

TempTrax temperature graphing with MRTG  
=====

Author: Marc Bigler / oriented.net

Summary: This document will guide you through configuring MRTG to graph and monitor the temperature using a TempTrax device.

Pre-requisites: - UNIX operating system like Linux  
- PERL  
- Device-SerialPort PERL module  
- MRTG

Optional:  
- Web server (for example Apache)

MRTG configuration part  
-----

MRTG was initially created to graph the incoming and outgoing network traffic from a router's interface card using the SNMP protocol. With it's big gain in popularity it has been enhanced and can now graph anything simply by using any kind of shell script including PERL and this is how we are going to use it.

MRTG will execute every 5 minutes a PERL script. This script opens the serial device where the TempTrax is connected to get the current temperature, MRTG will then read in this value, process it and generate the new graph. MRTG per default generates a daily, weekly, monthly and yearly graph it will even generate you an HTML page which can be used with your web server to nicely display those graphs.

First of all you will need to install MRTG on your system, for that you will have to follow the well documented MRTG installation guide which you can find on it's web site or in the MRTG package itself.

Once installed you will need to create a configuration file which will let you control everything about your graphs. To start with here is a sample configuration file which I will explain in more details:

```
# -----  
1: WorkDir: /opt/apache/htdocs/mrtg  
2: Refresh: 600  
  
3: Options[_]: growright,nopercent,gauge,noi,noinfo,nolegend,integer,nobanner  
  
4: XSize[_]: 450  
5: YSize[_]: 150  
  
6: YLegend[_]: Temperature (c)  
7: ShortLegend[_]: C  
8: Legend0[_]: Temp :  
9: YTicks[_]: 10  
  
10: Target[TEMPTRAX2000]: `/opt/scripts/temptrax.pl`  
11: MaxBytes[TEMPTRAX2000]: 50  
XX: #WithPeak[TEMPTRAX2000]: wm  
12: Title[TEMPTRAX2000]: oriented.net server room temperature
```

```
13: PageTop[TEMPTRAX2000]: <H1>oriented.net server room temperature</H1>
14: <TABLE>
15:   <TR><TD>Maintainer:</TD> <TD>Marc Bigler</TD></TR>
16:   <TR><TD>Description:</TD><TD>TempTrax 2000 Sensor</TD></TR>
17: </TABLE>
# -----
```

WARNING: If you use this sample configuration file don't forget to remove the number and the colon as it is only for explanation.

[1]: The workdir is the directory where you want MRTG to save the graphs, i would recommend you to be it a directory in your web server's document root. If you do so, you will be able to access those graphs from anywhere on your network by accessing your web server. If you don't use a web server then the directory doesn't mind. In my sample I am having MRTG save the graphs in the mrtg sub-directory of my Apache web server's root directory /opt/apache/htdocs.

[2]: This defines the refreshing rate in seconds of the HTML page which is generated by MRTG along with the graphs. I've got it here setted to 6 minutes.

[3]: The options configuration directive lets you configure quite a lot of boolean switches which directly affect the generated graph. I've found that those options are the most suited for temperature graphing. Explained quickly I want my graph to grow on the right side (growright), I don't want to display percentages (nopercent), the value of my temperature is a current status measurement and not an incrementing counter (gauge), don't display the second graph line which is not required because we have only one variable which is our temperature (noi), we do not want to display the uptime and device name as this is not valid for our case (noinfo), we don't want any extra legends (nolegend), we are using an integer value and want the summary lines below the graphs also as integers without comma (integer), we choose not to display the MRTG banner (nobanner).

[4-5]: By default all graphs generated by MRTG are 100x400 pixels wide, I want them to be slightly a bit larger, as you've guessed XSize is it's width and YSize is it's height. The unit is in pixels.

[6]: Here we define the legend of our Y axis which is the temperature and will be displayed along the Y axis. The X axis is the time but that's the default so we only need to change Y's axis legend.

[7]: A short version of the legend which will be used for the summary lines under the graph, the unit is a good choice as legend for example F or C, depending if you use Fahrenheit or Celcius.

[8]: This string will be displayed at the summary line under the graph.

[9]: The YTicks option will let use define the number of vertical lines displayed in the graph, I have changed this to 10 which will display 10 vertical dotted lines, this gives a better readability to the graph.

[10]: Here we come to the most important piece of the configuration, the Target option here is used to call our script which will return the temperature. We will talk later about the script, it's important here that you point to the correct path to the script and don't forget the "`" before and after the script's full path. This character is well-known in UNIX scripts, it permits to run a script for example.

[11]: The MaxBytes parameter has a misleading name but you would maybe have guessed it is to define the maximum value of our X axis, which is of course the temperature. MRTG uses this parameter to define the upper limit of the X axis. For those who use celcius as unit a value of 50 is a good choice, for those who use fahrenheit 100 should also be a good choice.

[12]: This is the title of the generated HTML page, this value is mainly used in the HTML <TITLE> tag.

[13-17]: This is HTML code which you would like to add at the top of the HTML page which MRTG generates. Usually as in my case it's just a title and some brief description but it can be anything for example you could use a <IMG> tag to display your company's logo.

With all these parameters you will already have a nice temperature graph but if you wish you can further customize it for example change the lines colors and so on. A good description of all the other extra parameters can be found on the website of MRTG.

Save this configuration file somewhere for example I have it in /etc/mrtg and have called the file mrtg.temptrax.cfg. From here on the next step is to have the PERL script in the right place specified in Target parameter on line 10.

PERL script

-----

Now for MRTG to work you need an interface which can speak or better which can retrieve the value of the TempTrax device. That's what the script defined in the Target parameter of MRTG's configuration file is for. Before you can use this script you will need to download and install the Device-SerialPort PERL module from CPAN. Just follow the installing instructions provided with the module.

Here is the script which will gather the temperature from the TempTrax and return it to MRTG:

```
# -----
#!/usr/bin/perl -w

use strict;

# Requires the Device-SerialPort PERL module

use Device::SerialPort;

# Define variables (change here to the device where your TempTrax is connected)

my $temptrax_device = "/dev/ttyS0";

# Initialize variables

my $pass;
my $return;
my @values;
my ($probel_value, $probe2_value, $battery_status);
```

```

my $temperature;

# Open serial device and set the connection parameters

my $ob = Device::SerialPort->new ($temptrax_device) || die "Can't open
${temptrax_device}: $!";

$ob->baudrate (9600) || die "fail setting baudrate";
$ob->parity ("none") || die "fail setting parity";
$ob->databits (8) || die "fail setting databits";
$ob->stopbits (1) || die "fail setting stopbits";
#$ob->handshake ("none") || die "fail setting handshake";
$ob->dtr_active (1) || die "fail setting dtr_active";

$ob->write_settings || die "no settings";

sleep 1;
# Send a dummy character to the TempTrax device to "wake it up", this will
return the temperature

$pass = $ob->write("a") or die ("Could not write to TempTrax: $!");

sleep 1;

# Process the output returned by the TempTrax device
if (($return = $ob->input) ne "") {
    $ob->write ($return);

    # Insert each value in it's respective variable
    ($probel_value, $probe2_value, $battery_status) = split("\r", $return);

    # Get the temperature in Fahrenheit
    chomp($temperature = $probel_value + 0);

    # Temperature is in Fahrenheit so we convert it here into Celcius
    $temperature = (($probel_value - 32) * 5) / 9;

    # Remove the decimal part as MRTG doesn't like decimals
    $temperature =~ s/([0-9]+\.[0-9]+)/$1/;

    # Print the result to be processed by MRTG

    print "0\n";
    print $temperature;
    print $battery_status, "\n";
    print "oriented.net server room\n" # This can be set to anything
} else {

    print "ERROR: Did not receive a temperature reading from TempTrax.\n";

```

```
}  
  
undef $ob;  
  
exit(0);  
  
# -----
```

If you know PERL the script is pretty self explaining, in a few words, it opens the device where the TempTrax is connected, retrieves the temperature and display is to the standard output.

The only two things which you will need to parameter is the device name of your TempTrax, in my case running Linux I have it connected on the first serial port of my PC (COM1) so the device is /dev/ttyS0. For a Sun Solaris box also on the first serial port device it would be /dev/term/a. So don't forget to modify the \$temptrax\_device variable at the beginning of the script if you are not running Linux and your TempTrax is not connected to COM1.

The second piece of code to tweak is if you want the output to be in Fahrenheit instead of Celcius, as I live in Switzerland our unit for the temperature is the Celcius. But for the folks living in the US it would be in Fahrenheit. I need to do some conversion in Celcius in the script because the TempTrax device, as it is built in the US, returns the temperature in Fahrenheit. So for those using Fahrenheit as temperature unit you will need to remove the following single line near the end of the script:

```
$temperature = (($probel_value - 32) * 5) / 9;
```

Then the output of the script will be in Fahrenheit. Now save this script in the same path and filename as you have used in MRTG's configuration file, in my example that is /opt/scripts/temptrax.pl. You will need to run this script as root or give the user which will run the script read access to the serial device. Also make sure the script is executable and that the path to PERL on the very first line of the script is correct. You should now try if your script works correctly by executing it like this:

```
/opt/scripts/temptrax.pl
```

A sample output would be:

```
0  
25  
Bat Ok  
oriented.net server room
```

The only important line of output for MRTG is the second one which is actually the temperature, here in Celcius. The other lines are required to fool MRTG as it expects this 4-line output format and won't work if it's not exactly like that.

Credits: This script is originaly from Sensatronics, I have found it on their website, I simply modified it to add support for the Celcius unit and to make it work with MRTG.

The next step is to execute MRTG automatically at regular intervals for that we will use UNIX crontab.

## UNIX crontab configuration part

-----

As you should know the crontab in UNIX lets you run anything at anytime or at regular intervals in background. We will use it to run MRTG with our configuration file every 5 minutes, this means that every 5 minutes the temperature will be read from the TempTrax and MRTG will re-generate the graphs taking in account the new value.

The easiest way is to edit the crontab of the root user with the following command on Linux:

```
crontab -e
```

and add the following lines at the end:

```
#  
# Crontab entry for MRTG  
#  
*/5 * * * * /opt/mrtg/bin/mrtg /etc/mrtg/mrtg.temptrax.cfg >/dev/null 2>&1
```

The value /opt/mrtg/bin/mrtg should point to you MRTG file which you installed at the beginning and the value /etc/mrtg/mrtg.temptrax.cfg should be the special configuration file of MRTG for TempTrax. The rest of that line simply tells to run MRTG every five minutes and to redirect all output to /dev/null.

The first two times MRTG is ran it won't produce much because it somehow initializes some files it needs. So you will have to wait a few minutes before seeing some output. All the files generated by MRTG will be located in the directory defined by the WorkDir parameter in the MRTG configuration file. If you use the computer where your TempTrax device is connected as a desktop computer you could point your browser to that directory. If it is a server I would recommend you to use Apache to display the graphs remotely on your local network or even onto the internet, as you wish.

## TIPS

----

If you have two probes on your TempTrax device you could slightly modify the MRTG configuration file and the PERL script to read the two values and have two lines in your graph, one per probe.

By adding the following parameter in the MRTG configuration file:

```
WithPeak[TEMPTRAX2000]: wmy
```

You can get MRTG to graph the maximum value, it will add a new line, per default the color is pink and will always represent the maximum value. Normally MRTG only graphs the line for the average value which is the one you see without this option. The "wmy" value instructs to draw that maximum line for the Weekly Montly and Yearly graphs. It's not possible to get an maximum value for the daily graph as it's already the real value.

## LINKS

-----

Here are all the links where you can find the software which is discussed in this document.

Device-SerialPort PERL module:

<http://search.cpan.org/author/COOK/Device-SerialPort-0.12/>

Apache web server:

<http://www.apache.org/>

MRTG itself:

<http://www.mrtg.org/>

PERL:

<http://www.perl.org/>

Linux:

<http://www.linux.org/>