Binario Etico – Soc.Cooperativa

Informatica Sostenibile

Virtual Distro Dispatcher

NLnet project

Mapping system level to desktop level performance

Index

1. Introduction

Server workloads are relatively predictable and GNU/Linux systems support such workloads well. Desktop workloads, instead, are more variable and harder to get right [1]. Till some years ago, GNU/Linux systems were mainly distributed as workstations or servers rather than as desktops. Behaviour and performance suitable for a developer, usually are not what a knowledge worker needs [2]. Lately things changed and GNU/Linux desktops are increasingly spread, hence Desktop Environments had to address new performance profiles. Unlike server workloads, the primary requirement of interactive applications is to respond to user events under human perception bounds rather than to maximize end-to-end throughput [3].

Benchmarks are used to identify performance problems and motivate improvements in system design. Current benchmarks typically report throughput, bandwidth, or end-to-end latency metrics . These benchmarks do not give a direct indication of performance that is relevant for interactive applications such as those that dominate modern desktop computing. The most important performance criterion for interactive applications is responsiveness, which determines the performance perceived by the user [5].

1.1 Towards a definition of desktop performance benchmarks

Parameters to be used for measuring performance and become a valid metric in a particular context, should generally be:

• critical to the success of the system in serving its purpose;

• persistent across successive system releases and in the face of technological change, and therefore valid as a basis for measuring performance improvements;

• manipulable by designers, in order to achieve specific performance targets [4].

End-to-end latency, throughput and bandwidth metrics measure system performance for repetitive, synchronous sequences of requests. Feeding user input as rapidly as the system can accept it is equivalent to modeling an infinitely fast user. In any case, the results of these benchmarks do not correlate directly with user-perceived performance. The performance of desktop applications depends on the speed at which the system can respond to an asynchronous stream of independent and diverse events that result from interactive user input or network packet arrival [5].

In order to measure what in [5] is called **event handling latency**, we defined a set of benchmarks, organized into the three categories in [5]:

1. **Microbenchmarks**, useful for understanding system behavior for simple interactive operations, such as interactions with Desktop Environment and GUIs.
2. **Task-oriented benchmarks** useful to understand the real impact of latency on the perceived interactive responsiveness of an application. Good examples are interactive computing workload from office suites.
3. **Application microbenchmarks** useful to evaluate isolated interactive events from the computations needed for particular operations within an application (e.g. apply a filter in a raster graphics editor or execute a function in a spreadsheet).

By analyzing such benchmarks, we develop an understanding of the low-level behavior of the system related to them.

1.2 Coupling system monitoring with desktop benchmarking

In order to analyze performance of VDD, we have at first collected quite an extensive amount of data from tests and benchmarks. They were related to system aspects, such as CPU usage, memory speed, network bandwidth, disk responsiveness, and so on. In [6] and [7] we summarized and commented the most relevant. Unfortunately, those tests do not say much about user experience. In an interactive system, the CPU is mostly idle. When an interactive event arrives, the CPU becomes busy and then returns to the idle state when the eventhandling is complete. A first idea is to record when the processor leaves and returns to an idle state, so that we can measure the time it takes to handle an interactive event, and the time during which a user might be waiting [5]. But we need to go further and generalize the measure of waiting times when concurrent interactive events occur. Hence we thought of observing system parameters while desktop operations, such as menu pop-up or application launching, take place, in order to describe correlations and hence trace a mapping.

What we needed was a sort of “electrocardiogram” measuring values of a set of system parameters and providing them real-time with a certain period (e.g. 1 second). We then defined a set of operations belonging to the aforementioned three categories. Tests consisted in executing operations, taking notes of timestamp, duration and incidental slowing downs or anomalies. In the meanwhile, values of the “electrocardiogram” are monitored and saved in files for subsequent analysis.

2 Test procedure

2.1 Dstat: the system “electrocardiogram”

There are several tools to monitor system state. Some of them are comprehensive suites (such as Phoronix Test Suite, xfbsuite) that include an incredible variety of utilities and functions, mostly not needed for our purpose. Others are interactive (top) or too textual (vmstat, iostat) to have outputs saved and analyzed automatically. Vmstat or iostat can not be used to produce graphs (e.g. via gnuplot) without preparing custom scripts. Moreover, they do not include a timestamp in the output. Finally we decided to use a good replacement of theirs, **dstat**, a simple and versatile tool for generating system resource statistics. Dstat makes exactly what we needed, i.e. monitoring system state instant after instant. It generates tables with configurable rows and columns. For example, one can choose to monitor quantity of memory, free, buffered and cached (columns) every 2 seconds (rows). It includes several interesting options and generates precise results in a standard form so that they can be eventually put on a graph. It works very well in combination with graph generators, such as gnuplot, but we had to write a patch (see appendix 1) in order to exclude the preface of the output that otherwise interfere and prevent graph creation.

In the following, we describe the way we used dstat providing examples. The command:

dstat -t -cC 0,1,total -l -ms -n -dD sda1,sda2,total --noheaders --output example.csv

generates this output in the terminal.

```
----time---- -------cpu0-usage-------  -------cpu1-usage-------  -------total-cpu-usage---- ---load-avg---> -net/total- --dsk/sda1---dsk/sda2--dsk/total-
  date/time  |usr sys idl wai hiq siq|usr sys idl wai hiq siq|usr sys idl wai hiq siq| 1m   5m  15m > recv  send|read  writ:read  writ:read  writ
18-04 21:20:15|  5   1  92   2   0   0:  5   1  93   0   0   0:  5   1  93   1   0   0|  0    0    0>   0     0 |7804B   13k:   0     0 :  25k   54k
18-04 21:20:16|  0   0 100   0   0   0:  2   1  97   0   0   0:  1   0  99   0   0   0|  0    0    0>   0     0 |   0     0 :   0     0 :   0     0
18-04 21:20:17|  0   0 100   0   0   0:  1   0  99   0   0   0:  0   0  99   0   0   0|  0    0    0>   0    56B|   0   264k:   0     0 :  48k  528k
18-04 21:20:18|  0   0 100   0   0   0:  0   1  99   0   0   0:  1   0 100   0   0   0|  0    0    0>   0    56B|   0     0 :   0     0 :   0     0
18-04 21:20:19|  1   0  92   7   0   0:  2   1  97   0   0   0:  1   0  95   3   0   0|  0    0    0>   0     0 |   0     0 :   0     0 :   0     0
18-04 21:20:20|  0   0 100   0   0   0:  1   0  99   0   0   0:  0   1  99   0   0   0|  0    0    0>   0   100B|   0     0 :   0     0 :   0    56k
18-04 21:20:21|  0   0  93   7   0   0:  1   1  98   0   0   0:  0   0  96   3   0   0|  0    0    0>   0     0 |   0     0 :   0     0 :   0     0
```

The first column of the generated table is time (-t option), given in the form hh:mm:ss. Dstat provides up to one shot per second. Second group of columns regards CPU. Six parameters for every CPU (or CPU core) are available: usr, sys, idl, wai, hiq, siq. In the example, in a system with two CPUs, cpu0, cpu1 and the total (-cC 0,1,total) are shown. The next option (-l) is the classic load avarage in three values: 1m, 5m and 15m (one minute, five minute and fifteen minutes). The third group of columns (option -ms) regards memory (used, buffered, cached, free) plus swap (used, free). The forth group of columns regards the network, a simple byte received, sent scheme (-n). The fifth group of column is about disk usage, separted into sda1, sda2 and total (-dD sda1,sda2,total). For each one, bytes read and written are monitored. Of course, it is possible to add sda3, sda4, etc. and it can also be done on a device base instead (sda, sdb, etc.). The option --noheader skips some initial output that could interfere with a subsequent process of analysis (not all, though, see appendix 1). The last

option (--output) is a great one, as provide a CSV formatted file as output (in addition to the terminal output shown above, that can be redirected to another file too, of course). With a CSV file, it is then possibile to use a spreadsheet, such as OpenOffice Calc, thus making the analysis and the graph generation straighforward. If one it is not interesed in monitoring each CPU plus the total, and each disk/partition plus the total, command could simplify this way:

dstat -t -c -l -ms -n -d --noheaders --output example.csv

As we will see, monitoring more CPUs (or CPU cores) can be very useful to see how workloads is distributed on them. Unfortunately, due to how desktop applications are currently written (not suitable for a parallel workload distribution), the only CPU core to be stressed during a standard desktop use is the first. But a big advantage of using virtual machines is the possibility of associating each of them to a different core. This way VDD can really increment efficiency even in a desktop situation.

2.2 Analysis process

VDD host and every every virtual machine in use (corresponding to one or more VDD desktops dispatched and in use) are monitored with dstat during tests. For every desktop under test, two dstat tables have to be analyzed: the one of the corresponding virtual machine and the one of the host. These are two CSV files to be imported in OpenOffice Calc for inspection and analysis and to generate graphs if and where needed. In order to simplify this process, we developed a macro (displayed in Appendix 2). Suppose, for example, that an anomaly or slow down is reported at a particular timestamp (hh:mm:ss). One can see what is the sample number at that timestamp (an unsigned int corresponding to the spreadsheet row number). The funciotn sys_graphs can then be called, specifying three parameters:

1) the number of the raw of the sample of interest;
2) how many samples before that one to consider;
3) how many samples after that one to consider.

The table containing the data of the host must be put on a second spreadsheet tab (the table of the virtual machine is on the first tab by default when the CSV file is imported). The macro generates 6 graphs per tab (so 12 in total) appropriately formatted, corresponding to system perfomance measures (CPU usage, load avarage, memory usage, swap usage, net load, disk load). Despite the use of --headers option and of our dstat patch (appendix 1), it is worthwhile to double check that the second raw of the spreadsheet is the one where the heading of columns is written (the first one must be empty and can be used for launching the function of the macro). Using this procedure, one can produce graphs to be inspected in a few seconds or regenerated with a larger range of samples if needed. For our experience, one or two

hundreads of samples (say 50-100 on the left and 50-100 on the right) are usually enough and the resulting graphs are still quite readable (too many samples make the x axis too crowded).

2.3 Operations

We chose the following operations to be performed on VDD desktop during tests.

2.3.1 Microbenchmark

- VDD Desktop start up: select a Desktop to be dispatched on your terminal
- Responsiveness of DE menus: browse menus using a fixed sequence
- Start up user programs: OpenOffice Write, OpenOffice Calc, OpenOffice Impress, Mozilla Firefox, Gimp, Mplayer, Thunderbird, acroread
- Copy and move files of fixed dimension: 100 KB, 10MB, 100 MB
- Switching between 2 VDD desktops
- Switching VDD desktops with full load (switch forward and backward all desktops)

2.3.2 Task-oriented benchmark

- **Multimedia multitasking test**: convert a file containg a video with a defined quality (encoding, fps, resolutions and megabyte) from mov to mpeg. The video conversion takes place in the foreground (in order to test how long it takes a system to perform only the conversion). This test exercises almost every major subsystem, including CPU, memory, and hard drive. Desktops with multicore CPUs are likely to perform better than comparable systems that use CPUs with fewer cores or single-core CPUs;
- **GIMP image-processing test**: apply a filter to an image of defined quality (camera RAW image files captured from a X-megapixel camera). Examples of filters could be Unsharp Mask, Lens Correction, and Dust and Scratches filters; as well as reducing image noise and converting the images to grayscale JPEGs. This test primarily exercises a system's CPU, memory, and chipset subsystem, but it also utilizes the graphics and hard drive subsystems to some extent. Some of the filters we use in the GIMP test can use multithreading, so desktops with multicore CPUs are likely to perform better than comparable systems that use CPUs with fewer cores or single-core CPUs;
- **Encoding test**: convert tracks of a music album (e.g. 10 tracks) to MP3 files. This test almost exclusively exercises a system's CPU capabilities. It supports multithreading, so

desktops with multicore CPUs are likely to perform better than comparable systems that use CPUs with fewer cores or single-core CPUs;

- **OpenOffice productivity test**: execute defined operations (using macros can be a good option). Operations under OpenOffice Write could be searching and replacing, changing font sizes, and creating columns. About OpenOffice Calc, operations could be performing functions on a spreadsheet, such as solving formulas and creating charts. In OpenOffice Impress, we could add graphics and text and moves images around in a presentation. This test exercises nearly every major subsystem, including CPU, memory and hard drive;
- **Cinelerra test**: perform rendering. This test almost exclusively exercises a system's CPU capabilities. Cinelerra supports multithreading, so desktops with multicore CPUs are likely to perform better than comparable systems that use CPUs with fewer cores or single-core CPUs (this set of tests is inspired by [8]).

2.3.3 Application microbenchmark

OpenOffice Write	Search and replace (e.g. "a" arial at 12 → "A" bold verdana at 14)
OpenOffice Calc	Sum up a certain amount of numbers in a column
OpenOffice Impress	Open an animated presentation
Firefox	Open page with fixed content and dimension (either local file system or apache web server could be used)
Gimp	Open an image with defined quality
Mplayer	Play a song with defined quality
Thunderbird	Import address book
Acroread	Search a word in a document

3 Testbed

3.1 Hardware setup

VDD has been tested in its development network, appropriately adapted and prepared for testing, consisting of 2 servers and 10 thin clients, with the following characteristics.

Servers (192.168.108.21, 192.168.108.22)

- CPU: Intel core i7 920 @ 2.67GHz 8MB SK1366 RET
- RAM: 8GB (4 DIMM KINGSTON 2GB PC1333 DDR III CL9)
- 3 Western Digital hard disks, RAPTOR 74GB 10000rpm 16MB SATA
- BOX COOLER MASTER ELITE 330 BLACK
- Fan Zalman 7500
- VD CAPTIVA GEFORCE 9400GT 1GB PCX
- TECHSOLO 730W
- Supplementary gigabit network card

Terminal 0 (192.168.108.10)

CPU	Intel(R) Pentium(R) 4 CPU 1.50GHz
cache size	256 KB
RAM	377MB

Terminal 1 (192.168.108.11)

CPU	Intel(R) Pentium(R) 4 CPU 1.50GHz
cache size	256 KB
RAM	377MB

Terminal 2 (192.168.108.12)

CPU	Intel(R) Pentium(R) 4 CPU 1.70GHz
cache size	256 KB

RAM 377MB

Terminal 3 (192.168.108.13)

CPU Intel(R) Pentium(R) 4 CPU 1.70GHz

cache size 256 KB

RAM 377MB

Terminal 4 (192.168.108.14)

CPU Intel(R) Pentium(R) 4 CPU 1.50GHz

cache size 256 KB

RAM 377MB

Terminal 5 (192.168.108.15)

CPU Intel(R) Pentium(R) 4 CPU 1.70GHz

cache size 256 KB

RAM 377MB

Terminal 6 (192.168.108.16)

CPU Intel(R) Pentium(R) 4 CPU 1.70GHz

cache size 256 KB

RAM 377MB

Postazione 7 (192.168.108.17)

CPU Intel(R) Pentium(R) 4 CPU 1.70GHz

cache size 256 KB

RAM 377MB

Postazione 8 (192.168.108.18)

CPU Intel(R) Pentium(R) 4 CPU 1.70GHz

cache size 256 KB

RAM 377MB

Postazione 9 (192.168.108.19)

CPU Intel(R) Pentium(R) 4 CPU 1.50GHz

cache size 256 KB

RAM 377MB

Hardware homogeneity has been double checked: RAM (quantity and frequency), CPU (frequency and type). Mouse responsiveness has been tested carefully, to avoid slowing down users during tests. Network has also been tested to exclude leakages. Tests using the Internet have been done when no other activities were running. During tests, we monitored VDD server too. So both guested machines and hosting machine have been taken into account in the analysis (see chapter 4).

3.2 Test batteries

VDD currently supports eight operating systems. Six of theese have got three Desktop Environments (Gnome, KDE and XFCE), so eighteen desktops in total. Plus the two Microsoft Windows desktops, it makes 20 different dispatchable desktops (see table).

	Gnome	KDE	XFCE	Windows
Debian Lenny	V	V	V	
Ubuntu Jaunty	V	V	V	
Ubuntu Karmic	V	V	V	
Centos 5	V	V	V	
Fedora 11	V	V	V	
Gentoo 10	V	V	V	
Windows XP				V
Windows 7				V

Tests have been executed on each of them. Performance analysis refers to the use of VDD from terminals, i.e. a real-life use. While tests run, system is costantly monitored and values of performance parameters defined in Section 2.1 are saved into files for future inspection. We defined four different batteries of tests.

- **Battery 1**: one user for one desktop at a time. All operations in Section 2.3 are performed twice (to avoid caching distortions). This battery verifies if peformance of a VDD desktop differ from performance of a standard local and physical desktop. Another result of the battery is the comparison of performance among desktops.
- **Battery 2**: ten different users on the same desktop from ten terminals. Repeat same tests, in a random order for each user, and without taking care of caching distortions (irrelevant in a random situation).
- **Battery 3**: ten different users on ten different desktops from ten terminals. Repeat same tests, in a random order for each user, and without taking care of caching distortions (irrelevant in a random situation). Battery 2 and 3 must not be done the same day (in order to keep the experience as random as possible).
- **Battery 4:** one user on one desktop. Repeat tests with artificial stressing of VDD. The traffic for stressing must occur from terminals: nine terminals using the same desktop or nine terminals using different desktops. Three stressing profiles are defined (low, intermediate, high), to be calibrated during tests, based on levels of I/O, memory access, CPU usage and network traffic. Tools for stressing systems: CPUBurn (http://pages.sbcglobal.net/redelm/), Mersenne Prime (GIMPS) (http://mersenne.org/freesoft/).

3.3 Software setup

The 20 desktops have been provided with all the software needed for tests. The following applications have been installed on Linux and Windows systems: OpenOffice, Firefox, Gimp, Mplayer, Thunderbird, Acroread. For converting a music album on Linux, *mp2enc* has been installed and enclosed with a supplementary script (useful to repeat execution for all CD tracks). Command line is very useful when time must be measured (every command can be preceded by the instruction *time*). Under Windows this is not possible, so we used Sound Converter, that provides a summary of time elapsed at the end of operations. To convert MOV video to MPEG, Avidemux has been installed both on Linux and Windows. On all users home in all desktops, test files have been put (or linked), see Section 3.4 for details.

The following is the association desktop-terminal used for battery 3.

IP	Machine	DE	Terminal

192.168.108.31	jauntyvm	XFCE	0
192.168.108.34	karmicvm	Gnome	1
192.168.108.34	karmicvm	KDE	2
192.168.108.43	fedora12vm	KDE	3
192.168.108.32	centos5vm	Gnome	4
192.168.108.43	fedora12vm	Gnome	5
192.168.108.30	lennyvm	Gnome	6
192.168.108.42	gentoo10vm	KDE	7
192.168.108.40	windows 7	Windows	8
192.168.108.41	windows xp	Windows	9

The following is the memory configuration of virtual machines in VDD during battery 3:

Name	**Memory**	**VCPUs**
Domain-0	5463	8
centos5vm	512	1
fedora12vm	512	1
gentoo10vm	512	1
jauntyvm	512	1
karmicvm	512	1
lennyvm	512	1
win7vm	1024	1
winxpvm	512	1
winxpvm2	512	1

Finally, it has been essential to syncronize time on all host and guests machines, in order to have exactly the same time during tests, so that testers' time references when reporting events are consistent.

3.4 People setup

On all users home directories, in all desktops, test files have been put (or linked). This is the list of files, including file specs:

FILE	SPECS		
davide.jpg	JPEG image data, JFIF standard 1.02	8,4MB	2592 x 3872 pixel
file_10M	10MB file		
file_100k	100KB file		
file_100M	100MB file		
oxford.pdf	PDF document, version 1.4		
thesis.odt	OpenDocument Text	93 pages	
vdd_test_protocol.odt	The tests explained to testers		
kubuntu.odp	OpenDocument Presentation	10 slides	
payment.ods	OpenDocument Spreadsheet	4000 cells	
Track_1.wav	WAVE audio, PCM, 16 bit, stereo 44100 Hz		
Track_2.wav	WAVE audio, PCM, 16 bit, stereo 44100 Hz		
Track_3.wav	WAVE audio, PCM, 16 bit, stereo 44100 Hz		
Track_4.wav	WAVE audio, PCM, 16 bit, stereo 44100 Hz		
Track_5.wav	WAVE audio, PCM, 16 bit, stereo 44100 Hz		
Track_6.wav	WAVE audio, PCM, 16 bit, stereo 44100 Hz		
Track_7.wav	WAVE audio, PCM, 16 bit, stereo 44100 Hz		
Track_8.wav	WAVE audio, PCM, 16 bit, stereo 44100 Hz		
wavlist2mp3	script for repeating mp2enc for all tracks and put them in a separate dir		
SoftonicDownloader90278.exe	SoundConverter installation file for Windows		
trashware.mov	ISO Media, Apple QuickTime movie		
	Duration:	7 minutes 39 seconds	
	Dimension:	640 x 480	
	Codec:	H.264 / AVC	
	Frame rate:	30 frames per second	
	Audio codec:	Raw 16-bit PCM audio	

Channel: Stereo

Sampling rate: 16000 Hz

All people were given two questionnaires during tests, one for performance, one for usability rating (see Report on usability for the latter). The performance questionnaire asked for name, system user, operating system, Desktop Environment, battery number. For every operation, users had to write id number of the operation, timestamp (hh:mm:ss of the event), performance evaluation (perfect, reasonable, not acceptable), time to complete (if applicable) and provide notes and comments when needed. About completion time, it has been tracked only when relevant: for operations that are launchable from command-line (GNU/Linux only), this is done by preceding the instruction *time* in the command syntax; otherwise, by using application summary if available (e.g. Sound Converter under Windows). In some cases, measuring conventions needed to be used. For example, for Desktop startup we consider the operation executed when hourglass (or equivalent) terminates. Another example is Firefox, which do not stop time after been loaded, so it must be stopped by hand.

All the operations are numbered. For executing them in a random order, random lists of numbers were generated and given to users as well. The association between numbers and operations (given in the following) are essential to execute (and read) the analysis (they refer to operations IDs (#) in Chapter 4 tables). Users were given written instructions and some initial training. For example, following are the operations for group tests in battery 2 and 3.

On host (192.168.108.22):

dstat -t -c -l -ms -n -d --noheaders --output host_individual.csv

For each GNU/Linux machine (Debian Lenny, Ubuntu Jaunty, Ubuntu Karmic, CentOS 5, Fedora 12, Gentoo 2010):

On guest (192.168.108.x):

dstat -t -c -l -ms -n -d --noheaders --output guest_machine_individual.csv

do (fill in the form after every operation, including time stamp and evaluation):

0) Desktop start up (one at a time)

00) Responsiveness of menus (one at a time, then all together)

Go to: application

→ accessories

→ internet

→ office

resources

system

→ preferences

click on desktop

on Windows:

Go to: start

→ accessories

→ control panel

→ programs

click on desktop

000) Start up (and then close): (one at a time)

a) Write

b) Calc

c) Impress

d) Firefox

e) Gimp

f) Mplayer

g) Acroread

. Start up, wait, open doc, wait, execute action and then close:

Note: files in home/user0/Desktop/testfiles

1) write open thesis.odt, search and replace "a" -> "A"

2) calc open payment.ods, sum all column B values

3) impress open kubuntu.odp, slide show (10 slides)

4) Firefox go to: http://www.gnu.org/

5) Gimp open davide.jpg

6) Mplayer listen Track 1

7) acroread open oxford.pdf, search "qgis" in document

.Data cp, mv (100k, 10 M, 100 M) (take note of time)

8) time mv /home/user0/Desktop/test_files/file_100k /home/user0/

time cp /home/user0/file_100k /home/user0/Desktop/

9) time mv /home/user0/Desktop/test_files/file_10M /home/user0/

time cp /home/user0/file_10M /home/user0/Desktop/

10) time mv /home/user0/Desktop/test_files/file_100M /home/user0/

time cp /home/user0/file_100M /home/user0/Desktop/

On Windows:

8) using drag and drop, move file_100k from test_file to Desktop

using drag and drop, copy file_100k from Desktop to test_file

9) using drag and drop, move file_10M from test_file to Desktop

using drag and drop, copy file_10M from Desktop to test_file

10) using drag and drop, move file_100M from test_file to Desktop

using drag and drop, copy file_100M from Desktop to test_file

11) Convert mov to mpeg

Open avidemux. Open trashware.mov, save, video, trahware.mpeg, close

12) GIMP open davide.jpg

filter → art → cubism

close without saving

13) Album (8 tracks) → mp3

wavlist2mp3

delete files into mp3encoded

End of test: Desktop log out

4 Analysis

Individual tests gave excellent results. It was not possible for users to distinguish between a virtual desktop and a physical one. Also differences between 32 and 64 bits are not perceivable for a standard desktop use. The only two batteries that gave some interesting results were battery 2 and 3. In this Section we report results of those, starting with battery 3 (called first battery in the following) that presented a few slow downs (rarely annoying) and then battery 2, where more critical issues arised. User questionnaires are reported in their integral form (see evaluation legend). Whenever a slow down or an anomaly are registered, graphs related to that timestamp are provided (using macro in Appendix 2) and briefly commented. About responsiveness of DE, events that complete in 0.1 seconds or less are believed to have imperceptible latency and do not contribute to user dissatisfaction, whereas events in the 2-4 second range invariably irritate users [9].

Evaluation legend:

0 = Exactly as it was a local desktop

-1 = A bit slow, but reasonable

-2 = Not acceptable

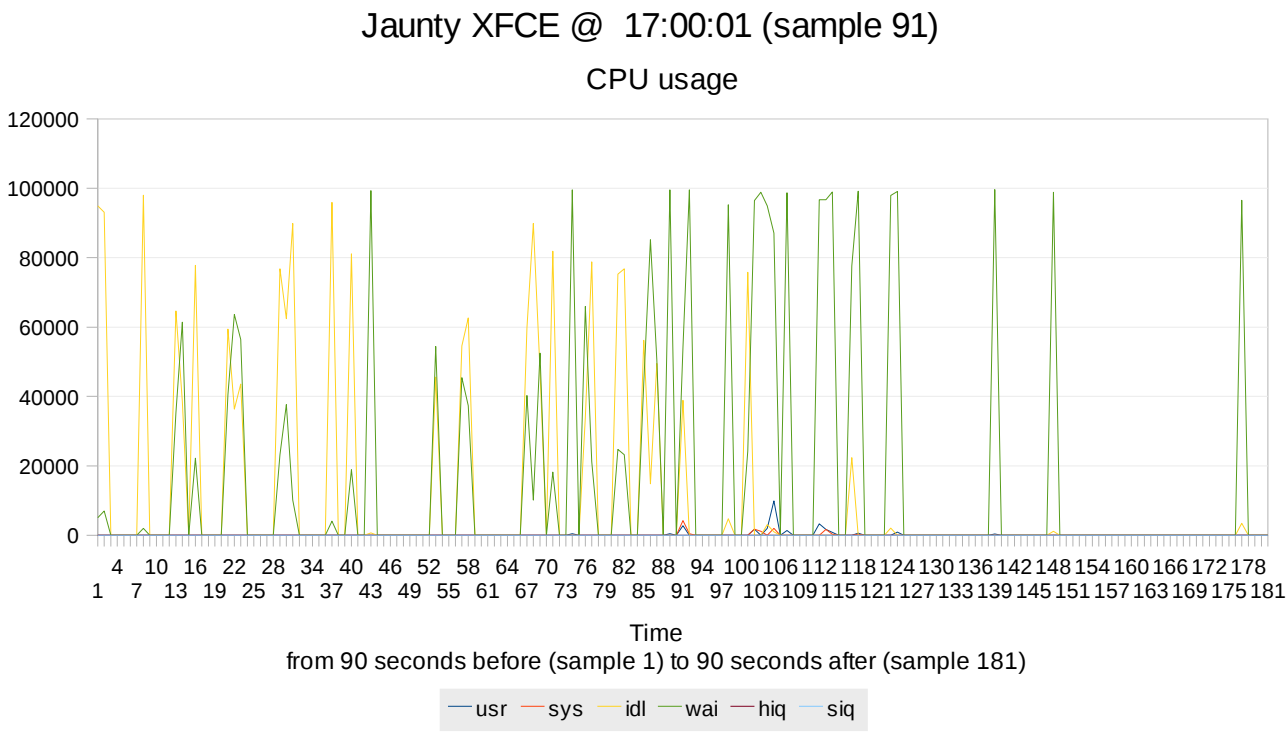
4.1 First battery

4.1.1 User0 Ubuntu Jaunty XFCE

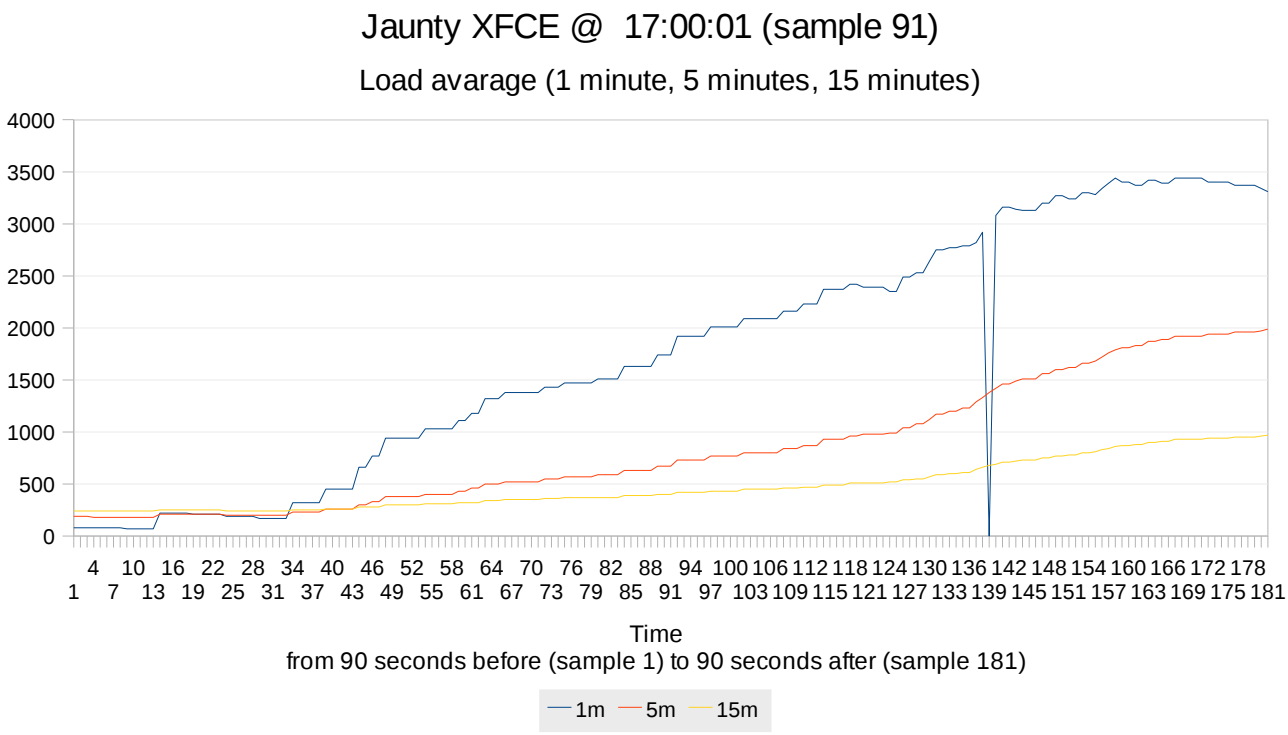
#	Timestamp	Evaluation	Time to complete	Notes
0	-	0	-	-
00	-	-1/0	-	-
a	-	0	29.701	-
b	-	0	1.85	-
c	-	0	1.12	-
d	-	0	13.48	-
e	-	-1	58,23	-
f	-	0	9,4	-
g	-	0	24,3	-
1	16:17:40	0		-
10	16:25:16	0	0.002 mv 3.967 cp	-
7	16:31:45	0		Slightly slow

9	16:34:50	0	0.001 mv 0.193 cp	-
8	16:40:02	0	0.001 mv 0.04 cp	-
13	16:41:15	0		-
2	16:47:18	0		-
4	16:51:10	0		-
6	16:56:35	0		Slightly slow
3	17:00:00	-2		Very slow!
12	17:08:15	0		-
11	17:14:25	0		-
5	17:16:30	-1		-

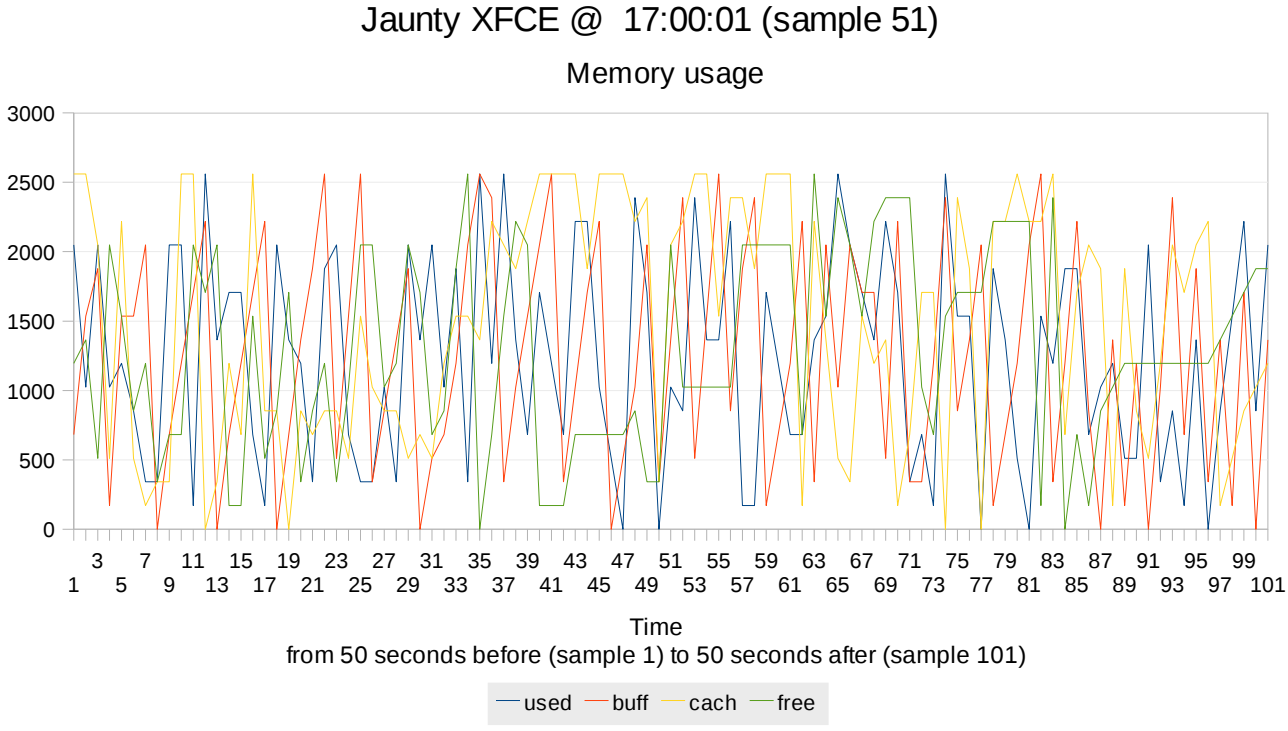
Let us see what happened at 17:00:00, from 50 seconds before and till 50 seconds after (=SYS_GRAPHS("Jaunty XFCE";5785;50;50)) and then from 90 seconds before and till 90 seconds after (=SYS_GRAPHS("Jaunty XFCE";5785;90;90)). We report all the 12 graphs, starting from the guest, and choosing for each graph the more convenient extent (between 50;50 and 90;90).



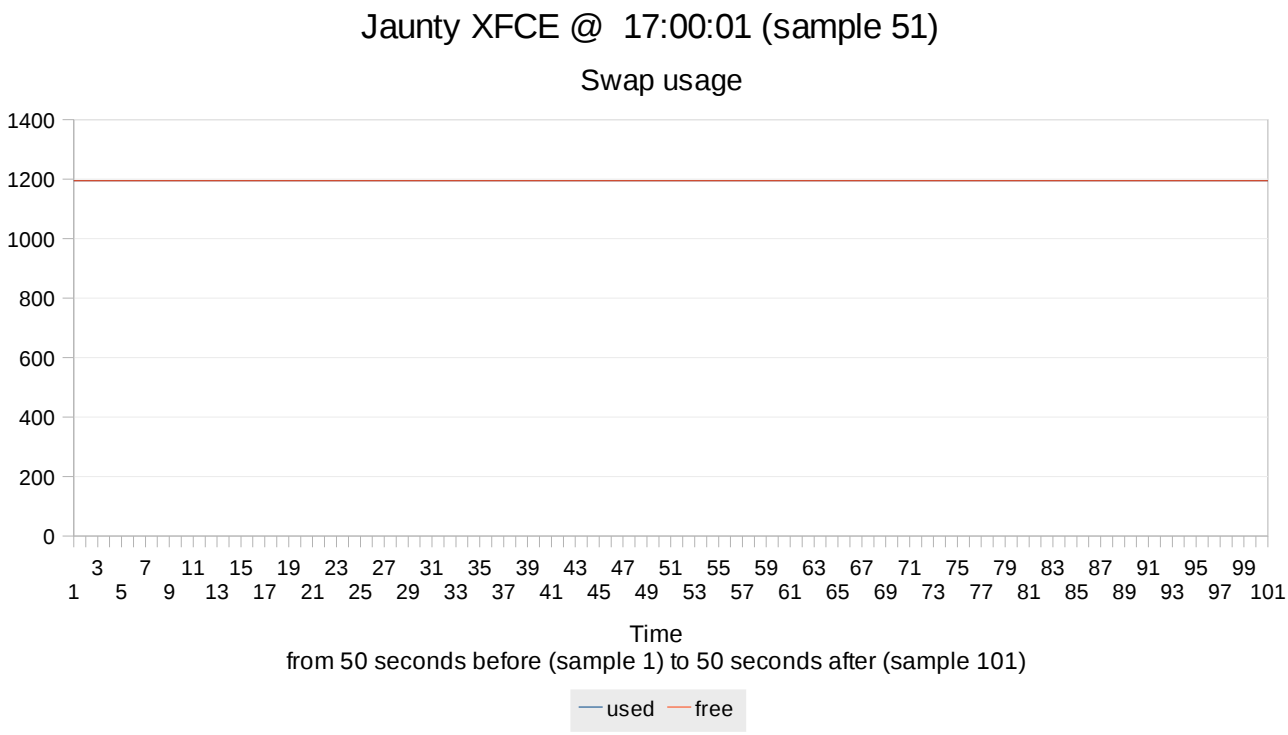
A set of spikes can be noticed around sample 91. The spikes of wai (that regards I/O, so network or disk) cease after one or two minutes.



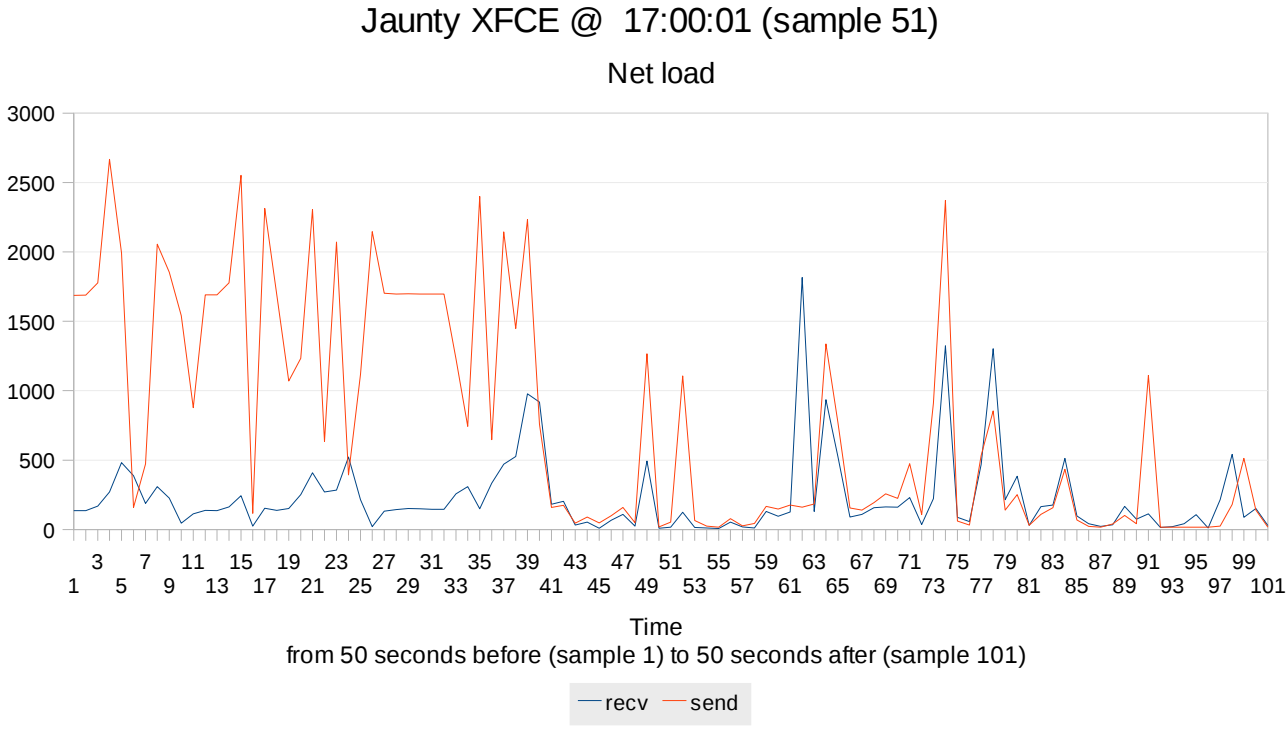
Load avarage is increasing and it is going to increase more after sample 91. It starts decreasing after two minutes.



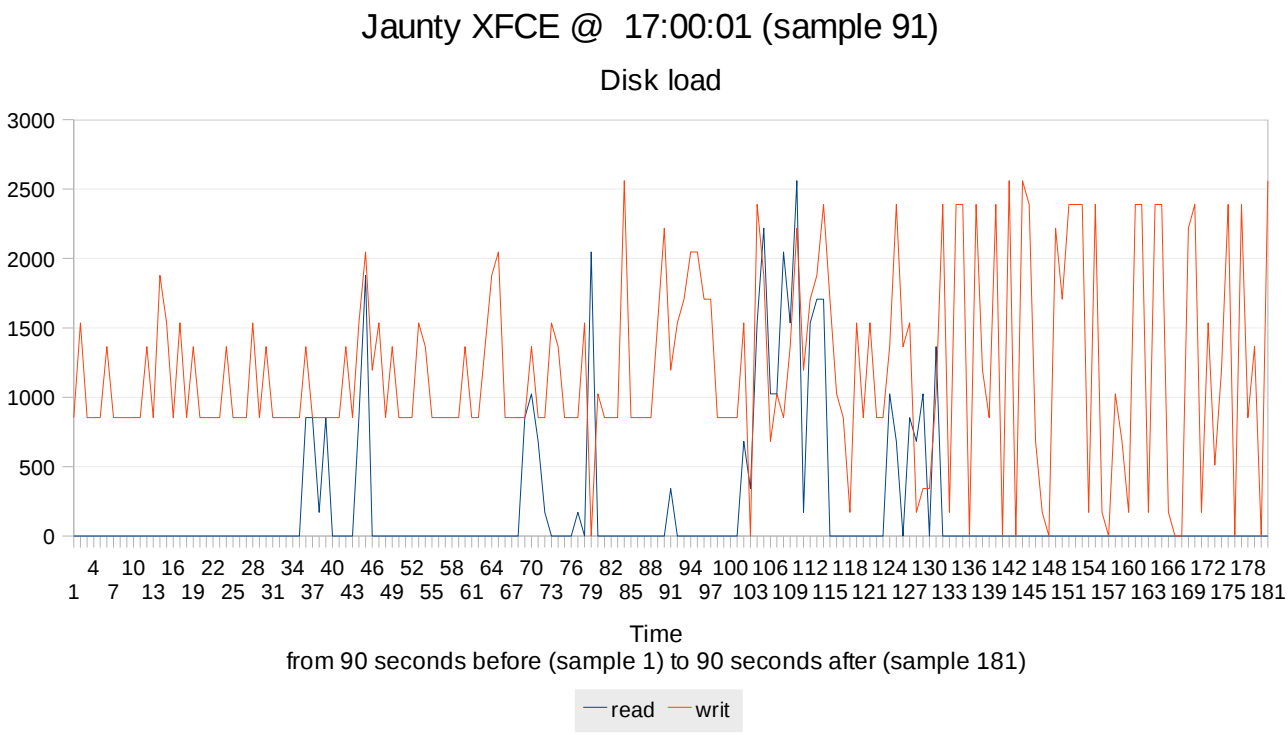
Memory seems ok around sample 51 and it starts to decrease anyway after a couple of minutes.



Swap is not giving problems.

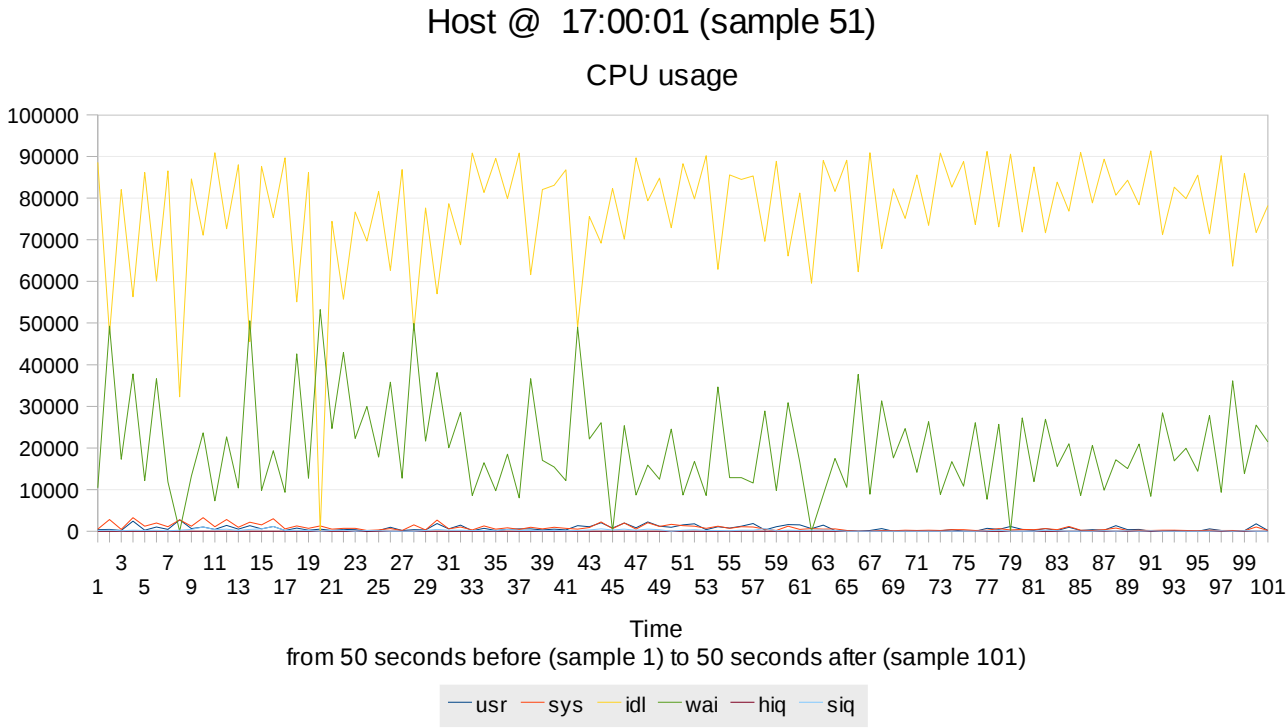


Small network activity registered. Minor but significant peaks aroung sample 51.

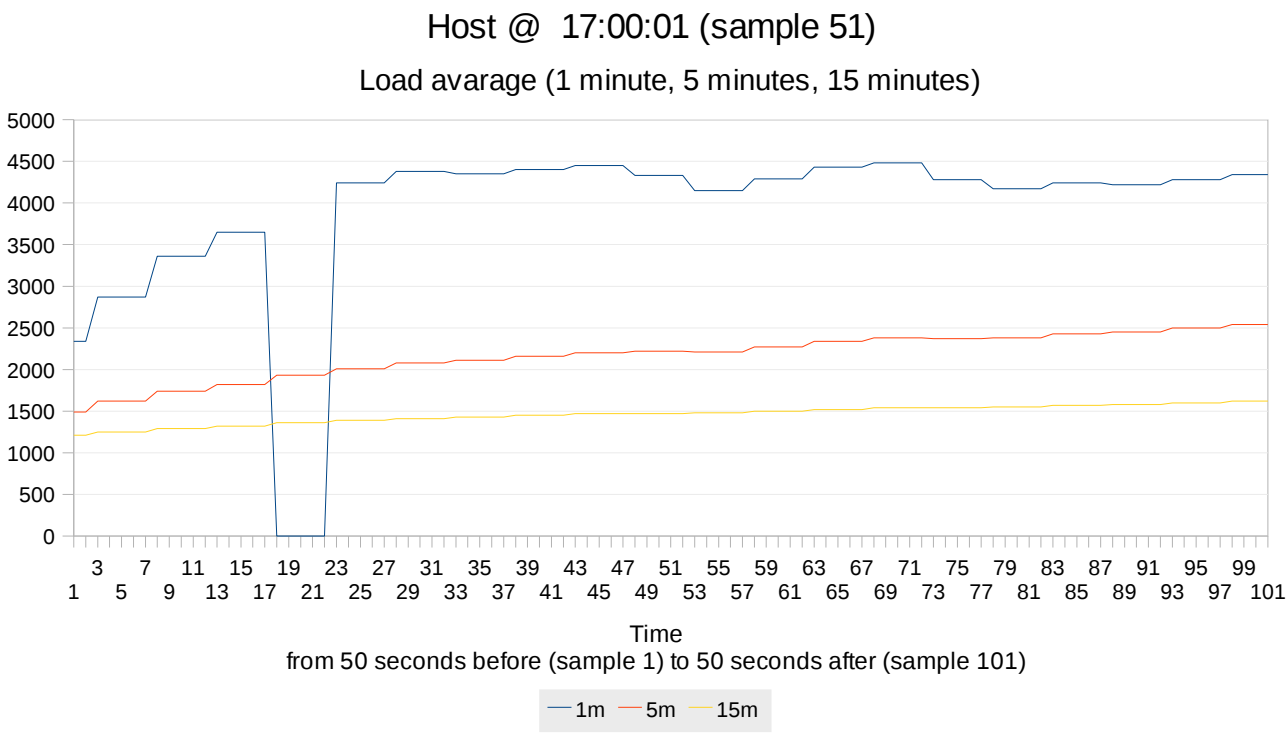


Some disk operations are going on. A writing burst is arriving in the next two minutes.

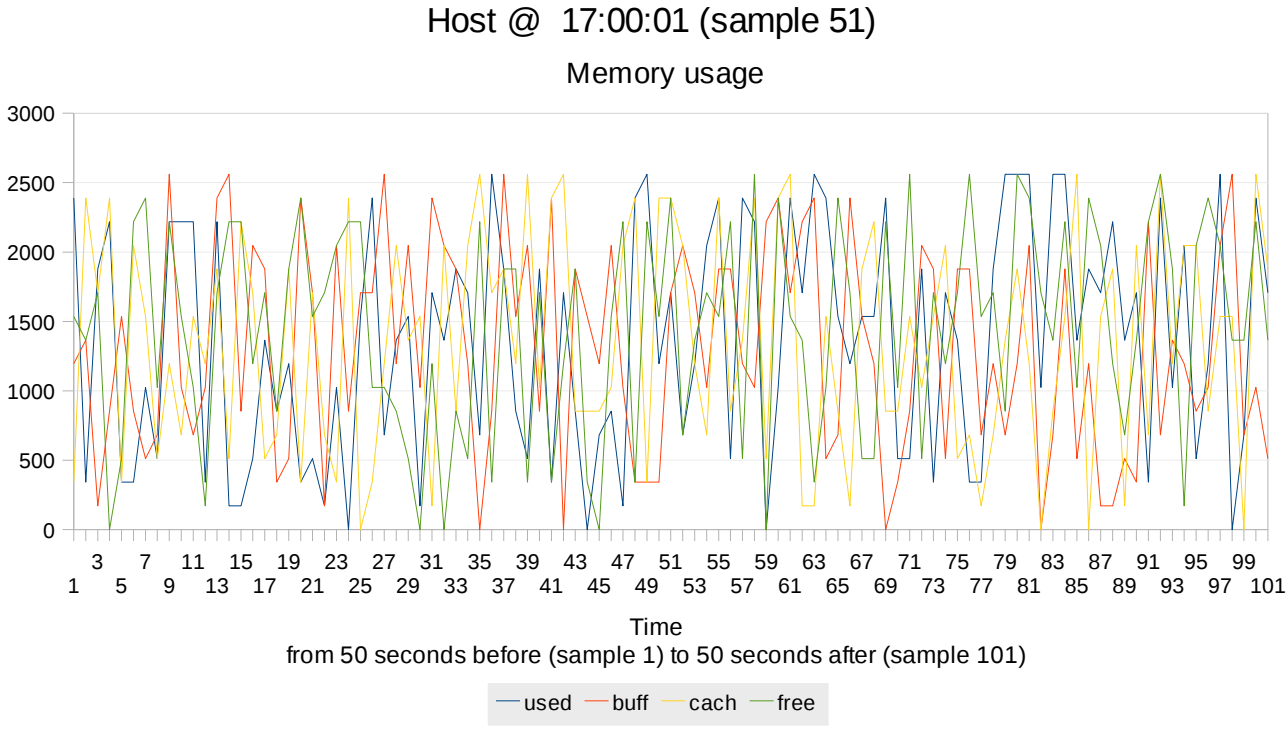
Let us go to the host now.



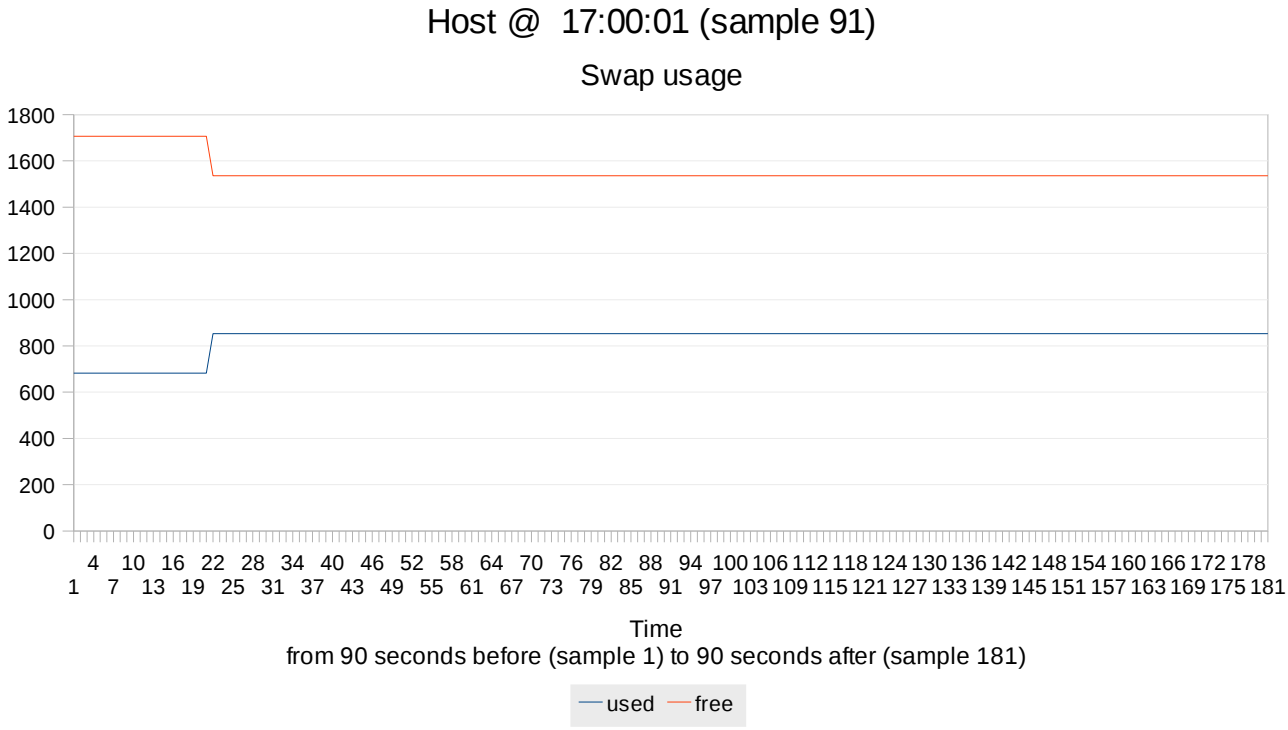
Host seems quite ok. Some, not worrying, wai peaks can be noticed.



Load avarage is a bit higher than normal and slightly increasing, but under control.



Memory is normally and costantly used.



Some swapping is going on. Even if a system does have an ample supply of fast RAM, it might decide to conserve it and use virtual disk memory, which is contained in the swap file on the hard drive. The hard drive runs 100 times slower than RAM, so that's one place system speed goes [2] . This is valid as a general comment (we will not repeat it in the rest of the report).

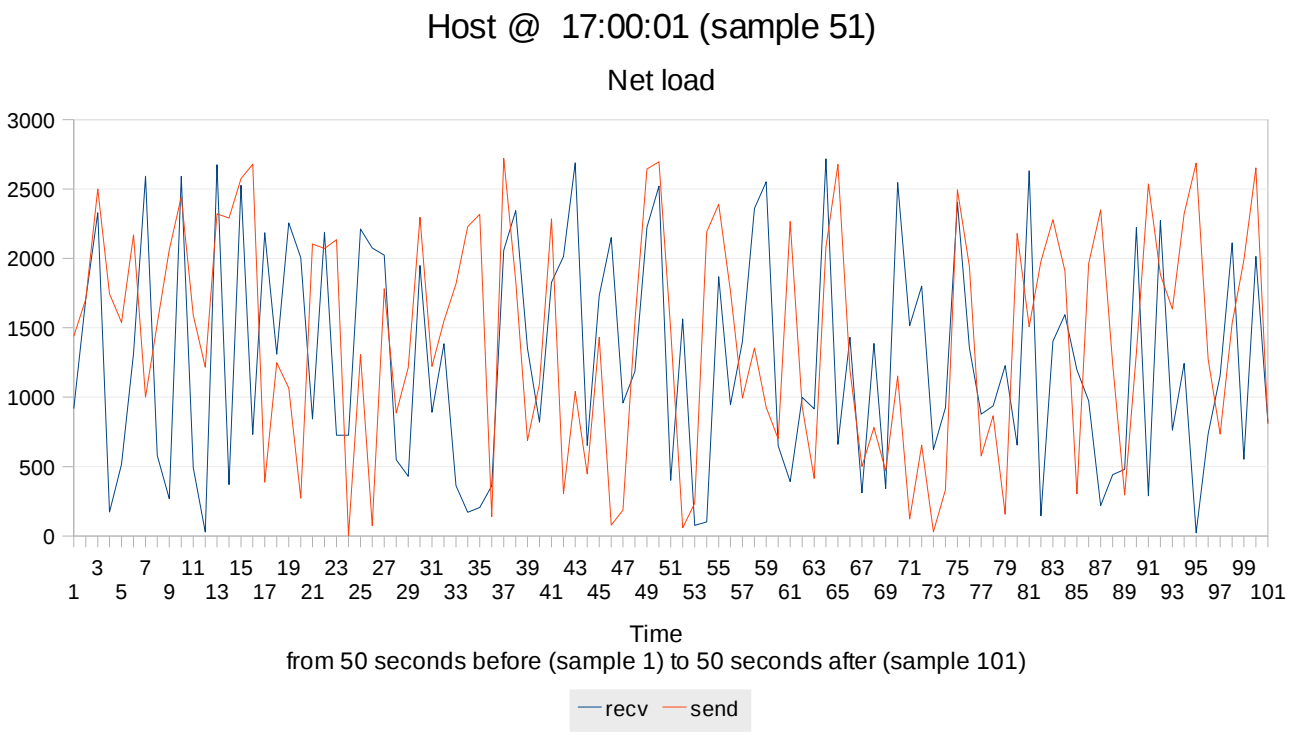
Here is a trick for tuning system in order to avoid swapping, anyway.

$ cat /proc/sys/vm/swappiness

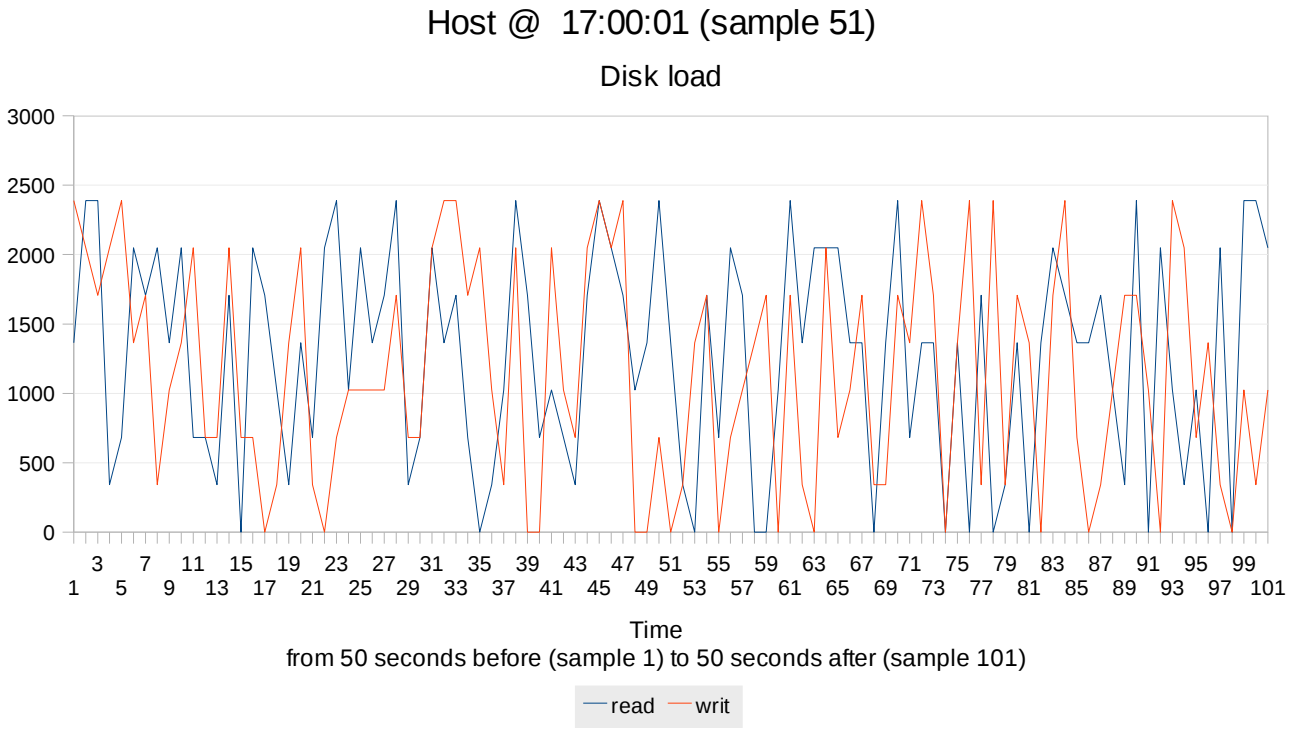
Usually is 60

$ sysctl -w vm.swappiness=10

Add: vm.swappiness=10 to /etc/sysctl.conf



Aboslutely normal network activity.



Minor and normal disk activity.

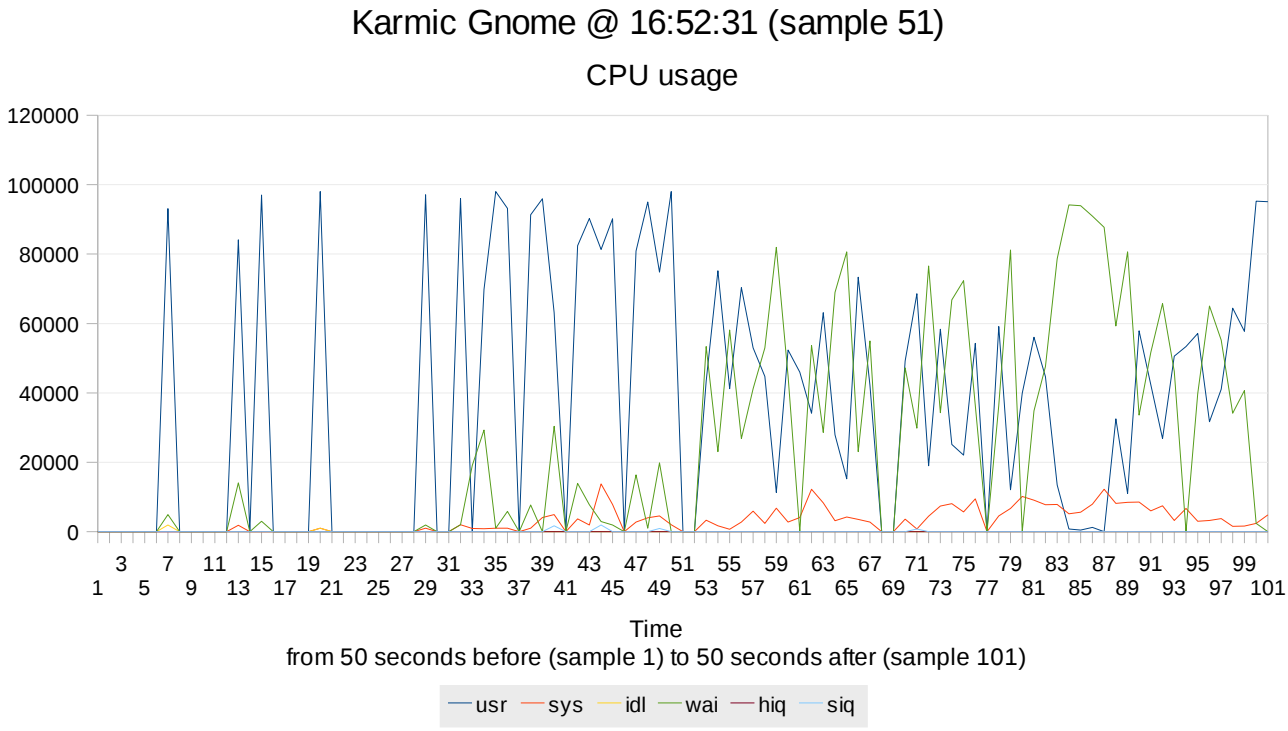
We can conclude that CPU of the guest was a bit stressed because of I/O peaks when a -2 was reported. This generated a temporary increment in load avarage and some disk burst for two

minutes, after which the system was again responsive. As a matter of fact, next operations are rated 0. Nothing relevant in charge of the host.

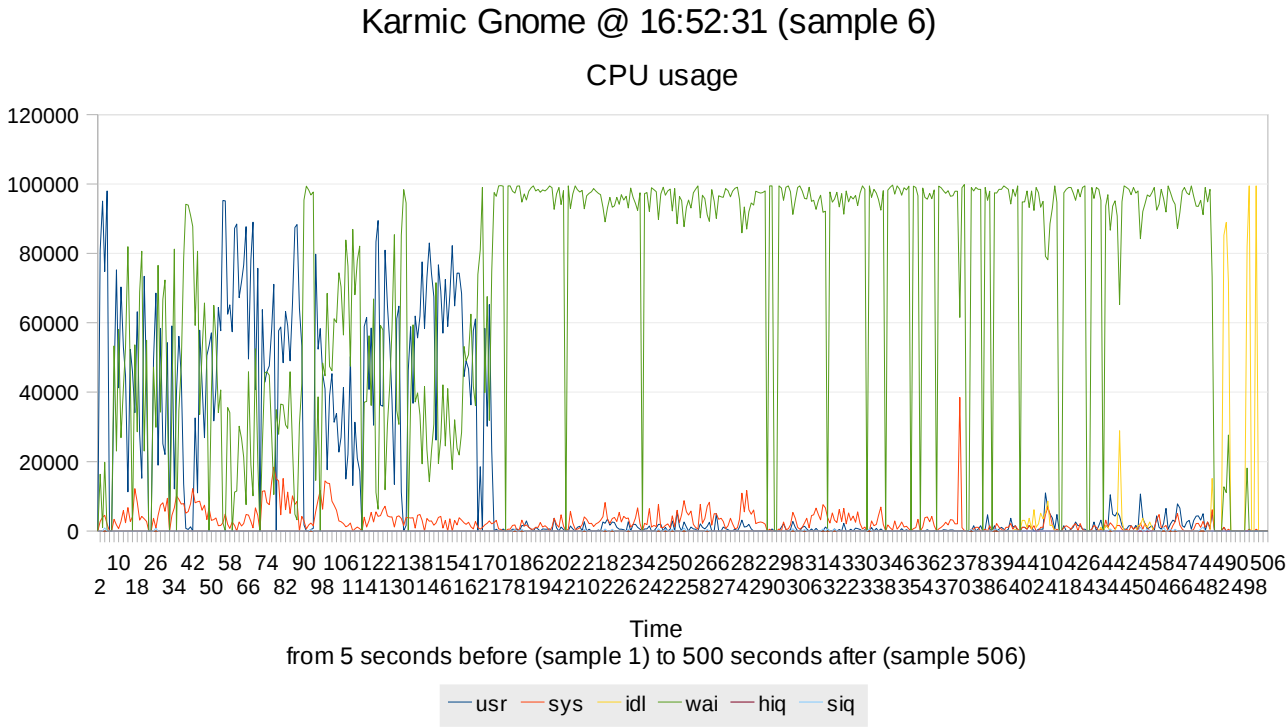
4.1.2 User1 Karmic Gnome

#	Timestamp	Evaluation	Time to complete	Notes
0	-	0	-	
00	-	0	-	
a	-	0	29.396	
b	-	0	3.04	
c	-	0	1.61	
d	-	0	11.5	
e	-	0	20.88	
f	-	0	5.5	
g	-	0	21.7	
5	16:17:50	-1		
2	16:27:13	0		
13	16:31:15	0		Slightly slow
8	16:38:30	0	mv 0.27 cp 0.13	
1	16:42:18	0		
3	16:45:38	0		
6	16:50:12	0		
12	16:52:30	-2		Very slow, do not open any filter
7	17:05:30	0		
10	17:06:15	0	mv 0.05 cp 5.97	
4	17:09:40	0		
11	17:11:50	0		
9	17:12:51	0	mv 0.2 cp 0.3	

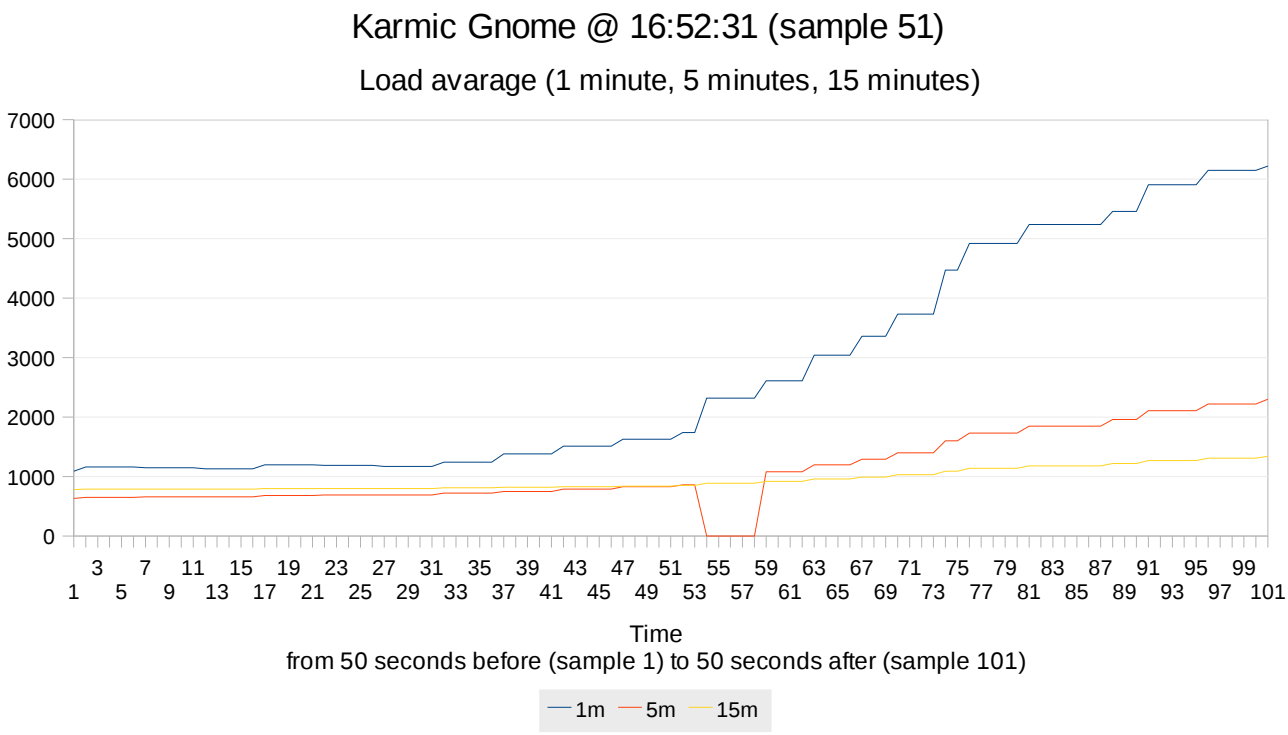
Let us see what happened at 16:52:30 (sample 5314), using sys_graphs function this way: =SYS_GRAPHS(“Karmic Gnome”;5314;50;50) and then enlarging extents, especially on the right, when needed (5;500, 10;200, 5;800, 5;1000).



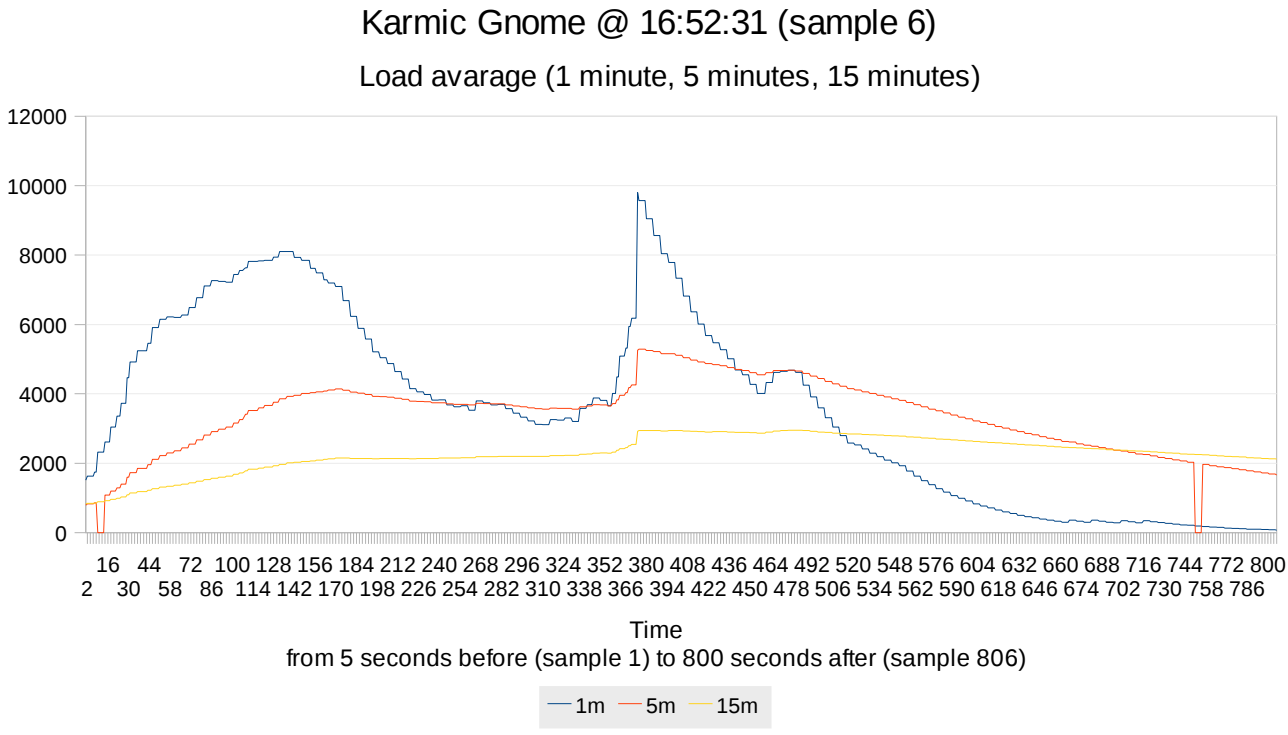
Around sample 51, when problems arise, we notice an increment of wai parameter. Let us enlarge the view on the right (=SYS_GRAPHS(“Karmic Gnome”;5314;5;500)).



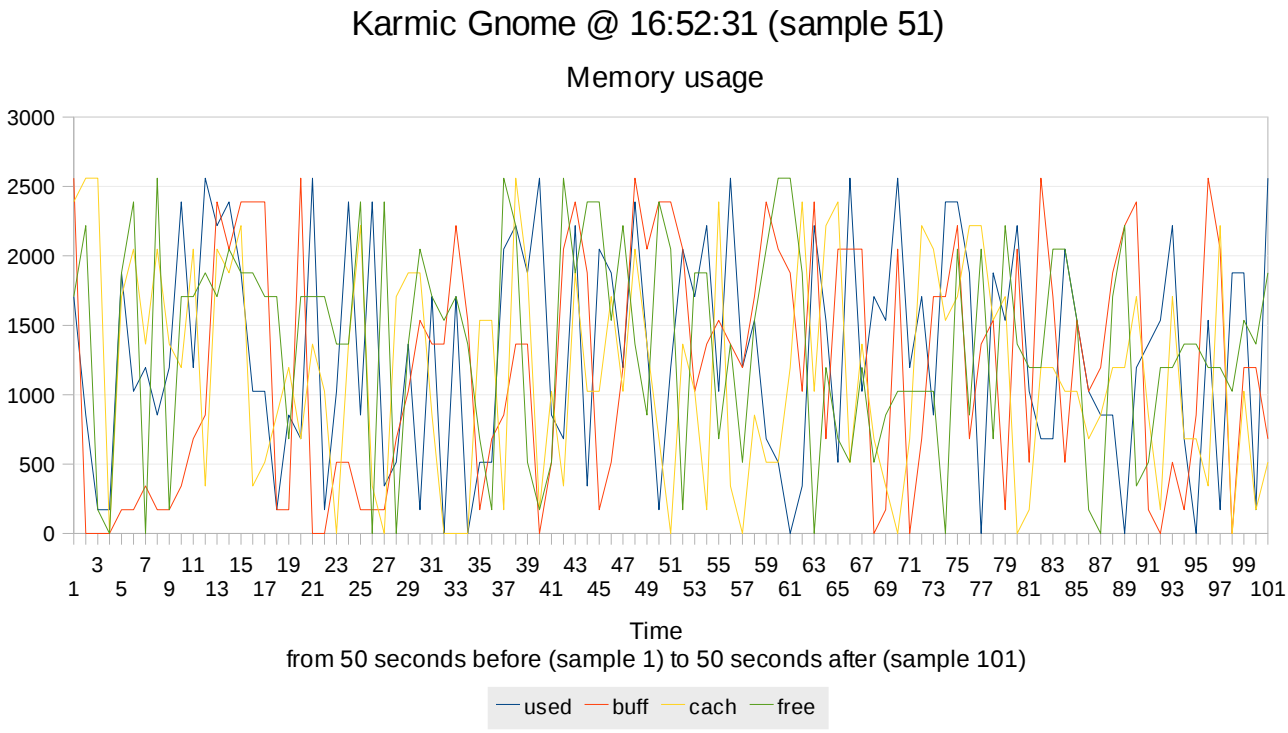
That's it! Wai parameter is the one to look at for 500 seconds (then, it comes down). If one of the wa or hi parameters is high, it can indicate a real problem. Normally, the wa parameter shows how much time the CPU has wasted waiting for I/O. This I/O can come from the hard disk or from the network. Therefore, a high value on the wa parameter often indicates a slow hard disk or network connection. Let us go on.



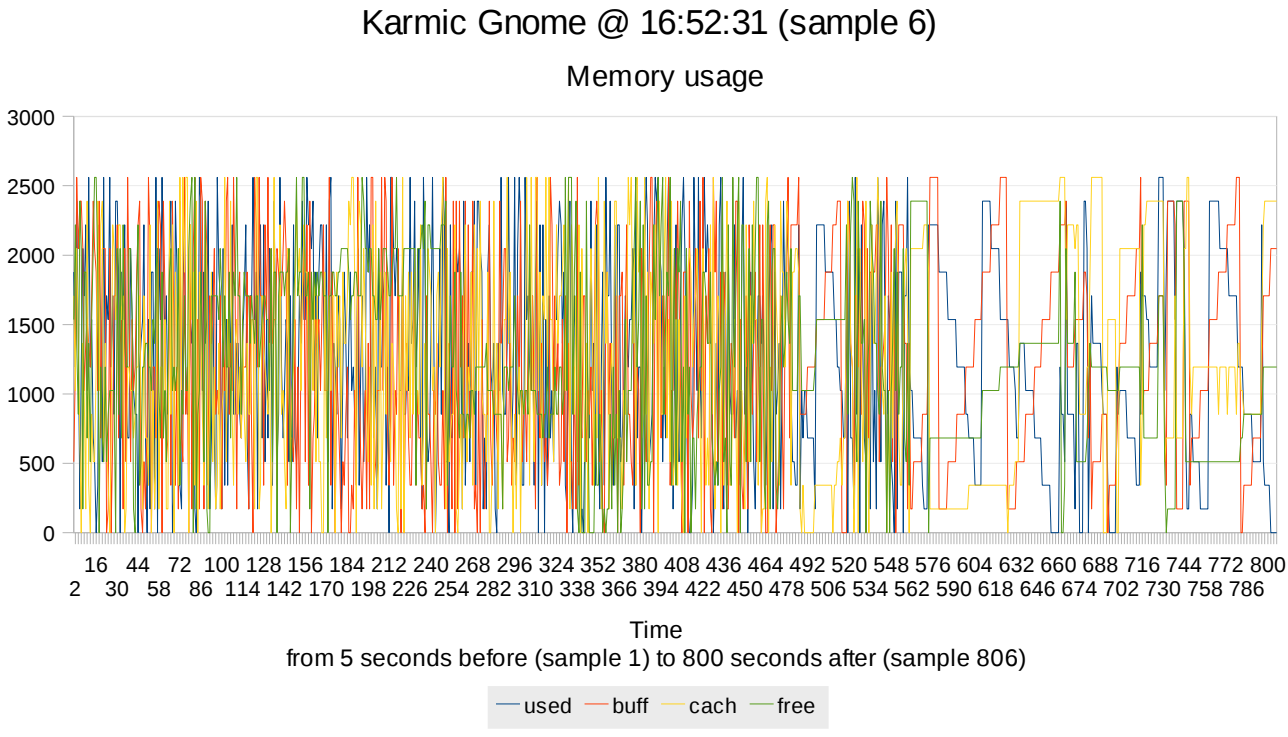
Load avarage increases. Let us see when it decreases back by enlarging again on the right (=SYS_GRAPHS(5314;5;800)).



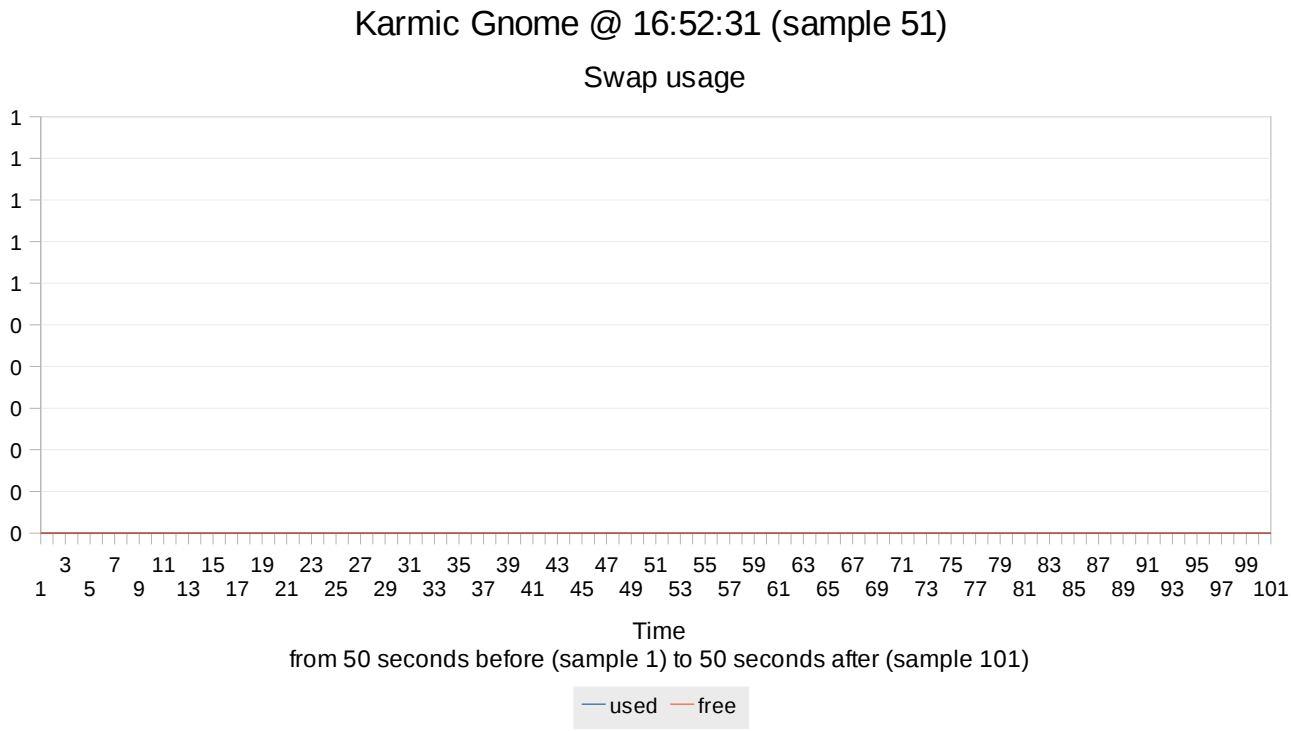
This conferms 500 seconds are needed for the system to come back working fine.



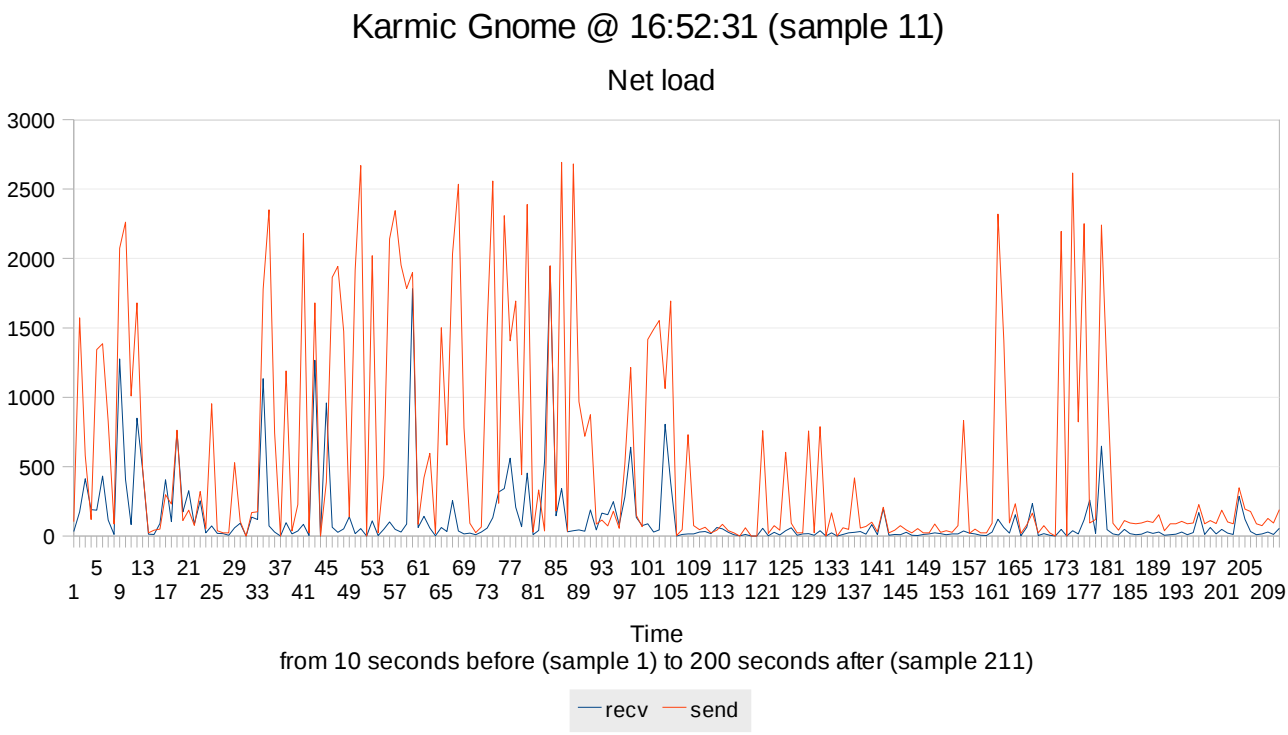
Apparentrly memory usage is not a problem. But let us enlarge the view again (=SYS_GRAPHS("Karmic Gnome";5314;5;800)).



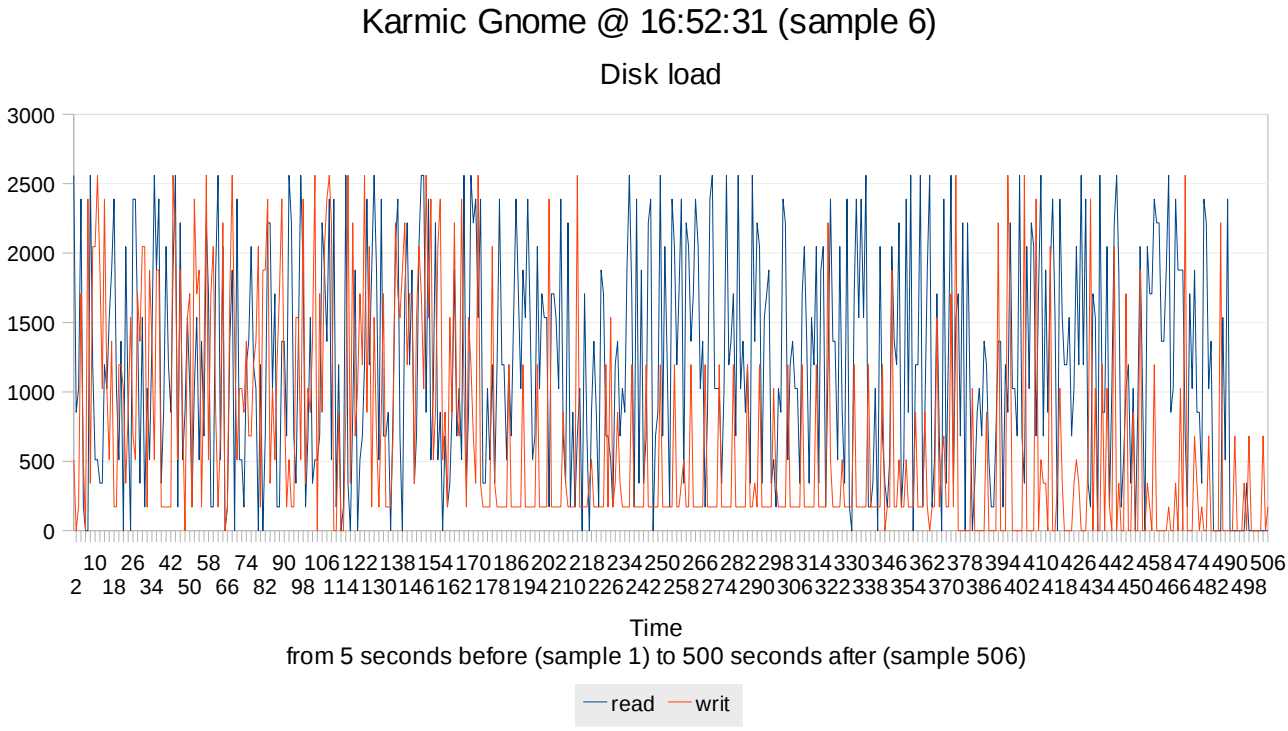
After 500 seconds, memory starts to be freed. This is another indicator of the same event, even if the memory consumption is not critical in this moment.



No swap at all, as a matter of fact.

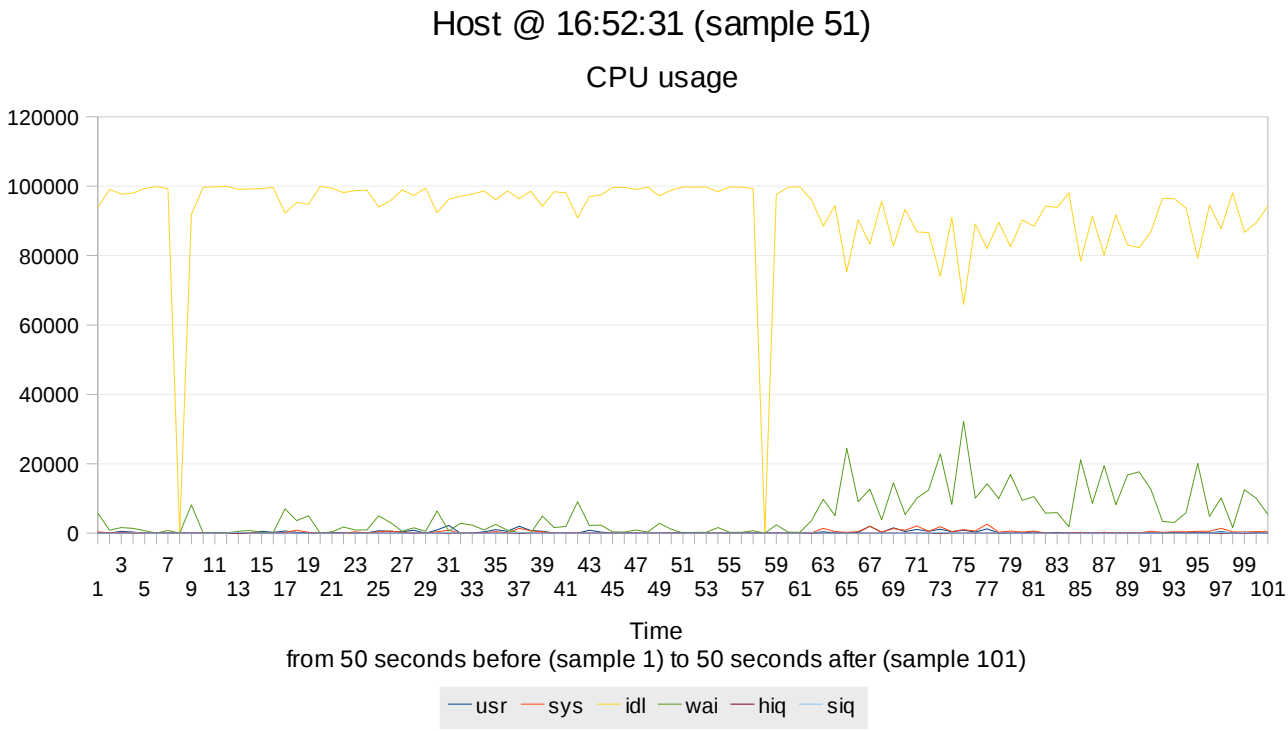


Net load is interesting, as we have said that net could be the reason for wai paramater to be so high. But the level is not high at all and after 100 seconds it decreases, while the rest of the system is experiencing troubles. Let's go to see the disk, then.

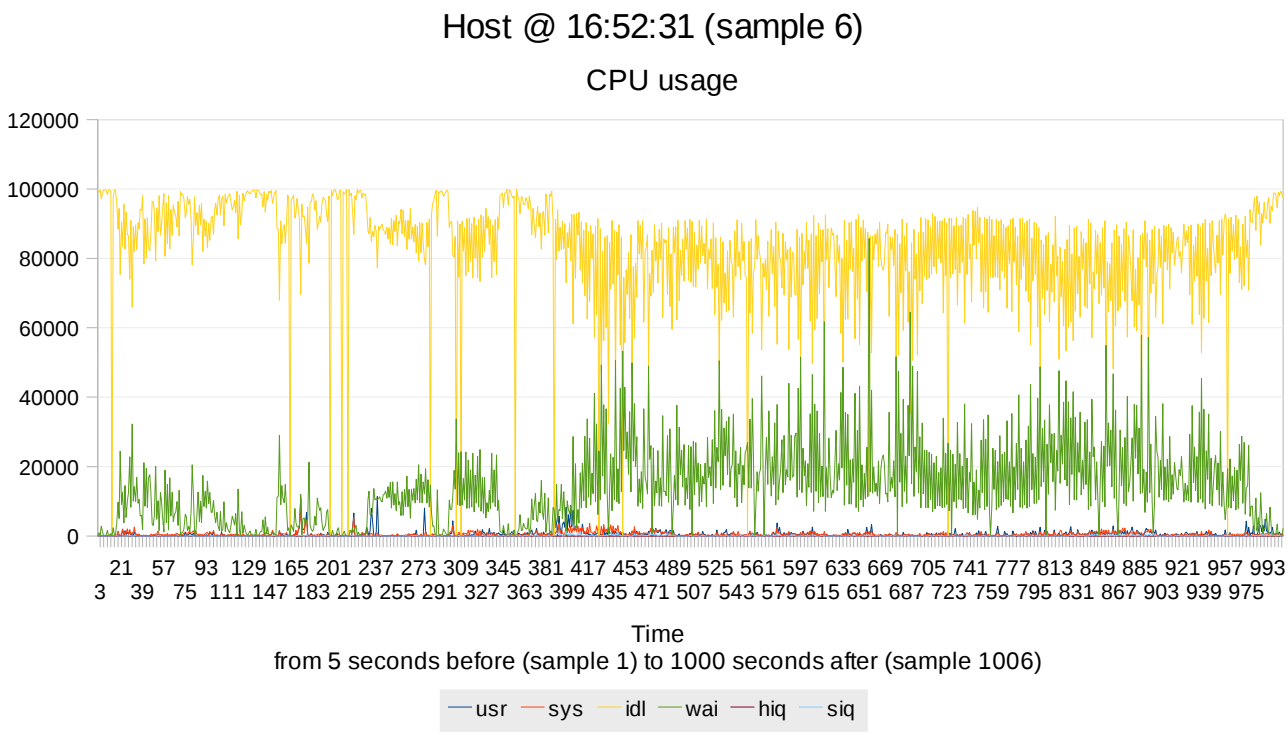


It was the disk to make CPU waiting, indeed.

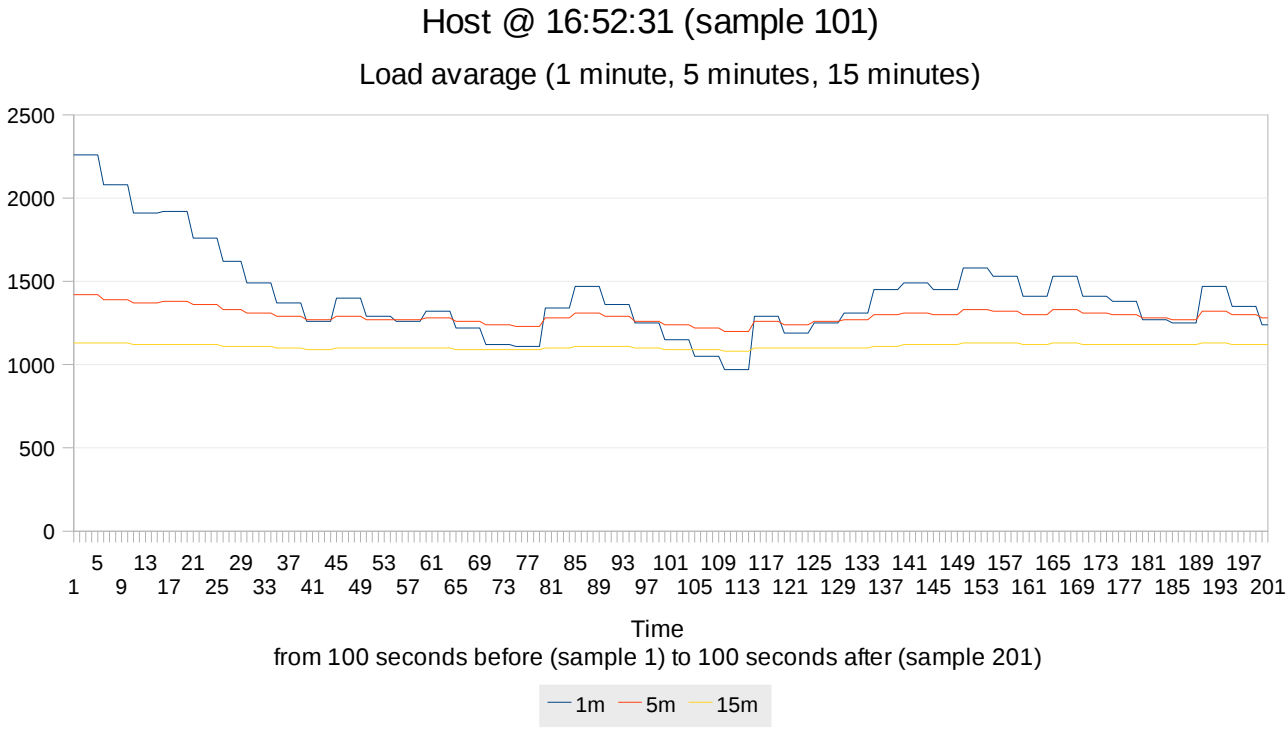
Let us see what happened in the host in the meanwhile.



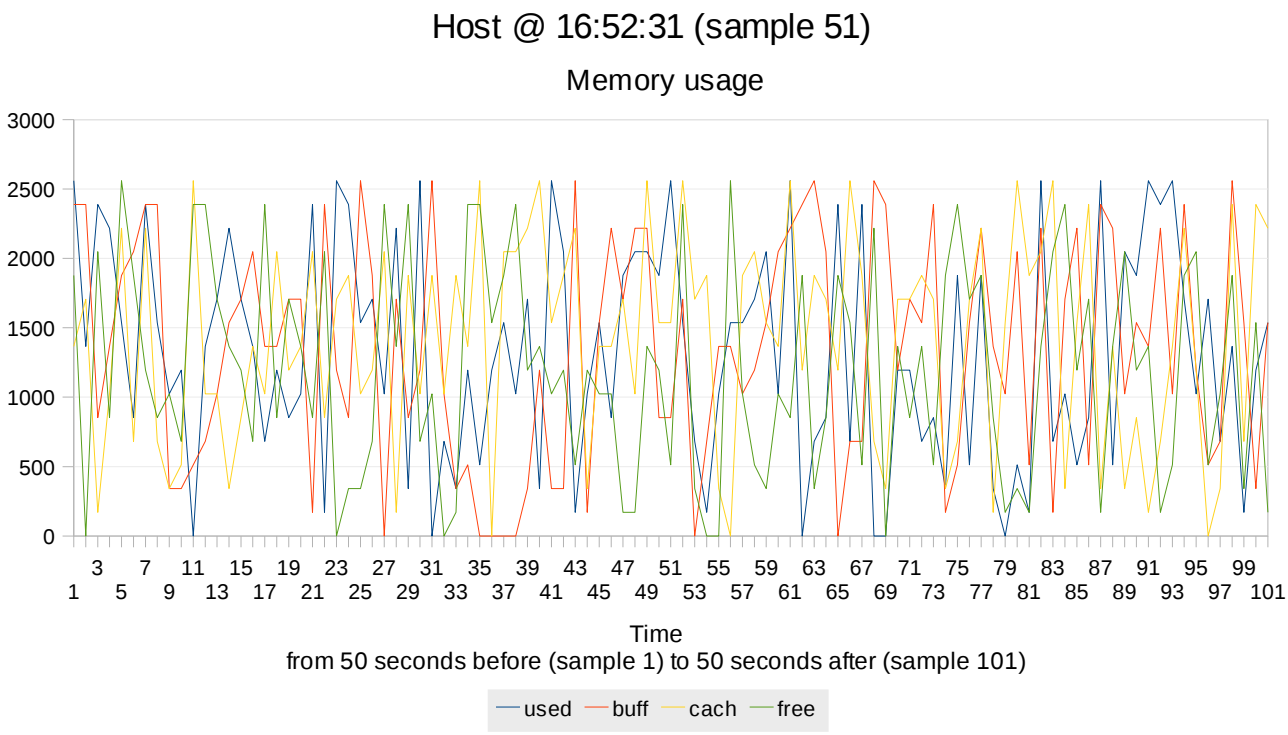
Something happened to the host as well in that very moment (just a few samples of difference, due to initial offset when launching subsequently dstat on more machines). Let us enlarge the view.



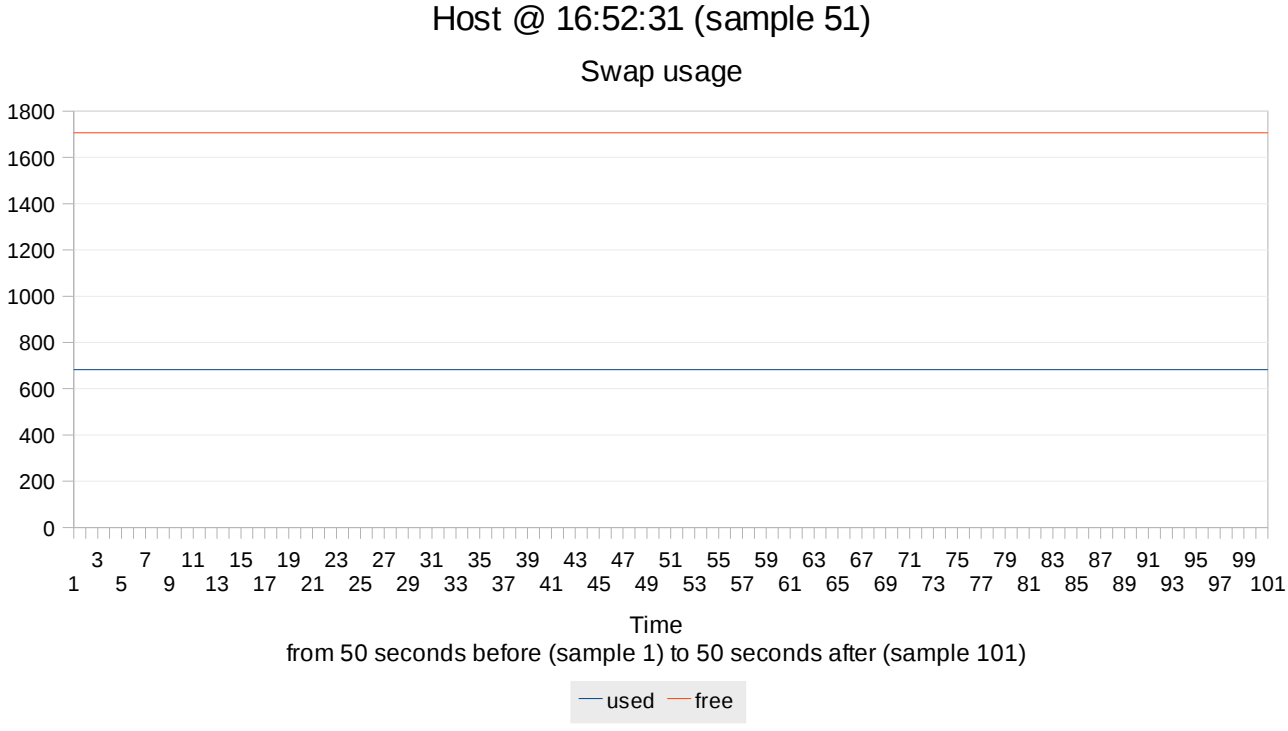
Apparently, the host is quite affected by what is happening in VDD in the next 1000 seconds. Let us see how much.



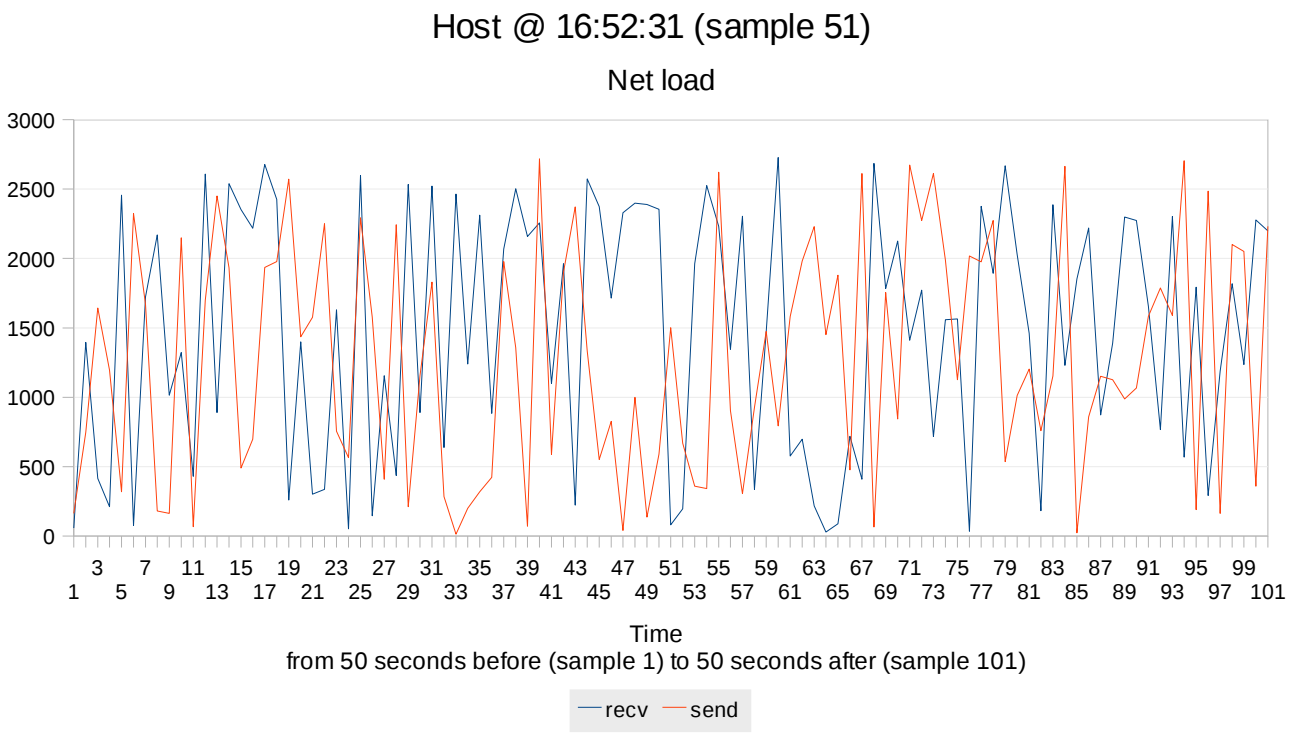
Not that much. Load avarage is quite flat and low. Host has got much more resources to catch up.



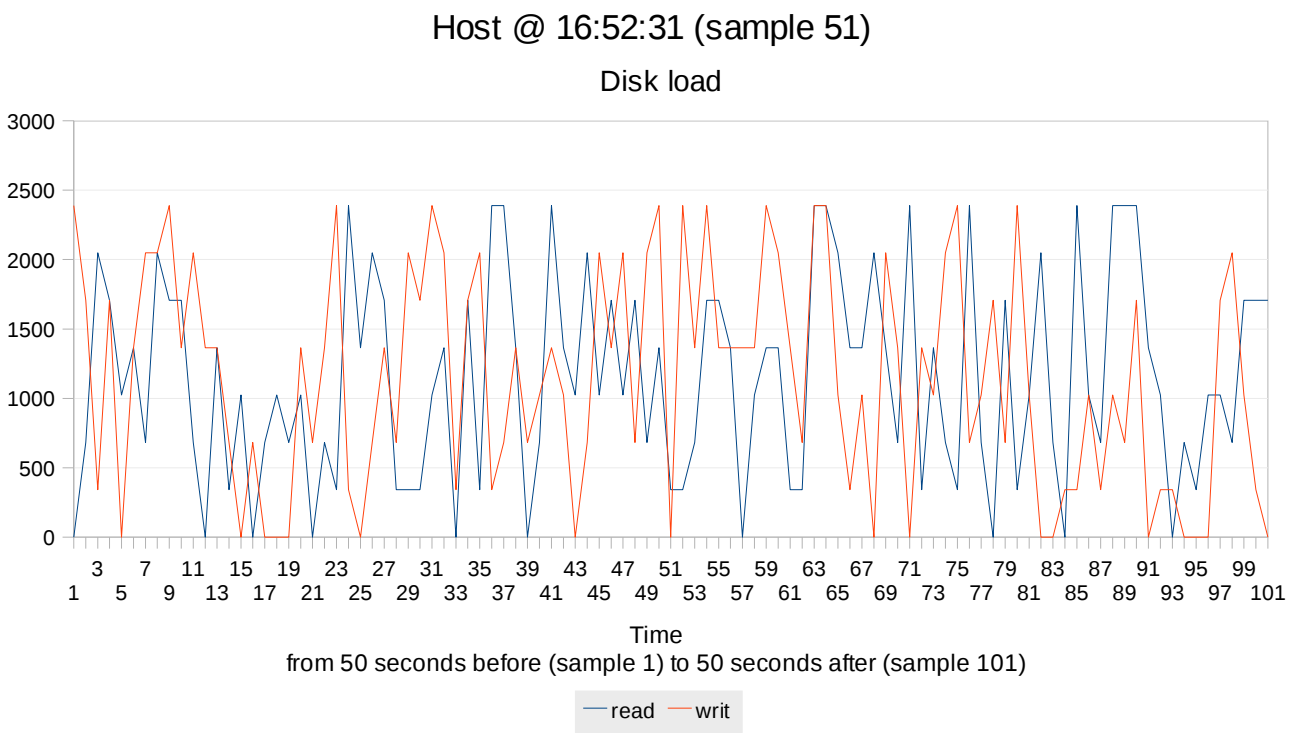
Memory usage is regular.



Swap is constant (see note in 4.1.1)



Network activity is regular.



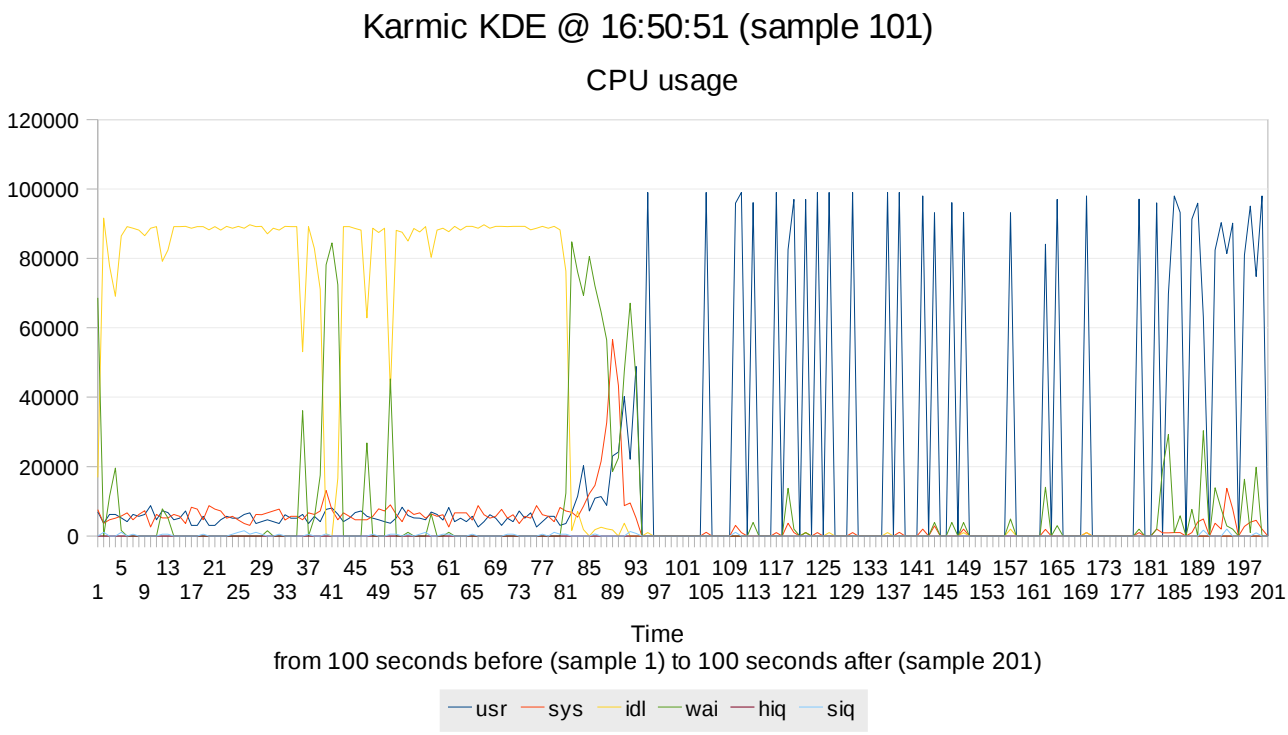
And disk load is regular too.

So the problem was I/O from disk for the dispatched desktop (Ubuntu Karmic). We need to take into account that karmic is being used by two users (the other with KDE). We will see what he or she was doing in that very moment (see 4.1.3).

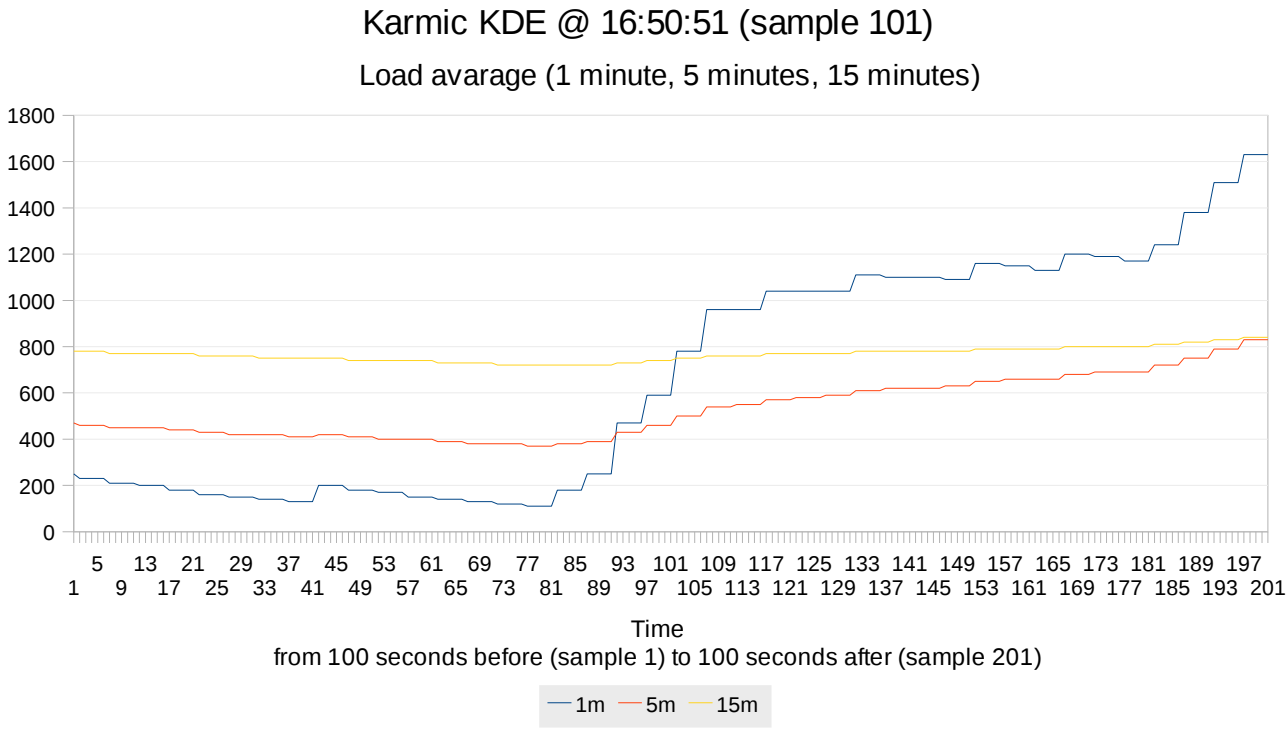
4.1.3 User2 Karmic KDE

#	Timestamp	Evaluation	Time to complete	Notes
0	-	0	-	
00	-	0	-	
a	-	0	9.3	
b	-	0	1.9	
c	-	0	0.7	
d	-	0	6.6	
e	-	0	10.2	
f	-	0	3.2	
g	-	0	11.2	
7	16:17:44	-1		Slightly slow when doing “open with” (no by double-clicking)
5	16:23:10	0		
1	16:25:20	0		
12	16:28:17	-2		Failed! Unable to run plugin (impossible to allocate memory)
3	16:32:40	-1		Slow to load
2	16:37:40	0		
4	16:39:04	0		
11	16:39:05	0		faster then fedora
6	16:43:40	-1		no audio
10	16:46:35	0	mv 0.05 cp 4.3	
8	16:48:55	0	mv 0.03 cp 0.06	
13	16:50:50	-2	4.57	!
9	16:55:30	0	mv 0.44 cp 1.35	

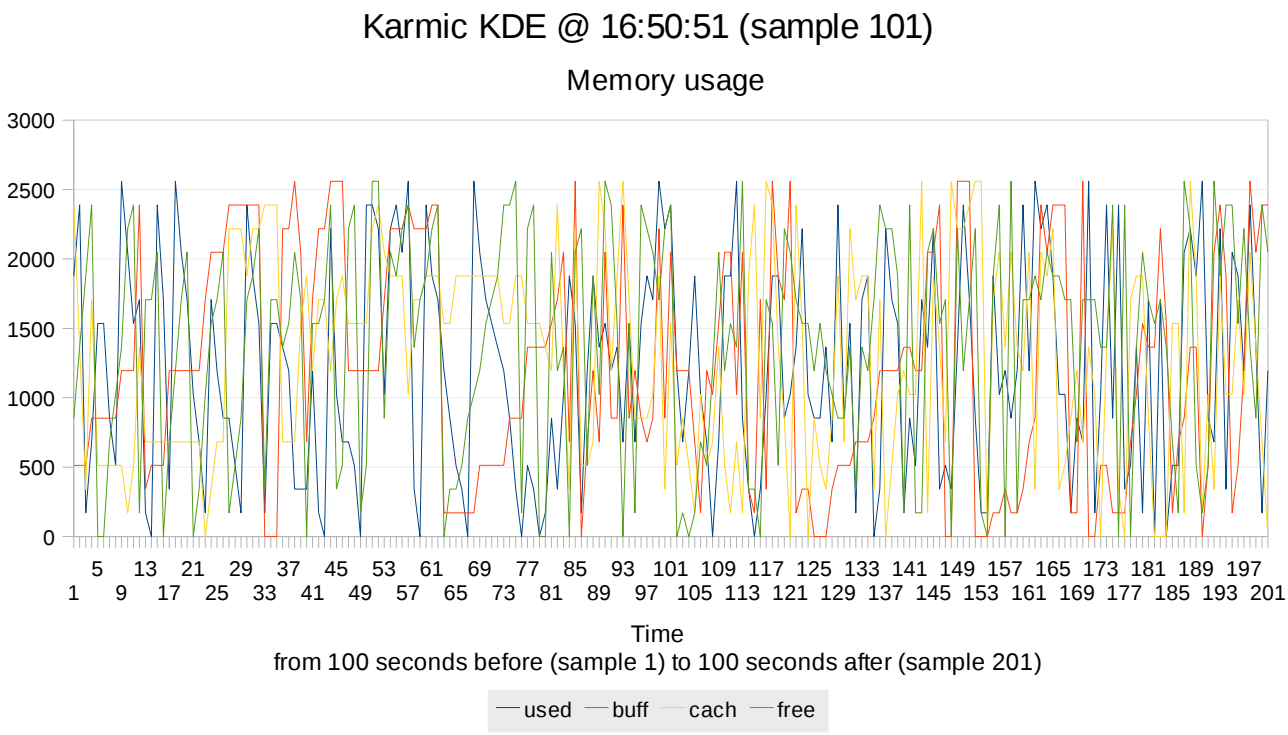
Time of trouble is 16:50:50, corresponding to 5214, just 100 seconds before the unacceptable event of user 1 analysed before. =SYS_GRAPHS(“Karmic KDE”;5214;100;100)



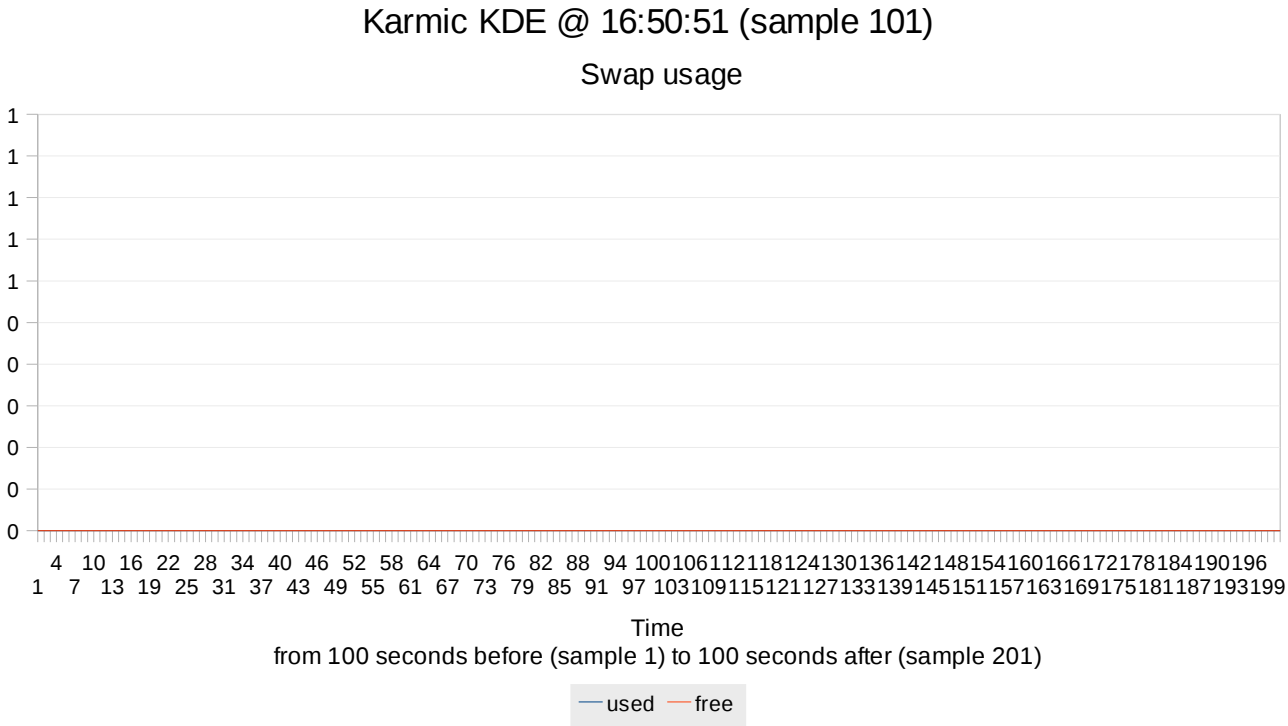
We can clearly see usr parameter bursting for three/four seconds. This indicates CPU-bound activity, in particular in user mode (whereas sys represents kernel mode). In fact, user2 is performing music tracks conversion from wav to mp3 and this is taking massively CPU resources.



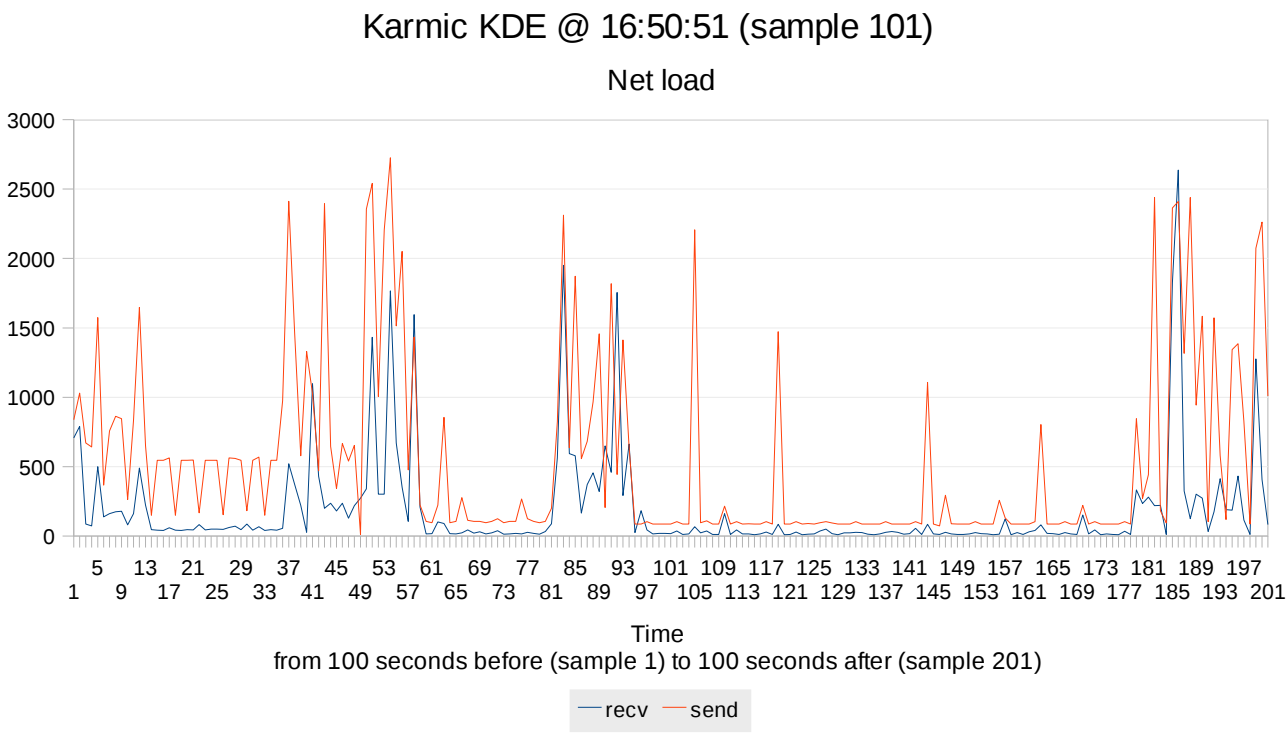
Load increases suddenly. It will last for 5 minutes and as we have said this influences Karmic Gnome desktop, where user1 is trying to apply graphic filters using GIMP (a CPU-bound activity which implies I/O as well).



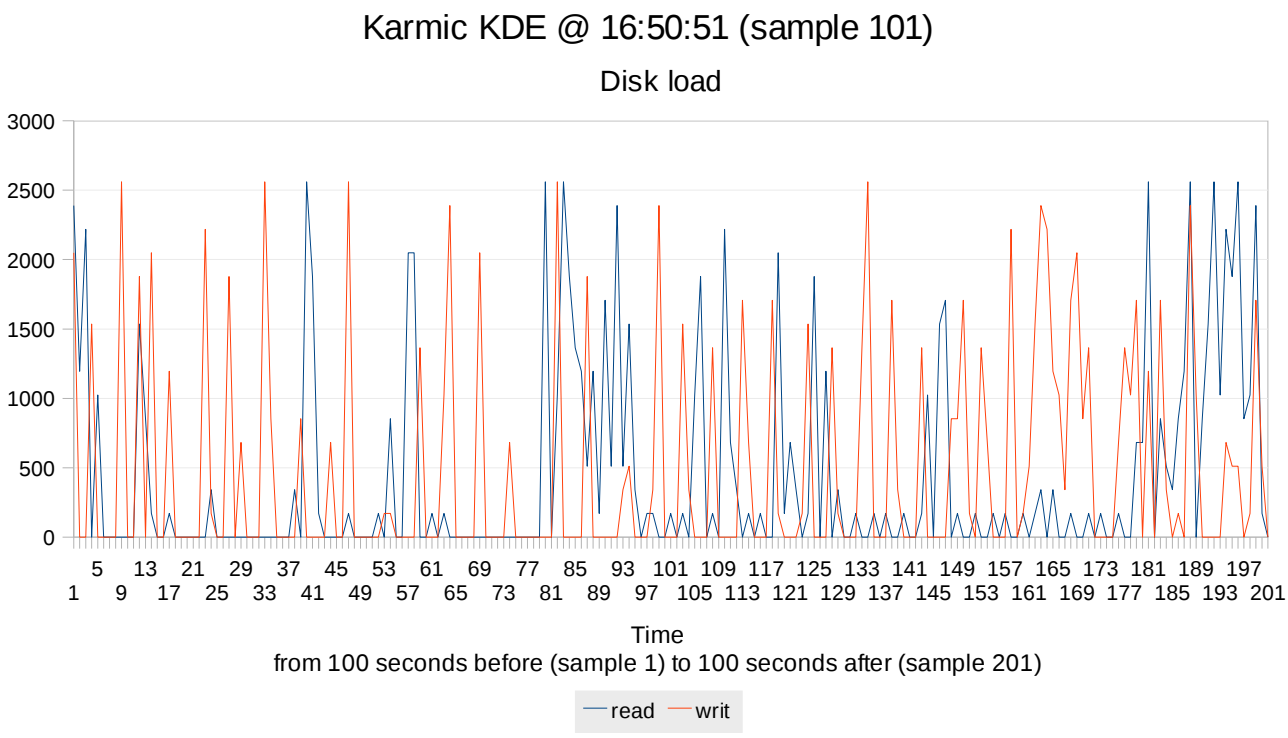
Memory is not affected at the moment.



And swap is zero.

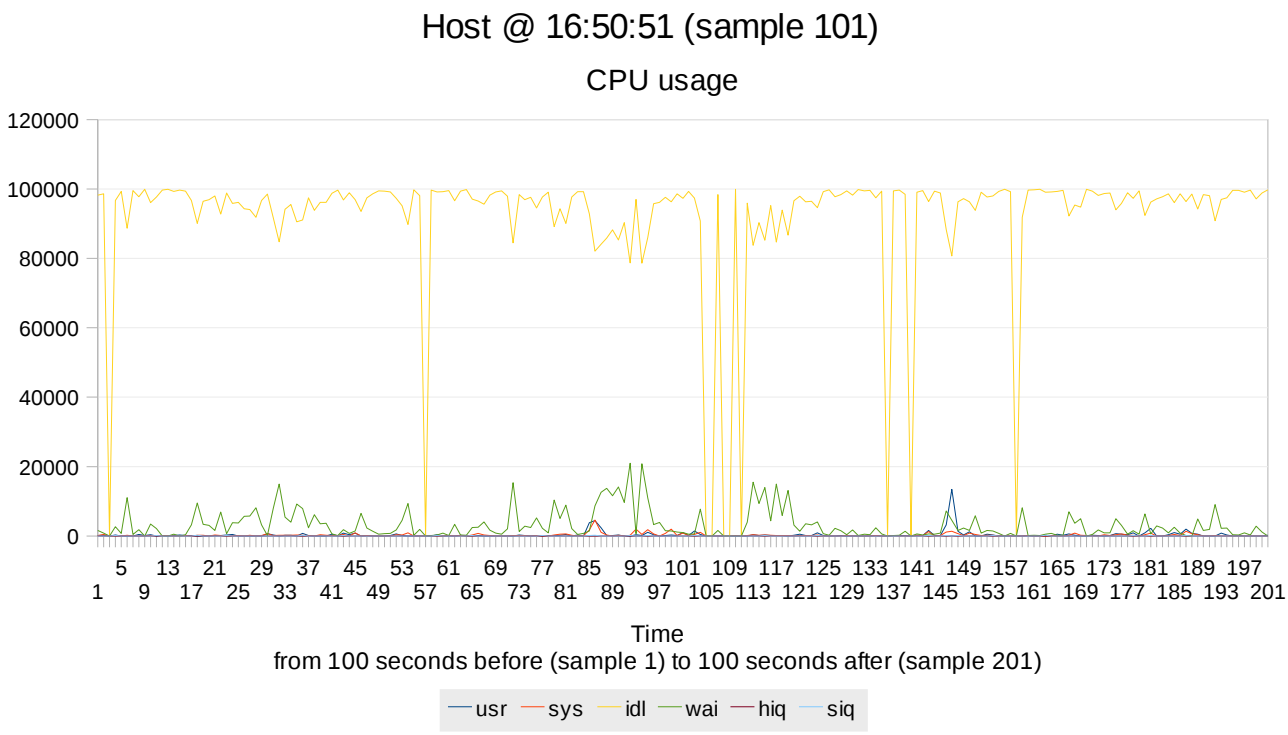


Network activity is normal conditions.

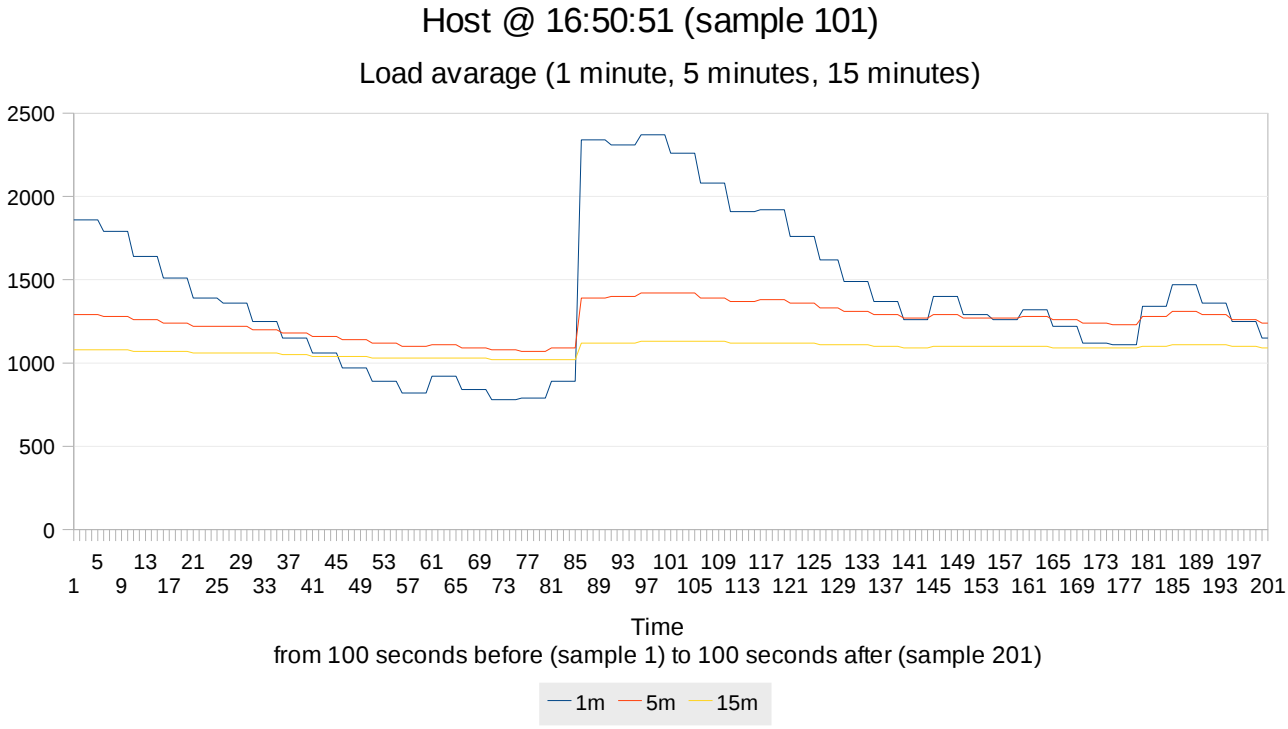


Disk activity presents some frequent spikes, as we have already seen in Karmic Gnome analysis.

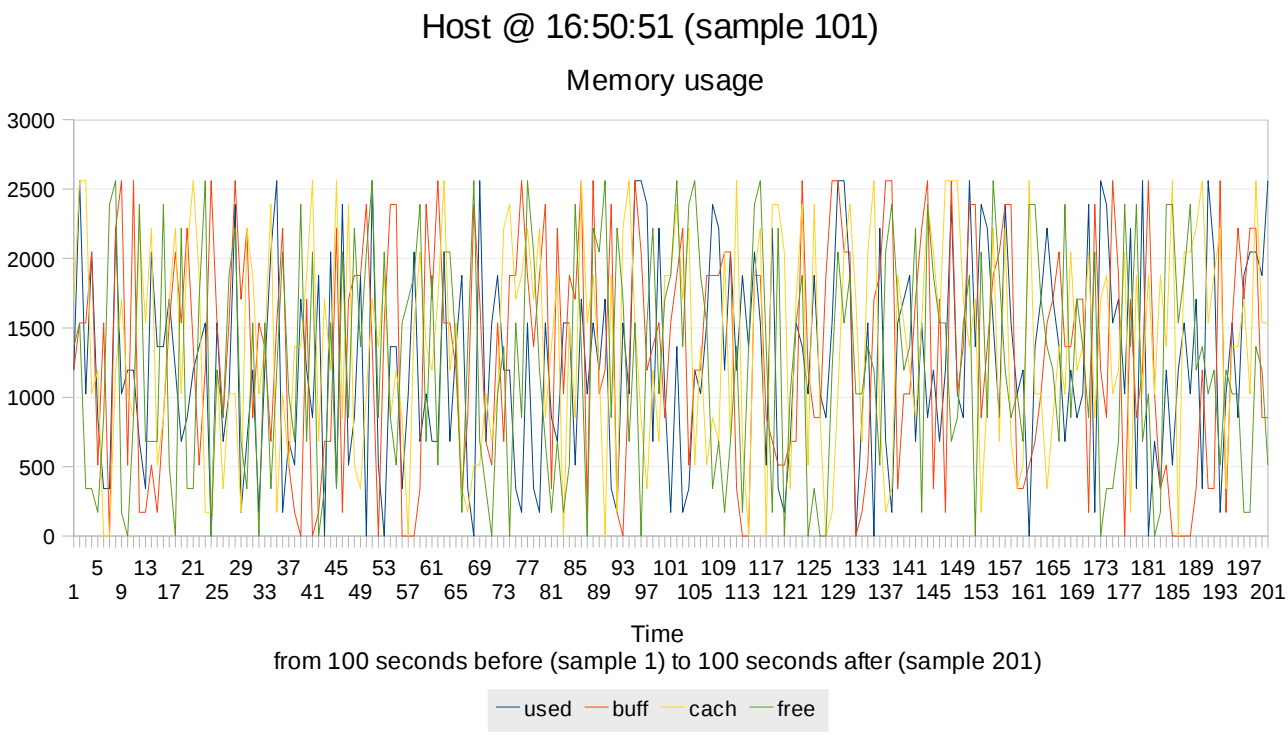
Let's analyze the host now.



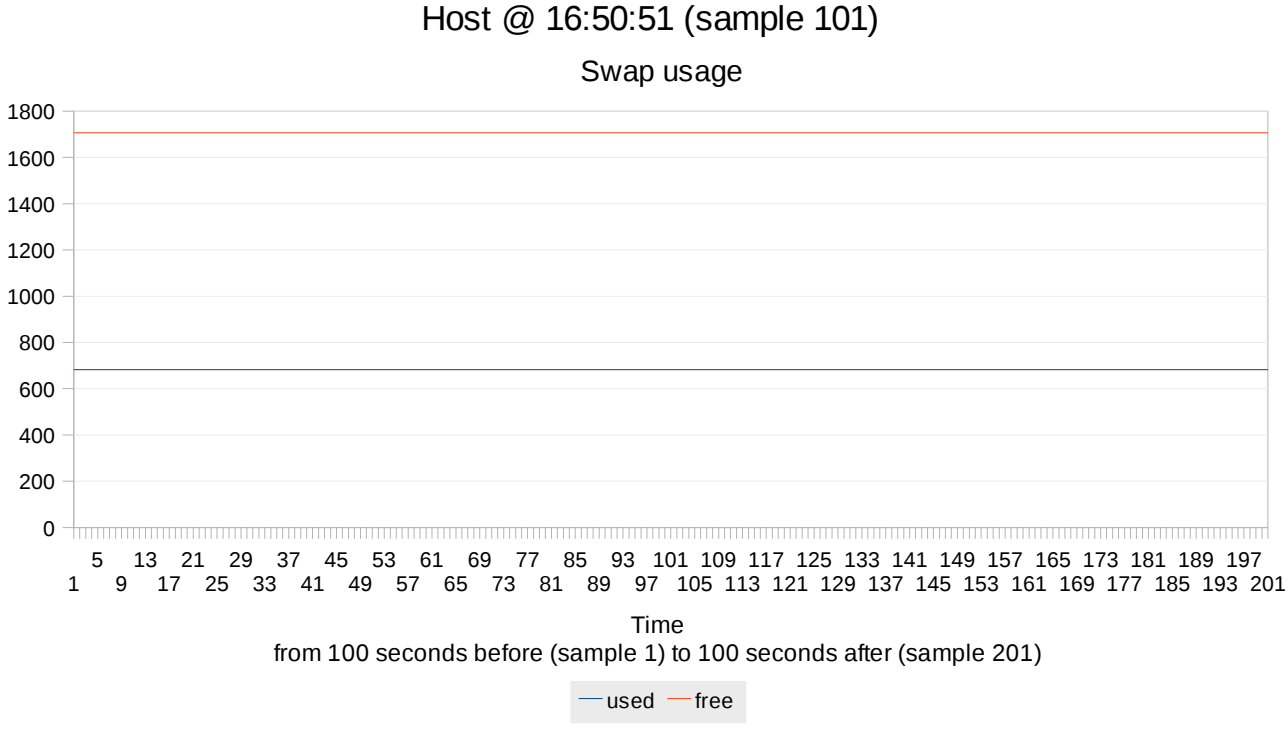
CPU presents a negative spike for idle around sample 101, but resources are absolutely under control.



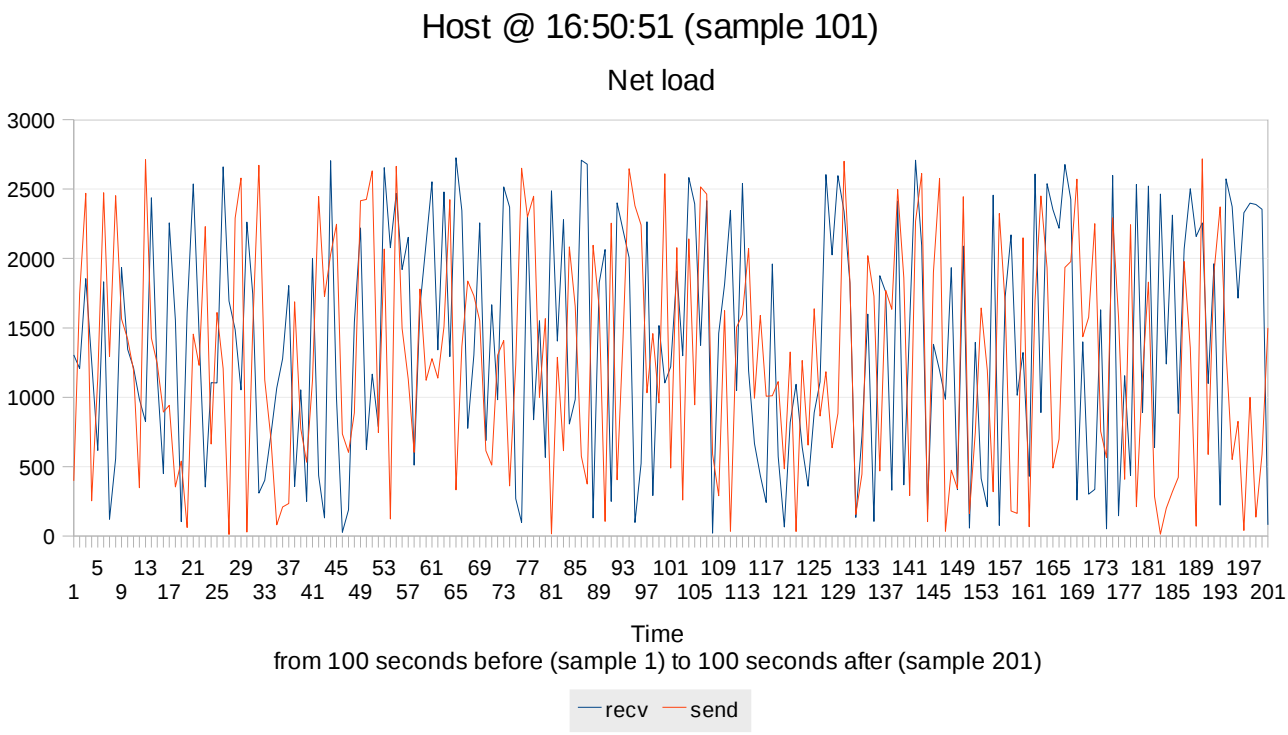
A small increment for load avarage, immediately subdued.



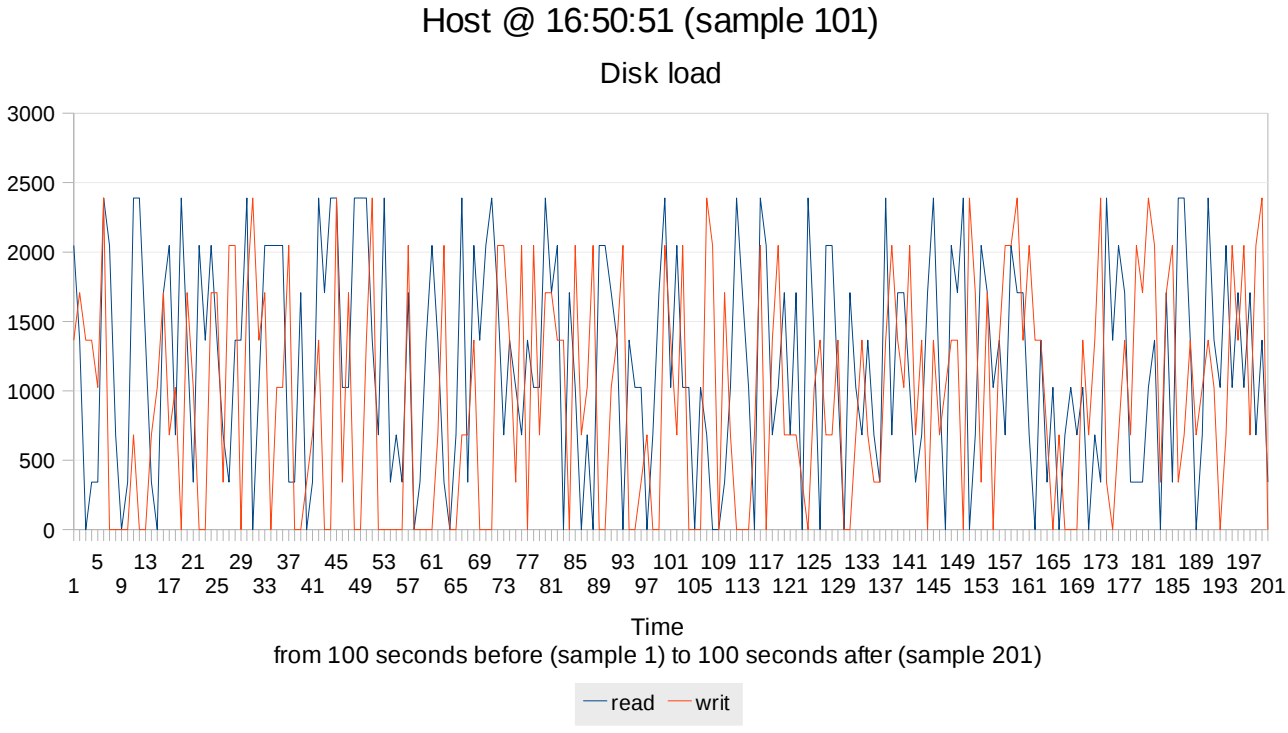
Memory usage is regular and not varying much.



A small swap usage (see 4.1.1).



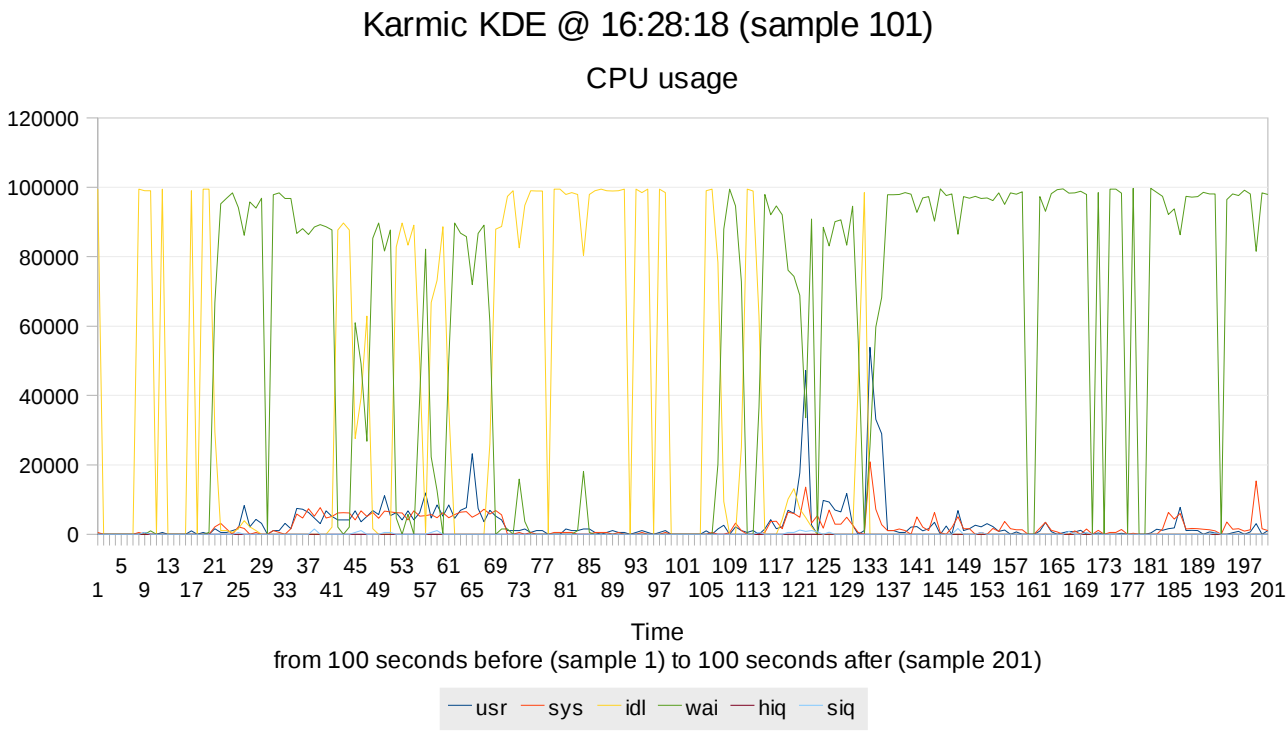
Net load is present, but under control.



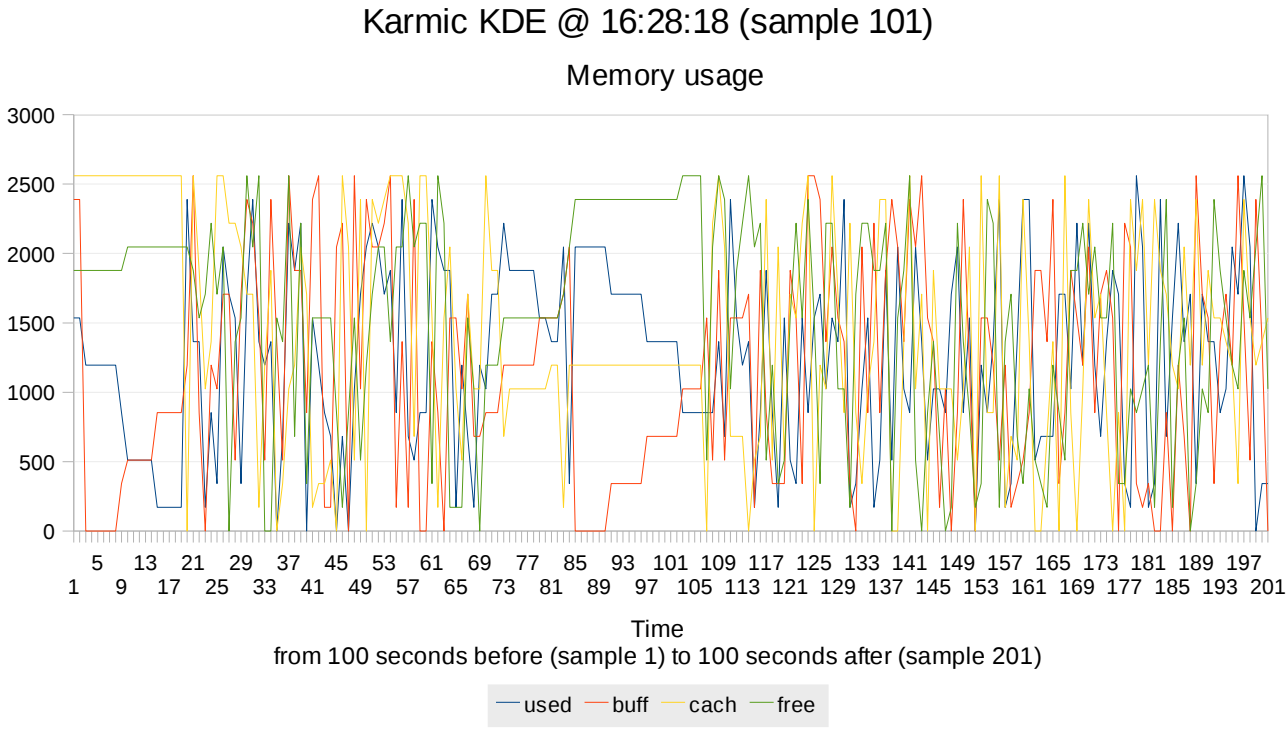
Disk is working fine.

So, the big problem was that two users were contemporary trying to do a thick CPU-bound activity nearly at the same time (user2 a couple of seconds before, so user1 had major malfunctiongs (not possible to apply filters in GIMP).

User2 indicates another big issue around 16:28:17 (sample 3861): =SYS_GRAPHS("Karmic KDE";3861;100;100). We report only two graphs to explain what happened (as the others do not present critical situations).



Big wai spikes associated with respectable usr spikes and minor but relevant sys spikes. User2 is trying to apply GIMP filters. User1 is using OpenOffice Calc and did not report any problem. The operation of user2 fails, it is not possible to run plugin. Let us see memory.



The message is that it is not possible to allocate memory. In fact we see that memory is mostly free in that very moment.

So what is happening is that the system is not able to use resources that are available, such as memory. User is tryng to perform an operation using quite intensely CPU (usr spikes) and including I/O (big wai spikes), system underneath is trying to allocate memory (sys spikes) but fails. It does not seem to be a problem of VDD, as other parameters are ok, host is ok and the other user of Karmic is ok too. Rather, it is a failure imputable to the functioning of the Ubuntu system itself, that could happen occasionally also outside VDD. Or, in any case, we do not have more means to describe it in a different way with our analysis process.

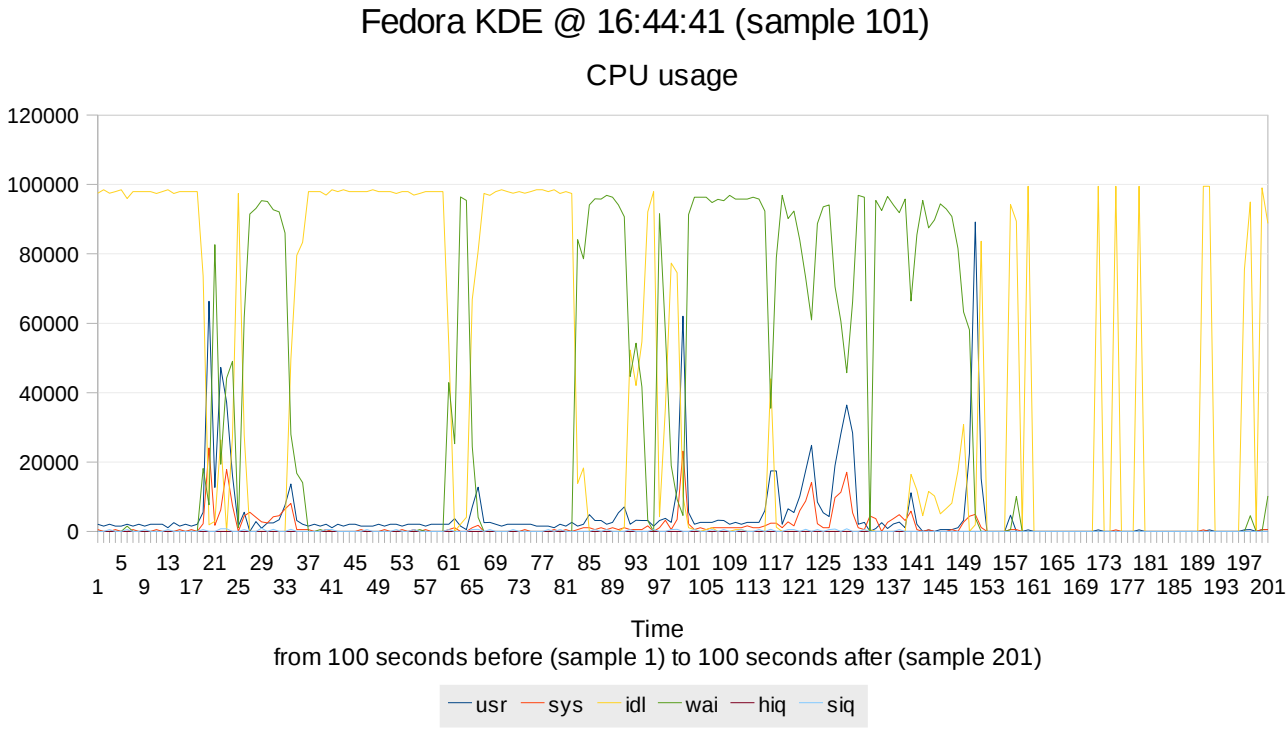
4.1.4 User3 Fedora12 KDE

#	Timestamp	Evaluation	Time to complete	Notes
0	-	0		
00	-	0		
a		-1	13.98	
b		-1	5.2	
c		-1	12	
d		-1	13.5	
e		-1	12	
f		-1	4.1	

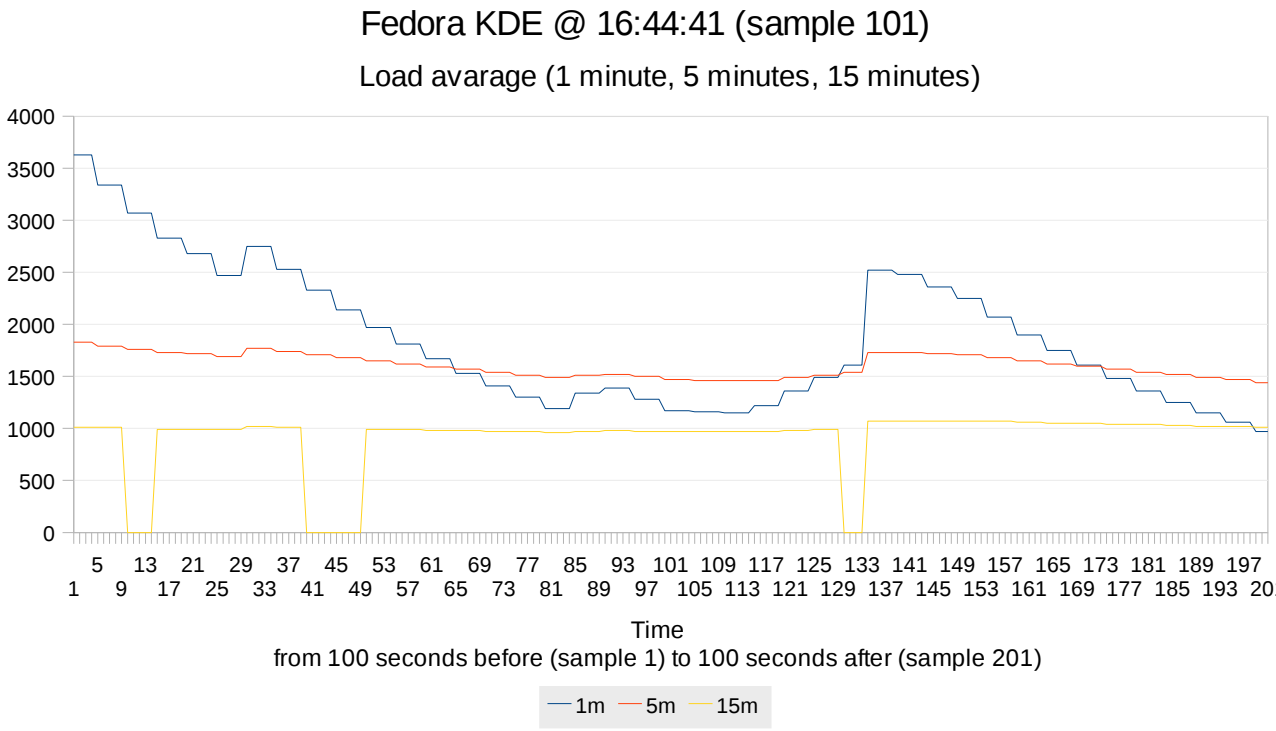
g		-1	8.5	
1	16:19:22	0		
4	16:24:10	0		
11	16:25:52	0		
2	16:30:40	-1		Slow to load
9	16:34:55	0	mv 0.3 cp 0.7	
3	16:36:00	0		
7	16:38:14	0		
10	16:41:20	0/-1	mv 0.5 cp 29	
5	16:44:40	-1		System crashes!
6	16:59:10	0		
8	17:00:00	0	mv 0.2 cp 0.1	
13	17:02:10	-1	2:30	
12	17:07:00	-1		Unable to load plugin

Desktop seems to run better than others. No critical situations are reported. Operations “a” to “g” are marked -1, but times are good, even better than other users marking them with 0. So it must be a subjective verdict.

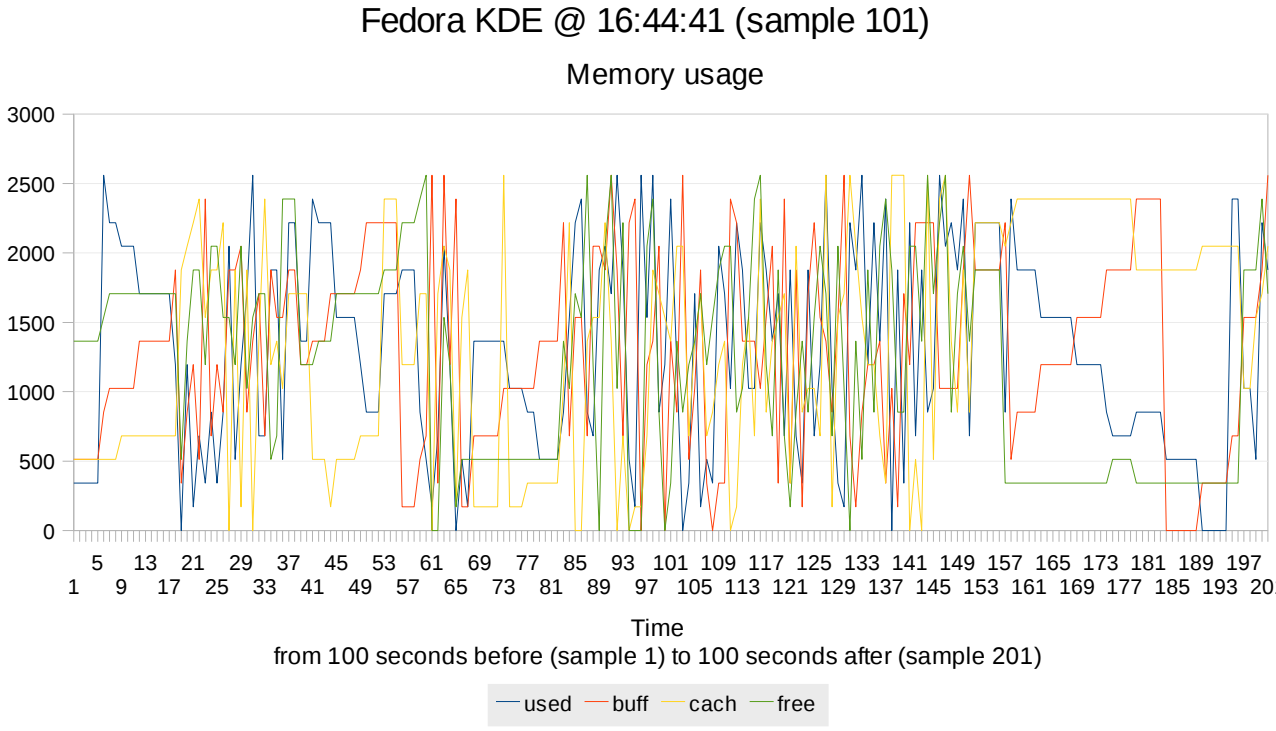
Since there are not significant slow down events to analyze, we take the opportunity to analyze an anomaly, i.e. what happens when one VDD desktop crashes (16:44:40, sample 4881): =SYS_GRAPHS("Fedora KDE";4881;100;100). We report only interesting graphs.



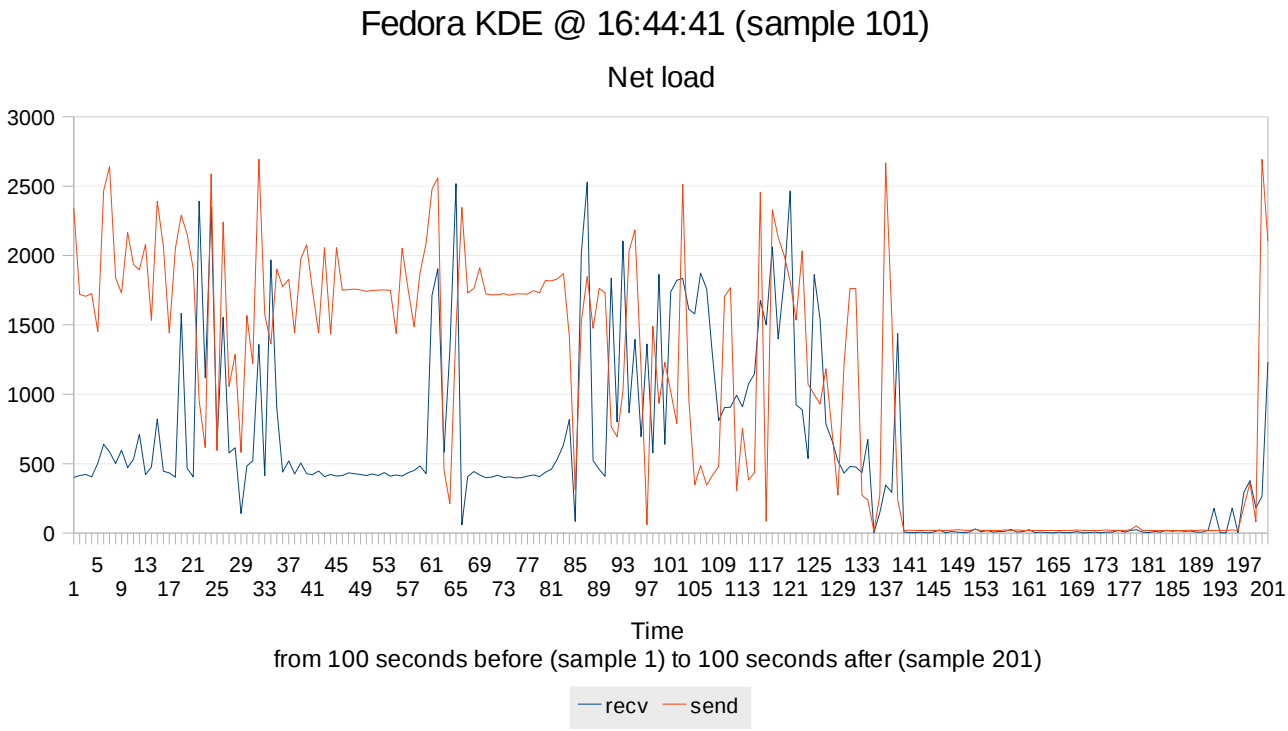
There are some remarkable I/O waiting times on the CPU (green spikes), when, after one minute, suddenly almost every activity ceases on the CPU.



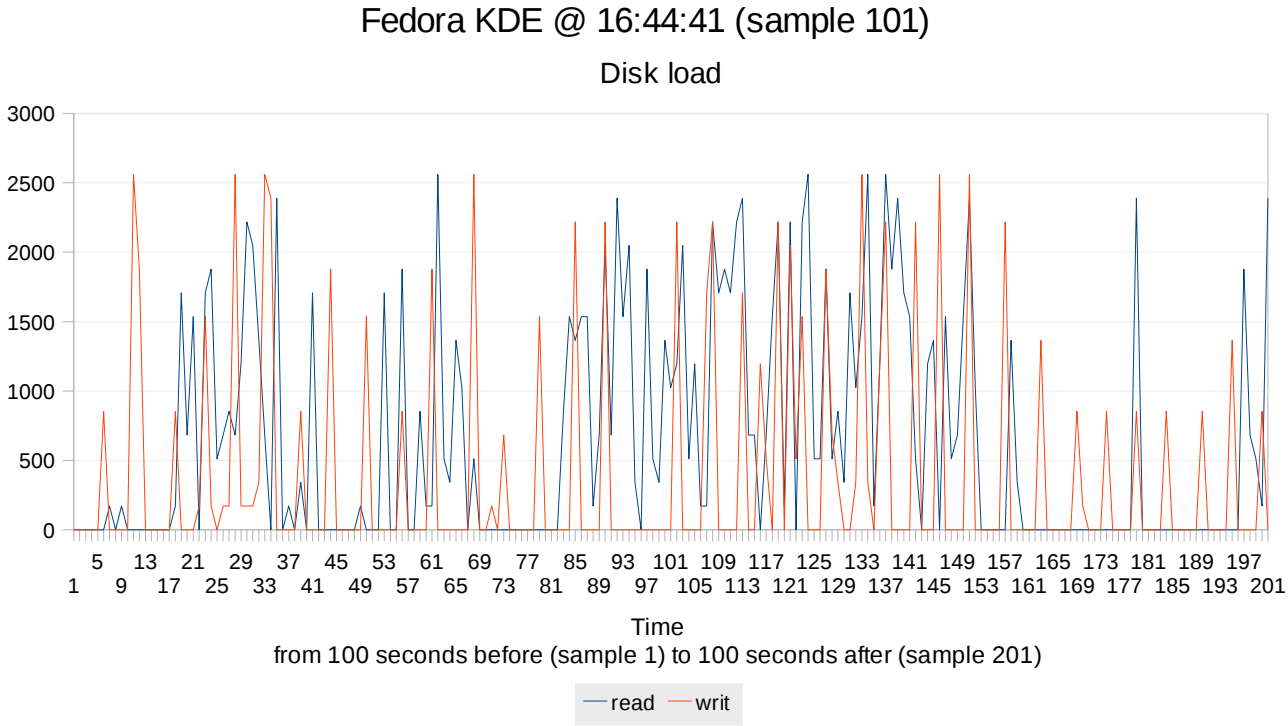
Load avarage is not high and tends to decrease. This indicates that it is not a matter of performance. It must be a creashes of X Window System. Let's see...



There is a respectful burst in the centre of the graph, describing intense but normal activity, then a sudden pause (when system crashes). Swap is zero for the whole extent, anyway.

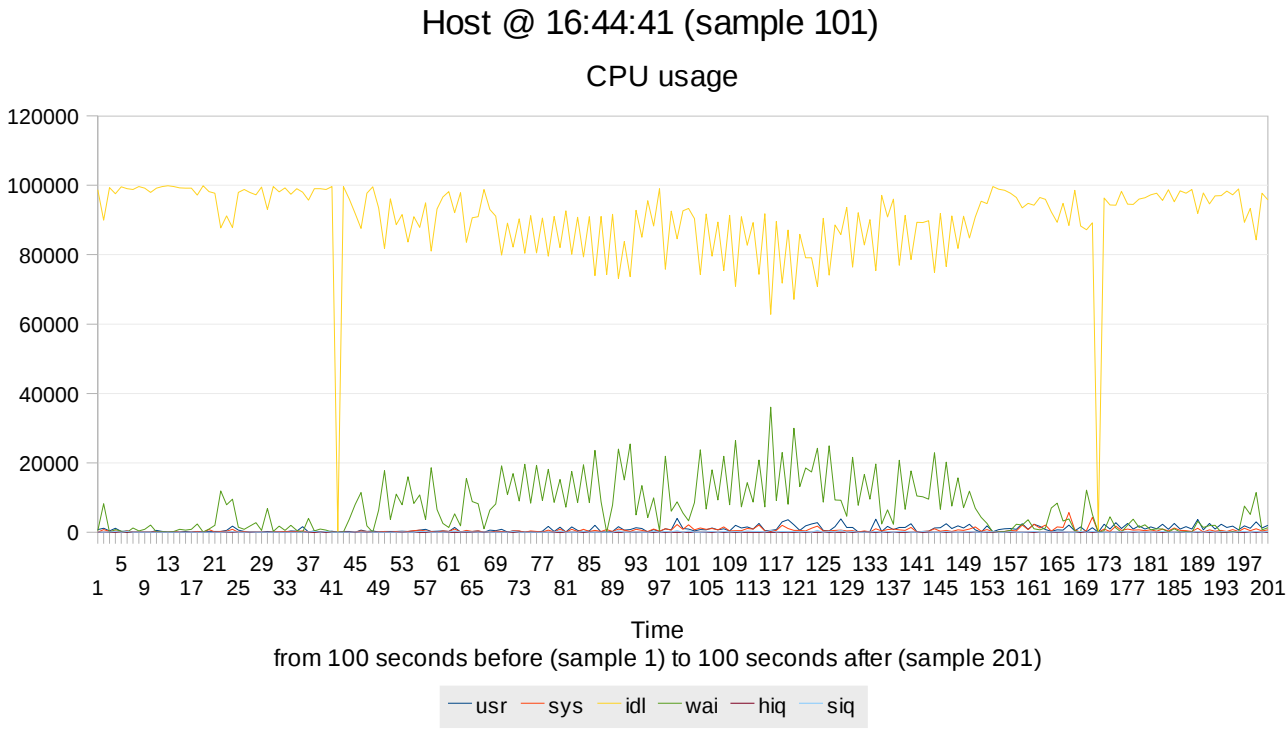


This is interesting. Network activity, after a quite regular burst (usually affordable in other situations), ceases suddenly. This is X Windows Server/Xephir network exchange not going on anymore. This confirms it is a crash due to crash of X. So the machine is up and running but not dispatching anymore.

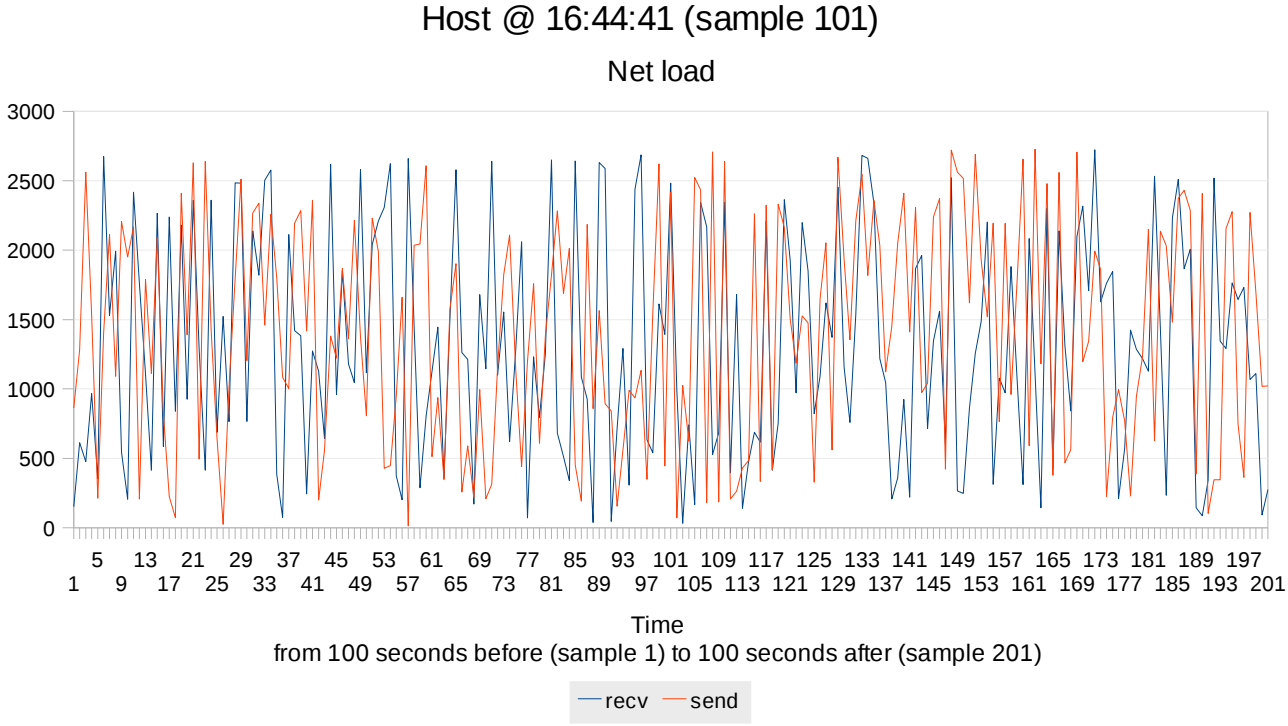


Also disk load decreases quite a bit when X crashes. The machine is still on, but not doing much (it is not dispatched, and hence not used by a human user).

Let's go to the host.



The wai burst of the host (absolutely bearable) ceases in that very moment (host is quite positively affected by one desktop crash).



Also network activity of the host decreases a bit.

This does not say much about reasons why X crashed, but we can at least exclude that they were due to perfomance (the scope of this piece of work). VDD is still unstable, but this is another issue, to be faced at development side.

4.1.5 User4 centOS5 Gnome

#	Timestamp	Evaluation	Time to complete	Notes
0		0		Really good!
00		0		
a		0	6.958	
b		0	4.991	
c		0	18.294	
d		0	14.822	
e		0	18.303	
f		0	4.814	
g		0	6.175	
9	16:20	0		
11	16:23	0		
4	16:29	0		Two more tabs present
12	16:33	0	123	
10	16:38	0	0.044	
2	16:46	-1		Slightly slow side scrolling
7	16:55	0		
3	16:57	0	42	
8	17:03	0	0.002	
5	17:06	0		Window slightly slow in response
1	17:09	0		
13	17:11	0	1.41	
6	17:15	0		

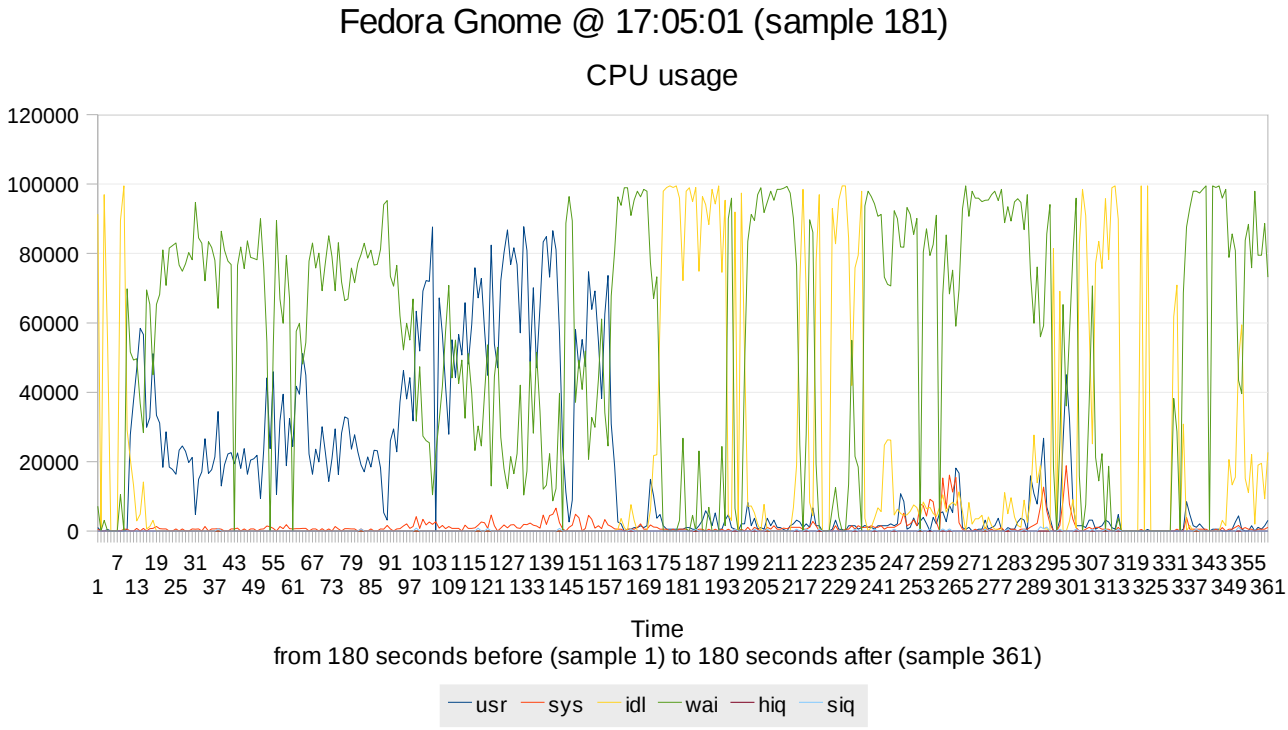
Results do not need to be commented. No critical situations were reported. Actually, CentOS desktop proved to have an excellent behaviour in all tests. The system is solid and responsive, and this include also the DE, which is not certainly the first objective of the distro. We do not report any graph, as they do not show anything special.

4.1.6 User5 Fedora12 Gnome

#	Timestamp	Evaluation	Time to complete	Notes
0		0		
00		-1		
a		0	16.153	
b		0	4.852	
c		0	8.319	
d		0	7.415	
e		0	11.278	
f		0	5.240	
g		0	6.354	
6	16:22			
9	16:26	-1		Latency in progress window
11	16:30	0		
8	16:37	0		Immediate
5	16:41	0	16 sec	
13	16:52	0	1.27	
4	16:56	0	16.56	
2	17:02	-2		No feedback
3	17:07	-1	39.139	
7	17:10	0		
1	17:13	0	41.91	
12	17:16	0	3 min	Long elaboration
10	17:20	0		

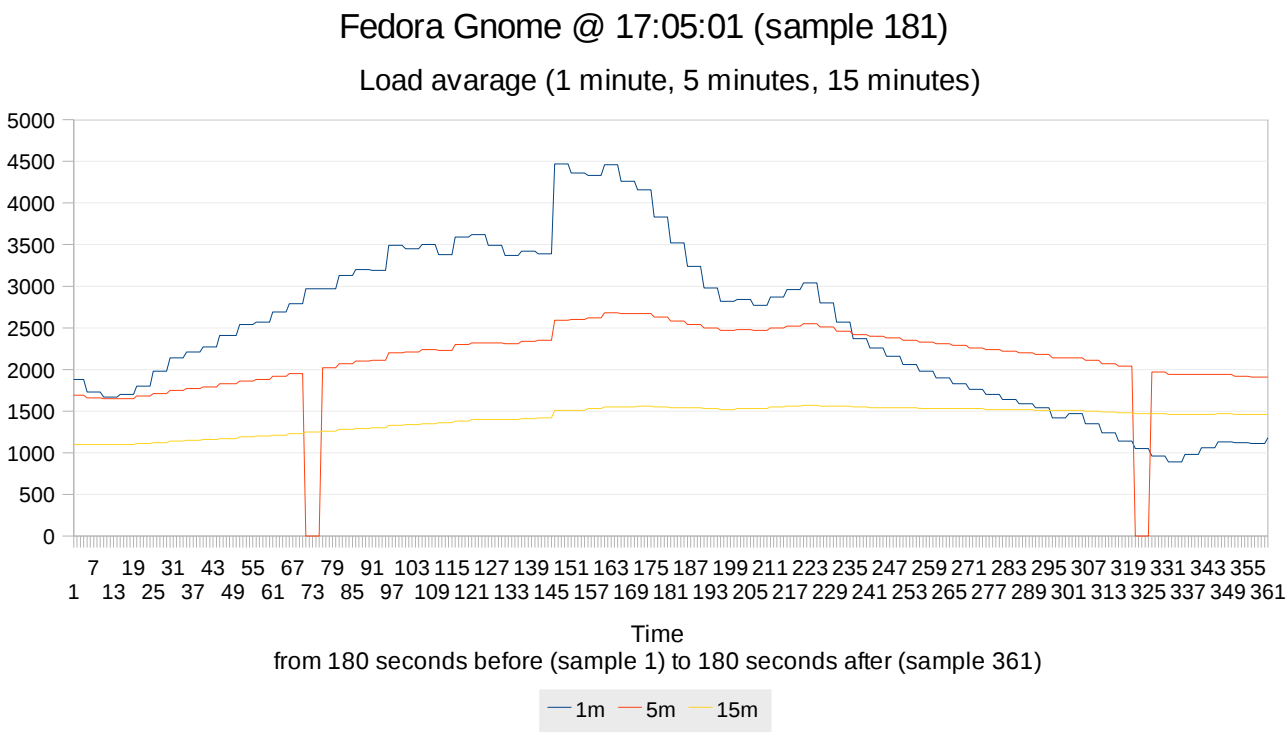
Fedora proved to be quite a good distro as well. Even more remarkable is the circumstance that user3 was contemporary using that machine with a different DE (KDE in that case, Gnome in this one). Indeed, user3 reported good results too (except for the crash of X, as we have seen). Just a note on the -1 for responsiveness of menu. We have seen this in 20% of our group tests. It does not seem a big issue and it is normally observed only the first time, before caching enters into action, both for GNU/Linux systems and for Microsoft Windows systems. It occasionally happens also on local Desktop, so we do not worry much about this, meaning that we will not increment VDD machines resources to cope with it and we will rather do it for other issues.

User5 signals something wrong happening around 17:05 (17:02/17:07). Let us see if there is any performance related problem: =SYS_GRAPHS("Fedora Gnome";6100;180;180).

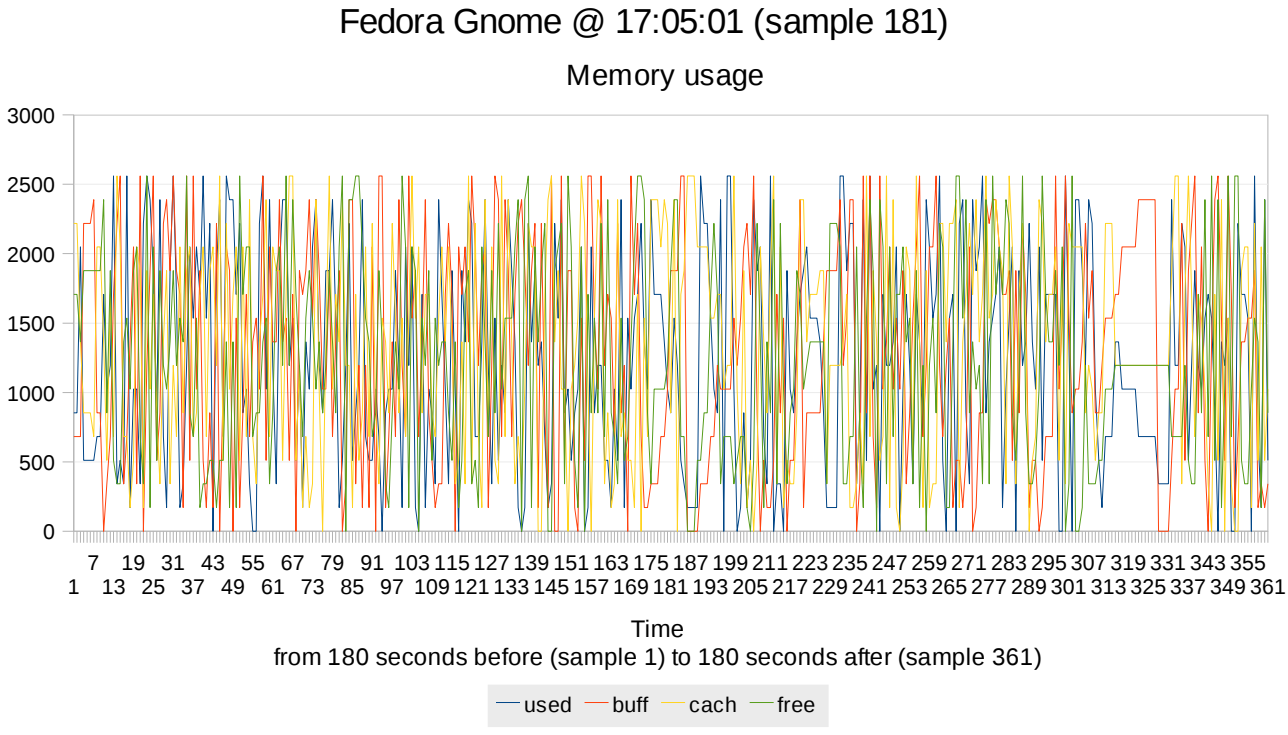


From 17:02 to 17:04 there is a remarkable CPU-bound activity at user level. This is due to user3, that is doing in the meanwhile mp3 conversion followed by GIMP filtering. User5 tries to make a sum in OpenOffice Calc and apperently he or she does not succeed. It does not seem to be related to system, though. CPU-bound activity is not so incredible to giustify Calc refusing a simple sum operation. There had to be other problems related to application, instead.

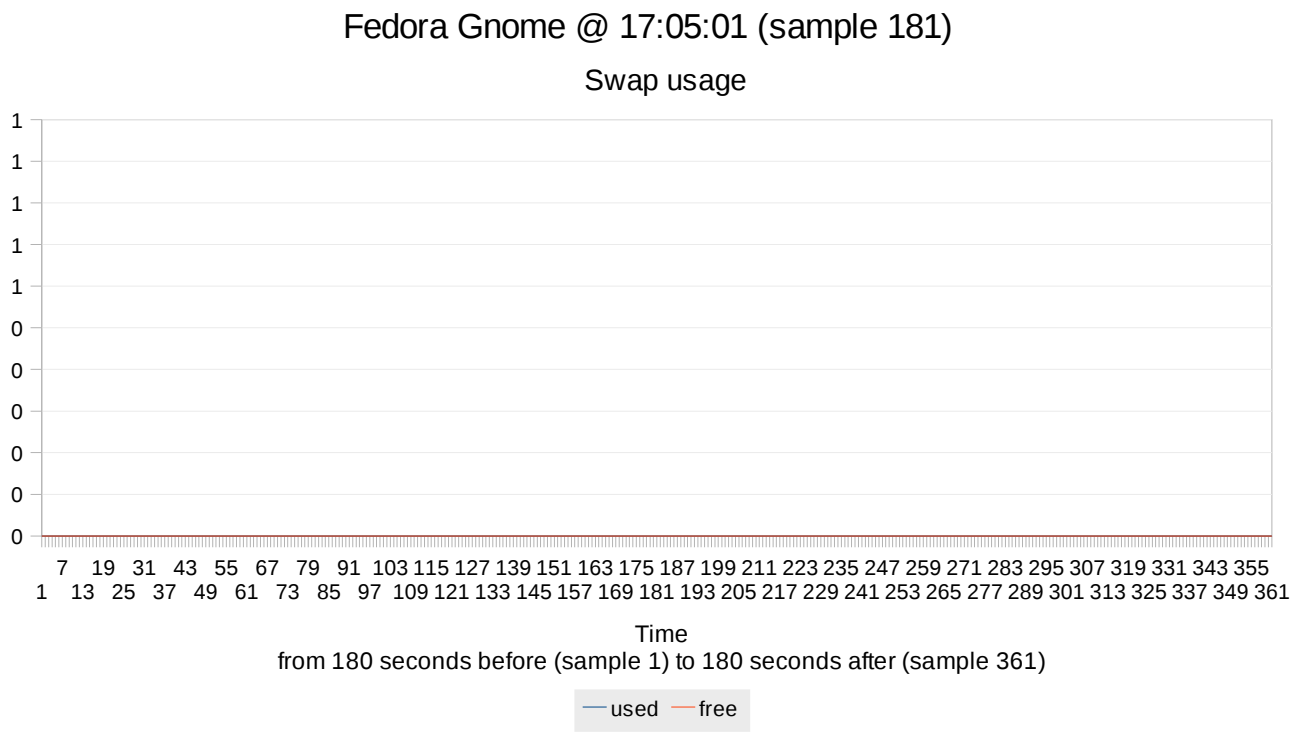
In the second part of the observed window (when user5 performs operation numer 3 and reports some slow downs), only some wai bursts can be observed, as for now.



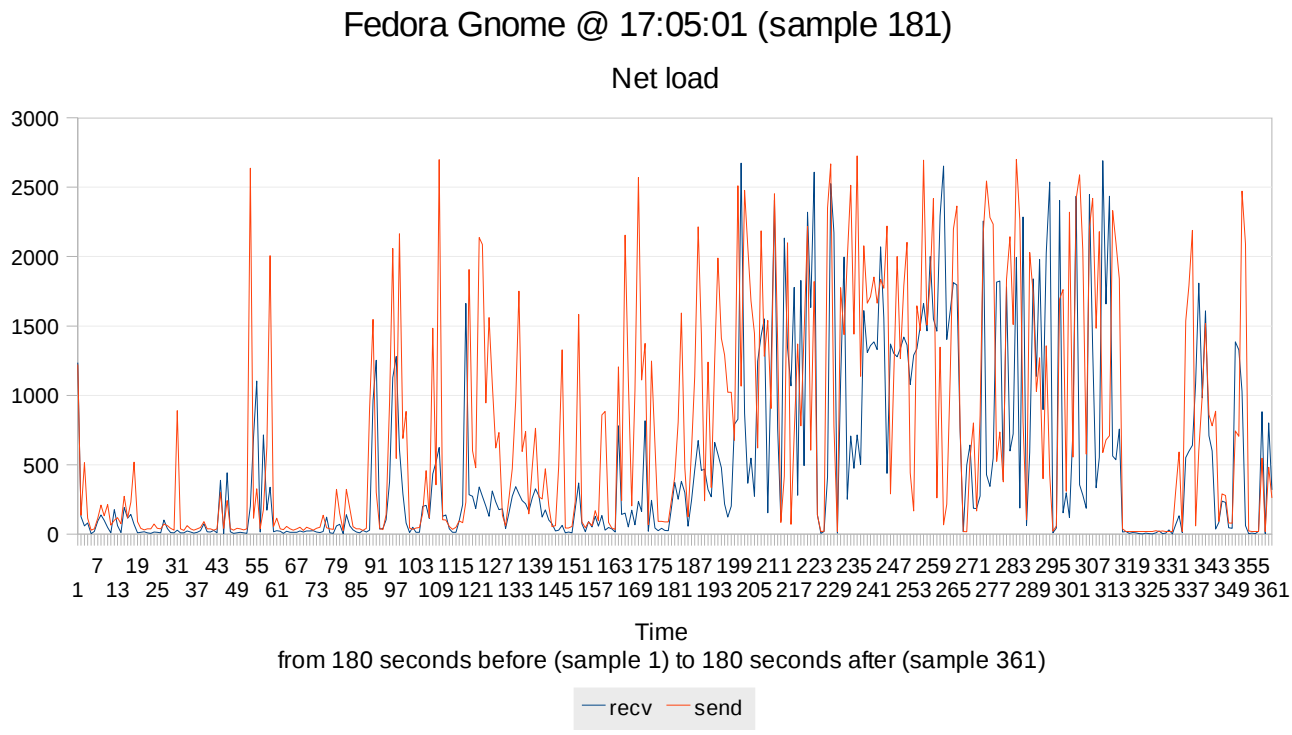
Load avarage has got a peak at 17:04, due to CPU activity. As a matter of fact, it decreases after, so I/O activity does not influence load that much.



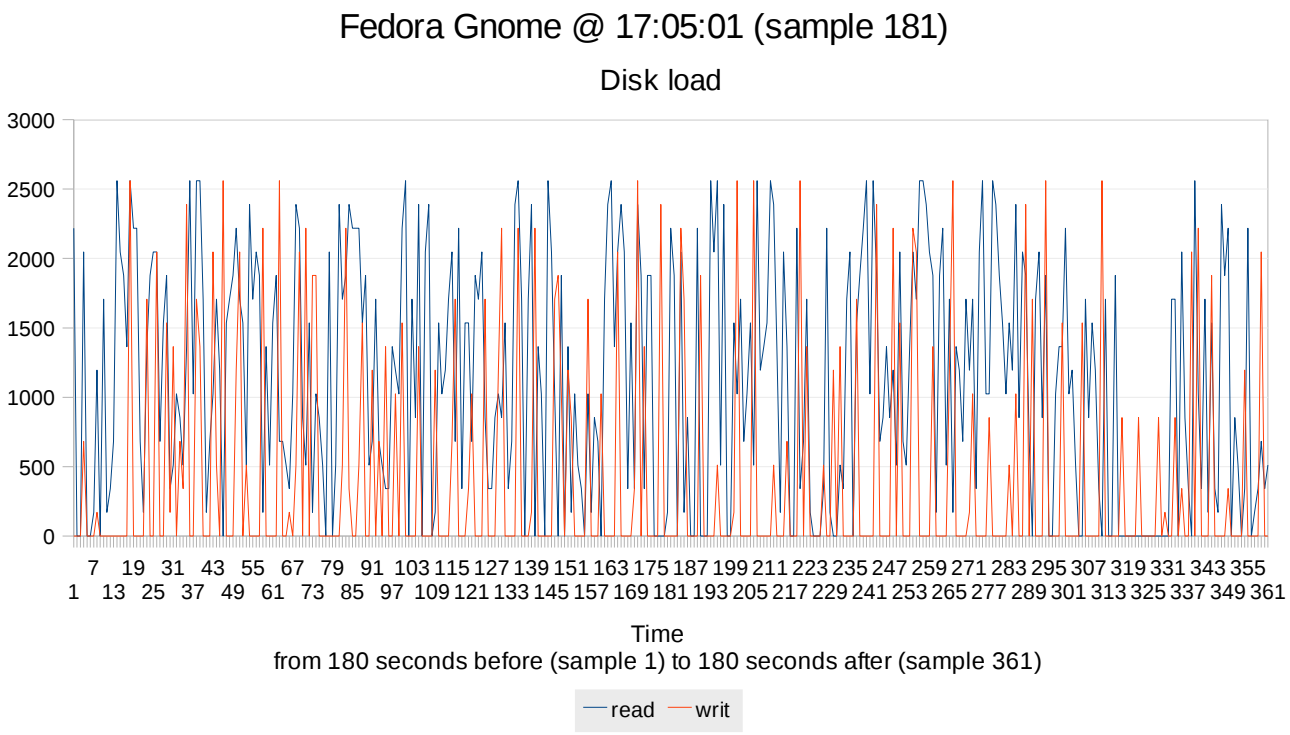
Memory usage is constant and regular under all the period observed, with a short pause at the end.



Swap is not a problem, indeed.



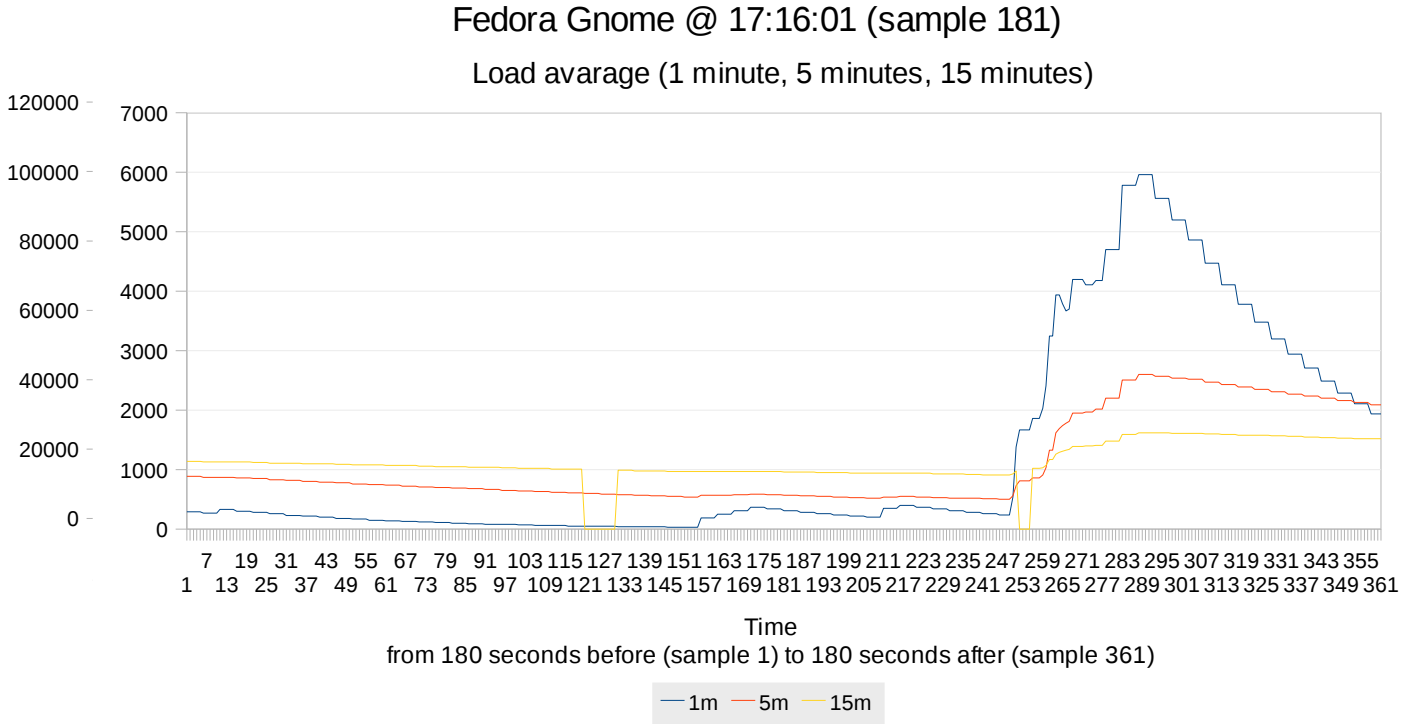
There is an interesting peak of network in the second part of the observation window, when OpenOffice Impress is showing a presentation. Just some more graphics to receive on the cable, but nothing special.



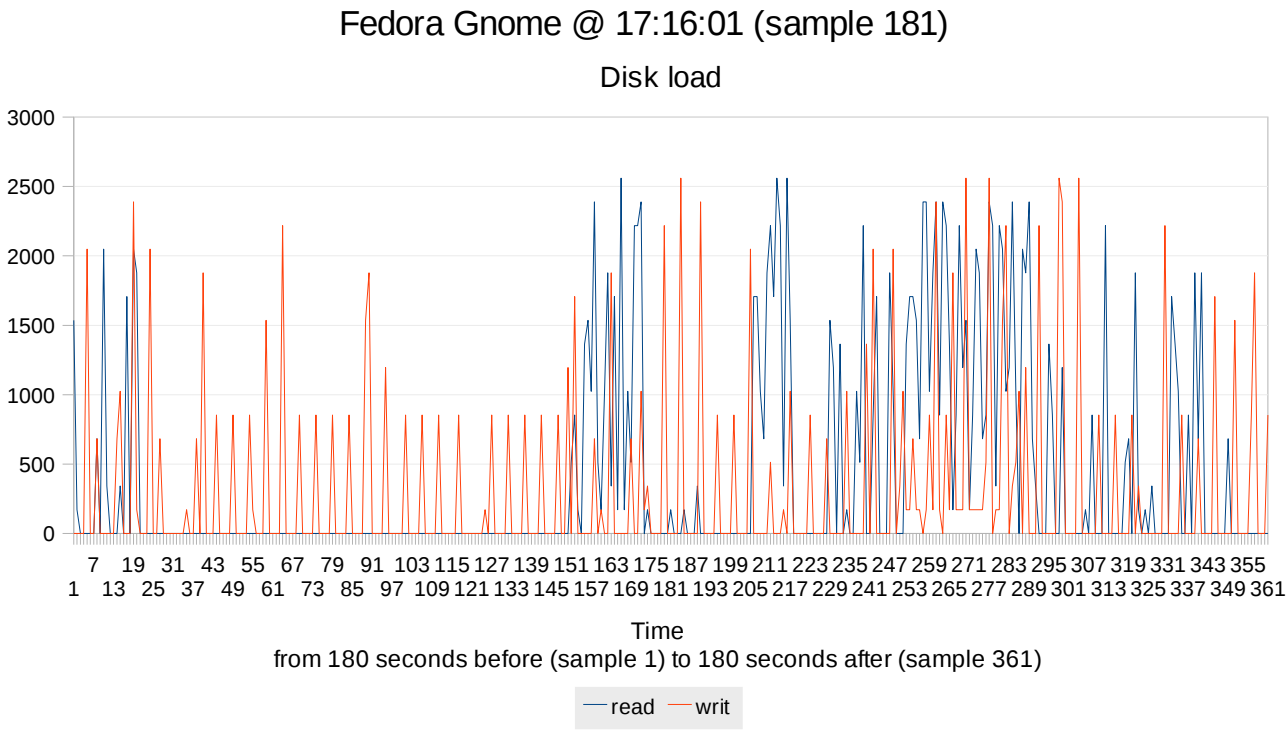
Disk load is definitely not a problem during all period and it even decreases in the end.

So we can conclude, even without analyzing the host, that performance were quite right at the time when -2 was rated by the tester, almost certainly due to application related problems, rather than system problems.

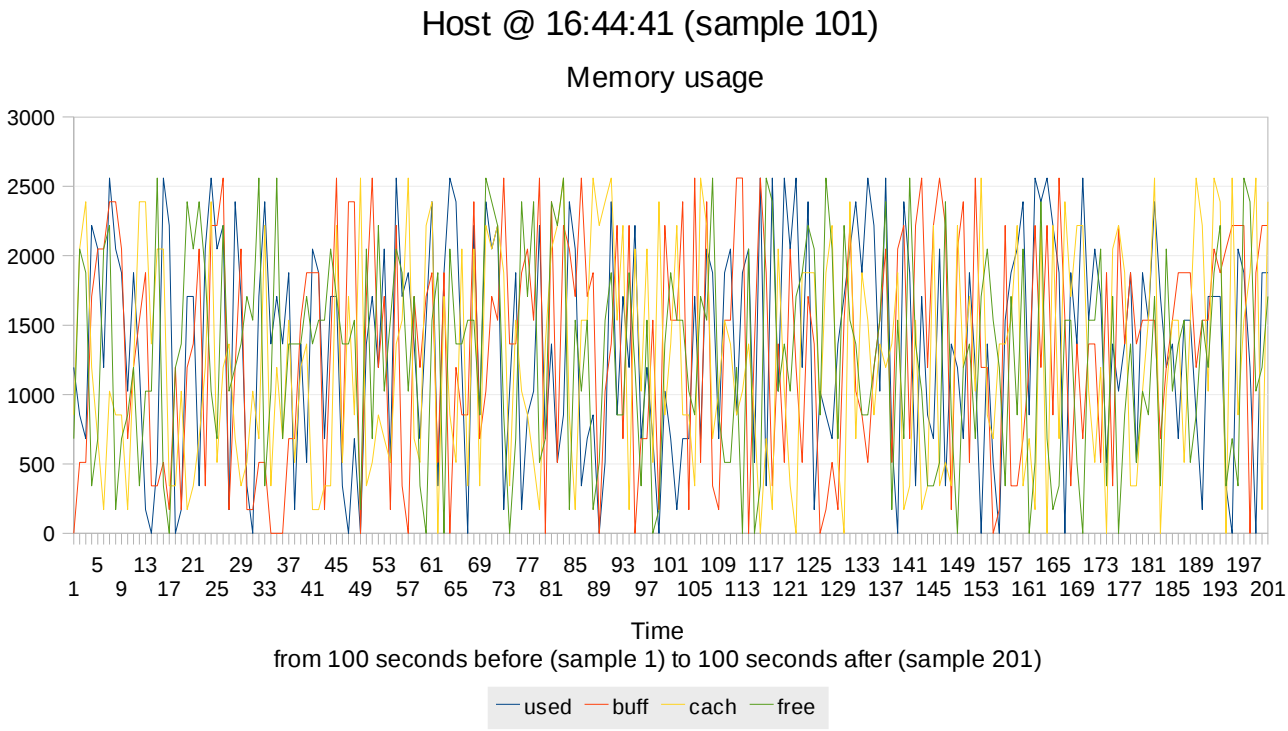
Another interesting note of user5 is about long elaboration period for the GIMP filter. Let's see.



Load raises quite a lot in that point.



And also disk load has got a burst.



On the host everything is ok, instead, and memory usage is quite right during the period, decreasing in the end.

So we conclude that Fedora system is a very good system, not presenting critical issues in our tests. If we really want to find a flaw in elaboration time, we have to consider that Fedora is intended for desktop use, so developers have probably issued a system that works fine with small and medium files and have longer CPU elaborations with respect to I/O, which is absolutely reasonable and correct.

4.1.7 User6 Gentoo10 KDE

#	Timestamp	Evaluation	Time to complete	Notes
0		0		
00		0		
a		0	14.5	
b		0	3.2	
c		0	5.6	
d		0	11.6	
e		0	10.5	
f		0	3.9	
g		0	4.7	Using evince instead of acroread
4	16:17:04	0		
7	16:17:50	0		
2	16:20:30	0		
1	16:22:44	0		Log out and log in again
5	17:06:21			Don't open with GIMP
8	17:11:05	0	mv 0.02 cp 0.28	
10	17:13:45	0	mv 0.02 cp 3.73	
12	-	-2		Don't open with GIMP
11	17:15:58	0		
6	17:17:20	-2		No audio
9	17:18:35	0	mv 0.048 cp 0.0832	
3	17:20:31	0		
13	17:21:25	0		

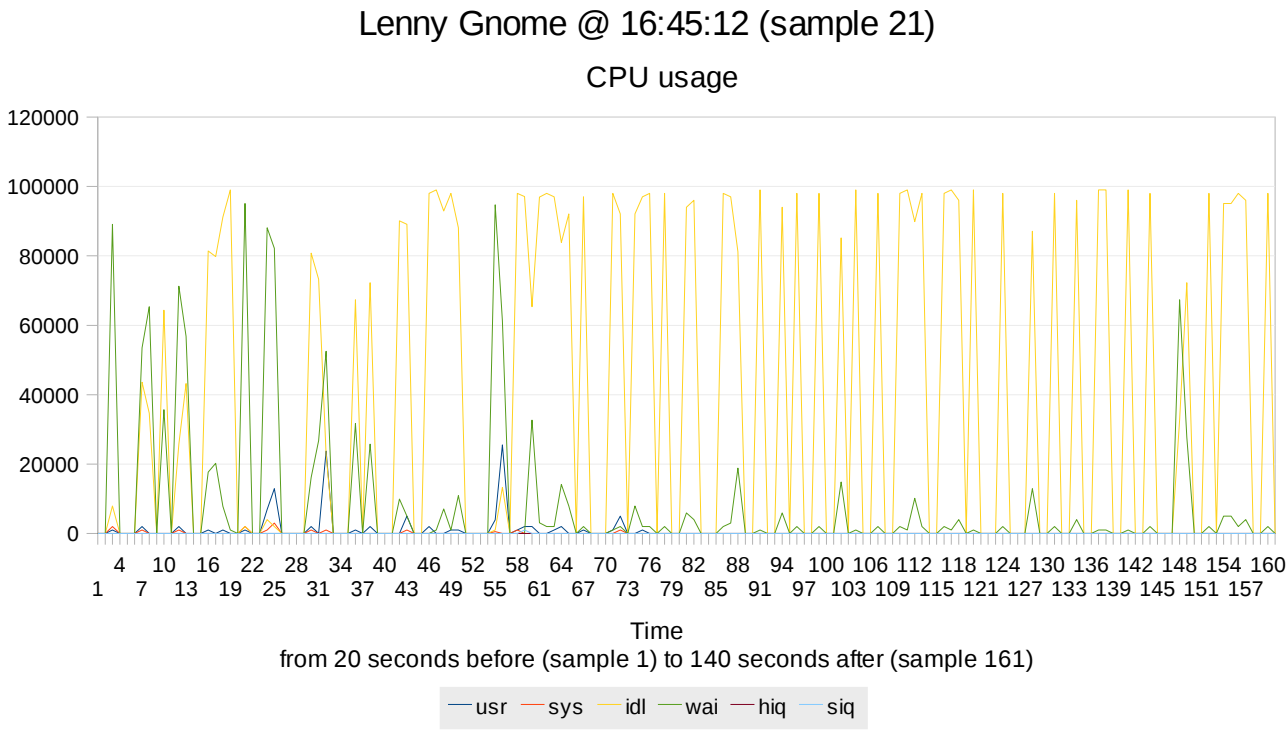
Gentoo with KDE scores great. The only problems reported regard the absence of audio output (this is a known issue of VDD) and the impossibility of using GIMP (application

problem). Graphs can not say much about these problems, so we do not report them. There was a problem between operation 1 and 5 (see the time). User had to log in back in the system. We do not know what happened. Must be addressed as a usability problem (see usability report).

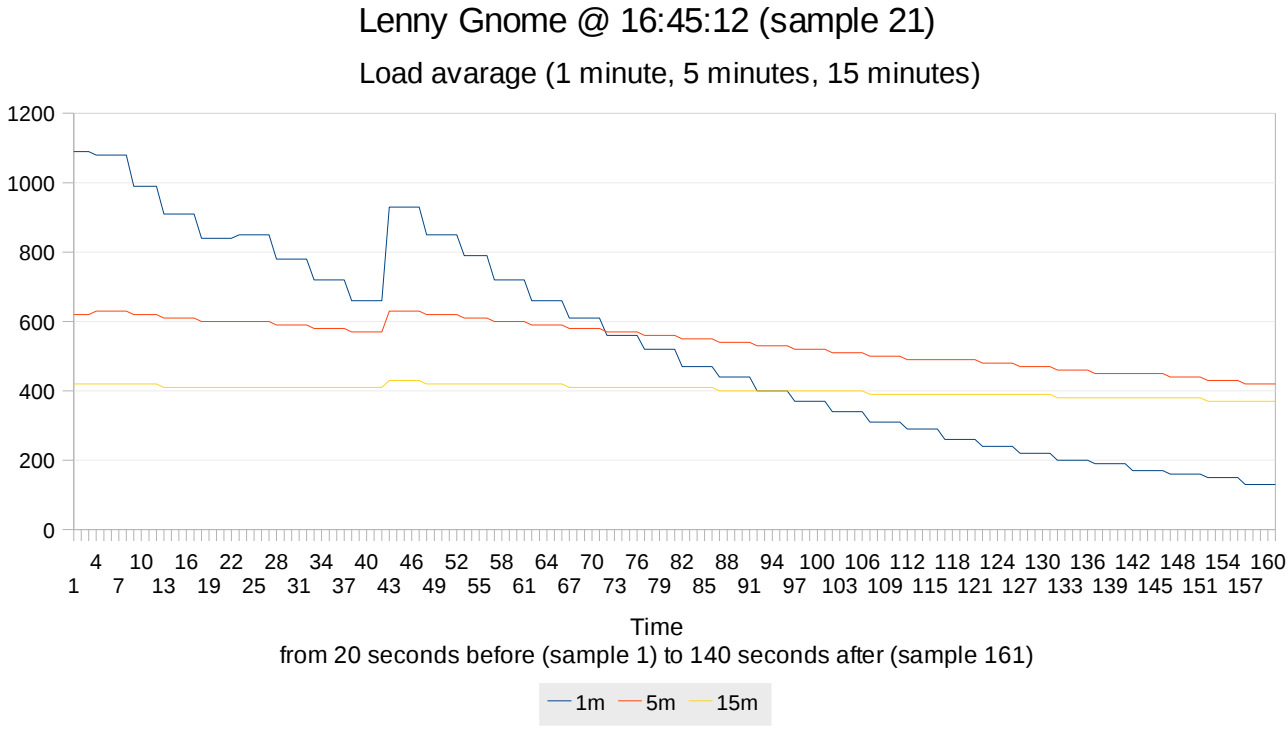
4.1.8 User7 Lenny Gnome

#	Timestamp	Evaluation	Time to complete	Notes
0		0		
00		0		
a			9.7	
b			0.5	
c			0.16	
d			9.8	
e			9.3	
f			6.1	
g			15.3	
13	16:28:50	0		
6	16:32:12	-2		No audio
3	16:33:00	0		
12	16:34:20	0		
4	16:36:13	0		
8	16:37:20	0	cp 0.03 mv 0.037	
9	16:37:55	0	cp 0.02 mv 0.002	
7	16:41:03	0		
5	16:41:48	0		
2	16:42:26	0		
11	16:43:24	0		
1	16:45:11	-1		
10	16:47:20	0/-1	mv 0.077 cp 13.27	

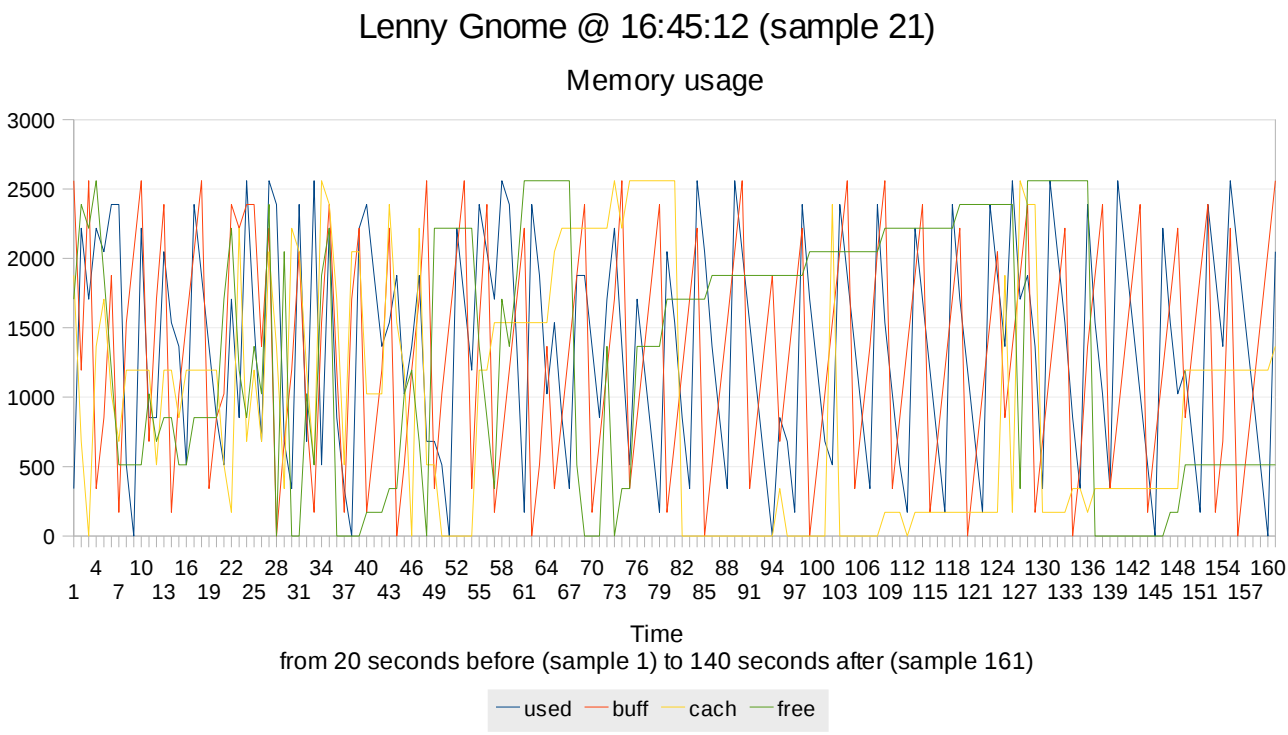
Yet another great result. Apart the -2 for lack of audio (known VDD issue), only some minor slow downs are reported at the end of the test. We go and have a quick look at that (=SYS_GRAPHS("Lenny Gnome";4864;20;140)) more for seeing the desktop at work, than for criticizing it.



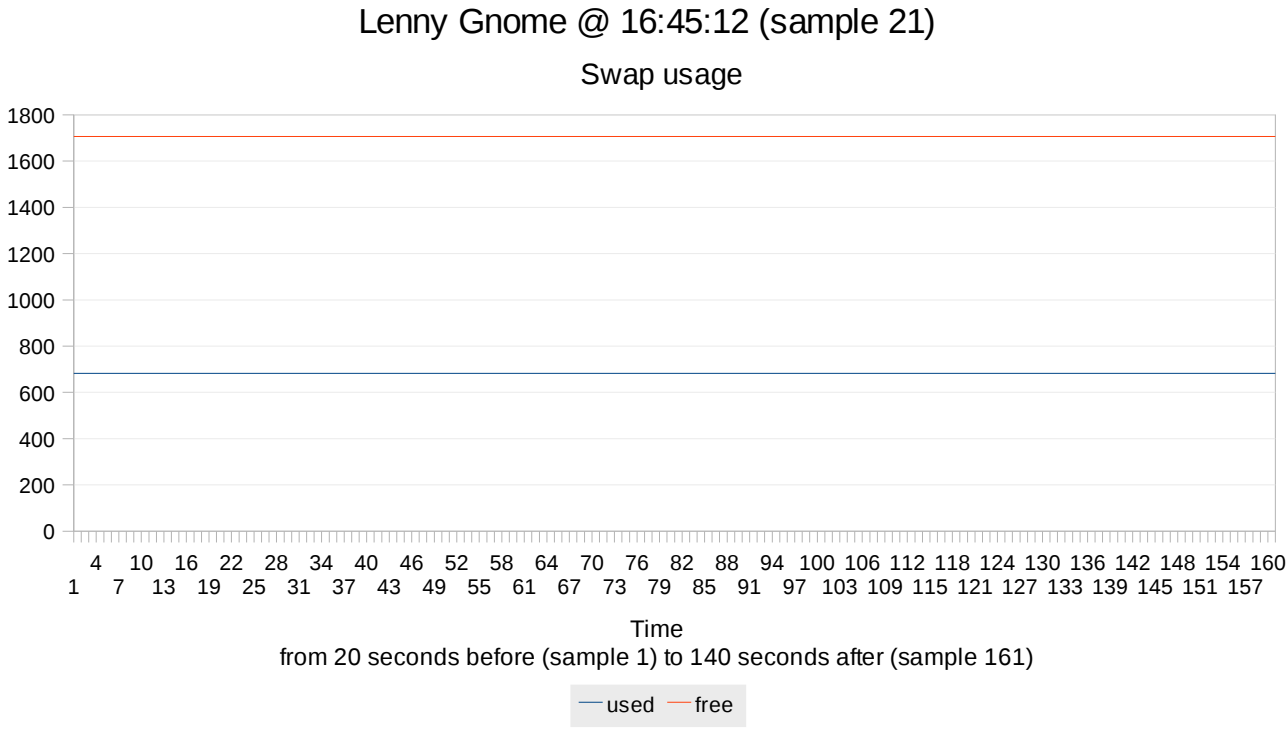
Really good profile. Everything seems to be managed at best.



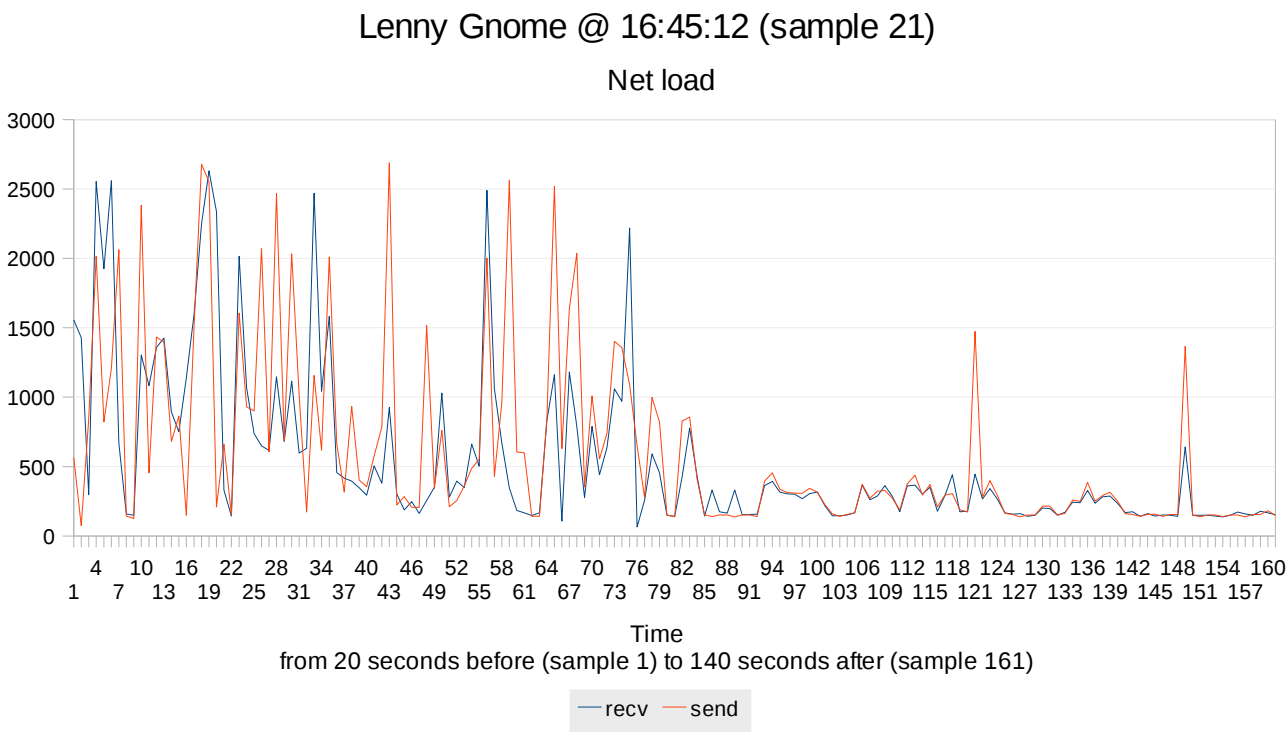
No load problems at all. Really low values.



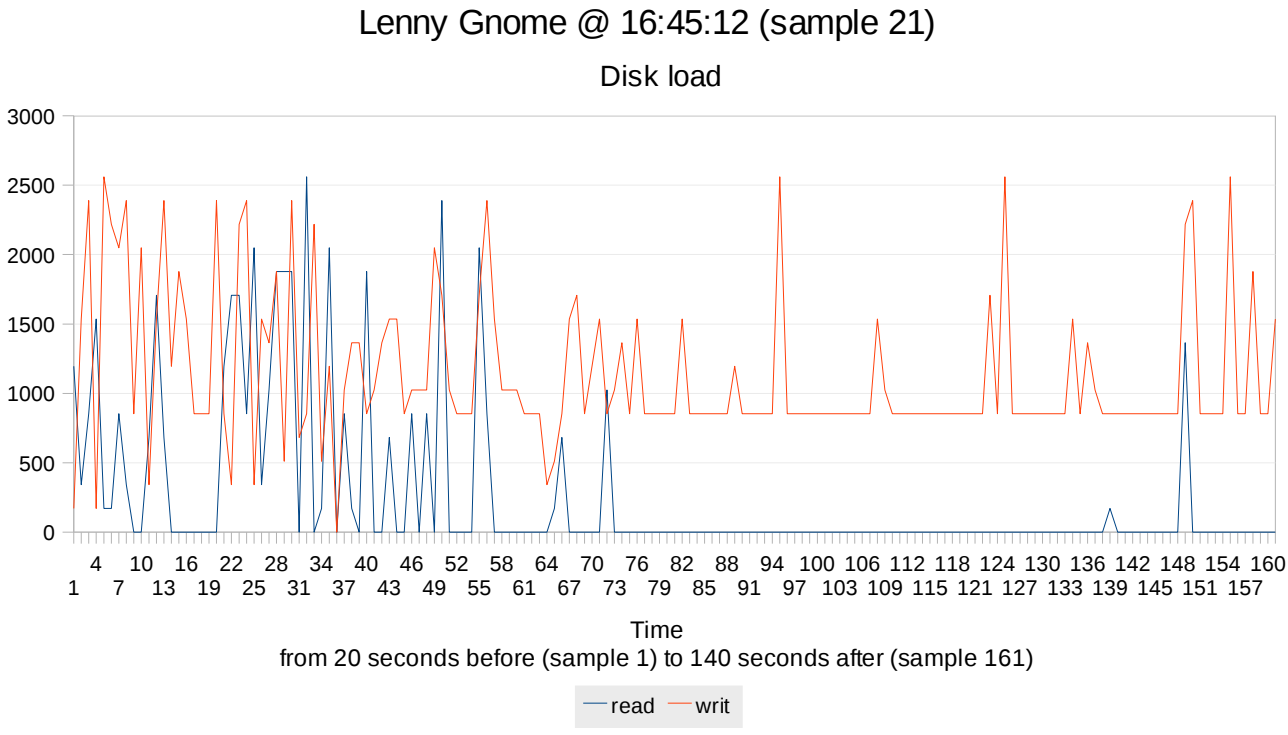
Memory profile is really fantastic! Not only the use of memory is rational, but also profile dynamics indicate a linear and fluid management.



Some swapping is going on. This is due to how debian systems work (see note in 4.1.1).



Neat and low. Debian performs great with networks.

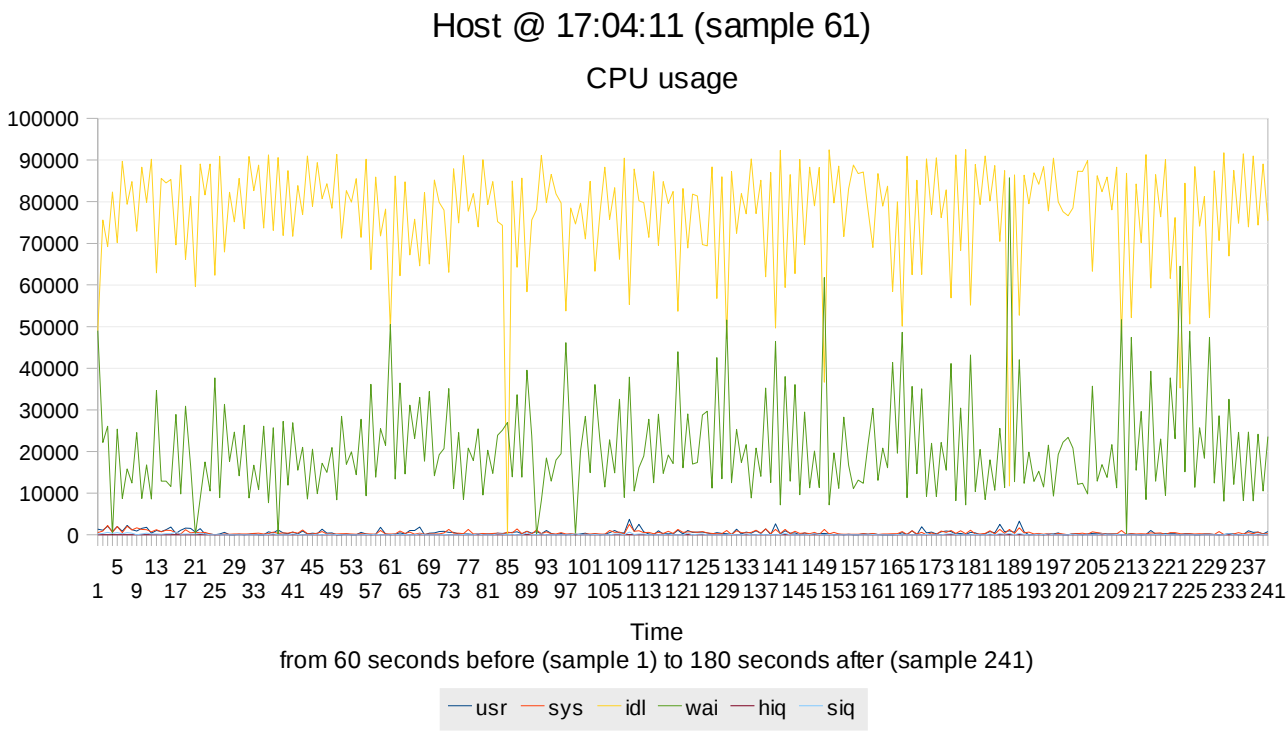


The lower we have seen in all tests. If this is -1, imagine the rest...

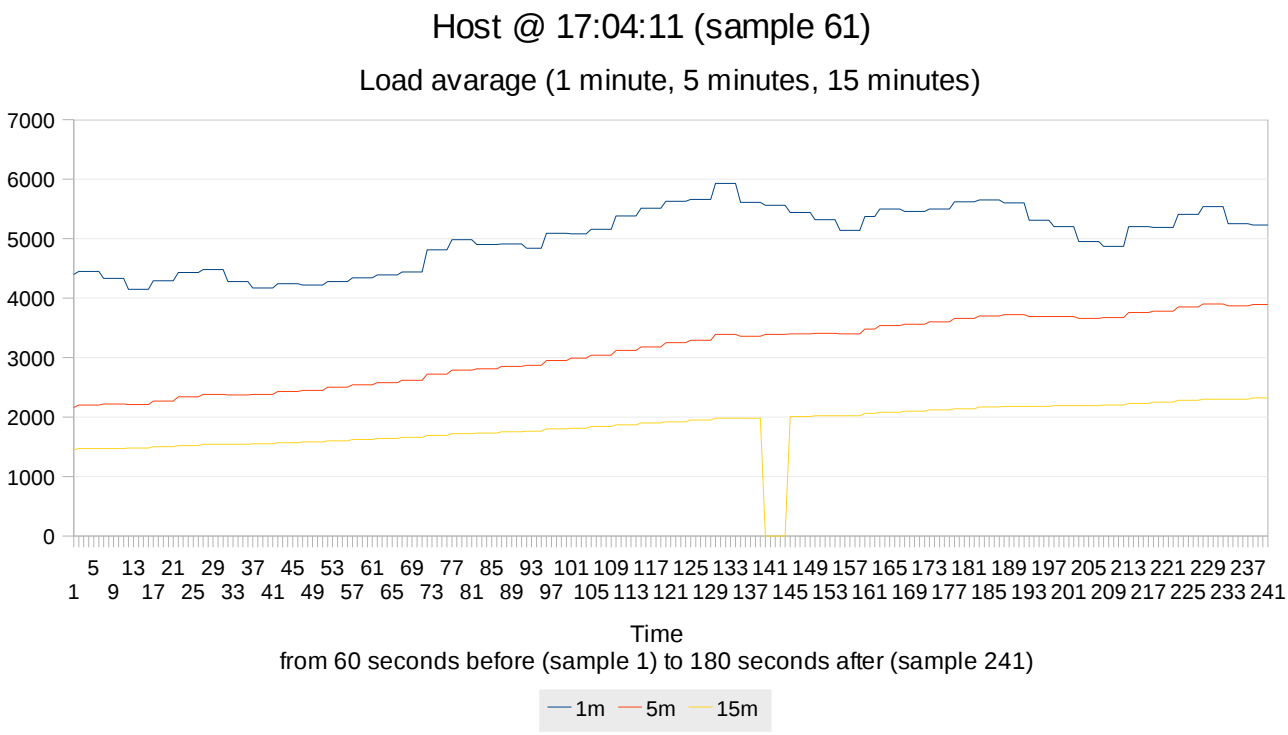
4.1.9 User8 Windows XP

#	Timestamp	Evaluation	Time to complete	Notes
0		0		
00		0		
a		0	5	
b		0	3	
c		0	3	
d		0	2	
e		0	3	
f		0	2	
g		0	3	
1	16:11:00	0	19	
6	16:22:00	0	18	
8	16:25:05	0	2	
7	16:32:20	0	7	
4	16:37:30	0	15	
2	16:40:00	0	17	
11	16:46:00	0	25	
9	16:48:00	0	6	
12	16:53:10	0	57	
10	16:56:45	0	32	
5	17:00:00	0	6	
13	17:04:10	-2	213	
3	17:09:10	-1	15	

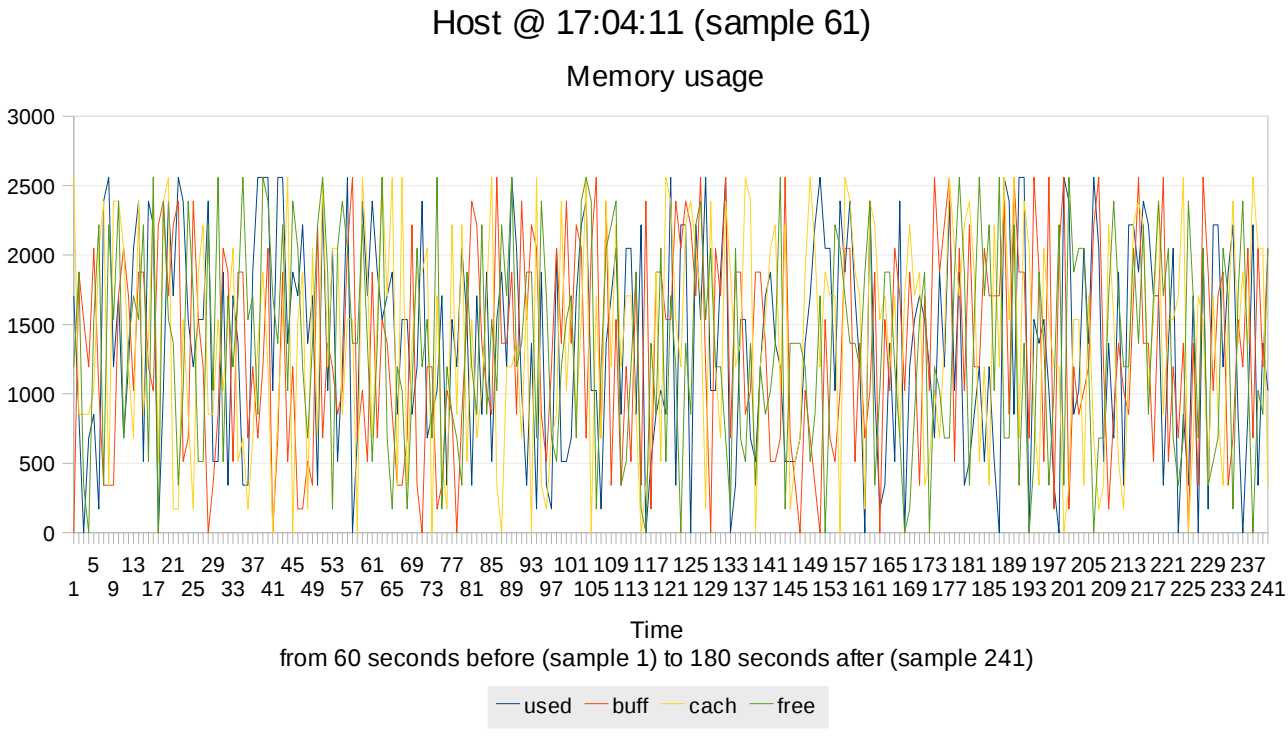
Great results, nothing to say. We go and see if host is affected at all around 17:04:10 and later. Windows does not have a valid equivalent of dstat, unfortunately. Even if dstat is written in python, it can not load *resource* module, as dstat calls operating system primitives to obtain statistics. Windows kernel has got different routines. Anyway, we still have dstat host statistics: =SYS_GRAPHS("Windows XP";5836;60;180).



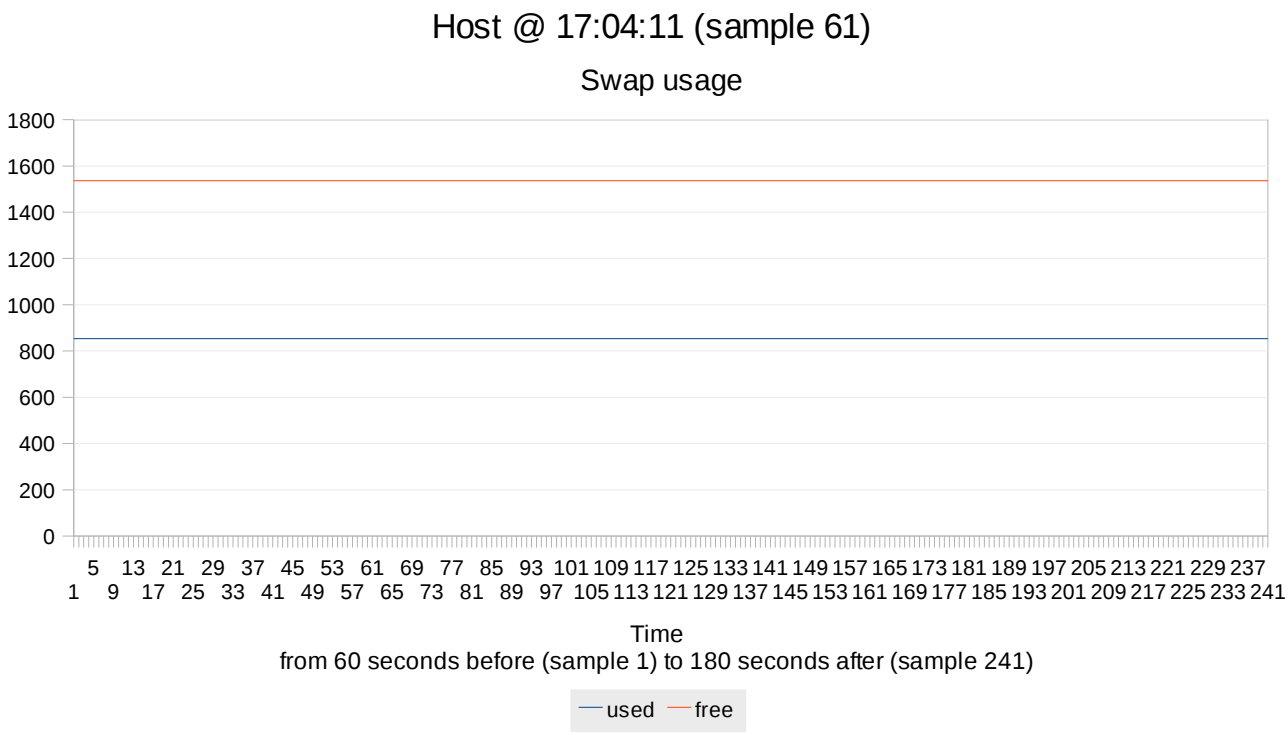
Intense I/O waiting for the CPU. Big I/O is going on, but still the system catchs up.



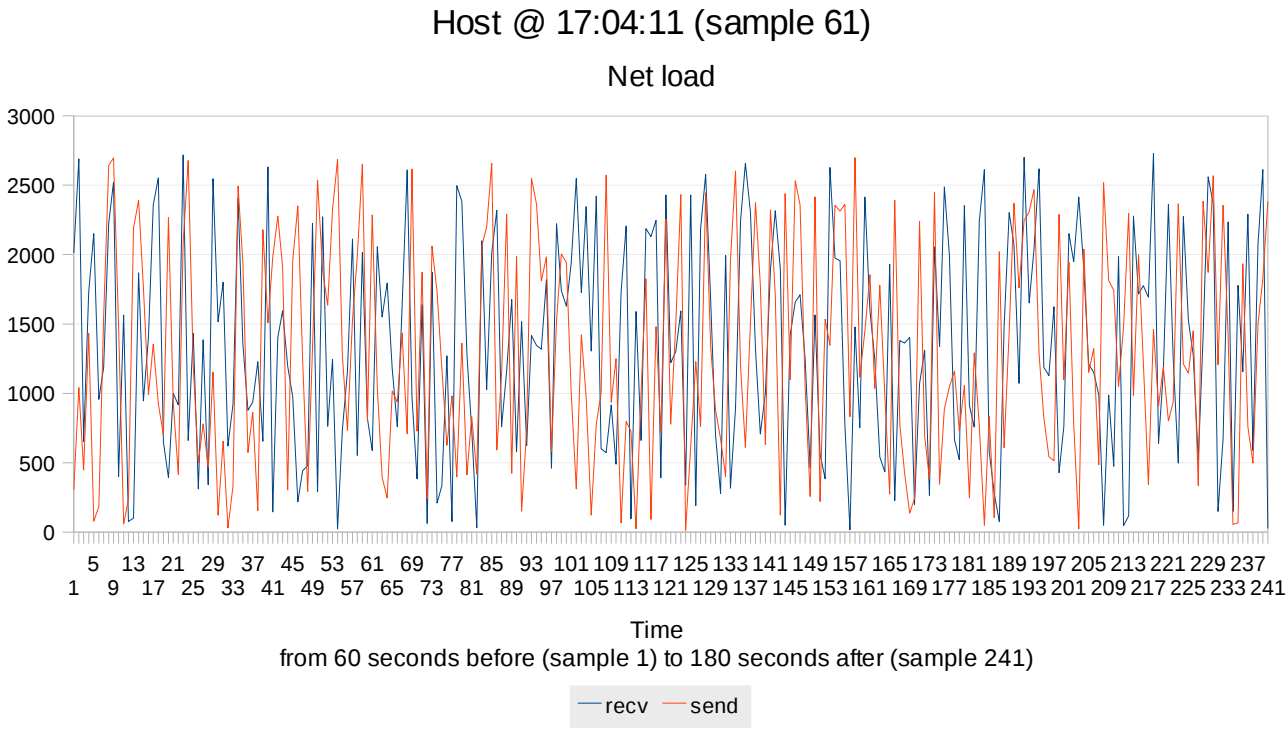
Load is quite higher than normal. Profile is almost flat and system still behaves well, but the absolute value is something.



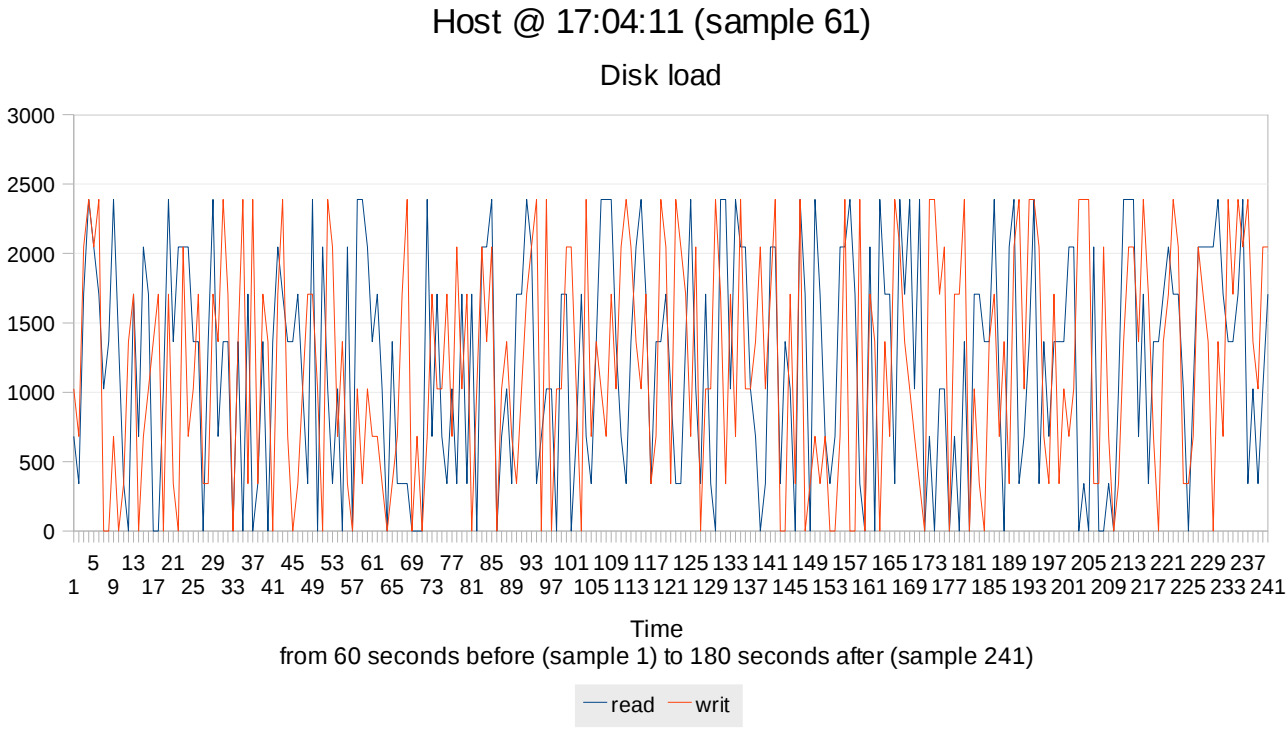
No problems with memory.



And the level of swap stared at the usual.



Constant and quite regular.



As for the network, it goes on linearly.

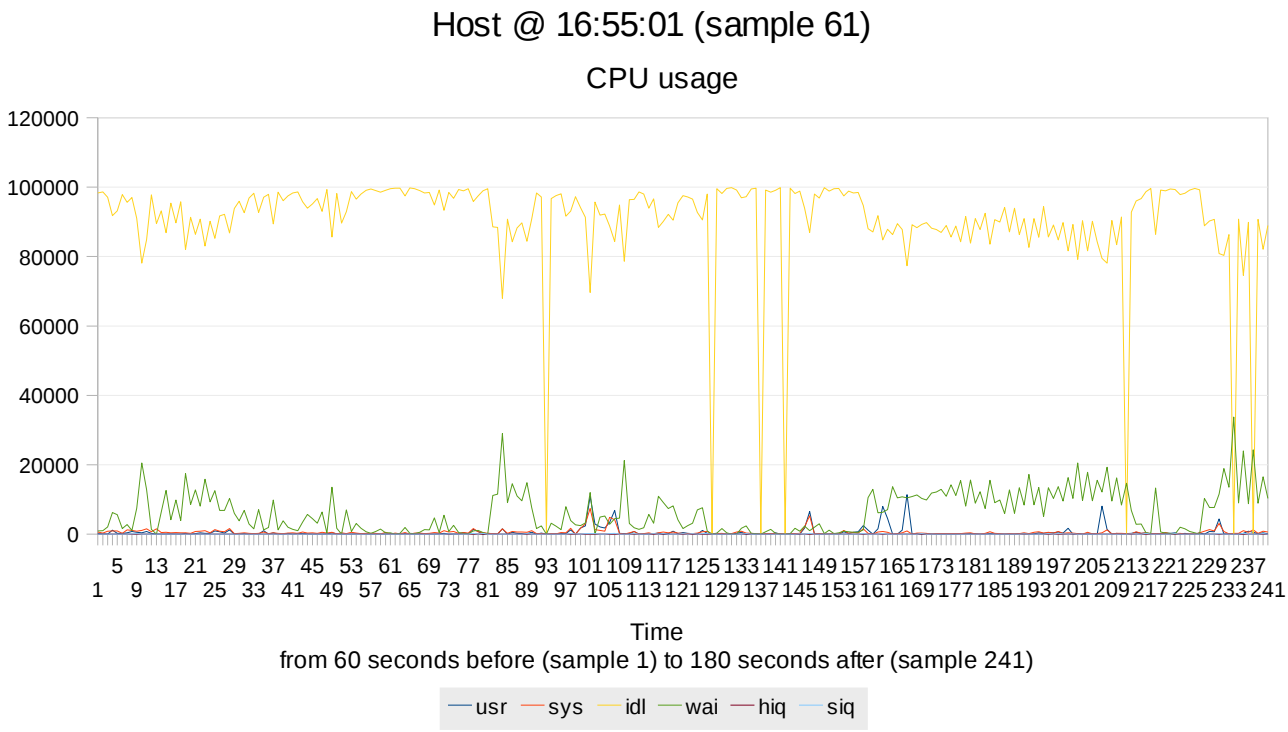
So, we can say that the system is doing fine, even if it is quite under stress. Long elaboration probably due to Windows XP system (i.e. obvoiusly optimized for a desktop use).

4.1.10 User9 Windows 7

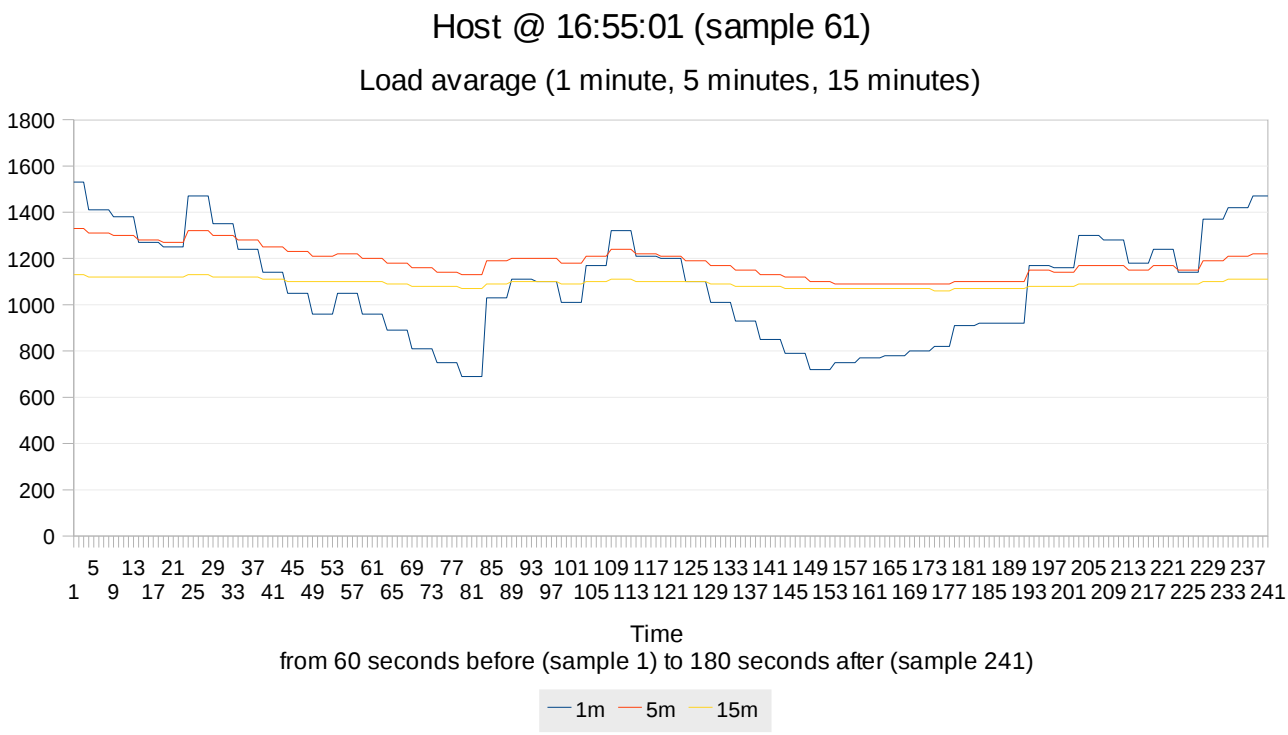
#	Timestamp	Evaluation	Time to complete	Notes
0		0		
00		-1		
a		0	2	
b		0	1	
c		0	1	
d		0	3	
e		0	4	
f		0	1	
g		0	3	
4	16:20:25	0	11	
9	16:24:00	0	2	
13	16:28:10	0	71	

2	16:35:00	0	17	
5	16:39:00	0	16	
1	16:41:10	0	14	
11	16:45:10	0	23	
7	16:47:25	0	6	
6	16:50:30	-2	5	Mplayer does not work (using winmp)
12	16:55:00	-1	55	
3	16:58:10	0	5	
10	17:01:18	-1	42	
8	17:08:10	0	4	

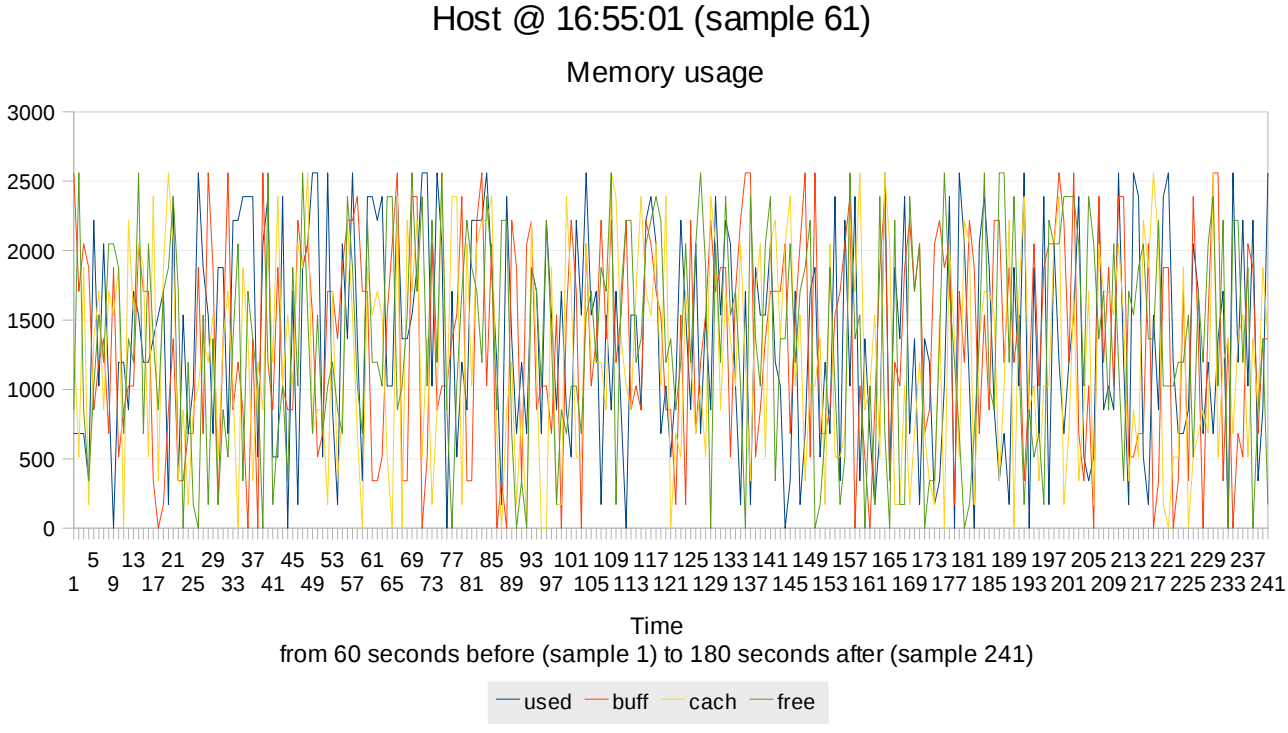
Another good result, slightly under XP. Let's see if the host is doing better or worst in the meanwhile: =SYS_GRAPHS("Windows 7";5440;60;180).



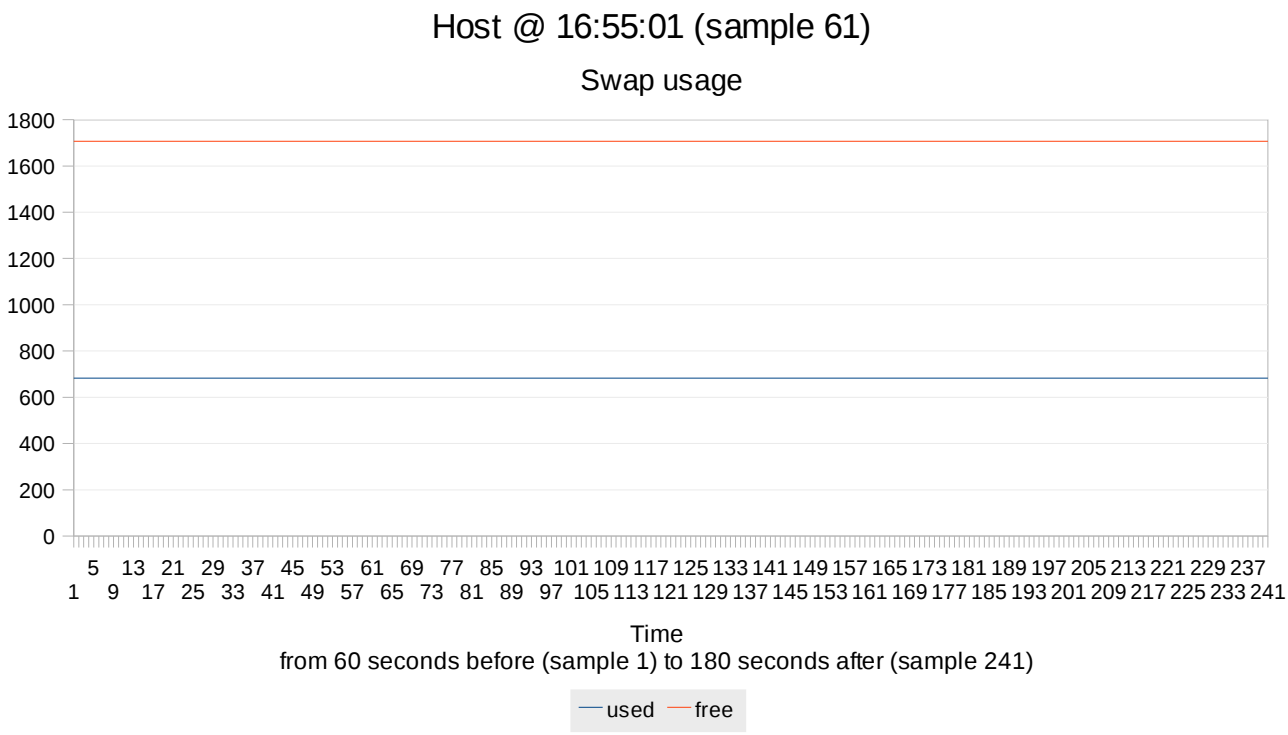
Values are absolutely under control. Much less than for Windows XP case.



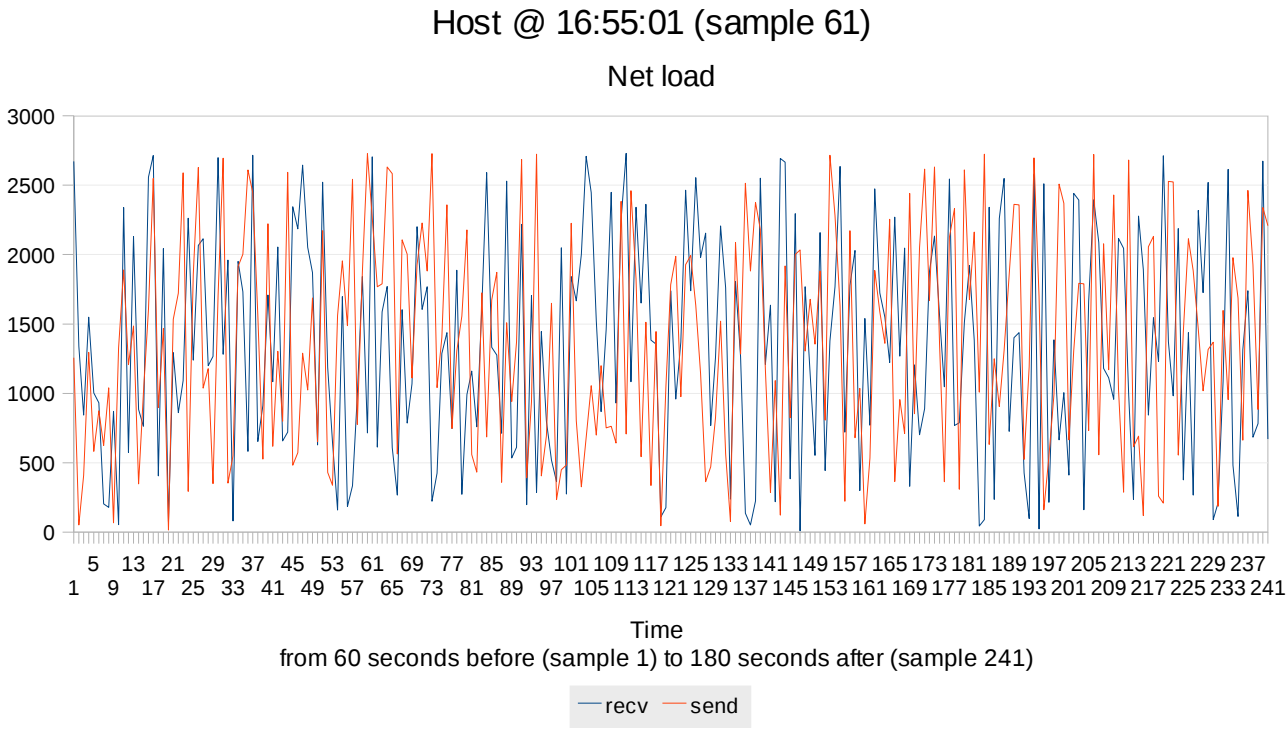
Load is low as well. A lot lower.



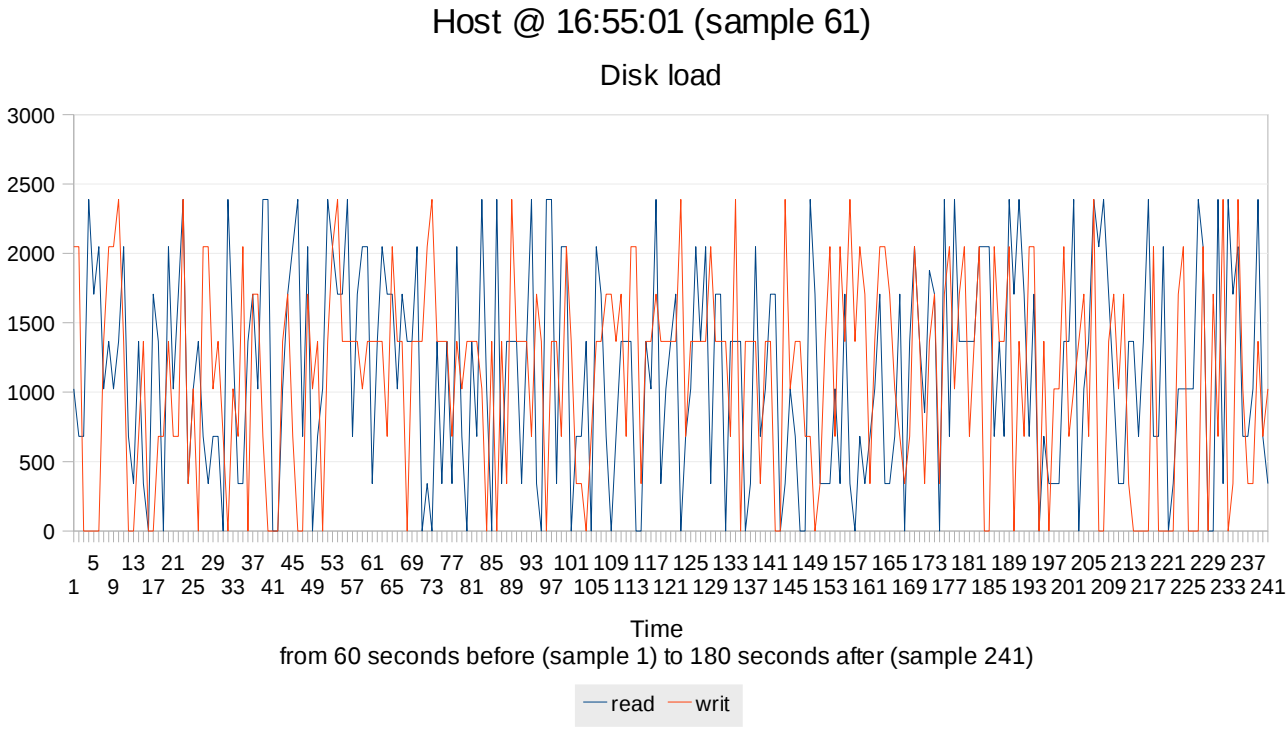
Memory is more or less the same.



Swap is lower.



Net load more or less the same, or just a bit more.



Disk slightly better than before.

This means that even if Windows 7 has got good results, as good as Windows XP, it scores so in quite better underneath conditions than before, so Windows 7 starts having problems (the one signaled by user9) much before than Windows XP would.

4.2 Second battery

With this battery we intended to stress one single desktop from 10 terminals. We repeated the same tests, in a random order for each user, and without taking care of caching distortions (irrelevant in a random situation). As we started tests, we immediately understood that it was not impossible to continue in that situation. Performances were really poor and systems resulted not usable. Ten terminals on the same machine was clearly too much. So we started looking for a threshold, by dispatching from the same virtual machine to more and more terminals, starting from two and adding one each time. We found that 5 was the limit for performing the tests in a reasonable way.

We then noticed that memory was scarse and the virtual machine we were on was swapping a lot. We tried different desktops and found the same situation for all of them. We then started to assign more memory to the desktop to be dispatched and we found that granting 4 times as much (despite of users being 5) the system started to work absolutely fine. We then started

the test procedure and we obtained the following results. They are so brilliant, that graphs do not say anything interesting, so we do not report them.

4.2.1 User0 CentOS KDE

#	Timestamp	Evaluation	Time to complete	Notes
00	17:00	0		
1	17:02	0		
10	17:05	0	0.02	
7	17:09	0		
9	17:12	0	0.01	
8	17:13	0	0.01	
13	17:15	0		
2	17:19	0		
4	17:26	0		
6	-			
3	17:31	0		
12	17:36	0		
11	17:39	0		
5	17:40	0		

4.2.2 User1 CentOS KDE

#	Timestamp	Evaluation	Time to complete	Notes
00	17:00	0		
5	17:03	0		
2	17:11	0		
13	17:18	0	1.46	
8	17:21	0	0.01	
1	17:22	0		
3	17:23	0		
12	17:26	0		
6	-	-		
7	17:31	0		
10	17:32	0	0.24	

4	17:33	0		
11	17:34	0		
9	17:35	0	0.01	

4.2.3 User5 CentOS KDE

#	Timestamp	Evaluation	Time to complete	Notes
00	17:00	0		
6	17:03	-	-	Error in opening file
9	17:05	0	0.003	
11	17:07	0		
8	17:10	0		
5	17:11	0		
13	17:12	-1	3.356	
4	17:17	0	5.786	
2	17:20	0		
3	17:23	0	20.030	Rapid and fluid
7	17:25	0	7.39	
1	17:27	0	13.801	
12	17:31	-1	29.904	
10	17:34	0	0.003	

4.2.4 User8 CentOS KDE

#	Timestamp	Evaluation	Time to complete	Notes
00	17:00	0	-	
1	17:03	0	13.289	
6	17:06	0	6.845	
8	17:11	0	0.028	
7	17:21	0	5.870	
4	17:25	0	5.443	
2	17:27	0	12.335	
11	17:32	0	24.452	
9	17:34	0	0.014	
12	17:30	0	126 - 923	

10	17:39	0	2	Test halted (no more to say)
5				
13				
3				

4.2.5 User9 CentOS KDE

#	Timestamp	Evaluation	Time to complete	Notes
00	17:00	0		
4	17:04	0	10.14	
9	17:09	0	0.004	
13	17:14	-1	2.06	
2	17:24	0	15.862	
5	17:27	0	8.548	
1	17:28	0	13.976	
11	17:33	0	27.676	
7	17:35	0	6.752	
6	17:38	0	3.889	
12	17:43	0	127 - 294	Test halted (no more to say)
3				
10				
8				

5 Final remarks

Common benchmarks do not give a direct indication of performance that is relevant for interactive applications, where the most important parameter is responsiveness perceived by the user. We developed a method for observing low-level system parameters while desktop operations take place, in order to describe correlations and hence trace a mapping. By analyzing such benchmarks, we can achieve an understanding of the low-level behavior of the system related to them.

This is primarily meant to be a metodology to be used for understanding system behaviour from a user point of view and hence dimensioning VDD components and resources accordingly, but there are also some practical lessons learned. Some distros performed better than others (see Section 4.1). In particular we were impressed by GNU/Linux CentOS, whereas Debian Lenny and Gentoo 10 were just a confirmation of the excellent systems they are. Also Fedora and Windows (especially XP) did quite well during tests. But in general every dispatched desktop, when tested one by one, made it impossible to understand that they were not physical and local.

Individual use performs very well, and so does group use when users are well distributed on different virtual machines. We discovered, though, that the number of concurrent users using the same virtual machine is critical. We saw that just when two do, some occasional problems may occur (see Section 4.1.3), if heavy CPU-bound operations are executed at the same time. This without making any system tuning.

We then started to tune the system in order to find a critical threshold, and we succeded: if the system is well dimensioned, 5 concurrent users can work perfectly, the 6 can not (Section 4.2). In particular, more memory needs to be given to the guest machine (roughly the sum of the memory given to each user if different virtual machines were used). This include situations where more then one desktop is dispatched on a single terminal and one can switch between terminals using virtual consoles. Each virtual console consumes memory, so the grant of memory works in the same way.

Another interesting insight in multiprocessing has been gained. CPU architects are designing processors that count on the availability of thread-level parallelism[4], even if this is not the case for desktop applications, at least the ones available today. We monitored the eight cores of our Intel i7 CPU and we understood how workloads is distributed on them. Since desktop applications are not suitable for a parallel workload distribution, the only CPU core to be stressed during a standard desktop use is the first. But a big advantage of using virtual machines is the possibility of associating each of them to a different core. This way VDD can really increment efficiency even in a desktop situation, exploiting CPU multicores indeed.

The difference between 32-bit and 64-bit could not be perceived by users during tests. The tendency of DE, to open thousands of little files as they start up affects performance and there is little that the kernel can do about it [1] .

Much work can still be done on this side. This experience can be considered just a starting point, maybe indicating an interesting way to explore. VDD is still at its early stages, but we believe that performance is a key issue when desktops are concerned. So whenever a specific deployment of VDD is needed such studies could assist to put final users in a confortable environment.

Appendix 1 - Dstat patch

```
--- dstat-0.6.9/dstat   2008-12-02 21:23:05.000000000 +0100
+++ dstat-0.6.9/dstat   2010-02-16 16:09:45.000000000 +0100

@@ -2211,9 +2211,9 @@
              ### Check when to display the header
              if op.header and rows >= 6:
                  if oldvislist != vislist:
-                     showheader = True
+                     showheader = False
                  elif step == 1 and loop % (rows - 1) == 0:
-                     showheader = True
+                     showheader = False

              oldvislist = vislist
          else:
```

Appendix 2 – Macro for supporting analysis

```
REM  *****  BASIC  *****

Sub Main

End Sub


Function sys_graphs ( desktop As String, time_of_event As Integer, seconds_before As Integer, seconds_after As Integer ) As String
	Dim left_end (0) As Integer
	Dim right_end (0) As Integer
	Dim left_displacement (0) As Integer
	Dim right_displacement (0) As Integer
	Dim sample_of_event (0) As Integer
	Dim last_sample (0) As Integer
	left_displacement (0) = seconds_before
	right_displacement (0) = seconds_after
	left_end (0) = time_of_event - left_displacement (0)
	right_end (0) = time_of_event + right_displacement (0)
	sample_of_event (0) = left_displacement (0) + 1
	last_sample (0) = left_displacement (0) + right_displacement (0) +1


Dim Doc As Object
Dim Charts As Object
Dim Chart1 as Object
Dim Chart2 as Object
Dim Chart3 As Object
Dim Chart4 As Object
Dim Chart5 As Object
Dim Chart6 As Object
Dim Chart1h as Object
Dim Chart2h as Object
```

```
Dim Chart3h As Object
Dim Chart4h As Object
Dim Chart5h As Object
Dim Chart6h As Object
Dim Diagram as Object
Dim Cell As Object

Doc = ThisComponent
Cell = Doc.Sheets(0).getCellByPosition(0, time_of_event)
newline = chr(10)

      REM CPU usage range

      Dim aRect as new com.sun.star.awt.Rectangle
    with aRect
            .X = 0 : .Y = 0 : .Width = 18000 : .Height = 10000
    end with

      Dim aRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with aRangeAddress(0)
            .Sheet = 0 : .StartColumn = 1 : .EndColumn = 6 : .StartRow = 2 :
.EndRow = 2
      end with
      with aRangeAddress(1)
            .Sheet = 0 : .StartColumn = 1 : .EndColumn = 6 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with


      REM Load avarage range
```

```
      Dim bRect as new com.sun.star.awt.Rectangle
    with bRect
            .X = 1000 : .Y = 1000 : .Width = 18000 : .Height = 10000
    end with

      Dim bRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with bRangeAddress(0)
            .Sheet = 0 : .StartColumn = 7 : .EndColumn = 9 : .StartRow = 2 :
.EndRow = 2
      end with
      with bRangeAddress(1)
            .Sheet = 0 : .StartColumn = 7 : .EndColumn = 9 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with


      REM Memory usage range

      Dim cRect as new com.sun.star.awt.Rectangle
    with cRect
            .X = 1000 : .Y = 0 : .Width = 18000 : .Height = 10000
    end with

      Dim cRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with cRangeAddress(0)
            .Sheet = 0 : .StartColumn = 10 : .EndColumn = 13 : .StartRow =
2 : .EndRow = 2
      end with
      with cRangeAddress(1)
            .Sheet = 0 : .StartColumn = 10 : .EndColumn = 13 : .StartRow =
left_end (0) : .EndRow = right_end (0)
```

```
      end with


      REM Swap usage range

      Dim dRect as new com.sun.star.awt.Rectangle
    with dRect
            .X = 2000 : .Y = 1000 : .Width = 18000 : .Height = 10000
    end with

      Dim dRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with dRangeAddress(0)
            .Sheet = 0 : .StartColumn = 14 : .EndColumn = 15 : .StartRow =
2 : .EndRow = 2
      end with
      with dRangeAddress(1)
            .Sheet = 0 : .StartColumn = 14 : .EndColumn = 15 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with


      REM Net load range

      Dim eRect as new com.sun.star.awt.Rectangle
    with eRect
            .X = 2000 : .Y = 0 : .Width = 18000 : .Height = 10000
    end with

      Dim eRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with eRangeAddress(0)
```

```
            .Sheet = 0 : .StartColumn = 16 : .EndColumn = 17 : .StartRow =
2 : .EndRow = 2
      end with
      with eRangeAddress(1)
            .Sheet = 0 : .StartColumn = 16 : .EndColumn = 17 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with



      REM Disk load range


      Dim fRect as new com.sun.star.awt.Rectangle
    with fRect
            .X = 3000 : .Y = 1000 : .Width = 18000 : .Height = 10000
    end with


      Dim fRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with fRangeAddress(0)
            .Sheet = 0 : .StartColumn = 18 : .EndColumn = 19 : .StartRow =
2 : .EndRow = 2
      end with
      with fRangeAddress(1)
            .Sheet = 0 : .StartColumn = 18 : .EndColumn = 19 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with



Charts = Doc.Sheets(0).Charts


REM Cpu usage chart
```

```
Charts.addNewByName("vdd_guest_1", aRect, aRangeAddress(), True, False)

Chart1 = Charts.getByName("vdd_guest_1").EmbeddedObject

Chart1.Diagram = Chart1.createInstance("com.sun.star.chart.LineDiagram")

Chart1.HasMainTitle = True

Chart1.Title.String = desktop + " @ " + Cell.String + " (sample " + sample_of_event (0) + ")"

Chart1.HasSubTitle = True

Chart1.Subtitle.String = "CPU usage"

Chart1.HasLegend = True

Chart1.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM

Chart1.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID

Chart1.Legend.FillColor = RGB(235, 235, 235)

Chart1.Legend.CharHeight = 8

Chart1.Diagram.HasYAxisGrid = True

Chart1.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)

Chart1.Diagram.HasXAxisTitle = True

Chart1.Diagram.XAxisTitle.String = "Time" + newline + "from " + left_displacement (0) + " seconds before (sample 1) to " + right_displacement (0) + " seconds after (sample " + last_sample (0) + ")"



REM Load avarage chart


Charts.addNewByName("vdd_guest_2", bRect, bRangeAddress(), True, False)

Chart2 = Charts.getByName("vdd_guest_2").EmbeddedObject

Chart2.Diagram = Chart2.createInstance("com.sun.star.chart.LineDiagram")

Chart2.HasMainTitle = True

Chart2.Title.String = desktop + " @ " + Cell.String + " (sample " + sample_of_event (0) + ")"

Chart2.HasSubTitle = True

Chart2.Subtitle.String = "Load avarage (1 minute, 5 minutes, 15 minutes)"
```

```
Chart2.HasLegend = True
Chart2.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM
Chart2.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID
Chart2.Legend.FillColor = RGB(235, 235, 235)
Chart2.Legend.CharHeight = 8
Chart2.Diagram.HasYAxisGrid = True
Chart2.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)
Chart2.Diagram.HasXAxisTitle = True
Chart2.Diagram.XAxisTitle.String = "Time" + newline + "from " +
left_displacement (0) + " seconds before (sample 1) to " + right_displacement
(0) + " seconds after (sample " + last_sample (0) + ")"


REM Memory usage chart


Charts.addNewByName("vdd_guest_3", cRect, cRangeAddress(), True, False)
Chart3 = Charts.getByName("vdd_guest_3").EmbeddedObject
Chart3.Diagram = Chart3.createInstance("com.sun.star.chart.LineDiagram")
Chart3.HasMainTitle = True
Chart3.Title.String = desktop + " @ " + Cell.String + " (sample " +
sample_of_event (0) + ")"
Chart3.HasSubTitle = True
Chart3.Subtitle.String = "Memory usage"
Chart3.HasLegend = True
Chart3.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM
Chart3.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID
Chart3.Legend.FillColor = RGB(235, 235, 235)
Chart3.Legend.CharHeight = 8
Chart3.Diagram.HasYAxisGrid = True
Chart3.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)
Chart3.Diagram.HasXAxisTitle = True
```

```
Chart3.Diagram.XAxisTitle.String = "Time" + newline + "from " + left_displacement (0) + " seconds before (sample 1) to " + right_displacement (0) + " seconds after (sample " + last_sample (0) + ")"


REM Swap usage chart


Charts.addNewByName("vdd_guest_4", dRect, dRangeAddress(), True, False)
Chart4 = Charts.getByName("vdd_guest_4").EmbeddedObject
Chart4.Diagram = Chart4.createInstance("com.sun.star.chart.LineDiagram")
Chart4.HasMainTitle = True
Chart4.Title.String = desktop + " @ " + Cell.String + " (sample " + sample_of_event (0) + ")"
Chart4.HasSubTitle = True
Chart4.Subtitle.String = "Swap usage"
Chart4.HasLegend = True
Chart4.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM
Chart4.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID
Chart4.Legend.FillColor = RGB(235, 235, 235)
Chart4.Legend.CharHeight = 8
Chart4.Diagram.HasYAxisGrid = True
Chart4.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)
Chart4.Diagram.HasXAxisTitle = True
Chart4.Diagram.XAxisTitle.String = "Time" + newline + "from " + left_displacement (0) + " seconds before (sample 1) to " + right_displacement (0) + " seconds after (sample " + last_sample (0) + ")"


REM Net load chart


Charts.addNewByName("vdd_guest_5", eRect, eRangeAddress(), True, False)
Chart5 = Charts.getByName("vdd_guest_5").EmbeddedObject
```

```
Chart5.Diagram = Chart5.createInstance("com.sun.star.chart.LineDiagram")
Chart5.HasMainTitle = True
Chart5.Title.String = desktop + " @ " + Cell.String + " (sample " + sample_of_event (0) + ")"
Chart5.HasSubTitle = True
Chart5.Subtitle.String = "Net load"
Chart5.HasLegend = True
Chart5.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM
Chart5.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID
Chart5.Legend.FillColor = RGB(235, 235, 235)
Chart5.Legend.CharHeight = 8
Chart5.Diagram.HasYAxisGrid = True
Chart5.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)
Chart5.Diagram.HasXAxisTitle = True
Chart5.Diagram.XAxisTitle.String = "Time" + newline + "from " + left_displacement (0) + " seconds before (sample 1) to " + right_displacement (0) + " seconds after (sample " + last_sample (0) + ")"


REM Disk load chart


Charts.addNewByName("vdd_guest_6", fRect, fRangeAddress(), True, False)
Chart6 = Charts.getByName("vdd_guest_6").EmbeddedObject
Chart6.Diagram = Chart6.createInstance("com.sun.star.chart.LineDiagram")
Chart6.HasMainTitle = True
Chart6.Title.String = desktop + " @ " + Cell.String + " (sample " + sample_of_event (0) + ")"
Chart6.HasSubTitle = True
Chart6.Subtitle.String = "Disk load"
Chart6.HasLegend = True
Chart6.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM
```

```
Chart6.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID

Chart6.Legend.FillColor = RGB(235, 235, 235)

Chart6.Legend.CharHeight = 8

Chart6.Diagram.HasYAxisGrid = True

Chart6.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)

Chart6.Diagram.HasXAxisTitle = True

Chart6.Diagram.XAxisTitle.String = "Time" + newline + "from " +
left_displacement (0) + " seconds before (sample 1) to " + right_displacement
(0) + " seconds after (sample " + last_sample (0) + ")"



      REM Host CPU usage range


      Dim ahRect as new com.sun.star.awt.Rectangle
    with ahRect
            .X = 0 : .Y = 20000 : .Width = 18000 : .Height = 10000
    end with


      Dim ahRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with ahRangeAddress(0)
            .Sheet = 1 : .StartColumn = 1 : .EndColumn = 6 : .StartRow = 2 :
.EndRow = 2
      end with
      with ahRangeAddress(1)
            .Sheet = 1 : .StartColumn = 1 : .EndColumn = 6 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with



      REM Host Load avarage range
```

```
      Dim bhRect as new com.sun.star.awt.Rectangle
    with bhRect
            .X = 1000 : .Y = 21000 : .Width = 18000 : .Height = 10000
    end with

      Dim bhRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with bhRangeAddress(0)
            .Sheet = 1 : .StartColumn = 7 : .EndColumn = 9 : .StartRow = 2 :
.EndRow = 2
      end with
      with bhRangeAddress(1)
            .Sheet = 1 : .StartColumn = 7 : .EndColumn = 9 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with


      REM Host Memory usage range

      Dim chRect as new com.sun.star.awt.Rectangle
    with chRect
            .X = 1000 : .Y = 20000 : .Width = 18000 : .Height = 10000
    end with

      Dim chRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with chRangeAddress(0)
            .Sheet = 1 : .StartColumn = 10 : .EndColumn = 13 : .StartRow =
2 : .EndRow = 2
      end with
      with chRangeAddress(1)
```

```
            .Sheet = 1 : .StartColumn = 10 : .EndColumn = 13 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with


      REM Host Swap usage range

      Dim dhRect as new com.sun.star.awt.Rectangle
    with dhRect
            .X = 2000 : .Y = 21000 : .Width = 18000 : .Height = 10000
    end with

      Dim dhRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with dhRangeAddress(0)
            .Sheet = 1 : .StartColumn = 14 : .EndColumn = 15 : .StartRow =
2 : .EndRow = 2
      end with
      with dhRangeAddress(1)
            .Sheet = 1 : .StartColumn = 14 : .EndColumn = 15 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with


      REM Host Net load range

      Dim ehRect as new com.sun.star.awt.Rectangle
    with ehRect
            .X = 2000 : .Y = 20000 : .Width = 18000 : .Height = 10000
    end with

      Dim ehRangeAddress(1) As New com.sun.star.table.CellRangeAddress
```

```
      with ehRangeAddress(0)
            .Sheet = 1 : .StartColumn = 16 : .EndColumn = 17 : .StartRow =
2 : .EndRow = 2
      end with
      with ehRangeAddress(1)
            .Sheet = 1 : .StartColumn = 16 : .EndColumn = 17 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with



      REM Host Disk load range


      Dim fhRect as new com.sun.star.awt.Rectangle
    with fhRect
            .X = 3000 : .Y = 21000 : .Width = 18000 : .Height = 10000
    end with


      Dim fhRangeAddress(1) As New com.sun.star.table.CellRangeAddress
      with fhRangeAddress(0)
            .Sheet = 1 : .StartColumn = 18 : .EndColumn = 19 : .StartRow =
2 : .EndRow = 2
      end with
      with fhRangeAddress(1)
            .Sheet = 1 : .StartColumn = 18 : .EndColumn = 19 : .StartRow =
left_end (0) : .EndRow = right_end (0)
      end with



REM Host Cpu usage chart
```

```
Charts.addNewByName("vdd_host_1", ahRect, ahRangeAddress(), True, False)

Chart1h = Charts.getByName("vdd_host_1").EmbeddedObject

Chart1h.Diagram = Chart1h.createInstance("com.sun.star.chart.LineDiagram")

Chart1h.HasMainTitle = True

Chart1h.Title.String = "Host @ " + Cell.String + " (sample " + sample_of_event
(0) + ")"

Chart1h.HasSubTitle = True

Chart1h.Subtitle.String = "CPU usage"

Chart1h.HasLegend = True

Chart1h.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM

Chart1h.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID

Chart1h.Legend.FillColor = RGB(235, 235, 235)

Chart1h.Legend.CharHeight = 8

Chart1h.Diagram.HasYAxisGrid = True

Chart1h.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)

Chart1h.Diagram.HasXAxisTitle = True

Chart1h.Diagram.XAxisTitle.String = "Time" + newline + "from " +
left_displacement (0) + " seconds before (sample 1) to " + right_displacement
(0) + " seconds after (sample " + last_sample (0) + ")"



REM Host Load avarage chart


Charts.addNewByName("vdd_host_2", bhRect, bhRangeAddress(), True, False)

Chart2h = Charts.getByName("vdd_host_2").EmbeddedObject

Chart2h.Diagram = Chart2h.createInstance("com.sun.star.chart.LineDiagram")

Chart2h.HasMainTitle = True

Chart2h.Title.String = "Host @ " + Cell.String + " (sample " + sample_of_event
(0) + ")"

Chart2h.HasSubTitle = True

Chart2h.Subtitle.String = "Load avarage (1 minute, 5 minutes, 15 minutes)"
```

```
Chart2h.HasLegend = True
Chart2h.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM
Chart2h.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID
Chart2h.Legend.FillColor = RGB(235, 235, 235)
Chart2h.Legend.CharHeight = 8
Chart2h.Diagram.HasYAxisGrid = True
Chart2h.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)
Chart2h.Diagram.HasXAxisTitle = True
Chart2h.Diagram.XAxisTitle.String = "Time" + newline + "from " + left_displacement (0) + " seconds before (sample 1) to " + right_displacement (0) + " seconds after (sample " + last_sample (0) + ")"


REM Host Memory usage chart


Charts.addNewByName("vdd_host_3", chRect, chRangeAddress(), True, False)
Chart3h = Charts.getByName("vdd_host_3").EmbeddedObject
Chart3h.Diagram = Chart3h.createInstance("com.sun.star.chart.LineDiagram")
Chart3h.HasMainTitle = True
Chart3h.Title.String = "Host @ " + Cell.String + " (sample " + sample_of_event (0) + ")"
Chart3h.HasSubTitle = True
Chart3h.Subtitle.String = "Memory usage"
Chart3h.HasLegend = True
Chart3h.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM
Chart3h.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID
Chart3h.Legend.FillColor = RGB(235, 235, 235)
Chart3h.Legend.CharHeight = 8
Chart3h.Diagram.HasYAxisGrid = True
Chart3h.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)
Chart3h.Diagram.HasXAxisTitle = True
```

```
Chart3h.Diagram.XAxisTitle.String = "Time" + newline + "from " + left_displacement (0) + " seconds before (sample 1) to " + right_displacement (0) + " seconds after (sample " + last_sample (0) + ")"


REM Host Swap usage chart


Charts.addNewByName("vdd_host_4", dhRect, dhRangeAddress(), True, False)
Chart4h = Charts.getByName("vdd_host_4").EmbeddedObject
Chart4h.Diagram = Chart4h.createInstance("com.sun.star.chart.LineDiagram")
Chart4h.HasMainTitle = True
Chart4h.Title.String = "Host @ " + Cell.String + " (sample " + sample_of_event (0) + ")"
Chart4h.HasSubTitle = True
Chart4h.Subtitle.String = "Swap usage"
Chart4h.HasLegend = True
Chart4h.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM
Chart4h.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID
Chart4h.Legend.FillColor = RGB(235, 235, 235)
Chart4h.Legend.CharHeight = 8
Chart4h.Diagram.HasYAxisGrid = True
Chart4h.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)
Chart4h.Diagram.HasXAxisTitle = True
Chart4h.Diagram.XAxisTitle.String = "Time" + newline + "from " + left_displacement (0) + " seconds before (sample 1) to " + right_displacement (0) + " seconds after (sample " + last_sample (0) + ")"


REM Host Net load chart


Charts.addNewByName("vdd_host_5", ehRect, ehRangeAddress(), True, False)
Chart5h = Charts.getByName("vdd_host_5").EmbeddedObject
```

```
Chart5h.Diagram = Chart5h.createInstance("com.sun.star.chart.LineDiagram")
Chart5h.HasMainTitle = True
Chart5h.Title.String = "Host @ " + Cell.String + " (sample " + sample_of_event (0) + ")"
Chart5h.HasSubTitle = True
Chart5h.Subtitle.String = "Net load"
Chart5h.HasLegend = True
Chart5h.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM
Chart5h.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID
Chart5h.Legend.FillColor = RGB(235, 235, 235)
Chart5h.Legend.CharHeight = 8
Chart5h.Diagram.HasYAxisGrid = True
Chart5h.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)
Chart5h.Diagram.HasXAxisTitle = True
Chart5h.Diagram.XAxisTitle.String = "Time" + newline + "from " + left_displacement (0) + " seconds before (sample 1) to " + right_displacement (0) + " seconds after (sample " + last_sample (0) + ")"


REM Host Disk load chart


Charts.addNewByName("vdd_host_6", fhRect, fhRangeAddress(), True, False)
Chart6h = Charts.getByName("vdd_host_6").EmbeddedObject
Chart6h.Diagram = Chart6h.createInstance("com.sun.star.chart.LineDiagram")
Chart6h.HasMainTitle = True
Chart6h.Title.String = "Host @ " + Cell.String + " (sample " + sample_of_event (0) + ")"
Chart6h.HasSubTitle = True
Chart6h.Subtitle.String = "Disk load"
Chart6h.HasLegend = True
Chart6h.Legend.Alignment = com.sun.star.chart.ChartLegendPosition.BOTTOM
```

```
Chart6h.Legend.FillStyle = com.sun.star.drawing.FillStyle.SOLID

Chart6h.Legend.FillColor = RGB(235, 235, 235)

Chart6h.Legend.CharHeight = 8

Chart6h.Diagram.HasYAxisGrid = True

Chart6h.Diagram.YMainGrid.LineColor = RGB(235, 235, 235)

Chart6h.Diagram.HasXAxisTitle = True

Chart6h.Diagram.XAxisTitle.String = "Time" + newline + "from " +
left_displacement (0) + " seconds before (sample 1) to " + right_displacement
(0) + " seconds after (sample " + last_sample (0) + ")"



sys_graphs = "VDD rulez!"

End Function
```

References

[1] The 2004 Kernel Summit: Desktop performance http://lwn.net/Articles/94203/

[2] Linux Journal. Linux in Government: Optimizing Desktop Performance
http://www.linuxjournal.com/article/8308

[3] Thread-level parallelism and interactive performance of desktop applications

http://portal.acm.org/citation.cfm?id=357001

[4] Towards a Methodology employing Critical Parameters to deliver Performance Improvements in Interactive Systems

http://scholar.google.com/scholar?hl=it&lr=&client=iceweasel-a&rls=org.debian:it:unofficial&q=author:%22Newman%22+intitle:%22Towards+a+methodology+employing+critical+parameters+to+...%22+&um=1&ie=UTF-8&oi=scholarr

[5] Using latency to evaluate interactive system performance

http://www.eecs.harvard.edu/~yaz/latency/latency.ps

[6] Miglioramento delle prestazioni e dell'affidabilità di Virtual Distro Dispatcher (Improving perfomance and reliability of Virtual Distro Dispatcher), final thesis by Salvatore Cristofaro

[7] Virtual Distro Dispatcher: a light-weight Desktop-as-a-Service solution.

Salvatore Cristofaro, Flavio Bertini, D. Davide Lamanna, Roberto Baldoni

First International Conference on Cloud Computing October 19 - 21, 2009, Munich, Germany.

[8] How we test: Desktops

http://reviews.cnet.com/4520-6603_7-5020816-1.html

[9] Ben Shneiderman, Designing the User Interface, Addison-Wesley, 1992.