

Chapter 12 Process Monitoring - Notes

12.2 Introduction

Keeping track running (and sleeping) processes -> essential system administration task. **ps** program -> main tool for doing so in UNIX-based operating systems for decades.

HOWever, because utility has long + complicated history of being used differently in more than one operating system variety, has large assortment of options that can be applied w/ often confusing combinations. Another trust tool provided by **top**, which interactively monitors the system's state.

12.3 Learning Objectives:

- Use **ps** to view characteristics and statistics associated w/ processes.
- Identify different **ps** output fields and customize the **ps** output.
- Use **pstree** to get a visual description of the process ancestry and multi-threaded applications.
- Use **top** to view system loads interactively.

12.4 Monitoring Tools

In this section, will concentrate on process monitoring. Linux administrators make use of many utilities for process monitoring, eg. **ps**, **pstree**, **top**. All have long histories in UNIX-like operating systems.

Review of some of the main tools for process monitoring:

Process and Load Monitoring Utilities

Utility	Purpose	Package
top	Process activity, dynamically updated	procps
uptime	How long system is running and average load	procps
ps	Detailed information about processes	procps
pstree	Tree of processes and their connections	psmisc (or pstree)
mpstat	Multiple processor usage	sysstat
iostat	CPU utilization and I/O statistics	sysstat
sar	Display and collect information about system activity	sysstat
numstat	Information about NUMA (Non-Uniform Memory Architecture)	numactl
strace	Information about all system calls a process makes	strace

12.5 Viewing Process States with ps

ps -> workhorse for displaying characteristics and statistics associated with processes, all of which garnered from `/proc` directory associated with process.

Command utility existed in all UNIX-like operating system variant. Diversity reflected in complicated potpourri of options that Linux version of **ps** accepts. Falls into three categories:

1. UNIX options, which must be preceded by `-`, and what may be grouped.
2. BSD options, which must *not* be preceded by `-`, and which may be grouped.
3. GNU long options, each of which must be preceded by `--`.

Having all these possible options can make life rather confusing. Most system administrators tend to use one or two standard combinations for daily use.

12.6 BSD Option Format for ps

Can see typical usage with BSD option format below, where `aux` option shows all processes. Commands surrounded by square brackets (as in `[ksoftirqd/0]`) -> threads that exist totally within kernel. If there is one for each CPU, command followed by `integer` specifying CPU it is running on.

```
student@FC-25:~  
File Edit View Search Terminal Help  
/home/student@student@FC-25 ~]$ ps aux  
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND  
root         1  0.6  0.1 146580  7632 ?        Ss   10:38   0:01 /usr/lib/systemd/systemd --switch  
h  
root         2  0.0  0.0      0     0 ?        S    10:38   0:00 [kthreadd]  
root         3  0.0  0.0      0     0 ?        S    10:38   0:00 [kworker/0:0]  
root         4  0.0  0.0      0     0 ?        S<   10:38   0:00 [kworker/0:0H]  
root         5  0.0  0.0      0     0 ?        S    10:38   0:00 [kworker/u256:0]  
root         6  0.0  0.0      0     0 ?        S    10:38   0:00 [ksoftirqd/0]  
.....  
student    1429 13.2  6.3 1994576 257244 tty2     Sl+  10:38   0:19 /usr/bin/gnome-shell  
student    1445  0.4  1.0 236360  41700 tty2     Sl+  10:38   0:00 /usr/bin/Xwayland :0 -rootless -  
n  
student    1455  0.0  0.1 344740   5780 ?        Ssl  10:38   0:00 /usr/libexec/at-spi-bus-launcher  
....  
student    1872  0.0  0.1 123292   4728 pts/0    Ss   10:38   0:00 bash  
student    2013  0.0  0.0 125020   2324 pts/0    S+   10:40   0:00 script /tmp/outfile  
student    2015  0.3  0.1 123300   4596 pts/1    Ss   10:40   0:00 bash -i  
student    2047  0.0  0.0 150020   3524 pts/1    R+   10:40   0:00 ps aux  
/home/student@student@FC-25 ~]$  
[student@FC-25 ~]$
```

12.7 ps Output Fields

Most fields in preceding example self-explanatory. Of others:

- **vsz** : process' virtual memory size in KB
- **rss** : resident set size; non-swapped physical memory task is using in KB.
- **STAT** : describes state of process. in example, only see **S** for sleeping, or **R** for running. Additional character in state (where it exists):
 - **<** : high priority (not *nice*)
 - **N** : low priority (*nice*)
 - **L** : having pages locked in memory
 - **s** : session leader
 - **l** : multi-threaded
 - **+** : being in foreground process group

Adding **f** option will show how processes connect by ancestry:

```
$ ps auxf
```

```
student@FC-25:/tmp
File Edit View Search Terminal Help
[student@FC-25 ~]$ ps auxf
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         2  0.0  0.0      0     0 ?        Ss   10:38   0:00 [kthreadd]
....
student   1866  0.6  1.0 742160 41704 ?        Ssl  10:38   0:03 \_ /usr/libexec/gnome-terminal-server
student   1872  0.0  0.1 123544   512 pts/0    Ss   10:38   0:00 | \_ bash
student   2260  0.0  0.0 125020   232 pts/0    S+   10:48   0:00 | \_ script /tmp/outfile
student   2262  0.5  0.1 123300   460 pts/1    Ss   10:48   0:00 | \_ bash -i
student   2294  0.0  0.0 150332   388 pts/1    R+   10:48   0:00 | \_ ps auxf
/home/student[student@FC-25 ~]$
[student@FC-25 tmp]$
```

12.8 UNIX Option Format for ps

Can see typical usage with UNIX option format below. Note: now showing **Parent Process ID** (`PPID`) and the niceness (`NI`). May observe that many processes show `PPID=2`, in this screenshot (taken from RHEL 7, using systemd) an internal kernel process `kthreadd` (designed to adopt children when parent process dies). In older kernels/systems, would see `PPID=1` for `sbin/init`, but really same thing going on.

```
student@FC-25:/tmp
File Edit View Search Terminal Help
[student@FC-25 ~]$ ps -elf
F S UID      PID  PPID  C PRI  NI ADDR SZ WCHAN  STIME TTY      TIME CMD
4 S root        1      0  0  80   0 - 53029 -    10:38 ?      00:00:01 /usr/lib/systemd/systemd --s
1 S root        2      0  0  80   0 -    0 -    10:38 ?      00:00:00 [kthreadd]
1 S root        4      2  0  60 -20 -    0 -    10:38 ?      00:00:00 [kworker/0:0H]
1 S root        6      2  0  80   0 -    0 -    10:38 ?      00:00:00 [ksoftirqd/0]
....
0 S student   1429   1357  6  80   0 - 500475 poll_s 10:38 tty2    00:00:59 /usr/bin/gnome-shell
0 S student   1445   1429  0  80   0 - 59085 ep_pol 10:38 tty2    00:00:02 /usr/bin/Xwayland :0 -rootle
0 S student   1455   1333  0  80   0 - 86185 poll_s 10:38 ?      00:00:00 /usr/libexec/at-spi-bus-laun
0 S student   1460   1455  0  80   0 - 12061 ep_pol 10:38 ?      00:00:00 /bin/dbus-daemon --config-fi
....
0 R student   2477   1872  0  80   0 - 31255 -    10:53 pts/0    00:00:00 script /tmp/outfile
0 S student   2479   2477  0  80   0 - 30825 wait  10:53 pts/1    00:00:00 bash -i
0 R student   2511   2479  0  80   0 - 37505 -    10:53 pts/1    00:00:00 ps -elf
...
[student@FC-25 ~]$
[student@FC-25 tmp]$
```

Some common selection options in UNIX format:

- `-A` or `-e` : Select all processes
- `N` : Negate selection (means do the opposite)
- `c` : Select by command name
- `g` : Select by real group ID (also supports names)
- `u` : Select by real user ID (also supports names)

12.9 Customizing the ps Output

If you use `-o` option, followed by comma-separated list of field identifiers, can print out customized list of `ps` fields:

- `pid` : Process ID number
- `uid` : User ID number
- `cmd` : Command with all arguments
- `cputime` : Cumulative CPU time
- `pmem` : Ratio of process's resident set size to physical memory on machine, expressed as percentage

Can see example below. Can consult **ps man** page for many other output options.

```
File Edit View Search Terminal Help
c7:/tmp>ps -o pid,uid,cputime,pmem,command
  PID   UID   TIME %MEM COMMAND
 2900   1000 00:00:00  0.0  bash
29145   1000 00:00:00  0.0  ps -o pid,uid,cputime,pmem,command
c7:/tmp>
```

12.10 Using pstree

pstree gives visual description of process ancestry and multi-threaded applications:

```
$ pstree -aAp 2408

bash,2408
|-emacs,24998 pmonitor.tex
|  |-{emacs},25002
|  '---{emacs},25002
|-evince,18036 LFS201-SLIDES.pdf
|  |-{evince},18040
|  |-{evince},18046
|  '---{evince},18047
```

Consult **man** page for **pstree** for explanation of many options. In above, have chosen just to show information for **pid-2408**.

Note: one of its child processes (**evince**, **pid=18036**) has three children of its own. Another way to see that:

```
$ ls -l /proc/18036/task

total 0
dr-xr-xr-x 5 coop coop 0 Sep 11 07:15 18036
dr-xr-xr-x 5 coop coop 0 Sep 11 07:15 18040
dr-xr-xr-x 5 coop coop 0 Sep 11 07:15 18046
dr-xr-xr-x 5 coop coop 0 Sep 11 07:15 18047
```

12.11 Viewing System Loads with top

When we want to know what system spending time on, first tool often used is **top**. Below shows what can be seen when using **top** without arguments.

By default, **top** refreshes every 3.0 seconds.

File Edit View Search Terminal Help											
top - 14:42:39 up 7:21, 6 users, load average: 1.90, 1.01, 0.57											
Tasks: 367 total, 8 running, 359 sleeping, 0 stopped, 0 zombie											
%Cpu(s): 74.7 us, 8.3 sy, 0.0 ni, 16.7 id, 0.3 wa, 0.0 hi, 0.0 si, 0.0 st											
KiB Mem : 16282768 total, 2446720 free, 2802652 used, 11033396 buff/cache											
KiB Swap: 8290300 total, 8290300 free, 0 used. 11506844 avail Mem											
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
27331	root	20	0	218716	95564	14812	R	51.2	0.6	0:01.54	cc1
2420	coop	20	0	2683668	326032	86584	S	23.9	2.0	12:06.35	gnome-shell
27477	root	20	0	176692	53736	14828	R	15.9	0.3	0:00.48	cc1
27507	root	20	0	166280	43504	14184	R	9.0	0.3	0:00.27	cc1
3395	coop	20	0	1389980	203716	70452	S	8.3	1.3	10:23.40	skypeforlinux
27513	root	20	0	167916	40384	10476	R	6.3	0.2	0:00.19	cc1
1420	root	20	0	538388	78476	62092	S	6.0	0.5	7:11.91	Xorg
26681	coop	20	0	644272	55812	50028	S	5.6	0.3	0:00.36	gnome-screensho
3367	coop	20	0	515344	65148	49820	S	3.0	0.4	1:46.33	skypeforlinux
2739	coop	20	0	664892	69636	50836	S	2.7	0.4	0:27.92	gnome-terminal-
3342	coop	20	0	1456344	114452	75568	S	2.3	0.7	2:49.98	skypeforlinux
27524	root	20	0	148956	21072	10444	R	2.0	0.1	0:00.06	cc1
27243	coop	20	0	2331612	285412	98476	S	1.3	1.8	1:23.69	thunderbird
2322	coop	20	0	28716	5004	2444	S	0.7	0.0	0:09.32	dbus-daemon
3831	coop	20	0	1015984	110092	59836	S	0.7	0.7	0:23.01	chrome
8	root	20	0	0	0	0	S	0.3	0.0	0:16.62	rcu preempt
11	root	20	0	0	0	0	S	0.3	0.0	0:02.36	rcuop/0

12.12 top Options

top: ancient utility and has tons of options, as well as interactive commands triggered when certain keys pressed. Eg. if hit **1**, each CPU shown separately, if hit **i** only active processes shown. Can see what doing both gives you below.

File Edit View Search Terminal Help											
top - 14:45:38 up 7:24, 6 users, load average: 2.01, 1.45, 0.84											
Tasks: 371 total, 9 running, 362 sleeping, 0 stopped, 0 zombie											
%Cpu0 : 61.1 us, 10.4 sy, 0.0 ni, 27.9 id, 0.7 wa, 0.0 hi, 0.0 si, 0.0 st											
%Cpu1 : 81.3 us, 10.0 sy, 0.0 ni, 8.7 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st											
%Cpu2 : 68.0 us, 11.8 sy, 0.0 ni, 20.2 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st											
%Cpu3 : 68.7 us, 12.0 sy, 0.0 ni, 19.0 id, 0.3 wa, 0.0 hi, 0.0 si, 0.0 st											
%Cpu4 : 80.7 us, 9.6 sy, 0.0 ni, 9.6 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st											
%Cpu5 : 68.6 us, 9.4 sy, 0.0 ni, 22.1 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st											
%Cpu6 : 74.7 us, 9.7 sy, 0.0 ni, 15.0 id, 0.7 wa, 0.0 hi, 0.0 si, 0.0 st											
%Cpu7 : 70.5 us, 9.7 sy, 0.0 ni, 19.8 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st											
KiB Mem : 16282768 total, 2234832 free, 2737992 used, 11309944 buff/cache											
KiB Swap: 8290300 total, 8290300 free, 0 used. 11374212 avail Mem											
PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
2420	coop	20	0	2765448	328120	86772	R	20.9	2.0	12:23.28	gnome-shell
5044	root	20	0	178232	51964	12884	R	13.6	0.3	0:00.41	cc1
3395	coop	20	0	1390760	204324	70220	S	7.6	1.3	10:34.47	skypeforlinux
4665	coop	20	0	644268	55736	49952	S	6.3	0.3	0:00.19	gnome-screensho
1420	root	20	0	538364	78576	62192	R	5.3	0.5	7:17.21	Xorg
3367	coop	20	0	515344	65148	49820	S	3.0	0.4	1:50.76	skypeforlinux
3342	coop	20	0	1505008	114524	75464	S	2.7	0.7	2:53.70	skypeforlinux
2739	coop	20	0	664892	69636	50836	S	2.3	0.4	0:29.37	gnome-terminal-
5147	root	20	0	152208	24368	10496	R	2.0	0.1	0:00.06	cc1
5148	root	20	0	154508	27072	10388	R	2.0	0.2	0:00.06	cc1

Have lot of control over how processes sorted, which fields displayed. Many others besides defaults. Eg. hitting **h** or **?** gives brief list of interactive commands, **q** quits.

Can **kill** task by hitting **k**, or **renice** (change its priority) with **r**.

man top will give extensive documentation on configuration possibilities, options, interactive possibilities.

Note: there are popular alternatives to standard **top** program. Some have more visual interfaces and/or additional information, such as **htop**, **ntop**, **atop**. Most Linux distributions have graphical system monitor (eg. **gnome-system-monitor** or

ksysguard), which has **top**-like display window that can be shown.

##

[Back to top](#)

[Previous Chapter](#) - [Table of Contents](#) - [Next Chapter](#)