RAID et LVM

RAID et LVM

Plan de la phase

Introduction

Partionnement avancé RAID

- *Définition*
- *Précautions et considérations d'usage*
- *RAID avec mdadm*
- *Etat du RAID*
- *Simuler une panne*
- *Remplacer un disque*
- *Arrêt et relance manuels*

Initiation au LVM

- *Principe*
- *Les volumes physiques*
- *Les groupes de volumes*
- *Les volumes logiques*
- *Agrandissements et réductions*
- *Supprimer un groupe de volumes*
- *Commandes supplémentaires*

Validation des acquis : questions/réponses

Travaux pratiques

RAID et LVM

Introduction

À la fin de ce chapitre, vous serez en mesure :

- De comprendre les différences entre les niveaux de RAID.
- De créer et modifier des matrices RAID.
- De gérer les pannes et les modes dégradés.
- De mettre en place un LVM.
- De modifier à volonté la taille de vos volumes logiques.
- D’agrandir et de réduire vos systèmes de fichiers.

RAID et LVM

Partitionnement avancé RAID - définition

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

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

RAID et LVM

Partitionnement avancé RAID - définition

RAID-0 : appelé ***stripe mode*** : **deux disques au moins forment un seul volume**. Les deux disques ont en principe la **même taille**. Chaque opération de **lecture/écriture** sera **fractionnée** et **effectuée** sur **chacun des disques**. Par exemple, 4 ko seront écrits sur le disque 0, 4 ko sur le disque 1, 4 ko sur le disque 2, puis 4 ko sur le disque 0, etc. Ainsi, les **performances** sont **accrues** puisque les opérations de **lecture et d'écriture** sont effectuées **en parallèle** sur les disques. Si N est le nombre de disques et P la vitesse de transfert, la vitesse de transfert du volume RAID est en principe proche de N*P mbps. Le RAID-0 n'a **aucune redondance**. En cas de **panne** d'un des **disques**, il est probable que l'**ensemble des données** soit **perdu**.

RAID-1 : appelé ***mirroring*** : **premier mode redondant**. Il peut être utilisé à partir de **deux disques ou plus** avec d'éventuels **disques de secours (Spare Disk)**. Chaque **information** écrite sur un disque est **dupliquée** sur les autres. Si **N-1 disques** du RAID viennent à **tomber**, les **données** restent **intactes**. Si un **disque de secours** est présent, en cas de panne, il est **automatiquement reconstruit** et **prend** la **place** du **disque défaillant**. Les **performances** en **écriture** peuvent être **mauvaises** : écriture sur N disques en même temps, risquant de saturer le contrôleur disque et le bus. Les **performances** en **lecture** sont **bonnes**, car RAID emploie un algorithme qui peut lire les données sur chaque disque (puisqu'ils sont identiques).

RAID-5 : ***RAID avec bande de parité redistribuée***. C'est le **mode le plus utilisé** car c'est celui qui offre le **meilleur compromis entre** le **nombre de disque**s, l'**espace disponible** et la **redondance**. Il faut **au moins trois disques** avec d'éventuels **disques de secours**. La **parité** est présente **sur chacun des disques**. La **taille** finale est celle de **N-1 disques**. Le RAID-5 **survit** à **une panne** de **disque**. Dans ce cas, si un disque de secours est présent, il sera automatiquement reconstruit. Les **performances** en **lecture** sont équivalentes à celles du **RAID-0** tandis qu'en **écriture**, elles **dépendent** de **l'algorithme** employé et de la mémoire de la machine.

RAID et LVM

Partitionnement avancé RAID - Précautions et considérations d'usage

a. Disque de secours

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

b. Disque défectueux

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

c. Boot

La partition de boot (celle qui contient le noyau, la configuration du bootloader, les fichiers images de disques) **ne doit pas être placée dans une matrice RAID** : le chargeur de démarrage est incapable de monter des partitions RAID (la prochaine version de GRUB en sera capable).
Référence : cf grub.pdf, page 10 (GRUB 2 can read files directly from LVM and RAID devices.)

- Voir aussi :
  - Linux Magazine Hors-série N°48
  - LM HS n°18
  - ressources/adm_part3.pdf
  - https://wiki.archlinux.org/index.php/GRUB2#LVM
  - http://komo.simoko.eu/glpi/front/knowbaseitem.form.php?id=124

d. Swap

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

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

Cependant, en cas de besoin de **haute disponibilité**, le swap sur le RAID est **possible (*cf note*)**.

e. Périphériques

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

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

f. IDE

Si les disques IDE ont longtemps été le SCSI du pauvre (matériel de moins bonne qualité, lenteur, manque de fiabilité) ce n'est plus vraiment le cas. Les derniers modèles sont totalement identiques aux disques SCSI, contrôleur excepté. **Vous pouvez donc monter pour un coût raisonnable des configurations RAID en IDE. Cependant une règle est à retenir :**

UN SEUL DISQUE IDE PAR BUS IDE

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

g. Hot Swap

IDE : **NE JAMAIS DEBRANCHER À CHAUD UN DISQUE IDE** ! C'est le meilleur moyen de détruire le disque, si ce n'était pas encore le cas et de détruire le contrôleur IDE (et donc éventuellement la carte mère ou additionnelle). L'IDE n'est pas prévu pour.

SCSI : les **contrôleurs SCSI** ne sont **pas prévus** pour le **Hot Swap** mais **devraient** en **théorie** tout de même **fonctionner**, si le disque est identique physiquement et logiquement. ***(note)***

SAS (*Serial Attached SCSI*) : supporte le Hot Swap

SATA : le SATA est reconnu comme du SCSI. La spécification **SATA** en **version 2 supporte théoriquement le Hot Swap**. Seulement, la plupart des contrôleurs actuels implémentent mal ou pas du tout cette possibilité, d'où les risques de plantages ou de griller son contrôleur. **Référez-vous à la documentation du constructeur de votre carte mère (chipset)**.

https://raid.wiki.kernel.org/index.php/Hardware_issues#Hot_Swap

a. Préparation

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

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

https://raid.wiki.kernel.org/index.php/RAID_setup

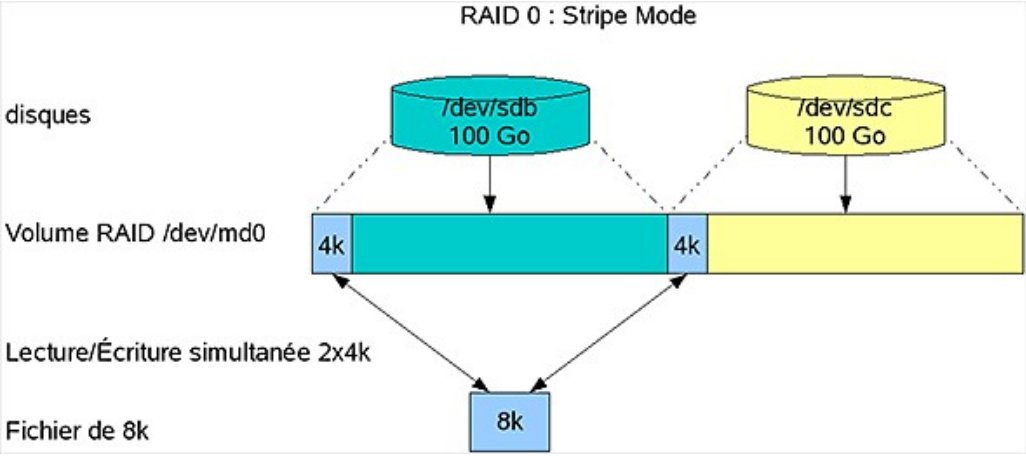
RAID et LVM

Partitionnement avancé RAID - RAID avec mdadm

b. Création

RAID-0

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




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

- `/dev/md0` : Nom du fichier périphérique de type bloc représentant la matrice RAID.
- `--level` : Type de RAID à créer : 0, raid0 et stripe pour du RAID0.
- `--raid-devices` : Nombre de partitions utilisées pour créer la matrice.

`/dev/sdb1, /dev/sdc1` : Partitions constituant la matrice, suivant le nombre indiqué dans --raid-devices.

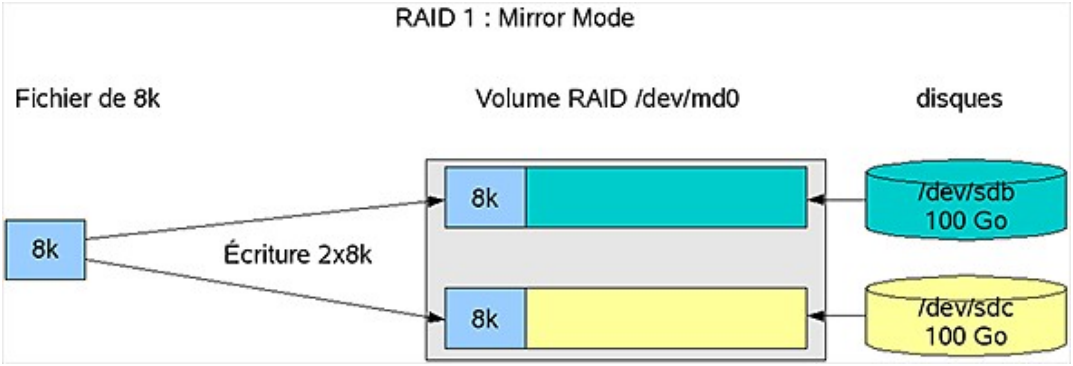
• Il ne reste plus qu'à installer le système de fichiers sur le disque RAID :

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

b. Création

RAID-1

C'est le même principe. Vous allez cette fois rajouter une partition de secours */dev/sdd1*.

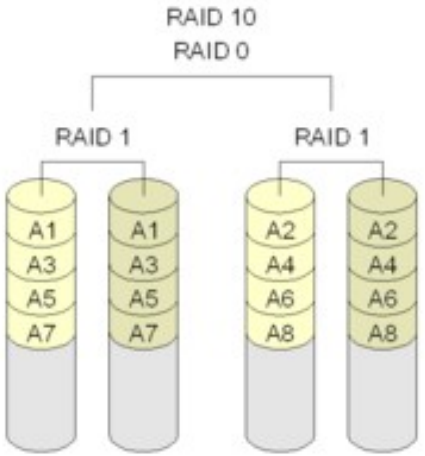



```
# mdadm --create /dev/md0 --level=raid1 --raid-devices=2 /dev/sdb1 /dev/sdc1 --spare-devices=1
/dev/sdd1
```

- `--level 1, mirror ou raid1` : sont de bonnes valeurs pour un RAID-1.
- `--spare-devices` : nombre de disques de secours à utiliser.
- `/dev/sdd1` : partitions constituant les disques de secours, suivant le nombre indiqué dans --spare-devices.
- Puis :

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

b. Création



RAID-1+0

Il faut au moins **quatre partitions**. Vous devez créer **deux matrices RAID-1** que vous allez **regrouper** en une **matrice RAID-0**.

Sa **fiabilité** est assez **grande** puisqu'il faut que tous les éléments d'une grappe soient défectueux pour entraîner un défaut global.
La **reconstruction** est assez **performante** puisqu'elle ne mobilise que les disques d'une seule grappe et non la totalité.

```
# mdadm --create /dev/md0 --level=raid1 --raid-devices=2 /dev/sdb1 /dev/sdc1

# mdadm --create /dev/md1 --level=raid1 --raid-devices=2 /dev/sdd1 /dev/sde1

# mdadm --create /dev/md2 --level=raid0 --raid-devices=2 /dev/md0 /dev/md1
```

Puis :

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

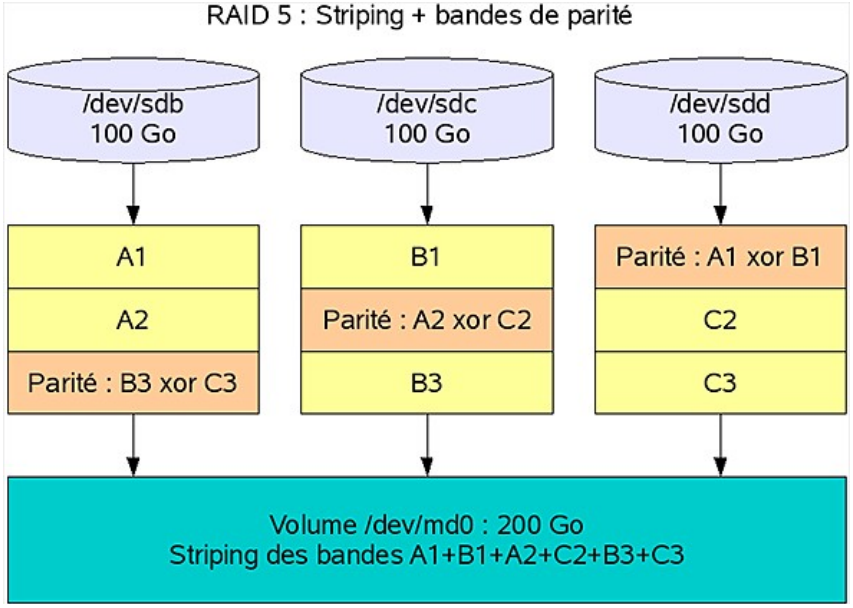
RAID et LVM

Partitionnement avancé RAID - RAID avec mdadm

b. Création

RAID-5

Vous allez utiliser trois disques de données */dev/sdb1*, */dev/sdc1*, */dev/sdd1*
et un disque de secours */dev/sde1*.



En cas de défaillance d'un disque, les données qui s'y trouvaient pourront être reconstituées par l'opération xor. En effet, l'opération XOR ($\oplus$) a la propriété suivante : si on considère N blocs de taille identique $A_1, A_2 \cdots A_N$ et si $A_1 \oplus A_2 \oplus \cdots A_N = X$ alors $X \oplus A_2 \oplus \cdots A_N = A_1$, et de façon générale, $A_1 \oplus \cdots A_{k-1} \oplus X \oplus A_{k+1} \oplus \cdots A_N = A_k$.

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

Puis :

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

c. Sauver la configuration

Pour faciliter la tâche de l'outil mdadm, **vous pouvez créer** (ce n'est pas obligatoire) le **fichier** de **configuration /etc/mdadm.conf**. Ce fichier peut être **créé** manuellement mais l'outil mdadm sait le générer. Il est préférable de le faire **APRÈS** la **création des matrices RAID**.

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

(sous **Debian** : c'est le fichier **`/etc/mdadm/mdadm.conf`**)

RAID et LVM

Partitionnement avancé RAID - État du RAID

Le fichier virtuel **/proc/mdstat** contient des **informations** sur le **RAID**. C'est ici que vous pouvez voir le **détail** d'un **RAID**, notamment si un des **volumes** de la matrice est **défectueux** (Faulty).

```
debian:/home/komo# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sdc1[1] sdb1[0]
      1044096 blocks [2/2] [UU]

unused devices: <none>
```

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

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

RAID et LVM

Partitionnement avancé RAID - État du RAID

Vous pouvez aussi utiliser `mdadm` avec le paramètre `--detail` :

```
debian:/home/komo# mdadm --detail /dev/md0
/dev/md0:
        Version : 00.90
  Creation Time : Sun Nov 15 22:07:50 2009
     Raid Level : raid1
     Array Size : 1044096 (1019.80 MiB 1069.15 MB)
  Used Dev Size : 1044096 (1019.80 MiB 1069.15 MB)
   Raid Devices : 2
  Total Devices : 2
Preferred Minor : 0
    Persistence : Superblock is persistent

    Update Time : Sun Nov 15 22:42:29 2009
          State : clean
 Active Devices : 2
Working Devices : 2
 Failed Devices : 0
  Spare Devices : 0

           UUID : 17090b20:92b93cf5:9d4deba6:47ca997f
         Events : 0.38

    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /dev/sdb1
       1       8       33        1      active sync   /dev/sdc1
```

Remarquez qu'avec cette dernière commande vous obtenez bien plus de détails, notamment quels sont les disques "spare" et "faulty".

RAID et LVM

Partitionnement avancé RAID - Simuler une panne

Vous allez **simuler une panne sur /dev/sdc1** :

```
debian:/home/komo# mdadm /dev/md0 -f /dev/sdc1
mdadm: set /dev/sdc1 faulty in /dev/md0
```

Regardez l'état du RAID dans `/proc/mdstat` durant l'exécution :

```
Personalities : [linear] [multipath] [raid0] [raid1] [raid6] [raid5] [raid4] [raid10]
md0 : active raid1 sdd1[2] sdc1[3](F) sdb1[0]
      1044096 blocks [2/1] [U_]
      [==========>.........]  recovery = 50.9% (532992/1044096) finish=0.0min speed=88832K/sec
```

Remarquez qu'un « (F) » est apparu près de sdc1, indiquant un disque Faulty. On voit aussi que sur les deux disques, un est en panne et que le RAID reconstruit sa matrice avec le spare disk. Après l'exécution, vous obtenez :

```
md0 : active raid1 sdd1[1] sdc1[2](F) sdb1[0]
      1044096 blocks [2/2] [UU]
```

RAID et LVM

Partitionnement avancé RAID - Simuler une panne

Le RAID est reconstruit et fonctionne à merveille

```
          State : clean
 Active Devices : 2
Working Devices : 2
 Failed Devices : 1
  Spare Devices : 0

           UUID : 37660f22:ea8ce952:6822a9fe:380d36f1
         Events : 0.10

    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /dev/sdb1
       1       8       49        1      active sync   /dev/sdd1

       2       8       33        -      faulty spare   /dev/sdc1
```

Le disque Faulty est bien /dev/sdc1 ; /dev/sdd1 a pris sa place en tant que disque de secours. Ainsi, le disque de secours devient un disque RAID de la matrice.

RAID et LVM

Partitionnement avancé RAID - Remplacer un disque

Puisque **/dev/sdc1** est en panne, vous allez le **remplacer**. Retirez-le avec -r (ou --remove) :

```
debian:/home/komo# mdadm /dev/md0 -r /dev/sdc1
mdadm: hot removed /dev/sdc1
```

```
md0 : active raid1 sdd1[1] sdb1[0]
      1044096 blocks [2/2] [UU]
```

Constatez que **sdc1 a disparu**. Vous pouvez éteindre la machine puis remplacer le disque défaillant. Rallumez la machine, puis repartitionnez le disque correctement. Il n'y a plus qu'à r**ajouter** le **disque** réparé **dans** la **matrice** RAID avec **-a** (--add) :

Pour r**éutiliser un disque** faisant partie d'un **ancien RAID** au lieu d'utiliser un disque neuf :

```
debian:/home/komo# mdadm --zero-superblock /dev/sdc1
```

```
debian:/home/komo# mdadm /dev/md0 -a /dev/sdc1
mdadm: added /dev/sdc1
```

```
md0 : active raid1 sdc1[2](S) sdd1[1] sdb1[0]
      1044096 blocks [2/2] [UU]
```

Le **disque sdc1** apparaît à nouveau et est devenu le nouveau **disque de secours** !

RAID et LVM

Partitionnement avancé RAID - Arrêt et relance manuels

Vous pouvez **arrêter** ponctuellement une **matrice RAID** avec -S (--stop) **APRÈS** avoir **démonté** le **périphérique** :

```
debian:/home/komo# mdadm -S /dev/md0
mdadm: stopped /dev/md0

Every 2,0s: cat /proc/mdstat                                    Sun Nov 15 23:43:25 2009

Personalities : [linear] [multipath] [raid0] [raid1] [raid6] [raid5] [raid4] [raid10]
unused devices: <none>
```

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

```
debian:/home/komo# mdadm -As /dev/md0
mdadm: /dev/md0 has been started with 2 drives and 1 spare.

md0 : active (auto-read-only) raid1 sdb1[0] sdc1[2](S) sdd1[1]
      1044096 blocks [2/2] [UU]
```

```
debian:/home/komo# cat /etc/mdadm/mdadm.conf
DEVICE /dev/sdb1 /dev/sdc1 /dev/sdd1
ARRAY /dev/md0 devices=/dev/sdb1,/dev/sdc1,/dev/sdd1
```

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

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

RAID et LVM

Initiation au LVM - Principe

Le Logical Volume Manager est **un système de gestion** très perfectionné **des supports de stockage**. **Le but est de dépasser**, voire transcender **la gestion physique des disques**, et leur organisation logique basique (les partitions) pour étendre la capacité globale des supports, **à l'aide d'une gestion entièrement logique de celle-ci**.

Un LVM permet, tout comme le RAID 0 par exemple, de créer des espaces de données logiques sur plusieurs disques. Il permet aussi de faire du mirroring, comme le RAID 1. Mais la comparaison s'arrête là. Le RAID logiciel se contente de créer une « partition » dans un espace de stockage défini par le RAID lui-même (par exemple une partition de 100 Go dans un RAID 0 de deux disques de 50 Go).

Le LVM regroupe les disques physiques, ou **tout** autre **support** de **stockage** dit **physique** (disque, RAID matériel, RAID logiciel, support de stockage en provenance d'un SAN), qu'il **appelle** des **volumes physiques *PV*** (Physical Volume) **en** un **groupe** de **volumes *VG*** (Volume Group). Ce groupe VG est **vu** par le LVM comme une sorte de **métadisque**, **dans lequel** vous allez **créer** des **volumes logiques *LV*** (Logical Volume) à volonté.

- Volume physique PV : un support de stockage de données dit physique : disque dur par exemple ;
- Groupe de volumes VG : un regroupement logique de 1 à n PV ;
- Volume logique LV : un découpage logique au sein d'un VG.

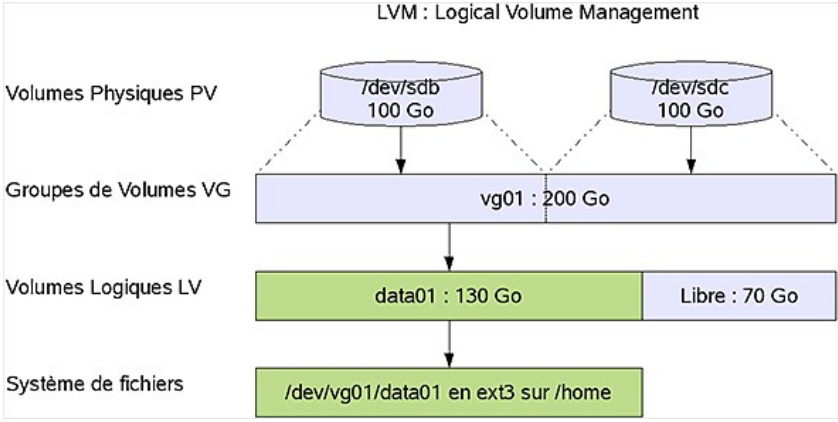
RAID et LVM

Initiation au LVM - Principe

Un **volume logique** est **vu** comme une **partition**, et est utilisable comme telle. Il peut **contenir** des **données**, il suffit de **créer un système de fichiers** ordinaire (ext3 par exemple) et de le **monter** de manière tout à fait classique.

Contrairement au RAID 0 logiciel où la partition de données doit occuper tout l'espace, il est **possible** de **créer** autant de **volumes logiques** de **toute taille** que souhaité. Mais cela va bien plus loin.

Le **LVM** est **dynamique**. Il est possible d'**ajouter** et de **supprimer** des **volumes physiques** d'un **groupe** de **volumes**. En ajoutant des volumes physiques, la capacité, et donc l'**espace disponible du groupe augmente**. Le nouvel espace disponible peut permettre de **créer** des nouveaux **volumes logiques**, mais aussi **d'agrandir** un **volume logique existant**.



Un volume logique est dynamique : il peut être agrandi ou réduit à volonté, ce qui **implique** qu'il faut aussi **pouvoir agrandir** un **système de fichiers**, ou le **réduire**.

Notez enfin qu'une **matrice RAID** peut être **utilisée** comme **volume physique**.

La **configuration** du **LVM** est **située** dans les fichiers et répertoires présents dans **/etc/lvm**. Le fichier **/etc/lvm/lvm.conf** contient la **configuration globale**. La **configuration des différents volumes** (physiques, groupes et logiques) ne s**e trouve** pas dans un fichier, mais dans une **structure** présente **au sein** des **périphériques** eux-mêmes, dans les **premiers blocs** : ce sont les **métadatas** des **volumes physiques**.

RAID et LVM

Initiation au LVM - Les volumes physiques

a. Créer un volume physique

Ici au 19/11/12 à 10h30

Un **volume physique** peut être un **disque complet** ou une **partition** classique au sein d'un disque. Dans ce cas, la **partition** doit être **de type *0x8e***.

Voici le retour de la commande fdisk sur */dev/sde*. Distinguez les partitions primaires 2 et 3 de type ***8e*** qui vont servir pour les exemples suivants.

```
debian:/home/komo# fdisk -l /dev/sde

Disk /dev/sde: 3221 MB, 3221225472 bytes
255 heads, 63 sectors/track, 391 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0x00000000

   Device Boot      Start         End      Blocks   Id  System
/dev/sde1               1         124      995998+  83  Linux
/dev/sde2             125         248      996030   8e  Linux LVM
/dev/sde3             249         391     1148647+  8e  Linux LVM
```

Une fois les partitions créées, utilisez la commande ***pvcreate* sur** une première **partition** (plusieurs partitions peuvent être précisées) :

```
debian:/home/komo# pvcreate /dev/sde2
  Physical volume "/dev/sde2" successfully created
```

b. Voir les volumes physiques

La commande ***pvdisplay*** permet de **visualiser** l'ensemble des **volumes physiques** accessibles sur votre système. Elle peut prendre aussi un nom de volume spécifique.

```
debian:/home/komo# pvdisplay /dev/sde2
  "/dev/sde2" is a new physical volume of "972,69 MB"
  --- NEW Physical volume ---
  PV Name               /dev/sde2
  VG Name
  PV Size               972,69 MB
  Allocatable           NO
  PE Size (KByte)       0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               parBMc-aht9-B03A-VBJW-9Sz7-5feh-ibc1i9
```

Pour le moment, les informations sont réduites. **Le PV n'appartient** encore à **aucun groupe** de **volumes** (ligne VG Name). Sa **taille** est de **1 Go**. Les lignes les plus intéressantes sont les lignes (pour l'instant vides car le PV n'appartient pas à un VG) où est indiqué PE. ***PE*** signifie Physical Extend, extension physique. **Chaque VG**, et donc **PV** le constituant, est **découpé** en **tranches appelées PE**. ***Le PE est l'unité de base de travail du LVM***. Si un PE fait 4 Mo, cela signifie que l'espace pourra être découpé au sein du groupe de volumes par tranches de 4 Mo. L'allocation se fait par PE : la création d'**un volume logique** de **500 PE de 4 Mo** fait donc **2000 Mo**.

Les valeurs à zéro seront remplies dès que le PE sera dans un VG.

RAID et LVM

Initiation au LVM - Les groupes de volumes

a. Créer un groupe de volumes

Pour créer un groupe de volumes, vous devez **disposer** d'**au moins un volume physique**. Vous pouvez **créer** un **groupe de volumes** avec la commande ***vgcreate***. Un groupe de volumes porte un **nom**, celui que vous voulez. C'est le **premier argument** de la commande. Vous passez **ensuite** comme argument **la liste des volumes physiques** composant le groupe de volumes, ici un seul, */dev/sdb2*.

```
debian:/home/komo# vgcreate vg01 /dev/sde2
  Volume group "vg01" successfully created
```

b. Propriétés d'un VG

Le groupe de volumes a de nombreuses **propriétés**. Il peut être **étudié avec** la commande ***vgdisplay***.

- **MAX LV** : indique le nombre maximum de volumes logiques qui pourront être créés dans ce groupe de volumes (La valeur zéro indique un nombre théoriquement infini)

- **MAX PV** : indique le nombre maximum de volumes physiques pouvant être ajoutés au groupe de volumes (Là encore, le zéro indique un nombre infini).

```
debian:/home/komo# vgdisplay vg01
  --- Volume group ---
  VG Name               vg01
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  1
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               972,00 MB
  PE Size               4,00 MB
  Total PE              243
  Alloc PE / Size       0 / 0
  Free  PE / Size       243 / 972,00 MB
  VG UUID               PgTGU0-mzT8-l28G-MhEq-IALe-Dx1E-O39bdY
```

b. Propriétés d'un VG

MAX LV, MAX PV et PE Size peuvent être **positionnées** à la création du groupe de volumes à l'aide des paramètres suivants de la commande *pvcreate* :

- **-l** Nombre maximum de volumes logiques
- **-p** Nombre maximum de volumes physiques
- **-s** Taille des extensions physiques (avec un suffixe k, m, g ou t pour préciser l'unité).

Les dernières lignes concernent les PE (extensions physiques). Le **groupe dispose** actuellement de **243 extensions de 4 Mo**, **soit 972 Mo**, toutes libres. Les volumes logiques occuperont ensuite un certain nombre de ces PE, selon leur taille.

La commande ***vgdisplay* accepte** le paramètre ***-v*** qui **donne plus de détails**, et notamment la liste des volumes physiques qui le composent.

```
--- Physical volumes ---
PV Name               /dev/sde2
PV UUID               parBMc-aht9-B03A-VBJW-9Sz7-5feh-ibc1i9
PV Status             allocatable
Total PE / Free PE    243 / 243
```

b. Propriétés d'un VG

Maintenant que le PV /dev/sdb2 fait partie d'un VG, plus d'informations sont disponibles :

```
debian:/home/komo# pvdisplay /dev/sde2
  --- Physical volume ---
  PV Name               /dev/sde2
  VG Name               vg01
  PV Size               972,69 MB / not usable 702,00 KB
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              243
  Free PE               243
  Allocated PE          0
  PV UUID               parBMc-aht9-B03A-VBJW-9Sz7-5feh-ibc1i9
```

RAID et LVM

Initiation au LVM - Les volumes logiques

a. Créer un volume logique

Un **volume logique** est un **découpage** d'un **VG** (groupe de volumes) qui est l'**équivalent** d'une **partition** dans laquelle vous pourrez **créer** un **système de fichiers**. Un volume logique **LV occupe** un certain nombre de **PE** (extensions physiques) **d'un VG**, **contigus ou non**. Ceci a son importance pour la suite car :

- il est **possible** d**'agrandir** un **LV** tant qu'il **reste** des **PE** de libres dans le VG.
- il est possible de **réduire** un **LV**, ce qui **libérera** des **PE** dans le VG, utilisables pour créer de nouveaux LV ou pour les agrandir.

Ceci signifie que le LVM gère une sorte d'index des PE, et leur ordre, pour savoir à quel LV appartient un PE.

Vous **créez un volume logique** avec la commande ***lvcreate***. Un volume logique **porte** un **nom**, dispose d'une **taille exprimée** soit **en** extensions logiques ***LE*** (Logical Extension) qui sont la **représentation** des **PE** au sein d'un LV, soit en Ko, Mo, Go… La **commande** suivante **crée** un **volume logique** appelé **data01** au sein du VG **vg01**, d'une taille de **512 Mo**. Le -L précise que l'unité est en Mo (m), Go (g), To (Teraoctet, t), Po (Petaoctet), ou Eo (Exaoctet). Pour préciser un nombre de PE, utilisez « -l ».

```
debian:/home/komo# lvcreate -n data01 -L 512m vg01
  Logical volume "data01" created
```

RAID et LVM

Initiation au LVM - Les volumes logiques

a. Créer un volume logique

Un **LV** est **vu** comme une **partition**, et **dispose** après sa création d'un **fichier périphérique** associé. Le fichier est **dans** le dossier **`/dev/<nom_du_vg>/<nom_du_lv>`**. Notez qu'**il s'agit** d'un **lien symbolique vers** un fichier de **`/dev/dm-0`**.

```
Debian:/home/komo# ls -l /dev/vg01/data01
lrwxrwxrwx 1 root root 7 21 nov.  01:00 /dev/vg01/data01 -> ../dm-0
```

b. Propriétés d'un volume logique

Les **propriétés** d'un volume logique sont accessibles par la commande ***lvdisplay*** :

```
debian:/home/komo# lvdisplay /dev/vg01/data01
  --- Logical volume ---
  LV Name                /dev/vg01/data01
  VG Name                vg01
  LV UUID                q99zB2-VQG0-6uUD-9VUW-aY1J-2gK6-bzu0Db
  LV Write Access        read/write
  LV Status              available
  # open                 0
  LV Size                512,00 MB
  Current LE             128
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:0
```

La répartition des extensions physiques occupées par le volume logique au sein de chaque volume physique est donnée par :

```
lvdisplay -m /dev/vg01/data01

  --- Segments ---
  Logical extent 0 to 127:
    Type                linear
    Physical volume     /dev/sde2
    Physical extents    0 to 127
```

c. Accès au volume logique

Vous pouvez créer un système de fichiers et monter le LV comme pour n'importe quelle partition :

```
debian:/home/komo# mkfs -t ext3 /dev/vg01/data01
```

Il ne reste plus qu'à monter le nouveau système de fichiers

```
debian:/home/komo# mount /dev/vg01/data01 /mnt
```

a. Les groupes de volumes

Pour le moment, tout reste assez classique. **La force du LVM est son dynamisme**. L'étape suivante consiste à l'exploiter. Vous voulez maintenant **créer** un **nouveau LV** de **512 Mo** appelé **data02** au sein du **VG vg01**. Voici l'état actuel de vg01 :

```
debian:/home/komo# vgdisplay vg01
  --- Volume group ---
  VG Name               vg01
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  2
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                1
  Open LV               1
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               972,00 MB
  PE Size               4,00 MB
  Total PE              243
  Alloc PE / Size       128 / 512,00 MB
  Free  PE / Size       115 / 460,00 MB
  VG UUID               PgTGU0-mzT8-l28G-MhEq-IALe-Dx1E-O39bdY
```

Il n'y a **plus assez de place**. Seuls **460 Mo** (115 PE) sont disponibles. Il faut r**ajouter** au sein de ce **VG** un **nouveau volume physique**. Ceci se fait avec la commande ***vgextend*** qui fonctionne de la même manière que *vgcreate* : **indiquez** le nom du **VG suivi** du ou **des PV** à rajouter.

```
debian:/home/komo# pvcreate /dev/sde3
  Physical volume "/dev/sde3" successfully created

debian:/home/komo# vgextend vg01 /dev/sde3
  Volume group "vg01" successfully extended
```

a. Les groupes de volumes

Voici le **nouvel état de vg01**. Remarquez que le **VG contient maintenant deux PV**, et que **1,54 Go** (395 PE) sont **disponibles**.

```
debian:/home/komo# vgdisplay vg01
  --- Volume group ---
  VG Name               vg01
  System ID
  Format                lvm2
  Metadata Areas        2
  Metadata Sequence No  3
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                1
  Open LV               1
  Max PV                0
  Cur PV                2
  Act PV                2
  VG Size               2,04 GB
  PE Size               4,00 MB
  Total PE              523
  Alloc PE / Size       128 / 512,00 MB
  Free  PE / Size       395 / 1,54 GB
  VG UUID               PgTGU0-mzT8-l28G-MhEq-IALe-Dx1E-O39bdY
```

Il ne reste plus qu'à créer le LV data02 de 512 Mo :

```
debian:/home/komo# lvcreate -n data02 -L 512m vg01
  Logical volume "data02" created
```

RAID et LVM

Initiation au LVM - Agrandissements et réductions

a. Les groupes de volumes

Quand vous créez un LV, le **LVM** cherche à **optimiser l'utilisation** des **PE** de manière à ce **qu'ils** soient les plus **contigus** possibles et **si possible** sur un **même PV**. Ceci se voit avec la commande **`lvdisplay`** et le paramètre **`-m`**, sur les lignes `Segments` et la liste des segments proposés.

Les commandes suivantes **créent** le **système de fichiers** et **montent** celui-ci :

```
debian:/home/komo# mkfs -t ext3 /dev/vg01/data02
debian:/home/komo# mount /dev/vg01/data02 /mnt/data02
```

```
debian:/home/komo# lvdisplay -m /dev/vg01/data02
  --- Logical volume ---
  LV Name                /dev/vg01/data02
  VG Name                vg01
  LV UUID                9VTKH4-M6hk-NmEa-aYEA-2rn4-tiqb-m9UmzL
  LV Write Access        read/write
  LV Status              available
  # open                 0
  LV Size                512,00 MB
  Current LE             128
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:1

  --- Segments ---
  Logical extent 0 to 127:
    Type                linear
    Physical volume     /dev/sde3
    Physical extents    0 to 127
```

b. Agrandir un volume logique

Il se trouve que le LV **data01** de 512 Mo est trop petit. Il **doit** en **faire** le **double**. Il faut lui **ajouter 512 Mo**, ce qui est **possible** car il **reste 1 Go** (267 PE) **dans** le groupe de volumes **vg01** :

```
debian:/home/komo# vgdisplay vg01 | grep Free
  Free  PE / Size       267 / 1,04 GB
```

L'agrandissement d'un volume logique s**e fait dans cet ordre** :

- **Agrandissement du LV avec la commande *lvextend***
- **Agrandissement du système de fichier avec *resize2fs* (*ext3*)**

Agrandissement du LV

La commande ***lvextend* autorise** les paramètres *-l* (nombre d'extensions logiques LE) ou ***-L* comme pour *lvcreate***. Vous **précisez** ensuite la **nouvelle taille** du LV **ou**, si vous rajoutez un **+ en préfixe**, la **taille additionnelle** souhaitée. Vous pouvez aussi préciser, **en dernier argument**, le nom du **PV** sur lequel forcer l'extension du **LV** (c'est aussi possible avec *lvcreate*). Ça ne marchera que si le ou les PV précisés disposent d'assez de PE.

b. Agrandir un volume logique

Agrandissement du LV

La commande suivante r**ajoute 128 LE** (**4x128=512 Mo**) dans data01 :

```
debian:/home/komo# lvextend -l +128 /dev/vg01/data01
  Extending logical volume data01 to 1,00 GB
  Logical volume data01 successfully resized
```

Regardez maintenant **sur quels PV les données sont situées** :

Le volume logique **data01 occupe** bien **1 Go**, **sur deux segments** de PE, ces segments étant **sur les PV /dev/sde2 et /dev/sde3**. Le LVM a donc attribué un espace sur l'ensemble des PV du VG.

```
debian:/home/komo# lvdisplay -m /dev/vg01/data01
  --- Logical volume ---
  LV Name                /dev/vg01/data01
  VG Name                vg01
  LV UUID                q99zB2-VQG0-6uUD-9VUW-aY1J-2gK6-bzu0Db
  LV Write Access        read/write
  LV Status              available
  # open                 1
  LV Size                1,00 GB
  Current LE             256
  Segments               2
  Allocation             inherit
  Read ahead sectors     auto
  - currently set to     256
  Block device           253:0

  --- Segments ---
  Logical extent 0 to 242:
    Type                linear
    Physical volume     /dev/sde2
    Physical extents    0 to 242

  Logical extent 243 to 255:
    Type                linear
    Physical volume     /dev/sde3
    Physical extents    128 to 140
```

b. Agrandir un volume logique

Extension du système de fichiers

Seul le volume logique a été agrandi. Pour le moment, **la taille du système de fichiers** contenu dans data01 **n'a pas changé** :

```
debian:/home/komo# df -h /mnt/data01
Sys. de fich.         Tail. Occ. Disp. %Occ. Monté sur
/dev/mapper/vg01-data01
                      504M   17M  462M   4% /mnt/data01
```

La commande ***resize2fs*** **permet** de **réduire** et **d'agrandir** un **système de fichiers**. Le premier argument est le système de fichiers, le second la taille, avec un éventuel suffixe K (Ko), M (Mo), ou G (Go). Sans suffixe, c'est le nombre de blocs du système de fichiers qui est indiqué. Si **la taille est absente**, le système de fichiers sera **adapté à la taille** de la **partition ou** du **LV**.

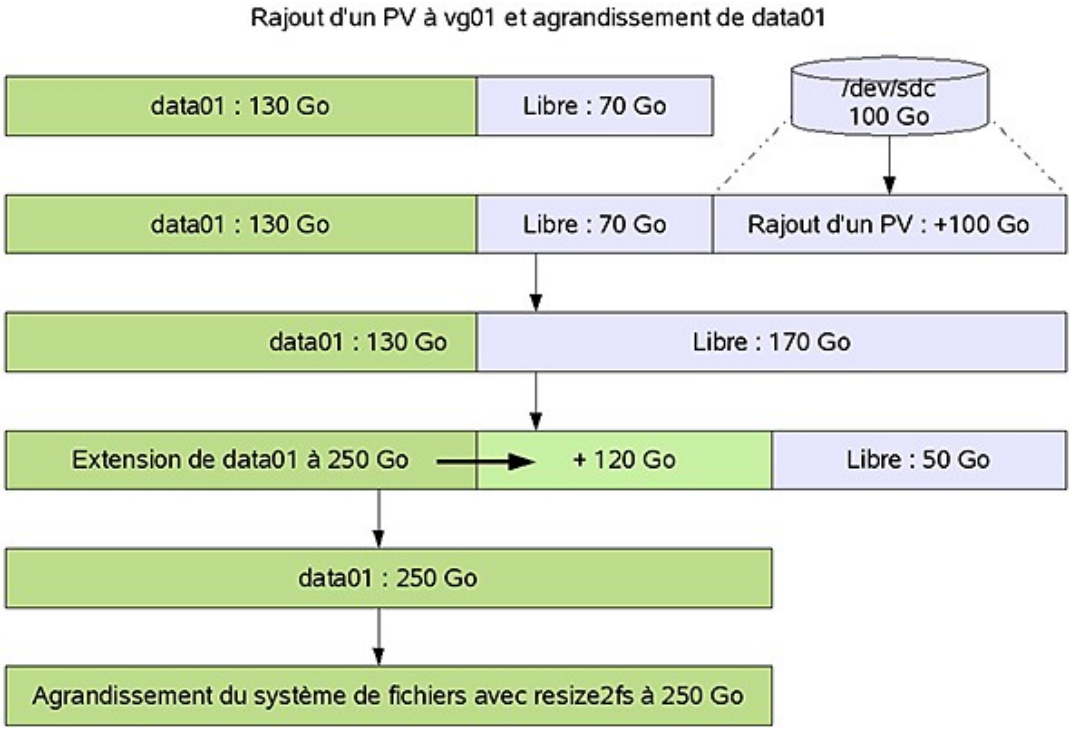
La commande ***resize2fs*** peut être **utilisée à chaud**, c'est-à-dire système de fichiers monté, pour les **agrandissements**. Il faudra par contre **démonter** le système de fichiers pour le **réduire**.

```
debian:/home/komo# resize2fs /dev/vg01/data01
...
Le système de fichiers /dev/vg01/data01 a maintenant une taille de 262144 blocs.

debian:/home/komo# df -h /mnt/data01
Sys. de fich.         Tail. Occ. Disp. %Occ. Monté sur
/dev/mapper/vg01-data01
                     1008M   17M  941M   2% /mnt/data01
```

b. Agrandir un volume logique

Vous voyez maintenant la puissance du LVM : rajout de volumes physiques et agrandissement de volumes logiques à la volée, de manière dynamique. Il n'y a plus de place ? Ce n'est pas grave : il suffit de rajouter un nouveau disque, le transformer en PV, l'ajouter dans le VG, et redimensionner le LV qui manque de place, sans avoir à repartitionner, recréer de système de fichiers, faire des backups, etc.



c. Réduire un volume logique

Pour réduire la taille d'un volume logique, vous devez **procéder dans cet ordre** :

- **Vérification du système de fichiers à réduire avec *fsck***.
- **Réduction du système de fichiers contenu dans le volume logique avec *resize2fs***.
- **Réduction du volume logique avec la commande *lvreduce***.

Vous allez **réduire le LV data01 à 512 Mo**. C'est uniquement **possible si ses données occupent moins de 512 Mo**. Dans un premier temps, vérifiez la taille actuelle du système de fichiers. Ici, il est quasiment vide :

```
debian:/home/komo# df -h /mnt/data01
Sys. de fich.          Tail. Occ. Disp. %Occ. Monté sur
/dev/mapper/vg01-data01
                      1008M   17M  941M   2% /mnt/data01
```

Le **système de fichiers** ne peut être réduit que s'il n'est pas monté ; **démontez-le** :

```
debian:/home/komo# umount /mnt/data01
```

c. Réduire un volume logique

Vérifiez le système de fichiers :

```
debian:/home/komo# fsck -f /dev/vg01/data01
fsck 1.41.3 (12-Oct-2008)
e2fsck 1.41.3 (12-Oct-2008)
Passe 1 : vérification des i-noeuds, des blocs et des tailles
Passe 2 : vérification de la structure des répertoires
Passe 3 : vérification de la connectivité des répertoires
Passe 4 : vérification des compteurs de référence
Passe 5 : vérification de l'information du sommaire de groupe

/dev/vg01/data01: ***** LE SYSTÈME DE FICHIERS A ÉTÉ MODIFIÉ *****
/dev/vg01/data01 : 11/65536 fichiers (0.0% non contigus), 8384/262144 blocs
```

Redimensionnez le système de fichiers à 512 Mo :

```
debian:/home/komo# resize2fs /dev/vg01/data01 512M
resize2fs 1.41.3 (12-Oct-2008)
Resizing the filesystem on /dev/vg01/data01 to 131072 (4k) blocks.
Le système de fichiers /dev/vg01/data01 a maintenant une taille de 131072 blocs.
```

Vérifiez la nouvelle taille du système de fichiers. 4096*131072 font bien 512 Mo.

```
debian:/home/komo# dumpe2fs -h /dev/vg01/data01 | grep ^Block
dumpe2fs 1.41.3 (12-Oct-2008)
Block count:              131072
Block size:               4096
Blocks per group:         32768
```

c. Réduire un volume logique

Enfin, **redimensionnez le LV à 512 Mo**. La **syntaxe** de ***lvreduce*** est la **même** que ***lvextend***, sauf qu'il n'est **pas possible de préciser de PV**. Veillez ici à **ne pas vous tromper** : si vous avez mal réduit le **système de fichiers**, vous **risquez de le détruire**. Répondez « y » à la question si vous êtes certain.

```
debian:/home/komo# lvreduce -L 512M /dev/vg01/data01
  WARNING: Reducing active logical volume to 512,00 MB
  THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce data01? [y/n]: y
  Reducing logical volume data01 to 512,00 MB
  Logical volume data01 successfully resized
```

Remontez le système de fichiers :

```
debian:/home/komo# mount /dev/vg01/data01 /mnt/data01
debian:/home/komo# df -h /mnt/data01
Sys. de fich.         Tail. Occ. Disp. %Occ. Monté sur
/dev/mapper/vg01-data01
                      504M   17M  462M   4% /mnt/data01
```

d. Déplacer le contenu d'un volume physique

Il est courant en entreprise de déplacer un PV vers un autre. Ce peut être dans le but de remplacer un disque contenant le PV par un autre (**pour agrandir par exemple**). Dans ce cas, vous pouvez déplacer le contenu d'un PV vers un autre, voire des PE d'un LV vers un autre PV, ou encore certains PE précis. Sans rien préciser comme destination, le LVM va déplacer tous les PE du PV dans les autres PV du groupe de volumes. **Attention : les volumes physiques doivent être dans le même groupe de volumes**.

La commande ***pvmove*** permet de **déplacer** les **PE d'un PV** vers un **autre**. Il s'agit ici pour vous de **déplacer** le contenu du PV **/dev/sde3 vers /dev/sde2**. /dev/sde3 contient 128 PE d'utilisés. Il contient tous les LE du LV data02.

```
debian:/home/komo# pvdisplay -m /dev/sde3
  --- Physical volume ---
  PV Name               /dev/sde3
  VG Name               vg01
  PV Size               1,10 GB / not usable 1,73 MB
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              280
  Free PE               152
  Allocated PE          128
  PV UUID               s5noVm-3rW1-pE6j-CYOC-Uy1I-blXb-8LxSvq

  --- Physical Segments ---
  Physical extent 0 to 127:
    Logical volume      /dev/vg01/data02
    Logical extents     0 to 127
  Physical extent 128 to 279:
    FREE
```

d. Déplacer le contenu d'un volume physique

Vérifiez si le volume physique ***/dev/sde2* dispose d'assez** de **place** pour **accueillir** le contenu de ***/dev/sde3***. Il reste 179 PE dans ce dernier, c'est donc possible.

```
debian:/home/komo# pvdisplay /dev/sde2
  --- Physical volume ---
  PV Name               /dev/sde2
  VG Name               vg01
  PV Size               972,69 MB / not usable 702,00 KB
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              243
  Free PE               179
  Allocated PE          64
  PV UUID               parBMc-aht9-B03A-VBJW-9Sz7-5feh-ibc1i9
```

d. Déplacer le contenu d'un volume physique

Déplacez le PV /dev/sde3 vers le PV /dev/sde2. Vous pouvez utiliser le paramètre *-v* pour suivre l'avancement. **Notez que l'opération s'effectue alors qu'aucun système de fichiers n'est démonté** :

```
debian:/home/komo# pvmove -v /dev/sde3 /dev/sde2
    Finding volume group "vg01"
    Archiving volume group "vg01" metadata (seqno 7).
    Creating logical volume pvmove0
    Moving 128 extents of logical volume vg01/data02
    Found volume group "vg01"
...
    Writing out final volume group after pvmove
    Creating volume group backup "/etc/lvm/backup/vg01" (seqno 10).
```

Vérifiez maintenant l'état du groupe de volumes :

Le second PV de vg01 est **entièrement libre**. Il est alors **maintenant possible de le supprimer du VG**.

```
debian:/home/komo# vgdisplay -v vg01 | grep -A 100 "Physical"
    Using volume group(s) on command line
    Finding volume group "vg01"
  --- Physical volumes ---
  PV Name               /dev/sde2
  PV UUID               parBMc-aht9-B03A-VBJW-9Sz7-5feh-ibc1i9
  PV Status             allocatable
  Total PE / Free PE    243 / 51

  PV Name               /dev/sde3
  PV UUID               s5noVm-3rW1-pE6j-CYOC-Uy1I-blXb-8LxSvq
  PV Status             allocatable
  Total PE / Free PE    280 / 280
```

e. Réduire un groupe de volumes

La commande ***vgreduce*** permet de **retirer** un ou plusieurs **PV d'un groupe de volumes**. Pour cela, **il faut** tout d'abord que les **PV** en question **soient vides** : leurs **PE** doivent être entièrement **libres**. **C'est le cas de /dev/sde3** que vous allez **retirer** du VG **vg01** :

```
debian:/home/komo# vgreduce vg01 /dev/sde3
  Removed "/dev/sde3" from volume group "vg01"
```

Contrôlez que le VG ne contient plus ce PV :

```
debian:/home/komo# vgdisplay -v vg01 | grep  -A 10 "Physical"
    Using volume group(s) on command line
    Finding volume group "vg01"
  --- Physical volumes ---
  PV Name               /dev/sde2
  PV UUID               parBMc-aht9-B03A-VBJW-9Sz7-5feh-ibc1i9
  PV Status             allocatable
  Total PE / Free PE    243 / 51
```

RAID et LVM

Initiation au LVM - Supprimer un groupe de volumes

a. Étapes

Pour **supprimer un groupe de volumes**, vous devez **suivre les étapes suivantes** :

- Démonter tous les systèmes de fichiers des LV associés.
- Supprimer tous les volumes logiques avec *lvremove*.
- Retirer tous les volumes physiques du VG avec *vgreduce*.
- Détruire le VG avec *vgremove*.

Vous allez détruire le groupe de volumes vg01.

b. Supprimer un volume logique

Démontez data01 et data02 :

```
debian:/home/komo# umount /mnt/data0{1,2}
```

Supprimez les volumes logiques avec *lvremove* :

```
debian:/home/komo# lvremove /dev/vg01/data0{1,2}
Do you really want to remove active logical volume "data01"? [y/n]: y
  Logical volume "data01" successfully removed
Do you really want to remove active logical volume "data02"? [y/n]: y
  Logical volume "data02" successfully removed
```

c. Retirer tous les volumes physiques

Utilisez la commande *vgreduce* avec le paramètre -a pour ALL :

```
debian:/home/komo# vgreduce -a vg01
  Can't remove final physical volume "/dev/sde2" from volume group "vg01"
```

Remarquez que la commande *vgreduce* laisse toujours au minimum un PV dans le VG, car il faut au moins un PV pour constituer un VG.

d. Détruire un groupe de volumes

Utilisez la commande *vgremove* pour détruire un groupe de volumes :

```
debian:/home/komo# vgremove vg01
  Volume group "vg01" successfully removed
```

Vérifiez que les fichiers et répertoires associés ont disparu :

```
debian:/home/komo# ls /dev/vg01
ls: ne peut accéder /dev/vg01: Aucun fichier ou répertoire de ce type
```

Enfin, la commande *vgdisplay* ne retourne plus rien :

```
debian:/home/komo# vgdisplay
```

e. Supprimer un volume physique

Les deux volumes physiques peuvent maintenant **être détruits** puisqu'ils ne sont plus utilisés. Vous pouvez détruire les informations contenues dans le volume **avec** la commande **pvremove**. Cependant, si vous détruisez la partition via **fdisk** ou que vous créez un système de fichiers dessus, l'effet est le même.

RAID et LVM

Initiation au LVM - Commandes supplémentaires

Vous n'avez eu qu'un bref aperçu des possibilités du LVM. De nombreuses autres commandes existent dont :

• *pvchange* : modifie l'état d'un volume physique, par exemple pour interdire l'allocation d'extensions physiques sur ce volume.
• *pvresize* : redimensionne un volume physique si sa partition ou disque d'origine a été agrandie ou réduite.
• *pvscan* : recherche tous les volumes physiques présents sur tous les supports de stockage du système.
• *vgchange* : modifie les attributs d'un groupe de volumes, pour l'activer ou le désactiver par exemple, mais aussi pour modifier les valeurs maximales de PV et de PE, ou pour interdire son agrandissement ou sa réduction.
• *vgscan* : recherche tous les groupes de volumes sur tous les supports.
• *vgrename* : renomme un groupe de volumes.
• *vgmerge* : regroupe deux groupes de volumes en un seul.
• *lvresize* : redimensionne un volume logique, équivaut tant à lvextend qu'à lvreduce.
• *lvchange* : modifie les attributs d'un volume logique.
• lvrename : renomme un volume logique.

RAID et LVM

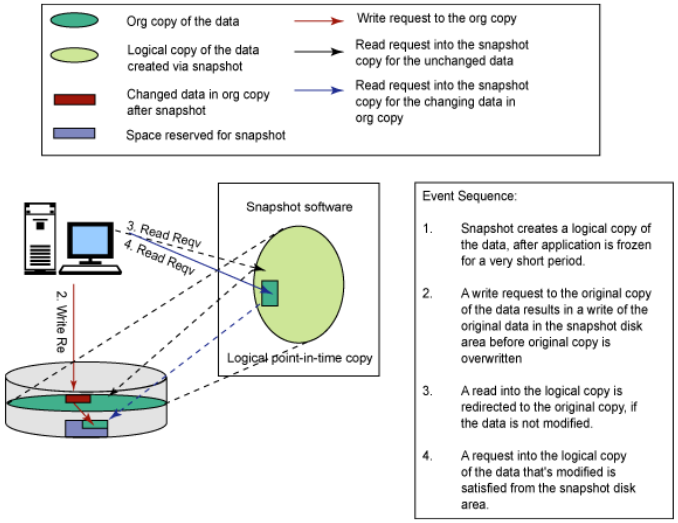
Initiation au LVM – Créer une image instantanée par Copy-on-Write d'un volume LVM

Copy-on-write

A snapshot of a storage volume is created using the pre-designated space for the snapshot. When the snapshot is first created, only the meta-data about where original data is stored is copied. No physical copy of the data is done at the time the snapshot is created. Therefore, the creation of the snapshot is almost instantaneous. The snapshot copy then tracks the changing blocks on the original volume as writes to the original volume are performed. The original data that is being written to is copied into the designated storage pool that is set aside for the snapshot before original data is overwritten, hence the name "copy-on-write".

Before a write is allowed to a block, copy-on-write moves the original data block to the snapshot storage. This keeps the snapshot data consistent with the exact time the snapshot was taken. Read requests to the snapshot volume of the unchanged data blocks are redirected to the original volume, while read requests to data blocks that have been changed are directed to the "copied" blocks in the snapshot. Snapshot contains the meta-data that describes the data blocks that have changed since the snapshot was first created. Note that original data blocks are copied only once into the snapshot storage when the first write request is received.

The following diagram illustrates a snapshot operation that creates a logical copy of the data using copy-on-write method.



Copy-on-write snapshot might initially impact performance on the original volume while it exists, because write requests to the original volume must wait while original data is being "copied out" to the snapshot. The read requests to snapshot are satisfied from the original volumes if data being read hasn't changed. However, this method is highly space efficient, because the storage required to create a snapshot is minimal to hold only the data that is changing. Additionally, the snapshot requires original copy of the data to be valid.

Suivre le lien

TODO

RAID et LVM

Validation des acquis : questions/réponses

1. Questions

Si l'état de vos connaissances sur ce chapitre vous semble suffisant, répondez aux questions ci-après.

RAID

1 Que signifie l'abréviation RAID ?
2 Soit un RAID 1 composé de trois disques ayant individuellement un MTBF de 30000 heures. Quel est le MTBF final ?
- A - 10000 heures
- B - 30000 heures
- C - 90000 heures

3 Un RAID 5 est composé de 4 disques de 100 Go. Quelle est la capacité totale de stockage ?
4 Deux disques sont en RAID0. Le second disque tombe en panne.
- A - Les données encore présentes sur le premier disque sont conservées.
- B - Les données sont réparties, donc tout est perdu.
- C - Les données sont présentes sur le premier disque, le second peut être reconstruit.

5 Un RAID5 est composé de trois disques ainsi que d'un disque de secours (spare). Deux disques tombent en panne, à quelque temps d'intervalle. Le RAID est-il encore fonctionnel ?
6 Quelle commande permet de manipuler du RAID logiciel ?
7 Est-il possible de créer un RAID 15 (1+5) de manière logicielle ?
8 La commande suivante a-t-elle un sens ?

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

9 Un cat `/proc/mdstat` indique ceci, que se passe-t-il ?

```
md0 : active raid1 hda10[2] hda9[0]F hda8[1]
```

10 Est-il normalement possible de changer un disque IDE à chaud ?

RAID et LVM

Validation des acquis : questions/réponses

LVM

11 Que signifie LVM ?
12 Un groupe de volumes peut-il être composé de volumes physiques de types différents ?
13 Quel est le type de partition à attribuer à un volume physique ?
•A - fd
•B - 83
•C - 82
•D - 8e
14 Soit trois volumes physiques de 30, 70 et 100 Go. Quelle est la taille maximale d'un volume logique taillé dans le groupe de volumes constitué de ces trois PV ?
15 Un groupe de volumes vgLOCAL est constitué de PE d'une taille de 64Mo. Quel est le paramètre à passer à la commande lvcreate pour créer un volume logique de 12 Go ?
16 On souhaite agrandir lv_L1 jusqu'à 20 Go. Or il n'y a plus de PE de libre dans le VG. Comment ajouter un volume physique /dev/sde de 20 Go ?
•A - vgextend -L +20G /dev/sde vgLOCAL
•B - pvextend /dev/sde vgLOCAL
•C - lvextend -l 320 /dev/vgLOCAL/lv_L1 /dev/sde
•D - vgextend vgLOCAL /dev/sde
17 Quelles syntaxes sont correctes pour augmenter à 20 Go le volume logique lv_L1 ?
•A - lvextend -L 20G /dev/vgLOCAL/lv_L1
•B - lvextend -l +128 /dev/vgLOCAL/lv_L1
•C - lvextend -l 320 /dev/vgLOCAL/lv_L1
•D - lvextend -L 20G /dev/vgLOCAL/lv_L1

RAID et LVM

Validation des acquis : questions/réponses

LVM

Malgré le passage à 20 Go de lv_L1, un df montre que la taille du système de fichiers est restée à 12 Go. Qu'avez-vous oublié ?

19 Pour réduire la taille d'un système de fichiers ext3, que devez-vous faire ?

20 Vous devez changer un PV et le remplacer par un autre. Quelle commande vous permet de déplacer tous les PE d'un PV vers un autre ?

RAID et LVM

Travaux pratiques

1. Gérer un RAID1

But : utiliser les commandes fdisk et mdadm pour gérer un RAID1

1. Sur votre disque dur, créez deux partitions de taille identique. La taille n'a pas d'importance. Donnez-leur le type fd, comme dans l'exemple suivant. Reportez-vous au chapitre Les disques et le système de fichiers pour l'utilisation de fdisk. Éventuellement utilisez la commande partprobe pour rafraîchir la table des partitions du noyau.

```
# fdisk /dev/sdb
Commande (m pour l'aide): p

Disque /dev/sdb: 160.0 Go, 160041885696 octets
255 heads, 63 sectors/track, 19457 cylinders
Units = cylindres of 16065 * 512 = 8225280 bytes
Disk identifier: 0x000eab03

Périphérique Amorce    Début         Fin      Blocs    Id  Système
/dev/sdb1               2       16846   135307462+  83  Linux
/dev/sdb2           16847       18152    10490445   fd  Linux raid
autodetect
/dev/sdb3           18153       19457    10482412+  fd  Linux raid
autodetect

# partprobe /dev/sdb
```

RAID et LVM

Travaux pratiques

1. Gérer un RAID1

2. Créez le RAID avec la commande mdadm. C'est la syntaxe courte qui est donnée ici :

```
# mdadm -C /dev/md0 -l 0 -n 2 /dev/sdb2 /dev/sdb3
mdadm: array /dev/md/0 started.
```

3. Vérifiez l'état du raid en regardant le contenu de /proc/mdstat :

```
# cat /proc/mdstat
Personalities : [raid0]
md0 : active raid0 sdb3[1] sdb2[0]
      20972672 blocks 64k chunks
```

RAID et LVM

Travaux pratiques

1. Gérer un RAID1

4. Comparez ce résultat avec celui de la commande mdstat pour un affichage détaillé. /proc/mdsta fournit un instantané de l'état du RAID vu du noyau. La sortie de mdstat fournit, outre l'état actuel, les informations sur la conception et le mode de fonctionnement du RAID :

```
slyserver:/home/seb # mdadm --detail /dev/md0
/dev/md0:
        Version : 0.90
  Creation Time : Wed May 13 20:16:49 2009
     Raid Level : raid0
     Array Size : 20972672 (20.00 GiB 21.48 GB)
   Raid Devices : 2
  Total Devices : 2
Preferred Minor : 0
    Persistence : Superblock is persistent

    Update Time : Wed May 13 20:16:49 2009
          State : clean
 Active Devices : 2
Working Devices : 2
 Failed Devices : 0
  Spare Devices : 0

     Chunk Size : 64K

           UUID : 3cac5942:d2777be8:1f7b0fb3:3386758f (local to host
slyserver)
         Events : 0.1

    Number   Major   Minor   RaidDevice State
       0       8       18        0      active sync   /dev/sdb2
       1       8       19        1      active sync   /dev/sdb3
unused devices: <none>
```

RAID et LVM

Travaux pratiques

1. Gérer un RAID1

5. Simulez une panne du second disque. Puis regardez l'état du RAID. Est-il encore en bon état ? Oui car le RAID 1 est un miroir, les données sont identiques sur les deux disques.

```
# mdadm /dev/md0 -f /dev/sdb3
mdadm: set /dev/sdb3 faulty in /dev/md0
# mdadm --detail /dev/md0|grep State
          State : clean
```

6. Stoppez le RAID et détruisez-le :

```
# mdadm --stop /dev/md0
mdadm: stopped /dev/md0
```

Pour détruire définitivement le RAID, il faut supprimer les informations des superblocs des disques composant le RAID. C'est là que sont écrites les informations sur le RAID. Ceci ce fait avec le paramètre --zero-superblock :

```
# mdadm --zero-superblock /dev/sdb2 /dev/sdb3
```

RAID et LVM

Travaux pratiques

2. Manipuler un LVM

But : Manipuler des PV, VG et LV.

1. Modifiez les partitions du TD précédent avec le type 8e :

```
/dev/sdb2          16847       18152    10490445   8e  Linux LVM
/dev/sdb3          18153       19457    10482412+  8e  Linux LVM
```

2. Créez deux PV avec ces deux partitions :

```
# pvcreate /dev/sdb2
  Physical volume "/dev/sdb2" successfully created
# pvcreate /dev/sdb3
  Physical volume "/dev/sdb3" successfully created
```

3. Créez un groupe de volumes vgLOCAL avec le premier PV :

```
slyserver:/home/seb # vgcreate vgLOCAL /dev/sdb2
  Volume group "vgLOCAL" successfully created
```

2. Manipuler un LVM

4. Vérifiez l'état du VG. Notamment, notez la taille d'un Physical Extend :

```
# vgdisplay vgLOCAL
  --- Volume group ---
  VG Name               vgLOCAL
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  1
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               10,00 GB
  PE Size               4,00 MB
  Total PE              2561
  Alloc PE / Size       0 / 0
  Free  PE / Size       2561 / 10,00 GB
  VG UUID               VYNM1X-D4al-PPZl-oD4t-7l5r-HyPP-SdZuDw
```

RAID et LVM

Travaux pratiques

2. Manipuler un LVM

5. Créez un volume logique lv_L1 de 8 Go (adaptez en fonction de la taille de votre VG), en utilisant les Logical extends. Ici il faut 2048 extends. Puis formatez le LV en ext3.

```
# lvcreate -n lv_L1 -l 2048 vgLOCAL
  Logical volume "lv_L1" created

# mkfs -t ext3 /dev/vgLOCAL/lv_L1
mke2fs 1.41.1 (01-Sep-2008)
…
Écriture des superblocs et de l'information de comptabilité du système de
fichiers : complété
```

6. Agrandissez lv_L1 à 15 Go. Pour cela, ajoutez le second PV dans le groupe de volumes puis agrandissez le LV et enfin agrandissez le système de fichiers.

```
# vgextend vgLOCAL /dev/sdb3
  Volume group "vgLOCAL" successfully extended
# lvextend -L +3G /dev/vgLOCAL/lv_L1
  Extending logical volume lv_L1 to 11,00 GB
  Logical volume lv_L1 successfully resized
# resize2fs /dev/vgLOCAL/lv_L1
resize2fs 1.41.1 (01-Sep-2008)
Resizing the filesystem on /dev/vgLOCAL/lv_L1 to 2883584 (4k) blocks.
Le système de fichiers /dev/vgLOCAL/lv_L1 a maintenant une taille de
2883584 blocs.
```

RAID et LVM

Travaux pratiques

2. Manipuler un LVM

7. Réduisez le volume logique lv_L1 à 5 Go. Pour cela, réduisez d'abord le système de fichiers à cette taille puis le volume logique. Vous devrez confirmer cette dernière opération car si vous oubliez de réduire la taille du système de fichiers elle peut être destructrice.

```
# resize2fs /dev/vgLOCAL/lv_L1 5G
resize2fs 1.41.1 (01-Sep-2008)
Resizing the filesystem on /dev/vgLOCAL/lv_L1 to 1310720 (4k) blocks.
Le système de fichiers /dev/vgLOCAL/lv_L1 a maintenant une taille de
1310720 blocs.

# lvreduce -L 5G /dev/vgLOCAL/lv_L1
  WARNING: Reducing active logical volume to 5,00 GB
  THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce lv_L1? [y/n]: y
  Reducing logical volume lv_L1 to 5,00 GB
  Logical volume lv_L1 successfully resized
```

8. Enfin, détruisez complètement le groupe de volumes. Commencez par détruire le volume logique lv_L1 et tous les volumes logiques éventuellement présents. Puis, retirez tous les volumes physiques du groupe de volumes, sauf un. Enfin, détruisez le groupe de volumes.

```
# lvremove /dev/vgLOCAL/lv_L1
Do you really want to remove active logical volume "lv_L1"? [y/n]: y
  Logical volume "lv_L1" successfully removed
# vgreduce vgLOCAL /dev/sdb3
  Removed "/dev/sdb3" from volume group "vgLOCAL"
# vgremove vgLOCAL
  Volume group "vgLOCAL" successfully removed
```