Lec-02-1
Process and Process Management

Dr Syed Faisal Hasan and Dr. Hymie Latif
Computing and Information Technology
College of Enterprise and Development
Otago Polytechnic
Dunedin, New Zealand

Bachelor of Information Technology
IN616 – Operating Systems Concepts
Semester 1, 2020

Schedule

- Recap
- Processes

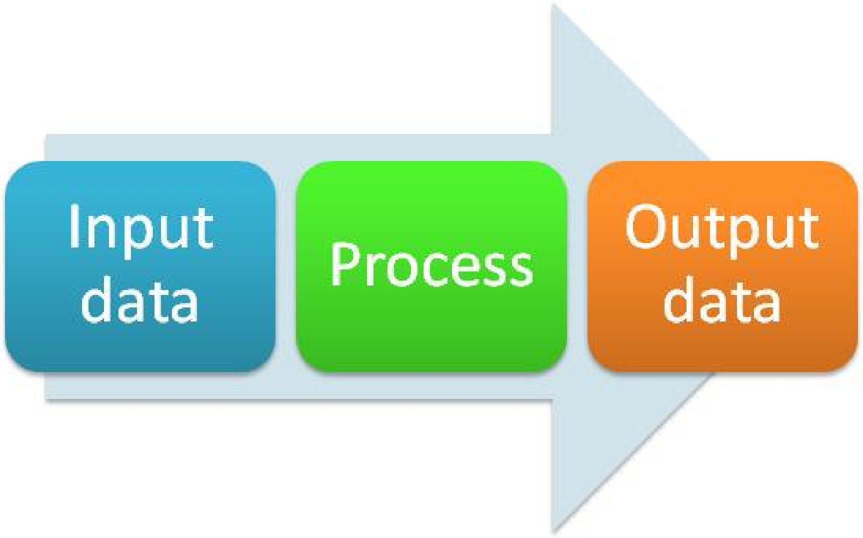
- Process Management

TOPIC:

Processes

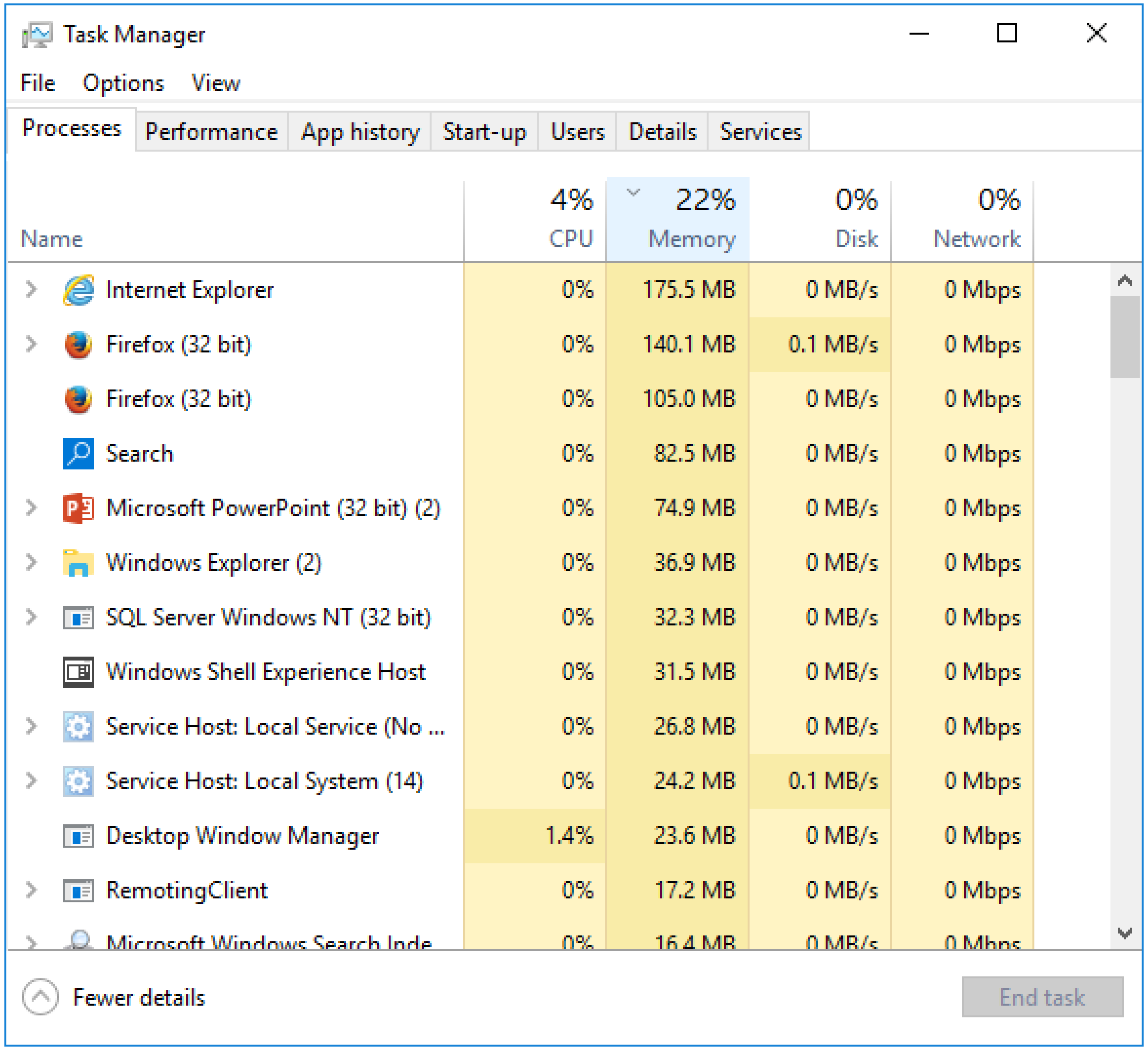
Processes

- ***"Instance of a computer program being executed"***
- ***"Program in execution"***



- Execution principles:
  - Program code loaded from disk to memory
  - Main method is called (not always!)
  - Program becomes a process

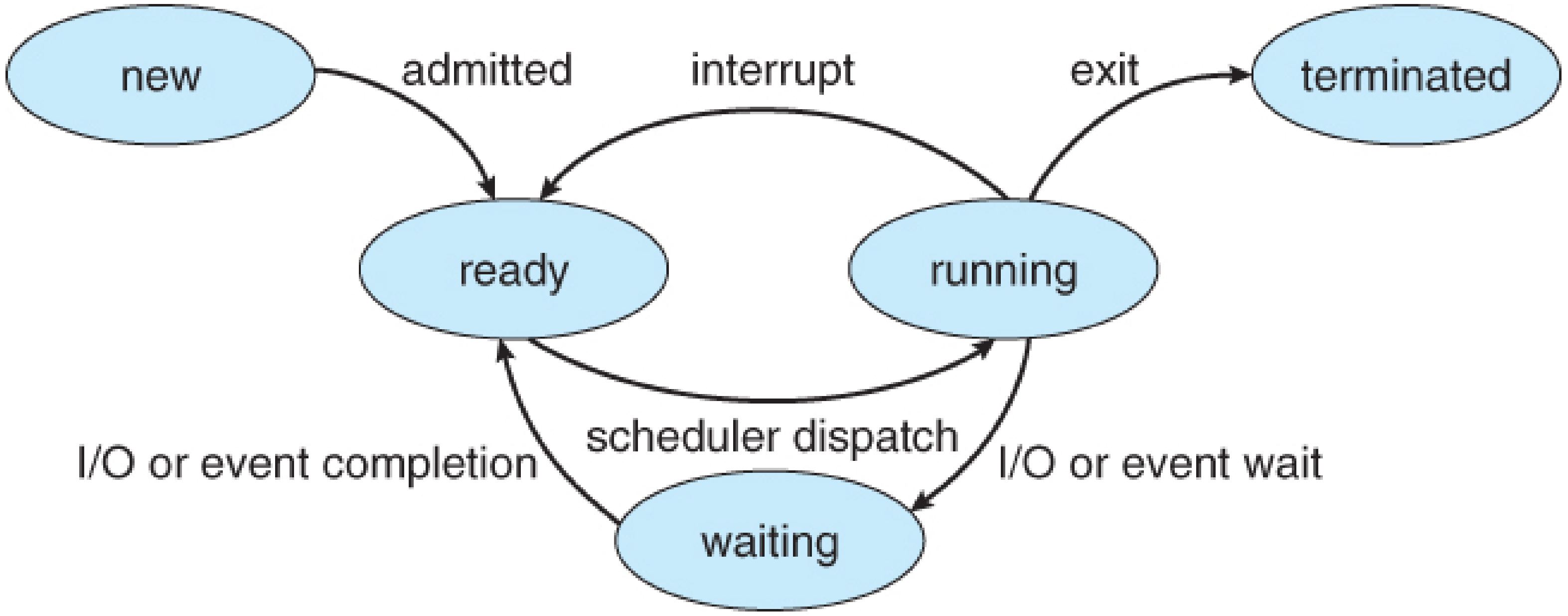
How we view processes


Task Manager
File Options View
Processes Performance App history Start-up Users Details Services
Fewer details
End task

Name	4% CPU	22% Memory	0% Disk	0% Network
Internet Explorer	0%	175.5 MB	0 MB/s	0 Mbps
Firefox (32 bit)	0%	140.1 MB	0.1 MB/s	0 Mbps
Firefox (32 bit)	0%	105.0 MB	0 MB/s	0 Mbps
Search	0%	82.5 MB	0 MB/s	0 Mbps
Microsoft PowerPoint (32 bit) (2)	0%	74.9 MB	0 MB/s	0 Mbps
Windows Explorer (2)	0%	36.9 MB	0 MB/s	0 Mbps
SQL Server Windows NT (32 bit)	0%	32.3 MB	0 MB/s	0 Mbps
Windows Shell Experience Host	0%	31.5 MB	0 MB/s	0 Mbps
Service Host: Local Service (No ...	0%	26.8 MB	0 MB/s	0 Mbps
Service Host: Local System (14)	0%	24.2 MB	0.1 MB/s	0 Mbps
Desktop Window Manager	1.4%	23.6 MB	0 MB/s	0 Mbps
RemotingClient	0%	17.2 MB	0 MB/s	0 Mbps
Microsoft Windows Search Inde	0%	16.4 MB	0 MB/s	0 Mbps

Processes: States

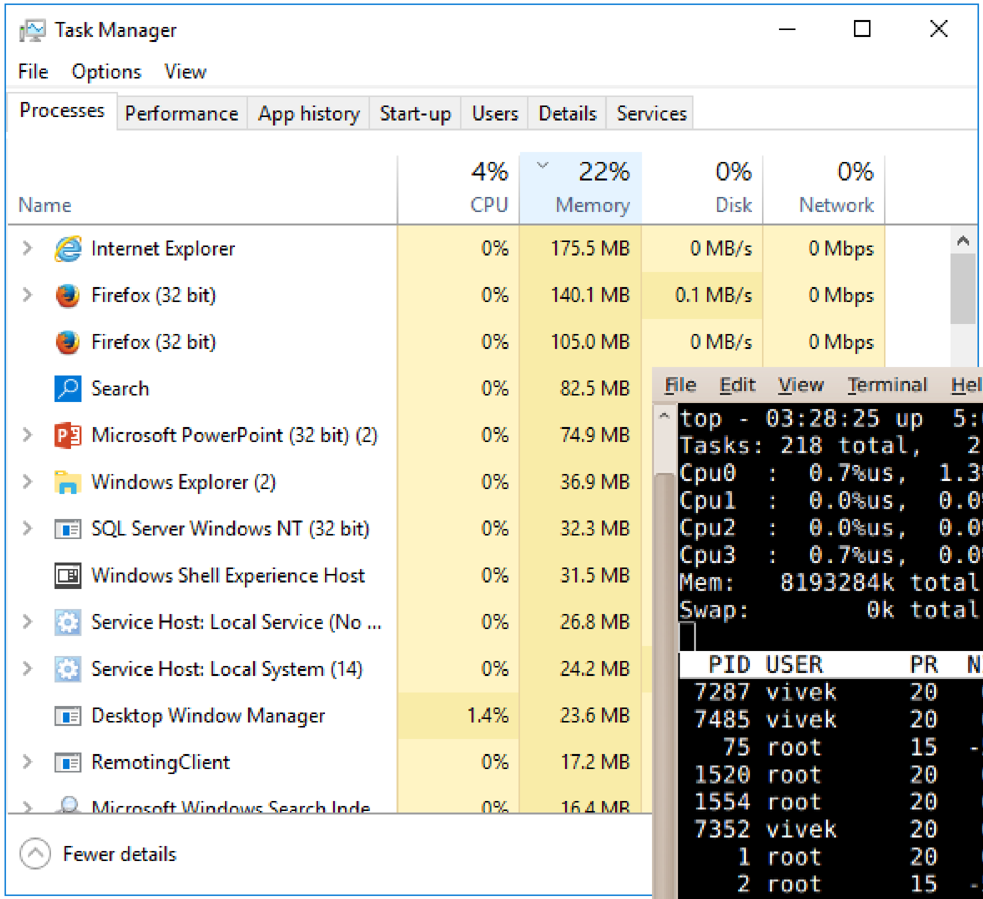
- As a program executes, it changes states



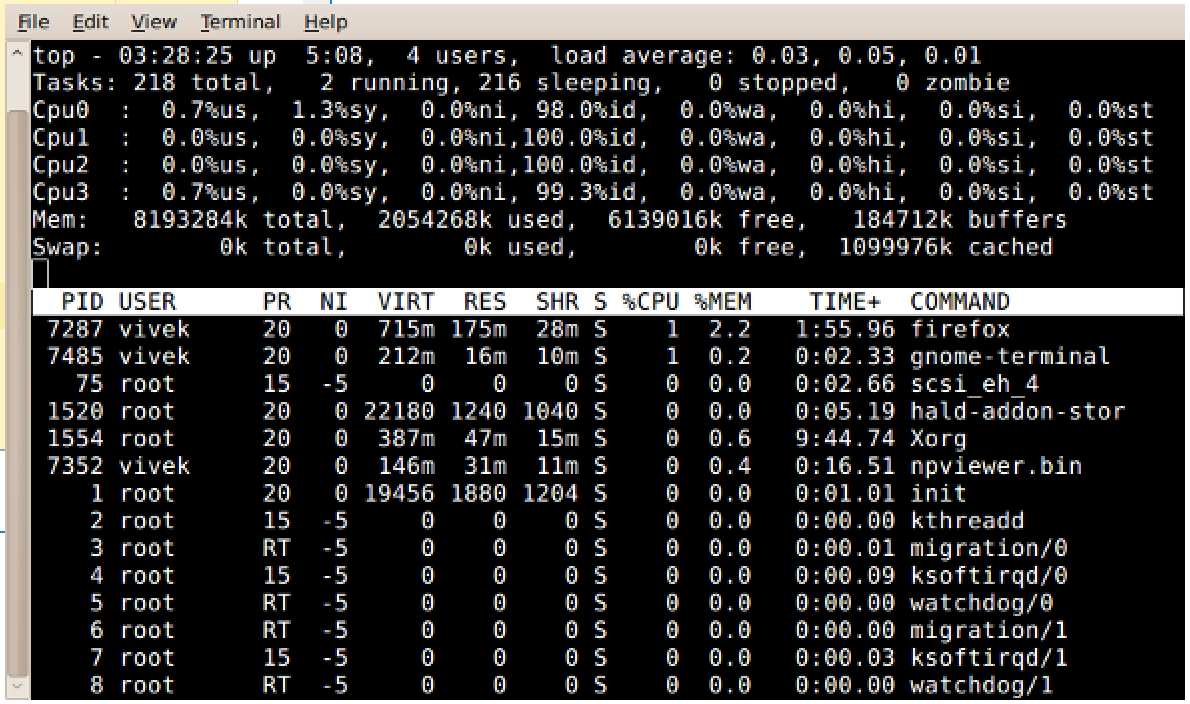
Processes: States (Examples)

- **New:** The process is being created
  - **Double click Firefox.exe, ./script.sh**
- **Running:** Instructions are being executed
  - **Java program in the main function, Bash script running**
- **Waiting:** The process is waiting for some event to occur
  - **wait(20), sleep 10, waiting for user input**
- **Ready:** The process is waiting to be assigned to processor
  - **Waiting for a scheduled event**
- **Terminated:** The process has finished execution
  - **exit 1, quit()**

Process Management


Task Manager
File Options View
Processes Performance App history Start-up Users Details Services
Fewer details

Name	4% CPU	22% Memory	0% Disk	0% Network
Internet Explorer	0%	175.5 MB	0 MB/s	0 Mbps
Firefox (32 bit)	0%	140.1 MB	0.1 MB/s	0 Mbps
Firefox (32 bit)	0%	105.0 MB	0 MB/s	0 Mbps
Search	0%	82.5 MB		
Microsoft PowerPoint (32 bit) (2)	0%	74.9 MB		
Windows Explorer (2)	0%	36.9 MB		
SQL Server Windows NT (32 bit)	0%	32.3 MB		
Windows Shell Experience Host	0%	31.5 MB		
Service Host: Local Service (No ...	0%	26.8 MB		
Service Host: Local System (14)	0%	24.2 MB		
Desktop Window Manager	1.4%	23.6 MB		
RemotingClient	0%	17.2 MB		
Microsoft Windows Search Inde	0%	16.4 MB		



```
top - 03:28:25 up  5:08,  4 users,  load average: 0.03, 0.05, 0.01
Tasks: 218 total,   2 running, 216 sleeping,   0 stopped,   0 zombie
Cpu0  :  0.7%us,  1.3%sy,  0.0%ni, 98.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1  :  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu2  :  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3  :  0.7%us,  0.0%sy,  0.0%ni, 99.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:   8193284k total,  2054268k used,  6139016k free,   184712k buffers
Swap:        0k total,        0k used,        0k free,  1099976k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 7287 vivek     20   0  715m 175m  28m S    1  2.2   1:55.96 firefox
 7485 vivek     20   0  212m  16m  10m S    1  0.2   0:02.33 gnome-terminal
   75 root      15  -5     0    0    0 S    0  0.0   0:02.66 scsi_eh_4
 1520 root      20   0 22180 1240 1040 S    0  0.0   0:05.19 hald-addon-stor
 1554 root      20   0  387m  47m  15m S    0  0.6   9:44.74 Xorg
 7352 vivek     20   0  146m  31m  11m S    0  0.4   0:16.51 npviewer.bin
    1 root      20   0 19456 1880 1204 S    0  0.0   0:01.01 init
    2 root      15  -5     0    0    0 S    0  0.0   0:00.00 kthreadd
    3 root      RT  -5     0    0    0 S    0  0.0   0:00.01 migration/0
    4 root      15  -5     0    0    0 S    0  0.0   0:00.09 ksoftirqd/0
    5 root      RT  -5     0    0    0 S    0  0.0   0:00.00 watchdog/0
    6 root      RT  -5     0    0    0 S    0  0.0   0:00.00 migration/1
    7 root      15  -5     0    0    0 S    0  0.0   0:00.03 ksoftirqd/1
    8 root      RT  -5     0    0    0 S    0  0.0   0:00.00 watchdog/1
```

TOPIC:

Process Management Tools

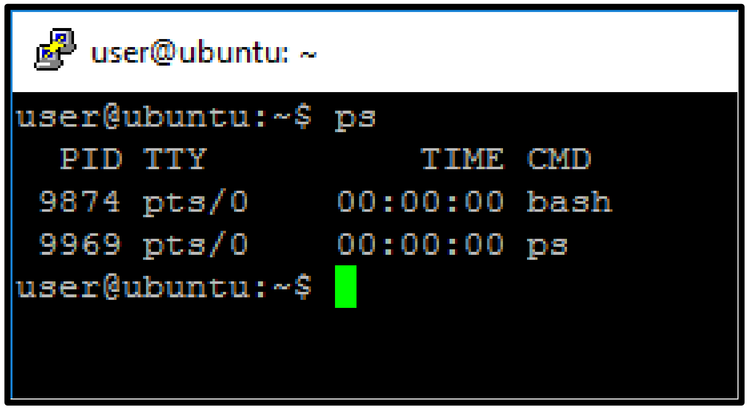
Process Management Tools

- **`ps`**
  - List a (snapshot) of processes
- **`pstree`**
  - Display a tree (snapshot) of processes
- **`top`**
  - Display processes and system information (dynamically)

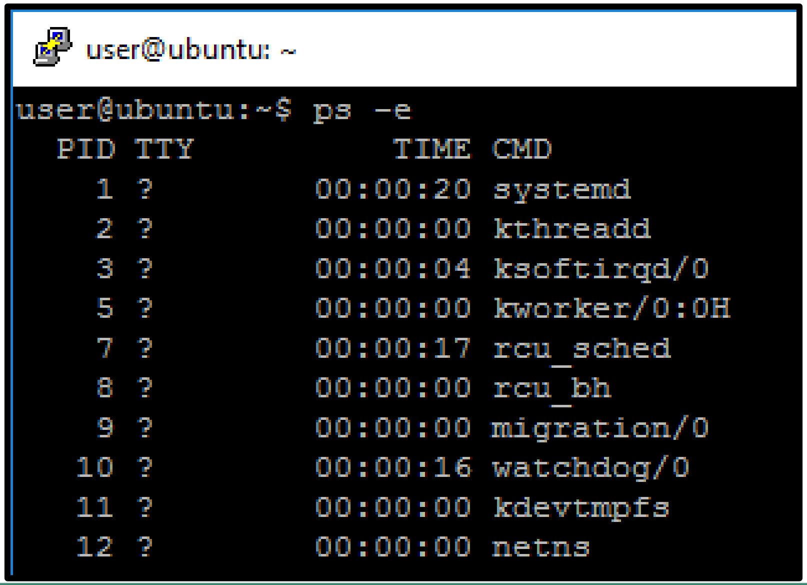
ps

- Reports a snapshot of current processes
  - By default for the user who executed the command

- The list is not dynamic
  - Only reports processes when executed



```
user@ubuntu:~$ ps
  PID TTY          TIME CMD
 9874 pts/0    00:00:00 bash
 9969 pts/0    00:00:00 ps
user@ubuntu:~$ 
```



```
user@ubuntu:~$ ps -e
  PID TTY          TIME CMD
    1 ?        00:00:20 systemd
    2 ?        00:00:00 kthreadd
    3 ?        00:00:04 ksoftirqd/0
    5 ?        00:00:00 kworker/0:0H
    7 ?        00:00:17 rcu_sched
    8 ?        00:00:00 rcu_bh
    9 ?        00:00:00 migration/0
   10 ?        00:00:16 watchdog/0
   11 ?        00:00:00 kdevtmpfs
   12 ?        00:00:00 netns
```

`ps`: Useful arguments

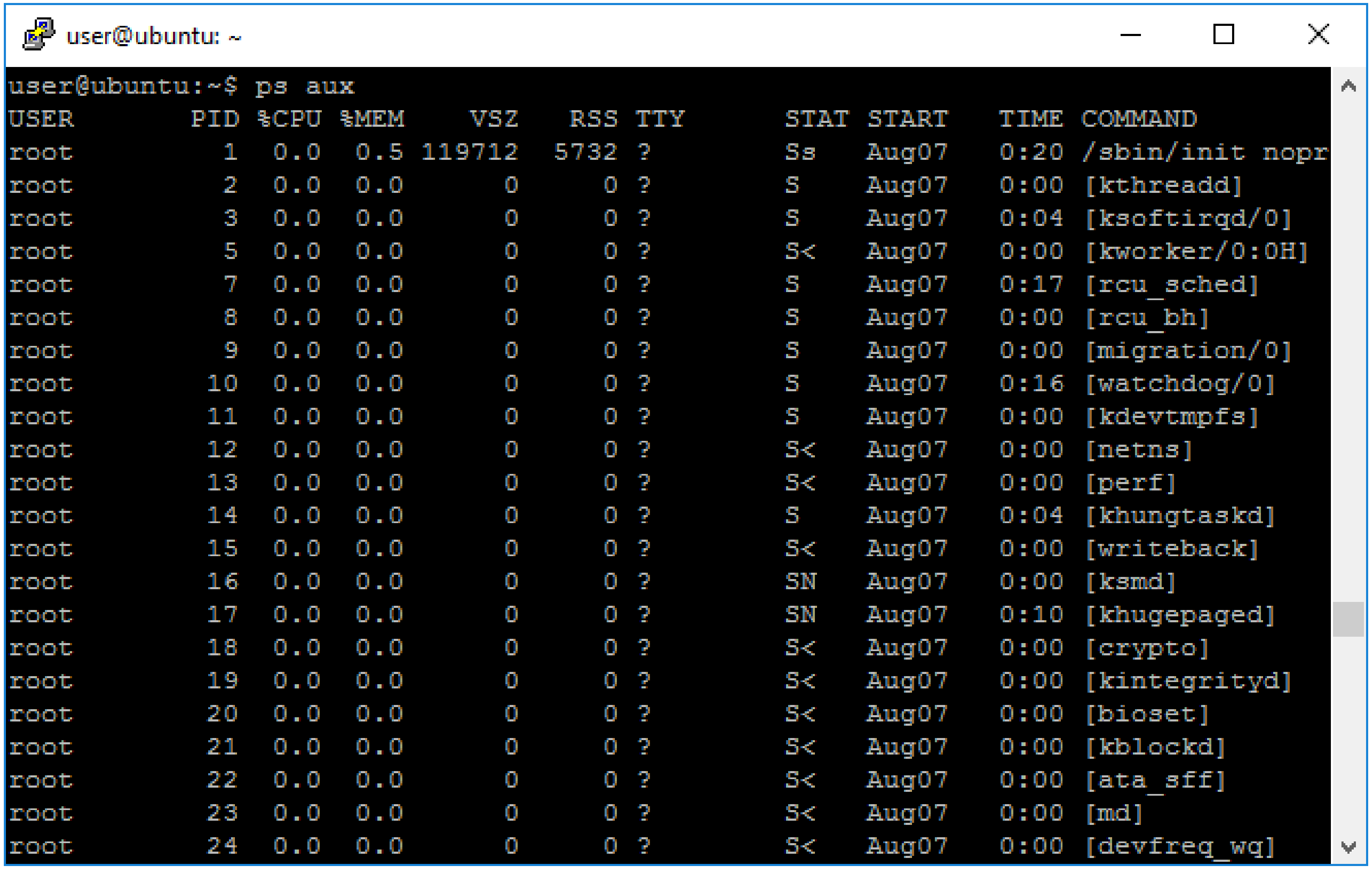
```
ps –e
ps -A
```
– Display all processes (not just user's processes)

```
ps -r
```
– Display only running processes

```
ps -F
```
– Display the full format

```
ps -e | grep ssh
```
– Display only processes matching keyword (using grep)

ps: Full Format

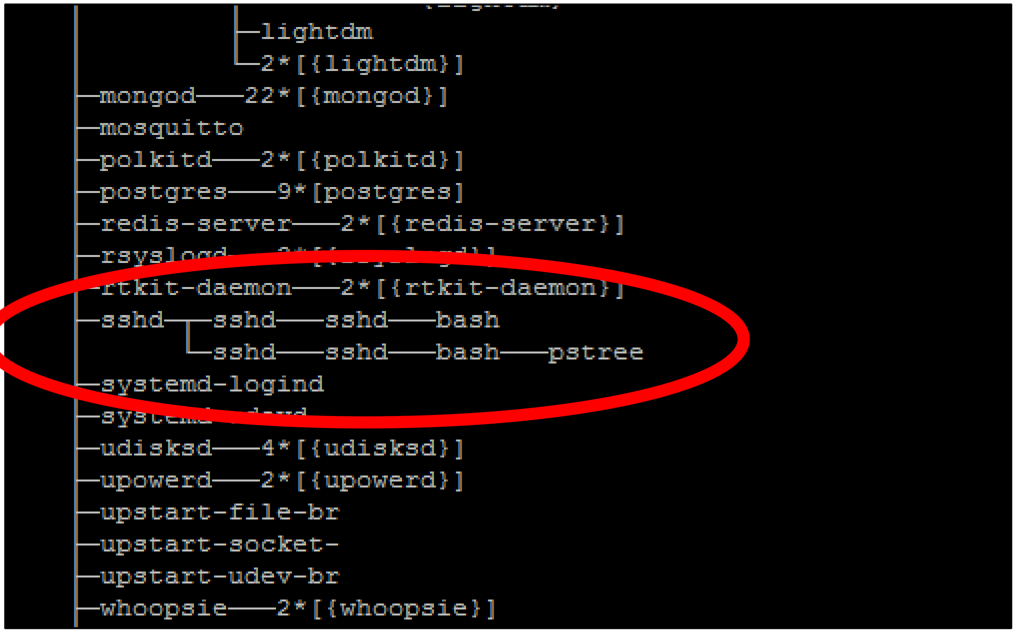


user@ubuntu: ~

```
user@ubuntu:~$ ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.5 119712  5732 ?        Ss   Aug07   0:20 /sbin/init nopr
root         2  0.0  0.0      0     0 ?        S    Aug07   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    Aug07   0:04 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   Aug07   0:00 [kworker/0:0H]
root         7  0.0  0.0      0     0 ?        S    Aug07   0:17 [rcu_sched]
root         8  0.0  0.0      0     0 ?        S    Aug07   0:00 [rcu_bh]
root         9  0.0  0.0      0     0 ?        S    Aug07   0:00 [migration/0]
root        10  0.0  0.0      0     0 ?        S    Aug07   0:16 [watchdog/0]
root        11  0.0  0.0      0     0 ?        S    Aug07   0:00 [kdevtmpfs]
root        12  0.0  0.0      0     0 ?        S<   Aug07   0:00 [netns]
root        13  0.0  0.0      0     0 ?        S<   Aug07   0:00 [perf]
root        14  0.0  0.0      0     0 ?        S    Aug07   0:04 [khungtaskd]
root        15  0.0  0.0      0     0 ?        S<   Aug07   0:00 [writeback]
root        16  0.0  0.0      0     0 ?        SN   Aug07   0:00 [ksmd]
root        17  0.0  0.0      0     0 ?        SN   Aug07   0:10 [khugepaged]
root        18  0.0  0.0      0     0 ?        S<   Aug07   0:00 [crypto]
root        19  0.0  0.0      0     0 ?        S<   Aug07   0:00 [kintegrityd]
root        20  0.0  0.0      0     0 ?        S<   Aug07   0:00 [bioset]
root        21  0.0  0.0      0     0 ?        S<   Aug07   0:00 [kblockd]
root        22  0.0  0.0      0     0 ?        S<   Aug07   0:00 [ata_sff]
root        23  0.0  0.0      0     0 ?        S<   Aug07   0:00 [md]
root        24  0.0  0.0      0     0 ?        S<   Aug07   0:00 [devfreq_wq]
```

pstree

- Displays a tree of processes (again, snapshot!)
- Visualise processes
- The list is not dynamic
  - Only reports processes when executed



```
 │            ├─lightdm
 │            └─2*[{lightdm}]
 ├─mongod───22*[{mongod}]
 ├─mosquitto
 ├─polkitd───2*[{polkitd}]
 ├─postgres───9*[postgres]
 ├─redis-server───2*[{redis-server}]
 ├─rsyslogd───3*[{rsyslogd}]
 ├─rtkit-daemon───2*[{rtkit-daemon}]
 ├─sshd─┬─sshd───sshd───bash
 │      └─sshd───sshd───bash───pstree
 ├─systemd-logind
 ├─systemd-udevd
 ├─udisksd───4*[{udisksd}]
 ├─upowerd───2*[{upowerd}]
 ├─upstart-file-br
 ├─upstart-socket-
 ├─upstart-udev-br
 ├─whoopsie───2*[{whoopsie}]
```

`pstree`: Useful arguments

```
pstree pid
```

– Display all processes from a Process ID (PID)

```
pstree student
```

– Display all processes from a specific user

```
pstree | less
pstree | more
```

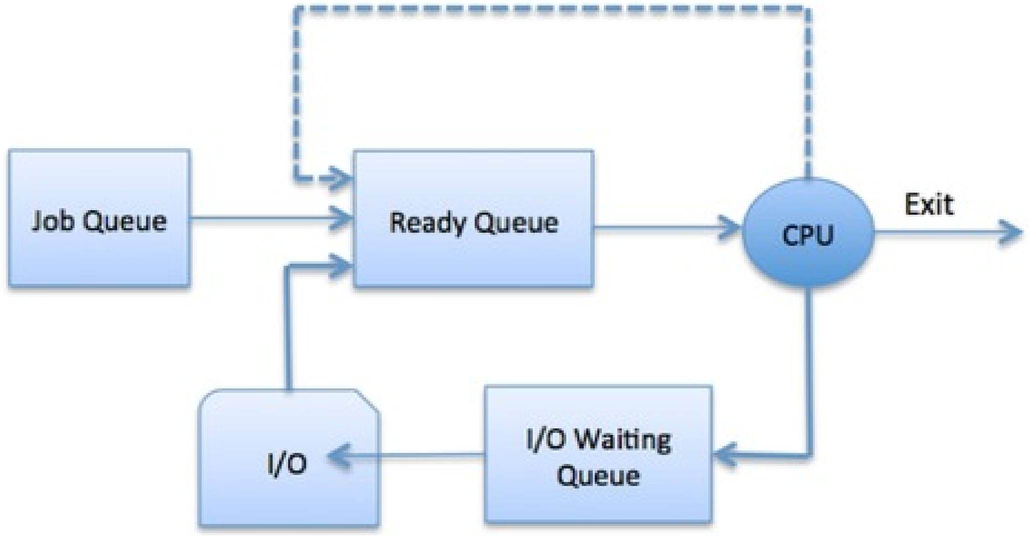
– Display the tree one page at a time

```
pstree -p
```

– Print the tree and include the PIDs

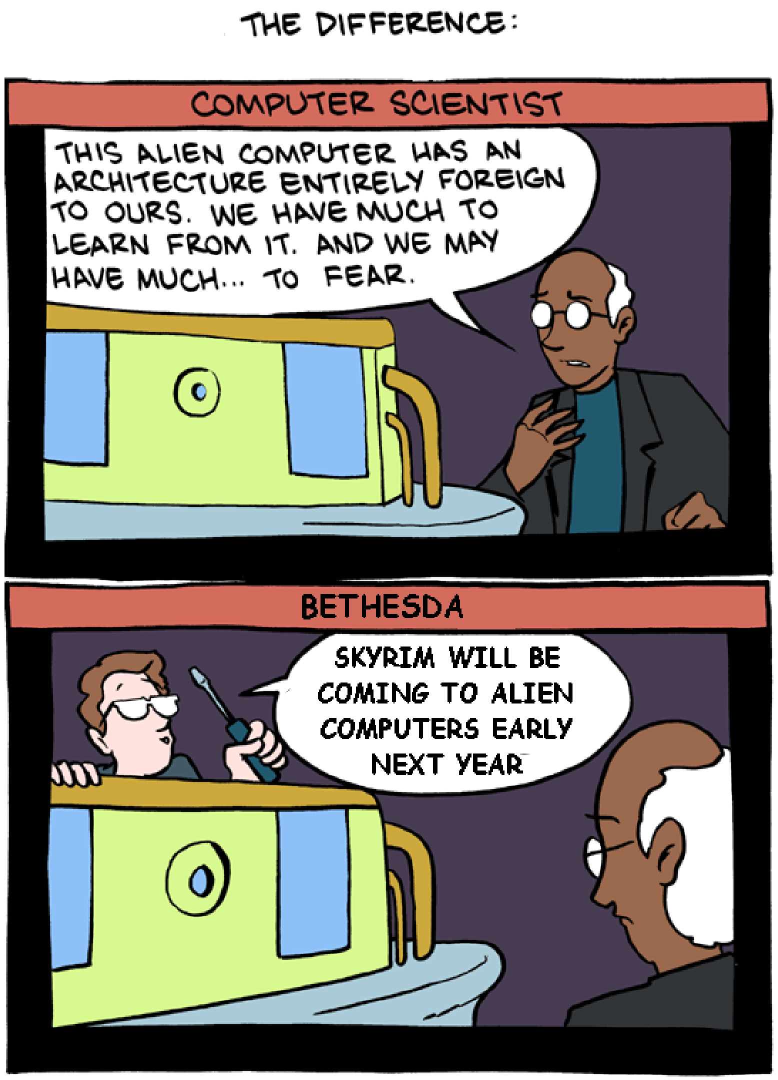
Process scheduling

- A manager that handles processes
  - Removes running processes
  - Selects another process to add
- Must have a strategy



Process Priorities

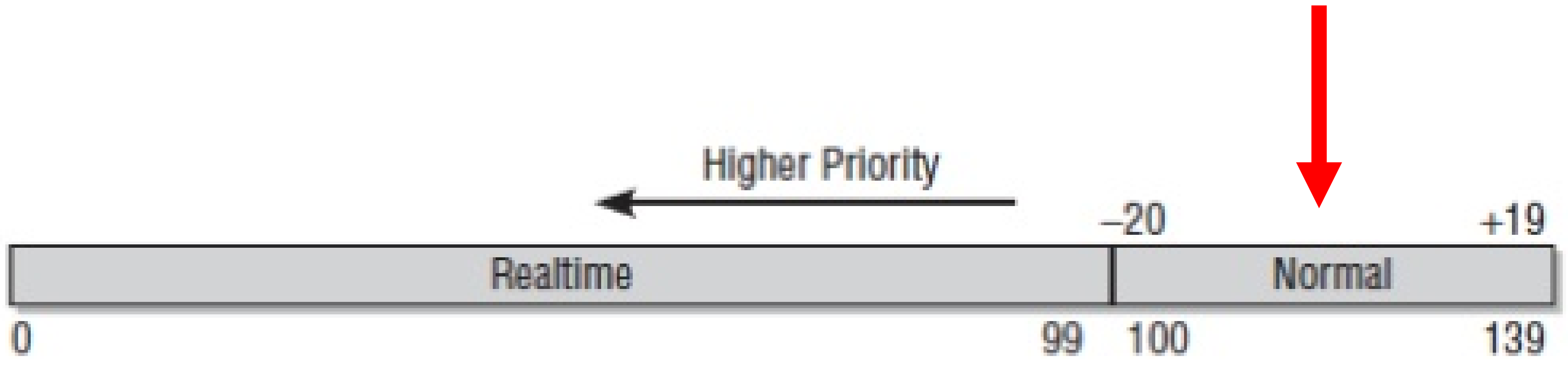
- Each process has a priority
  - How important is the process?
- Total priority range = 0 → 139
- Lower is higher?!?!
  - 0 is very important
  - 139 is not important at all
- 0 → 99 in the kernel
- 100 → 139 in user space


THE DIFFERENCE:
COMPUTER SCIENTIST
THIS ALIEN COMPUTER HAS AN ARCHITECTURE ENTIRELY FOREIGN TO OURS. WE HAVE MUCH TO LEARN FROM IT. AND WE MAY HAVE MUCH... TO FEAR.
BETHESDA
SKYRIM WILL BE COMING TO ALIEN COMPUTERS EARLY NEXT YEAR

Process Priorities (user space)

- 100 → 139 in user space
  - Translated to:
- 0 → 39

- Default priority in userspace is 20

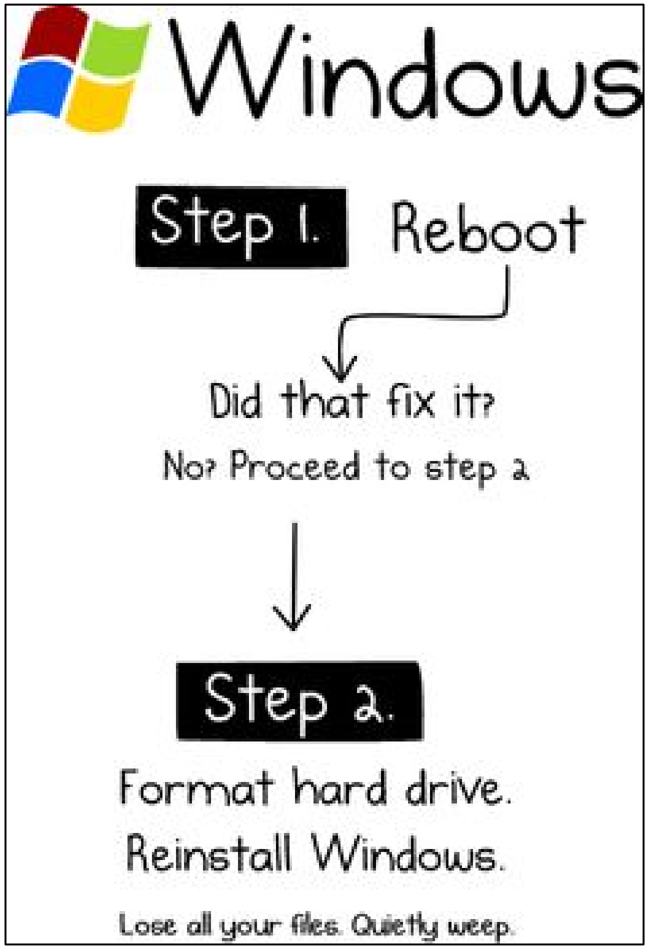


Nice Process Priorities

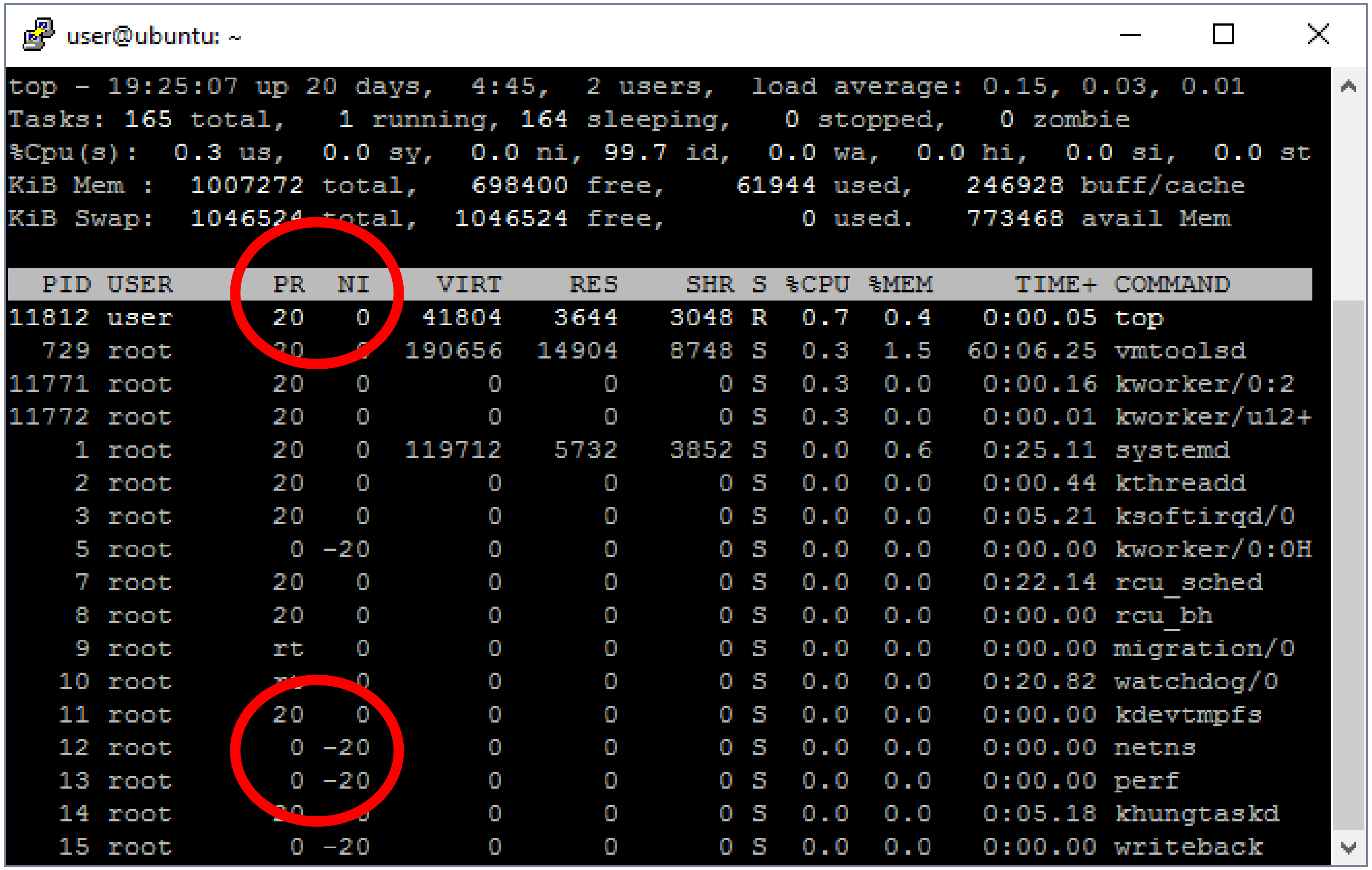
- Nice value?!
- Modify the priority of a process

- PR → Priority (default 20)
- NI → Nice value (-20 to +19)

- PR = Default Priority +/- Nice value



Process Priorities



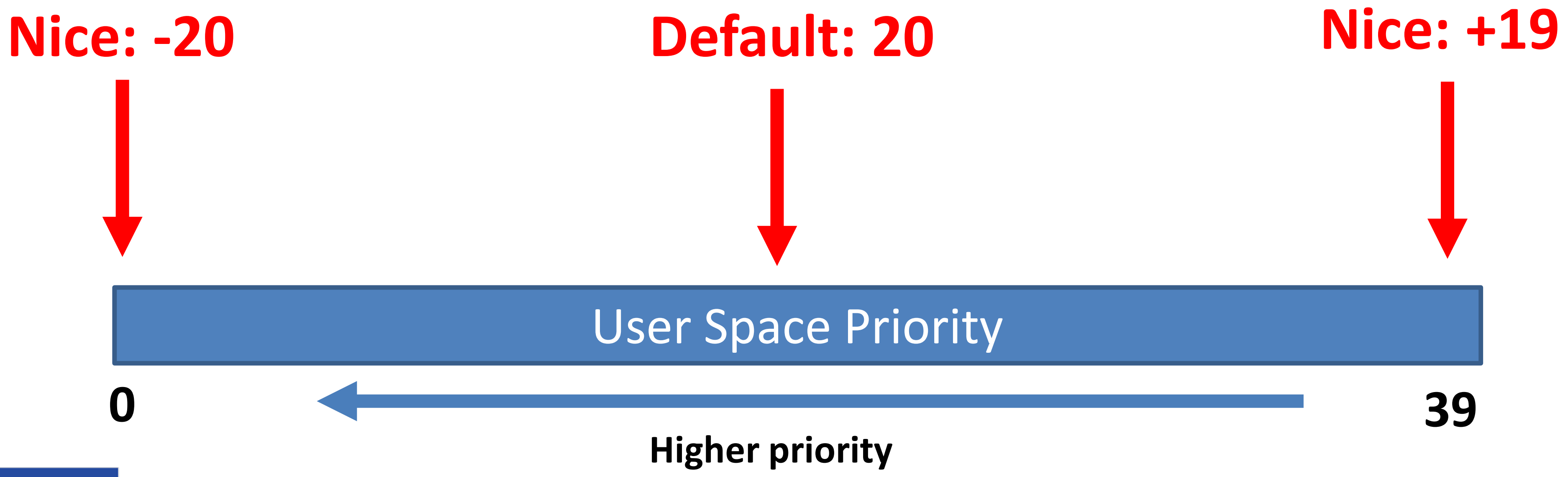
user@ubuntu: ~

```
top - 19:25:07 up 20 days,  4:45,  2 users,  load average: 0.15, 0.03, 0.01
Tasks: 165 total,   1 running, 164 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  0.0 sy,  0.0 ni, 99.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  1007272 total,   698400 free,    61944 used,   246928 buff/cache
KiB Swap:  1046524 total,  1046524 free,        0 used.   773468 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM     TIME+ COMMAND
11812 user      20   0   41804   3644   3048 R  0.7  0.4   0:00.05 top
  729 root      20   0  190656  14904   8748 S  0.3  1.5  60:06.25 vmtoolsd
11771 root      20   0       0      0      0 S  0.3  0.0   0:00.16 kworker/0:2
11772 root      20   0       0      0      0 S  0.3  0.0   0:00.01 kworker/u12+
    1 root      20   0  119712   5732   3852 S  0.0  0.6   0:25.11 systemd
    2 root      20   0       0      0      0 S  0.0  0.0   0:00.44 kthreadd
    3 root      20   0       0      0      0 S  0.0  0.0   0:05.21 ksoftirqd/0
    5 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H
    7 root      20   0       0      0      0 S  0.0  0.0   0:22.14 rcu_sched
    8 root      20   0       0      0      0 S  0.0  0.0   0:00.00 rcu_bh
    9 root      rt   0       0      0      0 S  0.0  0.0   0:00.00 migration/0
   10 root      rt   0       0      0      0 S  0.0  0.0   0:20.82 watchdog/0
   11 root      20   0       0      0      0 S  0.0  0.0   0:00.00 kdevtmpfs
   12 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 netns
   13 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 perf
   14 root      20   0       0      0      0 S  0.0  0.0   0:05.18 khungtaskd
   15 root       0 -20       0      0      0 S  0.0  0.0   0:00.00 writeback
```

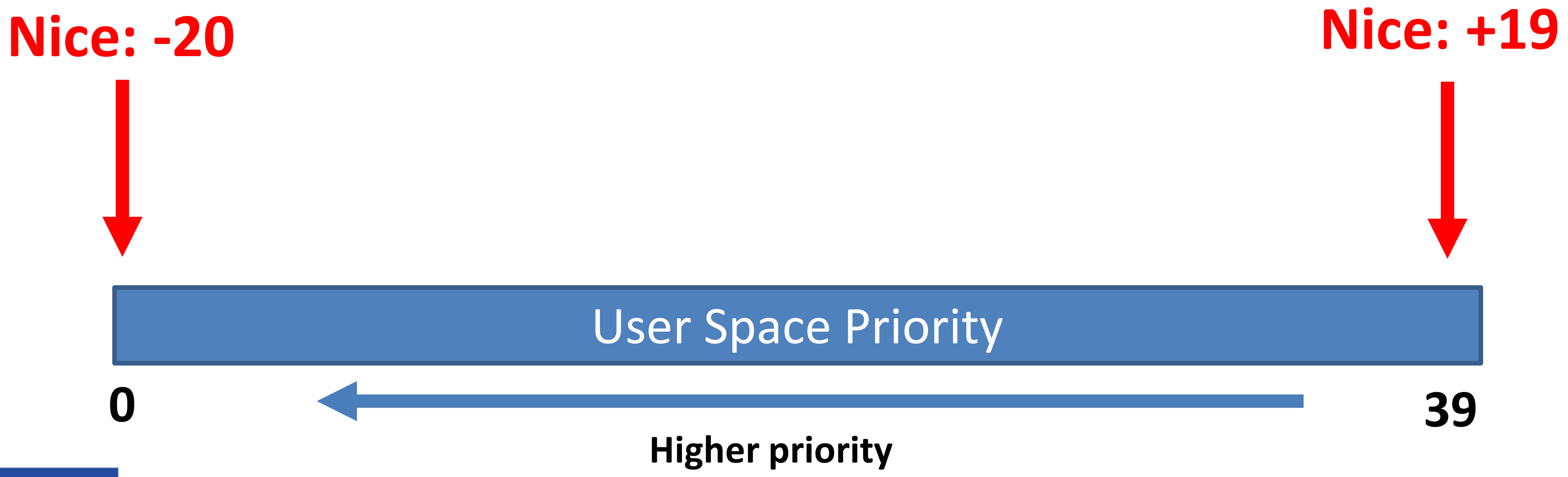
Nice Process Priorities

- PR = Default Priority +/- Nice value
- PR = 20 – 20
  - 0 has a higher priority
- PR = 20 + 19
  - 39 has a lower priority



Nice Process Priorities

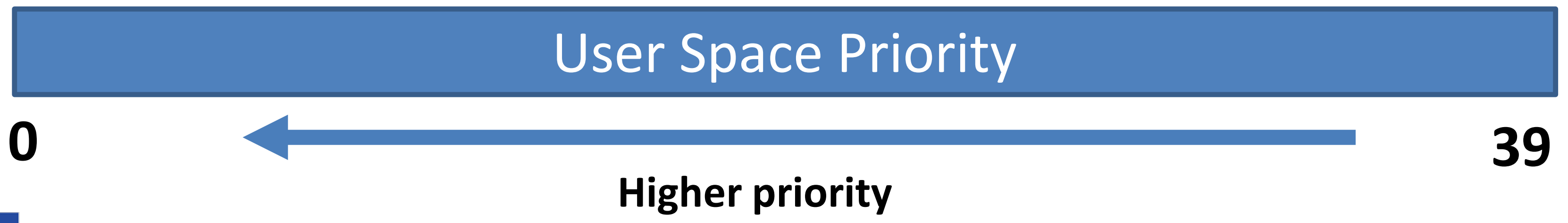
- PR = Default Priority +/- Nice value
- PR = 20 – 20
  - 0 has a higher priority
- PR = 20 + 19
  - 39 has a lower priority



The `nice` command

- `nice -n <nice-value> <command>`
- `nice -n -20 ./script.sh`
- `nice -n 19 ./calc_pi.sh`

- OK, that's nice…
- But how can we adjust when a command is running?

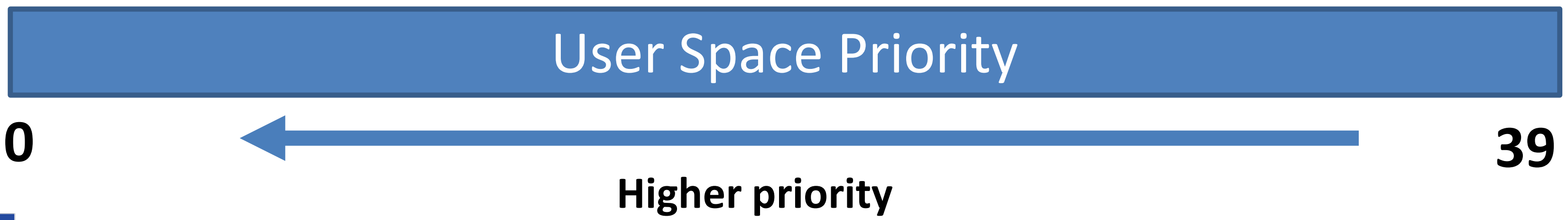


The **renice** command

- **`renice -n <nice value> <PID>`**

- **`ps –a; top`** (to get PID)
- **`renice -n 19 3849`**

- FYI – users can only lower priorities!
- **sudo** is needed to increase priority



Lab-09-1 – Start

- **TOPICS:**
- **Test out process tools**
- **Create a script to learn about processes**
- **Also, running processes in the background**

SOFTWARE DEVELOPMENT PROCESS

0. I can't fix this
1. Crisis of confidence
2. Questions career
3. Questions life
4. Oh it was a typo, cool

TOPIC:

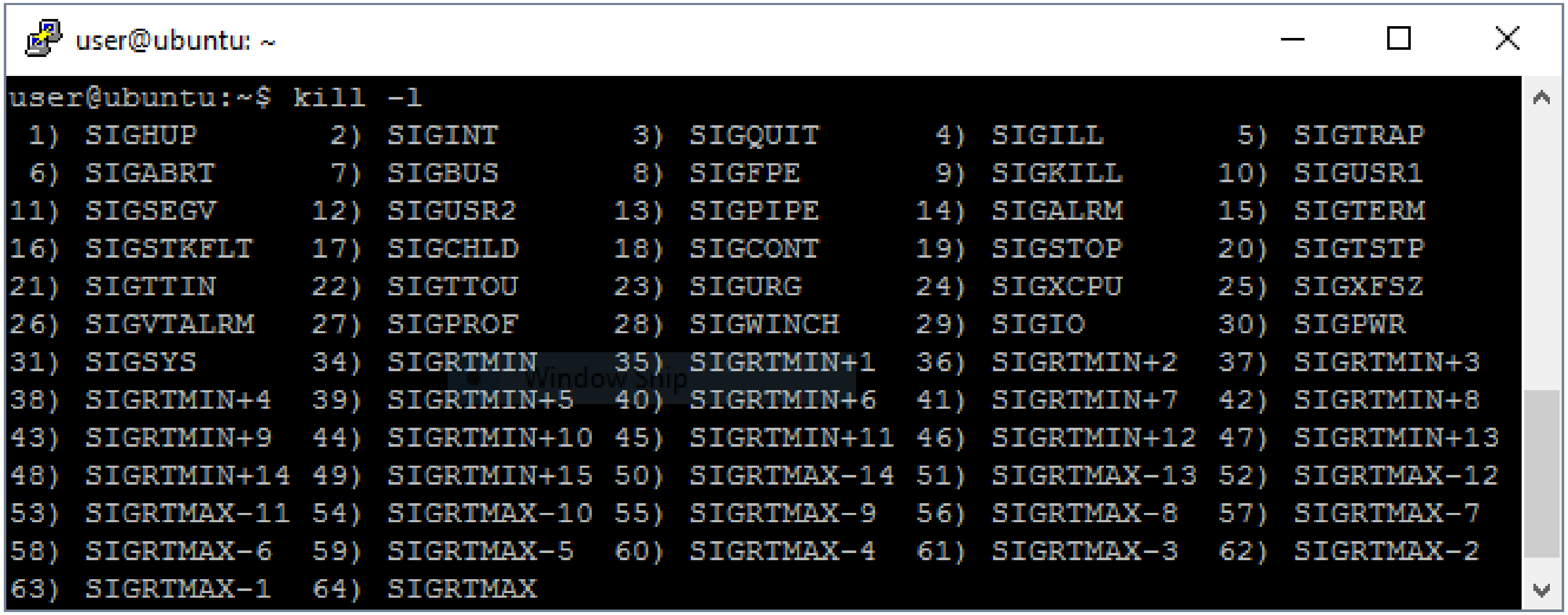
Process Communication and Signals

Process Communication

- Process communication relies on ***signals***
- Signals are used to notify a process of an event
- Therefore, processes can detect signals
- Processes can also send signals
  - Self-termination
  - Clean-up and exit

Signals in Linux

- Total of 64 different signals, not many are useful
  - Output of: **`kill -l`**

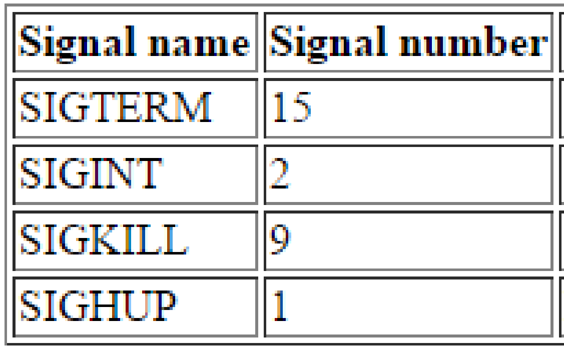


user@ubuntu: ~

```
user@ubuntu:~$ kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL       5) SIGTRAP
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) SIGKILL     10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30) SIGPWR
31) SIGSYS      34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
```

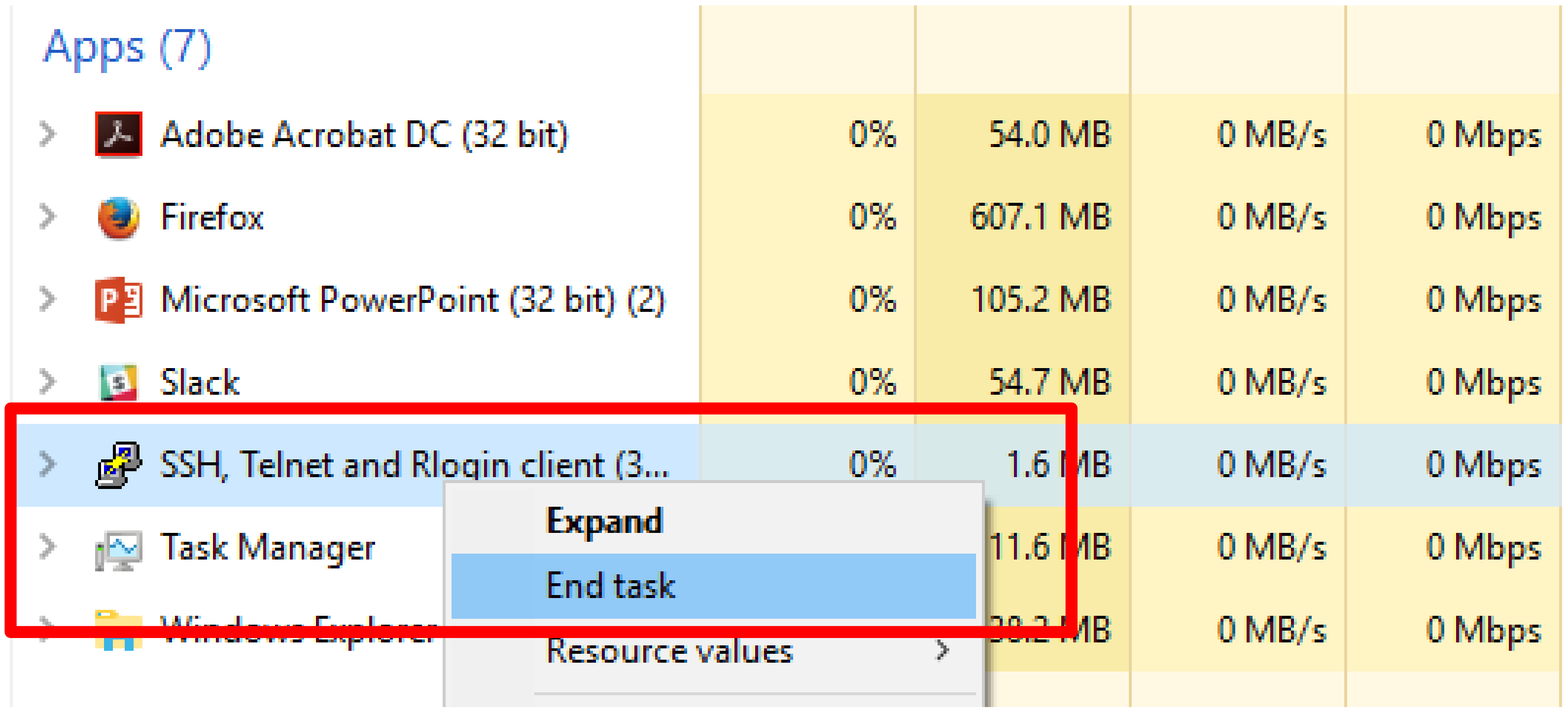
4 useful Signals

- SIGTERM (15)
  - Terminate a process and be nice about it!
  - Terminate in a controlled manner (close resources)
  - Process can ignore the interrupt
- SIGINT (2)
  - Interrupt a process → Same as `Ctrl + C`
  - Process can ignore the interrupt
- SIGKILL (9)
  - Terminate a process and be exceptionally forceful about it!
  - Use as a last option
- SIGHUP (1)
  - Signal hangup – mainly used for serial connection (historical)
  - Now used for pseudo/virtual terminals (e.g., PuTTY)
  - Daemons use SIGHUP to restart (e.g., re-read the configuration file)



Signal name	Signal number
SIGTERM	15
SIGINT	2
SIGKILL	9
SIGHUP	1

Killing processes



Apps (7)

Adobe Acrobat DC (32 bit)	0%	54.0 MB	0 MB/s	0 Mbps
Firefox	0%	607.1 MB	0 MB/s	0 Mbps
Microsoft PowerPoint (32 bit) (2)	0%	105.2 MB	0 MB/s	0 Mbps
Slack	0%	54.7 MB	0 MB/s	0 Mbps
SSH, Telnet and Rlogin client (3...	0%	1.6 MB	0 MB/s	0 Mbps
Task Manager		11.6 MB	0 MB/s	0 Mbps
Windows Explorer		30.2 MB	0 MB/s	0 Mbps

The `kill` command

- The **`kill`** command
  - Send a signal to a process

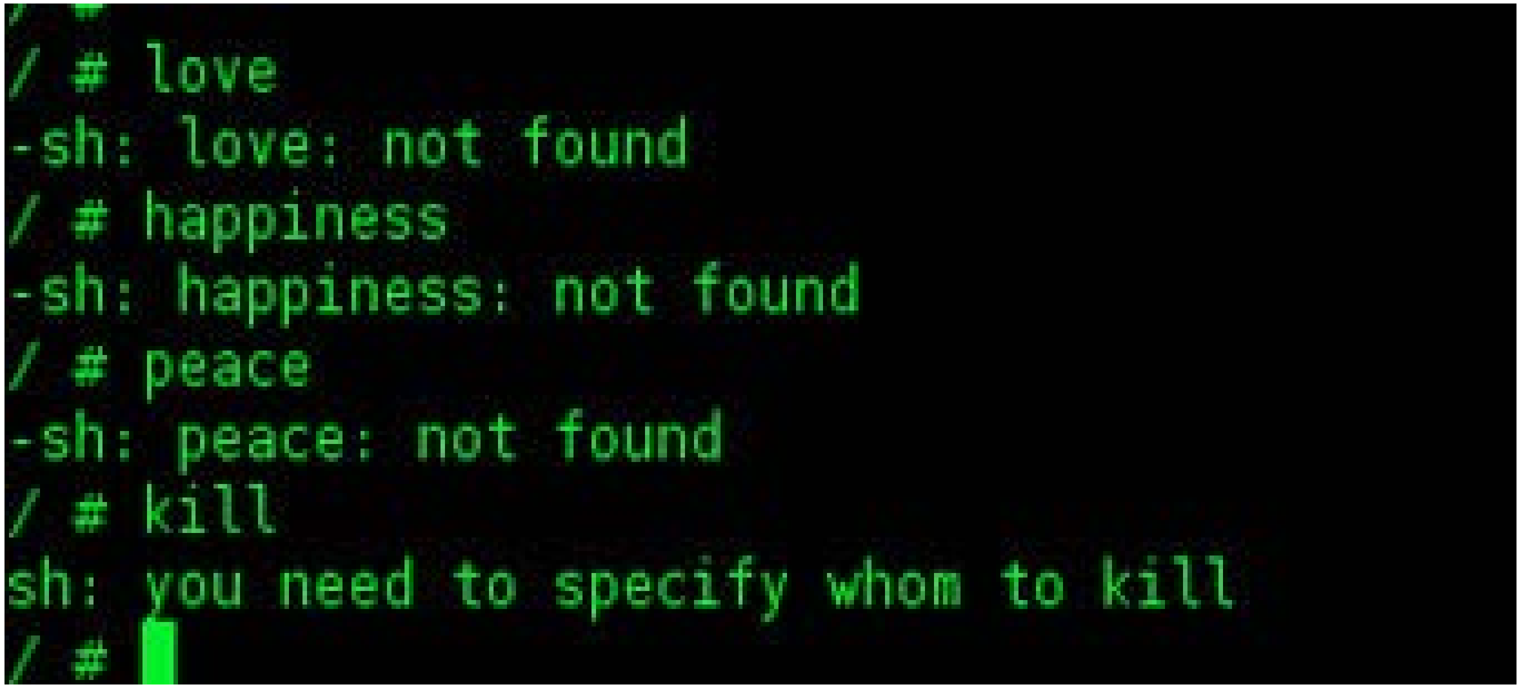
- Command syntax:

```
kill <PID>
```

- Generally requires process ID (PID)

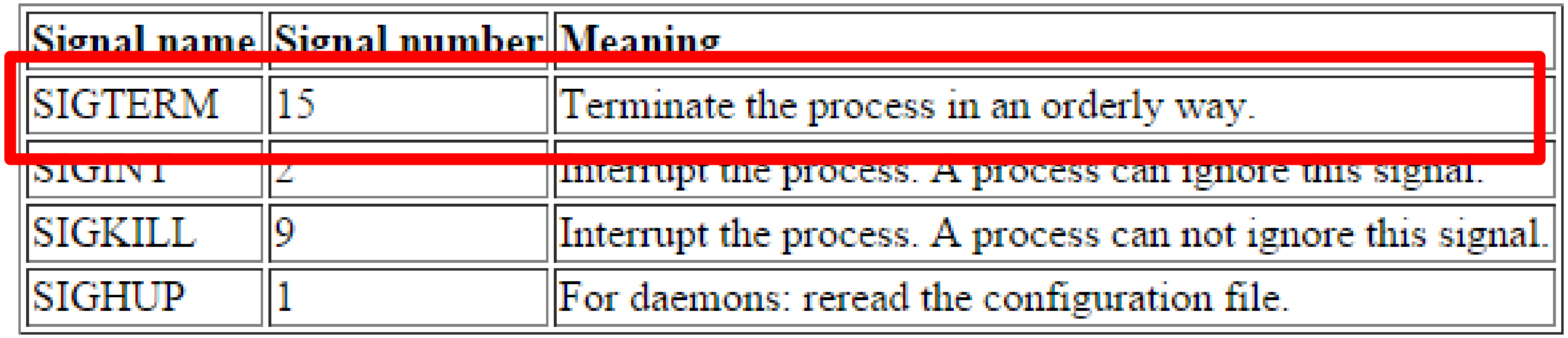
- ***How can we find the PID?***

Only Linux Things



```
/ # love
-sh: love: not found
/ # happiness
-sh: happiness: not found
/ # peace
-sh: peace: not found
/ # kill
sh: you need to specify whom to kill
/ #
```

`kill`: default signal



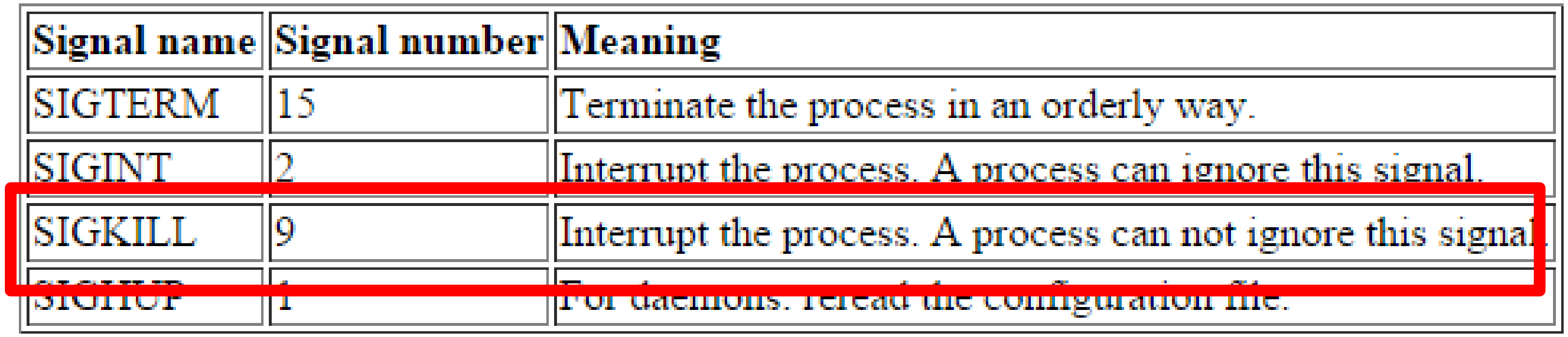
Signal name	Signal number	Meaning
SIGTERM	15	Terminate the process in an orderly way.
SIGINT	2	Interrupt the process. A process can ignore this signal.
SIGKILL	9	Interrupt the process. A process can not ignore this signal.
SIGHUP	1	For daemons: reread the configuration file.

- By default, the **`kill`** command uses **SIGTERM**
  - Terminate a process and be nice about it!
  - For example, try to save a file before exiting

- To use a different signal, you have to specify it

```
kill -s <signal_name> <PID>
kill -s <signal_number> <PID>
```

kill: specifying a signal



Signal name	Signal number	Meaning
SIGTERM	15	Terminate the process in an orderly way.
SIGINT	2	Interrupt the process. A process can ignore this signal.
SIGKILL	9	Interrupt the process. A process can not ignore this signal.
SIGHUP	1	For daemons: reread the configuration file.

- Specifying to use SIGKILL:

```
kill -s KILL <PID>
kill -s SIGKILL <PID>
kill -s 9 <PID>
```

`kill`: killing processes by PID

`kill <PID>`

- How to find the PID?
  - **`ps, top, pstree -p`**

- Example syntax:
  - **`ps -e`** (manually find)
  - **`top`** (manually find or sort)
  - **`ps -e | grep <process_name>`**
  - **`pgrep <process_name>`**
  - **`pidof <process_name>`**

`kill`: killing processes by name

```
killall <process_name>
pkill <process_name>
```

- The killall and pkill commands takes a process name argument:
  - `killall vi`
  - `killall sshd`
  - `pkill vi`

OTAGO POLYTECHNIC Te Kura Matatini ki Otago

Permissions and Processes

- Users can kill their own processes
  - Anything on **ps** output
  - But not everything on **ps –e** output

- Users <u>cannot</u> kill other users processes
- Users <u>cannot</u> kill system processes
  - Usually owned by root user

- As usual, *superuser* can do anything!

Managing System Shutdown/Reboot

- Essential command for shutdown/reboot:
  - **`shutdown`**

- Various parameters exist:

  `-H` → Halt the system

  `-P` → Power off the system

  `-r` → Reboot the system

  `-c` → Cancel the shutdown

```
shutdown <options> <time> <wall>
```

`shutdown`: Examples

- `shutdown <options> <time> <wall>`
  - General command syntax
- `shutdown -P +5 “Shutdown immanent!”`
  - Shutdown in 5 minutes seconds with a message
- `shutdown -r now`
  - Restart the system now
- `shutdown -c`
  - Cancel any pending shutdowns

`shutdown`: permissions & aliases

- Shutdown requires *superuser* privledge

```
sudo shutdown
```

- Various other related commands exist
- Not specifically aliases, but kind of…

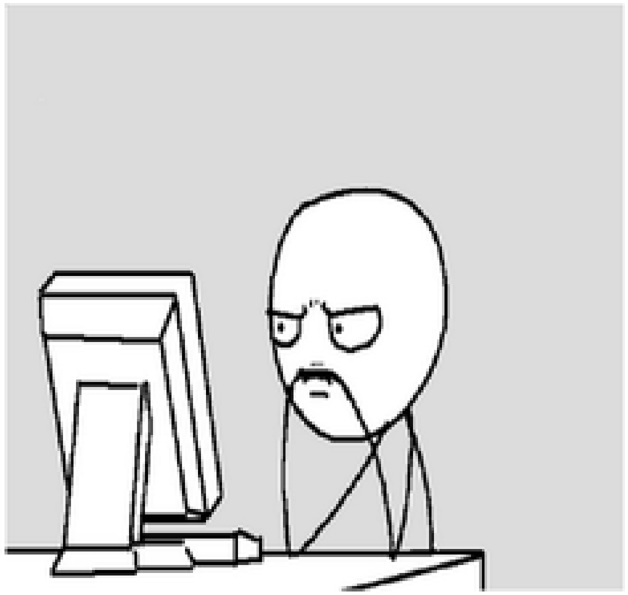
`poweroff`	→ Alias for **`shutdown –p now`**
`reboot`	→ Alias for **`shutdown -r`**

```
sudo poweroff
sudo reboot
```

`shutdown`: permissions & aliases

Differences between:

normaluser@linux:~$



root@linux:~#

Lab-09-1 – Continue

- **TOPICS:**
- **Killing processes**
- **Working with signals**
- **System runtime (shutdown)**

SOFTWARE DEVELOPMENT PROCESS

0. I can't fix this
1. Crisis of confidence
2. Questions career
3. Questions life
4. Oh it was a typo, cool