sysstat Performance statistics

START HERE

`iostat -xz 2`
Disk load, busy devices only

`sar -n DEV 2 5`
Network throughput

`iostat -c 2`
CPU split incl. iowait

`sar -q 2 5`
Load and run queue

`sar -u 2 5`
CPU, 5 samples

`pidstat -u 2`
CPU per process

`sar -r 2 5`
Memory use

`pidstat -d 2`
Disk I/O per process

Kernel counters → `sampled by` → **sadc** collector → `writes` → **/var/log/sa/saDD** daily file → `read by` → **sar -f** report

CONCEPTS

Give every command an interval (seconds) and a count; without them you only get an average since boot.

The first iostat report covers the time since boot; read the later ones, or add -y to drop it.

sar history only exists if the sadc collector runs (cron job or timer from the sysstat package).

High %util means saturation only on one-request-at-a-time disks; SSDs and RAID can stay busy yet keep up.

DISKS (IOSTAT)

`iostat -x 2`
Extended disk stats every 2 s
First report is the average since boot; add -y to skip it.

`iostat -xz 2`
Only devices with activity

`iostat -c 2`
CPU report only

`iostat -xy 1 5`
Five samples, no since-boot report

`iostat -xm 2`
Rates in MB per second

`iostat -d -h`
Human-readable device report

`iostat -p <device> 2`
Device with its partitions

`iostat -N -x 2`
Show LVM / device-mapper names

`iostat -t -x 5`
Timestamp every report

`iostat -o JSON -x 1 1`
One report as JSON

SYSTEM (SAR LIVE)

`sar -u 2 5`
CPU utilisation, 5 samples
Add -P ALL for every core separately.

SYSTEM (SAR LIVE) (cont.)

`sar -r 2 5`
Memory utilisation

`sar -n DEV 2 5`
Network throughput per interface

`sar -d -p 2 5`
Per-disk activity, readable names

`sar -q 2 5`
Load average and run queue

`sar -P ALL 2 5`
Each CPU core

`sar -S 2 5`
Swap usage

`sar -n EDEV 2 5`
Interface errors and drops

`sar -b 2 5`
Overall I/O transfer rates

`sar -B 2 5`
Paging and page faults

`sar -W 2 5`
Swap in and out per second

`sar -w 2 5`
Context switches and task creation

`sar -n TCP,ETCP 2 5`
TCP connections and errors

`sar -q PSI 2 5`
Pressure stall (CPU, I/O, memory)
Needs a recent sysstat and a kernel with PSI.

`sar -F MOUNT 2 5`
Filesystem usage by mount point
Only if the collector gathers filesystem data.

HISTORY (SAR -F)

`sar -u`
Today's CPU history
Reads the daily file; needs the collector running.

`sar -r -f /var/log/sa/sa<DD>`
Memory on day DD of the month

HISTORY (SAR -F) (cont.)

`sar -u -f /var/log/sa/sa<DD> -s 09:00 -e 11:00`
CPU in a time window

`sar -q -1`
Load history from yesterday

`sar -u -i 600`
One record per 10 minutes

`sar -o <file> 5 12 >/dev/null`
Record samples to your own file

`sar -u -r -f <file>`
Several reports from one file

`sar -u -f <file> -C`
Show sadc comments in the file

PROCESSES (PIDSTAT)

`pidstat -u 2 5`
CPU per process

`pidstat -r 2 5`
Memory and page faults per process

`pidstat -d 2 5`
Disk I/O per process

`pidstat -p <pid> 1`
Watch one process

`pidstat -t -p <pid> 1`
Per-thread numbers

`pidstat -w 2 5`
Context switches per process

`pidstat -l 2 5`
Full command line

`pidstat -C <name> 2`
Only commands matching name

`pidstat -U 2`
User names instead of UIDs

`pidstat -u -I 2`
CPU divided by core count

`pidstat -h -u -r -d 1`
CPU, memory, I/O on one line

READING THE NUMBERS

iostat -x columns

Column	Meaning
`%util`	time with I/O in flight
`await`	avg ms per request, incl. queue
`r_await / w_await`	same, reads / writes
`aqu-sz`	average queue length
`rareq-sz / wareq-sz`	avg request size, KiB
`%iowait`	CPU idle, I/O pending
`%steal`	CPU lost to hypervisor

pidstat counters

Column	Meaning
`cswch/s`	voluntary switches: blocked, waiting
`nvcswch/s`	involuntary: preempted by others
`majflt/s`	faults that read from disk
`minflt/s`	faults served from memory
`kB_ccwr/s`	writes cancelled before disk

Rising await together with a growing aqu-sz points at a saturated device; high %iowait with an idle CPU says the same.

RECIPES

Is the disk the bottleneck?

```
iostat -xz 2
pidstat -d 2   # who is doing the I/O
```

What happened at 3 am?

```
sar -u -f /var/log/sa/sa<DD> -s 02:30 -e 03:30
sar -q -f /var/log/sa/sa<DD> -s 02:30 -e 03:30
```

Capture, then analyse later

```
sar -o /tmp/cap.sa 5 60 >/dev/null
sar -u -r -f /tmp/cap.sa
```

Find the CPU and memory hog

```
pidstat -u -r -l 2 5
```

HELP

`man sar`
Manual page with all report fields

`sar -V`
Show the version