
Glances Documentation

Release 2.11.1

Nicolas Hennion

Sep 09, 2017

Contents

1	Table of Contents	3
1.1	Install	3
1.2	Quickstart	3
1.3	Command Reference	8
1.4	Configuration	13
1.5	Anatomy Of The Application	16
1.6	Gateway To Other Services	33
1.7	API Documentation	41
1.8	Support	41


```
xps (Ubuntu 14.04 64bit / Linux 3.13.0-85-generic) - IP 192.168.0.6/24 Uptime: 1 day, 20:23:55

1.80/1.80GHz CPU [|||||] 100% CPU 100.0% nice: 0.0% ctx sw: 7605 MEM 26.8% active: 878M SWAP 7.6% LOAD 4-core
CPU [|||||] 26.8% user: 97.9% irq: 0.0% inter: 5015 total: 7.71G inactive: 1.58G total: 7.91G 1 min: 3.21
MEM [|||||] 26.8% system: 2.1% iowait: 0.0% sw_int: 1273 used: 2.06G buffers: 5.73M used: 612M 5 min: 1.86
SWAP [||] 7.6% idle: 0.0% steal: 0.0% free: 5.64G cached: 645M free: 7.31G 15 min: 1.17

NETWORK Rx/s Tx/s CONTAINERS 2 (served by Docker 1.11.1)
docker0 0b 0b
lo 392b 392b
h2c39a99 0b 0b
h610b701 0b 0b
wlan0 10.5Mb 860Kb

Name Status CPU% MEM /MAX IOR/s IOW/s Rx/s Tx/s Command
_dbgrafana grafana_1 Up 2 mins 0.0 5.94M 7.71G 0b 0b 0b 0b /run.sh
_bgrafana influxdb_1 Up 2 mins 0.1 8.60M 7.71G 0b 0b 0b 0b /run.sh

TASKS 257 (780 thr), 5 run, 252 slp, 0 oth sorted automatically by cpu_percent, flat view

DISK I/O R/s W/s CPU% MEM% VIRT RES PID USER NI S TIME+ R/s W/s Command
sda1 0 0 96.0 0.0 7.13M 100K 20888 nicolargo 0 R 0:03.50 0 0 stress --cpu 4 -t 30
sda2 78K 2K 91.8 0.0 7.13M 100K 20890 nicolargo 0 R 0:03.33 0 0 stress --cpu 4 -t 30
sda3 66K 0 91.5 0.0 7.13M 100K 20891 nicolargo 0 R 0:03.26 0 0 stress --cpu 4 -t 30

FILE SYS Used Total CPU% MEM% VIRT RES PID USER NI S TIME+ R/s W/s Command
/ (sda2) 71.2G 226G 12.7 11.8 2.50G 933M 11378 nicolargo 0 S 2h21:43 0 1M /usr/lib/firefox/firefox
/boot/efi 3.38M 511M 5.1 0.3 548M 23.4M 19899 nicolargo 0 R 0:07.27 0 0 python -m glances
4.3 2.1 2.04G 163M 3278 nicolargo 0 S 36:17.83 0 0 /usr/bin/gnome-shell
2.7 0.0 0 577 root 0 S 1:36.94 0 0 irq/59-iwlwifi
1.5 1.3 477M 100M 2141 root 0 S 17:18.96 0 0 /usr/bin/X :0 -background none -verbose -auth /var/run/gdm/aut
temp1 27C 1.5 1.5 1.18G 122M 23657 nicolargo 0 S 0:07.93 0 0 /usr/bin/perl /usr/bin/shutter
temp2 29C 1.5 1.5 1.18G 122M 23657 nicolargo 0 S 0:07.93 0 0 /usr/bin/perl /usr/bin/shutter
Physical id 0 74C 0.6 0.2 914M 19.2M 19237 nicolargo 0 S 0:33.56 0 0 /usr/bin/python /usr/bin/terminator
Core 0 74C 0.6 0.3 606M 20.8M 2870 root 0 S 0:30.96 0 0 /usr/bin/docker daemon --raw-logs
Core 1 74C 0.6 0.1 358M 6.70M 3142 nicolargo 0 S 1:14.27 20K 0 /usr/bin/ibus-daemon --daemonize --xim
Battery 33% 0.3 0.1 612M 11.1M 3381 nicolargo 19 S 0:44.80 0 0 /usr/lib/tracker/tracker-miner-fs
0.3 0.0 201M 1.16M 3227 nicolargo 0 S 0:19.28 2K 0 /usr/lib/ibus/ibus-engine-simple
0.3 0.3 1.18G 22.2M 4835 nicolargo 0 S 0:23.10 0 0 nautilus --new-window
0.3 0.0 410M 2.81M 1112 root 0 S 0:07.11 0 0 NetworkManager
0.3 0.3 649M 27.3M 3099 nicolargo 0 S 0:08.46 0 0 /usr/lib/x86_64-linux-gnu/bamf/bamfdaemon
0.3 0.0 0 79 root 0 S 0:20.63 0 0 kworker/3:1
0.3 0.0 88.4M 240K 2083 www-data 0 S 0:02.72 0 0 nginx: worker process
0.3 0.3 2.07G 25.6M 2194 rabbitmq 0 S 5:41.18 0 0 /usr/lib/erlang/erts-5.10.4/bin/beam.smp -W w -K true -A30 -P
0.3 0.0 0 16492 root 0 S 0:01.94 0 0 kworker/2:2
0.0 0.0 0 18 root 0 S 0:00.00 0 0 rcuob/1
0.0 0.1 999M 10.0M 3315 nicolargo 0 S 0:09.78 0 0 /usr/lib/gnome-online-accounts/goa-daemon
0.0 0.0 0 27916 root 0 S 0:00.00 0 0 irq/61-me_i
0.0 0.0 0 39 root 0 S 0:00.70 0 0 ksoftirqd/3

Warning or critical alerts (last 4 entries)
2016-05-16 16:53:12 (ongoing) - CPU_USER (97.6): stress, stress, stress
2016-05-16 16:52:41 (0:00:18) - WARNING on MEM (76.0)
2016-05-16 16:52:01 (0:00:33) - CRITICAL on CPU_IOWAIT (Min:51.9 Mean:63.4 Max:77.1): bash, stress, firefox
2016-05-16 16:53:15 2016-05-16 16:51:31 (0:00:30) - CRITICAL on CPU_USER (Min:80.5 Mean:95.8 Max:98.3): stress, stress, stress
```

Glances is a cross-platform monitoring tool which aims to present a maximum of information in a minimum of space through a curses or Web based interface. It can adapt dynamically the displayed information depending on the terminal size.

It can also work in client/server mode. Remote monitoring could be done via terminal, Web interface or API (XML-RPC and RESTful).

Glances is written in Python and uses the `psutil` library to get information from your system.

Stats can also be exported to external time/value databases.

Table of Contents

Install

Glances is on [PyPI](#). By using [PyPI](#), you are sure to have the latest stable version.

To install, simply use `pip`:

```
pip install glances
```

Note: Python headers are required to install [psutil](#). For example, on Debian/Ubuntu you need to install first the *python-dev* package. For Fedora/CentOS/RHEL install first *python-devel* package. For Windows, just install PsUtil from the binary installation file.

You can also install the following libraries in order to use optional features (like the Web interface, export modules...):

```
pip install glances[action,browser,cloud,cpuinfo,chart,docker,export,folders,gpu,ip,  
↪raid,snmp,web,wifi]
```

To upgrade Glances and all its dependencies to the latest versions:

```
pip install --upgrade glances  
pip install --upgrade psutil  
pip install --upgrade glances[...]
```

For additional installation methods, read the official [README](#) file.

Quickstart

This page gives a good introduction in how to get started with Glances. Glances offers 3 modes:

- Standalone
- Client/Server

- Web server

Standalone Mode

If you want to monitor your local machine, open a console/terminal and simply run:

```
$ glances
```

Glances should start (press ‘q’ or ‘ESC’ to exit):

```
xps (Ubuntu 14.04 64bit / Linux 3.13.0-85-generic) - IP 192.168.0.6/24 Uptime: 1 day, 20:23:55

1.80/1.80GHz CPU [|||||] 100% CPU 100.0% nice: 0.0% ctx_sw: 7605 MEM 26.8% active: 878M SWAP 7.6% LOAD 4-core
MEM [|||||] 26.8% user: 97.9% irq: 0.0% inter: 5015 total: 7.71G inactive: 1.58G total: 7.91G 1 min: 3.21
SWAP [|||||] 7.6% system: 2.1% iowait: 0.0% sw_int: 1273 used: 2.06G buffers: 5.73M used: 612M 5 min: 1.86
idle: 0.0% steal: 0.0% free: 5.64G cached: 645M free: 7.31G 15 min: 1.17

NETWORK Rx/s Tx/s CONTAINERS 2 (served by Docker 1.11.1)
docker0 0b 0b
lo 392b 392b
h2c39a99 0b 0b
h610b701 0b 0b
wlan0 10.5Mb 860Kb

DISK I/O R/s W/s
sda1 0 0
sda2 78K 2K
sda3 66K 0

FILE SYS Used Total
/ (sda2) 71.2G 226G
/boot/efi 3.38M 511M

SENSORS
temp1 27C
temp2 29C
Physical id 0 74C
Core 0 74C
Core 1 74C
Battery 33%

TASKS 257 (780 thr), 5 run, 252 slp, 0 oth sorted automatically by cpu_percent, flat view

CPU% MEM% VIRT RES PID USER NI S TIME+ R/s W/s Command
96.0 0.0 7.13M 100K 20888 nicolargo 0 R 0:03.50 0 0 stress --cpu 4 -t 30
91.8 0.0 7.13M 100K 20890 nicolargo 0 R 0:03.33 0 0 stress --cpu 4 -t 30
91.5 0.0 7.13M 100K 20891 nicolargo 0 R 0:03.26 0 0 stress --cpu 4 -t 30
86.4 0.0 7.13M 100K 20889 nicolargo 0 R 0:03.19 0 0 stress --cpu 4 -t 30
12.7 11.8 2.50G 933M 11378 nicolargo 0 S 2h21:43 0 1M /usr/lib/firefox/firefox
5.1 0.3 548M 23.4M 19899 nicolargo 0 R 0:07.27 0 0 python -m glances
4.3 2.1 2.04G 163M 3278 nicolargo 0 S 36:17.83 0 0 /usr/bin/gnome-shell
2.7 0.0 0 0 577 root 0 S 1:36.94 0 0 irq/59-iwlwifi
1.5 1.3 477M 100M 2141 root 0 S 17:18.96 0 0 /usr/bin/X :0 -background none -verbose -auth /var/run/gdm/aut
1.5 1.5 1.18G 122M 23657 nicolargo 0 S 0:07.93 0 0 /usr/bin/perl /usr/bin/shutter
0.6 0.2 914M 19.2M 19237 nicolargo 0 S 0:33.56 0 0 /usr/bin/python /usr/bin/terminator
0.6 0.3 606M 20.8M 2870 root 0 S 0:30.96 0 0 /usr/bin/docker daemon --raw-logs
0.6 0.1 358M 6.70M 3142 nicolargo 0 S 1:14.27 20K 0 /usr/bin/ibus-daemon --daemonize --xim
0.6 0.1 612M 11.1M 3381 nicolargo 19 S 0:44.80 0 0 /usr/lib/tracker/tracker-miner-fs
0.3 0.0 201M 1.16M 3227 nicolargo 0 S 0:19.28 2K 0 /usr/lib/ibus/ibus-engine-simple
0.3 0.3 1.18G 22.2M 4835 nicolargo 0 S 0:23.10 0 0 nautilus --new-window
0.3 0.0 410M 2.81M 1112 root 0 S 0:07.11 0 0 NetworkManager
0.3 0.3 649M 27.3M 3099 nicolargo 0 S 0:08.46 0 0 /usr/lib/x86_64-linux-gnu/bamf/bamfdaemon
0.3 0.0 0 0 79 root 0 S 0:20.63 0 0 kworker/3:1
0.3 0.0 88.4M 240K 2083 www-data 0 S 0:02.72 0 0 nginx: worker process
0.3 0.3 2.07G 25.6M 2194 rabbitmq 0 S 5:41.18 0 0 /usr/lib/erlang/erts-5.10.4/bin/beam.smp -W w -K true -A30 -P
0.0 0.0 0 0 16492 root 0 S 0:01.94 0 0 kworker/2:2
0.0 0.0 0 0 18 root 0 S 0:00.00 0 0 rcuob/1
0.0 0.1 999M 10.0M 3315 nicolargo 0 S 0:09.78 0 0 /usr/lib/gnome-online-accounts/goa-daemon
0.0 0.0 0 0 27916 root 0 S 0:00.00 0 0 irq/61-mei me
0.0 0.0 0 0 39 root 0 S 0:00.70 0 0 ksoftirqd/3

Warning or critical alerts (last 4 entries)
2016-05-16 16:53:12 (ongoing) - CPU_USER (97.6): stress, stress, stress
2016-05-16 16:52:41 (0:00:18) - WARNING on MEM (76.0)
2016-05-16 16:52:01 (0:00:33) - CRITICAL on CPU IOWAIT (Min:51.9 Mean:63.4 Max:77.1): bash, stress, firefox
2016-05-16 16:51:31 (0:00:30) - CRITICAL on CPU_USER (Min:80.5 Mean:95.8 Max:98.3): stress, stress, stress
```

Client/Server Mode

If you want to remotely monitor a machine, called `server`, from another one, called `client`, just run on the server:

```
server$ glances -s
```

and on the client:

```
client$ glances -c @server
```

where `@server` is the IP address or hostname of the server.

In server mode, you can set the bind address with `-B ADDRESS` and the listening TCP port with `-p PORT`.

In client mode, you can set the TCP port of the server with `-p PORT`.

Default binding address is `0.0.0.0` (Glances will listen on all the available network interfaces) and TCP port is `61209`.

In client/server mode, limits are set by the server side.

You can set a password to access to the server `--password`. By default, the username is `glances` but you can change it with `--username`. It is also possible to set the password in the Glances configuration file:

```
[passwords]
# Define the passwords list
# Syntax: host=password
# Where: host is the hostname
#         password is the clear password
# Additionally (and optionally) a default password could be defined
localhost=mylocalhostpassword
default=mydefaultpassword
```

If you ask it, the SHA password will be stored in `username.pwd` file. Next time you run the server/client, password will not be asked.

Central client

2 Glances servers available

Name	LOAD	CPU%	MEM%	STATUS	IP	OS
> Win	?	100.0	90.4	ONLINE	192.168.0.17	Windows 7 (64bi
xps	0.66	55.1	59.6	ONLINE	192.168.0.7	Ubuntu 14.04 (64

Glances can centralize available Glances servers using the `--browser` option. The server list can be statically defined via the configuration file (section `[serverlist]`).

Example:

```
[serverlist]
# Define the static servers list
server_1_name=xps
server_1_alias=xps
server_1_port=61209
server_2_name=win
server_2_port=61235
```

Glances can also detect and display all Glances servers available on your network via the `zeroconf` protocol (not available on Windows):

To start the central client, use the following option:

```
client$ glances --browser
```

Note: Use `--disable-autodiscover` to disable the auto discovery mode.

SNMP

As an experimental feature, if Glances server is not detected by the client, the latter will try to grab stats using the SNMP protocol:

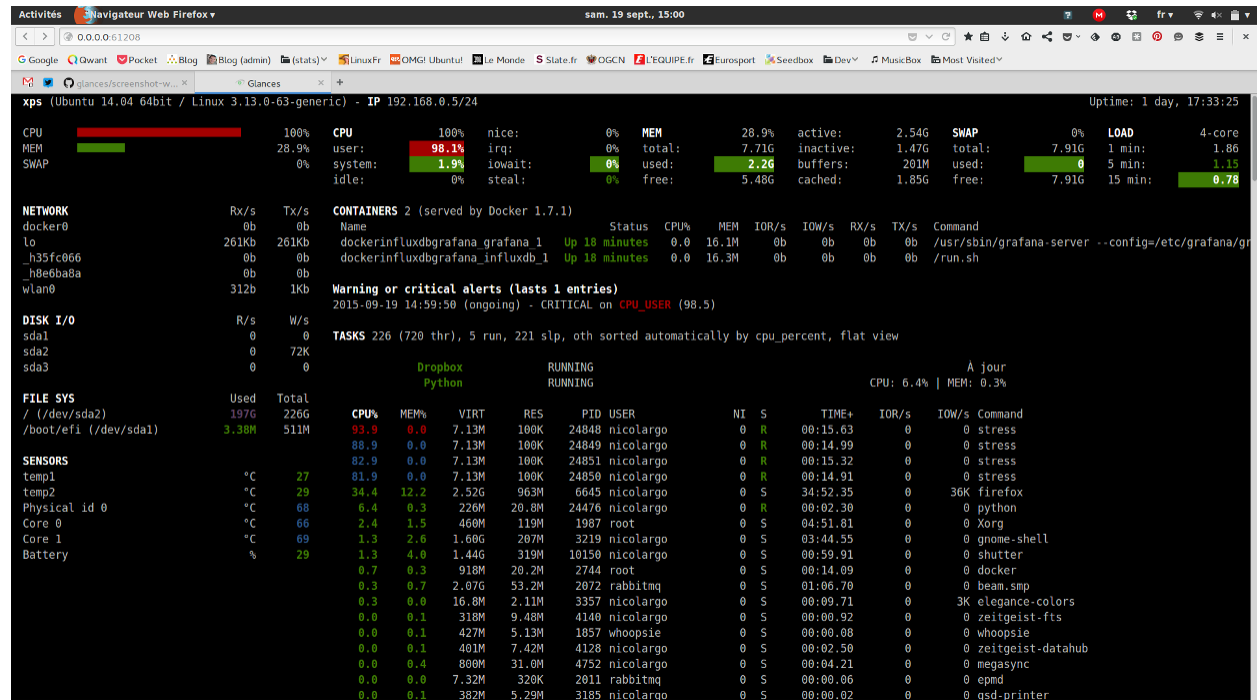
```
client$ glances -c @snmpserver
```

Note: Stats grabbed by SNMP request are limited and OS dependent. A SNMP server should be installed and configured...

IPv6

Glances is IPv6 compatible. Just use the `-B ::` option to bind to all IPv6 addresses.

Web Server Mode



If you want to remotely monitor a machine, called `server`, from any device with a web browser, just run the server with the `-w` option:

```
server$ glances -w
```

then on the client enter the following URL in your favorite web browser:

```
http://@server:61208
```

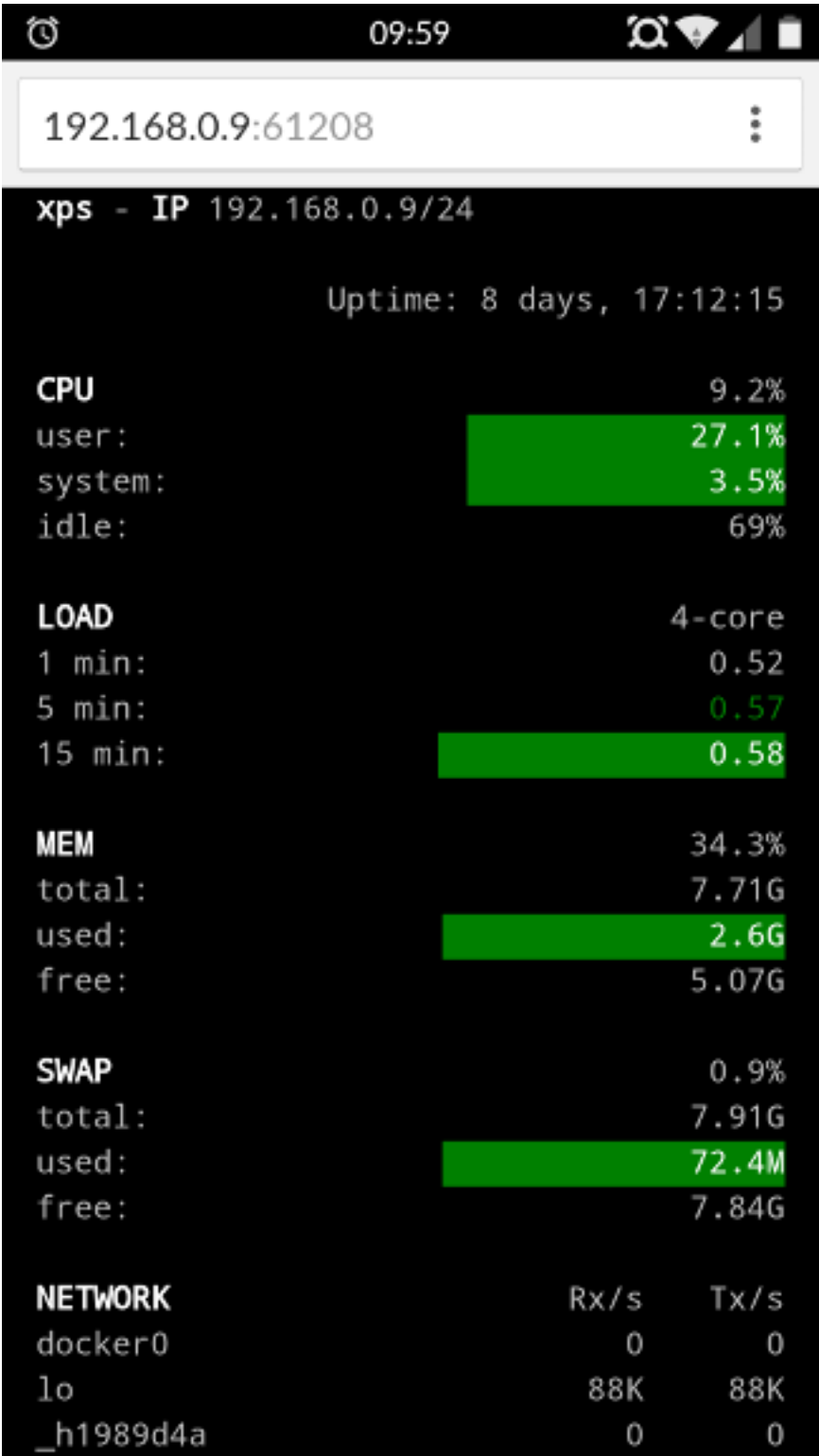
where `@server` is the IP address or hostname of the server.

To change the refresh rate of the page, just add the period in seconds at the end of the URL. For example, to refresh the page every 10 seconds:

```
http://@server:61208/10
```

The Glances web interface follows responsive web design principles.

Here's a screenshot from Chrome on Android:



Command Reference

Command-Line Options

- h, --help**
show this help message and exit
- V, --version**
show program's version number and exit
- d, --debug**
enable debug mode
- C CONF_FILE, --config CONF_FILE**
path to the configuration file
- disable-alert**
disable alert/log module
- disable-amps**
disable application monitoring process module
- disable-cpu**
disable CPU module
- disable-diskio**
disable disk I/O module
- disable-docker**
disable Docker module
- disable-folders**
disable folders module
- disable-fs**
disable file system module
- disable-hddtemp**
disable HD temperature module
- disable-ip**
disable IP module
- disable-irq**
disable IRQ module
- disable-load**
disable load module
- disable-mem**
disable memory module
- disable-memswap**
disable memory swap module
- disable-network**
disable network module
- disable-now**
disable current time module

--disable-ports
disable Ports module

--disable-process
disable process module

--disable-raid
disable RAID module

--disable-sensors
disable sensors module

--disable-wifi
disable Wifi module

-0, --disable-irix
task's CPU usage will be divided by the total number of CPUs

-1, --percpu
start Glances in per CPU mode

-2, --disable-left-sidebar
disable network, disk I/O, FS and sensors modules

-3, --disable-quicklook
disable quick look module

-4, --full-quicklook
disable all but quick look and load

-5, --disable-top
disable top menu (QuickLook, CPU, MEM, SWAP and LOAD)

-6, --meangpu
start Glances in mean GPU mode

--enable-history
enable the history mode (matplotlib lib needed)

--disable-bold
disable bold mode in the terminal

--disable-bg
disable background colors in the terminal

--enable-process-extended
enable extended stats on top process

--export-graph
export stats to graph

--path-graph PATH_GRAPH
set the export path for graph history

--export-csv EXPORT_CSV
export stats to a CSV file

--export-cassandra
export stats to a Cassandra/Scylla server (cassandra lib needed)

--export-couchdb
export stats to a CouchDB server (couchdb lib needed)

--export-elasticsearch
export stats to an Elasticsearch server (elasticsearch lib needed)

--export-influxdb
export stats to an InfluxDB server (influxdb lib needed)

--export-opentsdb
export stats to an OpenTSDB server (potsdb lib needed)

--export-rabbitmq
export stats to RabbitMQ broker (pika lib needed)

--export-statsd
export stats to a StatsD server (statsd lib needed)

--export-riemann
export stats to Riemann server (bernhard lib needed)

--export-zeromq
export stats to a ZeroMQ server (zmq lib needed)

-c CLIENT, --client CLIENT
connect to a Glances server by IPv4/IPv6 address, hostname or hostname:port

-s, --server
run Glances in server mode

--browser
start the client browser (list of servers)

--disable-autodiscover
disable autodiscover feature

-p PORT, --port PORT
define the client/server TCP port [default: 61209]

-B BIND_ADDRESS, --bind BIND_ADDRESS
bind server to the given IPv4/IPv6 address or hostname

--username
define a client/server username

--password
define a client/server password

--snmp-community SNMP_COMMUNITY
SNMP community

--snmp-port SNMP_PORT
SNMP port

--snmp-version SNMP_VERSION
SNMP version (1, 2c or 3)

--snmp-user SNMP_USER
SNMP username (only for SNMPv3)

--snmp-auth SNMP_AUTH
SNMP authentication key (only for SNMPv3)

--snmp-force
force SNMP mode

-t TIME, **--time** TIME
set refresh time in seconds [default: 3 sec]

-w, --webserver
run Glances in web server mode (bottle lib needed)

--cached-time CACHED_TIME
set the server cache time [default: 1 sec]

open-web-browser
try to open the Web UI in the default Web browser

-q, --quiet
do not display the curses interface

-f PROCESS_FILTER, **--process-filter** PROCESS_FILTER
set the process filter pattern (regular expression)

--process-short-name
force short name for processes name

--hide-kernel-threads
hide kernel threads in process list

--tree
display processes as a tree

-b, --byte
display network rate in byte per second

--diskio-show-ramfs
show RAM FS in the DiskIO plugin

--diskio-iops
show I/O per second in the DiskIO plugin

--fahrenheit
display temperature in Fahrenheit (default is Celsius)

--fs-free-space
display FS free space instead of used

--theme-white
optimize display colors for white background

--disable-check-update
disable online Glances version ckeck

Interactive Commands

The following commands (key pressed) are supported while in Glances:

ENTER Set the process filter

Note: On macOS please use CTRL-H to delete filter.

Filter is a regular expression pattern:

- `gnome`: matches all processes starting with the `gnome` string
- `.*gnome.*`: matches all processes containing the `gnome` string

- a** Sort process list automatically
 - If CPU >70%, sort processes by CPU usage
 - If MEM >70%, sort processes by MEM usage
 - If CPU iowait >60%, sort processes by I/O read and write
- A** Enable/disable Application Monitoring Process
- b** Switch between bit/s or Byte/s for network I/O
- B** View disk I/O counters per second
- c** Sort processes by CPU usage
- d** Show/hide disk I/O stats
- D** Enable/disable Docker stats
- e** Enable/disable top extended stats
- E** Erase current process filter
- f** Show/hide file system and folder monitoring stats
- F** Switch between file system used and free space
- g** Generate graphs for current history
- h** Show/hide the help screen
- i** Sort processes by I/O rate
- I** Show/hide IP module
- l** Show/hide log messages
- m** Sort processes by MEM usage
- M** Reset processes summary min/max
- n** Show/hide network stats
- N** Show/hide current time
- p** Sort processes by name
- q|ESC** Quit the current Glances session
- Q** Show/hide IRQ module
- r** Reset history
- R** Show/hide RAID plugin
- s** Show/hide sensors stats
- t** Sort process by CPU times (TIME+)
- T** View network I/O as combination
- u** Sort processes by USER
- U** View cumulative network I/O
- w** Delete finished warning log messages
- W** Show/hide Wifi module
- x** Delete finished warning and critical log messages

z Show/hide processes stats

0 Enable/disable Irix/Solaris mode

Task's CPU usage will be divided by the total number of CPUs

1 Switch between global CPU and per-CPU stats

2 Enable/disable left sidebar

3 Enable/disable the quick look module

4 Enable/disable all but quick look and load module

5 Enable/disable top menu (QuickLook, CPU, MEM, SWAP and LOAD)

6 Enable/disable mean GPU mode

/ Switch between process command line or command name

In the Glances client browser (accessible through the `--browser` command line argument):

ENTER Run the selected server

UP Up in the servers list

DOWN Down in the servers list

q|ESC Quit Glances

Configuration

No configuration file is mandatory to use Glances.

Furthermore a configuration file is needed to access more settings.

Location

Note: A template is available in the `/usr{,/local}/share/doc/glances` (Unix-like) directory or directly on [GitHub](#).

You can put your own `glances.conf` file in the following locations:

Linux, SunOS	<code>~/config/glances, /etc/glances</code>
*BSD	<code>~/config/glances, /usr/local/etc/glances</code>
macOS	<code>~/Library/Application Support/glances, /usr/local/etc/glances</code>
Windows	<code>%APPDATA%\glances</code>

- On Windows XP, `%APPDATA%` is: `C:\Documents and Settings\<USERNAME>\Application Data`.
- On Windows Vista and later: `C:\Users\<USERNAME>\AppData\Roaming`.

User-specific options override system-wide options and options given on the command line override either.

Syntax

Glances reads configuration files in the *ini* syntax.

A first section (called global) is available:

```
[global]
# Does Glances should check if a newer version is available on PyPI?
check_update=true
```

Each plugin, export module and application monitoring process (AMP) can have a section. Below an example for the CPU plugin:

```
[cpu]
user_careful=50
user_warning=70
user_critical=90
iowait_careful=50
iowait_warning=70
iowait_critical=90
system_careful=50
system_warning=70
system_critical=90
steal_careful=50
steal_warning=70
steal_critical=90
```

an InfluxDB export module:

```
[influxdb]
# Configuration for the --export-influxdb option
# https://influxdb.com/
host=localhost
port=8086
user=root
password=root
db=glances
prefix=localhost
#tags=foo:bar,spam:eggs
```

or a Nginx AMP:

```
[amp_nginx]
# Nginx status page should be enable (https://easyengine.io/tutorials/nginx/status-
↪page/)
enable=true
regex=/usr/sbin/nginx
refresh=60
one_line=false
status_url=http://localhost/nginx_status
```

Logging

Glances logs all of its internal messages to a log file.

DEBUG messages can be logged using the `-d` option on the command line.

By default, the `glances-USERNAME.log` file is under the temporary directory:

*nix	/tmp
Windows	%TEMP%

- On Windows XP, %TEMP% is: C:\Documents and Settings\<USERNAME>\Local Settings\Temp.
- On Windows Vista and later: C:\Users\<USERNAME>\AppData\Local\Temp.

If you want to use another system path or change the log message, you can use your own logger configuration. First of all, you have to create a `glances.json` file with, for example, the following content (JSON format):

```
{
  "version": 1,
  "disable_existing_loggers": "False",
  "root": {
    "level": "INFO",
    "handlers": ["file", "console"]
  },
  "formatters": {
    "standard": {
      "format": "%(asctime)s -- %(levelname)s -- %(message)s"
    },
    "short": {
      "format": "%(levelname)s: %(message)s"
    },
    "free": {
      "format": "%(message)s"
    }
  },
  "handlers": {
    "file": {
      "level": "DEBUG",
      "class": "logging.handlers.RotatingFileHandler",
      "formatter": "standard",
      "filename": "/var/tmp/glances.log"
    },
    "console": {
      "level": "CRITICAL",
      "class": "logging.StreamHandler",
      "formatter": "free"
    }
  },
  "loggers": {
    "debug": {
      "handlers": ["file", "console"],
      "level": "DEBUG"
    },
    "verbose": {
      "handlers": ["file", "console"],
      "level": "INFO"
    },
    "standard": {
      "handlers": ["file"],
      "level": "INFO"
    },
    "requests": {
      "handlers": ["file", "console"],
      "level": "ERROR"
    },
    "elasticsearch": {
```

```
        "handlers": ["file", "console"],
        "level": "ERROR"
    },
    "elasticsearch.trace": {
        "handlers": ["file", "console"],
        "level": "ERROR"
    }
}
```

and start Glances using the following command line:

```
LOG_CFG=<path>/glances.json glances
```

Note: Replace <path> by the folder where your `glances.json` file is hosted.

Anatomy Of The Application

This document is meant to give an overview of the Glances interface.

Legend:

GREEN	OK
BLUE	CAREFUL
MAGENTA	WARNING
RED	CRITICAL

Note: Only stats with colored background will be shown in the alert view.

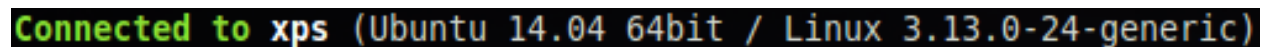
Header



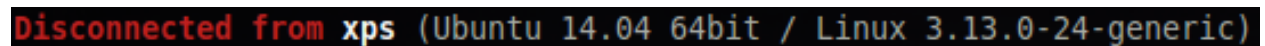
The header shows the hostname, OS name, release version, platform architecture IP addresses (private and public) and system uptime. Additionally, on GNU/Linux, it also shows the kernel version.

In client mode, the server connection status is also displayed.

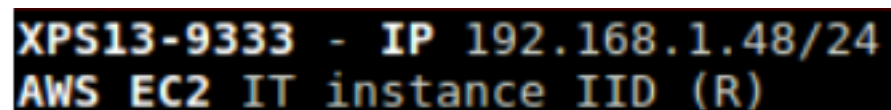
Connected:



Disconnected:



If you are hosted on an AWS EC2 instance, some additional information can be displayed (AMI-ID, region).



Quick Look

The `quicklook` plugin is only displayed on wide screen and proposes a bar view for CPU and memory (virtual and swap).

```
Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz - 0.77/1.80GHz
CPU [|||||] 8.6%
MEM [|||||] 24.9%
SWAP [ ] 0.0%
```

If the per CPU mode is on (by clicking the `1` key):

```
Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz - 0.77/1.80GHz
CPU0 [|||||] 9.1%
CPU1 [|||||] 9.6%
CPU2 [|||||] 9.5%
CPU3 [|||||] 8.0%
MEM [|||||] 25.2%
SWAP [ ] 0.0%
```

Note: Limit values can be overwritten in the configuration file under the `[quicklook]` section.

CPU

The CPU stats are shown as a percentage or values and for the configured refresh time.

The total CPU usage is displayed on the first line.

```
CPU / 23.7%
user: 20.2%
system: 3.4%
idle: 75.0%
```

If enough horizontal space is available, extended CPU information are displayed.

```
CPU / 8.2% nice: 0.0% ctx_sw: 3364
user: 6.3% irq: 0.0% inter: 1440
system: 1.9% iowait: 0.5% sw_int: 626
idle: 91.2% steal: 0.0%
```

A character is also displayed just after the CPU header and shows the trend value:

Trend	Status
-	CPU value is equal to the mean of the six latests refreshes
\	CPU value is lower than the mean of the six latests refreshes
/	CPU value is higher than the mean of the six latests refreshes

CPU stats description:

- **user:** percent time spent in user space. User CPU time is the time spent on the processor running your program's code (or code in libraries).

- **system**: percent time spent in kernel space. System CPU time is the time spent running code in the Operating System kernel.
- **idle**: percent of CPU used by any program. Every program or task that runs on a computer system occupies a certain amount of processing time on the CPU. If the CPU has completed all tasks it is idle.
- **nice** (**nix*): percent time occupied by user level processes with a positive nice value. The time the CPU has spent running users' processes that have been *niced*.
- **irq** (*Linux*, **BSD*): percent time spent servicing/handling hardware/software interrupts. Time servicing interrupts (hardware + software).
- **iowait** (*Linux*): percent time spent by the CPU waiting for I/O operations to complete.
- **steal** (*Linux*): percentage of time a virtual CPU waits for a real CPU while the hypervisor is servicing another virtual processor.
- **ctx_sw**: number of context switches (voluntary + involuntary) per second. A context switch is a procedure that a computer's CPU (central processing unit) follows to change from one task (or process) to another while ensuring that the tasks do not conflict.
- **inter**: number of interrupts per second.
- **sw_inter**: number of software interrupts per second. Always set to 0 on Windows and SunOS.
- **syscal**: number of system calls per second. Do not displayed on Linux (always 0).

To switch to per-CPU stats, just hit the 1 key:

```
PER CPU    6.0%    8.0%    9.0%    4.0%
user:      2.8%    6.2%    6.5%    2.8%
system:    3.2%    1.2%    2.5%    1.2%
idle:     94.0%   92.0%   91.0%   96.0%
iowait:    0.0%    0.6%    0.0%    0.0%
steal:     0.0%    0.0%    0.0%    0.0%
```

By default, steal CPU time alerts aren't logged. If you want that, just add to the configuration file:

```
[cpu]
steal_log=True
```

Legend:

CPU (user/system)	Status
<50%	OK
>50%	CAREFUL
>70%	WARNING
>90%	CRITICAL

Note: Limit values can be overwritten in the configuration file under the [cpu] and/or [percpu] sections.

GPU

Note: You need to install the `nvidia-ml-py3` library on your system.

The GPU stats are shown as a percentage of value and for the configured refresh time. The total GPU usage is displayed on the first line, the memory consumption on the second one.

```
3 GPU GeForce GTX
proc mean: 47%
mem mean: 40%
```

If you click on the 6 short key, the per-GPU view is displayed:

```
3 GPU GeForce GTX
0: 61% mem: 49%
1: 80% mem: 71%
2: 0% mem: 0%
```

Note: You can also start Glances with the `--meangpu` option to display the first view by default.

You can change the threshold limits in the configuration file:

```
[gpu]
# Default processor values if not defined: 50/70/90
proc_careful=50
proc_warning=70
proc_critical=90
# Default memory values if not defined: 50/70/90
mem_careful=50
mem_warning=70
mem_critical=90
```

Legend:

GPU (PROC/MEM)	Status
<50%	OK
>50%	CAREFUL
>70%	WARNING
>90%	CRITICAL

Load

Availability: Unix

```
LOAD 4-core
1 min: 0.03
5 min: 0.17
15 min: 0.44
```

On the *No Sheep* blog, Zachary Tirrell defines the [load average](#) on GNU/Linux operating system:

“In short it is the average sum of the number of processes waiting in the run-queue plus the number currently executing over 1, 5, and 15 minutes time periods.”

Be aware that Load on Linux and BSD are different things, high S'load on BSD' _ does not means high CPU load.

Glances gets the number of CPU core to adapt the alerts. Alerts on load average are only set on 15 minutes time period. The first line also displays the number of CPU core.

Legend:

Load avg	Status
<0.7*core	OK
>0.7*core	CAREFUL
>1*core	WARNING
>5*core	CRITICAL

Note: Limit values can be overwritten in the configuration file under the [load] section.

Memory

Glances uses two columns: one for the RAM and one for the SWAP.

MEM -	48.5%	SWAP -	0.0%
total:	7.33G	total:	7.53G
used:	3.56G	used:	0
free:	3.78G	free:	7.53G

If enough space is available, Glances displays extended information for the RAM:

MEM -	49.0%	active:	4.03G	SWAP -	0.0%
total:	7.33G	inactive:	1.81G	total:	7.53G
used:	3.59G	buffers:	441M	used:	0
free:	3.74G	cached:	2.99G	free:	7.53G

A character is also displayed just after the MEM header and shows the trend value:

Trend	Status
-	MEM value is equal to the mean of the six latests refreshes
\	MEM value is lower than the mean of the six latests refreshes
/	MEM value is higher than the mean of the six latests refreshes

Alerts are only set for used memory and used swap.

Legend:

RAM/Swap	Status
<50%	OK
>50%	CAREFUL
>70%	WARNING
>90%	CRITICAL

Note: Limit values can be overwritten in the configuration file under the [memory] and/or [memswap] sections.

Network

NETWORK	Rx/s	Tx/s
docker0	0b	0b
wlan0	10Kb	17Kb

Glances displays the network interface bit rate. The unit is adapted dynamically (bit/s, kbit/s, Mbit/s, etc).

If the interface speed is detected (not on all systems), the defaults thresholds are applied (70% for careful, 80% warning and 90% critical). It is possible to define this percents thresholds from the configuration file. It is also possible to define per interface bit rate thresholds. In this case thresholds values are define in bps.

Additionally, you can define:

- a list of network interfaces to hide
- per-interface limit values
- aliases for interface name

The configuration should be done in the `[network]` section of the Glances configuration file.

For example, if you want to hide the loopback interface (lo) and all the virtual docker interface (docker0, docker1, ...):

```
[network]
# Default bitrate thresholds in % of the network interface speed
# Default values if not defined: 70/80/90
rx_careful=70
rx_warning=80
rx_critical=90
tx_careful=70
tx_warning=80
tx_critical=90
# Define the list of hidden network interfaces (comma-separated regexp)
hide=docker.*,lo
# WLAN 0 alias
wlan0_alias=Wireless IF
# It is possible to overwrite the bitrate thresholds per interface
# WLAN 0 Default limits (in bits per second aka bps) for interface bitrate
wlan0_rx_careful=4000000
wlan0_rx_warning=5000000
wlan0_rx_critical=6000000
wlan0_rx_log=True
wlan0_tx_careful=700000
wlan0_tx_warning=900000
wlan0_tx_critical=1000000
wlan0_tx_log=True
```

Wi-Fi

Availability: Linux

```
NETWORK      Rx/s    Tx/s
docker0      0b       0b
lo           0b       0b
wlp2s0       6Kb      480b

WIFI
CANDNWIFI wpa      -79

DefaultGateway 11ms
```

Glances displays the Wi-Fi hotspot names and signal quality. If Glances is ran as root, then all the available hotspots are displayed.

Note: You need to install the `wireless-tools` package on your system.

In the configuration file, you can define signal quality thresholds:

- "Poor" quality is between -100 and -85dBm
- "Good" quality between -85 and -60dBm
- "Excellent" between -60 and -40dBm

It's also possible to disable the scan on a specific interface from the configuration file ([`wifi`] section). For example, if you want to hide the loopback interface (`lo`) and all the virtual docker interfaces:

```
[wifi]
hide=lo,docker.*
# Define SIGNAL thresholds in dBm (lower is better...)
careful=-65
warning=-75
critical=-85
```

You can disable this plugin using the `--disable-wifi` option or by hitting the `W` key from the user interface.

Ports

Availability: All

```
DefaultGateway 11ms
Home Box       100ms
My ISP         102ms
Internet ICMP  302ms
Internet Web   100ms
My Blog        Code 200
github.com     Code 200
Google Fr      Code 200
Intranet       Code 404
```

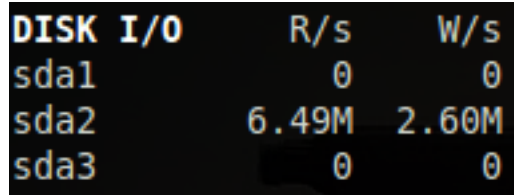
This plugin aims at providing a list of hosts/port and URL to scan.

You can define ICMP or TCP ports scans and URL (head only) check.

The list should be defined in the [ports] section of the Glances configuration file.

```
[ports]
# Ports scanner plugin configuration
# Interval in second between two scans
refresh=30
# Set the default timeout (in second) for a scan (can be overwrite in the scan list)
timeout=3
# If port_default_gateway is True, add the default gateway on top of the scan list
port_default_gateway=True
#
# Define the scan list (1 < x < 255)
# port_x_host (name or IP) is mandatory
# port_x_port (TCP port number) is optional (if not set, use ICMP)
# port_x_description is optional (if not set, define to host:port)
# port_x_timeout is optional and overwrite the default timeout value
# port_x_rtt_warning is optional and defines the warning threshold in ms
#
port_1_host=192.168.0.1
port_1_port=80
port_1_description=Home Box
port_1_timeout=1
port_2_host=www.free.fr
port_2_description=My ISP
port_3_host=www.google.com
port_3_description=Internet ICMP
port_3_rtt_warning=1000
port_4_host=www.google.com
port_4_description=Internet Web
port_4_port=80
port_4_rtt_warning=1000
#
# Define Web (URL) monitoring list (1 < x < 255)
# web_x_url is the URL to monitor (example: http://my.site.com/folder)
# web_x_description is optional (if not set, define to URL)
# web_x_timeout is optional and overwrite the default timeout value
# web_x_rtt_warning is optional and defines the warning respond time in ms_
↪ (approximatively)
#
web_1_url=https://blog.nicolargo.com
web_1_description=My Blog
web_1_rtt_warning=3000
web_2_url=https://github.com
web_3_url=http://www.google.fr
web_3_description=Google Fr
```

Disk I/O



DISK I/O	R/s	W/s
sda1	0	0
sda2	6.49M	2.60M
sda3	0	0

Glances displays the disk I/O throughput. The unit is adapted dynamically.

There is no alert on this information.

It's possible to define:

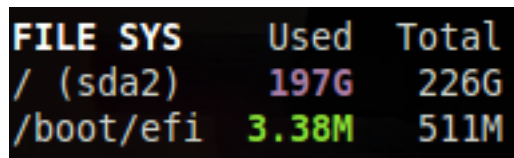
- a list of disks to hide
- aliases for disk name

under the `[diskio]` section in the configuration file.

For example, if you want to hide the loopback disks (loop0, loop1, ...) and the specific `sda5` partition:

```
[diskio]
hide=sda5,loop.*
```

File System



FILE SYS	Used	Total
/ (sda2)	197G	226G
/boot/efi	3.38M	511M

Glances displays the used and total file system disk space. The unit is adapted dynamically.

Alerts are set for used disk space.

Legend:

Disk usage	Status
<50%	OK
>50%	CAREFUL
>70%	WARNING
>90%	CRITICAL

Note: Limit values can be overwritten in the configuration file under the `[filesystem]` section.

By default, the plugin only displays physical devices (hard disks, USB keys). To allow other file system types, you have to enable them in the configuration file. For example, if you want to allow the `zfs` file system:

```
[fs]
allow=zfs
```

Also, you can hide mount points as well (in the following `/boot`):

```
[fs]
hide=/boot.*
```

RAID

Availability: Linux

Thanks to the `pymdstat` library, if a RAID controller is detected on you system, its status will be displayed as well:

```
RAID disks  Used  Avail
RAID1  md0    2     2
RAID1  md1    2     2
RAID1  md2    2     2
RAID5  md3   10    10
```

Folders

The folders plugin allows user, through the configuration file, to monitor size of a predefined folders list.

```
FOLDERS
/tmp                6.72M
_colargo/Videos    17.2G
/nonexisting        ?
```

If the size cannot be computed, a '?' (non-existing folder) or a '!' (permission denied) is displayed.

Each item is defined by:

- `path`: absolute path to monitor (mandatory)
- `careful`: optional careful threshold (in MB)
- `warning`: optional warning threshold (in MB)
- `critical`: optional critical threshold (in MB)

Up to 10 items can be defined.

For example, if you want to monitor the `/tmp` folder, the following definition should do the job:

```
[folders]
folder_1_path=/tmp
folder_1_careful=2500
folder_1_warning=3000
folder_1_critical=3500
```

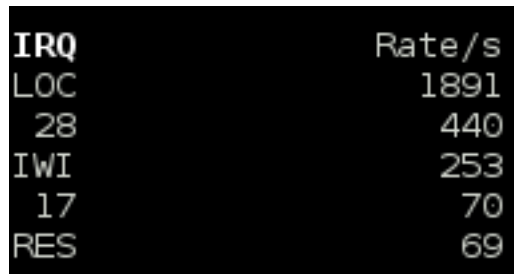
In client/server mode, the list is defined on the `server` side.

Warning: Do NOT define folders containing lot of files and subfolders.

IRQ

Availability: Linux

This plugin is disabled by default, please use the `--enable-irq` option to enable it.



IRQ	Rate/s
LOC	1891
28	440
IWI	253
17	70
RES	69

Glances displays the top 5 interrupts rate.

This plugin is only available on GNU/Linux (stats are grabbed from the `/proc/interrupts` file).

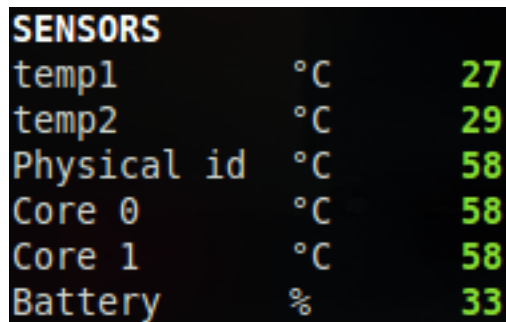
Note: `/proc/interrupts` file doesn't exist inside OpenVZ containers.

How to read the information:

- The first column is the IRQ number / name
- The second column says how many times the CPU has been interrupted during the last second

Sensors

Availability: Linux



SENSORS		
temp1	°C	27
temp2	°C	29
Physical id	°C	58
Core 0	°C	58
Core 1	°C	58
Battery	%	33

Glances can display the sensors information using `psutil` and/or `hddtemp`.

There is no alert on this information.

Note: Limit values and sensors alias names can be defined in the configuration file under the `[sensors]` section.

Processes List

Compact view:

```
TASKS 225 (562 thr), 1 run, 223 slp, 1 oth sorted automatically
```

CPU%	MEM%	PID	USER	NI	S	Command
4.1	0.2	27889	nicolargo	0	R	/home/nicolargo/virtualenvs/glances-de
3.2	3.4	1107	root	0	S	/usr/bin/X :0 -background none -verbo
1.6	0.9	22440	nicolargo	0	S	/usr/lib/firefox/plugin-container /usr
1.3	6.3	8411	nicolargo	0	S	/usr/bin/perl /usr/bin/shutter
1.0	4.7	22870	nicolargo	0	S	/usr/bin/gnome-shell
0.6	0.0	2112	nicolargo	0	S	syndaemon -i 1.0 -t -K -R
0.6	2.2	7042	nicolargo	0	S	vlc
0.6	11.4	7214	nicolargo	0	S	/usr/lib/firefox/firefox
0.3	0.6	303	nicolargo	0	S	nautilus --new-window
0.3	0.0	1033	root	0	S	/usr/sbin/irqbalance
0.3	0.0	1086	snmp	0	S	/usr/sbin/snmpd -Lsd -Lf /dev/null -u
0.3	0.6	4089	nicolargo	0	S	/usr/bin/python /usr/bin/terminator
0.0	0.1	1	root	0	S	/sbin/init
0.0	0.0	2	root	0	S	kthreadd
0.0	0.0	3	root	0	S	ksoftirqd/0
0.0	0.0	5	root	-20	S	kworker/0:0H

Full view:

```
TASKS 227 (578 thr), 1 run, 225 slp, 1 oth sorted automatically by cpu_percent
```

CPU%	MEM%	VIRT	RES	PID	USER	NI	S	TIME+	IOR/s	IOW/s	Command
4.4	0.2	79.0M	15.2M	27889	nicolargo	0	R	0:03.24	0	0	/home/nicolargo/virtualenvs/glances-develop/bin/
4.1	3.6	676M	284M	1107	root	0	S	5:09.56	0	0	/usr/bin/X :0 -background none -verbose -auth /
1.6	0.9	717M	70.8M	22440	nicolargo	0	S	1:16.40	0	0	/usr/lib/firefox/plugin-container /usr/lib/flash
1.6	4.7	2.10G	371M	22870	nicolargo	0	S	35:31.50	0	0	/usr/bin/gnome-shell
1.3	0.6	1.02G	47.2M	4089	nicolargo	0	S	0:49.50	0	0	/usr/bin/python /usr/bin/terminator
0.3	0.3	386M	27.5M	1982	nicolargo	0	S	8:37.84	0	0	/usr/bin/ibus-daemon --daemonize --xim
0.3	2.2	1.64G	176M	7042	nicolargo	0	S	23:44.30	0	0	vlc
0.3	6.5	1.98G	515M	8411	nicolargo	0	S	2:30.55	0	0	/usr/bin/perl /usr/bin/shutter
0.3	0.0	0	0	19741	root	0	S	0:01.75	0	0	kworker/0:0
0.3	0.0	0	0	26267	root	0	S	0:00.47	0	0	kworker/2:1
0.3	0.0	0	0	27127	root	0	S	0:00.11	0	0	kworker/u16:0
0.0	0.1	36.5M	6.45M	1	root	0	S	0:07.73	0	0	/sbin/init
0.0	0.0	0	0	2	root	0	S	0:00.80	0	0	kthreadd
0.0	0.0	0	0	3	root	0	S	0:01.32	0	0	ksoftirqd/0
0.0	0.0	0	0	5	root	-20	S	0:00.00	0	0	kworker/0:0H
0.0	0.0	0	0	7	root	0	S	1:05.30	0	0	rcu_sched

Filtered view:

```
Processes filter: .*firefox.* ('ENTER' to edit, 'E' to reset)
```

```
TASKS 2 (103 thr), 1 run, 1 slp, 0 oth sorted automatically by cpu_percent, flat view
```

CPU%	MEM%	VIRT	RES	PID	USER	NI	S	TIME+	R/s	W/s	Command
9.4	18.1	3.27G	1.40G	11378	nicolargo	0	S	6h41:25	0	0	/usr/lib/firefox/firefox
2.6	0.3	547M	25.4M	19801	nicolargo	0	R	0:03.00	0	0	python -m glances -f .*firefox.*
12.0	18.4	3.80G	1.42G						0	0	< current
8.4	18.4	3.80G	1.41G								< min ('M' to reset)
104.0	18.7	3.81G	1.44G								< max ('M' to reset)

The process view consists of 3 parts:

- Processes summary
- Monitored processes list (optional)
- Processes list

The processes summary line displays:

- Tasks number (total number of processes)
- Threads number
- Running tasks number
- Sleeping tasks number
- Other tasks number (not running or sleeping)
- Sort key

By default, or if you hit the `a` key, the processes list is automatically sorted by:

- CPU: if there is no alert (default behavior)
- CPU: if a CPU or LOAD alert is detected
- MEM: if a memory alert is detected
- DISK I/O: if a CPU iowait alert is detected

The number of processes in the list is adapted to the screen size.

Columns display

CPU%	% of CPU used by the process If Irix/Solaris mode is off, the value is divided by logical core number
MEM%	% of MEM used by the process
VIRT	Virtual Memory Size The total amount of virtual memory used by the process. It includes all code, data and shared libraries plus pages that have been swapped out and pages that have been mapped but not used.
RES	Resident Memory Size The non-swapped physical memory a process is using (what's currently in the physical memory).
PID	Process ID
USER	User ID
NI	Nice level of the process
S	Process status The status of the process: <ul style="list-style-type: none">• R: running or runnable (on run queue)• S: interruptible sleep (waiting for an event)• D: uninterruptible sleep (usually I/O)• Z: defunct ("zombie") process• T: traced/stopped by job control signal• X: dead (should never be seen)
TIME+	Cumulative CPU time used by the process
R/s	Per process I/O read rate in B/s
W/s	Per process I/O write rate in B/s
COMMAND	Process command line or command name User can switch to the process name by pressing on the <code>' / '</code> key

Process filtering

It's possible to filter the processes list using the ENTER key.

Filter syntax is the following (examples):

- `python`: Filter processes name or command line starting with *python* (regex)
- `.*python.*`: Filter processes name or command line containing *python* (regex)
- `username:nicolargo`: Processes of nicolargo user (key:regex)
- `cmdline:\usr\bin.*`: Processes starting by */usr/bin*

Extended info

```

CPU%  MEM%  VIRT  RES   PID USER      NI  S   TIME+ IOR/s IOW/s Command
42.7  13.8  2.47G 1.06G 3447 nicolargo  0   S   52:01.28  0  468K firefox
CPU affinity: 4 cores
Memory info: shared 28.2M text 104K lib 0 data 1.60G dirty 0 swap 0
Opened: threads 79 files 139 TCP 28 UDP 0
IO nice: No specific I/O priority

```

In standalone mode, additional information are provided for the top process:

CPU affinity	Number of cores used by the process
Memory info	Extended memory information about the process For example, on Linux: swap, shared, text, lib, data and dirty
Open	The number of threads, files and network sessions (TCP and UDP) used by the process
IO nice	The process I/O niceness (priority)

The extended stats feature can be enabled using the `--enable-process-extended` option (command line) or the `e` key (curses interface).

Note: Limit values can be overwritten in the configuration file under the `[process]` section.

Monitored Processes List

Warning: The monitored processes list feature has been removed. Use the new Application Monitoring Process (AMP) instead.

Applications Monitoring Process

Thanks to Glances and its AMP module, you can add specific monitoring to running processes. AMPs are defined in the Glances configuration file.

You can disable AMP using the `--disable-amps` option or pressing the `A` key.

Simple AMP

For example, a simple AMP that monitor the CPU/MEM of all Python processes can be defined as follows:

```
[amp_python]
enable=true
regex=.*python.*
refresh=3
```

Every 3 seconds (`refresh`) and if the `enable` key is true, Glances will filter the running processes list thanks to the `.*python.*` regular expression (`regex`).

The default behavior for an AMP is to display the number of matching processes, CPU and MEM:

A screenshot of a Glances UI snippet for the 'Python' AMP. It shows the word 'Python' in green, followed by the number '3' in green, and then 'CPU: 4.1% | MEM: 1.4%' in green. The background is black.

You can also define the minimum (`countmin`) and/or maximum (`countmax`) process number. For example:

```
[amp_python]
enable=true
regex=.*python.*
refresh=3
countmin=1
countmax=2
```

With this configuration, if the number of running Python scripts is higher than 2, then the AMP is displayed with a purple color (red if less than `countmin`):

A screenshot of a Glances UI snippet for the 'Python' AMP. It shows the word 'Python' in red, followed by the number '3' in red, and then 'CPU: 3.8% | MEM: 1.4%' in red. The background is black.

User defined AMP

If you need to execute a specific command line, you can use the `command` option. For example, if you want to display the Dropbox process status, you can define the following section in the Glances configuration file:

```
[amp_dropbox]
# Use the default AMP (no dedicated AMP Python script)
enable=true
regex=.*dropbox.*
refresh=3
one_line=false
command=dropbox status
countmin=1
```

The `dropbox status` command line will be executed and displayed in the Glances UI:

A screenshot of a Glances UI snippet for the 'Dropbox' AMP. It shows the word 'Dropbox' in green, followed by the number '1' in green, and then two lines of text in green: 'Synchronisation (1 fichier restant, 3 s restantes)' and 'Transfert de IMG_0132.JPG (148,9 Ko/s, 3 s restantes)'. The background is black.

You can force Glances to display the result in one line setting `one_line` to true.

Embedded AMP

Glances provides some specific AMP scripts (replacing the `command` line). You can write your own AMP script to fill your needs. AMP scripts are located in the `amps` folder and should be named `glances_*.py`. An AMP script

define an Amp class (GlancesAmp) with a mandatory update method. The update method call the `set_result` method to set the AMP return string. The return string is a string with one or more line (n between lines). To enable it, the configuration file section should be named `[amp_*]`.

For example, if you want to enable the Nginx AMP, the following definition should do the job (Nginx AMP is provided by the Glances team as an example):

```
[amp_nginx]
enable=true
regex=/usr/sbin/nginx
refresh=60
one_line=false
status_url=http://localhost/nginx_status
```

Here's the result:

```
Python      3      CPU: 3.8% | MEM: 1.4%
Xeyes       0      No running process
SystemV     1      Services running: 21 stopped: 28 upstart: 20
Nginx       1      Active connections: 1
              server accepts handled requests
              379 379 379
              Reading: 0 Writing: 1 Waiting: 0
Dropbox     1      A jour
```

In client/server mode, the AMP list is defined on the server side.

Logs

```
Warning or critical alerts (lasts 3 entries)
2014-05-31 11:26:58 (ongoing) - CPU_USER (Min:98.1 Mean:98.4 Max:98.7)
2014-05-31 11:25:28 (0:00:06) - WARNING on MEM (Min:71.0 Mean:74.0 Max:78.0)
2014-05-31 11:24:11 (0:00:32) - CRITICAL on CPU_USER (Min:70.4 Mean:94.5 Max:98.8)
```

A log messages list is displayed in the bottom of the screen if and only if:

- at least one WARNING or CRITICAL alert was occurred
- space is available in the bottom of the console/terminal

Each alert message displays the following information:

1. start datetime
2. duration if alert is terminated or *ongoing* if the alert is still in progress
3. alert name
4. {min,avg,max} values or number of running processes for monitored processes list alerts

Docker

If you use Docker, Glances can help you to monitor your containers. Glances uses the Docker API through the `docker-py` library.

CONTAINERS 2 (served by Docker 1.7.1)									
Name	Status	CPU%	MEM	IOR/s	IOW/s	Rx/s	Tx/s	Command	
_dbgrafana_grafana_1	Up 54 seconds	0.0	16.4M	0b	0b	0b	0b	/usr/sbin/grafana-server	
_bgrafana_influxdb_1	Up 55 seconds (Paused)	0.0	16.4M	0b	0b	0b	0b	/run.sh	

It is possible to define limits and actions from the configuration file under the `[docker]` section:

```
[docker]
# Global containers' thresholds for CPU and MEM (in %)
cpu_careful=50
cpu_warning=70
cpu_critical=90
mem_careful=20
mem_warning=50
mem_critical=70
# Per container thresholds
containername_cpu_careful=10
containername_cpu_warning=20
containername_cpu_critical=30
containername_cpu_critical_action=echo {{Image}} {{Id}} {{cpu}} > /tmp/container_{{name}}.alert
```

You can use all the variables (`{{foo}}`) available in the Docker plugin.

Actions

Glances can trigger actions on events.

By action, we mean all shell command line. For example, if you want to execute the `foo.py` script if the last 5 minutes load are critical then add the `_action` line to the Glances configuration file:

```
[load]
critical=5.0
critical_action=python /path/to/foo.py
```

All the stats are available in the command line through the use of the `{{mustache}}` syntax. Another example would be to create a log file containing used vs total disk space if a space trigger warning is reached:

```
[fs]
warning=70
warning_action=echo {{mnt_point}} {{used}}/{{size}} > /tmp/fs.alert
```

Note: You can use all the stats for the current plugin. See <https://github.com/nicolargo/glances/wiki/The-Glances-2.x-API-How-to> for the stats list.

It is also possible to repeat action until the end of the alert. Keep in mind that the command line is executed every refresh time so use with caution:

```
[load]
critical=5.0
critical_action_repeat=/home/myhome/bin/bipper.sh
```

Gateway To Other Services

Glances can exports stats to a CSV file. Also, it can act as a gateway to providing stats to multiple services (see list below).

CSV

It's possible to export stats to a CSV file.

```
$ glances --export-csv /tmp/glances.csv
```

CSV file description:

- Stats description (first line)
- Stats (other lines)

JSON

It's possible to export stats to a JSON file.

```
$ glances --export-json /tmp/glances.json
```

Cassandra

You can export statistics to a Cassandra or Scylla server. The connection should be defined in the Glances configuration file as following:

```
[cassandra]
host=localhost
port=9042
protocol_version=3
keyspace=glances
replication_factor=2
table=localhost
```

and run Glances with:

```
$ glances --export-cassandra
```

The data model is the following:

```
CREATE TABLE <table> (plugin text, time timeuuid, stat map<text,float>, PRIMARY KEY
↪ (plugin, time))
```

Only numerical stats are stored in the Cassandra table. All the stats are converted to float. If a stat cannot be converted to float, it is not stored in the database.

CouchDB

You can export statistics to a CouchDB server. The connection should be defined in the Glances configuration file as following:

```
[couchdb]
host=localhost
port=5984
user=root
password=root
db=glances
```

and run Glances with:

```
$ glances --export-couchdb
```

Documents are stored in native JSON format. Glances adds "type" and "time" entries:

- type: plugin name
- time: timestamp (format: "2016-09-24T16:39:08.524828Z")

Example of Couch Document for the load stats:

```
{
  "_id": "36cbbad81453c53ef08804cb2612d5b6",
  "_rev": "1-382400899bec5615cabb99aa34df49fb",
  "min15": 0.33,
  "time": "2016-09-24T16:39:08.524828Z",
  "min5": 0.4,
  "cpucore": 4,
  "load_warning": 1,
  "min1": 0.5,
  "history_size": 28800,
  "load_critical": 5,
  "type": "load",
  "load_careful": 0.7
}
```

You can view the result using the CouchDB utils URL: http://127.0.0.1:5984/_utils/database.html?glances.

Elasticsearch

You can export statistics to an Elasticsearch server. The connection should be defined in the Glances configuration file as following:

```
[elasticsearch]
host=localhost
port=9200
index=glances
```

and run Glances with:

```
$ glances --export-elasticsearch
```

The stats are available through the elasticsearch API. For example, to get the CPU system stats:

```
$ curl http://172.17.0.2:9200/glances/cpu/system
{
  "_index": "glances",
  "_type": "cpu",
  "_id": "system",
```

```
{
  "_version": 28,
  "found": true,
  "_source": {
    "timestamp": "2016-02-04T14:11:02.362232",
    "value": "2.2"
  }
}
```

InfluxDB

You can export statistics to an InfluxDB server (time series server). The connection should be defined in the Glances configuration file as following:

```
[influxdb]
host=localhost
port=8086
user=root
password=root
db=glances
tags=foo:bar,spam:eggs
```

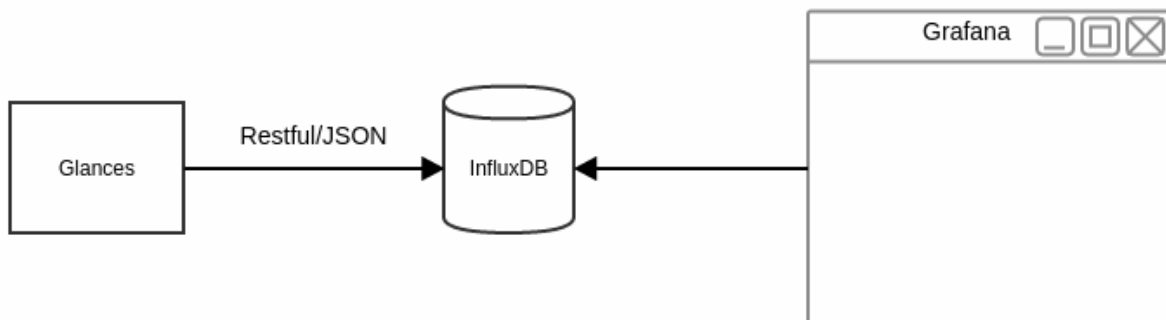
and run Glances with:

```
$ glances --export-influxdb
```

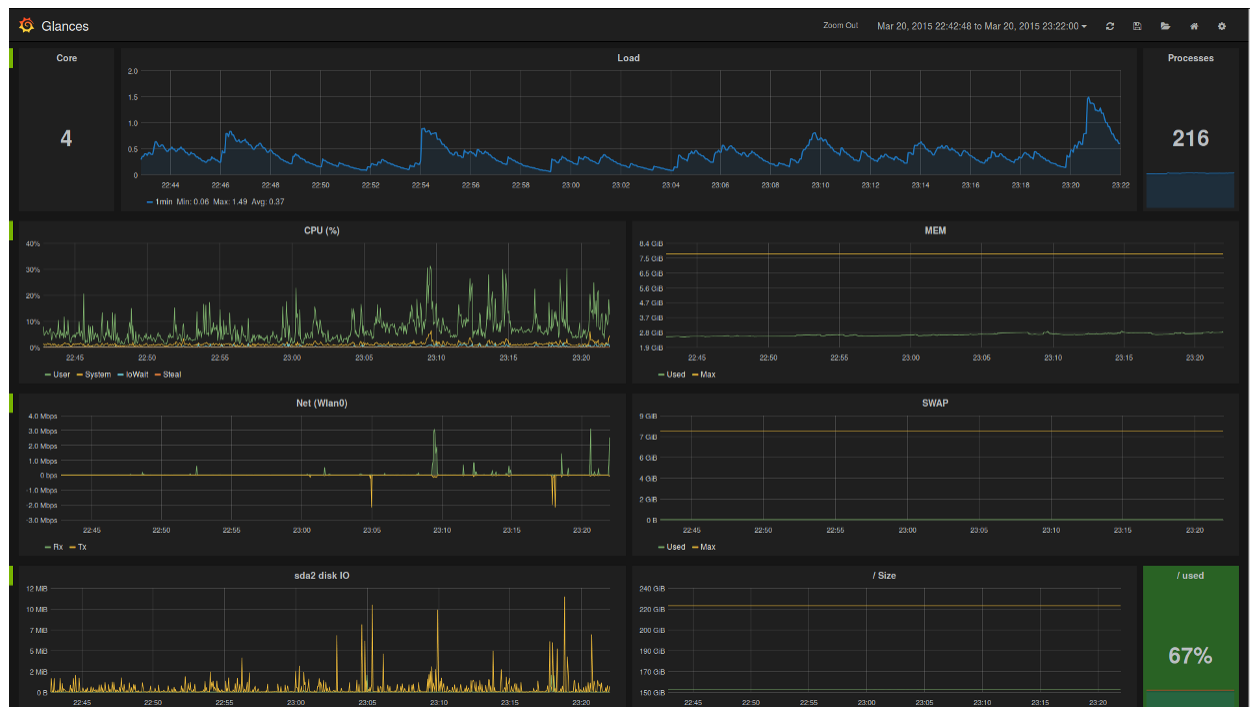
Glances generates a lot of columns, e.g., if you have many running Docker containers, so you should use the `tsml` engine in the InfluxDB configuration file (no limit on columns number).

Grafana

For Grafana users, Glances provides a dedicated [dashboard](#).



To use it, just import the file in your Grafana web interface.



Kafka

You can export statistics to a Kafka server. The connection should be defined in the Glances configuration file as following:

```
[kafka]
host=localhost
port=9092
topic=glances
#compression=gzip
```

Note: you can enable the compression but it consume CPU on your host.

and run Glances with:

```
$ glances --export-kafka
```

Stats are sent in native JSON format to the topic:

- key: plugin name
- value: JSON dict

Example of record for the memory plugin:

```
ConsumerRecord(topic=u'glances', partition=0, offset=1305, timestamp=1490460592248,
↳ timestamp_type=0, key='mem', value=u'{"available": 2094710784, "used": 5777428480,
↳ "cached": 2513543168, "mem_careful": 50.0, "percent": 73.4, "free": 2094710784,
↳ "mem_critical": 90.0, "inactive": 2361626624, "shared": 475504640, "history_size":
↳ 28800.0, "mem_warning": 70.0, "total": 7872139264, "active": 4834361344, "buffers":
↳ 160112640}', checksum=214895201, serialized_key_size=3, serialized_value_size=303)
```

Python code example to consume Kafka Glances plugin:


```
from kafka import KafkaConsumer
import json

consumer = KafkaConsumer('glances', value_deserializer=json.loads)
for s in consumer:
    print s
```

OpenTSDB

You can export statistics to an OpenTSDB server (time series server). The connection should be defined in the Glances configuration file as following:

```
[opentsdb]
host=localhost
port=4242
prefix=glances
tags=foo:bar,spam:eggs
```

and run Glances with:

```
$ glances --export-opentsdb
```

Prometheus

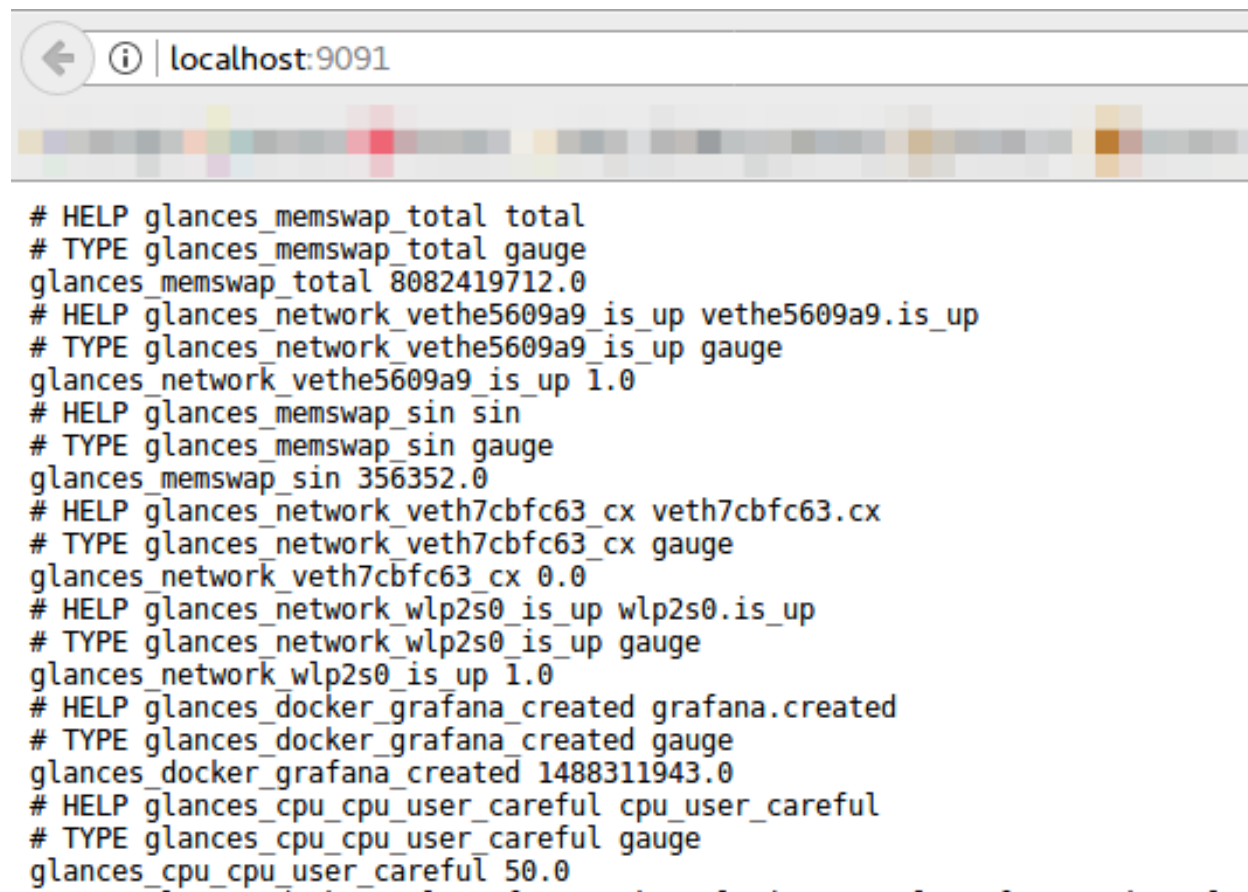
You can export statistics to a Prometheus server through an exporter. When the *--export-prometheus* is used, Glances creates a Prometheus exporter listening on <host:port> (define in the Glances configuration file).

```
[prometheus]
host=localhost
port=9091
prefix=glances
```

and run Glances with:

```
$ glances --export-prometheus
```

You can check that Glances exports the stats using this URL: <http://localhost:9091>



```
# HELP glances_memswap_total total
# TYPE glances_memswap_total gauge
glances_memswap_total 8082419712.0
# HELP glances_network_vethe5609a9_is_up vethe5609a9.is_up
# TYPE glances_network_vethe5609a9_is_up gauge
glances_network_vethe5609a9_is_up 1.0
# HELP glances_memswap_sin sin
# TYPE glances_memswap_sin gauge
glances_memswap_sin 356352.0
# HELP glances_network_veth7cbfc63_cx veth7cbfc63.cx
# TYPE glances_network_veth7cbfc63_cx gauge
glances_network_veth7cbfc63_cx 0.0
# HELP glances_network_wlp2s0_is_up wlp2s0.is_up
# TYPE glances_network_wlp2s0_is_up gauge
glances_network_wlp2s0_is_up 1.0
# HELP glances_docker_grafana_created grafana.created
# TYPE glances_docker_grafana_created gauge
glances_docker_grafana_created 1488311943.0
# HELP glances_cpu_cpu_user_careful cpu_user_careful
# TYPE glances_cpu_cpu_user_careful gauge
glances_cpu_cpu_user_careful 50.0
```

In order to store the metrics in a Prometheus server, you should add this exporter to your Prometheus server configuration with the following lines (in the `prometheus.yml` configuration file):

```
scrape_configs:
  - job_name: 'glances_exporter'
    scrape_interval: 5s
    static_configs:
      - targets: ['localhost:9091']
```



RabbitMQ

You can export statistics to an RabbitMQ server (AMQP Broker). The connection should be defined in the Glances configuration file as following:

```
[rabbitmq]
host=localhost
port=5672
user=glances
password=glances
queue=glances_queue
```

and run Glances with:

```
$ glances --export-rabbitmq
```

Restful

You can export statistics to a Restful JSON server. All the available stats will be exported in one big (~15 KB) POST request to the Restful endpoint.

The Restful endpoint should be defined in the Glances configuration file as following:

```
[restful]
# Configuration for the --export-restful option
# Example, export to http://localhost:6789/
host=localhost
port=6789
protocol=http
path=
```

URL Syntax:

```
http://localhost:6789/
|      |      |      |
```

```
|         |         | path
|         |         | port
|         | host
protocol
```

and run Glances with:

```
$ glances --export-restful
```

Glances will generate stats as a big JSON dictionary (see example [here](#)).

Riemann

You can export statistics to a Riemann server (using TCP protocol). The connection should be defined in the Glances configuration file as following:

```
[riemann]
host=localhost
port=5555
```

and run Glances with:

```
$ glances --export-riemann
```

StatsD

You can export statistics to a StatsD server (welcome to Graphite!). The connection should be defined in the Glances configuration file as following:

```
[statsd]
host=localhost
port=8125
prefix=glances
```

Note: The prefix is optional (glances by default)

and run Glances with:

```
$ glances --export-statsd
```

Glances will generate stats as:

```
'glances.cpu.user': 12.5,
'glances.cpu.total': 14.9,
'glances.load.cpucore': 4,
'glances.load.min1': 0.19,
...
```

ZeroMQ

You can export statistics to a ZeroMQ server.

The connection should be defined in the Glances configuration file as following:

```
[zeromq]
host=127.0.0.1
port=5678
prefix=G
```

Glances **envelopes** the stats before publishing it. The message is composed of three frames:

1. the prefix configured in the [zeromq] section (as STRING)
2. the Glances plugin name (as STRING)
3. the Glances plugin stats (as JSON)

Run Glances with:

```
$ glances --export-zeromq -C <path>/glances.conf
```

Following is a simple Python client to subscribe to the Glances stats:

```
# -*- coding: utf-8 -*-
#
# ZeroMQ subscriber for Glances
#

import json
import zmq

context = zmq.Context()

subscriber = context.socket(zmq.SUB)
subscriber.setsockopt(zmq.SUBSCRIBE, 'G')
subscriber.connect("tcp://127.0.0.1:5678")

while True:
    _, plugin, data_raw = subscriber.recv_multipart()
    data = json.loads(data_raw)
    print('{} => {}'.format(plugin, data))

subscriber.close()
context.term()
```

API Documentation

Glances provides an **XML-RPC** server and a **RESTful-JSON** API which can be used by other clients.

API documentation is available at:

- XML-RPC: <https://github.com/nicolargo/glances/wiki/The-Glances-2.x-API-How-to>
- RESTful-JSON: <https://github.com/nicolargo/glances/wiki/The-Glances-RESTFULL-JSON-API>

Support

To post a question about Glances use cases, please post it to the official Q&A [forum](#).

To report a bug or a feature request use the GitHub [issue](#) tracker.

Feel free to contribute!

Symbols

- browser
 - command line option, 10
- cached-time CACHED_TIME
 - command line option, 11
- disable-alert
 - command line option, 8
- disable-amps
 - command line option, 8
- disable-autodiscover
 - command line option, 10
- disable-bg
 - command line option, 9
- disable-bold
 - command line option, 9
- disable-check-update
 - command line option, 11
- disable-cpu
 - command line option, 8
- disable-diskio
 - command line option, 8
- disable-docker
 - command line option, 8
- disable-folders
 - command line option, 8
- disable-fs
 - command line option, 8
- disable-hddtemp
 - command line option, 8
- disable-ip
 - command line option, 8
- disable-irq
 - command line option, 8
- disable-load
 - command line option, 8
- disable-mem
 - command line option, 8
- disable-memswap
 - command line option, 8
- disable-network
 - command line option, 8
- disable-now
 - command line option, 8
- disable-ports
 - command line option, 8
- disable-process
 - command line option, 9
- disable-raid
 - command line option, 9
- disable-sensors
 - command line option, 9
- disable-wifi
 - command line option, 9
- diskio-iops
 - command line option, 11
- diskio-show-ramfs
 - command line option, 11
- enable-history
 - command line option, 9
- enable-process-extended
 - command line option, 9
- export-cassandra
 - command line option, 9
- export-couchdb
 - command line option, 9
- export-csv EXPORT_CSV
 - command line option, 9
- export-elasticsearch
 - command line option, 9
- export-graph
 - command line option, 9
- export-influxdb
 - command line option, 10
- export-opentsdb
 - command line option, 10
- export-rabbitmq
 - command line option, 10
- export-riemann
 - command line option, 10

- export-statsd
 - command line option, 10
- export-zeromq
 - command line option, 10
- fahrenheit
 - command line option, 11
- fs-free-space
 - command line option, 11
- hide-kernel-threads
 - command line option, 11
- password
 - command line option, 10
- path-graph PATH_GRAPH
 - command line option, 9
- process-short-name
 - command line option, 11
- snmp-auth SNMP_AUTH
 - command line option, 10
- snmp-community SNMP_COMMUNITY
 - command line option, 10
- snmp-force
 - command line option, 10
- snmp-port SNMP_PORT
 - command line option, 10
- snmp-user SNMP_USER
 - command line option, 10
- snmp-version SNMP_VERSION
 - command line option, 10
- theme-white
 - command line option, 11
- tree
 - command line option, 11
- username
 - command line option, 10
- 0, -disable-irix
 - command line option, 9
- 1, -percpu
 - command line option, 9
- 2, -disable-left-sidebar
 - command line option, 9
- 3, -disable-quicklook
 - command line option, 9
- 4, -full-quicklook
 - command line option, 9
- 5, -disable-top
 - command line option, 9
- 6, -meangpu
 - command line option, 9
- B BIND_ADDRESS, -bind BIND_ADDRESS
 - command line option, 10
- C CONF_FILE, -config CONF_FILE
 - command line option, 8
- V, -version
 - command line option, 8

- b, -byte
 - command line option, 11
- c CLIENT, -client CLIENT
 - command line option, 10
- d, -debug
 - command line option, 8
- f PROCESS_FILTER, -process-filter PROCESS_FILTER
 - command line option, 11
- h, -help
 - command line option, 8
- p PORT, -port PORT
 - command line option, 10
- q, -quiet
 - command line option, 11
- s, -server
 - command line option, 10
- t TIME, -time TIME
 - command line option, 10
- w, -webserver
 - command line option, 11

C

- command line option
 - browser, 10
 - cached-time CACHED_TIME, 11
 - disable-alert, 8
 - disable-amps, 8
 - disable-autodiscover, 10
 - disable-bg, 9
 - disable-bold, 9
 - disable-check-update, 11
 - disable-cpu, 8
 - disable-diskio, 8
 - disable-docker, 8
 - disable-folders, 8
 - disable-fs, 8
 - disable-hddtemp, 8
 - disable-ip, 8
 - disable-irq, 8
 - disable-load, 8
 - disable-mem, 8
 - disable-memswap, 8
 - disable-network, 8
 - disable-now, 8
 - disable-ports, 8
 - disable-process, 9
 - disable-raid, 9
 - disable-sensors, 9
 - disable-wifi, 9
 - diskio-iops, 11
 - diskio-show-ramfs, 11
 - enable-history, 9
 - enable-process-extended, 9

- export-cassandra, 9
- export-couchdb, 9
- export-csv EXPORT_CSV, 9
- export-elasticsearch, 9
- export-graph, 9
- export-influxdb, 10
- export-opentsdb, 10
- export-rabbitmq, 10
- export-riemann, 10
- export-statsd, 10
- export-zeromq, 10
- fahrenheit, 11
- fs-free-space, 11
- hide-kernel-threads, 11
- password, 10
- path-graph PATH_GRAPH, 9
- process-short-name, 11
- snmp-auth SNMP_AUTH, 10
- snmp-community SNMP_COMMUNITY, 10
- snmp-force, 10
- snmp-port SNMP_PORT, 10
- snmp-user SNMP_USER, 10
- snmp-version SNMP_VERSION, 10
- theme-white, 11
- tree, 11
- username, 10
- 0, -disable-irix, 9
- 1, -percpu, 9
- 2, -disable-left-sidebar, 9
- 3, -disable-quicklook, 9
- 4, -full-quicklook, 9
- 5, -disable-top, 9
- 6, -meangpu, 9
- B BIND_ADDRESS, -bind BIND_ADDRESS, 10
- C CONF_FILE, -config CONF_FILE, 8
- V, -version, 8
- b, -byte, 11
- c CLIENT, -client CLIENT, 10
- d, -debug, 8
- f PROCESS_FILTER, -process-filter PROCESS_FILTER, 11
- h, -help, 8
- p PORT, -port PORT, 10
- q, -quiet, 11
- s, -server, 10
- t TIME, -time TIME, 10
- w, -webserver, 11
- open-web-browser, 11

O

open-web-browser
 command line option, 11