Einführung RRDtool

Mario Haustein

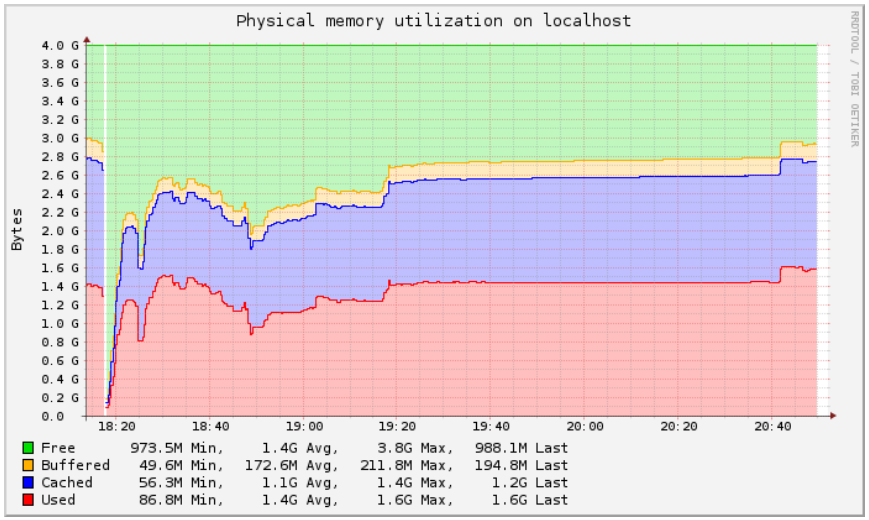
Chemnitzer Linux User Group

10. Oktober 2014

Sinn und Zweck von RRDTool

- RRDtool kann Messdaten ...
  - erfassen
  - archivieren
  - analysieren
  - visualisieren
- Typische Anwendungsfälle
  - Fehlerzustände in IT-Systemen erkennen.
  - Zukünftige Investitionen planen.
  - Komplizierte Sachverhältnisse verständlich machen.
  - Abrechnungen erstellen.

Typisches Erscheinungsbild



Trivia

- RRD $\mapsto$ **R**ound **R**obin **D**atabase
  - Art und Weise wie die Messdaten gespeichert werden
- Webseite: `http://oss.oetiker.ch/rrdtool/index.en.html`
- Autor: Tobias Oetiker
- Lizenz: GPL
- Version 1.0: 16. Juli 1999
- Aus dem Monitoring-System MRTG hervorgegangen

Archivierung von Messdaten

Zeitreihe

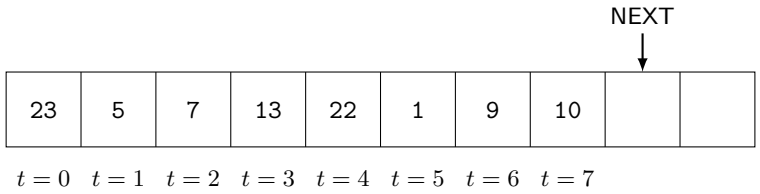
- Messungen einer Prozessgröße in **regelmäßigen** Abständen.

- Mehrere Archivierungsmodi für die Messgrößen definierbar, z.B.:
  - Minütliche Auflösung für die letzten 24 Stunden
  - 5-minütliche Auflösung für die letzten 7 Tage
  - Stündliche Auflösung für die letzten 31 Tage
  - Tägliche Auflösung für das letzte Jahr
- Ggf. werden mehrere Messungen zu einem Datenpunkt zusammengefasst.

- Mess- und Metadaten werden in einer RRD-Datei gespeichert.
  - ⇒ Durch Vergröberung der Auflösung sind lange Messreihen möglich.
  - ⇒ Größe der Datenbank zum Erstellungszeitpunkt bekannt.

Round Robin Datenbanken

- Effiziente Speicherung einer Zeitreihe
  - Zeitlich geordnete Liste von Messwerten.
  - Beim Einfügen wird der älteste Messwert überschrieben.
  - ⇒ Datenbank wächst nicht.

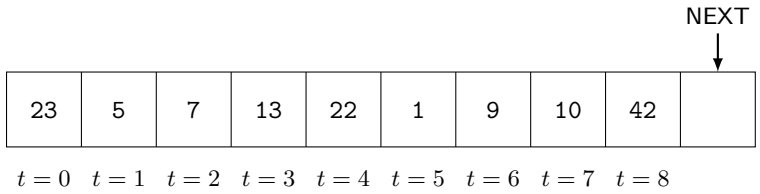


- Einfügen: $\mathcal{O}(1)$, Auslesen: $\mathcal{O}(1)$, Sortiert Auslesen: $\mathcal{O}(n)$

⇒ Besser geht es nicht!

Round Robin Datenbanken

- Effiziente Speicherung einer Zeitreihe
  - Zeitlich geordnete Liste von Messwerten.
  - Beim Einfügen wird der älteste Messwert überschrieben.
  - ⇒ Datenbank wächst nicht.

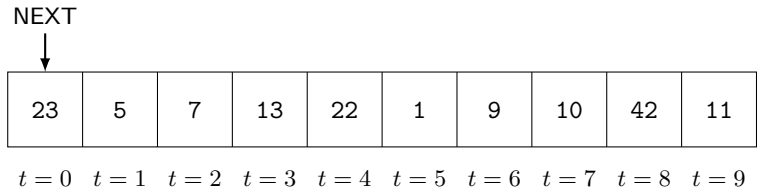


- Einfügen: $\mathcal{O}(1)$, Auslesen: $\mathcal{O}(1)$, Sortiert Auslesen: $\mathcal{O}(n)$

⇒ Besser geht es nicht!

Round Robin Datenbanken

- Effiziente Speicherung einer Zeitreihe
  - Zeitlich geordnete Liste von Messwerten.
  - Beim Einfügen wird der älteste Messwert überschrieben.
  - ⇒ Datenbank wächst nicht.



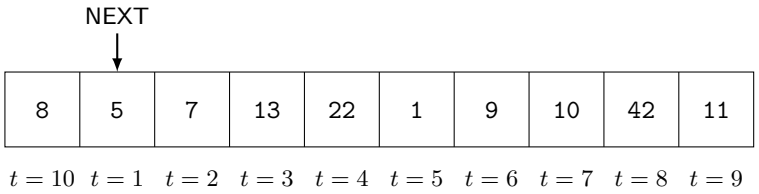
- Einfügen: $\mathcal{O}(1)$, Auslesen: $\mathcal{O}(1)$, Sortiert Auslesen: $\mathcal{O}(n)$

⇒ Besser geht es nicht!

Round Robin Datenbanken

- Effiziente Speicherung einer Zeitreihe
  - Zeitlich geordnete Liste von Messwerten.
  - Beim Einfügen wird der älteste Messwert überschrieben.

  $\Rightarrow$ Datenbank wächst nicht.



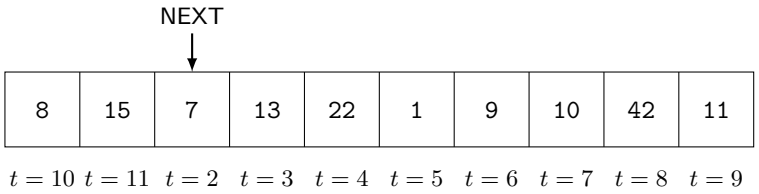
- Einfügen: $\mathcal{O}(1)$, Auslesen: $\mathcal{O}(1)$, Sortiert Auslesen: $\mathcal{O}(n)$

$\Rightarrow$ Besser geht es nicht!

Round Robin Datenbanken

- Effiziente Speicherung einer Zeitreihe
  - Zeitlich geordnete Liste von Messwerten.
  - Beim Einfügen wird der älteste Messwert überschrieben.

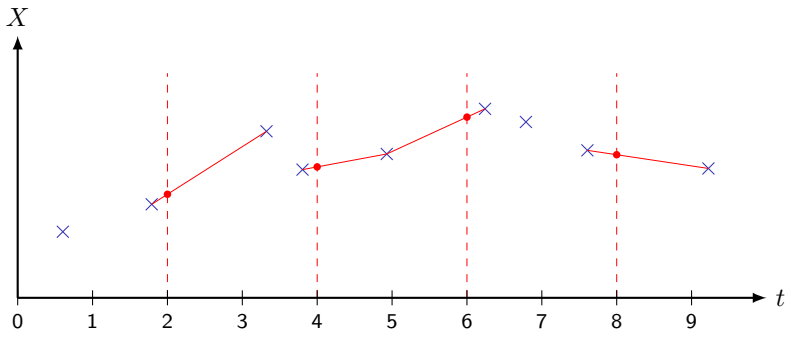
  $\Rightarrow$ Datenbank wächst nicht.



- Einfügen: $\mathcal{O}(1)$, Auslesen: $\mathcal{O}(1)$, Sortiert Auslesen: $\mathcal{O}(n)$

$\Rightarrow$ Besser geht es nicht!

Zeitreihen



- Messperiode wird bei Erzeugung des RRD festgelegt (hier: 2 Zeiteinheiten).
- Messungen ($\times$) erfolgen in der Praxis nie genau zu den Intervallgrenzen.

$\Rightarrow$ RRDtool interpoliert die Werte zur Zeit des Intervallwechsels ($\bullet$).
  - **Primary Data Point (PDP)**

Eine RRD anlegen

- Erforderliche Metadaten

  `--step` Messperiode

  `DS` Datenquellen (**D**ata **S**ource)

  `RRA` Archivierungsregeln (**R**ound **R**obin **A**rchive)
- Optional: Zeitpunkt zu dem die Aufzeichnung beginnt.

```
$ rrdtool create datenbank.rrd --step 60 \
  DS:temp_hdd:GAUGE:120:10:U \ # Datenquelle HDD-Temperatur
  DS:temp_cpu:GAUGE:120:10:U \ # Datenquelle CPU-Temperatur
  RRA:AVERAGE:0.5:1:300      \ # Minütliche Messungen für 5h
  RRA:AVERAGE:0.5:1440:365   \ # Tagesmittel für ein Jahr
  RRA:MAX:0.5:1440:365       \ # Tagesmax. für ein Jahr
  RRA:MIN:0.5:1440:365       \ # Tagesmin. für ein Jahr
```

Details: Datenquellen

- `DS:`⟨*Name*⟩`:`⟨*Typ*⟩`:`⟨*Heartbeat*⟩`:`⟨*Min*⟩`:`⟨*Max*⟩
- Bsp.: `DS:temp_hdd:GAUGE:120:10:U`
  - `temp_hdd` Beliebiger Bezeichner
  - `GAUGE` Der Messwert wird so wie eingegeben gespeichert.
  - `120` Liegen zwei Messungen weiter als 120 s auseinander, wird das Intervall als ungültige betrachtet.
  - `10` Temperaturen unter $10\,^{\circ}\mathrm{C}$ sind ungültig.
  - `U` Keine Obergrenze für die Temp. festgelegt.
- Typen
  - `GAUGE` Zeitlose Messgröße (der Messwert wird nicht differenziert).
  - `DERIVE` Die Änderungsrate bzgl. der letzten Messung wird gespeichert.
  - `COUNTER` Wie `DERIVE` nur mit Berücksichtigung von 32-Bit bzw. 64-Bit-Überläufen.
  - `ABSOULTE` Messwert wird durch die seit der letzten Messung vergangene Zeit dividiert.
  - `COMPUTE` Berechnung auf Basis anderen Datenquellen (siehe Manpage).

Details: Archivierungsregeln

- Ein RRA fasst (mehrere) primäre Datenpunkte (PDPs) zusammen.
- Diese konsolidierte Datenpunkte (CDPs) werden gespeichert.
- `RRA:`⟨*Funktion*⟩`:`⟨*Quote*⟩`:`⟨*PDPs*⟩`:`⟨*CDPs*⟩
  - ⟨*CDPs*⟩ Anzahl gespeicherter CDPs in diesem Archiv
  - ⟨*PDPs*⟩ Anzahl PDPs, die zu einem CDP zusammenfasst werden
  - ⟨*Quote*⟩ Anteil der PDPs, die höchstens ungültig sein können, damit der CDP noch gültig ist.
  - ⟨*Funktion*⟩ Art auf die sich der CDP aus den PDPs berechnet. `AVERAGE`, `MIN`, `MAX` oder `LAST`
- Bsp.: `RRA:AVERAGE:0.5:1:300`

Daten in eine RRD eintragen

```
$ rrdtool update datenbank.rrd \
  -t temp_hdd:temp_cpu -- N:30:50.6
```

- `-t` Reihenfolge in der die Messungen der Datenquellen zugeordnet werden.
- `N:` Zeitpunkt der Messung (`N` ↦ „jetzt“)
- `30:50.6` HDD-Temp. → $30\,^{\circ}\mathrm{C}$, CPU-Temp. → $50{,}6\,^{\circ}\mathrm{C}$

- Wird statt `N` ein Zeitstempel angegeben, muss dieser größer als der Zeitstempel der letzten Messung sein.

Beispiel: Erfassung der Systemlast

```
load=$(cat "/proc/loadavg")
load1=$(echo "$load" | cut -d ' ' -f 1)
load5=$(echo "$load" | cut -d ' ' -f 2)
load15=$(echo "$load" | cut -d ' ' -f 3)

rrdtool update load.rrd -t 1min:5min:15min -- \
  N:$load1:$load5:$load15
```

Eine RRD visualisieren

- Vielseitige Darstellungsoptionen
- Hier nur grundlegende Darstellungselemente gezeigt
- I.d.R. benutzt man ein Framework und nicht direkt RRDtool.
- Weitere Informationen
  - `man rrdgraph`
  - `man rrdgraph_data`
  - `man rrdgraph_graph`
  - `man rrdgraph_examples`

```
$ rrdtool graph output.png          \ # Ausgabedatei
  -w 640 -h 480 -t Systemlast       \ # Optionen
  --end now --start end-1h          \
  ⟨Datenquellen laden⟩
  ⟨Variablen berechnen⟩
  ⟨Grafikelemente zeichnen⟩
```

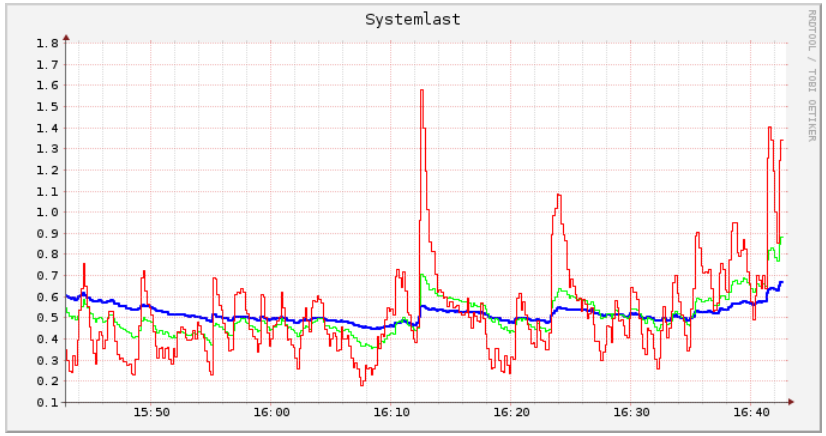

Beispiel: Systemlast

ohne Legende

```
$ rrdtool graph output_load.png     \
  -w 600 -h 300 -t Systemlast       \
  --end now --start end-1h          \
  DEF:1min=load.rrd:1min:AVERAGE    \
  DEF:5min=load.rrd:5min:AVERAGE    \
  DEF:15min=load.rrd:15min:AVERAGE  \
  "LINE2:15min#0000ff"              \
  "LINE:5min#00ff00"                \
  "LINE:1min#ff0000"
```


Beispiel: Systemlast

ohne Legende



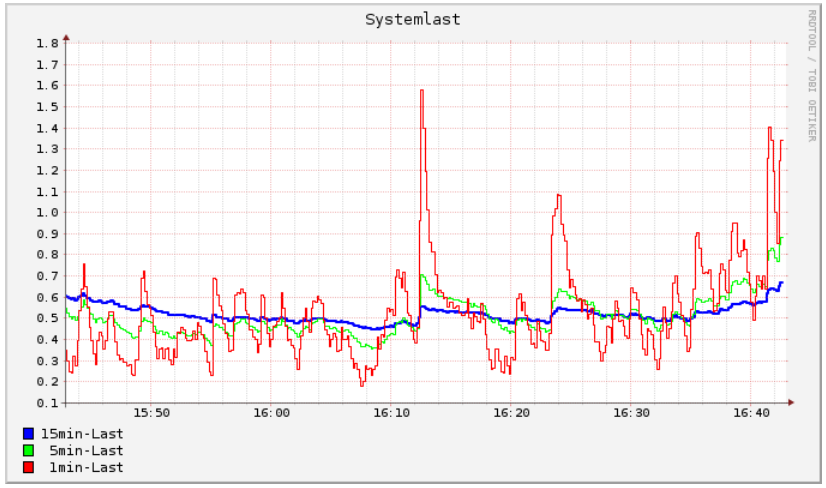
Beispiel: Systemlast

mit Legende

```
$ rrdtool graph output_load.png     \
  -w 600 -h 300 -t Systemlast       \
  --end now --start end-1h          \
  DEF:1min=load.rrd:1min:AVERAGE    \
  DEF:5min=load.rrd:5min:AVERAGE    \
  DEF:15min=load.rrd:15min:AVERAGE  \
  "LINE2:15min#0000ff:15min-Last\n" \
  "LINE:5min#00ff00:␣5min-Last\n"   \
  "LINE:1min#ff0000:␣1min-Last\n"
```

Beispiel: Systemlast

mit Legende



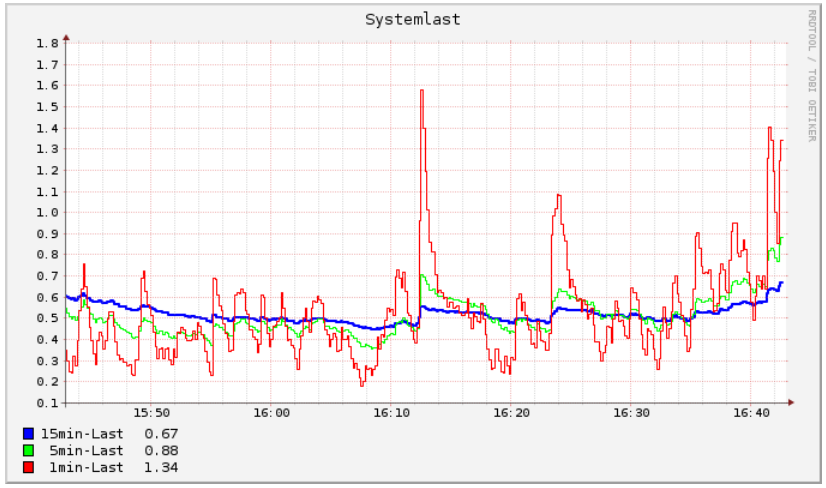
Beispiel: Systemlast

mit Legende und aktuellen Werten

```
$ rrdtool graph output_load.png      \
  -w 600 -h 300 -t Systemlast        \
  --end now --start end-1h           \
  DEF:1min=load.rrd:1min:AVERAGE     \
  DEF:5min=load.rrd:5min:AVERAGE     \
  DEF:15min=load.rrd:15min:AVERAGE   \
  VDEF:1min_value=1min,LAST          \
  VDEF:5min_value=5min,LAST          \
  VDEF:15min_value=15min,LAST        \
  "LINE2:15min#0000ff:15min-Last"    \
  "GPRINT:15min_value:%4.2lf\n"      \
  "LINE:5min#00ff00:␣5min-Last"      \
  "GPRINT:5min_value:%4.2lf\n"       \
  "LINE:1min#ff0000:␣1min-Last"      \
  "GPRINT:1min_value:%4.2lf\n"
```


Beispiel: Systemlast

mit Legende und aktuellen Werten



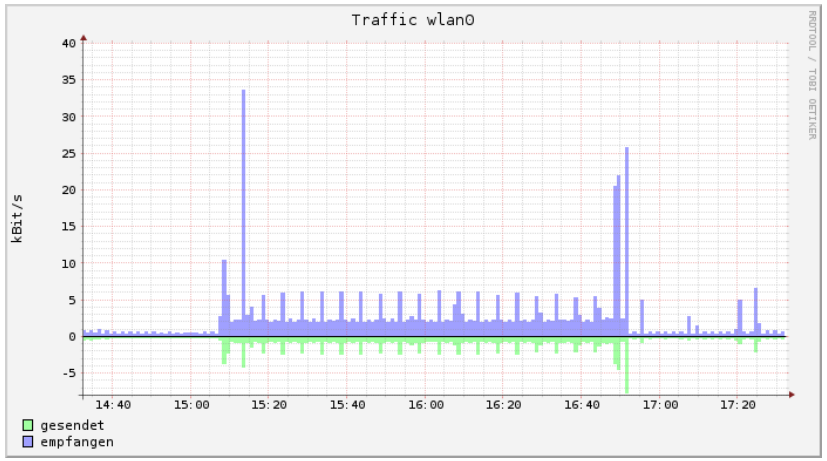
Beispiel: Netzwerktraffic

Ein- und Ausgehend

```
$ rrdtool graph output_wlan0_lin.png \
  -w 600 -h 300 -t "Traffic␣wlan0" \
  --end now --start end-3h         \
  -v "kBit/s"                      \
  DEF:tx=wlan0.rrd:bytestx:AVERAGE \
  DEF:rx=wlan0.rrd:bytesrx:AVERAGE \
  CDEF:txkb=tx,-1000,/             \
  CDEF:rxkb=rx,1000,/              \
  "AREA:txkb#a0ffa0:gesendet\n"    \
  "AREA:rxkb#a0a0ff:empfangen\n"   \
  "HRULE:0#000000"
```

Beispiel: Netzwerktraffic

Ein- und Ausgehend



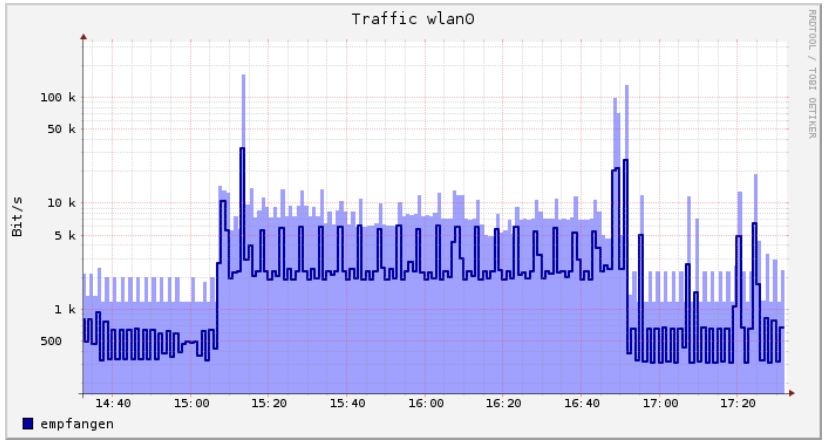
Beispiel: Netzwerktraffic

Durchschnitt und Maximum auf log. Skala

```
$ rrdtool graph output_wlan0_rx_log.png \
  -w 600 -h 300 -t "Traffic␣wlan0"    \
  --end now --start end-3h            \
  -v "Bit/s" -o --units=si            \
  DEF:rx=wlan0.rrd:bytesrx:AVERAGE    \
  DEF:rxmax=wlan0.rrd:bytesrx:MAX     \
  "AREA:rxmax#a0a0ff"                 \
  "LINE2:rx#0000a0:empfangen"         \
```


Beispiel: Netzwerktraffic

Durchschnitt und Maximum auf log. Skala



Weiterführendes

- Weitere Features
  - Vorhersage von Zeitreihen
  - Hervorhebung von Schwellwerten und Zeitmarken
  - Glättung von Messwerten
- Frontends und Frameworks[1]
  - drraw
  - DSreport
  - WeatherMap4RRD
  - PHP Network Weathermap
  - rrdUtils
  - BBStatus
  - Big Sister
  - Cacti
  - Cadmon
  - collectd
  - eLuna Graph System
  - Mailgraph
  - Monitorix
  - Munin
  - NAV
  - N2RRD Nagios Add-On
  - Observium
  - Systemgraph
  - Tstat
  - Zero RRD Framework
- Beispieldiagramme: `http://oss.oetiker.ch/rrdtool/gallery/`

[1] Auswahl aus `http://oss.oetiker.ch/rrdtool/rrdworld/`

Collectd

- https://collectd.org/
- Deamon & Framework zur Datenerfassung
  - Fragt regelmäßig (z.B. alle 10 s) alle Datenquellen ab.
- 93 Plugins zur Datenerfassung (Stand Oktober 2014)
  - Vom Betriebssystem bereitgestellte Leistungsdaten
  - Auswertung von Logs gängiger Server-Produkte
  - Anbindung eigener Skript möglich
- Überwachung mehrer Computer übers Netzwerk möglich

Collectd installieren

- Messwerte werden in RRD-Datenbanken gespeichert
  - /var/lib/collectd/rrd/
- Einfaches Setup
  - Installieren
  - /etc/collectd.conf bearbeiten
  - /etc/init.d/collectd start
  - Fertig.
- Frontends zur Datenvisualisierung müssen separat aufgesetzt werden.
  - https://collectd.org/wiki/index.php/List_of_front-ends

Collectd konfigurieren

```
1  BaseDir     "/var/lib/collectd"
2  PIDFile     "/var/run/collectd/collectd.pid"
3  Interval     10
4
5  LoadPlugin syslog
6  LoadPlugin cpu
7  LoadPlugin disk
8  LoadPlugin interface
9  # ...
10
11 <Plugin disk>
12   Disk "/^[hs]d[a-f][0-9]?$/"
13 </Plugin>
14
15 <Plugin interface>
16   Interface "eth0"
17   Interface "wlan0"
18   IgnoreSelected false
19 </Plugin>
20 # ...
```

- Dokumentation: man 5 collectd.conf

CGP: Ein Collectd-Frontend

- Webfrontend, PHP
- http://pommi.nethuis.nl/category/cgp/
- Auspacken
- /var/www/localhost/htdocs/cgp/conf/collectd.local.php

```
1 <?php
2 $CONFIG['datadir'] = '/var/lib/collectd/rrd';
3 $CONFIG['rrdtool'] = '/usr/bin/rrdtool';
4 $CONFIG['width'] = 400;
5 $CONFIG['heigth'] = 175;
6 $CONFIG['detail-width'] = 800;
7 $CONFIG['detail-heigth'] = 350;
8 ?>
```

- Fertig