By Falko Timme
Published: 2007-01-14 18:51

A Beginner's Guide To LVM

Version 1.0
Author: Falko Timme <ft [at] falkotimme [dot] com>
Last edited 01/08/2007

This guide shows how to work with LVM (Logical Volume Management) on Linux. It also describes how to use LVM together with RAID1 in an extra chapter. As LVM is a rather abstract topic, this article comes with a Debian Etch VMware image that you can download and start, and on that Debian Etch system you can run all the commands I execute here and compare your results with mine. Through this practical approach you should get used to LVM very fast.

However, I do not issue any guarantee that this tutorial will work for you!

1 Preliminary Note

This tutorial was inspired by two articles I read:

- **http://www.linuxdevcenter.com/pub/a/linux/2006/04/27/managing-disk-space-with-lvm.html**
- **http://www.debian-administration.org/articles/410**

These are great articles, but hard to understand if you've never worked with LVM before. That's why I have created **this Debian Etch VMware image** that you can download and run in VMware Server or VMware Player (see **http://www.howtoforge.com/import_vmware_images** to learn how to do that).

I installed all tools we need during the course of this guide on the Debian Etch system (by running

```
apt-get install lvm2 dmsetup mdadm reiserfsprogs xfsprogs
```

) so you don't need to worry about that.

The Debian Etch system's network is configured through DHCP, so you don't have to worry about conflicting IP addresses. The root password is *howtoforge*. You can also connect to that system with an SSH client like **PuTTY**. To find out the IP address of the Debian Etch system, run

```
ifconfig
```

The system has six SCSI hard disks, */dev/sda* - */dev/sdf*. */dev/sda* is used for the Debian Etch system itself, while we will use */dev/sdb* - */dev/sdf* for LVM and RAID. */dev/sdb* - */dev/sdf* each have 80GB of disk space. In the beginning we will act as if each has only 25GB of disk space (thus using only 25GB on each of them), and in the course of the tutorial we will "replace" our 25GB hard disks with 80GB hard disks, thus demonstrating how you can replace small hard disks with bigger ones in LVM.

The article **http://www.linuxdevcenter.com/pub/a/linux/2006/04/27/managing-disk-space-with-lvm.html** uses hard disks of 250GB and 800GB, but some commands such as *pvmove* take a long time with such hard disk sizes, that's why I decided to use hard disks of 25GB and 80GB (that's enough to understand how LVM works).

1.1 Summary

Download **this Debian Etch VMware image** (~310MB) and start it **like this**. Log in as root with the password *howtoforge*.

2 LVM Layout

Basically LVM looks like this:

You have one or more physical volumes (*/dev/sdb1* - */dev/sde1* in our example), and on these physical volumes you create one or more volume groups (e.g. *fileserver*), and in each volume group you can create one or more logical volumes. If you use multiple physical volumes, each logical volume can be bigger than one of the underlying physical volumes (but of course the sum of the logical volumes cannot exceed the total space offered by the physical volumes).

It is a good practice to not allocate the full space to logical volumes, but leave some space unused. That way you can enlarge one or more logical volumes later on if you feel the need for it.

In this example we will create a volume group called *fileserver*, and we will also create the logical volumes */dev/fileserver/share*, */dev/fileserver/backup*, and */dev/fileserver/media* (which will use only half of the space offered by our physical volumes for now - that way we can switch to RAID1 later on (also described in this tutorial)).

3 Our First LVM Setup

Let's find out about our hard disks:

```
fdisk -l
```

The output looks like this:

```
server1:~# fdisk -l

Disk /dev/sda: 21.4 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

```
   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          18      144553+  83  Linux
/dev/sda2              19        2450    19535040   83  Linux
/dev/sda4            2451        2610     1285200   82  Linux swap / Solaris

Disk /dev/sdb: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/sdb doesn't contain a valid partition table

Disk /dev/sdc: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/sdc doesn't contain a valid partition table

Disk /dev/sdd: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/sdd doesn't contain a valid partition table

Disk /dev/sde: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/sde doesn't contain a valid partition table

Disk /dev/sdf: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

```
Disk /dev/sdf doesn't contain a valid partition table
```

There are no partitions yet on /dev/sdb - /dev/sdf. We will create the partitions /dev/sdb1, /dev/sdc1, /dev/sdd1, and /dev/sde1 and leave /dev/sdf untouched for now. We act as if our hard disks had only 25GB of space instead of 80GB for now, therefore we assign 25GB to /dev/sdb1, /dev/sdc1, /dev/sdd1, and /dev/sde1:

```
fdisk /dev/sdb
```

```
server1:~# fdisk /dev/sdb

  The number of cylinders for this disk is set to 10443.
  There is nothing wrong with that, but this is larger than 1024,
  and could in certain setups cause problems with:
  1) software that runs at boot time (e.g., old versions of LILO)
  2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):
Command action
   a   toggle a bootable flag
   b   edit bsd disklabel
   c   toggle the dos compatibility flag
   d   delete a partition
   l   list known partition types
   m   print this menu
   n   add a new partition
   o   create a new empty DOS partition table
   p   print the partition table
   q   quit without saving changes
   s   create a new empty Sun disklabel
   t   change a partition's system id
```

```
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit
   x   extra functionality (experts only)

Command (m for help):
Command action
   e   extended
   p   primary partition (1-4)

Partition number (1-4):
First cylinder (1-10443, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-10443, default 10443):

Command (m for help):
Selected partition 1
Hex code (type L to list codes):

 0  Empty           1e  Hidden W95 FAT1 80  Old Minix       be  Solaris boot
 1  FAT12           24  NEC DOS         81  Minix / old Lin bf  Solaris
 2  XENIX root      39  Plan 9          82  Linux swap / So c1  DRDOS/sec (FAT-
 3  XENIX usr       3c  PartitionMagic  83  Linux           c4  DRDOS/sec (FAT-
 4  FAT16 <32M      40  Venix 80286     84  OS/2 hidden C:  c6  DRDOS/sec (FAT-
 5  Extended        41  PPC PReP Boot   85  Linux extended  c7  Syrinx
 6  FAT16           42  SFS             86  NTFS volume set da  Non-FS data
 7  HPFS/NTFS       4d  QNX4.x          87  NTFS volume set db  CP/M / CTOS / .
 8  AIX             4e  QNX4.x 2nd part 88  Linux plaintext de  Dell Utility
 9  AIX bootable    4f  QNX4.x 3rd part 8e  Linux LVM       df  BootIt
 a  OS/2 Boot Manag 50  OnTrack DM      93  Amoeba          e1  DOS access
 b  W95 FAT32       51  OnTrack DM6 Aux 94  Amoeba BBT      e3  DOS R/O
 c  W95 FAT32 (LBA) 52  CP/M            9f  BSD/OS          e4  SpeedStor
 e  W95 FAT16 (LBA) 53  OnTrack DM6 Aux a0  IBM Thinkpad hi eb  BeOS fs
```

```
 f  W95 Ext'd (LBA) 54  OnTrackDM6      a5  FreeBSD         ee  EFI GPT
10  OPUS            55  EZ-Drive        a6  OpenBSD         ef  EFI (FAT-12/16/
11  Hidden FAT12    56  Golden Bow      a7  NeXTSTEP        f0  Linux/PA-RISC b
12  Compaq diagnost 5c  Priam Edisk     a8  Darwin UFS      f1  SpeedStor
14  Hidden FAT16 <3 61  SpeedStor       a9  NetBSD          f4  SpeedStor
16  Hidden FAT16    63  GNU HURD or Sys ab  Darwin boot     f2  DOS secondary
17  Hidden HPFS/NTF 64  Novell Netware  b7  BSDI fs         fd  Linux raid auto
18  AST SmartSleep  65  Novell Netware  b8  BSDI swap       fe  LANstep
1b  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid ff  BBT
1c  Hidden W95 FAT3 75  PC/IX
Hex code (type L to list codes):
Changed system type of partition 1 to 8e (Linux LVM)

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

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

Now we do the same for the hard disks */dev/sdc* - */dev/sde*:

```
fdisk /dev/sdc

fdisk /dev/sdd

fdisk /dev/sde
```

Then run

```
fdisk -l
```

again. The output should look like this:

```
server1:~# fdisk -l

Disk /dev/sda: 21.4 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          18      144553+  83  Linux
/dev/sda2              19        2450    19535040   83  Linux
/dev/sda4            2451        2610     1285200   82  Linux swap / Solaris

Disk /dev/sdb: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1        3040    24418768+  8e  Linux LVM

Disk /dev/sdc: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1               1        3040    24418768+  8e  Linux LVM

Disk /dev/sdd: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdd1               1        3040    24418768+  8e  Linux LVM
```

```
Disk /dev/sde: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sde1               1        3040    24418768+  8e  Linux LVM

Disk /dev/sdf: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/sdf doesn't contain a valid partition table
```

Now we prepare our new partitions for LVM:

```
pvcreate /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
```

```
server1:~# pvcreate /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
  Physical volume "/dev/sdb1" successfully created
  Physical volume "/dev/sdc1" successfully created
  Physical volume "/dev/sdd1" successfully created
  Physical volume "/dev/sde1" successfully created
```

Let's revert this last action for training purposes:

```
pvremove /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
```

```
server1:~# pvremove /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
  Labels on physical volume "/dev/sdb1" successfully wiped
```

```
  Labels on physical volume "/dev/sdc1" successfully wiped
  Labels on physical volume "/dev/sdd1" successfully wiped
  Labels on physical volume "/dev/sde1" successfully wiped
```

Then run

```
pvcreate /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
```

again:

```
server1:~# pvcreate /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
  Physical volume "/dev/sdb1" successfully created
  Physical volume "/dev/sdc1" successfully created
  Physical volume "/dev/sdd1" successfully created
  Physical volume "/dev/sde1" successfully created
```

Now run

```
pvdisplay
```

to learn about the current state of your physical volumes:

```
server1:~# pvdisplay
  --- NEW Physical volume ---
  PV Name               /dev/sdb1
  VG Name
  PV Size               23.29 GB
  Allocatable           NO
  PE Size (KByte)       0
```

```
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               G8lu2L-Hij1-NVde-sOKc-OoVI-fadg-Jd1vyU

  --- NEW Physical volume ---
  PV Name               /dev/sdc1
  VG Name
  PV Size               23.29 GB
  Allocatable           NO
  PE Size (KByte)       0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               40GJyh-IbsI-pzhn-TDRq-PQ3l-3ut0-AVSE4B

  --- NEW Physical volume ---
  PV Name               /dev/sdd1
  VG Name
  PV Size               23.29 GB
  Allocatable           NO
  PE Size (KByte)       0
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               4mU63D-4s26-uL00-r0pO-Q0hP-mvQR-2YJN5B

  --- NEW Physical volume ---
  PV Name               /dev/sde1
  VG Name
  PV Size               23.29 GB
  Allocatable           NO
  PE Size (KByte)       0
```

```
  Total PE              0
  Free PE               0
  Allocated PE          0
  PV UUID               3upcZc-4eS2-h4r4-iBKK-gZJv-AYt3-EKdRK6
```

Now let's create our volume group *fileserver* and add */dev/sdb1* - */dev/sde1* to it:

```
vgcreate fileserver /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
```

```
server1:~# vgcreate fileserver /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
  Volume group "fileserver" successfully created
```

Let's learn about our volume groups:

```
vgdisplay
```

```
server1:~# vgdisplay
  --- Volume group ---
  VG Name               fileserver
  System ID
  Format                lvm2
  Metadata Areas        4
  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                4
  Act PV                4
  VG Size               93.14 GB
  PE Size               4.00 MB
  Total PE              23844
  Alloc PE / Size       0 / 0
  Free  PE / Size       23844 / 93.14 GB
  VG UUID               3Y1WVF-BLET-QkKs-Qnrs-SZxI-wrNO-dTqhFP
```

Another command to learn about our volume groups:

```
vgscan
```

```
server1:~# vgscan
  Reading all physical volumes.  This may take a while...
  Found volume group "fileserver" using metadata type lvm2
```

For training purposes let's rename our volumegroup *fileserver* into *data*:

```
vgrename fileserver data
```

```
server1:~# vgrename fileserver data
  Volume group "fileserver" successfully renamed to "data"
```

Let's run *vgdisplay* and *vgscan* again to see if the volume group has been renamed:

```
vgdisplay
```

```
server1:~# vgdisplay
  --- Volume group ---
  VG Name               data
  System ID
  Format                lvm2
  Metadata Areas        4
  Metadata Sequence No  2
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                0
  Open LV               0
  Max PV                0
  Cur PV                4
  Act PV                4
  VG Size               93.14 GB
  PE Size               4.00 MB
  Total PE              23844
  Alloc PE / Size       0 / 0
  Free  PE / Size       23844 / 93.14 GB
  VG UUID               3Y1WVF-BLET-QkKs-Qnrs-SZxI-wrNO-dTqhFP
```

```
vgscan
```

```
server1:~# vgscan
  Reading all physical volumes.  This may take a while...
  Found volume group "data" using metadata type lvm2
```

Now let's delete our volume group *data*:

```
vgremove data
```

```
server1:~# vgremove data
  Volume group "data" successfully removed
```

```
vgdisplay
```

No output this time:

```
server1:~# vgdisplay
```

```
vgscan
```

```
server1:~# vgscan
  Reading all physical volumes.  This may take a while...
```

Let's create our volume group *fileserver* again:

```
vgcreate fileserver /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
```

```
server1:~# vgcreate fileserver /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1
  Volume group "fileserver" successfully created
```

Next we create our logical volumes *share* (40GB), *backup* (5GB), and *media* (1GB) in the volume group *fileserver*. Together they use a little less than

50% of the available space (that way we can make use of RAID1 later on):

```
lvcreate --name share --size 40G fileserver
```

```
server1:~# lvcreate --name share  --size 40G fileserver
  Logical volume "share" created
```

```
lvcreate --name backup --size 5G fileserver
```

```
server1:~# lvcreate --name backup --size  5G fileserver
  Logical volume "backup" created
```

```
lvcreate --name media --size 1G fileserver
```

```
server1:~# lvcreate --name media  --size  1G fileserver
  Logical volume "media" created
```

Let's get an overview of our logical volumes:

```
lvdisplay
```

```
server1:~# lvdisplay
  --- Logical volume ---
  LV Name                /dev/fileserver/share
  VG Name                fileserver
  LV UUID                280Mup-H9aa-sn0S-AXH3-04cP-V6p9-lfoGgJ
```

```
  LV Write Access        read/write
  LV Status              available
  # open                 0
  LV Size                40.00 GB
  Current LE             10240
  Segments               2
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:0

  --- Logical volume ---
  LV Name                /dev/fileserver/backup
  VG Name                fileserver
  LV UUID                zZeuKg-Dazh-aZMC-Aa99-KUSt-J6ET-KRe0cD
  LV Write Access        read/write
  LV Status              available
  # open                 0
  LV Size                5.00 GB
  Current LE             1280
  Segments               1
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:1

  --- Logical volume ---
  LV Name                /dev/fileserver/media
  VG Name                fileserver
  LV UUID                usfvrv-BC92-3pFH-2NW0-2N3e-6ERQ-4Sj7YS
  LV Write Access        read/write
  LV Status              available
  # open                 0
  LV Size                1.00 GB
  Current LE             256
```



```
  Segments               1
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:2
```

```
lvscan
```

```
server1:~# lvscan
  ACTIVE            '/dev/fileserver/share' [40.00 GB] inherit
  ACTIVE            '/dev/fileserver/backup' [5.00 GB] inherit
  ACTIVE            '/dev/fileserver/media' [1.00 GB] inherit
```

For training purposes we rename our logical volume *media* into *films*:

```
lvrename fileserver media films
```

```
server1:~# lvrename fileserver media films
  Renamed "media" to "films" in volume group "fileserver"
```

```
lvdisplay
```

```
server1:~# lvdisplay
  --- Logical volume ---
  LV Name                /dev/fileserver/share
  VG Name                fileserver
  LV UUID                280Mup-H9aa-sn0S-AXH3-04cP-V6p9-lfoGgJ
  LV Write Access        read/write
```

```
  LV Status              available
  # open                 0
  LV Size                40.00 GB
  Current LE             10240
  Segments               2
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:0

  --- Logical volume ---
  LV Name                /dev/fileserver/backup
  VG Name                fileserver
  LV UUID                zZeuKg-Dazh-aZMC-Aa99-KUSt-J6ET-KRe0cD
  LV Write Access        read/write
  LV Status              available
  # open                 0
  LV Size                5.00 GB
  Current LE             1280
  Segments               1
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:1

  --- Logical volume ---
  LV Name                /dev/fileserver/films
  VG Name                fileserver
  LV UUID                usfvrv-BC92-3pFH-2NW0-2N3e-6ERQ-4Sj7YS
  LV Write Access        read/write
  LV Status              available
  # open                 0
  LV Size                1.00 GB
  Current LE             256
  Segments               1
```

```
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:2
```

```
lvscan
```

```
server1:~# lvscan
  ACTIVE            '/dev/fileserver/share' [40.00 GB] inherit
  ACTIVE            '/dev/fileserver/backup' [5.00 GB] inherit
  ACTIVE            '/dev/fileserver/films' [1.00 GB] inherit
```

Next let's delete the logical volume *films*:

```
lvremove /dev/fileserver/films
```

```
server1:~# lvremove /dev/fileserver/films
  Do you really want to remove active logical volume "films"? [y/n]:
   Logical volume "films" successfully removed
```

We create the logical volume *media* again:

```
lvcreate --name media --size 1G fileserver
```

```
server1:~# lvcreate --name media  --size  1G fileserver
  Logical volume "media" created
```

Now let's enlarge *media* from 1GB to 1.5GB:

```
lvextend -L1.5G /dev/fileserver/media
```

```
server1:~# lvextend -L1.5G /dev/fileserver/media
  Extending logical volume media to 1.50 GB
  Logical volume media successfully resized
```

Let's shrink it to 1GB again:

```
lvreduce -L1G /dev/fileserver/media
```

```
server1:~# lvreduce -L1G /dev/fileserver/media
  WARNING: Reducing active logical volume to 1.00 GB
  THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce media? [y/n]:
  Reducing logical volume media to 1.00 GB
  Logical volume media successfully resized
```

Until now we have three logical volumes, but we don't have any filesystems in them, and without a filesystem we can't save anything in them. Therefore we create an ext3 filesystem in *share*, an xfs filesystem in *backup*, and a reiserfs filesystem in *media*:

```
mkfs.ext3 /dev/fileserver/share
```

```
server1:~# mkfs.ext3 /dev/fileserver/share
mke2fs 1.40-WIP (14-Nov-2006)
```

```
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
5242880 inodes, 10485760 blocks
524288 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=0
320 block groups
32768 blocks per group, 32768 fragments per group
16384 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 23 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

```
mkfs.xfs /dev/fileserver/backup
```

```
server1:~# mkfs.xfs /dev/fileserver/backup
meta-data=/dev/fileserver/backup isize=256    agcount=8, agsize=163840 blks
         =                       sectsz=512   attr=0
data     =                       bsize=4096   blocks=1310720, imaxpct=25
         =                       sunit=0      swidth=0 blks, unwritten=1
naming   =version 2              bsize=4096
log      =internal log           bsize=4096   blocks=2560, version=1
```

```
         =                       sectsz=512   sunit=0 blks
realtime =none                   extsz=65536  blocks=0, rtextents=0
```

```
mkfs.reiserfs /dev/fileserver/media
```

```
server1:~# mkfs.reiserfs /dev/fileserver/media
mkfs.reiserfs 3.6.19 (2003 www.namesys.com)

A pair of credits:
Alexander  Lyamin  keeps our hardware  running,  and was very  generous  to our
project in many little ways.

Chris Mason wrote the journaling code for V3,  which was enormously more useful
to users than just waiting until  we could create a wandering log filesystem as
Hans would have unwisely done without him.
Jeff Mahoney optimized the bitmap  scanning code for V3,  and performed the big
endian cleanups.


Guessing about desired format.. Kernel 2.6.17-2-486 is running.
Format 3.6 with standard journal
Count of blocks on the device: 262144
Number of blocks consumed by mkreiserfs formatting process: 8219
Blocksize: 4096
Hash function used to sort names: "r5"
Journal Size 8193 blocks (first block 18)
Journal Max transaction length 1024
inode generation number: 0
UUID: 2bebf750-6e05-47b2-99b6-916fa7ea5398
ATTENTION: YOU SHOULD REBOOT AFTER FDISK!
        ALL DATA WILL BE LOST ON '/dev/fileserver/media'!
```

```
Continue (y/n):y
Initializing journal - 0%....20%....40%....60%....80%....100%
Syncing..ok

Tell your friends to use a kernel based on 2.4.18 or later, and especially not a
kernel based on 2.4.9, when you use reiserFS. Have fun.

ReiserFS is successfully created on /dev/fileserver/media.
```

Now we are ready to mount our logical volumes. I want to mount *share* in */var/share*, *backup* in */var/backup*, and *media* in */var/media*, therefore we must create these directories first:

```
mkdir /var/media /var/backup /var/share
```

Now we can mount our logical volumes:

```
mount /dev/fileserver/share /var/share

mount /dev/fileserver/backup /var/backup

mount /dev/fileserver/media /var/media
```

Now run

```
df -h
```

You should see your logical volumes in the output:

```
server1:~# df -h
```

```
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  665M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
udev                   10M   88K   10M   1% /dev
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
/dev/mapper/fileserver-share
                       40G  177M   38G   1% /var/share
/dev/mapper/fileserver-backup
                      5.0G  144K  5.0G   1% /var/backup
/dev/mapper/fileserver-media
                      1.0G   33M  992M   4% /var/media
```

Congratulations, you've just set up your first LVM system! You can now write to and read from */var/share*, */var/backup*, and */var/media* as usual.

We have mounted our logical volumes manually, but of course we'd like to have them mounted automatically when the system boots. Therefore we modify */etc/fstab*:

```
mv /etc/fstab /etc/fstab_orig

cat /dev/null > /etc/fstab
```

```
vi /etc/fstab
```

Put the following into it:

```
# /etc/fstab: static file system information.
#
# <file system> <mount point>   <type>  <options>     <dump>  <pass>
```

```
proc            /proc           proc    defaults        0       0
/dev/sda2       /               ext3    defaults,errors=remount-ro 0       1
/dev/sda1       /boot           ext3    defaults        0       2
/dev/hdc        /media/cdrom0   udf,iso9660 user,noauto     0       0
/dev/fd0        /media/floppy0  auto    rw,user,noauto  0       0
/dev/fileserver/share   /var/share      ext3       rw,noatime    0 0
/dev/fileserver/backup    /var/backup      xfs        rw,noatime    0 0
/dev/fileserver/media    /var/media      reiserfs   rw,noatime    0 0
```

If you compare it to our backup of the original file, */etc/fstab_orig*, you will notice that we added the lines:

```
/dev/fileserver/share   /var/share     ext3       rw,noatime    0 0
/dev/fileserver/backup    /var/backup      xfs        rw,noatime    0 0
/dev/fileserver/media    /var/media      reiserfs   rw,noatime    0 0
```

Now we reboot the system:

```
shutdown -r now
```

After the system has come up again, run

```
df -h
```

again. It should still show our logical volumes in the output:

```
server1:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  665M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
```

```
udev                   10M   88K   10M   1% /dev
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
/dev/mapper/fileserver-share
                       40G  177M   38G   1% /var/share
/dev/mapper/fileserver-backup
                      5.0G  144K  5.0G   1% /var/backup
/dev/mapper/fileserver-media
                      1.0G   33M  992M   4% /var/media
```

4 Resize Logical Volumes And Their Filesystems

In this chapter we will learn how to resize our logical volume *share* which has an ext3 filesystem. (I will show how to resize logical volumes with xfs and reiserfs filesystems further down this tutorial.)

First we must unmount it:

```
umount /var/share
```

share should not be listed anymore in the

```
df -h
```

output:

```
server1:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  665M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
udev                   10M   88K   10M   1% /dev
```

```
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
/dev/mapper/fileserver-backup
                      5.0G  144K  5.0G   1% /var/backup
/dev/mapper/fileserver-media
                      1.0G   33M  992M   4% /var/media
```

Now let's enlarge *share* from 40GB to 50GB:

```
lvextend -L50G /dev/fileserver/share
```

```
server1:~# lvextend -L50G /dev/fileserver/share
  Extending logical volume share to 50.00 GB
  Logical volume share successfully resized
```

Until now we have enlarged only *share*, but not the ext3 filesystem on *share*. This is what we do now:

```
e2fsck -f /dev/fileserver/share
```

```
server1:~# e2fsck -f /dev/fileserver/share
e2fsck 1.40-WIP (14-Nov-2006)
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/fileserver/share: 11/5242880 files (9.1% non-contiguous), 209588/10485760 blocks
```

Make a note of the total amount of blocks (*10485760*) because we need it when we shrink *share* later on.

```
resize2fs /dev/fileserver/share
```

```
server1:~# resize2fs /dev/fileserver/share
resize2fs 1.40-WIP (14-Nov-2006)
Resizing the filesystem on /dev/fileserver/share to 13107200 (4k) blocks.
The filesystem on /dev/fileserver/share is now 13107200 blocks long.
```

Let's mount *share*:

```
mount /dev/fileserver/share /var/share
```

and in the

```
df -h
```

output share should now have 50GB instead of 40:

```
server1:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  665M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
udev                   10M   88K   10M   1% /dev
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
/dev/mapper/fileserver-backup
                      5.0G  144K  5.0G   1% /var/backup
/dev/mapper/fileserver-media
```

```
                      1.0G   33M  992M   4% /var/media
/dev/mapper/fileserver-share
                       50G  180M   47G   1% /var/share
```

Shrinking a logical volume is the other way round: first we must shrink the filesystem before we reduce the logical volume's size. Let's shrink share to 40GB again:

```
umount /var/share
```

```
df -h
```

```
server1:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  665M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
udev                   10M   88K   10M   1% /dev
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
/dev/mapper/fileserver-backup
                      5.0G  144K  5.0G   1% /var/backup
/dev/mapper/fileserver-media
                      1.0G   33M  992M   4% /var/media
```

```
e2fsck -f /dev/fileserver/share
```

```
server1:~# e2fsck -f /dev/fileserver/share
e2fsck 1.40-WIP (14-Nov-2006)
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/fileserver/share: 11/6553600 files (9.1% non-contiguous), 251733/13107200 blocks
```

When resizing an ext3 filesystem to a certain size (instead of all available space), *resize2fs* takes the number of blocks as argument (you can as well specify the new size in MB, etc. See

```
man resize2fs
```

for more details). From our previous operation we know the 40GB equals *10485760* blocks so we run

```
resize2fs /dev/fileserver/share 10485760
```

```
server1:~# resize2fs /dev/fileserver/share 10485760
resize2fs 1.40-WIP (14-Nov-2006)
Resizing the filesystem on /dev/fileserver/share to 10485760 (4k) blocks.
The filesystem on /dev/fileserver/share is now 10485760 blocks long.
```

We've shrinked the filesystem, now we must shrink the logical volume, too:

```
lvreduce -L40G /dev/fileserver/share
```

```
server1:~# lvreduce -L40G /dev/fileserver/share
  WARNING: Reducing active logical volume to 40.00 GB
  THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce share? [y/n]:
```



```
  Reducing logical volume share to 40.00 GB
  Logical volume share successfully resized
```

We can ignore the warning that data might be destroyed because we have shrinked the filesystem before.

Let's mount *share* again:

```
mount /dev/fileserver/share /var/share
```

The output of

```
df -h
```

should now look like this:

```
server1:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  665M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
udev                   10M   88K   10M   1% /dev
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
/dev/mapper/fileserver-backup
                      5.0G  144K  5.0G   1% /var/backup
/dev/mapper/fileserver-media
                      1.0G   33M  992M   4% /var/media
/dev/mapper/fileserver-share
                       40G  177M   38G   1% /var/share
```

5 Adding A Hard Drive And Removing Another One

We haven't used */dev/sdf* until now. We will now create the partition */dev/sdf1* (25GB) and add that to our *fileserver* volume group.

```
fdisk /dev/sdf
```

```
server1:~# fdisk /dev/sdf
  Device contains neither a valid DOS partition table, nor Sun, SGI or OSF 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.


  The number of cylinders for this disk is set to 10443.
  There is nothing wrong with that, but this is larger than 1024,
  and could in certain setups cause problems with:
  1) software that runs at boot time (e.g., old versions of LILO)
  2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)
Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help):
Command action
   a   toggle a bootable flag
   b   edit bsd disklabel
   c   toggle the dos compatibility flag
   d   delete a partition
   l   list known partition types
   m   print this menu
   n   add a new partition
   o   create a new empty DOS partition table
   p   print the partition table
```

```
   q   quit without saving changes
   s   create a new empty Sun disklabel
   t   change a partition's system id
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit
   x   extra functionality (experts only)

Command (m for help):
Command action
   e   extended
   p   primary partition (1-4)

Partition number (1-4):
First cylinder (1-10443, default 1):
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-10443, default 10443):

Command (m for help):
Selected partition 1
Hex code (type L to list codes):
Changed system type of partition 1 to 8e (Linux LVM)

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

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

Let's prepare `/dev/sdf1` for LVM:

```
pvcreate /dev/sdf1
```

```
server1:~# pvcreate /dev/sdf1
  Physical volume "/dev/sdf1" successfully created
```

Add */dev/sdf1* to our *fileserver* volume group:

```
vgextend fileserver /dev/sdf1
```

Run

```
vgdisplay
```

VG Size should now be bigger than before:

```
server1:~# vgdisplay
  --- Volume group ---
  VG Name               fileserver
  System ID
  Format                lvm2
  Metadata Areas        5
  Metadata Sequence No  12
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                3
  Open LV               3
  Max PV                0
  Cur PV                5
  Act PV                5
  VG Size               116.43 GB
  PE Size               4.00 MB
```

```
  Total PE              29805
  Alloc PE / Size       11776 / 46.00 GB
  Free  PE / Size       18029 / 70.43 GB
  VG UUID               iWr1Vk-7h7J-hLRL-SHbx-3p87-Rq47-L1GyEO
```

That's it. */dev/sdf1* has been added to the fileserver volume group.

Now let's remove */dev/sdb1*. Before we do this, we must copy all data on it to */dev/sdf1*:

```
pvmove /dev/sdb1 /dev/sdf1
```

This can take some minutes:

```
server1:~# pvmove /dev/sdb1 /dev/sdf1
  /dev/sdb1: Moved: 1.9%
  /dev/sdb1: Moved: 3.8%
  /dev/sdb1: Moved: 5.8%
  /dev/sdb1: Moved: 7.8%
  /dev/sdb1: Moved: 9.7%
  /dev/sdb1: Moved: 11.6%
  /dev/sdb1: Moved: 13.6%
  /dev/sdb1: Moved: 15.6%
  /dev/sdb1: Moved: 17.5%
  /dev/sdb1: Moved: 19.4%
  /dev/sdb1: Moved: 21.4%
  [...]
  /dev/sdb1: Moved: 85.7%
  /dev/sdb1: Moved: 87.7%
  /dev/sdb1: Moved: 89.7%
  /dev/sdb1: Moved: 91.7%
  /dev/sdb1: Moved: 93.6%
```

```
  /dev/sdb1: Moved: 95.5%
  /dev/sdb1: Moved: 97.5%
  /dev/sdb1: Moved: 99.4%
  /dev/sdb1: Moved: 100.0%
```

Next we remove */dev/sdb1* from the *fileserver* volume group:

```
vgreduce fileserver /dev/sdb1
```

```
server1:~# vgreduce fileserver /dev/sdb1
  Removed "/dev/sdb1" from volume group "fileserver"
```

```
vgdisplay
```

```
server1:~# vgdisplay
  --- Volume group ---
  VG Name               fileserver
  System ID
  Format                lvm2
  Metadata Areas        4
  Metadata Sequence No  16
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                3
  Open LV               3
  Max PV                0
  Cur PV                4
  Act PV                4
```

```
  VG Size               93.14 GB
  PE Size               4.00 MB
  Total PE              23844
  Alloc PE / Size       11776 / 46.00 GB
  Free  PE / Size       12068 / 47.14 GB
  VG UUID               iWr1Vk-7h7J-hLRL-SHbx-3p87-Rq47-L1GyEO
```

Then we run

```
pvremove /dev/sdb1
```

/dev/sdb1 shouldn't be listed as a physical volume anymore:

```
pvdisplay
```

```
server1:~# pvdisplay
  --- Physical volume ---
  PV Name               /dev/sdc1
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               1682
  Allocated PE          4279
  PV UUID               40GJyh-IbsI-pzhn-TDRq-PQ3l-3ut0-AVSE4B

  --- Physical volume ---
  PV Name               /dev/sdd1
  VG Name               fileserver
```

```
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               4681
  Allocated PE          1280
  PV UUID               4mU63D-4s26-uL00-r0pO-Q0hP-mvQR-2YJN5B

  --- Physical volume ---
  PV Name               /dev/sde1
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               5705
  Allocated PE          256
  PV UUID               3upcZc-4eS2-h4r4-iBKK-gZJv-AYt3-EKdRK6

  --- Physical volume ---
  PV Name               /dev/sdf1
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes (but full)
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               0
  Allocated PE          5961
  PV UUID               1xgo2I-SBjj-0MAz-lmDu-OLZ1-3NdO-mLkS20
```

You could now remove `/dev/sdb` from the system (if this was a real system and not a virtual machine).

6 Return To The System's Original State

In this chapter we will undo all changes from the previous chapters to return to the system's original state. This is just for training purposes so that you learn how to undo an LVM setup.

First we must unmount our logical volumes:

```
umount /var/share

umount /var/backup

umount /var/media
```

```
df -h
```

```
server1:~# df -h
  Filesystem            Size  Used Avail Use% Mounted on
  /dev/sda2              19G  665M   17G   4% /
  tmpfs                  78M     0   78M   0% /lib/init/rw
  udev                   10M   92K   10M   1% /dev
  tmpfs                  78M     0   78M   0% /dev/shm
  /dev/sda1             137M   17M  114M  13% /boot
```

Then we delete each of them:

```
lvremove /dev/fileserver/share
```

```
server1:~# lvremove /dev/fileserver/share
  Do you really want to remove active logical volume "share"? [y/n]:
```

```
  Logical volume "share" successfully removed
```

```
lvremove /dev/fileserver/backup
```

```
server1:~# lvremove /dev/fileserver/backup
  Do you really want to remove active logical volume "backup"? [y/n]:
   Logical volume "backup" successfully removed
```

```
lvremove /dev/fileserver/media
```

```
server1:~# lvremove /dev/fileserver/media
  Do you really want to remove active logical volume "media"? [y/n]:
   Logical volume "media" successfully removed
```

Next we remove the volume group *fileserver*:

```
vgremove fileserver
```

```
server1:~# vgremove fileserver
  Volume group "fileserver" successfully removed
```

Finally we do this:

```
pvremove /dev/sdc1 /dev/sdd1 /dev/sde1 /dev/sdf1
```

```
server1:~# pvremove  /dev/sdc1 /dev/sdd1 /dev/sde1 /dev/sdf1
  Labels on physical volume "/dev/sdc1" successfully wiped
  Labels on physical volume "/dev/sdd1" successfully wiped
  Labels on physical volume "/dev/sde1" successfully wiped
  Labels on physical volume "/dev/sdf1" successfully wiped
```

```
vgdisplay
```

```
server1:~# vgdisplay
  No volume groups found
```

```
pvdisplay
```

should display nothing at all:

```
server1:~# pvdisplay
```

Now we must undo our changes in `/etc/fstab` to avoid that the system tries to mount non-existing devices. Fortunately we have made a backup of the original file that we can copy back now:

```
mv /etc/fstab_orig /etc/fstab
```

Reboot the system:

```
shutdown -r now
```

Afterwards the output of

```
df -h
```

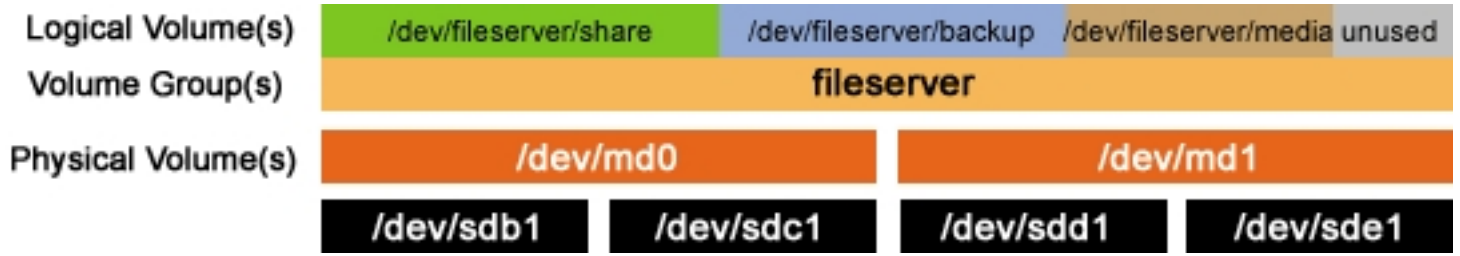
should look like this:

```
server1:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  666M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
udev                   10M   92K   10M   1% /dev
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
```

Now the system is like it was in the beginning (except that the partitions `/dev/sdb1` - `/dev/sdf1` still exist - you could delete them with `fdisk` but we don't do this now - as well as the directories `/var/share`, `/var/backup`, and `/var/media` which we also don't delete).

7 LVM On RAID1

In this chapter we will set up LVM again and move it to a RAID1 array to guarantee for high-availability. In the end this should look like this:



This means we will make the RAID array */dev/md0* from the partitions */dev/sdb1* + */dev/sdc1*, and the RAID array */dev/md1* from the partitions */dev/sdd1* + */dev/sde1*. */dev/md0* and */dev/md1* will then be the physical volumes for LVM.

Before we come to that, we set up LVM as before:

```
pvcreate /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1

vgcreate fileserver /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1

lvcreate --name share --size 40G fileserver

lvcreate --name backup --size 5G fileserver

lvcreate --name media --size 1G fileserver
```

```
mkfs.ext3 /dev/fileserver/share

mkfs.xfs /dev/fileserver/backup

mkfs.reiserfs /dev/fileserver/media
```

Then we mount our logical volumes:

```
mount /dev/fileserver/share /var/share

mount /dev/fileserver/backup /var/backup

mount /dev/fileserver/media /var/media
```

The output of

```
df -h
```

should now look like this:

```
server1:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  666M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
udev                   10M   92K   10M   1% /dev
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
/dev/mapper/fileserver-share
                       40G  177M   38G   1% /var/share
/dev/mapper/fileserver-backup
                      5.0G  144K  5.0G   1% /var/backup
/dev/mapper/fileserver-media
                      1.0G   33M  992M   4% /var/media
```

Now we must move the contents of */dev/sdc1* and */dev/sde1* (*/dev/sdc1* is the second partition of our future */dev/md0*, */dev/sde1* the second partition of our future */dev/md1*) to the remaining partitions, because we will afterwards remove them from LVM and format them with the type *fd* (Linux RAID autodetect) and move them to */dev/md0* resp. */dev/md1*.

```
modprobe dm-mirror

pvmove /dev/sdc1
```

```
vgreduce fileserver /dev/sdc1

pvremove /dev/sdc1
```

```
pvdisplay
```

```
server1:~# pvdisplay
  --- Physical volume ---
  PV Name               /dev/sdb1
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes (but full)
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               0
  Allocated PE          5961
  PV UUID               USDJyG-VDM2-r406-OjQo-h3eb-c9Mp-4nvnvu

  --- Physical volume ---
  PV Name               /dev/sdd1
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               4681
```

```
  Allocated PE          1280
  PV UUID               qdEB5d-389d-O5UA-Kbwv-mn1y-74FY-4zublN

  --- Physical volume ---
  PV Name               /dev/sde1
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               1426
  Allocated PE          4535
  PV UUID               4vL1e0-sr2M-awGd-qDJm-ZrC9-wuxW-2lEqp2
```

```
pvmove /dev/sde1
```

```
vgreduce fileserver /dev/sde1

pvremove /dev/sde1
```

```
pvdisplay
```

```
server1:~# pvdisplay
  --- Physical volume ---
  PV Name               /dev/sdb1
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes (but full)
  PE Size (KByte)       4096
```

```
  Total PE              5961
  Free PE               0
  Allocated PE          5961
  PV UUID               USDJyG-VDM2-r406-OjQo-h3eb-c9Mp-4nvnvu

  --- Physical volume ---
  PV Name               /dev/sdd1
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               146
  Allocated PE          5815
  PV UUID               qdEB5d-389d-O5UA-Kbwv-mn1y-74FY-4zublN
```

Now we format `/dev/sdc1` with the type `fd` (Linux RAID autodetect):

```
fdisk /dev/sdc
```

```
server1:~# fdisk /dev/sdc

  The number of cylinders for this disk is set to 10443.
  There is nothing wrong with that, but this is larger than 1024,
  and could in certain setups cause problems with:
  1) software that runs at boot time (e.g., old versions of LILO)
  2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):
Command action
```

```
   a   toggle a bootable flag
   b   edit bsd disklabel
   c   toggle the dos compatibility flag
   d   delete a partition
   l   list known partition types
   m   print this menu
   n   add a new partition
   o   create a new empty DOS partition table
   p   print the partition table
   q   quit without saving changes
   s   create a new empty Sun disklabel
   t   change a partition's system id
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit
   x   extra functionality (experts only)

Command (m for help):
Selected partition 1
Hex code (type L to list codes):

 0  Empty           1e  Hidden W95 FAT1 80  Old Minix       be  Solaris boot
 1  FAT12           24  NEC DOS         81  Minix / old Lin bf  Solaris
 2  XENIX root      39  Plan 9          82  Linux swap / So c1  DRDOS/sec (FAT-
 3  XENIX usr       3c  PartitionMagic  83  Linux           c4  DRDOS/sec (FAT-
 4  FAT16 <32M      40  Venix 80286     84  OS/2 hidden C:  c6  DRDOS/sec (FAT-
 5  Extended        41  PPC PReP Boot   85  Linux extended  c7  Syrinx
 6  FAT16           42  SFS             86  NTFS volume set da  Non-FS data
 7  HPFS/NTFS       4d  QNX4.x          87  NTFS volume set db  CP/M / CTOS / .
 8  AIX             4e  QNX4.x 2nd part 88  Linux plaintext de  Dell Utility
 9  AIX bootable    4f  QNX4.x 3rd part 8e  Linux LVM       df  BootIt
 a  OS/2 Boot Manag 50  OnTrack DM      93  Amoeba          e1  DOS access
 b  W95 FAT32       51  OnTrack DM6 Aux 94  Amoeba BBT      e3  DOS R/O
```

```
 c  W95 FAT32 (LBA) 52  CP/M            9f  BSD/OS          e4  SpeedStor
 e  W95 FAT16 (LBA) 53  OnTrack DM6 Aux a0  IBM Thinkpad hi eb  BeOS fs
 f  W95 Ext'd (LBA) 54  OnTrackDM6      a5  FreeBSD         ee  EFI GPT
10  OPUS            55  EZ-Drive        a6  OpenBSD         ef  EFI (FAT-12/16/
11  Hidden FAT12    56  Golden Bow      a7  NeXTSTEP        f0  Linux/PA-RISC b
12  Compaq diagnost 5c  Priam Edisk     a8  Darwin UFS      f1  SpeedStor
14  Hidden FAT16 <3 61  SpeedStor       a9  NetBSD          f4  SpeedStor
16  Hidden FAT16    63  GNU HURD or Sys ab  Darwin boot     f2  DOS secondary
17  Hidden HPFS/NTF 64  Novell Netware  b7  BSDI fs         fd  Linux raid auto
18  AST SmartSleep  65  Novell Netware  b8  BSDI swap       fe  LANstep
1b  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid ff  BBT
1c  Hidden W95 FAT3 75  PC/IX
Hex code (type L to list codes):
Changed system type of partition 1 to fd (Linux raid autodetect)

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

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

Now do the same with */dev/sde1*:

```
fdisk /dev/sde
```

The output of

```
fdisk -l
```

should now look like this:

```
server1:~# fdisk -l

Disk /dev/sda: 21.4 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          18      144553+  83  Linux
/dev/sda2              19        2450    19535040   83  Linux
/dev/sda4            2451        2610     1285200   82  Linux swap / Solaris

Disk /dev/sdb: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1        3040    24418768+  8e  Linux LVM

Disk /dev/sdc: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1               1        3040    24418768+  fd  Linux raid autodetect

Disk /dev/sdd: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdd1               1        3040    24418768+  8e  Linux LVM

Disk /dev/sde: 85.8 GB, 85899345920 bytes
```

```
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sde1               1        3040    24418768+  fd  Linux raid autodetect

Disk /dev/sdf: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdf1               1        3040    24418768+  8e  Linux LVM
```

Next we add */dev/sdc1* to */dev/md0* and */dev/sde1* to */dev/md1*. Because the second nodes (*/dev/sdb1* and */dev/sdd1*) are not ready yet, we must specify *missing* in the following commands:

```
mdadm --create /dev/md0 --auto=yes -l 1 -n 2 /dev/sdc1 missing
```

```
server1:~# mdadm --create /dev/md0 --auto=yes -l 1 -n 2 /dev/sdc1 missing
mdadm: array /dev/md0 started.
```

```
mdadm --create /dev/md1 --auto=yes -l 1 -n 2 /dev/sde1 missing
```

```
server1:~# mdadm --create /dev/md1 --auto=yes -l 1 -n 2 /dev/sde1 missing
mdadm: array /dev/md1 started.
```

Afterwards we prepare */dev/md0* and */dev/md1* for LVM:

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

```
server1:~# pvcreate /dev/md0 /dev/md1
  Physical volume "/dev/md0" successfully created
  Physical volume "/dev/md1" successfully created
```

and extend our *fileserver* volume group:

```
vgextend fileserver /dev/md0 /dev/md1
```

```
server1:~# vgextend fileserver /dev/md0 /dev/md1
  Volume group "fileserver" successfully extended
```

The outputs of

```
pvdisplay
```

and

```
vgdisplay
```

should look like this:

```
server1:~# pvdisplay
  --- Physical volume ---
  PV Name               /dev/sdb1
  VG Name               fileserver
```

```
  PV Size               23.29 GB / not usable 0
  Allocatable           yes (but full)
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               0
  Allocated PE          5961
  PV UUID               USDJyG-VDM2-r406-OjQo-h3eb-c9Mp-4nvnvu

  --- Physical volume ---
  PV Name               /dev/sdd1
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               146
  Allocated PE          5815
  PV UUID               qdEB5d-389d-O5UA-Kbwv-mn1y-74FY-4zublN

  --- Physical volume ---
  PV Name               /dev/md0
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               5961
  Allocated PE          0
  PV UUID               7JHUXF-1R2p-OjbJ-X1OT-uaeg-gWRx-H6zx3P

  --- Physical volume ---
  PV Name               /dev/md1
  VG Name               fileserver
```

```
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               5961
  Allocated PE          0
  PV UUID               pwQ5AJ-RwVK-EebA-0Z13-d27d-2IdP-HqT5RW


server1:~# vgdisplay
  --- Volume group ---
  VG Name               fileserver
  System ID
  Format                lvm2
  Metadata Areas        4
  Metadata Sequence No  14
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                3
  Open LV               3
  Max PV                0
  Cur PV                4
  Act PV                4
  VG Size               93.14 GB
  PE Size               4.00 MB
  Total PE              23844
  Alloc PE / Size       11776 / 46.00 GB
  Free  PE / Size       12068 / 47.14 GB
  VG UUID               dQDEHT-kNHf-UjRm-rmJ3-OUYx-9G1t-aVskI1
```

Now we move the contents of `/dev/sdb1` to `/dev/md0` and the contents of `/dev/sdd1` to `/dev/md1`, then we remove `/dev/sdb1` and `/dev/sdd1` from

LVM:

```
pvmove /dev/sdb1 /dev/md0
```

```
pvmove /dev/sdd1 /dev/md1
```

```
vgreduce fileserver /dev/sdb1 /dev/sdd1

pvremove /dev/sdb1 /dev/sdd1
```

Now only */dev/md0* and */dev/md1* should be left as physical volumes:

```
pvdisplay
```

```
server1:~# pvdisplay
  --- Physical volume ---
  PV Name               /dev/md0
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes (but full)
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               0
  Allocated PE          5961
  PV UUID               7JHUXF-1R2p-OjbJ-X1OT-uaeg-gWRx-H6zx3P

  --- Physical volume ---
  PV Name               /dev/md1
  VG Name               fileserver
```

```
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               146
  Allocated PE          5815
  PV UUID               pwQ5AJ-RwVK-EebA-0Z13-d27d-2IdP-HqT5RW
```

Now we format `/dev/sdb1` with `fd` (Linux RAID autodetect):

```
fdisk /dev/sdb
```

```
server1:~# fdisk /dev/sdb

  The number of cylinders for this disk is set to 32635.
  There is nothing wrong with that, but this is larger than 1024,
  and could in certain setups cause problems with:
  1) software that runs at boot time (e.g., old versions of LILO)
  2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):
Command action
   a   toggle a bootable flag
   b   edit bsd disklabel
   c   toggle the dos compatibility flag
   d   delete a partition
   l   list known partition types
   m   print this menu
   n   add a new partition
   o   create a new empty DOS partition table
```

```
   p   print the partition table
   q   quit without saving changes
   s   create a new empty Sun disklabel
   t   change a partition's system id
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit
   x   extra functionality (experts only)

Command (m for help):
Selected partition 1
Hex code (type L to list codes):
Changed system type of partition 1 to fd (Linux raid autodetect)

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

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

Do the same with */dev/sdd1*:

```
fdisk /dev/sdd
```

Next add */dev/sdb1* to */dev/md0* and */dev/sdd1* to */dev/md1*:

```
mdadm --manage /dev/md0 --add /dev/sdb1
```

```
server1:~# mdadm --manage /dev/md0 --add /dev/sdb1
mdadm: added /dev/sdb1
```

```
mdadm --manage /dev/md1 --add /dev/sdd1
```

```
server1:~# mdadm --manage /dev/md1 --add /dev/sdd1
mdadm: added /dev/sdd1
```

Now the two RAID arrays will be synchronized. This will take some time, you can check with

```
cat /proc/mdstat
```

when the process is finished. The output looks like this for an unfinished process:

```
server1:~# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid5] [raid4] [raid6] [raid10]
md1 : active raid1 sdd1[2] sde1[0]
      24418688 blocks [2/1] [U_]
      [=>...................]  recovery =  6.4% (1586560/24418688) finish=1.9min speed=198320K/sec

md0 : active raid1 sdb1[2] sdc1[0]
      24418688 blocks [2/1] [U_]
      [==>..................]  recovery = 10.5% (2587264/24418688) finish=2.8min speed=129363K/sec

unused devices: <none>
```

and like this when the process is finished:

```
server1:~# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid5] [raid4] [raid6] [raid10]
md1 : active raid1 sdd1[1] sde1[0]
      24418688 blocks [2/2] [UU]
```

```
md0 : active raid1 sdb1[1] sdc1[0]
      24418688 blocks [2/2] [UU]

unused devices: <none>
```

If you have a look at `PV Size` in the output of

```
pvdisplay
```

you will see that `2 * 23.29GB = 46.58GB` are available, however only `40GB (share) + 5GB (backup) + 1GB (media) = 46GB` are used which means we could extend one of our logical devices with about 0.5GB. I've already shown how to extend an ext3 logical volume (`share`), so we will resize `media` now which uses reiserfs. reiserfs filesystems can be resized without unmounting:

```
lvextend -L1.5G /dev/fileserver/media
```

```
server1:~# lvextend -L1.5G /dev/fileserver/media
  Extending logical volume media to 1.50 GB
  Logical volume media successfully resized
```

```
resize_reiserfs /dev/fileserver/media
```

```
server1:~# resize_reiserfs /dev/fileserver/media
resize_reiserfs 3.6.19 (2003 www.namesys.com)



resize_reiserfs: On-line resizing finished successfully.
```

The output of

```
df -h
```

looks like this:

```
server1:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  666M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
udev                   10M   92K   10M   1% /dev
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
/dev/mapper/fileserver-share
                       40G  177M   38G   1% /var/share
/dev/mapper/fileserver-backup
                      5.0G  144K  5.0G   1% /var/backup
/dev/mapper/fileserver-media
                      1.5G   33M  1.5G   3% /var/media
```

If we want our logical volumes to be mounted automatically at boot time, we must modify `/etc/fstab` again (like in chapter 3):

```
mv /etc/fstab /etc/fstab_orig

cat /dev/null > /etc/fstab
```

```
vi /etc/fstab
```

Put the following into it:


```
# /etc/fstab: static file system information.
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
proc            /proc           proc    defaults        0       0
/dev/sda2       /               ext3    defaults,errors=remount-ro 0       1
/dev/sda1       /boot           ext3    defaults        0       2
/dev/hdc        /media/cdrom0   udf,iso9660 user,noauto     0       0
/dev/fd0        /media/floppy0  auto    rw,user,noauto  0       0
/dev/fileserver/share   /var/share     ext3       rw,noatime    0 0
/dev/fileserver/backup    /var/backup      xfs        rw,noatime    0 0
/dev/fileserver/media    /var/media      reiserfs   rw,noatime    0 0
```

If you compare it to our backup of the original file, */etc/fstab_orig*, you will notice that we added the lines:

```
/dev/fileserver/share   /var/share     ext3       rw,noatime    0 0
  /dev/fileserver/backup    /var/backup      xfs        rw,noatime    0 0
  /dev/fileserver/media    /var/media      reiserfs   rw,noatime    0 0
```

Now we reboot the system:

```
shutdown -r now
```

After the system has come up again, run

```
df -h
```

again. It should still show our logical volumes in the output:

```
server1:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  666M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
udev                   10M  100K   10M   1% /dev
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
/dev/mapper/fileserver-share
                       40G  177M   38G   1% /var/share
/dev/mapper/fileserver-backup
                      5.0G  144K  5.0G   1% /var/backup
/dev/mapper/fileserver-media
                      1.5G   33M  1.5G   3% /var/media
```

Now we are finished with our LVM on RAID1 setup.

8 Replacing The Hard Disks With Bigger Ones

We are currently using four hard disks with a size of 25GB each (at least we are acting like that). Now let's assume this isn't enough anymore, and we need more space in our RAID setup. Therefore we will replace our 25GB hard disks with 80GB hard disks (in fact we will still use the current hard disks, but use their full capacity now - in the real life you would replace your old, small hard disks with new, bigger ones).

The procedure is as follows: first we remove `/dev/sdb` and `/dev/sdd` from the RAID arrays, replace them with bigger hard disks, put them back into the RAID arrays, and then we do the same again with `/dev/sdc` and `/dev/sde`.

First we mark `/dev/sdb1` as failed:

```
mdadm --manage /dev/md0 --fail /dev/sdb1
```

```
server1:~# mdadm --manage /dev/md0 --fail /dev/sdb1
mdadm: set /dev/sdb1 faulty in /dev/md0
```

The output of

```
cat /proc/mdstat
```

looks now like this:

```
server1:~# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid5] [raid4] [raid6] [raid10]
md0 : active raid1 sdc1[0] sdb1[2](F)
      24418688 blocks [2/1] [U_]

md1 : active raid1 sde1[0] sdd1[1]
      24418688 blocks [2/2] [UU]

unused devices: <none>
```

Then we remove `/dev/sdb1` from the RAID array `/dev/md0`:

```
mdadm --manage /dev/md0 --remove /dev/sdb1
```

```
server1:~# mdadm --manage /dev/md0 --remove /dev/sdb1
mdadm: hot removed /dev/sdb1
```

```
cat /proc/mdstat
```

```
server1:~# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid5] [raid4] [raid6] [raid10]
md0 : active raid1 sdc1[0]
      24418688 blocks [2/1] [U_]
```



```
md1 : active raid1 sde1[0] sdd1[1]
      24418688 blocks [2/2] [UU]

unused devices: <none>
```

Now we do the same with */dev/sdd1*:

```
mdadm --manage /dev/md1 --fail /dev/sdd1
```

```
server1:~# mdadm --manage /dev/md1 --fail /dev/sdd1
mdadm: set /dev/sdd1 faulty in /dev/md1
```

```
cat /proc/mdstat
```

```
server1:~# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid5] [raid4] [raid6] [raid10]
md0 : active raid1 sdc1[0]
      24418688 blocks [2/1] [U_]

md1 : active raid1 sde1[0] sdd1[2](F)
      24418688 blocks [2/1] [U_]

unused devices: <none>
```

```
mdadm --manage /dev/md1 --remove /dev/sdd1
```

```
server1:~# mdadm --manage /dev/md1 --remove /dev/sdd1
mdadm: hot removed /dev/sdd1
```

```
cat /proc/mdstat
```

```
server1:~# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid5] [raid4] [raid6] [raid10]
md0 : active raid1 sdc1[0]
      24418688 blocks [2/1] [U_]

md1 : active raid1 sde1[0]
      24418688 blocks [2/1] [U_]

unused devices: <none>
```

On a real system you would now shut it down, pull out the 25GB */dev/sdb* and */dev/sdd* and replace them with 80GB ones. As I said before, we don't have to do this because all hard disks already have a capacity of 80GB.

Next we must format */dev/sdb* and */dev/sdd*. We must create a */dev/sdb1* resp. */dev/sdd1* partition, type *fd* (Linux RAID autodetect), size 25GB (the same settings as on the old hard disks), and a */dev/sdb2* resp. */dev/sdd2* partition, type *fd*, that cover the rest of the hard disks. As */dev/sdb1* and */dev/sdd1* are still present on our hard disks, we only have to create */dev/sdb2* and */dev/sdd2* in this special example.

```
fdisk /dev/sdb
```

```
server1:~# fdisk /dev/sdb

  The number of cylinders for this disk is set to 10443.
  There is nothing wrong with that, but this is larger than 1024,
  and could in certain setups cause problems with:
  1) software that runs at boot time (e.g., old versions of LILO)
  2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)
```

```
Command (m for help):

Disk /dev/sdb: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1        3040    24418768+  fd  Linux raid autodetect

Command (m for help):
Command action
   e   extended
   p   primary partition (1-4)

Partition number (1-4):
First cylinder (3041-10443, default 3041):
Using default value 3041
Last cylinder or +size or +sizeM or +sizeK (3041-10443, default 10443):
Using default value 10443

Command (m for help):
Partition number (1-4):
Hex code (type L to list codes): <-- fd
Changed system type of partition 2 to fd (Linux raid autodetect)

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

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

Do the same for */dev/sdd*:

```
fdisk /dev/sdd
```

The output of

```
fdisk -l
```

looks now like this:

```
server1:~# fdisk -l

Disk /dev/sda: 21.4 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          18      144553+  83  Linux
/dev/sda2              19        2450    19535040   83  Linux
/dev/sda4            2451        2610     1285200   82  Linux swap / Solaris

Disk /dev/sdb: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1        3040    24418768+  fd  Linux raid autodetect
/dev/sdb2            3041       10443    59464597+  fd  Linux raid autodetect

Disk /dev/sdc: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

```
   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1               1        3040    24418768+  fd  Linux raid autodetect

Disk /dev/sdd: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdd1               1        3040    24418768+  fd  Linux raid autodetect
/dev/sdd2            3041       10443    59464597+  fd  Linux raid autodetect

Disk /dev/sde: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sde1               1        3040    24418768+  fd  Linux raid autodetect

Disk /dev/sdf: 85.8 GB, 85899345920 bytes
255 heads, 63 sectors/track, 10443 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sdf1               1        3040    24418768+  8e  Linux LVM

Disk /dev/md1: 25.0 GB, 25004736512 bytes
2 heads, 4 sectors/track, 6104672 cylinders
Units = cylinders of 8 * 512 = 4096 bytes

Disk /dev/md1 doesn't contain a valid partition table

Disk /dev/md0: 25.0 GB, 25004736512 bytes
2 heads, 4 sectors/track, 6104672 cylinders
```

```
Units = cylinders of 8 * 512 = 4096 bytes

Disk /dev/md0 doesn't contain a valid partition table
```

Now we add */dev/sdb1* to */dev/md0* again and */dev/sdd1* to */dev/md1*:

```
mdadm --manage /dev/md0 --add /dev/sdb1
```

```
server1:~# mdadm --manage /dev/md0 --add /dev/sdb1
mdadm: re-added /dev/sdb1
```

```
mdadm --manage /dev/md1 --add /dev/sdd1
```

```
server1:~# mdadm --manage /dev/md1 --add /dev/sdd1
mdadm: re-added /dev/sdd1
```

Now the contents of both RAID arrays will be synchronized. We must wait until this is finished before we can go on. We can check the status of the synchronization with

```
cat /proc/mdstat
```

The output looks like this during synchronization:

```
server1:~# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid5] [raid4] [raid6] [raid10]
md0 : active raid1 sdb1[1] sdc1[0]
      24418688 blocks [2/1] [U_]
      [=>...................]  recovery =  9.9% (2423168/24418688) finish=2.8min speed=127535K/sec
```

```
md1 : active raid1 sdd1[1] sde1[0]
      24418688 blocks [2/1] [U_]
      [=>...................]  recovery =  6.4% (1572096/24418688) finish=1.9min speed=196512K/sec

unused devices: <none>
```

and like this when it's finished:

```
server1:~# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid5] [raid4] [raid6] [raid10]
md0 : active raid1 sdb1[1] sdc1[0]
      24418688 blocks [2/2] [UU]

md1 : active raid1 sdd1[1] sde1[0]
      24418688 blocks [2/2] [UU]

unused devices: <none>
```

Now we do the same process again, this time replacing `/dev/sdc` and `/dev/sde`:

```
mdadm --manage /dev/md0 --fail /dev/sdc1

mdadm --manage /dev/md0 --remove /dev/sdc1

mdadm --manage /dev/md1 --fail /dev/sde1

mdadm --manage /dev/md1 --remove /dev/sde1
```

```
fdisk /dev/sdc

fdisk /dev/sde
```

```
mdadm --manage /dev/md0 --add /dev/sdc1

mdadm --manage /dev/md1 --add /dev/sde1
```

```
cat /proc/mdstat
```

Wait until the synchronization has finished.

Next we create the RAID arrays */dev/md2* from */dev/sdb2* and */dev/sdc2* as well as */dev/md3* from */dev/sdd2* and */dev/sde2*.

```
mdadm --create /dev/md2 --auto=yes -l 1 -n 2 /dev/sdb2 /dev/sdc2
```

```
server1:~# mdadm --create /dev/md2 --auto=yes -l 1 -n 2 /dev/sdb2 /dev/sdc2
mdadm: array /dev/md2 started.
```

```
mdadm --create /dev/md3 --auto=yes -l 1 -n 2 /dev/sdd2 /dev/sde2
```

```
server1:~# mdadm --create /dev/md3 --auto=yes -l 1 -n 2 /dev/sdd2 /dev/sde2
mdadm: array /dev/md3 started.
```

The new RAID arrays must be synchronized before we go on, so you should check

```
cat /proc/mdstat
```

```
server1:~# cat /proc/mdstat
Personalities : [linear] [multipath] [raid0] [raid1] [raid5] [raid4] [raid6] [raid10]
md3 : active raid1 sde2[1] sdd2[0]
      59464512 blocks [2/2] [UU]
      [=>...................]  resync =  5.1% (3044224/59464512) finish=5.5min speed=169123K/sec

md2 : active raid1 sdc2[1] sdb2[0]
      59464512 blocks [2/2] [UU]
      [=>...................]  resync =  5.5% (3312512/59464512) finish=9.3min speed=100379K/sec

md0 : active raid1 sdc1[0] sdb1[1]
      24418688 blocks [2/2] [UU]

md1 : active raid1 sde1[0] sdd1[1]
      24418688 blocks [2/2] [UU]

unused devices: <none>
```

After the synchronization has finished, we prepare */dev/md2* and */dev/md3* for LVM:

```
pvcreate /dev/md2 /dev/md3
```

```
server1:~# pvcreate /dev/md2 /dev/md3
  Physical volume "/dev/md2" successfully created
  Physical volume "/dev/md3" successfully created
```

and add */dev/md2* and */dev/md3* to our *fileserver* volume group:

```
vgextend fileserver /dev/md2 /dev/md3
```

```
server1:~# vgextend fileserver /dev/md2 /dev/md3
  Volume group "fileserver" successfully extended
```

Now let's run our *display commands:

```
pvdisplay
```

```
server1:~# pvdisplay
  --- Physical volume ---
  PV Name               /dev/md0
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes (but full)
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               0
  Allocated PE          5961
  PV UUID               7JHUXF-1R2p-OjbJ-X1OT-uaeg-gWRx-H6zx3P

  --- Physical volume ---
  PV Name               /dev/md1
  VG Name               fileserver
  PV Size               23.29 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              5961
  Free PE               18
  Allocated PE          5943
  PV UUID               pwQ5AJ-RwVK-EebA-0Z13-d27d-2IdP-HqT5RW

  --- Physical volume ---
```

```
  PV Name               /dev/md2
  VG Name               fileserver
  PV Size               56.71 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              14517
  Free PE               14517
  Allocated PE          0
  PV UUID               300kTo-evxm-rfmf-90LA-4YOJ-2LG5-t4JHnf

  --- Physical volume ---
  PV Name               /dev/md3
  VG Name               fileserver
  PV Size               56.71 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       4096
  Total PE              14517
  Free PE               14517
  Allocated PE          0
  PV UUID               LXFSW6-7LQX-ZGGU-dV95-jQgg-TK44-U5JOjO
```

```
vgdisplay
```

```
server1:~# vgdisplay
  --- Volume group ---
  VG Name               fileserver
  System ID
  Format                lvm2
  Metadata Areas        4
  Metadata Sequence No  26
  VG Access             read/write
```

```
  VG Status              resizable
  MAX LV                 0
  Cur LV                 3
  Open LV                3
  Max PV                 0
  Cur PV                 4
  Act PV                 4
  VG Size                159.98 GB
  PE Size                4.00 MB
  Total PE               40956
  Alloc PE / Size        11904 / 46.50 GB
  Free  PE / Size        29052 / 113.48 GB
  VG UUID                dQDEHT-kNHf-UjRm-rmJ3-OUYx-9G1t-aVskI1
```

```
lvdisplay
```

```
server1:~# lvdisplay
  --- Logical volume ---
  LV Name                /dev/fileserver/share
  VG Name                fileserver
  LV UUID                bcn3Oi-vW3p-WoyX-QlF2-xEtz-uz7Z-4DllYN
  LV Write Access        read/write
  LV Status              available
  # open                 1
  LV Size                40.00 GB
  Current LE             10240
  Segments               2
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:0
```

```
  --- Logical volume ---
  LV Name                /dev/fileserver/backup
  VG Name                fileserver
  LV UUID                vfKVnU-gFXB-C6hE-1L4g-il6U-78EE-N8Sni8
  LV Write Access        read/write
  LV Status              available
  # open                 1
  LV Size                5.00 GB
  Current LE             1280
  Segments               1
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:1

  --- Logical volume ---
  LV Name                /dev/fileserver/media
  VG Name                fileserver
  LV UUID                H1gagh-wTwH-Og0S-cJNQ-BgX1-zGlM-LwLVzE
  LV Write Access        read/write
  LV Status              available
  # open                 2
  LV Size                1.50 GB
  Current LE             384
  Segments               1
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:2
```

If your outputs look similar, you have successfully replaced your small hard disks with bigger ones.

Now that we have more disk space (`2* 23.29GB + 2 * 56.71GB = 160GB`) we could enlarge our logical volumes. Until now you know how to enlarge ext3 and reiserfs partitions, so let's enlarge our `backup` logical volume now which uses xfs:


```
lvextend -L10G /dev/fileserver/backup
```

```
server1:~# lvextend -L10G /dev/fileserver/backup
  Extending logical volume backup to 10.00 GB
  Logical volume backup successfully resized
```

To enlarge the xfs filesystem, we run

```
xfs_growfs /dev/fileserver/backup
```

```
server1:~# xfs_growfs /dev/fileserver/backup
meta-data=/dev/fileserver/backup isize=256    agcount=8, agsize=163840 blks
         =                       sectsz=512   attr=0
data     =                       bsize=4096   blocks=1310720, imaxpct=25
         =                       sunit=0      swidth=0 blks, unwritten=1
naming   =version 2              bsize=4096
log      =internal               bsize=4096   blocks=2560, version=1
         =                       sectsz=512   sunit=0 blks
realtime =none                   extsz=65536  blocks=0, rtextents=0
data blocks changed from 1310720 to 2621440
```

The output of

```
df -h
```

should now look like this:

```
server1:~# df -h
```

```
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              19G  666M   17G   4% /
tmpfs                  78M     0   78M   0% /lib/init/rw
udev                   10M  116K  9.9M   2% /dev
tmpfs                  78M     0   78M   0% /dev/shm
/dev/sda1             137M   17M  114M  13% /boot
/dev/mapper/fileserver-share
                       40G  177M   38G   1% /var/share
/dev/mapper/fileserver-backup
                       10G  272K   10G   1% /var/backup
/dev/mapper/fileserver-media
                      1.5G   33M  1.5G   3% /var/media
```

That's it! If you've made it until here, you should now be used to LVM and LVM on RAID.

9 Links

- Managing Disk Space with LVM: **http://www.linuxdevcenter.com/pub/a/linux/2006/04/27/managing-disk-space-with-lvm.html**
- A simple introduction to working with LVM: **http://www.debian-administration.org/articles/410**
- Debian: **http://www.debian.org**