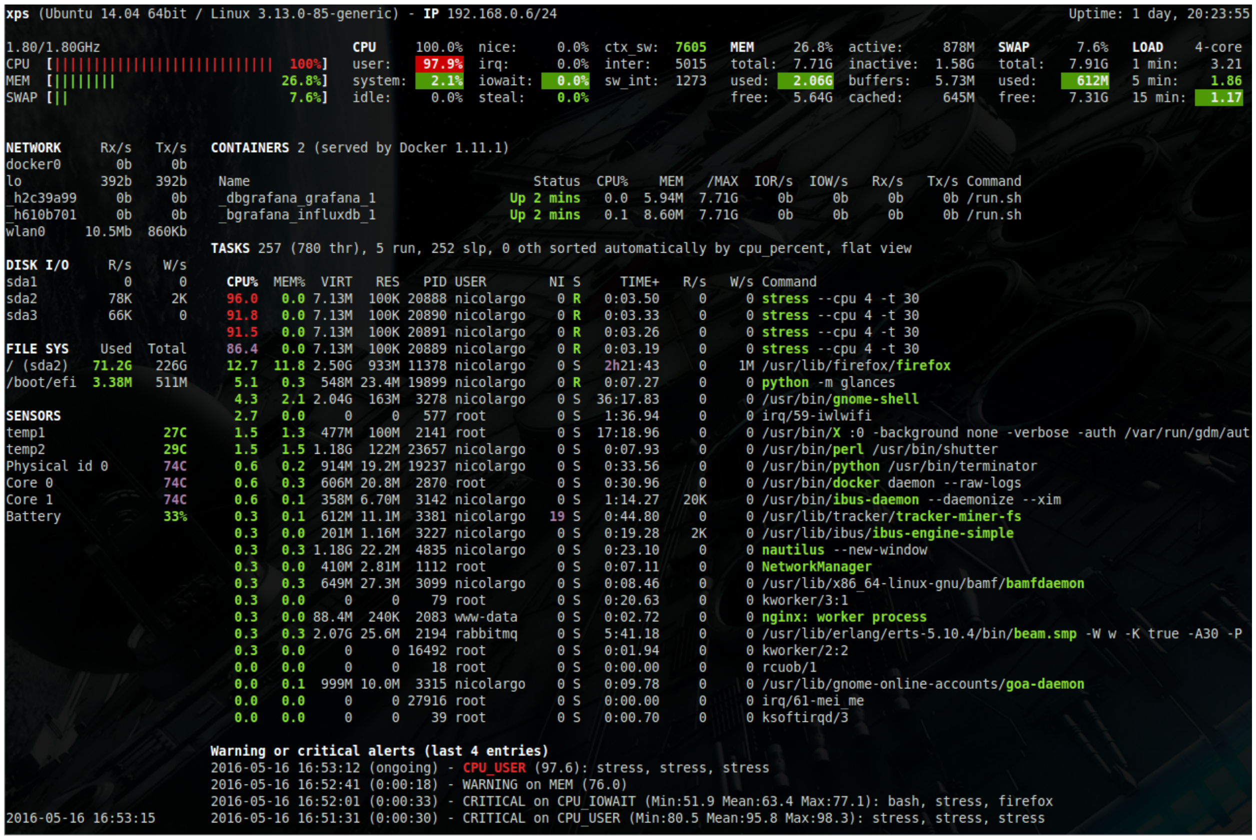
Glances Documentation

Release 3.4.0.3

Nicolas Hennion

Jul 08, 2023

Contents



```
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     100.0%  nice:      0.0%  ctx_sw:  7605   MEM     26.8%  active:      878M   SWAP     7.6%   LOAD    4-core
CPU  [|||||||||||||||||||||||||||||| 100%]  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    Name                                     Status  CPU%    MEM   /MAX  IOR/s  IOW/s   Rx/s   Tx/s Command
_h2c39a99     0b     0b    _dbgrafana_grafana_1                  Up 2 mins   0.0  5.94M  7.71G     0b     0b     0b     0b /run.sh
_h610b701     0b     0b    _bgrafana_influxdb_1                  Up 2 mins   0.1  8.60M  7.71G     0b     0b     0b     0b /run.sh
wlan0     10.5Mb  860Kb
                          TASKS 257 (780 thr), 5 run, 252 slp, 0 oth sorted automatically by cpu_percent, flat view
DISK I/O     R/s    W/s
sda1           0      0     CPU%  MEM%  VIRT   RES   PID USER         NI S    TIME+   R/s   W/s Command
sda2         78K     2K     96.0   0.0 7.13M  100K 20888 nicolargo     0 R  0:03.50     0     0 stress --cpu 4 -t 30
sda3         66K      0     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
FILE SYS    Used  Total     86.4   0.0 7.13M  100K 20889 nicolargo     0 R  0:03.19     0     0 stress --cpu 4 -t 30
/ (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
SENSORS                      2.7   0.0     0     0   577 root          0 S  1:36.94     0     0 irq/59-iwlwifi
temp1                27C     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
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     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     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 16:53:15       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.

CHAPTER 1

Table of Contents

1.1 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[all]
```

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

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

For additional installation methods, read the official README file.

1.2 Quickstart

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

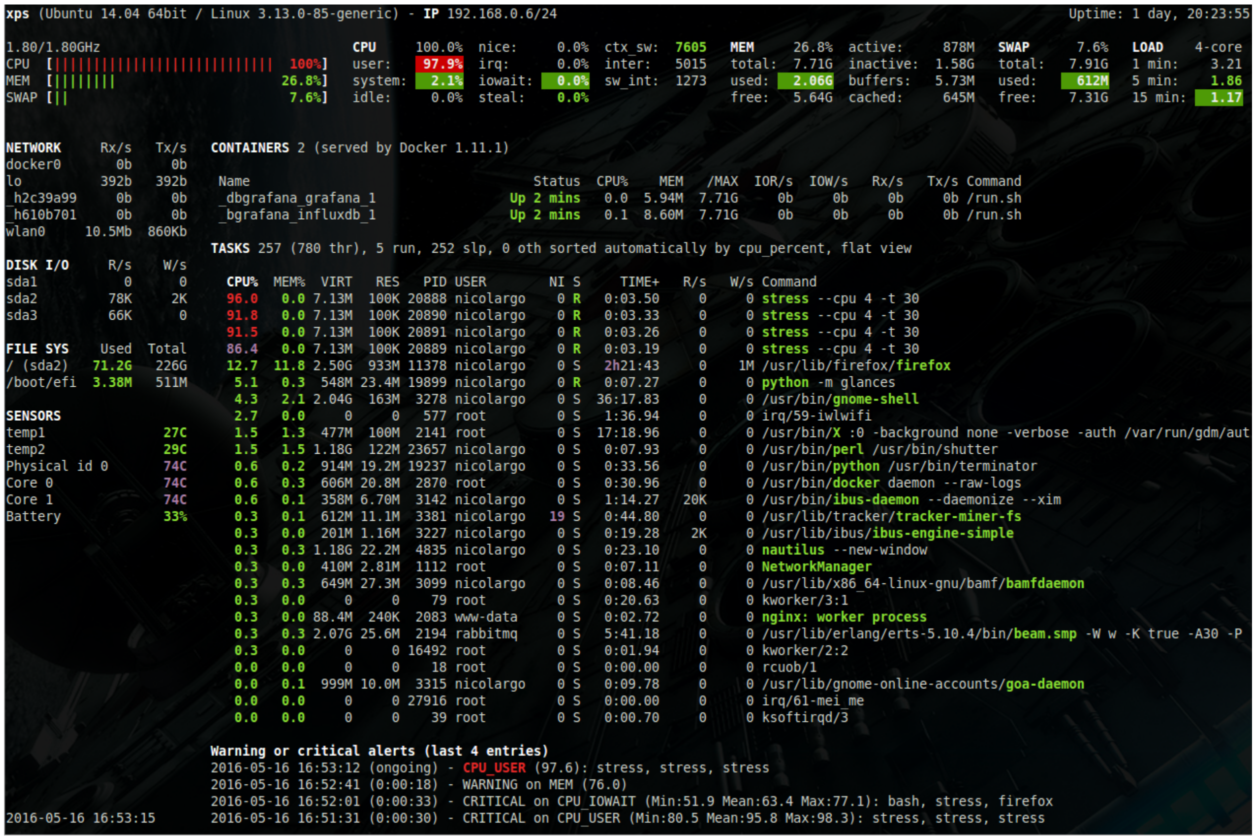
- Standalone
- Client/Server
- Web server

1.2.1 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     100.0%  nice:     0.0%  ctx_sw:  7605  MEM     26.8%  active:    878M  SWAP    7.6%  LOAD    4-core
CPU  [|||||||||||||||||||||||||||  100%]  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    Name                                Status  CPU%    MEM   /MAX  IOR/s  IOW/s   Rx/s    Tx/s Command
 h2c39a99     0b     0b    _dbgrafana_grafana_1             Up 2 mins   0.0  5.94M  7.71G     0b     0b     0b      0b /run.sh
 h610b701     0b     0b    _bgrafana_influxdb_1             Up 2 mins   0.1  8.60M  7.71G     0b     0b     0b      0b /run.sh
wlan0     10.5Mb  860Kb
                          TASKS 257 (780 thr), 5 run, 252 slp, 0 oth sorted automatically by cpu_percent, flat view
DISK I/O     R/s    W/s
sda1           0      0     CPU%  MEM%  VIRT   RES   PID USER        NI S    TIME+   R/s    W/s Command
sda2         78K     2K     96.0   0.0 7.13M  100K 20888 nicolargo    0 R  0:03.50     0      0 stress --cpu 4 -t 30
sda3         66K      0     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
FILE SYS     Used  Total    86.4   0.0 7.13M  100K 20889 nicolargo    0 R  0:03.19     0      0 stress --cpu 4 -t 30
/ (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
SENSORS                      2.7   0.0     0     0   577 root         0 S  1:36.94     0      0 irq/59-iwlwifi
temp1                 27C    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
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     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     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 16:53:15       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
```

It is also possible to display RAW (Python) stats directly to stdout using:

```
$ glances --stdout cpu.user,mem.used,load
cpu.user: 30.7
mem.used: 3278204928
load: {'cpucore': 4, 'min1': 0.21, 'min5': 0.4, 'min15': 0.27}
cpu.user: 3.4
mem.used: 3275251712
load: {'cpucore': 4, 'min1': 0.19, 'min5': 0.39, 'min15': 0.27}
...
```

or in a CSV format thanks to the stdout-csv option:

```
$ glances --stdout-csv now,cpu.user,mem.used,load
now,cpu.user,mem.used,load.cpucore,load.min1,load.min5,load.min15
2018-12-08 22:04:20 CEST,7.3,5948149760,4,1.04,0.99,1.04
2018-12-08 22:04:23 CEST,5.4,5949136896,4,1.04,0.99,1.04
...
```

or as a JSON format thanks to the stdout-json option (attribute not supported in this mode):

```
$ glances --stdout-json cpu,mem
cpu: {"total": 29.0, "user": 24.7, "nice": 0.0, "system": 3.8, "idle": 71.4, "iowait
↪": 0.0, "irq": 0.0, "softirq": 0.0, "steal": 0.0, "guest": 0.0, "guest_nice": 0.0,
↪"time_since_update": 1, "cpucore": 4, "ctx_switches": 0, "interrupts": 0, "soft_
↪interrupts": 0, "syscalls": 0}
mem: {"total": 7837949952, "available": 2919079936, "percent": 62.8, "used":␣
↪4918870016, "free": 2919079936, "active": 2841214976, "inactive": 3340550144,
↪"buffers": 546799616, "cached": 3068141568, "shared": 788156416}
...
```

Note: It will display one line per stat per refresh.

1.2.2 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.

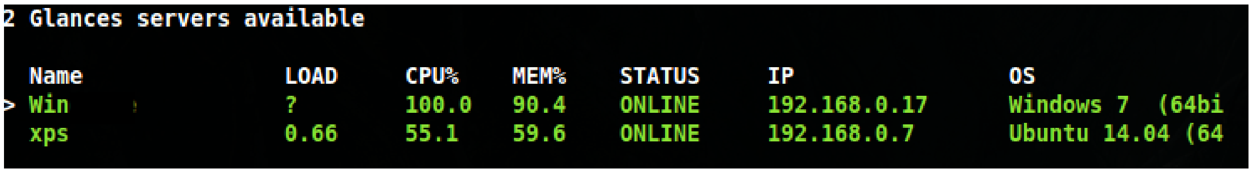
In server mode, you can set the bind address with `-B ADDRESS` and the listening TCP port with `-p PORT`.

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.

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.

When the list is displayed, you can navigate through the Glances servers with up/down keys. It is also possible to sort the server using: - '1' is normal (do not sort) - '2' is using sorting with ascending order (ONLINE > SNMP > PROTECTED > OFFLINE > UNKNOWN) - '3' is using sorting with descending order (UNKNOWN > OFFLINE > PROTECTED > SNMP > ONLINE)

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:

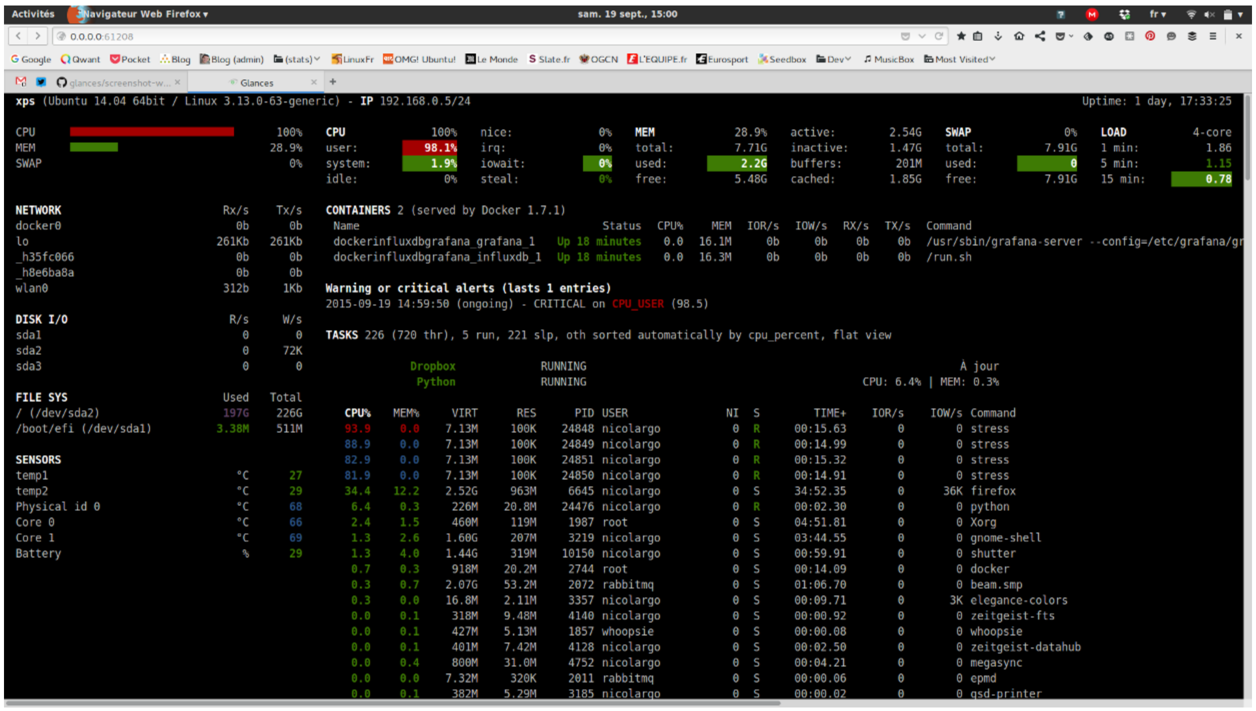
```
client$ glances -c @snmpserver
```

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.

1.2.3 Web Server Mode



```
xps (Ubuntu 14.04 64bit / Linux 3.13.0-63-generic) - IP 192.168.0.5/24                                              Uptime: 1 day, 17:33:25

CPU                              100%  CPU        100%  nice:       0%  MEM     28.9%  active:    2.54G  SWAP       0%  LOAD    4-core
MEM                             28.9%  user:     98.1%  irq:        0%  total:  7.71G  inactive:  1.47G  total:  7.91G  1 min:    1.86
SWAP                               0%  system:    1.9%  iowait:     0%  used:    2.2G  buffers:    201M  used:       0  5 min:    1.15
                                       idle:        0%  steal:      0%  free:   5.48G  cached:    1.85G  free:   7.91G  15 min:   0.78

NETWORK                 Rx/s     Tx/s  CONTAINERS 2 (served by Docker 1.7.1)
docker0                   0b       0b   Name                                    Status   CPU%   MEM  IOR/s  IOW/s  RX/s  TX/s  Command
lo                     261Kb    261Kb   dockerinfluxdbgrafana_grafana_1   Up 18 minutes   0.0  16.1M    0b     0b    0b    0b  /usr/sbin/grafana-server --config=/etc/grafana/gr
_h35fc066                 0b       0b   dockerinfluxdbgrafana_influxdb_1  Up 18 minutes   0.0  16.3M    0b     0b    0b    0b  /run.sh
_h8e6ba8a                 0b       0b
wlan0                   312b      1Kb  Warning or critical alerts (lasts 1 entries)
                                       2015-09-19 14:59:50 (ongoing) - CRITICAL on CPU_USER (98.5)
DISK I/O                 R/s      W/s
sda1                       0        0  TASKS 226 (720 thr), 5 run, 221 slp, oth sorted automatically by cpu_percent, flat view
sda2                       0      72K
sda3                       0        0               Dropbox             RUNNING                                          À jour
                                                     Python             RUNNING                                CPU: 6.4% | MEM: 0.3%
FILE SYS                Used    Total
/ (/dev/sda2)           197G     226G     CPU%   MEM%    VIRT    RES    PID USER          NI  S      TIME+   IOR/s   IOW/s Command
/boot/efi (/dev/sda1)  3.38M     511M     93.9    0.0   7.13M   100K  24848 nicolargo      0  R   00:15.63       0       0 stress
                                          88.9    0.0   7.13M   100K  24849 nicolargo      0  R   00:14.99       0       0 stress
SENSORS                                   82.9    0.0   7.13M   100K  24851 nicolargo      0  R   00:15.32       0       0 stress
temp1                     °C       27     81.9    0.0   7.13M   100K  24850 nicolargo      0  R   00:14.91       0       0 stress
temp2                     °C       29     34.4   12.2   2.52G   963M   6645 nicolargo      0  S   34:52.35       0     36K firefox
Physical id 0             °C       68      6.4    0.3    226M  20.8M  24476 nicolargo      0  R   00:02.30       0       0 python
Core 0                    °C       66      2.4    1.5    460M   119M   1987 root           0  S   04:51.81       0       0 Xorg
Core 1                    °C       69      1.3    2.6   1.60G   207M   3219 nicolargo      0  S   03:44.55       0       0 gnome-shell
Battery                    %       29      1.3    4.0   1.44G   319M  10150 nicolargo      0  S   00:59.91       0       0 shutter
                                           0.7    0.3    918M  20.2M   2744 root           0  S   00:14.09       0       0 docker
                                           0.3    0.7   2.07G  53.2M   2072 rabbitmq       0  S   01:06.70       0       0 beam.smp
                                           0.3    0.0   16.8M  2.11M   3357 nicolargo      0  S   00:09.71       0      3K elegance-colors
                                           0.0    0.1    318M  9.48M   4140 nicolargo      0  S   00:00.92       0       0 zeitgeist-fts
                                           0.0    0.1    427M  5.13M   1857 whoopsie       0  S   00:00.08       0       0 whoopsie
                                           0.0    0.1    401M  7.42M   4128 nicolargo      0  S   00:02.50       0       0 zeitgeist-datahub
                                           0.0    0.4    800M  31.0M   4752 nicolargo      0  S   00:04.21       0       0 megasync
                                           0.0    0.0   7.32M   320K   2011 rabbitmq       0  S   00:00.06       0       0 epmd
                                           0.0    0.1    382M  5.29M   3185 nicolargo      0  S   00:00.02       0       0 gsd-printer
```

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
```

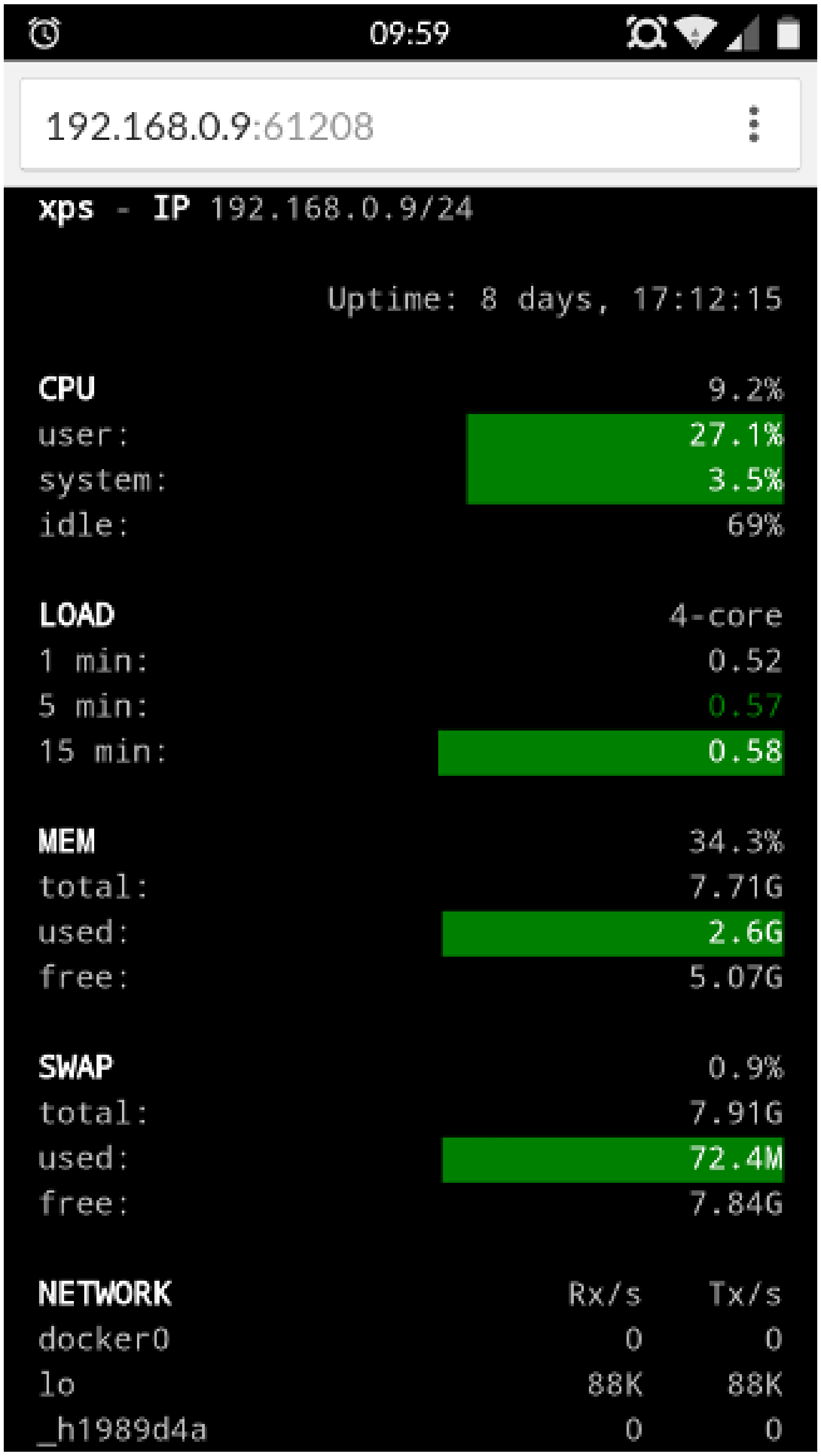
where `@server` is the IP address or hostname of the server.

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:



```
09:59
192.168.0.9:61208
xps - IP 192.168.0.9/24

                Uptime: 8 days, 17:12:15

CPU                             9.2%
user:                          27.1%
system:                         3.5%
idle:                            69%

LOAD                          4-core
1 min:                          0.52
5 min:                          0.57
15 min:                         0.58

MEM                            34.3%
total:                         7.71G
used:                           2.6G
free:                          5.07G

SWAP                            0.9%
total:                         7.91G
used:                          72.4M
free:                          7.84G

NETWORK               Rx/s      Tx/s
docker0                  0         0
lo                     88K       88K
_h1989d4a                0         0
```

1.2.4 How to protect your server (or Web server) with a login/password ?

You can set a password to access to the server using the `--password`. By default, the login is `glances` but you can change it with `--username`.

If you want, the SHA password will be stored in `<login>.pwd` file (in the same folder where the Glances configuration file is stored, so ~/.config/glances/ on GNU Linux operating system).

Next time your run the server/client, password will not be asked. To set a specific username you can use the -u <username> option.

It is also possible to set the default 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
```

1.3 Command Reference

1.3.1 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

`--modules-list`
display modules (plugins & exports) list and exit

`--disable-plugin` `PLUGIN`
disable PLUGIN (comma separated list)

`--enable-plugin` `PLUGIN`
enable PLUGIN (comma separated list)

`--stdout` `PLUGINS_STATS`
display stats to stdout (comma separated list of plugins/plugins.attribute)

`--export` `EXPORT`
enable EXPORT module (comma separated list)

`--export-csv-file` `EXPORT_CSV_FILE`
file path for CSV exporter

`--export-json-file` `EXPORT_JSON_FILE`
file path for JSON exporter

`--disable-process`
disable process module (reduce Glances CPU consumption)

`--disable-webui`
disable the Web UI (only the RESTful API will respond)

--light, --enable-light
light mode for Curses UI (disable all but top menu)

-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

--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

-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 (not available on Windows)

-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

1.3.2 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

C Enable/disable cloud stats

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

G Enable/disable GPU stats

h Show/hide the help screen

i Sort processes by I/O rate

I Show/hide IP module

+ Increase selected process nice level / Lower the priority (need right) - Only in standalone mode.

- Decrease selected process nice level / Higher the priority (need right) - Only in standalone mode.

k Kill selected process (need right) - Only in standalone mode.

K Show/hide TCP connections

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

P Enable/Disable ports stats

q|ESC|CTRL-C Quit the current Glances session

Q Show/hide IRQ module

r Reset history

R Show/hide RAID plugin

s Show/hide sensors stats

S Enable/disable spark lines

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

9 Switch UI theme between black and white

/ Switch between process command line or command name

F5 Refresh stats in curses user interface

LEFT Navigation leff through process sort

RIGHT Navigation right through process sort

UP Up in the processes list

DOWN Down in the processes list

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

1.4 Configuration

No configuration file is mandatory to use Glances.

Furthermore a configuration file is needed to access more settings.

1.4.1 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`	~/.config/glances/, /etc/glances/, /usr/share/docs/glances/
`*BSD`	~/.config/glances/, /usr/local/etc/glances/, /usr/share/docs/glances/
`macOS`	~/Library/Application Support/glances/, /usr/local/etc/glances/, /usr/share/docs/glances/
`Windows`	%APPDATA%\glances\glances.conf

- 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.

1.4.2 Syntax

Glances reads configuration files in the *ini* syntax.

A first section (called global) is available:

```
[global]
# Refresh rate (default is a minimum of 2 seconds)
# Can be overwrite by the -t <sec> option
# It is also possible to overwrite it in each plugin sections
refresh=2
# Does Glances should check if a newer version is available on PyPI ?
check_update=false
# History size (maximum number of values)
# Default is 28800: 1 day with 1 point every 3 seconds
history_size=28800
```

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

```
[cpu]
disable=False
refresh=3
user_careful=50
user_warning=70
user_critical=90
iowait_careful=50
iowait_warning=70
```

(continues on next page)

(continued from previous page)

```
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
```

With Glances 3.0 or higher it is also possible to use dynamic configuration value using system command. For example, if you to set the prefix of an InfluxDB export to the current hostname, use:

```
[influxdb]
...
prefix=`hostname`
```

Or if you want to add the Operating System name as a tag:

```
[influxdb]
...
tags=system:`uname -a`
```

1.4.3 Logging

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

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

The location of the Glances depends of your operating system. You could displayed the Glances log file full path using the‘‘glances -V‘‘ command line.

The file is automatically rotate when the size is higher than 1 MB.

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"
```

(continues on next page)

(continued from previous page)

```
        }
    }
}
```

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.

1.5 Anatomy Of The Application

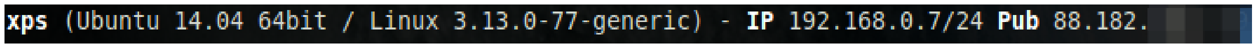
This document is meant to give an overview of the Glances interface.

Legend:

GREEN	OK
BLUE	CAREFUL
MAGENTA	WARNING
RED	CRITICAL

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

1.5.1 Header



```
xps (Ubuntu 14.04 64bit / Linux 3.13.0-77-generic) - IP 192.168.0.7/24 Pub 88.182.[redacted]
```

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.

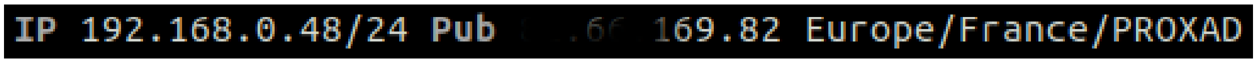
It is possible to disable or define time interval to be used for refreshing the public IP address (default is 300 seconds) from the configuration file under the `[ip]` section:

NOTE: Setting low values for *public_refresh_interval* will result in frequent HTTP requests to the IP detection servers. Recommended range: 120-600 seconds. Glances uses online services in order to get the IP addresses. Your IP address could be blocked if too many requests are done.

If the Censys options are configured, the public IP address is also analysed (with the same interval) and additional information is displayed.

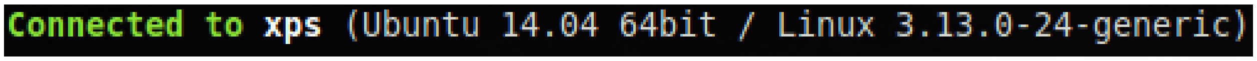
Note: Access to the Censys Search API need an account (https://censys.io/login).

Example:



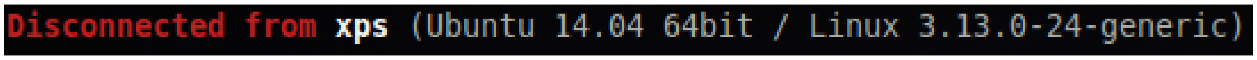
```
IP 192.168.0.48/24 Pub [redacted] 169.82 Europe/France/PROXAD
```

Connected:



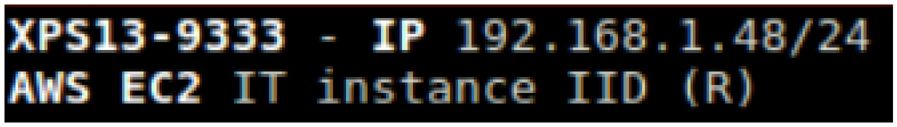
```
Connected to xps (Ubuntu 14.04 64bit / Linux 3.13.0-24-generic)
```

Disconnected:



```
Disconnected from xps (Ubuntu 14.04 64bit / Linux 3.13.0-24-generic)
```

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

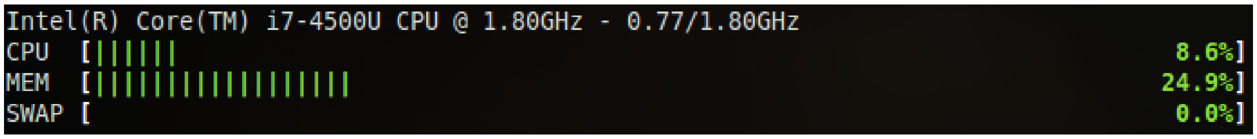


```
XPS13-9333 - IP 192.168.1.48/24
AWS EC2 IT instance IID (R)
```

1.5.2 Quick Look

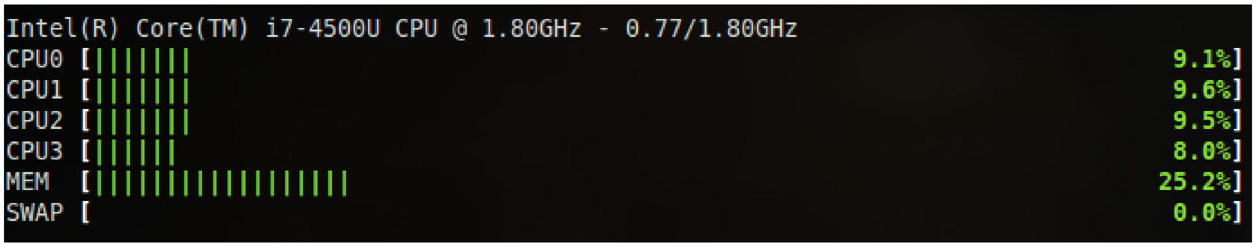
The `quicklook` plugin is only displayed on wide screen and proposes a bar view for CPU and memory (virtual and swap).

In the terminal interface, click on `3` to enable/disable it.



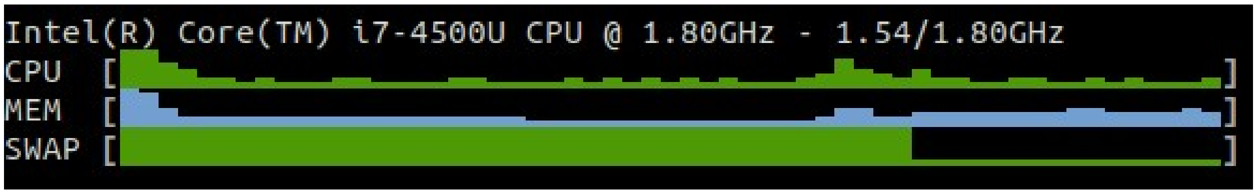
```
Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz - 0.77/1.80GHz
CPU  [||||||                                                    8.6%]
MEM  [|||||||||||||||||                                        24.9%]
SWAP [                                                          0.0%]
```

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%]
```

In the Curses/terminal interface, it is also possible to switch from bar to sparkline using 'S' hot key or –sparkline command line option (need the sparklines Python lib on your system). Please be aware that sparklines use the Glances history and will not be available if the history is disabled from the command line. For the moment sparkline is not available in client/server mode (see issue).



```
Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz - 1.54/1.80GHz
CPU  [                                                              ]
MEM  [                                                              ]
SWAP [                                                              ]
```

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

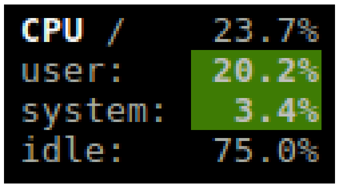
You can also configure the percentage char used in the terminal user interface.

```
[quicklook]
# Graphical percentage char used in the terminal user interface (default is |)
percentage_char=@
```

1.5.3 CPU

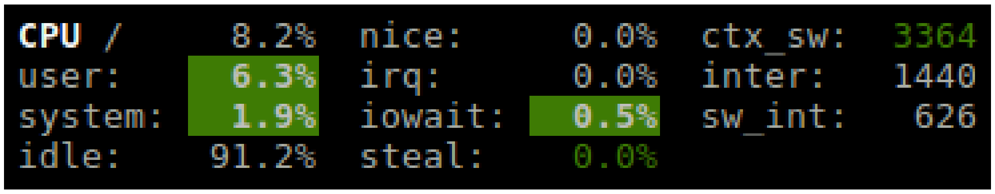
The CPU stats are shown as a percentage or values and for the configured refresh time.

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%
```

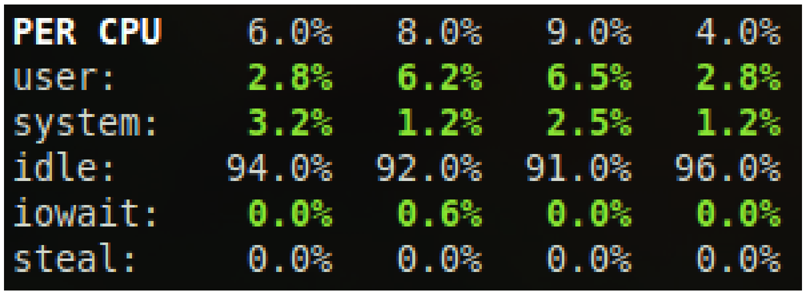
A character is also displayed just after the CPU header and shows the trend value:

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).
- **dpc**: *(Windows)*: time spent servicing deferred procedure calls.

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%
```

In this case, Glances will show on line per logical CPU on the system. Logical cores means the number of physical cores multiplied by the number of threads that can run on each core (this is known as Hyper Threading).

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

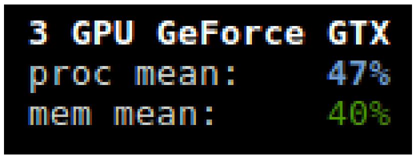
Note: Limit values can be overwritten in the configuration file under the `[cpu]` and/or `[percpu]` sections.

1.5.4 GPU

Note: You need to install the py3nvml library on your system. Or nvidia-ml-py3 for Glances 3.1.3 or lower.

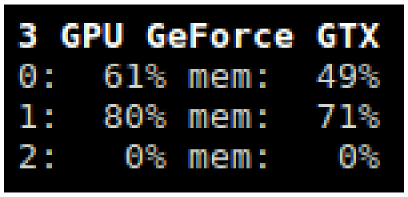
The GPU stats are shown as a percentage of value and for the configured refresh time. It displays:

- total GPU usage
- memory consumption
- temperature (Glances 3.1.4 or higher)



```
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:

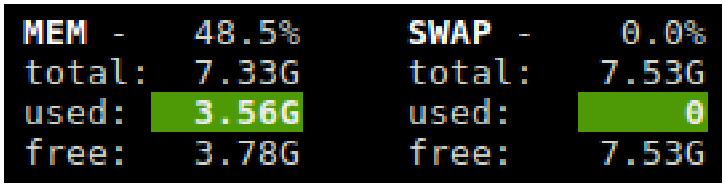
```
[gpu]
# Default processor values if not defined: 50/70/90
proc_careful=50
proc_warning=70
proc_critical=90
# Default memory values if not defined: 50/70/90
mem_careful=50
mem_warning=70
mem_critical=90
```

Legend:

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

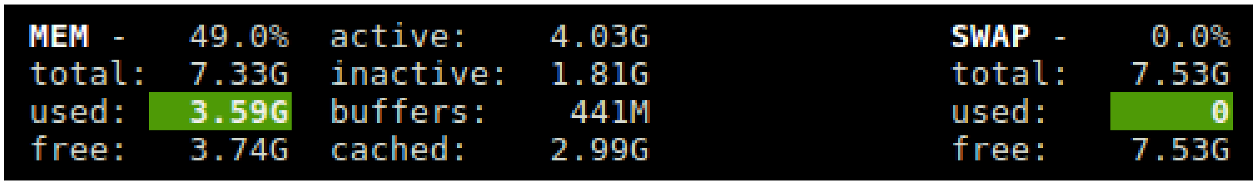
1.5.5 Memory

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
```

Stats description:

- **percent**: the percentage usage calculated as (total-available)/total*100.
- **total**: total physical memory available.

- **used**: memory used, calculated differently depending on the platform and designed for informational purposes only. It's compute as following:

  used memory = total - free (with free = available + buffers + cached)

- **free**: memory not being used at all (zeroed) that is readily available; note that this doesn't reflect the actual memory available (use 'available' instead).
- **active**: (UNIX): memory currently in use or very recently used, and so it is in RAM.
- **inactive**: (UNIX): memory that is marked as not used.
- **buffers**: (Linux, BSD): cache for things like file system metadata.
- **cached**: (Linux, BSD): cache for various things.

Additional stats available in through the API:

- **available**: the actual amount of available memory that can be given instantly to processes that request more memory in bytes; this is calculated by summing different memory values depending on the platform (e.g. free + buffers + cached on Linux) and it is supposed to be used to monitor actual memory usage in a cross platform fashion.
- **wired**: (BSD, macOS): memory that is marked to always stay in RAM. It is never moved to disk.
- **shared**: (BSD): memory that may be simultaneously accessed by multiple processes.

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.

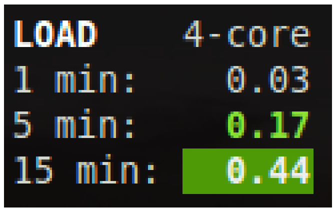
Legend:

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.

1.5.6 Load

Availability: Unix and Windows with a PsUtil version >= 5.6.2



```
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:

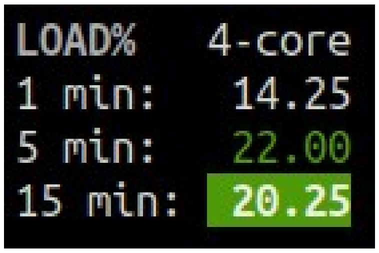
> "In short it is the average sum of the number of processes waiting in the run-queue plus the number currently executing over 1, 5, and 15 minutes time periods."

Be aware that Load on Linux, BSD and Windows are different things, high load on BSD does not means high CPU load. The Windows load is emulated by the PsUtil lib (see load on Windows)

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

Thresholds are computed by dividing the 5 and 15 minutes average load per CPU(s) number. For example, if you have 4 CPUs and the 5 minutes load is 1.0, then the warning threshold will be set to 2.8 (0.7 * 4 * 1.0).

From Glances 3.1.4, if Irix/Solaris mode is off ('0' key), the value is divided by logical core number and multiple by 100 to have load as a percentage.



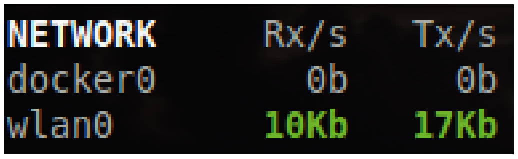
```
LOAD%    4-core
1 min:    14.25
5 min:    22.00
15 min:   20.25
```

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.

1.5.7 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

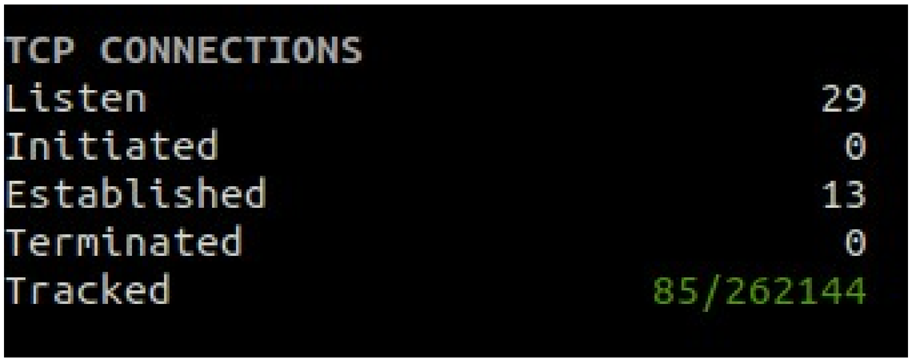
The configuration should be done in the `[network]` section of the Glances configuration file.

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
# Define the list of network interfaces to show (comma-separated regexp)
#show=eth0,eth1
# 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
```

Filtering is based on regular expression. Please be sure that your regular expression works as expected. You can use an online tool like regex101 in order to test your regular expression.

1.5.8 Connections



```
TCP CONNECTIONS
Listen                      29
Initiated                    0
Established                 13
Terminated                   0
Tracked              85/262144
```

This plugin display extended information about network connections.

The states are the following:

- Listen: all ports created by server and waiting for a client to connect
- Initialized: All states when a connection is initialized (sum of SYN_SENT and SYN_RECEIVED)
- Established: All established connections between a client and a server

- Terminated: All states when a connection is terminated (FIN_WAIT1, CLOSE_WAIT, LAST_ACK, FIN_WAIT2, TIME_WAIT and CLOSE)
- Tracked: Current number and maximum Netfilter tracker connection (nf_conntrack_count/nf_conntrack_max)

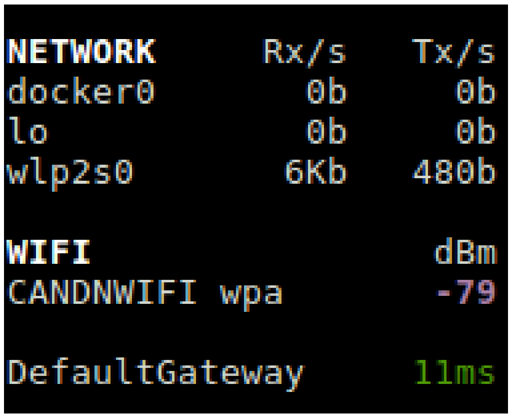
The configuration should be done in the `[connections]` section of the Glances configuration file.

By default the plugin is **disabled**. Please change your configuration file as following to enable it

```
[connections]
disable=False
# nf_conntrack thresholds in %
nf_conntrack_percent_careful=70
nf_conntrack_percent_warning=80
nf_conntrack_percent_critical=90
```

1.5.9 Wi-Fi

Availability: Linux



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

WIFI                   dBm
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

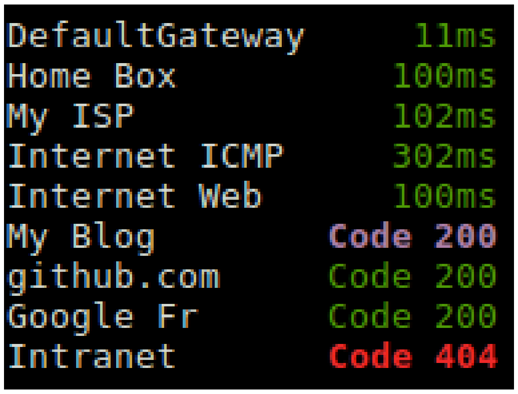
It's also possible to disable the scan on a specific interface from the configuration file (`[wifi]` section). For example, if you want to hide the loopback interface (lo) and all the virtual docker interfaces:

```
[wifi]
hide=lo,docker.*
#show=wlp2s0
# Define SIGNAL thresholds in dBm (lower is better...)
careful=-65
warning=-75
critical=-85
```

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

1.5.10 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.

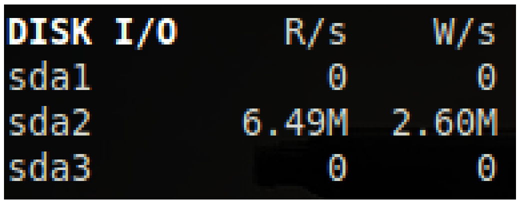
```
[ports]
# Ports scanner plugin configuration
# Interval in second between two scans
refresh=30
# Set the default timeout (in second) for a scan (can be overwrite in the scan list)
timeout=3
# If port_default_gateway is True, add the default gateway on top of the scan list
port_default_gateway=True
#
# 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)
```

(continues on next page)

(continued from previous page)

```
# 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␣
↪(approximately)
#
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
```

1.5.11 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.

You can display:

- bytes per second (default behavior / Bytes/s, KBytes/s, MBytes/s, etc)
- requests per second (using –diskio-iops option or *B* hotkey)

There is no alert on this information.

It's possible to define:

- a list of disk to show (white list)
- 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:

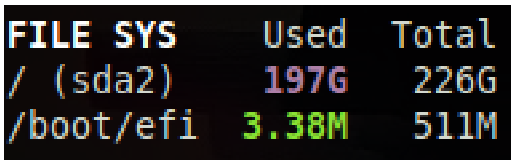
```
[diskio]
hide=sda5,loop.*
```

or another example:

```
[diskio]
show=sda.*
```

Filtering is based on regular expression. Please be sure that your regular expression works as expected. You can use an online tool like regex101 in order to test your regular expression.

1.5.12 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 user disk space usage.

Legend:

User disk space usage	Status
`<50%`	`OK`
`>50%`	`CAREFUL`
`>70%`	`WARNING`
`>90%`	`CRITICAL`

Note: Limit values can be overwritten in the configuration file under the `[fs]` 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 `shm` file system:

```
[fs]
allow=shm
```

Also, you can hide mount points using regular expressions.

To hide all mount points starting with /boot and /snap:

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

Filtering are also applied on device name (Glances 3.1.4 or higher).

It is also possible to configure a white list of devices to display. Example to only show /dev/sdb mount points:

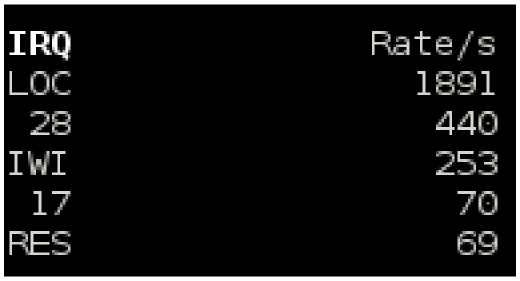
```
[fs]
show=/dev/sdb.*
```

Filtering is based on regular expression. Please be sure that your regular expression works as expected. You can use an online tool like regex101 in order to test your regular expression.

1.5.13 IRQ

Availability: Linux

This plugin is disable 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).

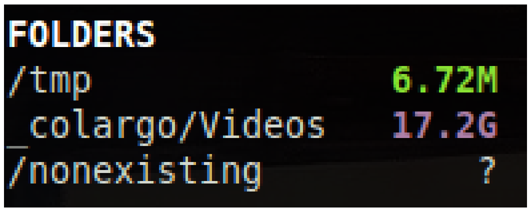
Note: `/proc/interrupts` file doesn't exist inside OpenVZ containers.

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

1.5.14 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)
- `refresh`: interval in second between two refresh (default is 30 seconds)

Up to `10` items can be defined.

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

```
[folders]
folder_1_path=/tmp
folder_1_careful=2500
folder_1_warning=3000
folder_1_critical=3500
folder_1_refresh=60
```

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

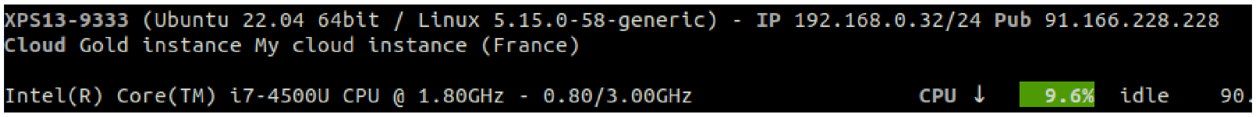
> **Warning:** Do **NOT** define folders containing lot of files and subfolders or use an huge refresh time…

1.5.15 CLOUD

This plugin diplays information about the cloud provider if your host is running on OpenStack.

The plugin use the standard OpenStack metadata service to retrieve the information.

This plugin is disable by default, please use the –enable-plugin cloud option to enable it.



```
XPS13-9333 (Ubuntu 22.04 64bit / Linux 5.15.0-58-generic) - IP 192.168.0.32/24 Pub 91.166.228.228
Cloud Gold instance My cloud instance (France)

Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz - 0.80/3.00GHz                CPU ↓   9.6%  idle    90.
```

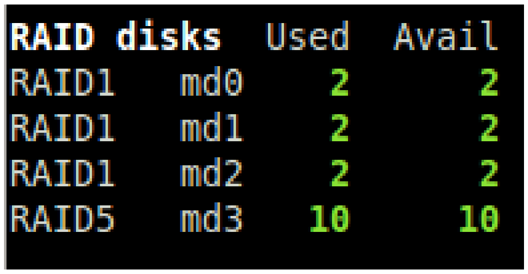
1.5.16 RAID

Availability: Linux

Dependency: this plugin uses the optional pymdstat Python lib

This plugin is disable by default, please use the –enable-plugin raid option to enable it.

In the terminal interface, click on `R` to enable/disable it.



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

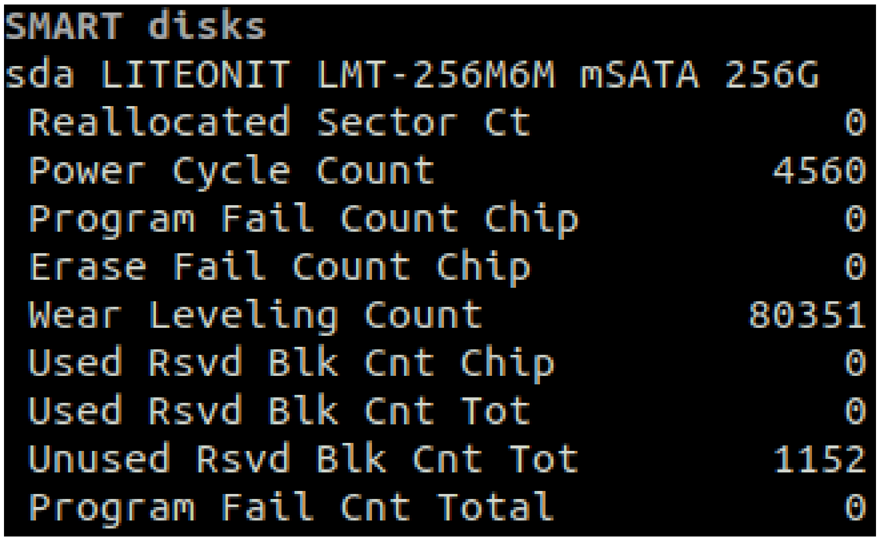
This plugin is only available on GNU/Linux.

1.5.17 SMART

Availability: all but Mac OS

Dependency: this plugin uses the optional pySMART Python lib

This plugin is disable by default, please use the –enable-plugin smart option to enable it.



```
SMART disks
sda LITEONIT LMT-256M6M mSATA 256G
 Reallocated Sector Ct              0
 Power Cycle Count               4560
 Program Fail Count Chip            0
 Erase Fail Count Chip              0
 Wear Leveling Count            80351
 Used Rsvd Blk Cnt Chip             0
 Used Rsvd Blk Cnt Tot              0
 Unused Rsvd Blk Cnt Tot         1152
 Program Fail Cnt Total             0
```

Glances displays all the SMART attributes.

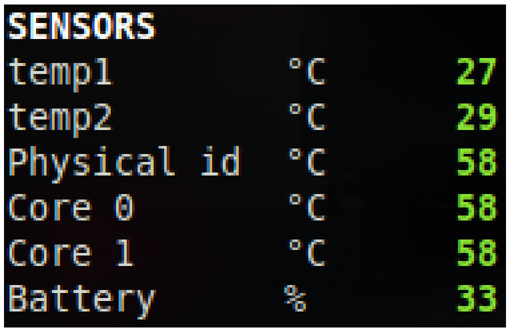
How to read the information:

- The first line display the name and model of the device
- The first column is the SMART attribute name
- The second column is the SMART attribute raw value

Warning: This plugin needs administrator rights. Please run Glances as root/admin.

1.5.18 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`, `hddtemp` and `batinfo`: - motherboard and CPU temperatures - hard disk temperature - battery capacity

There is no alert on this information.

1.5.19 HDD temperature sensor

Availability: Linux

This plugin will add HDD temperature to the sensors plugin.

On your Linux system, you will need to have: - hddtemp package installed - hddtemp service up and running (check it with systemctl status hddtemp) - the TCP port 7634 opened on your local firewall (if it is enabled on your system)

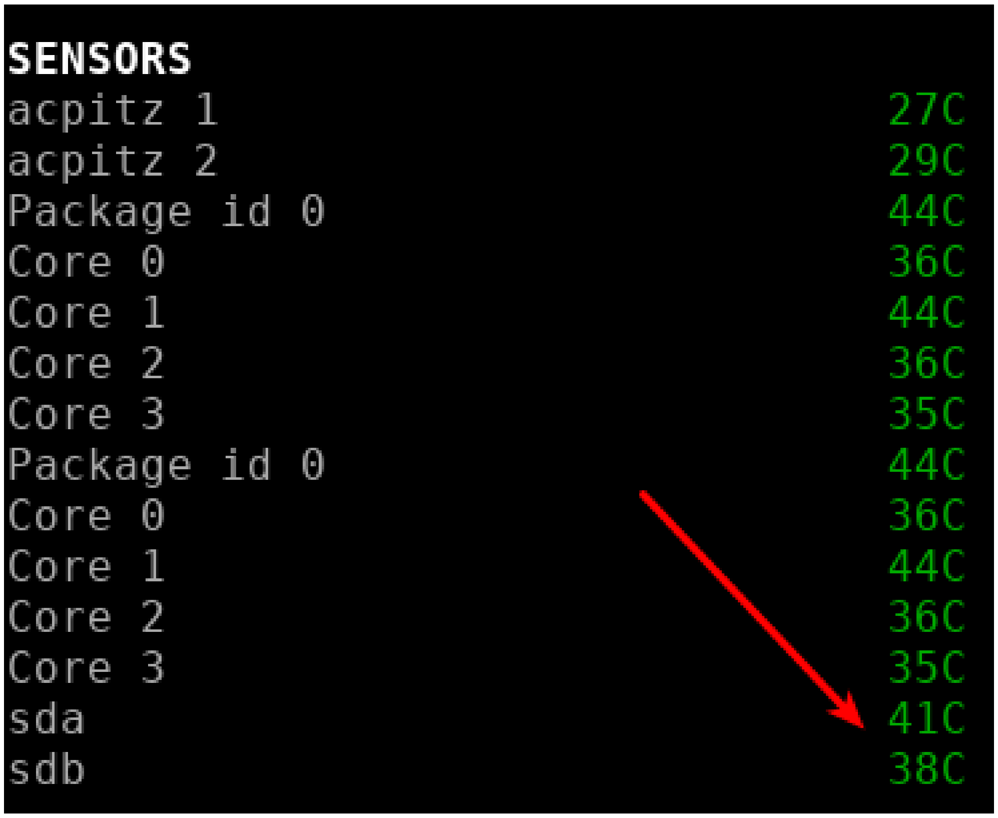
For example on a CentOS/Redhat Linux operating system, you have to:

> $ sudo yum install hddtemp
>
> $ sudo systemctl enable hddtemp
>
> $ sudo systemctl enable hddtemp

Test it in the console:

> $ hddtemp
>
> /dev/sda: TOSHIBA MQ01ACF050: 41°C
>
> /dev/sdb: ST1000LM044 HN-M101SAD: 38°C

It should appears in the sensors plugin.



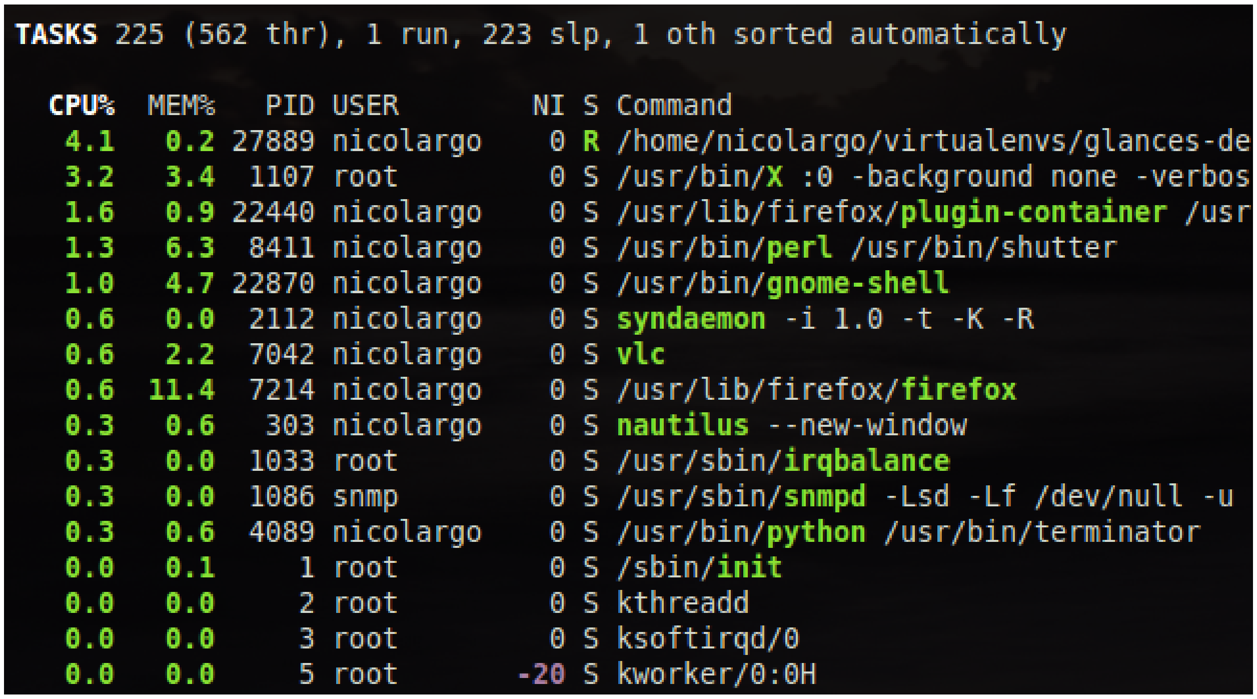
```
SENSORS
acpitz 1                          27C
acpitz 2                          29C
Package id 0                      44C
Core 0                            36C
Core 1                            44C
Core 2                            36C
Core 3                            35C
Package id 0                      44C
Core 0                            36C
Core 1                            44C
Core 2                            36C
Core 3                            35C
sda                               41C
sdb                               38C
```

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.

1.5.20 Processes List

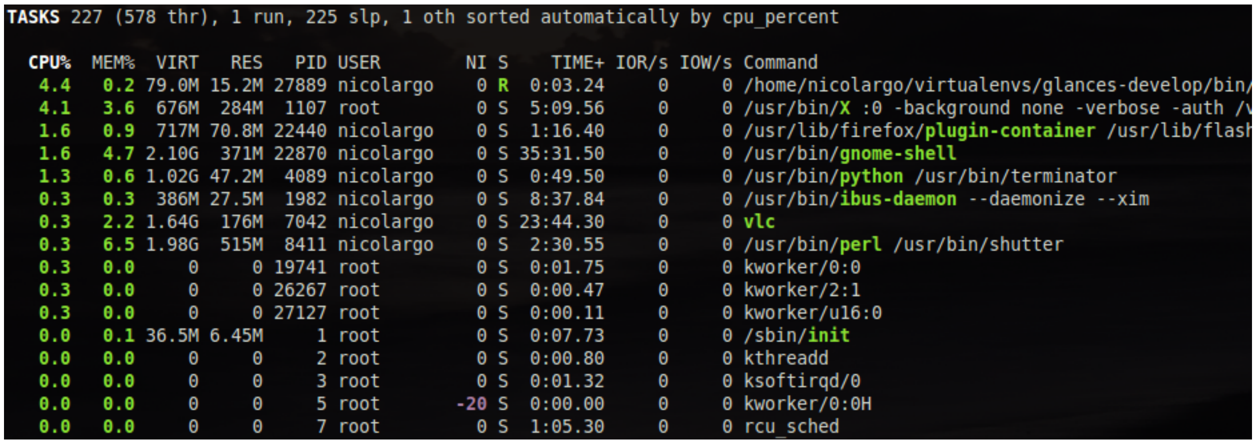
Compact view:



```
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 -verbos
   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
```

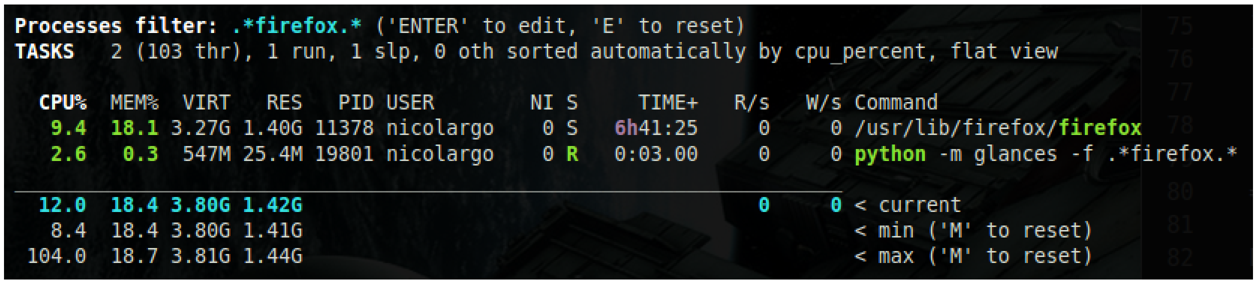
Full view:



```
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 /v
   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:

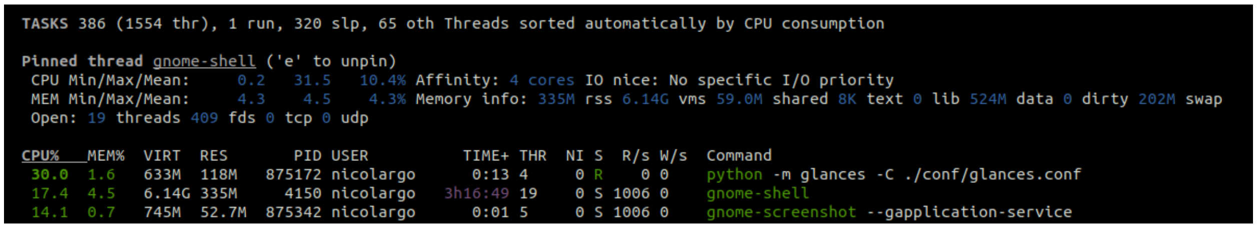


```
Processes filter: .*firefox.* ('ENTER' to edit, 'E' to reset)
TASKS   2 (103 thr), 1 run, 1 slp, 0 oth sorted automatically by cpu_percent, flat view

  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)
```

Extended view:



```
TASKS 386 (1554 thr), 1 run, 320 slp, 65 oth Threads sorted automatically by CPU consumption

Pinned thread gnome-shell ('e' to unpin)
 CPU Min/Max/Mean:     0.2   31.5   10.4% Affinity: 4 cores IO nice: No specific I/O priority
 MEM Min/Max/Mean:     4.3    4.5    4.3% Memory info: 335M rss 6.14G vms 59.0M shared 8K text 0 lib 524M data 0 dirty 202M swap
 Open: 19 threads 409 fds 0 tcp 0 udp

CPU%   MEM%  VIRT  RES       PID USER          TIME+ THR  NI S  R/s W/s  Command
 30.0  1.6   633M  118M   875172 nicolargo      0:13 4     0 R    0 0    python -m glances -C ./conf/glances.conf
 17.4  4.5   6.14G 335M     4150 nicolargo   3h16:49 19    0 S 1006 0    gnome-shell
 14.1  0.7   745M  52.7M  875342 nicolargo      0:01 5     0 S 1006 0    gnome-screenshot --gapplication-service
```

The process view consists of 3 parts:

- Processes summary
- Monitored processes list (optional, only in standalone mode)
- Extended stats for the selected process (optional)
- Processes list

The processes summary line displays:

- Total number of tasks/processes (aliases as total in the Glances API)
- Number of threads
- Number of running tasks/processes
- Number of sleeping tasks/processes
- Other number of tasks/processes (not in running or sleeping states)
- Sort key for the process list

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

You can also set the sort key in the UI:

- by clicking on left and right arrows
- by clicking on the following shortcuts or command line option:

Table 1: Title

Shortcut	Command line option	Description
a	Automatic sort	Default sort
c	–sort-processes cpu_percent	Sort by CPU
e	N/A	Pin the process and display extended stats
i	–sort-processes io_counters	Sort by DISK I/O
j	–programs	Accumulate processes by program (extended stats disable in this mode)
m	–sort-processes memory_percent	Sort by MEM
p	–sort-processes name	Sort by process name
t	–sort-processes cpu_times	Sort by CPU times
u	–sort-processes username	Sort by process username

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 ('0' key), the value is divided by logical core number
`MEM%`	% of MEM used by the process (RES divided by the total RAM you have)
`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. Virtual memory is usually much larger than physical memory, making it possible to run programs for which the total code plus data size is greater than the amount of RAM available. Most of the time, this is not a useful number.
`RES`	Resident Memory Size The non-swapped physical memory a process is using (what's currently in the physical memory).
`PID`	Process ID (column is replaced by NPROCS in accumulated mode)
`USER`	User ID
`THR`	Threads number of the process
`TIME+`	Cumulative CPU time used by the process
`NI`	Nice level of the process
`S`	Process status The status of the process: • `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 by job control signal • `t`: stopped by debugger during the tracing • `X`: dead (should never be seen)
`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 `'/'` key

Process filtering

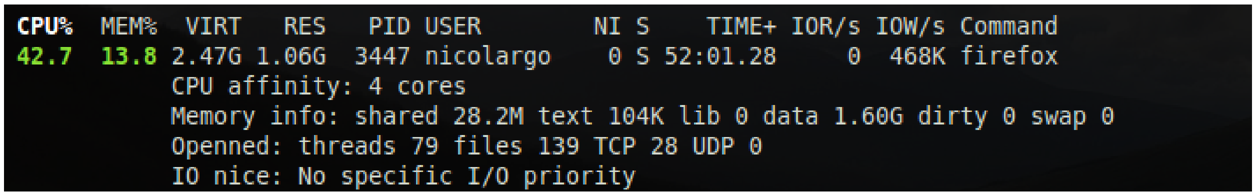
It's possible to filter the processes list using the `ENTER` key.

Filter syntax is the following (examples):

- `python`: Filter processes name or command line starting with *python* (regexp)
- `.*python.*`: Filter processes name or command line containing *python* (regexp)
- `username:nicolargo`: Processes of nicolargo user (key:regexp)

- `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
           Openned: 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).

In curses/standalone mode, you can select a process using `UP` and `DOWN` and press: - `k` to kill the selected process

Note: Limit for CPU and MEM percent values can be overwritten in the configuration file under the `[processlist]` section. It is also possible to define limit for Nice values (comma separated list). For example: nice_warning=-20,-19,-18

Accumulated per program — key 'j'

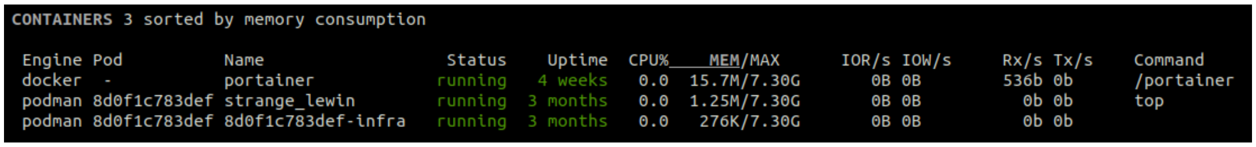
When activated ('j' hotkey or –programs option in the command line), processes are merged to display which programs are active. The columns show the accumulated cpu consumption, the accumulated virtual and resident memory consumption, the accumulated transferred data I/O. The PID columns is replaced by a NPROCS column which is the number of processes.

1.5.21 Containers

If you use `containers`, Glances can help you to monitor your Docker or Podman containers. Glances uses the containers API through the docker-py and podman-py libraries.

You can install this dependency using:

```
pip install glances[containers]
```



```
CONTAINERS 3 sorted by memory consumption

 Engine Pod          Name                  Status    Uptime  CPU%    MEM/MAX     IOR/s IOW/s     Rx/s Tx/s    Command
 docker  -           portainer            running   4 weeks   0.0  15.7M/7.30G     0B 0B       536b 0b      /portainer
 podman 8d0f1c783def strange_lewin        running  3 months   0.0  1.25M/7.30G     0B 0B         0b 0b      top
 podman 8d0f1c783def 8d0f1c783def-infra   running  3 months   0.0   276K/7.30G     0B 0B         0b 0b
```

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

```
[containers]
disable=False
# Only show specific containers (comma separated list of container name or regular␣
↪expression)
show=thiscontainer,andthisone,andthoseones.*
# Hide some containers (comma separated list of container name or regular expression)
hide=donotshowthisone,andthose.*
# Show only specific containers (comma separated list of container name or regular␣
↪expression)
#show=showthisone,andthose.*
# Define the maximum containers size name (default is 20 chars)
max_name_size=20
# 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
# By default, Glances only display running containers
# Set the following key to True to display all containers
all=False
# Define Podman sock
#podman_sock=unix:///run/user/1000/podman/podman.sock
```

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

Filtering (for hide or show) is based on regular expression. Please be sure that your regular expression works as expected. You can use an online tool like regex101 in order to test your regular expression.

1.5.22 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-plugin amps` option or pressing the `A` key.

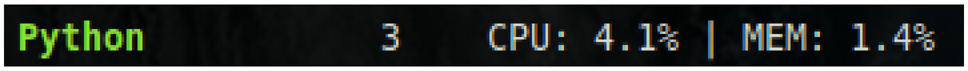
Simple AMP

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:

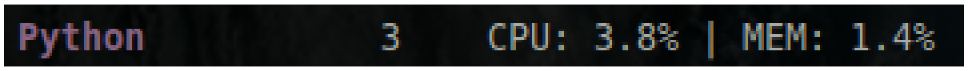


```
Python                3     CPU: 4.1% | MEM: 1.4%
```

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):



```
Python                3     CPU: 3.8% | MEM: 1.4%
```

If the regex option is not defined, the AMP will be executed every refresh time and the process count will not be displayed (countmin and countmax will be ignored).

For example:

```
[amp_conntrack]
enable=false
refresh=30
one_line=false
command=sysctl net.netfilter.nf_conntrack_count;sysctl net.netfilter.nf_conntrack_max
```

For security reason, pipe is not directly allowed in a AMP command but you create a sheel script with your command:

```
$ cat /usr/local/bin/mycommand.sh
#!/bin/sh
ps -aux | wc -l
```

and use it in the amps:

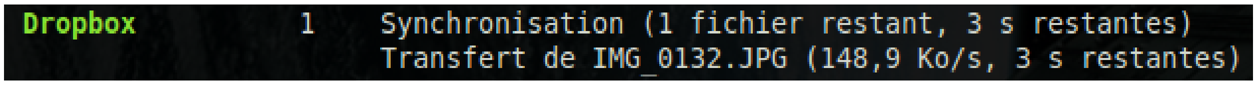
```
[amp_amptest]
enable=true
regex=.*
refresh=15
one_line=false
command=/usr/local/bin/mycommand.sh
```

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:



```
Dropbox          1    Synchronisation (1 fichier restant, 3 s restantes)
                      Transfert de IMG_0132.JPG (148,9 Ko/s, 3 s restantes)
```

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

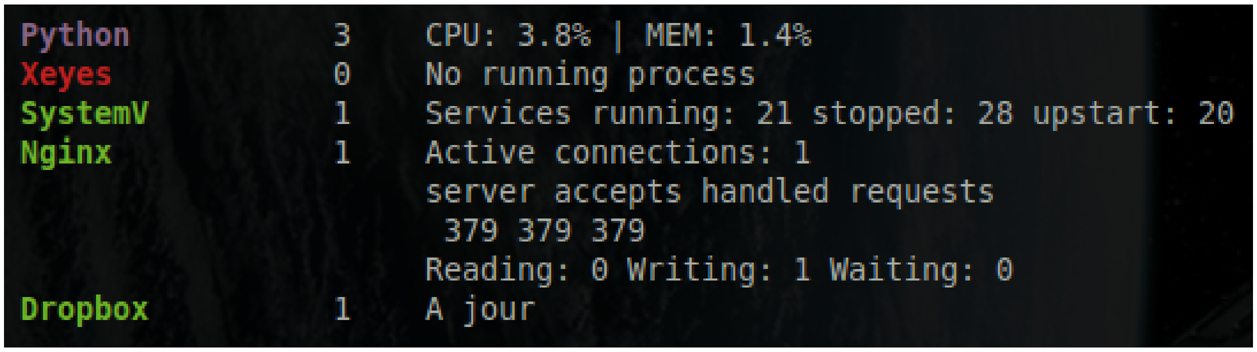
Embedded AMP

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
```

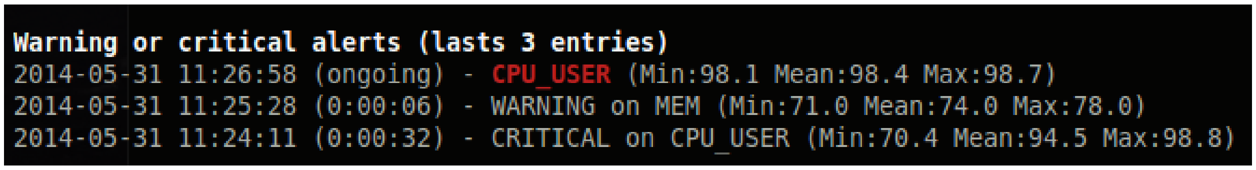
Here's the result:



```
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.

1.5.23 events



```
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)
```

Events 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 event 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

1.5.24 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. Chevron is required to render the mustache's template 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
```

A last example would be to create a log file containing the total user disk space usage for a device and notify by email each time a space trigger critical is reached:

```
[fs]
critical=90
critical_action_repeat=echo {{device_name}} {{percent}} > /tmp/fs.alert && python /
↪etc/glances/actions.d/fs-critical.py
```

Note: Use && as separator for multiple commands

Within `/etc/glances/actions.d/fs-critical.py`:

```
import subprocess
from requests import get

fs_alert = open('/tmp/fs.alert', 'r').readline().strip().split(' ')
device = fs_alert[0]
percent = fs_alert[1]
system = subprocess.check_output(['uname', '-rn']).decode('utf-8').strip()
ip = get('https://api.ipify.org').text

body = 'Used user disk space for ' + device + ' is at ' + percent + '%.\nPlease␣
↪cleanup the filesystem to clear the alert.\nServer: ' + str(system)+ '.\nIP␣
↪address: ' + ip
ps = subprocess.Popen(('echo', '-e', body), stdout=subprocess.PIPE)
subprocess.call(['mail', '-s', 'CRITICAL: disk usage above 90%', '-r',
↪'postmaster@example.com', 'glances@example.com'], stdin=ps.stdout)
```

Note: You can use all the stats for the current plugin. See https://github.com/nicolargo/glances/wiki/The-Glances-RESTFULL-JSON-API 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
```

1.6 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).

1.6.1 CSV

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

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

CSV file description:

- first line: Stats description (header)
- others lines: Stats (data)

By default, data will be append any existing CSV file (if header are compliant).

If the header did not match with a previous one, an error is logged.

The –export-csv-overwrite tag should be used if you want to delete the existing CSV file when Glances starts.

It is possible to remove some exported data using the –disable-plugin tag:

> $ glances –export csv –export-csv-file /tmp/glances.csv –disable-plugin load,swap –quiet

or by only enable some plugins:

> $ glances –export csv –export-csv-file /tmp/glances.csv –disable-plugin all –enable-plugin cpu,mem,load –quiet

1.6.2 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.

1.6.3 CouchDB

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

```
[mongodb]
host=localhost
port=27017
db=glances
user=root
password=example
```

and run Glances with:

```
$ glances --export mongodb
```

Documents are stored in native the configured database (glances by default) with one collection per plugin.

Example of MongoDB Document for the load stats:

```
{
    _id: ObjectId('63d78ffee5528e543ce5af3a'),
    min1: 1.46337890625,
    min5: 1.09619140625,
    min15: 1.07275390625,
    cpucore: 4,
    history_size: 1200,
    load_disable: 'False',
    load_careful: 0.7,
    load_warning: 1,
    load_critical: 5
}
```

1.6.4 Elasticsearch

Note: You need to install the elasticsearch library on your system.

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
```

1.6.5 Graph

You can generate dynamic graphs (SVG format) in a target folder. The generation starts every time the 'g' key is pressed in the CLI interface (if Glances has been started with the –export graph option).

The graph export module can be configured through the Glances configuration file:

```
[graph]
# Configuration for the --export graph option
# Set the path where the graph (.svg files) will be created
# Can be overwrite by the --graph-path command line option
path=/tmp
# It is possible to generate the graphs automatically by setting the
# generate_every to a non zero value corresponding to the seconds between
# two generation. Set it to 0 to disable graph auto generation.
generate_every=60
# See following configuration keys definitions in the Pygal lib documentation
# http://pygal.org/en/stable/documentation/index.html
width=800
height=600
style=DarkStyle
```

and run Glances with:

```
$ glances --export graph --export-graph-path /tmp
```

Example of output (load graph)

1.6.6 InfluxDB

You can export statistics to an `InfluxDB` server (time series server).

In Glances version 3.2.0 and higher, the way Glances exports stats to InfluxDB changes. The following fields will be added as tags:

- key stats (for example *interface_name* for network, container *name* for docker…)
- hostname (shortname)
- tags

Glances InfluxDB data model:

Measurement	Fields	Tags
cpu	user system iowait…	hostname
network	read_bytes write_bytes time_since_update…	hostname disk_name
diskio	rx tx time_since_update…	hostname interface_name
docker	cpu_percent memory_usage…	hostname name
gpu	proc mem temperature…	hostname gpu_id

InfluxDB (up to version 1.7.x)

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

```
[influxdb]
host=localhost
port=8086
protocol=http
user=root
password=root
db=glances
# Prefix will be added for all measurement name
# Ex: prefix=foo
#     => foo.cpu
#     => foo.mem
# You can also use dynamic values
#prefix=foo
# Following tags will be added for all measurements
# You can also use dynamic values.
# Note: hostname is always added as a tag
#tags=foo:bar,spam:eggs,domain:`domainname`
```

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 `tsm1` engine in the InfluxDB configuration file (no limit on columns number).

Note: if you want to use SSL, please set 'protocol=https'.

InfluxDB v2 (from InfluxDB v1.8.x/Flux and InfluxDB v2.x)

Note: The InfluxDB v2 client (https://pypi.org/project/influxdb-client/) is only available for Python 3.6 or higher.

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

```
[influxdb2]
host=localhost
port=8086
protocol=http
org=nicolargo
bucket=glances
token=EjFUTWe8U-MIseEAkaVIgVnej_TrnbdvEcRkaB1imstW7gapSqy6_6-8XD-yd51V0zUUpDy-
↪kAdVD1purDLuxA==
# Set the interval between two exports (in seconds)
# If the interval is set to 0, the Glances refresh time is used (default behavor)
#interval=0
```

(continues on next page)

(continued from previous page)

```
# Prefix will be added for all measurement name
# Ex: prefix=foo
#      => foo.cpu
#      => foo.mem
# You can also use dynamic values
#prefix=foo
# Following tags will be added for all measurements
# You can also use dynamic values.
# Note: hostname is always added as a tag
#tags=foo:bar,spam:eggs,domain:`domainname`
```

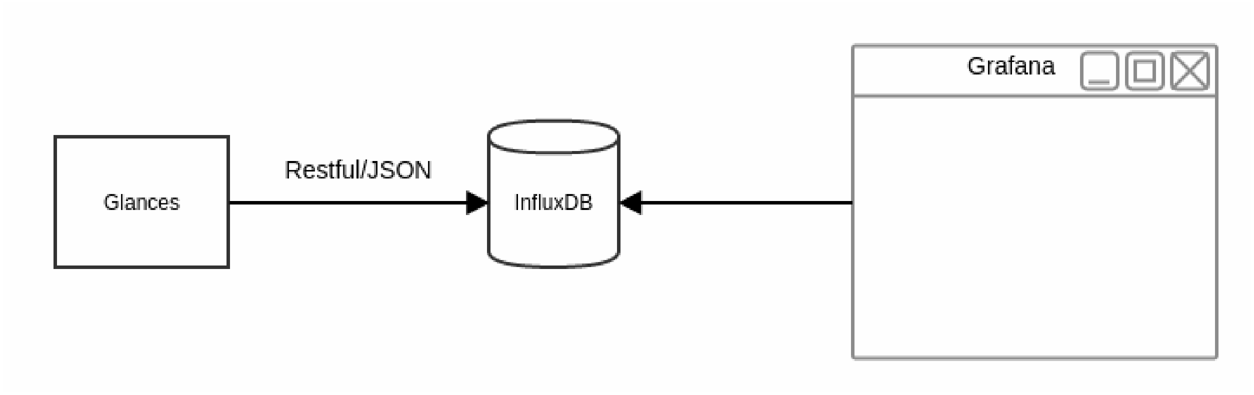
and run Glances with:

```
$ glances --export influxdb2
```

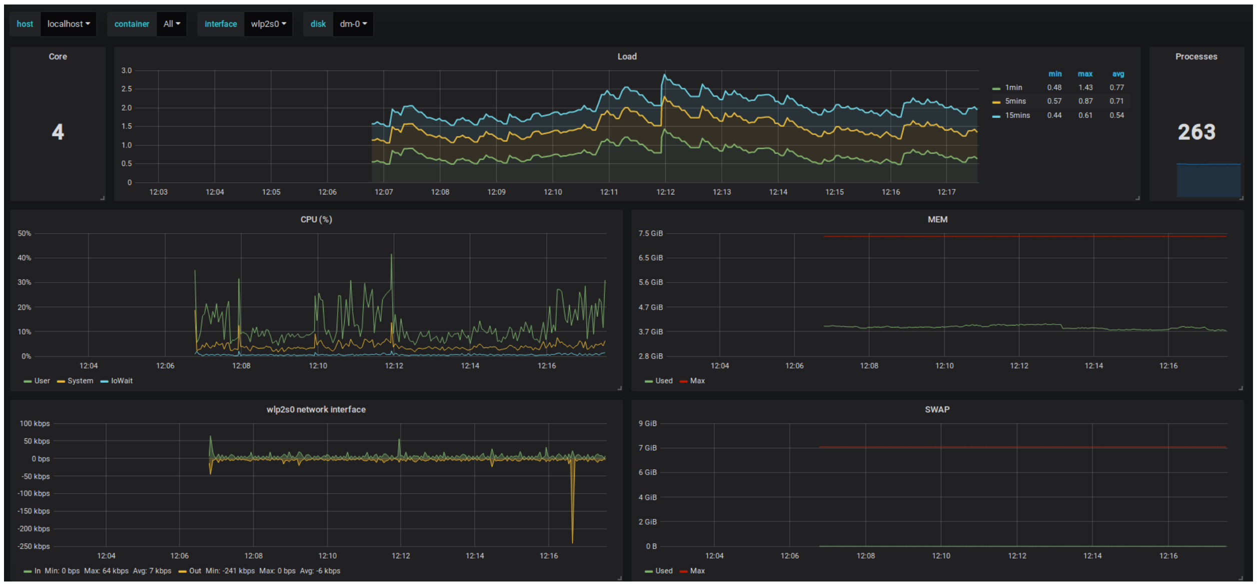
Note: if you want to use SSL, please set 'protocol=https'.

Grafana

For Grafana users, Glances provides a dedicated for InfluxQL or Flux InfluxDB datasource.



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



1.6.7 JSON

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

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

1.6.8 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
# Tags will be added for all events
#tags=foo:bar,spam:eggs
# You can also use dynamic values
#tags=hostname:`hostname -f`
```

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)
```

1.6.9 MQTT

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

```
[mqtt]
host=localhost
port=883
tls=true
user=glances
password=glances
topic=glances
topic_structure=per-metric
```

and run Glances with:

```
$ glances --export mqtt
```

The topic_structure field aims at configuring the way stats are exported to MQTT (see #1798): - per-metric: one event per metric (default behavior) - per-plugin: one event per plugin

1.6.10 MongoDB

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

```
[couchdb]
host=localhost
port=
user=root
password=example
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.

1.6.11 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
```

1.6.12 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
labels=src:glances
```

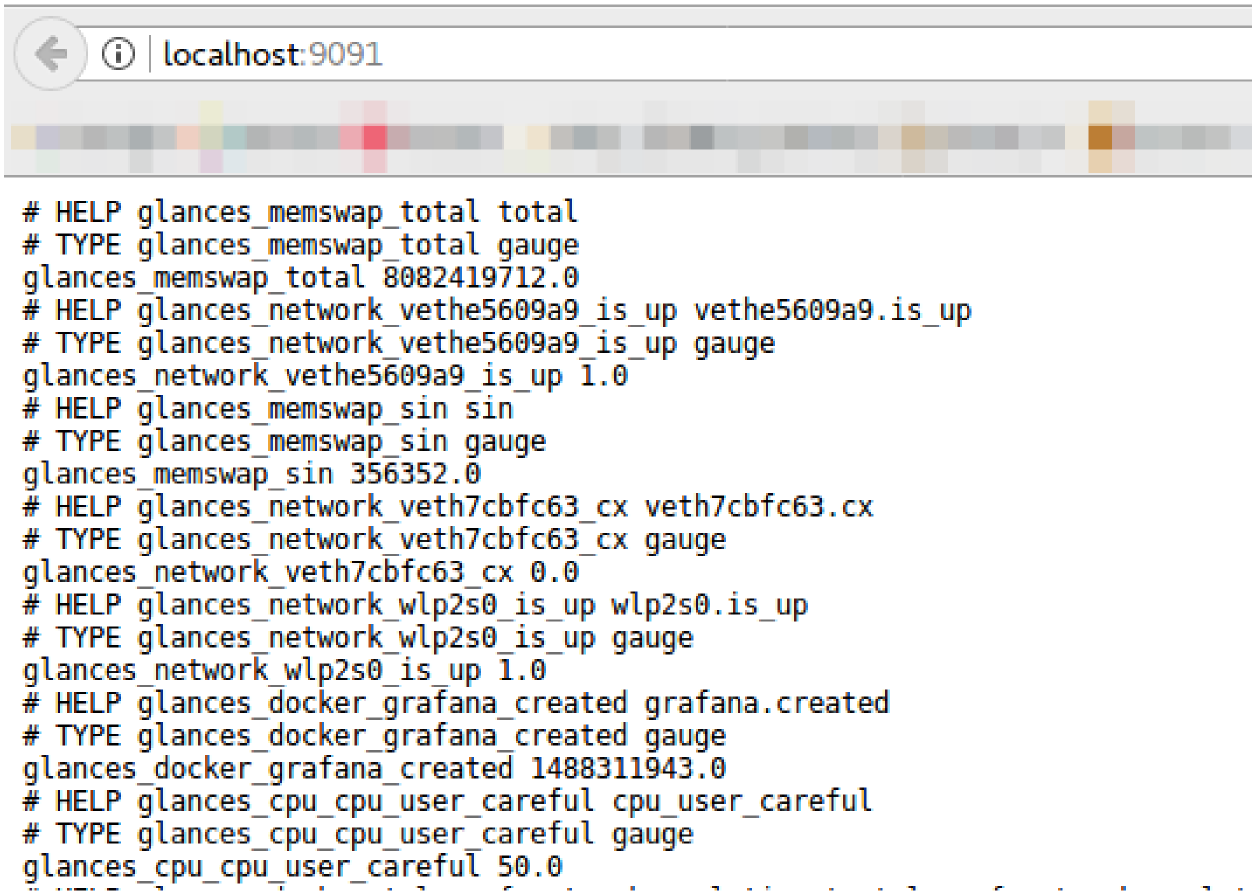
Note: When running Glances in a container, set `host=0.0.0.0` in the Glances configuration file.

Note: You can use dynamic fields for the label (ex: labels=system:*uname -s*)

and run Glances with:

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

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

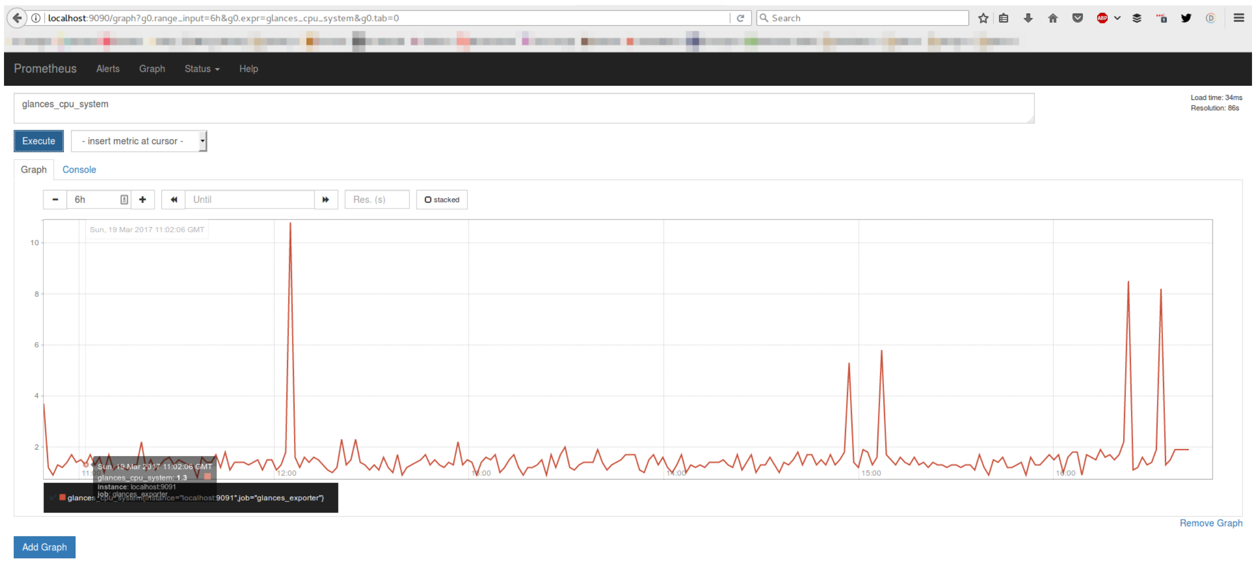


localhost:9091

```
# 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']
```



1.6.13 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
#protocol=amqps
```

and run Glances with:

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

1.6.14 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).

1.6.15 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
```

1.6.16 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,
...
```

1.6.17 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
```

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()
```

1.7 API (Restfull/JSON) documentation

The Glances Restfull/API server could be ran using the following command line:

```
# glances -w --disable-webui
```

Note: Change request URL api/3 by api/2 if you use Glances 2.x.

1.7.1 GET API status

This entry point should be used to check the API status. It will return nothing but a 200 return code if everythin is OK.

Get the Rest API status:

```
# curl -I http://localhost:61208/api/3/status
"HTTP/1.0 200 OK"
```

1.7.2 GET plugins list

Get the plugins list:

```
# curl http://localhost:61208/api/3/pluginslist
["alert",
 "amps",
 "cloud",
 "connections",
 "containers",
 "core",
 "cpu",
 "diskio",
 "folders",
 "fs",
 "gpu",
 "help",
 "ip",
 "irq",
 "load",
 "mem",
 "memswap",
 "network",
 "now",
 "percpu",
 "ports",
 "processcount",
 "processlist",
 "psutilversion",
 "quicklook",
 "raid",
 "sensors",
 "smart",
 "system",
 "uptime",
 "wifi"]
```

1.7.3 GET alert

Get plugin stats:

```
# curl http://localhost:61208/api/3/alert
[[1684593857.0,
  -1,
  "WARNING",
  "MEM",
```

(continues on next page)

(continued from previous page)

```
  74.95383222581204,
  74.95383222581204,
  74.95383222581204,
  74.95383222581204,
  1,
  [],
  "",
  "memory_percent"]]
```

1.7.4 GET amps

Get plugin stats:

```
# curl http://localhost:61208/api/3/amps
[{"count": 0,
  "countmax": None,
  "countmin": 1.0,
  "key": "name",
  "name": "Dropbox",
  "refresh": 3.0,
  "regex": True,
  "result": None,
  "timer": 1.7997314929962158},
 {"count": 0,
  "countmax": 20.0,
  "countmin": None,
  "key": "name",
  "name": "Python",
  "refresh": 3.0,
  "regex": True,
  "result": None,
  "timer": 1.7995269298553467}]
```

Get a specific field:

```
# curl http://localhost:61208/api/3/amps/name
{"name": ["Dropbox", "Python", "Conntrack", "Nginx", "Systemd", "SystemV"]}
```

Get a specific item when field matchs the given value:

```
# curl http://localhost:61208/api/3/amps/name/Dropbox
{"Dropbox": [{"count": 0,
              "countmax": None,
              "countmin": 1.0,
              "key": "name",
              "name": "Dropbox",
              "refresh": 3.0,
              "regex": True,
              "result": None,
              "timer": 1.7997314929962158}]}
```

1.7.5 GET connections

Get plugin stats:

```
# curl http://localhost:61208/api/3/connections
{"net_connections_enabled": True, "nf_conntrack_enabled": True}
```

Get a specific field:

```
# curl http://localhost:61208/api/3/connections/net_connections_enabled
{"net_connections_enabled": True}
```

1.7.6 GET containers

Get plugin stats:

```
# curl http://localhost:61208/api/3/containers
{"containers": [{"Command": ["top"],
                 "Created": "2023-05-08T15:29:34.918692365+02:00",
                 "Id":
↪"4b7f732d43e4bc5d92fe5298cba025b550e6a608754c1c38f9a90aaecd46b8f9",
                 "Image": "["docker.io/library/ubuntu:latest"]",
                 "Status": "running",
                 "Uptime": "1 weeks",
                 "cpu": {"total": 2.2993346332214973e-06},
                 "cpu_percent": 2.2993346332214973e-06,
                 "engine": "podman",
                 "io": {"ior": 0.0, "iow": 0.0, "time_since_update": 1},
                 "io_r": 0.0,
                 "io_w": 0.0,
                 "key": "name",
                 "memory": {"limit": 7836184576.0, "usage": 1142784.0},
                 "memory_usage": 1142784.0,
                 "name": "frosty_bouman",
                 "network": {"rx": 0.0, "time_since_update": 1, "tx": 0.0},
                 "network_rx": 0.0,
                 "network_tx": 0.0,
                 "pod_id": "8d0f1c783def",
                 "pod_name": "frosty_bouman"},
                {"Command": [],
                 "Created": "2022-10-22T14:23:03.120912374+02:00",
                 "Id":
↪"9491515251edcd5bb5dc17205d7ee573c0be96fe0b08b0a12a7e2cea874565ea",
                 "Image": "["k8s.gcr.io/pause:3.5"]",
                 "Status": "running",
                 "Uptime": "1 weeks",
                 "cpu": {"total": 2.754373096346692e-10},
                 "cpu_percent": 2.754373096346692e-10,
                 "engine": "podman",
                 "io": {"ior": 0.0, "iow": 0.0, "time_since_update": 1},
                 "io_r": 0.0,
                 "io_w": 0.0,
                 "key": "name",
                 "memory": {"limit": 7836184576.0, "usage": 208896.0},
                 "memory_usage": 208896.0,
                 "name": "8d0f1c783def-infra",
                 "network": {"rx": 0.0, "time_since_update": 1, "tx": 0.0},
                 "network_rx": 0.0,
                 "network_tx": 0.0,
                 "pod_id": "8d0f1c783def",
```

(continues on next page)

(continued from previous page)

```
                  "pod_name": "8d0f1c783def-infra"},
                 {"Command": ["/portainer"],
                  "Created": "2022-10-29T14:59:10.266701439Z",
                  "Id":
↪"3abd51c615968482d9ccff5afc629f267f6dda113ed68b75b432615fae3b49fb",
                  "Image": ["portainer/portainer-ce:2.9.3"],
                  "Status": "running",
                  "Uptime": "7 hours",
                  "cpu": {"total": 0.0},
                  "cpu_percent": 0.0,
                  "engine": "docker",
                  "io": {},
                  "io_r": None,
                  "io_w": None,
                  "key": "name",
                  "memory": {},
                  "memory_usage": None,
                  "name": "portainer",
                  "network": {},
                  "network_rx": None,
                  "network_tx": None}],
 "version": {},
 "version_podman": {}}
```

1.7.7 GET core

Get plugin stats:

```
# curl http://localhost:61208/api/3/core
{"log": 4, "phys": 2}
```

Fields descriptions:

- **phys**: Number of physical cores (hyper thread CPUs are excluded) (unit is *number*)
- **log**: Number of logical CPUs. A logical CPU is the number of physical cores multiplied by the number of threads that can run on each core (unit is *number*)

Get a specific field:

```
# curl http://localhost:61208/api/3/core/phys
{"phys": 2}
```

1.7.8 GET cpu

Get plugin stats:

```
# curl http://localhost:61208/api/3/cpu
{"cpucore": 4,
 "ctx_switches": 0,
 "guest": 1.1,
 "guest_nice": 0.0,
 "idle": 38.7,
 "interrupts": 0,
```

(continues on next page)

(continued from previous page)

```
"iowait": 3.2,
"irq": 0.0,
"nice": 0.0,
"soft_interrupts": 0,
"softirq": 0.4,
"steal": 0.0,
"syscalls": 0,
"system": 7.7,
"time_since_update": 1,
"total": 62.2,
"user": 50.0}
```

Fields descriptions:

- **total**: Sum of all CPU percentages (except idle) (unit is *percent*)
- **system**: percent time spent in kernel space. System CPU time is the time spent running code in the Operating System kernel (unit is *percent*)
- **user**: CPU 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) (unit is *percent*)
- **iowait**: *(Linux)*: percent time spent by the CPU waiting for I/O operations to complete (unit is *percent*)
- **dpc**: *(Windows)*: time spent servicing deferred procedure calls (DPCs) (unit is *percent*)
- **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 (unit is *percent*)
- **irq**: *(Linux and BSD)*: percent time spent servicing/handling hardware/software interrupts. Time servicing interrupts (hardware + software) (unit is *percent*)
- **nice**: *(Unix)*: 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* (unit is *percent*)
- **steal**: *(Linux)*: percentage of time a virtual CPU waits for a real CPU while the hypervisor is servicing another virtual processor (unit is *percent*)
- **ctx_switches**: 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 (unit is *number*)
- **interrupts**: number of interrupts per second (unit is *number*)
- **soft_interrupts**: number of software interrupts per second. Always set to 0 on Windows and SunOS (unit is *number*)
- **syscalls**: number of system calls per second. Always 0 on Linux OS (unit is *number*)
- **cpucore**: Total number of CPU core (unit is *number*)
- **time_since_update**: Number of seconds since last update (unit is *seconds*)

Get a specific field:

```
# curl http://localhost:61208/api/3/cpu/total
{"total": 62.2}
```

1.7.9 GET diskio

Get plugin stats:

```
# curl http://localhost:61208/api/3/diskio
[{"disk_name": "sda",
  "key": "disk_name",
  "read_bytes": 0,
  "read_count": 0,
  "time_since_update": 1,
  "write_bytes": 0,
  "write_count": 0},
 {"disk_name": "sda1",
  "key": "disk_name",
  "read_bytes": 0,
  "read_count": 0,
  "time_since_update": 1,
  "write_bytes": 0,
  "write_count": 0}]
```

Get a specific field:

```
# curl http://localhost:61208/api/3/diskio/disk_name
{"disk_name": ["sda", "sda1", "sda2", "sda5", "dm-0", "dm-1"]}
```

Get a specific item when field matchs the given value:

```
# curl http://localhost:61208/api/3/diskio/disk_name/sda
{"sda": [{"disk_name": "sda",
          "key": "disk_name",
          "read_bytes": 0,
          "read_count": 0,
          "time_since_update": 1,
          "write_bytes": 0,
          "write_count": 0}]}
```

1.7.10 GET fs

Get plugin stats:

```
# curl http://localhost:61208/api/3/fs
[{"device_name": "/dev/mapper/ubuntu--gnome--vg-root",
  "free": 4763168768,
  "fs_type": "ext4",
  "key": "mnt_point",
  "mnt_point": "/",
  "percent": 97.9,
  "size": 243334156288,
  "used": 226183532544},
 {"device_name": "zsfpool",
  "free": 41811968,
  "fs_type": "zfs",
  "key": "mnt_point",
  "mnt_point": "/zsfpool",
  "percent": 0.3,
  "size": 41943040,
  "used": 131072}]
```

Get a specific field:

```
# curl http://localhost:61208/api/3/fs/mnt_point
{"mnt_point": ["/", "/zsfpool", "/var/snap/firefox/common/host-hunspell"]}
```

Get a specific item when field matchs the given value:

```
# curl http://localhost:61208/api/3/fs/mnt_point//
{"/": [{"device_name": "/dev/mapper/ubuntu--gnome--vg-root",
        "free": 4763168768,
        "fs_type": "ext4",
        "key": "mnt_point",
        "mnt_point": "/",
        "percent": 97.9,
        "size": 243334156288,
        "used": 226183532544}]}
```

1.7.11 GET ip

Get plugin stats:

```
# curl http://localhost:61208/api/3/ip
{"address": "192.168.0.32",
 "gateway": "192.168.0.254",
 "mask": "255.255.255.0",
 "mask_cidr": 24,
 "public_address": "91.166.228.228",
 "public_info_human": ""}
```

Get a specific field:

```
# curl http://localhost:61208/api/3/ip/gateway
{"gateway": "192.168.0.254"}
```

1.7.12 GET load

Get plugin stats:

```
# curl http://localhost:61208/api/3/load
{"cpucore": 4,
 "min1": 3.00634765625,
 "min15": 1.19775390625,
 "min5": 1.4345703125}
```

Fields descriptions:

- **min1**: Average sum of the number of processes waiting in the run-queue plus the number currently executing over 1 minute (unit is *float*)
- **min5**: Average sum of the number of processes waiting in the run-queue plus the number currently executing over 5 minutes (unit is *float*)
- **min15**: Average sum of the number of processes waiting in the run-queue plus the number currently executing over 15 minutes (unit is *float*)
- **cpucore**: Total number of CPU core (unit is *number*)

Get a specific field:

```
# curl http://localhost:61208/api/3/load/min1
{"min1": 3.00634765625}
```

1.7.13 GET mem

Get plugin stats:

```
# curl http://localhost:61208/api/3/mem
{"active": 3130548224,
 "available": 1962663936,
 "buffers": 142680064,
 "cached": 2079936512,
 "free": 1962663936,
 "inactive": 3094474752,
 "percent": 75.0,
 "shared": 498302976,
 "total": 7836184576,
 "used": 5873520640}
```

Fields descriptions:

- **total**: Total physical memory available (unit is *bytes*)
- **available**: The actual amount of available memory that can be given instantly to processes that request more memory in bytes; this is calculated by summing different memory values depending on the platform (e.g. free + buffers + cached on Linux) and it is supposed to be used to monitor actual memory usage in a cross platform fashion (unit is *bytes*)
- **percent**: The percentage usage calculated as (total - available) / total * 100 (unit is *percent*)
- **used**: Memory used, calculated differently depending on the platform and designed for informational purposes only (unit is *bytes*)
- **free**: Memory not being used at all (zeroed) that is readily available; note that this doesn't reflect the actual memory available (use 'available' instead) (unit is *bytes*)
- **active**: *(UNIX)*: memory currently in use or very recently used, and so it is in RAM (unit is *bytes*)
- **inactive**: *(UNIX)*: memory that is marked as not used (unit is *bytes*)
- **buffers**: *(Linux, BSD)*: cache for things like file system metadata (unit is *bytes*)
- **cached**: *(Linux, BSD)*: cache for various things (unit is *bytes*)
- **wired**: *(BSD, macOS)*: memory that is marked to always stay in RAM. It is never moved to disk (unit is *bytes*)
- **shared**: *(BSD)*: memory that may be simultaneously accessed by multiple processes (unit is *bytes*)

Get a specific field:

```
# curl http://localhost:61208/api/3/mem/total
{"total": 7836184576}
```

1.7.14 GET memswap

Get plugin stats:

```
# curl http://localhost:61208/api/3/memswap
{"free": 3018182656,
 "percent": 62.7,
 "sin": 10520571904,
 "sout": 16592646144,
 "time_since_update": 1,
 "total": 8082419712,
 "used": 5064237056}
```

Fields descriptions:

- **total**: Total swap memory (unit is *bytes*)
- **used**: Used swap memory (unit is *bytes*)
- **free**: Free swap memory (unit is *bytes*)
- **percent**: Used swap memory in percentage (unit is *percent*)
- **sin**: The number of bytes the system has swapped in from disk (cumulative) (unit is *bytes*)
- **sout**: The number of bytes the system has swapped out from disk (cumulative) (unit is *bytes*)
- **time_since_update**: Number of seconds since last update (unit is *seconds*)

Get a specific field:

```
# curl http://localhost:61208/api/3/memswap/total
{"total": 8082419712}
```

1.7.15 GET network

Get plugin stats:

```
# curl http://localhost:61208/api/3/network
[{"alias": None,
  "cumulative_cx": 344481682,
  "cumulative_rx": 172240841,
  "cumulative_tx": 172240841,
  "cx": 7770,
  "interface_name": "lo",
  "is_up": True,
  "key": "interface_name",
  "rx": 3885,
  "speed": 0,
  "time_since_update": 1,
  "tx": 3885},
 {"alias": None,
  "cumulative_cx": 22624441944,
  "cumulative_rx": 21888194655,
  "cumulative_tx": 736247289,
  "cx": 146685,
  "interface_name": "wlp2s0",
  "is_up": True,
  "key": "interface_name",
  "rx": 135811,
  "speed": 0,
  "time_since_update": 1,
  "tx": 10874}]
```

Fields descriptions:

- **interface_name**: Interface name (unit is *string*)
- **alias**: Interface alias name (optional) (unit is *string*)
- **rx**: The received/input rate (in bit per second) (unit is *bps*)
- **tx**: The sent/output rate (in bit per second) (unit is *bps*)
- **cx**: The cumulative received+sent rate (in bit per second) (unit is *bps*)
- **cumulative_rx**: The number of bytes received through the interface (cumulative) (unit is *bytes*)
- **cumulative_tx**: The number of bytes sent through the interface (cumulative) (unit is *bytes*)
- **cumulative_cx**: The cumulative number of bytes reveived and sent through the interface (cumulative) (unit is *bytes*)
- **speed**: Maximum interface speed (in bit per second). Can return 0 on some operating-system (unit is *bps*)
- **is_up**: Is the interface up ? (unit is *bool*)
- **time_since_update**: Number of seconds since last update (unit is *seconds*)

Get a specific field:

```
# curl http://localhost:61208/api/3/network/interface_name
{"interface_name": ["lo",
                    "wlp2s0",
                    "docker0",
                    "br_grafana",
                    "mpqemubr0",
                    "vboxnet0",
                    "tap-1e376645a40",
                    "veth54fd604"]}
```

Get a specific item when field matchs the given value:

```
# curl http://localhost:61208/api/3/network/interface_name/lo
{"lo": [{"alias": None,
         "cumulative_cx": 344481682,
         "cumulative_rx": 172240841,
         "cumulative_tx": 172240841,
         "cx": 7770,
         "interface_name": "lo",
         "is_up": True,
         "key": "interface_name",
         "rx": 3885,
         "speed": 0,
         "time_since_update": 1,
         "tx": 3885}]}
```

1.7.16 GET now

Get plugin stats:

```
# curl http://localhost:61208/api/3/now
"2023-05-20 16:44:16 CEST"
```

1.7.17 GET percpu

Get plugin stats:

```
# curl http://localhost:61208/api/3/percpu
[{"cpu_number": 0,
  "guest": 1.5,
  "guest_nice": 0.0,
  "idle": 26.2,
  "iowait": 1.5,
  "irq": 0.0,
  "key": "cpu_number",
  "nice": 0.0,
  "softirq": 0.0,
  "steal": 0.0,
  "system": 4.4,
  "total": 73.8,
  "user": 68.0},
 {"cpu_number": 1,
  "guest": 0.0,
  "guest_nice": 0.0,
  "idle": 39.4,
  "iowait": 1.0,
  "irq": 0.0,
  "key": "cpu_number",
  "nice": 0.0,
  "softirq": 0.0,
  "steal": 0.0,
  "system": 4.8,
  "total": 60.6,
  "user": 54.8}]
```

Get a specific field:

```
# curl http://localhost:61208/api/3/percpu/cpu_number
{"cpu_number": [0, 1, 2, 3]}
```

1.7.18 GET ports

Get plugin stats:

```
# curl http://localhost:61208/api/3/ports
[{"description": "DefaultGateway",
  "host": "192.168.0.254",
  "indice": "port_0",
  "port": 0,
  "refresh": 30,
  "rtt_warning": None,
  "status": 0.00792,
  "timeout": 3}]
```

Get a specific field:

```
# curl http://localhost:61208/api/3/ports/host
{"host": ["192.168.0.254"]}
```

Get a specific item when field matchs the given value:

```
# curl http://localhost:61208/api/3/ports/host/192.168.0.254
{"192.168.0.254": [{"description": "DefaultGateway",
                    "host": "192.168.0.254",
                    "indice": "port_0",
                    "port": 0,
                    "refresh": 30,
                    "rtt_warning": None,
                    "status": 0.00792,
                    "timeout": 3}]}
```

1.7.19 GET processcount

Get plugin stats:

```
# curl http://localhost:61208/api/3/processcount
{"pid_max": 0, "running": 1, "sleeping": 347, "thread": 1761, "total": 415}
```

Get a specific field:

```
# curl http://localhost:61208/api/3/processcount/total
{"total": 415}
```

1.7.20 GET processlist

Get plugin stats:

```
# curl http://localhost:61208/api/3/processlist
[{"cmdline": ["/snap/firefox/2605/usr/lib/firefox/firefox"],
  "cpu_percent": 0.0,
  "cpu_times": [16463.82, 5059.85, 11961.09, 1725.84, 0.0],
  "gids": [1000, 1000, 1000],
  "io_counters": [9699624960, 15455608832, 0, 0, 0],
  "key": "pid",
  "memory_info": [524517376, 22345007104, 90595328, 618496, 0, 1481629696, 0],
  "memory_percent": 6.693530134632704,
  "name": "firefox",
  "nice": 0,
  "num_threads": 171,
  "pid": 10541,
  "status": "S",
  "time_since_update": 1,
  "username": "nicolargo"},
 {"cmdline": ["/snap/multipass/8465/usr/bin/qemu-system-x86_64",
              "-bios",
              "OVMF.fd",
              "--enable-kvm",
              "-cpu",
              "host",
              "-nic",
              "tap,ifname=tap-1e376645a40,script=no,downscript=no,model=virtio-net-
↪pci,mac=52:54:00:05:05:17",
              "-device",
              "virtio-scsi-pci,id=scsi0",
              "-drive",
```

(continues on next page)

(continued from previous page)

```
              "file=/var/snap/multipass/common/data/multipassd/vault/instances/
↪primary/ubuntu-22.04-server-cloudimg-amd64.img,if=none,format=qcow2,discard=unmap,
↪id=hda",
              "-device",
              "scsi-hd,drive=hda,bus=scsi0.0",
              "-smp",
              "1",
              "-m",
              "1024M",
              "-qmp",
              "stdio",
              "-chardev",
              "null,id=char0",
              "-serial",
              "chardev:char0",
              "-nographic",
              "-cdrom",
              "/var/snap/multipass/common/data/multipassd/vault/instances/primary/
↪cloud-init-config.iso"],
  "cpu_percent": 0.0,
  "cpu_times": [846.85, 90.96, 0.0, 0.0, 0.0],
  "gids": [0, 0, 0],
  "io_counters": [0, 0, 0, 0, 0],
  "key": "pid",
  "memory_info": [510238720, 3458437120, 2822144, 5304320, 0, 1366933504, 0],
  "memory_percent": 6.511315743668364,
  "name": "qemu-system-x86_64",
  "nice": 0,
  "num_threads": 4,
  "pid": 354319,
  "status": "S",
  "time_since_update": 1,
  "username": "root"}]
```

Get a specific field:

```
# curl http://localhost:61208/api/3/processlist/pid
{"pid": [10541,
         354319,
         10770,
         11043,
         374779,
         374071,
         3927,
         469948,
         374587,
         317865,
         10778,
         10774,
         429788,
         59195,
         469241,
         399766,
         430971,
         372037,
         11646,
         59069,
```

(continues on next page)

(continued from previous page)

```
         374904,
         10733,
         480322,
         480143,
         480228,
         374842,
         435889,
         10790,
         59161,
         480580,
         59523,
         4243,
         421,
         480591,
         466459,
         374575,
         374705,
         3810,
         457618,
         466460,
         374905,
         372303,
         4385,
         165661,
         417207,
         463383,
         1618,
         1771,
         372048,
         372151,
         2398,
         372186,
         372172,
         59182,
         4339,
         374703,
         313257,
         2636,
         431242,
         374702,
         4023,
         4666,
         1,
         3730,
         4179,
         59663,
         1584,
         427863,
         4075,
         10710,
         17997,
         430855,
         4308,
         4091,
         1630,
         1605,
         4403,
```

(continues on next page)

(continued from previous page)

```
         4000,
         4009,
         4090,
         3991,
         3719,
         11381,
         11380,
         1794,
         372168,
         431219,
         479905,
         372170,
         354726,
         4169,
         372169,
         4086,
         1727,
         4033,
         3745,
         4105,
         3901,
         3710,
         4442,
         3908,
         4046,
         4127,
         36919,
         1631,
         3743,
         14243,
         1583,
         4302,
         3956,
         1379,
         418247,
         74953,
         4126,
         3748,
         3115,
         20173,
         14266,
         1591,
         4196,
         59126,
         2116,
         4005,
         1764,
         4316,
         4145,
         4097,
         4080,
         1627,
         2168,
         2607,
         1818,
         3989,
         1579,
```

(continues on next page)

(continued from previous page)

```
        1628,
        3925,
        4079,
        4244,
        4157,
        3970,
        59127,
        2554,
        4099,
        1612,
        3819,
        1380,
        4078,
        10848,
        2341,
        1566,
        227509,
        1598,
        1624,
        4119,
        4074,
        3825,
        3947,
        4098,
        4062,
        3939,
        4107,
        3975,
        3753,
        3952,
        1575,
        1606,
        313277,
        461,
        354741,
        1593,
        480538,
        3934,
        3728,
        12480,
        3888,
        1616,
        59145,
        12489,
        1377,
        1582,
        18045,
        4332,
        1964,
        1825,
        3727,
        59130,
        1634,
        16182,
        1391,
        2361,
        2605,
```

(continues on next page)

(continued from previous page)

```
         1577,
         3118,
         2604,
         1390,
         12483,
         1567,
         431184,
         20396,
         354739,
         431178,
         431203,
         20180,
         480579,
         431197,
         4072,
         2358,
         469137,
         3503,
         12492,
         1726,
         1725,
         3794,
         3498,
         3499,
         3720,
         20400,
         313283,
         2345,
         2382,
         4593,
         2360,
         1392,
         1637,
         3573,
         20185,
         2,
         3,
         4,
         5,
         6,
         8,
         10,
         11,
         12,
         13,
         14,
         15,
         16,
         18,
         19,
         20,
         21,
         22,
         24,
         25,
         26,
         27,
```

(continues on next page)

(continued from previous page)

```
        28,
        30,
        31,
        32,
        33,
        34,
        36,
        37,
        38,
        39,
        40,
        41,
        42,
        43,
        44,
        45,
        92,
        93,
        94,
        96,
        97,
        98,
        99,
        100,
        101,
        103,
        106,
        107,
        109,
        110,
        112,
        117,
        118,
        119,
        129,
        132,
        138,
        181,
        183,
        206,
        219,
        223,
        226,
        228,
        231,
        232,
        233,
        234,
        249,
        250,
        255,
        256,
        313,
        361,
        362,
        439,
        440,
```

(continues on next page)

(continued from previous page)

```
        530,
        544,
        655,
        700,
        702,
        703,
        898,
        899,
        900,
        901,
        908,
        909,
        910,
        911,
        912,
        913,
        914,
        915,
        962,
        963,
        964,
        965,
        966,
        967,
        968,
        969,
        970,
        971,
        972,
        973,
        974,
        975,
        976,
        977,
        978,
        979,
        980,
        1001,
        1002,
        1009,
        1010,
        1031,
        1032,
        1033,
        1034,
        1035,
        1036,
        1037,
        2394,
        2410,
        2422,
        2491,
        2492,
        2493,
        2506,
        2508,
        2510,
```

(continues on next page)

(continued from previous page)

```
         2515,
         2525,
         3988,
         12486,
         313404,
         354325,
         354329,
         417249,
         417250,
         417251,
         417252,
         417254,
         417255,
         417257,
         417258,
         417259,
         417260,
         417261,
         417262,
         417263,
         417264,
         417265,
         417266,
         417267,
         427608,
         427609,
         427611,
         454059,
         459092,
         463267,
         463268,
         463271,
         463272,
         463274,
         463320,
         467345,
         467496,
         468536,
         468902,
         469637,
         469929,
         470092,
         471704,
         476042,
         477638,
         478302,
         479060,
         479698,
         479767,
         480036,
         480278,
         480424,
         480454]}
```

Get a specific item when field matchs the given value:

```
# curl http://localhost:61208/api/3/processlist/pid/10541
{"10541": [{"cmdline": ["/snap/firefox/2605/usr/lib/firefox/firefox"],
            "cpu_percent": 0.0,
            "cpu_times": [16463.82, 5059.85, 11961.09, 1725.84, 0.0],
            "gids": [1000, 1000, 1000],
            "io_counters": [9699624960, 15455608832, 0, 0, 0],
            "key": "pid",
            "memory_info": [524517376,
                            22345007104,
                            90595328,
                            618496,
                            0,
                            1481629696,
                            0],
            "memory_percent": 6.693530134632704,
            "name": "firefox",
            "nice": 0,
            "num_threads": 171,
            "pid": 10541,
            "status": "S",
            "time_since_update": 1,
            "username": "nicolargo"}]}
```

1.7.21 GET psutilversion

Get plugin stats:

```
# curl http://localhost:61208/api/3/psutilversion
[5, 9, 5]
```

1.7.22 GET quicklook

Get plugin stats:

```
# curl http://localhost:61208/api/3/quicklook
{"cpu": 62.2,
 "cpu_hz": 2025000000.0,
 "cpu_hz_current": 1273980750.0,
 "cpu_name": "Intel(R) Core(TM) i7-4500U CPU @ 1.80GHz",
 "mem": 75.0,
 "percpu": [{"cpu_number": 0,
             "guest": 1.5,
             "guest_nice": 0.0,
             "idle": 26.2,
             "iowait": 1.5,
             "irq": 0.0,
             "key": "cpu_number",
             "nice": 0.0,
             "softirq": 0.0,
             "steal": 0.0,
             "system": 4.4,
             "total": 73.8,
             "user": 68.0},
            {"cpu_number": 1,
```

(continues on next page)

(continued from previous page)

```
             "guest": 0.0,
             "guest_nice": 0.0,
             "idle": 39.4,
             "iowait": 1.0,
             "irq": 0.0,
             "key": "cpu_number",
             "nice": 0.0,
             "softirq": 0.0,
             "steal": 0.0,
             "system": 4.8,
             "total": 60.6,
             "user": 54.8},
            {"cpu_number": 2,
             "guest": 0.5,
             "guest_nice": 0.0,
             "idle": 36.0,
             "iowait": 4.3,
             "irq": 0.0,
             "key": "cpu_number",
             "nice": 0.0,
             "softirq": 0.9,
             "steal": 0.0,
             "system": 9.5,
             "total": 64.0,
             "user": 49.3},
            {"cpu_number": 3,
             "guest": 2.8,
             "guest_nice": 0.0,
             "idle": 36.6,
             "iowait": 6.5,
             "irq": 0.0,
             "key": "cpu_number",
             "nice": 0.0,
             "softirq": 0.0,
             "steal": 0.0,
             "system": 12.5,
             "total": 63.4,
             "user": 44.4}],
 "swap": 62.7}
```

Get a specific field:

```
# curl http://localhost:61208/api/3/quicklook/cpu
{"cpu": 62.2}
```

1.7.23 GET sensors

Get plugin stats:

```
# curl http://localhost:61208/api/3/sensors
[{"critical": 105,
  "key": "label",
  "label": "acpitz 0",
  "type": "temperature_core",
  "unit": "C",
```

(continues on next page)

(continued from previous page)

```
  "value": 27,
  "warning": 105},
 {"critical": 105,
  "key": "label",
  "label": "acpitz 1",
  "type": "temperature_core",
  "unit": "C",
  "value": 29,
  "warning": 105}]
```

Get a specific field:

```
# curl http://localhost:61208/api/3/sensors/label
{"label": ["acpitz 0",
           "acpitz 1",
           "Package id 0",
           "Core 0",
           "Core 1",
           "CPU",
           "Ambient",
           "SODIMM",
           "BAT BAT0"]}
```

Get a specific item when field matchs the given value:

```
# curl http://localhost:61208/api/3/sensors/label/acpitz 0
{"acpitz 0": [{"critical": 105,
               "key": "label",
               "label": "acpitz 0",
               "type": "temperature_core",
               "unit": "C",
               "value": 27,
               "warning": 105}]}
```

1.7.24 GET system

Get plugin stats:

```
# curl http://localhost:61208/api/3/system
{"hostname": "XPS13-9333",
 "hr_name": "Ubuntu 22.04 64bit",
 "linux_distro": "Ubuntu 22.04",
 "os_name": "Linux",
 "os_version": "5.15.0-71-generic",
 "platform": "64bit"}
```

Get a specific field:

```
# curl http://localhost:61208/api/3/system/os_name
{"os_name": "Linux"}
```

1.7.25 GET uptime

Get plugin stats:

```
# curl http://localhost:61208/api/3/uptime
"12 days, 3:42:29"
```

1.7.26 GET all stats

Get all Glances stats:

```
# curl http://localhost:61208/api/3/all
Return a very big dictionnary (avoid using this request, performances will be poor)...
```

1.7.27 GET stats history

History of a plugin:

```
# curl http://localhost:61208/api/3/cpu/history
{"system": [["2023-05-20T16:44:17.685943", 7.7],
            ["2023-05-20T16:44:18.817737", 7.7],
            ["2023-05-20T16:44:19.996995", 1.1]],
 "user": [["2023-05-20T16:44:17.685935", 50.0],
          ["2023-05-20T16:44:18.817731", 50.0],
          ["2023-05-20T16:44:19.996988", 4.3]]}
```

Limit history to last 2 values:

```
# curl http://localhost:61208/api/3/cpu/history/2
{"system": [["2023-05-20T16:44:18.817737", 7.7],
            ["2023-05-20T16:44:19.996995", 1.1]],
 "user": [["2023-05-20T16:44:18.817731", 50.0],
          ["2023-05-20T16:44:19.996988", 4.3]]}
```

History for a specific field:

```
# curl http://localhost:61208/api/3/cpu/system/history
{"system": [["2023-05-20T16:44:17.685943", 7.7],
            ["2023-05-20T16:44:18.817737", 7.7],
            ["2023-05-20T16:44:19.996995", 1.1]]}
```

Limit history for a specific field to last 2 values:

```
# curl http://localhost:61208/api/3/cpu/system/history
{"system": [["2023-05-20T16:44:18.817737", 7.7],
            ["2023-05-20T16:44:19.996995", 1.1]]}
```

1.7.28 GET limits (used for thresholds)

All limits/thresholds:

```
# curl http://localhost:61208/api/3/all/limits
{"alert": {"history_size": 1200.0},
 "amps": {"amps_disable": ["False"], "history_size": 1200.0},
 "containers": {"containers_all": ["False"],
                "containers_disable": ["False"],
```

(continues on next page)

(continued from previous page)

```
                "containers_max_name_size": 20.0,
                "history_size": 1200.0},
 "core": {"history_size": 1200.0},
 "cpu": {"cpu_ctx_switches_careful": 160000.0,
         "cpu_ctx_switches_critical": 200000.0,
         "cpu_ctx_switches_warning": 180000.0,
         "cpu_disable": ["False"],
         "cpu_iowait_careful": 20.0,
         "cpu_iowait_critical": 25.0,
         "cpu_iowait_warning": 22.5,
         "cpu_steal_careful": 50.0,
         "cpu_steal_critical": 90.0,
         "cpu_steal_warning": 70.0,
         "cpu_system_careful": 50.0,
         "cpu_system_critical": 90.0,
         "cpu_system_log": ["False"],
         "cpu_system_warning": 70.0,
         "cpu_total_careful": 65.0,
         "cpu_total_critical": 85.0,
         "cpu_total_log": ["True"],
         "cpu_total_warning": 75.0,
         "cpu_user_careful": 50.0,
         "cpu_user_critical": 90.0,
         "cpu_user_log": ["False"],
         "cpu_user_warning": 70.0,
         "history_size": 1200.0},
 "diskio": {"diskio_disable": ["False"],
            "diskio_hide": ["loop.*", "/dev/loop.*"],
            "history_size": 1200.0},
 "folders": {"folders_disable": ["False"], "history_size": 1200.0},
 "fs": {"fs_careful": 50.0,
        "fs_critical": 90.0,
        "fs_disable": ["False"],
        "fs_hide": ["/boot.*", "/snap.*"],
        "fs_warning": 70.0,
        "history_size": 1200.0},
 "gpu": {"gpu_disable": ["False"],
         "gpu_mem_careful": 50.0,
         "gpu_mem_critical": 90.0,
         "gpu_mem_warning": 70.0,
         "gpu_proc_careful": 50.0,
         "gpu_proc_critical": 90.0,
         "gpu_proc_warning": 70.0,
         "history_size": 1200.0},
 "help": {"history_size": 1200.0},
 "ip": {"history_size": 1200.0,
        "ip_censys_fields": ["location:continent",
                             "location:country",
                             "autonomous_system:name"],
        "ip_censys_url": ["https://search.censys.io/api"],
        "ip_disable": ["False"],
        "ip_public_ip_disabled": ["False"],
        "ip_public_refresh_interval": 300.0},
 "load": {"history_size": 1200.0,
          "load_careful": 0.7,
          "load_critical": 5.0,
          "load_disable": ["False"],
```

(continues on next page)

(continued from previous page)

```
         "load_warning": 1.0},
"mem": {"history_size": 1200.0,
        "mem_careful": 50.0,
        "mem_critical": 90.0,
        "mem_disable": ["False"],
        "mem_warning": 70.0},
"memswap": {"history_size": 1200.0,
            "memswap_careful": 50.0,
            "memswap_critical": 90.0,
            "memswap_disable": ["False"],
            "memswap_warning": 70.0},
"network": {"history_size": 1200.0,
            "network_disable": ["False"],
            "network_rx_careful": 70.0,
            "network_rx_critical": 90.0,
            "network_rx_warning": 80.0,
            "network_tx_careful": 70.0,
            "network_tx_critical": 90.0,
            "network_tx_warning": 80.0},
"now": {"history_size": 1200.0},
"percpu": {"history_size": 1200.0,
           "percpu_disable": ["False"],
           "percpu_iowait_careful": 50.0,
           "percpu_iowait_critical": 90.0,
           "percpu_iowait_warning": 70.0,
           "percpu_system_careful": 50.0,
           "percpu_system_critical": 90.0,
           "percpu_system_warning": 70.0,
           "percpu_user_careful": 50.0,
           "percpu_user_critical": 90.0,
           "percpu_user_warning": 70.0},
"ports": {"history_size": 1200.0,
          "ports_disable": ["False"],
          "ports_port_default_gateway": ["True"],
          "ports_refresh": 30.0,
          "ports_timeout": 3.0},
"processcount": {"history_size": 1200.0, "processcount_disable": ["False"]},
"processlist": {"history_size": 1200.0,
                "processlist_cpu_careful": 50.0,
                "processlist_cpu_critical": 90.0,
                "processlist_cpu_warning": 70.0,
                "processlist_disable": ["False"],
                "processlist_mem_careful": 50.0,
                "processlist_mem_critical": 90.0,
                "processlist_mem_warning": 70.0,
                "processlist_nice_warning": ["-20",
                                             "-19",
                                             "-18",
                                             "-17",
                                             "-16",
                                             "-15",
                                             "-14",
                                             "-13",
                                             "-12",
                                             "-11",
                                             "-10",
                                             "-9",
```

(continues on next page)

(continued from previous page)

```
                                                  "-8",
                                                  "-7",
                                                  "-6",
                                                  "-5",
                                                  "-4",
                                                  "-3",
                                                  "-2",
                                                  "-1",
                                                  "1",
                                                  "2",
                                                  "3",
                                                  "4",
                                                  "5",
                                                  "6",
                                                  "7",
                                                  "8",
                                                  "9",
                                                  "10",
                                                  "11",
                                                  "12",
                                                  "13",
                                                  "14",
                                                  "15",
                                                  "16",
                                                  "17",
                                                  "18",
                                                  "19"]},
 "psutilversion": {"history_size": 1200.0},
 "quicklook": {"history_size": 1200.0,
               "quicklook_cpu_careful": 50.0,
               "quicklook_cpu_critical": 90.0,
               "quicklook_cpu_warning": 70.0,
               "quicklook_disable": ["False"],
               "quicklook_mem_careful": 50.0,
               "quicklook_mem_critical": 90.0,
               "quicklook_mem_warning": 70.0,
               "quicklook_percentage_char": ["|"],
               "quicklook_swap_careful": 50.0,
               "quicklook_swap_critical": 90.0,
               "quicklook_swap_warning": 70.0},
 "sensors": {"history_size": 1200.0,
             "sensors_battery_careful": 80.0,
             "sensors_battery_critical": 95.0,
             "sensors_battery_warning": 90.0,
             "sensors_disable": ["False"],
             "sensors_refresh": 4.0,
             "sensors_temperature_core_careful": 60.0,
             "sensors_temperature_core_critical": 80.0,
             "sensors_temperature_core_warning": 70.0,
             "sensors_temperature_hdd_careful": 45.0,
             "sensors_temperature_hdd_critical": 60.0,
             "sensors_temperature_hdd_warning": 52.0},
 "system": {"history_size": 1200.0,
            "system_disable": ["False"],
            "system_refresh": 60},
 "uptime": {"history_size": 1200.0}}
```

Limits/thresholds for the cpu plugin:

```
# curl http://localhost:61208/api/3/cpu/limits
{"cpu_ctx_switches_careful": 160000.0,
 "cpu_ctx_switches_critical": 200000.0,
 "cpu_ctx_switches_warning": 180000.0,
 "cpu_disable": ["False"],
 "cpu_iowait_careful": 20.0,
 "cpu_iowait_critical": 25.0,
 "cpu_iowait_warning": 22.5,
 "cpu_steal_careful": 50.0,
 "cpu_steal_critical": 90.0,
 "cpu_steal_warning": 70.0,
 "cpu_system_careful": 50.0,
 "cpu_system_critical": 90.0,
 "cpu_system_log": ["False"],
 "cpu_system_warning": 70.0,
 "cpu_total_careful": 65.0,
 "cpu_total_critical": 85.0,
 "cpu_total_log": ["True"],
 "cpu_total_warning": 75.0,
 "cpu_user_careful": 50.0,
 "cpu_user_critical": 90.0,
 "cpu_user_log": ["False"],
 "cpu_user_warning": 70.0,
 "history_size": 1200.0}
```

1.8 Docker

Glances can be installed through Docker, allowing you to run it without installing all the python dependencies directly on your system. Once you have [docker installed](https://docs.docker.com/install/), you can

Get the Glances container:

```
docker pull nicolargo/glances:<version>
```

Available tags (all images are based on both Alpine and Ubuntu Operating System):

- *latest-full* for a full Alpine Glances image (latest release) with all dependencies
- *latest* for a basic Alpine Glances (latest release) version with minimal dependencies (Bottle and Docker)
- *dev* for a basic Alpine Glances image (based on development branch) with all dependencies (Warning: may be instable)
- *ubuntu-latest-full* for a full Ubuntu Glances image (latest release) with all dependencies
- *ubuntu-latest* for a basic Ubuntu Glances (latest release) version with minimal dependencies (Bottle and Docker)
- *ubuntu-dev* for a basic Ubuntu Glances image (based on development branch) with all dependencies (Warning: may be instable)

You can also specify a version (example: 3.4.0). All available versions can be found on DockerHub.

An Example to pull the *latest* tag:

```
docker pull nicolargo/glances:latest
```

Run the container in *console mode*:

```
docker run --rm -v /var/run/docker.sock:/var/run/docker.sock:ro --pid host --network␣
↪host -it docker.io/nicolargo/glances
```

Additionally, if you want to use your own glances.conf file, you can create your own Dockerfile:

```
FROM nicolargo/glances
COPY glances.conf /glances/conf/glances.conf
CMD python -m glances -C /glances/conf/glances.conf $GLANCES_OPT
```

Alternatively, you can specify something along the same lines with docker run options:

```
docker run -v `pwd`/glances.conf:/glances/conf/glances.conf -v /var/run/docker.sock:/
↪var/run/docker.sock:ro --pid host -it docker.io/nicolargo/glances
```

Where ‘pwd‘/glances.conf is a local directory containing your glances.conf file.

Run the container in *Web server mode* (notice the *GLANCES_OPT* environment variable setting parameters for the glances startup command):

```
docker run -d --restart="always" -p 61208-61209:61208-61209 -e GLANCES_OPT="-w" -v /
↪var/run/docker.sock:/var/run/docker.sock:ro --pid host docker.io/nicolargo/glances
```

Note: if you want to see the network interface stats within the container, add –net=host –privileged

You can also include Glances container in you own *docker-compose.yml*. Here’s a realistic example including a “traefik” reverse proxy serving an “whoami” app container plus a Glances container, providing a simple and efficient monitoring webui.

```
version: '3'

services:
  reverse-proxy:
    image: traefik:alpine
    command: --api --docker
    ports:
      - "80:80"
      - "8080:8080"
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock

  whoami:
    image: emilevauge/whoami
    labels:
      - "traefik.frontend.rule=Host:whoami.docker.localhost"

  monitoring:
    image: nicolargo/glances:latest-alpine
    restart: always
    pid: host
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    environment:
      - "GLANCES_OPT=-w"
    labels:
      - "traefik.port=61208"
      - "traefik.frontend.rule=Host:glances.docker.localhost"
```

1.8.1 How to protect your Dockerized server (or Web server) with a login/password ?

Below are two methods for setting up a login/password to protect Glances running inside a Docker container.

Option 1

You can enter the running container by entering this command (replacing `glances_docker` with the name of your container):

```
docker exec -it glances_docker sh
```

and generate the password file (the default login is `glances`, add the `--username` flag if you would like to change it):

```
glances -s --password
```

which will prompt you to answer the following questions:

```
Define the Glances server password (glances username):
Password (confirm):
Do you want to save the password? [Yes/No]: Yes
```

after which you will need to kill the process by entering `CTRL+C` (potentially twice), before leaving the container:

```
exit
```

You will then need to copy the password file to your host machine:

```
docker cp glances_docker:/root/.config/glances/glances.pwd ./secrets/glances_password
```

and make it visible to your container by adding it to `docker-compose.yml` as a `secret`:

```yaml
version: '3'

services:
  glances:
    image: nicolargo/glances:latest
    restart: always
    environment:
      - GLANCES_OPT="-w --password"
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock:ro
    pid: host
    secrets:
      - source: glances_password
        target: /root/.config/glances/glances.pwd

secrets:
  glances_password:
    file: ./secrets/glances_password
```

Option 2

You can add a `[passwords]` block to the Glances configuration file as mentioned elsewhere in the documentation:

```
[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
```

1.8.2 Using GPU Plugin with Docker (Only Nvidia GPUs)

Complete the steps mentioned in the docker docs to make the GPU accessible by the docker engine.

With *docker run*

Include the *–gpus* flag with the *docker run* command.

Note: Make sure the *–gpus* is present before the image name in the command, otherwise it won't work.

```
docker run --rm -v /var/run/docker.sock:/var/run/docker.sock:ro --gpus --pid host --
↪network host -it docker.io/nicolargo/glances:latest-full
```

With *docker-compose*

Include the *deploy* section in compose file as specified below in the example service definition.

```
version: '3'

services:
  monitoring:
    image: nicolargo/glances:latest-full
    pid: host
    network_mode: host
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    environment:
      - "GLANCES_OPT=-w"
    # For nvidia GPUs
    deploy:
      resources:
        reservations:
          devices:
            - driver: nvidia
              count: 1
              capabilities: [gpu]
```

Reference: https://docs.docker.com/compose/gpu-support/

1.9 F.A.Q

Any encoding issue ?

Try to run Glances with the following command line:

```
LANG=en_US.UTF-8 LC_ALL= glances
```

1.10 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!

Index