Monitoring von Linux-basierten Systemen mit Bordmitteln

(Stichwort: „Wartungsfreier Betrieb“)

Festspeicherplatz-Verfügbarkeit

`df` („Disk Filling“)

```
Filesystem           1024-blocks    Used Available Capacity Mounted on
/dev/root              51198432  50077208   1121224  98% /
/dev/mapper/crypt      45590156  40025472   5564684  88% /crypt
/dev/mmcblk0p1         30542348     44992  30480972   0% /media/mmcblk0p1
```

→ Warnung generieren, wenn der Platz (Capacity) knapp wird, oder Temporärdateien löschen.

RAM-Auslastung

`cat /proc/meminfo`

`free`

```
             total       used       free     shared    buffers     cached
Mem:       7966620    7857016     109604     885808      76472    1819548
-/+ buffers/cache:    5960996    2005624
Swap:     12385272     419328   11965944
```

→ Warnung / Aktion erforderlich, wenn der noch verfügbare Hauptspeicher ABZÜGLICH CACHE/BUFFERS verschwindet. (Gelb markiert, hier stehen spontan noch 2GB für Applikationen zur Verfügung).

Kontinuierliches Monitoring: `vmstat`

CPU-Auslastung

„System-Auslastung“: `uptime`

```
10:08:56 up 10 days, 12:52,  1 user,  load average: 20, 0,34, 0,37
```

Grün markiert: Gemittelte Auslastung in den 1 / 5 / 15 Minuten

In diesem Beispiel hat das System aktuell eine ÜBERLAST von Faktor 20.

`ps auxwww (nicht-interaktiv, leicht weiterverarbeitbar)`

`(h)top (interaktiv)`

→ Prozesse identifizieren, die (zu) viel Speicher/CPU/IO-Last verursachen.

`Abhilfe gegen Prozesse, die das System „lahmlegen“, aber die laufen MÜSSEN:`

`nice -10 ionice -c 3 programmname …`

`→ Setzt die CPU-Priorität HERUNTER, und sorgt dafür, dass nur auf die Festplatte / SSD zugegriffen werden kann, wenn diese im „Idle“ Zustand ist.`

Akkustatus / Temperatursensoren

Intel/AMD: Über ACPI-BIOS

`acpi -V`

```
Battery 0: Full, 100%
Battery 0: design capacity 1914 mAh, last full capacity 1914 mAh = 100%
Adapter 0: on-line

→ Der Akku ist zwar voll, hat aber nur noch eine Kapazität von knapp 2000
Milliamperestunden! (ca. 1 Stunde Laufzeit)

Thermal 0: ok, 28.7 degrees C
Thermal 0: trip point 0 switches to mode critical at temperature 128.0 degrees C
Thermal 0: trip point 1 switches to mode passive at temperature 55.0 degrees C
Thermal 1: ok, 39.0 degrees C
Thermal 1: trip point 0 switches to mode critical at temperature 128.0 degrees C
Thermal 2: ok, 59.0 degrees C
Thermal 2: trip point 0 switches to mode critical at temperature 128.0 degrees C
Cooling 0: Processor 0 of 10
Cooling 1: Processor 0 of 10
Cooling 2: LCD 0 of 24

→ Bei 128°C würde sich der Rechner abschalten, ab 55°C wird die CPU
heruntergetaktet („passive Kühlung“), der Lüfter schaltet sich in diesem
Beispiel anscheinend automatisch temperaturgeregelt ein.

Aktuell wird keine Kühlungsmaßnahme aktiv.
Über Dateien in /proc/acpi können die Temperaturregelungen verändert werden.
```

Logdateien / Systemstatusprotokoll / Andere Fehlerquellen

Vor systemd: Textdateien mit dem Systemprotokoll sind unter /var/log im Klartext lesbar. Der syslogd legt diese an und schreibt aktuelle Systemmeldungen von Kernel und Programmen hinein.

→ Permanent gespeichert

Neu mit systemd: Die Logdateien sind in einem „leicht mit Programmen durchsuchbaren“ Binärformat gespeichert, und können mit dem Programm „journalctl“ ausgewertet werden.

→ Auch permanent

Gefahr: System-Meldungen können die Logfiles anwachsen lassen

Tipp: Mit „logrotate“ lassen sich Logdateien in der Größe, Kompression, Anzahl von Backups oder Anzahl von Meldungen begrenzen.

Kernel-Meldungen in Echtzeit:

cat /proc/kmsg

(läuft kontinuierlich weiter bis ^C)

Aktueller Stand:

dmesg

(geht auch ohne system-Log-Daemon).

Z.B. die letzten Meldungen der WLAN-Karte anzeigen lassen:

`dmesg | grep wlan`

Automatischer Check zu bestimmten Zeiten (crontab)

Stichwort: **crontab**

Eine **crontab** ist eine Tabelle (Programm zur Verwaltung heißt auch so), die angibt, welche Dienste zu welchen Zeiten oder Intervallen laufen sollen. Details: `man 5 crontab`

```
# run five minutes after midnight, every day
5 0 * * *       $HOME/bin/daily.job >> $HOME/tmp/out 2>&1
# run at 2:15pm on the first of every month -- output mailed to paul
15 14 1 * *     $HOME/bin/monthly
# run at 10 pm on weekdays, annoy Joe
0 22 * * 1-5    mail -s "It's 10pm" joe%Joe,%%Where are your kids?%
23 0-23/2 * * * echo "run 23 minutes after midn, 2am, 4am ..., everyday"
5 4 * * sun     echo "run at 5 after 4 every sunday"
# Run on every second Saturday of the month
0 4 8-14 * *    test $(date +\%u) -eq 6 && echo "2nd Saturday"
```

Einfacher: Für Aufgaben, die „täglich“ der „wöchentlich“ oder „stündlich“ laufen sollen, sind Ordner vorhanden: `/etc/cron.daily /etc/cron.weekly /etc/cron.hourly` …

Unter `/var/spool/cron/crontabs/*` liegen die Benutzer-eigenen crontabs. Die Haupt-crontab ist unter `/etc/crontab` untergebracht.