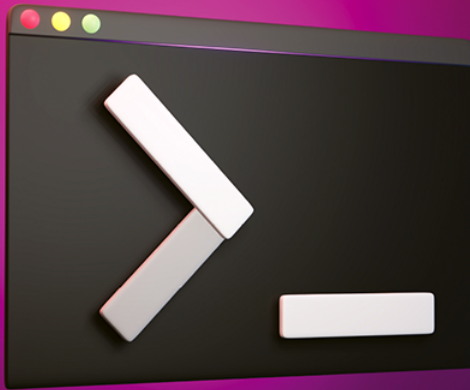


cat, ln, nano  
stat, tee, a  
bunzip2, he  
iconv, attr  
getcap, con  
at, batch, z  
iotop, addg  
usermod, ba



```
user$ rdfind verz1 verz2
Now scanning "verz1", found 211 files.
Now scanning "verz2", found 215 files.
Now have 426 files in total.
Removed 0 files due to nonunique device
Total size is 85754591 bytes or 82 MiB
Removed 103 files due to unique sizes from
Now eliminating candidates based on first
  removed 4 files from list. 319 files left.
Now eliminating candidates based on last
  removed 0 files from list. 319 files left.
Now eliminating candidates based on sha
  removed 2 files from list. 317 files left.
It seems like you have 317 files that are not
Totally, 39 MiB can be reduced. Now make
```

# Linux Command Reference

## Shell Commands from A to Z

Michael Kofler



Rheinwerk  
Computing

# Overview of Commands Sorted by Topic

## Managing Files

---

cat	Combines multiple files into one file .....	64
cd	Changes to another directory .....	65
chgrp	Changes the group membership of a file .....	69
chmod	Changes the access bits of a file .....	70
chown	Changes the owner of a file .....	72
cp	Copies files .....	78
dircolors	Helps with the configuration of the <code>ls</code> colors .....	95
erd	Combines <code>ls</code> , <code>tree</code> , <code>find</code> , and <code>du</code> .....	112
file	Attempts to determine the type of a file .....	120
inotifywait	Waits for file attributes to be changed .....	173
install	Combines <code>cp</code> with <code>chmod</code> and <code>chown</code> .....	175
j	Changes to another directory .....	185
ln	Creates fixed and symbolic links to files .....	197
ls	Displays the table of contents .....	203
mkdir	Creates a new directory .....	219
mv	Moves files or changes their names .....	241
namei	Shows the access rights of all directories of a file .....	248
rdfind	Searches for and eliminates duplicates .....	308
rename	Assigns new names to multiple files .....	312
rm	Deletes files .....	318
rmdir	Deletes directories .....	320
rsync	Synchronizes directories .....	324
stat	Provides detailed information on files .....	356
tee	Duplicates the standard input .....	370
touch	Creates an empty file or changes the modify time .....	376
tree	Represents the directory hierarchy .....	379
truncate	Reduces/enlarges a file .....	379
umask	Controls which access rights new files and directories receive ...	383
z	Changes to another directory .....	418

## Finding Files

---

ack	Serves as a <code>grep</code> alternative for programmers .....	28
erd	Combines <code>ls</code> , <code>tree</code> , <code>find</code> , and <code>du</code> .....	112

find	Searches for files by name, date, size, and so on .....	121
fzf	Enables an interactive search for files .....	139
grep	Searches for text in a text file .....	155
locate	Searches for files in a database prepared for this purpose .....	199
updatedb	Updates the search database for <code>locate</code> .....	388
whereis	Searches for files in predefined directories.....	407
which	Searches the <code>PATH</code> directories for commands.....	407

## Compressing and Archiving Files

---

bunzip2	Decompresses *.bz2 files.....	61
bzip2	Compresses files; more powerful than <code>gzip</code> .....	61
compress	Compresses files .....	76
cpio	Transfers archive files between file systems .....	80
gunzip	Decompresses *.gz files .....	162
gzip	Compresses files; more powerful than <code>compress</code> .....	162
lz4	Compresses files; significantly faster than <code>gzip</code> .....	214
lzop	Compresses files; significantly faster than <code>gzip</code> .....	214
tar	Creates a *.tar archive.....	364
uncompress	Decompresses files compressed using <code>compress</code> .....	384
unxz	Decompresses *.xz files .....	386
unzip	Decompresses *.zip files.....	386
xz	Compresses files; more powerful than <code>bzip2</code> .....	417
zip	Creates a Windows-compatible ZIP archive.....	420
zipinfo	Displays the contents of a ZIP archive .....	420

## Processing Text Files

---

awk	Programming language for text analysis .....	48
bat	Displays a text file with syntax highlighting .....	53
cat	Outputs a file or combines multiple texts .....	64
column	Formats information column by column.....	75
csplit	Splits the text into individual files at predefined points .....	82
cut	Extracts columns from each line of the text .....	85
delta	Compares two texts and highlights the differences in color .....	90
diff	Compares two texts .....	93
expand	Replaces tab characters with spaces.....	116
fold	Breaks down long lines of text into shorter ones .....	131
fx	Interactively scrolls through JSON files .....	138
grep	Searches for texts within the file .....	155
head	Outputs the first lines of the file .....	164

iconv	Changes the character set of text files.....	171
jq	Extracts data from JSON documents .....	189
less	Displays text files page by page (also backwards) .....	195
more	Displays text files page by page .....	226
multitail	Tracks the changes of multiple files .....	241
paste	Combines multiple texts line by line .....	281
patch	Changes text files according to a diff file .....	281
recode	Converts between different character sets .....	311
sed	Stream editor (programmable editor) .....	331
sort	Sorts files .....	348
split	Splits a file into subfiles of a specified size.....	350
strings	Extracts character strings from binary files .....	357
tac	Outputs text in reverse order, that is, the last line first .....	363
tail	Outputs the end of a file.....	363
tr	Replaces predefined characters with other characters.....	377
unexpand	Replaces spaces with tab characters.....	384
uniq	Eliminates multiple lines in a text file .....	385
zcat	Outputs a compressed text file .....	419
zless	Displays a compressed text file (also backwards).....	419
zmore	Displays a compressed text file page by page .....	419

## Access Control Lists (ACLs) and Extended Attributes (EAs)

---

attr	Manages the additional attributes of a file .....	48
chacl	Manages the ACLs of a file .....	65
getcap	Determines the capabilities of a file.....	139
getfacl	Determines the ACLs of a file .....	140
getfattr	Determines the additional attributes of a file.....	140
setcap	Changes the capabilities of a file .....	333
setfacl	Changes the ACLs of a file .....	334
setfattr	Changes the additional attributes of a file.....	337

## Converters

---

convert	Converts graphic files.....	76
dvips	Converts a DVI file into PostScript format .....	108
enscript	Converts text files into PostScript format.....	111
epstopdf	Converts EPS files into PDF files .....	112
exiftool	Reads or changes Exif data in JPG files.....	115
ffmpeg	Converts video files.....	119
iconv	Changes the character set of text files.....	171

lame	Generates MP3-compatible audio files .....	193
magick	Converts graphic files. ....	76
mogrify	Changes parameters of an image file. ....	226
pandoc	Creates documents from Markdown files. ....	272
paps	Converts UTF-8 text files to PostScript format .....	274
pdf2ps	Converts PDF files into PostScript files. ....	282
pdftimages	Extracts images from PDF files .....	283
pdftk	Manipulates PDF files .....	284
pdftops	Serves as an alternative to pdf2ps. ....	285
pdftotext	Converts a PDF document into a plain text file. ....	286
pdfunite	Merges several PDF files into a new document. ....	286
ps2pdf	Converts PostScript files into PDF files. ....	296
recode	Changes the character set of text files. ....	311

## Managing Processes

---

at	Executes a job at a predefined time .....	46
atq	Lists jobs that are to be executed later .....	47
atrm	Deletes a job that is to be executed later .....	48
batch	Executes a job as soon as the system is idle. ....	53
bg	Continues a process in the background. ....	54
chroot	Starts a shell in a modified root directory. ....	73
crontab	Helps with the administration of custom crontab entries .....	80
disown	Detaches a process from the shell. ....	96
fg	Continues a process in the foreground .....	120
fuser	Determines the program that accesses a file. ....	136
glances	Modern alternative to top .....	148
halt	Quits Linux and stops the computer. ....	163
history	Displays the last commands executed in the bash .....	165
htop	Displays a list of all processes every five seconds. ....	167
iftop	Monitors the network activity. ....	172
ionice	Controls the I/O priority of a process .....	176
iotop	Shows the processes with the highest I/O activity .....	177
kill	Sends signals (usually to terminate processes) .....	191
killall	Like kill; mentions the process by name. ....	192
ldconfig	Updates the cache file for the library search .....	194
ldd	Provides all required libraries of a program .....	195
lsof	Lists open files and the assigned processes .....	209
nice	Starts a program with reduced priority .....	260
nohup	Starts a process that is detached from the shell. ....	265
parallel	Starts multiple similar jobs in parallel .....	274

pidof	Determines the process number of a program .....	287
powertop	Analyzes the energy consumption of running processes. ....	292
ps	Displays the list of running processes .....	294
pstree	Like ps; makes the dependencies more visible .....	298
reboot	Quits Linux and reboots the computer .....	311
renice	Changes the priority of a running process .....	313
sudo	Runs a program as root. ....	358
timeout	Limits the execution time of a command .....	375
top	Displays a list of all processes every five seconds. ....	376
uptime	Shows how long the computer has been running .....	388
watch	Executes a command periodically and displays the output .....	403

## Managing Users and Groups

---

addgroup	Sets up a new group (Debian/Ubuntu) .....	32
adduser	Sets up a new user (Debian/Ubuntu) .....	32
chage	Changes the expiration date of an account or password .....	65
chgrp	Changes the group membership of a file .....	69
chown	Changes the owner of a file .....	72
chpasswd	Changes a user password without interaction .....	72
chsh	Changes the default shell of a user .....	74
delgroup	Deletes a group (Debian/Ubuntu) .....	90
deluser	Deletes a user (Debian/Ubuntu) .....	91
groupadd	Sets up a new group .....	159
groupdel	Deletes a group .....	159
groupmod	Changes group properties .....	159
groups	Displays the groups of the current user .....	160
gpasswd	Changes group passwords .....	150
id	Displays the current user and group ID .....	171
last	Reveals who was last logged in on this computer .....	194
lastb	Lists which login attempts have recently failed .....	194
makepasswd	Generates a new, random password .....	216
mkpasswd	Generates a new, random password (CentOS, Fedora, RHEL) .....	225
newgrp	Changes the active group of a user .....	254
newusers	Sets up multiple new users .....	255
passwd	Changes the password of a user .....	280
pwgen	Generates easy-to-remember passwords .....	300
useradd	Sets up a new user .....	389
userdel	Deletes a user .....	389
usermod	Changes user properties .....	390
vigr	Edits /etc/groups .....	393

vipw	Edits /etc/passwd .....	393
visudo	Edits /etc/sudoers .....	393
who	Provides information about the logged-in users .....	408

## Administrating the File System

badblocks	Tests whether data carriers contain defective sectors .....	52
blkid	Returns the UUID and the name of a file system .....	55
btrfs	Administrates a Btrfs file system .....	58
cfdisk	Partitions a hard disk .....	65
cryptsetup	Sets up an encrypted device .....	81
dd	Copies data blocks between devices .....	88
df	Displays the free memory on the hard disk .....	91
du	Determines the space requirement of a directory .....	106
dumpe2fs	Displays internal information about an ext file system .....	107
e4defrag	Defragments files of an ext4 file system .....	108
exfatlabel	Changes the name of an exFAT file system .....	115
findmnt	Provides a list of all active file systems .....	125
fstrim	Reports all free data blocks to the SSD .....	134
kpartx	Creates or deletes device files for virtual data carriers .....	192
lsblk	Lists all block devices .....	206
mdadm	Manages RAID partitions .....	217
mkfifo	Creates a FIFO file (a named pipe) .....	219
mkfs	Sets up a file system .....	220
mknod	Creates device files .....	224
mkswap	Sets up a file or partition as a swap area .....	225
mount	Integrates a file system into the directory tree .....	227
ncdu	Serves as an interactive, convenient du variant .....	250
parted	Partitions a hard disk .....	276
partprobe	Informs the kernel about the changed partitioning .....	278
partx	Reads partitions or changes the partition table of the kernel .....	279
resize2fs	Changes the size of an ext file system .....	313
sfdisk	Partitions a hard disk with an MBR partition table .....	339
sgdisk	Partitions a hard disk with a GUID partition table .....	340
smartctl	Controls the SMART functions of the hard disk .....	344
snapper	Manages Btrfs snapshots (SUSE) .....	347
swapon	Deactivates a swap file or partition .....	360
swapoff	Activates a swap file or partition .....	360
sync	Executes all buffered write operations .....	360
tune2fs	Changes system parameters of an ext file system .....	380
umount	Removes a file system from the directory tree .....	383

xfs_admin	Changes parameters of an XFS file system.....	413
xfs_growfs	Enlarges an XFS file system .....	414
xfs_info	Displays the key data of an XFS file system .....	414
xfs_repair	Repairs a defective XFS file system .....	414
zramctl	Controls the ZRAM swap memory.....	420

## Logical Volume Manager (LVM)

lvcreate	Sets up a new logical volume (LV) .....	211
lvdisplay	Provides detailed information about an LV .....	212
lvextend	Enlarges an LV.....	212
lvm	Serves as an LVM basic command.....	213
lvreduce	Reduces the size of an LV .....	213
lvremove	Deletes an LV .....	213
lvrename	Assigns a new name to the LV.....	213
lvscan	Lists all LVs.....	213
pvccreate	Identifies a partition as a physical volume (PV).....	298
pvddisplay	Provides detailed information about a PV .....	299
pvremove	Removes the PV identification of an unused PV.....	299
pvscan	Lists all PVs .....	299
vgchange	Changes the attributes of a volume group (VG).....	391
vgcreate	Creates a new VG from one or more PVs.....	391
vgdisplay	Provides detailed information about a VG .....	392
vgextend	Increases a VG by one PV .....	392
vgmerge	Combines two VGs .....	392
vgreduce	Reduces a VG by one unused PV.....	392
vgrename	Assigns a new name to a VG .....	392
vgscan	Lists all VGs .....	393

## SELinux and AppArmor

aa-complain	Logs AppArmor rule violations without penalizing them .....	27
aa-disable	Deactivates an AppArmor rule profile .....	27
aa-enforce	Ensures compliance with AppArmor rules .....	27
aa-status	Determines the status of the AppArmor system .....	27
chcon	Changes the SELinux context of files .....	68
getenforce	Determines the SELinux mode (Enforcing or Permissive).....	140
restorecon	Restores the default SELinux context.....	315
sealert	Helps to analyze SELinux rule violations.....	330
sestatus	Determines the status of the SELinux system.....	332



setenforce	Changes the SELinux mode between Enforcing and Permissive	334
setsebool	Changes Boolean parameters of the SELinux rules .....	338

## Package Management

---

apk	Manages packages in Alpine Linux .....	36
apt	Helps with DEB package management (Debian, Ubuntu).....	38
add-apt-repository	Sets up a PPA package source (Ubuntu).....	31
alien	Converts packages between different formats .....	33
alternatives	Sets up links in /etc/alternatives (Fedora, Red Hat) .....	35
apt-cache	Provides information about installed/available packages .....	39
apt-get	Helps with DEB package management (Debian, Ubuntu).....	41
apt-file	Searches for files even in uninstalled packages (Debian, Ubuntu)	40
apt-key	Sets up a key for an APT package source .....	42
aptitude	Helps with DEB package management .....	43
cnf	Reveals which package contains a command (SUSE) .....	75
dnf	Serves as an alternative to yum (Fedora).....	97
dpkg	(De)installs or updates DEB packages.....	104
flatpak	Manages Flatpak packages .....	129
pacman	Manages packages from Arch Linux .....	270
pip	Manages Python packages.....	289
pkcon	Manages packages across distributions (PackageKit).....	290
ppa-purge	Deactivates a PPA package source (Ubuntu) .....	293
pro	Controls Ubuntu Pro functions.....	293
rpm	(De)installs or updates RPM packages .....	322
rpm2archive	Converts a package into a TAR archive.....	324
rpm2cpio	Converts a package into a CPIO archive.....	324
snap	Manages snap packages (Ubuntu) .....	347
tasksel	(De)installs DEB package groups .....	367
ubuntu-security-status	Indicates the support period of the installed packages .....	381
update-alternatives	Sets up links in /etc/alternatives .....	386
yay	Manages Arch Linux packages including AUR.....	418
yum	Helps with RPM package management (Fedora, Red Hat) .....	418
zypper	Helps with RPM package management (SUSE).....	421

## Network Administration

---

cadaver	Transfers files interactively via WebDAV.....	62
curl	Transfers files from/to HTTP, FTP, and SSH servers .....	84
dhclient	Performs a DHCP network configuration .....	93

dig	Performs DNS queries .....	94
etherwake	Activates a Wake-on-LAN device .....	113
ethtool	Changes the parameters of an Ethernet adapter .....	114
exportfs	Reports the NFS configuration to the NFS server .....	117
firewall-cmd	Reads or changes the firewall configuration (RHEL/Fedora).....	126
ftp	Interactively transfers files via FTP .....	134
host	Resolves IP addresses or network names .....	165
hostname	Returns or changes the name of the local computer .....	166
hostnamectl	Changes the host name permanently .....	167
ifconfig	Configures a network interface .....	172
ifdown	Deactivates a network interface .....	172
ifup	Activates a network interface .....	172
ip	Displays or changes network settings .....	177
ipcalc	Calculates network areas and masks .....	179
iptables	Configures a Netfilter firewall .....	180
ip[6]tables-restore	Reads multiple firewall rules.....	183
ip[6]tables-save	Outputs all firewall rules in text format .....	183
ip[6]tables-xml	Outputs all firewall rules as an XML document .....	183
iw	Controls WLAN interfaces.....	184
mtr	Combines ping and traceroute results.....	240
nft	Configures an nftables firewall .....	255
netplan	Controls other network backends (Ubuntu).....	251
netstat	Analyzes the network activity of the local computer .....	252
networkctl	Provides the network status (systemd) .....	253
newaliases	Reports changes in /etc/aliases to the mail server.....	254
nmap	Analyzes the network activity of a foreign computer.....	261
nmblookup	Determines the NetBIOS names of SMB servers.....	263
nmcli	Controls the NetworkManager.....	264
openssl	Generates and administers SSL key files .....	266
ping	Checks the network connection to a computer.....	289
pnuke	Terminates a program in parallel on several hosts .....	297
postconf	Reads or changes the postfix configuration.....	291
postqueue	Displays the postfix queue.....	292
pscp	Copies files simultaneously from/to host(s) .....	297
pssh	Executes commands on several hosts via SSH.....	297
rdiff-backup	Creates incremental backups .....	310
resolvectl	Determines or changes the DNS configuration.....	314
rfkill	(De)activates Bluetooth, WLAN, and mobile radio adapters.....	316
route	Changes or displays the IP routing table .....	321
rpcinfo	Provides information about RPC and NFS services.....	322
rsync	Synchronizes network directories .....	324
scp	Transfers files encrypted via SSH.....	327

sftp	Transfers files via SFTP.....	340
showmount	Lists NFS directories .....	342
smbclient	Transfers files from Windows network directories .....	345
smbpasswd	Sets the password of a Samba account.....	346
smbstatus	Provides status information of the local Samba server .....	346
ss	Analyzes the network activity of the local computer .....	351
ssh	Enables logins on other computers in the network.....	352
ssh-keygen	Generates and manages SSH keys.....	355
ssh-copy-id	Transmits a public key to the SSH server.....	354
swaks	Helps with testing and debugging mail servers .....	359
telnet	Communicates interactively with a network service .....	370
traceroute[6]	Provides the intermediate stations for a network address .....	377
ufw	Configures the firewall (Ubuntu).....	381
wakeonlan	Activates a Wake-on-LAN device (Debian, Ubuntu).....	403
wget	Downloads files or directories.....	404
whois	Performs DNS queries .....	408
wol	Activates a Wake-on-LAN device (Fedora, Red Hat).....	409
wpa_passphrase	Helps with the WLAN configuration.....	410

## Hacking and Security

---

arp	Manages the ARP cache .....	44
arp-scan	Sends ARP packets to all addresses in a network .....	45
chkrootkit	Searches for known rootkits .....	69
fail2ban-client	Administrates Fail2ban.....	117
hydra	Serves as an online password cracker.....	169
john	Serves as an offline password cracker .....	185
nc	Netcat, redirects network data to the standard input or output ..	248
ngrep	Filters network streams using <code>grep</code> (packet sniffing) .....	259
nmap	Serves as a network and port scanner .....	261
rkhunter	Searches for known rootkits .....	317
tcpdump	Filters network streams (packet sniffing) .....	368

## Printer, Database, and Server Administration

---

acme.sh	Manages certificates from Let's Encrypt .....	29
htpasswd	Saves Apache login data in a password file .....	167
lpadmin	Sets up new printers or deletes them again.....	201
lpinfo	Lists all print devices, printer drivers, and so on .....	201
lpoptions	Displays or changes the printer options .....	202

lpq	Displays the contents of a printer queue .....	202
lpr	Prints a file.....	202
lprm	Deletes a print job from the queue .....	203
lpstat	Provides information about printers, print jobs, and so on.....	203
mysql	Executes SQL commands on a MySQL server .....	241
mysqladmin	Helps with MySQL administration.....	244
mysqlbinlog	Extracts data from binary MySQL logging files .....	245
mysqldump	Performs a MySQL backup .....	246
pg_dump	Creates a backup of a PostgreSQL database.....	286
psql	Executes SQL commands on a PostgreSQL server.....	296
smbpasswd	Changes a Samba password .....	346
sqlite3	Executes SQL commands in SQLite databases.....	351

## Audio Functions and Hardware Management

acpi	Provides information about the battery status.....	31
alsactl	Saves or loads all parameters of the audio system .....	33
alsamixer	Interactively adjusts the audio channels.....	34
amixer	Controls the audio channels through options.....	35
boltctl	Controls devices on the Thunderbolt interface .....	57
free	Displays the free memory space (RAM/Swap).....	132
fwupdmg	Manages firmware updates.....	137
hwclock	Reads or sets the hardware clock.....	168
inxi	Provides an overview of the hardware .....	175
kbdrate	Sets the key repetition rate .....	190
localectl	Changes the language and keyboard settings.....	198
lscpu	Provides information about the CPU .....	208
lshw	Creates a hierarchical list of hardware components.....	208
lspci	Provides information about PCI components .....	210
ls SCSI	Provides information about connected SCSI devices.....	210
lsusb	Provides information about connected USB devices.....	211
nproc	Provides the number of CPU cores.....	266
pactl	Controls the PulseAudio server.....	271
paplay	Plays a raw file via PulseAudio .....	273
parecord	Performs a raw audio recording via PulseAudio .....	273
powertop	Helps to optimize the energy consumption of notebooks.....	292
pw-cat	Transfers audio files from or to the PipeWire system.....	299
pw-cli	Controls the PipeWire system.....	299
pw-mon	Lists all PipeWire objects.....	299
pw-top	Displays the active PipeWire audio components .....	299
speaker-test	Tests the audio system and the connected speakers .....	349

timedatectl	Sets the date, time, and time zone .....	374
vcgencmd	Reads or changes hardware parameters of Raspberry Pi .....	390

## Bluetooth

---

bluetoothctl	Configures Bluetooth devices .....	55
hciconfig	(De)activates local Bluetooth adapters .....	163
hcitool	Manages Bluetooth devices .....	164
l2ping	Sends echo requests to Bluetooth devices .....	193
rfkill	(De)activates Bluetooth, WLAN, and mobile radio adapters .....	316
sdptool	Determines detailed information about Bluetooth devices .....	329

## Kernel

---

canonical-livepatch	Administrates kernel live patches (Ubuntu) .....	62
depmod	Creates a file with all module dependencies .....	91
dmesg	Displays the messages of the kernel .....	97
dracut	Creates a new initrd file (Fedora, RHEL, SUSE) .....	106
insmod	Loads a kernel module (low level) .....	174
kexec	Activates a different kernel .....	190
lsmod	Lists all loaded kernel modules .....	209
modinfo	Provides information on a kernel module .....	226
modprobe	Loads a kernel module, including dependencies .....	226
uname	Displays the current kernel version .....	384
update-initramfs	Creates a new initrd file (Debian, Ubuntu) .....	387

## System Start and Stop, init System, Logging, and GRUB

---

efibootmgr	Reads or changes the table of EFI boot entries .....	109
grub-install	Installs GRUB into the boot sector .....	160
grub-mkconfig	Creates a new GRUB configuration file .....	161
journalctl	Reads news from the journal .....	188
logger	Logs a message .....	199
loginctl	Controls the systemd login manager .....	200
needs-restarting	Reveals whether the system or individual services need to be restarted .....	251
shutdown	Terminates Linux .....	342
systemctl	Manages systemd processes .....	361
systemd-analyze	Helps with systemd troubleshooting .....	363
update-grub	Updates the GRUB configuration (Debian, Ubuntu) .....	387

## Virtualization, Containers, Cloud

---

aws	Controls Amazon cloud services.....	49
docker	Manages containers.....	101
kvm	Runs a virtual machine .....	193
podman	Docker, manages containers .....	101
qemu	Runs a virtual machine .....	302
qemu-img	Creates or edits image files .....	306
rclone	Synchronizes a local directory with a cloud directory.....	307
virsh	Executes libvirt commands .....	393
virt-clone	Creates a copy of a virtual machine .....	399
virt-install	Sets up a new virtual machine .....	400
virt-viewer	Allows the operation of a virtual machine via VNC.....	402
wsl	Manages Linux installations on Windows .....	410

## Terminal and Text Console

---

echo	Outputs one line of text .....	109
loadkeys	Loads a keyboard table for text consoles.....	197
printf	Enables a formatted output as under C.....	293
reset	Performs a reset for the terminal.....	313
screen	Manages several sessions in one terminal .....	328
setterm	Changes various terminal settings .....	338

## Online Help

---

apropos	Searches for commands on a topic .....	37
help	Displays the description of a shell command .....	165
info	Starts the info system .....	173
man	Displays the description of a command.....	216
tldr	Displays examples of the use of a command.....	375
whatis	Displays a short description of a command .....	406

## Graphics System and Gnome

---

chvt	Changes the active text console or activates the graphics system	74
dconf	Modifies the dconf database (low level) .....	87
fc-list	Lists all scalable fonts .....	118
fgconsole	Returns the number of the active console .....	120
gnome-session-quit	Initiates a logout or shutdown .....	149

grim	Creates a screenshot on Wayland .....	159
gsettings	Reads or changes settings of the dconf database .....	161
nvidia-xconfig	Helps with the configuration of the NVIDIA graphics driver .....	266
xdg-open	Opens a file with a suitable program .....	413
scrot	Creates a screenshot under X .....	328
slurp	Enables the selection of a rectangle on Wayland .....	343
wl-copy	Inserts data into the Wayland clipboard .....	409
wl-paste	Reads data from the Wayland clipboard .....	409
wlr.randr	Changes the resolution of the graphics system (Wayland) .....	409
xdg-open	Opens a file using a suitable program .....	413
xdpyinfo	Provides information on the running X server .....	413
xhost	Allows or blocks hosts for the X login .....	415
xinput	Configures input devices for X .....	415
xkill	Terminates a program with a mouse click .....	415
xrandr	Changes the resolution of the graphics system (X) .....	416
xset	Changes user settings of the graphics system (X) .....	417
zenity	Displays simple graphical dialogs .....	419

## Miscellaneous

alias	Defines an abbreviation .....	33
basename	Determines the file name of a path .....	53
bc	Performs simple calculations (like a pocket calculator) .....	54
cksum	Calculates the CRC checksum for a file .....	74
date	Displays the date and time .....	86
dirname	Determines the directory of a path .....	96
expr	Performs calculations and sample comparisons .....	117
git	Controls the version management system Git .....	142
gpioget	Reads the state of the input/output pins of Raspberry Pi .....	154
gpioset	Changes the state of the input/output pins of Raspberry Pi .....	154
hash	Displays the hash table .....	163
ldd	Displays the required libraries for a program .....	195
lsb_release	Determines the name and version of the distribution .....	207
mail	Transfers an email to the local mail server .....	215
md5sum	Calculates a checksum for a file .....	217
printenv	Displays only the environment variables .....	293
pinctrl	Controls the input/output pins of Raspberry Pi .....	287
qalc	Serves as a calculator for the terminal .....	301
rpicas-still	Takes a photo (Raspberry Pi OS) .....	318
rpicas-vid	Records a video (Raspberry Pi OS) .....	320
seq	Returns a numerical sequence .....	332

set	Displays all variables known to the shell. ....	333
sha	Calculates a checksum for a file. ....	341
sleep	Waits for a specified time. ....	343
strace	Reveals which functions a program calls. ....	357
time	Measures the execution time of a command. ....	373
tty	Displays the device name of the terminal. ....	380
type	Specifies the type of a command. ....	380
unalias	Deletes an abbreviation. ....	384
uname	Returns the operating system name and the kernel version. ....	384
xargs	Redirects the standard input to a command. ....	412

## **“bash” Programming**

---

break	Ends a loop prematurely. ....	58
case	Initiates a case differentiation. ....	63
continue	Skips the loop body. ....	76
exit	Terminates the shell program. ....	116
for	Initiates a loop. ....	132
function	Defines a new function. ....	135
getopts	Processes options passed to a script. ....	141
if	Initiates a branch. ....	171
local	Defines local variables in a function. ....	198
source	Runs the specified shell file. ....	349
test	Analyzes a condition. ....	371
until	Initiates a loop (variant 1). ....	385
while	Initiates a loop (variant 2). ....	407

## **“bash” Variable Management**

---

alias	Defines an abbreviation. ....	33
declare	Defines an environment variable. ....	90
export	Defines a specified shell variable as an environment variable. ....	116
local	Defines local variables in a function. ....	198
read	Imports a variable. ....	311
readonly	Displays all read-only variables. ....	311
shift	Moves the parameter list. ....	342
unalias	Deletes an abbreviation. ....	384
unset	Deletes a variable. ....	385



## Additional “bash” Commands and Special Characters

<code>dirs</code>	Displays the list of saved directories.....	96
<code>disown</code>	Detaches a process from the shell.....	96
<code>eval</code>	Analyzes the specified command.....	114
<code>popd</code>	Changes to the last saved directory .....	291
<code>pushd</code>	Saves the directory and changes to another one .....	298
<code>trap</code>	Executes a command when a signal occurs.....	378
<code>ulimit</code>	Controls the resources used by the shell.....	382
<code>wait</code>	Waits for the end of a background process .....	403

## Configuration Files

<code>adduser.conf</code>	Settings for new accounts (Debian, Ubuntu).....	427
<code>aliases</code>	Email forwarding.....	429
<code>bashrc</code>	Default settings for bash.....	430
<code>config.txt</code>	Raspberry Pi hardware parameters (Raspberry Pi OS).....	430
<code>crontab</code>	Running processes periodically.....	431
<code>deluser.conf</code>	Settings for <code>deluser</code> and <code>delgroup</code> (Debian, Ubuntu).....	434
<code>dnf.conf</code>	Configuration of <code>dnf</code> package management (Fedora).....	435
<code>fstab</code>	Automatically mounting file systems/partitions .....	437
<code>group</code>	Group names and group assignments .....	438
<code>grub</code>	Default settings for GRUB 2.....	439
<code>grub.cfg</code>	Configuration for GRUB 2.....	441
<code>gshadow</code>	Hash codes of the group passwords .....	441
<code>host.conf</code>	Configuration of the resolver library.....	442
<code>hostname</code>	Hostname of the computer .....	443
<code>hosts</code>	Static list of hostnames and IP addresses .....	443
<code>interfaces</code>	Network configuration (Debian, Ubuntu).....	444
<code>journald.conf</code>	Configuration of the Journal logging service.....	447
<code>locale.conf</code>	Localization settings (systemd) .....	448
<code>login.defs</code>	Options for creating new users and groups .....	449
<code>mdadm.conf</code>	Software RAID configuration.....	451
<code>modules</code>	Loading kernel modules automatically (Debian, Ubuntu).....	452
<code>netplan.yaml</code>	Network settings (Ubuntu) .....	452
<code>networkd.network</code>	<code>networkd</code> configuration (systemd) .....	454
<code>nsswitch.conf</code>	Configuration of the name service switch functions .....	457
<code>os-release</code>	Name and version number of the distribution (systemd).....	458
<code>passwd</code>	List of all users and home directories .....	459
<code>profile</code>	Configuration of system-wide environment variables.....	460
<code>rc.local</code>	Script executed at the end of the boot process.....	460
<code>resolv.conf</code>	IP address of the name server .....	461

rsyslog.conf	Configuration of the syslog service .....	462
services	Assignment between network services and ports .....	464
shadow	Hash codes of the login passwords .....	465
sources.list	APT package sources (Debian, Ubuntu) .....	466
sudoers	Configuration for sudo. ....	467
sysctl.conf	Default settings for kernel parameters .....	469
systemd.service	Configuration of systemd services .....	470
systemd.timer	Configuration of periodic systemd jobs .....	473
vconsole.conf	Keyboard settings (systemd) .....	475
wpa_supplicant.conf	WLAN configuration (Raspberry Pi OS) .....	475

## Keyboard Shortcuts

---

bash	Shell .....	477
emacs	Editor .....	477
gnome-terminal	Terminal window on Gnome .....	480
grub	Bootloader .....	480
info	Command for displaying help texts .....	481
joe	Simple editor .....	482
konsole	Terminal window on KDE .....	482
less	Command for displaying text files .....	483
man	Command for displaying help texts .....	483
mutt	Email client for text mode .....	484
nano	Basic editor .....	484
screen	Terminal multiplexer .....	328
Text console	Keyboard shortcuts in text consoles .....	485
vi/vim	Editor .....	486

# Introduction

This book contains short descriptions of the most important Linux commands for managing the file system, for starting and terminating processes, for editing text files, for other administrative tasks, and for `bash` programming. The book also summarizes the syntax of basic configuration files and contains a keyboard shortcut reference for the `emacs`, `nano`, and `vi` editors, as well as some other interactive commands such as `less` or `info`.

The aim of this book is to provide a compact reference work for using Linux in the terminal. The basic principle of this book is: *less is more*. The book cannot and is not intended to replace the `man` and `info` pages of complex commands! Thus, you will still have to look up or research exotic options yourself.

With this book, however, I'm trying to relieve you of the work of searching through the often dozens of pages of original documentation for options for everyday use. Numerous examples also show the basic use of a command at a glance.

Sometimes, you're looking for a command for a specific task, but don't know its name or have just forgotten it. The thematically organized table of contents is intended for these cases.

Depending on which distribution you are using, some commands aren't available by default and must be installed separately. There are also distribution-specific commands that are only available in certain distributions, such as the package management commands `dpkg` and `apt` (Debian, Ubuntu), `rpm` and `dnf` or `yum` (Fedora, Red Hat), and `zypper` (SUSE). This is pointed out in the respective command description.

## What Is a Command?

Linux doesn't distinguish between commands (as described in this book) and programs such as Firefox, LibreOffice, or GIMP. Here, "command" refers to programs without a graphical user interface, which are usually executed in a terminal window.

In this book, I also describe some commands that aren't real programs at all, but only commands of the currently active shell. I assume that you're using the `bash` (Bourne Again Shell) or the largely compatible `zsh`. These shells are used in most Linux distributions for the interactive execution of commands. An example of a shell command is `cd` to change the current directory.

## Options

Most of the commands described in this book are controlled by options. The options are specified *before* all other parameters. There are two ways of notation for many commands: `-x` for short options (one letter) and `--xyz` for long options (multiple letters).

Thus, the following two `ls` commands are equivalent and display all files and directories in the `/usr` directory:

```
user$ ls -l -A /usr
user$ ls --format=long --almost-all /usr
```

For some commands, multiple options can be specified as a group (i.e., `-ab` instead of `-a -b`). Some commands also work with options that are specified after the actual parameter(s). However, this shouldn't lead you to the conclusion that this applies to all commands!

```
user$ ls -lA /usr
user$ ls /usr -lA
```

For a few commands, the order of the parameters has an influence on how the command is executed. If options are specified that are logically mutually exclusive, the last option specified applies.

## man, info, and help

To avoid making this book unnecessarily long, I'll only describe the most important options. A complete overview of all options for the majority of commands is provided by `command name --help`. More detailed information is usually contained in the manual pages, which you can access via `man name` or `man 1 name`. For some commands, the `man` pages only contain a reference to the `info` texts, which are displayed accordingly via `info name`.

For commands that are directly integrated into the `bash` (e.g., `cd`), `man name` leads to the `man` page of the `bash`. The command is actually described there, but searching through the very long documentation is tedious. In this context, it's more helpful to use `help name`.

The help texts from `man` and `info` sometimes overshoot the mark. If you're just looking for a few examples of how to use a command, you should use either this book or the `tlldr` command. Incidentally, the command name stands for *Too long, didn't read*. :-)

```
ncdu [options] [directory]
```

`ncdu` is an interactive variant of the `du` command. Usually, it's simply called without options and parameters. It then displays the subdirectories in the current directory that take up the most space.

You can now navigate through the subdirectories using the cursor keys and `[Enter]`. Various other functions are accessible via keyboard shortcuts. An overview of the most important shortcuts can be obtained by entering `[?]`. But be careful—`[D]` deletes the current file or directory after a query.

- ▶ `-r`  
Runs the command in read-only mode. Accidental deletion of files or directories is therefore impossible.
- ▶ `-x`  
Remains in the current file system, that is, doesn't take any mount directories into account.

### Example

The space required by the directories in the home directory is also symbolized by bars:

```
user$ ncdu
--- /home/kofler ---
   1.5 GiB [#####] /Nextcloud
   1.0 GiB [#####] /Downloads
  528.5 MiB [###   ] /Images
  209.4 MiB [#     ] /Documents
...
```

```
needs-restarting [options]
```

The `needs-restarting` command available on Fedora, Red Hat, and related distributions specifies whether the computer or individual programs need to be restarted due to an update of the kernel, a basic library, or a firmware file.

The command isn't available for distributions based on Debian or Ubuntu. There, the `/var/run/reboot-required` file indicates that a reboot is required.

- ▶ `-r`  
Indicates whether and why a computer restart is required.
- ▶ `-u`  
Lists processes of the current user that need to be restarted. This option applies by default.

## Example

The following commands show the status of a computer that should be restarted:

```
root# needs-restarting -r
Core libraries or services have been updated since boot-up:
  * glibc
Reboot is required to fully utilize these updates.
More information: https://access.redhat.com/solutions/27943
root# needs-restarting | sort -n
782 : /usr/lib/polkit-1/polkitd --no-debug
783 : /usr/libexec/power-profiles-daemon
784 : /usr/bin/qemu-ga --method=virtio-serial ...
787 : /usr/libexec/accounts-daemon
...
```

### netplan [command]

Netplan (<https://netplan.io>) is a framework developed by Ubuntu that configures and integrates other network backends such as the NetworkManager and networkd (a systemd component, see `networkctl`). It's used in Ubuntu and evaluates the configuration files located in `/etc/netplan` (see `/etc/netplan/netplan.yaml`).

- ▶ `apply`  
Runs `netplan generate` and then activates the changed configuration.
- ▶ `generate`  
Reads the Netplan configuration from the YAML files in `/etc/netplan`, `/lib/netplan` and `/var/netplan`, and then generates the corresponding configuration files for the network backends. The new files end up in `/run/NetworkManager` or in `/run/systemd/network`.  
  
The new configuration isn't activated! The changed settings only take effect when the respective backend is requested to evaluate the files (or at the next restart).
- ▶ `ip leases interface`  
Outputs the current DHCP lease data for the relevant interface. This only works if the `networkd` backend and a DHCP configuration are in use. Otherwise, the command will return misleading error messages.

### netstat [options]

`netstat` provides information about the network activity on the local computer. If the command is called without options, it returns a list of all open internet connections and sockets.

- ▶ **-a**  
Also takes into account inactive sockets, that is, services that are waiting for a connection on a network port (LISTEN state).
- ▶ **-e**  
Also specifies the user for internet connections.
- ▶ **-n**  
Provides numerical addresses and port numbers instead of host and network service names.
- ▶ **-p**  
Also specifies the process number (PID) and the process name of the program responsible for the connection. `netstat` requires root permissions so that it can provide information about nonnative processes.
- ▶ **-t / -u / -w / -X**  
Limits the output to connections that use the TCP, UDP, Raw, or Unix protocol.

### Example

The following command lists all active connections (established) or monitored ports (LISTEN). The output is heavily abridged due to space limitations.

```
root# netstat -atupe
Active Internet connections (servers and established)
Proto Local Address           Foreign Addr    State   User      PID/Prog name
tcp    *:nfs                *:              LISTEN  root      -
tcp    *:ldap               *:              LISTEN  root      5842/slapd
tcp    localhost:mysql      *:              LISTEN  mysql     5785/mysqld
tcp6   [::]:ssh             [::]:*          LISTEN  root      5559/sshd
udp    *:nfs                *:              root    -
```

### **networkctl** [command]

`networkctl` is one of the administration commands from the `systemd` family. If `networkd` is used as the network backend, `networkctl` helps to analyze the network status.

Although `networkd` is installed by default on many distributions, it's only active on a few of them. One of the exceptions is Ubuntu Server, where `networkd` serves as a backend for Ubuntu's own Netplan system (see `netplan`). `networkd` evaluates configuration files from the `/etc/systemd/network`, `/lib/systemd/network`, and `/run/systemd/network` directories (see `/etc/systemd/network/networkd.network`).

- ▶ **list**  
Lists all network interfaces. The `SETUP` column of the result shows whether the interface is controlled by `networkd` (configured) or not (unmanaged).

- ▶ `lldp`  
Shows other devices in the network that were discovered via the *Link Layer Discovery Protocol* (LLDP).
- ▶ `status [interface]`  
Provides detailed information about the status of a network interface. If no interface is specified, the command attempts to summarize the entire network status.

## Examples

The following outputs of `networkctl` were created on an Ubuntu server in a virtual machine:

```
root# networkctl list
IDX  LINK      TYPE      OPERATIONAL  SETUP
  1  lo        loopback  carrier      unmanaged
  2  ens3      ether     routable     configured
root# networkctl status ens3
    Link File: /lib/systemd/network/99-default.link
    Network File: /run/systemd/network/10-netplan-ens3.network
        Type: ether
        State: routable (configured)
        ...
    HW Address: 52:54:00:0a:8d:fc
    Address: 138.201.20.182
           fe80::5054:ff:fe0a:8dfc
    Gateway: 138.201.20.176 (Fujitsu Technology Solutions GmbH)
    DNS: 213.133.100.100
        213.133.98.98
    Search Domains: ubuntu-buch.info
```

The second listing shows the status summary of another server installation with IPv6 configuration:

```
root# networkctl status
    State: routable
    Online state: online
    Address: 168.119.33.110 on eth0
           172.17.0.1 on docker0
           2a01:4f8:242:1f88::4 on eth0
           fe80::5054:ff:fe4a:7321 on eth0
    Gateway: 168.119.33.119 on eth0
           2a01:4f8:242:1f88::2 on eth0
    DNS: 213.133.100.100
```



```
213.133.99.99
2a01:4f8:0:1::add:1010
2a01:4f8:0:1::add:9999
```

### **newaliases**

The `/etc/aliases` file contains an alias list for the email server, which ensures, for example, that all emails addressed to `postmaster` are forwarded to `root`. For changes to this file to be taken into account by the mail server, you must run `newaliases`.

### **newgrp** [group name]

The `newgrp` command determines the currently active group of a user who belongs to several groups. The active group determines which group newly created files belong to. The groups available for selection can be determined via `groups`. If no group name is specified, the primary group will be used. This group is also automatically considered an active group after a login.

### **Example**

In the following example, the `newgroup` command makes `docuteam` the active group of the user `kofler`. As a result, the newly created `newfile` file is assigned to the `docuteam` group and can be edited by other members of the documentation team.

```
user$ groups
kofler docuteam wheel
user$ newgroups docuteam
user$ touch newfile
```

### **newusers** file

`newusers` reads a text file and creates a new user for each line. The text file basically has the same format as `/etc/passwd`. However, the passwords must be entered unencrypted in the second column. Most of the other parameters (UID, GID, etc.) are optional. `newusers` creates a new account for each specified user, whereby the associated groups are also created if required. For missing parameters, `newusers` selects suitable default values, taking into account the settings in `/etc/login.defs`.

Note that `newusers` creates new home directories if their location is specified in the sixth column. However, the command doesn't take care of copying the contents of `/etc/skel` there.

## Example

The following lines show the minimalist text file `users.txt`, which corresponds to the requirements of `newusers`. As `users.txt` contains passwords in plain text, you must make sure that no one other than `root` can read the file or that you delete the file again after running `newusers`.

```
hawks:secret1:::Howard Hawks:/home/hawks:/bin/bash
moore:secret2:::Grace Moore:/home/moore:/bin/bash
smith:secret3:::Peter Smith:/home/smith:/bin/bash
```

`newusers` now creates the three new users `hawks`, `moore`, and `smith`, as well as primary groups of the same name. `newusers` itself decides on suitable UIDs and GIDs.

```
root# newusers users.txt
```

```
nft command [options]
```

Most current Linux distributions now use the new *nftables* firewall system, which replaces its predecessor, *Netfilter*. However, the old `iptables` command is often still used to configure the firewall, which continues to work thanks to a compatibility layer.

However, new or `nftables`-specific functions can only be controlled using the `nft` command. The following description provides a rough overview of the most important sub-commands and options of this command. You can find a lot more details about the functionality of `nftables` and the syntax of `nft` in `man nft` and the following wiki:

<https://wiki.nftables.org>

Before you call `nft`, you should make sure that a firewall isn't already running on your distribution. On Fedora, RHEL, and SUSE, for example, *Firewalld* is active (see `firewall-cmd`). Custom-defined firewall rules then often result in conflicts with the firewall of your distribution.

## nft Terminology

A separate terminology applies to `nft`: Firewall *rules* are always part of a *chain*. *Tables*, in turn, consist of multiple chains. Unlike `iptables`, there are neither predefined tables nor predefined chains. Thus, you can compose as many custom tables as you like from chains also defined by yourself.

Tables, chains, and rules apply to different types of network packets, which are assigned to the following *families*:

- `ip`  
Rules for IPv4 only.

- ▶ `ip6`  
Rules for IPv6 only.
- ▶ `inet`  
Rules that apply equally to IPv4 and IPv6.
- ▶ `arp`  
Level 2 rules, which are analyzed before Level 3 rules.
- ▶ `bridge`  
Switching rules.
- ▶ `netdev`  
Low-level rules that are analyzed before all other rules and enable, for example, a particularly efficient defense against DDOS attacks.

### nft Options

- ▶ `-a`  
Integrates *handles* into the output of `nft list`. These are numbers that uniquely identify each object within a group (e.g., a rule within a chain). Handles make it possible to delete or replace individual objects or to specify the position where new objects are to be inserted.
- ▶ `-f fname`  
Reads the commands to be executed from the specified file. Two syntax variants are permitted: the file can either contain ordinary `nft` commands line by line or a hierarchically structured sequence of tables, chains, and rules. The structure of the file for the second variant corresponds to the output of `nft list ruleset`.
- ▶ `-S`  
Uses the names of network services specified in `/etc/services` instead of port numbers (e.g., `ssh` instead of `22`).
- ▶ `-v`  
Shows the version number.

### nft Commands

- ▶ `add chain family tname cname`  
Creates a new chain named *cname* in the specified table.
- ▶ `add rule family tname cname matches statements`  
Adds a new rule to the specified *cname* chain. Here, *matches* specifies the packets to which the rule applies. *statements* describes what should happen to the respective packets.

Instead of `add`, the `insert` and `replace` keywords are also permitted to insert a rule at a specific position in the rule list or to replace an existing rule with a new one.

- ▶ `add table family tname`  
Creates a new table for the specified family.
- ▶ `delete/flush chain family tname cname`  
Removes the specified chain from the table. `flush` deletes all rules of the chain, but not the chain itself.
- ▶ `delete rule family tname cname handle came`  
Deletes the rule specified by *handle* from the *cname* chain.
- ▶ `delete/flush table family tname`  
Deletes the specified table. With `flush`, all chains and rules in the table are deleted, but the now empty table is retained.
- ▶ `flush ruleset`  
Deletes all tables, including all chains and rules contained therein. This corresponds to a complete reset of the firewall. The firewall now accepts every packet.
- ▶ `list tables`  
`list table family name`  
`list chain family tname cname`  
Lists tables, chains, and rules.
- ▶ `list ruleset`  
Lists all tables, chains, and rules. The output structured by curly brackets fulfills the syntax rules for `nft -f` and can therefore be used as the basis for a new rule file. With the additional `-j` option, the output is in JSON format.
- ▶ `monitor [filter criteria]`  
Displays nftables events (e.g., as debugging help).

## Examples

The first two commands show which firewall tables are defined on the test computer (Fedora 32) and which chains and rules apply to the `firewalld` table within the `ip` family. (nftables allows the use of matching table names for different families. For this reason, the family must also be specified in the second `list` command.)

```
root# nft list tables
table bridge filter
table bridge nat
table inet firewalld
table ip firewalld
table ip6 firewalld
root# nft list table ip firewalld
table ip firewalld {
```

```
chain nat_PREROUTING {
    type nat hook prerouting priority dstnat + 10; policy accept;
    jump nat_PREROUTING_ZONES
}
chain nat_PREROUTING_ZONES {
    iifname "enp1s0" goto nat_PRE_FedoraWorkstation
    goto nat_PRE_FedoraWorkstation
}
...
```

systemctl now deactivates the default Fedora firewall daemon:

```
root# systemctl disable --now firewalld
```

Now, you can create your own firewall using `nft`. It's not usual to define the rules by repeatedly calling `nft`. It's much more elegant to formulate the rules in a text file and run this file like a script. To do this, you must enter `nft -f` as a shell interpreter in the line.

```
#!/sbin/nft -f
# delete existing firewall
flush ruleset
# new table for IPv4 and IPv6
table inet myfilter {
    chain input {
        type filter hook input priority 0; policy drop;
        # block faulty packets
        ct state invalid drop
        # always accept packets from self-generated connections
        ct state {established, related} accept
        # accept internal network traffic
        iif lo accept
        # Block packets to loopback but from external addresses
        iif != lo ip daddr 127.0.0.1/8 drop
        iif != lo ip6 daddr ::1/128 drop
        # accept ICMP packets
        ip protocol icmp accept
        ip6 nexthdr icmpv6 accept
        # accept SSH connection from outside
        tcp dport 22 accept
        # for server configuration: more rules for
        # HTTP + HTTPS (Port 80 + 443), Samba etc.
        # ...
    }
}
```

```

    }
    # no forwarding
    chain forward {
        type filter hook forward priority 0; policy drop;
    }
}

```

Now you can run the script:

```

root# chmod +x myfirewall
root# ./myfirewall

```

`nft -f` first performs a syntax check. If it detects errors, it aborts the process with an error message. In this case, the previous firewall is retained.

The preceding script has been slightly modified from the Gentoo wiki. You can find more examples there and in the ArchLinux Wiki:

<https://wiki.gentoo.org/wiki/Nftables/Examples>

<https://wiki.archlinux.org/title/Nftables>

```
ngrep [options] [grep-search-expression] [pcap-filter-expression]
```

`ngrep` is referred to as a *packet sniffer*. The command reads the network traffic of a port and filters it. `ngrep` provides similar functions to the `tcpdump` command and, like this, uses the `pcap` library to read the network packets. Unlike `tcpdump`, however, `ngrep` also takes the contents of the packages into account. Naturally, this only works for nonencrypted protocols, such as HTTP or FTP.

The search expression must be formulated as a regular pattern as in `grep`. The syntax summarized in `tcpdump` applies to the `pcap` filter expression.

- ▶ `-d interface|any`  
Determines the network interface.
- ▶ `-i`  
Ignores uppercase and lowercase in the `grep` search expression.
- ▶ `-v`  
Inverts the search. `ngrep` therefore only returns the packets in which the `grep` search pattern was *not* recognized.
- ▶ `-W`  
Interprets the `grep` search expression as a word.
- ▶ `-Wbyline`  
Takes line breaks into account during output, resulting in more legible output.

## Example

The following example listens on all interfaces for HTTP packets containing the keywords `user`, `pass`, and so on:

```
root# ngrep -d any -i 'user|pass|pwd|mail|login' port 80
interface: any
filter: (ip or ip6) and ( port 80 )
match: user|pass|pwd|mail|login
T 10.0.0.87:58480 -> 91.229.57.14:80 [AP] POST /index.php
  HTTP/1.1..Host: ... user=name&pass=geheim&login=Login
...
```

**nice** [options] program

`nice` starts the specified program with a reduced or increased priority. The command can be used to start non-time-critical programs with low priority so as not to affect the rest of the system too much.

► `-n +/-n`

Specifies the `nice` value. By default (i.e., without `nice`), programs are started with the `nice` value of 0. A value of `-20` means highest priority, and a value of `+19` means lowest priority. Values less than 0 may only be specified by `root`; that is, most users can only start programs with reduced priority with `nice`. If this option is omitted, `nice` starts the program with the `nice` value of `+10`.

Note that `nice` only controls the CPU load. If you want to reduce the I/O load of a command, it's better to use `ionice`.

## Example

The following command starts the `mybackup.sh` script with lower priority:

```
user$ nice -n 10 mybackup.sh
```

**nl** [options] file

`nl` numbers all nonempty lines of the specified text file and writes the result to the standard output. By setting the numerous options, you can have page-by-page numbering, the numbering of headers and footers, and more.

**nmap** [options] hostname/ip-address/ip-address-range

The `nmap` command (*network mapper*) from the package of the same name performs a port scan and attempts to determine which network services are active on the specified

computer or in the specified network. `nmap` should only be used to analyze your own computers or after consultation with the respective administrator. A port scan of other computers can be interpreted as an attempted attack!

- ▶ `-A`  
Performs an extensive (“aggressive”) scan, corresponding to `-sV -O -sC --traceroute`.
- ▶ `-F`  
Considers only the 100 most important ports from `/usr/share/nmap/nmap-services` (quick scan).
- ▶ `-iL file`  
Scans the IP addresses specified in the file.
- ▶ `-oN file / -oG file / -oX file.xml`  
Writes the results to an ordinary text file, to a text file that can be easily processed with `grep`, or to an XML file. Without the option, `nmap` uses the standard output and the ordinary text format.
- ▶ `-O`  
Attempts to recognize the operating system; this option must be combined with a scan option, such as `-sS`, `-sT`, or `-sF`.
- ▶ `-p1-10,22,80`  
Takes only the specified ports into account.
- ▶ `-Pn`  
Doesn’t perform a ping test. This means that `nmap` considers all hosts to be online and performs a scan in any case (slowly!).
- ▶ `-sL`  
Lists all ports and specifies host names assigned in the past. This is particularly fast, but provides outdated data even from devices that are currently no longer online.
- ▶ `-sP`  
Performs only a ping test (fast).
- ▶ `-sS`  
Performs a TCP SYN scan (applies by default).
- ▶ `-sU`  
Takes UDP into account as well. This option can be used together with another `-s` option.
- ▶ `-sV`  
Tries to find out which service is provided on open ports (*service version detection*) or which program is responsible for the service and in which version.
- ▶ `-T0` to `-T5`  
Selects a timing schema:
  - `-T5` is the fastest.



- -T3 is the default.
- -T0 and -T1 are extremely slow, but minimize the risk of the scan being detected.

► -v

Outputs detailed information (*verbose*).

You must choose an -s option when calling. Only -sU may be combined with other -s options. In general, the right choice of options is a trade-off between thoroughness and speed.

In many cases, `nmap -v -A name` is sufficient to get an initial overview of the network services of the specified computer. Advanced `nmap` users can find more details on the `man` page and on the following websites:

<https://insecure.org/nmap>

<https://nmap.org/book>

Graphical user interfaces also exist for `nmap`, such as `nmap-frontend` or `zenmap`.

## Examples

The following command performs a quick network scan on the local network (256 IP addresses). Thanks to the focus on the most important 100 ports, the job is done within about two seconds. The `nmap` outputs have been shortened for reasons of space and only show the results of two of the devices found:

```
root# nmap -F -T4 10.0.0.0/24
Nmap scan report for imac (10.0.0.2)
  Host is up (0.00019s latency).
  PORT      STATE SERVICE
  22/tcp    open  ssh
  88/tcp    open  kerberos-sec
  445/tcp   open  microsoft-ds
  548/tcp   open  afp
  MAC Address: AC:87:A3:1E:4A:87 (Apple)
Nmap scan report for raspberrypi (10.0.0.22)
  Host is up (0.00038s latency).
  Not shown: 99 closed ports
  PORT      STATE SERVICE
  22/tcp    open  ssh
  MAC Address: B8:27:EB:11:44:2E (Raspberry Pi Foundation)
...
Nmap done: 256 IP addresses (6 hosts up) scanned in 2.42 seconds
```

The second example shows the `nmap` result for a NAS device that is located in a different local network (again, it has been heavily shortened):

```

root# nmap -v -A 192.168.178.28
Nmap scan report for DiskStation.fritz.box (192.168.178.28)
Host is up (0.0023s latency).
...
Discovered open port 445/tcp on 192.168.178.28
Discovered open port 139/tcp on 192.168.178.28
Discovered open port 80/tcp on 192.168.178.28
Discovered open port 443/tcp on 192.168.178.28
Discovered open port 22/tcp on 192.168.178.28
Discovered open port 5001/tcp on 192.168.178.28
Discovered open port 5000/tcp on 192.168.178.28
Discovered open port 548/tcp on 192.168.178.28

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.4 (protocol 2.0)
    ssh-hostkey:
        2048 5a:e7:3a:66:f4:99:9f:0a:0a:... (RSA)
        256 06:1a:bf:9f:e9:d0:64:3a:92:49:... (ECDSA)
        256 ad:b7:7d:ab:ae:70:0a:c9:a6:0c:... (ED25519)
    ...
80/tcp    open  http      nginx
139/tcp   open  netbios-ssn Samba smbd 4.6.2
445/tcp   open  netbios-ssn Samba smbd 4.6.2
...
Nmap done: 1 IP address (1 host up) scanned in 64.19 seconds

```

```
nmblookup [options] workgroupname
```

**nmblookup** from the **samba-common-bin** (Debian/Ubuntu) or **samba-client** (Fedora/RHEL) package determines which devices or servers in the local network provide SMB shares. Previously, **smbtree** could be used for this, but this command requires the browsing functions of SMB1 and is obsolete in modern networks.

- **-S**  
Performs a status query for each server found and then displays the detailed results.
- **-T**  
Provides an ordered list of server names and IP addresses. In my experience, the results of **nmblookup** are more reliable if you run **nmblookup -S** first or repeat the command again after a few seconds. The first result is often incomplete.

## Example

There are two SMB servers, a NAS device, and an appropriately configured Raspberry Pi in the local network:

```
user$ nmblookup -T WORKGROUP
DiskStation.fritz.box, 192.168.178.43 WORKGROUP<00>
pi5.fritz.box,          192.168.178.123 WORKGROUP<00>
```

```
nmcli [options] con|dev|nm command
```

Network Manager is usually controlled via a menu in the Gnome or KDE panel. You can also use the `nmcli` command to control network connections via the command line or a script.

The commands available for selection depend on which object they refer to. Options `-t`, `-p`, and so on can be used to control the form of the `nmcli` output—depending on whether the output is to be formatted properly or processed further by a script (for details, see `man nmcli`).

- ▶ `con down name` or `con down uuid n`  
Deactivates the specified connection.
- ▶ `con show`  
Lists all configured connections, specifying the name and UUID of each connection.
- ▶ `con up name` or `con up uuid n`  
Activates the specified connection.
- ▶ `dev disconnect name`  
Terminates the connection for the specified interface.
- ▶ `dev`  
Lists all network interfaces known to Network Manager and specifies their properties.
- ▶ `dev show`  
Displays detailed information on all network interfaces.
- ▶ `dev wifi hotspot`  
Configures the Wi-Fi adapter as a router.
- ▶ `dev wifi list`  
Provides a list of all Wi-Fi networks within range, including the signal strength. The command requires the `wpa_supplicant` service to be running.
- ▶ `general [status]`  
Displays the status of Network Manager.

- ▶ `networking [on|off]`  
Indicates whether there's a network connection or not. All network connections can be interrupted or restored by selecting `off` or `on`.
- ▶ `radio [wifi|wwan|wimax|all] [on|off]`  
Displays the status of the wireless networks or changes their status. `wifi` refers to Wi-Fi connections, `wwan` to mobile connections, and `wimax` to WiMAX technology.

## Examples

The first command lists all connections known to Network Manager. The following two commands deactivate and reactivate the connection named *Wired connection 1*:

```
root# nmcli con show
NAME                                UUID                                TYP                                DEVICE
Wired connection 1                 effd094d-... 802-3-ethernet  enp0s3
virbr0                             207d9f4f-... bridge          virbr0
root# nmcli con down   id 'Wired connection 1'
root# nmcli con up     id 'Wired connection 1'
```

The second example filters out the IP addresses of the name server from the detailed interface information. This is particularly useful on Ubuntu, where these addresses aren't listed in `/etc/resolv.conf`. Rather, `resolv.conf` references a local name server in this distribution, that is, the address 127.0.0.1.

```
root# nmcli dev show | grep DNS
IP4.DNS[1]: 10.0.0.138
IP4.DNS[2]: 4.4.8.8
```

The third command sets up a hotspot on a Raspberry Pi connected to the LAN via ethernet cable:

```
root# device wifi hotspot ssid 'wlan-name' password 'wlan-pw'
```

## nohup command

If you start a command as a background process in a shell window and then close the window, or if you start the command in a text console and then log out, the background process is automatically terminated. As a rule, this is reasonable behavior.

However, sometimes you want to start a process that continues to run after you log out—and that's exactly what `nohup` is there for. The command must be specified with its full path. It can't write text outputs to the standard output. If necessary, such outputs are redirected to the `nohup.out` file in the local directory.

## Example

In the following example, an administrator logs in to a server via `ssh`, starts a backup script in the background, and then logs out again. The backup script continues to run.

```
someone@localhost$ ssh user@remotehost
user@remotehost$ nohup backup-script &
user@remotehost$ exit
```

## nproc

`nproc` indicates how many CPU cores are available. For CPUs with hyperthreading, the virtual cores are also included in the calculation. The `lscpu` command and the contents of the `/proc/cpuinfo` file provide more detailed information about the CPU.

## nvidia-xconfig [options]

`nvidia-xconfig` helps to configure the proprietary NVIDIA graphics driver. The command is only available if this driver is installed.

If the command is executed without parameters, it creates or changes the `/etc/X11/xorg.conf` file so that the NVIDIA graphics driver will be used in the future. If the graphics system is working after restarting the computer, the `nvidia-settings` graphical user interface can be used for further configuration.

- ▶ `-c file`  
Uses the specified file instead of `/etc/X11/xorg.conf`.
- ▶ `--mode=WxH`  
Adds a graphics mode for `WxH` pixels to `xorg.conf`.
- ▶ `--query-gpu-infos`  
Displays details of all available graphics cards and connected monitors.
- ▶ `-t`  
Reads `xorg.conf` and displays the settings it contains in a clear tree view.

## openssl command

`openssl` from the package of the same name helps with the creation and further administration of certificates as well as private and public keys. The command uses the OpenSSL library. New certificates or keys are written to the standard output if you don't specify a target file via `-out`.

- ▶ `dhparam [options] [n]`  
Generates or manipulates parameter files for the Diffie-Hellman method. The `n`

parameter specifies the key size in bits. In the simplest case, `openssl dhparam -out keyfile 512` generates a Diffie-Hellman key with a length of 512 bits.

► `enc [options]`

Reads data from the standard input, encodes or decodes it symmetrically (option `-d`), and writes the result to the standard output. You need to specify the algorithm to be used and the key with options, for example, in the format `-aes-256-cbc -pass file:mykeyfile`. Note that `openssl enc` is only suitable for relatively small amounts of data. To encrypt large files, it's better to use the `gpg` command.

► `genpkey [options]`

Generates a private key. To write a 2,048-bit RSA key to a file, the following options are required:

```
openssl genpkey -algorithm RSA -pkeyopt rsa_keygen_bits:2048 -out server.key
```

If the key itself is to be encrypted, you must specify the name of the encryption algorithm with an additional option, such as `-aes256`. During the execution of the command, `openssl` asks for an encryption password twice.

► `list-standard-commands | list-message-digest-commands |  
list-cipher-commands | list-cipher-algorithms |  
list-message-digest-algorithms | list-public-key-algorithms`

Lists the commands, algorithms, and so on that are supported by `openssl`.

► `rand [options] n`

Writes *n* bytes of pseudo-random data to the standard output. Note that the output is binary data, not ASCII text! The `-base64` option causes `openssl` to perform Base64 encoding in ASCII format. This increases the length of the resulting character string by around a third.

► `req [options]`

Generates a request to sign a certificate (i.e., a *certificate signing request* [CSR] file). With `-new`, you can specify that a new certificate must be set up. `-key` specifies the key file to be used. Without this option, `openssl` generates a new RSA key for this purpose, which you can save in a separate file using `-keyout`. `openssl req -new -sha256 -key server.key -out server.csr` `openssl` then interactively asks for the key data of the certificate, that is, the country code, location, your name, host name (common name), and more. With the additional `-x509 -days n` options, `openssl` generates a self-signed certificate with a validity of *n* days instead of a certificate request.

► `rsa [options]`

Processes RSA keys and converts them between different forms and formats. `-in file` and `-out file` specify the file from which the original key is read and where the new key is written. If you don't specify any further options, `openssl` removes the encryption (*pass phrase*) of the key. The form of the key can be specified with `-inform` and `-outform` (by default, it's PEM, alternatively DER or NET). With `-in file -text`, you

can determine the properties of an existing key, such as the key length (the number of bits).

► `sclient [options]`

Acts as an SSL/TLS client to test connections to an encrypted server. You can find an application example at the following address:

*<https://www.misterpki.com/openssl-s-client/>*

► `speed`

Performs multiple benchmark tests, each of which lasts three seconds. The result shows how many bytes were processed for the respective algorithm (the more there are, the faster it runs or the better optimized the hardware on which the test is performed).

► `x509 [options]`

Displays certificate data, signs certificates, and converts certificates between different forms. Again, `-in` and `-out` specify the file names of the source and target certificates. With `-req`, `openssl` expects a certificate request as input, not the certificate itself. `-CA` specifies which CA certificate is to be used for the signature. `-CAkey` specifies the corresponding private key.

There are separate `man` pages available for most `openssl` commands. For example, `man req` describes the countless options of `openssl req`, `man rsa` describes the options of `openssl rsa`, and so on.

## Examples

If you want to have a certificate signed for your web server by an official certification authority (e.g., Thawte), you should first generate a key (if you don't already have one) and then a certificate signing request. The key isn't encrypted here because otherwise the web server would have to ask for the encryption password every time it's started.

```
root# genpkey -algorithm RSA -pkeyopt rsa_keygen_bits:2048 -out server.key
root# chmod 400 server.key
root# openssl req -new -key server.key -out server.csr
...
```

Common Name (eg server FQDN or YOUR name) []: **company-abc.com**

You can take a look at the CSR file to check this:

```
root# openssl req -in server.csr -noout -text
```

Then, you need to send the CSR file to the certification authority. You will then receive the signed certificate from there for a fee. The usual file identifier is `.pem` or `.crt`. You must enter this file, your own key, and a CA certificate from the certification authority in the Apache configuration.

```
# Apache configuration file
SSLCertificateFile      /etc/apache2/server.pem
SSLCertificateKeyFile   /etc/apache2/server.key
SSLCertificateChainFile /etc/apache2/sub.class1.server.ca.pem
SSLCACertificateFile    /etc/apache2/ca.pem
```

If you don't want to pay for an official signature, you can sign your certificate yourself:

```
root# openssl x509 -req -days 1900 -in server.csr \
      -signkey server.key -sha256 -out server.pem
Signature ok
subject=/C=DE/L=Boston/O=John Doe/CN=www.company-abc.com/
      emailAddress=webmaster@company-abc.com
Getting Private key
```

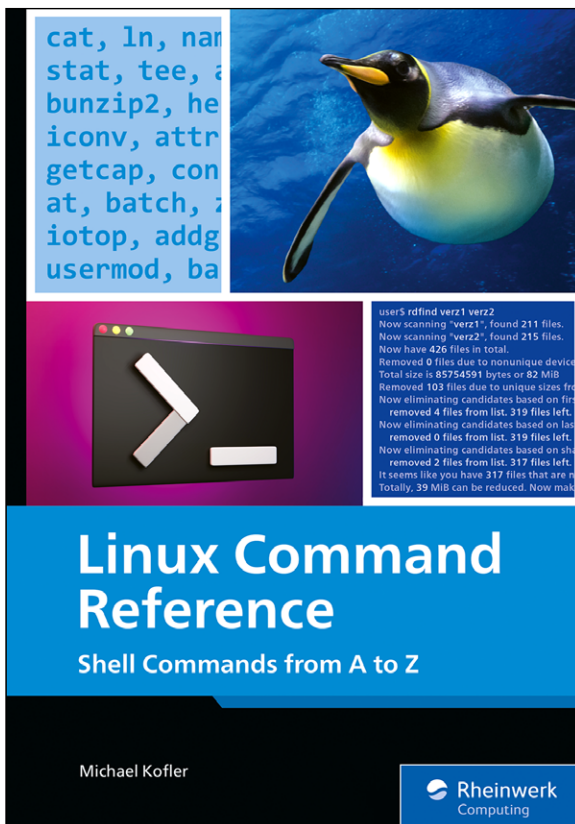
To take a look at the certificate file, you can run `openssl x509 -text`:

```
root# openssl x509 -text -in server.pem
Certificate:
  Data:
    Version: 1 (0x0)
    Serial Number: 12669601459972319941 (0xafd37766c36baac5)
    Signature Algorithm: sha256WithRSAEncryption
    Issuer: C=DE, L=Boston, O=John Doe,
            CN=www.company-abc.com/emailAddress=webmaster@company-abc.com
    Validity
      Not Before: Sep 28 14:48:03 2025 GMT
      Not After : Dec 10 14:48:03 2029 GMT      ...
```

The generation of a key, a certificate request, and self-signing can also be carried out in a single command: The following command generates a self-signed certificate for a mail server that is valid for 10 years. It's important that you enter the host name of your server as the common name when running `openssl`. Because the certificate itself is signed, your mail client will indicate during configuration that the certificate isn't trustworthy.

```
root# openssl req -new -x509 -days 3650 -nodes \
      -out /etc/ssl/certs/postfix.pem \
      -keyout /etc/ssl/private/postfix.key
...
Common Name (eg server FQDN or YOUR name) []: company-abc.com
root# chmod 400 /etc/ssl/private/postfix.key
```





Michael Kofler

# Linux Command Reference

## Shell Commands from A to Z

- More than 500 Linux commands and configuration files at your fingertips
- Explore commands by topic and alphabetically
- Learn to use commands with detailed examples and instructions



[rheinwerk-computing.com/6157](https://rheinwerk-computing.com/6157)

We hope you have enjoyed this reading sample. You may recommend or pass it on to others, but only in its entirety, including all pages. This reading sample and all its parts are protected by copyright law. All usage and exploitation rights are reserved by the author and the publisher.

Dr. Michael Kofler is a programmer and Linux administrator. He has been one of the most successful and versatile computing authors in the German-speaking world for many years. His current focuses include Linux, Docker, Git, hacking and security, Raspberry Pi, and the programming languages Swift, Java, Python, and Kotlin.

ISBN 978-1-4932-2749-5 • 493 pages • 07/2025

E-book: \$34.99 • Print book: \$39.95 • Bundle: \$49.99



**Rheinwerk**  
Publishing