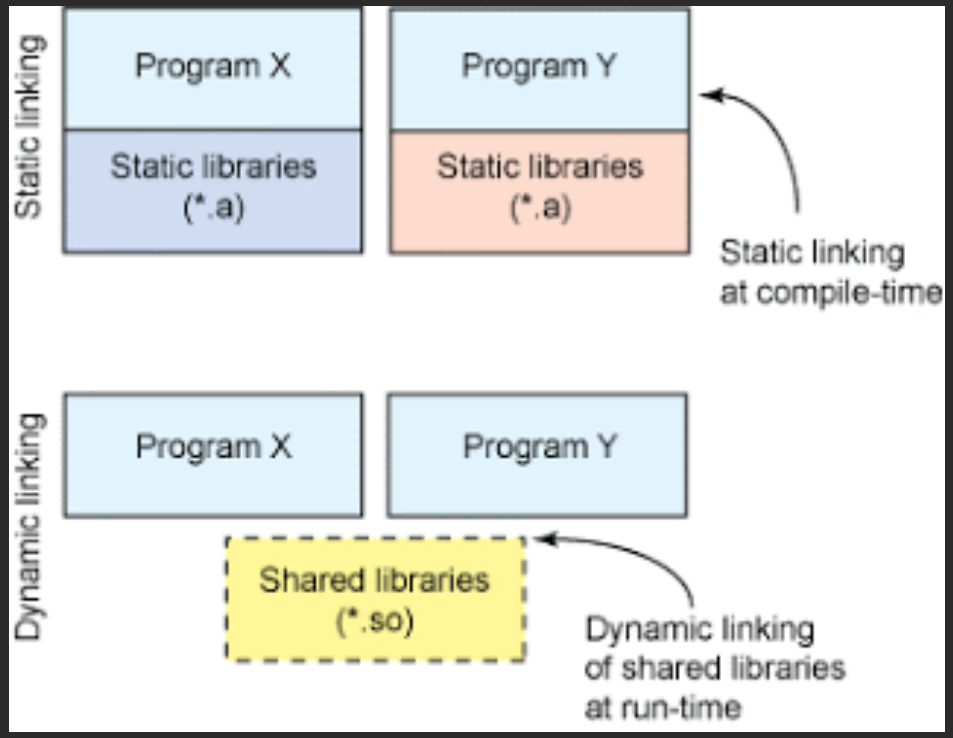
# Lesson 3: Managing Software and Processes

# Objectives covered

- 102.5 Use RPM and YUM package management (weight: 3)
- 102.4 Manage Debian package management (weight: 3)
- **102.3 Manage shared libraries (weight: 1)**
- **103.5 Create, monitor, and kill process (weight: 4)**
- **103.6 Modify process execution priorities (weight: 2)**

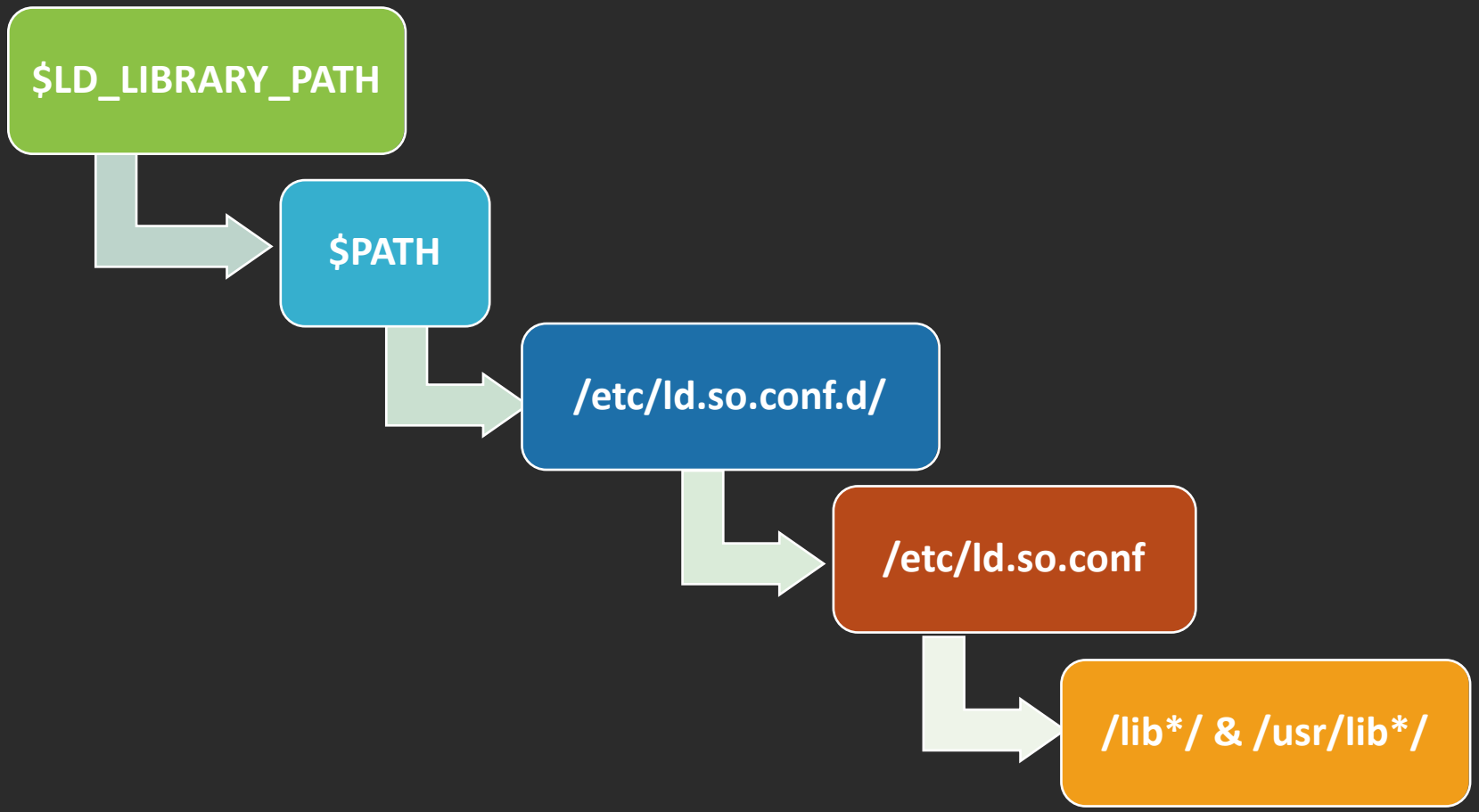
# Manage shared libraries

# Library types

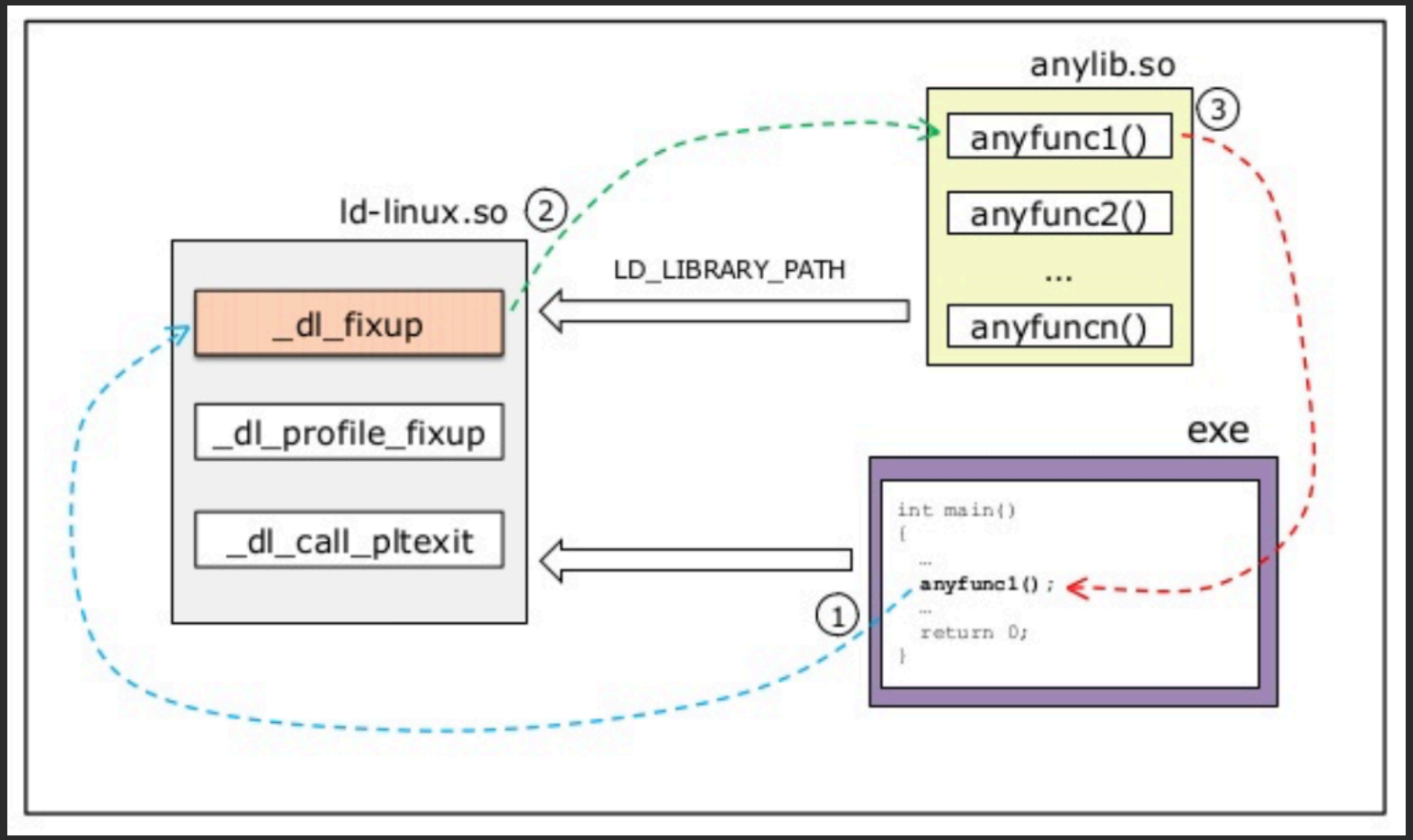



Shared library naming convention: libLIBRARYNAME.so.VERSION

# Library files location



# Dynamic linker/loader (ld-linux.so or ld.so)



# Working with Library cache (/etc/ld.so.cache)

```
# ldconfig -v | head
/usr/lib/mesa:
    libGL.so.1 -> libGL.so.1.2
/usr/lib/i686-linux-gnu:
    liblouis.so.2 -> liblouis.so.2.2.0
/usr/lib/alsa-lib:
    libasound_module_ctl_oss.so -> libasound_module_ctl_oss.so
    libasound_module_ctl_bluetooth.so -> libasound_module_ctl_bluetooth.so
    libasound_module_pcm_bluetooth.so -> libasound_module_pcm_bluetooth.so
    libasound_module_pcm_vdownmix.so -> libasound_module_pcm_vdownmix.so
    libasound_module_rate_speexrate.so -> libasound_module_rate_speexrate_me
```

```
# ldconfig -p | head -5
916 libs found in cache `/etc/ld.so.cache'
    libzephyr.so.4 (libc6) => /usr/lib/libzephyr.so.4
    libzbar.so.0 (libc6) => /usr/lib/libzbar.so.0
    libz.so.1 (libc6) => /lib/libz.so.1
    libz.so (libc6) => /usr/lib/libz.so
```

```
# ldconfig -n /opt/dummy/lib
```

```
# vi /etc/ld.so.conf

# ldconfig
```

# Library dependencies

```
$ ldd /usr/bin/echo
        linux-vdso.so.1 =>  (0x00007ffd3bd64000)
        libc.so.6 => /lib64/libc.so.6 (0x00007f7c39eff000)
        /lib64/ld-linux-x86-64.so.2 (0x00007f7c3a2cc000)
```

# Create, monitor and kill process

# Viewing processes

```
$ ps -ef
UID          PID  PPID  C STIME TTY          TIME CMD
root           1     0  0 10:18 ?        00:00:03 /sbin/init splash
root           2     0  0 10:18 ?        00:00:00 [kthreadd]
root           4     2  0 10:18 ?        00:00:00 [kworker/0:0H]
```

- **UID:** The user responsible for running the process
- **PID:** The process ID of the process
- **PPID:** The process ID of the parent process (if the process was started by another process)
- **C:** The processor utilization over the lifetime of the process
- **STIME:** The system time when the process was started
- **TTY:** The terminal device from which the process was started
- **TIME:** The cumulative CPU time required to run the process
- **CMD:** The name of the program that was started in the process

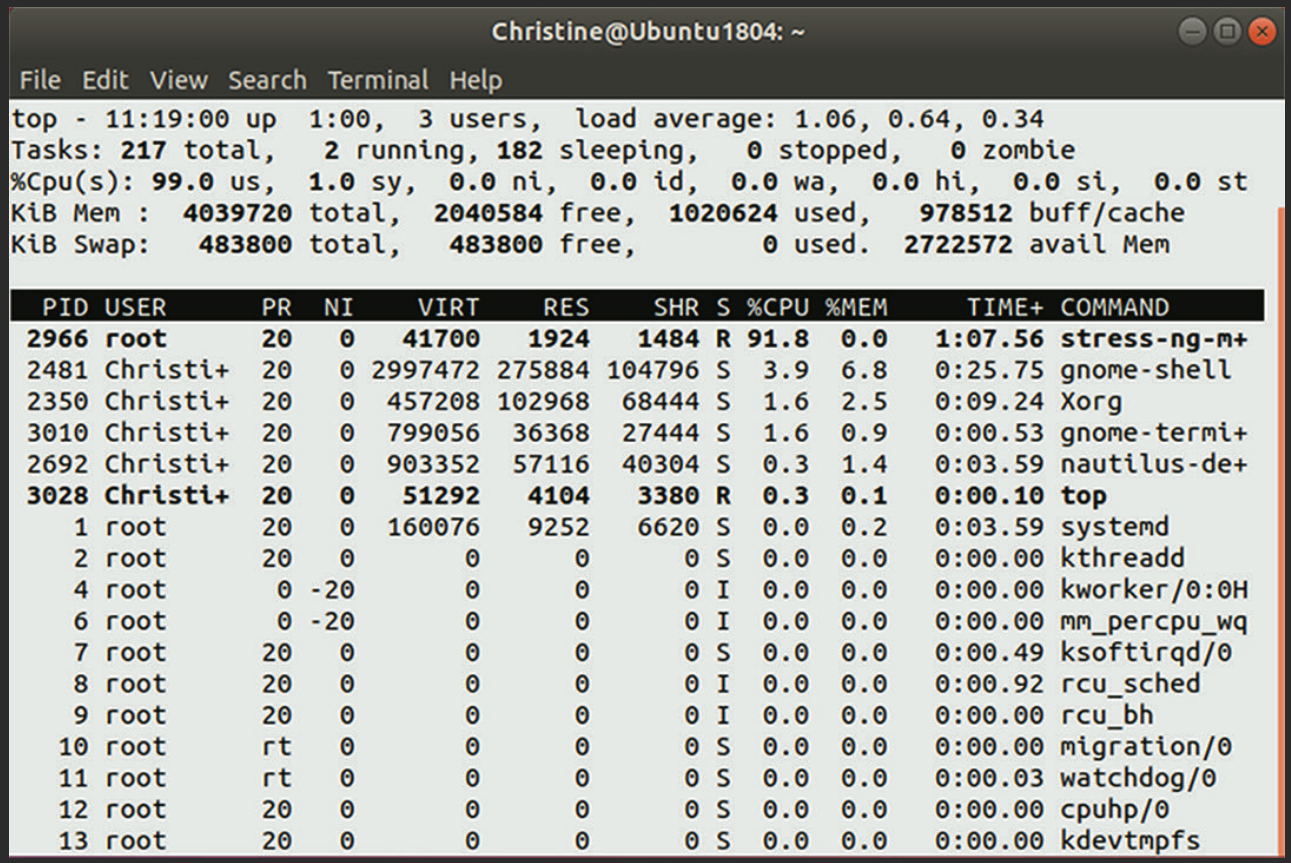
# Viewing processes

## ps [option]

| Option(s) | Description |
|---|---|
| a | Display every process on the system associated with a tty terminal |
| -A, -e | Display every process on the system |
| -C *CommandList* | Only display processes running a command in the *CommandList* |
| -g *GIDList*, or -group *GIDList* | Only display processes whose current effective group is in *GIDList* |
| -G *GIDList*, or -Group *GIDList* | Only display processes whose current real group is in *GIDList* |
| -N | Display every process except selected processes |
| p *PIDList*, -p *PIDList* or --pid *PIDList* | Only display *PIDList* processes |
| -r | Only display selected processes that are in a state of running |
| -t *ttyList*, or --tty *ttyList* | List every process associated with the *ttyList* terminals |
| -T | List every process associated with the current tty terminal |
| -u *UserList*, or --user *UserList* | Only display processes whose effective user (username or UID) is in *UserList* |
| -U *UserList*, or --User *UserList* | Only display processes whose real user (username or UID) is in *UserList* |
| x | Remove restriction of "associated with a tty terminal"; typically used with the a option |

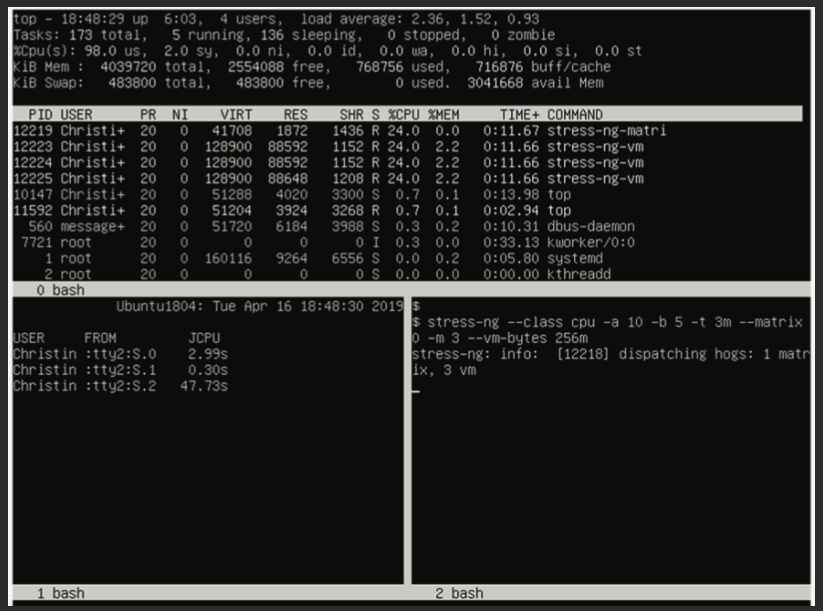
# Viewing processes

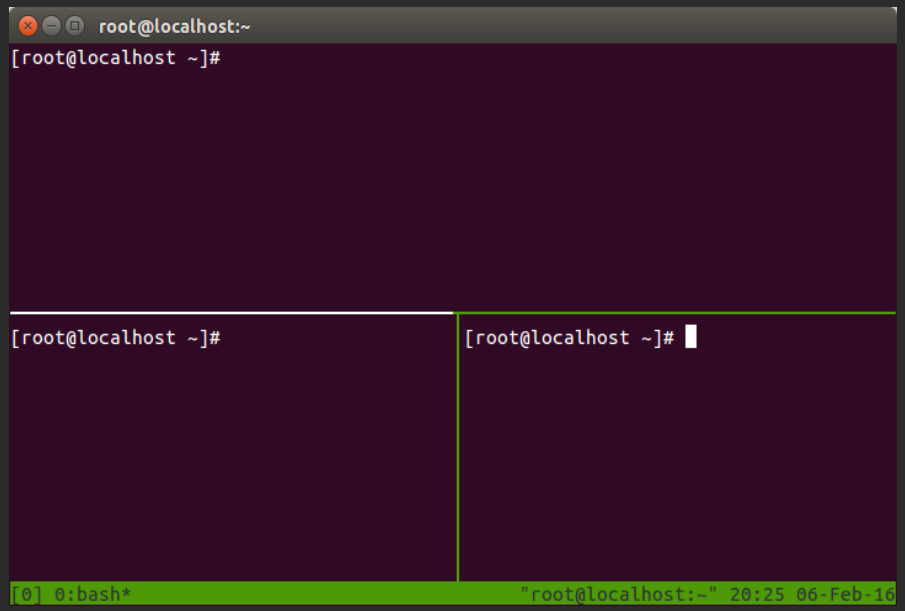
## top



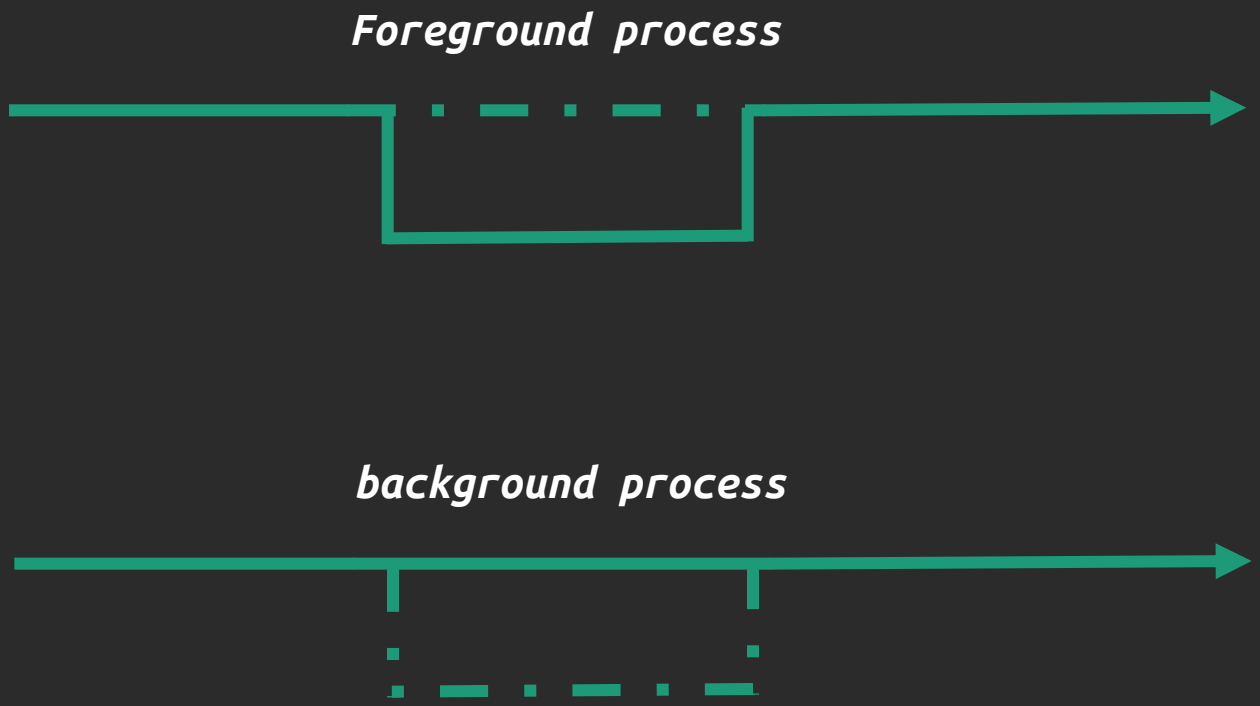
# Employing muliple screen

## screen & tmux



# Background and Foreground processes



## Manage background processes

# jobs [option] [job-id]

```
$ sleep 3000 &
[1] 1539
$
$ jobs
[1]+  Running                 sleep 3000 &
$
$ jobs -l
[1]+  1539 Running                 sleep 3000 &
$
```

# Bring jobs to foreground and vice versa

## Bring jobs to foreground

```
$ jobs -l
[1]-  1539 Running                 sleep 3000 &
[2]+  1540 Running                 bash CriticalBackups.sh &
$
$ fg %2
bash CriticalBackups.sh
```

## Send running program to background

```
$ bash CriticalBackups.sh
^Z
[2]+  Stopped                 bash CriticalBackups.sh
$
$ bg %2
[2]+ bash CriticalBackups.sh &
$
$ jobs -l
[1]-  1539 Running                 sleep 3000 &
[2]+  1540 Running                 bash CriticalBackups.sh &
$
```

# Process signals

| Number | Name | Description |
|---|---|---|
| 1 | HUP | Hangs up |
| 2 | INT | Interrupts |
| 3 | QUIT | Stops running |
| 9 | KILL | Unconditionally terminates |
| 11 | SEGV | Segments violation |
| 15 | TERM | Terminates if possible |
| 17 | STOP | Stops unconditionally, but doesn't terminate |
| 18 | TSTP | Stops or pauses, but continues to run in background |
| 19 | CONT | Resumes execution after STOP or TSTP |

# Sending signals to process

## kill [option] process

```
 $ ps 2285
   PID TTY      STAT   TIME COMMAND
  2285 pts/0    S      0:00 bash SecurityAudit.sh
 $
 $ kill 2285
 [1]+  Terminated              bash SecurityAudit.sh
 $
 $ ps 2285
   PID TTY      STAT   TIME COMMAND
 $
$ kill -s HUP  2305
$
$ ps 2305
  PID TTY      STAT   TIME COMMAND
 2305 pts/0    T      0:00 vi
$
$ kill -9  2305
[1]+  Killed                  vi
$
$ ps 2305
  PID TTY      STAT   TIME COMMAND
$
```

*Default signal is TERM*

# Sending signals to process

# killall, pkill & pgrep

```
$ ps
  PID TTY          TIME CMD
 1441 pts/0    00:00:00 bash
 1504 pts/0    00:00:00 stressor.sh
 1505 pts/0    00:00:00 stress-ng
 1506 pts/0    00:00:05 stress-ng-matri
 1507 pts/0    00:00:00 stressor.sh
 1508 pts/0    00:00:00 stress-ng
 1509 pts/0    00:00:02 stress-ng-matri
 1510 pts/0    00:00:00 stressor.sh
[…]
 1517 pts/0    00:00:00 ps
$
$ killall stress-ng
[…]
$
$ ps
  PID TTY          TIME CMD
 1441 pts/0    00:00:00 bash
 1519 pts/0    00:00:00 ps
```

```
$ pgrep -t tty3
1716
1804
1828
1829
1831
1832
1836
1837
1838
1839
1840
$
```

```
$ ps 1840
  PID TTY      STAT   TIME COMMAND
 1840 tty3     R+     0:39 stress-ng --class cpu -a 10 -b 5 -t 5m --matrix 0
$
$ sudo pkill -t tty3
$
$ pgrep -t tty3
1846
$
$ ps 1846
  PID TTY      STAT   TIME COMMAND
 1846 tty3     Ss+    0:00 /sbin/agetty -o -p -- \u --noclear tty3 linux
$
```

## Keep job running after log-out

# nohup program/script &

```
$ nohup bash CriticalBackups.sh &
[1] 2090
$ nohup: ignoring input and appending output to 'nohup.out'
```

# Modify process execution priorities

## Manage process priorities

# nice -n VALUE Command

```
 $ nice -n 10 bash CriticalBackups.sh
$ ps -l 1949
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
0 S  1001  1949  1527  0  90  10 -  4998 wait   pts/1      0:00 bash CriticalBac
$
```

*nice value in scale of -20 → 19 (with higher priority on the left)*

# *Manage process priorities*

## *renice [-n] VALUE [PID] [-u Users] [-g Groups]*

```
$ renice 15 -p 1949
1949 (process ID) old priority 10, new priority 15
$
$ sudo renice -n -5 -p 1949
1949 (process ID) old priority 15, new priority -5
$
$ sudo renice -10 -p 1949
1949 (process ID) old priority -5, new priority -10
$
```

# Question...