



LVM2 – logical volume management

Milan Brož
mbroz@redhat.com

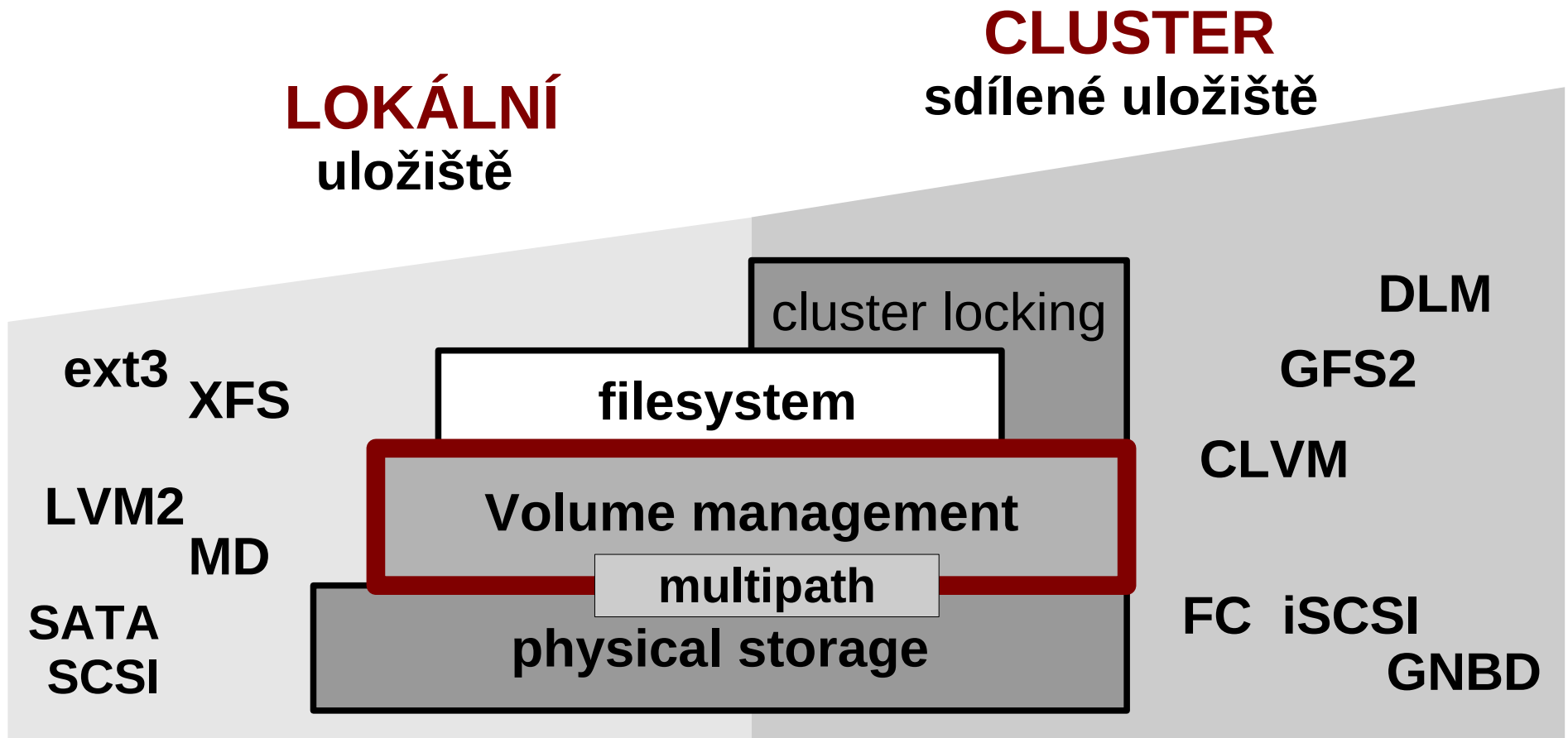
LinuxAlt 2007, 3.listopadu 2007, Brno

Proč virtualizace diskových oddílů ?

- **Abstrakce** nad omezení fyzických disků
- **Jednotné ovládání**
- **Vysoká dostupnost** (operace za běhu, HA)
- **Flexibilita** – změna parametrů, optimalizace, ...
- **Škálovatelnost**
- **Bezpečnost dat** - RAID, snímky obsahu
- **Celkové snížení nákladů na provoz**

Virtualizace diskového prostoru v Linuxu

Přehled technologií



MD versus LVM2

- **MD (multi-disk) je jaderný ovladač**
 - umožňuje vytvořit nové blokové zařízení nad existujícími disky
 - umožňuje použít RAID 0,1,4,5,6
 - stabilní, logika implementována v jádře, ovládání přes mdadm
- **DM (device mapper) je jaderný ovladač**
 - zcela obecný ovladač pro mapování blokových zařízení
 - umožňuje kombinovat všechny dostupné typy mapování a vytvořit z nich nové blokové zařízení
- **LVM2 je správce logických oddílů**
 - umožňuje kombinovat a spravovat fyzické disky, MD a DM zařízení a mapovat je do logických oddílů
 - vyžaduje funkční DM v jádře

LVM2 nad MD ?

▪ MD RAID ?

- MD poskytuje stabilní sw RAID 1,5,6
- LVM2 umí RAID 0,1, RAID 4,5 není zatím v jádře
 - vyžaduje userspace démona (dmeventd) pro rekonfigurace
 - zatím není podporováno v distribucích, potenciální problémy

▪ LVM2 nad MD RAID

- momentálně nejlepší kompromis
- výkonnostní problémy – odstranění je jedna z priorit vývoje LVM
- MD RAID nad LVM2 oddíly – možné, ale prakticky nemá smysl

▪ DMRAID

- sw RAID - využívá pouze device-mapper bez LVM vrstvy
- smysl má pouze při sdílení uložení s jiným OS

DMRAID

- **nástroj pro zpřístupnění diskových polí vytvořených pomocí sw RAID řadičů**

ATARAID – levné sw RAID řadiče (SiL, Highpoint, Promise, VIA, ...) často s vlastním BIOSem (s nástroji umožňujícími vytvářet raid pole a speciálními ovladači poskytujícími RAID funkce)

- **software RAID pole**, hw RAID nepotřebuje speciální ovladače a nepoužívá systémový procesor pro výpočet paritních bloků více na <http://linux-ata.org/faq-sata-raid.html>

- **speciální metadata** převádí na DM zařízení

- linear, JBOD (just bunch of disks)
- RAID 0,1
- RAID 4,5 (nemá zatím podporu ve stabilním jádře)

DMRAID – základní příkazy

dmraid -r [...] – vypíše všechna nalezená mapování

jméno disku, ovladač pro metadata, stav, ...

dmraid -a [y|n] [name] – aktivuje/deaktivuje raid

vytvoří nové dm blokové zařízení podle nalezených metadat na disku /dev/mapper/<drivername>_<devicename>

dmraid -s [...] – vypíše parametry raid pole

typ RAIDu, počet zařízení, velikost bloku, ...

dmraid -l [...] – vypíše podporované formáty metadat

*k aktivaci je třeba **dm raid jaderný ovladač** (nikoliv md/raid456)
dm raid5 není ve stabilním jádře dmraid nevytvěří mapování,
ačkoliv formátu metadat rozumí !*

kpartx

- zpřístupní jednotlivé diskové oddíly (podle tabulky oddílů) jako samostatná bloková zařízení

kpartx - příklad

kpartx -l device – seznam nalezených oddílů

kpartx -a device – vytvoří mapování oddílů

kpartx -d device – odstraní mapování oddílů

*zpřístupnění oddílů (particií) z obrazu disku v souboru
losetup /dev/loop0 file
kpartx -a /dev/loop0
a oddíly jsou přístupné pod /dev/mapper/loop0p1,p2,...*

LVM2 – správa logických oddílů

- **Abstraktní vrstva** nad fyzickými zařízeními
- Umožňuje vytvářet **logické oddíly (LV)**
nad kterými lze použít například souborový systém
- Logický oddíl je **blokové zařízení**
~ zhruba odpovídá fyzickému oddílu na disku

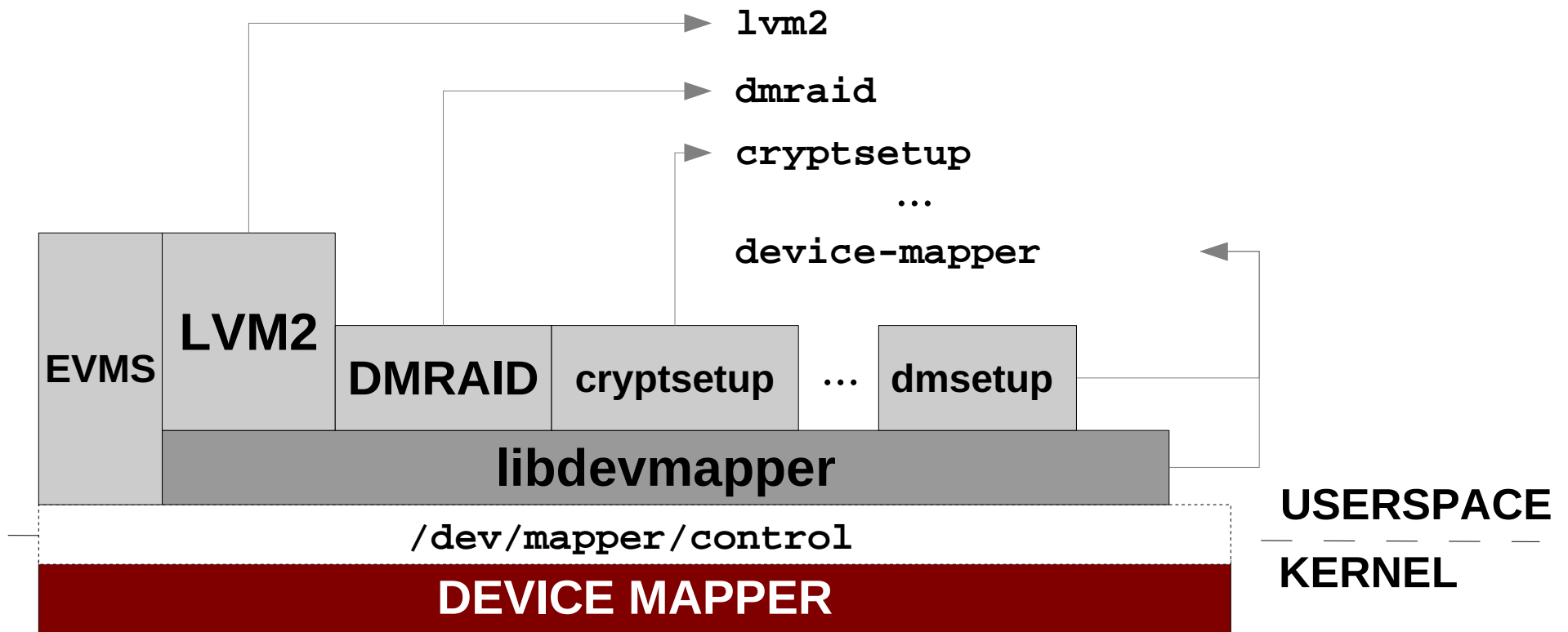
Linux LVM

- **LVM1 – pouze pro Linux jádro 2.4**
většina funkcí se provádí přímo v jádře

- **LVM2 – nový, obecný návrh**
 - zpětně podporuje LVM1 příkazy a formáty metadat
 - v jádře je jen nezbytná část kódu
 - logicky má tedy 2 části
 - **Modulární jaderný ovladač (DM)**
 - **Uživatelské nástroje (LVM2)**

- **EVMS**
 - Enterprise Volume Management System (IBM)
 - mimo jiné podporuje částečně i LVM metadata, používá DM

Aplikace nad jaderným DM



LVM2 – správa logických oddílů

(Logical Volume Manager – abstraktní vrstva nad DM)

3 VRSTVY

FYZICKÉ ODDÍLY (PHYSICAL VOLUMES) – PV

fyzická uložení ~ DISKY

jsou seskupeny

SKUPINY ODDÍLŮ (VOLUME GROUPS) – VG

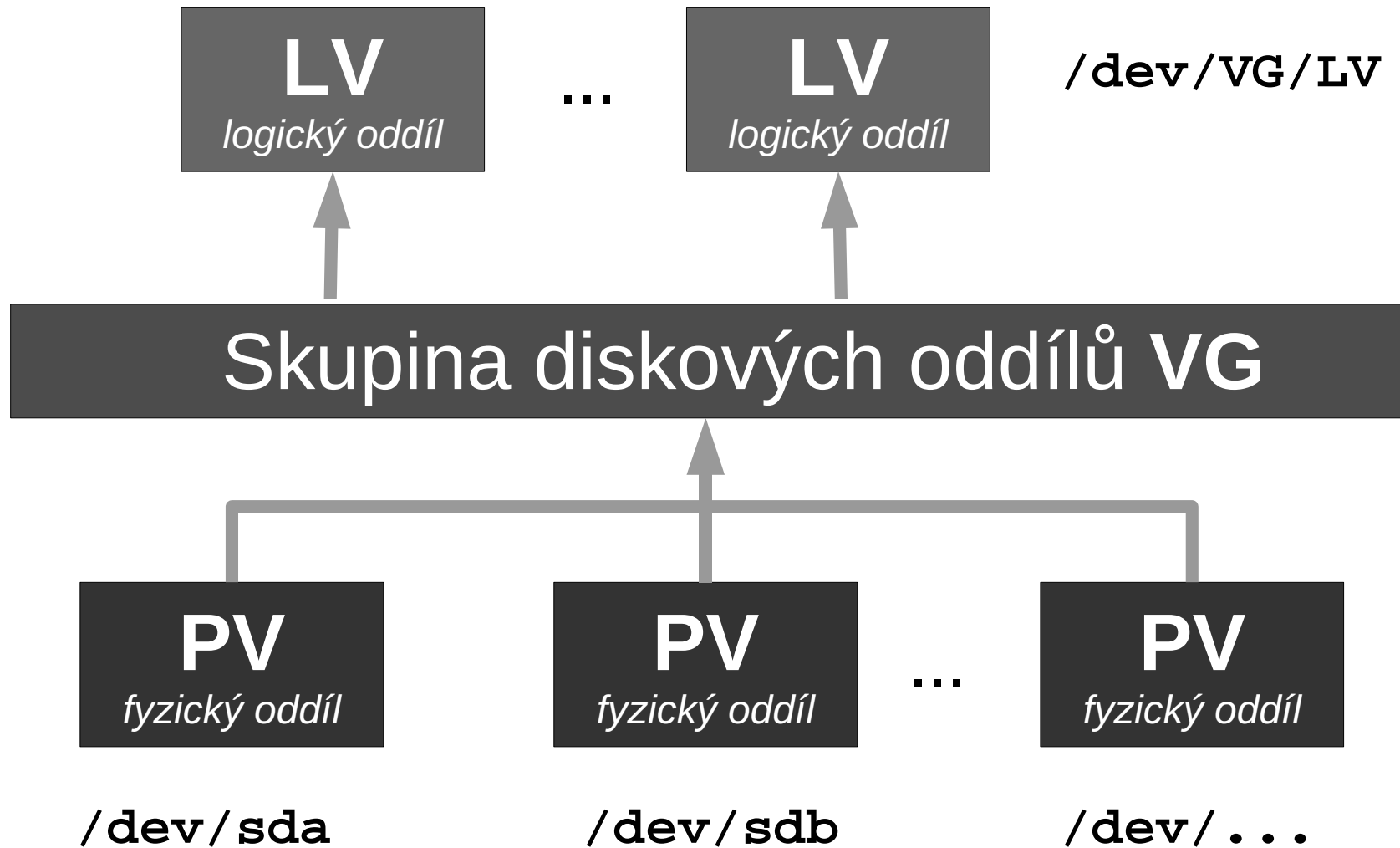
logické disky

úložná kapacita VG je přidělena pro

LOGICKÉ ODDÍLY (LOGICAL VOLUMES) – LV

logické diskové oddíly ~ PARTICIE

LVM2 – schéma



LVM2 – příkazy

- předpona: **pv***, **vg***, **lv***
- kompletní přehled příkazů: **lvm help**
- popis a přehled atributů: **pvs -o help**
- **man lvm**, **man <command>**

- **lvm** – jeden binární soubor (ostatní jsou symbolické linky), interaktivní mód
 - [pv,vg,lv] ***s[how] [*display]**, ***scan**, ***create**, ***remove**, ***change**
 - [vg,lv] ***reduce**, ***extend**, ***rename**
 - [pv,lv] ***resize**
 - **pvmove**
 - **vgmerge**, **vgsplit**
 - **lvconvert**
 - **lvmdiskscan**

- [vg] ***cfgbackup/restore**, ***import/export**, ***convert**, ***ck**, ***mknodes**

- Reportování problému - **lvmdump** (vytvoří archív se všemi potřebnými daty pro analýzu)

LVM2 – /etc/lvm, lvm.conf

- konfigurace je uložena v souboru **/etc/lvm/lvm.conf**
- **popis parametrů přímo v konfiguračním souboru**
příklady:
 - (devices) `filter = ["a/loop/", "r/*/"]`
nastavení zařízení, na kterých se budou hledat oddíly
 - `cache_dir = "/etc/lvm/cache"`
adresář, kde LVM udržuje cache
(lze smazat, jen pro urychlení některých operací)
 - (global) `locking_type = 1`
typ zamykání metadat – lokální, pro cluster (clvmd), ...
- `/etc/lvm/backup, archive` – historie změn metadat
(`vgcfgrestore, vgcfgbackup`)

LVM2 – METADATA obsahující popis logických oddílů

▪ Kde LVM ukládá metadata ?

- **PV** (blokové zařízení) **obsahuje hlavičku**
- metadata mají **textový formát** – plain ASCII
- **zálohy metadat** jsou v adresáři **/etc/lvm** (dle konfigurace)
- všechny **objekty mají jednoznačné UUID**
(které lze použít místo jména v případě duplicity)
- Metadata jsou **uložena na každém PV** (redundance)
(lze explicitně zakázat – v případě desítek PV jde o výrazné zrychlení práce při změně metadat)
- obnovení mapování může být při fyzickém selhání disku a ztrátě metadat poměrně komplikované
Je velmi vhodné metadata zálohovat !

LVM2 – ukázka formátu METADAT

```
creation_time = ...
...
vg_strip { id = "xxxxxxxx-xxxx-xxxx-xxxx-xxxx-xxxx-xxxxxxxx"
  ...
  physical_volumes {
    pv0 { id = "xxxxxxxx-xxxx-xxxx-xxxx-xxxx-xxxx-xxxxxxxx"
          device = "/dev/sdb1" # Hint only
          ...
          pe_start = 384
          pe_count = 50 # 200 Megabytes }
    pv1 { ... }
  }
  logical_volumes {
    lv_strip { id = "xxxxxxxx-xxxx-xxxx-xxxx-xxxx-xxxx-xxxxxxxx"
      ...
      segment1 {
        start_extent = 0
        extent_count = 100 # 400 Megabytes
        type = "striped"
        stripe_count = 2
        stripe_size = 128 # 64 Kilobytes
        stripes = [ "pv0", 0,
                   "pv1", 0 ]
      }
    }
  }
}
```

LVM2 – RAID0 (striped) zařízení

fdisk /dev/sd[bc] → oddíly sdb1,sdc1 typu 0x8e [LVM, není nutné]

pvcreate /dev/sdb1 /dev/sdc1 – vytvoření PV

vgcreate vg_strip /dev/sdb1 /dev/sdc1 – VG

pvs

vgs

lvcreate -L 400M -i 2 -n lv_strip vg_strip – LV

ls -l /dev/mapper

ls -l /dev/vg_strip/lv_strip

mke2fs /dev/vg_strip/lv_strip – formatování, připojení

mount /dev/vg_strip/lv_strip /mnt/tst

...

umount /mnt/tst

lvremove vg_strip lv_strip – odstranění LV, VG, PV

vgremove vg_strip

pvremove /dev/sdb1 /dev/sdc1

PVMOVE – přesun dat mezi PV

- **přesun libovolného rozsahu**
- **online operace** – využívá kcopyd (v jádře)
- operaci je možné znovu spustit (i po pádu systému)
metadata obsahují informace o průběhu
- vytváří dočasný RAID1 – **dm-mirror**
- pvmove spustí operaci, dále jen monitoruje
- přesun root – potenciální problémy

LVM2 – příklad PVMOVE

/dev/sdc1 → /dev/sdd1

pvcreate /dev/sdd1 – rozšíření VG o nový disk

lvs -o +devices

pvs

vgextend vg_strip /dev/sdd1

pvmove /dev/sdc1 /dev/sdd1 – vlastní přesun

pvs

vgreduce vg_strip /dev/sdc1 – odstranění disku

pvremove /dev/sdc1

Užitečné parametry pvmove

pvmove --abort – přeruší probíhající operaci

pvmove src:PE-PE dst:PE-PE – konkrétní rozsah PE

pvmove – (bez parametrů) – pokračuje v poslední operaci

LVM2 snímky (snapshots)

- **(zapisovatelné) snímky na úrovni blokového zařízení** (nezávislé na souborovém systému)
 - používá COW (**C**opy **O**n **W**rite) uložistě
 - COW je uloženo na disku (speciální LV)
 - prostor pro uložení změněných bloků (**menší než původní LV**)
 - Pozor: zaplněné snímky jsou automaticky zneplatněny !
- LVM2 zatím nepodporuje inkrementální (diferenciální) snímky
 - vícenásobné snímky stejného LV = obrovské zpomalení
- snímek zatím nelze vrátit na původní LV

SNAPSHOT I/O

read

- if block unchanged – read origin LV
- otherwise read COW space

write = overwrite old blocks

READ

WRITE

Select if block
original / changed

Logical Volume

Snap. COW

VOLUME I/O

read = read origin LV

write = backup old blocks to COW

1. copy old data
2. finish write

READ

WRITE

LVM2 – vytvoření snímku

```
vgextend vg_strip /dev/sdc1 – příprava místa VG (pro snímek)
```

Vytvoření snímku o velikosti <size> nad lv_strip LV

```
lvcreate -s -L<size> -n lv_snap vg_strip/lv_strip
```

```
mount /dev/vg_strip/lv_snap ...
```

lvs – výpis aktivních logických oddílů

/dev/vg_strip/lv_strip – původní oddíl

/dev/vg_strip/lv_snap – zapisovatelný snímek

Změna velikosti logického oddílu

- Je nutné dodržet pořadí operací !

- **Zvětšení logického oddílu**
 1. rozšíření skupiny oddílů o nové uložistě - disk)
 2. zvětšení logického oddílu
 3. zvětšení souborového systému na logickém oddílu (online)

- **Zmenšení logického oddílu**
 1. zmenšení souborového systému (pokud jde online, jinak offline)
 2. zmenšení logického oddílu
 3. odstranění prázdného uložistě ze skupiny oddílů)

LVM2 – změna velikosti oddílu ext3

zmenšení na <size>

```
pvs --units k
```

```
resize2fs -p /dev/vg_strip/lv_strip <size>K
```

```
lvresize -L <size>k /dev/vg_strip/lv_strip
```

použité jednotky se musí shodovat (velikost parametru k a K se zde opravdu liší !)

```
mount /dev/vg_strip/lv_strip ...
```

zvětšení na maximum

```
lvresize -l +100%FREE /dev/vg_strip/lv_strip
```

```
lvs
```

```
df -h
```

```
resize2fs -p /dev/vg_strip/lv_strip
```

```
df -h
```

Boot proces & LVM2

▪ GRUB

- částečná podpora LVM2 je až v GRUB2 zavaděči
- dnešní distribuce používají oddělený startovací (boot) oddíl a initramdisk

▪ initrd

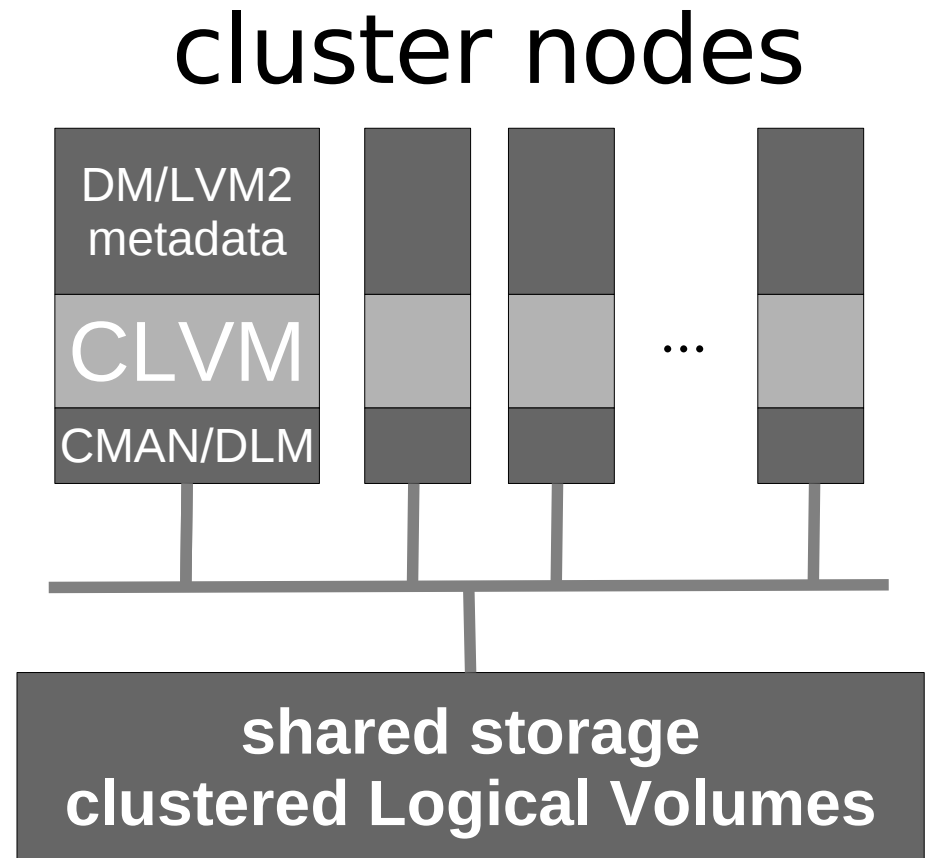
- obsahuje a zavede ovladače pro fyzické diskové řadiče
- vytvoří potřebné záznamy v /dev/ (device nodes)
- zavede ovladače pro device-mapper (a MD)
- spustí **vgchange -a y** čímž aktivuje logické oddíly
- nebo spustí **dmraid** a aktivuje ATARAID mapování
- připojí nový root (který teď již může být na LVM2)

▪ System boot (součást init skriptů)

- spustí **vgchange -a y** a aktivuje LVM monitoring démon **dmeventd** (Fedora, RHEL)

Cluster LVM (CLVM)

- Rozšiřuje funkce o podporu LVM2 v rámci clusteru.
- Poskytuje infrastrukturu pro manipulaci v rámci sdíleného uložště, **aktivaci logických oddílů a správu LVM metadat.**
- **clvmd démon**
 - vyžaduje DLM (distributed lock manager) a běžící cluster management sw (CMAN, ...)



Co je třeba pro LVM v clusteru ?

- **Sdílené uložště** (blokové zařízení)
 - **SAN** přímo připojený k uzlům (FC)
 - **iSCSI**
 - **GNBD** (Global Network Block Device)
- Volume management – **CLVM**
- clusterový souborový systém, např. **GFS**
- (existují řešení bez CLVM, kdy není třeba současné aktivace)
 - musí se vyhnout konfliktům při přístupu k LVM
 - např. striktní aktivací na jednom uzlu a neměnnými LVM metadaty

Odkazy

- Device mapper <http://sources.redhat.com/dm/>
- LVM2 <http://sources.redhat.com/lvm2/>
- LVM wiki <http://sources.redhat.com/lvm2/wiki>
- Cluster suite <http://sources.redhat.com/cluster/>
- dmraid <http://people.redhat.com/heinzm/sw/dmraid/>
- multipath <http://christophe.varoqui.free.fr/wiki/>
- dm-crypt <http://www.saout.de/misc/dm-crypt/>
- EVMS <http://evms.sourceforge.net/>