31/3/22

1) Ans: P What is process Management in Linux and commands

The process means program in execution. It generally takes an input, process it and gives us the appropriate output.

i) Foreground processes: It is also called as interactive process. These are the process which are to be executed or initiated by user or the programmer. they can not be initialized by the system services. Such process take input from the user and return the output. While these processes are running we can not directly initiate a new process from the same terminal.

Background processes: Such kind of process are also known as non interactive process. These are the processes that are to be executed or initiated by the system itself or by users, though they can even managed by users.

These processes have a unique PID or process if assigned to them and we can initiate other processes within the same terminal from which they are initiated.

Process states

A process in linux go through different states after it's created and before it's terminated.

The states are:

- Running
- Sleeping

- Interruptible sleep
- Uninterruptible sleep
- Stopped
- Zombie
- A process in running state means that it is running or it ready to run
- The process is in sleeping state when it is waiting for a resource to be available.
- A process in Interruptible sleep will wakeup to handle signals, whereas a process in Uninterruptible sleep will not
- A process enters a stopped state when it recives a stop signal.
- Zombie state is when a process is dead but the entry for the process is still present in the table.

2) Ans:- Command for Manging Linux Processes

$ top: To track the running processes on your machine you can use the top command

$ PID: Unique Process ID given to each process.

$ PS - Process status. It displays the currently-running processes.

$ kill - To stop a process in Linux. Kill comonand is used.

$ Top : This tell the user about all the running processes on the Linux machine.

$ nice : To tell machine to prioritize processes as per your requirements. this priority is called niceness in linux.

$ DF : It reports the free disk space (Hard Disk) on all the file systems.

$ fg : To run a stopped process in the foreground.

2) what is Parent process, child process, orphan process, Zombie process, Deamon process.

Ans: i) Parent process - all the processes in OS are created when a process executes the fork() system call except the startup process. The process. In other word a parent process is one that creates a child process. A parent process may have multiple child processes but a child process only one parent process.

On the success of a fork() system call, the PID of the child process is returned to the parent process and 0 is returned to child process. On the failure of a fork() system call, -1 is returned to the parent process and a child process is not created.

ii) Child process: A child process is a process created by parent process in OS using a fork() system call. A child process may also be called a subprocess or a subtask. A child process is created as its parent process's copy and inherits most of its attributes. If a child process has no parent process, it was created directly by the kernel.

iii) Zombie Process: A zombie is a process that has completed its task but still it shows an entry in a process table. The zombie process usually occurs in the child process. Very short time the process is a zombie. After the process has completed all of its tasks it reports the parent process that it has about to terminate.

iv) Orphan Process: A child process that remains running even after its parent process is terminated or completed without waiting for the child process execution is called an orphan. A process becomes an orphan unintentionally. Some time intentionally becomes orphan due to long-running time to complete the assigned task without user attention. The orphan process has controlling terminals.

v) Deamon process: Deamon processes are started working when the system will be bootstrapped and terminated only when the system is shutdown. It does not have a controlling terminal. It always runs in the background.