

LVM

Материал из Xgu.ru.

Повесть о Linux и LVM (Logical Volume Manager).

+ дополнения

Автор: Иван Песин

Автор: [Игорь Чубин](#) (автор дополнений)

В основу этой страницы положена работа «*Повесть о Linux и LVM*» Ивана Песина, которая, в свою очередь, написана на основе Linux LVM HOWTO. Она дополнена новыми ссылками и небольшими уточнениями, а также углублённым рассмотрением нескольких дополнительных вопросов.

Один из вопросов это использование **kpartx** из пакета **multipath-tools** для построения карты устройства (device map) и рекурсивного доступа к томам LVM (когда LVM развернут на разделах, созданных внутри логического тома LVM более низкого уровня). Это может быть полезно при использовании LVM совместно с системами виртуализации.

Второй вопрос --- это использование постоянных снимков (persistent snapshot) для быстрого *клонирования разделов*. Эта возможность может быть полезна как при выполнении резервного копирования, так и при быстром создании виртуальных машин в системах виртуализации (вопрос создания снапшотов затрагивался и в *повести*, но здесь он рассмотрен более детально).

Третий вопрос --- это *сравнение LVM и файловой системой ZFS*, набирающей в последнее время большую популярность. На первый взгляд такое сравнение может показаться странным, ведь ZFS -- это файловая система, а LVM -- система управления томами, то есть нечто, что находится на уровень ниже файловой системы. В действительности, сравнение вполне имеет право на существование, поскольку ZFS это не просто файловая система, а нечто большее. В ней присутствует уровень "storage pool", который берёт на себя те же задачи, что и LVM.

Введение

Цель статьи -- описать процесс установки и использования менеджера логических томов на Linux-системе. LVM (Logical Volume Manager), менеджер логических томов -- это система управления дисковым пространством, абстрагирующаяся от физических устройств. Она позволяет эффективно использовать и легко управлять дисковым пространством. LVM обладает хорошей масштабируемостью, уменьшает общую сложность системы. У логических томов, созданных с помощью LVM, можно легко изменить размер, а их названия могут нести большую смысловую нагрузку, в отличие от традиционных /dev/sda, /dev/hda ...

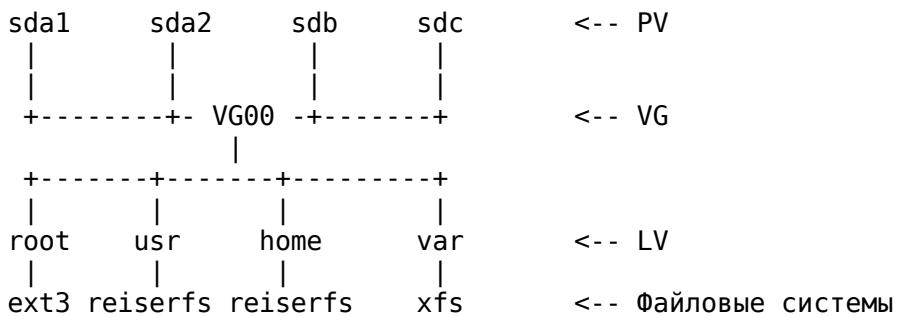
Реализации менеджеров логических томов существуют практически во всех UNIX-подобных операционных системах. Зачастую они сильно отличаются в реализации, но все они основаны на одинаковой идеи и преследуют аналогичные цели. Одна из

основных реализаций была выполнена Open Software Foundation (OSF) и сейчас входит в состав многих систем, например IBM AIX, DEC Tru64, HP/UX. Она же послужила и основой для Linux-реализации LVM.

Данная статья является переработкой и дополнением LVM-HOWTO.

Терминология.

Поскольку система управления логическими томами использует собственную модель представления дискового пространства, нам будет необходимо определиться с терминами и взаимосвязями понятий. Рассмотрим схему, основанную на диаграмме Эрика Бегфорса (Erik Begfors), приведенную им в списке рассылки linux-lvm. Она демонстрирует взаимосвязь понятий системы LVM:



Обозначения и понятия:

- *PV, Physical volume, физический том.* Обычно это раздел на диске или весь диск. В том числе, устройства программного и аппаратного RAID (которые уже могут включать в себя несколько физических дисков). Физические тома входят в состав группы томов.
- *VG, Volume group, группа томов.* Это самый верхний уровень абстрактной модели, используемой системой LVM. С одной стороны группа томов состоит из физических томов, с другой -- из логических и представляет собой единую административную единицу.
- *LV, Logical volume, логический том.* Раздел группы томов, эквивалентен разделу диска в не-LVM системе. Представляет собой блочное устройство и, как следствие, может содержать файловую систему.
- *PE, Physical extent, физический экстент.* Каждый физический том делится на порции данных, называющиеся физическими экстентами. Их размеры те же, что и у логических экстентов.
- *LE, Logical extent, логический экстент.* Каждый логический том делится на порции данных, называющиеся логическими экстентами. Размер логических экстентов не меняется в пределах группы томов.

Давайте теперь соединим все эти понятия в общую картину. Пусть у нас имеется группа томов VG00 с размером физического экстента 4Мб. В эту группу мы добавляем два раздела, /dev/hda1 и /dev/hdb1. Эти разделы становятся физическими томами, например PV1 и PV2 (символьные имена присваивает администратор, так что они могут быть более осмыслившими). Физические тома делятся на 4-х мегабайтные порции данных, т.к. это размер логического экстента. Диски имеют разный размер: PV1 получается размером в 99 экстентов, а PV2 --

размером в 248 экстентов. Теперь можно приступать к созданию логических томов, размером от 1 до 347 (248+99) экстентов. При создании логического тома, определяется отображение между логическими и физическими экстентами.

Например, логический экстент 1 может отображаться в физический экстент 51 тома PV1. В этом случае, данные, записанные в первые 4Мб логического экстента 1, будут в действительности записаны в 51-й экстент тома PV1.

Администратор может выбрать алгоритм отображения логических экстентов в физические. На данный момент доступны два алгоритма:

1. Линейное отображение последовательно назначает набор физических экстентов области логического тома, т.е. LE 1 - 99 отображаются на PV1, а LE 100 - 347 -- на PV2.

2. "Расслоенное" (striped) отображение разделяет порции данных логических экстентов на определенное количество физических томов. То есть:

- 1-я порция данных LE[1] -> PV1[1],
- 2-я порция данных LE[1] -> PV2[1],
- 3-я порция данных LE[1] -> PV3[1],
- 4-я порция данных LE[1] -> PV1[2], и т.д.

Похожая схема используется в работе RAID нулевого уровня. В некоторых ситуациях этот алгоритм отображения позволяет увеличить производительность логического тома. Однако он имеет значительное ограничение: логический том с данным отображением не может быть расширен за пределы физических томов, на которых он изначально и создавался.

Великолепная возможность, предоставляемая системой LVM -- это "снапшоты". Они позволяют администратору создавать новые блочные устройства с точной копией логического тома, "замороженного" в какой-то момент времени. Обычно это используется в пакетных режимах. Например, при создании резервной копии системы. Однако при этом вам не будет нужно останавливать работающие задачи, меняющие данные на файловой системе. Когда необходимые процедуры будут выполнены, системный администратор может просто удалить устройство-"снапшот". Ниже мы рассмотрим работу с таким устройством. Работа с LVM

Давайте теперь рассмотрим задачи, стоящие перед администратором LVM системы. Помните, что для работы с системой LVM ее нужно инициализировать командами:

```
%# vgscan  
%# vgchange -ay
```

Первая команда сканирует диски на предмет наличия групп томов, вторая активирует все найденные группы томов. Аналогично для завершения всех работ, связанных с LVM, нужно выполнить деактивацию групп:

```
%# vgchange -an
```

Первые две строки нужно будет поместить в скрипты автозагрузки (если их там нет), а последнюю можно дописать в скрипт shutdown.

Инициализация дисков и разделов

Перед использованием диска или раздела в качестве физического тома необходимо его инициализировать:

Для целого диска:

```
%# pvcreate /dev/hdb
```

Эта команда создает в начале диска дескриптор группы томов.

Если вы получили ошибку инициализации диска с таблицей разделов -- проверьте, что работаете именно с нужным диском, и когда полностью будете уверены в том, что делаете, выполните следующие команды

```
%# dd if=/dev/zero of=/dev/diskname bs=1k count=1  
%# blockdev --rereadpt /dev/diskname
```

Эти команды уничтожат таблицу разделов на целевом диске.

Для разделов:

Установите программой fdisk тип раздела в 0x8e.

```
%# pvcreate /dev/hdb1
```

Команда создаст в начале раздела /dev/hdb1 дескриптор группы томов.

Создание группы томов

Для создания группы томов используется команда 'vgcreate'

```
%# vgcreate vg00 /dev/hda1 /dev/hdb1
```

 Если вы используете devfs важно указывать полное имя в devfs, а не ссылку в каталоге /dev. Таким образом приведенная команда должна выглядеть в системе с devfs так:

```
# vgcreate vg00 /dev/ide/host0/bus0/target0/lun0/part1 /dev/ide/host0/bus0/target1/lun0/part1
```

Кроме того, вы можете задать размер экстента при помощи ключа "-s", если значение по умолчанию в 32Мб вас не устраивает. Можно, также, указать ограничения возможного количества физических и логических томов.

Активация группы томов

После перезагрузки системы или выполнения команды vgchange -a, ваши группы томов и логические тома находятся в неактивном состоянии. Для их активации необходимо выполнить команду

```
%# vgchange -a y vg00
```

Удаление группы томов

Убедитесь, что группа томов не содержит логических томов. Как это сделать, показано в следующих разделах.

Деактивируйте группу томов:

```
%# vgchange -a n vg00
```

Теперь можно удалить группу томов командой:

```
%# vgremove vg00
```

Добавление физических томов в группу томов

Для добавления предварительно инициализированного физического тома в существующую группу томов используется команда 'vgextend':

```
%# vgextend vg00 /dev/hdc1  
^~~~~~  
      новый физический том
```

Удаление физических томов из группы томов

Убедитесь, что физический том не используется никакими логическими томами.

Для этого используйте команду 'pvdisplay':

```
%# pvdisplay /dev/hda1
```

```
--- Physical volume ---  
PV Name          /dev/hda1  
VG Name          vg00  
PV Size          1.95 GB / NOT usable 4 MB [LVM: 122 KB]  
PV#              1  
PV Status        available  
Allocatable      yes (but full)  
Cur LV           1  
PE Size (KByte) 4096  
Total PE         499  
Free PE          0  
Allocated PE     499  
PV UUID          Sd44tK-9IRw-SrMC-M0kn-76iP-iftz-0VSen7
```

Если же физический том используется, вам нужно будет перенести данные на другой физический том. Эта процедура будет описана в следующих разделах.

После этого можно использовать 'vgreduce' для удаления физических томов:

```
%# vgreduce vg00 /dev/hda1
```

Создание логического тома

Для того, чтобы создать логический том "lv00", размером 1500Мб, выполните команду:

```
%# lvcreate -L1500 -n lv00 vg00
```

Для создания логического тома размером в 100 логических экстентов с расслоением по двум физическим томам и размером блока данных 4 KB:

```
%# lvcreate -i2 -I4 -l100 -n lv01 vg00
```

Если вы хотите создать логический том, полностью занимающий группу томов, выполните команду vgdisplay, чтобы узнать полные размеры группы томов, после чего используйте команду lvcreate.

```
%# vgdisplay vg00 | grep "Total PE"  
Total PE      10230  
%# lvcreate -l 10230 vg00 -n lv02
```

Эти команды создают логический том testvg, полностью заполняющий группу томов.

Удаление логических томов

Логический том должен быть размонтирован перед удалением:

```
%# umount /dev/vg00/home  
%# lvremove /dev/vg00/home  
lvremove -- do you really want to remove "/dev/vg00/home"? [y/n]: y  
lvremove -- doing automatic backup of volume group "vg00"  
lvremove -- logical volume "/dev/vg00/home" successfully removed
```

Увеличение логических томов

Для увеличения логического тома вам нужно просто указать команде lvextend до какого размера вы хотите увеличить том:

```
%# lvextend -L12G /dev/vg00/home  
lvextend -- extending logical volume "/dev/vg00/home" to 12 GB  
lvextend -- doing automatic backup of volume group "vg00"  
lvextend -- logical volume "/dev/vg00/home" successfully extended
```

В результате /dev/vg00/home увеличится до 12Гбайт.

```
%# lvextend -L+1G /dev/vg00/home  
lvextend -- extending logical volume "/dev/vg00/home" to 13 GB  
lvextend -- doing automatic backup of volume group "vg00"  
lvextend -- logical volume "/dev/vg00/home" successfully extended
```

Эта команда увеличивает размер логического тома на 1Гб.

После того как вы увеличили логический том, необходимо соответственно увеличить размер файловой системы. Как это сделать зависит от типа используемой файловой системы.

По умолчанию большинство утилит изменения размера файловой системы увеличивают ее размер до размера соответствующего логического тома. Так что вам не нужно беспокоиться об указании одинаковых размеров для всех команд.

ext2

Если вы не пропатчили ваше ядро патчем ext2online, вам будет необходимо размонтировать файловую систему перед изменением размера:

```
# umount /dev/vg00/home  
# resize2fs /dev/vg00/home  
# mount /dev/vg00/home /home
```

Если у вас нет пакета e2fsprogs 1.19 его можно загрузить с сайта

ext2resize.sourceforge.net.

Для файловой системы ext2 есть и другой путь. В состав LVM входит утилита e2fsadm, которая выполняет и lvextend, и resize2fs (она также выполняет и уменьшение размера файловой системы, это описано в следующем разделе). Так что можно использовать одну команду:

```
# e2fsadm -L+1G /dev/vg00/home
```

что эквивалентно двум следующим:

```
# extend -L+1G /dev/vg00/home
# resize2fs /dev/vg00/home
```

Замечание: вам все равно нужно будет размонтировать файловую систему перед выполнением e2fsadm.

reiserfs

Увеличивать размер файловых систем Reiserfs можно как в смонтированном, так и в размонтированном состоянии.

Увеличить размер смонтированной файловой системы:

```
# resize_reiserfs -f /dev/vg00/home
```

Увеличить размер размонтированной файловой системы:

```
# umount /dev/vg00/homevol
# resize_reiserfs /dev/vg00/homevol
# mount -t reiserfs /dev/vg00/homevol /home
```

xfs

Размер файловой системы XFS можно увеличить только в смонтированном состоянии. Кроме того, утилите в качестве параметра нужно передать точку монтирования, а не имя устройства:

```
# xfs_growfs /home
```

Уменьшение размера логического тома

Логические тома могут быть уменьшены в размере, точно также как и увеличены. Однако очень важно помнить, что нужно в первую очередь уменьшить размер файловой системы, и только после этого уменьшать размер логического тома. Если вы нарушите последовательность, вы можете потерять данные.

ext2

При использовании файловой системы ext2, как уже указывалось ранее, можно использовать команду e2fsadm:

```
# umount /home
# e2fsadm -L-1G /dev/vg00/home
# mount /home
```

Если вы хотите выполнить операцию по уменьшению логического тома вручную, вам нужно знать размер тома в блоках:

```
# umount /home
# resize2fs /dev/vg00/home 524288
# lvreduce -L-1G /dev/vg00/home
# mount /home
```

reiserfs

При уменьшении размера файловой системы Reiserfs, ее нужно размонтировать:

```
# umount /home
# resize_reiserfs -s-1G /dev/vg00/home
# lvreduce -L-1G /dev/vg00/home
# mount -treiserfs /dev/vg00/home /home
```

xfs

Уменьшить размер файловой системы XFS нельзя.

Примечание: обратите внимание на то, что для уменьшения размера файловых систем, необходимо их размонтировать. Это вносит определенные трудности, если вы желаете уменьшить размер корневой файловой системы. В этом случае можно применить следующий метод: загрузится с CD дистрибутива, поддерживающего LVM. Перейти в командный режим (обычно это делается нажатием клавиш Alt+F2) и выполнить команды сканирования и активации группы томов:

```
%# vgs
%# vgchange -a y
```

Теперь вы имеете доступ к логическим томам и можете изменять их размеры:

```
%# resize_reiserfs -s-500M /dev/vg00/root
%# lvreduce -L-500M /dev/vg00/root
%# reboot
```

Перенос данных с физического тома

Для того, чтобы можно было удалить физический том из группы томов, необходимо освободить все занятые на нем физические экстенты. Это делается путем перераспределения занятых физических экстентов на другие физические тома. Следовательно, в группе томов должно быть достаточно свободных физических экстентов. Описание операции удаления физического тома приведено в разделе примеров.

Примеры

Настройка LVM на трех SCSI дисках

В первом примере мы настроим логический том из трех SCSI дисков. Устройства дисков: /dev/sda, /dev/sdb и /dev/sdc.

Перед добавлением в группу томов диски нужно инициализировать:

```
%# pvcreate /dev/sda  
%# pvcreate /dev/sdb  
%# pvcreate /dev/sdc
```

После выполнения этих команд в начале каждого диска создастся область дескрипторов группы томов.

Теперь создадим группу томов vg01, состоящую из этих дисков:

```
%# vgcreate vg01 /dev/sda /dev/sdb /dev/sdc/
```

Проверим статус группы томов командой vgdisplay:

```
%# vgdisplay  
--- Volume Group ---  
VG Name          vg01  
VG Access        read/write  
VG Status        available/resizable  
VG #             1  
MAX LV           256  
Cur LV           0  
Open LV          0  
MAX LV Size     255.99 GB  
Max PV           256  
Cur PV           3  
Act PV           3  
VG Size          1.45 GB  
PE Size          4 MB  
Total PE         372  
Alloc PE / Size  0 / 0  
Free PE / Size   372 / 1.45 GB  
VG UUID          nP2PY5-5T0S-hLx0-FDu0-2a6N-f37x-0BME0Y
```

Обратите внимание на первые три строки и строку с общим размером группы томов. Она должна соответствовать сумме всех трех дисков. Если всё в порядке, можно переходить к следующей задаче.

Создание логического тома

После успешного создания группы томов, можно начать создавать логические тома в этой группе. Размер тома может быть любым, но, естественно, не более всего размера группы томов. В этом примере мы создадим один логический том размером 1 Гб. Мы не будем использовать "расслоение", поскольку при этом невозможно добавить диск в группу томов после создания логического тома, использующего данный алгоритм.

```
%# lvcreate -L1G -nusr1v vg01  
lvcreate -- doing automatic backup of "vg01"  
lvcreate -- logical volume "/dev/vg01/usr1v" successfully created
```

Создание файловой системы

Создадим на логическом томе файловую систему ext2:

```
%# mke2fs /dev/vg01/usrlv
mke2fs 1.19, 13-Jul-2000 for EXT2 FS 0.5b, 95/08/09
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
131072 inodes, 262144 blocks
13107 blocks (5.00%) reserved for the super user
First data block=0
9 block groups
32768 blocks per group, 32768 fragments per group
16384 inodes per group
Superblock backups stored on blocks:
32768, 98304, 163840, 229376

Writing inode tables: done
Writing superblocks and filesystem accounting information: done
```

Тестирование файловой системы

Смонтируйте логический том и проверьте все ли в порядке:

```
%# mount /dev/vg01/usrlv /mnt
%# df
Filesystem      1k-blocks  Used Available Use% Mounted on
/dev/hda1        1311552  628824    616104  51% /
/dev/vg01/usrlv  1040132     20    987276   0% /mnt
```

Если вы все сделали правильно, у вас должен появиться логический том с файловой системой ext2, смонтированный в точке /mnt.

Создание логического тома с "расслоением"

Рассмотрим теперь вариант логического тома, использующего алгоритм "расслоения". Как уже указывалось выше, минусом этого решения является невозможность добавления дополнительного диска.

Процедура создания данного типа логического тома также требует инициализации устройств и добавления их в группу томов, как это уже было показано.

Для создания логического тома с "расслоением" на три физических тома с блоком данных 4Кб выполните команду:

```
%# lvcreate -i3 -I4 -L1G -nvarlv vg01
lvcreate -- rounding 1048576 KB to stripe boundary size 1056768 KB / 258 PE
lvcreate -- doing automatic backup of "vg01"
lvcreate -- logical volume "/dev/vg01/varlv" successfully created
```

После чего можно создавать файловую систему на логическом томе.

Добавление нового диска

Рассмотрим систему со следующей конфигурацией:

```
%# pvscan
pvscan -- ACTIVE  PV "/dev/sda"  of VG "dev"  [1.95 GB / 0 free]
```

```

pvscan -- ACTIVE PV "/dev/sdb" of VG "sales" [1.95 GB / 0 free]
pvscan -- ACTIVE PV "/dev/sdc" of VG "ops" [1.95 GB / 44 MB free]
pvscan -- ACTIVE PV "/dev/sdd" of VG "dev" [1.95 GB / 0 free]
pvscan -- ACTIVE PV "/dev/sde1" of VG "ops" [996 MB / 52 MB free]
pvscan -- ACTIVE PV "/dev/sde2" of VG "sales" [996 MB / 944 MB free]
pvscan -- ACTIVE PV "/dev/sdf1" of VG "ops" [996 MB / 0 free]
pvscan -- ACTIVE PV "/dev/sdf2" of VG "dev" [996 MB / 72 MB free]
pvscan -- total: 8 [11.72 GB] / in use: 8 [11.72 GB] / in no VG: 0 [0]

```

```

%# df
Filesystem      1k-blocks   Used   Available  Use% Mounted on
/dev/dev/cvs    1342492    516468    757828  41% /mnt/dev/cvs
/dev/dev/users  2064208    2060036    4172 100% /mnt/dev/users
/dev/dev/build  1548144    1023041    525103  66% /mnt/dev/build
/dev/ops/databases 2890692    2302417    588275  79% /mnt/ops/databases
/dev/sales/users 2064208    871214   1192994  42% /mnt/sales/users
/dev/ops/batch  1032088    897122   134966  86% /mnt/ops/batch

```

Как видно из листинга, группы томов "dev" и "ops" практически заполнены. В систему добавили новый диск /dev/sdg. Его необходимо разделить между группами "ops" и "dev", поэтому разобьем его на разделы:

```

%# fdisk /dev/sdg
Device contains neither a valid DOS partition table, nor Sun or SGI
disklabel Building a new DOS disklabel. Changes will remain in memory
only, until you decide to write them. After that, of course, the
previous content won't be recoverable.

```

```

Command (m for help): n
Command action
  e  extended
  p  primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1000, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-1000, default 1000): 500

```

```

Command (m for help): n
Command action
  e  extended
  p  primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (501-1000, default 501):
Using default value 501
Last cylinder or +size or +sizeM or +sizeK (501-1000, default 1000):
Using default value 1000

```

```

Command (m for help): t
Partition number (1-4): 1
Hex code (type L to list codes): 8e
Changed system type of partition 1 to 8e (Unknown)

```

```

Command (m for help): t
Partition number (1-4): 2
Hex code (type L to list codes): 8e
Changed system type of partition 2 to 8e (Unknown)

```

```
Command (m for help): w
The partition table has been altered!
```

```
Calling ioctl() to re-read partition table.
```

```
WARNING: If you have created or modified any DOS 6.x partitions,
please see the fdisk manual page for additional information.
```

Перед тем как добавить разделы в группу томов, их необходимо инициализировать:

```
%# pvcreate /dev/sdg1
pvcreate -- physical volume "/dev/sdg1" successfully created
```

```
# pvcreate /dev/sdg2
pvcreate -- physical volume "/dev/sdg2" successfully created
```

Теперь можно добавлять физические тома в группы томов:

```
%# vgextend ops /dev/sdg1
vgextend -- INFO: maximum logical volume size is 255.99 Gigabyte
vgextend -- doing automatic backup of volume group "ops"
vgextend -- volume group "ops" successfully extended
```

```
# vgextend dev /dev/sdg2
vgextend -- INFO: maximum logical volume size is 255.99 Gigabyte
vgextend -- doing automatic backup of volume group "dev"
vgextend -- volume group "dev" successfully extended
```

```
# pvscan
pvscan -- reading all physical volumes (this may take a while...)
pvscan -- ACTIVE PV "/dev/sda" of VG "dev" [1.95 GB / 0 free]
pvscan -- ACTIVE PV "/dev/sdb" of VG "sales" [1.95 GB / 0 free]
pvscan -- ACTIVE PV "/dev/sdc" of VG "ops" [1.95 GB / 44 MB free]
pvscan -- ACTIVE PV "/dev/sdd" of VG "dev" [1.95 GB / 0 free]
pvscan -- ACTIVE PV "/dev/sde1" of VG "ops" [996 MB / 52 MB free]
pvscan -- ACTIVE PV "/dev/sde2" of VG "sales" [996 MB / 944 MB free]
pvscan -- ACTIVE PV "/dev/sdf1" of VG "ops" [996 MB / 0 free]
pvscan -- ACTIVE PV "/dev/sdf2" of VG "dev" [996 MB / 72 MB free]
pvscan -- ACTIVE PV "/dev/sdg1" of VG "ops" [996 MB / 996 MB free]
pvscan -- ACTIVE PV "/dev/sdg2" of VG "dev" [996 MB / 996 MB free]
pvscan -- total: 10 [13.67 GB] / in use: 10 [13.67 GB] / in no VG: 0 [0]
```

Наконец, увеличим размеры логических томов и расширим файловые системы до размеров логических томов:

```
%# umount /mnt/ops/batch
%# umount /mnt/dev/users

# export E2FSADM_RESIZE_CMD=ext2resize
# e2fsadm /dev/ops/batch -L+500M
e2fsck 1.18, 11-Nov-1999 for EXT2 FS 0.5b, 95/08/09
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/ops/batch: 11/131072 files (0.0<!-- non-contiguous), 4127/262144 blocks
lvextend -- extending logical volume "/dev/ops/batch" to 1.49 GB
lvextend -- doing automatic backup of volume group "ops"
```

```

lvextend -- logical volume "/dev/ops/batch" successfully extended

ext2resize v1.1.15 - 2000/08/08 for EXT2FS 0.5b
e2fsadm -- ext2fs in logical volume "/dev/ops/batch" successfully extended to
1.49 GB

# e2fsadm /dev/dev/users -L+900M
e2fsck 1.18, 11-Nov-1999 for EXT2 FS 0.5b, 95/08/09
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
/dev/dev/users: 12/262144 files (0.0% non-contiguous), 275245/524288 blocks
lvextend -- extending logical volume "/dev/dev/users" to 2.88 GB
lvextend -- doing automatic backup of volume group "dev"
lvextend -- logical volume "/dev/dev/users" successfully extended

ext2resize v1.1.15 - 2000/08/08 for EXT2FS 0.5b
e2fsadm -- ext2fs in logical volume "/dev/dev/users" successfully extended to
2.88 GB

```

Нам осталось смонтировать системы и посмотреть их размеры:

```

## mount /dev/ops/batch
## mount /dev/dev/users
## df
Filesystem      1k-blocks    Used Available Use% Mounted on
/dev/dev/cvs        1342492   516468    757828  41% /mnt/dev/cvs
/dev/dev/users       2969360  2060036   909324  69% /mnt/dev/users
/dev/dev/build       1548144  1023041   525103  66% /mnt/dev/build
/dev/ops/databases    2890692  2302417   588275  79% /mnt/ops/databases
/dev/sales/users      2064208   871214  1192994  42% /mnt/sales/users
/dev/ops/batch        1535856   897122   638734  58% /mnt/ops/batch

```

Резервное копирование при помощи "снапшотов"

Развивая приведенный пример, предположим, что нам нужно выполнить резервирование базы данных. Для этой задачи мы будем использовать устройство-"снапшот".

Этот тип устройства представляет собой доступную только на чтение копию другого тома на момент выполнения процедуры "снапшот". Это дает возможность продолжать работу не заботясь о том, что данные могут измениться в момент резервного копирования. Следовательно, нам не нужно останавливать работу базы данных на время выполнения резервного копирования. Останов нужен только на момент создания устройства-"снапшот", который значительно короче самого копирования.

В группе томов ops у нас осталось около 600Мб свободного места, его мы и задействуем для "снапшот"-устройства. Размер "снапшот"-устройства не регламентируется, но должен быть достаточен для сохранения всех изменений, которые могут произойти с томом, с которого он сделан, за время жизни снапшота. 600Мб должно хватить для наших целей:

```
%# lvcreate -L592M -s -n dbbackup /dev/ops/databases  
lvcreate -- WARNING: the snapshot must be disabled if it gets full  
lvcreate -- INFO: using default snapshot chunk size of 64 KB for  
"/dev/ops/dbbackup"  
lvcreate -- doing automatic backup of "ops"  
lvcreate -- logical volume "/dev/ops/dbbackup" successfully created
```

Если вы делаете "снапшот" файловой системы XFS, нужно выполнить на смонтированной файловой системе команду `xfs_freeze`, и лишь после этого создавать "снапшот":

```
%# xfs_freeze -f /mnt/point; lvcreate -L592M -s -n dbbackup /dev/ops/databases;  
xfs_freeze -u /mnt/point
```

Если устройство-"снапшот" полностью заполняется, оно автоматически деактивируется. В этом случае "снапшот" не может более использоваться, потому крайне важно выделять достаточное пространство для него.

После того как мы создали "снапшот", его нужно смонтировать:

```
%# mkdir /mnt/ops/dbbackup  
%# mount /dev/ops/dbbackup /mnt/ops/dbbackup  
mount: block device /dev/ops/dbbackup is write-protected, mounting read-only
```

Если вы работаете с файловой системой XFS, вам будет нужно при монтировании указать опцию `nouuid`:

```
%# mount -o nouuid,ro /dev/ops/dbbackup /mnt/ops/dbbackup
```

Выполним резервное копирование раздела:

```
%# tar -cf /dev/rmt0 /mnt/ops/dbbackup  
tar: Removing leading `/' from member names
```

После выполнения необходимых процедур, нужно удалить устройство-"снапшот":

```
%# umount /mnt/ops/dbbackup  
%# lvremove /dev/ops/dbbackup  
lvremove -- do you really want to remove "/dev/ops/dbbackup"? [y/n]: y  
lvremove -- doing automatic backup of volume group "ops"  
lvremove -- logical volume "/dev/ops/dbbackup" successfully removed
```

Подробнее:

- [Consistent backup with Linux Logical Volume Manager \(LVM\) snapshots](#) (англ.)
- [Back Up \(And Restore\) LVM Partitions With LVM Snapshots](#) (англ.)
- [Linux Kernel Documentation: device-mapper/snapshot.txt](#) (англ.)

Резервное копирование MySQL и LVM:

- [Using LVM for MySQL Backup and Replication Setup](#) (англ.)
- [MySQL Backups using LVM Snapshots](#) (англ.)
- [mylvmbackup](#) (англ.)

Удаление диска из группы томов

Скажем, вы хотите освободить один диск из группы томов. Для этого необходимо выполнить процедуру переноса использующихся физических экстентов. Естественно, что на других физических томах должно быть достаточно свободных физических экстентов.

Выполните команду:

```
%# pvmove /dev/hdb
pvmove -- moving physical extents in active volume group "dev"
pvmove -- WARNING: moving of active logical volumes may cause data loss!
pvmove -- do you want to continue? [y/n] y
pvmove -- 249 extents of physical volume "/dev/hdb" successfully moved
```

Учтите, что операция переноса физических экстентов занимает много времени. Если вы хотите наблюдать за процессом переноса экстентов, укажите в команде ключ -v .

После окончания процедуры переноса, удалите физический том из группы томов:

```
%# vgreduce dev /dev/hdb
vgreduce -- doing automatic backup of volume group "dev"
vgreduce -- volume group "dev" successfully reduced by physical volume:
vgreduce -- /dev/hdb
```

Теперь данный диск может быть физически удален из системы или использован в других целях. Например, добавлен в другую группу томов.

Перенос группы томов на другую систему

Физический перенос группы томов на другую систему организовывается при помощи команд vgexport и vgimport.

Сперва необходимо размонтировать все логические тома группы томов и деактивировать группу:

```
%# umount /mnt/design/users
%# vgchange -an design
vgchange -- volume group "design" successfully deactivated
```

После этого экспортируем группу томов. Процедура экспорта запрещает доступ к группе на данной системе и готовит ее к удалению:

```
%# vgexport design
vgexport -- volume group "design" sucessfully exported
```

Теперь можно выключить машину, отсоединить диски, составляющие группу томов и подключить их к новой системе. Остается импортировать группу томов на новой машине и смонтировать логические тома:

```
%# pvscan
pvscan -- reading all physical volumes (this may take a while...)
pvscan -- inactive PV "/dev/sdb1"  is in EXPORTED VG "design" [996 MB / 996 MB
free]
pvscan -- inactive PV "/dev/sdb2"  is in EXPORTED VG "design" [996 MB / 244 MB
free]
```

```
pvscan -- total: 2 [1.95 GB] / in use: 2 [1.95 GB] / in no VG: 0 [0]
# vgimport design /dev/sdb1 /dev/sdb2
vgimport -- doing automatic backup of volume group "design"
vgimport -- volume group "design" successfully imported and activated

%# mkdir -p /mnt/design/users
%# mount /dev/design/users /mnt/design/users
```

Все! Группа томов готова к использованию на новой системе.

Конвертация корневой файловой системы в LVM

В данном примере имеется установленная система на двух разделах: корневом и /boot. Диск размером 2Гб разбит на разделы следующим образом:

```
/dev/hda1 /boot
/dev/hda2 swap
/dev/hda3 /
```

Корневой раздел занимает все пространство, оставшееся после выделения swap и /boot разделов. Главное требование, предъявляемое к корневому разделу в нашем примере: он должен быть более чем на половину пуст. Это нужно, чтобы мы могли создать его копию. Если это не так, нужно будет использовать дополнительный диск. Процесс при этом останется тот же, но уменьшать корневой раздел будет не нужно.

Для изменения размера файловой системы мы будем использовать утилиту GNU parted.

Загрузитесь в однопользовательском режиме, это важно. Запустите программу parted для уменьшения размера корневого раздела. Ниже приведен пример диалога с утилитой parted:

```
# parted /dev/hda
(parted) p
.
.
.
```

Изменим размер раздела:

```
(parted) resize 3 145 999
```

Первое число -- это номер раздела (hda3), второе -- начало раздела hda3, не меняйте его. Последнее число -- это конец раздела. Укажите приблизительно половину текущего размера раздела.

Создадим новый раздел:

```
(parted) mkpart primary ext2 1000 1999
```

Этот раздел будет содержать LVM. Он должен начинаться после раздела hda3 и заканчиваться в конце диска.

Выходите из утилиты parted:

```
(parted) q
```

Перезагрузите систему. Убедитесь, что ваше ядро содержит необходимые установки. Для поддержки LVM должны быть включены параметры CONFIG_BLK_DEV_RAM и CONFIG_BLK_DEV_INITRD.

Для созданного раздела необходимо изменить тип на LVM (8e). Поскольку parted не знает такого типа, воспользуемся утилитой fdisk:

```
%# fdisk /dev/hda
Command (m for help): t
Partition number (1-4): 4
Hex code (type L to list codes): 8e
Changed system type of partition 4 to 8e (Unknown)
Command (m for help): w
```

Инициализируем LVM, физический том; создаем группу томов и логический том для корневого раздела:

```
%# vgscan
%# pvcreate /dev/hda4
%# vgcreate vg /dev/hda4
%# lvcreate -L250M -n root vg
```

Создадим теперь файловую систему на логическом томе и перенесем туда содержимое корневого каталога:

```
%# mke2fs /dev/vg/root
%# mount /dev/vg/root /mnt/
%# find / -xdev | cpio -pvmd /mnt
```

Отредактируйте файл /mnt/etc/fstab на логическом томе соответствующим образом. Например, строку:

```
/dev/hda3      /      ext2      defaults 1 1
```

замените на:

```
/dev/vg/root   /      ext2      defaults 1 1
```

Создаем образ initrd, поддерживающий LVM:

```
%# lvmcreate_initrd
Logical Volume Manager 1.0.6 by Heinz Mauelshagen 25/10/2002
lvmcreate_initrd -- make LVM initial ram disk /boot/initrd-lvm-2.4.20-inpl-up-
rootlvm.gz

lvmcreate_initrd -- finding required shared libraries
lvmcreate_initrd -- stripping shared libraries
lvmcreate_initrd -- calculating initrd filesystem parameters
lvmcreate_initrd -- calculating loopback file size
lvmcreate_initrd -- making loopback file (6491 kB)
lvmcreate_initrd -- making ram disk filesystem (19125 inodes)
lvmcreate_initrd -- mounting ram disk filesystem
lvmcreate_initrd -- creating new /etc/modules.conf
lvmcreate_initrd -- creating new modules.dep
lvmcreate_initrd -- copying device files to ram disk
lvmcreate_initrd -- copying initrd files to ram disk
lvmcreate_initrd -- copying shared libraries to ram disk
lvmcreate_initrd -- creating new /linuxrc
```

```
lvmcreate_initrd -- creating new /etc/fstab
lvmcreate_initrd -- ummounting ram disk
lvmcreate_initrd -- creating compressed initrd /boot/initrd-lvm-2.4.20-inpl-up-
rootlvm.gz
```

Внимательно изучите вывод команды. Обратите внимание на имя нового образа и его размер. Отредактируйте файл /etc/lilo.conf. Он должен выглядеть приблизительно следующим образом:

```
image    = /boot/KERNEL_IMAGE_NAME
label    = lvm
root     = /dev/vg/root
initrd   = /boot/INITRD_IMAGE_NAME
ramdisk = 8192
```

KERNEL_IMAGE_NAME -- имя ядра, поддерживающего LVM. INITRD_IMAGE_NAME -- имя образа initrd, созданного командой lvmcreate_initrd. Возможно, вам будет нужно увеличить значение ramdisk, если у вас достаточно большая конфигурация LVM, но значения 8192 должно хватить в большинстве случаев. Значение по умолчанию параметра ramdisk равно 4096. Если сомневаетесь, проверьте вывод команды lvmcreate_initrd в строке lvmcreate_initrd -- making loopback file (6189 kB).

После этого файл lilo.conf нужно скопировать и на логический том:

```
%# cp /etc/lilo.conf /mnt/etc/
```

Выполните команду lilo:

```
%# lilo
```

Перезагрузитесь и выберите образ lvm. Для этого введите "lvm" в ответ на приглашение LILO. Система должна загрузится, а корневой раздел будет находиться на логическом томе.

После того как вы убедитесь, что все работает нормально, образ lvm нужно сделать загружаемым по умолчанию. Для этого укажите в конфигурационном файле LILO строку default=lvm, и выполните команду lilo.

Наконец, добавьте оставшийся старый корневой раздел в группу томов. Для этого измените тип раздела утилитой fdisk на 8e, и выполните команды:

```
%# pvcreate /dev/hda3
%# vgextend vg /dev/hda3
```

Организация корневой файловой системы в LVM для дистрибутива ALT Master 2.2

При установке данного дистрибутива оказалось невозможным разместить корневой раздел в системе LVM. Связано это с тем, как выяснилось позже, что в ядре, поставляемом с данным дистрибутивом, поддержка файловой системы ext2 организована в виде загружаемого модуля. Образ же initrd использует файловую систему romfs, поддержка которой вкомпилирована в ядро. При выполнении команды lvmcreate_initrd генерируется файл-образ initrd с системой ext2. Если после этого вы попытаетесь загрузиться, то получите примерно следующее:

Kernel panic: VFS: Unable to mount root fs on 3a:00

Первое, что вам нужно будет сделать -- это скомпилировать ядро со встроенной поддержкой файловой системы ext2, установить его и проверить. После этого снова выполните команду lvmcreate_initrd. Дальнейшие действия зависят от вашей конфигурации. Смонтируйте созданный образ:

```
%# gzip -d /boot/initrd-lvm-2.4.20-inpl-up-rootlvm.gz  
%# mount -o loop /boot/initrd-lvm-2.4.20-inpl-up-rootlvm /mnt/initrd
```

И копируйте туда модули, необходимые для работы с вашими дисковыми накопителями и файловыми системами (если они не вкомпилированы в ядро). Так, для системы с RAID-контроллером ICP-Vortex и корневой файловой системой reiserfs нужны модули: gdth.o mod_scsi.o sd_mod.o reiserfs.o. Добавьте их загрузку в файл /mnt/initrd/linuxrc.

Обратите внимание на оставшееся свободное место на образе:

Filesystem	Size	Used	Avail	% Used	Mounted on	
	... /boot/initrd-lvm-2.4.20-inpl-up-rootlvm	4.0M	3.9M	0.1M	100%	/mnt/initrd

В файловой системе должно быть свободно еще 200-300Кб, в зависимости от вашей LVM-конфигурации. Если же у вас ситуация похожа на приведенную в листинге, будет необходимо создать новый образ, с большим размером файловой системы и повторить операции добавления модулей.

Наконец, отмонтируйте образ, сожмите его, запустите программу lilo и перезагрузитесь:

```
%# umount /mnt/initrd  
%# gzip /boot/initrd-lvm-2.4.20-inpl-up-rootlvm  
%# lilo  
%# reboot
```

Заключение

Система управления логическими томами особенно полезна в работе с серверами, поскольку обеспечивает масштабируемость и удобное управление дисковым пространством. Она упрощает планирование дискового пространства и предотвращает проблемы, возникающие при неожиданно быстром росте занятого места в разделах. LVM не предназначен для обеспечения отказоустойчивости или высокой производительности. Потому он часто используется в сочетании с системами RAID.

Использованные источники:

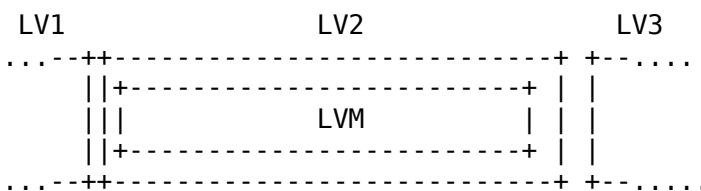
1. LVM-HOWTO
2. <http://www.gweep.net/~sfoskett/linux/lvmlinux.html>

Дополнительные вопросы по LVM

Дисковые разделы и LVM внутри LVM

LVM может находиться рекурсивно внутри логического тома LVM.

Например, пусть есть три логических тома: *LV1*, *LV2* и *LV3*. Один из которых, *LV2* разбит на разделы, каждый из которых является физическим томом LVM, который в свою очередь объединены в группу томов, на которых созданы логические тома и так далее.



Такая ситуация очень легко может возникнуть при использовании виртуальных машин, например в том случае, если том *LV2* выдан как диск фильтруальной машине [Xen](#) или эмулятору [QEMU](#).

Может возникнуть необходимость залезть внутрь системы томов, которые установлены внутрь логического тома. Например, это может произойти в случае, когда виртуальная машина (в результате сбоя или по какой-то другой причине), не загружается, а нужно добраться до её данных (напомним, что всё это при условии, что LVM используется не только снаружи, то есть в домене 0, но и внутри, то есть в домене U).

Решение, если говорить кратко, заключается в том, что использовать **kpartx** из пакета **multipath-tools**, которая даёт возможность строить карты устройств (device maps) для разделов.

Далее процедура использования **kpartx** описывается подробно.

Блоочное устройство, внутри которого находятся тома LVM может быть любым, в частности, это может быть том LVM, том EVMS, простой дисковый раздел или физическое устройство.

Убедитесь что пакет multipath-tools установлен:

```
%# apt-get install multipath-tools
```

Посмотрите как выполнено разбиение на разделы:

```
dom0:~ # fdisk -l /dev/evms/san2/vm3
Disk /dev/evms/san2/vm3: 5368 MB, 5368708096 bytes
255 heads, 63 sectors/track, 652 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

Device	Boot	Start	End	Blocks	Id	System
/dev/evms/san2/vm3p1	*	1	9	72261	83	Linux
/dev/evms/san2/vm3p2		10	652	5164897+	f	W95 Ext'd (LBA)
/dev/evms/san2/vm3p5		10	75	530113+	82	Linux swap / Solaris

```
/dev/evms/san2/vm3p6          76      493    3357553+  83  Linux  
/dev/evms/san2/vm3p7          494     652    1277136   83  Linux
```

Создайте карту устройств для блочного устройства:

```
dom0:~ # kpartx -a /dev/evms/san2/vm3
```

Карты находятся здесь:

```
%# ls -l /dev/mapper  
total 0  
lrwxrwxrwx 1 root root 16 Aug 26 16:28 control -> ../device-mapper  
brw----- 1 root root 253, 1 Aug 26 16:28 mpath1  
brw----- 1 root root 253, 0 Aug 26 16:28 mpath2  
brw----- 1 root root 253, 13 Aug 29 08:55 san2|vm3p1  
brw----- 1 root root 253, 14 Aug 29 08:55 san2|vm3p2  
brw----- 1 root root 253, 15 Aug 29 08:55 san2|vm3p5  
brw----- 1 root root 253, 16 Aug 29 08:55 san2|vm3p6  
brw----- 1 root root 253, 17 Aug 29 08:55 san2|vm3p7
```

Имена выглядят несколько странно, но вообще это нормально.

Можно попробовать смонтировать раздел:

```
%# mount -o rw /dev/mapper/san2\|vm3p6 /mnt
```

В данном случае была смонтирована корневая файловая система виртуальной машины:

```
%# ls -l /mnt  
total 96  
dr-xr-xr-x 3 root root 4096 Aug 28 10:12 automount  
drwxr-xr-x 2 root root 4096 Aug 25 16:51 bin  
drwxr-xr-x 2 root root 4096 Aug 25 16:45 boot  
drwxr-xr-x 5 root root 4096 Aug 25 16:45 dev  
drwxr-xr-x 69 root root 8192 Aug 29 08:53 etc  
drwxr-xr-x 2 root root 4096 May 4 01:43 home  
drwxr-xr-x 11 root root 4096 Aug 25 17:10 lib  
drwx----- 2 root root 16384 Aug 25 16:45 lost+found  
drwxr-xr-x 2 root root 4096 May 4 01:43 media  
drwxr-xr-x 3 root root 4096 Aug 28 15:08 mnt  
drwxr-xr-x 4 root root 4096 Aug 25 16:49 opt  
drwxr-xr-x 2 root root 4096 Aug 25 16:45 proc  
drwx----- 10 root root 4096 Aug 28 14:56 root  
drwxr-xr-x 3 root root 8192 Aug 25 16:52 sbin  
drwxr-xr-x 4 root root 4096 Aug 25 16:45 srv  
-rw-r--r-- 1 root root 0 Aug 29 08:53 success  
drwxr-xr-x 3 root root 4096 Aug 25 16:45 sys  
drwxrwxrwt 6 root root 4096 Aug 29 08:49 tmp  
drwxr-xr-x 12 root root 4096 Aug 25 16:50 usr  
drwxr-xr-x 3 root root 4096 Aug 25 16:54 var
```

Аналогичным образом монтируем остальные разделы:

```
%# mount -o rw /dev/mapper/san2\|vm3p1 /mnt/boot  
%# mount -o rw /dev/mapper/san2\|vm3p7 /mnt/var
```

С разделами можно работать как обычно.

По завершению работы нужно

- размонтировать разделы;
- удалить карты устройств.

Размонтировать разделы и удалить карту устройств особенно важно, если эти разделы принадлежат какой-либо виртуальной машине, которую вы собираетесь запускать.

```
%# umount /mnt/var  
%# umount /mnt/boot  
%# umount /mnt  
%# kpartx -d /dev/evms/san2/vm3  
  
%# ls -l /dev/mapper  
total 0  
lrwxrwxrwx 1 root root 16 Aug 26 16:28 control > ../device-mapper  
brw----- 1 root root 253, 1 Aug 26 16:28 mpath1  
brw----- 1 root root 253, 0 Aug 26 16:28 mpath2
```

При условии что разделы находятся не на блочном устройстве, а в обычно файле, сначала нужно воспользоваться программой *losetup*.

В данном примере *loop0* это первое свободное loopback-устройство, но возможно, что оно будет занято, и тогда вместо *loop0* нужно будет использовать *loopX* с более высоким номером X:

```
%# cd /etc/xen/images  
%# losetup /dev/loop0 /xen/images/vm1.img  
%# losetup -a  
/dev/loop0: [6806]:318859 (vm1.img)  
  
%# fdisk -l /dev/loop0  
Disk /dev/loop0: 1638 MB, 1638400000 bytes  
255 heads, 63 sectors/track, 199 cylinders  
Units = cylinders of 16065 * 512 = 8225280 bytes  
 Device Boot Start End Blocks Id System  
/dev/loop0p1 * 1 9 72261 83 Linux  
/dev/loop0p2 10 199 1526175 5 Extended  
/dev/loop0p5 10 26 136521 82 Linux swap / Solaris  
/dev/loop0p6 27 151 1004031 83 Linux  
/dev/loop0p7 152 199 385528+ 83 Linux  
  
%# kpartx -a /dev/loop0  
%# ls -l /dev/mapper/  
total 0  
lrwxrwxrwx 1 root root 16 Sep 12 15:38 control -> ../device-mapper  
brw----- 1 root root 253, 15 Sep 30 13:19 loop0p1  
brw----- 1 root root 253, 16 Sep 30 13:19 loop0p2  
brw----- 1 root root 253, 17 Sep 30 13:19 loop0p5  
brw----- 1 root root 253, 18 Sep 30 13:19 loop0p6  
brw----- 1 root root 253, 19 Sep 30 13:19 loop0p7
```

Можно монтировать, копировать, восстанавливать, форматировать эти разделы, короче, делать с ними всё, что делается с обычными дисковыми разделами.

```
%# mount /dev/mapper/loop0p6 /mnt
%# mount /dev/mapper/loop0p1 /mnt/boot
%# mount /dev/mapper/loop0p7 /mnt/var
%# mount
-snip-
/dev/mapper/loop0p6 on /mnt type ext3 (rw)
/dev/mapper/loop0p1 on /mnt/boot type ext2 (rw)
/dev/mapper/loop0p7 on /mnt/var type ext3 (rw)
```

Другое решение:

```
%# losetup -diskimage vm1.img -partition 1 /mnt
%# mount
-snip-
/xen/images/vm1.img on /mnt type ext2 (rw,loop=/dev/loop0,offset=32256)
```

С одной стороны при таком подходе:

- не нужно вручную подключать loopback-устройства;
- не нужно использовать *kpartx*;

Но с другой стороны:

- это решение позволяет смонтировать только простые разделы, размещённые внутри тома LVM.

Если внутри логического тома выполнено другое разбиение, такой подход не поможет.

Если внутри тома находится расширенный раздел, подход работает, но требует дополнительных манипуляций:

```
%# fdisk -l -u /xen/images/vm1.img
You must set cylinders.
You can do this from the extra functions menu.
Disk vm1.img: 0 MB, 0 bytes
255 heads, 63 sectors/track, 0 cylinders, total 0 sectors
Units = sectors of 1 * 512 = 512 bytes
   Device Boot   Start     End   Blocks   Id  System
vm1.img1   *       63    144584     72261   83  Linux
vm1.img2          144585   3196934    1526175     5  Extended
vm1.img5          144648   417689     136521   83  Linux
vm1.img6          417753   2425814    1004031   83  Linux
vm1.img7          2425878   3196934    385528+   83  Linux
```

Найти начало интересующего раздела можно путём умножения значения поля *Start* (показанного fdiks) на 512.

$417753 \times 512 = 213889536$

Если мы хотим смонтировать корневой раздел:

```
%# mount -o loop,offset=213889536 /xen/images/vm1.img /mnt
%# mount
-snip-
/xen/images/vm1.img on /mnt type ext3 (rw,loop=/dev/loop0,offset=213889536)
```

Сравнение LVM и ZFS

Основная страница: [ZFSvsLVM](#)

На первый взгляд такое сравнение может показаться странным, ведь ZFS --- это файловая система, а LVM --- система для управления томами, то есть нечто, что находится на уровень ниже файловой системы.

В действительности, сравнение вполне имеет право на существование, поскольку ZFS это не просто файловая система, а нечто большее. В ней присутствует уровень "storage pool", который берёт на себя те же задачи, что и LVM.

В таком случае возникает вопрос, а какую функциональность ZFS может дать сама, без применения LVM, и если что-то она делает лучше, то что именно?

Восстановления LVM после сбоя

- [Recover Data From RAID1 LVM Partitions With Knoppix Linux LiveCD](#) (англ.)
- [LVM Recovery Tale](#) (англ.)
- [Recovery of RAID and LVM2 Volumes](#) (англ.)

Дополнительная информация

- [LVM-HOWTO](#) (англ.)
- [LVM2 Resource Page](#) (англ.)
- [Повесть о Linux и LVM \(Logical Volume Manager\)](#), Иван Песин
- [LVM](#) в Википедии
- [LVM](#) в Wikipedia (англ.)
- [EVMS](#) в Википедии
- [EVMS](#) в Wikipedia (англ.)
- [Enterprise Volume Management System](#)

Xen и LVM:

- [\(Xen-users\) Xen with LVM](#) (англ.)
- [Mount/access Files Residing on Xen Virtual Machines](#) (англ.)

Материалы по хранению данных на xgu.ru

- [RAID](#)
- [LVM](#)
- [EVMS](#)

Файловые системы:

- [ext4](#)
- [Reiser4](#)
- [XFS](#)

- [ZFS](#)

Кластерные файловые системы:

- [GFS](#)
- [OCFS2](#)
- [Lustre](#)

Сетевые хранилища:

- [AoE](#)
- [GNBD](#)
- [iSCSI](#)

Реплицируемое блочное устройство:

- [DRBD](#)
- [Использование доменов Xen поверх DRBD-устройств](#)
- [Построение SAN для Xen на основе DRBD, LVM и GNBD](#)

Получено с <http://xgu.ru/wiki/LVM>

К этой странице обращались 2327 раз. Последнее изменение этой страницы: 17:42, 6 января 2008.