**ASSIGNMENT – 2 (ps Command in Linux)**

Name - Ashutosh Soni

Id - 2018ucp1505

**Question 1: What is the process state reflected by following symbols in ps command? R, D, S, Z, T, N, L, s, <, +**

**Answer:**

The Process state reflected by the following symbols in ps command are:

R: running or runnable process or we can say the process on run queue.

D: uninterruptable sleep usually it is IO.

S: interruptible sleep means waiting for some process to complete.

Z: Defunct("Zombie") process, terminated but not reaped by its parent.

T: Stopped either by the job control signal or because it is being traced.

N: low-priority or we can say nice to other users.

L: has paged locked into memory. It’s for real time and custom IO.

s: is a session header.

<: high-priority or we can say not nice to other users.

+: is in the foreground process group.

**Question 2: Explain the following system calls related to process management, in brief, with help of a suitable example - fork (), wait (), kill (), exec (), getpid (), getppid (), exit ();**

**Answer:**

**fork ()** - The fork () system call is used to create the copy of the process executing. The executing process called as Parent process as the process it creates known as Child Process. The execution of parent process got suspended until the chid process completes its execution.

This function returns:

- pid > 0 for parent process.

- pid = 0 for child process.

- pid <0 for unsuccessful execution.

where pid stands for process id.

**wait ()** - The wait () system call is used to blocks the calling process until one of its child processes exits normally or it receives a signal.

After child process completes its execution then control come back to parent process it starts executing from where it goes to wait.

Child process may terminate because of following reasons:

- It calls exit ().

- It returns an int from main.

- It receives a signal form the OS or another process whose default action is to terminate.

**kill ()** - The kill () system call is used by OS to send a termination signal to a process that urges the process to exit. However, the kill system call does not necessary mean killing the parent process and can have various meaning.

This function return’s:

- 0 in case of successful execution.

- negative values otherwise.

**exec ()** - This exec () system call runs when an executable file in the context of an already running process that replaces the older executable file. However, the original process identifier remains as a new process is not built, but stack, data, head, data, etc. are replaced by the new process.

**getpid ()** - The getpid () system call is used when we need the process id (pid) of running process. This system call returns the pid of the calling process. This system call never throws an error it always gets successful.

**getppid ()** - The getppid () system call is used when we need the process id (pid) of the parent process of the calling process.

In case of the child process created by fork () system call and the parent process still exists then it returns the pid of the parent process Otherwise, this system call return’s 1 which is the process id of init process. This system call also nevers throws an error and always get successful

like getpid () system call.

**exit ()** - The exit () system call is used to terminate the program execution. In case of multi-threaded programming this call defines that the thread

execution is completed. The Operating System claims for the resources used by the process after calling exit () system call.

This system call has no return value as the process that calls got terminated and so couldn't receive a value anyway. This system call takes status as a parameter while calling.

**Question 3: What is a Zombie process? How is it different from an orphan process?**

**Answer:**

**Zombie Process:** Zombie process are those process who had completed their execution but still have an entry in the process table.

Let’s take an example,

A process created a child process via fork () system call and child process completed its execution and the parent process had not yet call wait () then in that case child process is known as zombie process. Once the system call wait () got executed, the process identifier of the zombie process

and the entry in the process table is released and then no longer child process is counted as Zombie process.

A child process always first become the Zombie Process before being removed form the Process Table.

whereas the **orphan process** is that process whose parent process no more exits i.e., the parent process either finished or terminated without waiting for the child process to terminate. In such cases they have entry in the process table and their entry is not being deleted by and program.

In this case new parent need to be assign to such process in order to remove their entry form the process table. The **init process** is assigned as new parent to such process or say orphan process, the init process invokes the wait () thereby allowing the exit status of the orphaned process to be collected and releasing orphan's process identifier and entry form the process table.