System-Informationen und Konfiguration

LPI Essentials

Andreas B. Mundt
andreas.mundt@zsl-bw.de

8. Mai 2023

4.3 Datenspeicherung

Gewichtung: 3

Beschreibung: Wo verschiedene Arten von Informationen in einem Linux-System gespeichert werden.

Hauptwissensgebiete:

- Programme und Konfiguration
- Prozesse
- Speicheradressen
- Systembenachrichtigungen
- Protokollierung

Auszugsweise Liste wichtiger Dateien, Begriffe und Hilfsprogramme:

- `ps`, `top`, `free`
- `syslog`, `dmesg`
- `/etc/`, `/var/log/`
- `/boot/`, `/proc/`, `/dev/`, `/sys/`

Aus- und Überblick

Aufbau eines Computersystems

Anwendungsprogramme

Desktop, Browser, Textverarbeitung, Tabellenkalkulation, ...

Systemprogramme (Dienste/Daemone) und -Tools

Web-Server, Email-Server, FTP-Server, ...

Verwaltungs- und Konfigurationsprogramme, ...

Betriebssytem-Kern (Kernel)

Steuert und kontrolliert Hardware, vermittelt und verteilt Ressourcen an Programme, verwaltet Zugriffsrechte, ...

Hardware

Prozessor (CPU), Speicher (Cache, RAM, Festplatte), Netzwerk, Schnittstellen, Peripheriegeräte, ...

Aufbau eines Computersystems

Anwendungsprogramme
Desktop, Browser, Textverarbeitung, Tabellenkalkulation, ...

Systemprogramme (Dienste/Daemone) und -Tools
Web-Server, Email-Server, FTP-Server, ...
Verwaltungs- und Konfigurationsprogramme, ...

Betriebssytem-Kern (Kernel)
Steuert und kontrolliert Hardware, vermittelt und verteilt Ressourcen an Programme, verwaltet Zugriffsrechte, ...

Hardware
Prozessor (CPU), Speicher (Cache, RAM, Festplatte), Netzwerk, Schnittstellen, Peripheriegeräte, ...

Aufbau eines Computersystems

Anwendungsprogramme
Desktop, Browser, Textverarbeitung, Tabellenkalkulation, ...

Systemprogramme (Dienste/Daemone) und -Tools
Web-Server, Email-Server, FTP-Server, ...
Verwaltungs- und Konfigurationsprogramme, ...

Betriebssytem-Kern (Kernel)
Steuert und kontrolliert Hardware, vermittelt und verteilt Ressourcen an Programme, verwaltet Zugriffsrechte, ...

Hardware
Prozessor (CPU), Speicher (Cache, RAM, Festplatte), Netzwerk, Schnittstellen, Peripheriegeräte, ...

Aufbau eines Computersystems

Anwendungsprogramme

Desktop, Browser, Textverarbeitung, Tabellenkalkulation, ...

Systemprogramme (Dienste/Daemone) und -Tools

Web-Server, Email-Server, FTP-Server, ...

Verwaltungs- und Konfigurationsprogramme, ...

Betriebssytem-Kern (Kernel)

Steuert und kontrolliert Hardware, vermittelt und verteilt Ressourcen an Programme, verwaltet Zugriffsrechte, ...

Hardware

Prozessor (CPU), Speicher (Cache, RAM, Festplatte), Netzwerk, Schnittstellen, Peripheriegeräte, ...

Schichten-Architektur

In Linux hier enthalten: Tools zur Systemverwaltung, Paketmanagement

GUI-Programme
Für interaktive Nutzung mit Grafik
Bsp: Browser, Textverarbeitung, ...

Dienst / Daemon
Läuft im Hintergrund ohne Grafik / Konsole

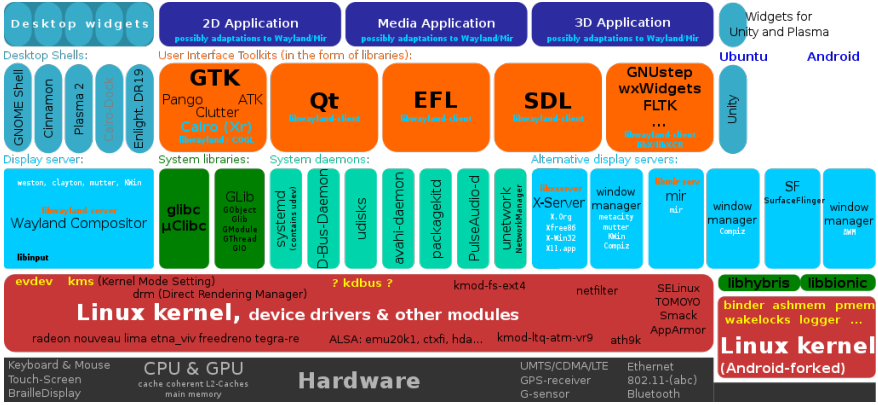
CLI-Programme
Für interaktive Nutzung der Konsole ohne Grafik

Grafische Oberfläche
Bsp: X-Server. Verteilt Ressourcen an GUI-Programme

Betriebssystem-Kern (Kernel)
Verwaltet und verteilt alle Ressourcen des Systems:
Prozesse, Rechenzeit, Speicherverbrauch, Zugriff auf Geräte, Rechte, Nutzer
Linux-Kernel selbst kann durch Module, Treiber, … im Betrieb verändert werden

Hardware
Prozessor (CPU), Speicher (Cache, RAM, Festplatte), Grafikkarte, Netzwerkkarte, Schnittstellen, Peripheriegeräte, ...

Aufbau des GNU/Linux Betriebssytems[1]



[1] https://de.wikipedia.org/wiki/Datei:Free_and_open-source-software_display_servers_and_UI_toolkits.svg

Filesystem Hierarchy Standard

- `/bin` : Essential user command binaries (for use by all users)
- `/boot` : Static files of the boot loader
- `/dev` : Device files
- `/etc` : Host-specific system configuration
- `/home` : User home directories (optional)
- `/lib` : Essential shared libraries and kernel modules
- `/lib<qual>` : Alternate format essential shared libraries (optional)
- `/media` : Mount point for removable media
- `/mnt` : Mount point for a temporarily mounted filesystem
- `/opt` : Add-on application software packages
- `/root` : Home directory for the root user (optional)
- `/run` : Run-time variable data
- `/sbin` : System binaries
- `/srv` : Data for services provided by this system
- `/tmp` : Temporary files
- `/usr` : Second major section of the filesystem, shareable, read-only data
- `/var` : Variable data files

`https://refspecs.linuxfoundation.org/fhs`

Wichtige Verzeichnisse

- `/etc/` Systemweite Konfiguration
- `/lib/` und `/usr/lib/`: Programm-Bibliotheken[2]
- `/bin/`, `/usr/bin/` und `/sbin/`, `/usr/sbin/`: (ausführbare[3]) Programme

Der Zweig `/var/` enthält sich mehr oder weniger ändernde (**var**iable) Daten:

- `/var/mail/` : eMails der Benutzer
- `/var/spool/` : z.B. Druckerwarteschlangen, ...
- `/var/www/` : lokaler Webserver
- `/var/log/` : Log-Dateien
- lokale Datenbanken
- ...

[2] engl. *Library*

[3] von engl. *binary*

Wichtige Verzeichnisse

- `/etc/` Systemweite Konfiguration
- `/lib/` und `/usr/lib/`: Programm-Bibliotheken[2]
- `/bin/`, `/usr/bin/` und `/sbin/`, `/usr/sbin/`: (ausführbare[3]) Programme

Der Zweig `/var/` enthält sich mehr oder weniger ändernde (**var**iable) Daten:

- `/var/mail/` : eMails der Benutzer
- `/var/spool/` : z.B. Druckerwarteschlangen, ...
- `/var/www/` : lokaler Webserver
- `/var/log/` : Log-Dateien
- lokale Datenbanken
- ...

[2] engl. *Library*
[3] von engl. *binary*

Systemkonfiguration in `/etc/`

Bis auf wenige Ausnahmen befindet sich die gesamte **systemweite Konfiguration** im Verzeichnis `/etc/` und erfolgt fast immer über normale Textdateien:

- Änderungen können mit jedem Editor vorgenommen werden.
- Die gesamte Konfiguration lässt sich wie Quellcode in einem VCS[4] verwalten (z.B. `etckeeper`).
- Die Flexibilität des einfachen Formats erlaubt Software verschiedenster Projekte auf verschiedensten Plattformen mit überall verfügbaren Werkzeugen zu konfigurieren.
- Durch die Möglichkeit von Kommentaren (beginnend mit #) enthält die Konfiguration z.T. gleich (einen Teil) ihrer Dokumentation.

[4] VCS: Version-Control–System

Systemkonfiguration in `/etc/`

Bis auf wenige Ausnahmen befindet sich die gesamte **systemweite Konfiguration** im Verzeichnis `/etc/` und erfolgt fast immer über normale Textdateien:

- Änderungen können mit jedem Editor vorgenommen werden.
- Die gesamte Konfiguration lässt sich wie Quellcode in einem VCS[4] verwalten (z.B. `etckeeper`).
- Die Flexibilität des einfachen Formats erlaubt Software verschiedenster Projekte auf verschiedensten Plattformen mit überall verfügbaren Werkzeugen zu konfigurieren.
- Durch die Möglichkeit von Kommentaren (beginnend mit #) enthält die Konfiguration z.T. gleich (einen Teil) ihrer Dokumentation.

[4] VCS: Version-Control–System

Systemkonfiguration in `/etc/`

Bis auf wenige Ausnahmen befindet sich die gesamte **systemweite Konfiguration** im Verzeichnis `/etc/` und erfolgt fast immer über normale Textdateien:

- Änderungen können mit jedem Editor vorgenommen werden.
- Die gesamte Konfiguration lässt sich wie Quellcode in einem VCS[4] verwalten (z.B. `etckeeper`).
- Die Flexibilität des einfachen Formats erlaubt Software verschiedenster Projekte auf verschiedensten Plattformen mit überall verfügbaren Werkzeugen zu konfigurieren.
- Durch die Möglichkeit von Kommentaren (beginnend mit #) enthält die Konfiguration z.T. gleich (einen Teil) ihrer Dokumentation.

[4] VCS: Version-Control–System

Systemkonfiguration in `/etc/`

Bis auf wenige Ausnahmen befindet sich die gesamte **systemweite Konfiguration** im Verzeichnis `/etc/` und erfolgt fast immer über normale Textdateien:

- Änderungen können mit jedem Editor vorgenommen werden.
- Die gesamte Konfiguration lässt sich wie Quellcode in einem VCS[4] verwalten (z.B. `etckeeper`).
- Die Flexibilität des einfachen Formats erlaubt Software verschiedenster Projekte auf verschiedensten Plattformen mit überall verfügbaren Werkzeugen zu konfigurieren.
- Durch die Möglichkeit von Kommentaren (beginnend mit #) enthält die Konfiguration z.T. gleich (einen Teil) ihrer Dokumentation.

[4] VCS: Version-Control–System

Konfigurationsdateien und -Verzeichnisse in /etc/:

```
ls -FC /etc | head -n 25
```

```
adduser.conf            initramfs-tools/    pulse/
adjtime                 inputrc             python3/
aliases                 insserv.conf.d/     python3.9/
alsa/                   ipp-usb/            qemu-ifdown*
alternatives/           iproute2/           qemu-ifup*
anacrontab              issue               radvd.conf
apache2/                issue.net           rc0.d/
apparmor/               java/               rc1.d/
apparmor.d/             java-11-openjdk/    rc2.d/
apt/                    java-17-openjdk/    rc3.d/
apt-cacher-ng/          kernel/             rc4.d/
avahi/                  kernel-img.conf     rc5.d/
bash.bashrc             ldap/               rc6.d/
bash_completion         ld.so.cache         rcS.d/
bash_completion.d/      ld.so.conf          reader.conf.d/
bindresvport.blacklist  ld.so.conf.d/       reportbug.conf
binfmt.d/               libaudit.conf       request-key.conf
bluetooth/              libblockdev/        request-key.d/
ca-certificates/        libccid_Info.plist  resolv.conf@
ca-certificates.conf    libibverbs.d/       resolvconf/
chatscripts/            libnl-3/            rlinetd.conf
console-setup/          libpaper.d/         rmt@
cron.d/                 libreoffice/        rpc
cron.daily/             libvirt/            rsyslog.conf
cron.hourly/            lighttpd/           rsyslog.d/
```

Beispiel /etc/fstab:

```
sed -r "s/[[:space:]]+/ /g" /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# systemd generates mount units based on this file, see systemd.mount(5).
# Please run 'systemctl daemon-reload' after making changes here.
#
# <file system> <mount point> <type> <options> <dump> <pass>
/dev/mapper/tuxedo--vg-root / ext4 errors=remount-ro 0 1
# /boot was on /dev/nvme0n1p2 during installation
UUID=598284a1-e1e7-41ad-87b7-afc1632a1eb0 /boot ext2 defaults 0 2
# /boot/efi was on /dev/nvme0n1p1 during installation
UUID=DA64-4A09 /boot/efi vfat umask=0077 0 1
/dev/mapper/tuxedo--vg-home /home ext4 defaults 0 2
/dev/mapper/tuxedo--vg-tmp /tmp ext4 defaults 0 2
/dev/mapper/tuxedo--vg-var /var ext4 defaults 0 2
/dev/mapper/tuxedo--vg-swap_1 none swap sw 0 0
/usr/lib/debian-installer/ /var/lib/tftpboot/d-i/n-pkg/ none bind,ro 0 0
/home/andi/Downloads/debian-live-11.2.0-amd64-kde+nonfree.iso /var/lib/tftpboot/d-i/n-live/debian-l
```

Manual Page von fstab:

```
man fstab | head -n4
FSTAB(5)                        File Formats                        FSTAB(5)

NAME
       fstab - static information about the filesystems
```

`/proc/, /sys/, /dev/ ...`

Der Kernel bietet virtuelle Dateisystem zum Abfragen und Verwalten von Informationen an:

- `/proc/` : Jeder Prozess hat dort ein Unterverzeichnis (PID). Zusätzlich liegen dort einige Daten über das System (RAM, CPU, geladene Kernelmodule, Laufwerke, ...).
- `/dev/` : Geräteschnittstellen
- `/sys/` : Informationen über die Geräte

In der Praxis greift man selten direkt auf die Kernel-Schnittstellen zu sondern verwendet Programme, die die Informationen aufbereitet darstellen.

`/proc/`, `/sys/`, `/dev/` ...

Der Kernel bietet virtuelle Dateisystem zum Abfragen und Verwalten von Informationen an:

- `/proc/` : Jeder Prozess hat dort ein Unterverzeichnis (PID). Zusätzlich liegen dort einige Daten über das System (RAM, CPU, geladene Kernelmodule, Laufwerke, ...).
- `/dev/` : Geräteschnittstellen
- `/sys/` : Informationen über die Geräte

In der Praxis greift man selten direkt auf die Kernel-Schnittstellen zu sondern verwendet Programme, die die Informationen aufbereitet darstellen.

`/proc/, /sys/, /dev/ ...`

Der Kernel bietet virtuelle Dateisystem zum Abfragen und Verwalten von Informationen an:

- `/proc/` : Jeder Prozess hat dort ein Unterverzeichnis (PID). Zusätzlich liegen dort einige Daten über das System (RAM, CPU, geladene Kernelmodule, Laufwerke, ...).
- `/dev/` : Geräteschnittstellen
- `/sys/` : Informationen über die Geräte

In der Praxis greift man selten direkt auf die Kernel-Schnittstellen zu sondern verwendet Programme, die die Informationen aufbereitet darstellen.

`/proc/, /sys/, /dev/ ...`

Der Kernel bietet virtuelle Dateisystem zum Abfragen und Verwalten von Informationen an:

- `/proc/` : Jeder Prozess hat dort ein Unterverzeichnis (PID). Zusätzlich liegen dort einige Daten über das System (RAM, CPU, geladene Kernelmodule, Laufwerke, ...).
- `/dev/` : Geräteschnittstellen
- `/sys/` : Informationen über die Geräte

In der Praxis greift man selten direkt auf die Kernel-Schnittstellen zu sondern verwendet Programme, die die Informationen aufbereitet darstellen.

`/proc/`, `/sys/`, `/dev/` ...

Der Kernel bietet virtuelle Dateisystem zum Abfragen und Verwalten von Informationen an:

- `/proc/` : Jeder Prozess hat dort ein Unterverzeichnis (PID). Zusätzlich liegen dort einige Daten über das System (RAM, CPU, geladene Kernelmodule, Laufwerke, ...).
- `/dev/` : Geräteschnittstellen
- `/sys/` : Informationen über die Geräte

In der Praxis greift man selten direkt auf die Kernel-Schnittstellen zu sondern verwendet Programme, die die Informationen aufbereitet darstellen.

Kernelparameter in /proc/sys/

Die virtuellen Dateien in /proc sind fast alle read-only. In /proc/sys/ kann root[5] Kernelparameter verändern:

```
echo 1 >/proc/sys/net/ipv4/ip_forward  # aktiviert v4-Routing
echo 0 >/proc/sys/net/ipv4/ip_forward  # deaktiviert es
cat /proc/sys/net/ipv4/ip_forward      # prüft den Routing-Status
echo 50000 >/proc/sys/kernel/pid_max   # Obergrenze Prozess-IDs
```

Statt mit echo/cat kann man auch sysctl verwenden:

```
sysctl kernel.pid_max
sysctl -w kernel.pid_max=50000
```

Persistente Einstellungen in /etc/sysctl.conf bzw. /etc/sysctl.d/.

[5] Als sudo-User: `echo XYZ | sudo tee /proc/sys/.../PSEUDOFILE`

Beispiele für Informationen in `/proc/`

- `/proc/cmdline` : Kernel Boot Parameter
- `/proc/cpuinfo` : $\rightarrow$ `lscpu`
- `/proc/modules` : $\rightarrow$ `lsmod`
- `/proc/meminfo` : $\rightarrow$ `free`

Aus- und Überblick

Prozesse anzeigen mit ps

Alle Arbeiten, die der Rechner verrichtet, wird in Prozessen verwaltet. Das Kommando ps zeigt die auf dem aktuellen Terminal laufenden Prozesse an:

```
ps
    PID TTY          TIME CMD
 139598 pts/10   00:00:00 sh
 139599 pts/10   00:00:04 pdflatex
 139633 pts/10   00:00:00 sh
 139634 pts/10   00:00:00 bash
 139635 pts/10   00:00:00 bash
 139636 pts/10   00:00:00 ps
```

```
ps -l  ## -l long
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000  139598    3485  0  80   0 -   621 do_wai pts/10   00:00:00 sh
0 S  1000  139599  139598 99  80   0 - 31265 do_wai pts/10   00:00:04 pdflatex
0 S  1000  139637  139599  0  80   0 -   621 do_wai pts/10   00:00:00 sh
0 S  1000  139638  139637  0  80   0 -  1705 do_wai pts/10   00:00:00 bash
0 S  1000  139639  139638  0  80   0 -  1705 do_wai pts/10   00:00:00 bash
4 R  1000  139640  139639  0  80   0 -  2420 -      pts/10   00:00:00 ps
```

Prozesse anzeigen mit ps

Alle Arbeiten, die der Rechner verrichtet, wird in Prozessen verwaltet. Das Kommando ps zeigt die auf dem aktuellen Terminal laufenden Prozesse an:

```
ps
    PID TTY          TIME CMD
 139598 pts/10   00:00:00 sh
 139599 pts/10   00:00:04 pdflatex
 139641 pts/10   00:00:00 sh
 139642 pts/10   00:00:00 bash
 139643 pts/10   00:00:00 bash
 139644 pts/10   00:00:00 ps
```

```
ps -l  ## -l long
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1000  139598    3485  0  80   0 -   621 do_wai pts/10   00:00:00 sh
0 S  1000  139599  139598 99  80   0 - 31265 do_wai pts/10   00:00:04 pdflatex
0 S  1000  139645  139599  0  80   0 -   621 do_wai pts/10   00:00:00 sh
0 S  1000  139646  139645  0  80   0 -  1705 do_wai pts/10   00:00:00 bash
0 S  1000  139647  139646  0  80   0 -  1705 do_wai pts/10   00:00:00 bash
4 R  1000  139648  139647  0  80   0 -  2420 -      pts/10   00:00:00 ps
```

Optionen für ps

```
ps -e -f | head  ## -e everything -f full
UID          PID    PPID  C STIME TTY          TIME CMD
root           1       0  0 Apr17 ?        00:00:06 /sbin/init splash
root           2       0  0 Apr17 ?        00:00:00 [kthreadd]
root           3       2  0 Apr17 ?        00:00:00 [rcu_gp]
root           4       2  0 Apr17 ?        00:00:00 [rcu_par_gp]
root           5       2  0 Apr17 ?        00:00:00 [slub_flushwq]
root           6       2  0 Apr17 ?        00:00:00 [netns]
root           8       2  0 Apr17 ?        00:00:00 [kworker/0:0H-events_highpri]
root          10       2  0 Apr17 ?        00:00:00 [mm_percpu_wq]
root          11       2  0 Apr17 ?        00:00:00 [rcu_tasks_kthread]
```

⋮ ⋮ ⋮ ⋮

```
ps -e -f | tail  ## -e everything -f full
root      138854       2  0 13:35 ?        00:00:00 [kworker/u32:12-kcryptd/254:0]
root      138926       2  0 13:37 ?        00:00:00 [kworker/0:1]
root      139145       2  0 13:42 ?        00:00:00 [kworker/0:2-events]
andi      139598    3485  0 13:46 pts/10   00:00:00 /bin/sh -c pdflatex
-file-line-error -shell-escape  -interaction=nonstopmode 4.3_SystemInfoConfig.tex
andi      139599  139598 99 13:46 pts/10   00:00:04 pdflatex -file-line-error -shell-escape -intera
andi      139654  139599  0 13:46 pts/10   00:00:00 sh -c bash -c "bash 4.3_SystemInfoConfig.sh >4.
andi      139655  139654  0 13:46 pts/10   00:00:00 bash -c bash 4.3_SystemInfoConfig.sh >4.3_Syste
andi      139656  139655  0 13:46 pts/10   00:00:00 bash 4.3_SystemInfoConfig.sh
andi      139657  139656  0 13:46 pts/10   00:00:00 ps -e -f
andi      139658  139656  0 13:46 pts/10   00:00:00 tail
```

Prozessbaum anzeigen mit pstree

```
pstree | head -n 25  ## alternativ: ps -ef --forest oder -H (hierarchy)
systemd-+-ModemManager---2*[{ModemManager}]
        |-NetworkManager---2*[{NetworkManager}]
        |-Xwayland---59*[{Xwayland}]
        |-apt-cacher-ng---2*[{apt-cacher-ng}]
        |-avahi-daemon---avahi-daemon
        |-bluetoothd
        |-colord---2*[{colord}]
        |-cron
        |-cups-browsed---2*[{cups-browsed}]
        |-cupsd
        |-dbus-daemon
        |-dmeventd---2*[{dmeventd}]
        |-dnsmasq---dnsmasq
        |-exim4
        |-firefox-esr-+-36*[Isolated Web Co---26*[{Isolated Web Co}]]
        |             |-Isolated Web Co---27*[{Isolated Web Co}]
        |             |-4*[Isolated Web Co---25*[{Isolated Web Co}]]
        |             |-Isolated Web Co---28*[{Isolated Web Co}]
        |             |-Isolated Web Co---24*[{Isolated Web Co}]
        |             |-Isolated Web Co---31*[{Isolated Web Co}]
        |             |-Privileged Cont---26*[{Privileged Cont}]
        |             |-RDD Process---2*[{RDD Process}]
        |             |-Socket Process---5*[{Socket Process}]
        |             |-Utility Process---2*[{Utility Process}]
        |             |-3*[Web Content---9*[{Web Content}]]
```

Prozessbaum anzeigen mit ps -forest

```
ps --forest
    PID TTY          TIME CMD
 139598 pts/10   00:00:00 sh
 139599 pts/10   00:00:04  \_ pdflatex
 139665 pts/10   00:00:00      \_ sh
 139666 pts/10   00:00:00          \_ bash
 139667 pts/10   00:00:00              \_ bash
 139668 pts/10   00:00:00                  \_ ps
```

Signale an Prozesse senden

Verhält sich ein Prozess nicht so wie man es sich von ihm wünscht, kann er z.B. durch Senden eines Signals beendet werden. Hat man die PID ausfindig gemacht, so sendet:

```
kill <PID>
```

dem Prozess <PID> ein TERM-Signal: Der Prozess wird aufgefordert sich umgehend zu beenden. Ist der Prozess dazu nicht mehr in der Lage, so kann man ihn auf etwas rustikalere Art mit dem KILL-Signal beenden:

```
kill -9 <PID>
```

Allen Prozessinstanzen eines Programms mit Namen <NAME> sendet der Befehl:

```
killall <NAME>
```

ein TERM-Signal. Auch hier kann optional ein KILL-Signal gesendet werden.

Es gibt zahlreiche weitere Signale; siehe: `man 7 signal`.

Speicherverbrauch anzeigen mit `free`

Den gesamten Speicherverbrauch zeigt das Programm `free` an:

```
free
               total        used        free      shared  buff/cache   available
Mem:        65258688    14056816    47023784      186724     4178088    50283528
Swap:         999420           0      999420
```

Optional auch „human readable“:

```
free -h
               total        used        free      shared  buff/cache   available
Mem:            62Gi        13Gi        44Gi       182Mi       4.0Gi        47Gi
Swap:          975Mi          0B       975Mi
```

total	Total installed memory (MemTotal and SwapTotal in `/proc/meminfo`)
used	Used memory (calculated as total - free - buffers - cache)
free	Unused memory (MemFree and SwapFree in `/proc/meminfo`)
shared	Memory used (mostly) by tmpfs (Shmem in `/proc/meminfo`)
buffers	Memory used by kernel buffers (Buffers in `/proc/meminfo`)
cache	Memory used by the page cache and slabs (Cached and SReclaimable in `/proc/meminfo`)
buff/cache	Sum of buffers and cache
available	Estimation of how much memory is available for starting new applications

Speicherverbrauch anzeigen mit `free`

Den gesamten Speicherverbrauch zeigt das Programm `free` an:

```
free
               total        used        free      shared  buff/cache   available
Mem:        65258688    14057320    47023280      186724     4178088    50283024
Swap:         999420           0      999420
```

Optional auch „human readable“:

```
free -h
               total        used        free      shared  buff/cache   available
Mem:            62Gi        13Gi        44Gi       182Mi       4.0Gi        47Gi
Swap:          975Mi          0B       975Mi
```

total	Total installed memory (MemTotal and SwapTotal in `/proc/meminfo`)
used	Used memory (calculated as total - free - buffers - cache)
free	Unused memory (MemFree and SwapFree in `/proc/meminfo`)
shared	Memory used (mostly) by tmpfs (Shmem in `/proc/meminfo`)
buffers	Memory used by kernel buffers (Buffers in `/proc/meminfo`)
cache	Memory used by the page cache and slabs (Cached and SReclaimable in `/proc/meminfo`)
buff/cache	Sum of buffers and cache
available	Estimation of how much memory is available for starting new applications, ...

Systemressourcen interaktiv verfolgen mit `top`

```
top -n 1
top - 13:46:30 up 21 days,  2:40,  1 user,  load average: 1.49, 1.14, 0.98
Tasks: 441 total,   2 running, 439 sleeping,   0 stopped,   0 zombie
%Cpu(s):  5.3 us,  0.7 sy,  0.0 ni, 94.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  63729.2 total,  45922.8 free,  13726.2 used,   4080.2 buff/cache
MiB Swap:    976.0 total,    976.0 free,      0.0 used.  49106.2 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  51488 andi      20   0 4162704 852012 203376 R  76.5   1.3   2410:38 firefo+
   2257 andi      20   0 4579656 128992  77384 S   5.9   0.2 130:25.58 Xwayla+
 118936 libvirt+  20   0 5242948   4.2g  28300 S   5.9   6.7  17:31.59 qemu-s+
 139688 andi      20   0   10420   3932   3196 R   5.9   0.0   0:00.02 top
      1 root      20   0  165176  11348   7592 S   0.0   0.0   0:06.51 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.23 kthrea+
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_pa+
      5 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 slub_f+
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 netns
      8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworke+
     10 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_per+
     11 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_ta+
     12 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_ta+
     13 root      20   0       0      0      0 I   0.0   0.0   0:00.00 rcu_ta+
     14 root      20   0       0      0      0 S   0.0   0.0   0:17.88 ksofti+
     15 root      20   0       0      0      0 I   0.0   0.0   4:16.05 rcu_pr+
```

h oder ? für Hilfe und weitere (Um-)schaltmöglichkeiten!

Erste Zeile auch mit: `uptime`. Kernel Version: `uname -a`

Anwendungsbeispiel mit top

```
while true ; do echo hallo > /dev/null ; done &
ps  ## Prozess sollte im Hintergrund laufen
top  ## Prozess ~100% CPU → k evtl. 9
ps
```

Aus- und Überblick

Protokollierung[6] von Systemereignissen

- Kernel und Systemdienste (Daemone) informieren den Syslog-Dienst (`systemd-journal` oder `rsyslogd`) über Ereignisse
- Syslog-Dienst kümmert sich u.a. um Komprimierung etc.
- Umfang/Wichtigkeit ist jeweils konfigurierbar
- Nachrichten werden im Verzeichnis `/var/log/` in die entsprechenden Dateien abgelegt
- Nachrichten des Kernels werden durch den klog-Daemon (`/dev/kmsg`) ebenfalls über den Syslog-Dienst weitergereicht
- `dmesg` zeigt die letzen Kernel-Meldungen an (Kernel-Ring-Buffer, root-Rechte erforderlich:

dmesg: aktuelles Kernel-Log

```
dmesg | tail -n 25
dmesg: read kernel buffer failed: Operation not permitted
```

[6] *logging, man denke an das „Logbuch“ in der Schifffahrt*

Protokollierung[6] von Systemereignissen

- Kernel und Systemdienste (Daemone) informieren den Syslog-Dienst (`systemd-journal` oder `rsyslogd`) über Ereignisse
- Syslog-Dienst kümmert sich u.a. um Komprimierung etc.
- Umfang/Wichtigkeit ist jeweils konfigurierbar
- Nachrichten werden im Verzeichnis `/var/log/` in die entsprechenden Dateien abgelegt
- Nachrichten des Kernels werden durch den klog-Daemon (`/dev/kmsg`) ebenfalls über den Syslog-Dienst weitergereicht
- `dmesg` zeigt die letzen Kernel-Meldungen an (Kernel-Ring-Buffer, root-Rechte erforderlich:

dmesg: aktuelles Kernel-Log

```
dmesg | tail -n 25
dmesg: read kernel buffer failed: Operation not permitted
```

[6] *logging, man denke an das „Logbuch“ in der Schifffahrt*

Protokollierung[6] von Systemereignissen

- Kernel und Systemdienste (Daemone) informieren den Syslog-Dienst (`systemd-journal` oder `rsyslogd`) über Ereignisse
- Syslog-Dienst kümmert sich u.a. um Komprimierung etc.
- Umfang/Wichtigkeit ist jeweils konfigurierbar
- Nachrichten werden im Verzeichnis `/var/log/` in die entsprechenden Dateien abgelegt
- Nachrichten des Kernels werden durch den klog-Daemon (`/dev/kmsg`) ebenfalls über den Syslog-Dienst weitergereicht
- `dmesg` zeigt die letzen Kernel-Meldungen an (Kernel-Ring-Buffer, root-Rechte erforderlich:

dmesg: aktuelles Kernel-Log

```
dmesg | tail -n 25
dmesg: read kernel buffer failed: Operation not permitted
```

[6] *logging, man denke an das „Logbuch“ in der Schifffahrt*

Protokollierung[6] von Systemereignissen

- Kernel und Systemdienste (Daemone) informieren den Syslog-Dienst (`systemd-journal` oder `rsyslogd`) über Ereignisse
- Syslog-Dienst kümmert sich u.a. um Komprimierung etc.
- Umfang/Wichtigkeit ist jeweils konfigurierbar
- Nachrichten werden im Verzeichnis `/var/log/` in die entsprechenden Dateien abgelegt
- Nachrichten des Kernels werden durch den klog-Daemon (`/dev/kmsg`) ebenfalls über den Syslog-Dienst weitergereicht
- `dmesg` zeigt die letzen Kernel-Meldungen an (Kernel-Ring-Buffer, root-Rechte erforderlich:

dmesg: aktuelles Kernel-Log

```
dmesg | tail -n 25
dmesg: read kernel buffer failed: Operation not permitted
```

[6] *logging, man denke an das „Logbuch“ in der Schifffahrt*

Protokollierung[6] von Systemereignissen

- Kernel und Systemdienste (Daemone) informieren den Syslog-Dienst (`systemd-journal` oder `rsyslogd`) über Ereignisse
- Syslog-Dienst kümmert sich u.a. um Komprimierung etc.
- Umfang/Wichtigkeit ist jeweils konfigurierbar
- Nachrichten werden im Verzeichnis `/var/log/` in die entsprechenden Dateien abgelegt
- Nachrichten des Kernels werden durch den klog-Daemon (`/dev/kmsg`) ebenfalls über den Syslog-Dienst weitergereicht
- `dmesg` zeigt die letzen Kernel-Meldungen an (Kernel-Ring-Buffer, root-Rechte erforderlich:

`dmesg`: aktuelles Kernel-Log

```
dmesg | tail -n 25
dmesg: read kernel buffer failed: Operation not permitted
```

[6] *logging, man denke an das „Logbuch“ in der Schifffahrt*

Protokollierung[6] von Systemereignissen

- Kernel und Systemdienste (Daemone) informieren den Syslog-Dienst (`systemd-journal` oder `rsyslogd`) über Ereignisse
- Syslog-Dienst kümmert sich u.a. um Komprimierung etc.
- Umfang/Wichtigkeit ist jeweils konfigurierbar
- Nachrichten werden im Verzeichnis `/var/log/` in die entsprechenden Dateien abgelegt
- Nachrichten des Kernels werden durch den klog-Daemon (`/dev/kmsg`) ebenfalls über den Syslog-Dienst weitergereicht
- `dmesg` zeigt die letzen Kernel-Meldungen an (Kernel-Ring-Buffer, root-Rechte erforderlich:

dmesg: aktuelles Kernel-Log

```
dmesg | tail -n 25
dmesg: read kernel buffer failed: Operation not permitted
```

[6] *logging, man denke an das „Logbuch“ in der Schifffahrt*

Protokollierung[6] von Systemereignissen

- Kernel und Systemdienste (Daemone) informieren den Syslog-Dienst (`systemd-journal` oder `rsyslogd`) über Ereignisse
- Syslog-Dienst kümmert sich u.a. um Komprimierung etc.
- Umfang/Wichtigkeit ist jeweils konfigurierbar
- Nachrichten werden im Verzeichnis `/var/log/` in die entsprechenden Dateien abgelegt
- Nachrichten des Kernels werden durch den klog-Daemon (`/dev/kmsg`) ebenfalls über den Syslog-Dienst weitergereicht
- `dmesg` zeigt die letzen Kernel-Meldungen an (Kernel-Ring-Buffer, root-Rechte erforderlich:

dmesg: aktuelles Kernel-Log

```
dmesg | tail -n 25
dmesg: read kernel buffer failed: Operation not permitted
```

[6] *logging, man denke an das „Logbuch“ in der Schifffahrt*

Protokollierung[6] von Systemereignissen

- Kernel und Systemdienste (Daemone) informieren den Syslog-Dienst (`systemd-journal` oder `rsyslogd`) über Ereignisse
- Syslog-Dienst kümmert sich u.a. um Komprimierung etc.
- Umfang/Wichtigkeit ist jeweils konfigurierbar
- Nachrichten werden im Verzeichnis `/var/log/` in die entsprechenden Dateien abgelegt
- Nachrichten des Kernels werden durch den klog-Daemon (`/dev/kmsg`) ebenfalls über den Syslog-Dienst weitergereicht
- `dmesg` zeigt die letzen Kernel-Meldungen an (Kernel-Ring-Buffer, root-Rechte erforderlich:

`dmesg`: aktuelles Kernel-Log

```
dmesg | tail -n 25
dmesg: read kernel buffer failed: Operation not permitted
```

[6] *logging, man denke an das „Logbuch“ in der Schifffahrt*

Log-Dateien in /var/log/

Logs in /var/log/:

```
ls -FC /var/log/ | head -n 25
```

```
alternatives.log         boot.log.3        dpkg.log.2.gz   messages
alternatives.log.1       boot.log.4        dpkg.log.3.gz   messages.1
alternatives.log.10.gz   boot.log.5        dpkg.log.4.gz   messages.2.gz
alternatives.log.11.gz   boot.log.6        dpkg.log.5.gz   messages.3.gz
alternatives.log.12.gz   boot.log.7        dpkg.log.6.gz   messages.4.gz
alternatives.log.2.gz    btmp              dpkg.log.7.gz   mysql/
alternatives.log.3.gz    btmp.1            dpkg.log.8.gz   openvpn/
alternatives.log.4.gz    cups/             dpkg.log.9.gz   private/
alternatives.log.5.gz    daemon.log        exim4/          runit/
alternatives.log.6.gz    daemon.log.1      faillog         syslog
alternatives.log.7.gz    daemon.log.2.gz   firebird/       syslog.1
alternatives.log.8.gz    daemon.log.3.gz   firewalld       syslog.2.gz
alternatives.log.9.gz    daemon.log.4.gz   fontconfig.log  syslog.3.gz
apt/                     debug             installer/      syslog.4.gz
apt-cacher-ng/           debug.1           journal/        user.log
auth.log                 debug.2.gz        kern.log        user.log.1
auth.log.1               debug.3.gz        kern.log.1      user.log.2.gz
auth.log.2.gz            debug.4.gz        kern.log.2.gz   user.log.3.gz
auth.log.3.gz            dpkg.log          kern.log.3.gz   user.log.4.gz
auth.log.4.gz            dpkg.log.1        kern.log.4.gz   wtmp
boot.log                 dpkg.log.10.gz    lastlog
boot.log.1               dpkg.log.11.gz    libvirt/
boot.log.2               dpkg.log.12.gz    lighttpd/
```

Logs mit systemd-journal

/var/log/README:

```
cat 4.3_README.txt
You are looking for the traditional text log files in /var/log, and they are
gone?

Here's an explanation on what's going on:

You are running a systemd-based OS where traditional syslog has been replaced
with the Journal. The journal stores the same (and more) information as classic
syslog. To make use of the journal and access the collected log data simply
invoke "journalctl", which will output the logs in the identical text-based
format the syslog files in /var/log used to be. For further details, please
refer to journalctl(1).

Alternatively, consider installing one of the traditional syslog
implementations available for your distribution, which will generate the
classic log files for you. Syslog implementations such as syslog-ng or rsyslog
may be installed side-by-side with the journal and will continue to function
the way they always did.

Thank you!

Further reading:
        man:journalctl(1)
        man:systemd-journald.service(8)
        man:journald.conf(5)
        http://0pointer.de/blog/projects/the-journal.html
```

Logs mit `journalctl`

`zless /usr/share/doc/systemd/README.Debian.gz`:

```
zcat /usr/share/doc/systemd/README.Debian.gz | head -n 12 | tail -4
systemd will make the journal files owned by the "systemd-journal" group and
add an ACL for read permissions for users in the "adm" group.
To grant a user read access to the system journal, add them to one of the two
groups.
```

Also z.B. `addgroup andi adm`

Logs mit journalctl

journalctl:

```
journalctl -b | head
-- Journal begins at Mon 2021-05-17 21:36:47 CEST, ends at Mon 2023-05-08 13:46:27 CEST. --
Apr 17 11:05:29 tuxe kernel: Linux version 6.1.0-0.deb11.5-amd64 (debian-kernel@lists.debian.org) (
Apr 17 11:05:29 tuxe kernel: Command line: BOOT_IMAGE=/vmlinuz-6.1.0-0.deb11.5-amd64 root=/dev/mapp
Apr 17 11:05:29 tuxe kernel: x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point registers
Apr 17 11:05:29 tuxe kernel: x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
Apr 17 11:05:29 tuxe kernel: x86/fpu: Supporting XSAVE feature 0x004: 'AVX registers'
Apr 17 11:05:29 tuxe kernel: x86/fpu: xstate_offset[2]:  576, xstate_sizes[2]:  256
Apr 17 11:05:29 tuxe kernel: x86/fpu: Enabled xstate features 0x7, context size is 832 bytes, using
Apr 17 11:05:29 tuxe kernel: signal: max sigframe size: 1776
Apr 17 11:05:29 tuxe kernel: BIOS-provided physical RAM map:
```

⋮ ⋮ ⋮ ⋮

```
journalctl -b | tail
May 08 13:43:17 tuxe kernel: atkbd serio0: Unknown key pressed (translated set 2, code 0xf8 on isa0
May 08 13:43:17 tuxe kernel: atkbd serio0: Use 'setkeycodes e078 <keycode>' to make it known.
May 08 13:43:32 tuxe kernel: atkbd serio0: Unknown key released (translated set 2, code 0xf8 on isa
May 08 13:43:32 tuxe kernel: atkbd serio0: Use 'setkeycodes e078 <keycode>' to make it known.
May 08 13:44:14 tuxe audit[139225]: AVC apparmor="DENIED" operation="file_inherit" profile="man_gro
May 08 13:44:14 tuxe kernel: audit: type=1400 audit(1683546254.086:247): apparmor="DENIED" operatio
May 08 13:46:14 tuxe audit[139456]: AVC apparmor="DENIED" operation="file_inherit" profile="man_gro
May 08 13:46:14 tuxe kernel: audit: type=1400 audit(1683546374.776:248): apparmor="DENIED" operatio
May 08 13:46:27 tuxe audit[139628]: AVC apparmor="DENIED" operation="file_inherit" profile="man_gro
May 08 13:46:27 tuxe kernel: audit: type=1400 audit(1683546387.691:249): apparmor="DENIED" operatio
```

Log eines Dienstes journalctl

Evtl. nach Eingabe von `journalctl -u` mit TAB mögliche Dienste anzeigen lassen. Beispiele:

journalctl -u fstrim.service:

```
journalctl -b -2 -u fstrim.service | tail
Apr 03 08:31:42 tuxe systemd[1]: Finished Discard unused blocks on filesystems from /etc/fstab.
Apr 10 09:20:50 tuxe systemd[1]: Starting Discard unused blocks on filesystems from /etc/fstab...
Apr 10 09:21:06 tuxe fstrim[544323]: /var: 3.4 GiB (3602112512 bytes) trimmed on /dev/mapper/tuxedo
Apr 10 09:21:06 tuxe fstrim[544323]: /tmp: 202.3 MiB (212111360 bytes) trimmed on /dev/mapper/tuxed
Apr 10 09:21:06 tuxe fstrim[544323]: /home: 6.5 GiB (6985957376 bytes) trimmed on /dev/mapper/tuxed
Apr 10 09:21:06 tuxe fstrim[544323]: /boot/efi: 497.4 MiB (521543680 bytes) trimmed on /dev/nvme0n1
Apr 10 09:21:06 tuxe fstrim[544323]: /boot: 7.2 MiB (7524352 bytes) trimmed on /dev/nvme0n1p2
Apr 10 09:21:06 tuxe fstrim[544323]: /: 1.9 GiB (2015641600 bytes) trimmed on /dev/mapper/tuxedo--v
Apr 10 09:21:06 tuxe systemd[1]: fstrim.service: Succeeded.
Apr 10 09:21:06 tuxe systemd[1]: Finished Discard unused blocks on filesystems from /etc/fstab.
```

journalctl -u logrotate.service:

```
journalctl -b -u logrotate.service | tail
May 02 10:48:15 tuxe systemd[1]: Starting Rotate log files...
May 02 10:48:15 tuxe systemd[1]: logrotate.service: Succeeded.
May 02 10:48:15 tuxe systemd[1]: Finished Rotate log files.
May 06 08:33:00 tuxe systemd[1]: Condition check resulted in Rotate log files being skipped.
May 07 11:24:54 tuxe systemd[1]: Starting Rotate log files...
May 07 11:24:54 tuxe systemd[1]: logrotate.service: Succeeded.
May 07 11:24:54 tuxe systemd[1]: Finished Rotate log files.
```

Aufgaben und Übungen

Konfigurationsdateien unter `/etc/`

1. Wechseln Sie ins Verzeichnis `/etc/` und studieren Sie einige Konfigurationsdateien Ihrer Wahl.
2. Suchen Sie Konfigurationsdateien zu denen eine Manual-Page existiert. Werfen Sie einen Blick auf selbige.

Systemanalyse mit `ps, pstree, free` und `top`

1. Machen Sie sich mit den Programme `ps, pstree, free` und `top` vertraut. Werfen Sie u.a. einen Blick auf die jeweilige Manual-Page und nützliche Optionen.
2. Starten Sie einige Prozess (z.B. mit `sleep 600`) und suchen sie die zugehörigen Zeilen in den Ausgaben von `ps, pstree` und `top`.
3. Terminieren Sie die Prozesse mit `kill`, `killall` und durch Auswahl mittels k in `top`.

Logs analysieren in `/var/log/` und mit `journalctl`

1. Schauen Sie sich Logdateien des Systems sowie einzelner Dienste in `/var/log/` an.
2. Machen Sie sich mit Optionen für `journalctl` vertraut und probieren Sie einige aus. Was bewirkt der Aufruf von `journalctl -f`? Probieren Sie `journalctl -u` TAB TAB aus.
3. Führen Sie `dmesg` aus und vergleichen Sie die Ausgabe mit `journalctl -b` und `journalctl -k`.

Zusammenfassung

1. Aufbau und Konfiguration des Betriebssystems
   - Aufbau eines Betriebssystems
   - Verzeichnisstruktur: Filesystem Hierarchy Standard
   - Wichtige Verzeichnisse
   - Systemweite Konfiguration in `/etc/`
   - Virtuelle Dateisysteme
2. Systemressourcen und Systeminformationen
   - Systemressourcen (interaktiv)
     - Prozesse anzeigen mit `ps` und `pstree`
     - Speicherverbrauch anzeigen mit `free`
     - Systemressourcen interaktiv verfolgen mit `top`
   - Systeminformationen (protokolliert)
     - Log-Dateien in `/var/log/`
     - Logs mit `systemd-journal`
     - Logs betrachten mit `journalctl`
3. Aufgaben