Linux LVM

A basic introduction to LVM

Logical **V**olume **M**anagement

by Peter Sjöberg*

peters @ techwiz.ca

Slides can be found at

http:// www.techwiz.ca/~peters/presentations/lvm

*with input from Frank Kannemann

Agenda

Some basic background about Files and Filesystems under Linux

What is LVM and why should I use it

How to create a simple LVM environment

If time allows

- A little about snapshots and a advanced FC4 slide demo

Audience Participation

Please raise you hands to allow me to taylor this presentation, not meant to embarrass anyone:

- Beginner User < 1 year: haven't touched the CLI
- Intermediate User > 1 year: Using CLI sometimes
- Advanced user: linux is my first language

Do you have more have 1 disk attached to your Linux box?

When to and not use LVMs

USE WHEN:

- Lots of DISKs and you need to simplify
- Need flexiblity of file system expansion
- When you want to learn an advanced storage management topic

DON'T USE:

- SINGLE Partition Environments
- If Command line (CLI) is scary to you
- Without access to a System Administrator :-)

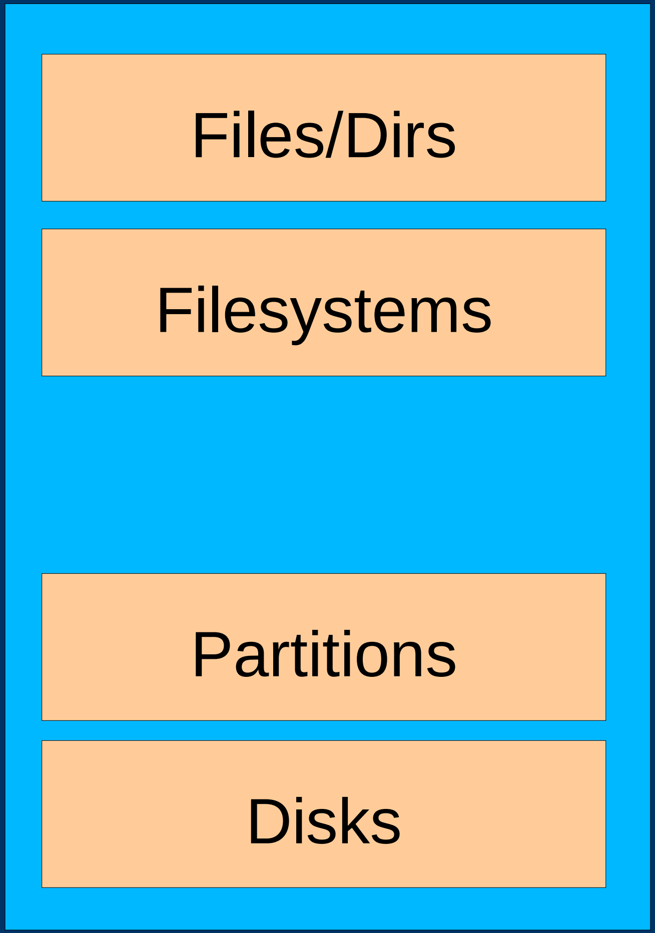
Background Info

Background

- A hard disk on a modern system can be seen as one continuous row of logical blocks.
- To store data on a disk this row needs to be cut in sections called partitions
- It can be
  - one huge partition covering the whole disk
  - several small partitions on one disk
  - or a combination over several disks.
- One partition must be one continuous chunk of blocks (and herein is part of the problem)
- Once you created a partition you have to live with it

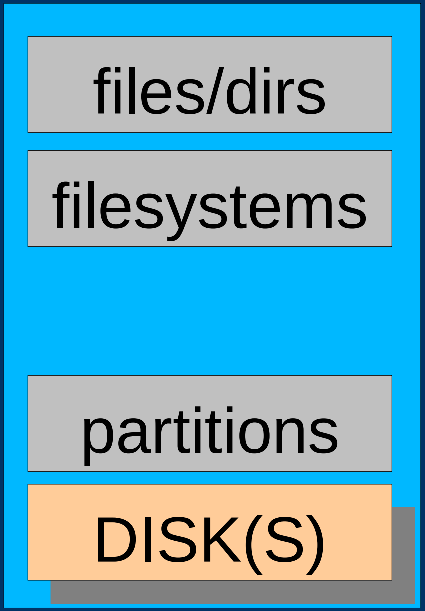
Layers to make it easier

Layers in a typical system (before LVM)



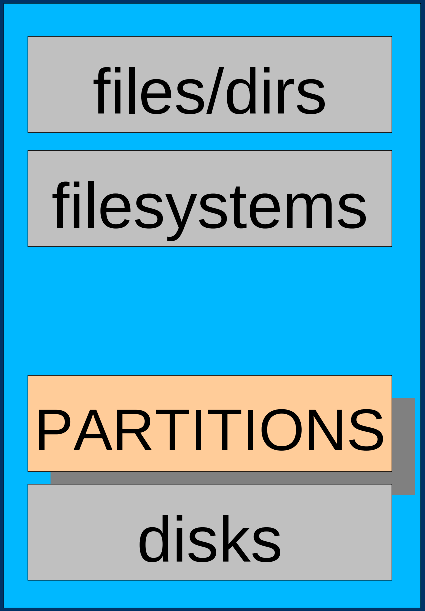
What is a Disk?

- Lets look at some terms first, starting from the bottom and working our way up
- The harddisk is the only Hardware piece we're going to talk about here
- It has various techie things in it:
  - Cylinders, Heads, Sectors, mbr, partitions
  - It's own cpu, cache, firmware etc
- but for this discussion a hard disk on a modern system can be seen as one continuous row of logical blocks.
- Typical disk names in Linux are hda, sda



Partitions

- To store data on a disk this continuous row of logical blocks needs to be cut in sections called partitions
- The original IBM PC from 1981 only had 4 primary partitions
  - On linux you can see them as hda1-hda4
- This was later improved by adding extended partitions
  - On linux they show up as partition hda5 and up
- Normally managed by fdisk or some graphical partition/volume manager
- **NOTE:** One partition must be a **continuous** chunk of blocks

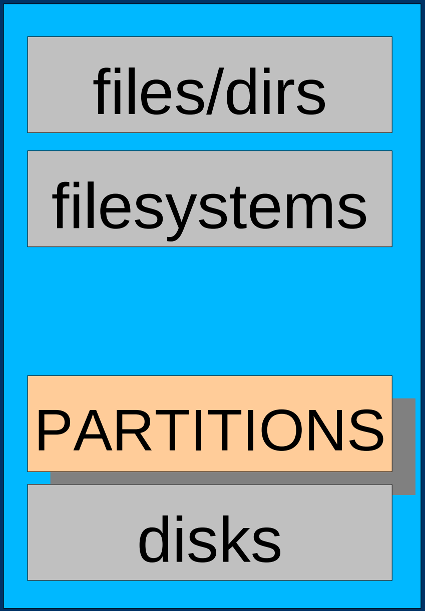


Partition tools

- I have to at least mention that it does exist tools that can change the existing partitions without loosing existing data.
  - I never tried this tools
  - They are all very risky
- parted
  - A free tool that can handle ext2 and vfat but not NTFS
- Partition magic
  - Commercial tool that can do more including NTFS
- Fips
- Linux-ntfs

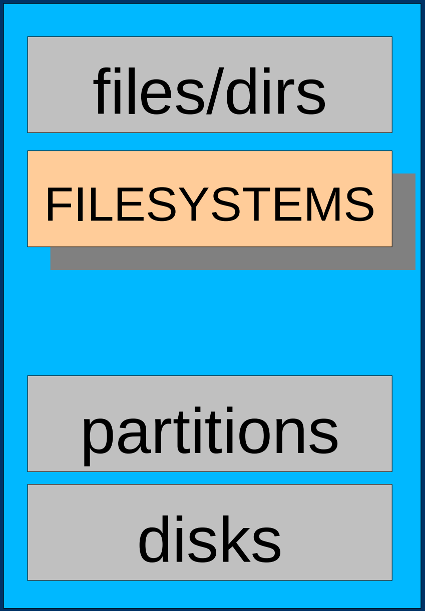
Partitions - cont

- Due to historical reasons partition sizes are based on Cylinders
- On a typical linux system you have something like
  - hda1 /
  - hda2 swap
  - hda3 /home
- Or for a dual boot
  - hda1 M$ Windows C:
  - hda2 swap
  - hda3 /
  - hda5 /home



Filesystems

- Filesystems are a fancy way to hold together a group of files and directories
- Without LVM one filesystem=one partition
- Common File Systems:
  - In the Windows World:
    - FAT (dos) FAT12/16/32
    - NTFS
  - In Linux
    - Ext2/3, Reiserfs,jfs,xfs
    - FAT/NTFS
    - And many many more

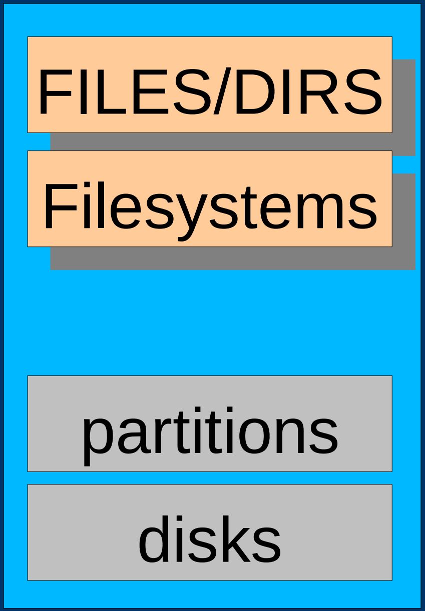


Example File systems & Partitions

```
# df
Filesystem              Size  Used Avail Use% Mounted on
/dev/hda1               510M  269M  242M  53% /
/dev/hda3               4.0G  351M  3.7G   9% /home
/dev/hda5               4.0G  2.1G  2.0G  52% /opt
/dev/hda6               4.0G  3.1G  976M  77% /usr
/dev/hda7               1.0G  121M  904M  12% /var
```

Files on a linux (and unix) system

- At last we are at the top, the user level
- Within User Level we have
  - Files
  - Directories
  - Filesystems
- Files Hold Data
- Directories hold files and directories
- Filesystems holds directories and files
- Mount points hold Filesystems



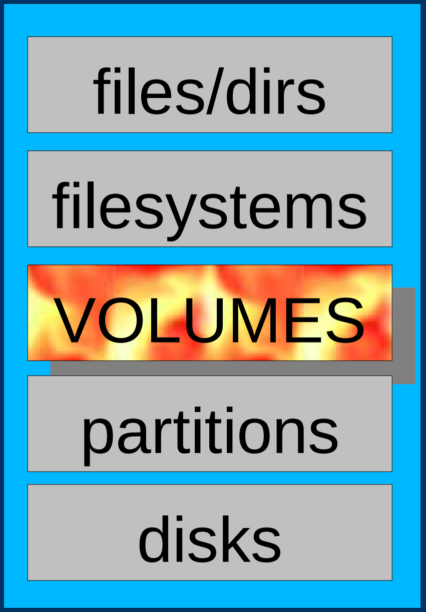
Intro to LVM

WHY Volume Management?

- Because
  - We can and it is there
- Actually
  - To make your life as System Administrator easier
- To give you a bigger Sandbox to play in
- Is Linux Volume Management Special?
  - NO :-)
  - A more advanced way of keeping track of data
  - Exists on other OSes also (even some versions of M$ win)
  - Commonly not used by default

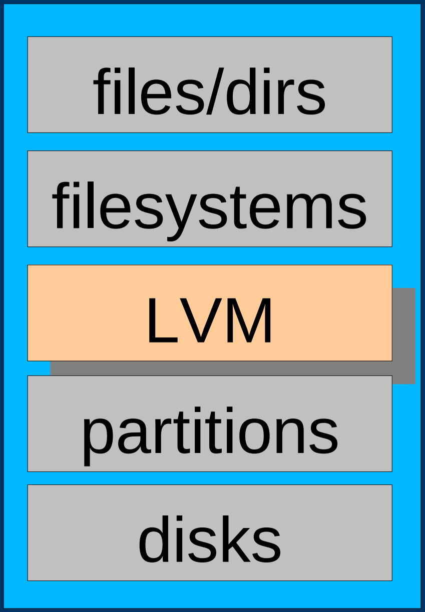
Time for volumes

- Time to fill in that hole in the middle
- By adding another layer between partitions and filesystems we break the OneFS=OnePartition relation
- This have many advantages
- 2 special reasons:
  - Can carve out non continues filesystems
  - Can add disks partitions together so they look like one big disk



What is LVM?

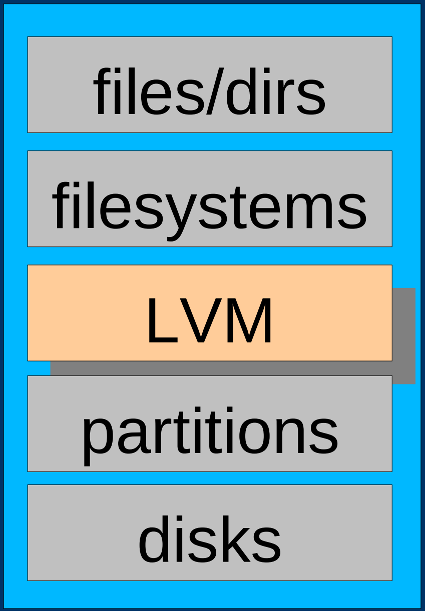
- LVM
  - stands for Logical Volume Management
- It's the Linux implementation of Volume Management
- It is placed between the filesystems and disk partitions



What is LVM? cont

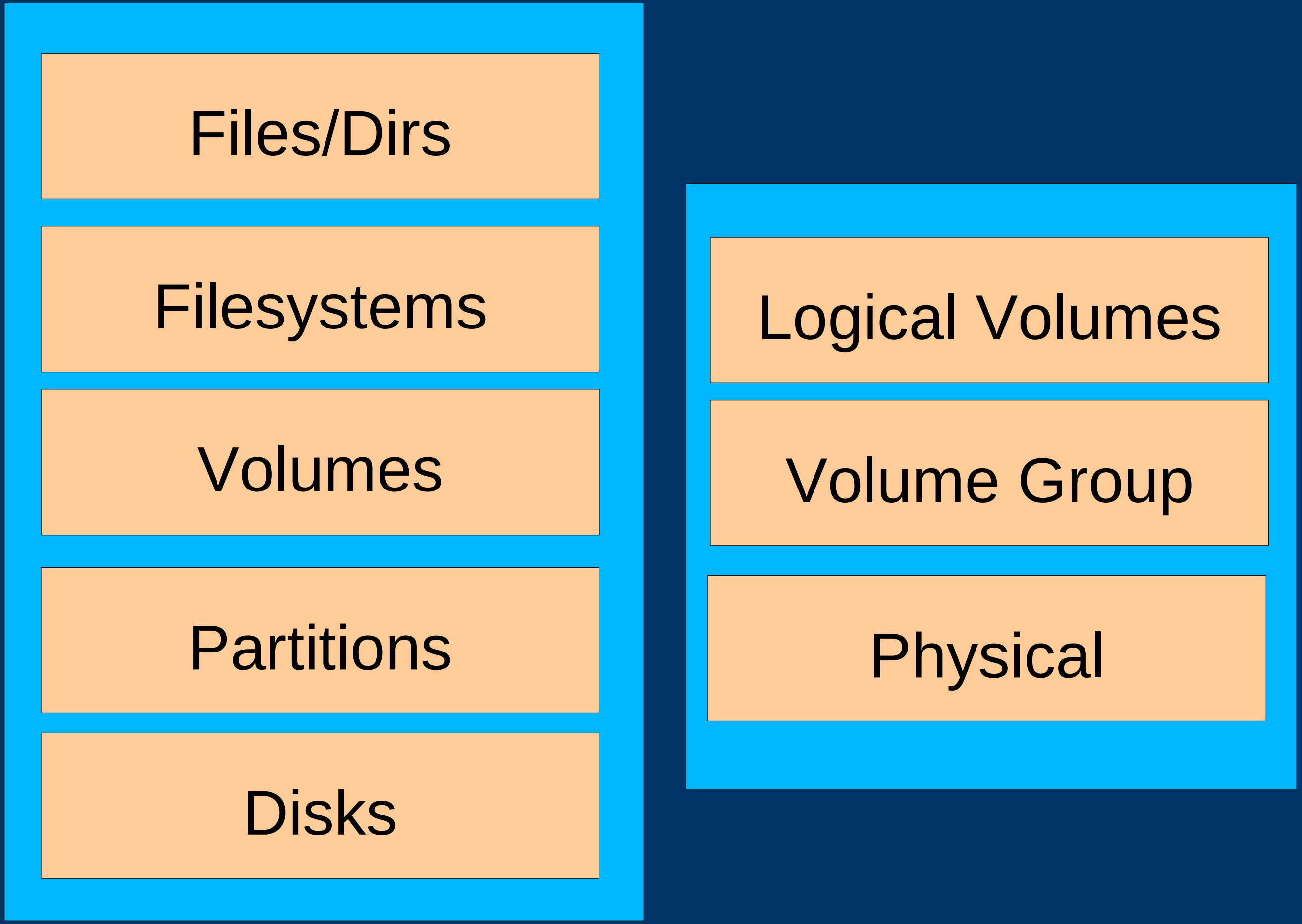
New Terms

- Physical Volumes PVs
  - collects all disk partitions
- Volume Group VGs
  - creates one big virtual disk
- Logical Volumes LVs
  - from the VG you can then create filesystems within LVs



LVM in Layers

- Here is a simplified picture of the LVM structure showing PV, VG and LG



LVM - cont

- LVM can be used to solve the previous problems with static partitions and many other problems also.
- With LVM it's no problem to reduce or extend an existing file system.
- In many cases you can even increase the size without dismounting the file system
  - Good when you discover that that 4.5G download won't fit
- Adding another disk is also an easy task
- Most benefits are on multi disk systems

How does LVM work?

- It collects all bits and pieces of diskspace and make a volume group (VG) of it
- From that volume group you then carve out a pile of blocks and create a logical volume (LV)
- The logical volume doesn't have to be a chunk of continues blocks, it doesn't even have to be on the same disk.
- LVM takes care of finding all the pieces of the cake and make it appear as one big cake to the next level (the filesystem)

LVM partitions

- Leo has a 100G disk
- He takes a safer route and makes 3 partitions
  - one smaller for / (which includes /boot, /dev, /etc, /lib...)
  - one swap
  - and one big LVM (AKA VG) of the rest of the disk.
- Out of the VG he
  - create LVs for /home, /usr, /var, /data, /tmp etc.
  - The created LVs are not using all the space in the LV, some is left as spare
  - As you soon see leaving some unused is a good thing

LVM partitions - cont

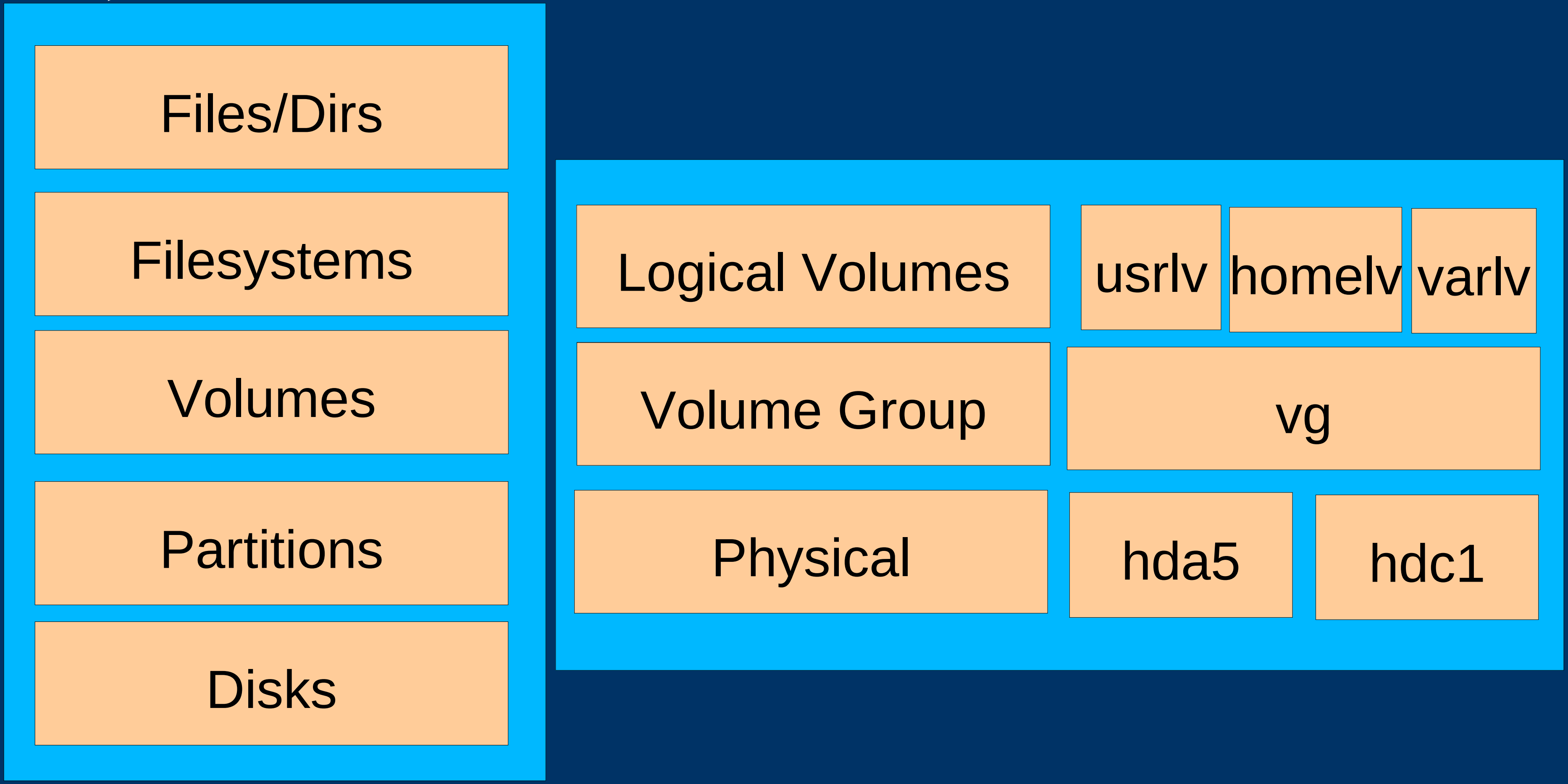
- After a while Leo discovers that /usr is now full so he can't install the latest OpenOffice/KDE/Gnome package.
  - Since it's space left in the VG he executes some commands and voila! – /usr is bigger
- Later he needs to expand /data but by now all free VG space is used up
- Some space is left in /home so he
  - Use some lvm magic to reduce the size of /home
  - Use more lvm magic to use the recovered space to increase the size of /data
- Life goes on

LVM partitions - cont

- After a while Leo needs some more space and buys another hard disk
- With just 2 LVM commands the new disk is added to the total pool of VG space to be used as needed
- He can now expand any LV as needed until that disk is used up
  - At what point he can keep on adding disks as long as the hardware supports it
- Life is good

LVM example

- Here is a simplified picture of the LVM structure showing PV, VG and LG



Df with lvm

```
$ df
Filesystem            Size  Used Avail Use% Mounted on
/dev/hda1              4.0G  2.4G  1.7G  60% /
/dev/hda2              46M   10M   34M  24% /boot
/dev/mapper/vg-homelv
                       95G   95G  774M 100% /home
/dev/mapper/vg-usrlv
                      7.0G  6.1G  943M  87% /usr
/dev/mapper/vg-varlv
                      2.0G  1.6G  486M  77% /var
```

How do I get the LVM joy?

- Many distros gives the option use LVM and some even does it by default (FC4 is one).
  - Type “mount”
  - Do you see something like “/dev/**mapper**/*something*”
    - If yes; Good – you're already using it
    - If no; follow closely
- When you install a new system it can be implemented when partitioning the disk.
  - Probably need to select manual/custom/expert and then look for something with LV when partitioning the disk.
- Be careful with putting /boot or / on LVM, all distro versions can't handle it.

How do I get tools?

If you already have a system running but no volume groups you may have to install the tools first

- Mandrake:
  - urpmi lvm2
- Debian:
  - apt-get install lvm2
- Fedora:
  - yum install lvm2
- Gentoo:
  - emerge lvm2 (?)
- Suse:
  - yast -i lvm2 (?)

How do I get the LVM joy? - cont

- Once the tools are there next problem is to find somewhere to play.
- You need a partition to convert to LVM
  - New disk
  - Unused partition (that old Windows partition works fine)

Your first LV

- After identifying a partition you need to prepare it
- Here are some examples with /dev/hdc1 as the target
  - With some tool, change the partition type to 0x8e or LV
  - Prepare it for lvm with:
    - pvcreate /dev/hdc1
  - Create a new volume group out of it
    - vgcreate vg /dev/hdc1
  - Create a Logical Volume AKA filesystem
    - lvcreate -L10G –name datalv vg

```
[root@dhcp132 ~]# pvcreate /dev/hdc1
...
[root@dhcp132 ~]# vgcreate vg /dev/hdc1
...
[root@dhcp132 ~]# lvcreate -L10G -name datalv vg
```

Your first LVM - cont

- Now you have a new device in
  - /dev/vg/datalv
  - This is a symbolic link that points to /dev/mapper/vg-datalv
  - The 'mapper' part is there as of version2 of LVM.
  - Don't worry about it, just accept it
    - or spend some hours googling about the new device mapper
- This device is very much like a normal partition like the usual /dev/hda1
  - It's just much easier to change the size of it

Your first LVM - cont

- Now when you have this device it's time to start using it.
  - Format the filesystem
    - mkfs -t reiserfs /dev/vg/datalv
  - Mount it somewhere
    - mkdir /data
    - mount /dev/vg/datalv /data
- That's it, you have started to use LVM

```
[root@dhcp132 ~]# mkfs -t reiserfs  /dev/vg/datalv
....
[root@dhcp132 ~]# mkdir /data

[root@dhcp132 ~]# mount /dev/vg/datalv /data
```

Advanced LVM

First LVM Done

- Of course just creating a filesystem was the least interesting part
  - So lets move on to the more interesting part
- Scenario
  - You have both usr and data on LVM
  - You need to expand usr
  - It is free space in data
- When dealing with LVM you have to remember that it's two layers involved here
    - The layer under the filesystem previously known as partition
    - The filesystem it self

Filesystem/LV/partitions

- When you create a filesystem with mkfs it automagically makes a filesystem of the same size as the partition it's created on
  - No surprise here
- Since with LVM the underlaying partition can change size we have to make the filesystem match it
  - Having the real filesystem smaller then the partition
    - No problem
  - Having the real filesystem bigger then the partition
    - Huge problem
    - Probably loose data
    - Maybe even corrupt other filesystems

Reduce

- In order to expand /usr you need to have some free space in your VG
- You can see how much is free with vgdisplay

```
[root@dhcp132 ~]# vgdisplay | grep Free
  Free  PE / Size        44 / 1.38 GB
```

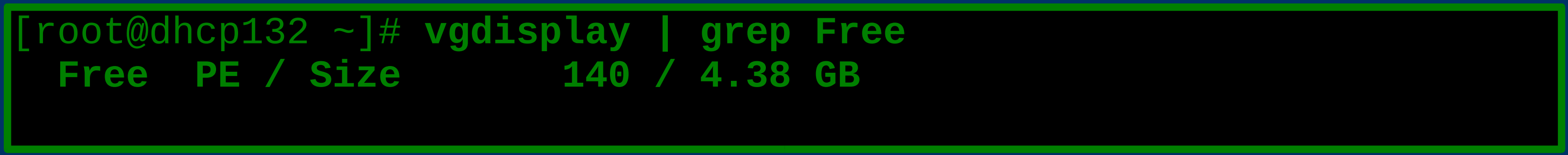
- Ops, only 1.38G free, and we needed 3G
- Now we have to start with reduce something else like datalv

Reduce – cont

- In all cases you have to umount the filesystem before this can be done so
  - umount /data
- Step 1, reduce the filesystem it self with 3G
  - Reiserfs
    - resize_reiserfs -s-3G /dev/vg/datalv
  - ext2/ext3
    - resize2fs /dev/vg/datalv 7G
      - Sorry, no “-nG” here, you have to calculate your self
  - xfs/jfs
    - There is (currently?) no way to shrink this

Reduce - cont2

- Step 2, reduce the logical volume
  - lvreduce -L-3G /dev/vg/datalv
- Last step, remount it
  - mount /data



```
[root@dhcp132 ~]# vgdisplay | grep Free
  Free  PE / Size       140 / 4.38 GB
```

- And now you have 3G more free space in your VG and this without to much pain

Expand

- Now when we have the space, lets expand /usr
- This is done in the reverse order, first LV then FS
- Step1, increase the LV
  - lvresize -L+3G /dev/vg/usrlv
- Step2, increase the filesystem
  - Reiserfs:
    - resize_reiserfs /dev/vg/usrlv
  - Xfs
    - xfs_growfs /usr
  - Jfs
    - mount -o remount,resize /usr

Expand - cont

- Ext2/3
  - Must be umounted first and for /usr that can be tricky but
    - umount /usr
    - resize2fs /dev/vg/usrlv
    - mount /usr
- And we are done
  - We reduce the size used by /data
  - And used that to expand /usr

Even more Advanced LVM

What's next?

- So, now you seen that the filesystem size can be changed without to much problem
  - What's next?

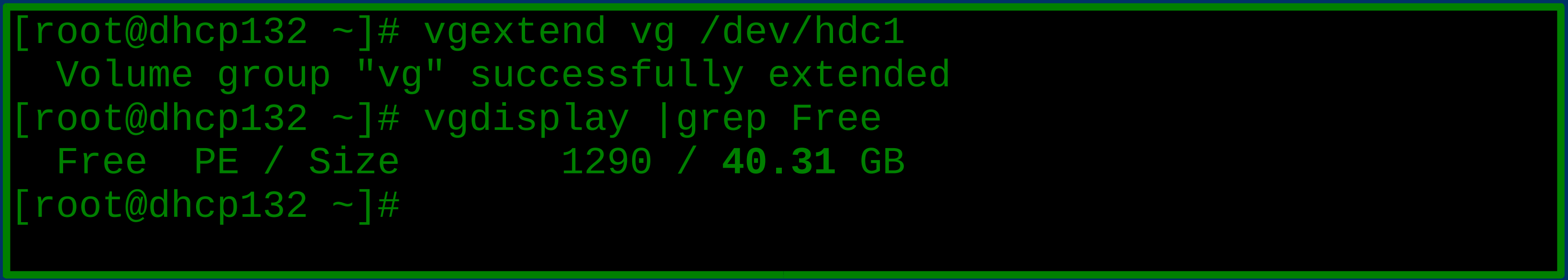
- How about
  - expanding /data again
  - but this time by adding a disk

Adding a disk

- For that we first need to physically install the disk first
- Once it's installed we partition it as one big LV partition
  - echo "/dev/hdc1:start=63,size=,Id=8e"|sfdisk /dev/hdc
- Great, now we have a disk partitioned and ready for LVM

Add disk

- Lets initialize it
  - pvcreate /dev/hdc1
- And then we add it to the total VG pool
  - vgextend vg /dev/hdc1
- Verify how much is free now



```
[root@dhcp132 ~]# vgextend vg /dev/hdc1
  Volume group "vg" successfully extended
[root@dhcp132 ~]# vgdisplay |grep Free
  Free  PE / Size       1290 / 40.31 GB
[root@dhcp132 ~]#
```

- More then enough for what we need
- Expanding /data in the same way we expanded /usr
- DONE

The END

QUESTIONS?

References

http://tldp.org/HOWTO/LVM-HOWTO

Google for “LVM HOWTO”

Slides can be found at

http://www.techwiz.ca/~peters/presentations/lvm

Snapshots

- Snapshots have at least two functions
- #1 What if we could “freeze” the current state for that database volume and back it up while the DB is still up
  - This is called ReadOnly snapshot
  - Works fine to get one volume in a consistent state
- #2 I would like to test this new app. It's going to write a pile of stuff somewhere but then I want to forget that and repeat the test
  - This is ReadWrite snapshot
  - Works for many other applications also

Snapshots - cont

- #2 is also excellent for Virtual Machines
  - You have one base os install
  - You create several snapshots of that base os
    - One for eache Virtual Machine
  - Any file that gets changed is stored in the individual VMs snapshot copy
  - Unchanged files are only stored once
    - Saving overall space

Create a snapshot device

- Lets create a snapshot volume of the datalv
  - lvcreate -L512M -s –name databackuplv /dev/vg/datalv
- Since it's a copy of an existing volume it doesn't need to be initialized, you just to mount it
  - mount /dev/vg/databackuplv /backup/data
- Run the backup
  - fancybackupsw /backup/data
- Remove the snapshot
  - umount /backup/data
  - lvremove /dev/vg/databackuplv

LVM Commands, PV

- Here are some of the PV commands
  - pvscan - scan all disks for physical volumes
  - pvdisplay - display attributes of a physical volume
  - pvcreate - initialize a disk or partition for use by LVM
  - pvchange - change attributes of a physical volume
  - pvmove – move physical extents from one disk to another disk

LVM Commands, VG

- Here are some of the VG commands
  - vgscan - scan all disks for volume groups
  - vgrename - rename a volume group
  - vgremove - remove a volume group
  - vgreduce - reduce a volume group
  - vgmerge - merge two volume groups
  - vgextend - add physical volumes to a volume group
  - vgdisplay - display attributes of volume groups
  - vgcreate - create a volume group

LVM Commands, LV

- Here are some of the LV commands
  - lvscan - scan (all disks) for logical volumes
  - lvrename - rename a logical volume
  - lvremove - remove a logical volume
  - lvreduce - reduce the size of a logical volume
  - lvextend - extend the size of a logical volume
  - lvdisplay - display attributes of a logical volume
  - lvcreate - create a logical volume in an existing volume group

vgdisplay

```
[root@dhcp132 ~]# vgdisplay
    Finding all volume groups
    Finding volume group "VolGroup00"
  --- Volume group ---
  VG Name               VolGroup00
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  4
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                2
  Open LV               2
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               9.88 GB
  PE Size               32.00 MB
  Total PE              316
  Alloc PE / Size       144 / 4.50 GB
  Free  PE / Size       172 / 5.38 GB
  VG UUID               xFS58l-ZXRS-vBaG-xD64-sbhn-7bod-4rc13o
```

lvdisplay -v

```
[root@dhcp132 ~]# lvdisplay /dev/VolGroup00/LogVol00
  --- Logical volume ---
  LV Name                /dev/VolGroup00/LogVol00
  VG Name                VolGroup00
  LV UUID                UMQ8ro-nTt6-p5PA-IsXN-JC6n-JuaE-APWYiO
  LV Write Access        read/write
  LV Status              available
  # open                 1
  LV Size                4.00 GB
  Current LE             128
  Segments               1
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:0
```

Don't use /dev/mapper

```
 [root@dhcp132 ~]# lvdisplay -v /dev/mapper/VolGroup00-datalv
    Using logical volume(s) on command line
    Wiping cache of LVM-capable devices
  Volume group "mapper" not found

[root@dhcp132 ~]# lvdisplay -v /dev/VolGroup00/datalv
    Using logical volume(s) on command line
  --- Logical volume ---
  LV Name                /dev/VolGroup00/datalv
  VG Name                VolGroup00
  LV UUID                tSJfYV-rzy6-fynl-vCql-WGRs-MFF6-lM3PEf
  LV Write Access        read/write
  LV Status              available
  # open                 0
  LV Size                2.00 GB
  Current LE             64
  Segments               2
  Allocation             inherit
  Read ahead sectors     0
  Block device           253:2
```

pvdisplay

```
[root@dhcp132 ~]# pvdisplay
  --- Physical volume ---
  PV Name               /dev/hda2
  VG Name               VolGroup00
  PV Size               9.88 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       32768
  Total PE              316
  Free PE               12
  Allocated PE          304
  PV UUID               uKwIoE-NcLK-lEqF-Zyuf-bq4i-6HTv-gXqHgK

  --- Physical volume ---
  PV Name               /dev/hdc1
  VG Name               VolGroup00
  PV Size               39.97 GB / not usable 0
  Allocatable           yes
  PE Size (KByte)       32768
  Total PE              1279
  Free PE               1279
  Allocated PE          0
  PV UUID               Diy7y9-RfoX-11CZ-0dtt-S4ke-Hoei-GwSjHa
```

How to do it under Fedora Core 4

- This is as close as it comes to a demo
  - Since FC4 use LVM by default it seems to be the best candidate for a scary test
  - The default install (Personal WS) use one big root partition covering the whole disk
  - To change that a little you need to reduce it's size
- Be prepared to loose everything
  - as with all disk operations, make sure you have a good backup
- Check sizes
  - df -h can be used to see how much space is used
    - Should probably print out the output of this

“Demo”

- Steps to reduce it
  - Boot on the cd and type “linux rescue” to get to rescue mode
  - When asked about scan for partitions, select “skip”
- Now you're at a position where you can do lots of damage so be careful
  - Scan and activate all logical volumes
    - lvm.static vgscan
    - lvm.static vgchange -ay
    - lvm.static vgdisplay

Demo 2

- Now all devices are listed under /dev/mapper
  - “ls /dev/mapper” to verify that it's there
- Lets reduce the size of the root partition.
- Assuming the “df -h” before indicated that we would get away with only 4G you can do

```
#resize2fs /dev/VolGroup00/LogVol00 4G
Resizing the filesystem on /dev/VolGroup00/LogVol00 to 1048576 (4k) blocks.
The filesystem on /dev/VolGroup00/LogVol00 is now 1048576 blocks long.

#lvm.static lvreduce -L4G /dev/VolGroup00/LogVol00
  WARNING: Reducing active logical volume to 4.00 GB
  THIS MAY DESTROY YOUR DATA (filesystem etc.)
Do you really want to reduce LogVol00? [y/n]:
Reducing logical volume LogVol00 to 4.00 GB
  Logical volume LogVol00 successfully resized
```

Demo 3

- Now root's size is reduced
- Reboot into normal mode
- You have a VG that has lots of free space so you can
  - Create partitions for /home. /var, /usr etc
    - And move over the data to create a multi partition system

Example time

- Now we have gone over the basics of a basic system
- Next are two examples to show some of the issues with the normal/default way of creating filesystems and partitioning.

Multiple partitions

- Marie has a 100G disk
- She makes several partitions
  - /boot
  - Swap
  - /
  - /var
  - /usr
  - /home
  - /data
  - ...
- She has problems guessing the size of each partition

Multiple partitions - cont

- After a while she discovers that /usr is now full so she can't install the latest OpenOffice/KDE/Gnome package.
- There is some space left in /data but that's not where she needs it
- Her options are limited
  - Reformat the hard disk and start over with different sizes
    - And hit the same problem later
  - Have fun with symlinks
    - And loose track of where data is actually stored
  - Expanding the total diskspace by adding another disk
    - this is anything but simple and you may end up with another symlink blues

Single partition

- Sven also has one 100G hard disk
- To avoid any issues with filesystem filling up he makes one huge partition of the whole hard disk.
  - swap
  - /   everything
- This gives room for all the stuff no matter if mp3, video or programs ends up taking most space.
- Life is good or?

Single partition - cont

- Some of the problems with this are
  - Hard to catch if something like /var/log, /var/spool or /tmp is growing out of control.
  - After a bad crash it takes “forever” to fsck
  - A user application (running amok) can easily fill up the disk and cripple/block important system functions
  - If the filesystem gets to corrupt to recover you have no fence and loose everything
- Expanding the total diskspace by adding another disk is anything but simple
  - Same problems as with multi partitions

Copyright & License