Linux/Unix commandline...

Tips'n'Tricks

Bash, Kommandozeile

Subcommands mit Bash

Subcommands können entweder mit **`Backticks`** oder mit **$(Klammern)** durchgeführt werden.

- Backticks sollten nur aus Kompatibilitätsgründen verwendet werden.
- Klammern ansonsten immer !

find

Falsch

```
rm -rf `find /var/tmp/ -type d -name .svn`
```

Richtig

```
find /var/tmp/ -type d -name .svn -exec rm -rf {} \;
```

Tolle und verkannte Befehle und Programme

sar : System Activity Reporter

```
sar -A
```

dd : Disk Dump

```
dd if=/dev/null of=/tmp/bigfile bs=1M count=1 seek=8192
```

ethtool

```
ethtool -p eth0          # same as "ethtool –identify eth0"
ethtool -t eth0          # selftest
ethtool -S eth0          # same as "ethtool --statistics eth0"
```

nc : NetCat

Datentransfer

```
$ nc -l 10.1.1.149 1234 > filename.out       # on the « server »
$ nc 10.1.1.149 1234 < filename.in           # on the « client »
```

Portscanning

```
$ nc -zv 10.1.1.149 20-30
$ nc -zv 10.1.1.149 22 80 443
```

at

```
echo "tar czf /tmp/homes.tgz /home" | at 1337
```

cal

```
$ cal
   Décembre 2012
di lu ma me je ve sa
                   1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31

$ cal jan 1337
    Janvier 1337
di lu ma me je ve sa
          1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31

$ cal dec 9999
   Décembre 9999
di lu ma me je ve sa
          1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31
```

tr

```
echo "123 456 789" | tr " " "\n"
echo "abcd" | tr "a-d" "p-s"
echo "abcd" | tr "a-d" "1-4"
echo "abcd" | tr "abd" "678"
echo "abcd" | tr "ac" "xvpb"
```

awk

```
sentence="year-month-day hours:minutes:seconds :ms \"this is a comment\""
printf "%s\n" "$sentence" | awk '{print $1}'
printf "%s\n" "$sentence" | awk '{print $NF}'
printf "%s\n" "$sentence" | awk 'BEGIN{FS = "\""} {print $(NF-1)}'
printf "%s\n" "$sentence" | awk 'BEGIN{FS = " :"} {print $2}'
printf "%s\n" "$sentence" | awk '/minutes/ {print $0}'

lines="$(echo "line1 line2 line3" | tr " " "\n")"
printf "%s\n" "$lines" | awk '$1 ~ /line2/ {print $0}'
printf "%s\n" "$lines" | awk '/line2/ {print $0}'
printf "%s\n" "$lines" | awk '/e3/ {print $0}'
```

jnettop

```
jnettop -i eth0
```

find

```
find /var/ -maxdepth 1 -type f -exec ls -lh {} \;
```

iostat

```
iostat -k5 /dev/sda
```

sed (Stream Editor)

```
faw@mahaliah:~$ echo "123456" | sed "s/345/abcdefgh/"
12abcdefgh6
faw@mahaliah:~$ echo "/1/2/3/3/4/5" | sed "s/\//XXX/"
XXX1/2/3/3/4/5
faw@mahaliah:~$ echo "/1/2/3/3/4/5" | sed "s/\//XXX/g"
XXX1XXX2XXX3XXX3XXX4XXX5
faw@mahaliah:~$ echo "/1/2/3/3/4/5" | sed "s_/_XXX_g"
XXX1XXX2XXX3XXX3XXX4XXX5
faw@mahaliah:~$ echo "/1/2/3/3/4/5" | sed "s_^/_XXX_g"
XXX1/2/3/3/4/5
faw@mahaliah:~$ echo "/1/2/3/3/4/5" | sed "s_4/5\$_XXX_"
/1/2/3/3/XXX
faw@mahaliah:~$ echo "/1/2/3/3/4/5" | sed "s_4/5\$_XXX_"
/1/2/3/3/XXX
faw@mahaliah:~$ echo -e "123\n456\nabc\ndef" | sed "s_abc_xyz_g"
123
456
xyz
def
faw@mahaliah:~$ echo -e "123\n456\nabc" | sed -e "s_abc_xyz_g" -e "s/123/999/" -e "s|
456|654|"
999
654
xyz
```

screen (« GNU screen »)

```
C-a ?       show help
C-a d       detach session
C-a "       list windows
C-a l       refresh
C-a F       fit size
C-a c       create a new window
C-a 0       go to window 0
...
C-a 9       go to window 9
C-a '       asks for a window number
C-a p       go to previous window
C-a n       go to next window
C-a C-a     go to previously displayed window
C-a k       kill window
C-a X       kill current region
C-a S       split region
C-a |       split region vertically
C-a TAB     switch to next region
C-a Q       « unsplit »
C-a x       lock terminal
C-a *       show a listing of all currently attached displays

screen -ls         # list sessions for connected user
screen -r          # re-attach session (if only one)
screen -r ID       # re-attach session « ID »
screen -x          # shared session
```

locate

```
locate abc*.jpg
```

Die *locate* Datenbank wird mit **updatedb** aktualisiert.

grep

Der Begriff *grep* steht für ***g**lobal/**r**egular **e**xpression/**p**rint* oder auch ***g**lobal search for a **r**egular **e**xpression and **p**rint out matched lines*. Historisch entwickelte sich der Name aus dem Kommando g/re/p des Unix-Standardeditors ed.

```
printf "Zeile1\nZeile2\nZeile3" | grep -v -e "1" -e "3" | grep --color "eil"
```

pgrep : PID-Suche

```
$ pgrep mysql
3436
3777
$ ps -ef | grep mysql
root      3436     1  0 nov.27 ?       00:00:00 /bin/sh /usr/bin/mysqld_safe
mysql     3777  3436  0 nov.27 ?       00:05:12 /usr/sbin/mysqld [blah] --port=3306
```

pbzip2 : Parallel bzip2

```
faw@mahaliah:~$ cat /proc/cpuinfo | grep ^proc
processor : 0
processor : 1
processor : 2
processor : 3
faw@mahaliah:~$ dd if=/dev/urandom > /tmp/testfile.big bs=1M count=128
134217728 octets (134 MB) copiés, 8,19632 s, 16,4 MB/s
faw@mahaliah:~$ ll /tmp/testfile.big
-rw-r--r-- 1 faw faw 128M déc.   4 11:35 /tmp/testfile.big
faw@mahaliah:~$ time bzip2 /tmp/testfile.big

real      0m19.317s
user      0m19.137s
sys       0m0.116s
faw@mahaliah:~$ time pbzip2 /tmp/testfile.big

real      0m6.526s
user      0m24.414s
sys       0m0.236s
```

Nur sinnvoll auf Rechnern (bzw. VM's), die mehr als nur ein Core haben !

tail

```
tail -f /var/log/syslog
```

less : Warum eine Datei editieren, wenn man sie nur betrachten will ?

```
nano /var/log/syslog            # falsch !
vi /var/log/syslog              # falsch !
emacs /var/log/syslog           # falsch !

less /var/log/syslog            # richtig !
```

Seltsame Fehler

Geschütztes Leerzeichen

Ältere Systeme

```
Faw@xxlws1:~$ echo "123" | grep 2
-bash:  grep: command not found
```

Neuere Systeme

```
faw@lianli:~$ echo "123" | grep 2
-bash: $'\302\240grep': command not found
faw@lianli:~$ echo "123" | grep 2
123
```

vi/vim

Replace

```
:10,42 %s/aaa/bbb/gc
```

Macros

Makro in den ersten Register aufzeichnen :

```
[ESC]q1
(do what you want the macro to do)
q
```

Makro im ersten Register aufrufen :

```
@1
7@1
```