1           01..00..00..00..00
hdiskpower106     active            871         0           00..00..00..00..00
hdiskpower112     active            871         0           00..00..00..00..00
hdiskpower88      active            435         0           00..00..00..00..00
hdiskpower95      active            217         0           00..00..00..00..00
hdiskpower67      active            217         0           00..00..00..00..00
hdiskpower114     active            435         0           00..00..00..00..00
hdiskpower107     active            435         0           00..00..00..00..00
hdiskpower113     active            435         0           00..00..00..00..00
hdiskpower108     active            435         0           00..00..00..00..00
hdiskpower18      active            435         0           00..00..00..00..00
hdiskpower19      active            435         0           00..00..00..00..00
hdiskpower63      active            217         0           00..00..00..00..00
hdiskpower91      active            217         0           00..00..00..00..00
hdiskpower25      active            435         0           00..00..00..00..00
hdiskpower22      active            435         0           00..00..00..00..00
hdiskpower23      active            435         0           00..00..00..00..00
hdiskpower26      active            435         0           00..00..00..00..00
hdiskpower98      active            54          0           00..00..00..00..00
hdiskpower70      active            54          0           00..00..00..00..00
hdiskpower99      active            54          0           00..00..00..00..00
hdiskpower71      active            54          0           00..00..00..00..00
hdiskpower28      active            217         32          00..00..00..00..32
hdiskpower33      active            217         32          00..00..00..00..32
```

- Pour chaque PV sans PP de libre, trouver une PP d'un LV à bouger. On liste les LV d'un PV

```
lspv -l hdiskpower56
hdiskpower56:
LV NAME               LPs   PPs   DISTRIBUTION          MOUNT POINT
lv_mtshist            3     3     00..00..00..00..03    /apps/mtshist
lv_oraMT2P_exp        39    39    00..00..00..00..39    /apps/oracle/exp/MTSEUP02
lv_oraMT2P_back       125   125   00..00..00..83..42    /apps/oracle/backup/MTSEUP02
lv_oraMT2P_dat2       105   105   87..00..14..04..00    /apps/oracledata/MTSEUP02/data002
lv_oraMT2P_dat1       163   163   00..87..73..00..03    /apps/oracledata/MTSEUP02/data001
```

- On se mefie de l'upperbound, et on l'augmente si necessaire

```
lslv lv_mtshist|awk '/UPPER BOUND/ {print $NF}'
3
=> chlv -u 4 lv_mtshist
```

- On recherche une LP à migrer dans le LV concernant notre PV. Ici, on cherche une LP dans lv_mtshist positionnée sur hdiskpower56.

```
lslv -m lv_mtshist|egrep "hdiskpower56|LP"|head -3
LP    PP1  PV1               PP2  PV2               PP3  PV3
0045  0394 hdiskpower56      0394 hdiskpower84
0046  0395 hdiskpower56      0395 hdiskpower84

Je sais que le 56 doit aller sur le 28 et le 84 sur le 33
(je suis pas feignasse, tant qu'à faire, autant migrer les 2 disques en même temps)
```

- Migration d'une LP du LV vers le PV cible

```
migratelp lv_mtshist/45/1 hdiskpower28
migratelp lv_mtshist/45/2 hdiskpower33
```

- Check pour s'en assurer

```
lslv -m lv_mtshist|egrep "hdiskpower56|LP|0045"
LP    PP1  PV1               PP2  PV2               PP3  PV3
0045  0186 hdiskpower28      0186 hdiskpower33
0046  0395 hdiskpower56      0395 hdiskpower84
0047  0396 hdiskpower56      0396 hdiskpower84
```

Disques : Commandes de base

Ajouter un disque

- Etat des PV avant

```
root@machine:/home$ lspv|grep hdiskpower
hdiskpower0     0054784a70359330                    vgdata
hdiskpower1     0054784a4b0474b2                    vgdata
hdiskpower2     0054784a9a0b3b42                    vgdata
hdiskpower3     0040b8ba901706fa                    vgdata
hdiskpower4     0040b8ba901fef37                    None
hdiskpower5     0054784a4f672c94                    vgdata
hdiskpower6     0054784a4f61aa7d                    vgdata
hdiskpower7     0054784a4b2b9dd4                    vgdata
hdiskpower8     0054784a4b8e0d72                    vgdata
hdiskpower9     0054784a702dbf42                    vgdata
hdiskpower10    0040b8ba8fecb4b4                    vgdata
hdiskpower11    0054784a1323a625                    vgdata
hdiskpower12    005477caae32d00a                    vgdata
hdiskpower13    0054784a4f646e17                    vgdata
```

- Lancer la commande `cfgmgr` pour découvrir les nouveaux devices
- Etat des PV après

```
hdiskpower0     0054784a70359330                    vgdata
hdiskpower1     0054784a4b0474b2                    vgdata
hdiskpower2     0054784a9a0b3b42                    vgdata
hdiskpower3     0040b8ba901706fa                    vgdata
hdiskpower4     0040b8ba901fef37                    None
hdiskpower5     0054784a4f672c94                    vgdata
hdiskpower6     0054784a4f61aa7d                    vgdata
hdiskpower7     0054784a4b2b9dd4                    vgdata
hdiskpower8     0054784a4b8e0d72                    vgdata
hdiskpower9     0054784a702dbf42                    vgdata
hdiskpower10    0040b8ba8fecb4b4                    vgdata
hdiskpower11    0054784a1323a625                    vgdata
hdiskpower12    005477caae32d00a                    vgdata
hdiskpower13    0054784a4f646e17                    vgdata
hdiskpower14    none                                None
hdiskpower15    0054784a312689df                    None
```

Dans ce cas la le hdiskpower 14 n'a pas de PVID alors que le hdiskpower 15 en possède déjà un. Cela est du au fait qu'il existait déjà un hdiskpower15 qui avait été supprimé auparavant. Pour attribuer un PVID au hdiskpower14 il faut lancer la commande :

```
root@machine:/home$ chdev -l hdiskpower14 -a pv=yes
hdiskpower14 changed
```

On peut vérifier ensuite que tout est OK :

```
root@machine:/home$ lsattr -El hdiskpower14|grep pvid
pvid_takeover yes                               Takeover PVIDs from hdisks True
pvid          005477ca1dac607900000000000000000 Physical volume identifier False
```

Ensuite on ajoute le disque avec la commmande *extendvg*. Parfois un disque n'appartient à aucun VG d'après *lspv* mais *extendvg* indique le contraire ... Dans ce cas on peut vérifier avec cette *lqueryvg* :

```
lqueryvg -p hdiskpower26 -At
```

Activer les statistiques disques

```
chdev -l sys0 -a "iostat=true"
```

Afficher les détails d'un disque

```
lscfg -vl hdisk0
```

Retrouver ses disques après la perte d'une baie + reboot serveur

```
cfgmgr
```

ou

```
cfgmgr -vl fcs0
cfgmgr -vl fcs1
```

```
powermt restore
powermt display
```

Récupérer la VGDA suite à un changement de PVID d'un disque

Méthode soft

Soit le volume group :

```
root@server:/apps/sys/back# lsvg -l vg_toto
vg_toto:
LV NAME             TYPE       LPs   PPs   PVs  LV STATE      MOUNT POINT
fsfsfslv_WA_NFD     jfs2       3     3     1    closed/syncd  /applix/WA/PANF01P/NFDMUL_P
fsfslv00            jfs2       3     3     1    closed/syncd  /applix/WA/PANF01P/NFDMUL_D
lv_arv_PA           jfs2       50    50    1    closed/syncd  /archive/PAZO01P
lv_bkp_PA           jfs2       50    50    1    closed/syncd  /backup/PAZO01P
loglv01             jfs2log    1     1     1    closed/syncd  N/A
```

On simule l'explosion du bordel :

```
root@server:/apps/sys/back# varyoffvg vg_toto
root@server:/apps/sys/back# exportvg vg_toto
root@server:/apps/sys/back# chdev -l hdiskpower20 -a pv=clear
hdiskpower20 changed
root@server:/apps/sys/back# chdev -l hdiskpower20 -a pv=yes
hdiskpower20 changed
```

On récupère la VGDA qui est toujours avec l'anvien PVID :

```
root@server:/apps/sys/back# lqueryvg -p hdiskpower20  -At
0516-320 lqueryvg: Physical volume hdiskpower20 is not assigned to
        a volume group.
Max LVs:        256
PP Size:        28
Free PPs:       328
LV count:       5
PV count:       1
Total VGDAs:    2
Conc Allowed:   0
MAX PPs per PV  1016
MAX PVs:        32
Conc Autovaryo  0
Varied on Conc  0
Logical:        00ca4cde00004c0000000117a74a8870.1   fsfsfslv_WA_NFD 1
                00ca4cde00004c0000000117a74a8870.2   fsfslv00 1
                00ca4cde00004c0000000117a74a8870.3   lv_arv_PA 1
                00ca4cde00004c0000000117a74a8870.4   lv_bkp_PA 1
                00ca4cde00004c0000000117a74a8870.5   loglv01 1
Physical:       00ca4cde5d832ea7                2   0
Total PPs:      435
LTG size:       128
HOT SPARE:      0
AUTO SYNC:      0
VG PERMISSION:  0
SNAPSHOT VG:    0
IS_PRIMARY VG:  0
```

```
PSNFSTPP:       4352
VARYON MODE:    0
```

On tente de réimporter le VG :

```
root@server:/apps/sys/back# importvg -y vg_willy hdiskpower20
0516-304 getlvodm: Unable to find device id 00ca4cde5d832ea7 in the Device
        Configuration Database.
0516-022 : Illegal parameter or structure value.
0516-780 importvg: Unable to import volume group from hdiskpower20.
```

→ ça ne marche pas, forcément.

Solution miracle :

```
root@server:/apps/sys/back# recreatevg -y vg_willy hdiskpower20
vg_willy
root@server:/apps/sys/back# lsvg -l vg_willy
vg_willy:
LV NAME             TYPE       LPs   PPs   PVs  LV STATE      MOUNT POINT
fsfsfsfslv_WA_N     jfs2       3     3     1    closed/syncd  /fs/applix/WA/PANF01P/NFDMUL_P
fsfsfslv00          jfs2       3     3     1    closed/syncd  /fs/applix/WA/PANF01P/NFDMUL_D
fslv_arv_PA         jfs2       50    50    1    closed/syncd  /fs/archive/PAZO01P
fslv_bkp_PA         jfs2       50    50    1    closed/syncd  /fs/backup/PAZO01P
fsloglv01           jfs2log    1     1     1    closed/syncd  N/A
root@server:/apps/sys/back#
```

Par contre tous les lv sont prefixés avec lv et les fs avec /fs/ il faut faire un *chfs -m nouveau_fs ancien_fs* et *chlv -n nouveau_lv ancien_lv*.

Méthode pour les warriors et pour se la péter

On vérifie que c'est le bon PVID vu dans la VGDA :

```
root@server:/tmp# dd if=/dev/hdiskpower20  skip=4896  bs=16 count=1 | od -x
0000000  00ca 4cde 5d83 2ea7 0000 0000 0000 0000
1+0 records in
1+0 records out
```

On prépare un le fichier qui va bien :

```
PVID                        00      ca      4c      de     5d      83     2e      a7
en octal                    000     312     114     336    135     203    056     247
```

```
echo "\0000\0312\0114\0336\0135\0203\0056\0247\c "  > /tmp/oldpvid
```

On a bien le mauvais PVID :

```
root@server:/apps/sys/back# cat /tmp/oldpvid | od -x
0000000  00ca 4cde 5d83 2ea7 0000 0000 0000 0000
0000020
root@server:/apps/sys/back# dd if=/dev/hdiskpower20 bs=16 skip=8 count=1 | od -x
0000000  00ca 4cde 5dd3 8fea 0000 0000 0000 0000
```

On fait la modif :

```
root@server:/apps/sys/back# dd if=/tmp/oldpvid of=/dev/hdiskpower20 bs=16 count=1 seek=8
1+0 records in
1+0 records out
root@server:/apps/sys/back# dd if=/dev/hdiskpower20 bs=16 skip=8 count=1 | od -x
0000000  00ca 4cde 5d83 2ea7 0000 0000 0000 0000
1+0 records in
1+0 records out
0000020
```

Et, ça marche !

```
root@server:/apps/sys/back# lspv | grep diskpower20
hdiskpower20    00ca4cde5dd38fea
root@server:/apps/sys/back# powermt remove dev=20
root@server:/apps/sys/back# powermt config
root@server:/apps/sys/back# lspv | grep diskpower20
hdiskpower20    00ca4cde5d832ea7                    None
root@server:/apps/sys/back# importvg -y vg_toto hdiskpower20
```

```
vg_toto
```

Retirer un disque d'un VG

```
unmirrorvg rootvg hdisk1
sysdumpdev -l
rmlv lvdump1
reducevg rootvg hdisk1
rmdev -vl hdisk1 ou rmdev -Rdl hdisk1
```

Coredump sur extendlv

Si un jour vous avez un pb de coredump en faisant un extendlv sur un lv en superstrict en version d'AIX antérieur à 5.3, il faut vérifier l'upper bound du LV.

En fait, on doit avoir:

```
upper bound < nb_disques ds VG / nb_copie
```

Ce qui finalement est logique mais bon le pb est corrigé à partir de la 5.3

LVs partiellement mirrorés

```
0516-1147 : Warning - logical volume lv_cft may be partially mirrored.
lv_cft              jfs        10    12    3    open/syncd    /apps/cft
```

- D'abord, vous récupérer l'ID du LV en question

```
root@server3002909:/$ lslv lv_cft
LOGICAL VOLUME:     lv_cft                 VOLUME GROUP:   rootvg
LV IDENTIFIER:      00ce117b00004c000000010944969a6b.23 PERMISSION:     read/write
```

- Ensuite,

→ soit vous passer par un lslv -m pour voir la map et les PP mal foutu

```
root@server3002909:/$ lslv -m lv_cft
lv_cft:/apps/cft
LP    PP1  PV1               PP2  PV2               PP3  PV3
0001  0167 hdisk1            0101 hdisk0
0002  0168 hdisk1            0102 hdisk0
0003  0183 hdisk1
0004  0184 hdisk1
0005  0185 hdisk1
0006  0186 hdisk1
0007  0187 hdisk1
0008  0188 hdisk1
0009  0189 hdisk1
0010  0190 hdisk1
```

Et vous créez un fichier avec la map en erreur. Le format est PVID PP2 LP.

```
root@server3002909:/$ lspv
hdisk0          00ce117b4496972c                    rootvg          active
```

Dans mon cas, le fichier :

```
00ce117b4496972c 101 1
00ce117b4496972c 102 2
```

→ soit vous faites un lquerylv, vous redirigez dans un fichier et vous ne gardez que les lignes avec les PP en erreur:

```
lquerylv -L 00ce117b00004c000000010944969a6b.23 -r > /tmp/map
```

- Vous comptez le nombre de ligne dans votre fichier

```
cat /tmp/map | wc -l
```

- Enfin vous balancez la commande magique. (le paramètre -s correspond au nb de PP à corriger)

```
lreducelv -l 00ce117b00004c000000010944969a6b.23 -s 2 /tmp/map
```

- Vous faites un mklvcopy puis syncg -l comme d'hab pour remirrorer et resynchroniser tout le bazar

Passer un disque de Defined en Available

```
mkdev -l hdiskX
```

MPIO

- Lister les paths

```
root@client:/root # lspath |grep -w "hdisk[0,1]"
Enabled hdisk0  vscsi4
Enabled hdisk0  vscsi0
Enabled hdisk1  vscsi4
Enabled hdisk1  vscsi0
```

- Attributs des disques

```
root@client:/root # lsattr -El hdisk0
PCM             PCM/friend/vscsi                Path Control Module        False
algorithm       fail_over                       Algorithm                  True
hcheck_cmd      test_unit_rdy                   Health Check Command       True+
hcheck_interval 60                              Health Check Interval      True+
hcheck_mode     nonactive                       Health Check Mode          True+
max_transfer    0x40000                         Maximum TRANSFER Size      True
pvid            00f6a14eab7fa4720000000000000000 Physical volume identifier False
queue_depth     32                              Queue DEPTH                True
reserve_policy  no_reserve                      Reserve Policy             True+
```

- Priorités

```
root@client:/tmp # chpath -l hdisk0 -p vscsi0 -a priority=2
path Changed
```

Documentations

Intitulé	Format
Understanding DIO	
AIX Logical Volume Manager from A to Z: Troubleshooting and Commands	