NAME

rrdupdate − Store a new set of values into the RRD

SYNOPSIS

rrdtool {**update** | **updatev**} *filename* [**−−template**|**−t** *ds-name*[**:***ds-name*]...] [**−−skip−past−updates**|**−s**] [**−−daemon**|**−d** *address*] [**−−**] **N:***value*[**:***value*]... *timestamp***:***value*[**:***value*]... *at-timestamp***@***value*[**:***value*]...

DESCRIPTION

The **update** function feeds new data values into an **RRD**. The data is time aligned (interpolated) according to the properties of the **RRD** to which the data is written.

updatev This alternate version of **update** takes the same arguments and performs the same function. The *v* stands for *verbose*, which describes the output returned. **updatev** returns a list of any and all consolidated data points (CDPs) written to disk as a result of the invocation of update. The values are indexed by timestamp (time_t), RRA (consolidation function and PDPs per CDP), and data source (name). Note that depending on the arguments of the current and previous call to update, the list may have no entries or a large number of entries.

Since **updatev** requires direct disk access, the **−−daemon** option cannot be used with this command.

filename The name of the **RRD** you want to update.

−−template|**−t** *ds-name*[**:***ds-name*]...

By default, the **update** function expects its data input in the order the data sources are defined in the RRD, excluding any COMPUTE data sources (i.e. if the third data source **DST** is COMPUTE, the third input value will be mapped to the fourth data source in the **RRD** and so on). This is not very error resistant, as you might be sending the wrong data into an RRD.

The template switch allows you to specify which data sources you are going to update and in which order. If the data sources specified in the template are not available in the RRD file, the update process will abort with an error message.

While it appears possible with the template switch to update data sources asynchronously, **RRDtool** implicitly assigns non-COMPUTE data sources missing from the template the **UNKNOWN** value.

Do not specify a value for a COMPUTE **DST** in the **update** function. If this is done accidentally (and this can only be done using the template switch), **RRDtool** will ignore the value specified for the COMPUTE **DST**.

The caching daemon rrdcached cannot be used together with templates yet.

−−skip−past−updates|**−s**

When updateing an rrd file with data earlier than the latest update already applied, rrdtool will issue an error message an abort. This option instructs rrdtool to silently skip such data. It can be useful when re-playing old data into an rrd file and you are not sure how many updates have already been applied.

−−daemon|**−d** *address*

If given, **RRDTool** will try to connect to the caching daemon rrdcached at *address*. If the connection is successfully established the values will be sent to the daemon instead of accessing the files directly. If the connection cannot be established it will fall back to direct file-access. While this is convenient, it can silently create problems so please read the warning in the examples.

For a list of accepted formats, see the **−l** option in the rrdcached manual.

{**N** | *timestamp*}**:***value*[**:***value*]...

The data used for updating the RRD was acquired at a certain time. This time can either be defined in seconds since 1970−01−01 or by using the letter 'N', in which case the update time is set to be the current time. Negative time values are subtracted from the current time. An

AT_STYLE TIME SPECIFICATION (see the *rrdfetch* documentation) may also be used by delimiting the end of the time specification with the '@' character instead of a ':'. Getting the timing right to the second is especially important when you are working with data-sources of type **COUNTER**, **DERIVE**, **DCOUNTER**, **DDERIVE** or **ABSOLUTE**.

When using negative time values, options and data have to be separated by two dashes (`--`), else the time value would be parsed as an option. See below for an example.

The remaining elements of the argument are DS updates. The order of this list is the same as the order the data sources were defined in the RRA. If there is no data for a certain data-source, the letter **U** (e.g., N:0.1:U:1) can be specified.

The format of the value acquired from the data source is dependent on the data source type chosen. Normally it will be numeric, but the data acquisition modules may impose their very own parsing of this parameter as long as the colon (**:**) remains the data source value separator.

ENVIRONMENT VARIABLES

The following environment variables may be used to change the behavior of `rrdtool update`:

RRDCACHED_ADDRESS

If this environment variable is set it will have the same effect as specifying the `--daemon` option on the command line. If both are present, the command line argument takes precedence.

RRDCACHED_STRIPPATH

If this environment variable is set it will strip the leading string from the filename prior to sending the filename to rrdcached. This is mostly intended to allow rrdcached to work with xymon and cacti tools without having to modify those tools.

EXAMPLES

- `rrdtool update demo1.rrd N:3.44:3.15:U:23`

  Update the database file demo1.rrd with 3 known and one **UNKNOWN** value. Use the current time as the update time.

- `rrdtool update demo2.rrd 887457267:U 887457521:22 887457903:2.7`

  Update the database file demo2.rrd which expects data from a single data-source, three times. First with an **UNKNOWN** value then with two regular readings. The update interval seems to be around 300 seconds.

- `rrdtool update demo3.rrd -- -5:21 N:42`

  Update the database file demo3.rrd two times, using five seconds in the past and the current time as the update times.

- `rrdtool update --daemon unix:/tmp/rrdd.sock demo4.rrd N:23`

  Use the UNIX domain socket `/tmp/rrdd.sock` to contact the caching daemon. If the caching daemon is not available, update the file `demo4.rrd` directly. **WARNING:** Since a relative path is specified, the following disturbing effect may occur: If the daemon is available, the file relative to the working directory **of the daemon** is used. If the daemon is not available, the file relative to the current working directory of the invoking process is used. **This may update two different files depending on whether the daemon could be reached or not.** Don't do relative paths, kids!

AUTHORS

Tobias Oetiker <tobi@oetiker.ch>, Florian Forster <octo at verplant.org>