NAME

atop - Advanced System & Process Monitor

SYNOPSIS

Live measurement in bar graph mode:

atop −B[H] [-t [absdir]] [interval [samples]]

Live measurement cgroups in text mode:

atop −G [-t [absdir]] [-2|-3|-4|-5|-6|-7|-8|-9] [−a] [−C|−M|−D|−A] [interval [samples]]

Live measurement processes in text mode:

atop [-t [absdir]] [−g|−m|−d|−n|−u|−p|−s|−c|−v|−o|−y|−Y] [−C|−M|−D|−N|−A] [−fFX1xR] [interval [samples]]

Live generation of parsable output (white-space separated or JSON):

atop [−Plabel[,label]... [-Z]] [−Jlabel[,label]...] [interval [samples]]

Write raw log files:

atop −w rawfile [−a] [−S] [interval [samples]]

Analyze raw log files in bar graph mode:

atop −B[H] -r [rawfile|yyy...] [−b [YYYYMMDD]hhmm[ss]] [−e [YYYYMMDD]hhmm[ss]]

Analyze cgroups from raw log files in text mode:

atop −G [-2|-3|-4|-5|-6|-7|-8|-9] [−a] −r [rawfile|yyy...] [−b [YYYYMMDD]hhmm[ss]] [−e [YYYYMMDD]hhmm[ss]]

Analyze processes from raw log files in text mode:

atop −r [rawfile|yyy...] [−b [YYYYMMDD]hhmm[ss]] [−e [YYYYMMDD]hhmm[ss]] [−g|−m|−d|−n|−u|−p|−s|−c|−v|−o|−y|−Y] [−C|−M|−D|−N|−A] [−fFX1xR]

Generate parsable output from raw log files (white-space separated or JSON):

atop −r [rawfile|yyy...] [−b [YYYYMMDD]hhmm[ss]] [−e [YYYYMMDD]hhmm[ss]] [−Plabel[,label]... [-Z]] [−Jlabel[,label]...]

DESCRIPTION

The program *atop* is an interactive monitor to view the load on a Linux system. Every *interval* seconds (default: 10 seconds) information is gathered about the resource occupation on system level (CPUs, memory, disks and network interfaces) and on cgroup level (version 2). Besides, information is gathered about the processes and threads that are responsible for the utilization of the CPUs, memory and disks. Network load per process is shown only when the *netatop* kernel module or the *netatop-bpf* BPF module has been installed.

TWIN MODE

With the *-t* flag you can run *atop* interactively in 'twin mode'. This mode allows to run a live measurement with the possibility to review and analyze an earlier sample. Meanwhile, the live measurement continues.

When started in twin mode, *atop* spawns a child process that gathers the counters and writes them to a temporary raw file. The parent process reads the counters from the temporary raw file and presents them to the user. The reading of the parent process keeps in pace with the written samples of the child process for live measurements.

While the gathering continues by the child process, key 'r', key 'b' or key 'T' can be pressed to review earlier samples. The parent process implicitly pauses the live measurement by pressing one of these keys or by pressing key 'z' (pause) explicitly.

After browsing through the earlier samples with the keys 't' (next sample), 'T' (previous sample), 'r' (reset to begin of measurement), 'Z' (fast-forward to end of measurement) and 'b' (branch to timestamp), the live measurement can be continued by pressing key 'z' (resume after pause).

The temporary raw file will be written in the **/tmp** directory by default. Optionally the absolute path name of an alternative directory can be added behind the *-t* flag (e.g. when there is not enough space in the **/tmp** directory). In any case, the parent process will terminate the child process when the measurement is finished and the temporary raw file will be removed.

BAR GRAPH MODE

When running *atop* you can choose to view the system load in bar graph mode or in text mode. In bar graph mode the resource utilization of CPUs, memory, disks and network interfaces is shown via (character-based) bar graphs, but only on system level. When you want to view more detailed information on system level or when you want to view the resource consumption on process or thread level, you can switch to text mode by pressing the 'B' key. Alternatively, you can use the 'B' key (again) to switch from text mode to bar graph mode.
By default, *atop* starts in text mode unless the *-B* flag is used or unless 'B' has been configured as a default flag in the *.atoprc* file (for further information about default flags, refer to the **atoprc** man page).

In bar graph mode the terminal will be subdivided into four character-based windows, i.e. one window for each hardware resource:

Processors

The first bar shows the average busy percentage of all CPUs with the bar label 'Avg' (might be abbreviated to 'Av' or even just 'A'). The subsequent bars show the busy percentages of single CPUs.
When there is not enough horizontal space to show all CPUs, only the most busy CPUs per sample will be shown after the width of each bar has been reduced to a minimum.

By default, the categories of CPU consumption are shown by different colors in the bars, marked with a character 'S' (system mode), 'U' (user mode), 'I' (interrupt handling), 's' (steal) and 'G' (guest, i.e. consumed by virtual machines).
The top of the bar might consist of an unmarked color representing a 'neutral' category. Suppose that the scale unit is 5% per line and the total busy percentage is 54% consisting of two categories of 27%. The two categories will be rounded to 25% (5 lines of 5% each) but the total busy percentage will be rounded to 55% (11 lines of 5%). Then the top line will represent a 'neutral' category.
By pressing the 'H' key or by starting *atop* with the '-H' flag, no categories are shown.

A red line is drawn in the bar graph as critical threshold. By default this value is 90% and can be modified by the 'cpucritperc' option in the configuration file (see separate **atoprc** man page). When this value is set to zero, no threshold line will be drawn.

Memory and swap space

Memory is presented as a column in which the specific categories of memory consumption are shown. These categories are (code, data and stack of) processes/kernel, slab caches (i.e. dynamically allocated kernel memory), shared memory, tmpfs, static huge pages, page cache and free memory.
Swap space (if present) is also presented as a column in which the categories processes/tmpfs, shared memory and free space are shown.

At the right side memory-related event counters are shown.
The bottom three counters are colored green when there is no memory pressure. When considerable activity is noticed such counter might be colored orange and with high activity red.
When memory pressure starts, usually memory page scanning will be activated first. When pressure increases, memory pages of processes might be swapped out to swap space (if present).
The 'oomkills' counter (Out Of Memory killing) is most serious: it reflects the number of processes that are killed due to lack of memory (and swap). Therefore this counter shows the absolute number (not per second) of processes being killed during the last interval and will immediately be colored red when it is 1 or more. Besides, after *atop* has noticed OOM killing the 'oomkills' counter remains

orange for the next 15 minutes, just in case that you have missed the OOM killing event itself.
When there is enough vertical space in the memory window, event counters are shown about the number of memory pages being swapped in, the number of memory pages paged out to block devices and the number of memory pages paged in from block devices.

Memory and swap space consumption will preferably be shown in a character-based window that vertically uses the entire screen for optimal granularity. However, when there are a lot of disks and/or network interfaces the memory and swap space consumption will be shown in a character-based window that only uses the upper half of the screen.

Disks

For each disk the busy percentage is shown as a bar.
When there is not enough horizontal space to show all disks, only the most busy disks per sample will be shown.

By default, categories of disk consumption are shown by different colors in the bars, marked with a character 'R' (read) and 'W' (write).
The top of the bar might consist of an unmarked color representing a 'neutral' category. Suppose that the scale unit is 5% per line and the total busy percentage is 54% consisting of two categories of 27%. The two categories will be rounded to 25% (5 lines of 5% each) but the total busy percentage will be rounded to 55% (11 lines of 5%). Then the top line will represent a 'neutral' category.
By pressing the 'H' key or by starting *atop* with the '-H' flag, no categories are shown.

A red line is drawn in the bar graph as critical threshold. By default this value is 90% and can be modified by the 'dskcritperc' option in the configuration file (see separate **atoprc** man page). When this value is set to zero, no threshold line will be drawn.

Interfaces

For each non-virtual network interface a double bar graph is shown with a dedicated scale that reflects the traffic rate. One of the bars shows the transmit rate ('TX') and the other bar the receive rate ('RX'). The traffic scale of each network interface remains at its highest level. All interface scales can be reset during the measurement by pressing the 'L' key.

Most often the real speed (maximum bandwidth) of network interfaces is not known, e.g. in case of the network interfaces of virtual machines. Therefore it is not possible to show the interface utilization as a percentage. However, when the real speed of an interface is known it will be shown underneath the concerning bar graph.

When there is not enough horizontal space to show all network interfaces, only the most busy interfaces per sample will be shown.

Usually the bar graphs will not be sorted on busy percentage when there is enough horizontal space. However, after switching from text mode to bar graph mode the bar graphs might have been sorted because this was needed for the presentation in text mode. The next interval in bar graph mode shows the bars unsorted again unless the window width is unsufficient for all bars.

For the CPUs, disks and memory resources also a bar graph with two bars is shown reflecting the PSI values 'some' (S) and 'full' (F). The 'some' (S) percentage indicates the time in which at least some tasks were stalled on a given resource. The 'full' (F) percentage indicates the time in which all non-idle tasks were stalled on this resource simultaneously, which had severe impact on the performance.
For some Linux distributions PSI support might be disabled by default. In that case you need to pass **psi=1** on the kernel command line during boot.

The remaining part of this manual page mainly describes the information shown in text mode. When certain descriptions also apply to bar graph mode it will be mentioned explicitly.

TEXT MODE IN GENERAL

With every interval information is shown about the resource occupation on system level (CPU, memory, disks and network layers) in the upper part of the screen. When a resource has been unused during the interval, the line is suppressed unless the 'f' key is active.

In the bottom part of the screen cgroup level (key 'G') or process level information is shown (keys 'g', 'c', 's', 'm', 'd', 'n', 'v', 'u' or 'p'). By default, only cgroups are shown without assigned processes in the cgroup itself nor in the cgroups underneath, unless the 'a' key (all) is active. By default, only processes are shown that were active during the interval unless the 'a' key (all) is active.
The intervals are repeated till the number of *samples* (specified as command argument) is reached, or till the key 'q' is pressed in interactive mode.

When *atop* is started, it checks whether the standard output channel is connected to a screen, or to a file/pipe. In the first case it produces screen control codes (via the ncurses library) and behaves interactively. In the second case it produces flat text output.

In interactive mode, the output of *atop* scales dynamically to the current dimensions of the screen/window.
If the window is resized horizontally, columns will be added or removed automatically. For this purpose, every column has a particular weight. The columns with the highest weights that fit within the current width will be shown.
If the window is resized vertically, lines of the process/thread list will be added or removed automatically.

In interactive mode the output of *atop* can be controlled by pressing particular keys. However it is also possible to specify such key as **flag** on the command line. In that case *atop* switches to the indicated mode on beforehand. This mode can be modified again interactively. Specifying such key as flag is especially useful when running *atop* with output to a pipe or file (non-interactively). These flags are the same as the keys that can be pressed in interactive mode (see section INTERACTIVE COMMANDS).
Additional flags are available to support storage of atop-data in raw format (see section RAW DATA STORAGE).

By default, *atop* produces its output full-screen unless a flag is passed to direct the output to a raw log (-w) or direct the output in parseable (-P) or JSON output (-J). It is possible however to produce full-screen output while the output is also directed in another way. In that case a flag that relates to full-screen output (like -g) has to be passed on the command line explicitly.

The status line in the initial screen shows if *atop* runs with restricted view (as unprivileged user) or unrestricted view (as privileged user). In case of restricted view *atop* does not have the privileges (no root identity nor the necessary capabilities) to retrieve all counter values on system level and on process level.

PROCESS ACCOUNTING

With every interval, *atop* reads the kernel administration to obtain information about all running processes. However, it is likely that processes have terminated during the interval. These processes might have consumed system resources during this interval before they terminated. Therefore, *atop* tries to read the process accounting records that contain the accounting information of terminated processes and report these processes too. Only when the process accounting mechanism in the kernel is activated, the kernel writes such process accounting record to a file for every process that terminates.

There are various ways for *atop* to get access to the process accounting records (tried in this order):

1. When the environment variable ATOPACCT is set, it specifies the name of the process accounting file. In that case, process accounting for this file should have been activated on beforehand. Before opening this file for reading, *atop* drops its root privileges (if any).
   When this environment variable is present but its contents is empty, process accounting will not be used at all.

2. **This is the preferred way of handling process accounting records!**
   When the *atopacctd* daemon is active, it has activated the process accounting mechanism in the kernel and transfers to original accounting records to shadow files. In that case, *atop* drops its root privileges

and opens the current shadow file for reading.
This way is preferred, because the *atopacctd* daemon maintains full control of the size of the original process accounting file written by the kernel and the shadow files read by the *atop* process(es).

The *atopacct* service will be activated before the *atop* service to enable *atop* to detect that process accounting is managed by the *atopacctd* daemon. As a forking service, *atopacctd* takes care that all directories and files are initialized before the parent process dies. The child process continues as the daemon process.

For further information, refer to the **atopacctd** man page.

3. When the *atopacctd* daemon is not active, *atop* verifies if the process accounting mechanism has been switched on via the separate **psacct** or **acct** package (the package name depends on the Linux distro). In that case, one of the files **/var/log/pacct, /var/account/pacct** or **/var/log/account/pacct** is in use as process accounting file and *atop* opens this file for reading.

4. As a last possibility, *atop* itself tries to activate the process accounting mechanism (requires root privileges) using the file **/var/cache/atop.d/atop.acct** (to be written by the kernel, to be read by *atop* itself). Process accounting remains active as long as at least one *atop* process is alive. Whenever the last *atop* process stops (either by pressing 'q' or by 'kill −15'), it deactivates the process accounting mechanism again. Therefore you should never terminate *atop* by 'kill −9', because then it has no chance to stop process accounting. As a result, the accounting file may consume a lot of disk space after a while.
To avoid that the process accounting file consumes too much disk space, *atop* verifies at the end of every sample if the size of the process accounting file exceeds 200 MiB and if this *atop* process is the only one that is currently using the file. In that case the file is truncated to a size of zero.

Notice that root-privileges are required to switch on/off process accounting in the kernel. You can start *atop* as a root user or specify setuid-root privileges to the executable file. In the latter case, *atop* switches on process accounting and drops the root-privileges again.
If *atop* does not run with root-privileges, it does not show information about finished processes. It indicates this situation with the message 'no procacct' in the top-right corner (instead of the counter that shows the number of exited processes).

When during one interval a lot of processes have finished, *atop* might grow tremendously in memory when reading all process accounting records at the end of the interval. To avoid such excessive growth *atop* will never read more than 50 MiB with process information from the process accounting file per interval (approx. 54000 finished processes). In interactive mode a warning is given whenever processes have been skipped for this reason.

COLORS

For the resource consumption on system level, *atop* uses colors in text mode to indicate that a critical occupation percentage has been (almost) reached. A critical occupation percentage means that is likely that this load causes a noticeable negative performance influence for applications using this resource. The critical percentage depends on the type of resource: e.g. the performance influence of a disk with a busy percentage of 80% might be more noticeable for applications/users than a CPU with a busy percentage of 90%.
Currently *atop* uses the following default values to calculate a weighted percentage per resource:

Processor
A busy percentage of 90% or higher is considered 'critical' (also in bar graph mode).

Disk
A busy percentage of 90% or higher is considered 'critical'.

Network
A busy percentage of 90% or higher for the load of an interface is considered 'critical'.

Memory

An occupation percentage of 90% is considered 'critical'. Notice that this occupation percentage is the accumulated memory consumption of the kernel (including slab) and all processes. The memory for the page cache ('cache' and 'buff' in the MEM-line) and the reclaimable part of the slab ('slrec') is not implied!
If the number of pages swapped out ('swout' in the PAG-line) is larger than 10 per second, the memory resource is considered 'critical'. A value of at least 1 per second is considered 'almost critical'.
If the committed virtual memory exceeds the limit ('vmcom' and 'vmlim' in the SWP-line), the SWP-line is colored due to overcommitting the system.

Swap

An occupation percentage of 80% is considered 'critical' because swap space might be completely exhausted in the near future. It is not critical from a performance point-of-view.

These default values can be modified in the configuration file (see separate **atoprc** man page).

When a resource exceeds its critical occupation percentage, the concerning values in the screen line are colored red by default.
When a resource exceeds (by default) 80% of its critical percentage (so it is almost critical), the concerning values in the screen line are colored cyan by default. This 'almost critical percentage' (one value for all resources) can be also modified in the configuration file (see separate **atoprc** man page).
The default colors red and cyan can be modified in the configuration file as well (see separate **atoprc** man page).

With the key 'x' (or flag −x), the use of colors can be suppressed in text mode. The use of colors is however mandatory in case of bar graph mode.

NETATOP OR NETATOP-BPF MODULE

Per-process and per-thread network activity can be measured by the *netatop* kernel module or the *netatop-bpf* BPF module that can be separately installed.
When *atop* has been started with the '-K' flag, it verifies if the *netatop* or *netatop-bpf* module is currently active. If so, *atop* obtains the relevant network counters from this module for each interval and shows the number of sent and received packets per process/thread in the generic screen. Besides, detailed counters can be requested by pressing the 'n' key.
When the *netatopd* daemon is running in combination with the *netatop* module, *atop* also reads the network counters of exited processes that are logged by this daemon (comparable with process accounting).

More information about the optional *netatop* kernel module and the *netatopd* daemon can be found in the concerning man-pages and on the website mentioned at the end of this manual page.

GPU STATISTICS GATHERING

GPU statistics can be gathered by *atopgpud* which is a separate data collection daemon process. It gathers cumulative utilization counters of every Nvidia GPU in the system, as well as utilization counters of every process that uses a GPU. When *atop* has been started with the '-k' flag, it reads these GPU utilization counters with every interval when the daemon is active.

The *atopgpud* daemon is written in Python, so a Python interpreter should be installed on the target system. For the gathering of the statistics, the *pynvml* module is used by the daemon. Be sure that this module is installed on the target system before activating the daemon, by running the command *pip* as root user:

pip install nvidia-ml-py

The *atopgpud* daemon is installed by default as part of the **atop** package, but it is *not* automatically enabled. The daemon can be enabled and started now by running the following commands (as root):

systemctl enable atopgpu
systemctl start atopgpu

Find a description about the utilization counters in the section OUTPUT DESCRIPTION.

INTERACTIVE COMMANDS

When running *atop* interactively (no output redirection), keys can be pressed to control the output. In general, lower case keys can be used to show other information for the active processes while certain upper case keys can be used to influence the sort order of the active process/thread list. Some of these keys can also be used to switch from bar graph mode to particular detailed process information in text mode.

g Show generic output (default).

Per process the following fields are shown in case of a window-width of 80 positions: process-id, CPU consumption during the last interval in system and user mode, the virtual and resident memory growth of the process.
The data transfer per process for read/write on disk can only be shown when *atop* runs with root privileges.
When the optional module *netatop* or *netatop-bpf* is loaded, the data transfer for send/receive of network packets is shown for each process.
The last columns contain the state, the occupation percentage for the chosen resource (default: CPU) and the process name.

When more than 80 positions are available, other information is added.

m Show memory related output.

Per process the following fields are shown in case of a window width of 80 positions: process-id, minor and major memory faults, size of virtual shared text, total virtual process size, total resident process size, virtual and resident growth during last interval, memory occupation percentage and process name.

When more than 80 positions are available, other information is added.

For memory consumption, always all processes are shown (also the processes that were not active during the interval).

d Show disk-related output.

When *atop* runs with root privileges, the following fields are shown: process-id, amount of data read from disk, amount of data written to disk, amount of data that was written but has been withdrawn again (WCANCL), disk occupation percentage and process name.

n Show network related output.

Per process the following fields are shown in case of a window width of 80 positions: process-id, thread-id, total bandwidth for received packets, total bandwidth for sent packets, number of received TCP packets with the average size per packet (in bytes), number of sent TCP packets with the average size per packet (in bytes), number of received UDP packets with the average size per packet (in bytes), number of sent UDP packets with the average size per packet (in bytes), the network occupation percentage and process name.
This information can only be shown when the optional module *netatop* or *netatop-bpf* is installed.

When more than 80 positions are available, other information is added.

s Show scheduling characteristics.

Per process the following fields are shown in case of a window width of 80 positions: process-id, number of threads in state 'running' (R), number of threads in state 'interruptible sleeping' (S), number of threads in state 'uninterruptible sleeping' (D), number of threads in state 'idle' (I), scheduling policy (normal timesharing, realtime round-robin, realtime fifo), nice value, priority, realtime priority, current processor, status, exit code, state, the occupation percentage for the chosen resource and the

process name.

When more than 80 positions are available, other information is added.

v Show various process characteristics.

Per process the following fields are shown in case of a window width of 80 positions: process-id, user name and group, start date and time, status (e.g. exit code if the process has finished), state, the occupation percentage for the chosen resource and the process name.

When more than 80 positions are available, other information is added.

c Show the command line of the process.

Per process the following fields are shown: process-id, the occupation percentage for the chosen resource and the command line including arguments.

G Show cgroup v2 information.

Show a hierarchical structure of cgroups and related metrics. Optionally, the processes assigned to each cgroup can be shown. With the keys/flags '2' till '7' the level depth of the cgroups can be chosen ('7' is default). With key/flag '8' the assigned active processes are shown as well, except the kernel processes (usually in the root cgroup). With key/flag '9' all active processes are shown, including the kernel processes.

With key/flag 'a' (toggle) all cgroups and all processes are shown instead of only the active cgroups and processes. A cgroup is considered inactive when no processes are assigned to that cgroup nor to the cgroups underneath.

With key/flag 'C' the output is sorted on CPU consumption, with key/flag 'M' sorted on memory consumption and with key/flag 'D' sorted on disk consumption.

e Show GPU utilization.

Per process at least the following fields are shown: process-id, range of GPU numbers on which the process currently runs, GPU busy percentage on all GPUs, memory busy percentage (i.e. read and write accesses on memory) on all GPUs, memory occupation at the moment of the sample, average memory occupation during the sample, and GPU percentage.

When the *atopgpud* daemon does not run with root privileges, the GPU busy percentage and the memory busy percentage are not available on process level. In that case, the GPU percentage on process level reflects the GPU memory occupation instead of the GPU busy percentage (which is preferred).

o Show the user-defined line of the process.

In the configuration file the keyword *ownprocline* can be specified with the description of a user-defined output-line.
Refer to the man-page of **atoprc** for a detailed description.

y Show the individual threads within a process (toggle).

Single-threaded processes are still shown as one line.
For multi-threaded processes, one line represents the process while additional lines show the activity per individual thread (in a different color). Depending on the option 'a' (all or active toggle), all threads are shown or only the threads that were active during the last interval. Depending on the option 'Y' (sort threads), the threads per process will be sorted on the chosen sort criterium or not.
Whether this key is active or not can be seen in the header line.

Y Sort the threads per process when combined with option 'y' (toggle).

u Show the process activity accumulated per user.

Per user the following fields are shown: number of processes active or terminated during last interval (or in total if combined with command 'a'), accumulated CPU consumption during last interval in system and user mode, the current virtual and resident memory space consumed by active processes (or all processes of the user if combined with command 'a').
When *atop* runs with root privileges, the accumulated read and write throughput on disk is shown.
When the optional module *netatop* or *netatop-bpf* has been installed, the accumulated number of received and sent network packets is shown.
The last columns contain the accumulated occupation percentage for the chosen resource (default: CPU) and the user name.

p Show the process activity accumulated per program (i.e. process name).

Per program the following fields are shown: number of processes active or terminated during last interval (or in total if combined with command 'a'), accumulated CPU consumption during last interval in system and user mode, the current virtual and resident memory space consumed by active processes (or all processes of the user if combined with command 'a').
When *atop* runs with root privileges, the accumulated read and write throughput on disk is shown.
When the optional module *netatop* or *netatop-bpf* has been installed, the accumulated number of received and sent network packets is shown.
The last columns contain the accumulated occupation percentage for the chosen resource (default: CPU) and the program name.

j Show the process activity accumulated per container/pod.

Per container (e.g. Docker/Podman) or pod (e.g. Kubernetes) the following fields are shown: number of processes active or terminated during last interval (or in total if combined with command 'a'), accumulated CPU consumption during last interval in system and user mode, the current virtual and resident memory space consumed by active processes (or all processes of the user if combined with command 'a').
When *atop* runs with root privileges, the accumulated read and write throughput on disk is shown.
When the optional module *netatop* or *netatop-bpf* has been installed, the accumulated number of received and sent network packets is shown.
The last columns contain the accumulated occupation percentage for the chosen resource (default: CPU) and the container/pod name (CID/POD).

C Sort the current list in the order of CPU consumption (default). The one-but-last column changes to 'CPU'.

E Sort the current list in the order of GPU utilization (preferred, but only applicable when the *atopgpud* daemon runs under root privileges) or the order of GPU memory occupation). The one-but-last column changes to 'GPU'.

M Sort the current list in the order of resident memory consumption. The one-but-last column changes to 'MEM'. In case of sorting on memory, the full process list will be shown (not only the active processes).

D Sort the current list in the order of disk accesses issued. The one-but-last column changes to 'DSK'.

N Sort the current list in the order of network bandwidth (received and transmitted). The one-but-last column changes to 'NET'.

A Sort the current list automatically in the order of the most busy system resource during this interval. The one-but-last column shows either 'ACPU', 'AMEM', 'ADSK' or 'ANET' (the preceding 'A' indicates automatic sorting-order). The most busy resource is determined by comparing the weighted busy-percentages of the system resources, as described earlier in the section COLORS.
This option remains valid until another sorting-order is explicitly selected again.

A sorting order for disk is only possible when *atop* runs with root privileges.
A sorting order for network is only possible when the optional module *netatop* or *netatop-bpf* is loaded.

Miscellaneous interactive commands:

? Request for help information (also the key 'h' can be pressed).

V Request for version information (version number and date).

R Gather and calculate the proportional set size of processes (toggle). Gathering of all values that are needed to calculate the PSIZE of a process is a very time-consuming task, so this key should only be active when analyzing the resident memory consumption of processes.

W Get the WCHAN per thread (toggle). Gathering of the WCHAN string per thread is a relatively time-consuming task, so this key should only be made active when analyzing the reason for threads to be in sleep state.

x Suppress colors to highlight critical resources (toggle).
Whether this key is active or not can be seen in the header line.

z The pause key can be used to freeze the current situation in order to investigate the output on the screen. While *atop* is paused, the keys described above can be pressed to show other information about the current list of processes. Whenever the pause key is pressed again, atop will continue with a next sample.
The pause key can be used in text mode and bar graph mode.

i Modify the interval timer (default: 10 seconds). If an interval timer of 0 is entered, the interval timer is switched off. In that case a new sample can only be triggered manually by pressing the key 't'.
The interval can be modified in text mode and bar graph mode.

t Trigger a new sample manually. This key can be pressed if the current sample should be finished before the timer has exceeded, or if no timer is set at all (interval timer defined as 0). In the latter case *atop* can be used as a stopwatch to measure the load being caused by a particular application transaction, without knowing on beforehand how many seconds this transaction will last.
This key can be used in text mode and bar graph mode.

When viewing the contents of a raw file this key can be used to show the next sample from the file. This key can also be used when viewing raw data via a pipe.

T When viewing the contents of a raw file this key can be used to show the previous sample from the file, however not when reading raw data from a pipe.
This key can be used in text mode and bar graph mode.

b When viewing the contents of a raw file, this key can be used to branch to a certain timestamp within the file either forward or backward. When viewing raw data from a pipe only forward branches are possible.
This key can be used in text mode and bar graph mode.

r Reset all counters to zero to see the system and process activity since boot again.
This key can be used in text mode and bar graph mode.

When viewing the contents of a raw file, this key can be used to rewind to the beginning of the file again (except when reading raw data from a pipe).

Z When viewing the contents of a raw file this key can be used to fast-forward to the end of the file, however not when reading raw data from a pipe.
This key can be used in text mode and bar graph mode.

U Specify a regular expression string for specific user names or a numerical UID. From now on, only (active) processes will be shown from a user which matches the regular expression or numerical UID. The system statistics are still system wide. If the Enter-key is pressed without specifying a name, (active) processes of all users will be shown again.

Whether this key is active or not can be seen in the header line.

I Specify a list with one or more PIDs to be selected. From now on, only processes will be shown with a PID which matches one of the given list. The system statistics are still system wide. If the Enter-key is pressed without specifying a PID, all (active) processes will be shown again.
Whether this key is active or not can be seen in the header line.

P Specify a search string for specific process names as a regular expression. From now on, only processes will be shown with a name which matches the regular expression. The system statistics are still system wide. If the Enter-key is pressed without specifying a name, all (active) processes will be shown again.
Whether this key is active or not can be seen in the header line.

/ Specify a specific command line search string as a regular expression. From now on, only processes will be shown with a command line which matches the regular expression. The system statistics are still system wide. If the Enter-key is pressed without specifying a string, all (active) processes will be shown again.
Whether this key is active or not can be seen in the header line.

J Specify a container id (e.g. Docker or Podman) or pod name (e.g. Kubernetes) of maximum 15 characters. In case the name is longer, the last 15 characters are expected. From now on, only processes will be shown that run in that specific container or pod. The system statistics are still system wide. If the Enter-key is pressed without specifying a container id or pod name, all (active) processes will be shown again.
Whether this key is active or not can be seen in the header line.

Q Specify a comma-separated list of process/thread state characters. From now on, only processes/threads will be shown that are in those specific states. Accepted states are: R (running), S (sleeping), D (disk sleep), I (idle), T (stopped), t (tracing stop), X (dead), Z (zombie) and P (parked). The system statistics are still system wide. If the Enter-key is pressed without specifying a state, all (active) processes/threads will be shown again.
Whether this key is active or not can be seen in the header line.

S Specify search strings for specific logical volume names, specific disk names and specific network interface names. All search strings are interpreted as a regular expressions. From now on, only those system resources are shown that match the concerning regular expression. If the Enter-key is pressed without specifying a search string, all (active) system resources of that type will be shown again.
Whether this key is active or not can be seen in the header line.

a The 'all/active' key can be used to toggle between only showing/accumulating the processes that were active during the last interval (default) or showing/accumulating all processes.
Whether this key is active or not can be seen in the header line.

X By default, *atop* shows/accumulates the processes that are alive and the processes that are exited during the last interval. With this key (toggle), showing/accumulating the processes that are exited can be suppressed.
Whether this key is active or not can be seen in the header line.

f Show a fixed (maximum) number of header lines for system resources (toggle). By default only the lines are shown about system resources (CPUs, paging, logical volumes, disks, network interfaces) that really have been active during the last interval. With this key you can force *atop* to show lines of inactive resources as well.
Whether this key is active or not can be seen in the header line.

F Suppress sorting of system resources (toggle). By default system resources (CPUs, logical volumes, disks, network interfaces) are sorted on utilization.
Whether this key is active or not can be seen in the header line.

1 Show relevant counters as an average per second (in the format '..../s') instead of as a total during the interval (toggle).
Whether this key is active or not can be seen in the header line.

l Limit the number of system level lines for the counters per-cpu, the active disks and the network interfaces. By default lines are shown of all CPUs, disks and network interfaces which have been active during the last interval. Limiting these lines can be useful on systems with huge number CPUs, disks or interfaces in order to be able to run *atop* on a screen/window with e.g. only 24 lines.
For all mentioned resources the maximum number of lines can be specified interactively. When using the flag **-l** the maximum number of per-cpu lines is set to 0, the maximum number of disk lines to 5 and the maximum number of interface lines to 3. These values can be modified again in interactive mode.

k Send a signal to an active process (a.k.a. kill a process).

q Quit the program.
This key can be used in text mode and bar graph mode.

PgDn
Show the next page of the process/thread list.
With the arrow-down key the list can be scrolled downwards with single lines.

^F Show the next page of the process/thread list (forward).
With the arrow-down key the list can be scrolled downwards with single lines.

PgUp
Show the previous page of the process/thread list.
With the arrow-up key the list can be scrolled upwards with single lines.

^B Show the previous page of the process/thread list (backward).
With the arrow-up key the list can be scrolled upwards with single lines.

^L Redraw the screen.

RAW DATA STORAGE

In order to store system and process level statistics for long-term analysis (e.g. to check the system load and the active processes running yesterday between 3:00 and 4:00 PM), *atop* can store the system and process level statistics in compressed binary format in a raw file with the flag **-w** followed by the filename. If this file already exists and is recognized as a raw data file, *atop* will append new samples to the file (starting with a sample which reflects the activity since boot). If the file does not exist, it will be created.
All information about system, processes and thread activity is stored in the raw file.
The interval (default: 10 seconds) and number of samples (default: infinite) can be passed as last arguments. Instead of the number of samples, the flag **-S** can be used to indicate that *atop* should finish anyhow before midnight.

A raw file can be read and visualized again with the flag **-r** followed by the filename. If no filename is specified, the file **/var/log/atop/atop_***YYYYMMDD* is opened for input (where *YYYYMMDD* are digits representing the current date). If a filename is specified in the format YYYYMMDD (representing any valid date), the file **/var/log/atop/atop_***YYYYMMDD* is opened. If a filename with the symbolic name **y** is specified, yesterday's daily logfile is opened (this can be repeated so 'yyyy' indicates the logfile of four days ago). If the filename **-** is used, stdin will be read.
The samples from the file can be viewed interactively by using the key 't' to show the next sample, the key 'T' to show the previous sample, the key 'b' to branch to a particular time, the key 'r' to rewind to the begin of the file or the 'Z' key to fast-forward to the end of the file. These keys can be used in text mode as well as in bar graph mode.
When output is redirected to a file or pipe, **atop** prints all samples in plain ASCII. The default line length is 80 characters in that case. With the flag **-L** followed by an alternate line length, more (or less) columns will be shown.
With the flag **-b** (begin time) and/or **-e** (end time) followed by a time argument of the form [YYYYMMDD]hhmm[ss], a certain time period within the raw file can be selected.

Every day at midnight **atop** is restarted by the **atop-rotate.timer** and **atop-rotate.service** unit files, to write compressed binary data to the file **/var/log/atop/atop_***YYYYMMDD* with an interval of 10 minutes by default.

Furthermore all raw files are removed that are older than 28 days (by default).
The mentioned default values can be overruled in the file **/etc/default/atop** that might contain other values for **LOGOPTS** (by default without any flag), **LOGINTERVAL** (in seconds, by default 600), **LOGGENERATIONS** (in days, by default 28), and **LOGPATH** (directory in which logfiles are stored).

Unfortunately, it is not always possible to keep the format of the raw files compatible in newer versions of **atop** especially when many new counters have to be maintained. Therefore, the program **atopconvert** is installed to convert a raw file created by an older version of **atop** to a raw file that can be read by a newer version of **atop** (see the man page of **atopconvert** for more details).

OUTPUT DESCRIPTION

The header line always shows the host name, the end date and end time when the sample has been taken, the current options that are active (only in text mode), and the elapsed time (duration) of the sample. The verb 'PAUSED' will be shown behind the time when **atop** is currently paused with the 'z' key.

The first sample shows the system level activity since boot (the elapsed time in the header shows the time since boot).

In text mode, *atop* first shows the lines related to system level activity for every sample. If a particular system resource has not been used during the interval, the entire line related to this resource is suppressed. So the number of system level lines may vary for each sample.
After that a list is shown of processes which have been active during the last interval. This list is sorted on CPU consumption by default, but this order can be changed by the keys which are previously described.

If values have to be shown by *atop* which do not fit in the column width, another format is used. If e.g. a CPU consumption of 233216 milliseconds should be shown in a column width of 4 positions, it is shown as '233s' (in seconds). For large memory figures, another unit is chosen if the value does not fit (Mb instead of Kb, Gb instead of Mb, Tb instead of Gb, etcetera). For other values, a kind of exponent notation is used (value 123456789 shown in a column of 5 positions gives 123e6).

OUTPUT DESCRIPTION - SYSTEM LEVEL

The system level information in text mode consists of the following output lines:

PRC

Process and thread level totals.
This line contains the total CPU time consumed in system mode ('sys') and in user mode ('user'), the total number of processes present at this moment ('#proc'), the total number of threads present at this moment in state 'running' ('#trun'), 'sleeping interruptible' ('#tslpi'), 'sleeping uninterruptible' ('#tslpu') and 'idle' ('#tidle'), the number of zombie processes ('#zombie'), the number of clone system calls ('clones'), and the number of processes that ended during the interval ('#exit') when process accounting is used. Instead of '#exit' the last column may indicate that process accounting could not be activated ('no procacct').
If the screen-width does not allow all of these counters, only a relevant subset is shown.

CPU

CPU utilization.
At least one line is shown for the total occupation of all CPUs together.
In case of a multi-processor system, an additional line is shown for every individual processor (with 'cpu' in lower case), sorted on activity. Inactive CPUs will not be shown by default. The lines showing the per-cpu occupation contain the CPU number in the field combined with the wait percentage.

Every line contains the percentage of CPU time spent in kernel mode by all active processes ('sys'), the percentage of CPU time consumed in user mode ('user') for all active processes (including processes running with a nice value larger than zero), the percentage of CPU time spent for interrupt handling ('irq') including softirq, the percentage of unused CPU time while no processes were waiting for disk I/O ('idle'), and the percentage of unused CPU time while at least one process was waiting for disk I/O ('wait').

In case of per-cpu occupation, the CPU number and the wait percentage ('w') for that CPU. The number of lines showing the per-cpu occupation can be limited.

For virtual machines, the steal-percentage ('steal') shows the percentage of CPU time stolen by other virtual machines running on the same hardware.
For physical machines hosting one or more virtual machines, the guest percentage ('guest') shows the percentage of CPU time used by the virtual machines. Notice that this percentage overlaps the user percentage!

When PMC performance monitoring counters are supported by the CPU and the kernel (and *atop* runs with root privileges), the number of instructions per CPU cycle ('ipc') is shown. The first sample always shows the value 'initial', because the counters are just activated at the moment that *atop* is started.
When the *CPU busy percentage is high* and the IPC is less than 1.0, it is likely that the CPU is frequently waiting for memory access during instruction execution (larger CPU caches or faster memory might be helpful to improve performance). When the *CPU busy percentage is high* and the IPC is greater than 1.0, it is likely that the CPU is instruction-bound (more/faster cores might be helpful to improve performance).
Furthermore, per CPU the effective number of cycles ('cycl') is shown. This value can reach the current CPU frequency if such CPU is 100% busy. When an idle CPU is halted, the number of effective cycles can be (considerably) lower than the current frequency.
Notice that the *average* instructions per cycle and number of cycles is shown in the CPU line for all CPUs.
Beware that reading the cycle counter in virtual machines (guests) might introduce performance delays. Therefore this metric is by default disabled in virtual machines. However, with the keyword 'perfevents' in the atoprc file this metric can be explicitly set to 'enable' or 'disable' (see separate man-page of atoprc).
See also: http://www.brendangregg.com/blog/2017-05-09/cpu-utilization-is-wrong.html

In case of frequency scaling, all previously mentioned CPU percentages are relative to the used scaling of the CPU during the interval. If a CPU has been active for e.g. 50% in user mode during the interval while the frequency scaling of that CPU was 40%, only 20% of the full capacity of the CPU has been used in user mode.
In case that the kernel module 'cpufreq_stats' is active (after issuing 'modprobe cpufreq_stats'), the *average* frequency ('avgf') and the *average* scaling percentage ('avgscal') is shown. Otherwise the *current* frequency ('curf') and the *current* scaling percentage ('curscal') is shown at the moment that the sample is taken. Notice that *average* values for frequency and scaling are shown in the CPU line for every CPU.

If the screen-width does not allow all of these counters, only a relevant subset is shown.

CPL CPU load information.
This line contains the load average figures reflecting the number of threads that are available to run on a CPU (i.e. part of the runqueue) or that are waiting for disk I/O. These figures are averaged over 1 ('avg1'), 5 ('avg5') and 15 ('avg15') minutes.
Furthermore the number of context switches ('csw'), the number of serviced interrupts ('intr') and the number of available CPUs are shown.

If the screen-width does not allow all of these counters, only a relevant subset is shown.

GPU

GPU utilization (Nvidia).
Read the section GPU STATISTICS GATHERING in this document to find the details about the activation of the *atopgpud* daemon.

In the first column of every line, the bus-id (last nine characters) and the GPU number are shown.

The subsequent columns show the percentage of time that one or more kernels were executing on the GPU ('gpubusy'), the percentage of time that global (device) memory was being read or written ('membusy'), the occupation percentage of memory ('memocc'), the total memory ('total'), the memory being in use at the moment of the sample ('used'), the average memory being in use during the sample time ('usavg'), the number of processes being active on the GPU at the moment of the sample ('#proc'), and the type of GPU.

If the screen-width does not allow all of these counters, only a relevant subset is shown.
The number of lines showing the GPUs can be limited.

MEM

Memory occupation (two lines).
These lines contain the total amount of physical memory ('tot'), the amount of memory which is currently free ('free'), the amount of memory that is available for new workloads without pushing the system into swap ('avail'), the amount of memory in use as page cache including the total resident shared memory ('cache'), the amount of memory within the page cache that has to be flushed to disk ('dirty'), the amount of memory used for filesystem meta data ('buff'), the amount of memory being used for kernel mallocs ('slab'), the amount of slab memory that is reclaimable ('slrec'), the resident size of SYSV shared memory including tmpfs but excluding static huge pages ('shmem'), the resident size of SYSV shared memory including static huge pages ('shrss'), the amount of SYSV shared memory that is currently swapped ('shswp'), the amount of memory that is currently used for page tables ('pgtab'), the number of NUMA nodes in this system ('numnode'), the amount of memory that is currently claimed by vmware's balloon driver ('vmbal'), the amount of memory that is currently claimed by the ARC (cache) of ZFSonlinux ('zfarc'), the amount of memory for anonymous transparent huge pages ('anthp'), the amount of memory that is claimed for huge pages ('hptot'), the amount of huge page memory that is really in use ('hpuse'), the amount of memory that is used for TCP sockets ('tcps'), and the amount of memory that is used for UDP sockets ('udps').

If the screen-width does not allow all of these counters, only a relevant subset is shown.

SWP

Swap occupation and overcommit info.
This line contains the total amount of swap space on disk ('tot'), the amount of free swap space ('free'), the size of the swap cache ('swcac'), the size of compressed storage used for zswap ('zswap'), the real (decompressed) size of the pages stored in zswap ('zstor'), the total size of the memory used for KSM ('ksuse', i.e. shared), and the total size of the memory saved (deduped) by KSM ('kssav', i.e. sharing).
Furthermore the committed virtual memory space ('vmcom') and the maximum limit of the committed space ('vmlim', which is by default swap size plus 50% of memory size) is shown. The committed space is the reserved virtual space for all allocations of private memory space for processes. The kernel only verifies whether the committed space exceeds the limit if strict overcommit handling is configured (vm.overcommit_memory is 2).

LLC

Last-Level Cache of CPU info.
This line contains the total memory bandwidth of LLC ('tot'), the bandwidth of the local NUMA node ('loc'), and the percentage of LLC in use ('LLCXX YY%').

Note that this feature depends on the 'resctrl' pseudo filesystem. Be sure that the kernel is built with the relevant config and take care that the pseudo-filesystem is mounted:

mount -t resctrl resctrl -o mba_MBps /sys/fs/resctrl (on Intel)
mount -t resctrl resctrl -o cdp /sys/fs/resctrl (on AMD)

NUM

Memory utilization per NUMA node (not shown for single NUMA node).
This line shows the total amount of physical memory of this node ('tot'), the amount of free memory

('free'), the amount of memory for cached file data ('file'), modified cached file data ('dirty'), recently used memory ('activ'), less recently used memory ('inact'), memory being used for kernel mallocs ('slab'), the amount of slab memory that is reclaimable ('slrec'), shared memory including tmpfs ('shmem'), total huge pages ('hptot'), used huge pages('hpuse'), and the fragmentation percentage ('frag').

NUC

CPU utilization per NUMA node (not shown for single NUMA node).
This line shows the utilization percentages of all CPUs related to this NUMA node, categorized for system mode ('sys'), user mode ('user'), user mode for niced processes ('niced'), idle mode ('idle'), wait mode ('w' preceded by the node number), irq mode ('irq'), softirq mode ('sirq'), steal mode ('steal'), and guest mode ('guest') overlapping user mode.

PAG Paging frequency.
This line contains the number of scanned pages ('scan') due to the fact that free memory drops below a particular threshold, the number of reclaimed pages('steal') due to the fact that free memory drops below a particular threshold, the number times that the kernel tries to reclaim pages due to an urgent need ('stall'),the number of process stalls to run memory compaction to allocate huge pages ('compact'), the number of NUMA pages migrated ('numamig'), and the total number of memory pages migrated successfully e.g. between NUMA nodes or for compaction ('migrate') are shown.
Also the number of memory pages the system read from block devices ('pgin'), the number of memory pages the system wrote to block devices ('pgout'), the number of memory pages swapped in from zswap ('zswin'), the number of memory pages swapped out to zswap ('zswout'), the number of memory pages the system read from swap space ('swin'), the number of memory pages the system wrote to swap space ('swout'), and the number of out-of-memory kills ('oomkill').

PSI Pressure Stall Information.
This line contains percentages about resource pressure related to CPU, memory and disk I/O. Certain percentages refer to 'some' meaning that some processes/threads were delayed due to resource overload. Other percentages refer to 'full' meaning a loss of overall throughput due to resource overload.
The values 'cpusome', 'memsome', 'memfull', 'iosome' and 'iofull' show the pressure percentage during the entire interval.
The values 'cs' (cpu some), 'ms' (memory some), 'mf' (memory full), 'is' (I/O some) and 'if' (I/O full) each show three percentages separated by slashes: pressure percentage over the last 10, 60 and 300 seconds.
For some Linux distributions PSI support might be disabled by default. In that case you need to pass **psi=1** on the kernel command line during boot.

LVM/MDD/DSK

Logical volume/multiple device/disk utilization.
Per active unit one line is produced, sorted on unit activity. Such line shows the name (e.g. VolGroup00-lvtmp for a logical volume or sda for a hard disk), the percentage of elapsed time during which I/O requests were issued to the device ('busy') (note that for devices serving requests in parallel, such as RAID arrays, SSD and NVMe, this number does not reflect their performance limits), the number of read requests issued ('read'), the number of write requests issued ('write'), the number of discard requests issued ('discrd') if supported by kernel version, the number of KiBytes per read ('KiB/r'), the number of KiBytes per write ('KiB/w'), the number of KiBytes per discard ('KiB/d') if supported by kernel version, the number of MiBytes per second throughput for reads ('MBr/s'), the number of MiBytes per second throughput for writes ('MBw/s'), requests issued to the device driver but not completed ('inflt'), the average queue depth while busy ('avq') and the average number of milliseconds needed by a request ('avio') for seek, latency and data transfer.
If the screen-width does not allow all of these counters, only a relevant subset is shown.

The number of lines showing the units can be limited per class (LVM, MDD or DSK) with the 'l' key or statically (see separate man-page of atoprc). By specifying the value 0 for a particular class, no lines will be shown any more for that class.

NFM

Network Filesystem (NFS) mount at the client side.
For each NFS-mounted filesystem, a line is shown that contains the mounted server directory, the name of the server ('srv'), the total number of bytes physically read from the server ('read') and the total number of bytes physically written to the server ('write'). Data transfer is subdivided in the number of bytes read via normal read() system calls ('nread'), the number of bytes written via normal read() system calls ('nwrit'), the number of bytes read via direct I/O ('dread'), the number of bytes written via direct I/O ('dwrit'), the number of bytes read via memory mapped I/O pages ('mread'), and the number of bytes written via memory mapped I/O pages ('mwrit').

NFC

Network Filesystem (NFS) client side counters.
This line contains the number of RPC calls issues by local processes ('rpc'), the number of read RPC calls ('read') and write RPC calls ('rpwrite') issued to the NFS server, the number of RPC calls being retransmitted ('retxmit') and the number of authorization refreshes ('autref').

NFS Network Filesystem (NFS) server side counters.
This line contains the number of RPC calls received from NFS clients ('rpc'), the number of read RPC calls received ('cread'), the number of write RPC calls received ('cwrit'), the number of Megabytes/second returned to read requests by clients ('MBcr/s'), the number of Megabytes/second passed in write requests by clients ('MBcw/s'), the number of network requests handled via TCP ('nettcp'), the number of network requests handled via UDP ('netudp'), the number of reply cache hits ('rchits'), the number of reply cache misses ('rcmiss') and the number of uncached requests ('rcnoca'). Furthermore some error counters indicating the number of requests with a bad format ('badfmt') or a bad authorization ('badaut'), and a counter indicating the number of bad clients ('badcln').

NET

Network utilization (TCP/IP).
One line is shown for activity of the transport layer (TCP and UDP), one line for the IP layer and one line per active interface.
For the transport layer, counters are shown concerning the number of received TCP segments including those received in error ('tcpi'), the number of transmitted TCP segments excluding those containing only retransmitted octets ('tcpo'), the number of UDP datagrams received ('udpi'), the number of UDP datagrams transmitted ('udpo'), the number of active TCP opens ('tcpao'), the number of passive TCP opens ('tcppo'), the number of TCP output retransmissions ('tcprs'), the number of TCP input errors ('tcpie'), the number of TCP output resets ('tcpor'), the number of UDP no ports ('udpnp'), the number of UDP input errors ('udpie'), and the number of TCP incorrect checksums ('csumie').
If the screen-width does not allow all of these counters, only a relevant subset is shown.
These counters are related to IPv4 and IPv6 combined.

For the IP layer, counters are shown concerning the number of IP datagrams received from interfaces, including those received in error ('ipi'), the number of IP datagrams that local higher-layer protocols offered for transmission ('ipo'), the number of received IP datagrams which were forwarded to other interfaces ('ipfrw'), the number of IP datagrams which were delivered to local higher-layer protocols ('deliv'), the number of received ICMP datagrams ('icmpi'), and the number of transmitted ICMP datagrams ('icmpo').
If the screen-width does not allow all of these counters, only a relevant subset is shown.
These counters are related to IPv4 and IPv6 combined.

For every active network interface one line is shown, sorted on the interface activity. Such line shows the name of the interface and its busy percentage in the first column. The busy percentage for half duplex is determined by comparing the interface speed with the number of bits transmitted and received per second; for full duplex the interface speed is compared with the highest of either the transmitted or the received bits. When the interface speed can not be determined (e.g. for the loopback

interface), '---' is shown instead of the percentage.
Furthermore the number of received packets ('pcki'), the number of transmitted packets ('pcko'), the line speed of the interface ('sp'), the effective amount of bits received per second ('si'), the effective amount of bits transmitted per second ('so'), the number of collisions ('coll'), the number of received multicast packets ('mlti'), the number of errors while receiving a packet ('erri'), the number of errors while transmitting a packet ('erro'), the number of received packets dropped ('drpi'), and the number of transmitted packets dropped ('drpo').
If the screen-width does not allow all of these counters, only a relevant subset is shown.
The number of lines showing the network interfaces can be limited.

IFB Infiniband utilization.
For every active Infiniband port one line is shown, sorted on activity. Such line shows the name of the port and its busy percentage in the first column. The busy percentage is determined by taking the highest of either the transmitted or the received bits during the interval, multiplying that value by the number of lanes and comparing it against the maximum port speed.
Furthermore the number of received packets divided by the number of lanes ('pcki'), the number of transmitted packets divided by the number of lanes ('pcko'), the maximum line speed ('sp'), the effective amount of bits received per second ('si'), the effective amount of bits transmitted per second ('so'), and the number of lanes ('lanes').
If the screen-width does not allow all of these counters, only a relevant subset is shown.
The number of lines showing the Infiniband ports can be limited.

OUTPUT DESCRIPTION - CGROUPS LEVEL (V2)

In the bottom part of the screen a hierarchical list of cgroups can be shown. By default, cgroups without assigned processes in the cgroup itself and the cgroups underneath will be suppressed. By pressing the 'a' key (toggle) all cgroups will be shown.

When depth level '8' or '9' has been selected (by pressing the corresponding key) the assigned processes are shown as well. By default, only the processes that were active during the interval. By pressing the 'a' key (toggle) all processes will be shown.

Per cgroup the following fields may be shown (in alphabetical order), depending on the current width of your window:

CGROUP
Name of the cgroup (version 2), i.e. the directory name in the cgroup hierarchy. This root cgroup '/' corresponds to the cgroup root directory, which is usually '/sys/fs/cgroup'.

CPUBUSY
The consumed CPU percentage by this cgroup and all cgroups underneath (as a percentage of one CPU). Value -1 means undefined.

CPUMAX
The 'cpu.max' value of this cgroup (version 2), calculated as percentage of one CPU. Value -1 means maximum, while value -2 means undefined.

CPUPS The CPU pressure percentage ('some') in this cgroup and cgroups underneath.
For some Linux distributions PSI support might be disabled by default. In that case you need to pass **psi=1** on the kernel command line during boot.

CPUWGT
The 'cpu.weight' value of this cgroup (version 2). Value -2 means undefined.

DISKIO The amount of data transferred to/from physical disks in this cgroup and the cgroups underneath. Value -1 means undefined.

DSKPS The disk pressure percentage ('full') in this cgroup and cgroups underneath.
For some Linux distributions PSI support might be disabled by default. In that case you need to pass **psi=1** on the kernel command line during boot.

IOWGT The 'io.weight' value of this cgroup (version 2). Value -2 means undefined.

MEMMAX
The 'memory.max' value of this cgroup (version 2). Value -1 means maximum, while value -2 means undefined.

MEMORY
The current memory occupation of this cgroup and all cgroups underneath. Value -1 means undefined.

MEMPS The memory pressure percentage ('full') in this cgroup and cgroups underneath.
For some Linux distributions PSI support might be disabled by default. In that case you need to pass **psi=1** on the kernel command line during boot.

NPROCS
The number of processes assigned to this cgroup.

PBELOW
The number of processes assigned to the cgroups underneath this cgroup.

SWPMAX
The 'memory.swap.max' value of this cgroup (version 2). Value -1 means maximum, while value -2 means undefined.

OUTPUT DESCRIPTION - PROCESS LEVEL

In the bottom part of the screen, a list of processes can be shown from which the resource utilization has changed during the last interval. These processes might have used CPU time or might have issued disk or network requests. However a process is also shown if part of it has been paged out due to lack of memory (while the process itself was in sleep state).

Per process the following fields may be shown (in alphabetical order), depending on the current output mode as described in the section INTERACTIVE COMMANDS and depending on the current width of your window:

AVGRSZ The average size of one read-action on disk.

AVGWSZ
The average size of one write-action on disk.

BANDWI
Total bandwidth for received TCP and UDP packets consumed by this process (bits-per-second). This value can be compared with the value 'si' on interface level (used bandwidth per interface).
This information will only be shown when the optional module *netatop* or *netatop-bpf* is loaded.

BANDWO
Total bandwidth for sent TCP and UDP packets consumed by this process (bits-per-second). This value can be compared with the value 'so' on interface level (used bandwidth per interface).
This information will only be shown when the optional module *netatop* or *netatop-bpf* is loaded.

BDELAY
Aggregated block I/O delay, i.e. time waiting for disk I/O.

CID/POD
Container id (e.g. Docker or Podman) or pod name (e.g. Kubernetes) referring to the container/pod in which the process/thread is running. When a pod name is longer than 15 characters, only the last 15 characters are shown.

If a process has been started and finished during the last interval, a '?' is shown because the container id or pod name is not part of the standard process accounting record.

This column will only be shown when *atop* runs with superuser privileges and when at least one containerized process is detected.

CMD The name of the process. This name can be surrounded by "less/greater than" signs ('<name>') which means that the process has finished during the last interval. A single accounting record is written for the entire process on termination of the last thread in the process. When the main thread exits, the process name is changed to the thread name.
Behind the abbreviation 'CMD' in the header line, the current page number and the total number of pages of the process/thread list are shown.

COMMAND-LINE
The full command line of the process (including arguments). If the length of the command line exceeds the length of the screen line, the arrow keys -> and <- can be used for horizontal scroll.

The '-z <regex>' command line option can be used to prepend matching environment variables to the displayed command line. POSIX Extended Regular Expression syntax are used (see regex(3)). When a matching environment variable is too long (exceeding the buffer that should contain the command line), it will be truncated. Usually only the environment variables will be shown of the processes that are running under your own user ID, unless you are a privileged user.
WARNING: Don't use this flag when *writing* to a (publicly readable) raw file! You might expose sensitive environment variables to the readers of that raw file.

Behind the verb 'COMMAND-LINE' in the header line, the current page number and the total number of pages of the process/thread list are shown.

CPU The occupation percentage of this process related to the available capacity for this resource on system level.

CPUNR The identification of the CPU the (main) thread is running on or has recently been running on.

CTID Container ID (OpenVZ). If a process has been started and finished during the last interval, a '?' is shown because the container ID is not part of the standard process accounting record.

DSK The occupation percentage of this process related to the total load that is produced by all processes (i.e. total disk accesses by all processes during the last interval).
This information is shown when per process "storage accounting" is active in the kernel.

EGID Effective group-id under which this process executes. When the name is longer than 8 or when the *-I* flag is used, the number is shown instead of the name.

ENDATE
Date that the process has been finished. If the process is still running, this field shows 'active'.

ENTIME
Time that the process has been finished. If the process is still running, this field shows 'active'.

ENVID Virtual environment identified (OpenVZ only).

EUID Effective user-id under which this process executes. When the name is longer than 8 or when the *-I* flag is used, the number is shown instead of the name.

EXC The exit code of a terminated process (second position of column 'ST' is E) or the fatal signal number (second position of column 'ST' is S or C).

FSGID Filesystem group-id under which this process executes. When the name is longer than 8 or when the *-I* flag is used, the number is shown instead of the name.

FSUID Filesystem user-id under which this process executes. When the name is longer than 8 or when the *-I* flag is used, the number is shown instead of the name.

GPU When the *atopgpud* daemon does not run with root privileges, the GPU percentage reflects the GPU memory occupation percentage (memory of all GPUs is 100%).

When the *atopgpud* daemon runs with root privileges, the GPU percentage reflects the GPU busy percentage.

GPUBUSY
Busy percentage on all GPUs (one GPU is 100%).
When the *atopgpud* daemon does not run with root privileges, this value is not available.

GPUNUMS
Comma-separated list of GPUs used by the process during the interval. When the comma-separated list exceeds the width of the column, a hexadecimal value is shown.

LOCKSZ
The virtual amount of memory being locked (i.e. non-swappable) by this process (or user).

MAJFLT
The number of page faults issued by this process that have been solved by creating/loading the requested memory page.

MEM The occupation percentage of this process related to the available capacity for this resource on system level.

MEMAVG
Average memory occupation during the interval on all used GPUs.

MEMBUSY
Busy percentage of memory on all GPUs (one GPU is 100%), i.e. the time needed for read and write accesses on memory.
When the *atopgpud* daemon does not run with root privileges, this value is not available.

MEMNOW
Memory occupation at the moment of the sample on all used GPUs.

MINFLT The number of page faults issued by this process that have been solved by reclaiming the requested memory page from the free list of pages.

NET The occupation percentage of this process related to the total load that is produced by all processes (i.e. consumed network bandwidth of all processes during the last interval).
This information will only be shown when the optional module *netatop* or *netatop-bpf* is loaded.

NICE The more or less static priority that can be given to a process on a scale from -20 (high priority) to +19 (low priority).

NIVCSW
Number of times the process/thread was context-switched involuntarily, in case that the time slice expired.

NPROCS
The number of active and terminated processes accumulated for this user or program.

NVCSW Number of times that the process/thread was context-switched voluntarily in case of a blocking system call, e.g. to wait for an I/O operation to complete.

PID Process-id. If a process has been started and finished during the last interval, a '?' is shown because the process-id is not part of the standard process accounting record.

POLI The policies 'norm' (normal, which is SCHED_OTHER), 'btch' (batch) and 'idle' refer to time-sharing processes. The policies 'fifo' (SCHED_FIFO) and 'rr' (round robin, which is SCHED_RR) refer to realtime processes.

PPID Parent process-id. If a process has been started and finished during the last interval, value 0 is shown because the parent process-id is not part of the standard process accounting record.

PRI The process' priority ranges from 0 (highest priority) to 139 (lowest priority). Priority 0 to 99 are used for realtime processes (fixed priority independent of their behavior) and priority 100 to

139 for timesharing processes (variable priority depending on their recent CPU consumption and the nice value).

PSIZE The proportional memory size of this process (or user).
Every process shares resident memory with other processes. E.g. when a particular program is started several times, the code pages (text) are only loaded once in memory and shared by all incarnations. Also the code of shared libraries is shared by all processes using that shared library, as well as shared memory and memory-mapped files. For the PSIZE calculation of a process, the resident memory of a process that is shared with other processes is divided by the number of sharers. This means, that every process is accounted for a proportional part of that memory. Accumulating the PSIZE values of all processes in the system gives a reliable impression of the total resident memory consumed by all processes.
Since gathering of all values that are needed to calculate the PSIZE is a very time-consuming task, the 'R' key (or '-R' flag) should be active. Gathering these values also requires superuser privileges (otherwise '?K' is shown in the output).
If a process has finished during the last interval, no value is shown since the proportional memory size is not part of the standard process accounting record.

RDDSK The read data transfer issued physically on disk (so reading from the disk cache is not accounted for).
Unfortunately, the kernel aggregates the data transfer of a process to the data transfer of its parent process when terminating, so you might see transfers for (parent) processes like cron, bash or init, that are not really issued by them.

RDELAY
Runqueue delay, i.e. time spent waiting on a runqueue.

RGID The real group-id under which the process executes. When the name is longer than 8 or when the *-I* flag is used, the number is shown instead of the name.

RGROW The amount of resident memory that the process has grown during the last interval. A resident growth can be caused by touching memory pages which were not physically created/loaded before (load-on-demand). Note that a resident growth can also be negative e.g. when part of the process is paged out due to lack of memory or when the process frees dynamically allocated memory. For a process which started during the last interval, the resident growth reflects the total resident size of the process at that moment.
If a process has finished during the last interval, no value is shown since resident memory occupation is not part of the standard process accounting record.

RNET The number of TCP- and UDP packets received by this process. This information will only be shown when the optional module *netatop* or *netatop-bpf* is installed.
If a process has finished during the last interval, no value is shown since network counters are not part of the standard process accounting record.

RSIZE The total resident memory usage consumed by this process (or user). Notice that the RSIZE of a process includes all resident memory used by that process, even if certain memory parts are shared with other processes (see also the explanation of PSIZE).
If a process has finished during the last interval, no value is shown since resident memory occupation is not part of the standard process accounting record.

RTPR Realtime priority according the POSIX standard. Value can be 0 for a timesharing process (policy 'norm', 'btch' or 'idle') or ranges from 1 (lowest) till 99 (highest) for a realtime process (policy 'rr' or 'fifo').

RUID The real user-id under which the process executes. When the name is longer than 8 or when the *-I* flag is used, the number is shown instead of the name.

S The current state of the (main) thread: 'R' for running (currently processing or in the runqueue), 'S' for sleeping interruptible (wait for an event to occur), 'D' for sleeping non-interruptible, 'Z' for zombie (waiting to be synchronized with its parent process), 'T' for stopped (suspended or

traced), 'W' for swapping, and 'E' (exit) for processes which have finished during the last interval.

SGID The saved group-id of the process. When the name is longer than 8 or when the *-I* flag is used, the number is shown instead of the name.

SNET The number of TCP and UDP packets transmitted by this process. This information will only be shown when the optional module *netatop* or *netatop-bpf* is installed.

ST The status of a process.
The first position indicates if the process has been started during the last interval (the value *N* means 'new process').

The second position indicates if the process has been finished during the last interval.
The value *E* means 'exit' on the process' own initiative; the exit code is displayed in the column 'EXC'.
The value *S* means that the process has been terminated unvoluntarily by a signal; the signal number is displayed in the in the column 'EXC'.
The value *C* means that the process has been terminated unvoluntarily by a signal, producing a core dump in its current directory; the signal number is displayed in the column 'EXC'.

STDATE The start date of the process.

STTIME The start time of the process.

SUID The saved user-id of the process. When the name is longer than 8 or when the *-I* flag is used, the number is shown instead of the name.

SWAPSZ The swap space consumed by this process (or user).

SYSCPU CPU time consumption of this process in system mode (kernel mode), usually due to system call handling.

TCPRASZ
The average size of a received TCP buffer in bytes. This information will only be shown when the optional module *netatop* or *netatop-bpf* is installed.

TCPRCV
The number of TCP packets received for this process. This information will only be shown when the optional module *netatop* or *netatop-bpf* is installed.

TCPSASZ
The average size of a transmitted TCP buffer in bytes. This information will only be shown when the optional module *netatop* or *netatop-bpf* is installed.

TCPSND The number of TCP packets transmitted for this process. This information will only be shown when the optional module *netatop* or *netatop-bpf* is installed.

THR Total number of threads within this process. All related threads are contained in a thread group, represented by *atop* as one line or as a separate line when the 'y' key (or -y flag) is active.

On Linux 2.4 systems it is hardly possible to determine which threads (i.e. processes) are related to the same thread group. Every thread is represented by *atop* as a separate line.

TID Thread-id. All threads within a process run with the same PID but with a different TID. This value is shown for individual threads in multi-threaded processes (when using the key 'y').

TIDLE Number of threads within this process that are in the state 'idle' (I), i.e. uninterruptible sleeping threads that do not count for the load average.

TRUN Number of threads within this process that are in the state 'running' (R).

TSLPI Number of threads within this process that are in the state 'interruptible sleeping' (S).

TSLPU Number of threads within this process that are in the state 'uninterruptible sleeping' (D).

UDPRASZ
The average size of a received UDP packet in bytes. This information will only be shown when the optional module *netatop* or *netatop-bpf* is installed.

UDPRCV
The number of UDP packets received by this process. This information will only be shown when the optional module *netatop* or *netatop-bpf* is installed.

UDPSASZ
The average size of a transmitted UDP packets in bytes. This information will only be shown when the optional module *netatop* or *netatop-bpf* is installed.

UDPSND
The number of UDP packets transmitted by this process. This information will only be shown when the optional module *netatop* or *netatop-bpf* is installed.

USRCPU
CPU time consumption of this process in user mode, due to processing the own program text.

VDATA The virtual memory size of the private data used by this process (including heap and shared library data).

VGROW The amount of virtual memory that the process has grown during the last interval. A virtual growth can be caused by e.g. issuing a malloc() or attaching a shared memory segment. Note that a virtual growth can also be negative by e.g. issuing a free() or detaching a shared memory segment. For a process which started during the last interval, the virtual growth reflects the total virtual size of the process at that moment.
If a process has finished during the last interval, no value is shown since virtual memory occupation is not part of the standard process accounting record.

VPID Virtual process-id (within an OpenVZ container). If a process has been started and finished during the last interval, a '?' is shown because the virtual process-id is not part of the standard process accounting record.

VSIZE The total virtual memory usage consumed by this process (or user).
If a process has finished during the last interval, no value is shown since virtual memory occupation is not part of the standard process accounting record.

VSLIBS The virtual memory size of the (shared) text of all shared libraries used by this process.

VSTACK
The virtual memory size of the (private) stack used by this process

VSTEXT
The virtual memory size of the (shared) text of the executable program.

WCHAN Wait channel of thread in sleep state, i.e. the name of the kernel function in which the thread has been put asleep.
Since determining the name string of the kernel function is a relatively time-consuming task, the 'W' key (or '-W' flag) should be active.

WRDSK The write data transfer issued physically on disk (so writing to the disk cache is not accounted for). This counter is maintained for the application process that writes its data to the cache (assuming that this data is physically transferred to disk later on). Notice that disk I/O needed for swapping is not taken into account.
Unfortunately, the kernel aggregates the data transfer of a process to the data transfer of its parent process when terminating, so you might see transfers for (parent) processes like cron, bash or init, that are not really issued by them.

WCANCL

The write data transfer previously accounted for this process or another process that has been cancelled. Suppose that a process writes new data to a file and that data is removed again before the cache buffers have been flushed to disk. Then the original process shows the written data as WRDSK, while the process that removes/truncates the file shows the unflushed removed data as WCANCL.

PARSABLE OUTPUT

With the flag **-P** followed by a list of one or more labels (comma-separated), parsable output is produced for each sample. The labels that can be specified for system-level statistics correspond to the labels (first verb of each line) that can be found in the interactive output: "CPU", "cpu", "CPL", "GPU", "MEM", "SWP", "PAG", "PSI", "LVM", "MDD", "DSK", "NFM", "NFC", "NFS", "NET", "IFB", "LLC", "NUM" and "NUC".

For cgroup-level statistics the label "CGR" is available.

For process-level statistics special labels are available: "PRG" (general), "PRC" (CPU), "PRE" (GPU), "PRM" (memory), "PRD" (disk, but requires special privileges) and "PRN" (network, only if the optional module *netatop* or *netatop-bpf* is installed).
With the label "ALL", all system and process level statistics are shown.

The command and command line in the parsable output might contain spaces and are therefore by default surrounded by parenthesis. However, since a space is often used as separator between the fields by parsing tools, with the additional flag **-Z** it is possible to exchange the spaces in the command (line) by underscores and omit the parenthesis.

For every interval all requested lines are shown whereafter **atop** shows a line just containing the label "SEP" as a separator before the lines for the next sample are generated.
When a sample contains the values since boot, **atop** shows a line just containing the label "RESET" before the lines for this sample are generated.

The first part of each output-line consists of the following six fields: **label** (the name of the label), **host** (the name of this machine), **epoch** (the time of this interval as number of seconds since 1-1-1970), **date** (date of this interval in format YYYY/MM/DD), **time** (time of this interval in format HH:MM:SS), and **interval** (number of seconds elapsed for this interval).

The subsequent fields of each output-line depend on the label:

CPU Subsequent fields: total number of clock-ticks per second for this machine, number of processors, consumption for all CPUs in system mode (clock-ticks), consumption for all CPUs in user mode (clock-ticks), consumption for all CPUs in user mode for niced processes (clock-ticks), consumption for all CPUs in idle mode (clock-ticks), consumption for all CPUs in wait mode (clock-ticks), consumption for all CPUs in irq mode (clock-ticks), consumption for all CPUs in softirq mode (clock-ticks), consumption for all CPUs in steal mode (clock-ticks), consumption for all CPUs in guest mode (clock-ticks) overlapping user mode, frequency of all CPUs, frequency percentage of all CPUs, instructions executed by all CPUs and cycles for all CPUs.

cpu Subsequent fields: total number of clock-ticks per second for this machine, processor-number, consumption for this CPU in system mode (clock-ticks), consumption for this CPU in user mode (clock-ticks), consumption for this CPU in user mode for niced processes (clock-ticks), consumption for this CPU in idle mode (clock-ticks), consumption for this CPU in wait mode (clock-ticks), consumption for this CPU in irq mode (clock-ticks), consumption for this CPU in softirq mode (clock-ticks), consumption for this CPU in steal mode (clock-ticks), consumption for this CPU in guest mode (clock-ticks) overlapping user mode, frequency of this CPU, frequency percentage of this CPU, instructions executed by this CPU and cycles for this CPU.

CPL Subsequent fields: number of processors, load average for last minute, load average for last five minutes, load average for last fifteen minutes, number of context-switches, and number of device interrupts.

GPU Subsequent fields: GPU number, bus-id string, type of GPU string, GPU busy percentage during last second (-1 if not available), memory busy percentage during last second (-1 if not available), total memory size (KiB), used memory (KiB) at this moment, number of samples taken during interval, cumulative GPU busy percentage during the interval (to be divided by the number of samples for the average busy percentage, -1 if not available), cumulative memory busy percentage during the interval (to be divided by the number of samples for the average busy percentage, -1 if not available), and cumulative memory occupation during the interval (to be divided by the number of samples for the average occupation).

MEM Subsequent fields: page size for this machine (in bytes), size of physical memory (pages), size of free memory (pages), size of page cache (pages), size of buffer cache (pages), size of slab (pages), dirty pages in cache (pages), reclaimable part of slab (pages), total size of vmware's balloon pages (pages), total size of shared memory (pages), size of resident shared memory (pages), size of swapped shared memory (pages), smaller huge page size (in bytes), total size of smaller huge pages (huge pages), size of free smaller huge pages (huge pages), size of ARC (cache) of ZFSonlinux (pages), size of sharing pages for KSM (pages), size of shared pages for KSM (pages), size of memory used for TCP sockets (pages), size of memory used for UDP sockets (pages), size of pagetables (pages), larger huge page size (in bytes), total size of larger huge pages (huge pages), size of free larger huge pages (huge pages), size of available memory (pages) for new workloads without swapping, and size of anonymous transparent huge pages ('normal' pages).

SWP Subsequent fields: page size for this machine (in bytes), size of swap (pages), size of free swap (pages), size of swap cache (pages), size of committed space (pages), limit for committed space (pages), size of the swap cache (pages), the real (decompressed) size of the pages stored in zswap (pages), and the size of compressed storage used for zswap (pages).

LLC Subsequent fields: LLC id, percentage of LLC in use, total memory bandwidth of this LLC (in bytes), and memory bandwidth on local NUMA node of this LLC (in bytes).

PAG Subsequent fields: page size for this machine (in bytes), number of page scans, number of allocstalls, 0 (future use), number of swapins, number of swapouts, number of oomkills (-1 when counter not present), number of process stalls to run memory compaction, number of pages successfully migrated in total, number of NUMA pages migrated, number of pages read from block devices, number of pages written to block devices, number of swapins from zswap, and number of swapouts to zswap.

PSI Subsequent fields: PSI statistics present on this system (n or y), CPU some avg10, CPU some avg60, CPU some avg300, CPU some accumulated microseconds during interval, memory some avg10, memory some avg60, memory some avg300, memory some accumulated microseconds during interval, memory full avg10, memory full avg60, memory full avg300, memory full accumulated microseconds during interval, I/O some avg10, I/O some avg60, I/O some avg300, I/O some accumulated microseconds during interval, I/O full avg10, I/O full avg60, I/O full avg300, and I/O full accumulated microseconds during interval.

LVM/MDD/DSK

For every logical volume/multiple device/hard disk one line is shown.
Subsequent fields: name, number of milliseconds spent for I/O, number of reads issued, number of sectors transferred for reads, number of writes issued, number of sectors transferred for write, number of discards issued (-1 if not supported), number of sectors transferred for discards, number of requests currently in flight (not yet completed), and the average queue depth while the disk was busy.

NFM Subsequent fields: mounted NFS filesystem, total number of bytes read, total number of bytes written, number of bytes read by normal system calls, number of bytes written by normal system calls, number of bytes read by direct I/O, number of bytes written by direct I/O, number of pages read by memory-mapped I/O, and number of pages written by memory-mapped I/O.

NFC Subsequent fields: number of transmitted RPCs, number of transmitted read RPCs, number of transmitted write RPCs, number of RPC retransmissions, and number of authorization refreshes.

NFS Subsequent fields: number of handled RPCs, number of received read RPCs, number of received write RPCs, number of bytes read by clients, number of bytes written by clients, number of RPCs with bad format, number of RPCs with bad authorization, number of RPCs from bad client, total number of handled network requests, number of handled network requests via TCP, number of handled network requests via UDP, number of handled TCP connections, number of hits on reply cache, number of misses on reply cache, and number of uncached requests.

NET First, one line is produced for the upper layers of the TCP/IP stack.
Subsequent fields: the verb "upper", number of packets received by TCP, number of packets transmitted by TCP, number of packets received by UDP, number of packets transmitted by UDP, number of packets received by IP, number of packets transmitted by IP, number of packets delivered to higher layers by IP, number of packets forwarded by IP, number of input errors (UDP), number of noport errors (UDP), number of active opens (TCP), number of passive opens (TCP), number of passive opens (TCP), number of established connections at this moment (TCP), number of retransmitted segments (TCP), number of input errors (TCP), number of output resets (TCP), and number of checksum errors on received packets (TCP).

Next, one line is shown for every interface.
Subsequent fields: name of the interface, number of packets received by the interface, number of bytes received by the interface, number of packets transmitted by the interface, number of bytes transmitted by the interface, interface speed, duplex mode (0=half, 1=full), receive errors, transmit errors, receive drops, transmit drops, receive fifo, transmit fifo, receive compressed, transmit compressed, receive framing errors, receive multicast, transmit collisions, and transmit carrier.

IFB Subsequent fields: name of the InfiniBand interface, port number, number of lanes, maximum rate (Mbps), number of bytes received, number of bytes transmitted, number of packets received, and number of packets transmitted.

NUM Subsequent fields: NUMA node number, page size for this machine (in bytes), the fragmentation percentage of this node, size of physical memory (pages), size of free memory (pages), recently (active) used memory (pages), less recently (inactive) used memory (pages), size of cached file data (pages), dirty pages in cache (pages), slab memory being used for kernel mallocs (pages), slab memory that is reclaimable (pages), shared memory including tmpfs (pages), total huge pages (huge pages), and free huge pages (huge pages).

NUC Subsequent fields: NUMA node number, number of processors for this node, consumption for node CPUs in system mode (clock-ticks), consumption for node CPUs in user mode (clock-ticks), consumption for node CPUs in user mode for niced processes (clock-ticks), consumption for node CPUs in idle mode (clock-ticks), consumption for node CPUs in wait mode (clock-ticks), consumption for node CPUs in irq mode (clock-ticks), consumption for node CPUs in softirq mode (clock-ticks), consumption for node CPUs in steal mode (clock-ticks), and consumption for node CPUs in guest mode (clock-ticks) overlapping user mode.

CGR For every cgroup (level) one or two lines are shown. The first line shows the utilization and configuration values of the cgroup. The optional second line shows the PIDs of the processes assigned to this cgroup.

Subsequent fields of first line: character 'C', full path name of cgroup, number of processes assigned to this group, number of processes assigned to the cgroups underneath, CPU usage in user mode (microseconds), CPU usage in system mode (microseconds), CPU weight, CPU max, current memory usage (pages), anonymous memory usage (pages), file memory usage (pages), kernel memory usage (pages), shared memory usage (pages), memory max (pages), swap max (pages), number of bytes read from disk, number of bytes written to disk, number of read requests from disk, number of write requests to disk, the disk weight, cpu some pressure

(microseconds), cpu total pressure (microseconds), memory some pressure (microseconds), memory total pressure (microseconds), disk some pressure (microseconds), and disk total pressure (microseconds).

Subsequent fields of second line (only when processes assigned to this cgroup): character 'P', full path name of cgroup, and list of PIDs separated by spaces.

PRG For every process one line is shown.
Subsequent fields: PID (unique ID of task), name (between parenthesis or underscores for spaces), state, real uid, real gid, TGID (group number of related tasks/threads), total number of threads, exit code (in case of fatal signal: signal number + 256), start time (epoch), full command line (between parenthesis or underscores for spaces), PPID, number of threads in state 'running' (R), number of threads in state 'interruptible sleeping' (S), number of threads in state 'uninterruptible sleeping' (D), effective uid, effective gid, saved uid, saved gid, filesystem uid, filesystem gid, elapsed time of terminated process (hertz), is_process (y/n), OpenVZ virtual pid (VPID), OpenVZ container id (CTID), container/pod name (CID/POD), indication if the task is newly started during this interval ('N'), cgroup v2 path name (between parenthesis or underscores for spaces), end time (epoch or 0 if still active), and number of threads in state 'idle' (I).

PRC For every process one line is shown.
Subsequent fields: PID, name (between parenthesis or underscores for spaces), state, total number of clock-ticks per second for this machine, CPU-consumption in user mode (clockticks), CPU-consumption in system mode (clockticks), nice value, priority, realtime priority, scheduling policy, current CPU (-1 for exited process), sleep average, TGID (group number of related tasks/threads), is_process (y/n), runqueue delay in nanoseconds for this thread or for all threads (in case of process), wait channel of this thread (between parenthesis or underscores for spaces), block I/O delay (clockticks), cgroup v2 'cpu.max' calculated as percentage (-3 means no cgroup v2 support, -2 means undefined and -1 means maximum), cgroup v2 most restrictive 'cpu.max' in upper directories calculated as percentage (-3 means no cgroup v2 support, -2 means undefined and -1 means maximum), number of voluntary context switches, and number of involuntary context switches.

PRE For every process one line is shown.
Subsequent fields: PID, name (between parenthesis or underscores for spaces), process state, GPU state (A for active, E for exited, N for no GPU user), number of GPUs used by this process, bitlist reflecting used GPUs, GPU busy percentage during interval, memory busy percentage during interval, memory occupation (KiB) at this moment cumulative memory occupation (KiB) during interval, and number of samples taken during interval.

PRM For every process one line is shown.
Subsequent fields: PID, name (between parenthesis or underscores for spaces), state, page size for this machine (in bytes), virtual memory size (KiB), resident memory size (KiB), shared text memory size (KiB), virtual memory growth (KiB), resident memory growth (KiB), number of minor page faults, number of major page faults, virtual library exec size (KiB), virtual data size (KiB), virtual stack size (KiB), swap space used (KiB), TGID (group number of related tasks/threads), is_process (y/n), proportional set size (KiB) if in 'R' option is specified, virtually locked memory space (KiB), cgroup v2 'memory.max' in KiB (-3 means no cgroup v2 support, -2 means undefined and -1 means maximum), cgroup v2 most restrictive 'memory.max' in upper directories in KiB (-3 means no cgroup v2 support, -2 means undefined and -1 means maximum), cgroup v2 'memory.swap.max' in KiB (-3 means no cgroup v2 support, -2 means undefined and -1 means maximum), and cgroup v2 most restrictive 'memory.swap.max' in upper directories in KiB (-3 means no cgroup v2 support, -2 means undefined and -1 means maximum).

PRD For every process one line is shown.
Subsequent fields: PID, name (between parenthesis or underscores for spaces), state, obsoleted kernel patch installed ('n'), standard io statistics used ('y' or 'n'), number of reads on disk, cumulative number of sectors read, number of writes on disk, cumulative number of sectors

written, cancelled number of written sectors, TGID (group number of related tasks/threads), obsoleted value ('n'), and is_process (y/n).

PRN For every process one line is shown.
Subsequent fields: PID, name (between parenthesis or underscores for spaces), state, kernel module *netatop* or *netatop-bpf* installed ('y' or 'n'), number of TCP-packets transmitted, cumulative size of TCP-packets transmitted, number of TCP-packets received, cumulative size of TCP-packets received, number of UDP-packets transmitted, cumulative size of UDP-packets transmitted, number of UDP-packets received, cumulative size of UDP-packets transmitted, number of raw packets transmitted (obsolete, always 0), number of raw packets received (obsolete, always 0), TGID (group number of related tasks/threads) and is_process (y/n).
If the kernel module is not active, the network I/O counters per process are not relevant.

JSON OUTPUT

With the flag **-J** followed by a list of one or more labels (comma-separated), JSON output is produced for each sample. The syntax and name of JSON labels are the same as for the parsable output.

SIGNALS

By sending the SIGUSR1 signal to *atop* a new sample will be forced, even if the current timer interval has not exceeded yet. The behavior is similar to pressing the 't' key in an interactive session.

By sending the SIGUSR2 signal to *atop* a final sample will be forced after which *atop* will terminate.

EXAMPLES

Monitor the current system load in text mode with an interval of (default) 10 seconds:

atop

Monitor the current system load as bar graphs with an interval of 5 seconds:

atop -B 5

Monitor the current system load in text mode with an interval of 4 seconds with the possibility to browse back and forth while measuring continues (twin mode):

atop -t 4

Live monitoring of the current system load while writing the measurement to a raw log in parallel (to be viewed later on):

atop -g -w /tmp/sessionx

View the cgroups with the related processes (except kernel processes) with the possibility to review previous samples during the live measurement:

atop -G8 -t

Store information about the system and process activity in binary compressed form to a file with an interval of ten minutes during an hour:

atop -w /tmp/atop.raw 600 6

View the contents of this file interactively:

atop -r /tmp/atop.raw

View the processor and disk utilization of this file in parsable format:

atop -PCPU,DSK -r /tmp/atop.raw

View the contents of today's standard logfile interactively:

atop -r

View the contents of the standard logfile of the day before yesterday interactively:

atop -r yy

View the contents of the standard logfile of 2024, November 5 from 02:00 PM onwards interactively:

atop -r 20241105 -b 1400

Read the contents of today's logfile and show the CPU utilization on system level as parseable output and JSON output:

atop -r -PCPU -JCPU

Concatenate all raw log files of November 2024 and generate parsable output about the CPU utilization:

atopcat /var/log/atop/atop_202403?? | atop -r - -PCPU

Monitor the system load and write it to a file (in plain ASCII) with an interval of one minute during half an hour with active processes sorted on memory consumption:

atop -M 60 30 > /log/atop.mem

FILES

/var/run/pacct_shadow.d/
Directory containing the process accounting shadow files that are used by *atop* when the *atopacctd* daemon is active.

/var/cache/atop.d/atop.acct
File in which the kernel writes the accounting records when *atop* itself has activated the process accounting mechanism.

/etc/atoprc
Configuration file containing system-wide default values. For further information about the default values, refer to the **atoprc** man page).

~/.atoprc
Configuration file containing personal default values. For further information about the default values, refer to the **atoprc** man page).

/etc/default/atop
Configuration file to overrule the settings of *atop* that runs in the background to create the daily logfile. This file is created when *atop* is installed. The default settings are:

```
LOGOPTS=""
LOGINTERVAL=600
LOGGENERATIONS=28
```

/var/log/atop/atop_*YYYYMMDD*
Raw file, where *YYYYMMDD* are digits representing the current date. This name is used by **atop** running in the background as default name for the output file, and by **atop** as default name for the input file when using the **-r** flag.
All binary system and process level data in this file has been stored in compressed format.

/var/run/netatop.log
File that contains the netpertask structs containing the network counters of exited processes. These structs are written by the *netatopd* daemon (which is related to the *netatop* module) and read by *atop* after reading the standard process accounting records.

SEE ALSO

atopsar(1), atopconvert(1), atopcat(1), atophide(1), atoprc(5), atopacctd(8), netatop(4), netatopd(8), atopgpud(8), logrotate(8)
https://www.atoptool.nl

AUTHOR

Gerlof Langeveld (gerlof.langeveld@atoptool.nl)
JC van Winkel