

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
 - Wear leveling count
 - pocet prepisu nejstarsi bunky.
 - z toho se pocita normalizovana hodnota, ktera klesa od 100% u noveho disku k 0% pak uz je to otazka...
 - Media Wearout Indicator
 - Asi noco podobnyho, ale presne nevim... evidentne maj diskы bud jedno nebo druhы
- Znacky
 - aktualne mam velmi dobre skusenosti s Samsungom, Samsung SSD 840 PRO Series
 - od znamych mam este info ze intely 500 a vysia rada su gut tiez

SW Aspeky

Rozdeleni disku

- **MBR** (disky ≤ 2TB)
 - fdisk, sfdisk, cfdisk
 - kopie rozdeleni a → b: `sfdisk -d /dev/sda | sfdisk /dev/sdb`
 - nutno jeste zmenit id disku: `sfdisk --disk-id /dev/sdb 0x141eee23` (vymyslim si nebo pouziju generator)
- **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 pomocí mdadm

- 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 -1 -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)

Přidání dalšího disku do raidu s UEFI boot

- mdamd --grow --raid-devices=2 --force /dev/md127 (-force je potreba pokud byl puvodne raid1 zalozen jen s jedním diskem)
- mdadm --manage /dev/md127 --add /dev/sdb3
- umount /boot/efi - je potreba odpojit puvodni EFI partiton, ktera je pravdepodobne na sda2
- mkfs.vfat /dev/sdb2 - musime prirpavit EFI partici na novem disku
- mount /dev/sdb2 /boot/efi nyni prijime novou EFI partici do puvodniho umistení.
- grub-install /dev/sdb
- standardne nepotrebujeme mit pripojenou /boot/efi v FSTAB, ale je potreba myslet na to, ze pri upgradu kernelu musime OBE partice postupne pripojit a zapsat na ne aktualni verzi grubu !!
- pro /boot/efi pridat do fstab options nofail > PARTUUID="eeefac33-598c-1540-acfa-e401d9d44d15" /boot/efi vfat defaults,nofail 0 1
- overit, jestli v /boot/efi na sdb2 je nějaký obsah (musí tam být adresář EFI) pokud tam není, překopírujte z disku sda2 !
- nakonec pro jistotu zavolat update-grub

Pri tomto postupu dokazeme nabootovat do zalozniho kopie systemu v pripade potizi s primarnim diskem.

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 samořejmě větší než aktivní partici 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 mensi disk a pridame 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 particie)

- 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 nestartuje 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`

Připadnou 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
 - autodetecte vgscan
 - prejmenujem vgrename fileserver data
 - smazem vgremove fileserver
- **LV** `lvcreate -name backup -size 5G fileserver`
 - overime lvs nebo lvdisplay
 - autodetecte 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 nyni pridat do proxmoxu a ten si v nem bude delat thinly pro virtualy a kontejnery
 - Nebo si v nem vytvorime vlastni thinly
- **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 -zmenovat 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 jiny LV k pouziti 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

- Rozdelime disky: typ partice BF(00) = Solaris Root (ale muzem pouzit i primo cely disky, v takovym pripade se partice vytvorí samy)
- Vytvorime hlavni systemovy ZFS uloziste s mirrorom (neco jako VG u LVM)
 - zpool create tank mirror /dev/sda /dev/sdb
 - Nekdy je dobrý pridat parametr -o ashift=12, kde hodnota ashift = exponent 2 pro zarovnani na sektory. $2^9=512B$ sektory, $2^{12}=4096B$ sektory,... Nejde to menit po naformatovani, nicmene default by mel byt vic nez rozumny reseni, takze bych se spis vyhnul rejpani 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 neni potreba, tyka se jen vypisu "zfs list", jinak se snapshoty daj normalne i bez toho vypsat ls -la /tank/.zfs/snapshot/
 - #zfs set recordsize=16k tank (nastavit recordsize na stejnou velikost jako pouziva db. aplikuje se jen pro nove soubory: default=128k, innodb-data=16k, innodb-log=128k, myisam=8k, postgresql=8k, sysbench=16k)
 - #zfs inherit recordsize tank (nastavit recordsize na vychozi hodnotu)
 - zfs set atime=off tank (vypne atime)
 - #zfs set logbias=throughput tank (vypne dvoji zapis dat pres SLOG, nejsem si jisty, ze je to vzdy dobre pro výkon, zalezi na konfiguraci SLOGu)
 - zfs set compression=on tank (default kompresni algoritmus by mel byt vcelku rychlej a napr. eliminuje dlouhy retezce nul)
 - #zfs set dedup=on tank #deduplikace zere MOOC ramky = asi 8GB ram na 1TB dat! pomocí prikazu zdb -S tank lze zjistit jestli se deduplikace vyplati (pokud to vypise dedup=2.00 nebo vic, tak se vyplati. Pokud vynasobime total allocated blocks cislem 320, vyjde nam potrebna ramka)
 - zpool list a zpool status
 - zfs get all tank
- V tomto ulozisti vytvorime ZFS pro LXC a pripojime
 - zfs create tank/vps
 - **Pokud pouzivame proxmox, tak tady zkonzime a zbytek naklikame v proxmoxu!**
 - zfs set mountpoint=/var/lib/lxc tank/vps
 - mountpoint nesmi existovat, ZFS si automaticky adresar vytvorí pri pripojeni a smaze pri odpojeni.
- Vytvorime novej virtual s quotou (tohle uz dela LXC automaticky)
 - zfs create tank/vps/test
 - zfs set quota=10G tank/vps/test
- Fsck: zpool scrub tank (prubeh sledujem v zpool status)
- Vsechno smazem (neni dobrý napad): #zpool destroy tank
- zpool status Vypise vsechny disky v poolu, jestli nedochazi k datovym chybám, jestli probíha scrub, resilver, atd...
- Log a cache na SSD
 - Typy pomocnejch devicu
 - "log", take nazvyvaný "zil" nebo "slog" je zurnal synchronních zapisu
 - "cache" neboli "l2arc" ("level 2 arc") je cache pro čtení
 - ani u jednoho nevadí výpadek, v cache jsou jen data přečteny z rotacního disku a všechno co je v logu je i v RAMce, log by se použil jen při výpadku napájení, když bysme o RAMku přišli
 - Dimenzování

- velikost logu staci cca 0.6 GB na kazdy gigabit sitovky co mame na systemu. 16GB by tedy melo s obri rezervou stacit i pro pripady ze budem mit 2x10G sitovku.
Flushuje se jednou za 5 sekund, takze staci kdyz se do nej vejdou vsechny synchronni zapisy co probihaj za 5 sekund. (= ano, vic nez 16GB tezko budem v soucasnosti potrebovat, i to je fakt hodne, prakticky staci asi 1GB nebo min)
- na cache je potom dobre vyuzit vsechno misto co nam na SSD zbyde po logu (cim vic, tim lip. dava smysl, aby byla cache radove vetsi nez log i L1ARC, ale jejich velikosti spolu primo nijak nesouvisi)
 - log a cache vzdy pridavame jako /dev/disk/by-id/ nemaji metadata!!!
 - zpool add tank log /dev/loop0
 - zpool add tank cache /dev/loop1
 - zpool iostat -v vypise obsazeni logu a cache (a vsech jednotek v poolu)
 - zpool remove tank /dev/loop0 /dev/loop1
- Vymena disku v RAIDu (konzultovat s harviem!!!)
 - zpool detach tank sdb2
 - zpool attach tank sda2 sdb2 (asi dava spis smysl attachnout celej praznej disk, oddily si vytvori ZFS samo)
 - zpool replace -f tank <old_drive> <new_drive> (varianta predchoziho v jednom kroku, pouziva se pro vymenu vadneho disku ve stejnem slotu.)
- Aktivace automatickyho zvetsovani zrcadla (asi dobrý udelat uz pred vymenou disku za vetsi)
 - zpool set autoexpand=on tank
- ZVOL (= noco jako LVM uvnitr ZFS poolu, doporucony na swapy a image virtualu!) - **Pro ZVOL je povinný parameter -V, pokud neuvedeme, založí se klasicky zfs filesystem !!**
 - zfs create -V 5gb tank/vol (vytvoří jednotku /dev/zvol/tank/vol, taky znamy jako /dev/zd0, parametr -s udela zvol bez rezervace diskovyho prostoru v poolu = thin-provisioning)
 - zfs list -t volume vypiseme si zvoly (bez -t volume to vypise vše v poolu)
 - zfs destroy tank/vol
- ZVOL můžeme následně také naformátovat na libovolný FS - např. mkfs.xfs -b size=4096 -m reflink=1,crc=1 /dev/zvol/tank/vol
- Autodetecte existujiciho ZFS
 - zpool import nebo zpool import -a pro exportnutý
 - zfs mount -a
- SWAP na ZFS
 - Swapovani do souboru na ZFS není podporovano, ale snad se da pouzit ZVOL jako swap
 - Nicmene jsou tam problemy (ZFS nekdy potrebuje pri zapisu do zvolu samo swapovat, coz je dost nemily predpoklad pro zapis do swapu = zpusobuje zamrznutí)
 - nekdo tvrdi, ze pomuze ZVOLu nastavit logbias=throughput a compress=zle, nicmene jsem netestoval
 - urcite to chce mit nastaveny hung_task a reboot pri panicu
- Vylepseni pro backup destinace
 - zfs set readonly=yes tank/backup - zabranime systemu menit zalohy (recv stale funguje)
 - zfs set volmode=none tank/backup - schovame ZVOLy pred systemem (aby nam je nedetekoval treba mdraid nebo lvm)
 - Aby nemusela protistrana mit pristup na root, je mozne povolit userovi prava na konkretni dataset pomocí zfs allow
 - <https://github.com/oetiker/znapzend#running-by-an-unprivileged-user>
 - Sending end: destroy,hold,mount,send,snapshot,userprop
 - Receiving end: create,destroy,mount,receive,userprop
 - Eg.: zfs allow -u backup-pve1

- `create,destroy,mount,receive,userprop tank/backup/pve1`
- Verify: `zfs allow tank/backup/pve1`

• Testovani 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 dobrý omezit kolik RAM může sezrat ARC (defaultně si bere 1/2 všech pamětí 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 strojciловъ sudo zfs recv -F tank/vps/subvol-300-disk-1`
- `cil#`

NILFS2

- `mkfs.nilfs2 -L LABEL /dev/sdx1`
- `lscp /dev/sdx1` Vypíše 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 zrusíme snapshot #94 (uděláme z nej zpátky 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)

Preci si o tom neco nez to zacnes pouzivat!

- Zakazat 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 -O extents,uninit_bg,dir_index /dev/md5
- fsck.ext4 -yfDtv /dev/md5

Podrobnosti na

http://www.debian-administration.org/article/643/Migrating_a_live_system_from_ext3_to_ext4_filesystems

Migrace Windows 10 na mensi SSD

Migoval jsem uspesne 1TB HDD na 960GB SSD. Delat to z windows je nemozne (nezkousel jsem placene windows nastroje tretich stran, ale vsechny freeware nastroje i programy dodavane k SSD 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 nabidce start najit CMD a "spustit jako spravce"
- Na windows pustit chkdsk /f, nabidne 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 predinstalovane vse potrebne, bezi z RAM a vejde se na flashku.
- Pomoci cfdisku 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, detaile v man sgdisk. Jake GUID ma puvodni disk jsem vycital z cfdisku.
- Podivam se jak velka je cilova particie se systemem a resiznu 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 resiznuti znova rebootnu do windows a znova udelam chkdsk /f vcetne 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 ntfsclone, napr.: ntfsclone --overwrite /dev/c1l /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 stejna 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=1754408209>

Last update: **2025/08/05 17:36**