Upwatch documentation

Ron Arts

Upwatch documentation
by Ron Arts

Table of Contents

List of Tables

Preface

People, especially managers, like to have facts and gures when taking decisions, either because a lot of money may be involved, or their job (or both). If you want to prove your website (or switch, or basically any other device) was available, showed the proper performance, or just want to know current and past CPU load, you've come to the right place.

UpWatch is very scalable (built for hundres of thousands of measurements per minute), fast, extensible, built on proven opensource tools, and tries very hard not to loose data.

Building and installing upwatch is not for the faint of heart. It uses lots of external libraries which may or may not be available on your platform. I myself use Redhat 8 for development, and test compilation on RH7, and Yellowdog Linux 2.3.

Chapter 1. Installation

1.1. Getting upwatch

Currently, upwatch is not released, and is not allowed to be distributed. The only way to get it, is through written permission of UpWatch BV.

If you aquired that, you will either receive access to CVS, or will receive a tar.gz le, or .RPM's.

Building and installing upwatch is not for the faint of heart. It uses lots of external libraries which may or may not be available on your platform. I myself use Redhat 8/9 for development, and test compilation on RH7, SuSE8.2, Solaris, and Yellowdog Linux 2.3.

1.2. Requirements

1.2.1. Run-time requirements

First ensure that the time/date on all hosts is correctly set.

Run time requirements differ per probe. Look in the corresponding .def le (or in the spec le for the probe), here's a list of everything we expect on a machine running all probes, and the database (I'll also list the version we use ourselves):

- glib2 >= 2.0.4
- xml2 (any version will do)
- freetds >= 0.6.0 compiled with --enable-threadsafe
- mysql 3.23.49
- postgreSQL 7.1.0
- net-snmp 5.0.6
- -lcrypto
- libnet 1.0.2a (www.packetfactory.net/libnet/)
- libpcap 0.6.2

Delivered with upwatch are libstatgrab and the State Threads Library.

1.2.2. Build requirements

Of course you can build the software yourself. Apart from the normal GNU compilation tools, and the development versions of the aforementioned packages, you'll need the following on your system to build upwatch:

- autogen 5.3.6 (autogen.sourceforge.net)
- libxslt 1.0.15
- docbook 1.48, including the entire toolchain: openjade, jadetex, tetex, netpbm, perl-SGMLSpm
- lynx 2.8.4

- RPM tools, if you want to build RPM's

If you run redhat, debian or SuSe, don't forget to install the devel packages if there are any.

1.3. Compiling upwatch

Just in case you really want to (or need to) compile upwatch yourself, it's pretty easy:

```
$ tar xzvf upwatch-x.x.tar.gz
$ cd upwatch-x.x
$ ./con gure
$ make
```

Nothing to it... In case of problems, you're probably missing some library or header les, or they are in unexpected places. Look in con g.log.

You can optionally specify --enable-monitors, --enable-iptraf and/or --enable-server to con gure. Default con gure only builds the client, docs, and utils.

1.4. Actual Installation

Before you install the software decide on the architecture. If you know in advance you'll have to monitor thousands of hosts, or the probes will exhaust your machine otherwise, you may have to split your installation across several machines. There may be more reasons to do that. Consult Scaling up and How it all works

For simplicity we assume you run everything on the same host. In this case just install all rpm's on this host. What if you can't use RPM's? Then type the following command as root:

```
$ make install
```

1.5. PHP pages

The PHP pages can just be copied to any directory. There is an include directory. Copy that a some location ouside the web root, and enter its location in the .htaccess le in the web root dir. Also enter the database details in con g.php.

1.6. Security considerations

All upwatch directories are readable and writable by members of the group upwatch. Most all executables run as user upwatch. Some probes need root-access, most notable uw_ping, and they will be installed suid root. These probes drop root privileges wherever possible. Further you can assign each probe its own database user and grant that user access rights to its own database tables. The probes itself don't write to the database, they only read from the prt_xxx_def tables.

The PHP web user should have SELECT, UPDATE, DELETE access to all tables.

uw_accessd and uw_accessdb are the programs most vulnerable to crackers, as they wait for incoming connections on a TCP port (1985/1984). If possible, use chroot and firewall rules to limit connections to real probes only. Something similar holds for mysql. Most probes will want access, and passwords can be sniffed. For real security use ssh-tunnels.

1.7. Database

First things first. Depending on the size of your installation you may run out of database or record space. It happened to me on the iptraf probe. I was measuring traffic for 4000 IP addresses and ran out of space after a month on the pr_iptraf_raw table - it hit the max_data_length limit. I had to issue the following commands:

```
$ mysql -u root --password=PASSWORD
mysql> alter table pr_iptraf_raw max_rows = 1000000000;
```

and this took almost two hours! So you better look at your own situation and adjust the settings MAX_ROWS and AVG_RECORD_SIZE accordingly for each table.

Create the database as follows. You DO have a root password set for mysql don't you?

```
$ mysqladmin -u root --password=PASSWORD create upwatch
$ mysql -u root --password=PASSWORD < upwatch.mysql
```

Of course you need to assign users and GRANT them access. Note that mosts probes will want read access to their definition table. In many situations you can use just one user for that. Give that user access with:

```
  $ mysql -u root --PASSWORD=PASSWORD mysql
mysql> GRANT SELECT ON upwatch.pr_ping_def TO user@'192.168.170.23' IDENTIFIED BY 'PASSW
```

Chapter 2. Con guration

2.1. Probe con guration

Each probe rst reads the general con guration le /etc/upwatch.conf and then its own con guration le in /etc/upwatch.d if it exists. Normally some general things like the debug and logging level, and the database access are speci ed in the rst le, and any probe-speci c setting in the second le. You can also override settings from the generic le in the probe-speci c le.

Each program has a manual page that documents options. Every long commandine option can also be entered in a con guration le.

2.2. Database con guration

Chapter 3. Administration

3.1. Logging

The upwatch package contains various ways of logging errors. The standard way is to its own log le /var/log/upwatch/upwatch.log. Other ways are logging to stderr (probably not practical) and to the syslog. Tweak the debug to increase the amount of logging. Setting the debug level higher than 2 should only be used for debugging serious problems, for example it causes daemons to stay always in the foreground. In debuglevel 0 only errors are logged, in debuglevel 1 some progress information is logged.

The website has its own log le in log/error.log

3.2. Managing daemons

In most Linux distributions you can start/stop daemons using the scripts in /etc/init.d. Don't forget: you will miss sample data in the database if a probe is not running. You can watch what a probe is running if you run ps ax.

3.3. Queues

Queues play an important part in upwatch. The queues are so-called maildir queues. This means that while the queue le is written, it is written to a temporary directory, and when it's closed it is hardlinked to the actual queue directory. This way you can be absolutely sure that if you nd a le in the queue, it is complete and nobody has the le open. Only one process reads from the queue and deletes the le when done.

Chapter 4. How it all works

4.1. General Overview

The system primary function is to ll lots of database tables, to offer views on those tables, and to page operators in case things go wrong. To enable this upwatch consists of a MySQL database, lots of probe daemons (one daemon per probe, usually one probe per daemon), some supporting daemons, a PHP website, and other software, like SMS and mail interfaces.

The software can be divided into four parts:

- upwatch client - runs on a machine
- server, accepts and processes results
- monitors, contains software for remotely monitoring.
- special software, like iptraf

4.1.1. The Upwatch client

The client consists of two programs: uw_sysstat and uw_send. uw_sysstat every minute collects information like CPU load, disk I/O, swapping activity and so on, and writes it to an XML le in the spooldirectory. This directory is checked every 5 seconds, and all les appearing there are sent by uw_send to the central repository. uw_send has a commandline option (--once) to let it be started by cronjobs, or for example when an ISDN connection has become online.

4.1.2. The Upwatch server

The server consists of three programs: uw_accept, uw_setip and uw_process. The monitoring results are accept by uw_accept which listen on port 1985 (con gurable), and drop the XML results into the uw_process spool directory, where it is picked up, and stored into the database by uw_process. For compatibility with Big Brother (www.bb4.com) clients, there is an uw_acceptbb daemon, which listens on the Big Brother port (1984), and converts Big Brother messages into upwatch XML les. Lastly, ,uw_setip listens to messages from the uw_tellip script, which should be started by clients whenever their IP address changes.

4.1.3. The Upwatch monitors

The monitors are daemons that run on some central monitoring server, and run checks on servers remotely, such as POP3, HTTP, SNMP or other services. Alll their results are sent by uw_send, as usual.

4.1.4. Special programs

Their are special programs that don't fall into any other category, for example uw_iptraf. This is a daemon that should run on a border gateway router, and that measures IP traf c on a per-IP basis.

4.2. A Detailed Description

4.2.1. Database Layout

Things start at the database. For every probe it contains the following tables:

- De nition table
- Raw results table
- Tables for compressed results per day, week, month, year and 5 year
- A table with an overview of state changes

The de nition table contains, of course, the de nition of this particular probe, this is of course probe speci c but at a minimum it contains usually contains the target ip address. We'll see what the other tables are for later on.

4.2.2. What a probe does

There are actually three kinds of probes:

- Probes with database access, that measure a remote server
- Probes without database access the measure remote servers
- Probes without database access thatg measure localhost

Every probe performs a repetative task: measuring some speci c function on a speci c host. So rst step is to know what to measure and on which host. For this it reads from the probe de nition table, or from its con g le if it does not depend on database access It creates a local - in memory - copy of that table just in case the database becomes unreachable for a period of time. It routinely walks this list and performs its task. The result are written in XML format to a queue which is speci ed in the probe con guration (note: all queues normally reside in /var/spool/upwatch). After that the probe just waits for the next round.

Many probes have to do a lot of work. They are programmed to do this as ef cient as possible. For example: the uw_ping probe is coded as a tight loop around a single select statement. This is the most ef cient way (as far as I know) to ping thousands of hosts in, say, 20 seconds. Other probes use pools of threads (like httpget) or are build using the State Threads library.

4.2.3. What happens to the probe results?

First, all results with status non-green are handed over to uw_examine , which tries to nd out why the probe failed, and attaches a report to the probe. After this the results are put in the same queue as every other probe: uw_notify . uw_notify reads the result, looks at the probe status, and at previous statuses, and decides if someone should be noti ed by sms, email, or if it should be put into a high-priority queue.

The outgoing queue may be either the uw_process queue, or an uw_send queue, which is emptied by the uw_send process which sends all les to a remote queue on another host (received and queued by uw_accept).

uw_examine can do some additional tests like traceroute to the target host. It attaches this report to the probe result, and in its turn puts everything in an uw_process or uw_send queue.

4.2.4. uw_process: storing results in the database

When the probe results arrive in the uw_process queue it is picked up by the workhorse of the lot, uw_process . It lls the result tables for the probes.

The raw results table contains just that, raw probe results.

Raw results are compressed into period tables in the following way (using week as an example): a week is divided into 100 equal timeslots. For computing the plot values for a slot the process reads all values from the day table in the same timeslot. These values are averaged and put in the week table. The same process happens for the month and year tables. This way we ensure that we never have to read more then 100 database records to produce a graph for a day, week, month, year or 5-year period.

Status changes are logged in a 'current status' table and in a status history (pr_hist). These two accomodate for easy retrieval by the webpages.

The pseudo-code below shows an example of how uw_process takes a probe result and puts this result in the database. as an example I'll take a pop3 result (class = 5, and our example probe has id 25)

```
   IF PROBEDEFINITION NOT IN THE CACHE OR IT'S TOO OLD
     select server, color, stattime, yellow, red from pr_status where  class = '8' and probe = '25'"
     IF NOT FOUND IN STATUS FILE
       select server, yellow, red  from   pr_pop3_def where  id = '25'
       IF NOT FOUND IN DEFINITION TABLE
         SKIP THIS PROBE
         probes without id (because they don't have database access) may be added here
       ENDIF not in de nition table
     ENDIF not in status  le
     GET MOST RECENT PROBE RESULT TIME:
   select stattime from pr_pop3_raw use index(probstat) where probe = '25' order by stattime desc limit 1
   ENDIF not in cache
   STORE RESULT:
  insert into pr_pop3_raw set probe = '25', yellow = '1', red = '2', stattime = 'xxxxx', color = 'xx',
    connect = '1', total = '2', message = 'none'
   IF CURRENT PROBE IS NEWER THEN ANY WE'VE SEEN SO FAR
     copy previous record stattime from def record
   ELSE
   select  color, stattime from pr_pop3_raw use index(probstat) where probe = '25' and stattime < 'xxxx'
     order by stattime desc limit 1
   ENDIF

   IF THIS IS THE FIRST RESULT EVER SEEN FOR THIS PROBE
    insert into pr_status set class = '8', probe = '25', stattime = 'xxx', expires = 'xxx', color = '200',
       server = '2', message = 'none', yellow = '1', red = '2'
   ELSE IF WE HAVE NOT SEEN THIS PROBE BEFORE
     IF THE COLOR DIFFERS FROM THE PREVIOUS RECORD
       CREATE HISTORY RECORD:
     insert into pr_hist set server = '2', class = '8', probe = '25', stattime = 'xxx', prv_color = '500', color = '200', r
sage = 'none'
       RETRIEVE FOLLOWING RECORD:
     select color, stattime from pr_pop3_raw use index(probstat) where probe = '25' and stattime < 'xxxx'
       order by stattime desc limit 1
```

```
        IF FOUND AND HAS THE SAME COLOR DELETE ANY HISTORY RECORDS:
          delete from pr_hist where stattime = 'xxxx' and probe = '25' and class = '8'
    delete from pr_status where stattime = 'xxx' and probe = '25' and class = '8'
        ENDIF following found and has same color
        IF CURRENT RECORD IS THE NEWEST UPDATE STATUS AND SERVER STATUS
       update pr_status set stattime = 'xxx', expires = 'xxxy', color = '200', message = 'none', yellow = '1', red =
          where probe = '25' and class = '8'
          update server set color = '20' where id = '2'
        ENDIF newest
      ENDIF color differs
      IF CURRENT RAW RECORD IS THE MOST RECENT
        FOR EACH PERIOD
          IF WE ENTERED A SLOT TIMESLOT IN THE PERIOD
            SUMMARIZE:
         select avg(connect), avg(total), max(color), avg(yellow), avg(red) from pr_pop3_day use index(probst
            where probe = '25' and stattime >= slotlow and stattime < slothigh
         insert into pr_pop3_week set connect = '1', total = '2', probe = 25, color = '200', stattime = slot,
              yellow = '1', red = '2', slot = '34'
          ENDIF
        ENDFOR
      ELSE
        FOR EACH PERIOD
         IF THE FIRST RECORD FOR THE NEXT SLOT HAS BEEN SEEN
           RE-SUMMARIZE CURRENT SLOT
        select avg(connect), avg(total), max(color), avg(yellow), avg(red) from pr_pop3_day use index(probsta
           where probe = '25' and stattime >= slotlow and stattime < slothigh
        insert into pr_pop3_week set connect = '1', total = '2', probe = 25, color = '200', stattime = slot, yel-
low = '1',
             red = '2', slot = '34'
         ENDIF
      ENDIF
    ENDIF
```

4.3. Scaling up

Various parts of the system may need more resources. Luckily Upwatch is designed to scale up considerably. Of course it cannot scale in nately. The last bottleneck will probably be the database. Although MySQL is known for its speed, even that has its limits.

The probes may be scaled up, sometimes by giving them more lehandles, later by moving them to another host

The website may be scaled up by spreading it out across several hosts

The database may be scaled up by putting it on separate hardware, using faster CPU and more spindles (disks), and ultimately using MySQL mirrorring to divide reading and writing across separate machines, or spreading out the tables across multtiple machines. MySQL has lots of info on increasing performance..

Appendix A. Interfaces and le layouts

A.1. Probe result le

Every probe result is written in XML format into a queue le. This le will be picked up by the process emptying the queue, usually uw_send, or uw_process. The le must have a speci c name, and a speci c layout.

A.1.1. Probe le name

The name of the le is composed of the current epoch time in seconds, microseconds, process id, and hostname on which the queue resides, all separated by dots. An example would be:

- 1031601982.341878.27470.ron-ibook.nbs.arts-betel.org

From a shell you can generate such a name using echo `date +%s`.500.$$.`hostname`

A.1.2. Generic probe le layout

The probe result le is in XML format, described in /usr/lib/upwatch/dtdt/result.dtd .

A.1.3. uw_accept protocol

The protocol used by uw_accept is almost exactly like the POP3 protocol. login with USER and PASS, then enter DATA lesize and start uploading

Appendix B. Probe speci cations

B.1. bb - Big Brother generic probe

B.1.1. bb result record layout

Table bb attributes. bb attributes

Name	Type	Required	Default	Description
host	NMTOKEN	NO		host where this element originated

Table bb elements. bb elements

Name	Optional	Description
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.1.2. bb database layout

Table bb de nition record layout. bb de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO			value for yellow alert

red	oat	NO			value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing
bbname	char	NO			Big Brother name of this probe

Table bb result record layout. bb result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO			value for yellow alert
red	oat	NO			value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value

B.2. bb_cpu - Big Brother System probe

B.2.1. bb_cpu result record layout

Table bb_cpu attributes. bb_cpu attributes

Name	Type	Required	Default	Description
host	NMTOKEN	NO		host where this element originated
loadavg	NMTOKEN	NO		Load average as computed by upwatch
user	NMTOKEN	NO		CPU user time
system	NMTOKEN	NO		CPU system time
idle	NMTOKEN	NO		CPU idle time
swapped	NMTOKEN	NO		Amount of blocks written to swap device
free	NMTOKEN	NO		Free memory
buffered	NMTOKEN	NO		Amount of memory used for OS buffers

cached	NMTOKEN	NO		Amount of memory used for disk buffers
used	NMTOKEN	NO		Amount of memory used by processes

Table bb_cpu elements. bb_cpu elements

Name	Optional	Description
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.2.2. bb_cpu database layout

Table bb_cpu de nition record layout. bb_cpu de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing

Table bb_cpu result record layout. bb_cpu result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value
loadavg	oat	NO	0		Load average as computed by upwatch
user	tinyint	NO	0		CPU user time
system	tinyint	NO	0		CPU system time
idle	tinyint	NO	0		CPU idle time
swapped	int	NO	0		Amount of blocks written to swap device
free	int	NO	0		Free memory
buffered	int	NO	0		Amount of memory used for OS buffers
cached	int	NO	0		Amount of memory used for disk buffers
used	int	NO	0		Amount of memory used by processes

B.3. httpget - Do a HTTP GET request

B.3.1. httpget result record layout

Table httpget attributes. httpget attributes

Name	Type	Required	Default	Description
lookup	NMTOKEN	NO		time needed for DNS lookup
connect	NMTOKEN	NO		time for connection to complete
pretransfer	NMTOKEN	NO		time for any pre-transfer actions
total	NMTOKEN	NO		total time needed

Table httpget elements. httpget elements

Name	Optional	Description
id	NO	id of this probe in the database
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.3.2. httpget database layout

Table httpget de nition record layout. httpget de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO	1		value for yellow alert
red	oat	NO	3		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing
hostname	varchar(80)	NO			Hostname for the HTTP request
uri	varchar(255)	NO			URI part

Table httpget result record layout. httpget result record layout

Field	Type	Key	Default	Extra	Description

id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	1		value for yellow alert
red	oat	NO	3		value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value
lookup	oat	NO	0		time needed for DNS lookup
connect	oat	NO	0		time for connection to complete
pretransfer	oat	NO	0		time for any pre-transfer actions
total	oat	NO	0		total time needed

B.4. imap - Test a IMAP server, optionally with user/password

B.4.1. imap result record layout

Table imap attributes. imap attributes

Name	Type	Required	Default	Description
connect	NMTOKEN	NO		time for connection to complete
total	NMTOKEN	NO		total time needed

Table imap elements. imap elements

Name	Optional	Description
id	NO	id of this probe in the database
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.4.2. imap database layout

Table imap de nition record layout. imap de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing
username	varchar(64)	NO			Username
password	char	NO			Password

Table imap result record layout. imap result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value
connect	oat	NO	0		time for connection to complete
total	oat	NO	0		total time needed

B.5. iptraf - Incoming and outgoing traf c to an IP adddress, network or interface

B.5.1. iptraf result record layout

Table iptraf attributes. iptraf attributes

Name	Type	Required	Default	Description
incoming	NMTOKEN	NO		total incoming bytes
outgoing	NMTOKEN	NO		total outgoing bytes

Table iptraf elements. iptraf elements

Name	Optional	Description
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server
interval	NO	time between measurements

B.5.2. iptraf database layout

Table iptraf de nition record layout. iptraf de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes

yellow	oat	NO	300		value for yellow alert
red	oat	NO	500		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing

Table iptraf result record layout. iptraf result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	300		value for yellow alert
red	oat	NO	500		value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value
incoming	oat	NO	0		total incoming bytes
outgoing	oat	NO	0		total outgoing bytes

B.6. mssql - Do a Microsoft SQL Server query

B.6.1. mssql result record layout

Table mssql attributes. mssql attributes

Name	Type	Required	Default	Description
connect	NMTOKEN	NO		time for connection to complete
total	NMTOKEN	NO		total time needed

Table mssql elements. mssql elements

Name	Optional	Description
id	NO	id of this probe in the database
ipaddress	NO	target ip address

date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.6.2. mssql database layout

Table mssql de nition record layout. mssql de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO	0.5		value for yellow alert
red	oat	NO	0.8		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing
dbname	char	NO			Name of the database for the query
dbuser	char	NO			Database user
dbpasswd	char	NO			Database password
query	text	NO			Query to perform. This should return at least 1 row

Table mssql result record layout. mssql result record layout

Field	Type	Key	Default	Extra	Description

id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	0.5		value for yellow alert
red	oat	NO	0.8		value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value
connect	oat	NO	0		time for connection to complete
total	oat	NO	0		total time needed

B.7. mysql - Do a MySQL query

B.7.1. mysql result record layout

Table mysql attributes. mysql attributes

Name	Type	Required	Default	Description
connect	NMTOKEN	NO		time for connection to complete
total	NMTOKEN	NO		total time needed

Table mysql elements. mysql elements

Name	Optional	Description
id	NO	id of this probe in the database
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.7.2. mysql database layout

Table mysql de nition record layout. mysql de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO	0.3		value for yellow alert
red	oat	NO	0.5		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing
dbname	char	NO			Name of the database for the query
dbuser	char	NO			Database user
dbpasswd	char	NO			Database password
query	text	NO			Query to perform. This should return at least 1 row

Table mysql result record layout. mysql result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	0.3		value for yellow alert
red	oat	NO	0.5		value for red alert
stattime	int unsigned	YES	0		time when result was generated

color	smallint unsigned	YES	200		color value
connect	oat	NO	0		time for connection to complete
total	oat	NO	0		total time needed

B.8. ping - send ICMP echo requests

Five ICMP echo requests are sent. For each request the time is measured between the echo and the resulting ICMP reply packet.

B.8.1. ping result record layout

Table ping attributes. ping attributes

Name	Type	Required	Default	Description
value	NMTOKEN	NO		Average turn-around time
lowest	NMTOKEN	NO		lowest turn-around time
highest	NMTOKEN	NO		highest turn-around time

Table ping elements. ping elements

Name	Optional	Description
id	NO	id of this probe in the database
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.8.2. ping database layout

Table ping de nition record layout. ping de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id

server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing
count	int unsigned	NO	5		Number of ping packets to send

Table ping result record layout. ping result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value
value	oat	NO	0		Average turn-around time
lowest	oat	NO	0		lowest turn-around time
highest	oat	NO	0		highest turn-around time

B.9. pop3 - Test a POP3 server, optionally with user/password

B.9.1. pop3 result record layout

Table pop3 attributes. pop3 attributes

Name	Type	Required	Default	Description
connect	NMTOKEN	NO		time for connection to complete
total	NMTOKEN	NO		total time needed

Table pop3 elements. pop3 elements

Name	Optional	Description
id	NO	id of this probe in the database
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.9.2. pop3 database layout

Table pop3 de nition record layout. pop3 de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe

hide	enum('yes', 'no')	NO	no		hide probe results from viewing
username	varchar(64)	NO			Username
password	char	NO			Password

Table pop3 result record layout. pop3 result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value
connect	oat	NO	0		time for connection to complete
total	oat	NO	0		total time needed

B.10. postgresql - Do a PostgreSQL query

B.10.1. postgresql result record layout

Table postgresql attributes. postgresql attributes

Name	Type	Required	Default	Description
connect	NMTOKEN	NO		time for connection to complete
total	NMTOKEN	NO		total time needed

Table postgresql elements. postgresql elements

Name	Optional	Description
id	NO	id of this probe in the database
ipaddress	NO	target ip address
date	NO	date/time for this result

expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.10.2. postgresql database layout

Table postgresql de nition record layout. postgresql de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing
dbname	char	NO			Name of the database for the query
dbuser	char	NO			Database user
dbpasswd	char	NO			Database password
query	text	NO			Query to perform. This should return at least 1 row

Table postgresql result record layout. postgresql result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result

probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value
connect	oat	NO	0		time for connection to complete
total	oat	NO	0		total time needed

B.11. snmpget - Query an SNMP variable using an SNMP GET

B.11.1. snmpget result record layout

Table snmpget attributes. snmpget attributes

Name	Type	Required	Default	Description
value	NMTOKEN	NO		Value of OID queried

Table snmpget elements. snmpget elements

Name	Optional	Description
id	NO	id of this probe in the database
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.11.2. snmpget database layout

Table snmpget de nition record layout. snmpget de nition record layout

Field	Type	Key	Default	Extra	Description

id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing
community	char	NO	public		community string for SNMPv1/v2c transactions
OID	varchar(255)	NO			Object ID
dispname	char	NO			Display Name
dispunit	char	NO			Display Unit
multiplier	oat	NO	1		Multiplier for result values
mode		NO	absolute		plot absolute or relative values

Table snmpget result record layout. snmpget result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value
value	oat	NO	0		Value of OID queried

B.12. sysstat - System information like load average, CPU/MEM usage etc

B.12.1. sysstat result record layout

Table sysstat attributes. sysstat attributes

Name	Type	Required	Default	Description
loadavg	NMTOKEN	NO		The load average as reported by the system
user	NMTOKEN	NO		CPU user time
system	NMTOKEN	NO		CPU system time
idle	NMTOKEN	NO		CPU idle time
swapin	NMTOKEN	NO		Amount of blocks swapped in from disk
swapout	NMTOKEN	NO		Amount of blocks swapped out to disk
blockin	NMTOKEN	NO		Amount of blocks read from block devices
blockout	NMTOKEN	NO		Amount of blocks written to block devices
swapped	NMTOKEN	NO		Amount of blocks written to swap device
free	NMTOKEN	NO		Free memory
buffered	NMTOKEN	NO		Amount of memory used for OS buffers
cached	NMTOKEN	NO		Amount of memory used for disk buffers
used	NMTOKEN	NO		Amount of memory used by processes
systemp	NMTOKEN	NO		System temperature in Celsius

Table sysstat elements. sysstat elements

Name	Optional	Description
server	NO	id of this server in the database
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.12.2. sysstat database layout

Table sysstat de nition record layout. sysstat de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing

Table sysstat result record layout. sysstat result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO	3		value for yellow alert
red	oat	NO	5		value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value
loadavg	oat	NO	0		The load average as reported by the system
user	tinyint	NO	0		CPU user time
system	tinyint	NO	0		CPU system time
idle	tinyint	NO	0		CPU idle time

swapin	int	NO	0		Amount of blocks swapped in from disk
swapout	int	NO	0		Amount of blocks swapped out to disk
blockin	int	NO	0		Amount of blocks read from block devices
blockout	int	NO	0		Amount of blocks written to block devices
swapped	int	NO	0		Amount of blocks written to swap device
free	int	NO	0		Free memory
buffered	int	NO	0		Amount of memory used for OS buffers
cached	int	NO	0		Amount of memory used for disk buffers
used	int	NO	0		Amount of memory used by processes
systemp	tinyint	NO	0		System temperature in Celsius

B.13. errlog - System error log analysis

B.13.1. errlog result record layout

Table errlog attributes. errlog attributes

Name	Type	Required	Default	Description
host	NMTOKEN	NO		host where this element originated

Table errlog elements. errlog elements

Name	Optional	Description
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.13.2. errlog database layout

Table errlog de nition record layout. errlog de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO			value for yellow alert
red	oat	NO			value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing

Table errlog result record layout. errlog result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO			value for yellow alert
red	oat	NO			value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value

B.14. diskfree - Free disk space

B.14.1. diskfree result record layout

Table diskfree attributes. diskfree attributes

Name	Type	Required	Default	Description
host	NMTOKEN	NO		host where this element originated

Table diskfree elements. diskfree elements

Name	Optional	Description
ipaddress	NO	target ip address
date	NO	date/time for this result
expires	NO	when this result expires
color	NO	color as this probe thinks it should be
received	NO	date/time this result was received by the upwatch server

B.14.2. diskfree database layout

Table diskfree de nition record layout. diskfree de nition record layout

Field	Type	Key	Default	Extra	Description
id	int	PRI		auto_increment	probe unique numerical id
pgroup	int unsigned	NO	2		group id
server	int	NO	1		server id
contact	int unsigned	YES	1		user eld: pointer to contact database
notify	int unsigned	YES	1		noti er id
ipaddress	varchar(15)	YES			target ipaddress
description	text	NO			description
freq	smallint unsigned	NO	1		frequency in minutes
yellow	oat	NO			value for yellow alert
red	oat	NO			value for red alert
disable	enum('yes', 'no')	NO	no		disable this probe
hide	enum('yes', 'no')	NO	no		hide probe results from viewing

Table diskfree result record layout. diskfree result record layout

Field	Type	Key	Default	Extra	Description
id	bigint unsigned	PRI		auto_increment	unique id for result
probe	int unsigned	YES	1		probe identi er
yellow	oat	NO			value for yellow alert
red	oat	NO			value for red alert
stattime	int unsigned	YES	0		time when result was generated
color	smallint unsigned	YES	200		color value

Appendix C. Adding a probe

C.1. So you want to add a probe?

Are you really sure? Adding a probe involves writing C code, creating and designing database tables and queue result les, creating PHP pages, and PHP graphs, writing documentation and submitting these changes to CVS. It is a lot of work, how rewarding it may be.

In the following overview we'll show you how to add a probe. All man-pages, spec- les, documentation will be auto-generated if you follow instructions below.

C.2. Basic steps for adding a probe

C.2.1. Think carefully before you start

You should not think lightly of adding a probe. Think things over before you start. Isn't there a probe available you can use? Or maybe you can get away with extending an existing one? If not, perhaps you can copy and modify one?

If not, you're in for the rewarding process of adding a new probe. Go to the next step.

C.2.2. Pick a real good name

You should think of a real good descriptive name for your probe. It should describe exactly what it does. Leave room for future probes that do something similar, also think about future extension to the probe itself.. Don't be satis ed too soon. For the rest of this small tutorial, we'll assume you probe willed be called `cputemp` for monitoring the host CPU temperature (which in fact already exists as part of the `sysstat` probe).

C.2.3. Design the Database Fields

Also a probe needs to enter its status into pr_status, and its history into pr_history and you should add code for this in the uw_process part..

C.2.4. Write the code

Create a new directory named `uw_cputemp`, copy all les in `templates/probe` to it. Look into those les, do a search and replace all occurrences of `template` with `cputemp`. Go one directory up, edit `configure.in`. Add a line `uw_cputemp/Makefile` to the `AC_CONFIG_FILES` section. Add `uw_cputemp` to the `PROGNAMES` variable in `Makefile.am`. Run:

```
$ ./autogen.sh
$ ./con gure
$ cd uw_cputemp
$ make clean
$ make
```

No errors should show up.

Now start coding in uw_cputemp/run.c , speci cally in the function run() . You should have enough examples in the other probes. Basically the probe should read a list of probede nitions from a database, execute all probes, and writes the results into a spool le. There are utility functions for doing this in libupwatch.

Important: Test your code thoroughly for memory leaks and error conditions.

C.2.5. Write the uw_process extension

The output of your probe is processed by uw_process . You should add a new source le called process_cputemp.c , that reads the probe results and writes them to the database tables. Be careful for the logic in this part. Add an entry in the struct _probe_proc array in uw_process/run.c , and a extern int process_cputemp(char *spec, GString *remark); just above it.

C.2.6. Add PHP pages_to_CMS

- Copy all *.php les from templates/php-cms to the en/database directory. Rename every *template* le to *cputemp*.
- Copy the pr_template_def.rec le to /home/cms/home/cms/www/php/cms . Do the usual replace, and take care this def le re ects the layout of pr_cputemp_def table in the database.
- Go to www/php/cms/custforms.php . Find the line which says START OF PROBES. Add an entry to the $f_probes array.
- Copy a section of another probe. Adapt as needed.
-

Important: Create an empty record in the database with id = 1i

C.2.7. Write documentation

Note that every probe is documented brie y in the cmd_options.def le. You should also document the probe in this manual. Personally I use KDE's Kate with the XML plugin. Go the the doc directory. Copy template-specs.xml to cputemp-specs.xml . Add a line to probesspecs.xml , and an ENTITY line at the top of upwatch.xml . Add cputemp-specs.xml to the XMLFILES line in Makefile.am . Rerun ./autogen.sh, ./con gure in the top directory. Run make in the doc directory.

C.3. Non-standard Probes

To be done

Index