UNIX

Lesson 08 :Process and related commands

Lesson Objectives

- In this lesson, you will learn:
  - processes
  - Process Status Command – ps
  - Running processes in background mode
  - Terminate process
  - Process scheduling

8.1: Process
Processes

A process can be defined as an instance of an executing program.

Though there might be several processes running in memory at any given moment, the CPU can cater to only one of these processes as the other processes await their turn.

There is a program called “SCHEDULER” running in memory which decides which process should get the CPU time & attention & when.

Processes (Contd.)

At any given moment a process in memory can be in any one of the following state.

When you execute a program the Scheduler submit your process to a queue called Process Queue. At this instant the process is said to be in submit state. Once submitted the process waits its turn in the queue for sometime .At this stage the process is said to be in HOLD state.

As the process advances in the queue at some instant it would become the next one in the queue to receive CPU attention. At this stage it is in READY state.

Finally the process gets the attention of CPU & starts getting executed and thereby attains the RUN state.

A process whose execution comes to an end goes into complete_state & is then removed from the process queue.

Processes (Contd.)

- Whenever you enter a command at the shell prompt, it invokes a program. While this program is running it is called a process. Your login shell is also a process, created for you upon logging in and existing until you logout.
- UNIX is a multi-tasking operating system. Any user can have multiple processes running simultaneously, including multiple login sessions. As you do your work within the login shell, each command creates at least one new process while it executes.
- Process id: every process in a UNIX system has a unique PID - process identifier.

8.2: ps Command
Processes

- ps - displays information about processes. Note that the ps command differs between different UNIX systems - see the local ps man page for details.

- To see your current shell's processes:

```
$ ps
   PID    TTY  TIME CMD
 26450  pts/9  0:00 ps
 66801  pts/9  0:00 -csh
```

Processes (Contd.)

init process is the process having PID always 1(parent process of all other processes).
sh process is born the moment you loging & dies only when you log out of the system.

$ps –a (processes for all the users)
$ps –u username (processes for particular user)
$ps –t tty1(processes for particular terminal)
$ps –f (full listing of information)
$ps –e (select all processes)

ps command also contains ps itself as one of the running processes since it is obviously active when the snapshot is taken.

8.3: Suspend Process
Processes

- Suspend a process: Use CTRL-Z.
- Background a process: Normally, commands operate in the foreground - you can not do additional work until the command completes. Backgrounding a command allows you to continue working at the shell prompt.
- To start a job in the background, use an ampersand (&) when you invoke the command:
- myprog & To put an already running job in the background, first suspend it with CRTL-Z and then use the "bg" command:
- myprog - *execute a process*
- CTRL-Z - *suspend the process*
- bg - *put suspended process in background*

8.4: Jobs in the Background
Process in Background Mode

Processes can run in foreground or background mode.

- Only one process can run in foreground mode but multiple processes can run in background mode.
- The processes, which do not require user intervention can run in background mode, e.g. sort, find.
- To run a process in background, use & operator
  - $sort -0 emp.lst emp.lst &

nohup (no hangup) - permits execution of process even if user has logged off.

- $nohup sort emp.lst &   (sends output to nohup.out)

8.5: Killing process
Kill Process

Terminating a process

- **kill - use the kill command to send a signal to a process. In most cases, this will be a kill signal, hence the command name. However, other types of signals are usually supported. Note that you can only kill processes which you own. The command syntax is:**

```
kill [-signal] process_identifier(PID)
```

Examples:

```
kill  63878     - kills process 63878
kill -9 1225    - kills (kills!) process 1225. Use if simple kill doesn't work.
kill -STOP 2339 - stops process 2339
kill -CONT 2339 - continues stopped process 2339
kill -l         - list the supported kill signals
```

- **You can also use CTRL-C to kill the currently running process.**

8.6:Lowering job execution priority – nice command nice and wait command

nice - runs a program with modified scheduling priority.

Syntax :

```
nice [OPTION] [COMMAND [ARG]...]
```

- $ nice cat chap?? | nice wc –l > wclist &

Wait - waits for child process to complete.

Syntax :

```
wait  [ process id…  ]
```

- $wait 138 - waits for background job with pid 138

8.7:Scheduling process
cron

A system daemon which performs a specific task at regular intervals

The command and schedule information is kept in the directory /var/spool/cron/crontabs or in /usr/spool/cron/crontabs.

Each user has a crontab file. cron wakes up periodically and executes any job that are scheduled for that minute.

Only users who are listed in /etc/cron.allow or not listed in cron.deny can make an entry in the crontab.

Crontab <filename>   -used to make an entry in the crontab file.

- where the file contains the commands to execute

```
  MIN      HOUR       DOM      MOY        DOW          COMMAND
(0-50)   (0-23)     (1-31)   (1-12)      (0-6)          ---
   $         0         18       *           *              *       /home/gather
```

SUMMARY

- Unix processes
- Process related commands
  - ps
  - nohup
  - wait
  - kill
  - nice
- Background processes

Review Questions

❖Complete The Following :

- A unique number called the _______________ identifies each process.
- To bring a last background process in foreground we use ______________ command.

❖True / False

- A signal number of 9 is used, by default, by the kill command to terminate a process.
- You can kill any process, including the system process, using the kill command.