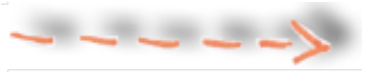
Zeitgesteuerte Vorgänge

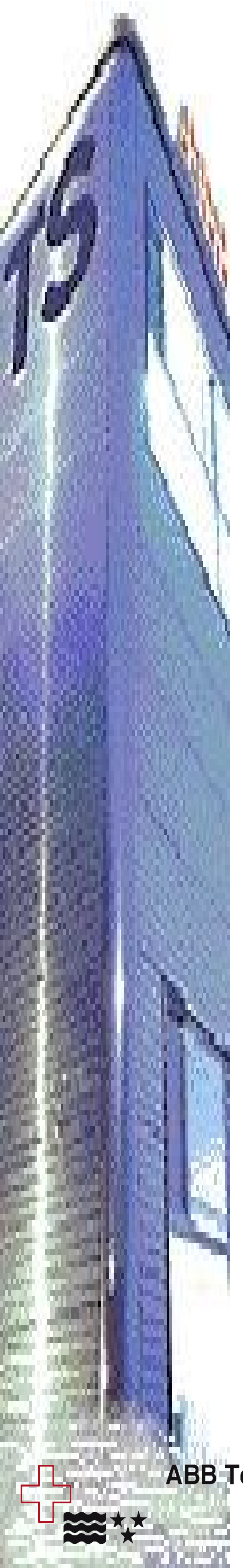
Lernziele



- Kommandos mit *at* zu einem zukünftigen Zeitpunkt ausführen können
- Kommandos periodisch mit *cron* ausführen können

Allgemeines

- Unter UNIX/Linux gibt es immer wieder Vorgänge, die automatisiert werden können
  - Zu einer bestimmten Zeit
  - Wenn die Systemlast tief ist
  - Wiederholt ablaufende Vorgänge
- Dazu dienen die Kommandos
  - *at*, *batch*
  - *cron*



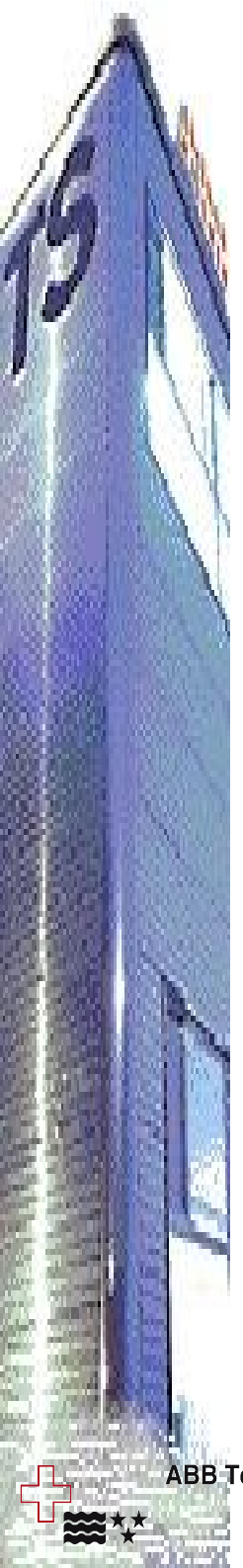
Einmalige Ausführung von Kommandos

- Mit dem Kommando *at* lassen sich Shell-Befehle zu einem bestimmten Zeitpunkt ausführen
- Ein Kommando wird also zeitversetzt, einmalig ausgeführt
  - Z.B. ein Backup in der Nacht
  - Ein- und ausschalten einer Abwesenheits-meldung im Mailsystem
  - Eine Erinnerungsmail zu einer bestimmten Zeit versenden

at und *batch*

- *at* kann interaktiv mit einer Folge von Kommandos ausgeführt werden
  - Zum Beispiel ein Backup Nachts um 1 Uhr:

```
$ at 01:00
warning: commands will be executed using /bin/sh
at> tar cvzf /dev/st0 $HOME
at> echo "Backup fertig" | mail -s Backup $USER
at> Strg + D
Job 123 at 2003-11-08 01:00
```



at und *batch*

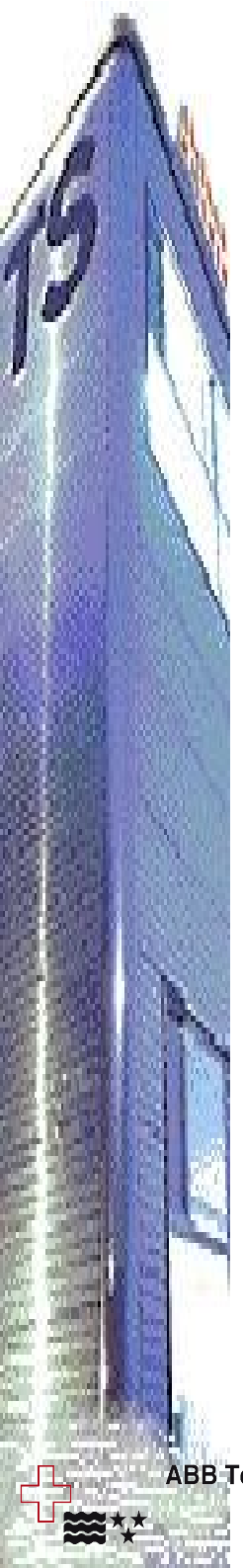
- Das Argument von *at* ist eine Zeitangabe
- Zeiten im Format »HH:MM« bezeichnen den nächstmöglichen solchen Zeitpunkt
- Durch Anhängen von *today* und *tomorrow* wird der Zeitpunkt eindeutig
- Angelsächsische Zeitangaben wie *01:00am* oder *14:20pm* sind auch möglich
- Symbolischen Namen *midnight* (0 bzw. 24 Uhr), *noon* (12 Uhr) und *teatime* (16 Uhr)

at und *batch*

- Datumsangaben in der Form:
  - amerikanisch MMTTJJ oder MM/TT/JJ
  - europäisch TT.MM.JJ
- Ausgeschriebene Datumsangaben wie »Monatsname Tag Jahr« und »Monatsname Tag« sind auch erlaubt
- Bei Datumsangaben werden die Kommandos zur aktuellen Zeit ausgeführt
- Zeit- und Datumsangaben kann man kombinieren

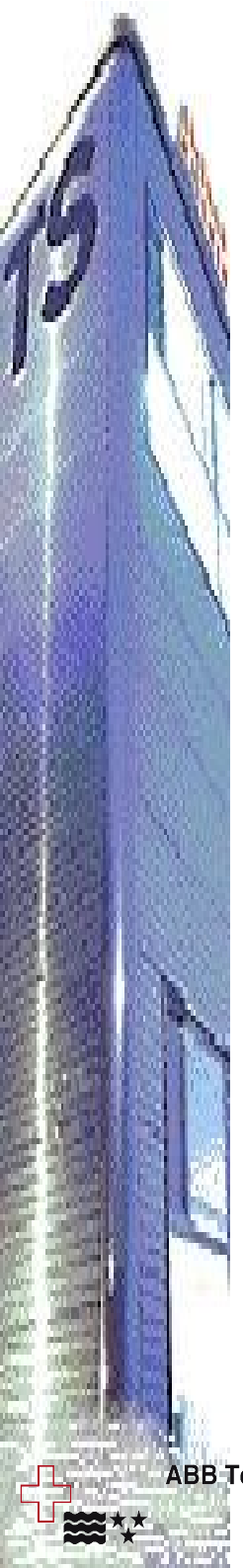
at und *batch*

- Ausser »expliziten« Zeit- und Datumsangaben können auch relative Angaben einen Zeitpunkt bestimmen
  - at now + 5 minutes
  - at noon + 2 days
  - Nicht erlaubt sind Kombinationen wie
    - at noon + 2 hours + 30 minutes
- *at* liest die Kommandos von der Standardeingabe oder von einer Datei (Option -f)



at und *batch*

- *at* führt die Kommandos in einer dem Benutzer ähnlichen Umgebung aus
- Arbeitsverzeichnis, die umask und einige Umgebungsvariablen werden verwendet
- Ausgaben der per *at* gestarteten Kommandos werden dem Benutzer per Mail zugestellt
- Es gibt keine Rückmeldung, dass der Job gelaufen ist



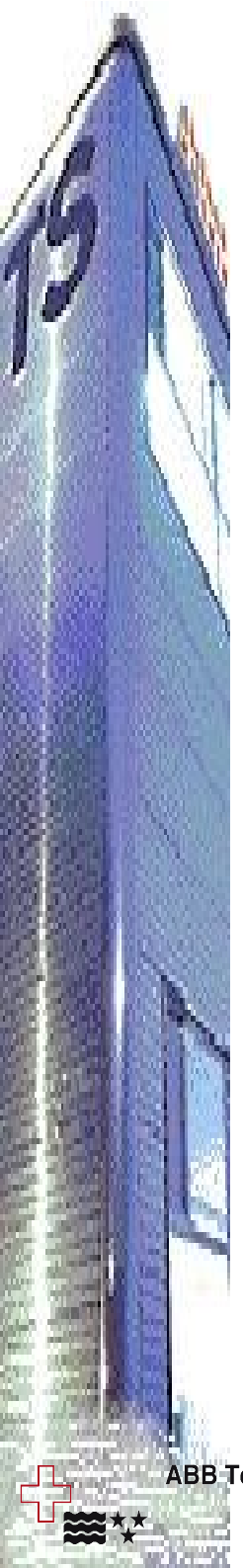
at und *batch*

- *at* führt Kommandos zu einem bestimmten Zeitpunkt aus
- Der Befehl *batch* kann eine Folge von Kommandos zum »nächstmöglichen Zeitpunkt« ausführen
- Wann das passiert, hängt von der aktuellen Systemlast ab
- Eine Zeitangabe wie bei *at* ist auch bei *batch* möglich

Fragen?

Übungen

- [10.1] Nehmen wir an, jetzt ist es der 1. März, 15 Uhr.
- Wann werden die mit den folgenden Kommandos eingereichten Aufträge ausgeführt?
  1. `at 17:00`
  2. `at 02:00pm`
  3. `at teatime tomorrow`
  4. `at now + 10 hours`



Übungen

- [10.2] Verwenden Sie das Programm logger, um in 3 Minuten eine Nachricht ins Systemprotokoll zu schreiben.

at-Hilfsprogramme

- Die mit *at* registrierten Aufträge kommen in eine Warteschlange
- Diese kann mit *atq* angezeigt werden
  - `atq`

```
123 2003-11-08 01:00 a hugo
124 2003-11-11 11:11 a hugo
125 2003-11-08 21:05 a hugo
```

  - Das »a« in der Liste bezeichnet die Auftragsklasse
    - »a« für at- und »b« für batch-Aufträge

Der *at*-Daemon

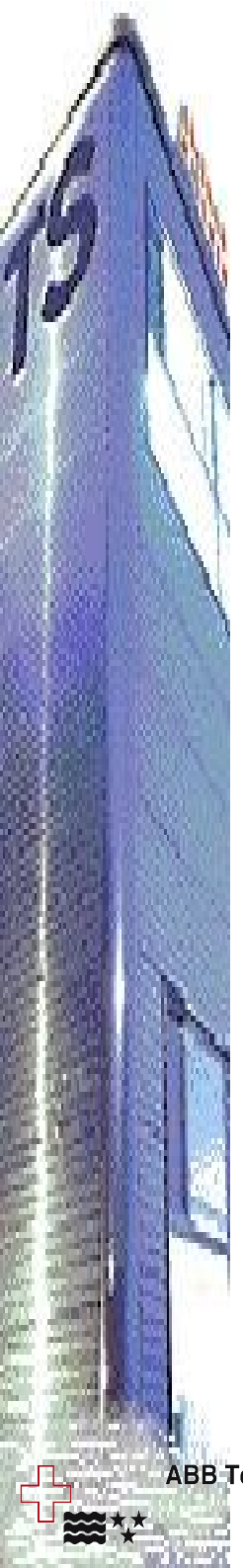
- Mit *atrm* kann ein registrierter Auftrag aus der Warteschlange entfernt werden
  - `atrm <job-id>`
- Die Job-Id kann mit *atq* oder *at -l* angezeigt werden
- Den Job selbst kann man mit *at -c <job-id>* auf die Konsole ausgeben

Der *at*-Daemon

- Zuständig für die Ausführung der at-Aufträge ist ein Daemon namens *atd*
- Beim Start von *atd* sind Optionen möglich
  - `-b` (engl. batch) Legt das Intervall von batch-Aufträgen fest (Default 60 Sek.)
  - `-l` (engl. load) Legt einen Grenzwert fest, wann batch-Aufträge nicht mehr ausgeführt werden (Default Wert 0,8)

Der *at*-Daemon

- Der Daemon *atd* benötigt zu seiner Arbeit die folgenden Verzeichnisse:
  - Im Verzeichnis /var/spool/atjobs werden die at-Aufträge abgelegt.
    - Der Zugriffsmodus sollte 700 sein
    - Der Eigentümer ist *at*
  - Das Verzeichnis /var/spool/atspool dient zur Zwischenspeicherung von Ausgaben
    - Auch hier sollten der Eigentümer *at*
    - und der Zugriffsmodus 700 sein



Übungen

- [10.3] Registrieren Sie mit *at* einige Aufträge und lassen Sie sich eine Auftragsliste anzeigen.
- Entfernen Sie die Aufträge wieder.

Übungen

- [10.4] Wie würden Sie eine Liste von at-Aufträgen erzeugen, die nicht nach der Auftragsnummer, sondern nach dem vorgesehenen Ausführungszeitpunkt sortiert ist?

Zugangskontrolle

- Wer mit *at* und *batch* Aufträge einreichen darf, wird durch zwei Dateien bestimmt
  - /etc/at.allow — Nur die Benutzer, die in der Datei stehen, sind berechtigt Jobs zu starten
  - /etc/at.deny — Benutzer, die nicht in dieser Datei stehen, dürfen Jobs starten
  - Existiert keine Datei, dann kann nur root *at* und *batch* Jobs ausführen

Übungen

- [10.5] Wer darf auf Ihrem System *at* und *batch* benutzen?

Wiederholte Ausführung von Kommandos

- Der Daemon *cron* dient zur automatischen Ausführung von sich regelmässig wiederholenden Aufgaben
- Wie auch der *atd* sollte der *cron* beim Booten von einem Init-Skript gestartet werden
- Neben den systemweiten Aufgabenlisten gibt es auch Aufgabenlisten für Benutzer

Aufgabenlisten für Benutzer

- Jeder Benutzer hat seine eigene Aufgabenliste (crontab)
- Diese ist im Verzeichnis */var/cron/tabs* bzw. */var/spool/cron/tabs* unter dem jeweiligen Benutzernamen abgelegt
- Die Kommandos in den Corntabs werden mit den Rechten des betreffenden Benutzers ausgeführt

crontab-Syntax

- Crontab-Dateien sind zeilenweise aufgebaut
- jede Zeile beschreibt einen (wieder-kehrenden) Zeitpunkt und ein Kommando
- Leerzeilen und Kommentarzeilen (mit einem »#« am Anfang) werden ignoriert
- Die Zeilen bestehen aus fünf Zeitfeldern und dem auszuführenden Kommando

crontab-Syntax

- Die fünf Zeitfelder beschreiben
  - die Minute (0–59)
  - die Stunde (0–23)
  - der Tag im Monat (1–31)
  - der Monat (1–12)
  - den Wochentag (0–7)
    - 0 und 7 stehen für Sonntag
    - Für den Monat und den Wochentag kann auch der englische Name eingesetzt werden

crontab-Syntax

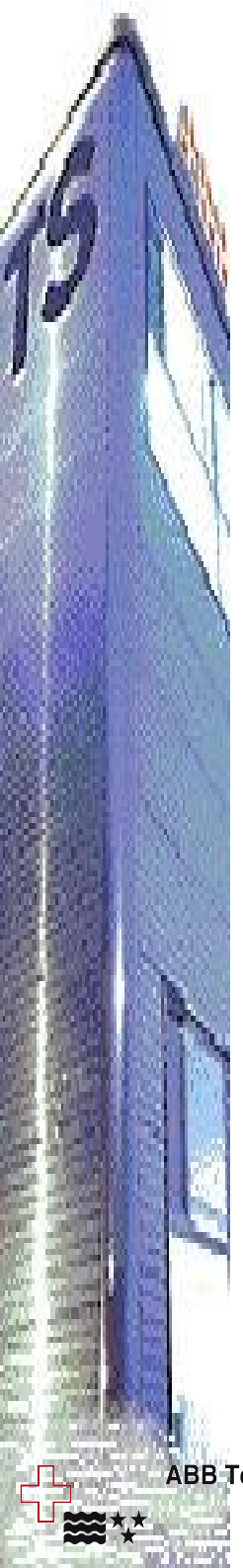
- Jedes der fünf Zeitfelder kann auch einen Stern »*« für eine beliebige Zeit enthalten
  - Zum Beispiel:

```
58 19 * * * echo "Gleich ist es 20 Uhr"
```

    - das Kommando wird täglich um 19.58 Uhr ausgeführt (Tag, Monat und Wochentag sind egal)

```
1 0 13 * 5 echo "Kurz nach Mitternacht"
```

    - Die Meldung wird an jedem Monats-Dreizehnten und an jedem Freitag ausgegeben, nicht nur am Freitag, dem 13.



crontab-Syntax

- *cron* akzeptiert in den Zeitfeldern auch kommaseparierte Listen
  - Die Angabe »0,30« im Minutenfeld bedeutet, dass das Kommando zu jeder vollen und halben Stunde ausgeführt wird
- Es sind auch Bereiche erlaubt
  - »8-11« ist äquivalent zu »8,9,10,11«
  - »8-10,14-16« entspricht »8,9,10,14,15,16«

crontab-Syntax

- *cron* akzeptiert in den Zeitfeldern auch die Angabe einer Schrittweite
  - Die Angabe »0-59/10« im Minutenfeld bedeutet, dass das Kommando alle 10 Minuten ausgeführt wird
  - Wird der komplette Wertebereich abgedeckt, so kann man auch »*/10« schreiben
- Die bei Monats- und Wochentags-angaben erlaubten englischen Namen dürfen nicht in Listen verwendet werden

crontab-Syntax

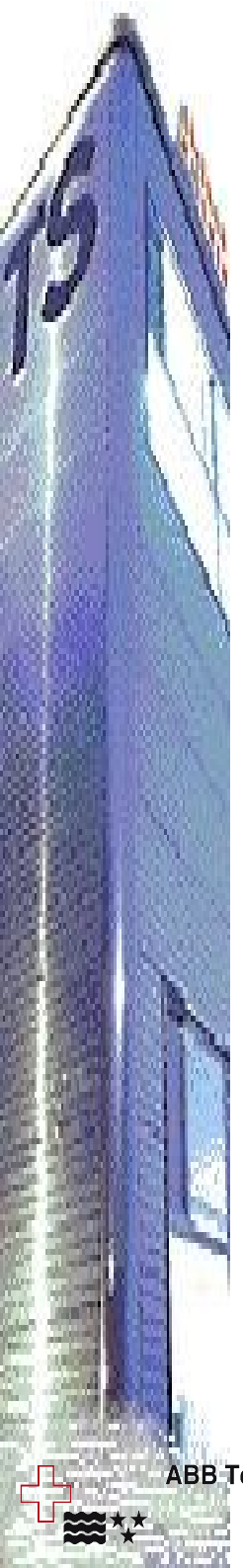
- *cron* verwendet standardmässig eine Born-Shell (`/bin/sh`)
- Prozentzeichen (%) in einem Kommando müssen markiert werden (\%) sonst gelten sie als Zeilentrenner
- Ein »-« als erstes Zeichen der Zeile unterdrückt die Meldung an Syslogd
- Variablen können in der Crontab-Datei definiert und in den Kommandos verwendet werden

crontab-Syntax

- SHELL
  - Mit dieser Shell werden die eingegebenen Befehle ausgeführt (Default */bin/sh*)
- LOGNAME
  - Der Benutzername wird aus /etc/passwd übernommen und kann nicht geändert werden
- HOME
  - Das Heimatverzeichnis wird ebenfalls aus /etc/passwd übernommen
  - Es kann geändert werden

crontab-Syntax

- Ausgaben werden per Mail an den Besitzer der Crontab-Datei gemeldet
- MAILTO
  - An diese Adresse schickt *cron* Nachrichten mit der Ausgabe des aufgerufenen Kommandos
- Soll *cron* keine Nachrichten verschicken, muss die Variable leer gesetzt sein
  - MAILTO=""



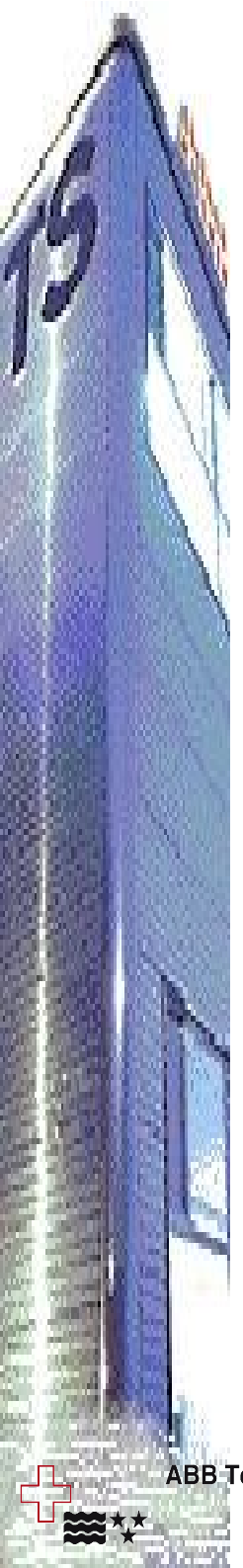
Systemweite Aufgabenlisten

- Die systemweite Aufgabenliste befindet sich in */etc/crontab*
- Sie gehört dem Systemadministrator *root*
- Die Syntax unterscheidet sich geringfügig von der benutzereigenen crontab
- Vor dem auszuführenden Kommando kann noch der Name eines Benutzers angegeben werden, mit dessen Rechten das Kommando ausgeführt wird

Systemweite Aufgabenlisten

- Diverse Linux-Distributionen haben ein */etc/cron.d*-Verzeichnis
- Auch Verzeichnisse wie
  - /etc/cron.hourly
  - /etc/cron.daily
  - /etc/cron.weekly
  - /etc/cron.monthly
- Diese werden als »Erweiterung« von */etc/crontab* interpretiert

Fragen?



Übungen

- [10.6] Wie können Sie dafür sorgen, dass eine Aufgabe nur am Freitag, dem 13. ausgeführt wird?

Übungen

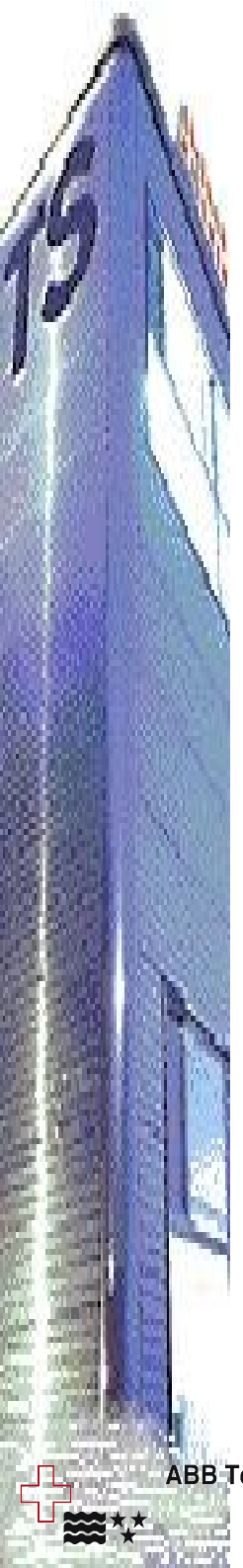
- [10.7] Wie stellt das System sicher, dass die Aufgaben in /etc/cron.hourly, /etc/cron.daily, . . . tatsächlich einmal in der Stunde, einmal am Tag usw. ausgeführt werden?

Zugangskontrolle

- Welche Anwender mit *cron* arbeiten dürfen, ist ähnlich wie bei *at* in zwei Dateien festgelegt
- /etc/cron.allow
  - Führt Benutzer auf, die cron verwenden dürfen
- /etc/cron.deny
  - Alle, ausser diejenigen Benutzer in dieser Datei dürfen automatisiert Befehle ausführen

Das Kommando *crontab*

- Benutzer sollten ihre crontab-Datei nicht von Hand ändern
- Dazu gibt es das Kommando *crontab*
  - `crontab -e` bearbeitet die Crontab-Datei
  - `crontab -l` zeigt die Crontab auf der Konsole
  - `crontab -r` löscht eine Crontab-Datei
  - `crontab <Datei>` liest die Crontab aus Datei
- Der Administrator kann mit der Option `-u` Crontabs von Benutzern lesen



Übungen

- [10.8] Testen Sie cron und crontab, indem Sie eine Aufgabe registrieren, die zum Beispiel alle zwei Minuten eine Meldung ins Systemprotokoll schreibt.

Übungen

- [10.9] Sorgen Sie dafür, dass der Benutzer hugo den cron-Dienst nicht verwenden darf.
- Vergewissern Sie sich, dass Ihre Massnahme funktioniert.