Using gnuplot to display data in your Web pages

David Tansley

January 19, 2010

Use gnuplot to dynamically generate Web pages from your system using raw data to provide graphic images. This raw data typically contains MIS-related information, on system performance, storage, or database growth.

Introduction

Gnuplot is a tool used to generate trends and graphs. It is typically used for time-series-based data gathering, but is not restricted to this; static data can also be used. Gnuplot can be run in either batch or on the fly, with the results being presented by a graphics viewer or Web browser. This article demonstrates how to use gnuplot using a batch file to generate data collected from sar and other data sources. Gnuplot is very rich in command options using the set operator. However, to generate basic graphs using line or boxes, it requires some knowledge of the documentation.

For this demonstration, I will present the graphs using a Web server.

Gnuplot overview

Gnuplot converts raw tabular data into a graphic file image. Popular formats are png, pdf, and jpeg, and they can then be presented either on the fly or directly to an X terminal, Web page, or a general graphic viewer utility. The image can also be imported into documents. A command interface is used to interact with gnuplot using the 'set' commands to specify how the image is to be formatted and presented. Typically this includes the sizing of the graph, colors to be used, the scale, x,y coordinates, and the output image name. The plot command is then used to actually generate the image using the gnuplot engine. There is also the splot command that will draw 3-D graph images. Though commands can be carried out interactively using the command interface, it is best done with a config file, as this is reusable by using variables in the file (if required) using the shell 'here' document method. The config file is then piped through gnuplot to generate the image file. If there are any errors in the config file, these will be displayed during this process, highlighting where the error is. With the newly generated graphics file, the image can then be displayed.

As with any data-gathering process that is to be used to generate graphs, some tinkering of the data gathered will have to be carried out prior to piping through gnuplot. This filtering may well include deleting headers or trailers from the data file; sed and awk are your friends for any text-filtering requirements.

Installing Gnuplot

Gnuplot version 4.2 can be downloaded as an rpm from the AIX® 5L Source Packages website at:http://www.perzl.org/aix/index.php.

As well as gnuplot, you will also require a running http server for this demonstration.

The required dependent libraries and http server can also be downloaded from here or from the AIX toolbox website.

Make sure you have the following rpm libraries installed prior to installing gnuplot, as these are prequisites:

```
fontconfig-2.7.2-1.aix5.1.ppc.rpm
expat-2.0.1-2.aix5.1.ppc.rpm
freetype2-2.3.9-1.aix5.1.ppc.rpm
zlib-1.2.3-5.aix5.1.ppc.rpm
libpng-1.2.40-1.aix5.1.ppc.rpm
gd-2.0.35-4.aix5.1.ppc.rpm
libjpeg-7-1.aix5.1.ppc.rpm
libXpm-3.5.7-2.aix5.1.ppc.rpm
 gettext-0.17-1.aix5.1.ppc.rpm
glib2-2.20.5-1.aix5.1.ppc.rpm
```

To list the installed rpms use:

```
# rpm –qa
```

Finally, install the actual gnuplot package:

```
# rpm -ivh gnuplot-4.2.4-1.aix5.1.ppc.rpm
```

Run gnuplot to test. You will be presented with the gnuplot interface command (issue `quit` to exit the interface):

```
$ gnuplot
      G N U P L O T
        Version 4.2 patchlevel 4
        last modified Sep 2008
        System: 5.3
        Copyright (C) 1986 - 1993, 1998, 2004, 2007, 2008
        Thomas Williams, Colin Kelley and many others
        Type `help` to access the on-line reference manual.
        The gnuplot FAQ is available from http://www.gnuplot.info/faq/
        Send bug reports and suggestions to <http://sourceforge.net/projects/gnu
plot>
Terminal type set to 'unknown'
gnuplot>
```

Creating a graph with sar

sar is probably the most common method of gathering performance data, so let's use that as just one example. Listing 1, sarx.txt, contains the filtered output from running sar at an hourly interval for five hours.

Listing 1. sarx.txt

```
14:10:50        33  27    4     36      4.00
15:10:50        29  14    3     28      4.00
16:10:50        35  21    1     31      4.00
17:10:49        38 29    2     39      4.00
18:10:40        42  29    3     35      4.00
```

Next, create a file to contain all the gnuplot commands required to generate the graph. Tthe file can be called anything you like; in this demonstration, I am calling it sarx.conf.

To be able to generate the image, you must inform gnuplot what format the graphics file should be and how it should be presented visually. Listing 2, sarx.conf, contains the configuration file to issue the set operations. Any line starting with a # character is considered a comment. Let's look more closely at Listing 2.

```
set terminal png truecolor
```

First the terminal type is set, which informs gnuplot what the resulting image will be formatted into. This article uses the png (Portable Network Graphics) format.

```
set output "sarimage.png"
```

Next, inform gnuplot the actual output image filename. In this case, call it sarimage.png.

```
set autoscale
```

When generating graphs, you need to specify an x and y axis range for your data. This example tells gnuplot to make that decision and lets gnuplot calculate the range values. However, this can be overridden, as demonstrated later.

```
set xdata time
set timefmt "%H:%M:%S"
```

Because this example uses date values as a point of reference for the data, we inform gnuplot of this with the format of the values. In the example contained in Listing 1 sarx.txt it is in the format:

`Hour:Minute:Seconds`

Based on the UNIX date notation, note the use of double quotes surrounding the date variable. Other common formats are:

```
%d  - day of month 1 -31
%m - month of year 1 -12
%y  - year 0-99
%b  - three character of month name , ie: jan ,feb
%B  - name of month
```

If we had a date column in the format: `Hour-Minute`

Then this would be represented by`: set timefmt "%H-%M"`

```
set style data lines
```

When the graph is presented the data plotted should be in a smooth data line. Other common drawing formats are: dots, boxes, errorbars, candlesticks.

```
plot "sarx1.txt" using 1:2 title "%user", '' using 1:3 title "%sys"
```

Next, the graph is actually plotted or generated using the plot command. First, the data input file name is given, then we inform gnuplot what columns to plot. In this example we are going to use column 1 as the x axis, then plot column 2 data with the title ″%user″, followed by column 3 with the title ″%sys″. The titles or labels will be displayed at the top right of the graph. Columns 2 and 3 will use column 1 as their x value reference when plotting. Each 'using' statement in the plot command is separated by a comma. The use of the two single quotes is discussed in the next section.

To generate the image file, the format is:

```
cat < conf file> | gnuplot
```

For this demonstration, I would use:

```
$ cat sarx.conf | gnuplot
```

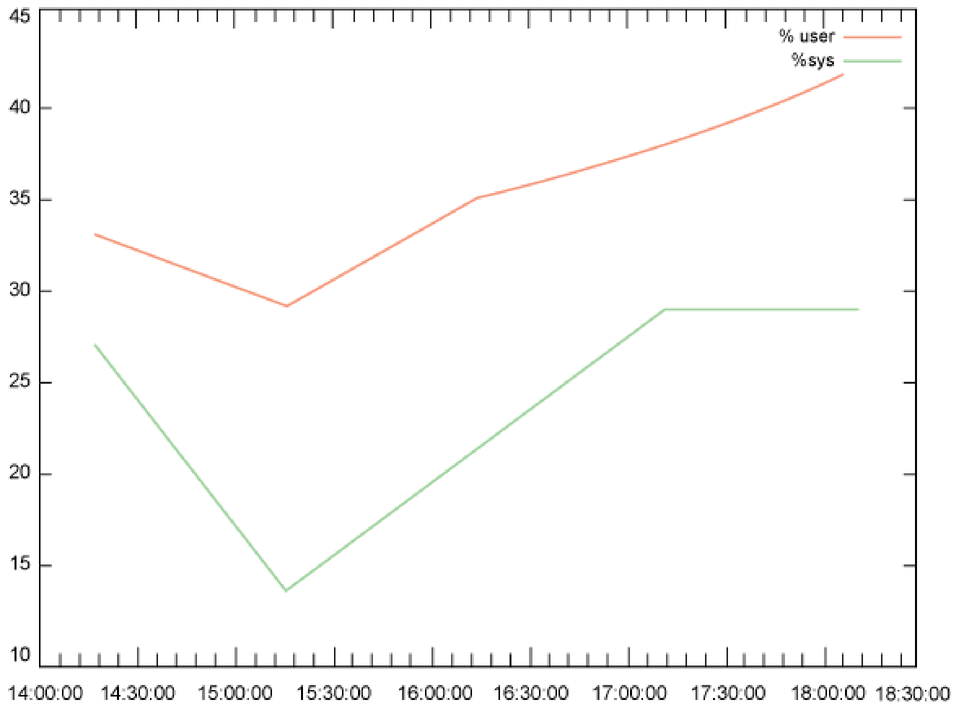
The sarimage.png file will now be produced. To view the image copy the file to your htdocs directory within your Web-server file system, for viewing.

Figure 1. sarimage, demonstrates the resulting output on a Web browser using the sample data provided.

Listing 2. sarx.conf

```
#sarx.conf
set terminal png truecolor
set output "sarimage.png"
set autoscale
set xdata time
set timefmt "%H:%M:%S"
set style data lines
plot "sarx.txt" using 1:2 title "%user", '' using 1:3 title "%sys"
```

Figure 1. sarimage



Presentation of the plots

Shortcuts in the command line can be used in the plot command. For example, after the initial plot command, any further options to the plot command can be abbreviated, using the first letter of the option. However, for this demonstration, I will only be abbreviating the input file, which is done using two single quotes instead of giving the input file again (sarx1.txt). The following demonstrates this explanation. The first example is the notation I am using in this demonstration, the second example is the abbreviated notation, and the third example gives the full command statement without any abbreviations. All three examples produce the same output.

```
plot "sarx1.txt" using 1:2 title "%user", '' using 1:3 title "%sys"

plot "sarx1.txt" using 1:2 title "%user", '' u 1:3 t "%sys"

plot "sarx1.txt" using 1:2 title "%user",\
"sarx1.txt'' using 1:3 title "%sys"
```

When displaying graphs to other users, it makes sense to sometimes include a worded label and a title for easy identification of the subject matter. To include an x and y label, use the xlabel and ylabel commands and enclose your text with quotes:

```
set ylabel " y line info here"
set xlabel " x line info here"
```

To include a title header for the graph, use the title command:

```
set title "main title info here"
```

Gnuplot uses its own default colors when generating a graph. By default, the graph generated will be on a white background; this makes sense, as the graph would probably be printed. However, you can change any of the colors using color codes denoted in the Hex range, prefixed with the letter x. The format for the hex codes are:

```
xrrggbb
```

A Google search of 'hex color codes' produces the full range of colors in Hex.

The order of the colors to override gnuplots default colors are:

```
background
border
X
Y
plotting lines
```

A light gray color in hex equates to: `C9C9C9`

To generate a graph with a light gray background I could then use:

```
set terminal png  xC9C9C9
```

Notice that in the above command I had to replace the truecolour option to the terminal type png, which overrides the default colors gnuplot uses. As I have not specified any more colors to override the default settings, gnuplot will then use its own default colors for the rest of the graph.

A grid can also be very useful as a point of reference when viewing graphs. Using the grid option, the following will inform gnuplot to insert a grid on the graph:

```
set grid
```

You can implicitly set the x and y coordinates; however, be sure to not specify the range that is less than the spread of data or no graph will be drawn. Using the sample data contained in Listing 1 you can see the range for the time x coordinates are from 14:10:50 through to 18:10:40.

The y coordinates range (that's columns 2 and 3) from 14 to 50.

With that information, you can specify your own ranges. This example uses the x range from 14:00 through to 18:15, and the y range from 10 to 50.

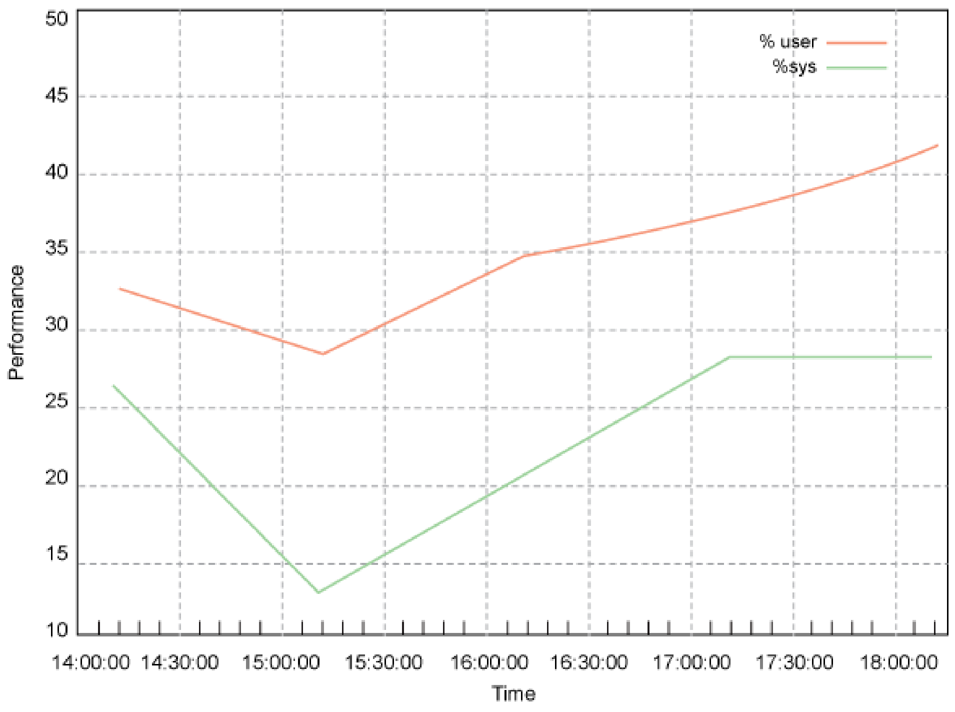
```
set xrange ["14:00:00" : "18:15:00"]
set yrange ["10:00" : "50:00" ]
```

In Listing 3, these changes are put into use, with the resulting graph output contained in Figure 2.

Listing 3. sarx2.conf

```
#sarx2.conf
set terminal png  xC9C9C9
set output "sarimage.png"
set autoscale
set xdata time
set timefmt "%H:%M:%S"
set ylabel "Performance"
set xlabel "Time"
set title "Sar Output Example"
set xrange ["14:00:00":"18:15:00"]
set yrange ["10:00" : "50:00" ]
set grid
set style data lines
plot "sarx1.txt" using 1:2 title "%user", '' using 1:3 title "%sys"
```

Figure 2. sarimage2



Gnuplot with histograms

Histograms or boxes can also be used to represent data and for some it is a more visual representation than using static data. Listing 4 represents data that contains the total number of user groups taken from an AIX applications box. Column one contains the actual AIX group names and column two contains the total members.

Listing 4. grpdata.txt

```
staff   54
apps    22
sybgrp  12
db2grp1 29
dasdm   8
dstage 21
dsgrp  8
batch  28
db2prd 1
```

To use histograms, simply inform gnuplot that the graph will be generated using a histogram:

```
set style data histograms
```

The default histogram generated will not have bold edges to their boxes and none of the boxes will be filled with color. However, if you specify a border, it may contain a double strike of bold lines at the bottom of the box, along the x axis line (Cosmetically, this wouldn't look right.).

Gnuplot draws the borders of the boxes in this order: top, bottom, left, and right, with their respective values: 1, 2, 4, 8. To erase one or more border lines, supply the sum of those values. In this example, the bottom border line is to be deleted, so that gets erased with the -1 option. Specifying the fill option fills the boxes with the default color:

```
set style fill solid 1.00 border -1
```

For the x coordinates, you are not using time spreads, but rather you are going to use the group names instead. Using the xtic option, this tells gnuplot to put the tic along the x axis along with the data labels (column one). In this case, it is the group names. However, there are occasions when supplying labels that contain many characters or indeed time formats for the xtics that will just not fit between the tics on the graph. You will discover overlaps in the labels. To avoid this, rotate the labels by 90 degrees (experiment to suit) so that they are vertical. You can achieve this using the following command:

```
set xtic rotate by 90
```

The data in column two is to use column one (x data) as reference:

```
 ( 2:xtic (1)
```

When generating the graph, give a title to the data "apps groups numbers":

```
plot "grpdata.txt" using 2:xtic(1) title "apps group numbers"
```

To generate the image grpimage.png, use:

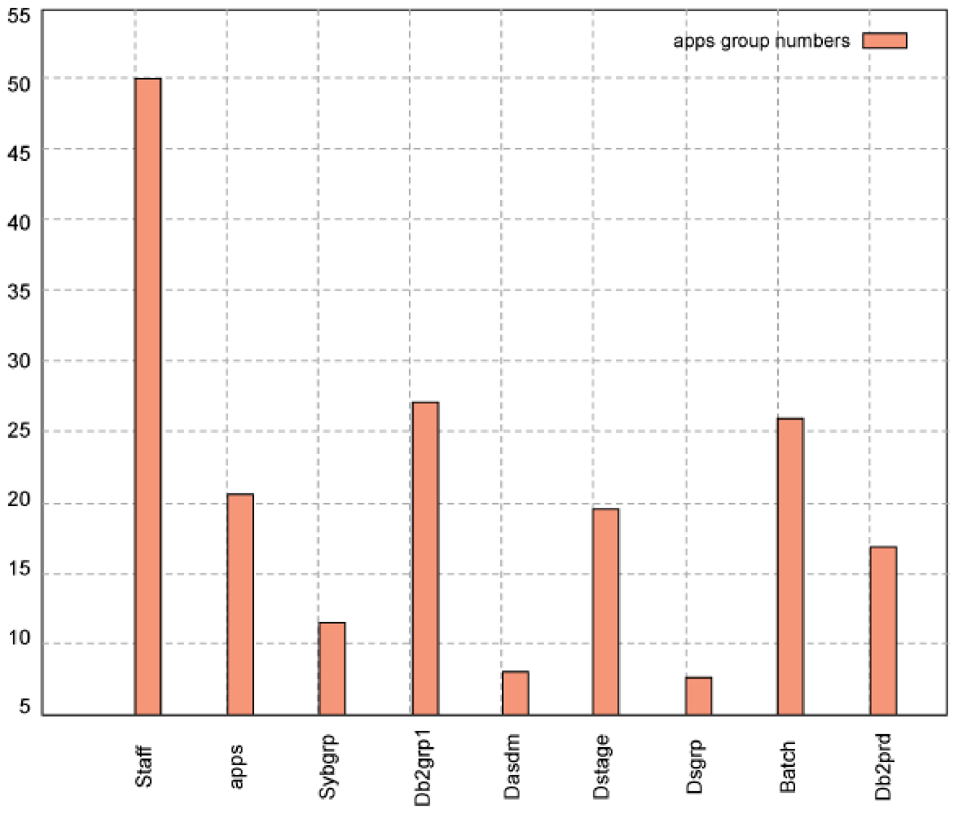
```
$ cat grphist.conf | gnuplot
```

Listing 5 contains the gnuplot commands to generate the image, which is displayed in Figure 3, below.

Listing 5. grphist.conf

```
# grphist.conf
set terminal png truecolor
set output "grpimage.png"
set grid
set xtic rotate by 90
set style data histograms
set style fill solid 1.00 border -1
plot "grpdata.txt"  using 2:xtic(1) title "apps group numbers"
```

Figure 3. grpimage



Of course, you can have more than one set of data drawn in histograms, as was shown in the sar example. Now look at another source of data. Listing 6 contains data that reflects data growth and reduction contained on disk arrays over a three-month period. Column one is the disk array name, column two is one month's disk growth usage, column three is the next month's disk growth usage, and the following month's growth is contained in column four.

Listing 6. disk.txt

```
hdisk2 420 425 410
hdisk3 700 780 760
hdisk4 450 450 452
hdisk5 680 702 690
hdisk6 320 330 329
hdisk7 530 515 514
```

The conf file for generating the data is contained in Listing 7. Looking more closely at the plot command, you again use the xtic command as you want to override gnuplot for the x axis data. Columns two, three, and four use the x axis as reference when generating the histogram. Once you have referenced that, column two is to use the x axis:

```
2:xtic(1)
```

Gnuplot assumes the rest of the columns to be plotted will reference that as well, so there is no need to specify the xtic again within the plot command:

```
plot "disk.txt"  using 2:xtic(1) title "Oct-09 data growth(gb)", '' using 3 title "N
ov-09 data growth(gb)", '' using 4 title "Dec-09 data growth(gb)"
```

To generate the image, use:

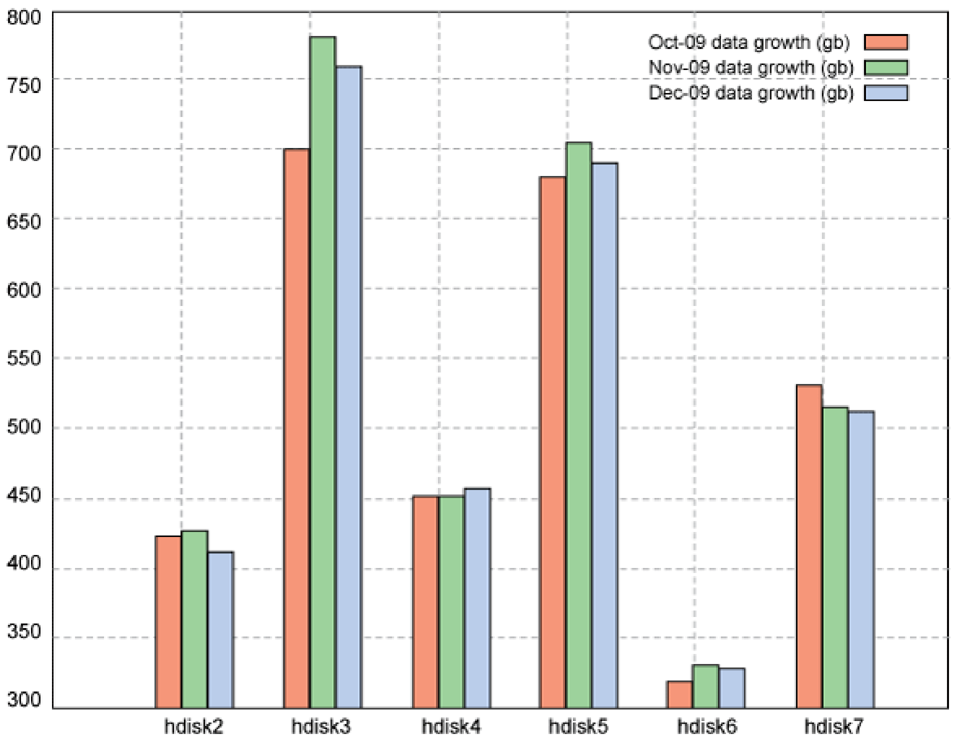
```
$ cat diskhist.conf | gnuplot
```

The resulting image is displayed in Figure 4.

Listing 7. diskhist.conf

```
# diskhist.conf
set terminal png truecolor
set output "diskimage.png"
set grid
set style data histograms
set style fill solid 1.00 border -1
plot "disk.txt"  using 2:xtic(1) title "Oct-09 data growth(gb)", '' using 3 title "N
ov-09 data growth(gb)", '' using 4 title "Dec-09 data growth(gb)"
```

Figure 4. diskimage



Conclusion

Gnuplot is a tool that can be used to generate different graphs from different types of data. To automate the process of graph generation using shell scripts, I recommend using the 'here' document method, as this process can be used to generate graphs using gnuplot on the fly. To stop the Web server from caching your graphs, use the appropriate HTML META tags (for example, the 'no-cache' statement) in your scripts.

Related topics

- Gnuplot Home Page provides downloads, FAQs, documentation, and much more.
- Download Gnuplot rpm
- >Download gnuplot prerequisite libraries