Watch your elephants

PostgreSQL Performance Analysis using collectd

Sebastian „tokkee“ Harl <sh@teamix.net>

teamix GmbH / **collect**d core team

PGConf.EU 2012
October 24, 2012
Prague

teamix
SOLIDE IT-INFRASTRUKTUR

Disclaimer

WARNING: I'm not a database (performance) expert!

This talk is an overview about a tool that may be used for the purpose of performance analysis.

Solid IT-Infrastructure
Location: Nuremberg, Munich, Frankfurt
`http://teamix.net/`

Open-Source	Monitoring	Network
N-IX	NetApp	Juniper
Riverbed	VMWare	Trainings

collectd Overview
- Overview
- Main Features

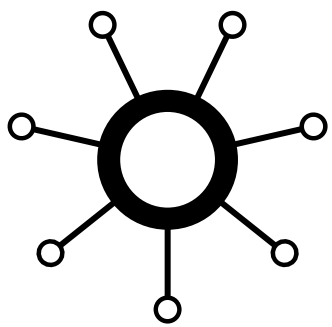
Feature Overview

PostgreSQL Processes

Querying statistics from PostgreSQL

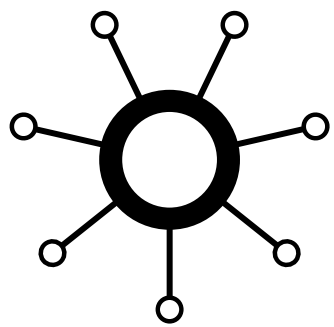
collectd overview

- **collect**d collects performance data of systems
- some (simple) examples:
  - CPU utilization
  - memory utilization
  - network traffic
- **collect**d collects and stores the performance data
- stored data is usually used to generate graphs
- → performance analysis, capacity planing
- not to be confused with *monitoring!*
- Homepage: `http://collectd.org/`



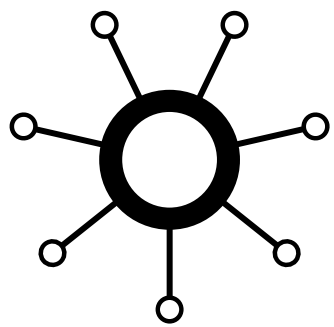
Main features

- daemon
- free software (mostly GPL)
- portable (Linux, *BSD, Solaris, . . .)
- scalable (OpenWrt, . . . , Cluster / Cloud)
- sophisticated network support
- efficient (default resolution: 10 seconds)
- flexible architecture
- modular (more than 100 plugins in Version 5.1)

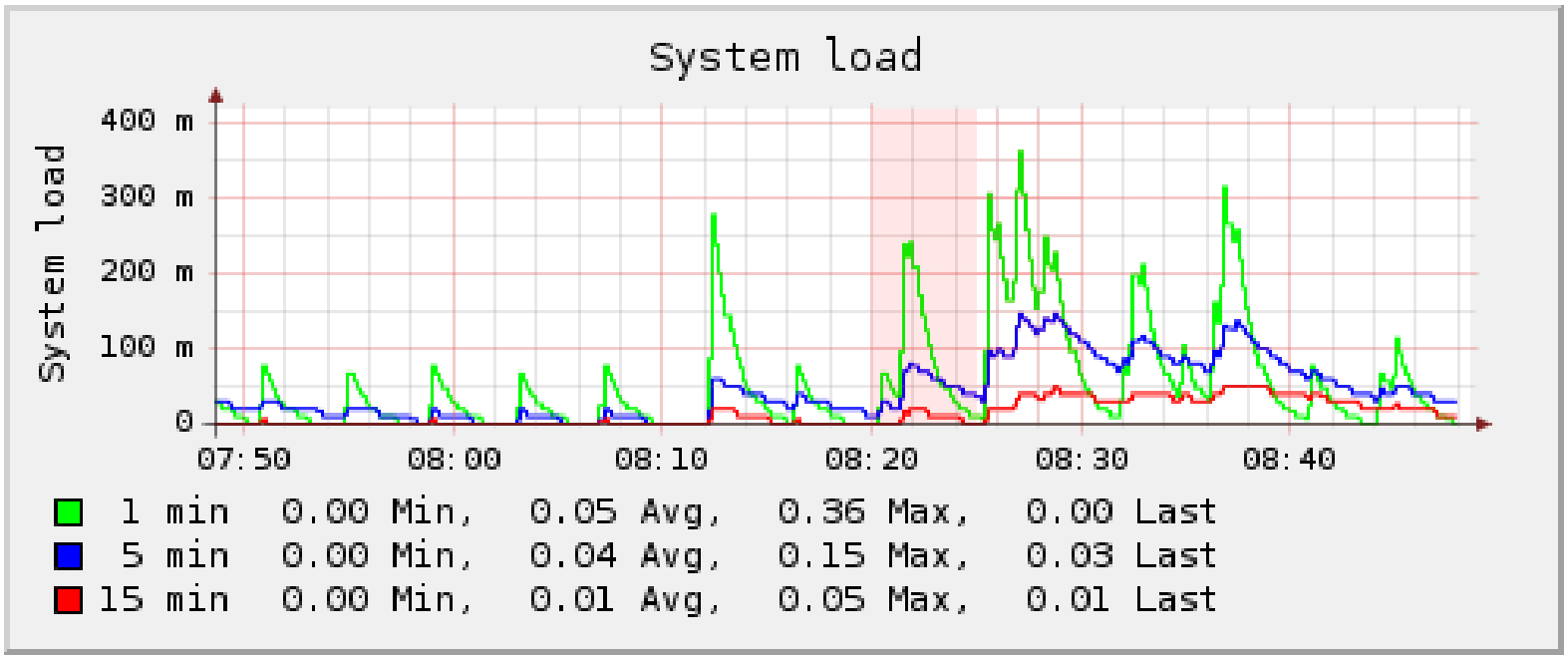


Main features

- daemon
- free software (mostly GPL)
- portable (Linux, *BSD, Solaris, . . .)
- scalable (OpenWrt, . . . , Cluster / Cloud)
- sophisticated network support
- efficient (default resolution: 10 seconds)
- flexible architecture
- modular (more than 100 plugins in Version 5.1)

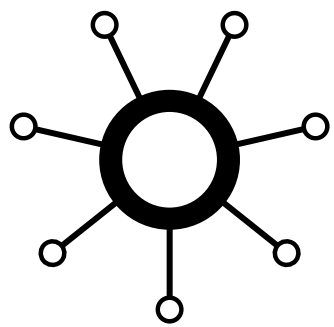


10 seconds resolution



Main features

- daemon
- free software (mostly GPL)
- portable (Linux, *BSD, Solaris, . . .)
- scalable (OpenWrt, . . . , Cluster / Cloud)
- sophisticated network support
- efficient (default resolution: 10 seconds)
- flexible architecture
- modular (more than 100 plugins in version 5.1)



available plugins (version 5.1)

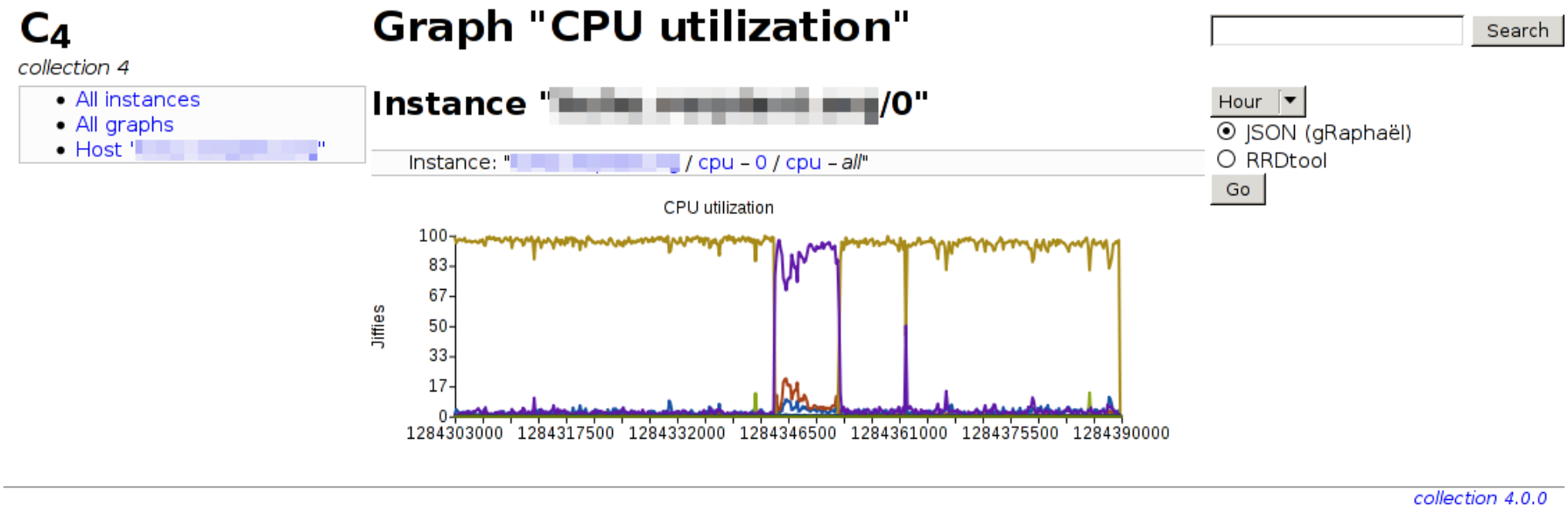
teamix
SOLIDE IT-INFRASTRUKTUR

amqp
apache
apcups
apple_sensors
ascent
battery
bind
conntrack
contextswitch
cpu
cpufreq
csv
curl
curl_json
curl_xml
dbi
df
disk
dns
email
entropy
ethstat
exec
filecount
fscache
GenericJMX.java
gmond
hddtemp
interface
ipmi
iptables
ipvs
irq
java
libvirt
load
logfile
lpar
madwifi
match_empty_counter
match_hashed
match_regex
match_timediff
match_value
mbmon
md
memcachec
memcached
memory
modbus
Monitorus.pm
multimeter
mysql
netapp
netlink
network
nfs
nginx
notify_desktop
notify_email
ntpd
numa
nut
olsrd
onewire
openvpn
OpenVZ.pm
oracle
perl
pinba
ping
postgresql
powerdns
processes
protocols
python
redis
routeros
rrdcached
rrdtool
sensors
serial
snmp
swap
syslog
table
tail
tape
target_notification
target_replace
target_scale
target_set
target_v5upgrade
tcpconns
teamspeak2
ted
thermal
threshold
tokyotyrant
unixsock
uptime
users
uuid
varnish
vmem
vserver
wireless
write_graphite
write_http
write_mongodb
write_redis
xmms
zfs_arc

collectd setup

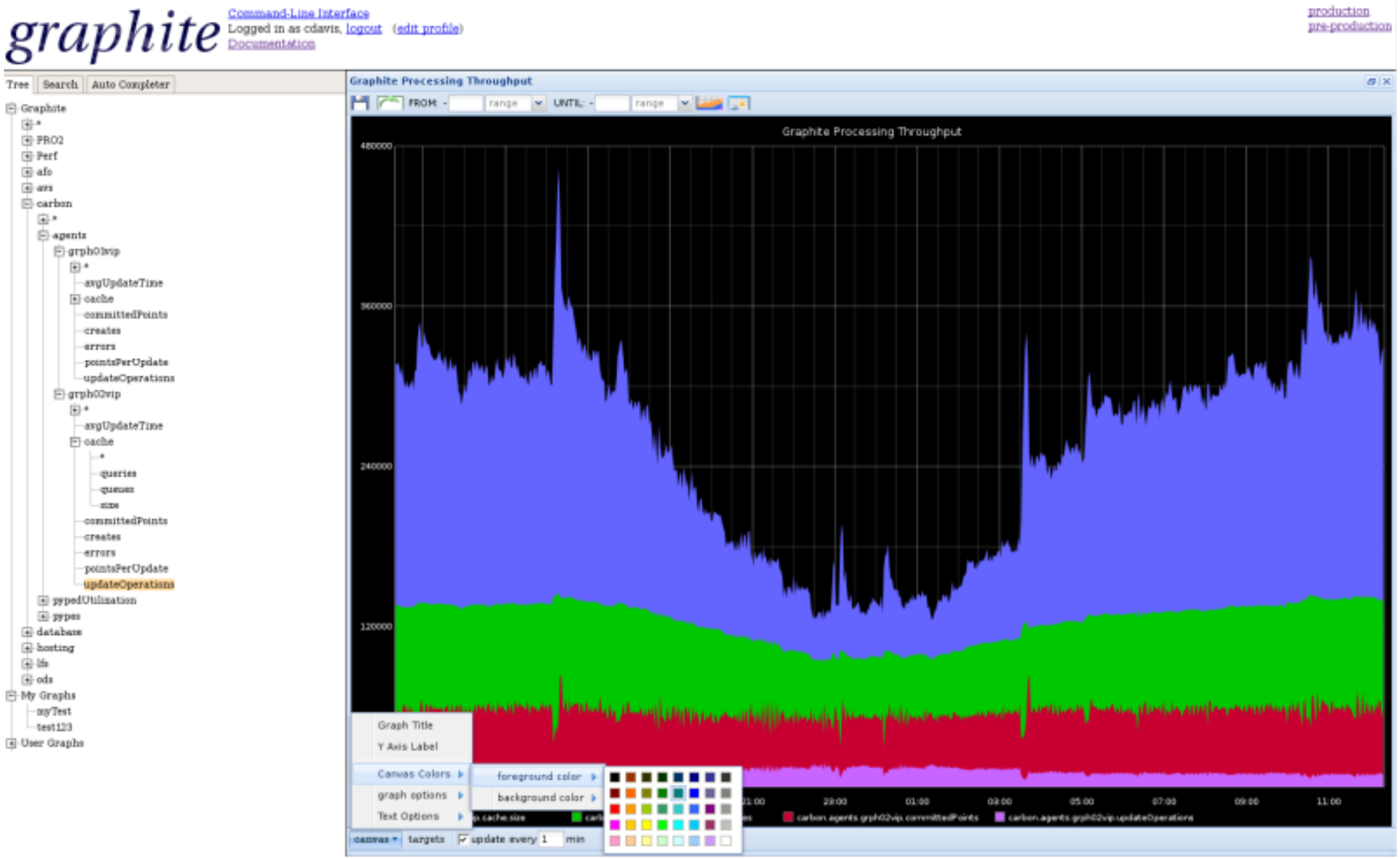
- daemon collects data locally ⇒ runs on every client system (exceptions: SNMP, databases, etc.)
- one or more central servers
- clients push their data to the central servers
- first steps: `install; select plugins; start daemon; enjoy` ;-)

data display: Collection 4 (C4)



- provide collected data through JSON
- different frontends possible
- efficiently handles large amounts of data
- flexible configuration of graphs

data display: Graphite



collectd Overview

Feature Overview

- CPU, memory, network I/O
- Networking Support
- RRDtool Support
- Generic Plugins (Overview)

PostgreSQL Processes

Querying statistics from PostgreSQL

plugin overview

- specialized read plugins
  - CPU, memory, network interfaces, ...
- IO plugins
  - network plugin
  - RRDtool, RRDCacheD
  - Graphite
  - MongoDB, Redis
  - AMQP
- generic plugins
  - SNMP
  - tail
  - PostgreSQL
- filter-chains

configuration synopsis

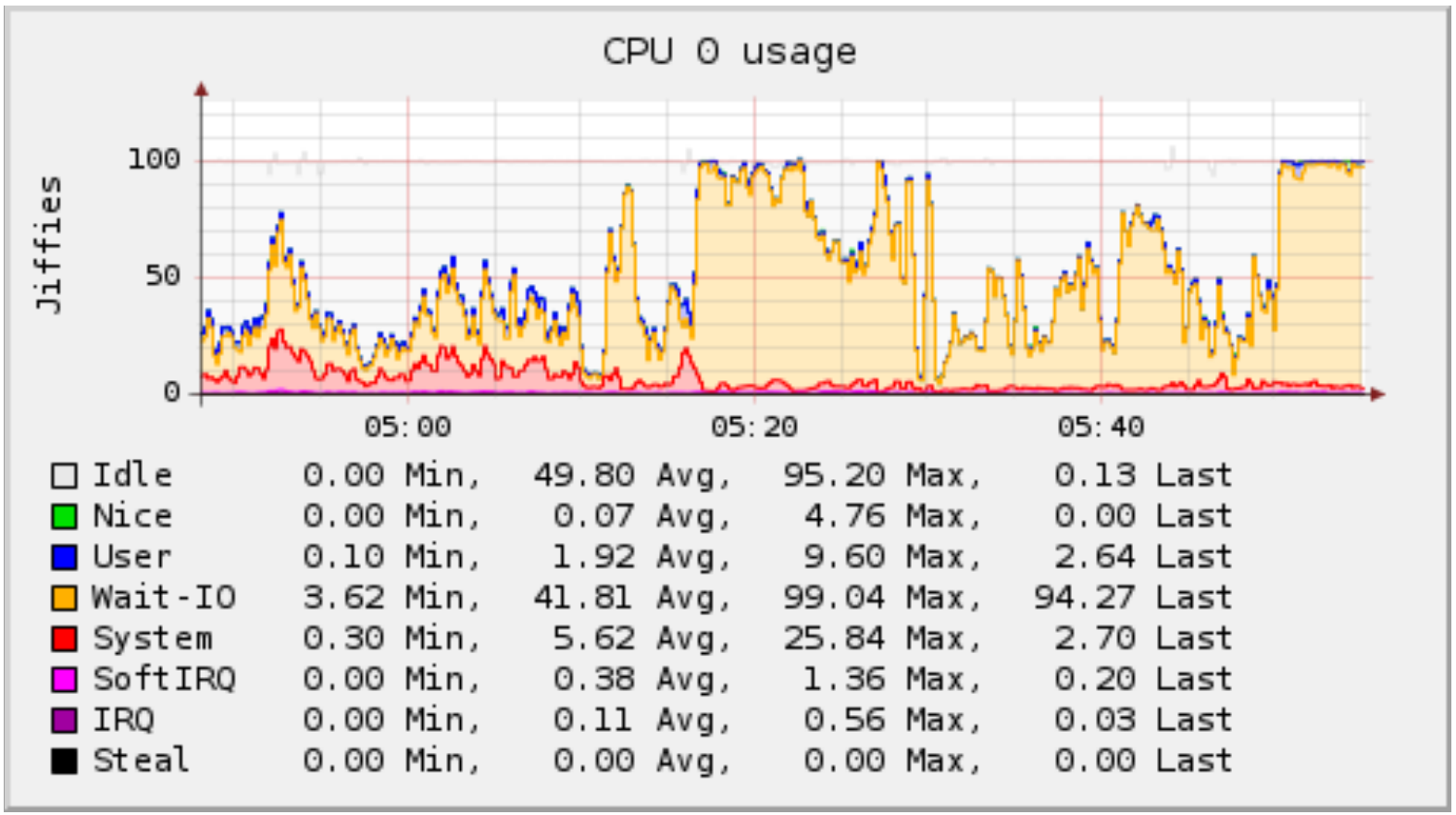
```
LoadPlugin "cpu"
LoadPlugin "memory"
LoadPlugin "interface"
```

configuration synopsis

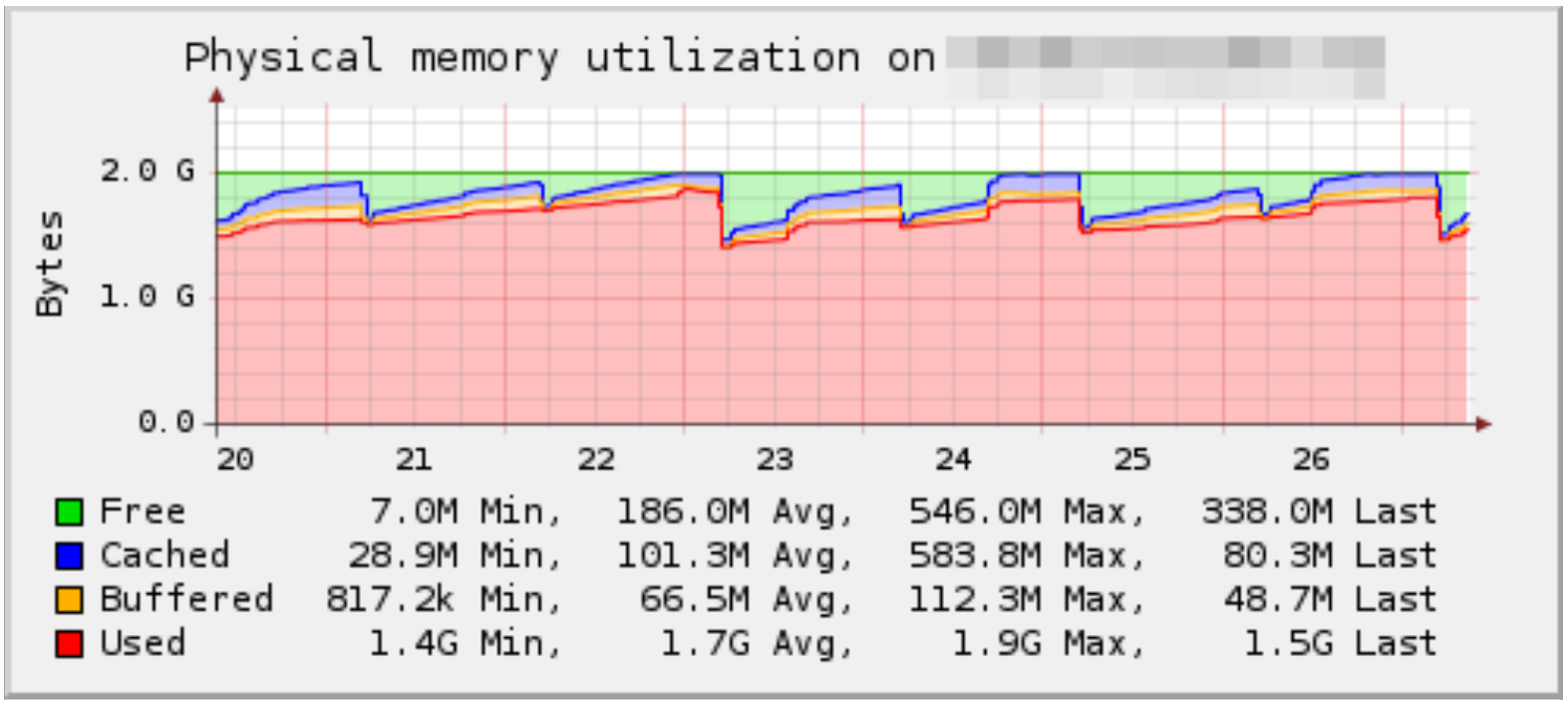
```
LoadPlugin "cpu"
LoadPlugin "memory"
LoadPlugin "interface"

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

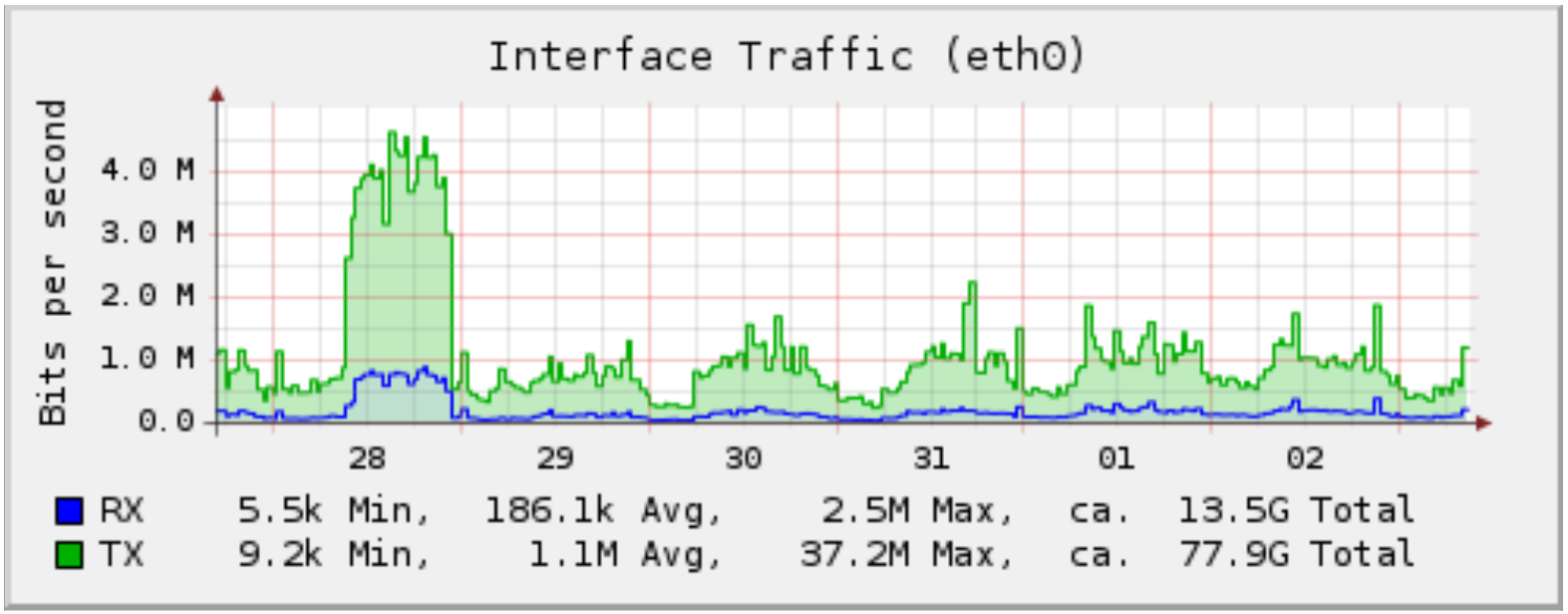
CPU, memory, network I/O



CPU, memory, network I/O



CPU, memory, network I/O



modes of operation

- send data *("client")*
- receive data *("server")*
- forward data *("proxy")*
- Unicast *("point-to-point")*
- Multicast *("point-to-group")*
- IPv4 and IPv6

rule them all

Modes may be mixed arbitrarily.

synopsis: client

```
LoadPlugin "network"

<Plugin "network">
  Server "collectd0.example.com"
  Server "collectd1.example.com"
  Server "ff18::efc0:4a42"
</Plugin>
```

network plugin: server / client

synopsis: server

```
LoadPlugin "network"

<Plugin "network">
  Listen "collectd0.example.com"
  Listen "ff18::efc0:4a42"
</Plugin>
```

synopsis: proxy

```
LoadPlugin "network"

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

RRDtool plugin

- writes data to RRD files **efficiently** → caching
- functionality now also available in RRDtool as stand-alone RRD Caching Daemon (RRDCacheD)

synopsis

```
LoadPlugin "rrdtool"

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

RRDtool plugin (caching)

configuration synopsis

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

  CacheTimeout 3600   # 1 hour
  CacheFlush 86400    # 1 day

  WritesPerSecond 30
</Plugin>
```

- `FLUSH` command allows for graphing of current values

generic plugins (overview)

- idea: generic approaches rather than specialized solutions
- $\rightarrow$ user configuration determines behavior
- $\Rightarrow$ new equipment does not require a new version of **collect**d
- examples: SNMP, tail, curl, DBI, PostgreSQL

collectd Overview

Feature Overview

PostgreSQL Processes

Querying statistics from PostgreSQL

PostgreSQL Processes

```
% ps ax | grep postgres
20177 ?  S    0:05 /usr/lib/postgresql/9.1/bin/postgres
    -D /var/lib/postgresql/9.1/main
    -c config_file=/etc/postgresql/9.1/main/postgresql.conf
20183 ?  Ss   0:09 postgres: writer process
20184 ?  Ss   0:05 postgres: wal writer process
20185 ?  Ss   0:04 postgres: autovacuum launcher process
20186 ?  Ss   0:13 postgres: stats collector process
20312 ?  Ss   2:04 postgres: collectd mail 127.0.0.1(33027) idle
```

- processes handling client connections:
  `postgres: `*`user database host activity`*

The collectd `processes` plugin

- The `processes` plugin collects various information about (groups of) processes
  - RSS and VM size
  - user and system time
  - number of page-faults
  - I/O estimates
- processes are selected either by process name or by regex of it's command line (`/proc/cmdline` on Linux)

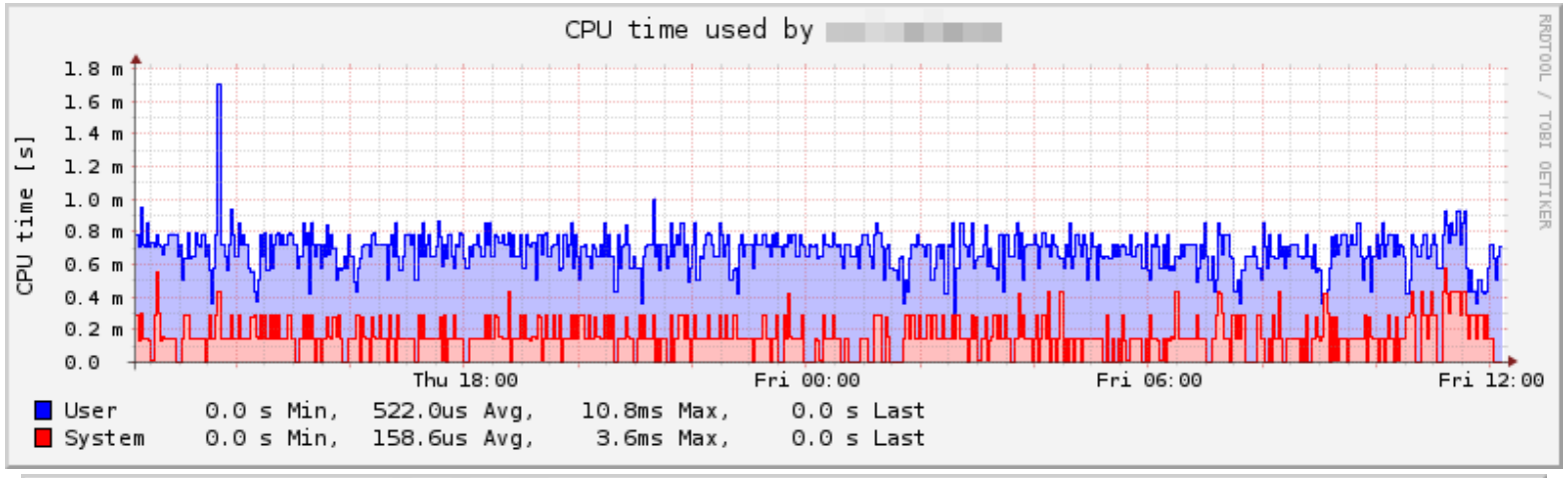
The processes plugin: configuration

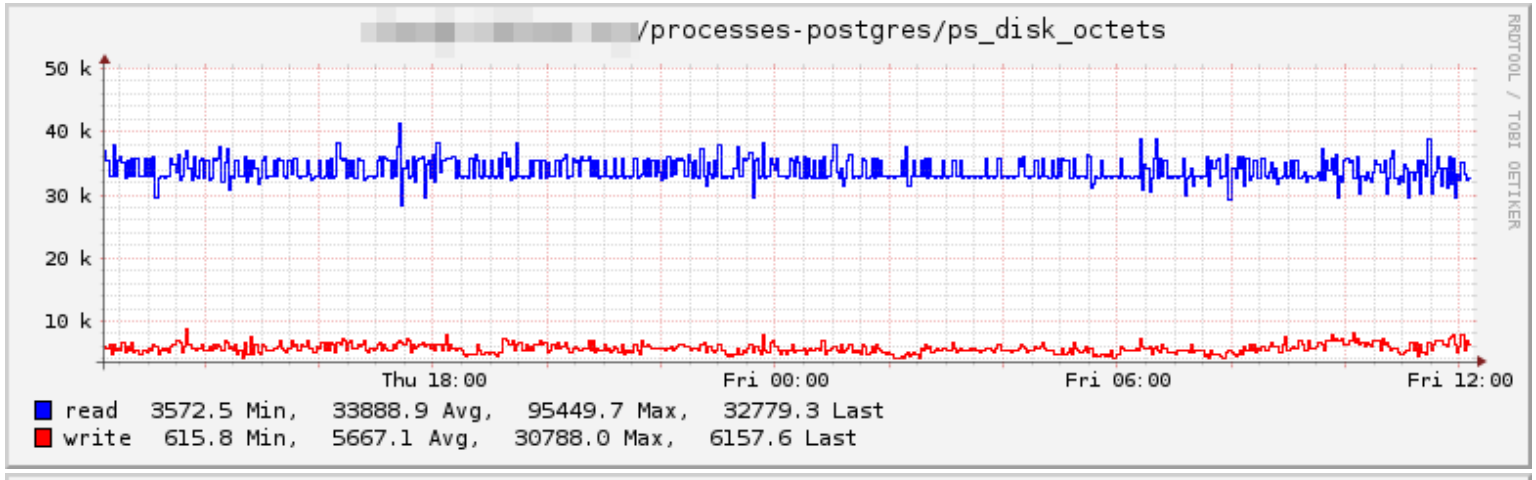
teamix
SOLIDE IT-INFRASTRUKTUR

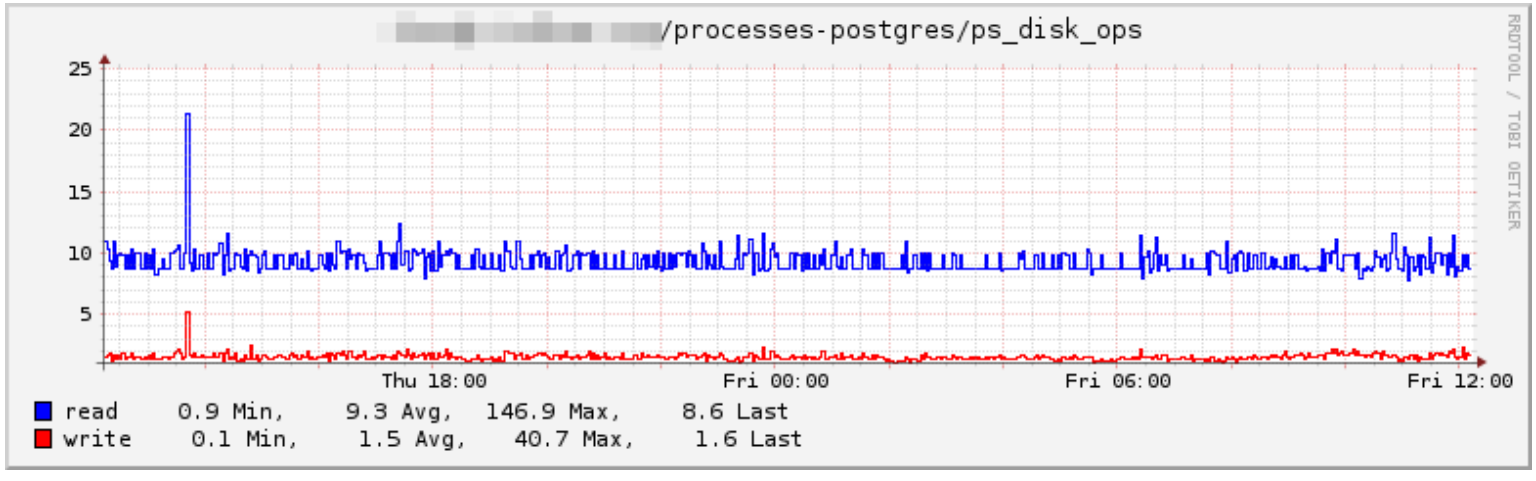
collectd.conf

```
<Plugin "processes">
    ProcessMatch pg_writer "postgres:.writer.process"
    ProcessMatch pg_wal_writer "postgres:.wal.writer.process"
    ProcessMatch pg_autovacuum "postgres:.*autovacuum"
    ProcessMatch pg_stats_collector \
        "postgres:.stats.collector.process"
    # database connections by 'user'
    ProcessMatch pg_user_mail "postgres:.user"
    # database connections to database 'mail'
    ProcessMatch pg_db_mail "postgres:.[A-Za-z0-9]+.mail"
</Plugin>
```

(versions before 5.0.1 did not support whitespace in regexes)


CPU time used by
CPU time [s]
1.8 m
1.6 m
1.4 m
1.2 m
1.0 m
0.8 m
0.6 m
0.4 m
0.2 m
0.0
Thu 18:00
Fri 00:00
Fri 06:00
Fri 12:00
RRDTOOL / TOBI OETIKER
User 0.0 s Min, 522.0us Avg, 10.8ms Max, 0.0 s Last
System 0.0 s Min, 158.6us Avg, 3.6ms Max, 0.0 s Last


/processes-postgres/ps_disk_octets
50 k
40 k
30 k
20 k
10 k
Thu 18:00
Fri 00:00
Fri 06:00
Fri 12:00
RRDTOOL / TOBI OETIKER
read 3572.5 Min, 33888.9 Avg, 95449.7 Max, 32779.3 Last
write 615.8 Min, 5667.1 Avg, 30788.0 Max, 6157.6 Last


/processes-postgres/ps_disk_ops
25
20
15
10
5
Thu 18:00
Fri 00:00
Fri 06:00
Fri 12:00
RRDTOOL / TOBI OETIKER
read 0.9 Min, 9.3 Avg, 146.9 Max, 8.6 Last
write 0.1 Min, 1.5 Avg, 40.7 Max, 1.6 Last

collectd Overview

Feature Overview

PostgreSQL Processes

Querying statistics from PostgreSQL

- The PostgreSQL statistics collector
- The collectd `postgresql` plugin

The PostgreSQL statistics collector

teamix
SOLIDE IT-INFRASTRUKTUR

postgresql.conf

```
# currently running command
track_activities = on
# access to tables and indices
track_counts = on
# user-defined functions
track_functions = none  # none, pl, all
```

Storing statistics on, for example, flash storage:

```
stats_temp_directory = '/mnt/flash/pg_stat_tmp'
```

Collecting statistics

- server processes submit statistics before going idle
- collector generates report each `PGSTAT_STAT_INTERVAL` milliseconds
- during each transaction, a snapshot of the report will be used → see `pg_stat_clear_snapshot()`

Querying statistics

- some predefined views
  - `pg_stat_bgwriter`
  - `pg_stat_database`
  - `pg_stat_all_indexes`
  - `pg_statio_all_tables`
  - many more (see table 27.1 in the documentation)
- also there are various functions to query single values

Querying statistics: example

```
sh=# select datname, numbackends,
sh-# xact_commit, xact_rollback
sh-# from pg_stat_database;
  datname  | numbackends | xact_commit | xact_rollback
-----------+-------------+-------------+---------------
 template1 |           0 |           0 |             0
 template0 |           0 |           0 |             0
 postgres  |           0 |         611 |             0
 sh        |           1 |         661 |            12
(4 rows)
```

The collectd `postgresql` plugin

- generic plugin which collects arbitrary (numeric) values from a database
- by default, queries various values from the statistics collector
- configuration has two parts
  - SQL query and specifications how to interpret the values
  - database connection plus queries assigned to it

The collectd naming scheme

- each data-set uses a unique identifier
  - hostname
  - plugin name
  - plugin instance (optional)
  - type
  - type instance (optional)
- `hostname/plugin[-instance]/type[-instance]`
- the type specifies how **collect**d is supposed to handle the data-set (cf. RRDtool's data-source types)
- any type needs to be pre-defined (`types.db(5)`)

- example: `server1.ex.com/cpu-0/cpu-idle`

The postgresql plugin: Query Definition

collectd.conf

```
<Plugin postgresql>
    <Query disk_usage>
        Statement "SELECT pg_database_size($1) AS size;"
        Param database

        <Result>
            Type pg_db_size
            ValuesFrom "size"
        </Result>
    </Query>
</Plugin>
```

The `postgresql` plugin: DB Definition

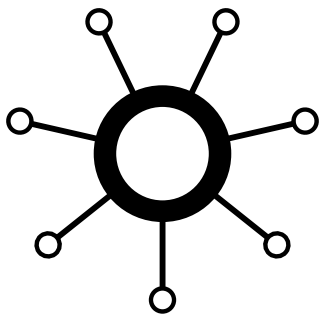
teamix
SOLIDE IT-INFRASTRUKTUR

collectd.conf

```
<Plugin postgresql>
    <Database mail>
        Host "db.ex.com"
        User "user"
        Password "secret"
        Query disk_usage
        Query disk_io
    </Database>
</Plugin>
```

Thanks for your attention!

Any questions?



Contact:
Sebastian „tokkee“ Harl
teamix GmbH, Nürnberg
<sh@teamix.net>

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

http://github.com/collectd/collectd — http://github.com/tokkee/collectd