Process Management

Process- All running instances of a lunched or executable program is called process
It consists of:
A set of instruction which are loaded into memory and running in diff status.
All process having their own PID (Process ID)

RHEL 7 & 8 pid 1 = Systemd
Before it was initd(PID1)

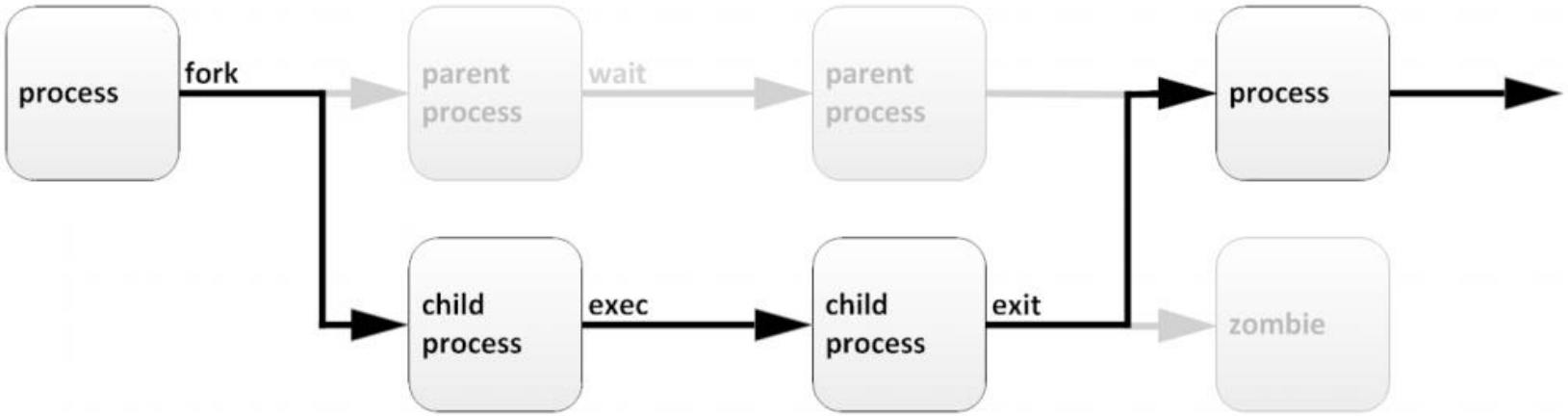
eg:
Vim, cat etc.

It consists of:
- An address space of allocated memory
- Security properties including ownership credentials and privileges
- Process state

The environment of a process includes:
- Local and global variables
- A current scheduling context
- Allocated system resources, such as file descriptors and network ports

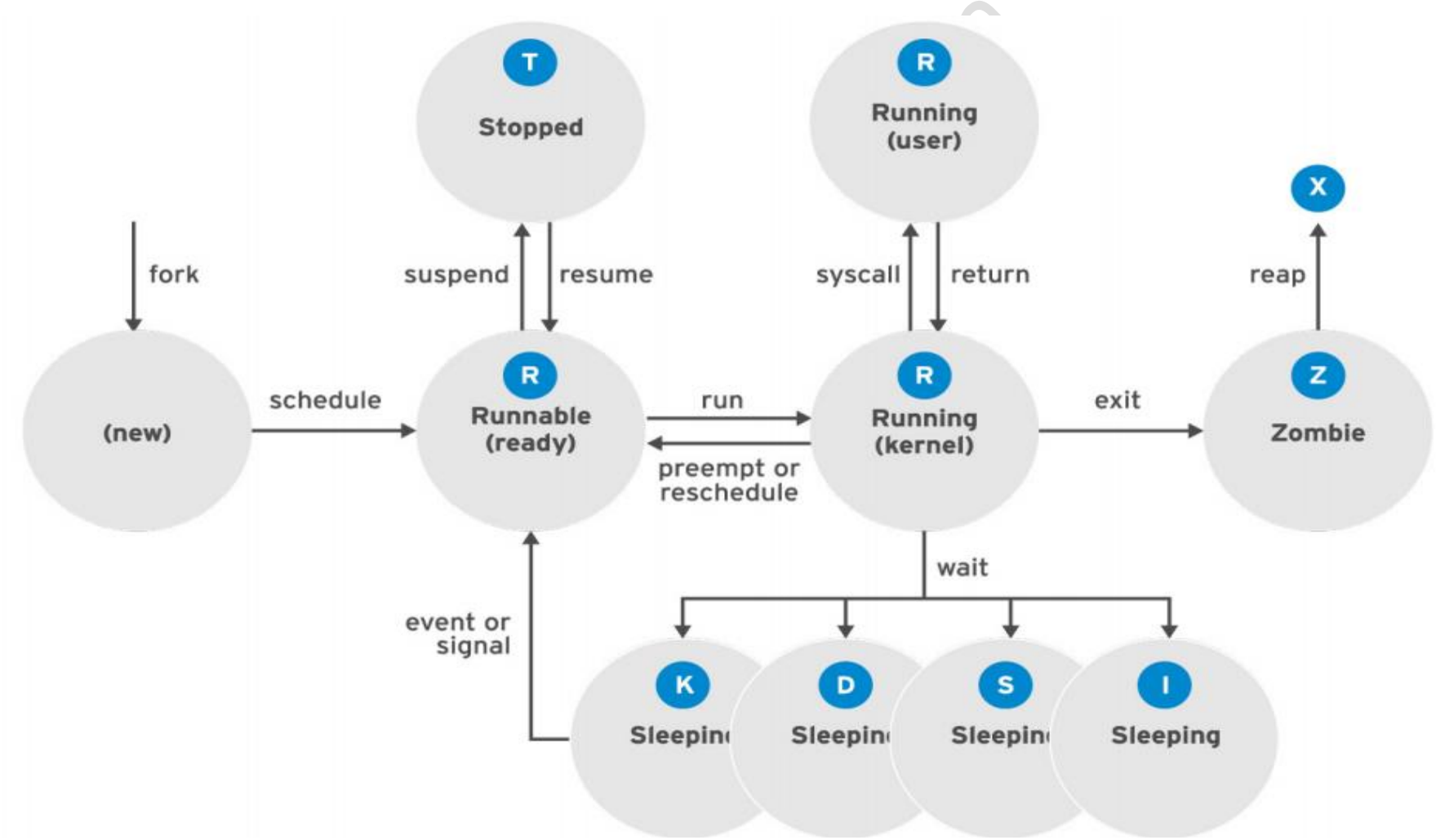
When we are executing any process, sometimes it duplicates its own Address space / FORK and creates some *child* process with new PID. So every child process has its own PPID (parent PID)

From the FROK of parent process, child inherit all like
Security details
PORT AND resource privileges
Environmental variable
Program codes
When a child process is going on, parent will wait until it complete.
Then child send a signal to its parent for exit, and close all its resources.
Still we can find the child process in process table as Zombie state
Then parent clean all child entry of it



States:

In a multitasking operating system, each cpu/cpu-core can be work on one process at a single time
So for managing that we have some states to assign them.



Running **(R)**: running or ready to run

Sleep **(S)**: process is waiting for some condition: H/W resources

(D): unlike S, it may be due to some device state

(K): Killable

(I): it's also an in sleep mode but it doesn't have any load.

Stopped **(T)**: stopped (suspended) may resume

(T): Traced its debugged

Zombie **(Z)**: it clears all except pid

(X): when parent clean up the remain child process structer

Listing out:

#ps command we can check the process list

UID: PID: CPU&REALTIME EXPANCE: MEMORY: LOCATION: STATE

Note: all process in () are from kernel

Controlling Jobs

Job controlling is a feature of shell which allow a single shell to manage multiple command.

e.g;

sleep 1000 & (& print your job id)
Ctrl + z (to stop the process)
Jobs
#bg %1 (% job id)
#fg %1 (% job id)

Killing Process

Usually when we or system start a process, it close itself when it done / we need to send a request to stop, kill etc.

Some time process can be hanging up, consuming much CPU / ram that time also for managing our resources we send kill request to a process.
We can use tool to check PID : top, ps, pidof, pgrep

If you are a normal user you can only kill your process
However root can kill all

Now to kill a process we need a Process Name or PID

SIGNALS: it's a software interrupt delivered to a process

1 HUP : request to terminate & reinitiate (refresh)
2 INT : "ctrl+c" can be blocked or handled
3 Quit : "ctrl+\" it produce process dump as well
9 KILL : can't blocked or handled
15 TERM: can be blocked or handled, in polite, self cleanup
(Default)
18 CONT: to resume
19 STOP: SUSPENDED
20 TSTP: "ctrl+z" can be handled

To find out all signals of kill
#kill -l

PKILL-

We use pkill to send a signal to one or more process which matches selection criteria. Criteria may be the command name, process owner, ppid etc.

Command
UID
GID
ppid
Terminal

#pkill PNAME
pkill -u username

pgrep- {to filter out process}
pgrep -l -u student
pstree- {to filter out process in tree structure}

****

W command

Owner: location: from: since: idle: jcpu: pcpu: what

user tty&pts f1 12:1 5 allP fgP command

JCPU: CPU resources consumed by current jobs, including background tasks and child processes.

PCPU: Current foreground process CPU consumption

MONITORING PROCESS ACTIVITY

Load average:- It's a measurement provided by Linux kernel , which mean load per a time

Normally the number of process ready to run or waiting for some condition per a time

uptime {to check load average , no of users & time since sys active}

Load average per 1min 5min 15min

#lscpu {to check no. of cpu sys have}

REAL-TIME PROCESS MONITORING

#top show dynamic sys-process

PID

OWNER

PR - priority of task

SHR -amount of shared memory utilization

VIRT- Virtual memory utilization

%CPU

%MEM- memory usage of task

S - Process state

RES- RAM utilization

CPU time (TIME)

The process command name (COMMAND).

KEY	PURPOSE
? *or* **H**	Help for interactive keystrokes.
L, **T**, **M**	Toggles for load, threads, and memory header lines.
1	Toggle showing individual CPUs or a summary for all CPUs in header.
S (1)	Change the refresh (screen) rate, in decimal seconds (e.g., 0.5, 1, 5).
B	Toggle reverse highlighting for **Running** processes; default is bold only.
B	Enables use of bold in display, in the header, and for *Running* processes.
Shift+H	Toggle threads; show process summary or individual threads.
U, **Shift+U**	Filter for any user name (effective, real).
Shift+M	Sorts process listing by memory usage, in descending order.
Shift+P	Sorts process listing by processor utilization, in descending order.

KEY	PURPOSE
K (1)	Kill a process. When prompted, enter **PID**, then **signal**.
R (1)	Renice a process. When prompted, enter **PID**, then **nice_value**.
Shift+W	Write (save) the current display configuration for use at the next **top** restart.
Q	Quit.
Note:	(1) Not available if top started in secure mode. See **top**(1).