LINKS:

Links are pointers
Connection between two strings

2 types of Links

1)Hard Links
2)Soft Links

Hard Links:
The two files having hard links will have the same inode number. If we update the data to the source file the same data we can access from the destination file & vice versa.
If we delete the source file we can access the data from the destination file with the same inode number. Hard links works as duplicate copies.

Limitations:
1)We cannot identify the files having hard links
2)hard links are applicable for only files
3)Hard links need to be configured with in the same filesystem.

```
#ln              <SF>            <DF>
                   Cap11      cap12
```

Soft Links :

The two files having soft link will be having different inode number. If we update the data to the source file the same data we can access from the destination file & vice versa.
If we delete the source file we cannot access the data from the destination file. Soft links acts as short cuts.

Advantages:
1)We can identify the files having soft links by seeing its perm.
2)Soft links are applicable for both files & directories.
3)Soft links can span across multiple file systems.

```
#ln     -s      <SF>            <DF>
              Cap12          cap13
```

Process management

Process:

Any Activity / task we execute in OS called as Process. Every process which is running in our OS will purly depends on another process.

Every process is identified with Unique ID called as PID (Process id). Several processes are started at boot time & which are running continuously in background called as Daemons.

Daemon is the continous process running in the background. These are handeled by the system & Process are handled by users.

→ The first process which starts while OS is booted in RHEL 6 is initd.

→The first process which starts while oS is booted in RHEL 7 & 8 is SystemD & its PID is 1.

→The Linux kernel communicates with the process using the PID.

Process States:

Zombie Process

orphan Process

Parent process

Child Process

Running Process

Seeping Process

Waiting Process

Stopped process.

Zombie Process :

The process which is running without child process called as Zombie Process. When we start parent process , It will start some child process. After some time the child process will die because of not knowing the parent process. These parent (which are running without child) process are called as zombie process.

Orphan:

The process which is running without parent process called s Orphan process.

Parent Process: The process which starts & creates another process called as Parent Process.

Parent Process:

The process which starts and create another process called as Parent process. Every process will have PP expect initd process & system(Which gets started at boot time by kernel with PID 1) only after this system gets started the remaining process are started. Parent Process is identified as PPID.

Child Process:

The process which is started, Created by the PP called as Child process. It is identified b PID.

Running Process:

The process which is in RUNNING state (**" r "**)

Sleeping Process :

The process which is in SLEEPY state (**" s "**)

Waiting :

The process which is in WAITING state (**" w "**)

Stopped / Stopping :

The process which is in stopped state (**" T "**)

#ps **(Lists all the process information in the current terminal)**

#ps -a **(Lists all the process information from all the terminals)**

#top **(We can monitor the process continuously, By default every 3 seconds it will refresh the data)**

#ps -ef **(It displays the total process information with Parent process id & PID)**

#ps -elf **(It displays the total process information with Parent process id & PID, process nice value, Location of the process information)**

#ps aux **(It displays the total process information with Parent process id & PID, process ncie value, Location of the process information along with that it displays CPU / RAM Information)**

#ps aux |wc -l **(Lists all the process running in an OS)**

Signals : Signals are the way of sending the message to process. We have around 64 Signals.

V IMP Signals :

Signal Name	**Value of Signal**	**Effect**
SIGHUP	1	Hangup
SIGINT	2	Interrupting from Keyboard
SIGKILL	9	Kinng the Parent & Child Process
SIGTERM	15	Termination Signal
SIGSTOP	17, 19,23	Stops the Process

Killing process with process name

```
#pkill          <process name>
```

Killing process with PID

```
#kil     l<signalvalue>          <PID>
```

```
#ps     -U      <Username>
```
(Lists all the process related to user)

```
#ps  -U  <Username> |wc  -l
```
(Lists the total process in number of a specific user)