CS2211b

Software Tools and Systems Programming

Week 4a

Processes and Jobs

Announcements

Week 3 Q&A on OWL

Lab 3 on OWL

Assignment #1 Due Tomorrow

MC244 Virtual Machine Image Available

Quiz Review

POSIX is a _____.

Quiz Review

A relative path to file2 in dir1 in your home dir is _____ if your current working directory is ~/dir2.

Quiz Review

_____ gives *only* read and execute permissions to others.

Week 3 Practice Questions

9.3 How do you select from a file (i) lines 5 to 10, (ii) second-to-last line?

Week 3 Practice Questions

9.10 How do you display a listing of all directories in the PATH list?

Week 3 Practice Questions

10.3 Explain the significance of the * in this command:

```
grep 'botswana.*birds' *.htm*.
```

Week 3 Practice Questions

10.18 How do you look for *bill christie* in a file, without knowing whether *bill* exists as *william* or *bill* and whether *christie* also exists as *christy*?

Week 3 Practice Questions

10.23 Frame a regular expression to locate lines longer than 100 and smaller than 150 characters using grep.

Processes & Jobs

Processes

- Each instance of a running program is considered to be a process in UNIX.
- These programs can either be started by the user (e.g. by issuing commands via the shell) or the system.
- Processes are managed by the kernel through a scheduling service called scheduler that allocates the system's resources (memory, CPU time, etc) to each process.

Processes

States/status

Processes have one status (out of three) at a time:

Running

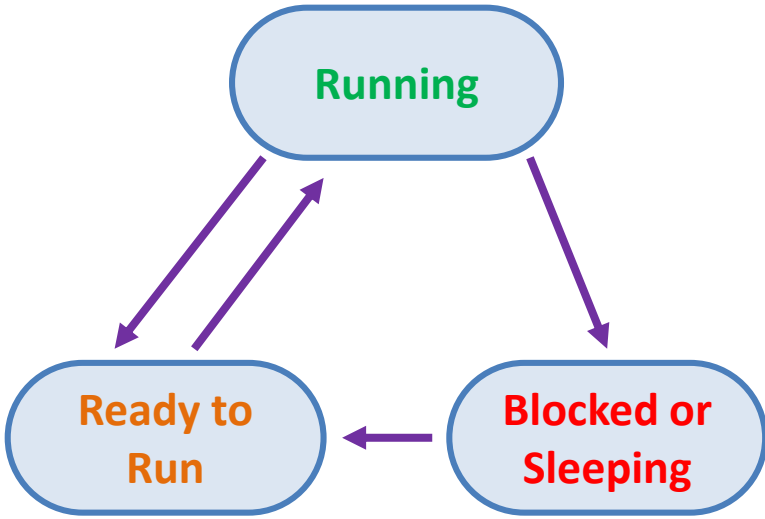
- For a set slice of time allocated by the scheduler.

Ready to Run

- Ready to be executed and waiting for the scheduler to allow it to run.
- The scheduler decides which of the ready processes to run next.

Blocked or Sleeping

- Sleeping for a set time period or waiting for an event to occur before continuing (e.g. IO or signal).



Processes

Attributes

- Like files, processes have a number of attributes associated with them.
- These attributes are stored in a process table maintained by the kernel that tracks each process by a process id (PID).
- This is similar to the idea of inodes and inode numbers as used in the UNIX file system.

Processes Attributes

Some process attributes include:

Attribute	Description
PID	A unique number assigned to the process used to identify it.
PPID	Process ID of the parent process.
Name	Name of the process. Normally the name of the command or program.
Real UID	UID of the user that initiated the process.
Real GID	GID of the group of the user who initiated the process.
Effective UID	Normally the same as Real UID. But in some cases programs may be ran as users that did not initiate them (SUID, SUDO, etc).
Effective GID	Same as above but for group/GID.
CWD	The current working directory from where the process was run.
List of Open Files	List of files opened by the process.
Environment Variables	Current environment variables like HOME, PATH and SHELL.
TTY	The terminal the process is connected to (if it is connected to one).

Processes

Hierarchy

- Every process is initialized (started) by another process expect for init the first process (normally given the PID 1) which is started by the kernel.
- Processes inherit many of their attributes from the parent processes who created them.
- When running commands via the shell, the shell is normally the parent process of the commands you issue.
- These parent/child relations create a hierarchy of processes leading back to the init process.

Processes

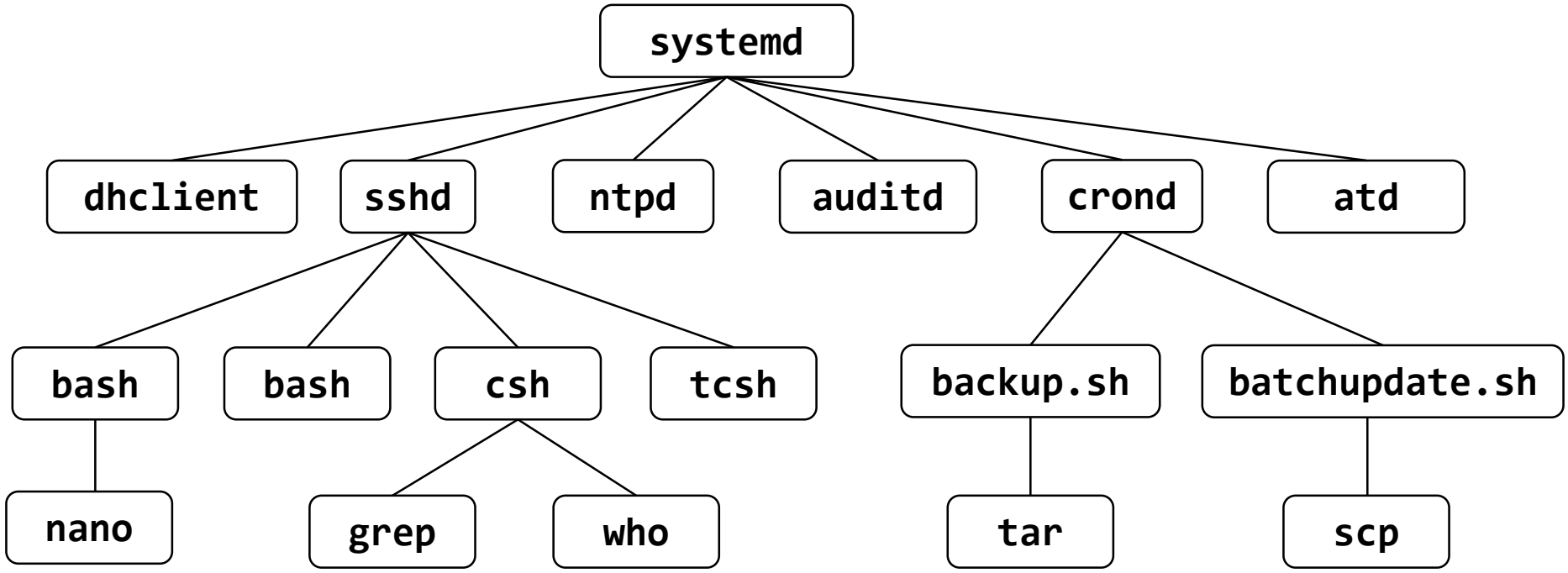
Hierarchy

On the course server the **init** process is **systemd**

- Every process is initialized (started) by another process expect for init the first process (normally given the PID 1) which is started by the kernel.
- Processes inherit many of their attributes from the parent processes who created them.
- When running commands via the shell, the shell is normally the parent process of the commands you issue.
- These parent/child relations create a hierarchy of processes leading back to the init process.

Processes

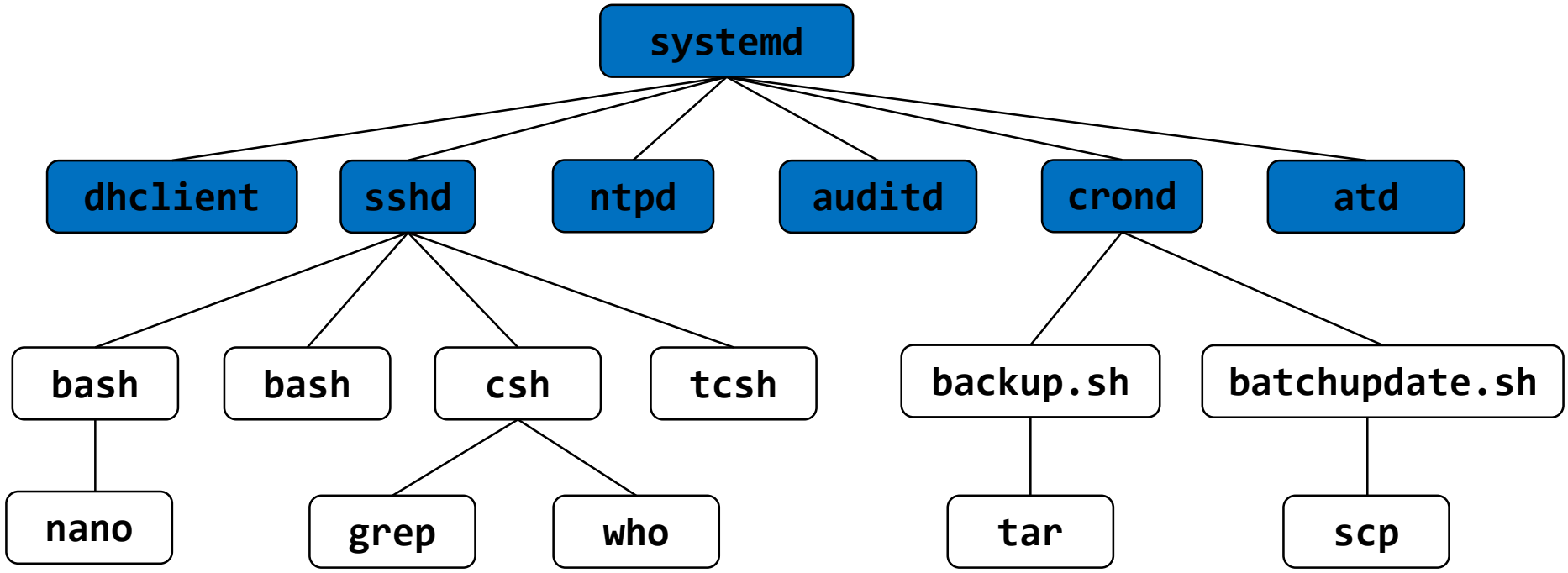
Hierarchy



Processes
Hierarchy

Processes with no controlling terminal are called daemons

They can not write to the terminal or be terminated by pressing Ctrl-C



Processes

Viewing Processes & Attributes

We can view a listing of processes and their attributes with the **`ps`** command.

```
ps [options]
```

Processes

Viewing Processes & Attributes

Some useful **ps** options:

no options	List all processes for the current terminal and user.
-e, -A	List all processes on the system (not just yours).
-p *PID*	List a process by PID.
--ppid *PPID*	List processes by PPID.
-C *CMD*	List processes by command name.
-t *TTY*	List processes by terminal.
-U *UID* OR *Name*	List processes by real user.
-f, -F	Full format and extra full format listing.
-l, -j	Long format and jobs format.
-H	Show process hierarchy.
-h	No header.

Processes

Viewing Processes & Attributes

Examples:

List processes for my terminal

```
[dservos5@cs2211b ~]$ ps
  PID TTY          TIME CMD
 9717 pts/7    00:00:00 bash
10553 pts/7    00:00:00 ps
```

Same but with long formatting

```
[dservos5@cs2211b ~]$ ps -l
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S 17789  9717  9716  0  80   0 - 31454 do_wai pts/7    00:00:00 bash
0 R 17789 10585  9717  0  80   0 - 37234 -      pts/7    00:00:00 ps
```

Processes

Viewing Processes & Attributes

Examples:

List processes for user dservos5 with full formatting

```
[dservos5@cs2211b ~]$ ps -fU dservos5
UID        PID  PPID  C STIME TTY          TIME CMD
dservos5  9716  9710  0 20:57 ?        00:00:00 sshd: dservos5@pts/7
dservos5  9717  9716  0 20:57 pts/7    00:00:00 -bash
dservos5  9799 31386  0 20:58 pts/6    00:00:00 man ps
dservos5  9812  9799  0 20:58 pts/6    00:00:00 less -s
dservos5 10742  9717  0 21:12 pts/7    00:00:00 ps -fU dservos5
dservos5 31385 31378  0 18:11 ?        00:00:00 sshd: dservos5@pts/6
dservos5 31386 31385  0 18:11 pts/6    00:00:00 -bash
```

Processes

Viewing Processes & Attributes

Examples:

List processes for user dservos5 with full formatting

```
[dservos5@cs2211b ~]$ ps -fU dservos5
UID          PID  PPID  C STIME TTY          TIME CMD
dservos5    9716  9710  0 20:57 ?        00:00:00 sshd: dservos5@pts/7
dservos5    9717  9716  0 20:57 pts/7    00:00:00 -bash
dservos5    9799 31386  0 20:58 pts/6    00:00:00 man ps
dservos5    9812  9799  0 20:58 pts/6    00:00:00 less -s
dservos5   10742  9717  0 21:12 pts/7    00:00:00 ps -fU dservos5
dservos5   31385 31378  0 18:11 ?        00:00:00 sshd: dservos5@pts/6
dservos5   31386 31385  0 18:11 pts/6    00:00:00 -bash
```

Looking at man page for ps in pts/6

Processes

Viewing Processes & Attributes

Examples:

List processes for user dservos5 with full formatting

```
[dservos5@cs2211b ~]$ ps -fU dservos5
UID          PID  PPID  C STIME TTY          TIME CMD
dservos5    9716  9710  0 20:57 ?        00:00:00 sshd: dservos5@pts/7
dservos5    9717  9716  0 20:57 pts/7    00:00:00 -bash
dservos5    9799 31386  0 20:58 pts/6    00:00:00 man ps
dservos5    9812  9799  0 20:58 pts/6    00:00:00 less -s
dservos5   10742  9717  0 21:12 pts/7    00:00:00 ps -fU dservos5
dservos5   31385 31378  0 18:11 ?        00:00:00 sshd: dservos5@pts/6
dservos5   31386 31385  0 18:11 pts/6    00:00:00 -bash
```

Running ps –fU dservos5 in terminal pts/7

Processes

Viewing Processes & Attributes

Examples:

List processes for user dservos5 with full formatting

```
[dservos5@cs2211b ~]$ ps -fU dservos5
UID          PID  PPID  C STIME TTY          TIME CMD
dservos5    9716  9710  0 20:57 ?        00:00:00 sshd: dservos5@pts/7
dservos5    9717  9716  0 20:57 pts/7    00:00:00 -bash
dservos5    9799 31386  0 20:58 pts/6    00:00:00 man ps
dservos5    9812  9799  0 20:58 pts/6    00:00:00 less -s
dservos5   10742  9717  0 21:12 pts/7    00:00:00 ps -fU dservos5
dservos5   31385 31378  0 18:11 ?        00:00:00 sshd: dservos5@pts/6
dservos5   31386 31385  0 18:11 pts/6    00:00:00 -bash
```

Parent of ps command is bash

Processes

Viewing Processes & Attributes

Examples:

List processes for user dservos5 with full formatting

```
[dservos5@cs2211b ~]$ ps -fU dservos5
UID          PID  PPID  C STIME TTY          TIME CMD
dservos5    9716  9710  0 20:57 ?        00:00:00 sshd: dservos5@pts/7
dservos5    9717  9716  0 20:57 pts/7    00:00:00 -bash
dservos5    9799 31386  0 20:58 pts/6    00:00:00 man ps
dservos5    9812  9799  0 20:58 pts/6    00:00:00 less -s
dservos5   10742  9717  0 21:12 pts/7    00:00:00 ps -fU dservos5
dservos5   31385 31378  0 18:11 ?        00:00:00 sshd: dservos5@pts/6
dservos5   31386 31385  0 18:11 pts/6    00:00:00 -bash
```

Parent of bash command is sshd

Processes

Viewing Processes & Attributes

Examples:

Find the children of process 1 (init)

```
[dservos5@cs2211b ~]$ ps --ppid 1
  PID TTY          TIME CMD
  386 ?        00:01:06 systemd-journal
  438 ?        00:00:00 systemd-udevd
  473 ?        00:00:10 auditd
  560 ?        00:00:05 polkitd
  562 ?        00:00:22 dbus-daemon
  586 ?        00:01:45 rsyslogd
  587 ?        00:00:04 sssd
  592 ?        00:01:20 irqbalance
  609 ?        00:01:37 ntpd
  636 ?        00:00:00 gssproxy
  652 ?        00:00:00 rpc.gssd
  782 ?        00:00:18 systemd-logind
  820 ?        00:00:00 dhclient
  879 ?        00:03:17 tuned
  980 ?        00:00:14 master
 1034 ?        00:00:06 crond
 1036 ttyS0    00:00:00 agetty
 1041 ?        00:00:00 atd
 1042 tty1     00:00:00 agetty
20592 ?        00:00:00 sshd
20976 ?        00:00:00 sshd
21200 ?        00:00:00 oddjobd
26848 ?        00:00:03 sshd
```

```
26848 ?        00:00:03 sshd
```

Processes

Controlling Processes

- We can control processes we are the owner of (that have our UID as an attribute).
- The **`kill`** command allows us to send signals to processes to terminate or suspend them.
- In most cases, terminating a parent process will also terminate the child processes.
- In cases when the child is not terminated, they are considered orphans and become the child of the init process.

Processes

Controlling Processes

`kill` command syntax:

```
kill [options] pid …
```

Some options:

no options	Sends SIGTERM signal (default termination signal).
-s *SIGNAL*	The signal to send to the process.
-*n*	Same as -s but uses the signal number.

Processes

Controlling Processes

Example:

Kill the process with PID 12418

```
[dservos5@cs2211b ~]$ kill 12418
```

In some cases, a process will not terminate even after you send the SIGTERM signal. You can ensure it terminates with SIGKILL

```
[dservos5@cs2211b ~]$ kill -s KILL 12418
```

OR

```
[dservos5@cs2211b ~]$ kill -9 12418
```

Jobs

- Unix is a multi-tasking operating system.
  - Some of these tasks are being done by other users logged in.
  - Some are being done by you in the background.
  - So far we have only been running one command at a time in the foreground.
- We are able to run multiple commands at once and control these commands using jobs and job control.

Jobs

- Most shells support the concept of jobs and job control.
- Jobs are a group of processes used together to accomplish some goal.
- **Example:**

```
ls | wc
```

- Would create two processes (one for ls and one for wc) but would be one job.

Jobs

Background

- By default when we run a command via the shell, it creates a process that runs in the foreground.
- We can use the & symbol to run a job in the background.
- **Example:**

```
ls /usr/bin > binlist.txt &
```

Jobs

Background

When you put a task in background

- task keeps running, but you continue to work at the shell in the foreground
- if any output is done, it appears on your screen immediately (can be confusing)
- if input is required, process prints a message and stops
- when it is done, a message will be printed

Jobs

Background

Example:

/cs2211/week4/printdate.sh

Prints timestamp every second for 30 seconds.

```
[dservos5@cs2211b week4]$ ./printdate.sh
1517281896
1517281897
1517281898
1517281899
1517281900
1517281901
…
```

Jobs

Background

Example:

Run printdate.sh in background

```
[dservos5@cs2211b week4]$ ./printdate.sh &
[2] 14918
[dservos5@cs2211b week4]$ 1517282086
1517282087
1517282088
...
[2]-  Done      ./printdate.sh
```

Jobs

Background

Example:

Run printdate.sh in background

Job ID PID

```
[dservos5@cs2211b week4]$ ./printdate.sh &
[2] 14918
[dservos5@cs2211b week4]$ 1517282086
1517282087
1517282088
...
[2]-  Done                    ./printdate.sh
```

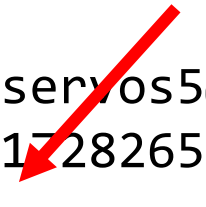
Jobs

Job Control

We can move jobs currently in the foreground to the background two different ways:

Ctrl-Z	Suspend job and move to background.
kill -s STOP *PID*	Sends the suspend signal to a process.

Example: **Ctrl-Z Pressed**



```
[dservos5@cs2211b week4]$ ./printdate.sh
1517282652
^Z
[2]+  Stopped                 ./printdate.sh
```

Jobs

Job Control

We can view a list of background jobs for the current shell session using the **`job`** command.

Example:

```
[dservos5@cs2211b week4]$ jobs
[1]   Stopped                 nano  (wd: ~)
[2]-  Stopped                 ./printdate.sh
[3]+  Stopped                 ./printdate.sh
```

Jobs

Job Control

We can view a list of background jobs for the current shell session using the **`job`** command.

Example:

```
[dservos5@cs2211b week4]$ jobs
[1]   Stopped                 nano  (wd: ~)
[2]-  Stopped                 ./printdate.sh
[3]+  Stopped                 ./printdate.sh
```

Job ID

Jobs

Job Control

We can view a list of background jobs for the current shell session using the **`job`** command.

Example:

```
[dservos5@cs2211b week4]$ jobs
[1]   Stopped                 nano  (wd: ~)
[2]-  Stopped                 ./printdate.sh
[3]+  Stopped                 ./printdate.sh
```

+ denotes last job

- denotes pervious job

Jobs

Job Control

We can view a list of background jobs for the current shell session using the **`job`** command.

Example:

```
[dservos5@cs2211b week4]$ jobs
[1]   Stopped                 nano  (wd: ~)
[2]-  Stopped                 ./printdate.sh
[3]+  Stopped                 ./printdate.sh
```

Job status

Jobs

Job Control

We can view a list of background jobs for the current shell session using the **`job`** command.

Example:

```
[dservos5@cs2211b week4]$ jobs
[1]   Stopped                 nano  (wd: ~)
[2]-  Stopped                 ./printdate.sh
[3]+  Stopped                 ./printdate.sh
```

Command

Jobs

Job Control

We can bring a job back to the foreground and resume them using the **`fg`** command.

Example:

Resume job number 2 and bring to foreground

```
[dservos5@cs2211b week4]$ fg 2
./printdate.sh
1517283568
1517283569
…
```

Jobs

Job Control

We can resume a job in the background using the **bg** command. It will resume but be running in the background like if we press Ctrl-Z.

Example:

Resume job number 3 and in the background

```
[dservos5@cs2211b week4]$ bg 3
[3]+ ./printdate.sh &
[dservos5@cs2211b week4]$ 1517283683
1517283684
1517283685
…
[3]+  Done                    ./printdate.sh
```

Jobs

Job Control

We can terminate a suspended or backgrounded job using the **`kill`** command and its PID like we saw previously or we can use the % symbol to denote a Job ID rather than a PID.

Example:

Terminate backgrounded job 1:

```
[dservos5@cs2211b week4]$ jobs
[1]+  Stopped                 ./printdate.sh
[dservos5@cs2211b week4]$ kill %1
[1]+  Stopped                 ./printdate.sh
```

`top` Command

- **`top`** command displays the top *n* processes on the system by CPU time, memory, etc.
- Information is updated live to the terminal.
- Supports interactive commands while running:
  - h help
  - c display full command
  - M sort by memory usage
  - P sort by CPU usage (%)
  - T sort by time (CPU time)
  - i show/hide idle processes
  - n set the number of processes displayed
  - u display the processes of a given user
  - k kill a given process
  - q quit

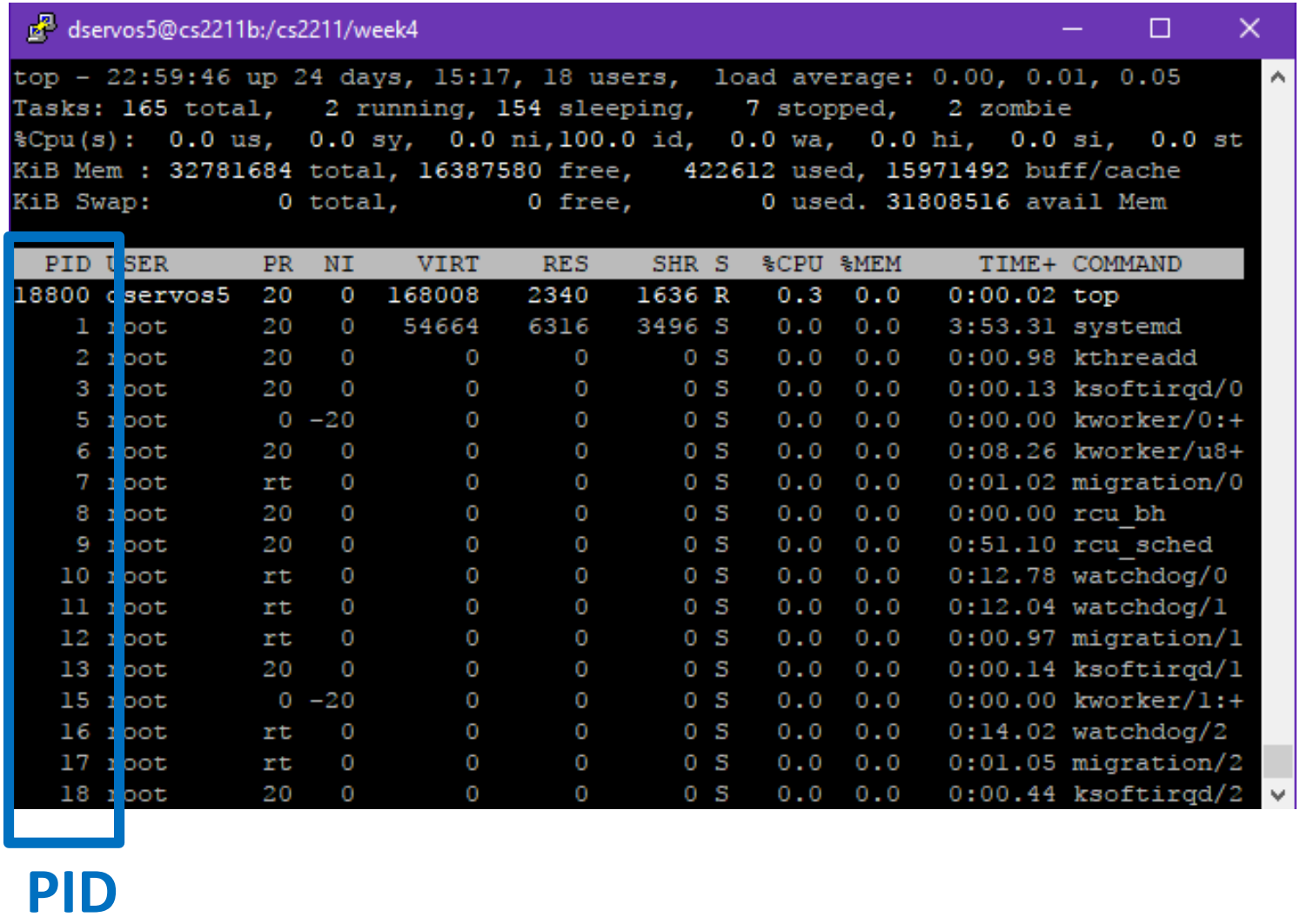
top Command

dservos5@cs2211b:/cs2211/week4

```
top - 22:59:46 up 24 days, 15:17, 18 users,  load average: 0.00, 0.01, 0.05
Tasks: 165 total,   2 running, 154 sleeping,   7 stopped,   2 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 32781684 total, 16387580 free,   422612 used, 15971492 buff/cache
KiB Swap:        0 total,        0 free,        0 used. 31808516 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
18800 dservos5  20   0  168008   2340   1636 R   0.3  0.0   0:00.02 top
    1 root      20   0   54664   6316   3496 S   0.0  0.0   3:53.31 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.98 kthreadd
    3 root      20   0       0      0      0 S   0.0  0.0   0:00.13 ksoftirqd/0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:+
    6 root      20   0       0      0      0 S   0.0  0.0   0:08.26 kworker/u8+
    7 root      rt   0       0      0      0 S   0.0  0.0   0:01.02 migration/0
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 rcu_bh
    9 root      20   0       0      0      0 S   0.0  0.0   0:51.10 rcu_sched
   10 root      rt   0       0      0      0 S   0.0  0.0   0:12.78 watchdog/0
   11 root      rt   0       0      0      0 S   0.0  0.0   0:12.04 watchdog/1
   12 root      rt   0       0      0      0 S   0.0  0.0   0:00.97 migration/1
   13 root      20   0       0      0      0 S   0.0  0.0   0:00.14 ksoftirqd/1
   15 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/1:+
   16 root      rt   0       0      0      0 S   0.0  0.0   0:14.02 watchdog/2
   17 root      rt   0       0      0      0 S   0.0  0.0   0:01.05 migration/2
   18 root      20   0       0      0      0 S   0.0  0.0   0:00.44 ksoftirqd/2
```

top Command



PID

top Command

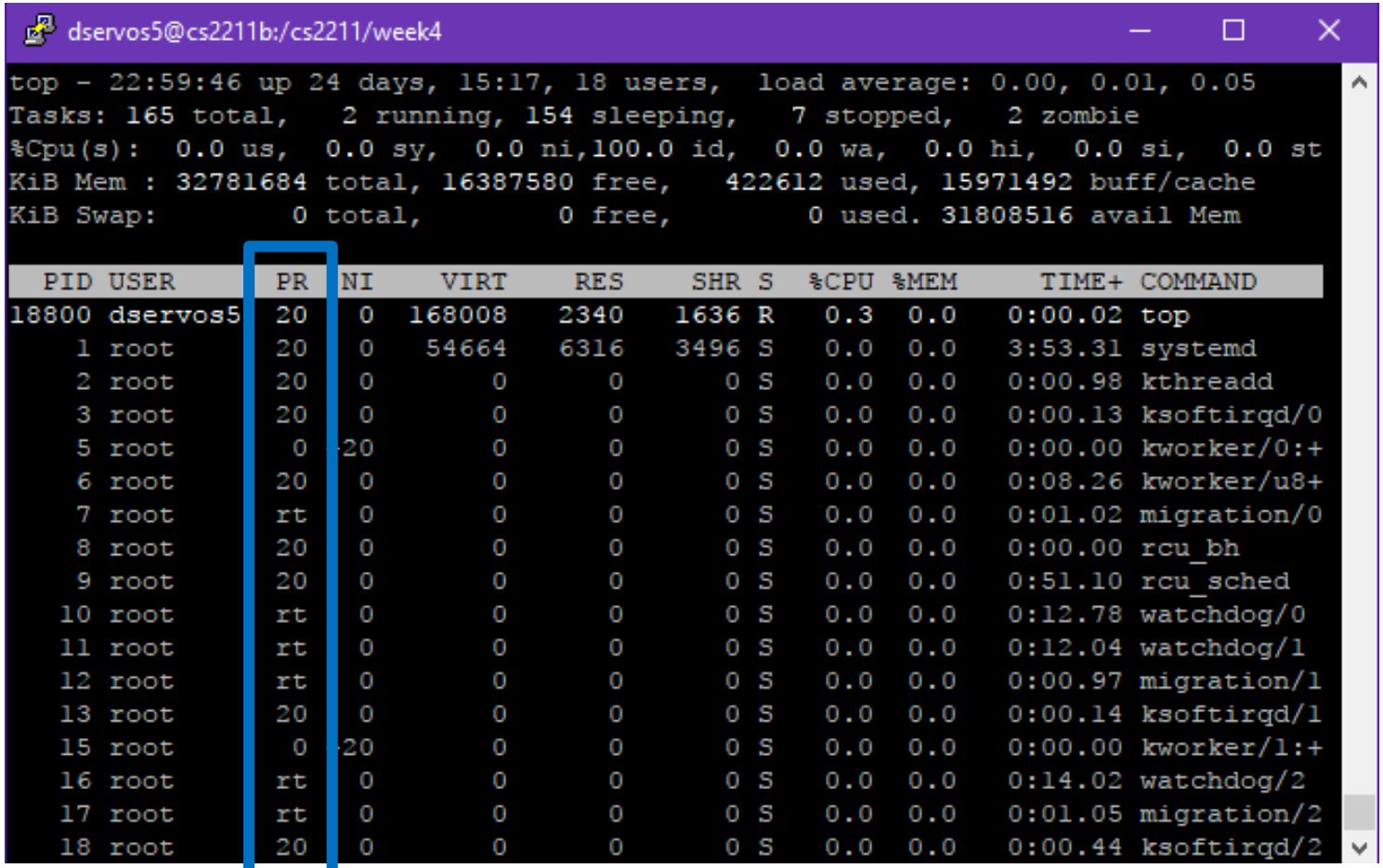
dservos5@cs2211b:/cs2211/week4

```
top - 22:59:46 up 24 days, 15:17, 18 users,  load average: 0.00, 0.01, 0.05
Tasks: 165 total,   2 running, 154 sleeping,   7 stopped,   2 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 32781684 total, 16387580 free,   422612 used, 15971492 buff/cache
KiB Swap:        0 total,        0 free,        0 used. 31808516 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
18800 dservos5  20   0  168008   2340   1636 R   0.3  0.0   0:00.02 top
    1 root      20   0   54664   6316   3496 S   0.0  0.0   3:53.31 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.98 kthreadd
    3 root      20   0       0      0      0 S   0.0  0.0   0:00.13 ksoftirqd/0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:+
    6 root      20   0       0      0      0 S   0.0  0.0   0:08.26 kworker/u8+
    7 root      rt   0       0      0      0 S   0.0  0.0   0:01.02 migration/0
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 rcu_bh
    9 root      20   0       0      0      0 S   0.0  0.0   0:51.10 rcu_sched
   10 root      rt   0       0      0      0 S   0.0  0.0   0:12.78 watchdog/0
   11 root      rt   0       0      0      0 S   0.0  0.0   0:12.04 watchdog/1
   12 root      rt   0       0      0      0 S   0.0  0.0   0:00.97 migration/1
   13 root      20   0       0      0      0 S   0.0  0.0   0:00.14 ksoftirqd/1
   15 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/1:+
   16 root      rt   0       0      0      0 S   0.0  0.0   0:14.02 watchdog/2
   17 root      rt   0       0      0      0 S   0.0  0.0   0:01.05 migration/2
   18 root      20   0       0      0      0 S   0.0  0.0   0:00.44 ksoftirqd/2
```

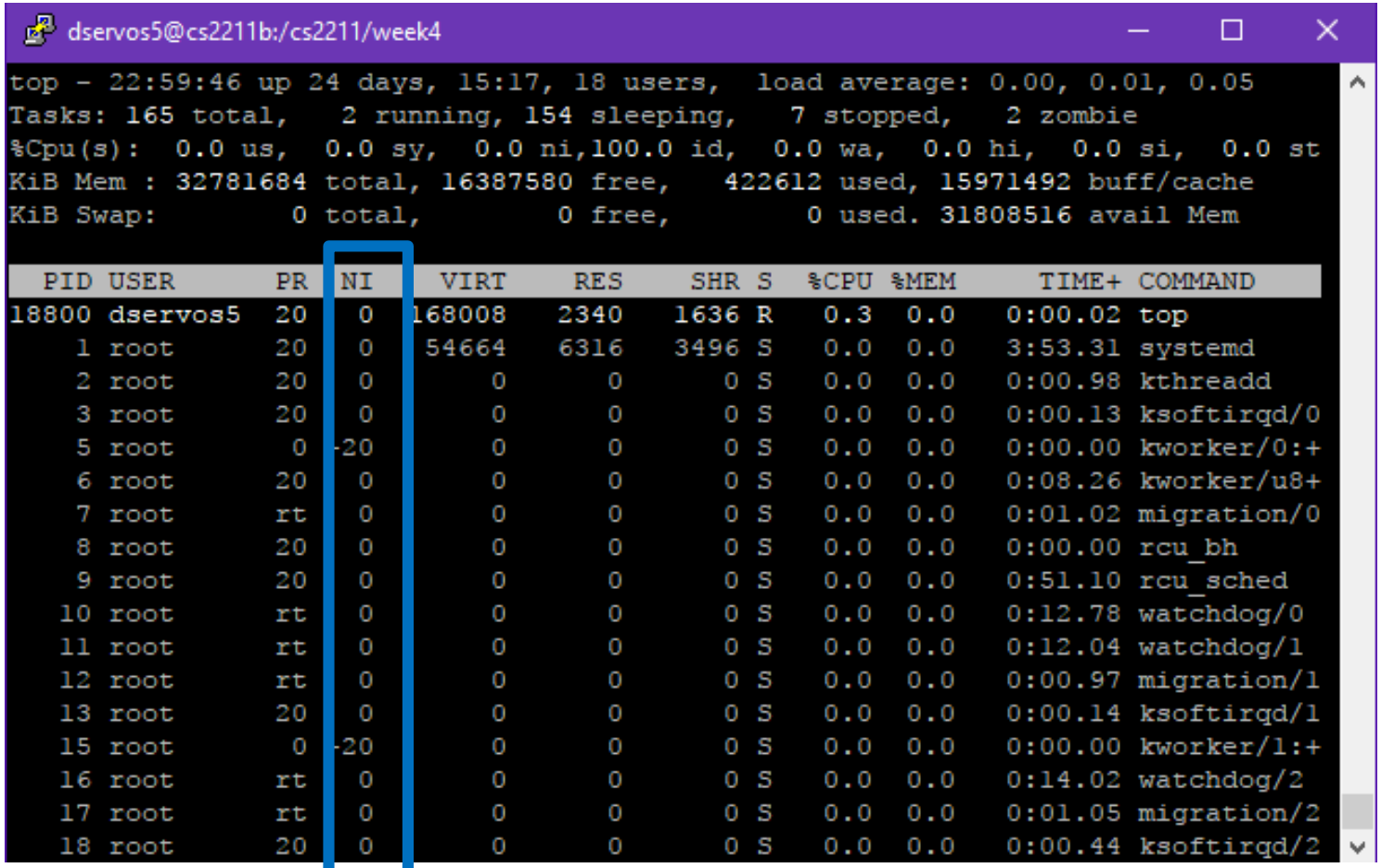
User Name

top Command



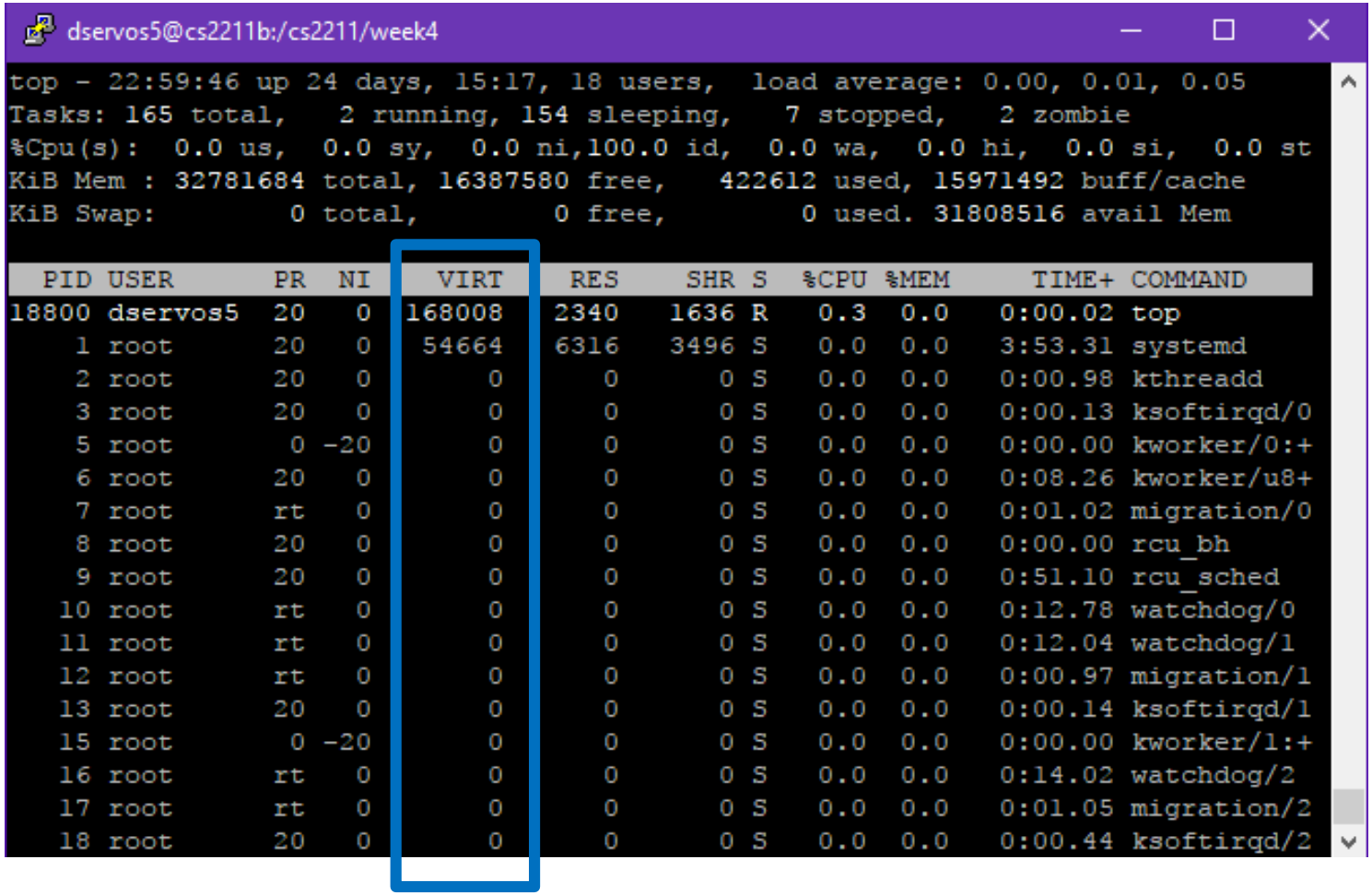
Priority

top Command



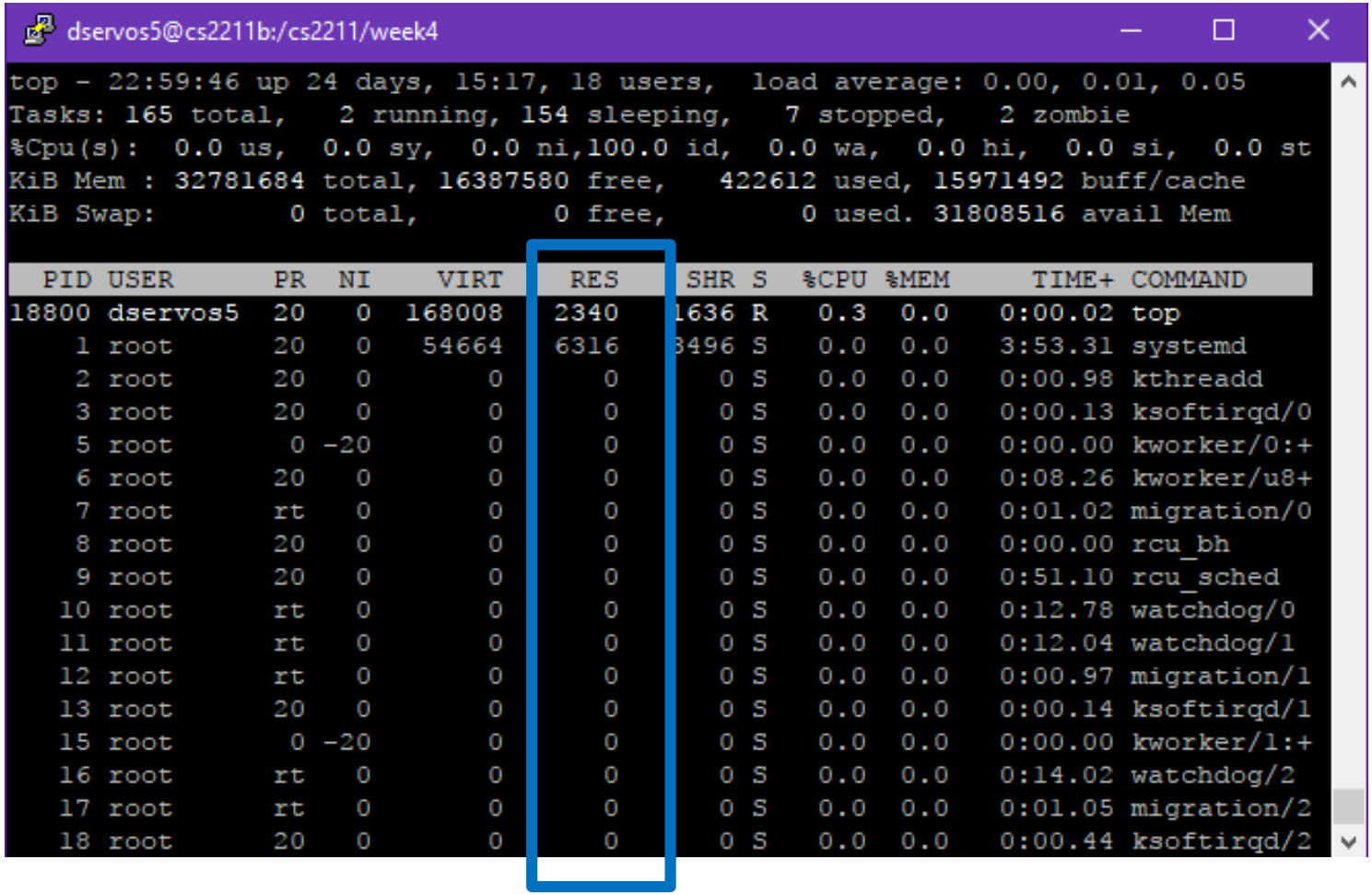
Nice Value

top Command



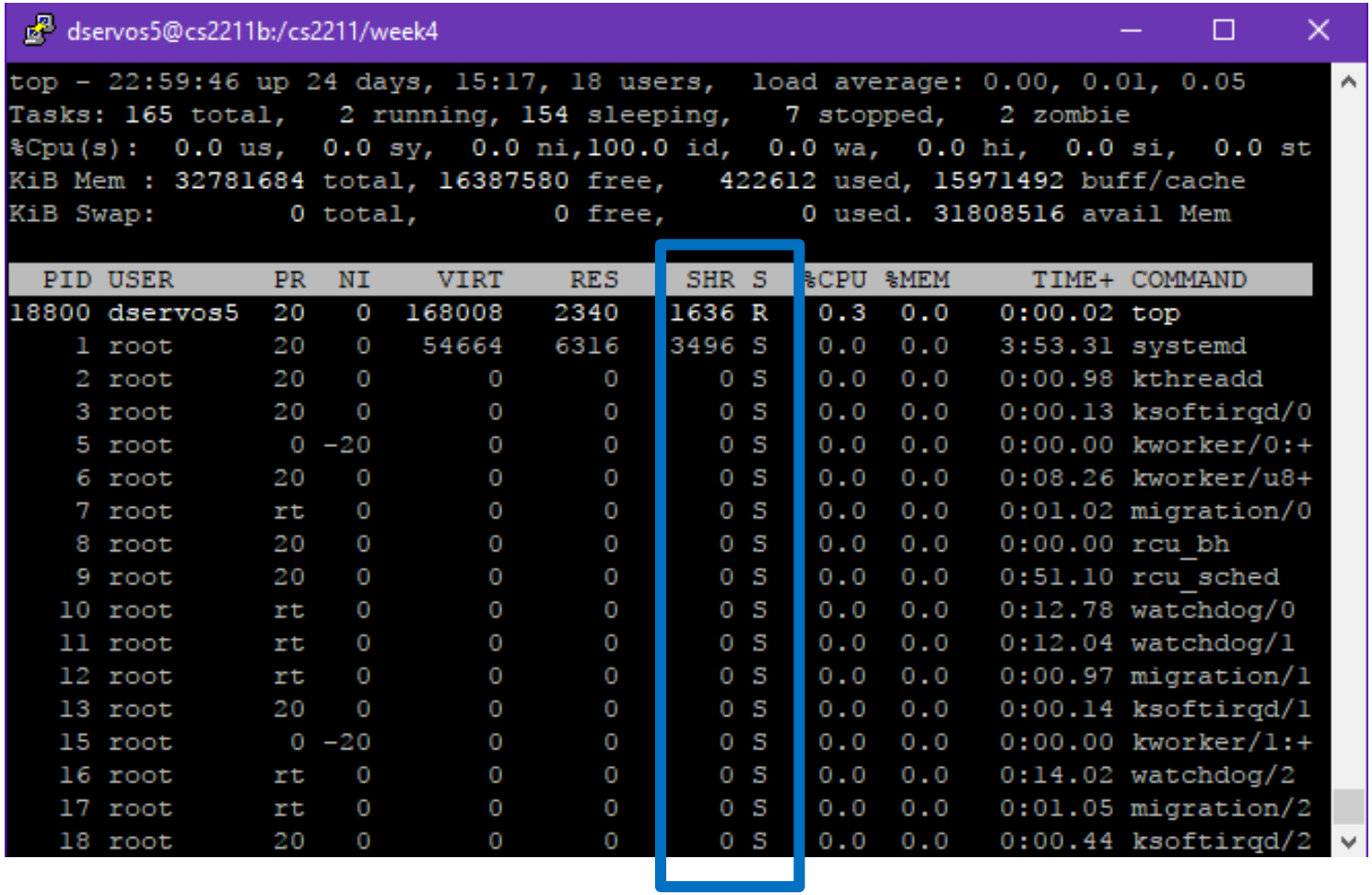
Virtual Memory Size

top Command



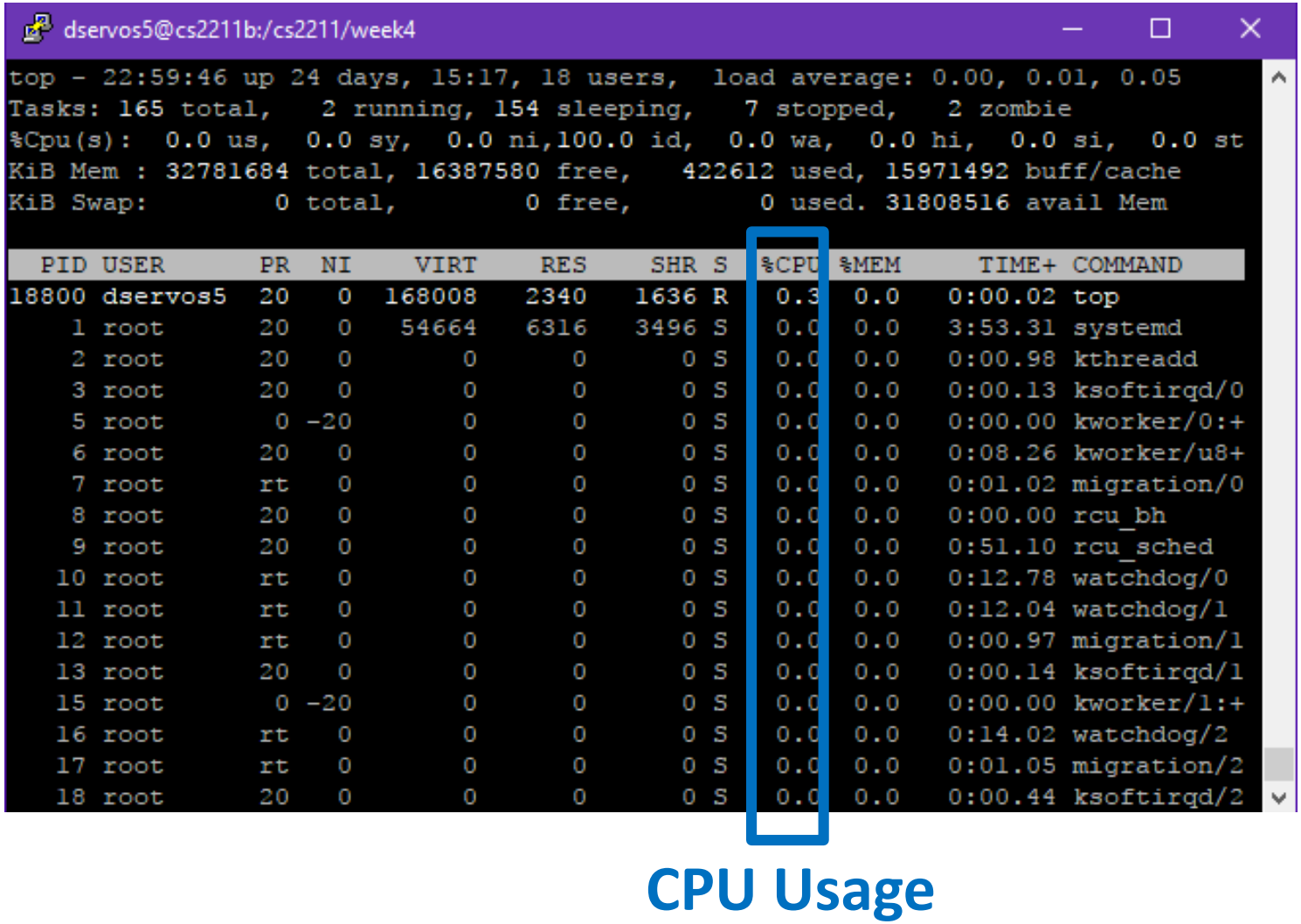
Resident Size (Physical Memory)

top Command



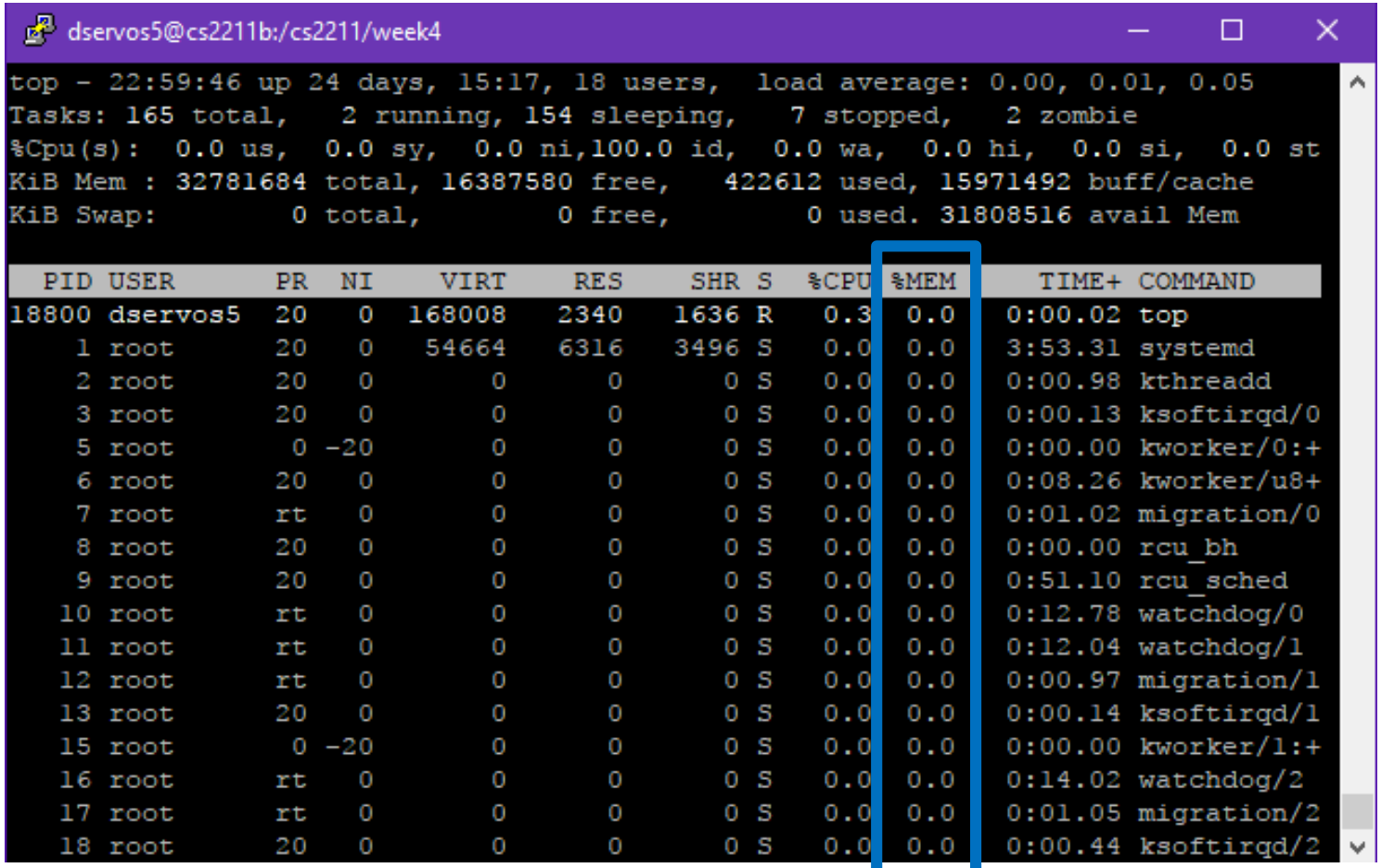
How much of the VIRT size is shared/shareable

top Command



CPU Usage

top Command



Memory Usage (Physical Only)

top Command

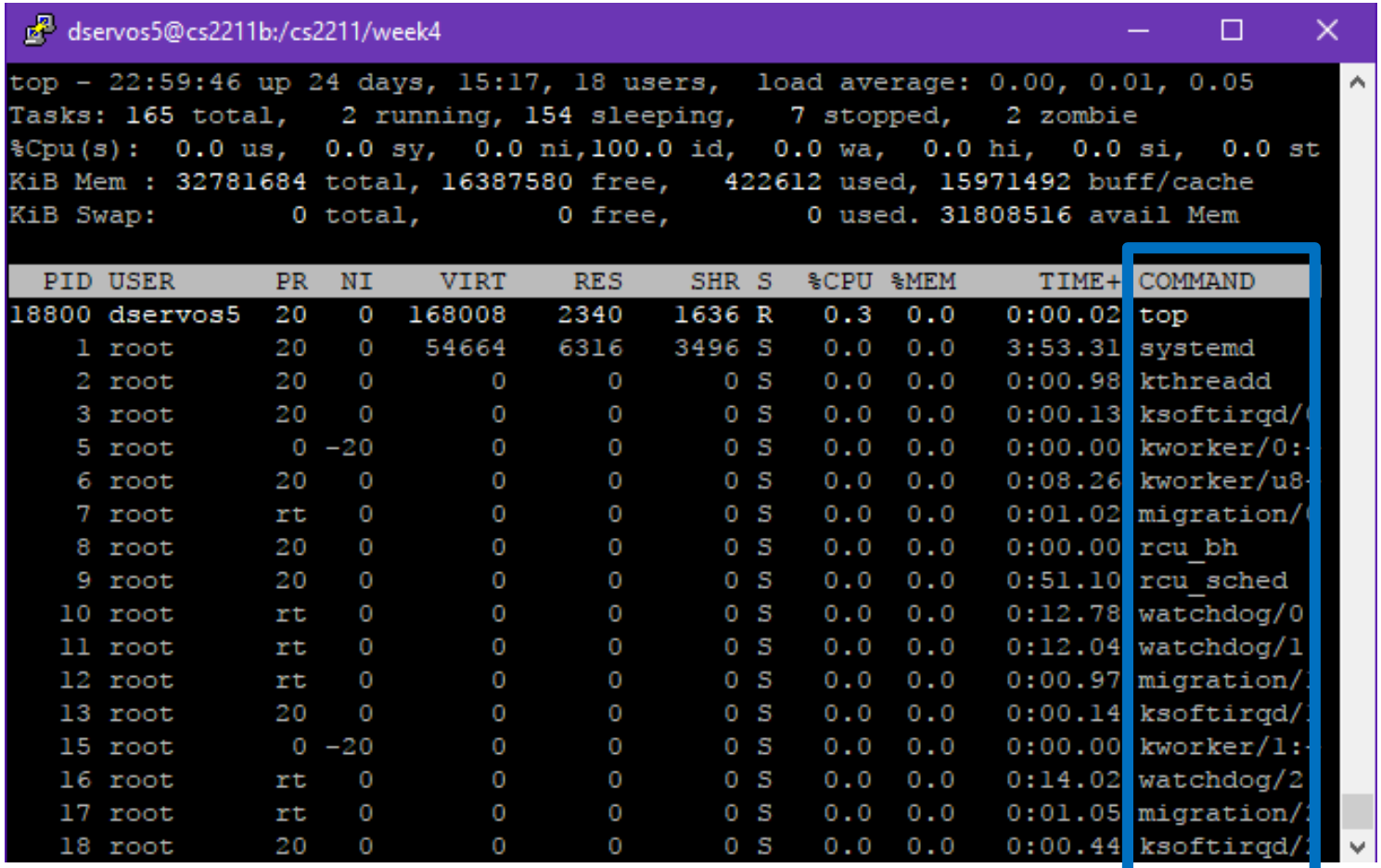
dservos5@cs2211b:/cs2211/week4

```
top - 22:59:46 up 24 days, 15:17, 18 users,  load average: 0.00, 0.01, 0.05
Tasks: 165 total,   2 running, 154 sleeping,   7 stopped,   2 zombie
%Cpu(s):  0.0 us,  0.0 sy,  0.0 ni,100.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem : 32781684 total, 16387580 free,   422612 used, 15971492 buff/cache
KiB Swap:        0 total,        0 free,        0 used. 31808516 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
18800 dservos5  20   0  168008   2340   1636 R   0.3  0.0   0:00.02 top
    1 root      20   0   54664   6316   3496 S   0.0  0.0   3:53.31 systemd
    2 root      20   0       0      0      0 S   0.0  0.0   0:00.98 kthreadd
    3 root      20   0       0      0      0 S   0.0  0.0   0:00.13 ksoftirqd/0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:+
    6 root      20   0       0      0      0 S   0.0  0.0   0:08.26 kworker/u8+
    7 root      rt   0       0      0      0 S   0.0  0.0   0:01.02 migration/0
    8 root      20   0       0      0      0 S   0.0  0.0   0:00.00 rcu_bh
    9 root      20   0       0      0      0 S   0.0  0.0   0:51.10 rcu_sched
   10 root      rt   0       0      0      0 S   0.0  0.0   0:12.78 watchdog/0
   11 root      rt   0       0      0      0 S   0.0  0.0   0:12.04 watchdog/1
   12 root      rt   0       0      0      0 S   0.0  0.0   0:00.97 migration/1
   13 root      20   0       0      0      0 S   0.0  0.0   0:00.14 ksoftirqd/1
   15 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/1:+
   16 root      rt   0       0      0      0 S   0.0  0.0   0:14.02 watchdog/2
   17 root      rt   0       0      0      0 S   0.0  0.0   0:01.05 migration/2
   18 root      20   0       0      0      0 S   0.0  0.0   0:00.44 ksoftirqd/2
```

CPU Time

top Command


dservos5@cs2211b:/cs2211/week4
top - 22:59:46 up 24 days, 15:17, 18 users, load average: 0.00, 0.01, 0.05
Tasks: 165 total, 2 running, 154 sleeping, 7 stopped, 2 zombie
%Cpu(s): 0.0 us, 0.0 sy, 0.0 ni,100.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 32781684 total, 16387580 free, 422612 used, 15971492 buff/cache
KiB Swap: 0 total, 0 free, 0 used. 31808516 avail Mem
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
18800 dservos5 20 0 168008 2340 1636 R 0.3 0.0 0:00.02 top
1 root 20 0 54664 6316 3496 S 0.0 0.0 3:53.31 systemd
2 root 20 0 0 0 0 S 0.0 0.0 0:00.98 kthreadd
3 root 20 0 0 0 0 S 0.0 0.0 0:00.13 ksoftirqd/
5 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kworker/0:
6 root 20 0 0 0 0 S 0.0 0.0 0:08.26 kworker/u8
7 root rt 0 0 0 0 S 0.0 0.0 0:01.02 migration/
8 root 20 0 0 0 0 S 0.0 0.0 0:00.00 rcu_bh
9 root 20 0 0 0 0 S 0.0 0.0 0:51.10 rcu_sched
10 root rt 0 0 0 0 S 0.0 0.0 0:12.78 watchdog/0
11 root rt 0 0 0 0 S 0.0 0.0 0:12.04 watchdog/1
12 root rt 0 0 0 0 S 0.0 0.0 0:00.97 migration/
13 root 20 0 0 0 0 S 0.0 0.0 0:00.14 ksoftirqd/
15 root 0 -20 0 0 0 S 0.0 0.0 0:00.00 kworker/1:
16 root rt 0 0 0 0 S 0.0 0.0 0:14.02 watchdog/2
17 root rt 0 0 0 0 S 0.0 0.0 0:01.05 migration/
18 root 20 0 0 0 0 S 0.0 0.0 0:00.44 ksoftirqd/
Command