Shell und Dateisysteme

http://www.gnu.org/software/bash/manual/bashref.html

https://www.shellscript.sh/

http://www.linux-services.org/shell/

Screen

Siehe auch https://www.mattcutts.com/blog/a-quick-tutorial-on-screen/

Screen Arbeitsablauf

Normalerweise im User, nicht mit sudo

```
screen -S [name]
```

alternativ screen -R (erzeugt neue Session wenn noch keine da oder re-attached an vorhandene)

Weitere Screen Session wird mit Ctrl + a eröffnet.

Wechseln zwischen den Sessions mit Control-a [Nummer], wobei 0 die erste ist

Hilfe: Ctrl A ?

Liste aller Fenster in einer Session Ctrl A w

Von Session abmelden

```
Ctrl-A d
```

Danach aus ssh abmelden, die screen session läuft weiter. Von einem anderen Ort wieder per ssh auf die Arbeitsmaschine und dann wieder an die Session binden

`screen -d -R sessionname` will resume a session that you started in a different location.

Alle Screen Sesions auflisten

screen -ls

Session killen

$ screen -X -S [session # you want to kill] quit

Shadow

Einfügen eines Passwort hashes

```
thommie@odysseus3:~> python -c "import crypt, getpass, pwd; \
>          print crypt.crypt('[klartext]', '\$6\$SALTsalt\$')"
$6$SALTsalt$0[hashwert]
```

Platte voll - grösste Dateien/Verzeichnisse finden

```
 { find ocdata/ -type f -name "*.*" -printf "%s+"; echo 0; } | bc | numfmt --to=si
```

```
 du -hsx -- * | sort -rh | head -10
```

Danach die Dateien in eine Datei kopieren und gezielt löschen

```
xargs rm -r < delete.txt
```

Out of inodes

Eine Platte kann auch wegen wegen ausgegangener inodes voll sein "no space left on device". Dann zuerst inode Status prüfen mit df -i. Danach auflisten, welches Verzeichnis die meisten Inodes belegt:

```
$ for i in /*; do echo $i; find $i |wc -l; done
```

Siehe https://www.ivankuznetsov.com/2010/02/no-space-left-on-device-running-out-of-inodes.html

Wenn es zu viele Einzel-Dateien sind, kommt bei rm -rf eventuell dieser Fehler

```
-bash: /bin/rm: Die Argumentliste ist zu lang (Argument list too long
```

In diesem Fall diee Dateien einzeln finden und übergeben:

```
find -type f -print0 | xargs -0 rm
```

Danach müsste alles wieder OK sein:

```
root@ruru:/mnt/data/log/letsencrypt# for i in /mnt/data/log/*; do echo $i; find $i |wc -l; done /mnt/data/log/apache2 1 /mnt/data/log/dokuwiki 1 /mnt/data/log/joomla 1 /mnt/data/log/letsencrypt 1 /mnt/data/log/owncloud 2 /mnt/data/log/roundcube 1
```

Access Control Lists ACL

siehe https://wiki.ubuntuusers.de/ACL#ACLs-fuer-Dateien-und-Verzeichnisse

setzen

```
setfacl -m u:hermann:rwx,g:hermann:r-x shared/win702/.
```

Settings lesen

```
root@server3:/srv/vm/shared/win702# getfacl /srv/vm/shared/win702 getfacl: Entferne führende '/' von absoluten Pfadnamen # file: srv/vm/shared/win702 # owner: hekneis # group: hekneis user::rwx group::r-x mask::r-x other::r-x default:user::rwx default:user:hekneis:rw- default:group::r-x default:group:hekneis:rw- default:mask::rwx default:other::r-x
```

BTRFS

Standard-Datei-System bei OpenSUSE und SLES

Platte läuft mit Snapshots voll

Die Grundeinstellungen stehen in /etc/snapper/configs/root. Empfehlungen siehe http://www.nrtm.org/index.php/2012/03/13/the-joys-of-btrfs-and-opensuse-or-no-space-left-on-device/comment-page-1/

Im akuten Fall:

```
/etc/cron.daily/suse.de-snapper
```

und

```
/etc/cron.weekly/btrfs-balance.sh
```

btrfs Maintenance

There is btrfsmaintenance package that should be installed by default that provides cron script. Somewhat interesting implementation is, these cron scripts are not installed directly but there is a service that does it. And *this* service is disabled by default

```
systemctl enable btrfsmaintenance-refresh
```

```
systemctl start btrfsmaintenance-refresh
```

and check /etc/cron.{daily,weekly,monthly}

It is configurable in /etc/sysconfig/btrfsmaintenance

I enabled it now (with the default config), which created cron.weekly/btrfs-balance.sh and cron.monthly/btrfs-scrub.sh symlinks.

Alte Snapshots auf einen Satz löschen:

```
for i in `seq 2309 2605`; do snapper delete $i; done
```

Weitere BTRFS Kommandos

Füllgrad feststellen:

```
btrfs filesystem df /
```

Snapshots auflisten

```
snapper list
```

Snapshot Diffs anzeigen

```
snapper diff 71..72
```

Bestimmten Snapshot löschen

```
snapper delete 65
```

Älteren Snapshot aktivieren

```
snapper rollback [ID]
```

BTRFS Dateisystem reparieren

```
fsck.btrfs /dev/…
```

Hohe Last durch BTRFS quota

https://forums.opensuse.org/showthread.php/523354-High-CPU-load-related-to-btrfs-causes-lock-up

https://bugzilla.opensuse.org/show_bug.cgi?id=1017461

Software RAID Reparatur

Ablauf bei Plattentausch

RAID Array anzeigen

```
mdadm –detail /dev/md0
```

und

```
cat /proc/mdstat
```

Platte aus Array entfernen

```
mdadm /dev/md2 –remove /dev/sdk1
```

Ganzes Array stoppen

```
mdadm –manage /dev/md0 –stop
```

XFS

XFS Fehler können unter OpenSuse zum emergency mode führen. Fix:

```
umount /dev/sda3 xfs_repair
```

Wenn das xfs Log überschrieben werden kann

```
xfs_repair -L /dev/sda1
```

Doku: http://docs.cray.com/books/S-2377-22/html-S-2377-22/z1029470303.html und http://xfs.org/index.php/XFS_FAQ

LUKS

Basisinfo: https://de.opensuse.org/SDB:Sicherheit_Verschl%C3%BCsselung_mit_LUKS

https://wiki.ubuntuusers.de/LUKS/

Vor LEAP: Image .img reparieren Die *.key Datei ist auch verschlüsselt, daher geht es nicht direkt siehe dazu https://forums.opensuse.org/showthread.php/501003-How-to-check-encrypted-home-directory-by-fsck `openssl aes-256-cbc -d -in /home/image.key | cryptsetup luks0pen /home/image.img my_home` Danach fsck auf /dev/mapper/my_home Mit luksclose wird das Image geschlossen `cryptsetup luksClose my_home` Ab Leap werden normale LUKS Partitionen benutzt. ==== LUKS Partitionen ab OpenSUSE Leap ==== Die Partition wird über ein Loop Setup ins Dateisystem gemountet: <code> dev/sda1 932G 352G 578G 38% /srv/vm /dev/mapper/cr-auto-1 120G 89G 32G 74% /home tmpfs 3.2G 0 3.2G 0% /run/user/497 </code> Die Zuordnung des gemappten Partition zur Partition auf der Platte steht in in /etc/crypttab <code> cr-auto-1 /dev/nvme0n1p3 </code> Die Befehle für cryptsetup funktionieren nur an der Originalpartition: <code> odysseus3:~ # cryptsetup luksDump /dev/nvme0n1p3 LUKS header information for /dev/nvme0n1p3 Version: 1 Cipher name: aes Cipher mode: xts-plain64 Hash spec: sha256 Payload offset: 4096 MK bits: 256 MK digest: 1f 06 0e 96 37 13 1c 25 d8 03 cd 64 df 2a 67 94 26 a5 6b 69 MK salt: e2 b4 a9 e0 c3 89 84 e6 cc 6f cb d0 fc da 3a 92 ce 52 95 ce c4 ca fa 65 7b bf 06 a8 ea 8a 03 3e MK iterations: 173146 UUID: 7b1703a0-0ff9-4836-b67a-9e9e951b5182 Key Slot 0: ENABLED Iterations: 2770346 Salt: f1 de c8 30 e1 80 5e eb 66 93 0d 03 b6 9a ee 90 75 5b a5 29 1c 50 17 79 18 b9 4d 5f c2 82 61 38 Key material offset: 8 AF stripes: 4000 Key Slot 1: ENABLED Iterations: 3912596 Salt: 89 fc dd 4c 1c f9 6f ff b2 4e 2e 40 03 a7 a4 5f de 7a 7a 08 3e 72 16 58 b2 5f 24 c8 b6 87 86 c0 Key material offset: 264 AF stripes: 4000 Key Slot 2: DISABLED Key Slot 3: DISABLED Key Slot 4: DISABLED Key Slot 5: DISABLED Key Slot 6: DISABLED Key Slot 7: DISABLED </code> Passphrase hinzufügen: <code> cryptsetup luksAddKey /dev/nvme0n1p3 –key-slot 0 </code> Passphrase in Slot gezielt ändern: <code> cryptsetup luksChangeKey /dev/nvme0n1p3 –key-slot 3 </code>

Passphrase entfernen <code> cryptsetup luksKillSlot /dev/nvme0n1p3 –key-slot 3 </code>