GETTING STARTED
WITH
PROCESS MANAGEMENT

➢ PROCESS:

- A process is a unit for provisioning system resources. It is any program, application or command that runs on the system.
- A process is simply instance of a running program
- A process is created in memory in its own address space when a command, program or application is initiated.
- Each process has a parent process. A single parent process may have one or more child process and processes many of its attributes to them when they are created.
- Each process is assigned a unique identification number known as PID (Process Identifier), which is used by the kernel to manage and control the process through its lifecycle.

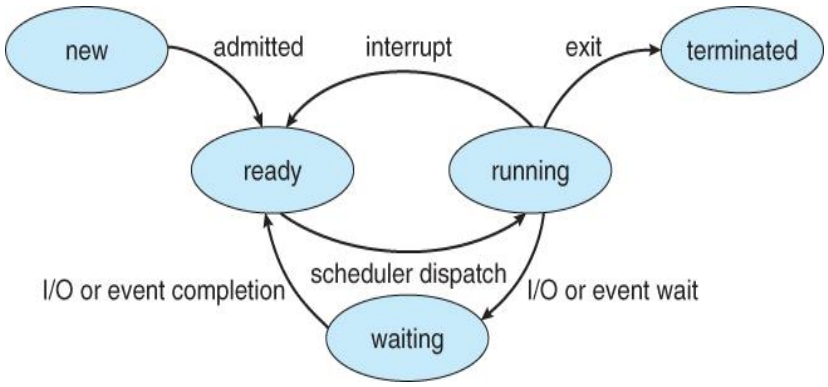
STATES OF A PROCESS:

RUNNING : The process is either running or it is ready to run.

WAITING : The process is waiting for an event or for a resource.

STOPPED : The process has been stopped, usually by receiving a signal.

ZOMBIE : The process is dead but have not been removed from the process table. It is also known as "**defunct process".**



JOB TYPES:

FOREGROUND:

- Every process when started runs in foreground by default.
- A user can be done only one process at a time.

→ To run a job:

$firefox

→ To cancel foreground jobs:

$ctrl + c

BACKGROUND:

- A user can be done in parallel with the process running in background since they do not have to wait for the previous process to be completed.
- Adding **"&"** along with the command starts it as a background process

→ To run a background job:

$firefox &

→ To list running background jobs:

$jobs

→ To list a process:

$ps

$top

→ Listing a specific process:

$pidof cron

$pgrep cron

→ To see every process on the system:

$ps -ef

$ps -aux

$ps -aef | grep cron

→ View processes by User and Group ownership:

$ps -U raju

$ps -G developers

UNDERSTANDING PROCESS NICENESS:

- Each process has an assigned priority, which is established at initiation. It is based on a numeric value called niceness (or a nice value).
- There are 40 niceness values, **with -20 being the highest** and **+19 being the lowest value.**
- Most processes started by the system use the default niceness of 0.
- A process running at higher priority gets more CPU attention.
- A child process inherits the niceness of its parent or calling process.
- Normally, we run programs at the default niceness but we can change it based on system load and urgency.

$nice (0 is default)

$ps -l

→ Priority is calculated based on the niceness value, column 8 - NI.

$ps -efl

→ To run the top command at a **lower priority** with nice value of +2:

$nice -2 top

From another terminal you can check with:

$ps -el |grep top

→ To run the top command at a **highest priority** with nice value of -10.

$nice --10 top

$ps -el |grep top

RENICING A RUNNING PROCESS:

- The niceness of a running process can be changed using the renice command.
- Renicing will change the priority at which the process is currently running.

→ To change the nice of a running top session from old priority to +5:

$pidof crond

$renice 5 pid

$ps -el|grep crond

NOTE: Options -u and -g can be used with renice to change nice values of processes owned by a user or group members.

CONTROLLING PROCESSES WITH SIGNALS:

- A system may have hundreds or thousands of processes running simultaneously on it. It is sometimes necessary to alert a process of an event.
- We can accomplish that by send a control signal to the process.
- A process can send a signal to alert each other as well
- A process will halt its execution as soon as it gets the signal and take appropriate action as per the enclosed instructions in that signal.
- A signal can instruct the process to terminate gracefully, kill it abruptly or force it to re-read its configuration.
- There are many signals are available for use but we well mostly deal with only a few of them.
- Each signal is associated with a unique numeric identifier, name and action.

→ To list all signals:

$kill -l

→ To kill a process:

$Kill [signal] <PID>

$ps

$kill -9 PID

KILL: The **kill command** needs PID(s) to send a signal.

PKILL: The **pkill command** needs process name(s) to send a signal to.

KILLALL: It can be terminating all processes that match provided criteria.

$killall top

➢ DAEMONS:

- These are special types of background processes that **start at system start-up** and keep running forever as a service; they don't die.
- They are started as system tasks, spontaneously. However, they can be controlled by a user via the **init / Systemd** process.

➢ INIT / SYSTEMD PROCESS:

- **Init/systemd** is the first program (**PID of 1**) in the Linux that is executed when the Linux boots up.
- Simply it is a mother of all processes on the system.
- It is started by the kernel itself, so in principle it does not have a parent process.

$pidof systemd