Pi Weather Station RRD Data Correction

Introduction

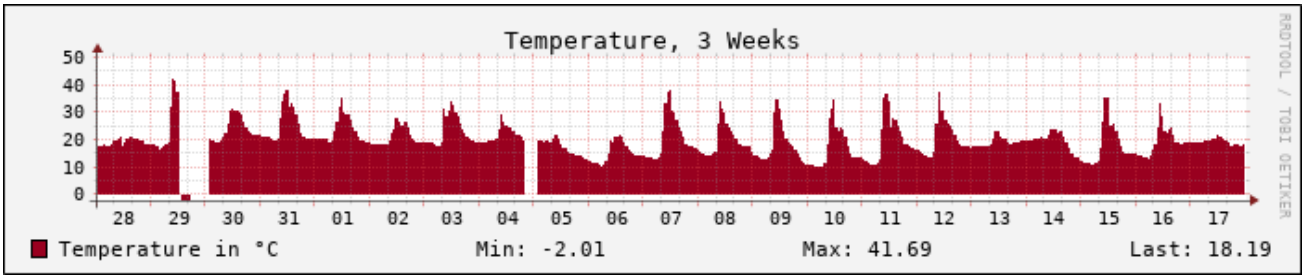
The PI weather station writes sensor data into the RRDtool database - https://oss.oetiker.ch/rrdtool/.

In severe weather conditions, a sensor may fail which results in recording wrong values, such as high negative temperatures at -37 C, humidity at 0%, or barometric pressure at 600 hPa.

Although the sensor update code function eliminates only very short one-time outliers, sensor failures occurring over several minutes to hours will create wrong database entries. Below are outlier.log entries showing the start of a failure.

```
Mon, 29 Mar 2021 13:35:01 +0900 [1616992446 Temp=-2.00*C Humidity=0.00% Pressure=nanPa] ->
temperature outlier [-2.00].
Mon, 29 Mar 2021 13:36:01 +0900 [1616992506 Temp=99.73*C Humidity=100.00% Pressure=406245.38Pa] ->
temperature outlier [99.73].
...
Mon, 29 Mar 2021 16:22:01 +0900 [1617002466 Temp=-89.14*C Humidity=0.00% Pressure=909878.38Pa] ->
temperature outlier [-89.14].
...
Sun, 04 Apr 2021 19:29:01 +0900 [1617532086 Temp=-2.00*C Humidity=0.00% Pressure=nanPa] ->
temperature outlier [-2.00].
Sun, 04 Apr 2021 19:29:01 +0900 [1617532086 Temp=-2.00*C Humidity=0.00% Pressure=nanPa] ->
humidity outlier [0.00].
...
Mon, 05 Apr 2021 01:05:01 +0900 [1617552246 Temp=-143.99*C Humidity=0.00% Pressure=165870.38Pa] ->
temperature outlier [-143.99].
```

Sensor failures on March 29 and April 4, shown as gaps in the database-extracted temperature graph:



The Bosch BME280 sensor often recovers from these failures when weather conditions improve. Appropriate sensor protection from direct water intake in severe storms is very important. Here is the procedure to manually update wrong sensor data inside RRD to restore proper data aggregation. Keeping wrong recorded values distorts the average, and sets from min/max values.

Identify Failed Database Entries:

The first step is to create the instructions for database record updates in the file **`rrdrepair.vi`** located under **`/home/pi/pi-weather/install`**. We extract the database values as readable XML format into the tmp folder:

```
pi@pi-ws01:~/pi-weather/install $ rrdtool dump /home/pi/pi-ws01/rrd/weather.rrd /tmp/tmp.xml
pi@pi-ws01:~/pi-weather/install $ ls -l /tmp/tmp.xml
-rw-r--r-- 1 pi pi 13307568 Apr 26 09:47 /tmp/tmp.xml
pi@pi-ws01:~/pi-weather/install $ head -3 /tmp/tmp.xml
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE rrd SYSTEM "http://oss.oetiker.ch/rrdtool/rrdtool.dtd">
<!-- Round Robin Database Dump -->
```

Now we identify failed records inside the XML extract. A sensor failure would record NaN values as below:

```
<!-- 2021-03-29 12:00:00 JST / 1616986800 --> <row><v>3.744648925e+01</v><v>2.866359677e+01</v><v>1
<!-- 2021-03-29 13:00:00 JST / 1616990400 --> <row><v>NaN</v><v>NaN</v><v>NaN</v><v>NaN</v></row>
<!-- 2021-03-29 14:00:00 JST / 1616994000 --> <row><v>-1.964666667e+00</v><v>0.000000000e+00</v><v>
<!-- 2021-03-29 15:00:00 JST / 1616997600 --> <row><v>-2.000000000e+00</v><v>0.000000000e+00</v><v>
<!-- 2021-03-29 16:00:00 JST / 1617001200 --> <row><v>-2.000166667e+00</v><v>0.000000000e+00</v><v>
<!-- 2021-03-29 17:00:00 JST / 1617004800 --> <row><v>-2.011205882e+00</v><v>4.852941176e-01</v><v>
<!-- 2021-03-29 18:00:00 JST / 1617008400 --> <row><v>NaN</v><v>NaN</v><v>NaN</v><v>NaN</v></row>
<!-- 2021-03-29 19:00:00 JST / 1617012000 --> <row><v>NaN</v><v>NaN</v><v>NaN</v><v>NaN</v></row>
<!-- 2021-03-29 20:00:00 JST / 1617015600 --> <row><v>NaN</v><v>NaN</v><v>NaN</v><v>NaN</v></row>
<!-- 2021-03-29 21:00:00 JST / 1617019200 --> <row><v>NaN</v><v>NaN</v><v>NaN</v><v>NaN</v></row>
<!-- 2021-03-29 22:00:00 JST / 1617022800 --> <row><v>NaN</v><v>NaN</v><v>NaN</v><v>NaN</v></row>
<!-- 2021-03-29 23:00:00 JST / 1617026400 --> <row><v>NaN</v><v>NaN</v><v>NaN</v><v>NaN</v></row>
<!-- 2021-03-30 00:00:00 JST / 1617030000 --> <row><v>NaN</v><v>NaN</v><v>NaN</v><v>NaN</v></row>
<!-- 2021-03-30 01:00:00 JST / 1617033600 --> <row><v>NaN</v><v>NaN</v><v>NaN</v><v>NaN</v></row>
<!-- 2021-03-30 02:00:00 JST / 1617037200 --> <row><v>2.033110000e+01</v><v>6.907510000e+01</v><v>1
```

Create Database Record Update Instructions:

The database update instructions will be written into a VI instruction file named **`repair.vi`**. We can re-use a old **`repair.vi.<date>`** backup as an example. The content typically has a positioning instruction that identifies the wrong entry by its data and time, followed by a substitute instruction that updates the wrong values. Important is the correct escaping of forward slash (/) and dot (.) characters to replace correctly.

```
pi@pi-ws01:~/pi-weather/install $ cp rrdrepair.vi.20190803 rrdrepair.vi
pi@pi-ws01:~/pi-weather/install $ vi rrdrepair.vi
:" ############################################################
:" rrdrepair.vim: This file is read by rrdrepair.sh as input.
:" Below lines have been prepared to fix the RRD data lines
:" with wrong sensor readings, e.g. -13.7 Celsius instead of 30
:" ############################################################
:/<!-- 2021-03-29 13:00:00 JST \/ 1616990400 --> <row><v>NaN
:.,+12:s/NaN<\/v><v>NaN<\/v><v>NaN<\/v><v>NaN/2.844648925e+01<\/v><v>2\.866359677e+01<\/v><v>1\.00
0874246e+05<\/v><v>0\.                 000000000e+00/
:1
:/<!-- 2021-03-29 14:00:00 JST \/ 1616994000 --> <row><v>-1\.964666667e+00
:.,+4s/-
[12]\.........e+00<\/v><v>.\.........e.0.<\/v><v>NaN/2.944648925e+01<\/v><v>2\.866359677e+01<\/v
><v>1\.0008742                46e+05/
:1
:" 1st set done
```

For position control, its good practice to jump back to the beginning of the file with ':1'.

Test Database Record Update Instructions:

The DB instructions should be tested on the temporary extracted DB. Verify the update of the correct records. Check correct line selection, and check correct update format, e.g. no missing field markers.

Update the DB:

To make the DB update run **`rrdrepair.sh.`** If run without options, the update is simulated. The argument '-p' does the real production update. The script catches the moment right after the last DB update by monitoring the file age. With an update frequency of every minute, there are 60 seconds time to repair before the next data update comes in.

```
pi@pi-ws01:~/pi-weather/install $ ./rrdrepair.sh
rrdrepair.sh start: Mon Apr 26 12:30:57 JST 2021
############################################################
# 1. Check for prd argument, and load the config file
############################################################
rrdrepair.sh: Reading file [../etc/pi-weather.conf] with [17] values
rrdrepair.sh: test run, simulating execution. Use ./rrdrepair.sh -p for PRD run.
Done.

############################################################
# 2. Catch the time after the existing RRD DB was updated
############################################################
date: Mon Apr 26 12:30:57 JST 2021 file update: Mon Apr 26 12:30:11 JST 2021 rrdage: 46
...
```