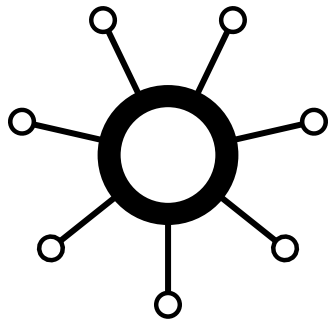
Performance-Analyse mit **collect**d

Wisse, was deine Rechner tun

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

collectd core team

Grazer Linuxtage 2010
24. April 2010



Was ist collectd?
Wichtige Eigenschaften

Wichtige Plugins

Optional

Was ist **collect**d?

- **collect**d sammelt Leistungsdaten von Rechnern
- Leistungsdaten sind zum Beispiel:
  - CPU-Auslastung
  - Speichernutzung
  - Netzwerkverkehr
- Daten werden erhoben, verarbeitet und gespeichert
- Häufig: Darstellung als Graphen
- $\rightarrow$ Performance-Analyse, Kapazitätsplanung
- Nicht verwechseln mit *Monitoring!*
- Homepage: `http://collectd.org/`

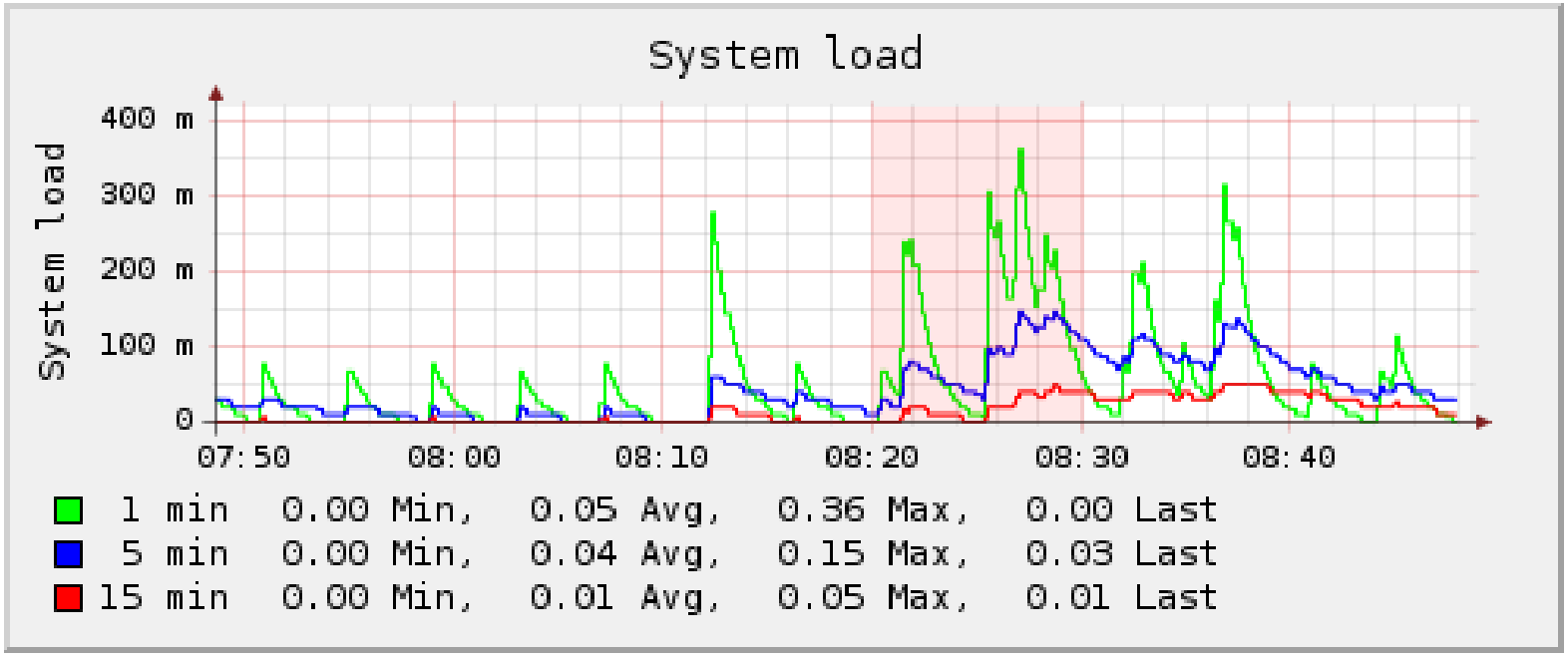
Wichtige Eigenschaften

- Daemon
- Freie Software (größtenteils GPLv2)
- Portierbar (Linux, *BSD, Solaris, . . .)
- Skalierbar (OpenWrt, . . . , Cluster / Cloud)
- Effizient (Default-Auflösung: 10 Sekunden)
- Modular (über 90 Plugins in Version 4.9)

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 4.9)

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 4.9)

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 4.9)

Verfügbare Plugins (Auswahl; Stand: Version 4.9)

apache	apcups	apple_sensors	ascent	battery
bind	conntrack	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	tokyotyrant
unixsock	uptime	users	uuid	vmem
vserver	wireless	write_http	xmms	zfs_arc

Technische Details

- Aktuelle Version ist 4.9.2
- Pakete für diverse Distributionen vorhanden
  (Debian, RedHat, FreeBSD, OpenWrt, . . .)
- Major-Version 3.$*$ ist veraltet und inkompatibel
  $\rightarrow$ noch irgendwelche Debian Etch Benutzer hier? ;-)
- Geschrieben in C
- Versionsverwaltung mit Git
  $\rightarrow$ `git://git.verplant.org/collectd.git`

Was ist collectd?

Wichtige Plugins

- CPU, Speicher, Netzwerk-Schnittstellen
- Netzwerk-Plugin
- RRDtool-Plugin (Überblick)
- Generische Plugins (Überblick)
- Eigene Erweiterungen (Überblick)

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"
```

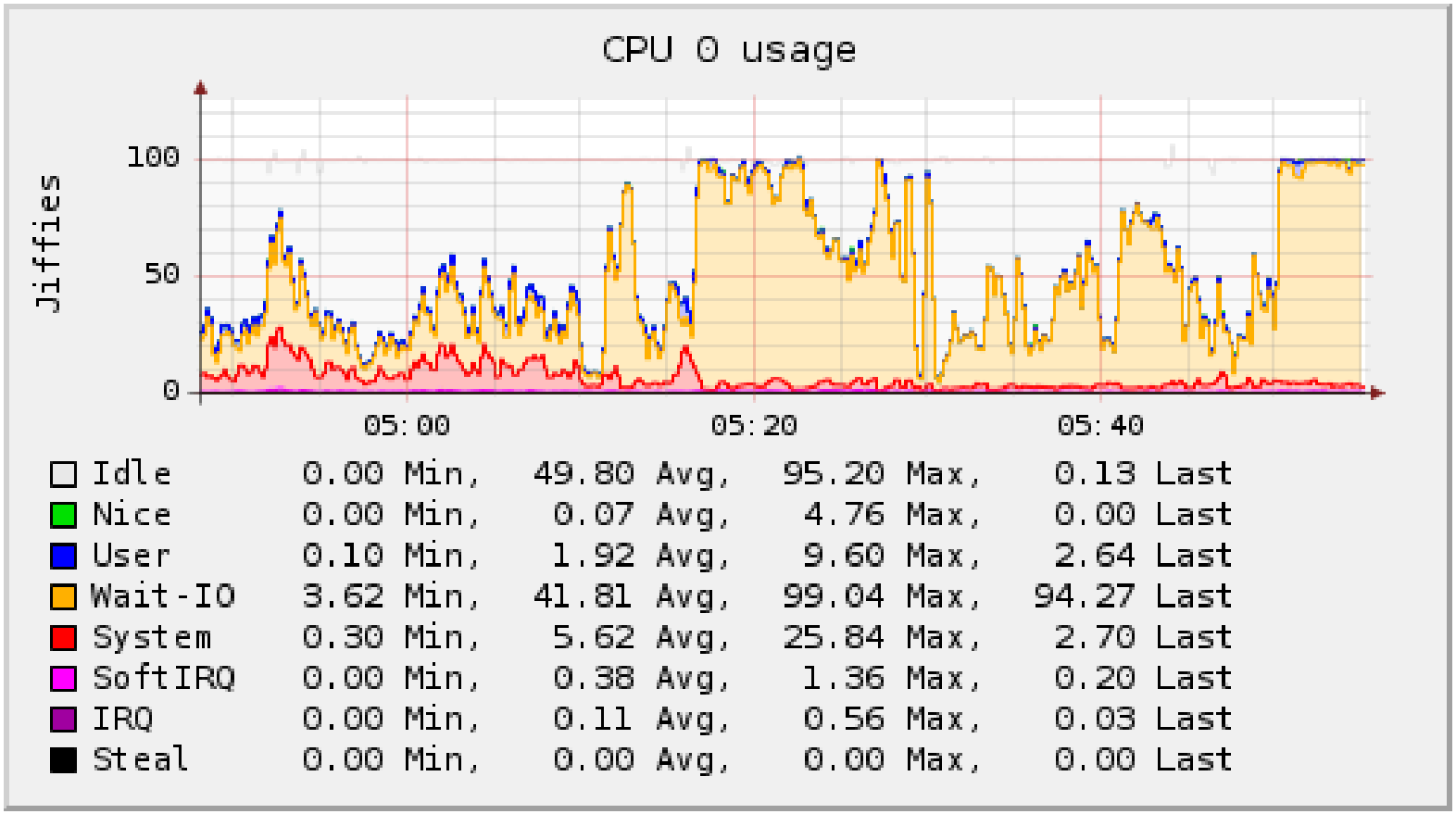

CPU, Speicher, Netzwerk-Schnittstellen

Synopsis

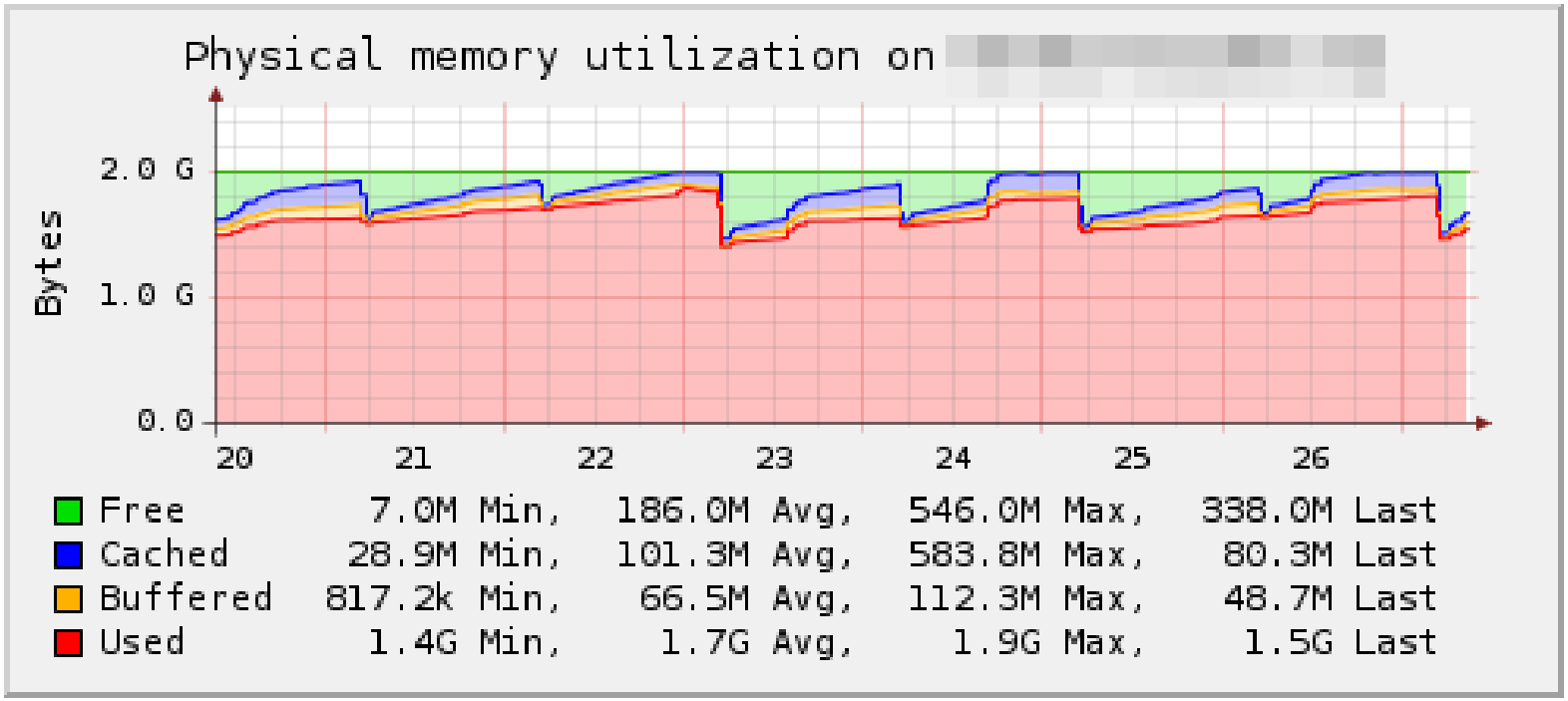
```
LoadPlugin "cpu"
LoadPlugin "memory"
LoadPlugin "interface"

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

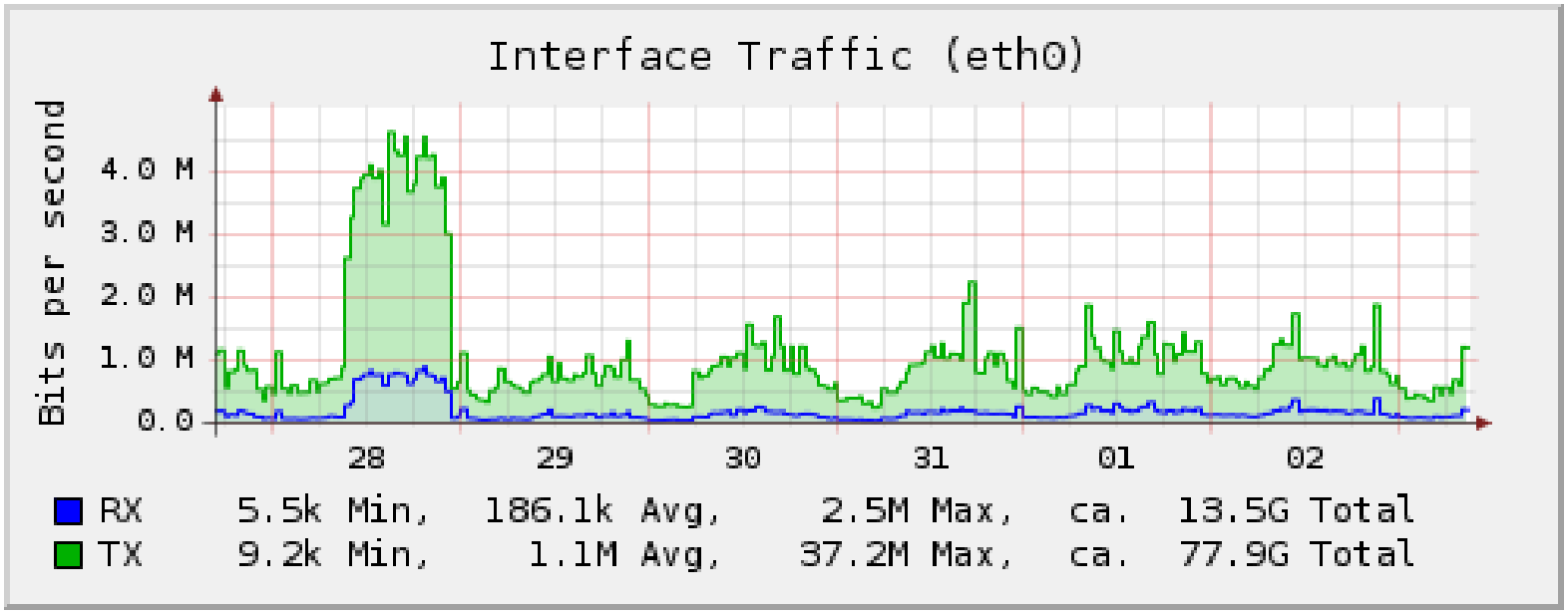

CPU, Speicher, Netzwerk-Schnittstellen



CPU, Speicher, Netzwerk-Schnittstellen



CPU, Speicher, Netzwerk-Schnittstellen



Netzwerk-Plugin

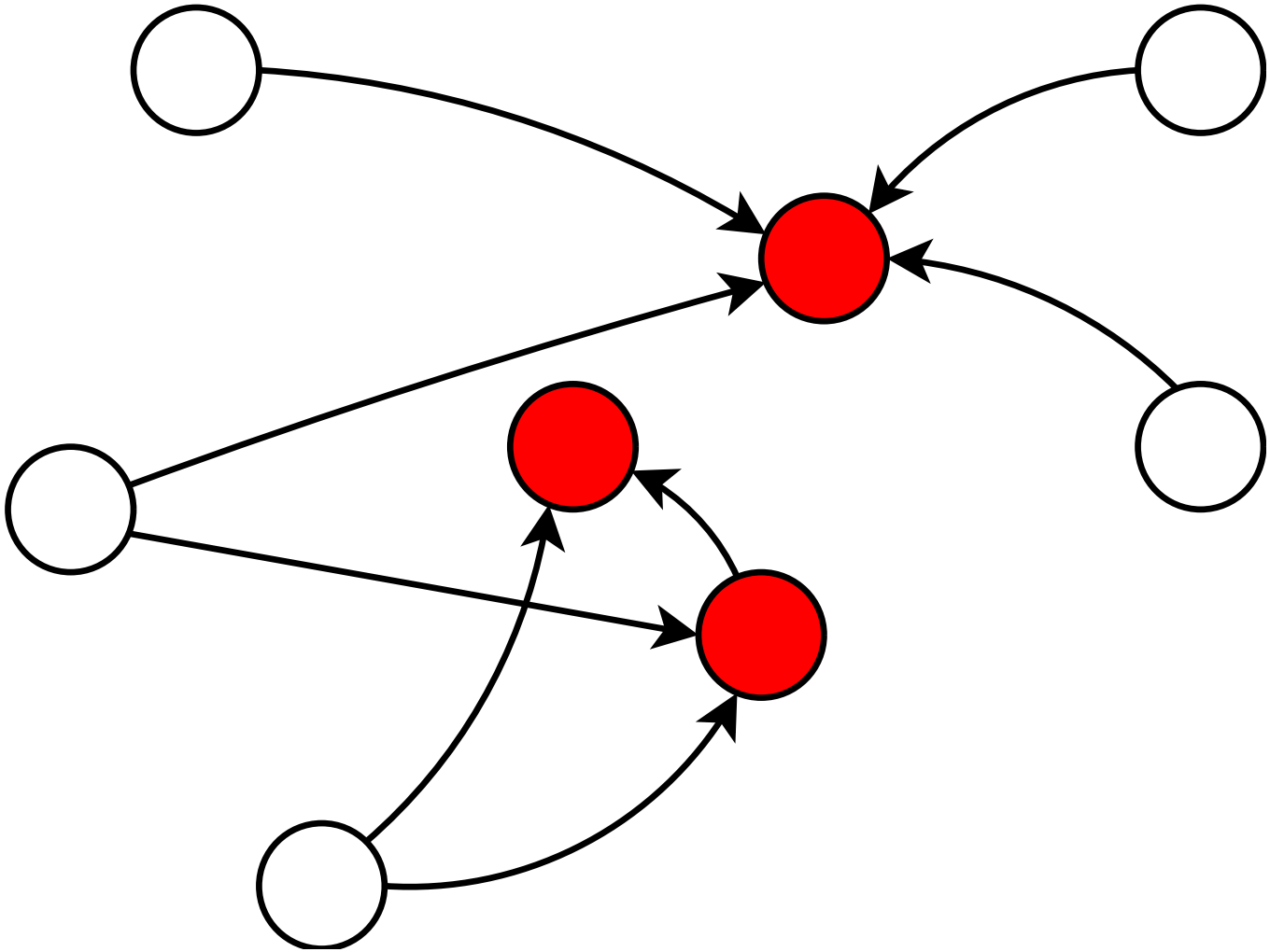
Betriebsarten

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

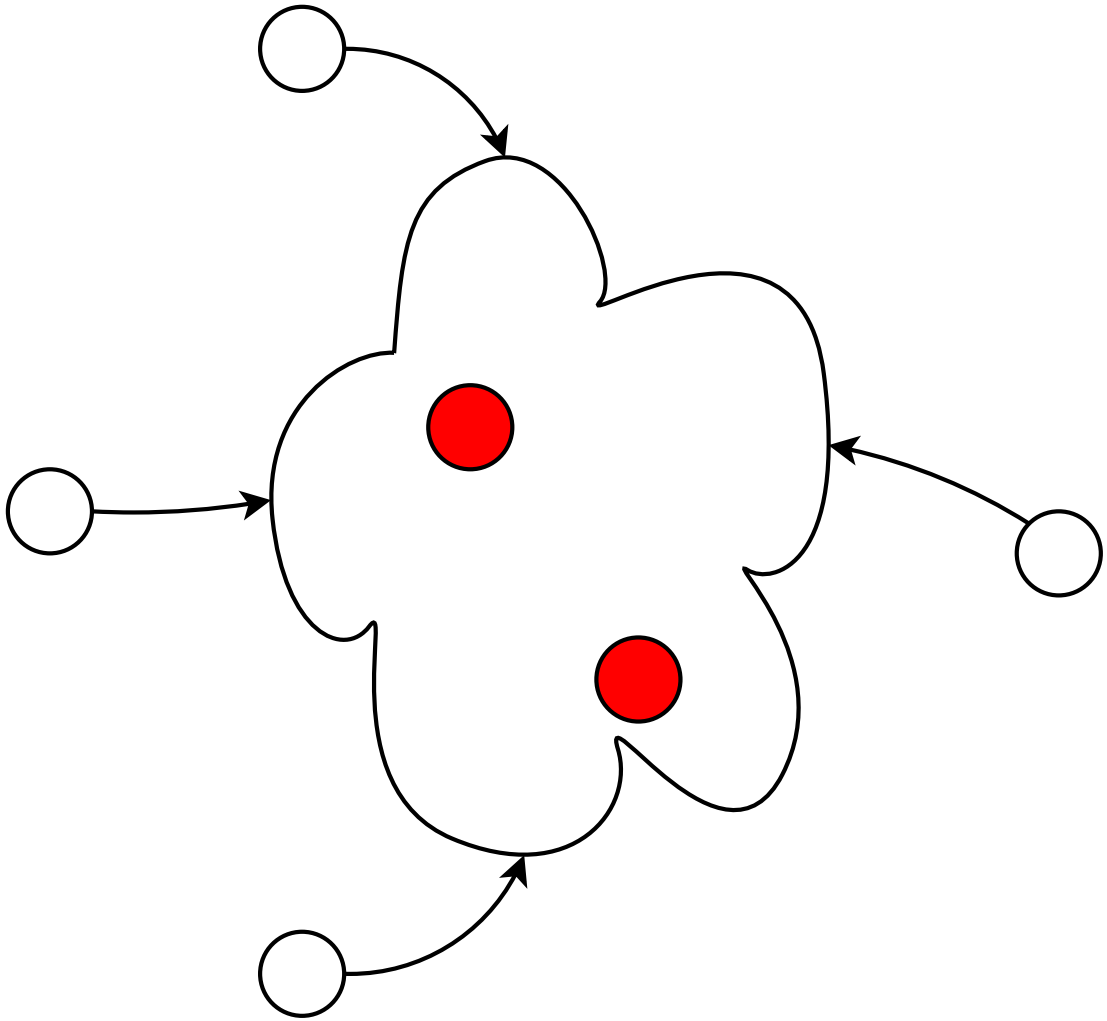
Ein Daemon für alles

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>
```

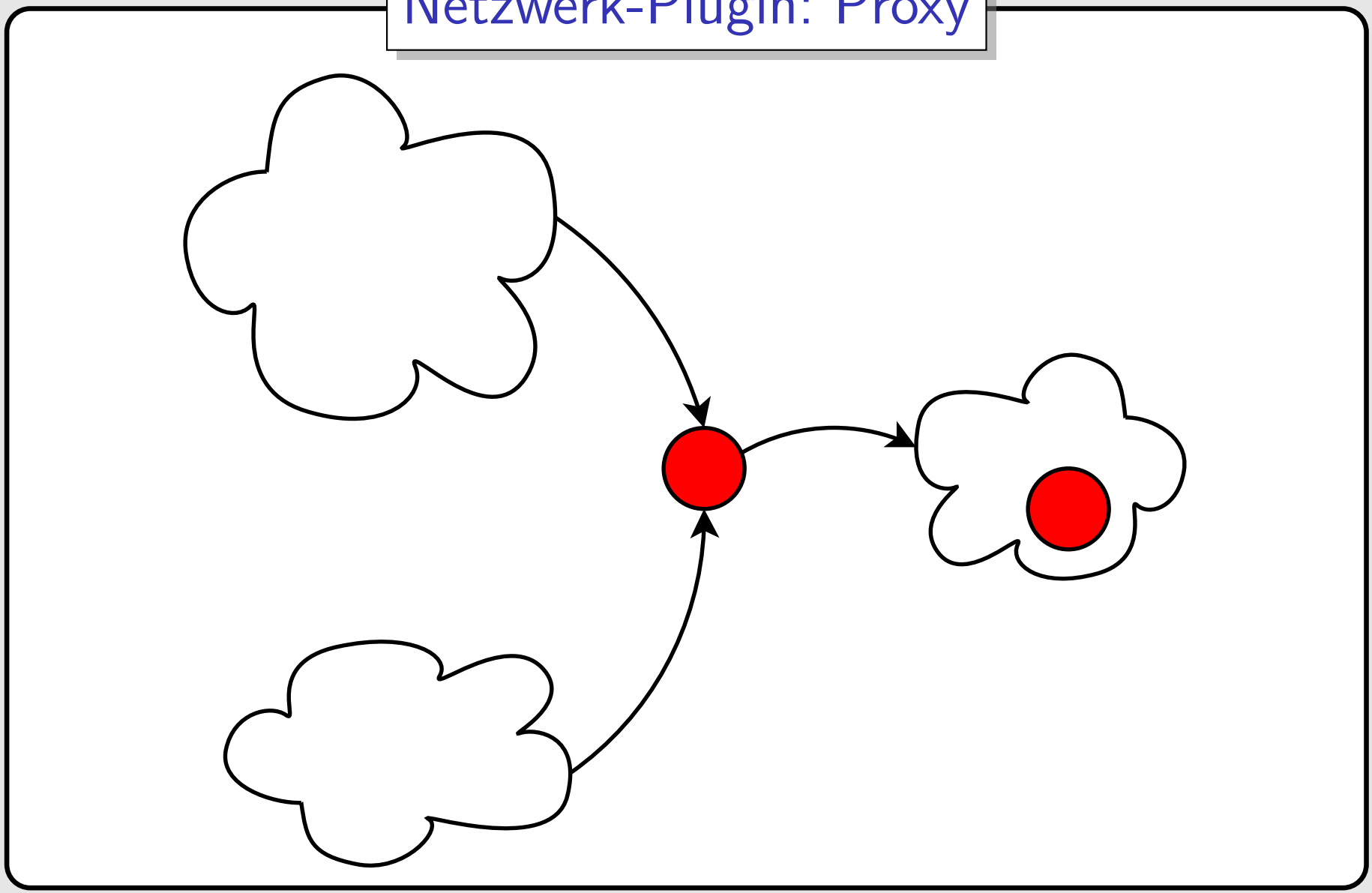

Netzwerk-Plugin: Server/Client

Synopsis: Server

```
LoadPlugin "network"

<Plugin "network">
  Listen "collectd0.musterfirma.de"
  Listen "ff18::efc0:4a42"
</Plugin>
```


Netzwerk-Plugin: Proxy



Netzwerk-Plugin: Proxy

Synopsis: Proxy

```
LoadPlugin "network"

<Plugin "network">
  Listen "collectgw.extern.musterfirma.de"
  Server "collectd1.intern.musterfirma.de"
  Forward true
</Plugin>
```


Netzwerk-Plugin

Authentifizierung / Verschlüsselung

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

Netzwerk-Plugin

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 RRDCache Daemon verfügbar

Synopsis

```
LoadPlugin "rrdtool"

<Plugin "rrdtool">
  DataDir "/var/lib/collectd/rrd"
</Plugin>
```


Generische Plugins (Überblick)

- Idee: Generische Ansätze, statt Speziallösungen
- $\rightarrow$ Benutzerkonfiguration bestimmt das Verhalten
- $\Rightarrow$ Unterstützung für neue Geräte braucht i.d.R. keine neue Version von **collect**d
- Beispiele: SNMP, tail, curl, DBI

Eigene Erweiterungen (Überblick)

- **collect**d API: C, Perl, Python, Java
- Externe Programme mittels unixsock- oder exec-Plugin

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)

Wisse, was deine Rechner tun

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

Optional

- SNMP-Plugin
- tail-Plugin
- RRDtool- und RRDCacheD-Plugins
- Eigene Erweiterungen
- Über den Tellerrand

SNMP-Plugin

Allgemeines

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

Konfiguration

- „Data“-Blöcke
- „Host“-Blöcke

SNMP-Plugin

Synopsis: Data-Block

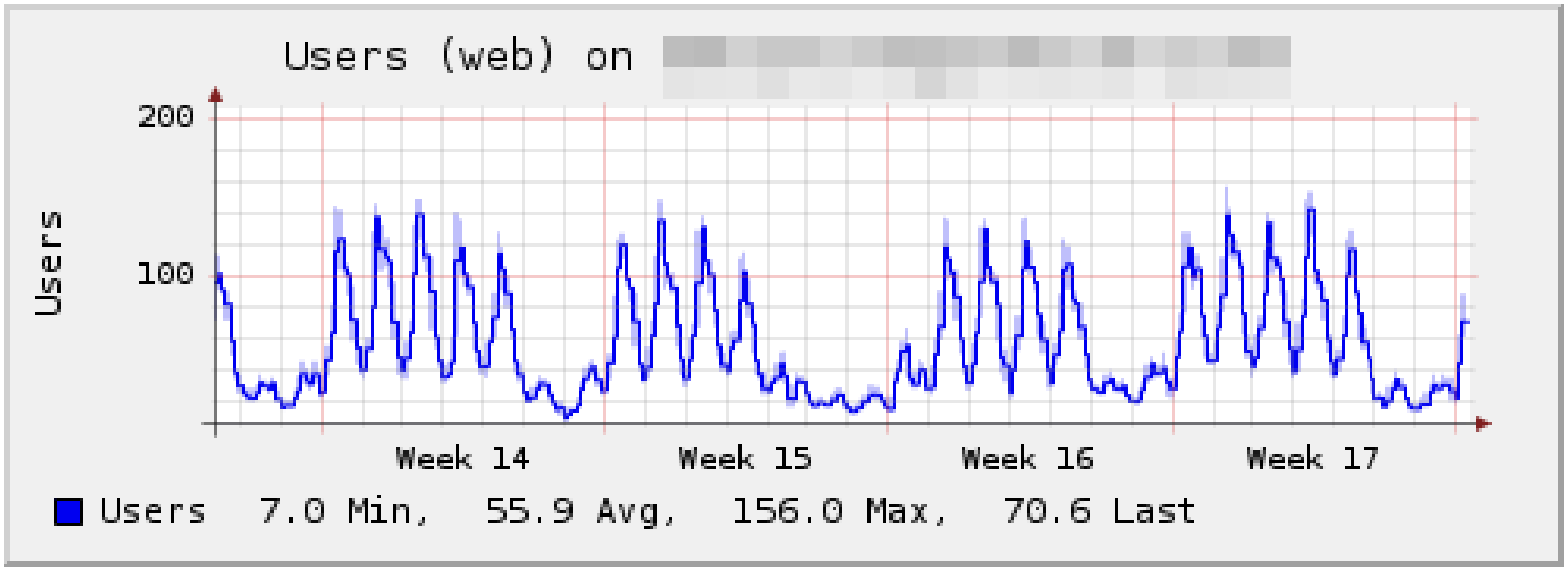
```
<Plugin "snmp">
  <Data "ifmib_if_octets64">
    Type "if_octets"
    Table true
    Instance "IF-MIB::ifName"
    Values "IF-MIB::ifHCInOctets" \
           "IF-MIB::ifHCOutOctets"
  </Data>
</Plugin>
```


SNMP-Plugin

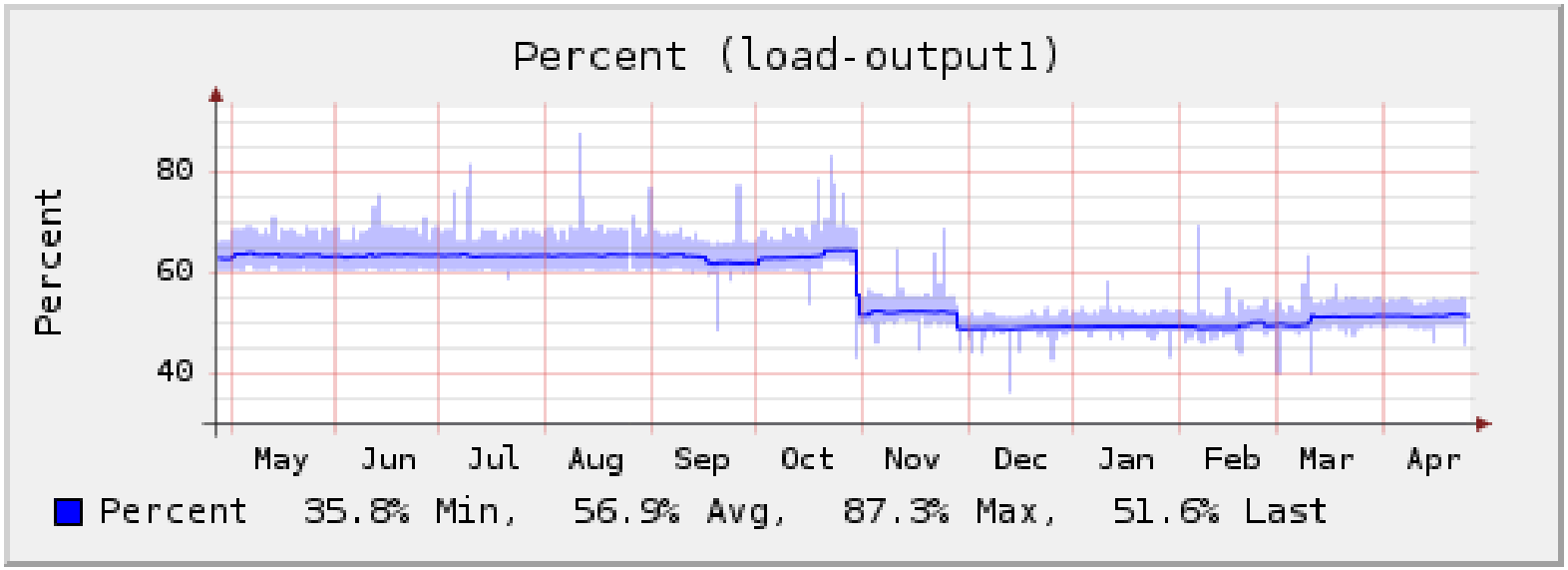
Synopsis: Host-Block

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

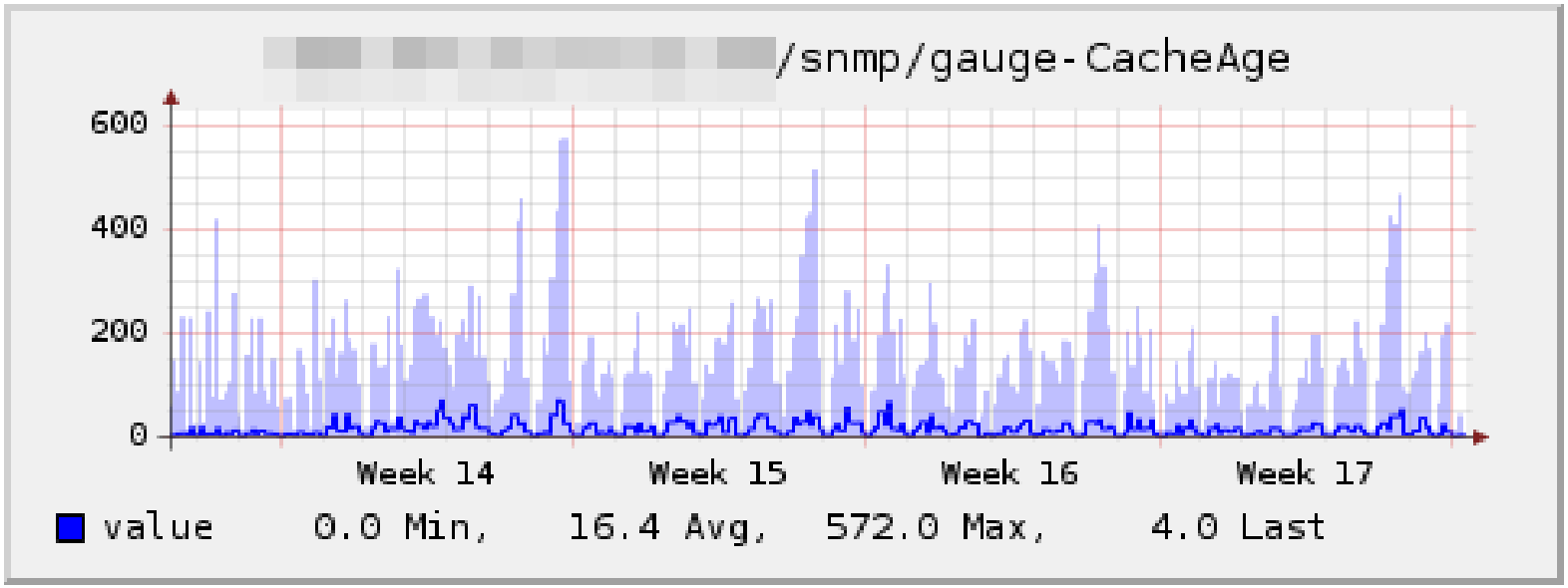

SNMP-Plugin: Users



SNMP-Plugin: USV-Last



SNMP-Plugin: Cache-Alter

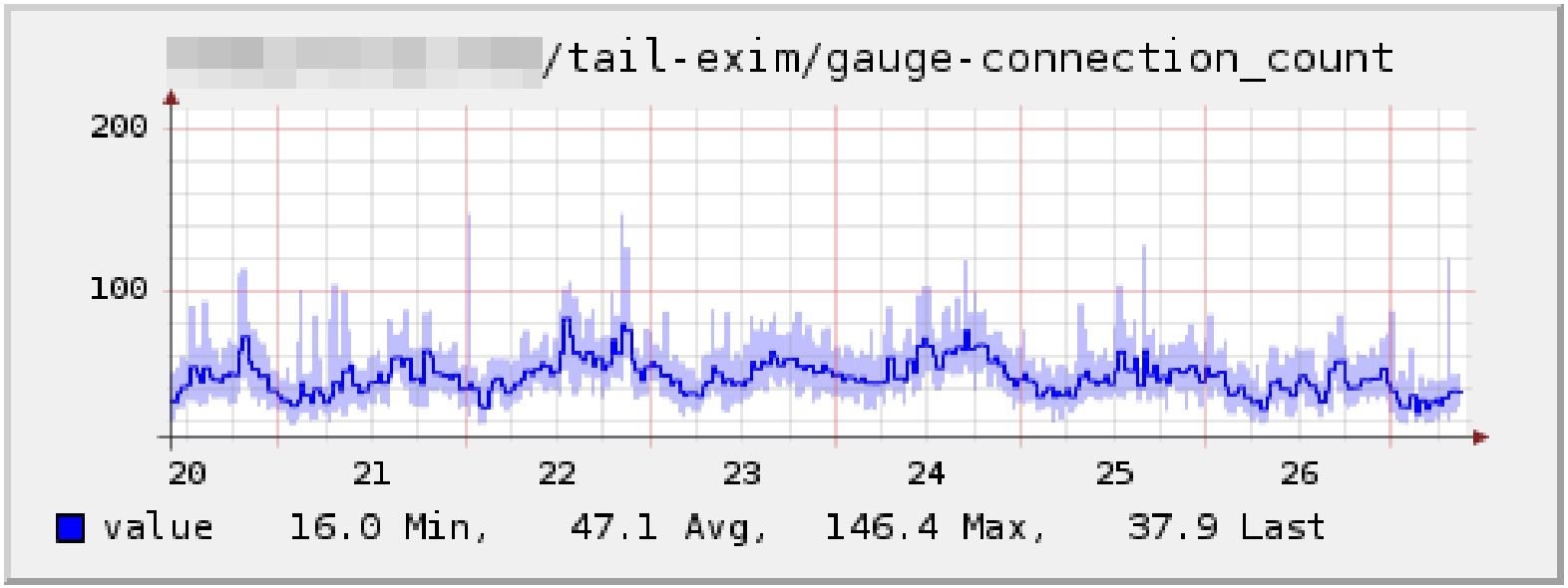


tail-Plugin

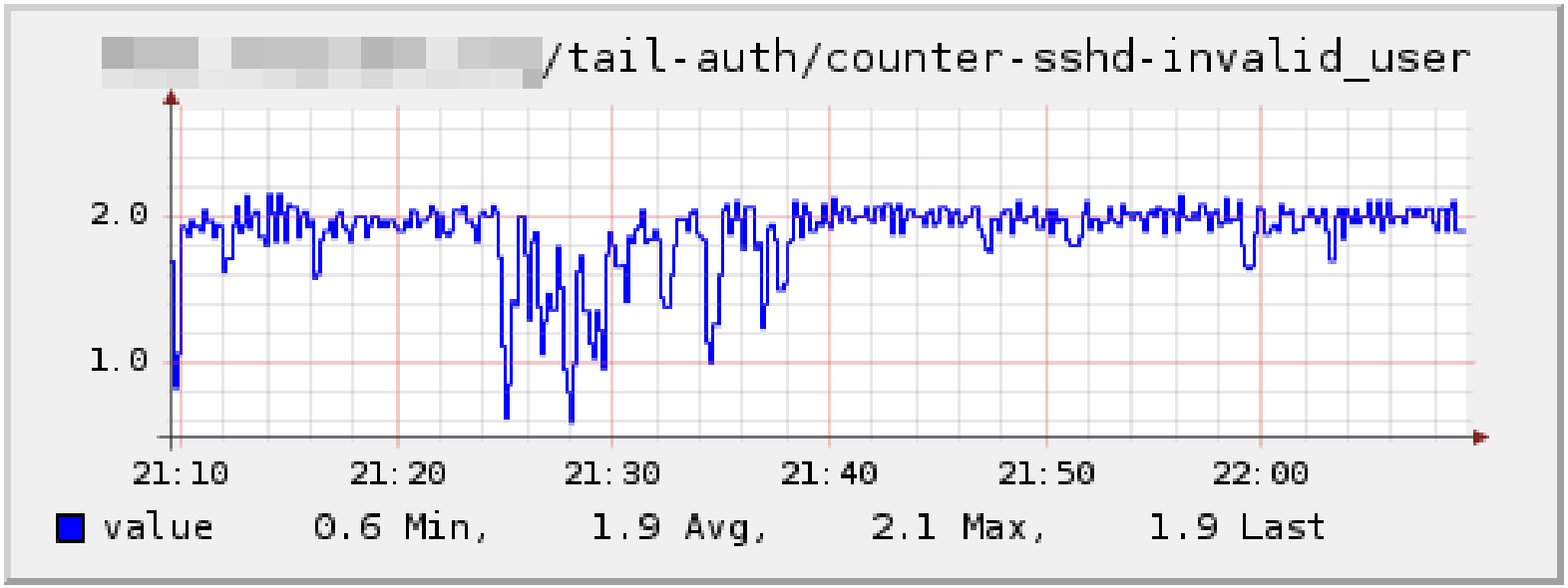
Allgemeines

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

tail-Plugin: Verbindungen von Exim



tail-Plugin: SSH Brute-Force-Attacke



RRDtool-Plugin

Allgemeines

- Schreibt Daten in RRD-Dateien

RRDtool-Plugin

Allgemeines

- Schreibt Daten *effizient* in RRD-Dateien

RRDtool-Plugin

Allgemeines

- Schreibt Daten *effizient* in RRD-Dateien
- *Caching* um Performance-Problemen zu begegnen

RRDtool-Plugin

Allgemeines

- Schreibt Daten *effizient* in RRD-Dateien
- *Caching* um Performance-Problemen zu begegnen
- *Flushing* für aktuelle Daten

RRDtool-Plugin

Allgemeines

- Schreibt Daten *effizient* in RRD-Dateien
- *Caching* um Performance-Problemen zu begegnen
- *Flushing* für aktuelle Daten
- *Throttling* für gleichmäßige Last

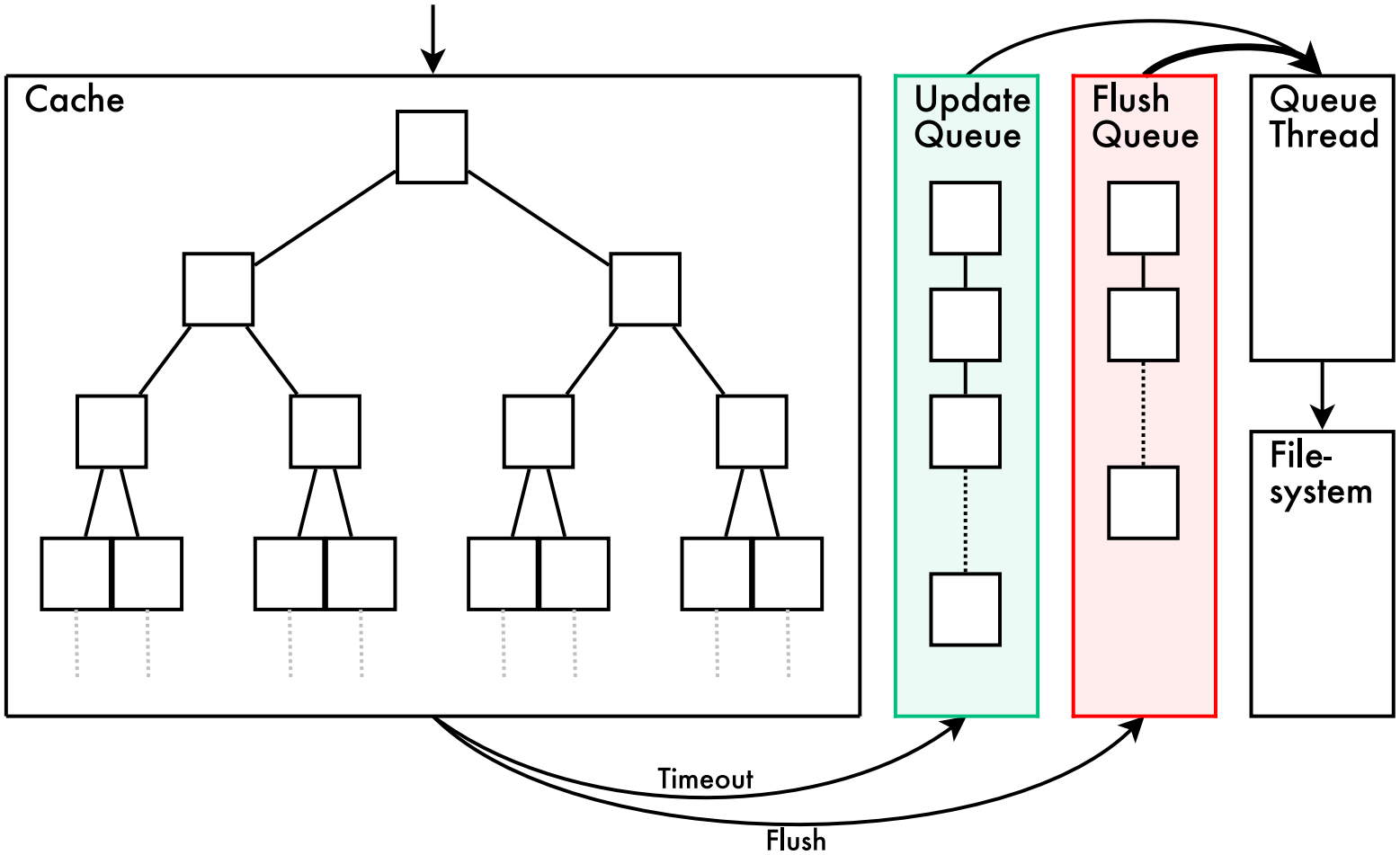
RRDtool-Plugin

Synopsis

```
LoadPlugin "rrdtool"

<Plugin "rrdtool">
  DataDir "/var/lib/collectd/rrd"
  CacheTimeout 300
  WritesPerSecond 30
</Plugin>
```


RRDtool-Plugin



RRDCacheD-Plugin

Allgemeines

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

Perl-Plugin

Allgemeines

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

Perl-Plugin: Beispiel

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


unixsock-Plugin

Allgemeines

- Öffnet einen UNIX-Domain-Socket
- Kennt mehrere Befehle
  (z. B. `PUTVAL`, `FLUSH`, `LISTVAL`)
- Interaktion mit externen Programmen möglich
- `cussh.pl`: *„collectd UNIX socket shell“*

unixsock-Plugin: Beispiel

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


exec-Plugin

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
```


Java-Plugin

Allgemeines

- Integriert eine „Java Virtual Maschine“ (JVM)
- Exportiert die API
  ($\rightarrow$ 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

- **collect**d API nutzen
  C, Perl, Python und Java möglich

Eigene Erweiterungen: Zusammenfassung

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

Eigene Erweiterungen: Zusammenfassung

- **collect**d API nutzen
  C, Perl, Python und Java möglich
- Externe Programme erweitern
  unixsock-Plugin ermöglicht Kommunikation
- Eigenes Programm / Skript schreiben
  $\rightarrow$ exec-Plugin

Über den Tellerrand: Zubehör

- `snmp-probe-host.px`
  Erzeugt semi-automatisch `<Host />`-Blöcke für das SNMP-Plugin
- jcollectd
  Java-Implementierung des Netzwerk-Protokolls *(→ JMX)*
- kcollectd
  KDE-Programm zur Near-Realtime-Anzeige von Graphen
- `Collectd::Unixsock`
  Perl-Modul für die Kommunikation mit dem unixsock-Plugin

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