Linux basic

Shell (1)

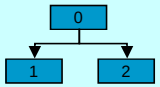
Version 2012-01-31-eng
Dipl.Inf. (FH) Bernd Schwägerl, Dolby Labs
Dipl.Inf. (FH) Stefan Edenhofer

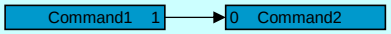
Helpfunctions	
man [Section] [Command]	Helppage for the section of a command
man -k [term]	Short description for the spesific term
man -a [Command]	Shows the man pages for all sections of the command
man -f [Command]	Display all occurences of the spesified command with shortdescription
whatis [Command]	Display the section related to the command

Data	
file [Filename|Directory]	Display the filetype
cp [Source] [Destination]	Copy a file or directory to destination
scp [user1]@[host1]:[absPfad] [user2]@[host2]:[absPfad]	
rm [Option] [Target]	Delet files or directories
-R	Subdirectories
-f	Force, ignore nonexisting files
mv [Source] [Destination]	Move files or directories to destination
ln -s [Target] [Link]	Create links to file or directory
cat [Filename]	Displays data
cat >[Datei]	Datei interaktiv erzeugen (Ende:<strg><d>)
touch [Filename]	Create file
tar -xfzp [Filename].tar.gz	.tar.gz. Unpack data
find [Path] [Option] [target]	Find a file in specified path
-name	Filename
-size +/-n; -mtime +/-n	filesize, fileage

Directories	
ls -la	Displays content of directory
mkdir	Create directoy (delete with rm -Rf)
du	Display diskusage
-s	total
-h	humanreadable
-a	display all, also files
pwd	Displays current location
cd [Directory]	Change to target directory
cd /	Change to root of system
cd ~	Change to current users homedirectory
cd ..	Change to one level above
"." is the current directory. For the root user it is NOT part of the searchpath (security reasons)	

User information	
whoami / who am i	Displays the current user (Multiuser-OS!)
who / users	Displays all user currently logged on
Which devices: ps aux | grep bash | cut -b37-43 | sort | uniq	

Input and output	
	0 Standardinput (Tastatur) stdin 1 Standardoutput (Screen) stdout 2 Standarderror (Screen) stderr
[cmd] 1> [file]	Redirect standard output
[cmd] 2> [file]	redirect standard error
[cmd] 1>[file1] 2>[file2]	Redirectinput and output to file
[cmd] 1>[file] 2>&1	stderr -> stdout; stdout -> file
[cmd] >> [file]	Append to file[file]
[cmd] < [file]	stdin is the input [file]

Input and output	
	
[cmd1] | [cmd2]	Output from cmd1 is Input to cmd2
[cmd1] | tee [file]	Output to screen and file [file]
Note: for filtering see "grep"	

Shell and Environment variables	
Shellvariable	Only valid in current shell(since some commands use a separate shell, the variable is not valid there)
Environmentvariable	An Environment variable will be inherited by subshells. But Parent shells ist not inherited!
[VARNAME]=["value"]	Set a Shellvariable
$[VARNAME]	Use the variable
export [VARNAME]	Converts the Shellvariable to Environmentvariable
unset [VARNAME]	Delete the variable

Systemresources	
free	Outputs the free memory
df [Devicefile]	Lagringsplass harddrive/CDs
	First IDE-HD: /dev/hda
	First SCSI-HD: /dev/sda
top	Utilization, running processes
-n 1 -b	one interation - batchmode

File permissions/ see also special rights/privileges	
ls -la	Outputs the file permissions
chmod 0421 [Datei|Dir]	Absolute change in permissions
chmod o+r g-x u+x [Datei|Dir]	Relative change in permissions
chown [user]:[group] [datei]	Change owner and group owner
umask 022	"Default permissions" (Complementary vector)
"-rwxrw--r-- 1 user group ..."	
The file is owned by the user and the user group. user may write (w = write = 2), read (r = read = 4) and execute (x = execute = 1), the group members of group may "r and w". ALL the other "r".	

Fitler / Comparing files and streams	
sort	Sort the rows
cut	cut fildes per line
cut -f n -d " "	Space delimiter, field n
uniq	ignores repeated identical lines
diff [-u|-dtw] [datei1] [datei2]	Compare two files
wc -l	Word count -l = lines
cmp [Datei1] [Datei2]	Compare rows
grep [Muster]	Specifies what to output should contain
grep -v [Muster]	…specifies what it shouldn't be
tail [Datei]	sends end of file to output
head [Datei]	Sends beginning og file to output
split --bytes=1024 [Datei]	Divide in 1KB files/streams
cat [Dat1] [Dat2] > [Datei]	Joining data/streams

Regular expressions / patterns	
Characters and charaterclasses:	
.	Anny character
^	Beginning og line
$	End of line
[0-9]	Characterclass (0,1,2,3...9)
[abxyz]	Characterclass (a,b,x,y,z)
[[:blank:]]	All Whitespaces
[[:alnum:]]	Alle lletters and numbers
[[:alpha:]]	Alle letters
Repetitions /Occurence:	
?	One or nothing
*	Noone or sveral
+	One or multiple
{n}; {n,}; {n,m}	exactly n times, n or more frequent at least n times, and most m times

Special rights		
Data		**Directory**
	1	Included files can only be deleted by fileowner
will run with the rights as the owner	2	
Will run with the rights of the owner group	4	New files in this folder will have the same rights as the groupwoner
"The special rights are simply a chmod forward. (For an absolute set!) Example: chmod 4000 [file | dir]. Displayed (lsmod), the special rights as s the owner (4), s in the group (2) and t (1). "		

Linux basic

Shell (2)

Version 2012-01-31-eng

Dipl.Inf. (FH) Bernd Schwägerl, Dolby Labs
Dipl.Inf. (FH) Stefan Edenhofer

Partitioning	
directories need to be on the "/"-Partition	/bin (important commands) /sbin (important root-commands) /dev (access handles for devices) /etc (all configuration files) /lib (Libraries) /proc (virt.!! Kernel communication)
Directories, that should be on the "/"-Partition	/root (Homedirectory of the admin user)
Directories that should be on different partitions	/usr (Unix System Ressource) /var (variable data, Log & Spool) /home (Homedirectories User) /tmp (Temporary files) /boot (Bootloader, Kernel) /swap ("swap partition")

Runlevels	
Configuration	/etc/inittab /etc/init.d
init [runlevel]	Change of "runlevels"
runlevel	Output of former and current runlevel
runlevels:	(most common)
0	stop the computer
1	single user
2	multi user w/o network
3	multiuser with network
4	(unused)
5	graphical user interface
6	restarting the system

Links (shortcuts)	
There are 2 different kinds of links (shortcuts): soft- and hardlinks. Hardlinks got the EXACT same meta data like the original file. Softlinks got their own meta data (inode)	
ln [target] [linkname]	create hardlink
ln -s [target] [linkname]	create softlink
ls -i	show directory with inodes
Nearly everything works with softlinks - They are mostly the better choice	

fast file search	
Just walks through the searchpath	
which [filename]	checks for executable files (result: first hit!)
whereis [filename]	Shows all files in searchpath (multiple)
PATH="$PATH:[/newDir]"	Extend searchpath
export PATH	Inherit the PATH to subshells
Searching via a database - just indexed files will be found.	
updatedb	refresh index
locate [file]	find indexed files
slocate [file]	same, but just if access exists
Security aspect: Just gather access to slocate to users	

hostinformation	
uptime	time, the system is up
hostname	print hostname
hostname -d	print domain name (DNS not Mickysoft)
date	date / time of the host

VI - the standard editor	
There are two important modi (Command <ESC>and Insert <ESC><I>)	
d[n][d|w|c]	delete n lines, words, chars
y[n][y|w|c]	copy n lines, words, chars
p	insert what has been deleted or copied
x	delete one character
r	replace one character
:[n]	goto line n
:$	goto file end
$	goto line end
0	goto line start
/pattern	search for "pattern" (forward)
?pattern	search for "pattern" (backwards)
:1,$s/m1/m2/ig	replace (s=substitute) between line 1 and file end ($) - mutliple occurances of "m1" with "m2" per line (g=global) and no matter if capitals or small (i=ignore)
:wq!	(w=write), (q=quit) - (! = without ask back if write protected etc.)
J	Add a line below
<strg>g	show statistics
u	undo - you can go back till the point you started the VI
k h l j	If there are no cursorkeys on the keyboard - or the terminal is misconfigured - NAVIGATION

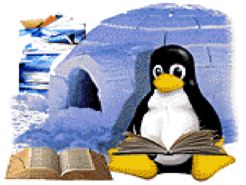
Tipps &Tricks
Show all users of the system (also daemon/system users)
cat /etc/passwd | cut -d":" -f1
show process changes of the last "SECONDS"
diff <(ps ax) <(sleep [SECONDS]; ps ax)
Show all process-Ids, that contain in the ps-output "MUSTER"
ps aux | grep [MUSTER] | grep -v grep | cut -b10-14
Mail to EMAIL with subject SUBJECT - Mailtext: All filenames of [DIR] - Attachement is the aggregated content (tar) of [DIR] in demo.tar.
tar cvf demo.tar [DIR] | elm -s [SUBJECT] [EMAIL] -A demo.tar

Directorysynchronisation via different hosts (backup!)
Encrypted (-e ssh) ; archived (a), recursive (r), verbose (v)
rsync -avr -e ssh [user@host:]:/srcpath [user@host:]/destpath

Prozesse	
ps aux	Show all running processes
pstree	dito with dependencies
kill [signal] [PID]	send "kill" (or other) signal to process
killall [Muster]	All processes containing "MUSTER"
jobs	Show jobs and jobids
bg [cmd1]	execute in background
[cmd1] &	
fg [JOBID]	bring back to foreground
<strg><z>	send to background Attention! Restart with bg [JOBID] - because the process is on PAUSE
nohup [cmd1]	Continue executing command after logout
Processpriotities	
processes can be executed with different priorities. There is a nice-value (-20 = highest priority = lowest value) to 19 (loweset priority = highest value) Defaultvalue is 0;increase.	
nice -n[VALUE] [CMD]	Command CMD with a Nice-value of VALUE. (Common Users just >-1)
renice +/-[DELTA] [PID]	Change the nice value (Common users nur >-1!!!)
renice +/-[DELTA] -u root	Change all process that belong to root.

Network	
Set ipV4-address	
ifconfig eth[n] netmask [Maske] broadcast [BC-Adr] [IP]	
ifconfig [NIC] up	bring NIC up
ifconfig [NIC] down	bring NIC down
route to ZIEL with netmask via gateway	
route add –net/host [ZIEL] netmask [netmask] gw [gateway]	
delete route	
route del –net [ZIEL] netmask [netmask]	
route	Alle Routes anzeigen

Dependencies of commands	
cmd1; cmd2	ececute cmd1 and than cmd2
cmd1 && cmd2	cmd2 just if cmd1 was success
cmd1 || cmd2	cmd2 just if cmd1 was without success (error code)
cmd1 & cmd2	start cmd1 in background and cmd2 simultanously in foreground
helps e.g. With kernel compiling (time consuming): &&	



Linux basic

network

Version 2012-01-31-eng

Dipl.Inf. (FH) Bernd Schwägerl, Dolby Labs
Dipl.Inf. (FH) Stefan Edenhofer

query and set network settings (IPv4)	
Netzwerk-Interface-Card (NIC):	
ifconfig	query current settings
ifconfig [interface] [adresse] [parameter]	
ifconfig eth0 172.17.21.11 netmask 255.255.255.0 broadcast 172.17.21.255	
most important parameters	
broadcast [BC-Adresse]	set broadcast address
down	deactivate the interface
mtu [number]	set MTU
netmask [MASK]	set netmask
up	activate interface
Routing:	
arp -n	show ARP-table without nameresolution
route -n	Show Routing table without name resolution.
route [add|del] [-net|-host] [Ziel] netmask [NM] gw [Gateway] dev [IF]	
route add -net 172.16.1.0 netmask 255.255.255.0 gw 172.17.21.1 dev eth0	
Activate routing (ipv4)	
echo "1" > /proc/sys/net/ipv4/ip_forward	
Deactivate routing (ipv4)	
echo "0" > /proc/sys/net/ipv4/ip_forward	
Show if routing is active (ipv4)	
cat /proc/sys/net/ipv4/ip_forward	
Additional important options of route	
metric [METRIK]	The metric sets the "costs" of a routing direction. The higher the metric the less propable it is taken.
mss [MAXSIZE]	The maximum fragment size (packet size) - sometimes needed for different topologies/hardware. This is also needed for some (bad) firewalls.
Attention! Every information you set with those tools - are being lost after a reboot. Workaround (e.g.) a startscript (see: programming)	
SuSE-configuration files	/etc/sysconfig/network/ifcfg....

netcat / nc
netcat, or. nc enables the communication with a tcp- or udp-connection via stdin/stdout, e.g. From shellscripts
Check cleartext protocols (Example for HTTP)
printf 'GET / HTTP/1.0\n\n' | netcat -w [TIMEOUT] [SERVER] [PORT]
Redirect local ports to remote computers (see also rinetd)
entry in /etc/inetd.conf (Port <-> program usage!): [L-Port] stream tcp nowait [User] nc nc -w 5 [R-host] [R-port]
With netcat / nc you can create shellprogramms that act as a daemon. netcat need to be compiled with (-DGAPING_SECURITY_HOLE)
nc -l -p [PORT] -e [Pfad zu einem Shellskript]

Important tools you should know	
ping	Test of reachability
-c [Anzahl]	(count) count of pings
-n	(numeric) no names
traceroute -n	Route of a packet (=tracert) (-n without name resolution)
tcpdump -i [NIC]	Is a small sniffer
ssh	SSL encrypted access
scp [user1]@[host1]:[absPfad] [user2]@[host2]:[absPfad]	
	encrypted remote copy (see a ls shell - cheat sheet)

Nameresolution (nslookup / dig)	
nslookup is obsolete	
nslookup	interactive nslookup-shell with a connection to the standard nameserver
set q=[TYP]	(see resource records RR)
server [IP/NAME]	Use "IP/NAME"-Server
dig -t [TYP] @[NServer]	Query Nameserver NSERVER for Ressource Record of TYP

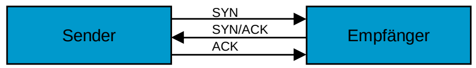
Ressource Records (TYP) for nslookup and dig	
SOA	Managementinformation
A	Adressentry (NAME .-> IP)
NS	Nameserver
CNAME	Aliasname
PTR	Adressentry (IP -> NAME)
MX	MaileXchanger - Mailserver for domain
HINFO	Hostinfos (if set)
TXT	comment
ALL	(nearly) all - (Is a QueryType - but NOT a Ressource Record)

show network relevant Information (netstat) (verbose -v)	
netstat -s	Statistics
netstat -c	continous output
netstat -l	Listening sockets
netstat -a	Sending and recieving sockets
netstat -p	Include program names

Space for improvement	

TCP-Header		
1. Byte	2. Byte	
Source Port (Dienstkennung)		
Destination Port (Dienstkennung)		
Sequenz Nummer (4 Byte)		
Sequenz Nummer (4 Byte)		
Acknowledge Nummer (4 Byte)		
Acknowledge Nummer (4 Byte)		
Daten Offset	Reserviert (6 Bit) |	Flags (6 Bit)
Window (Puffergröße des Empfängers)		
Checksumme (Prüfsumme Header UND Daten)		
Urgent Pointer (Zeigt auf "vorrangig zu bearbeitende Daten")		
Options (Always n-fold of 32-bit)		
Options + Padding ("to fill up to a n-fold of 32")		
file content		

Successful TCP-connection (3-way-handshake)



TCP-Flags:	
URG	1 = Urgent Pointer-field need to be read(„priority“ data)
ACK	1 = Acknowledge number field need to be read
PSH	1 =data need to be send immediatly to the corresponding service
RST	1 = connection should be stopped
SYN	1 = connection should be established
FIN	1 =shows final termination of connection

Gängige TCP-Ports	
(TCP) 20	data channel of FTP
(TCP) 21	controll channel of FTP
(TCP) 22	Secure Login
(TCP) 23	telnet
(TCP) 25	Simple Mail Transfer Protocol (SMTP)
(TCP/UDP) 53	DomainNameService (DNS)
(TCP) 80	HyperTextTransferProtocol (HTTP)
(TCP) 110	PostOfficeProtocol 3 (POP3)
(TCP) 443	https
Most important port/service relations can be found in /etc/services	

Reference to "Linux basic - shell(1) ; shell (2)"!
(Variablen, Pipes, concatenation of commands....)

Variables	
$* oder "$@"	All transfered parameters (list)
$* oder $@	All transfered parameters (word)
$#	count of parameters
$?	return value of last executed command
$$	process-id of current shell
$!	process-id of last executed subprocess
$0	The program name
$1 - $9	the first 9 parameters

shift shifts all transfered parameters to the left. The most left value drops down according to the "lemming method - just over the rock". The former $2 is now $1. The former value of $1 has been lost. (It was the lemming that jumped)

[VARIABLE]="[WERT]"	set Variable VARIABLE to value WERT
$[VARIABLE]	access the value of VARIABLE

Rechenoperationen mit dem Befehl expr	
Addition	ERGEBNIS=`expr 1 + 2`
Subtraction	ERGEBNIS=`expr 1 - 2`
Multiplication	ERGEBNIS=`expr 2 * 2`
integer division	ERGEBNIS=`expr 8 / 2`
Leftover of division	ERGEBNIS=`expr 7 % 2`
binary OR	ERGEBNIS=`expr 1 | 0`
binary AND	ERGEBNIS=`expr 1 & 1`
comparre operations	< <= = == != >= >

Those examples shows also, how to set a variable (ERGEBNIS) to the return value of a command. The same way you can use to get a VARIABLE comntaining the return value of other commands, e.g. "date", "ifconfig" etc. You can access the variable with pre-adding a "$" (e.g. $ERGEBNIS)
EXAMPLE:
echo $ERGEBNIS - prints content of ERGEBNIS to screen

results of compare operations are 1=true 0=false

Howto create "Standard-For-loop"

```
for ZAEHLER in `seq 1 100`
do
echo "Durchlauf $ZAEHLER"
done
```

`[BEFEHL]` executes the command and contains the return value

```
ZAEHLER=1
while [ $ZAEHLER -ne 101 ]
do
ZAEHLER=`expr $ZAEHLER + 1`
echo "Durchlauf: $ZAEHLER"
done
```

for-loop

The for-loop is not a standard counting-mechanismn like in other programming languages, it walks just through a "list". A common for-loop need to use while and arithmetic operations "expr"-statements

```
for [VARIABLE] in [LISTE]
do
....
done
```

while-loop

A while statement is just being executed if the statement "Bedingung" is true.

```
while [Bedingung]
do
....
done
```

until-loop

An until statement is being executed at least once - and until the condition "Bedingung" is true

```
until [Bedingung]
do
....
done
```

Startscript(goal of the progtamming excurse)

```
#! /bin/sh
# 2004-12-20-bs-0001 bs@muekno.de
start ()
{
Datum=`date`
echo "Starte den Dienst um $Datum"
}
stop ()
{
Datum=`date`
echo "Stoppe den Dienst um $Datum"
}
case $1 in
        status) PIDS=`ps aux | grep sdjadhagh | grep -v grep | cut -b9-14`
            if [ "$PIDS" = "" ]
            then echo "Is not running!"
            else echo "Is running with PID(s): $PIDS"
            fi
            }
        start)  start ;;
        stop) stop ;;
        restart) stop; start ;;
        *) echo "Usage $0 [start|stop|restart|status]
esac
```

Startscripts need to react on the parameter ($1) start and stop. They are located in /etc/init.d. Starten (Stoppen = K99startskript) in Runlevel 3:
ln -s /etc/init.d/startskript.sh /etc/rc.d/rc3.d/S99startskript
Attention: see also start-stop-daemon

Linux basic programming

Version 2012-01-31-eng

Dipl.Inf. (FH) Bernd Schwägerl, Dolby Labs
Dipl.Inf. (FH) Stefan Edenhofer

statements w conditions

if-Statement:

```
if [CONDITION]
then
 echo „condition is true=0“
else
 echo „condition is false=1“
fi
```

case-Statement:

```
case [VARIABLENVALUE] in
                    1) echo „value is 1“ ;;
                    2) echo „value is 2“ ;;
                    *) echo „value was something different" ;;
esac
```

subroutines (functions)

define functions

```
[functionname] ()
{
# example to return an value
echo $1
}
```

execute functions

```
[functionname] PARAMETER1 PARAMETER2
```

Teststatements (most common way of conditions)

[$A = $B]	$A equal $B (Chars, Strings)
[$A != $B]	$A not $B (Chars, Strings)
[$A -eq $B]	$A = $B (numeric)
[$A -ne $B]	$A nicht $B (numeric)
[$A -lt $B]	$A < $B (numeric)
[$A -gt $B]	$A > $B (numeric)
[$A -le $B]	$A <= $B (numeric)
[$A -ge $B]	$A=> $B (numeric)
[-r file]	file exists and is readable
[-w file]	file exists and is writeable
[-x file]	file exists and is executable
[-f file]	file is a FILE und exists
[-d file]	file is a directory and exists
AND/OR connections of test statements	
[$A > $B -o $C < $B]	OR
[$A > $B -a $C < $B]	AND

Every scriptexecution is an execution of an own shell, so common variables are being lost in subscripts (Solution - ".[SPACE]/script")