How to find out which CPU core a process is running on

Author : Dan Nanni

Categories : System, Kernel

Tagged as : cpu, top, htoptaskset

Question: I have a Linux process running on my multi-core processor system. How can I find out which CPU core the process is running on?

When you run performance-critical HPC applications or network-heavy workload on multi-core NUMA processors, CPU/memory affinity is one important factor to consider to maximize their performance. Scheduling closely related processes on the same NUMA node can reduce slow remote memory access. On processors like Intel's Sandy Bridge processor which has an integrated PCIe controller, you want to schedule network I/O workload on the same NUMA node as the NIC card to exploit PCI-to-CPU affinity.

As part of performance tuning or troubleshooting, you may want to know on which CPU core (or NUMA node) a particular process is currently scheduled.

Here are several ways to **find out which CPU core is a given Linux process or a thread is scheduled on**.

Method One

If a process is explicitly pinned to a particular CPU core using commands like `taskset`, you can find out the pinned CPU using the following `taskset` command:

```
$ taskset -c -p
```

For example, if the process you are interested in has PID 5357:

```
$ taskset -c -p 5357
```

```
pid 5357's current affinity list: 5
```

The output says the process is pinned to CPU core 5.

However, if you haven't explicitly pinned the process to any CPU core, you will get something like the following as the affinity list.

```
pid 5357's current affinity list: 0-11
```

The output indicates that the process can potentially be scheduled on *any* CPU core from 0 to 11. So in this case, `taskset` is not useful in identifying which CPU core the process is currently assigned to, and you should use other methods as described below.

Method Two

The `ps` command can tell you the CPU ID each process/thread is currently assigned to (under "PSR" column).

```
$ ps -o pid,psr,comm -p
```

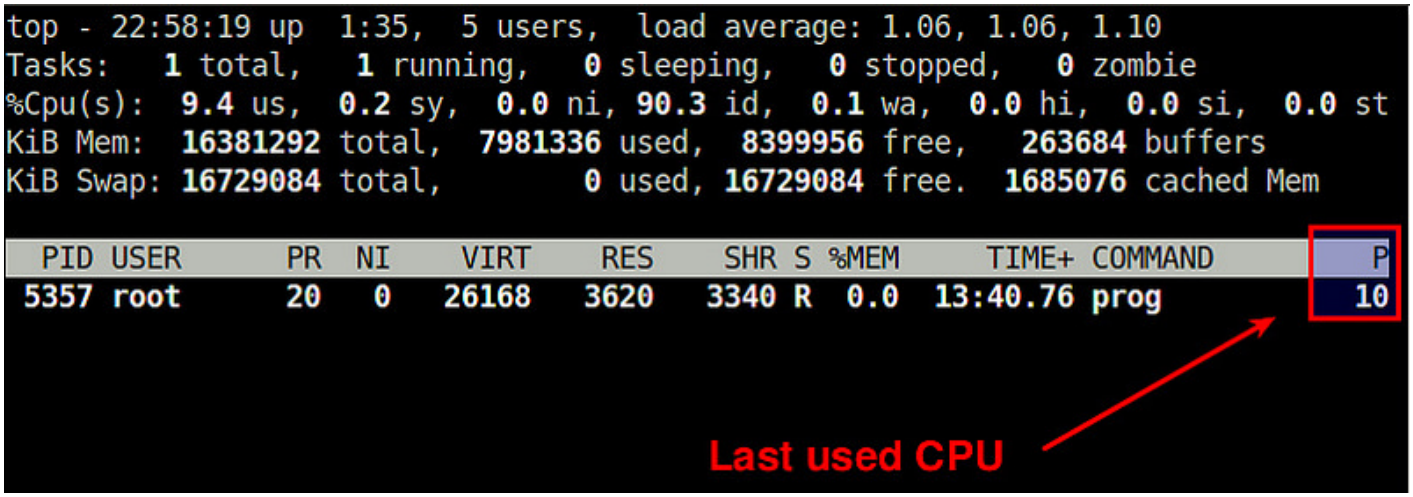
```
  PID PSR COMMAND   5357  10 prog
```

The output says the process with PID 5357 (named "prog") is currently running on CPU core 10. If the process is not pinned, the PSR column can keep changing over time depending on where the kernel scheduler assigns the process.

Method Three

The `top` command can also show the CPU assigned to a given process. First, launch `top` command with "p" option. Then press 'f' key, and add "Last used CPU" column to the display. The currently used CPU core will appear under "P" (or "PSR") column.

```
$ top -p 5357
```



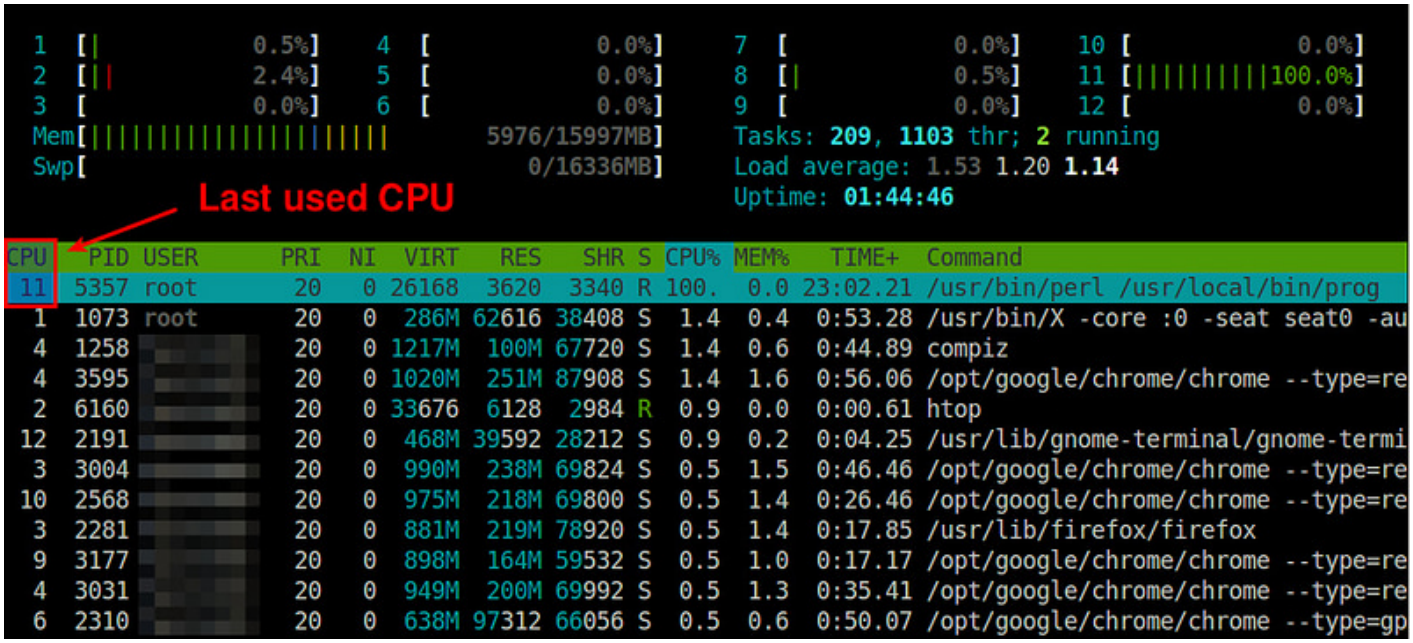
Compared to `ps` command, the advantage of using `top` command is that you can continuously monitor how the assigned CPU changes over time.

Method Four

Yet another method to check the currently used CPU of a process/thread is to use `htop` command.

Launch `htop` from the command line. Press key, go to "Columns", and add PROCESSOR under "Available Columns".

The currently used CPU ID of each process will appear under "CPU" column.



Note that all previous commands `taskset`, `ps` and `top` assign CPU core IDs 0, 1, 2, ..., N-1. However, `htop` assigns CPU core IDs starting from 1 (upto N).