Upwatch documentation

Ron Arts

Upwatch documentation

by Ron Arts

Table of Contents

List of Tables

Preface

People, especially managers, like to have facts and figures 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 scalable, fast, extensible, built on proven opensource tools, and is built not to loose data.

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 file, or .RPM’s.

1.2. Requirements

1.2.1. Run-time requirements

Run time requirements differ per probe. Look in the corresponding .def file (or in the spec file 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 2.0.1
- gnet 1.1.2
- mysql 3.23.49
- cURL 7.9.5
- libnet 1.0.2a
- libpcap 0.6.2

1.2.2. Build requirements

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

- autogen 5.3.6
- libxslt 1.0.15
- docbook 1.48
- 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
$ ./configure
$ make
```

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

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 follow the following procedure as root:

```
$ mkdir /etc/upwatch.d
$ cp config/upwatch.conf /etc
$ for i in uw_*
> do
> cp $i/$i /usr/sbin
> cp $i/$i.conf /etc/upwatch.d
> done
$ for i in uw_send uw_process uw_notify uw_traceroute
> do
> mkdir -p /var/spool/upwatch/$i/tmp
> mkdir -p /var/spool/upwatch/$i/new
> done
```

1.5. Securiy considerations

All of upwatch can be run as a non-root user, except the probes which need root-access, most notable, uw_ping. You can assign each probe its own user and grant that user access rights to its own database tables. The probes only read from the `pr_xxx_def` and the `server` tables.

1.6. Database

Create the database as follows:

Chapter 2. Configuration

Chapter 3. Administration

Chapter 4. How it all works

4.1. General Overview

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

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

- Definition 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 definition table contains, of course, the definition of this particular probe, this is of course probe specific but in practive usually contains frequency, target ip address, port number, current status.

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

Raw results are compressed into period tables in the foilowing way (using week as an example): the week is divided into 200 equal timeslots. For computing the plot values for a slot the process averages and takes the minimum and maximum value of all day-values in that timeframe. The same process happens for the month and year tables. This way we ensure that we never have to read more then 200 database records to produce a graph for a day, week, month, year or 5-year period.

4.2. What a probe does

What a probe does

4.3. What happens to the probe results?

What happens to the probe results

4.4. uw_process: storing results in the database

uw_process: storing results in the database

4.5. Scaling up

Scaling up

Appendix A. Interfaces and file layouts

A.1. Probe result file

Every probe result is written into a queue file. This file will be picked up by the process emptying the queue, usually uw_send, or uw_process. The file must have a specific name, and a specific layout.

A.1.1. Probe file name

The name of the file is composed of the current epoch time in seconds, microseconds, process id, and hostname, 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 file layout

The probe result file consists of ASCII lines, separated by linefeeds. The first line has a special format, any following lines are free format, can contain an error text, or some detail text like the HTTP header on a HTTP probe. The format of the first line is:

Table Header record layout. Header record layout

Field:	*Type*
method	Probe name (for example ’ping’)
lines	Total number of lines including this one
probeid	Probe id from the database
user	username for logging into uw_accept
password	password for loggin in to uw_accept
date	current date in seconds since epoch
expires	When this probe expires (seconds since epoch)
ipaddress	Probe target ip address
color	200 (green), 300 (yellow) or 500 (red)
	The following fields are probe specific

A.1.3. ping file layout

A.2. uw_accept protocol

Appendix B. Probe specifications

B.1. Iptraf-any

The iptraf-any probe measures ip traffic to or from a specified ip address passing a certain point. Currently it supports iptables/netfilter setup. It is used to measure data traffic at some network gateway.

B.1.1. Iptraf-any result record

Table iptraf-any record layout. iptraf-any record layout

Field:	*Type*
.....	*Standard header fields...*
outload	Outgoing link load
inload	Incoming link load
outbytes	Outgoing bytes in this timeslice
inbytes	Incoming bytes in this timeslice

Any lines following the header record contain error messages.

B.1.2. iptraf-any database layout

B.1.3. uw_iptraf-any daemon

There is no such thing. There *is* a shell script though, it reads the iptables, and produces a queue file.

B.2. Ping

The ping probe sends a configurable amount of ping packets to the target host and measures reply time. How many packets it can miss until turning yellow, or red is configurable. The packets send are normal ICMP packets as are send by the infamous ‘ping’ command.

B.2.1. Ping result record

Table ping result record layout. ping result record layout

Field:	*Type*
.....	*Standard header fields...*
minpingtime	Shortes measured ping time
avgpingtime	Average measured ping time
maxpingtime	Longest measured ping time
hostname	hostname that was pinged

Any lines following the header record contain error messages.

B.2.2. Ping database layout

Table ping result record layout. ping result record layout

Field	Type	Null	Key	Default	Extra
id	int(8)		PRI	NULL	auto_increment
server	int(8)			0	
address	varchar(80)				
freq	int(4)			1	
count	int(2) unsigned			5	
yellowmiss	int(2) unsigned			1	
redmiss	int(2) unsigned			3	
stattime	int(10) unsigned			0	
expires	int(10) unsigned			0	
color	smallint(5) unsigned			0	
lasthist	int(10) unsigned			0	
message	text				

B.2.3. uw_ping daemon

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 files, creating PHP pages, and PHP graphs, writing documentation and submitting these changes to CVS. It is a lot of work, though potentially very rewarding.

Index