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 neco podobnyho, ale presne nevim... evidentne maj disky bud jedno nebo druhy
- Znacky
 - $\circ\,$ aktualne mam velmi dobre skusenosti s Samsungom, Samsung SSD 840 PRO Series
 - $\circ\,$ od znamych mam este info ze intely 500 a vyssia rada su gut tiez

SW Aspekty

Rozdeleni disku

- **MBR** (disky ⇐2TB)
 - fdisk, sfdisk, cfdisk
 - \circ kopie rozdeleni a → b: sfdisk -d /dev/sda | sfdisk /dev/sdb
- GPT (disky >2TB)
 - gdisk, sgdisk, cgdisk
 - \circ 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
- 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 jednim 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 pripojime novou EFI partici do puvodniho umisteni.
- 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-acfae401d9d44d15" /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 vetší než aktivní partice v raidu.
- Novou partici přídáme 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 partice)
- Kontrola filesystemu e2fsck f /dev/md0
- Zvětšíme filesystém 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 prvni. 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

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
 - $\circ\,$ 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
 - \circ overime lvs nebo lvdisplay
 - autodetekce lvscan
 - \circ prejmenujem lvrename fileserver backup zalohy
 - o smazem lvremove /dev/fileserver/backup
 - \circ zvetsime
 - pvresize /dev/sdb1 roztahne PV pres celou zvetsenou partici
 - Ivextend -L5.5G /dev/fileserver/backup
 - Ivextend -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
 - Ivreduce -L5G /dev/fileserver/backup

LVM Thin

- LV lvcreate -L 100G -n data pve
- LV → thin-pool lvconvert -type thin-pool pve/data
 - $\circ~$ Thin-pool muzem nyni 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 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/
 - o 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 -0 ^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 vytvori samy)
- Vytvorime hlavni systemovy ZFS uloziste s mirrorem (neco jako VG u LVM)
 - zpool create tank mirror /dev/sda /dev/sdb
 - Nekdy je dobry 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 bejt 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 vypsatls -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 vykon, zalezi na konfiguraci SLOGu)
 - \circ zfs set compression=on tank (default kompresni algoritmus by mel bejt vcelku rychlej a napr. eliminuje dlouhy retezce nul)
 - #zfs set dedup=on tank #deduplikace zere MOOC ramky = asi 8GB ram na 1TB dat! pomoci 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
 - \circ zfs get all tank
- V tomto ulozisti vytvorime ZFS pro LXC a pripojime
 - zfs create tank/vps
 - $\circ\,$ Pokud pouzivame proxmox, tak tady zkoncime a zbytek naklikame v proxmoxu!
 - o zfs set mountpoint=/var/lib/lxc tank/vps
 - mountpoint nesmi existovat, ZFS si automaticky adresar vytvori pri pripojeni a smaze pri odpojeni.
- Vytvorime novej virtual s quotou (tohle uz dela LXC automaticky)
 - \circ zfs create tank/vps/test
 - $^\circ$ zfs set quota=10G tank/vps/test
- Fsck: zpool scrub tank (prubeh sledujem v zpool status)
- Vsechno smazem (neni dobry napad): #zpool destroy tank
- zpool status Vypise vsechny disky v poolu, jestli nedochazi k datovym chybam, jestli probiha scrub, resilver, atd...
- Log a cache na SSD
 - $\circ\,$ Typy pomocnejch devicu
 - "log", take nazyvany "zil" nebo "slog" je zurnal synchronnich zapisu
 - "cache" neboli "l2arc" ("level 2 arc") je cache pro cteni
 - ani u jednoho nevadi vypadek, v cache jsou jen data precteny z rotacniho disku a vsechno co je v logu je i v RAMce, log by se pouzil jen pri vypadku napajeni, kdy bysme o RAMku prisli
 - Dimenzovani

- 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)
- $\circ~$ log a cache vzdy pridavame jako /dev/disk/by-id/ nemaji metadata!!!
- \circ zpool add tank log /dev/loop0
- zpool add tank cache /dev/loop1
- \circ 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!!!)
 - o zpool detach tank sdb2
 - zpool attach tank sda2 sdb2 (asi dava spis smysl attachnout celej prazdnej disk, oddily si vytvori ZFS samo)
- • Aktivace automatickyho zvetsovani zrcadla (asi dobry udelat uz pred vymenou disku za vetsi) $_{\circ}$ zpool set autoexpand=on tank
- ZVOL (= neco jako LVM uvnitr ZFS poolu, doporuceny na swapy a image virtualu!)
 - zfs create -V 5gb tank/vol (vytvori jednotku /dev/zvol/tank/vol, taky znamy jako /dev/zd0, parametr - s udela zvol bez rezervace diskovyho prostoru v poolu = thinprovisioning)
 - zfs list -t volume vypiseme si zvoly (bez -t volume to vypise vse v poolu)
 - zfs destroy tank/vol
- Autodetekce existujiciho ZFS
 - zpool import nebo zpool import -a pro exportnuty
 - ∘ zfs mount -a
- SWAP na ZFS
 - $^\circ\,$ Swapovani do souboru na ZFS neni 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 zamrznuti)
 - nekdo tvrdi, ze pomuze ZVOLu nastavit logbias=troughput 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 pomoci 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-pvel
 - create,destroy,mount,receive,userprop tank/backup/pvel
 - Verify: zfs allow tank/backup/pve1
- Testovani ZFS
 - Ztest NESLOUŽÍ k testování zfs modulu v jádře!!! Pro otestovani systemu je naprosto nevhodny.

- ztest f /tmp VV vytvori v /tmp blockfily s testovacim ZFS a spusti na nem unit testy (musi tam byt dost mista).
- Jde prodlouzit cas v sekundach, defaultne T 300
- Dalsi 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 dobry omezit kolik RAM muze sezrat ARC (defaultne si bere 1/2 veskery ramky 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 Vypiseme checkpointy (historii)
- chcp ss /dev/sdx1 94 Nastavime checkpoint #94 jako snapshot (= nebude smazan cleanerem a pujde pripojit)
- mount -o ro,cp=94 /dev/sdx1 /mnt/ Pripojime readonly snapshot #94
- chcp cp /dev/sdx1 94 Po odpojeni zrusime snapshot #94 (udelame z nej zpatky checkpoint)

SATA HotSwap

- readlink /sys/block/sda (zjisti na jaky sbernici je disk sda)
- echo 1 > /sys/block/sda/device/delete (odpoji disk sda)
- echo "- -" > /sys/class/scsi_host/host0/scan (najde disky na sbernici 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!

- 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 -0 extents,uninit_bg,dir_index /dev/md5
- fsck.ext4 -yfDtv /dev/md5

Podrobnosti na

 $http://www.debian-administration.org/article/643/Migrating_a_live_system_from_ext3_to_ext4_filesystem_from_ext3_files$

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 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, 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 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:

- $\circ\,$ 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 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 znovu rebootnu do windows a znovu 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/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 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=1715088352

Last update: 2024/05/07 15:25

