ps List processes

START HERE

`ps aux`
Every process, user format

`ps -ef`
Every process, full format

`ps -p <pid> -o pid,etime,cmd`
One process, chosen columns

`ps -u <user>`
A user's processes

`ps -C <name>`
Match command name

`ps -ejH`
Process tree

`ps aux --sort=-%mem | head`
Top memory users

`ps -eLf`
Include threads

Select -e -u -p -C --ppid → which rows → **Format** -f u -o <cols> → which columns → **Sort** --sort=-%cpu → order → **Filter** | grep, | head

CONCEPTS

ps takes UNIX options (-ef, with a dash) and BSD options (aux, no dash). They differ in meaning: ps -aux is not ps aux.

ps is a snapshot. For a live view use top; to act on a result, feed the PID to kill.

SELECT PROCESSES

`ps`
Your processes on this terminal

`ps -e / ps -A`
All processes

`ps -u <user>`
By effective user

`ps -p <pid>[,<pid>]`
By PID

`ps -C <name>`
By exact command name

`ps --ppid <pid>`
Children of a process

`ps -t <tty>`
By terminal

`ps -U <user>`
By real user

`ps -G <group>`
By real group

`ps -s <sid>`
By session ID

`ps -N -u <user>`
Negate the selection
Same as --deselect: everything except that user.

`ps -q <pid>`
Quick lookup, no scan
Reads only the listed PIDs and keeps your order; -p sorts by PID.

`ps r`
Only running processes

`ps x`
Include processes without a tty

OUTPUT FORMATS

`ps aux`
BSD user-oriented, all

`ps -ef`
UNIX full format, all

`ps -f`
Full format (UID, PPID, args)

`ps -l`
Long format (nice, state, size)

`ps -F`
Extra full (adds RSS, CPU id)

`ps -j`
Job format (PGID, SID)

`ps v`
Virtual memory format

`ps -w / ps ww`
Do not truncate long lines
Without it, output is cut to the terminal width when writing to a tty.

`ps --no-headers`
Drop the header line

`ps --headers`
Repeat header every page

`ps -n / ps n`
Numeric UID and wchan

`ps e`
Append the environment

`ps -M`
Add SELinux context

CUSTOM COLUMNS AND SORTING

`ps -eo pid,ppid,user,%cpu,%mem,cmd`
Choose your own columns

`ps -o pid=,cmd= -p <pid>`
Column without header

`ps -o pid=PROCESS -p <pid>`
Rename a column header

`ps -O <cols>`
Default columns plus extras

`ps L`
List all output column names

`ps --sort=-%cpu`
Highest CPU first

`ps --sort=-rss`
Largest resident memory first

CUSTOM COLUMNS AND SORTING (cont.)

`ps --sort=<key>,-<key>`
Multi-key sort
+ or nothing is ascending, - is descending.

`ps -eo pid,lstart,etime,cmd`
Start time and uptime

`ps -eo pid,etimes,cmd`
Elapsed time in seconds

`ps -eo pid,stat,wchan:20,cmd`
State and kernel wait point

`ps -eo pid,ni,pri,cls,cmd`
Nice and scheduling class

`ps -eo pid,comm:20,args`
Set a column width

TREES AND THREADS

`ps -ejH`
Hierarchy, indented

`ps axjf`
ASCII-art forest

`ps -ef --forest`
Forest with full format

`ps -eLf`
Threads with LWP and NLWP

`ps -T -p <pid>`
Threads of one process

`ps -eo pid,nlwp,cmd --sort=-nlwp`
Most threads first

`ps -m -p <pid>`
Threads listed after process

`ps -ww --ppid <pid> -f`
Direct children, full args

READING THE OUTPUT

STAT first letter

Code	Meaning
R	Running or runnable
S	Sleeping, interruptible
D	Uninterruptible I/O wait
T	Stopped by job control signal
t	Stopped by a debugger
Z	Zombie, awaiting parent
I	Idle kernel thread

READING THE OUTPUT (cont.)

STAT extra flags

Flag	Meaning
s	Session leader
+	In foreground group
l	Multi-threaded
<	High priority
N	Low priority (nice)
L	Pages locked in memory

Common columns

Column	Meaning
%CPU	CPU time / elapsed time
VSZ	Virtual size in KiB
RSS	Resident memory in KiB
TIME	Total CPU time used
ELAPSED	Time since start
NI / PRI	Nice value / priority

%CPU is an average over the whole life of the process, not current load. Use top for what is busy right now.

Names in [brackets] are kernel threads. Many D or Z states for long usually point to a hung disk or a parent that never reaps.

RECIPES

Find a process without matching grep

```
ps -ef | grep '[n]ginx'
# or: ps -C nginx -o pid,args
# or: pgrep -af nginx
```

Top 5 memory hogs

```
ps -eo pid,user,%mem,rss,comm
--sort=-rss | head -n 6
```

Find zombies and their parents

```
ps -eo pid,ppid,stat,cmd | awk
'$3 ~ /^Z/'
# kill or fix the parent
(PPID), not the zombie
```

How long has it been running?

```
ps -o pid,lstart,etime,cmd -p
<pid>
# then: kill -TERM <pid>
```