

---

# RRDtool

**RRDtool** \aHr-aHr-´deE-t:ul\ n

[E, fr. round robin database tool]

:a system to store and display time-series data (i.e. utilized bandwidth, machine-room temperature, server load average).

R. stores the data in a very compact way so that it will not expand over time, and presents it in useful graphs.

Tobi Oetiker <[tobi@caida.org](mailto:tobi@caida.org)>

*Sponsors:*

AboveNet – [www.above.net](http://www.above.net)

CAIDA – [www.caida.org](http://www.caida.org)

ETH Zurich – [www.ethz.ch](http://www.ethz.ch)

---

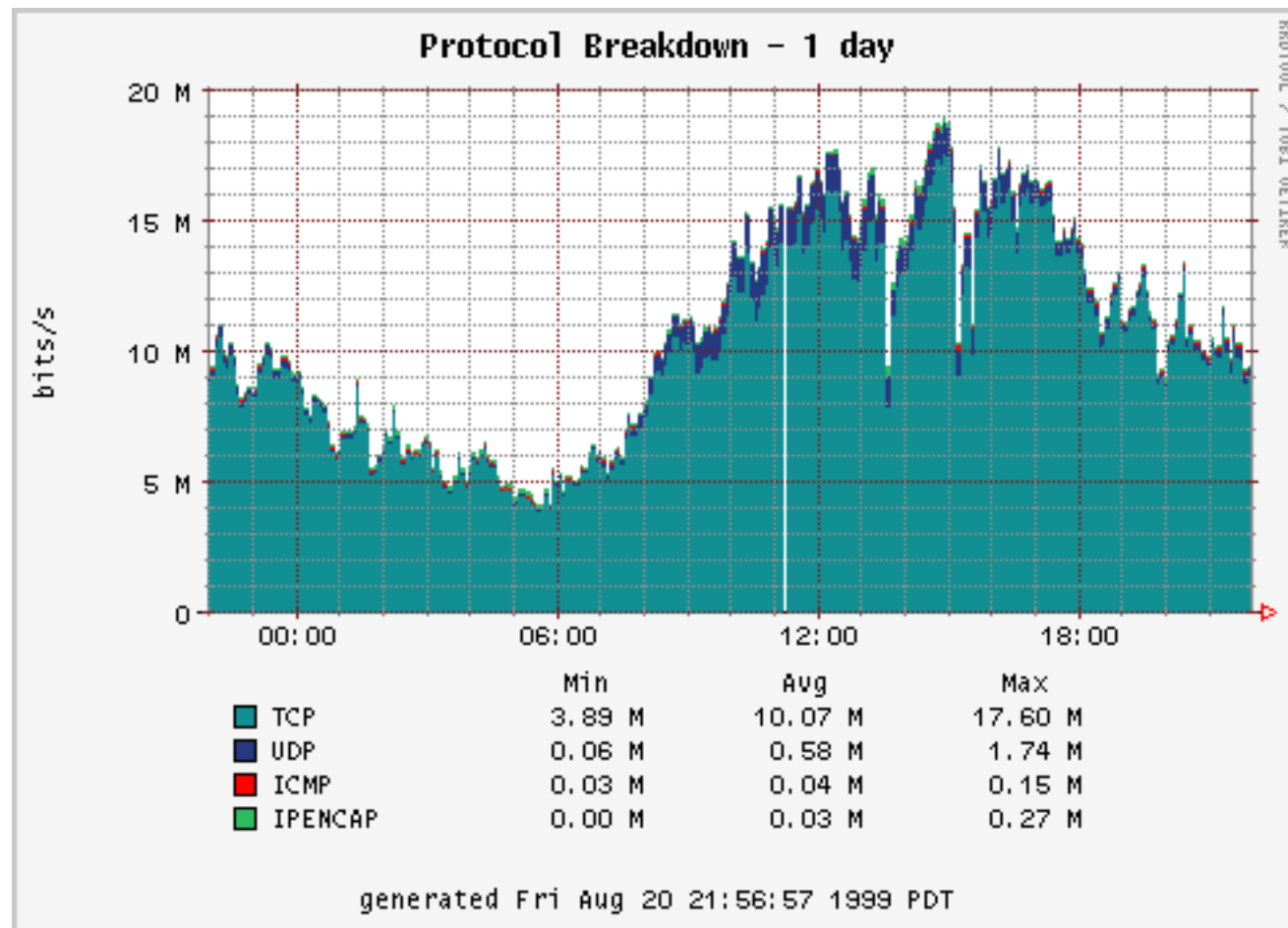
# Contents

- The Results
- The roots of RRDtool
- Design
- Implementation
- Programming with RRDtool
- Pre-Packaged solutions
- MRTG-3
- Future Work

---

# CoralReef Data

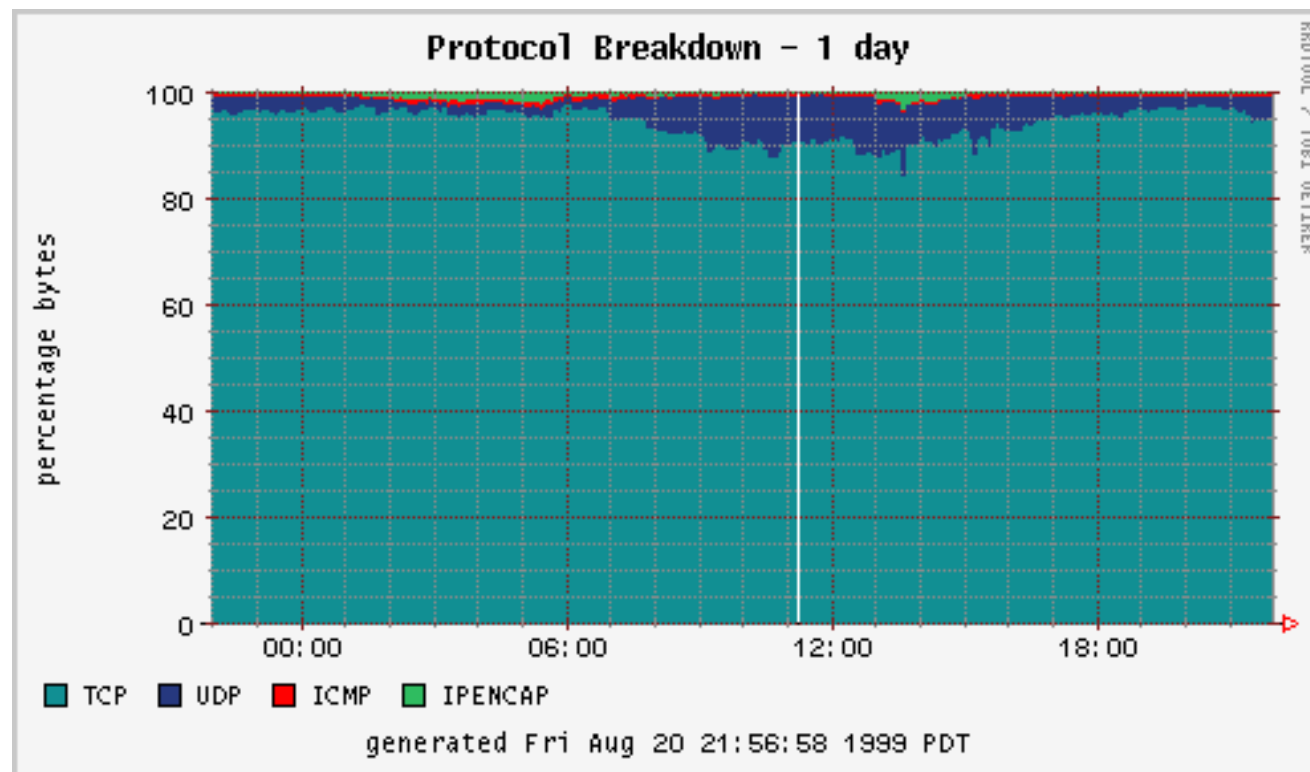
Visualization of Data collected with CAIDA's CoralReef package.



---

## CoralReef in Percent

The same data but normalized to 100% visualizing protocol distribution.



...and just in case: <http://www.caida.org/Tools/CoralReef/>

---

## MRTG - the roots of RRDtool

- 1995 creation of MRTG for a small site
- constant size log files
- growing popularity and thus many patches render code base difficult to maintain
- performance problems in large installations
- lack of flexibility: only two values in graph, integer log format, 5 minute interval, no concept of 'the unknown'

---

## RRDtool design

- end of 1997 start of RRDtool design
- full documentation
- floating point math
- \*unknown\* representation as “Not a Number”
- round robin logging for better performance
- flexible and powerful graphing
- terse data format which prevents invalid config by design

---

# RRDtool Implementation

The following slides will talk about the implementation of RRDtool.

- how RRDtool logging works
- how to create graphs
- general points on the RRDtool syntax
- methods for accessing RRDtool from within your applications

---

# RRDtool Logging

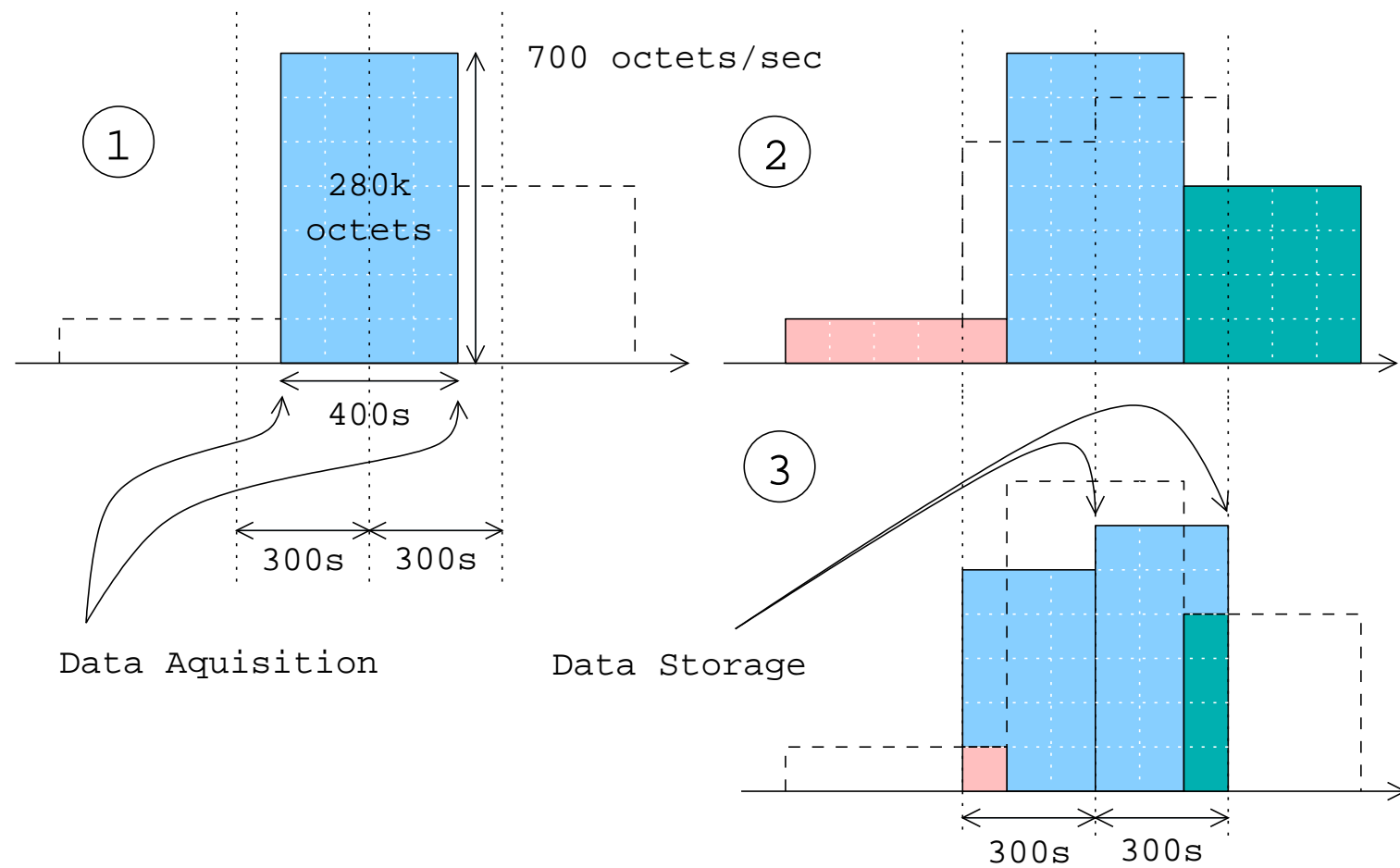
- round robin logging results in minimal data transfer
- fixed resolution logging with random data arrival
- storage of data at different resolutions in parallel
- on-the-fly data consolidation

The next slides will explain these topics in more detail so hang on!



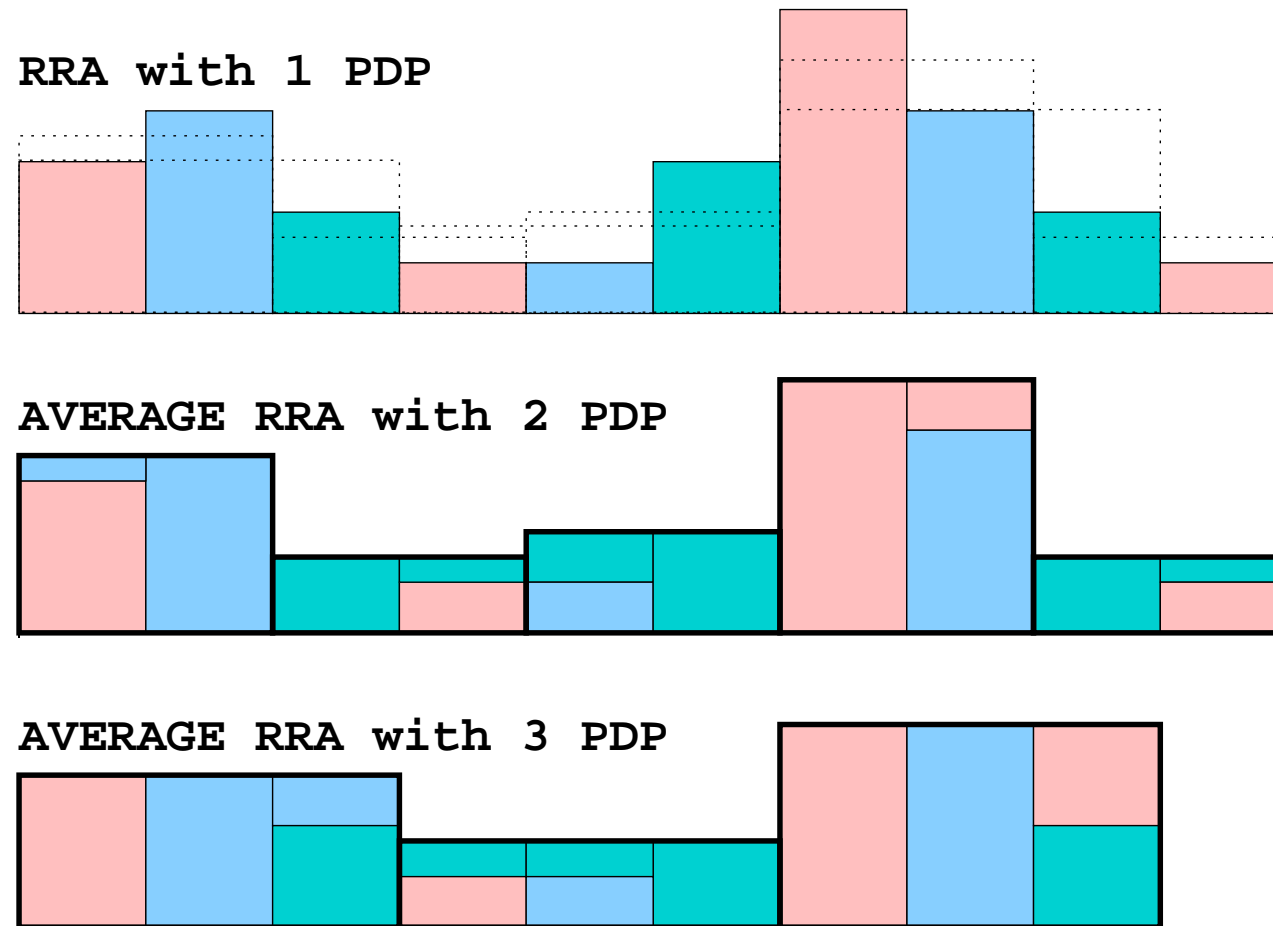
---

# Data Input “Data Re-Binning”



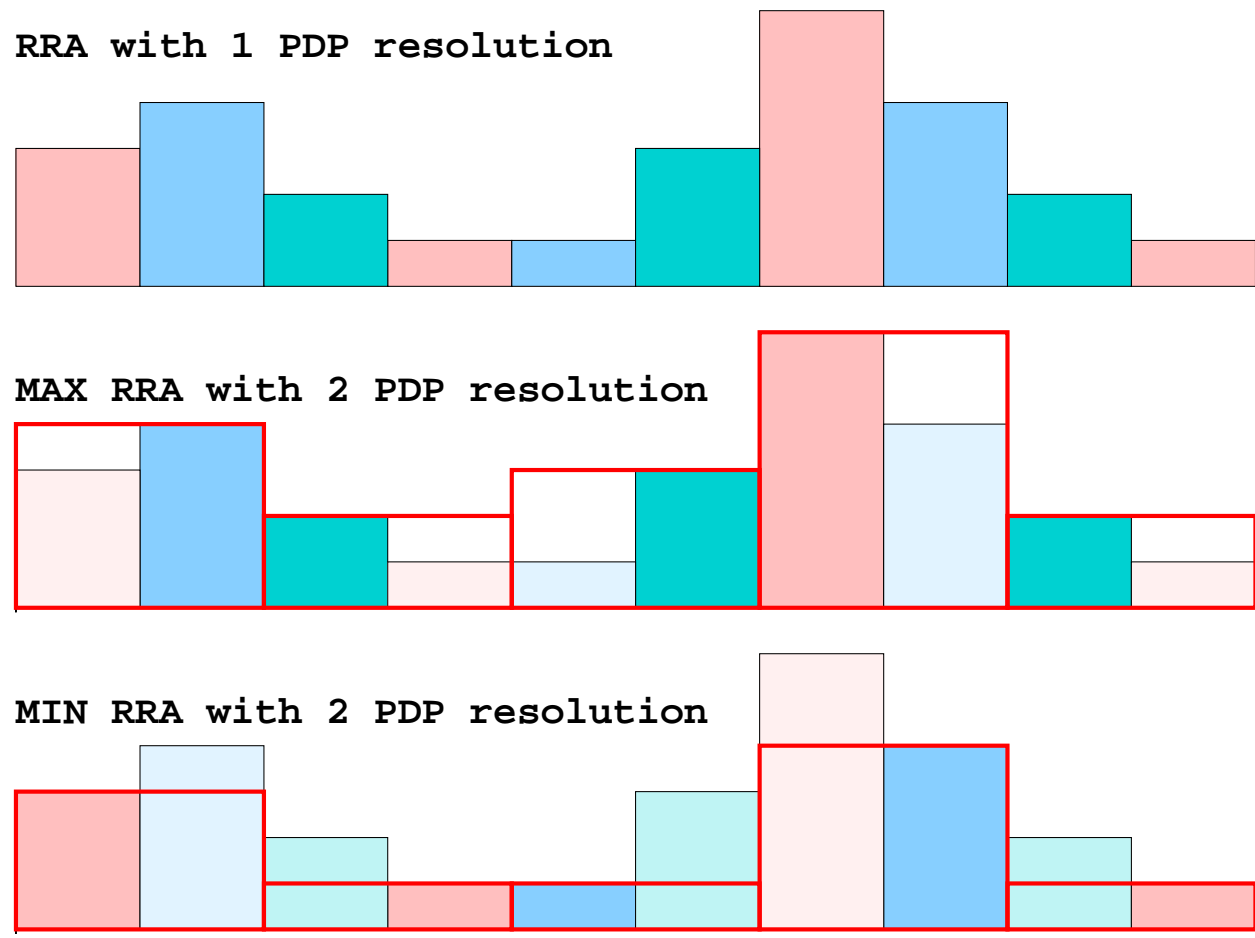
---

# Data Consolidation



---

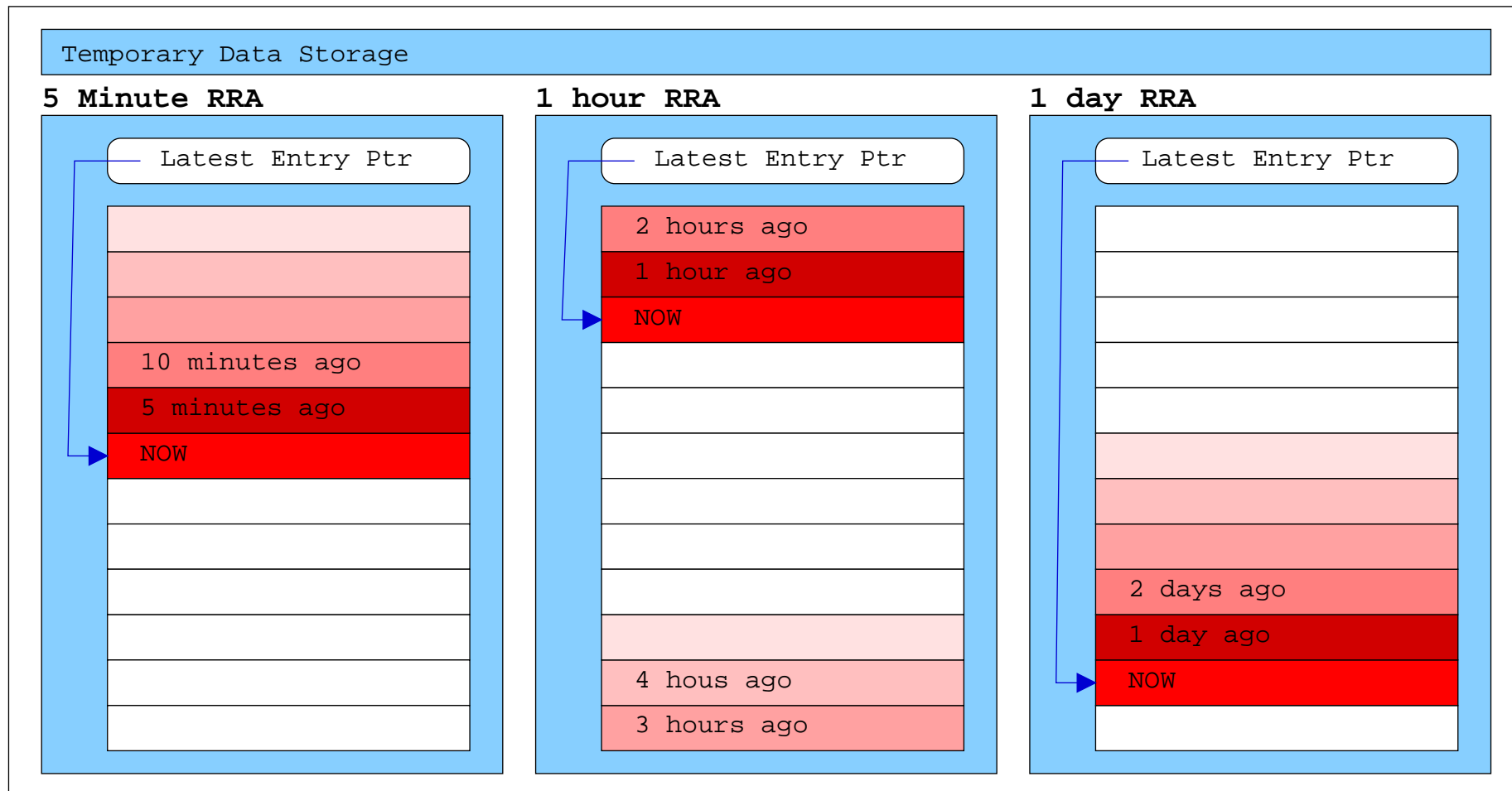
# Data Consolidation Methods



---

# Round Robin Storage

RRD with 3 RRAs



---

## Database Setup

Definition of an RRD Date Source (DS). The definition determines how the data-binning is performed:

```
DS : name : COUNTER | GAUGE | ABSOLUTE : heart-beat : min : max
```

Definition of a Round Robin Archive (RRA) which receives its data by aggregating a *number* of re-binned input values using a *Consolitation Function* (CF). If less than *part* of the input values are known, the result will be \*UNKNOWN\*. The results are stored in a rotary buffer of *length*:

```
RRA : AVERAGE | MIN | MAX : part : number : length
```

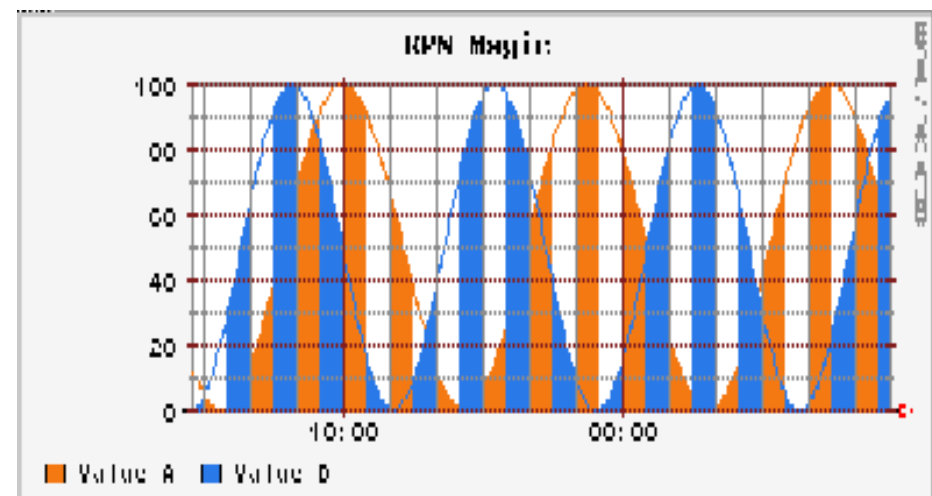
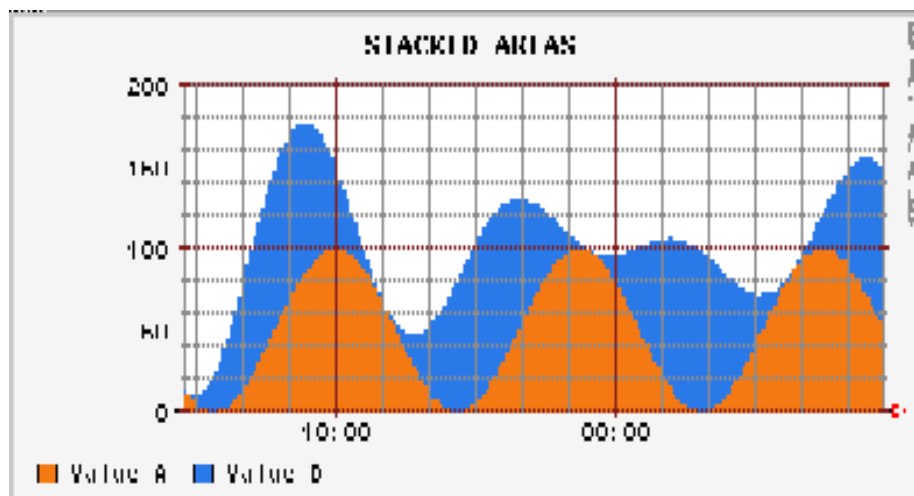
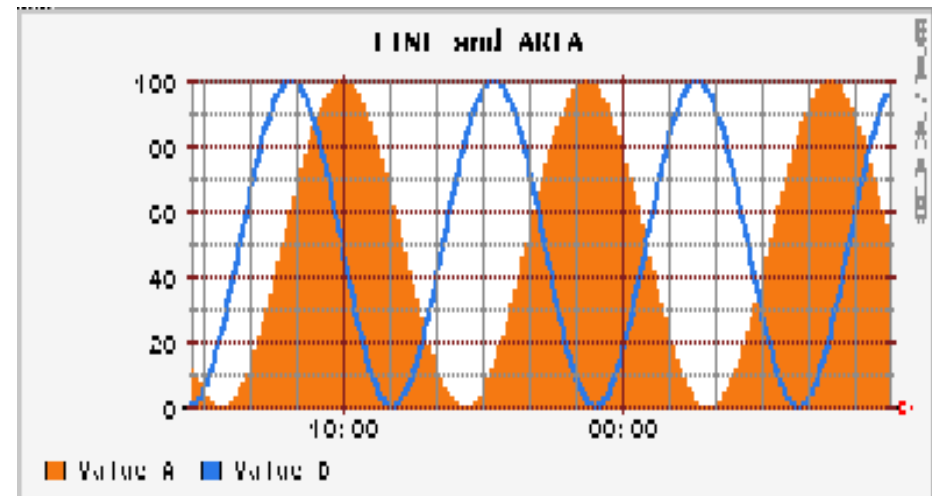
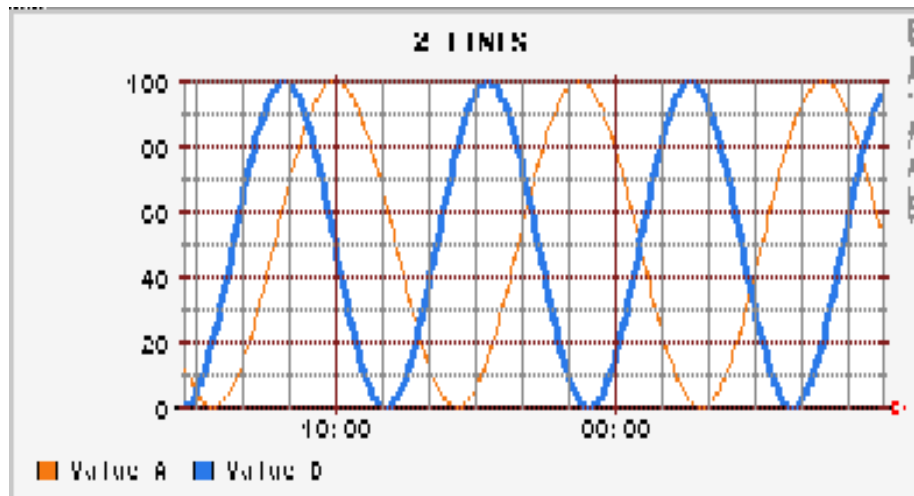
---

# Graphing

- use data from any number of RRDs
- line, area and stack charts
- perform powerful RPN math on data before graphing it

---

# Chart Type Demo



---

## RPN math

Pick a data source (octet counter):

```
DEF:a=file.rrd:incoming:AVERAGE
```

Modify the data: (People want to see bits while we get Octets from the router)

```
CDEF:b=a,8,*
```

Plot the data: (#ff0000 is the hex-triple representation for the color red)

```
AREA:b#ff0000:Incoming
```



---

## RRDtool Syntax

- many simple functions
- each one does its own argument evaluation
- functions do not exit or output
- caller has to deal with return values
- optional command line arguments start with '--'
- full documentation is available as man, txt and html

**rrdtool** *function arguments ...*

---

## Using RRDtool

- stand-alone: command line or pipe
- perl-module: accessing stand-alone or using a shared module
- RRDtool can be compiled as a shared library
- bindings for python and tcl being developed by contributors
- rrdcgi for web pages

Regardless of the language you are using RRDtool from, the syntax is always the same apart from requirements of the language.

---

## RRDtool functions I

Currently RRDtool has the following functions

**create** sets up a new Round Robin Database (RRD).

**update** stores new data values into an RRD.

**graph** creates a graph from data stored in one or several RRDs. Apart from generating graphs, data can also be extracted to stdout.

**dump** dumps the contents of an RRD in plain ASCII. In connection with `restore` you can use it to transport an RRD from one architecture to another.

continues on next slide ...

---

## RRDtool functions II

**restore** restores an RRD from XML format to a binary RRD.

**fetch** gets data for a certain time period from an RRD. The graph function uses fetch to retrieve its data from an RRD

**tune** alters the configuration of an RRD

**last** finds last update time of an RRD

**rrdresize** changes the size of individual RRAs ... Dangerous!

---

## Creating an RRDtool database

<code>rrdtool create demo.rrd</code>	\	By default, RRDtool will align incoming data into 5 minute (300 sec) bins.
<code>--step=300</code>	\	this would not be necessary
<code>DS:in:COUNTER:600:0:1.25e6</code>	\	max update interval 600s
<code>DS:out:COUNTER:600:0:1.25e6</code>	\	min val 0, max val 1.25MB
<code>RRA:AVERAGE:0.5:1:288</code>	\	5min avg for a day
<code>RRA:AVERAGE:0.5:12:168</code>	\	1hr avg for a week
<code>RRA:MIN:0.5:12:168</code>	\	1hr minima for a week
<code>RRA:MAX:0.5:12:168</code>	\	1hr maxima for a week

---

## Updating an RRDtool database

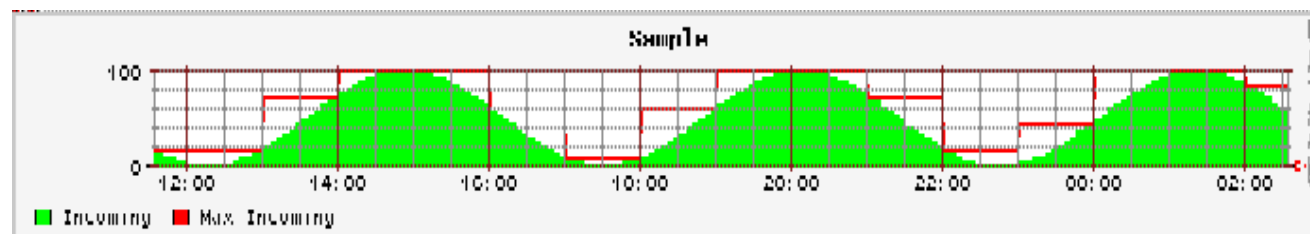
```
rrdtool update demo.rrd \  
--template in:out      \  
N:11222:1              \  
                        \  
                        \ 'N'=now could be sec since 1970
```

---

## Generating a graph

```
rrdtool graph ouput.png \
--imgformat=PNG \
--lower-limit=0 \
DEF:a=demo.rrd:in:AVERAGE \
DEF:b=demo.rrd:in:MAX \
AREA:a#00ff00:Incoming \
LINE1:b#ff0000:"Max Incoming"
```

\ makes the y axis start at 0  
\ 'a' gets data from the first RRA matching the resolution of the graph and covering the required time-span  
\ get data from an RRA with MAX consolidation  
\ a green AREA  
\ a thin red line



---

## Points to keep in mind

- RRDtool will make sure no garbage data goes into an RRD database. But in order to do so it must know what is sensible. Always specify MIN and MAX boundaries when setting up a new RRD.
- Use version numbers in the `use` statement for the RRD's perl module. This ensures you load the version of the module you expect to load. Module versions are in sync with RRDtool versions:

`x.y.z = x.yzzzz1` (eg. `1.0.6 = 1.000061`)

- Use `--lower-limit` if you want to make sure that the graph starts at 0.



---

# RRDcgi

Creating graphs is quite expensive, so why not do it on demand.

```
#!/usr/local/rrdtool-1.0.7/bin/rrdcgi
<HTML>
<HEAD><TITLE>RRD CGI Demo</TITLE></HEAD>
<BODY>
<H1>Demo Graph</H1>

<P><RRD::GRAPH
  output.png --imgformat=PNG --lower-limit=0 --lazy
  DEF:a=demo.rrd:in:AVERAGE DEF:b=demo.rrd:in:MAX
  AREA:a#00ff00:Incoming LINE1:b#ff0000:"Max In..."
></P>

</BODY></HTML>
```

---

## RRDtool Frontends

Why write your own Frontend when you can use someone else's:

**Cricket:** by Jeff R. Allen of WebTV. Filesystem based hierarchical configuration model. Dynamic graphing.

<http://www.munitions.com/~jra/cricket/>

**Orca:** by Blair Zajac of Yahoo!/GeoCities. Reads data from text files. Static graphing. <http://www.geocities.com/~bzking/>

**Remstats:** by Thomas Erskine of CRC. Remote monitoring Agents. Alerts.

<http://www.crc.ca/~terskine/remstats/>

**FlowScan:** by Dave Plonka at University of Wisconsin. Bringing Cflowd and RRDtool together. [plonka@doit.wisc.edu](mailto:plonka@doit.wisc.edu)

**MRTG-3:** Sales pitch ahead ...

---

## MRTG-3

- some code exists, but primarily in the concept stage.
- OO Perl module provides high-level functions for setting up your MRTG-3 Network monitor.
- Code instead of Configure.
- Datasources and Visualizations are defined in a tree structure. The tree knows how to handle the data. E.g. you Run the Data source tree and it gets new data from the different sources and updates the RRDs accordingly.
- send mail to [mrtg-developers-request@list.ee.ethz.ch](mailto:mrtg-developers-request@list.ee.ethz.ch) (subject: subscribe) if you are interested.

---

## Future Work on RRDtool

Version numbers will follow linux concept: Even sub-versions for stable code, odd numbers for development versions.

1.1.x will most likely break command line interface backward compatibility for certain parts of RRDtool.

- even more flexibility for the graphing module (compat break)
- sub-second resolution for RRD files
- anti aliased graphs with TrueType font support
- complex data analysis functions like 95 percentile
- general conceptual review ... no prisoners
- **full backward compatibility for data-format**

---

# Thanks!

Thanks to my Sponsors:

AboveNet – [www.above.net](http://www.above.net)

CAIDA – [www.caida.org](http://www.caida.org)

ETH Zurich – [www.ethz.ch](http://www.ethz.ch)

<http://www.caida.org/Tools/RRDtool>