



LVM & software RAID

under Linux

GNS Systems GmbH
Am Gaußberg 2
38114 Braunschweig / Germany

Tue 23 Mar 2010, Revision 0.1

Please send comments about this document to: norman.krebs@gns-systems.de

Contents

1	Preface	3
2	Logical Volume Manager - LVM	4
2.1	Create a Logical Volume	5
2.2	Resize or Extend a Logical Volume	6
2.3	Mount a Foreign Logical Volume	7
2.4	Destroy a LVM	8
2.5	Move or Modify a LVM	8
2.5.1	Remove a Physical Volume	8
2.5.2	Rename items	8
2.5.3	Make a snapshot	8
2.5.4	Move a LVM to another machine	9
2.6	LVM commands	10
2.6.1	Physical Volumes	10
2.6.2	Volume Groups	10
2.6.3	Logical Volumes	11
2.6.4	Outputs	11
3	RAID	13
3.1	RAID levels	13
3.2	Notes	14
3.3	Performance issues	14
3.4	Create a RAID	15
3.5	Test, check and manage the RAID	16

4 Appendix	17
4.1 URLs and Reference	17
4.2 Sample setup	18

1 Preface

This document denotes in short steps how to handle the logical volume manager (LVM) and software RAID under Linux. It is not a description nor intended to be complete in any way.

2 Logical Volume Manager - LVM

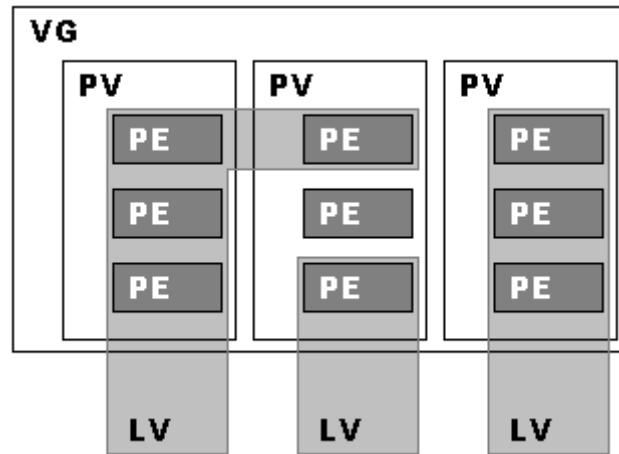


Figure 1: LVM - overview

PV physical volume == Linux block device
PE physical extent
VG volume group
LV logical volume - can contain a file system

LVM has the following design ideas:

- map physical to logical volumes
- deliver volume sizes independent from the sizes of the underlying hard drives
- no redundancy - (it is not a RAID!)
- no additional CPU load

2.1 Create a Logical Volume

Look at the section 2.6.4 for some sample outputs. The most commands accept more than one device, p.e.

```
# pvcreate /dev/sdc1 /dev/sdd1
```

The device names are not consistent here, but this doesn't matter.

1. install lvm2

```
# apt-get install lvm2
# rpm -ivh lvm2-2.02.21-5.el4.x86_64.rpm (the exact version doesn't matter)
# yum install lvm2
```

2. create partitions

```
/dev/sdc1    /dev/sdd1
```

3. initialize a Physical Volume (PV) and show the physical volumes:

```
# pvcreate /dev/sdc1 ; pvdisplay
```

4. set the partition type to "Linux LVM": "8e"

5. create a new Volume Group (VG) named "vg1" with a Physical Extend (PE) size of 32 Mb and assign the Physical Volume /dev/sdc1 to it and show the volume groups

```
# vgcreate -s 32M vg1 /dev/sdc1 ; vgdisplay
```

or, if you want to include all PVs at once:

```
# vgcreate -s 32M vg1 /dev/sdc1 /dev/sdd1 ; vgdisplay
```

6. create a new Logical Volume (LV) with name "lv1" 7.4 GB size in the Volume Group "vg1" and show it

```
# lvcreate -L 7.4G -n lv1 vg1 ; lvdisplay /dev/vg1/lv1
```

or assign the amount of extends as seen in "vgdisplay" -> "Free PE"

```
# lvcreate -l 236 -n lv1 vg1 ; lvdisplay /dev/vg1/lv1
```

Create a new Logical Volume spanned over two Physical Volumes in vg1 - a RAID 0

```
# lvcreate -n lvstriped -l 236 -i 2 vg1
```

7. create a file system on the Logical Volume:

```
# mkfs.ext3 /dev/vg1/lv1
```

2.2 Resize or Extend a Logical Volume

Two notes:

- (A) File systems cannot be modified when they are in use. Boot into single user mode before changes or reboot the machine without the file system mounted. Don't forget to modify `/etc/fstab` and reboot the machine finally.
- (B) If (A) is not possible, keep in mind: NFS caches file system sizes. Restart NFS after changes and make an `exportfs -r`.

The procedure:

1. create a new Physical Volume

```
# pvcreate /dev/sdd1 ; pvdisplay
```

2. enhance the existing Volume Group "vg1" and check the amount of new Physical Extends (p.e.: 430)

```
# vgextend vg1 /dev/sdd1 ; vgdisplay
```

3. assign the new, free Physical Extends to the logical volume "lv1"

```
# lvextend -l 430 /dev/vg1/lv1 ; lvdisplay
```

or assign the (additional) amount of space

```
# lvextend -L +1000M /dev/vg1/lv1 ; lvdisplay
```

4. resize the file system on the logical volume

- file system: ext2/ext3

```
# umount /dev/vg1/lv1 ; resize2fs -p /dev/vg1/lv1
```

- file system: XFS

```
# mount /dev/vg1/lv1 /mnt/vg1/lv1 ; xfs_growfs /mnt/vg1/lv1
```

2.3 Mount a Foreign Logical Volume

Assume a system crash and boot a live medium, p.e. the grml-CD (<http://grml.org>).

1. check the available physical volumes

```
# pvs
  PV          VG  Fmt Attr PSize PFree
  /dev/sda1   vg1 lvm2 a-  74.50G  0
  /dev/sdb1   vg1 lvm2 a-  74.50G  0
```

2. check the logical volume

```
# lvdisplay vg1
--- Logical volume ---
LV Name            /dev/vg1/lv1
VG Name            vg1
LV UUID            mmv77R-zrhD-Hu02-CvQq-piys-FFNL-swiNB2
LV Write Access    read/write
LV Status          available
# open             0
LV Size            149.00 GB
Current LE         4768
Segments          2
Allocation         inherit
Read ahead sectors auto
 - currently set to 256
Block device       253:0
```

we have now the device name: /dev/vg1/lv1

3. start the LVM if there is some kind of init script

```
# /etc/init.d/lvm2 start
```

4. or start it manually. Try these:

```
# modprobe dm-mod ; modprobe lvm-mod
# /sbin/vgscan --ignorelockingfailure --mknodes
# /sbin/vgchange -aly --ignorelockingfailure
# mkdir -p /mnt/vg1/lv1 ; mount /dev/vg1/lv1
```

5. stop it after unmounting:

```
# umount /dev/vg1/lv1
# /sbin/vgchange -aln --ignorelockingfailure
```

2.4 Destroy a LVM

```
# lvremove -f /dev/vg1/lv1  
  
# vgremove /dev/vg1  
  
# pvremove /dev/sde1 /dev/sdg1
```

2.5 Move or Modify a LVM

2.5.1 Remove a Physical Volume

Move the data from one Physical Volume away to remove the PV. There must be still free Extends in the volume group.

```
# pvmove -v /dev/sde1
```

or move them onto another Physical Volume:

```
# pvmove -v /dev/sde1 /dev/sdf1
```

and take out the cleaned Physical Volume from the Volume Group:

```
# vgreduce vg1 /dev/sde1
```

2.5.2 Rename items

```
# vgrename /dev/vg1 /dev/vlgrp1  
  
# lvrename /dev/vlgrp1/lv1 /dev/vlgrp1/lgv11
```

2.5.3 Make a snapshot

```
# lvcreate -L 500M --snapshot -n snapshot1 /dev/vg1/lv1
```

The 500Mb is the amount of changed data possible until the snapshot has to be neglected.

2.5.4 Move a LVM to another machine

Use drives containing a LVM in another machine:

old machine:

```
# pvscan
# umount /mnt/vg1/lv1
# vgchange -a n /dev/vg1
# vgexport /dev/vg1
```

new machine:

```
# vgimport vg2 /dev/hdb5 /dev/hdb6
# vgchange -a y /dev/vg2
# mkdir /mnt/vg1/lv1
# mount -t ext2 /dev/vg2/lv1 /mnt/vg1/lv1
```

2.6 LVM commands

lvm	interactive command line tool for all lvm functions
lvmdiskscan	scan data storage devices for PVs

2.6.1 Physical Volumes

pvcreate	initialize a harddrive or partition for LVM
pvdisplay	show properties of a PV
pvchange	change properties of a PV
pvremove	delete a PV
pvmove	move the content of a PV into another PV
pvscan	scan all harddrives for PVs
pvs	report information about PVs

2.6.2 Volume Groups

vgcreate	create a VG
vgdisplay	show properties of a VG
vgs	report information about VGs
vgchange	change properties of a VG
vgremove	delete a VG
vgextend	add PVs to a VG
vgreduce	reduce the size of a VG
vgrename	rename a VG
vgexport	log off a VG from the system for transfer to other machine(s)
vgimport	log on an imported VG
vgsplit	split a VG p.e. for export/import
vgmerge	concatenate two VG
vgcfgbackup	save VG-descriptors into /etc/lvm/backup
vgcfgrestore	restore VG-descriptors from /etc/lvm/backup
vgck	check meta-data of a VG
vgconvert	convert meta-data-format from lvm1 to lvm2
vgmknodes	restore the VG-directory and the needed device-files
vgscan	scan all harddrives for VGs

2.6.3 Logical Volumes

lvcreate	create a LV in an existing VG
lvdisplay	show properties of a LV
lvchange	change properties of a LV
lvremove	delete a LV
lvextend	extend the size of a LV
lvreduce	reduce the size of a LV
lvresize	change the size of a LV
lvrename	rename a LV
lvscan	scan all harddrives for LVs
lvs	report information about LVs

2.6.4 Outputs

```
# pvdisplay
--- Physical volume ---
PV Name           /dev/hde1
VG Name           vg1
PV Size           74.53 GB / not usable 31.06 MB
Allocatable       yes (but full)
PE Size (KByte)   32768
Total PE          2384
Free PE           0
Allocated PE      2384
PV UUID           cEbIFe-zoEn-WWtX-I9V8-jE3m-GNgj-IIoYvi

--- Physical volume ---
PV Name           /dev/hdg1
VG Name           vg1
PV Size           74.51 GB / not usable 5.46 MB
Allocatable       yes (but full)
PE Size (KByte)   32768
Total PE          2384
Free PE           0
Allocated PE      2384
PV UUID           TAECKa-CHdC-3N46-CIKF-9muR-qoDI-Yz2s2T
```

```
# vdisplay
```

```
--- Volume group ---
VG Name          vg1
System ID
Format           lvm2
Metadata Areas   2
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          2
Act PV           2
VG Size          149.00 GB
PE Size          32.00 MB
Total PE         4768
Alloc PE / Size  0 / 0
Free PE / Size   4768 / 149.00 GB // free physical extends
VG UUID          KDXvtE-ID6R-Y2Fg-chg6-OPU1-KjDz-NzMtjI
```

```
# lvdisplay
```

```
--- Logical volume ---
LV Name          /dev/vg1/lv1
VG Name          vg1
LV UUID          GbrEW3-ICGL-Rysl-9rmB-spIm-1gQE-7kWfeK
LV Write Access  read/write
LV Status        available
# open           1 // File system mounted: 1
//              // not mounted: 0

LV Size          7.4 GB
Current LE       236 // logical extends
Segments        2
Allocation       inherit
Read ahead sectors 0
Block device     253:0 // major/minor
//              // block device number
```

3 RAID

3.1 RAID levels

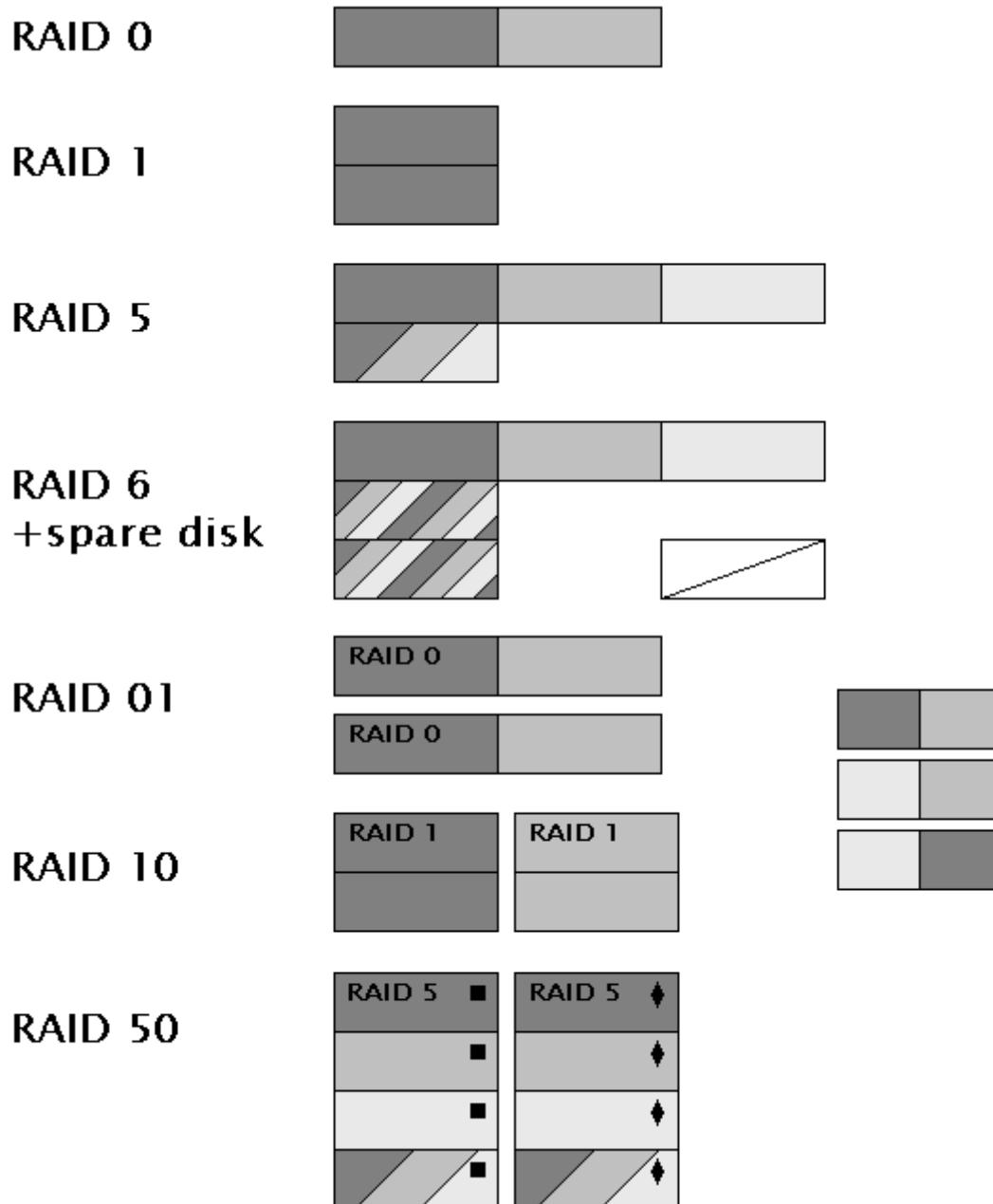


Figure 2: RAID levels, but a very small selection only

3.2 Notes

RAID has the following design parameters:

- improve performance or data safety
- be cheap (hardware RAID isn't)
- deliver devices bigger than the underlying hard drives

The most more complicated RAID Levels are available in hardware only. Linux software RAID supports at least the levels: 0, 1, 5, 6 and a few more.

3.3 Performance issues

This represents ideal values not cared about bottlenecks caused by other components (CPU, RAM, BUS etc.)

RAID level	0	1	10/01	2	3	4	5	6	Z
number of drives	$N > 1$	$N = 2$	$N > 2$	$N = 10$	$N > 2$	$N > 2$	$N > 2$	$N > 3$	$N < 10$
redundant drives	0	1	1**	2	1	1	1	2	1
capacity overhead [%]	0	50	50	20	$\frac{100}{N}$	$\frac{100}{N}$	$\frac{100}{N}$	$\frac{200}{N}$	$\frac{100}{N}$
parallel reads	N	2	$\frac{N}{2}$	8	$N - 1$	$N - 1$	$N - 1$	$N - 2$	1
parallel writes	N	1	1	1	1	1	$\frac{N}{2}$	$\frac{N}{3}$	1
read throughput	N	2	$\frac{N}{2}$	8	$N - 1$	$N - 1$	$N - 1$	$N - 2$	1
write throughput	N	1	1	1	1	1	$\frac{N}{2}$	$\frac{N}{3}$	1

(*) factor compared with a single drive

(**) worst case; best case: $n/2$ drives can fail without data loss

3.4 Create a RAID

1. install the raid utils: mdadm

```
# apt-get install mdadm
# rpm -ivh mdadm-1.12.0-2.x86_64.rpm (the exact version doesn't matter)
# yum install mdadm
```

2. take two identical hard drives: /dev/sdb and /dev/sdc and prepare them

- `dd if=/dev/zero of=/dev/sdb count=1 bs=512`
- `# fdisk /dev/sdb`
 - `n` create a new partition - default values: one partition per disk
 - `t` set the RAID-signature: `fd`
 - `w` write the changes to disk and exit

and repeat this step for the second disk

3. create the RAID

- make sure that the RAID daemons are started. Under Debian look for `/etc/default/mdadm` and look for `AUTOSTART` - however you'll see if the RAID isn't up yet.
- **RAID-0: *stripe***
`# mdadm -C /dev/md0 -a yes -l 0 -n 2 /dev/sdb1 /dev/sdc1`
- **RAID-1: *mirror***
`# mdadm -C /dev/md0 -a yes -l 1 -n 2 /dev/sdb1 /dev/sdc1`
- **RAID-5: *stripe with one parity disk***
`# mdadm -C /dev/md0 -a yes -l 5 -n 3 /dev/sdb1 /dev/sdc1 /dev/sdd1`
- **RAID-5+1: *stripe with one parity disk and one spare disk***
`# mdadm -C /dev/md0 -a yes -l 5 -n 5 -x 1 /dev/sd{b,c,d,e,f}1`

4. fix the configuration

- `/etc/mdadm/mdadm.conf:`
`DEVICE /dev/sdb1 /dev/sdc1`
- `# mdadm -Dvb /dev/md0 >> /etc/mdadm/mdadm.conf`
`-Dvb: --brief --detail --verbose`

5. finish

- `# mkfs.ext3 /dev/md0`
- `/etc/fstab:`
`/dev/md0 /usr2 ext3 defaults 0 0`
- `# /sbin/reboot`
- at boot time:
`mdadm --assemble /dev/md0 /dev/sdb1 /dev/sdc1`
`Debian: /etc/default/mdadm AUTOSTART=true`

3.5 Test, check and manage the RAID

- monitor with:
 - # mdadm; dmesg
 - # cat /proc/mdstat
- simulate a disk failure:
 - # mdadm /dev/md0 -f /dev/sdb1
- remove a drive (p.e. to boot after a disk failure)
 - # mdadm /dev/md0 -r /dev/sdb1
- restore the RAID (p.e. after exchanging the broken drive /dev/sdb):
 - # mdadm /dev/md0 -a /dev/sdb1
- disassemble a RAID:
 - # mdadm -S /dev/md0

4 Appendix

4.1 URLs and Reference

<http://tldp.org/guides.html>

<http://linuxwiki.de/mdadm>

http://www.brandonhutchinson.com/Mounting_a_Linux_LVM_volume.html

<http://www.linuxhaven.de/dlhp/HOWTO/DE-LVM-HOWTO-{1..7}.html>

4.2 Sample setup

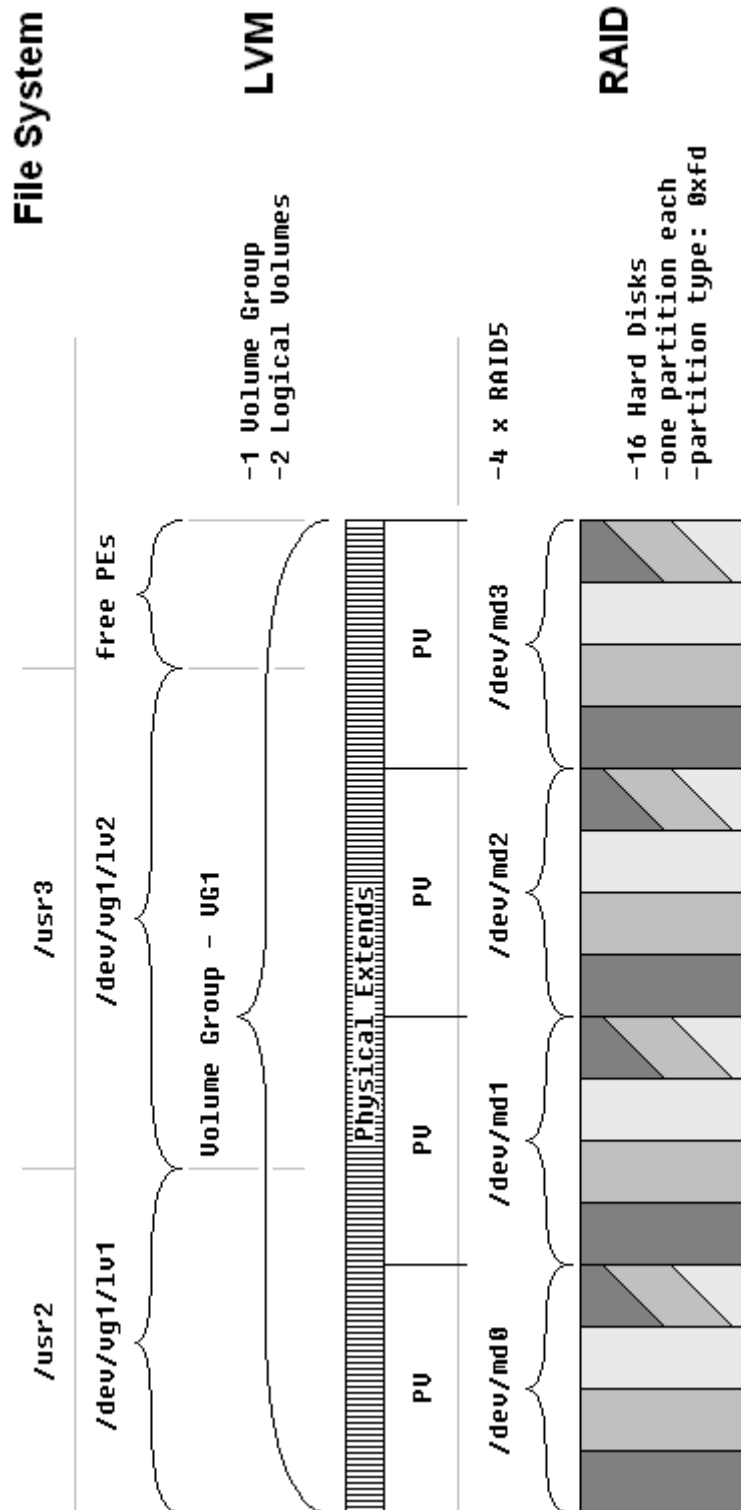


Figure 3: Sample setup - how RAID and LVM may work together