

# HardDisky

## Plotnové

- <http://blog.backblaze.com/2014/01/21/what-hard-drive-should-i-buy/>

## SSD

- ve SMARTu je potreba sledovat
  - [http://en.wikipedia.org/wiki/Wear\\_leveling](http://en.wikipedia.org/wiki/Wear_leveling)
  - Wear leveling count
    - pocet prepisu najstarsi bunky.
    - z toho se pocita normalizovana hodnota, ktora klesa od 100% u noveho disku k 0% pak uz je to otazka...
  - Media Wearout Indicator
    - Asi neco podobnyho, ale presne nevim... evidentne maj disky bud jedno nebo druhy
- Znacky
  - aktualne mam velmi dobre skusenosti s Samsungom, Samsung SSD 840 PRO Series
  - od znamych mam este info ze intely 500 a vyssia rada su gut tiez

## SW Aspekty

### Rozdeleni disku

- **MBR** (disky ≤2TB)
  - fdisk, sfdisk, cfdisk
  - kopie rozdeleni a → b: sfdisk -d /dev/sda | sfdisk /dev/sdb
- **GPT** (disky >2TB)
  - gdisk, sgdisk, cgdisk
  - kopie rozdeleni a → b: sgdisk -R=/dev/sdb /dev/sda; sgdisk -G /dev/sdb
  - jina varianta pokud predchozi nefunguje: a → b: sgdisk --backup=table /dev/sda; sgdisk --load-backup=table /dev/sdb

### RAID pomoci mdadm

- [https://raid.wiki.kernel.org/index.php/RAID\\_setup](https://raid.wiki.kernel.org/index.php/RAID_setup)
- Partition type: Linux RAID autodetect **0xFD** (0xFD00 u GPT)
- mdadm --create /dev/md0 --level=1 --bitmap=internal --raid-devices=2 /dev/sda1 missing
- mdadm --manage /dev/md0 --add /dev/sdb1
- mdadm --manage /dev/md0 --fail /dev/sdb1
- mdadm --manage /dev/md0 --remove /dev/sdb1
- mdadm /dev/md2 --fail /dev/sdc3 --remove /dev/sdc3
- mdadm --grow --bitmap=internal /dev/md0

- `mdadm --grow --bitmap=none /dev/md0`
- `mdadm --monitor --scan -l -t` (test sending of error e-mails)
- `echo "idle" > /sys/block/md0/md/sync_action` (defer active resync)
- `sysctl -w dev.raid.speed_limit_min=500000000; sysctl -w dev.raid.speed_limit_max=500000000` (unthrottle raid sync)

## Zvětšení RAID pole

Chceme zvětšit pole např. o velikosti 500G na nové disky o velikosti 1TB. Ideální je to dělat na degradovaném poli, protože tím získáme zároveň zálohu původního pole.

- Vyměníme jeden z disků za větší. Po startu systému bude pole degradované. Na novém disku vytvoříme nové partice typu Linux RAID autodetect **0xFD** (0xFD00 u GPT) po celé délce disku, případně jak potřebujeme, pokud raidů máme více. Bude samozřejmě větší než aktivní partice v raidu.
- Novou partici přidáme do raidu - `mdadm --manage /dev/md0 --add /dev/sdb1` a počkáme, až se pole syncne.
- Nezapomeneme zapsat grub na nový disk, abychom po dokončení operace nastartovali system - tj. `grub-install /dev/sdb!!!`

**následující operaci bude nutné provádět v jiném systému = např. nějaká live distribuce s podporou raidu, např. <https://partedmagic.com/downloads/>**

- Odebereme původní menší disk a přidáme druhý větší disk
- Nastartujeme live distribuci a složíme pole. Bude nyní degradované s novým diskem.
- Zadáme `mdadm --grow /dev/md0 --size=max` (zvětšíme pole na maximum velikosti partice)
- Kontrola filesystemu - `e2fsck -f /dev/md0`
- Zvětšíme filesystem - `resize2fs -p /dev/md0`
- Upravíme partice na druhém disku a přidáme do raid pole viz. výše.
- Nyní můžeme restartovat zpět do původního systému, synchronizace raidu bude potom pokračovat. V biosu bude nutné bootování přepnout na disk, který jsme měnili jako první. Po nastartování systému bude nutné také zapsat grub na druhý disk.

## Pokud system nespustí na degradovanem sw raidu

V Debianu Jessie se nám stalo, že po vyjmutí jednoho z disků začal debian startovat do ramdisku s chybou, že nemůže najít rootfs. Z nějakého důvodu nedošlo k automatickému startu degradovaného pole a proto bylo potřeba provést následující úpravu ramdisku:

- Přidat skript

[/etc/initramfs-tools/scripts/init-premount/mdadm-start](#)

```
#!/bin/sh
mdadm --run /dev/md*
exit 0
```

- `chmod 755 /etc/initramfs-tools/scripts/init-premount/mdadm-start`
- `update-initramfs -u`

Pripadnou chybu, ze /dev/md je adresarem muzete ignorovat.

## LVM

- Partition type: Linux LVM **0x8E**
- **PV** `pvcreate /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1`
  - overime pvs nebo `pvdisplay`
  - smazem `pvremove ...`
- **VG** `vgcreate fileserver /dev/sdb1 /dev/sdc1 /dev/sdd1 /dev/sde1`
  - overime vgs nebo `vgdisplay`
  - autodetekce `vgscan`
  - prejmenujem `vgrename fileserver data`
  - smazem `vgremove fileserver`
- **LV** `lvcreate -name backup -size 5G fileserver`
  - overime lvs nebo `lvdisplay`
  - autodetekce `lvscan`
  - prejmenujem `lvrename fileserver backup zalohy`
  - smazem `lvremove /dev/fileserver/backup`
  - zvetsime
    - `pvresize /dev/sdb1` roztahne PV pres celou zvetsenou partici
    - `lvextend -L5.5G /dev/fileserver/backup`
    - `lvextend -L+5.5G /dev/fileserver/backup` zvetsim o dalsich 5.5G
    - `e2fsck -f /dev/fileserver/backup`
    - `resize2fs /dev/fileserver/backup`
      - **POZOR: U ext4 provadej `resize2fs` online - nejprve mount napr do /mnt**
    - `xfs_growfs /mnt` zvetsi xfs pripojene do adresare /mnt
  - zmensime (**opatrne!**)
    - `e2fsck -f /dev/fileserver/backup`
    - `resize2fs /dev/fileserver/backup 10485760`
    - `lvreduce -L5G /dev/fileserver/backup`

## LVM Thin

- **LV** `lvcreate -L 100G -n data pve`
- **LV → thin-pool** `lvconvert -type thin-pool pve/data`
  - Thin-pool muzem nyní pridat do proxmoxu a ten si v nem bude delat thinlv pro virtualy a kontejnery
  - Nebo si v nem vytvorime vlastni thinlv
- **ThinLV** `lvcreate -n thin1 -V 1T -thinpool data pve`
- **Zobrazit lvm vcetne thin-pool** - `lvs -a`
- **Zvetseni thin-poolu o 256G** - `lvextend -L+256G /dev/vg/thinpool` - zmensovat nejde
- **Zvetseni metadat thin poolu** - `lvextend -poolmetadatasize +100M vg/thinpool`
- **Zvetseni vcetne metadat** - `lvresize -size +<size[\M,G,T]> -poolmetadatasize +<size[\M,G]> <VG>/<LVThin_pool>`
- Oprava (POZOR! OPATRNE!)

- Prohodit jiný LV k použití jako metadata thin poolu: `lvconvert --thinpool <VG>/<THIN_POOL_LV> --poolmetadata <NOVY_LV_METADATA>`
- **Oprava LV** `lvconvert --repair <VG>/<LVThin_pool>` ASI NENI UPLNE DOBRY NAPAD
- <https://github.com/jthornber/thin-provisioning-tools/>
- <https://www.unixrealm.com/?p=12000>

<https://www.redhat.com/archives/linux-lvm/2016-January/msg00010.html> So if you feel the time spend on thin\_checking doesn't pay-off - you can try to add option '-skip-mappings' (see lvm.conf field global/thin\_check\_options)

[/etc/lvm/lvm.conf](#)

```
...
thin_check_options = [ "--skip-mappings" ]
...
```

## Ext4

Zvetsit zurnal:

```
tune2fs -O ^has_journal /dev/hdXX
tune2fs -l /dev/hdXX
tune2fs -J size=128 /dev/hdXX
```

## ZFS

- Rozdelíme disky: typ partice BF(00) = Solaris Root (ale můžeme použít i přímo celé disky, v takovém případě se partice vytvoří samy)
- Vytvoríme hlavní systémový ZFS uložení s zrcadlem (něco jako VG u LVM)
  - `zpool create tank mirror /dev/sda /dev/sdb`
    - Někdy je dobré přidat parametr `-o ashift=12`, kde hodnota `ashift` = exponent 2 pro zarovnání na sektory.  $2^9=512\text{B}$  sektory,  $2^{12}=4096\text{B}$  sektory, ... Nejde to menit po naformátování, nicméně default by měl být víc než rozumný řešení, takže bych se spíš vyhnul rejpaní se v tomhle
    - Také se doporučuje přidávat disky podle ID, např.: `/dev/disk/by-id/wwn-NECONECO` kvůli spolehlivější detekci
  - `#zpool set listsnapshots=on tank` není potřeba, týká se jen výpisu "zfs list", jinak se snapshoty dají normálně i bez toho vypsat `ls -la /tank/.zfs/snapshot/`
  - `#zfs set recordsize=16k tank` (nastavit `recordsize` na stejnou velikost jako používá db. aplikuje se jen pro nové soubory: `default=128k`, `innodb-data=16k`, `innodb-log=128k`, `mysam=8k`, `postgresql=8k`, `sysbench=16k`)
  - `#zfs inherit recordsize tank` (nastavit `recordsize` na výchozí hodnotu)
  - `zfs set atime=off tank` (vypne `atime`)
  - `#zfs set logbias=throughput tank` (vypne dvojitý zápis dat přes SLOG, nejsem si jistý, že je to vždy dobře pro výkon, záleží na konfiguraci SLOGu)
  - `zfs set compression=on tank` (default kompresní algoritmus by měl být vcelku)

- rychlejš a napr. eliminuje dlouhý řetězce nul)
  - `#zfs set dedup=on tank` #deduplikace zere MOOC ramky = asi 8GB ram na 1TB dat! pomocí příkazu `zdb -S tank` lze zjistit jestli se deduplikace vyplatí (pokud to vypíše `dedup=2.00` nebo víc, tak se vyplatí. Pokud vynásobíme total allocated blocks číslem 320, vyjde nám potřebná ramka)
  - `zpool list` a `zpool status`
  - `zfs get all tank`
- V tomto uložisti vytvoříme ZFS pro LXC a připojíme
  - `zfs create tank/vps`
  - **Pokud používáme proxmox, tak tady zkončíme a zbytek naklikáme v proxmoxu!**
  - `zfs set mountpoint=/var/lib/lxc tank/vps`
  - mountpoint nesmí existovat, ZFS si automaticky adresář vytvoří při připojení a smaže při odpojení.
- Vytvoříme nový virtual s kvotou (tohle už dělá LXC automaticky)
  - `zfs create tank/vps/test`
  - `zfs set quota=10G tank/vps/test`
- `fsck: zpool scrub tank` (průběh sledujem v `zpool status`)
- Všechno smažem (není dobrý nápad): `#zpool destroy tank`
- `zpool status` Vypíše všechny disky v poolu, jestli nedochází k datovým chybám, jestli probíhá scrub, resilver, atd...
- Log a cache na SSD
  - Typy pomocných zařízení
    - "log", také nazývají "zil" nebo "slog" je záznam synchronních zápisů
    - "cache" neboli "l2arc" ("level 2 arc") je cache pro čtení
    - ani u jednoho nevadí výpadek, v cache jsou jen data přečtené z rotačního disku a všechno co je v logu je i v RAMce, log by se použil jen při výpadku napájení, kdy bychom o RAMku přišli
  - Dimenzování
    - velikost logu stačí cca 0.6 GB na každý gigabit síťovky co máme na systému. 16GB by tedy mělo s obří rezervou stačit i pro případy že budu mít 2x10G síťovku. Flushuje se jednou za 5 sekund, takže stačí když se do něj vejdou všechny synchronní zápisy co probíhají za 5 sekund. (= ano, víc než 16GB těžko budu v současnosti potřebovat, i to je fakt hodně, prakticky stačí asi 1GB nebo méně)
    - na cache je potom dobře využít všechno místo co nám na SSD zbyde po logu (cím víc, tím líp. dává smysl, aby byla cache radově větší než log i L1ARC, ale jejich velikosti spolu přímo nijak nesouvisí)
  - log a cache vždy přidáváme jako `/dev/disk/by-id/` nemají metadata!!!
  - `zpool add tank log /dev/loop0`
  - `zpool add tank cache /dev/loop1`
  - `zpool iostat -v` vypíše obsazení logu a cache (a všech jednotek v poolu)
  - `zpool remove tank /dev/loop0 /dev/loop1`
- Výměna disku v RAIDu (konzultovat s harviem!!!)
  - `zpool detach tank sdb2`
  - `zpool attach tank sda2 sdb2`
- Aktivace automatického zvětšování zrcadla (asi dobrý udělat už před výměnou disku za větší)
  - `zpool set autoexpand=on tank`
- ZVOL (= něco jako LVM uvnitř ZFS poolu, doporučeno na swapy a image virtualu!)
  - `zfs create -V 5gb tank/vol` (vytvoří jednotku `/dev/zvol/tank/vol`, taky známý jako `/dev/zd0`, parametr `-s` udělá zvol bez rezervace diskového prostoru v poolu = thin-provisioning)
  - `zfs list -t volume` vypíšeme si zvoly (bez `-t volume` to vypíše vše v poolu)

- zfs destroy tank/vol
- Autodetekce existujícího ZFS
  - zpool import nebo zpool import -a pro exportnutý
  - zfs mount -a
- Testování ZFS
  - Ztest NESLOUŽÍ k testování zfs modulu v jádře!!! Pro otestování systému je naprosto nevhodný.
    - ~~ztest -f /tmp -VV~~ vytvoří v /tmp blockfile s testovacím ZFS a spustí na něm unit testy (musí tam být dost místa).
    - ~~Jde prodloužit čas v sekundách, defaultně -T 300~~
- Další zdroje
  - ZoL manual <https://pthree.org/2012/04/17/install-zfs-on-debian-gnulinux/>
  - Arch Linux ZFS <https://wiki.archlinux.org/index.php/ZFS>
  - Things Nobody Told You About ZFS <http://nex7.blogspot.cz/2013/03/readme1st.html>

Je dobré omezit kolik RAM může sežrat ARC (defaultně si bere 1/2 veškeré paměti na Linuxu a 3/4 na Illumosu):

[/etc/modprobe.d/zfs.conf](#)

```
#echo '(1024^3)*10' | bc
#update-initramfs -u -k all
options zfs zfs_arc_max=10737418240
```

## ZFS replikace

- zdroj# zfs snapshot tank/vps/subvol-300-disk-1@mujsnapshot
- zdroj\$ su -c "zfs send tank/vps/subvol-300-disk-1@mujsnapshot" | pv | ssh strojcilovy sudo zfs recv -F tank/vps/subvol-300-disk-1
- cil#

## NILFS2

- mkfs.nilfs2 -L LABEL /dev/sdx1
- lscp /dev/sdx1 Vypíšeme checkpointy (historii)
- chcp ss /dev/sdx1 94 Nastavíme checkpoint #94 jako snapshot (= nebude smazán cleanerem a půjde připojit)
- mount -o ro,cp=94 /dev/sdx1 /mnt/ Připojíme readonly snapshot #94
- chcp cp /dev/sdx1 94 Po odpojení zrušíme snapshot #94 (uděláme z něj zpatky checkpoint)

## SATA HotSwap

- readlink /sys/block/**sda** (zjistí na jaké sběrnici je disk **sda**)
- echo 1 > /sys/block/**sda**/device/delete (odpojí disk **sda**)
- echo "- - -" > /sys/class/scsi\_host/**host0**/scan (najde disky na sběrnici **host0**)

## Fyzická identifikace disku

### Pomocí LED

Pokud má stroj modré identifikační led diody u jednotlivých šuplíčků disku, je možné je rozsvítit nebo zhasnout následujícím příkazem z balíčku `ledmon`:

- `ledctl locate=/dev/sda`
- `ledctl locate_off=/dev/sda`

Je možné použít i hierarchické označení disku (např. `/dev/disk/by-id/[drive-id]`, atd...)

### Host Protected Area (HPA)

Precti si o tom neco nez to zacnes pouzivat!

- Zakázat v GRUBu: `libata.ignore_hpa=1`
- Overit: `cat /sys/module/libata/parameters/ignore_hpa`
- Zjistit stav HPA u disku: `hdparm -N /dev/sd?`
- Nastavit HPA u disku na hodnotu XXX: `hdparm -N XXX /dev/sdx` (HPA se zrusi nastavenim hodnoty XXX na max co disk umi)
  - Pokud se ma hodnota XXX zachovat permanentne i po rebootu, tak se musi uvadet s "p" jako pXXX, jde to udelat jen jednou za power-cycle.

### Migrace ext3 na ext4

Minimální požadavky: **kernel 2.6.30; grub 1.96+20090808; e2fsprogs 1.41.6; mount 2.16**

- `umount /dev/md5`
- `fsck.ext3 -ftv /dev/md5`
- `tune2fs -0 extents,uninit_bg,dir_index /dev/md5`
- `fsck.ext4 -yDtv /dev/md5`

Podrobnosti na

[http://www.debian-administration.org/article/643/Migrating\\_a\\_live\\_system\\_from\\_ext3\\_to\\_ext4\\_filesystem](http://www.debian-administration.org/article/643/Migrating_a_live_system_from_ext3_to_ext4_filesystem)

### Migrace Windows 10 na mensi SSD

Migroval jsem uspesne 1TB HDD na 960GB SSD. Delat to z windows je nemozne (nezkousel jsem placene windows nastroje tretich stran, ale vsechny freeware nastroje selhaly, primo ve windows neni nic co by to zvladlo). Na Linuxu to jde, ale je to trochu makačka. Muj postup byl nasledujici:

- Na windows 10 v nabídce start najít CMD a "spustit jako správce"
- Na windows pustit `chkdsk /f`, nabídne to naplanovani opravy FS pri dalsim rebootu, rebootnout a zcekovat. Linux tohle neumi.
- Kdyz je disk opravenej, muze clovek nabootovat live linux, pouzil jsem ArchBang. Ma vse

potrebne, bezi z RAM a vejde se na flashku.

- Pomoci `cfdisk` si otevru stary a novy disk. Vsechny oddily vytvorim stejne na GPT(!), ale ten nejvetsi zmensim aby se to tam vse veslo. Velikosti zadavam v sektorech, stejne jako na puvodnim disku. Pouziva se k tomu suffix "S", napr.: 123456789S. Bohuzel to neumim naskriptovat, takze jsem to delal docela rucne:
  - Nastavim stejny typ u vsech partic, zapisu a zavru `cfdisk`.
  - Pomoci `sgdisk` nastavim GUID disku (-U) i jednotlivych partici (-u) a jejich atributy (-A) tak, aby byly stejne jako na puvodnim disku. Jinak to nebude bootovat!!! Je to mravenci prace, dela se to po jednom, detaily v `man sgdisk`. Jake GUID ma puvodni disk jsem vycital z `cfdisku`.
- Podivam se jak velka je cilova partice se systemem a reziznu zdrojovy system aby se tam s rezervou vesel. Napr.: `ntfsresize -size 850G /dev/sda2`. Musi tam byt volne misto aby to slo. Toto selze, pokud jsi na zacatku neudelal `chkdsk /f`
- Po reziznuti znovu rebootnu do windows a znovu udelam `chkdsk /f` vctne rebootu jako predtim, aby se disk opravil. pak se vratim do linuxu.
- Pomoci `dd` prekopiruju obsah jednotlivych partic krome te nejvetsi systemove.
- Nejvetsi systemovou prekopiruju pomoci `ntfscclone`, napr.: `ntfscclone --overwrite /dev/cil /dev/zdroj`. Funguje to jako `dd`, ale nekopiruje to bloky nealokovane fs, takze na poloprazdnem fs to usetri pulku casu. Necham bezet pres noc.
- `sync`
- Znovu rebootnu do windows a udelam `chkdsk /f+reboot`, vratim se do linuxu
- Protoze jsem si nechal par GB rezervu a chci ji ziskat zpatky, tak na cilovem systemu pustim `ntfsresize /dev/sdb2` na systemovy oddil, abych filesystem roztahl opet pres celou novou partici
- Znovu rebootnu do windows a udelam `chkdsk /f+reboot`, windows by mely v poradku bootovat
- Protoze oba disky maji stejnou GUID, neni mozne je oba mit ve windows pripojene najednou. Puvodni disk necham 1-3 mesice, nez se zahori SSD a pak na nem prepisu GPT s novym oddilem, ten naformatuju a mam externi disk na zalohy, nebo cokoliv jineho.

## Poznámky

- **Test rychlosti**
  - `hdparm --direct -t /dev/sd?`
- **Číslo ATA portu v dmesgu - převod na device**
  - `ata=4; ls -l /sys/block/sd* | grep $(grep $ata /sys/class/scsi_host/host*/unique_id | awk -F'/' '{print $5}')`
- **Souhrn co zapisuje na disk**
  - `iotop -aoP`
- **Kdo pouziva mountpoint**
  - `fuser -mv /mnt/point`
- **Záchrana dat**
  - <http://www.forensicswiki.org/wiki/Ddrescue>

From:

<https://wiki.spoje.net/> - **SPOJE.NET**

Permanent link:

<https://wiki.spoje.net/doku.php/howto/hw/disky?rev=1599402203>

Last update: **2020/09/06 16:23**

