

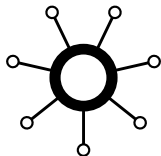
Performance-Analyse mit **collectd**

Wisse, was deine Rechner tun

Sebastian „tokkee“ Harl
<tokkee@collectd.org>

collectd core team

FrOSCamp 2010
18. September 2010



Was ist collectd?

Wichtige Eigenschaften

Wichtige Plugins

Über den Tellerrand

Optional



Was ist **collectd**?

- ▶ **collectd** sammelt Leistungsdaten von Rechnern
- ▶ Leistungsdaten sind zum Beispiel:
 - ▶ CPU-Auslastung
 - ▶ Speichernutzung
 - ▶ Netzwerkverkehr
- ▶ Daten werden erhoben, verarbeitet und gespeichert
- ▶ Häufig: Darstellung als Graphen
- ▶ → Performance-Analyse, Kapazitätsplanung
- ▶ Nicht verwechseln mit *Monitoring!*
- ▶ Homepage: <http://collectd.org/>



Wichtige Eigenschaften

- ▶ Daemon
- ▶ Freie Software (größtenteils GPLv2)
- ▶ Portierbar (Linux, *BSD, Solaris, ...)
- ▶ Skalierbar (OpenWrt, ..., Cluster / Cloud)
- ▶ Effizient (Default-Auflösung: 10 Sekunden)
- ▶ Modular (über 90 Plugins in Version 4.10)

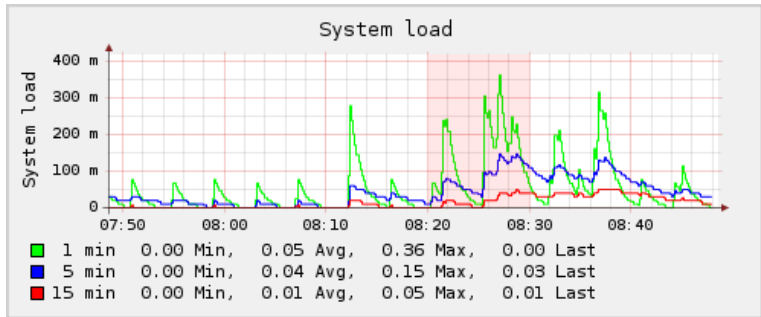


Wichtige Eigenschaften

- ▶ Daemon
- ▶ Freie Software (größtenteils GPLv2)
- ▶ Portierbar (Linux, *BSD, Solaris, ...)
- ▶ Skalierbar (OpenWrt, ..., Cluster / Cloud)
- ▶ **Effizient** (Default-Auflösung: 10 Sekunden)
- ▶ Modular (über 90 Plugins in Version 4.10)



Wichtige Eigenschaften: 10-Sekunden-Auflösung



Wichtige Eigenschaften

- ▶ Daemon
- ▶ Freie Software (größtenteils GPLv2)
- ▶ Portierbar (Linux, *BSD, Solaris, ...)
- ▶ Skalierbar (OpenWrt, ..., Cluster / Cloud)
- ▶ Effizient (Default-Auflösung: 10 Sekunden)
- ▶ Modular (über 90 Plugins in Version 4.10)



Wichtige Eigenschaften

- ▶ Daemon
- ▶ Freie Software (größtenteils GPLv2)
- ▶ Portierbar (Linux, *BSD, Solaris, ...)
- ▶ Skalierbar (OpenWrt, ..., Cluster / Cloud)
- ▶ Effizient (Default-Auflösung: 10 Sekunden)
- ▶ **Modular** (über 90 Plugins in Version 4.10)



Verfügbare Plugins (Auswahl; Stand: Version 4.10)

apache	apcups	apple_sensors	ascent	battery
bind	contrack	contextswitch	cpu	cpufreq
csv	curl	curl_json	dbi	df
disk	dns	email	entropy	exec
filecount	fscache	GenericJMX	gmond	hddtemp
interface	ipmi	iptables	ipvs	irq
java	libvirt	load	logfile	madwifi
match_regex	mbmon	memcachec	memcached	memory
Monitorus	multimeter	mysql	netapp	netlink
network	nfs	nginx	notify_email	ntpd
nut	olsrd	onewire	openvpn	OpenVZ
oracle	perl	ping	postgresql	powerdns
processes	protocols	python	routeros	rrdcached
rrdtool	sensors	serial	snmp	swap
syslog	table	tail	tape	target_scale
tcpconns	teamspeak2	ted	thermal	tokyotyrrant
unixsock	uptime	users	uuid	vmem
vserver	wireless	write_http	xmms	zfs_arc



Technische Details

- ▶ Aktuelle Version ist 4.10.1 (Release: Juli 2010)
- ▶ Pakete für diverse Distributionen vorhanden (Debian, RedHat, FreeBSD, OpenWrt, OpenSolaris [WIP], ...)
- ▶ Major-Version 3.* ist veraltet und inkompatibel
→ noch irgendwelche Debian Etch Benutzer hier? ;-)
- ▶ Geschrieben in C
- ▶ Versionsverwaltung mit Git
→ `git://git.verplant.org/collectd.git`



Verwendung

- ▶ Daemon läuft auf jedem Client
- ▶ üblicherweise: ein oder mehrere zentrale Server, die Werte von Clients empfangen (Push-Modell)
- ▶ First steps: `install; select plugins; start daemon; enjoy ;-)`



Collection 4 (C4)

C4

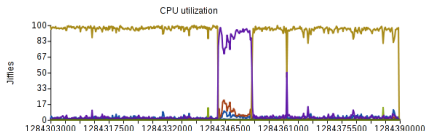
collection 4

- All instances
- All graphs
- Host "huhu.verplant.org"

Graph "CPU utilization"

Instance "huhu.verplant.org/0"

Instance: "huhu.verplant.org / cpu - 0 / cpu - all"

 Search

Hour ▾

JSON (gRaphaël)

RRDtool

Go

collection 4.0.0

- ▶ Grundidee: Daten über, z. B., JSON zur Verfügung stellen
- ▶ verschiedene Frontends davor möglich
- ▶ effiziente Handhabung von vielen Datensätzen durch Caching
- ▶ flexible Konfiguration von Graphen



Was ist collectd?

Wichtige Plugins

CPU, Speicher, Netzwerk-Schnittstellen

Netzwerk-Plugin

RRDtool-Plugin (Überblick)

Generische Plugins (Überblick)

Eigene Erweiterungen (Überblick)

Über den Tellerrand

Optional



Wichtige Plugins

- ▶ Spezielle Lese-Plugins
 - ▶ CPU, Speicher, Netzwerk-Schnittstellen
- ▶ Schreib- bzw. IO-Plugins
 - ▶ Netzwerk-Plugin
 - ▶ RRDtool, RRDCacheD (optional)
- ▶ Generische Plugins (optional)
 - ▶ SNMP
 - ▶ tail



CPU, Speicher, Netzwerk-Schnittstellen

Synopsis

```
LoadPlugin "cpu"  
LoadPlugin "memory"  
LoadPlugin "interface"
```



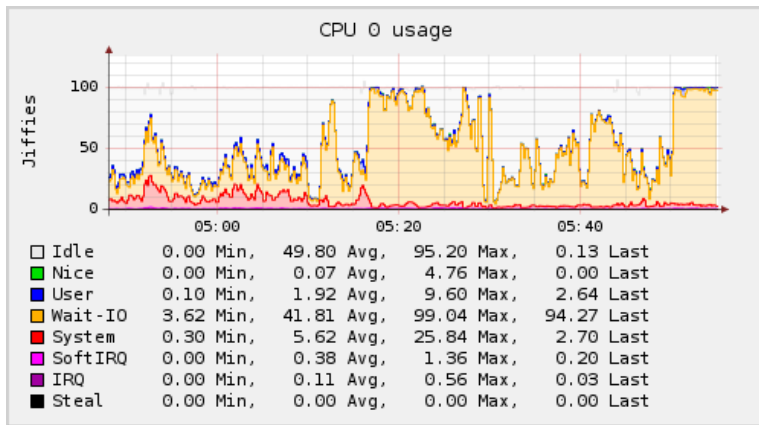
Synopsis

```
LoadPlugin "cpu"  
LoadPlugin "memory"  
LoadPlugin "interface"
```

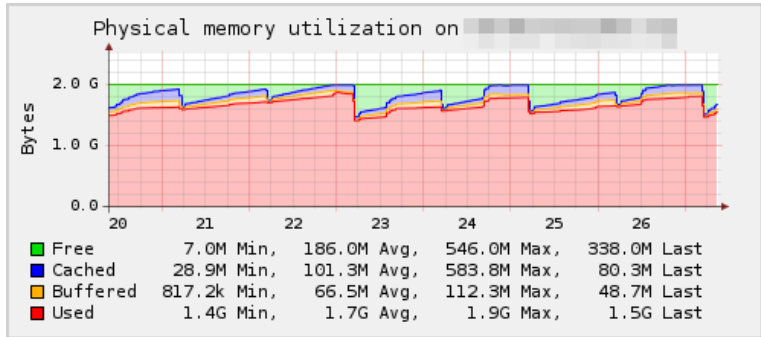
```
<Plugin interface>  
  Interface lo  
  Interface sit0  
  IgnoreSelected true  
</Plugin>
```



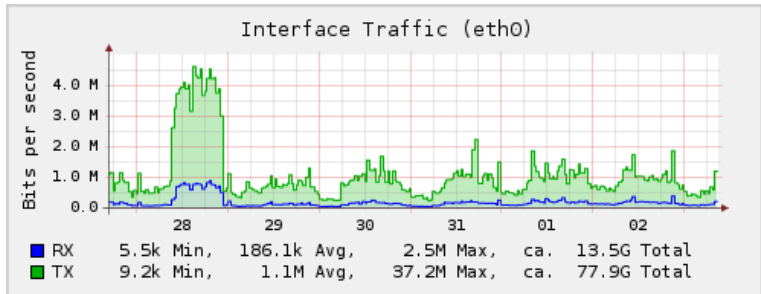
CPU, Speicher, Netzwerk-Schnittstellen



CPU, Speicher, Netzwerk-Schnittstellen



CPU, Speicher, Netzwerk-Schnittstellen



Betriebsarten

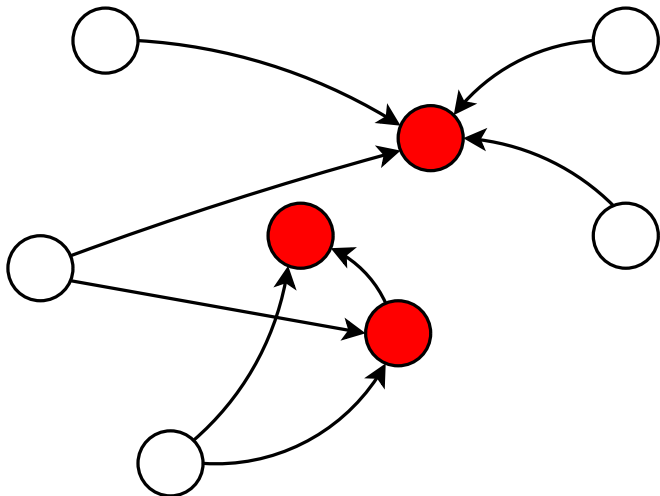
- ▶ Daten versenden („*Client*“)
- ▶ Daten empfangen („*Server*“)
- ▶ Weiterleiten („*Proxy*“)
- ▶ Unicast („*Punkt-zu-Punkt*“)
- ▶ Multicast („*Punkt-zu-Gruppe*“)
- ▶ IPv4 und IPv6

Ein Daemon für alles

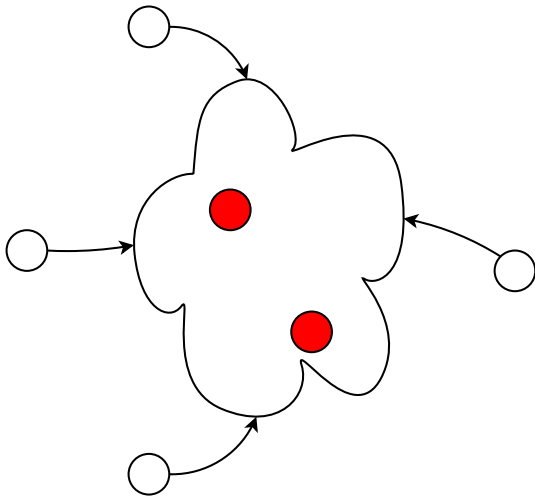
Rolle des Daemon hängt von der Konfiguration ab.



Netzwerk-Plugin: Unicast



Netzwerk-Plugin: Multicast



Netzwerk-Plugin: Server/Client

Synopsis: Client

```
LoadPlugin "network"
```

```
<Plugin "network">
```

```
  Server "collectd0.musterfirma.de"
```

```
  Server "collectd1.musterfirma.de"
```

```
  Server "ff18::efc0:4a42"
```

```
</Plugin>
```



Netzwerk-Plugin: Server/Client

Synopsis: Server

```
LoadPlugin "network"
```

```
<Plugin "network">
```

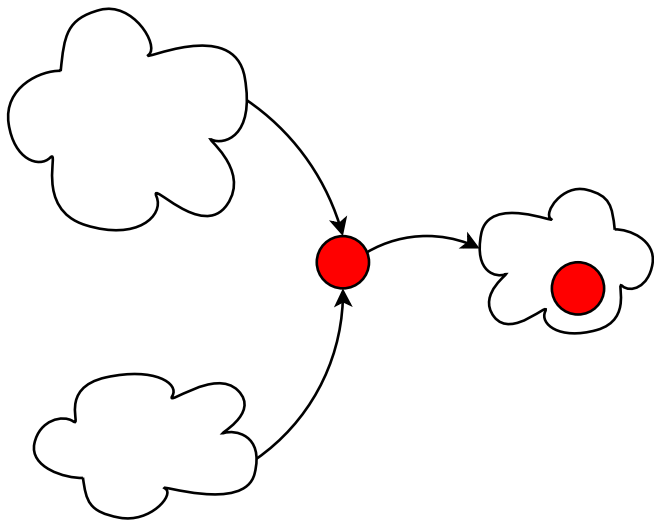
```
  Listen "collectd0.musterfirma.de"
```

```
  Listen "ff18::efc0:4a42"
```

```
</Plugin>
```



Netzwerk-Plugin: Proxy



Synopsis: Proxy

```
LoadPlugin "network"
```

```
<Plugin "network">
```

```
  Listen "collectgw.extern.musterfirma.de"
```

```
  Server "collectd1.intern.musterfirma.de"
```

```
  Forward true
```

```
</Plugin>
```



Authentifizierung / Verschlüsselung

- ▶ (seit Version 4.7.0)
- ▶ Authentifizierung via HMAC-SHA-256
- ▶ Verschlüsselung mit AES-256 (OFB)



Authentifizierung / Verschlüsselung

		Client		
		Nichts	Sign	Encrypt
Server	Nichts	akzeptiert	akzeptiert	nicht möglich
	AuthFile	akzeptiert	akzeptiert	akzeptiert
	Sign	nicht akzeptiert	akzeptiert	akzeptiert
	Encrypt	nicht akzeptiert	nicht akzeptiert	akzeptiert



RRDtool-Plugin (Überblick)

- ▶ Schreibt Daten effizient in RRD-Dateien → Caching
- ▶ Funktionalität nun in RRDtool als RRD Caching Daemon verfügbar

Synopsis

```
LoadPlugin "rrdtool"
```

```
<Plugin "rrdtool">
```

```
  DataDir "/var/lib/collectd/rrd"
```

```
</Plugin>
```



RRDtool-Plugin (Caching)

Konfiguration

```
<Plugin "rrdtool">  
  DataDir "/var/lib/collectd/rrd"  
  
  CacheTimeout 3600 # 1 hour  
  CacheFlush 86400 # 1 day  
  
  WritesPerSecond 30  
</Plugin>
```

- ▶ FLUSH ermöglicht dennoch die graphische Darstellung von aktuellen Daten



Generische Plugins (Überblick)

- ▶ Idee: Generische Ansätze, statt Speziallösungen
- ▶ → Benutzerkonfiguration bestimmt das Verhalten
- ▶ ⇒ Unterstützung für neue Geräte braucht i.d.R. keine neue Version von **collectd**
- ▶ Beispiele: SNMP, tail, curl, DBI

- ▶ Details (SNMP oder tail) im optionalen Teil



Eigene Erweiterungen (Überblick)

- ▶ **collectd** API: C, Perl, Python, Java
- ▶ Externe Programme mittels unixsock- oder exec-Plugin
- ▶ Beispiele (Perl, unixsock, exec, Java) im optionalen Teil



Wichtige Plugins: Zusammenfassung

- ▶ Plugins für zahlreiche Systemmetriken existieren (z. B. für CPU, Speicher und Netzwerk-Schnittstellen)



Wichtige Plugins: Zusammenfassung

- ▶ Plugins für zahlreiche Systemmetriken existieren (z. B. für CPU, Speicher und Netzwerk-Schnittstellen)
- ▶ Vielfältige Netzwerk-Möglichkeiten (IPv4, IPv6, Unicast, Multicast, Proxies)



Wichtige Plugins: Zusammenfassung

- ▶ Plugins für zahlreiche Systemmetriken existieren (z. B. für CPU, Speicher und Netzwerk-Schnittstellen)
- ▶ Vielfältige Netzwerk-Möglichkeiten (IPv4, IPv6, Unicast, Multicast, Proxies)
- ▶ Bewährtes Caching-Modell für RRD-Dateien



Wichtige Plugins: Zusammenfassung

- ▶ Plugins für zahlreiche Systemmetriken existieren (z. B. für CPU, Speicher und Netzwerk-Schnittstellen)
- ▶ Vielfältige Netzwerk-Möglichkeiten (IPv4, IPv6, Unicast, Multicast, Proxies)
- ▶ Bewährtes Caching-Modell für RRD-Dateien
- ▶ Mächtige, generische Ansätze statt Speziallösungen (z. B. SNMP- und tail-Plugins)



Was ist collectd?

Wichtige Plugins

Über den Tellerrand

Optional



Über den Tellerrand: Interaktion

- ▶ `collectd-nagios`
Fragt Daten via `unixsock`-Plugin ab und erzeugt Nagios-kompatible Ausgabe
- ▶ `exec-nagios.px`
Perl-Skript welches *Nagios*-Plugins ausführt (→ *exec-Plugin*)
- ▶ `exec-munin.px`
Perl-Skript welches *Munin*-Plugins ausführt (→ *exec-Plugin*)
- ▶ `gmond-Plugin`
Empfängt und verarbeitet *Ganglia* Multicast-Pakete



Vielen Dank für die Aufmerksamkeit!

Gibt es Fragen?

Kontakt:

Sebastian „tokkee“ Harl

<tokkee@collectd.org>

<collectd@verplant.org> — irc.freenode.net/#collectd — <http://identi.ca/collectd>

Danke an Florian Forster für die initiale Version dieser Folien!



Was ist collectd?

Wichtige Plugins

Über den Tellerrand

Optional

- SNMP-Plugin

- tail-Plugin

- RRDCacheD-Plugin

- Eigene Erweiterungen

- Über den Tellerrand



Allgemeines

- ▶ Fragt Netzwerk-Zubehör via SNMP ab
- ▶ *Generisch*: Nicht für ein gestimmtes Gerät geschrieben
- ▶ Mehrere Geräte werden parallel abgefragt

Konfiguration

- ▶ „Data“-Blöcke
- ▶ „Host“-Blöcke



Synopsis: Data-Block

```
<Plugin "snmp">
  <Data "ifmib_if_octets64">
    Type "if_octets"
    Table true
    Instance "IF-MIB::ifName"
    Values "IF-MIB::ifHCInOctets" \
          "IF-MIB::ifHCOutOctets"
  </Data>
</Plugin>
```

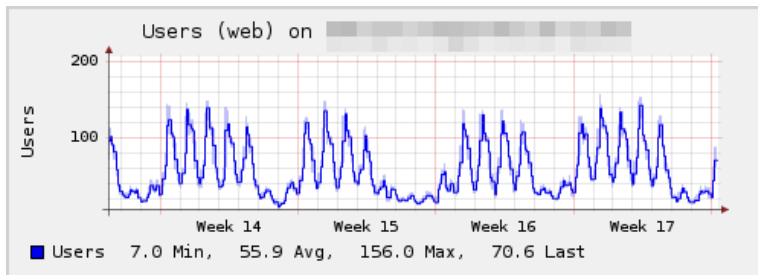


Synopsis: Host-Block

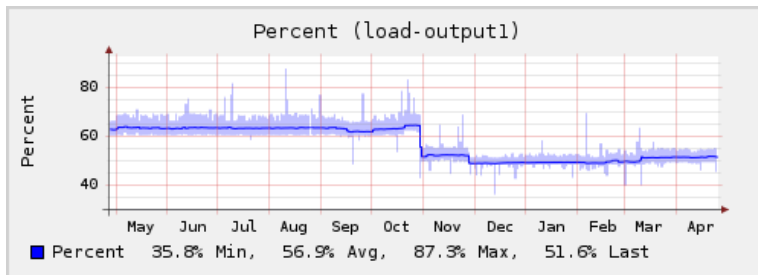
```
<Plugin "snmp">  
  <Host "switch0.intern.musterfirma.de">  
    Address "10.0.42.2"  
    Version 1  
    Community "public"  
    Collect "ifmib_if_octets64"  
    Interval 60  
  </Host>  
</Plugin>
```



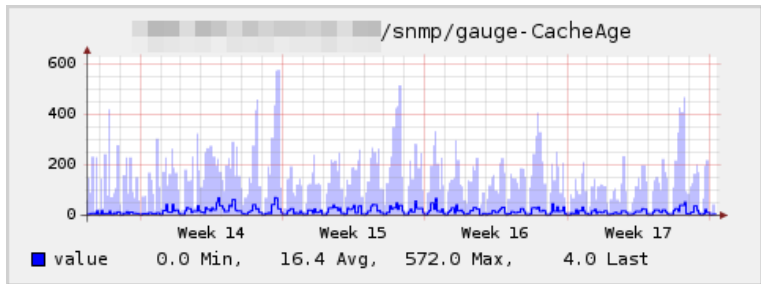
SNMP-Plugin: Users



SNMP-Plugin: USV-Last



SNMP-Plugin: Cache-Alter



Allgemeines

- ▶ Verfolgt Log-Dateien
- ▶ Extrahiert Werte oder zählt Ereignisse
- ▶ Selektion der Zeilen / Werte mit regulären Ausdrücken
- ▶ Verwendbar für MTAs, Web-Server, ...

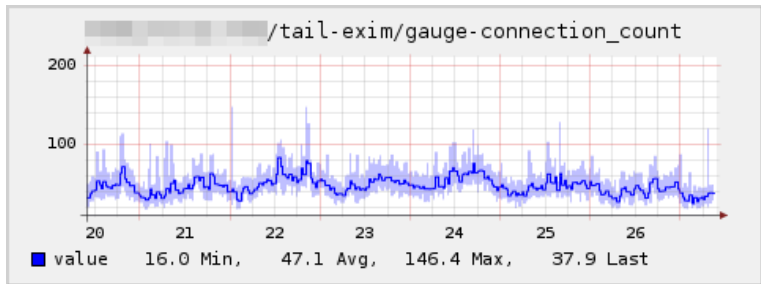


Konfiguration

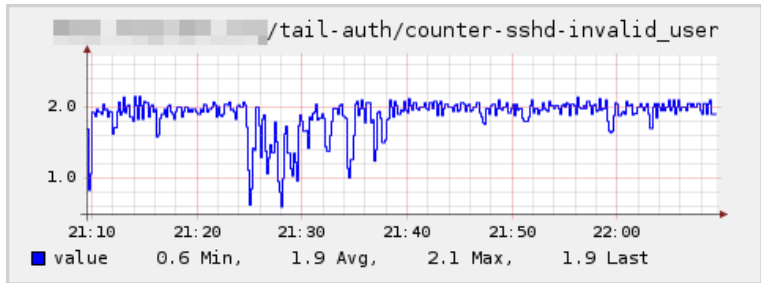
```
<Plugin "tail">
  <File "/var/log/exim4/mainlog">
    Instance "exim"
    <Match>
      Regex "S=([1-9][0-9]*)"
      DStype "CounterAdd"
      Type "ipt_bytes"
      Instance "total"
    </Match>
  </File>
</Plugin>
```



tail-Plugin: Verbindungen von Exim



tail-Plugin: SSH Brute-Force-Attacke

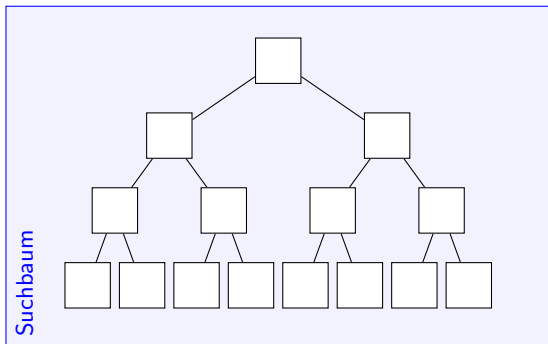


Allgemeines

- ▶ Update-Prinzip des RRDtool-Plugins
- ▶ Eigenständiger Daemon
- ▶ Integration in RRDtool 1.4
- ▶ Weitere Funktionen, z. B. Journaling
- ▶ Vorteil: Neustart von **collectd** ohne Cache-Verlust



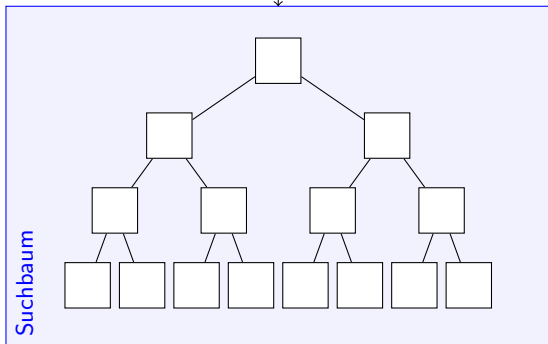
RRDCacheD-Plugin



Graphik © Florian „octo“ Forster



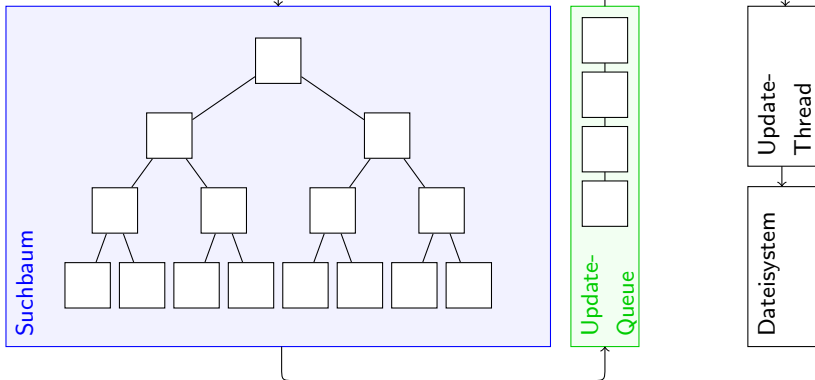
RRDCacheD-Plugin



Graphik © Florian „octo“ Forster



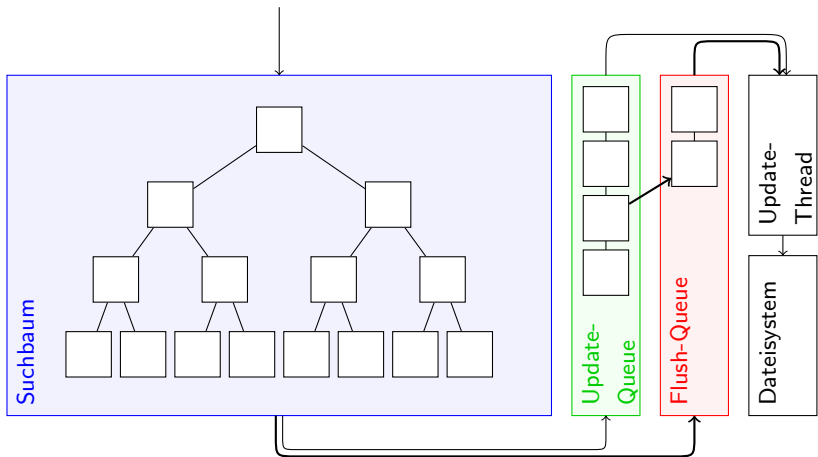
RRDCacheD-Plugin



Graphik © Florian „octo“ Forster



RRDCacheD-Plugin



Graphik © Florian „octo“ Forster



Allgemeines

- ▶ Integriert einen Perl-Interpreter
(vergleichbar zu Apaches `mod_perl`)
- ▶ Instanziierung und Syntax-Analyse nur einmal
- ▶ Exportiert die API
(→ nicht nur Lese-Plugins möglich)



Perl-Plugin: Beispiel

```
package Collectd::Plugin::Magic;
use Collectd qw( :all );
sub magic_read
{
    my $v1 = { plugin => 'magic',
               values => [Magic->getCurrentLevel ()] };
    plugin_dispatch_values ('magic_level', $v1);
}
plugin_register (TYPE_READ, 'magic', 'magic_read');
```



Allgemeines

- ▶ Öffnet einen UNIX-Domain-Socket
- ▶ Kennt mehrere Befehle
(z. B. PUTVAL, FLUSH, LISTVAL)
- ▶ Interaktion mit externen Programmen möglich
- ▶ collectdctl (ab Version 5.0, eta: dieses Jahr ;-))
- ▶ `cussh.pl`: „*collectd UNIX socket shell*“



unixsock-Plugin: Beispiel

```
-> | PUTVAL "testhost/magic/magic_level" \  
    interval=10 1179574444:42  
<- | 0 Success
```



Allgemeines

- ▶ Führt Programme aus
- ▶ Liest von deren Standard-Ausgabe
- ▶ Können über längere Perioden laufen
(vgl. `init`)



exec-Plugin: Beispiel

```
#!/bin/sh
INTVL=${COLLECTD_INTERVAL:-10}
CHOST="${COLLECTD_HOSTNAME:-localhost}"
IDENT="$CHOST/magic/magic_level"
while sleep $INTVL
do
    VALUE='magic --level'
    echo "PUTVAL \"$IDENT\" interval=$INTVL N:$VALUE"
done
```



Allgemeines

- ▶ Integriert eine „Java Virtual Maschine“ (JVM)
- ▶ Exportiert die API
(→ nicht nur Lese-Plugins möglich)
- ▶ Prinzipielle Ähnlichkeit zum Perl-Plugin



Java-Plugin: Beispiel

```
import org.collectd.api.Collectd;
import org.collectd.api.CollectdReadInterface;
public class MagicPlugin
    implements CollectdReadInterface
{
    public int read ();    /* Callback-Funktion */
    public MagicPlugin (); /* Konstruktor */
}
```



Java-Plugin: Beispiel

```
public int read ()
{
    ValueList vl = new ValueList ();
    vl.setHost ("testhost");
    vl.setPlugin ("magic");
    vl.setType ("magic_level");
    vl.addValue (Magic.getCurrentLevel ());
    return (Collectd.dispatchValues (vl));
}
```



Java-Plugin: Beispiel

```
public MagicPlugin ()  
{  
    /* Callback-Funktion anmelden */  
    Collectd.registerRead ("MagicPlugin", this);  
}
```



Eigene Erweiterungen: Zusammenfassung

- ▶ **collectd** API nutzen
C, Perl, Python und Java möglich



Eigene Erweiterungen: Zusammenfassung

- ▶ **collectd** API nutzen
C, Perl, Python und Java möglich
- ▶ Externe Programme erweitern
unixsock-Plugin ermöglicht Kommunikation



Eigene Erweiterungen: Zusammenfassung

- ▶ **collectd** API nutzen
C, Perl, Python und Java möglich
- ▶ Externe Programme erweitern
unixsock-Plugin ermöglicht Kommunikation
- ▶ Eigenes Programm / Skript schreiben
→ exec-Plugin



Über den Tellerrand: Zubehör

- ▶ `snmp-probe-host.px`
Erzeugt semi-automatisch `<Host />`-Blöcke für das SNMP-Plugin
- ▶ `jcollected`
Java-Implementierung des Netzwerk-Protokolls (→ *JMX*)
- ▶ `kcollected`
KDE-Programm zur Near-Realtime-Anzeige von Graphen
- ▶ Perl, Ruby, Python Module und C-Bibliothek für die Kommunikation mit dem `unixsock`-Plugin verfügbar

