Вячеслав Калошин

Что такое хорошо, или Немного об LVM

С появлением LVM вам больше не надо ломать голову над тем, сколько места под какой раздел необходимо выделить. Теперь доступное дисковое пространство любого раздела вы можете изменять динамически и в реальном времени.

В последнее время в мире компьютерных технологий стало появляться все больше и больше различных аббревиатур. Почти для каждой комбинации букв придуманы невразумительные руководства, которые без применения исконно русских напитков читать практически невозможно. Примерно такая же ситуация сложилась и с LVM. Данная технология, позволяющая упростить жизнь при работе с дисками и разделами, оказалась незаслуженно отодвинутой в тень. Многие избегают ее использования по причине того, что все существующие руководства и прочая документация написаны слишком сухим языком. Но этим материалом мы попытаемся переломить общее отчуждение и поделиться с вами удобствами LVM.

Что же такое LVM

В переводе с английского LVM (Logical Volume Management) означает управление логическими томами. Многие из вас сразу же зададутся вопросом: «А зачем мне еще одна система управления логическими разделами? Мне и fdisk хватает. А на крайний случай есть parted». Однако все дело в том, что это не те логические разделы, к которым мы привыкли, а просто еще один уровень между физическими дисками и файловой подсистемой. Примерно такой же, как и RAID. Только RAID призван обеспечить большую сохранность данных, а LVM — удобство их размещения. Но количество преимуществ, которые дает LVM, несоизмеримо больше, чем абстрактное падение производительности, которое можно будет заметить только на синтетических тестах.

Чтобы не превращать статью в очередной пересказ документации, давайте сразу же попробуем оценить преимущества LVM на конкретной гипотетической системе. Во-первых, для этого не надо что-то куда-то переставлять или ломать в уже существующей системе. Во-вторых, сейчас во все современные дистрибутивы встраивается поддержка LVM, чем мы и воспользуемся.

Начало работы

Итак, представим, что у нас есть Linux-машина, которая прекрасно работает и не требует к себе пристального внимания. Но вот ведь незадача: современные программные средства имеют тенденцию увеличивать размер файлов до неприличия. Одним словом, на диске, присоединенном к /test, кончилось место:

```
df -h | grep test
/dev/sda1 241M 239M 0 100% /test
```

Самым первым порывом, который возникает у многих, будет мысль купить новый диск и поставить его на место старого. Или, что еще хуже, монтировать новый диск в один из каталогов на старом и перенести часть данных туда. Если в первом случае ничего страшного в принципе не случится (за исключением появления свободного диска, который тоже как-то надо использовать), то во втором через некоторое время начнется неразбериха с каталогами, обязательно завершающаяся тем, что на обоих дисках суммарно окажется много места, а файлы положить будет некуда. Давайте же поступим более разумно и попытаемся рационально использовать все доступное нам дисковое пространство. Мы намеренно опускаем здесь вопросы отказоустойчивости, потому что это прерогатива RAID.

Создание PV и VG

Для начала приведем наглядный пример, иллюстрирующий принцип работы LVM. Представьте себе обычную стеклянную банку, в которую положили несколько ненадутых воздушных шариков. Мы можем надуть один шарик так, чтобы он стал занимать абсолютно все доступное пространство внутри банки. А можем надуть два, чтобы они делили место поровну. А можем и три, только нужно чуть сильнее надуть один из шариков, чтобы он был больше. Если перевести этот при-

мер в термины LVM, то банка — это Physical Volume, или PV. В качестве PV может выступать что угодно, позволяющее записывать данные. Шарики маскируются под термином Logical Volume (LV), после чего на LV создаются файловые системы. Основное отличие PV от LV состоит в том, что у перового невозможно поменять объем. То же самое, возвращаясь к аналогии, — банку нельзя накачать.

Есть еще один термин — Volume Group (VG). В нашей баночно-шариковой классификации он соответствует обычной авоське, в которую можно положить сразу несколько банок. Но вернемся к компьютерам. У нас появился новый диск, и мы непременно желаем воспользоваться им в полном объеме. А раз так, значит, никаких разделов создавать на нем не будем — все отдадим под LVM:

```
pvcreate -v /dev/hde
Set up physical volume for "/dev/hde" with 2001504 available sectors
Physical volume "/dev/hde" successfully created
```

Итак, стеклянная банка возникла, и нам даже известен ее точный объем (1 Гбайт). Если у вас есть еще свободные диски, их можно проинициализировать таким же образом. Но у нас подобной возможности нет, поэтому давайте просто положим банку в авоську и прицепим к авоське бирку с надписью «test», чтобы не потерять:

```
vgcreate test /dev/hde
Volume group "test" successfully created
```

Создание LV

Теперь приподнимем авоську и посмотрим, что у нас в итоге получилось. Или, говоря правильными словами, активизируем LV для нашей системы:

```
vgchange -a y test
0 logical volume(s) in volume group "test" now active
```

Система сообщила, что она видит все кроме шариков. Банка пуста. Давайте исправим этот недостаток:

```
lvcreate -L768 -nstor test
Logical volume "stor" created
```

В данном примере мы создали LV размером 768 Мбайт и дали ему название «stor» (поместили шарик в банку и чуть-чуть накачали). Давайте посмотрим более внимательно, что у нас получилось. Сначала изучим банку:

```
pvdisplay /dev/hde
Physical volume
PV Name /dev/hde
VG Name test
PV Size 976.00 MB / not usable 0
Allocatable yes
PE Size (KByte) 4096
Total PE 244
Free PE 52
Allocated PE 192
PV UUID 477bCS-EKut-Ql2b-OulY-F67k-uHJv-Vny5He
```

А далее шарик:

```
lvdisplay test
Logical volume
LV Name /dev/test/stor
VG Name test
LV UUID yaKBSz-lkLP-6AwM-JAvR-V7C7-omW4-mmAzJV
LV Write Access read/write
LV Status available
LV Size 768.00 MB
Current LE 192
Segments 1
Allocation inherit
Read ahead sectors 0
Block device 253:2
```

Теперь нам все про них известно: и где размещаются, и как выглядят, и каков объем. Не будем останавливаться отдельно на каждом параметре, значения их всех весьма прозрачны. Не откладывая дело в долгий ящик, создадим в уже готовом разделе файловую систему:

```
mkfs.reiserfs /dev/test/stor
```

Как вы увидите из вывода mkfs, такой элемент для системы абсолютно ничем не отличается от обычного дискового устройства. Теперь создадим временную точку монтирования для нового устройства, после чего скопируем все данные со старого диска на новый:

```
mkdir /newtest
mount /dev/test/stor /newtest
cd /test
cp -R * /newtest
cd
umount /newtest
```

Дождемся момента, когда /test никому не будет нужен, и перемонтируем каталог /test на новое устройство:

```
umount /test
mount /dev/test/stor /test
rmdir /newtest
```

Посмотрим, что получилось:

```
mount | grep test && df -h | grep test
/dev/mapper/test-stor on /test type reiserfs (rw)
/dev/mapper/test-stor
768M 269M 500M 35% /test
```

Увеличение LV

Вроде все так, как мы и предполагали. В каталоге /test появилось свободное место, что нам и требовалось получить. Но вот наличие ничем не занятого пространства на диске (диск 1 Гбайт, а мы заняли 768 Мбайт) ничем не обосновано. Ведь изначально в планировалось создать там еще один LV, но ситуация изменилась, и лишние 300 Мбайт остались свободными. Имея дело с обычными разделами, мы бы опять ломали голову, как и куда можно пристроить лишний объем. В случае же с LVM поступим немного иначе. Для начала выясним, сколько раз надо двинуть поршнем насоса, чтобы шарик полностью заполнил банку:

```
vgdisplay test | grep Free
Free PE / Size 52 / 208.00 MB
```

Видно, что 208 Мбайт свободны. Теперь увеличиваем размер LV на это количество свободных мегабайт:

```
lvextend -L+208M /dev/test/stor
Extending logical volume stor to 976.00 MB
Logical volume stor successfully resized
```

Затем командуем файловой системе занять все свободное место и наслаждаемся полученным результатом:

```
resize_reiserfs -f /dev/test/stor
resize_reiserfs 3.6.18 (2003 www.namesys.com)
resize_reiserfs: On-line resizing finished successfully.

df -h | grep test
/dev/mapper/test-stor
976M 269M 708M 28% /test
```

Как видите, мы смогли спокойно увеличить размер каталога, не прерывая работы программ и вообще ничего не трогая. Правда, с EXT2/EXT3 пока что подобные трюки проводить рискованно, и разделы, использующие эту файловую систему, необходимо предварительно размонтировать.

Создание нового PV и объединение двух PV в один VG

Мы поставили новый диск, смогли увеличить его доступный объем на лету, однако и старый диск вполне может еще послужить. Давайте попробуем задействовать и его объем. Создадим на старом диске PV (достаем новую банку):

```
pvcreate /dev/sda
Physical volume "/dev/sda" successfully created
```

Добавим PV в VG (кладем банку в авоську):

```
vgextend test /dev/sda
Volume group "test" successfully extended
```

И дальше идем уже проверенным путем:

```
vgdisplay test | grep Free
Free PE / Size 61 / 244.00 MB

lvextend -L+244 /dev/test/stor
/dev/hde1: open failed: No such device or address
Extending logical volume stor to 1.19 GB
Logical volume stor successfully resized

resize_reiserfs -f /dev/test/stor
resize_reiserfs 3.6.18 (2003 www.namesys.com)
resize_reiserfs: On-line resizing finished successfully.

df -h | grep test
/dev/mapper/test-stor
1.2G  269M  952M  22% /test
```

Уменьшение LV и замена одного из PV

Таким образом, мы объединили пространство двух дисков, не растеряв в процессе ни одного мегабайта. Но, как нам всем отлично известно, жизнь не стоит на месте и постоянно движется вперед. Так же и в нашей ситуации: вдруг появляется большой и хороший жесткий диск, которым можно было бы заменить один из уже используемых в системе. На роль жертвы нами был выбран /dev/sda.

К сожалению, чтобы освободить диск, необходимо убрать с него LV (чтобы выкинуть банку, надо вынуть шарики из нее). А ведь LV полностью занят файловой системой. Значит, надо уменьшить объем раздела. К сожалению, такие операции возможны только на несмонтированных файловых системах, да и то не на всех:

```
umount /test

resize_reiserfs -s-300M /dev/test/stor
resize_reiserfs: Resizing finished successfully.

mount /dev/test/stor /test

df -h|grep test
/dev/mapper/test-stor
920M 269M 652M 30% /test
```

Размер раздела уменьшился. Теперь уменьшаем размер LV (выпускаем воздух из шариков). Лучше всего производить все манипуляции на отключенном разделе, но в крайнем случае можно сделать это и на лету. Только не забудьте создать резервную копию критически важных данных и проверить запас энергии в аккумуляторах UPS:

```
lvreduce -L-300M /dev/test/stor
WARNING: Reducing active and open logical volume to 920.00 MB
THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce stor? [y/n]: y
Reducing logical volume stor to 920.00 MB
Logical volume stor successfully resized
```

Теперь выводим диск из обращения (перекладываем шарики в другие банки и выкидываем ненужную):

```
vgreduce -v test /dev/sda
Finding volume group "test"
Using physical volume(s) on command line
Archiving volume group "test" metadata.
Removing "/dev/sda" from volume group "test"
Creating volume group backup "/etc/LVM/backup/test"
Removed "/dev/sda" from volume group "test"
```

На самом деле все так легко получилось благодаря тому, что никаких файлов на /dev/sda физически не оказалось. В реальной ситуации (которая описывается ниже) необходимо воспользоваться pvmove. Итак, еще раз проверяем, что все в порядке и диск больше ничем не занят:

```
pvscan | grep sda
PV /dev/sda LVM2 [244.00 MB]
```

Как и ожидалось, он полностью пустой. Вынимаем и заменяем его новым, повторяя уже описанные операции. Теперь же мы перенесем на новый диск вообще все данные:

```
pvcreate /dev/sda
Physical volume "/dev/sda" successfully created

vgextend test /dev/sda
Volume group "test" successfully extended

lvextend -L+2G /dev/test/stor
Extending logical volume stor to 2.90 GB
Logical volume stor successfully resized
```

У нас появилось много свободного места. Скомандуем перенести все данные с /dev/hde на другие диски, входящие в VG:

```
pvmove -v /dev/hde
Finding volume group "test"
Archiving volume group "test" metadata.
Creating logical volume pvmove0
Moving 244 extents of logical volume test/stor
Found volume group "test"
Updating volume group metadata
Creating volume group backup "/etc/LVM/backup/test"
Found volume group "test"
Loading test-pvmove0
Found volume group "test"
Loading test-stor
Checking progress every 15 seconds
Found volume group "test"
Loading test-pvmove0
Found volume group "test"
Found volume group "test"
Loading test-stor
Found volume group "test"
Removing temporary pvmove LV
Writing out final volume group after pvmove
Creating volume group backup "/etc/LVM/backup/test"
```

Теперь мы можем окончательно удалить диск из VG:

```
vgreduce test /dev/hde
Removed "/dev/hde" from volume group "test"
```

Не правда ли, здорово? Наши данные сменили место жительства три раза, и столько же раз изменялся размер раздела. И все это время (не считая того момента, когда мы уменьшали раздел) они были доступны для использования. |

Дополнительные возможности

Создание резервных копий

Однако на этом возможности LVM не заканчиваются. Рассмотрим процесс создания резервных копий данных. В современных системах очень редко можно найти промежуток времени, удовлетворяющий двум условиям: его должно хватить для проведения резервного копирования, и к тому же резервируемые файлы не должны изменяться. К примеру, на веб-серверах такими «нехорошими» являются файлы, куда записываются данные о посещениях сайта. С помощью LVM вполне можно убрать второе условие. Для начала создадим специальный LV, который зафиксирует состояние нужного нам раздела, или, говоря терминами LVM, создаст снапшот. Размер, который мы указываем при создании LV, будет использован для записи изменений на разделе, которые происходят во время резервного копирования. Если вдруг за время резервного копирования размер изменений превысит заданный, то наш раздел со снапшотом просто отключится:

```
lvcreate -L1000M -s -nstorbackup /dev/test/stor
Logical volume "storbackup" created
```

Теперь мы можем спокойно создать (если ее еще нет) точку монтирования и приступить к выполнению резервного копирования раздела, совершенно не опасаясь получить в итоговой копии не те файлы:

```
mkdir /storbackup
mount /dev/test/storbackup /storbackup/
```

После этого давайте проверим, коснутся ли созданной копии раздела изменения на диске:

```
ls /storbackup/
lost+found  Photo

mkdir /test/testsnap

ls /storbackup/
lost+found  Photo
```

Очевидно, что тот раздел, с которого происходит бэкап, совершенно не изменился. Как только резервное копирование будет выполнено, можно спокойно удалять LV со снапшотом:

```
umount /storbackup/

lvremove /dev/test/storbackup
Do you really want to remove active logical volume "storbackup"? [y/n]: y
Logical volume "storbackup" successfully removed
```