

# L V M

## A Logical Volume Manager for Linux

by

Heinz Mauelshagen

<linux-LVM@ez-darmstadt.telekom.de>





# Goals

- ◆ build a flexible I/O-system extension for Linux to gain a virtual view on mass storage (get rid of those physical disk limits)
- ◆ flexible scaling of mass storage capacities in wide ranges
- ◆ extension and reduction of storage without system interruption
- ◆ release it GNU licensed



## *Concept<sub>1</sub>*

- ◆ usage of physical disks, disk partitions, multiple or loop devices as PVs (Physical Volumes; basic allocation device)
- ◆ gather PVs in VGs (Volume Groups; virtual disks)
- ◆ allocation of VG storage to LVs (Logical Volumes; virtual partitions) in basic units called PEs (Physical Extents)



## *Concept<sub>2</sub>*

- ◆ map the address space of the LVs in units of LEs (Logical Extents; kind of block) to the PEs on the PVs
- ◆ extend and reduce VGs and LVs online
- ◆ use LVs like disks, disk partitions etc. for filesystems, databases etc.



## *Concept<sub>3</sub>*

- ◆ access VGs and LVs via device special files named `/dev/VolumeGroupName/*`
- ◆ store configuration information (VGDA or Volume Group Descriptor Area) on each PV of a VG and for performance reasons in a working copy on file in `/etc/lvmtab.d/`
- ◆ the VGDA holds all the attributes of the PV, VG, LVs and the allocation of the PEs



## *Concept<sub>4</sub>*

- ◆ map the LEs (Logical Extents) of LVs to the PEs on the PVs
- ◆ hold the attributes and the mapping tables in a LVM driver (loadable as a module too)
- ◆ add calls to the mapping function of the driver in  
“/usr/src/linux/drivers/block/l1\_rw\_blk.c”



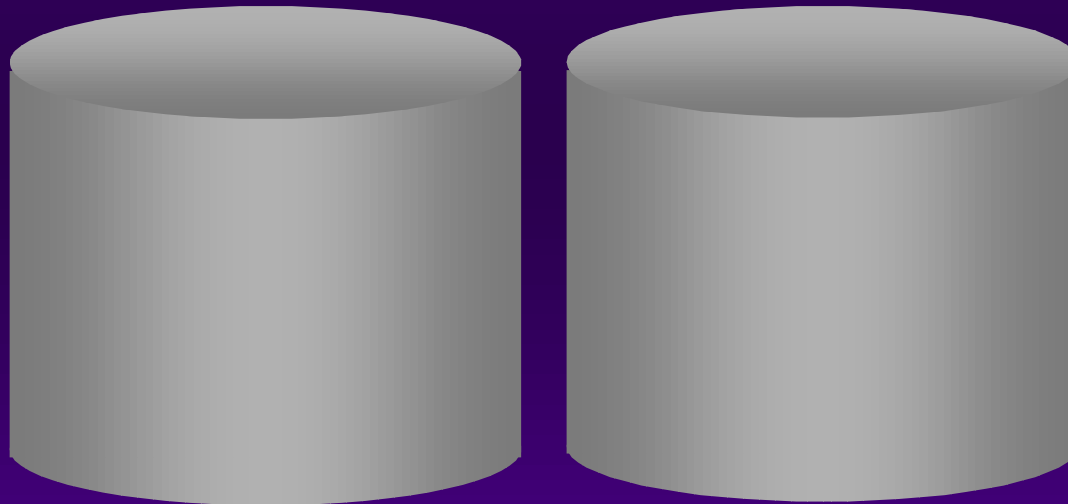
## *Concept<sub>5</sub>*

- ◆ build a command and a library layer to ease the integration of different or additional tools or graphical user interfaces
- ◆ export and import VGs to bring them to or get them from different computer systems
- ◆ support linear and striped (RAID0) LVs



# *Storage Architecture*

◆ 2 PVs

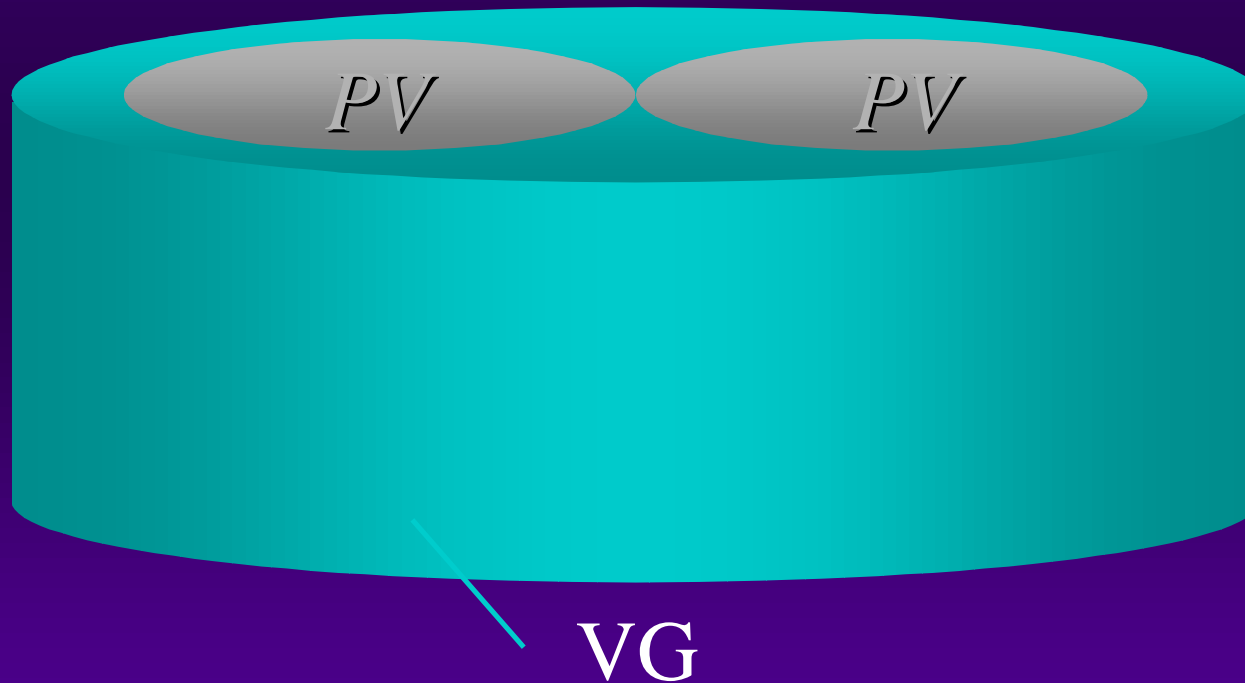


2 PVs



# *Storage Architecture*

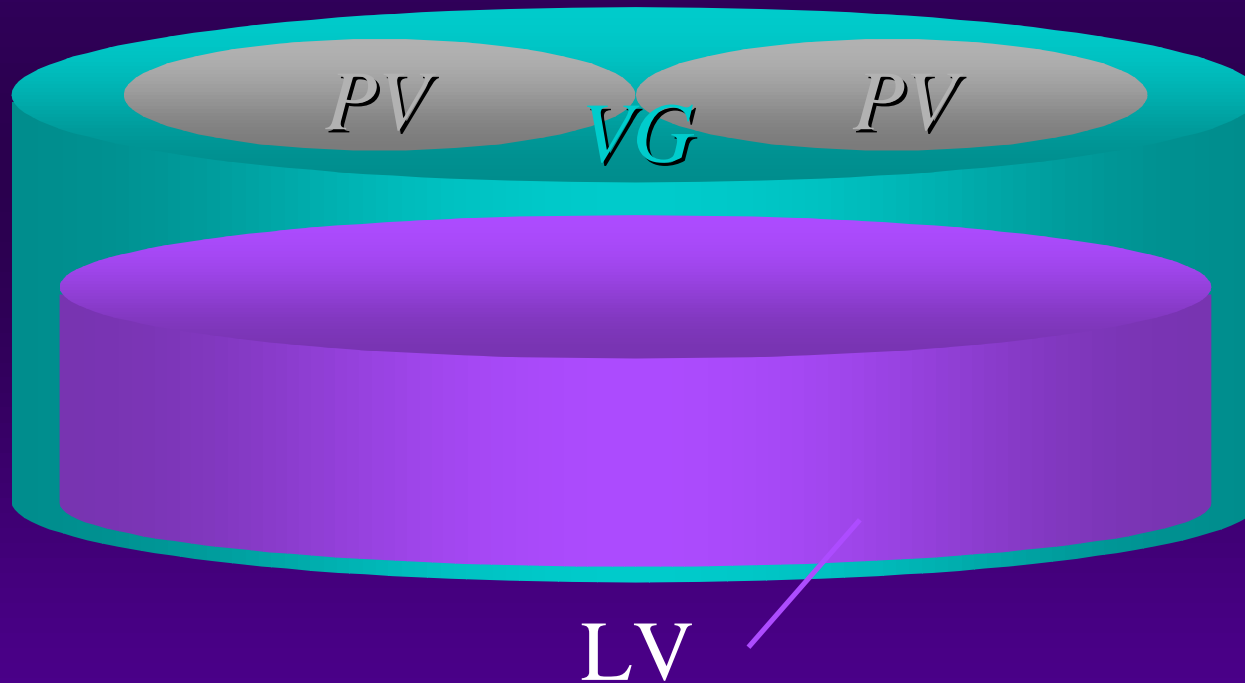
- ◆ VG with 2 PVs





# *Storage Architecture*

- ◆ VG with 2 PVs and 1 LV





# *PV Commands*

- ◆ pvchange - changes PV attributes
- ◆ pvcreate - initializes VGDA on a PV
- ◆ pvdata - displays VGDA (debugging)
- ◆ pvdisplay - shows PV attributes
- ◆ pvmove - moves PEs/LEs between PVs
- ◆ pvscan - searches for PVs



# *VG Commands<sub>1</sub>*

- ◆ `vgcfgbackup` - does a VGDA backup
- ◆ `vgcfgrestore` - restores VGDA to a PV
- ◆ `vgchange` - changes VG attributes
- ◆ `vgck` - check VG
- ◆ `vgcreate` - creates a new VG
- ◆ `vgdisplay` - shows VG attributes
- ◆ `vgexport` - make VG „unknown“



## *VG Commands<sub>2</sub>*

- ◆ `vgextend` - extend a VG online
- ◆ `vgimport` - make a VG „known“
- ◆ `vgmerge` - merge two VGs
- ◆ `vgmknodes` - create device specials
- ◆ `vgreduce` - reduce VG online
- ◆ `vgremove` - delete empty VG



## *VG Commands<sub>3</sub>*

- ◆ `vgrename` - rename an inactive VG
- ◆ `vgscan` - search for VG(s)
- ◆ `vgsplit` - split a VG into 2



# *LV Commands<sub>1</sub>*

- ◆ e2fsadm
  - extend or reduce LV and a contained ext2 Dateisystem (resize2fs needed)
- ◆ lvchange
  - change LV attributes
- ◆ *lvcreate*
  - *create a new LV*
- ◆ lvdisplay
  - show LV attributes
- ◆ lvextend
  - extend LV online



## *LV Commands<sub>2</sub>*

- ◆ `lvreduce` - reduce LV online
- ◆ `lvremove` - delete inactive LV
- ◆ `lvrename` - rename an inactive LV
- ◆ `lvscan` - search for LVs



# *LVM Commands*

- ◆ `lvmchange` - reset LVM in case of a test emergency
- ◆ `lvmdiskscan` - search for disks, disk partitions or multiple devices
- ◆ `lvmsadc` - collect LV read/write data
- ◆ `lvmsar` - report of collected LV statistic data



# *Software Metrics<sub>1</sub>*

- ◆ 99 VGs max
- ◆ 256 LVs total (8 bit minor #) in all VGs
- ◆ up to 65534 PEs total
- ◆ up to 65534 LEs per LV
- ◆ PE/LE sizes between several KB 8\*) and GB
- ◆ more than 1000 TB but limited by int variables in the Linux I/O subsystem today



## *Software Metrics<sub>2</sub>*

- ◆ about 800 hours work
- ◆ more than 34000 LOC (Lines Of Code) including all sources, comments, manual pages, scripts, makefiles, READMEs, ...
- ◆ about 29000 LOC sources
- ◆ modul/driver with 3200 LOC
- ◆ 192 library functions in 127 source modules
- ◆ 35 commands



## *Example*

Create a VG “test” with 2 PVs (/dev/sd[kl]1) and 1 LV “tlv” containing an EXT2FS:

```
# fdisk /dev/sdk          # set partition type identifier to 0xFE
# fdisk /dev/sdl          #
# pvcreate /dev/sd[kl]1
pvcreate -- physical volume "/dev/sdk1" successfully created
pvcreate -- physical volume "/dev/sdl1" successfully created
# vgcreate test /dev/sd[kl]1
vgcreate -- INFO: using default physical extent size of 4 MB
vgcreate -- INFO: maximum logical volume size is 256 Gigabyte
vgcreate -- doing automatic backup of "test"
vgcreate -- volume group "test" successfully created
#
```



## *Example Results*

- ◆ VGDA stored on `/dev/sd[kl]1`
- ◆ character special file `/dev/test/group`
- ◆ VGDA backup stored in `/etc/lvmconf/test.conf`
- ◆ VG name added to `/etc/lvmtab` and VGDA working copy stored in `/etc/lvmtab.d/test`
- ◆ VGDA is loaded into the driver so that VG “test” can be accessed



## *Example continued*

```
# lvcreate -L 300 -n tlv test # /dev/sdk1
lvcreate -- doing automatic backup of "test"
lvcreate -- logical volume "/dev/test/tlv" successfully created
```

```
# mke2fs /dev/test/tlv
mke2fs 1.10, 24-Apr-97 for EXT2 FS ....
```

```
<SNIP>
```

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

```
# mount /dev/test/tlv /usr1
```

```
. . . .
```



## *Example Results*

- ◆ block special file `/dev/test/tlv` mapping a size of 300 MB
- ◆ new working copy stored in `/etc/lvmtab.d/test`
- ◆ `/etc/lvmtab.d/test.conf` renamed to `/etc/lvmtab.d/test.conf.old`



## *Example Results*

- ◆ new backup of VGDA stored in `/etc/lvmtab.d/test.conf`
- ◆ changed VGDA loaded into the driver to be able to access `/dev/test/tlv`
- ◆ mounted EXT2 filesystem in `/dev/test/tlv`



## *The Future*

- + integrate driver into Linux 2.3 kernel
- + root-filesystem in a LV
- + associate UUIDs (Uniform Unique Identifiers) to VGs and PVs
- + extend VGDA by additional attributes like LV creation time...



## *Who and Where?*

- + **LVM software:**  
<http://linux.msede.com/lvm>
- + **Mail-list:**  
[linux-lvm@msede.com](mailto:linux-lvm@msede.com) (majordomo)
- + **Author:**  
[linux-lvm@ez-darmstadt.telekom.de](mailto:linux-lvm@ez-darmstadt.telekom.de)



*Thank you :-)*