Desktopia

A THOUSAND WORDS

It's much easier to grasp information when it's seen rather than explained. When it comes to information about your PC the display of data is handled by the system monitor ProcMeter. Jo Moskalewski takes a look

Ensuring your system is secure entails more than being cautious when dealing with system-wide services or relying on a firewall to surround your computer or network. A major element of keeping a system secure is inspecting the system log files at regular intervals. There is one serious drawback to this however, it is usually a look into the past and is often only done following some disastrous mishap. If your memory is overflowing, your Linux computer will no longer be stable, but you may not be aware of a memory problem until the system has fallen over. Early detection is therefore a must and not just when it comes to your machine's health.

Control room

Fortunately, the Linux kernel keeps lots of relevant data readily available and up to date at all times. This information can be read out in the */proc* directory, the contents of which is created virtually by the kernel (and therefore does not actually take up any disk space, not even the biggest file in it, *kcore* – a replica of your working memory). A *cat /proc/meminfo* lists a slew of information to do with the status of the working memory (see Listing 1).

Listing 1: Working memory

```
                   total:       used:       free:
Mem:            393928704   368910336    25018368
Swap:           777719808       32768   777687040
MemTotal:          384696 kB
MemFree:            24432 kB
MemShared:              0 kB
Buffers:            17088 kB
Cached:            164664 kB
SwapCached:            32 kB
Active:             50348 kB
Inact_dirty:       128592 kB
Inact_clean:         2844 kB
Inact_target:          12 kB
HighTotal:              0 kB
HighFree:               0 kB
LowTotal:          384696 kB
LowFree:            24432 kB
SwapTotal:         759492 kB
SwapFree:          759460 kB
```

Now all that remains is to find the line with the information you seek, and when found it is quite easy to follow, which is not true in all cases. A *cat /proc/uptime* presents the user with the much greater task of interpreting a string of figures such as *10756.43 7123.55*. The only thing relevant here for the period since the last reboot is the first of the two blocks of figures, and this is shown in seconds. Some tedious conversion is now necessary to get to the more meaningful 2 hours, 59 minutes and 16.43 seconds.

Help wanted

Rummaging around in the */proc* directory is a highly systematic and reliable, although extremely time-consuming way of searching information. That's why there are system monitors, which call up and graphically process the kernel information constantly. One such representative is *ProcMeter*, which is now available in version 3.3a. A copy of this can be found on the coverdisc or else (together with additional information) at *http://www.gedanken.demon.co.uk/procmeter3/*.

This piece of software now informs you not only about RAM locks and times, but also about incoming emails, your power supply, hard disk activity and capacity, network traffic, CPU-utilisation, and so on and so forth. Everything that can be recorded you will see displayed either in a numerical – human readable – form, as a graphical display of the last few moments or as a bar with the current status (also graphical), all together, with or without a label. Each scrap of information can, if desired, sparkle in a range of different colour combinations.

Obviously there is also a catch to all this. Although everything can be adjusted by mouse on the fly it's not possible to save your customisations. The associated configuration file must be typed by hand, which also demands a bit of attention when you're under pressure. The effort does pay off though, especially since it only needs doing once.

If you want to convince yourself of the utility and the abilities of this tool, then a simple *tar –xvfz procmeter3-3.3a.tgz* will unpack this archive into your own, new directory. This now contains the source code but we don't have the executable program just yet. It can be moulded to fit your computer with the command *make all*; the result of which needs to be installed afterwards with a *make install*, for which you will briefly need root privileges.

Twinpack

The program itself comes in two main parts. As well as the actual, finely adjustable *procmeter3* there is also the *gprocmeter3* incarnation, which is based on the popular GTK+ toolkit, thus giving it a similar appearance to other GTK and GNOME applications. This becomes especially clear in the display of the menus, which in *gprocmeter3* are oriented, not to the manually typed configuration file, but to the elected theme for the system.

The first time you call up either of the two programs the effect is very sobering. Only the program name and the time and date are displayed in mouse-grey (Figure 1). To add a fourth element, press the right mouse button. A menu with all displayable information in all available display modes now pops up (all in all an impressive number, which on any computer must run into three figures). If you now select an element this will be added to the bottom of ProcMeter window. The sequence in which the elements are displayed can be altered by selecting one of the two Move To options from the left mouse menu and assigning which items you wish to move to a new spot.

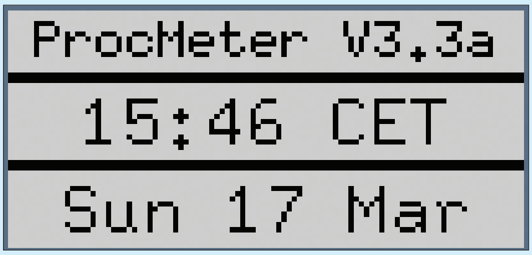
Internal

When the left mouse button is pressed, additional options can be accessed (Figure 2). Via Properties you can obtain additional information on the respective element (Figure 3). This menu entry becomes especially helpful when you want to transfer your settings into a configuration file because then the respective internal ProcMeter designators will also be displayed. If you do not close this window, it adapts its content at the next mouse click to the current element.

Another menu entry bears the title Run, which can be used to call up an external (and freely configurable) program, which varies from element to element. For example a mouse click on a memory element in the standard configuration results in the output of the command *free* being displayed in an xterm.

A little work

The much-bemoaned configuration file can be found in *~/.procmeterrc*. If this does not exist in your system, you can copy the skeletal default configuration from the unpacked tarball in its place and alter it accordingly. The structure of this file is simple and logical but can quickly become very large – and therefore very confusing. This is probably one reason why the author Andrew M Bishop split it into individual sections. First you should show ProcMeter the path to follow for its program files, or to be more precise, to its modules:



ProcMeter V3.3a

15:46 CET

Sun 17 Mar

Fig. 1: First impressions

```
[library]
path=/usr/local/lib/↵
X11/ProcMeter3/modules
```

These take over the actual consultation of your system, while ProcMeter merely outputs what the modules deliver. So there is a module for memory questions (*meminfo.so*), one for your processor (*stat-cpu.so*) etc. This modular structure has the advantage that it can be easily and quickly enhanced by new functions.

The *[library]* section is now followed in the *[start-up]* section by the elements to be displayed when the program is started. You can find out what these are called from the instructions, which can be viewed with *man procmeterrc* or else from the already described Properties dialog: it is advisable to click the structure together with the mouse in order then to type it into the configuration file.

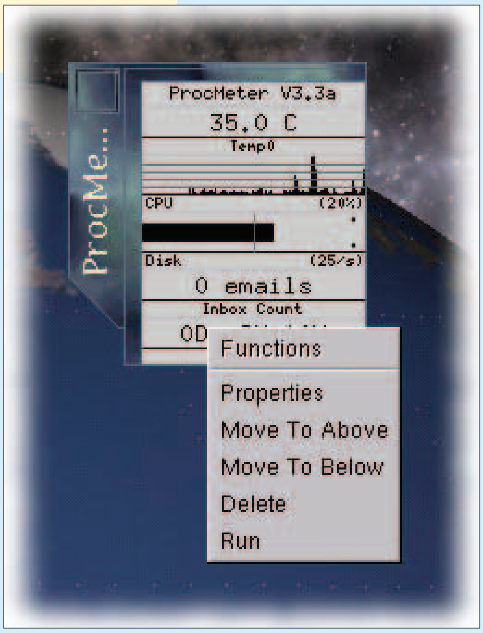
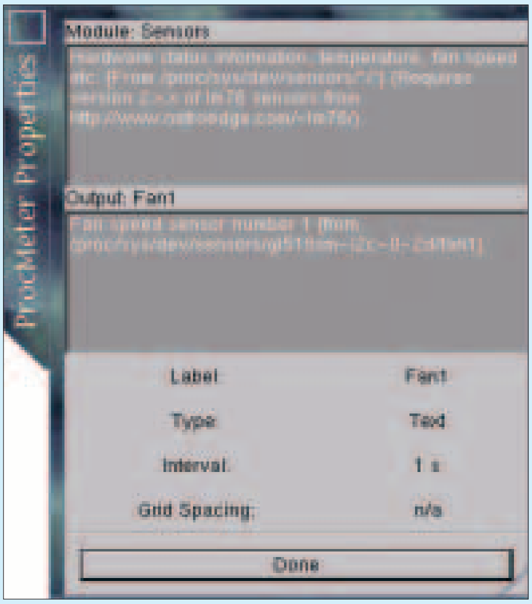


Fig. 2: Hidden behind the left mouse button



Module: Sensors

Hardware status information, temperature, fan speed etc. (From /proc/sys/dev/sensors/*/*) (Requires version 2.x.x of lm78 sensors from http://www.netroedge.com/~lm78/)

Output: Fan1

Fan speed sensor number 1 (from /proc/sys/dev/sensors/gl518sm-i2c-0-2d/fan1)

Label Fan1

Type Text

Interval 1 s

Grid Spacing n/a

Fig. 3: Properties explained

Xresources Ingenious system, with which one or more X application(s) can be configured at the same time. Mostly, one merely defines colour, font or geometry details in this. Newer toolkits (such as GTK+ or Qt) are based on so-called themes and no longer take any notice of the Xresources, as past Desktopias have discussed.

From the Properties window in Figure 3, for example, you can see that the corresponding module is called Sensors and under Output is being asked for a statement on Fan1 and thus on the rpm of the first fan. The display mode is defined in the configuration with a single letter (*t* for Text, *g* for Graphics and *b* for Bar display). Thus a .procmeterrc corresponding to the figure would contain the line *Sensors.Fan1-t*, with which a text output of the fan rpm is desired. Module and Output are here connected by a dot, while the type of display is attached with a hyphen.

You can only access the sensors module if you have installed a current Lm_sensors package or using a kernel from version 2.4.13 on with I2C support.

Out of the box, on the other hand, you get a complete overview of your CPU-utilisation displayed graphically. Enter *Stat-CPU.CPU-g* in the required place in the *[startup]* section. *Stat-CPU* here refers to the module to be used, with *CPU* you are saying that you would like to consult this module about the CPU in general (each module has several items of information ready), and with the final *g* you are setting the output to "graphics".

If one line of your configuration file becomes too long, you can indicate with the symbol\ *that it continues on the next with associated entries:*

```
[startup]
order = Stat-CPU.CPU-g\
        Processes.Processes-t\
        Memory.Mem_Free-b
```

Personal note

In the following *[resources]* section you can define general options, such as colours and fonts, for all the elements. The only things which have to be included are those which you want to be different from the standard settings. The following lines are all it takes to adjust foreground and background colours:

```
[resources]
foreground = white
background = #445566
```

The colours can, as will be familiar from HTML-pages, be defined both as hexadecimal RGB-values or else with names. You can get a good overview with the tool *xcolorsel*. If on the other hand you are looking for a nicer font, then *xfontsel* will be glad to help.

Authorities

ProcMeter also obeys classic XWindow options and so the system monitor can be influenced on start-up via suffixes: *–geometry –0+0* moves the window to the upper right corner. If you don't want to type in this flag every time, then inputs in the classic **Xresources** will help you out. To do this, write the following line in the file *~/.Xdefaults*:

```
procmeter3*geometry: -0+0
```

Should this file not yet exist, simply make it from new. By the time the command *xrdb –merge ~/.Xdefaults has been given, your entry will show effect. At least the GTK-free variant procmeter3* can be filled with a background graphic in the *.Xdefaults* instead of with plain colours:

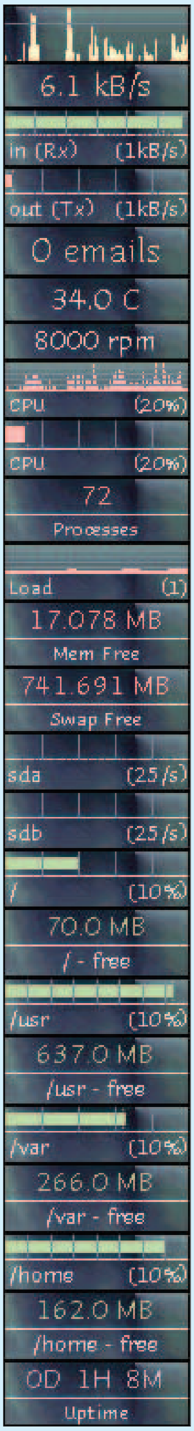
```
procmeter3*pane*backgroundPixmap: graphic.xpm
```

But that's not all: Maybe you would rather have the information on the file system displayed in a separate window? No problem, knit yourself a second configuration file under a different name (such as *.procmeterrc-2*), and make your Xresources entries under a different designator again (*procmeter3-2*geometry: +0+0*, for example). The latter prevents the two windows sticking over each other in a corner. Now call up ProcMeter with the command:

```
procmeter3 --rc=.procmeterrc-
2 -name procmeter3-2
```

and the second settings come into effect.

As you can see, ProcMeter can be configured and made highly individual. Once one has grasped the possibilities, you must also put the brakes on straight away. All too quickly, you'll find yourself configuring this superb system monitor so that it fills up a major portion of the screen, thus creating a new hurdle in the path of an easily understood status report.



```
6.1 kB/s
in (Rx)    (1kB/s)
out (Tx)   (1kB/s)
0 emails
34.0 C
8000 rpm
CPU        (20%)
CPU        (20%)
72
Processes
Load       (1)
17.078 MB
Mem Free
741.691 MB
Swap Free
sda        (25/s)
sdb        (25/s)
/          (10%)
70.0 MB
/ - free
/usr       (10%)
637.0 MB
/usr - free
/var       (10%)
266.0 MB
/var - free
/home      (10%)
162.0 MB
/home - free
0D 1H 8M
Uptime
```

Fig. 5: ProcMeter with background graphic