Anomaly detection and trend analysis of critical system states based on Nagios payload

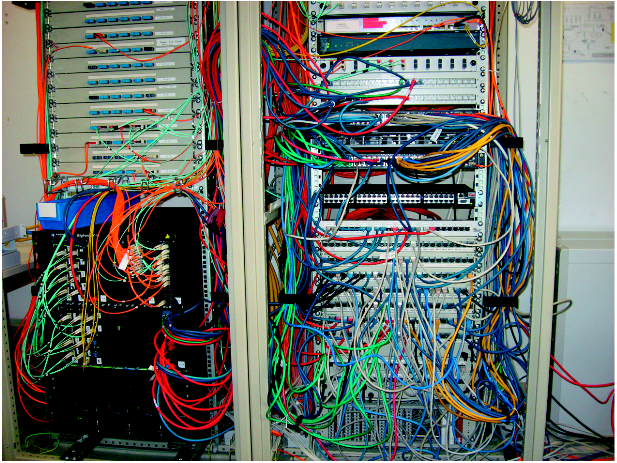
Daniel Borkmann <dborkmann@acm.org>

(Open Source Monitoring Conference 2009)

Leipzig University of Applied Science, Faculty of Computer Science, Mathematics, and Natural Sciences
Max Planck Institute for Human Cognitive and Brain Sciences, IT Department

October 28, 2009

Motivation



- **Worst-Case:** User detects failure within system and reports to IT-Hotline (*frustration on both sides*)

- **Best-Case:** Failure is detected and fixed before user notices anything (*frustration on only one side*)

We face the situation ...

- Nagios can only detect failures via static thresholds[1]
- Is perfect for requesting boolean values (e.g. host is up or not)
- But not so much in combination with numerical data from certain processes ...
- ... process behaviour changes over the time, our static thresholds do not
- ... what if behaviour is anomalous within thresholds?
  - Network traffic,
  - # Mail users,
  - Temperature sensors, ...

[1] OK, WARNING, CRITICAL, UNKNOWN

We face the situation ...

- Can we predict the future?[2]
- Nagios has no possibility for payload extrapolation (trend)
- Possible Use Cases
  - Software licences (e.g. Matlab),
  - Network: volume-based restrictions,
  - HD capacity,
  - Stock quotes ;)

[2] ... rhetorical question

1. Data collection
   - Sniffing bytes over the net ... netsniff-ng
   - Integration into Nagios
2. Data analysis
   - Assumptions about our data ...
   - Anomaly detection techniques
     - Holt-Winters-Forecasting (and some improvements)
     - Clusteranalysis approach
     - Interface to NagiosGrapher
   - Trend analysis
     - Levenberg-Marquardt
     - REA-Framework
3. Links & Q+A

Data collection

Data collection

- For our analysis we need some numerical data ...
- ... as an example, we fetch network packets and generate statistics
- Let's have a short look ...

Sniffing bytes over the net ...
netsniff-ng

tcpdump, libpcap

- Lets evaluate what could be used ...
- `tcpdump` is a userspace network sniffer based on libpcap
- `libpcap`: libpcap is a system-independent interface for user-level packet capture. libpcap provides a portable framework for low-level network monitoring. Applications include network statistics collection, security monitoring, network debugging, etc.[3]

[3] http://sourceforge.net/projects/libpcap/

tcpdump, libpcap

- `libpcap` programming isn't hard, so here we go ... writing a sniffer for Nagios based on `libpcap`
- Sounds great, doesn't it?

Hochschule
für Technik, Wirtschaft
und Kultur Leipzig (FH)
University of Applied Sciences

tcpdump, libpcap

- `tcpdump -i eth0 -n arp`
- `strace` clarifies our question ...

```
recvfrom(3, "\377\377\377\377\377\377\0\32ME|\211\10\6\0\1\10\0\6\4"..
  ., 96, MSG_TRUNC, {sa_family=AF_PACKET, proto=0x806, if50, pkttype=
  PACKET_BROADCAST, addr(6)={1, 001a4d457c89}, [18]) = 60
ioctl(3, SIOCGSTAMP, 0xbfb43e70)        = 0
write(1, "16:14:50.538813 arp who-has 10.0"..., 5516:14:50.538813 arp
  who-has 10.0.53.26 tell 10.0.63.10) = 55
recvfrom(3, "\377\377\377\377\377\377\0\4v\243\330\242\10\6\0\1\10\0"..
  ., 96, MSG_TRUNC, {sa_family=AF_PACKET, proto=0x806, if50, pkttype=
  PACKET_BROADCAST, addr(6)={1, 000476a3d8a2}, [18]) = 60
ioctl(3, SIOCGSTAMP, 0xbfb43e70)        = 0
write(1, "16:14:50.624868 arp who-has 10.0"..., 5616:14:50.624868 arp
  who-has 10.0.63.53 tell 10.0.54.184) = 56
```

tcpdump, libpcap

- For every incoming frame a *recvfrom*-Syscall is executed
- What does that mean?
  - `recvfrom(...)`
  - Buffer will be copied from Userspace to Kernelspace[4]
  - Context switch (done by Scheduler / Dispatcher)
  - Buffer will be copied from Kernelspace back to Userspace[5]
  - Context switch (done by Scheduler / Dispatcher)
- ... very time intensive if done for each frame!
- ... possible frame drops for socket during high traffic

[4] `copy_from_user()`

[5] `copy_to_user()`

netsniff-ng

- High performance network sniffer
- Consists of
  - *netsniff-ng*
  - *check_packets* (client for Nagios)

netsniff-ng features

- The sniffer itself ...
- Runs in promiscuous mode
- Bypasses the complete network stack
- Uses Kernelspace Berkeley Packet Filter (`BPF`)
- Allocates 128 MB or less (probing) Kernelspace Receive Ring (`RX_RING`)
- Ring is Memory-Mapped into Userspace (so no Syscalls like `recvfrom()` needed → Zero-Copy)
- Branchfree critical path (so we won't smash the Pipeline)
- Tested on Gigabit without packet loss
- Can be run as Sysdaemon (silent, creates `UDS` Server for communication) or in foreground

HTWK Hochschule für Technik, Wirtschaft und Kultur Leipzig (FH) University of Applied Sciences

MAX-PLANCK-GESELLSCHAFT

netsniff-ng features

- `netsniff-ng -d eth0 -f /etc/netsniff-ng/rules/arp.bpf -C`
- strace again, looks better now ...

```
rt_sigprocmask(SIG_BLOCK, [USR1 ALRM], NULL, 8) = 0
rt_sigprocmask(SIG_UNBLOCK, [USR1 ALRM], NULL, 8) = 0
poll([{fd=3, events=POLLIN|POLLERR, revents=POLLIN}], 1, -1) = 1
write(2, "I: ", 3I: )                          = 3
write(2, "60 bytes from 00:1a:4d:45:7c:89 "..., 5360 bytes from 00:
  1a:4d:45:7c:89 to ff:ff:ff:ff:ff:ff) = 53
rt_sigprocmask(SIG_BLOCK, [USR1 ALRM], NULL, 8) = 0
rt_sigprocmask(SIG_UNBLOCK, [USR1 ALRM], NULL, 8) = 0
poll([{fd=3, events=POLLIN|POLLERR, revents=POLLIN}], 1, -1) = 1
write(2, "I: ", 3I: )                          = 3
write(2, "60 bytes from 00:10:5a:d8:9a:a4 "..., 5360 bytes from 00:10:
  5a:d8:9a:a4 to ff:ff:ff:ff:ff:ff) = 53
```

netsniff-ng features

- `netsniff-ng -d eth0 -f /etc/netsniff-ng/rules/icmp.bpf`
- ICMP-Flooding; only 0-3% CPU usage of netsniff-ng during tests

```
I: elapsed time: 0 d, 0 h, 4 min, 45 s
I: -----------+------------------+------------------+------------------
I:            |  per sec         |  per min         |  total
I: -----------+------------------+------------------+------------------
I:   frames   |            80201 |          4696273 |         20149411
I: -----------+------------------+------------------+------------------
I:   in B     |        119622775 |       7003546464 |      30048067864
I:   in KB    |           116819 |          6839400 |         29343816
I:   in MB    |              114 |             6679 |            28656
I:   in GB    |                0 |                6 |               27
I: -----------+------------------+------------------+------------------
[...]
I: 23724545 frames incoming
I: 23724545 frames passed filter
I: 0 frames failed filter (due to out of space)
I: captured frames: 23724545, captured bytes:
   35377999772 [34548827 KB, 33739 MB, 32 GB]
```

check_packets features

- Is a Unix Domain Socket Client for netsniff-ng
- Fetches collected network statistics at runtime via `UDS` inode
- `-n` option for creating Nagios one-liner → Performance data
- Simple Nagios integration with `NRPE` or `check_by_ssh`

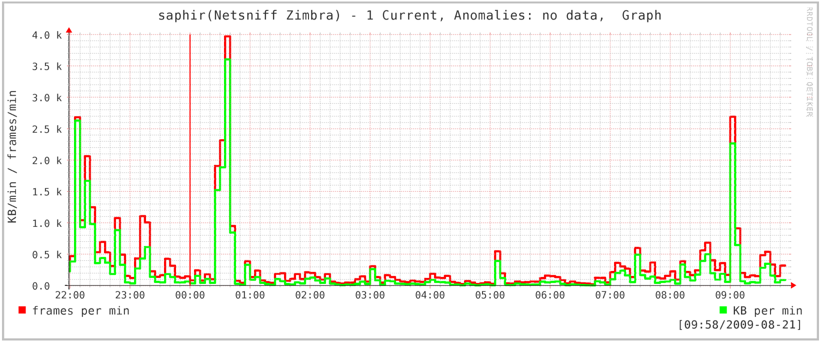
Integration into Nagios

Integration into Nagios

- Runs 24x7 on MPI router (DMZ ⇔ Router ⇔ Internet)
- 6 instances running with following BPFs:
  - All
  - HTTP
  - Broadcast
  - SSH
  - Webserver
  - Mailserver traffic
- Data will be passed via check_by_ssh[6] (Port 22) through our internal Firewall to our Nagios server
- Visualization realized by NagiosGrapher via Round Robin Databases

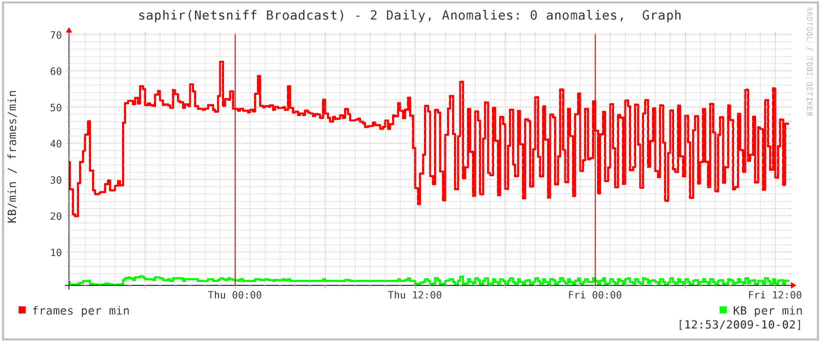
[6] NRPE inappropriate in our case

Some results



({Sec|}IMAP, {Sec|}SMTP, POP3, HTTP traffic to our Zimbra mailserver)

Some results



(Broadcast traffic)

Data analysis

Assumptions about our data ...

Assumptions

- Generally our monitored processes equal Stochastic processes
- Incoming anomalies itself are regarded as Poisson processes
- A single data point has ...
  - Baseline („intercept“) or irregular component
  - Linear trend („slope“) component
  - Seasonal trend (at least one)
- We assume components are additive

Anomaly detection techniques

Holt-Winters-Forecasting (and some improvements)

Holt-Winters-Forecasting

- Based on exponential smoothing (recent data have higher weights than older ones): $\hat{y}_{t+1} = \alpha y_t + (1-\alpha)\hat{y}_t$ with $0 < \alpha < 1$
- Decomposition of time series into components with triple exponential smoothing technique
- Multiplicative and additive versions
- Forecast of the near future (with $+/-$ deviation), check if actual value falls into expected interval (if not: anomaly after breaking thresholds)

Holt-Winters-Forecasting

What the math model looks like:

- **Prediction:**

  $\hat{y}_{t+1} = a_t + b_t + c_{t+1-m}$

  $$\begin{aligned} a_t &= \alpha(y_t - c_{t-m}) + (1-\alpha)(a_{t-1} + b_{t-1}) \\ b_t &= \beta(a_t - a_{t-1}) + (1-\beta)b_{t-1} \\ c_t &= \gamma(y_t - a_t) + (1-\gamma)c_{t-m} \end{aligned}$$

  Baseline,

  Linear trend,

  Seasonal trend

- **Confidence band coefficient:**

  $d_t = \gamma|y_t - \hat{y}_t| + (1-\gamma)d_{t-m}$

  $0 < \alpha, \beta, \gamma < 1$

  m: Len season

- **Confidence band itself:**

  $\mathbb{I}(\hat{y}_t - \delta_- * d_{t-m}, \hat{y}_t + \delta_+ * d_{t-m}) \quad 2 < \delta_-, \delta_+ < 3$

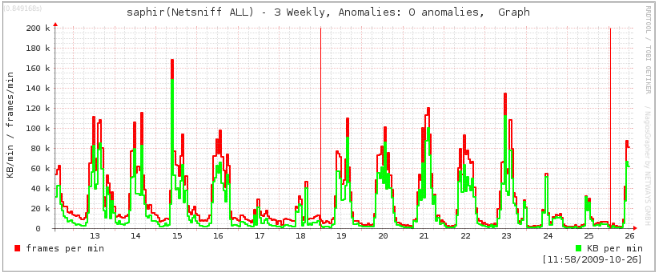
- **Sliding window as threshold for anomalies**

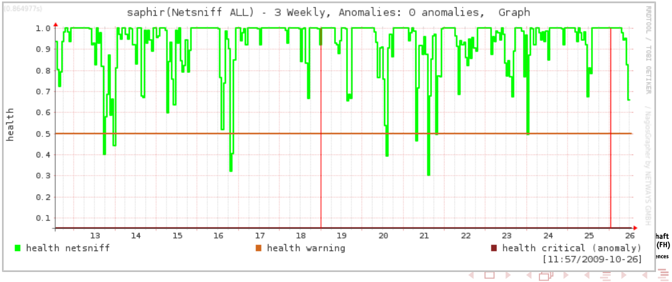
Holt-Winters-Forecasting

Possible improvements:

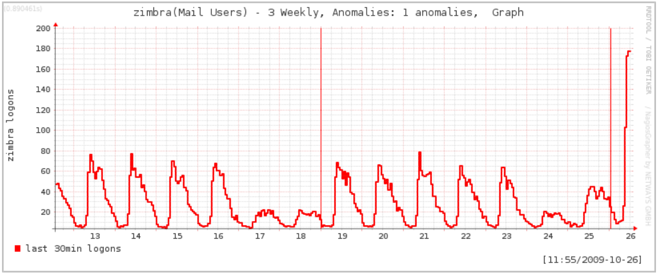
- Double seasonality (periods of week, day)
  - Up to now only single seasonality (day) → parameter disturbances of passing business days into non-business days and vice versa
- Parameter stabilization via Kalman filter [Gelper et al. 2007]
  - Will make parameters more robust towards interferences of anomalies
- Miller extension for improving prediction of low traffic values [Miller 2007]
  - Longterm zero-values decreases confidence band to minimal width → slight payload variance will result in anomaly even though it isn't

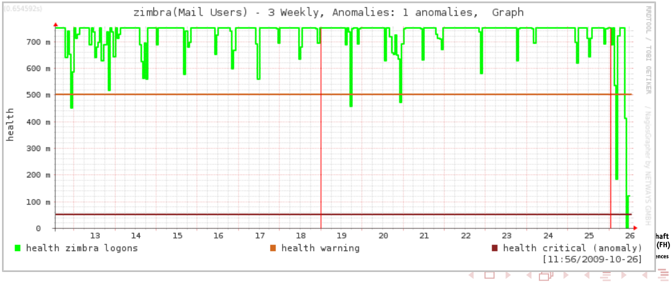
Holt-Winters-Forecasting, Results (all traffic)





Holt-Winters-Forecasting, Results (active Zimbra users)



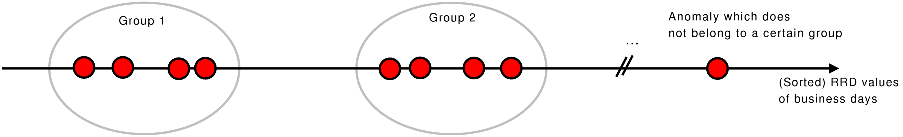


Clusteranalysis approach

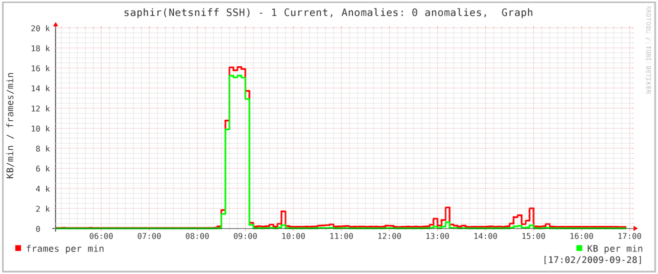
Clusteranalysis approach

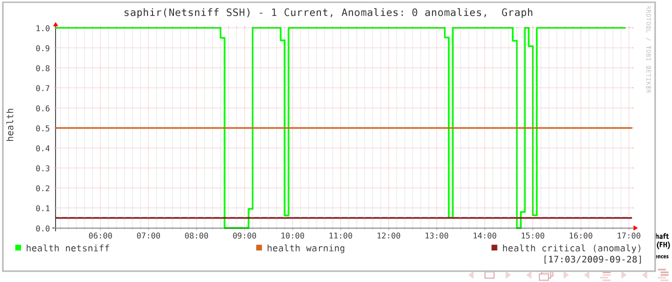
- Idea: one-dimensional grouping of data points to interval bands in order to separate anomalous from normal behaviour
- Segregation of business days and non-business days
- Grouping threshold set by standard deviation of Poisson Process (adapted with coeffcient)
- Groups consist of at least 25 per cent of data points (as per definition)

Clusteranalysis approach

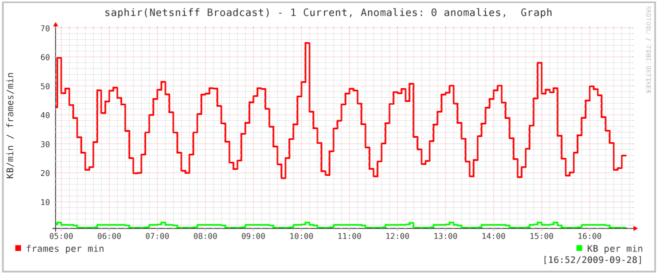


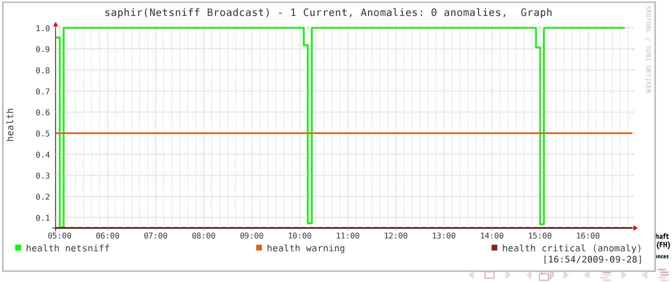
Clusteranalysis, Results (SSH traffic)





Clusteranalysis, Results (Broadcast traffic)





Interface to NagiosGrapher

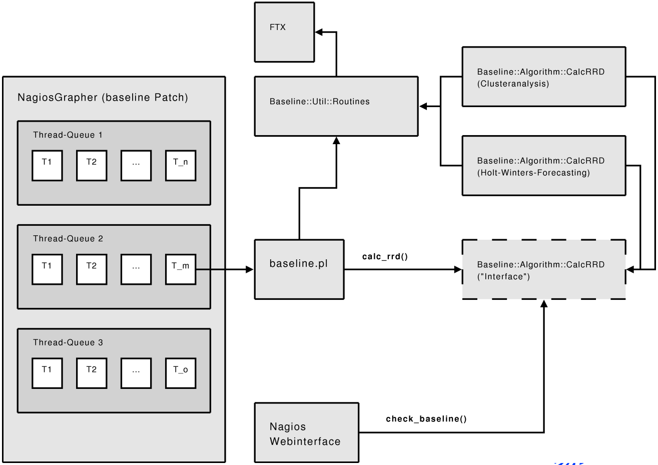
Interface to NagiosGrapher

- Three thread queues filled during NGs **collect2.pl** runtime (*patched against nagiosgrapher (1.6.1rc5-6), Debian Lenny (stable)*)
- Detached threads check for anomalies according to given algorithm (start **baseline.pl** delegate after RRD update)
- Module **Baseline::Algorithm::CalcRRD** with **calc_rrd()** and **check_baseline()** implemented for specific algorithm

Interface to NagiosGrapher

- Algorithms swappable during runtime without payload loss!
- Anomaly information on Nagios ‘Service Detail‘ page
- **Baseline::Util::Routines** contain helper routines for CalcRRD implementation (as **fetch_stepping()**, **fetch_lastupdate()**, **fetch_table()**, ...)

Interface to NagiosGrapher

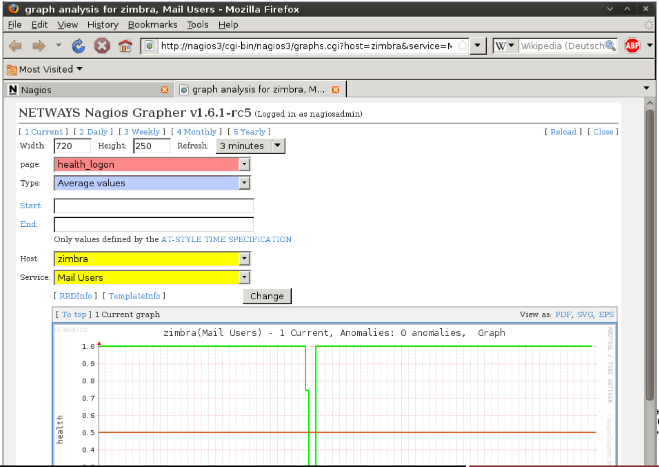

FTX
NagiosGrapher (baseline Patch)
Thread-Queue 1
T1
T2
...
T_n
Thread-Queue 2
T1
T2
...
T_m
Thread-Queue 3
T1
T2
...
T_o
Baseline::Util::Routines
Baseline::Algorithm::CalcRRD (Clusteranalysis)
Baseline::Algorithm::CalcRRD (Holt-Winters-Forecasting)
baseline.pl
calc_rrd()
Baseline::Algorithm::CalcRRD ("Interface")
Nagios Webinterface
check_baseline()

NagiosGrapher GUI, config integration

```
define ngraph{
        service_name                    Netsniff *
        graph_log_regex                 .*per\sminute:\s+([0-9\.]+)\sframes
        graph_value                     pktfr
        graph_units                     frames/min
        graph_legend                    frames per min
        rrd_plottype                    LINE2
        rrd_color                       FF0000
        hide                            no
        page                            data
}
define ngraph{
        service_name                    Netsniff *
        type                            BASELINE      <-- unknown, will be ignored within perfdata parsing
        graph_value                     pktfrbase
        graph_units                     health
        graph_legend                    health temperature
        rrd_plottype                    LINE2
        rrd_color                       00ff00
        hide                            no
        page                            health_frames
}
```

Interface to NagiosGrapher

- Two (or more) pages per Service (‘data‘, ‘health_*‘)
- Actual data (payload, anomaly) still separated


graph analysis for zimbra, Mail Users - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://nagios3/cgi-bin/nagios3/graphs.cgi?host=zimbra&service=M
Wikipedia (Deutsch
Most Visited
Nagios
graph analysis for zimbra, M...

NETWAYS Nagios Grapher v1.6.1-rc5 (Logged in as nagiosadmin)

[1 Current] [2 Daily] [3 Weekly] [4 Monthly] [5 Yearly]
[Reload] [Close]

Width: 720 Height: 250 Refresh: 3 minutes

page: health_logon

Type: Average values

Start:

End:

Only values defined by the AT-STYLE TIME SPECIFICATION

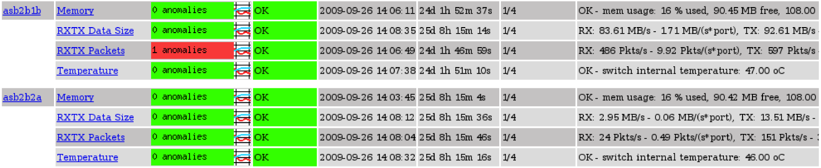
Host: zimbra

Service: Mail Users

[RRDInfo] [TemplateInfo]
Change
[To top] 1 Current graph
View as: PDF, SVG, EPS
(0.460871s)
zimbra(Mail Users) - 1 Current, Anomalies: 0 anomalies, Graph
1.0
0.9
0.8
0.7
0.6
0.5
0.4
health
RRDTOOL / TOBI OETIKER
/ NagiosGrapher

Interface to NagiosGrapher

- Integration into Nagios ‘Service Detail‘ page
- **check_baseline()** part of **Baseline::Algorithm::CalcRRD** called



asb2b1b	Memory	0 anomalies		OK	2009-09-26 14:06:11	24d 1h 52m 37s	1/4	OK - mem usage: 16 % used, 90.45 MB free, 108.00
	RXTX Data Size	0 anomalies		OK	2009-09-26 14:08:35	25d 8h 15m 14s	1/4	RX: 83.61 MB/s - 1.71 MB/(s*port), TX: 92.61 MB/s -
	RXTX Packets	1 anomalies		OK	2009-09-26 14:06:49	24d 1h 46m 59s	1/4	RX: 486 Pkts/s - 9.92 Pkts/(s*port), TX: 597 Pkts/s -
	Temperature	0 anomalies		OK	2009-09-26 14:07:38	24d 1h 51m 10s	1/4	OK - switch internal temperature: 47.00 oC
asb2b2a	Memory	0 anomalies		OK	2009-09-26 14:03:45	25d 8h 15m 4s	1/4	OK - mem usage: 16 % used, 90.42 MB free, 108.00
	RXTX Data Size	0 anomalies		OK	2009-09-26 14:08:12	25d 8h 15m 36s	1/4	RX: 2.95 MB/s - 0.06 MB/(s*port), TX: 13.51 MB/s -
	RXTX Packets	0 anomalies		OK	2009-09-26 14:08:04	25d 8h 15m 46s	1/4	RX: 24 Pkts/s - 0.49 Pkts/(s*port), TX: 151 Pkts/s -
	Temperature	0 anomalies		OK	2009-09-26 14:08:32	25d 8h 15m 16s	1/4	OK - switch internal temperature: 46.00 oC

Trend analysis

Levenberg-Marquardt

Levenberg-Marquardt

- Non-linear fitting algorithm
- Fits a math model (e.g. $f(x) := acos(bx) + bsin(ax) + c$) into a series of data points with minimal residuals (parameter search → $\{a, b, c\}$)
- More ‘stable‘ in finding local minimum (even with a bad chosen start vector) than Gauss-Newton method (with ‘lambda decay‘)
- Non-linearity approximated by iterative solving of linear equation systems (Taylor series)
- Note: quality of the fit depends on your defined model!

HTWK Hochschule für Technik, Wirtschaft und Kultur Leipzig (FH) University of Applied Sciences

MAX-PLANCK-GESELLSCHAFT

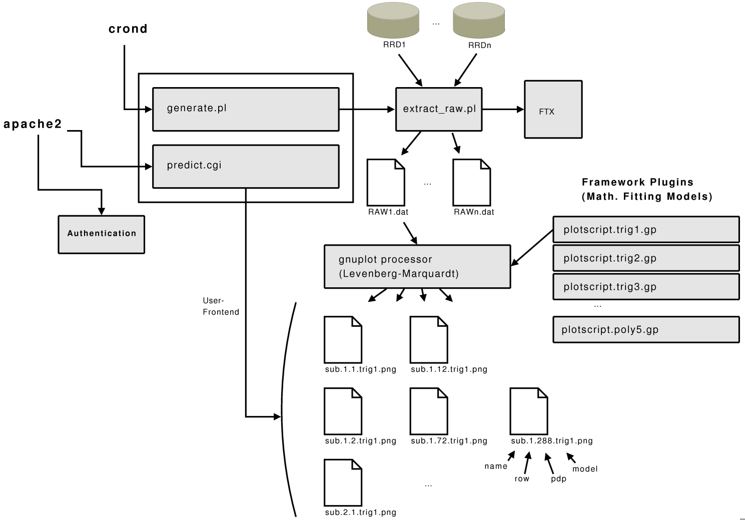
REA[7]-Framework

[7] RRD extrapolation and analysis

REA-Framework

- NG-independet script collection (works with all possible RRDs)
- **generate.pl**
  - Called via nightly cronjob
  - Automatically extracts RRD payload (extract_raw.pl) → triple: (timestamp, normalized idx, data value)
  - Runs configured plugins via gnuplot processor
  - Generates graph images (if possible) and copies them into WWW-dir
- **predict.cgi**
  - User interface
  - User assigns configured ‘subject‘ (e.g. host service) to ‘analysis type‘ (e.g. monomial extrapolation)
  - All RRA specific graphs with chosen fits are shown

REA-Framework



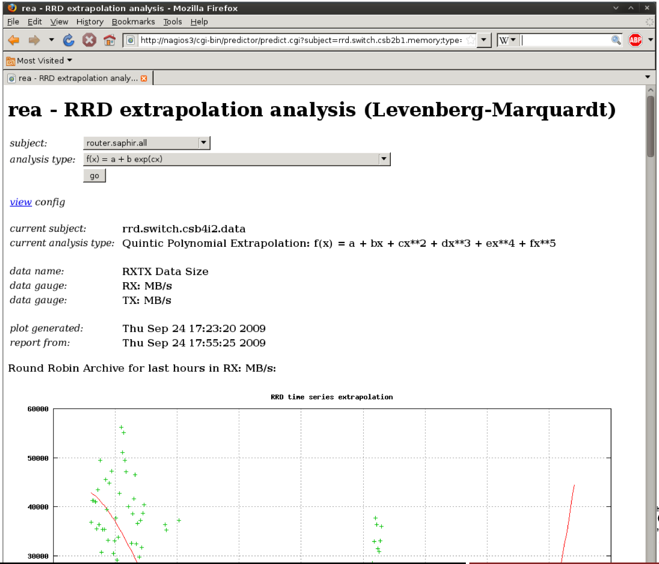
University of Applied Sciences

MAX-PLANCK-GESELLSCHAFT

REA-Framework, plugin for Monomial extrapolation

```
#!/usr/bin/gnuplot

FIT_LIMIT = 1e-6
FIT_MAXITER = 80

f(x) = a * (x - b)**n + c
fit f(x) "exp.dat" using 2:3 via a, b, c, n

set grid
set title "RRD time series extrapolation"
set timefmt "%s"
set xdata time
set format x "%d.%m.%Y, %H/%M"
set xlabel " "
set ylabel " "
set xtics rotate by 90 scale 0
set key below

plot "< cat exp.dat fore.dat" using 1:(f($2)) title "best \'a * (x - b)^n + c\' fit" with lines, \
     "exp.dat" using 1:3 title "time series data" with points

set term png size 1024, 768
set output "plot.png"
replot

set term dumb
```

REA-Framework, frontend


rea - RRD extrapolation analysis - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://nagios3/cgi-bin/predictor/predict.cgi?subject=rrd.switch.csb2b1.memory;type=
Most Visited
rea - RRD extrapolation analy...

rea - RRD extrapolation analysis (Levenberg-Marquardt)

subject: router.saphir.all
analysis type: f(x) = a + b exp(cx)
go

view config

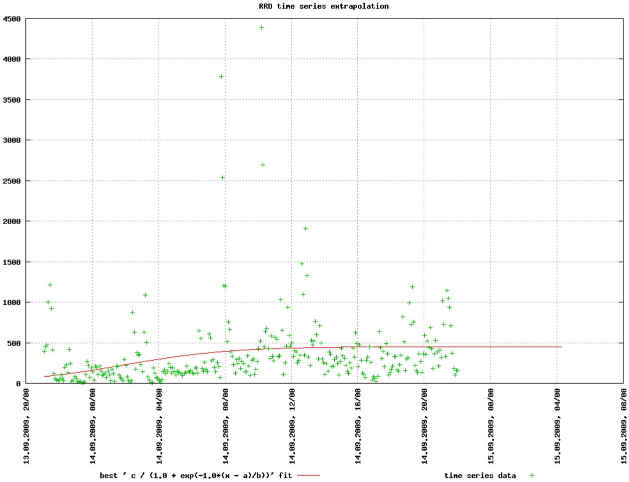
current subject: rrd.switch.csb4i2.data
current analysis type: Quintic Polynomial Extrapolation: f(x) = a + bx + cx**2 + dx**3 + ex**4 + fx**5

data name: RXTX Data Size
data gauge: RX: MB/s
data gauge: TX: MB/s

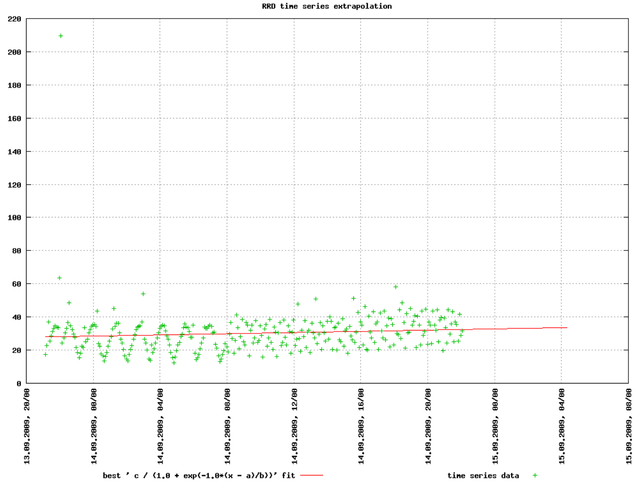
plot generated: Thu Sep 24 17:23:20 2009
report from: Thu Sep 24 17:55:25 2009

Round Robin Archive for last hours in RX: MB/s:

REA-Framework, Results (Traffic to webserver, KB/min)



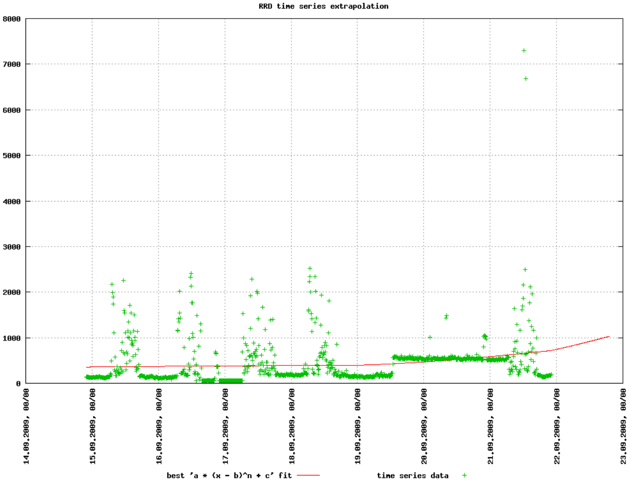
REA-Framework, Results (Broadcast traffic, Frames/min)



haft
und Kultur Leipzig (FH)
University of Applied Sciences

MAX-PLANCK-GESELLSCHAFT

REA-Framework, Results (SSH traffic, Frames/min)



haft
und Kultur Leipzig (FH)
University of Applied Sciences

MAX-PLANCK-GESELLSCHAFT

Acknowledgement

- Dr. Helmut Hayd, *Max Planck Institute for Human Cognitive and Brain Sciences*
- Prof. Dr.-Ing. Dietmar Reimann, *Leipzig University of Applied Science, Faculty of Computer Science, Mathematics, and Natural Sciences*

HTWK
Hochschule
für Technik, Wirtschaft
und Kultur Leipzig (FH)
University of Applied Sciences

MAX-PLANCK-GESELLSCHAFT

Links

- **Thesis:** http://edoc.mpg.de/437809
- **netsniff-ng:** http://netsniff-ng.googlecode.com
- **All the rest[8]:** <dborkmann@acm.org>

[8] It's still a prototype ...

Q+A?