Lecture 8.Introduction to Linux Process Management

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

Types of processes in Linux

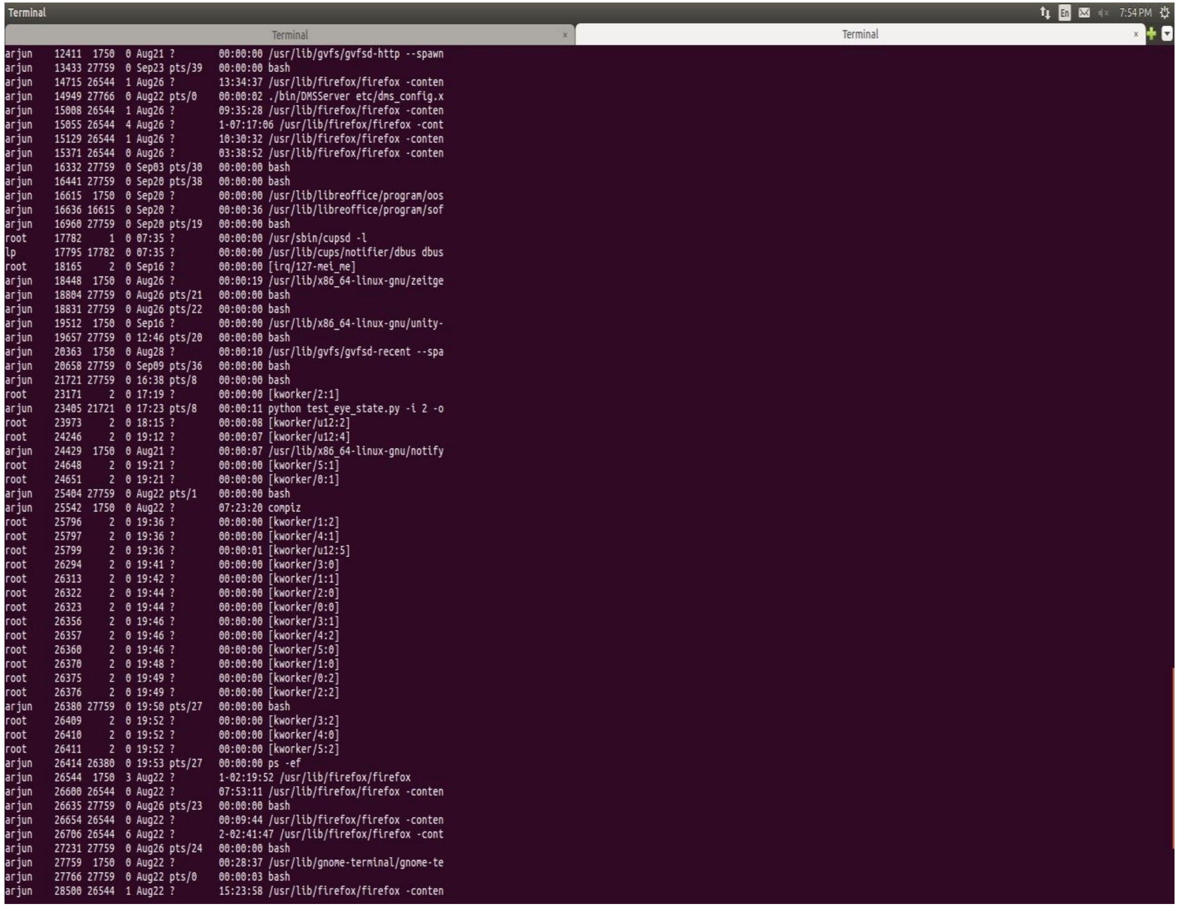
There are basically 2 types of processes.

1. **Foreground processes:** Such kind of processes are also known as **interactive processes**. These are the processes which are to be executed or initiated by the user or the programmer, they can not be initialized by system services. Such processes 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.
2. **Background processes:** Such kind of processes are also known as **non interactive processes**. These are the processes that are to be executed or initiated by the system itself or by users, though they can even be 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.The background process will be in stop state till input from the keyboard is given (usually 'Enter' key) then becomes a foreground process and gets executed. Only after the background process becomes a foreground process, that process gets completed else it will be a stop state.

Basic commands used for process management in Linux

Process status(ps): displays all the process in execution

To list all process in the background using ‘ps –f’ and to know more info on process use ‘ps -ef’

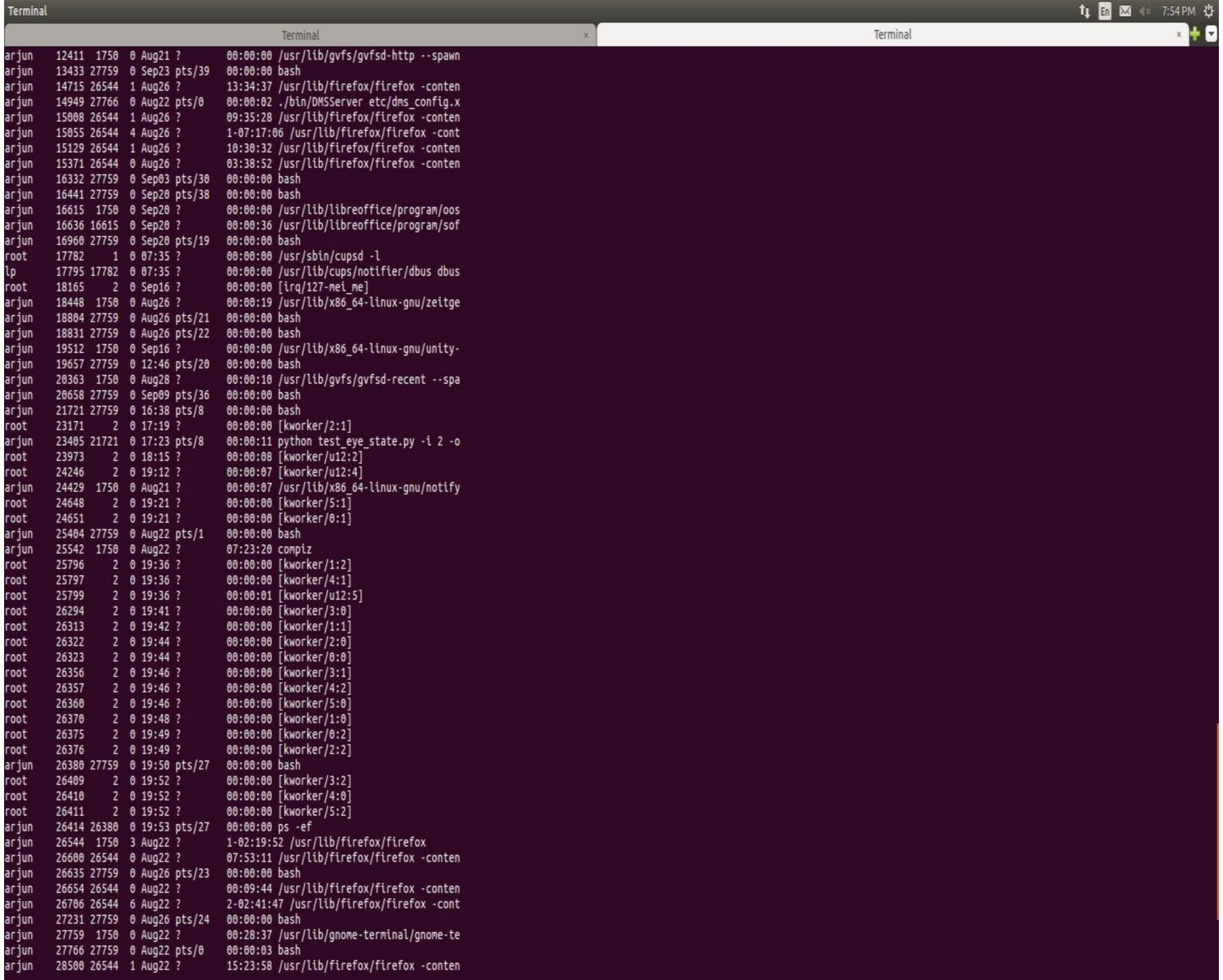


```
arjun    12411  1750  0 Aug21 ?        00:00:00 /usr/lib/gvfs/gvfsd-http --spawn
arjun    13433 27759  0 Sep23 pts/39   00:00:00 bash
arjun    14715 26544  1 Aug26 ?        13:34:37 /usr/lib/firefox/firefox -conten
arjun    14949 27766  0 Aug22 pts/0    00:00:02 ./bin/DMSServer etc/dms_config.x
arjun    15008 26544  1 Aug26 ?        09:35:28 /usr/lib/firefox/firefox -conten
arjun    15055 26544  4 Aug26 ?        1-07:17:06 /usr/lib/firefox/firefox -cont
arjun    15129 26544  1 Aug26 ?        10:30:32 /usr/lib/firefox/firefox -conten
arjun    15371 26544  0 Aug26 ?        03:38:52 /usr/lib/firefox/firefox -conten
arjun    16332 27759  0 Sep03 pts/30   00:00:00 bash
arjun    16441 27759  0 Sep20 pts/38   00:00:00 bash
arjun    16615  1750  0 Sep20 ?        00:00:00 /usr/lib/libreoffice/program/oos
arjun    16636 16615  0 Sep20 ?        00:00:36 /usr/lib/libreoffice/program/sof
arjun    16960 27759  0 Sep20 pts/19   00:00:00 bash
root     17782     1  0 07:35 ?        00:00:00 /usr/sbin/cupsd -l
lp       17795 17782  0 07:35 ?        00:00:00 /usr/lib/cups/notifier/dbus dbus
root     18165     2  0 Sep16 ?        00:00:00 [irq/127-mei_me]
arjun    18448  1750  0 Aug26 ?        00:00:19 /usr/lib/x86_64-linux-gnu/zeitge
arjun    18804 27759  0 Aug26 pts/21   00:00:00 bash
arjun    18831 27759  0 Aug26 pts/22   00:00:00 bash
arjun    19512  1750  0 Sep16 ?        00:00:00 /usr/lib/x86_64-linux-gnu/unity-
arjun    19657 27759  0 12:46 pts/20   00:00:00 bash
arjun    20363  1750  0 Aug28 ?        00:00:10 /usr/lib/gvfs/gvfsd-recent --spa
arjun    20658 27759  0 Sep09 pts/36   00:00:00 bash
arjun    21721 27759  0 16:38 pts/8    00:00:00 bash
root     23171     2  0 17:19 ?        00:00:00 [kworker/2:1]
arjun    23405 21721  0 17:23 pts/8    00:00:11 python test_eye_state.py -i 2 -o
root     23973     2  0 18:15 ?        00:00:08 [kworker/u12:2]
root     24246     2  0 19:12 ?        00:00:07 [kworker/u12:4]
arjun    24429  1750  0 Aug21 ?        00:00:07 /usr/lib/x86_64-linux-gnu/notify
root     24648     2  0 19:21 ?        00:00:00 [kworker/5:1]
root     24651     2  0 19:21 ?        00:00:00 [kworker/0:1]
arjun    25404 27759  0 Aug22 pts/1    00:00:00 bash
arjun    25542  1750  0 Aug22 ?        07:23:20 compiz
root     25796     2  0 19:36 ?        00:00:00 [kworker/1:2]
root     25797     2  0 19:36 ?        00:00:00 [kworker/4:1]
root     25799     2  0 19:36 ?        00:00:01 [kworker/u12:5]
root     26294     2  0 19:41 ?        00:00:00 [kworker/3:0]
root     26313     2  0 19:42 ?        00:00:00 [kworker/1:1]
root     26322     2  0 19:44 ?        00:00:00 [kworker/2:0]
root     26323     2  0 19:44 ?        00:00:00 [kworker/0:0]
root     26356     2  0 19:46 ?        00:00:00 [kworker/3:1]
root     26357     2  0 19:46 ?        00:00:00 [kworker/4:2]
root     26360     2  0 19:46 ?        00:00:00 [kworker/5:0]
root     26370     2  0 19:48 ?        00:00:00 [kworker/1:0]
root     26375     2  0 19:49 ?        00:00:00 [kworker/0:2]
root     26376     2  0 19:49 ?        00:00:00 [kworker/2:2]
arjun    26380 27759  0 19:50 pts/27   00:00:00 bash
root     26409     2  0 19:52 ?        00:00:00 [kworker/3:2]
root     26410     2  0 19:52 ?        00:00:00 [kworker/4:0]
root     26411     2  0 19:52 ?        00:00:00 [kworker/5:2]
arjun    26414 26380  0 19:53 pts/27   00:00:00 ps -ef
arjun    26544  1750  3 Aug22 ?        1-02:19:52 /usr/lib/firefox/firefox
arjun    26600 26544  0 Aug22 ?        07:53:11 /usr/lib/firefox/firefox -conten
arjun    26635 27759  0 Aug26 pts/23   00:00:00 bash
arjun    26654 26544  0 Aug22 ?        00:09:44 /usr/lib/firefox/firefox -conten
arjun    26706 26544  6 Aug22 ?        2-02:41:47 /usr/lib/firefox/firefox -cont
arjun    27231 27759  0 Aug26 pts/24   00:00:00 bash
arjun    27759  1750  0 Aug22 ?        00:28:37 /usr/lib/gnome-terminal/gnome-te
arjun    27766 27759  0 Aug22 pts/0    00:00:03 bash
arjun    28500 26544  1 Aug22 ?        15:23:58 /usr/lib/firefox/firefox -conten
```

- First column: User Id
- Second column: PID (process Id) – this is the 5-digit number assigned by OS for a process. No PID can be the same.
- Third column: PPID (parent process Id) – PID of the parent process
- Fourth column: CPU utilization of process
- Fifth column: STIME – Process start time
- Sixth column: TTY – the Terminal type associated with the process
- Seventh column: CMD – the command that started that process
- kill: Used to a process whose PID is known. To kill a process forcefully and unconditionally use
- “kill -9 PID”
- bg: A job control command that resumes suspended jobs while keeping them running in the background
- fg: It continues a stopped job by running it in the foreground

There are five types of Process in Linux

1. Parent process: The process created by the user on the terminal. All processes have a parent process, If it was created directly by user then the parent process will be the kernel process.



```
arjun    12411  1750  0 Aug21 ?        00:00:00 /usr/lib/gvfs/gvfsd-http --spawn
arjun    13433 27759  0 Sep23 pts/39   00:00:00 bash
arjun    14715 26544  1 Aug26 ?        13:34:37 /usr/lib/firefox/firefox -conten
arjun    14949 27766  0 Aug22 pts/0    00:00:02 ./bin/DMSServer etc/dms_config.x
arjun    15008 26544  1 Aug26 ?        09:35:28 /usr/lib/firefox/firefox -conten
arjun    15055 26544  4 Aug26 ?        1-07:17:06 /usr/lib/firefox/firefox -cont
arjun    15129 26544  1 Aug26 ?        10:30:32 /usr/lib/firefox/firefox -conten
arjun    15371 26544  0 Aug26 ?        03:38:52 /usr/lib/firefox/firefox -conten
arjun    16332 27759  0 Sep03 pts/30   00:00:00 bash
arjun    16441 27759  0 Sep20 pts/38   00:00:00 bash
arjun    16615  1750  0 Sep20 ?        00:00:00 /usr/lib/libreoffice/program/oos
arjun    16636 16615  0 Sep20 ?        00:00:36 /usr/lib/libreoffice/program/sof
arjun    16960 27759  0 Sep20 pts/19   00:00:00 bash
root     17782     1  0 07:35 ?        00:00:00 /usr/sbin/cupsd -l
lp       17795 17782  0 07:35 ?        00:00:00 /usr/lib/cups/notifier/dbus dbus
root     18165     2  0 Sep16 ?        00:00:00 [irq/127-mei_me]
arjun    18448  1750  0 Aug26 ?        00:00:19 /usr/lib/x86_64-linux-gnu/zeitge
arjun    18804 27759  0 Aug26 pts/21   00:00:00 bash
arjun    18831 27759  0 Aug26 pts/22   00:00:00 bash
arjun    19512  1750  0 Sep16 ?        00:00:00 /usr/lib/x86_64-linux-gnu/unity-
arjun    19657 27759  0 12:46 pts/20   00:00:00 bash
arjun    20363  1750  0 Aug28 ?        00:00:10 /usr/lib/gvfs/gvfsd-recent --spa
arjun    20658 27759  0 Sep09 pts/36   00:00:00 bash
arjun    21721 27759  0 16:38 pts/8    00:00:00 bash
root     23171     2  0 17:19 ?        00:00:00 [kworker/2:1]
arjun    23405 21721  0 17:23 pts/8    00:00:11 python test_eye_state.py -i 2 -o
root     23973     2  0 18:15 ?        00:00:08 [kworker/u12:2]
root     24246     2  0 19:12 ?        00:00:07 [kworker/u12:4]
arjun    24429  1750  0 Aug21 ?        00:00:07 /usr/lib/x86_64-linux-gnu/notify
root     24648     2  0 19:21 ?        00:00:00 [kworker/5:1]
root     24651     2  0 19:21 ?        00:00:00 [kworker/0:1]
arjun    25404 27759  0 Aug22 pts/1    00:00:00 bash
arjun    25542  1750  0 Aug22 ?        07:23:20 compiz
root     25796     2  0 19:36 ?        00:00:00 [kworker/1:2]
root     25797     2  0 19:36 ?        00:00:00 [kworker/4:1]
root     25799     2  0 19:36 ?        00:00:01 [kworker/u12:5]
root     26294     2  0 19:41 ?        00:00:00 [kworker/3:0]
root     26313     2  0 19:42 ?        00:00:00 [kworker/1:1]
root     26322     2  0 19:44 ?        00:00:00 [kworker/2:0]
root     26323     2  0 19:44 ?        00:00:00 [kworker/0:0]
root     26356     2  0 19:46 ?        00:00:00 [kworker/3:1]
root     26357     2  0 19:46 ?        00:00:00 [kworker/4:2]
root     26360     2  0 19:46 ?        00:00:00 [kworker/5:0]
root     26370     2  0 19:48 ?        00:00:00 [kworker/1:0]
root     26375     2  0 19:49 ?        00:00:00 [kworker/0:2]
root     26376     2  0 19:49 ?        00:00:00 [kworker/2:2]
arjun    26380 27759  0 19:50 pts/27   00:00:00 bash
root     26409     2  0 19:52 ?        00:00:00 [kworker/3:2]
root     26410     2  0 19:52 ?        00:00:00 [kworker/4:0]
root     26411     2  0 19:52 ?        00:00:00 [kworker/5:2]
arjun    26414 26380  0 19:53 pts/27   00:00:00 ps -ef
arjun    26544  1750  3 Aug22 ?        1-02:19:52 /usr/lib/firefox/firefox
arjun    26600 26544  0 Aug22 ?        07:53:11 /usr/lib/firefox/firefox -conten
arjun    26635 27759  0 Aug26 pts/23   00:00:00 bash
arjun    26654 26544  0 Aug22 ?        00:09:44 /usr/lib/firefox/firefox -conten
arjun    26706 26544  6 Aug22 ?        2-02:41:47 /usr/lib/firefox/firefox -cont
arjun    27231 27759  0 Aug26 pts/24   00:00:00 bash
arjun    27759  1750  0 Aug22 ?        00:28:37 /usr/lib/gnome-terminal/gnome-te
arjun    27766 27759  0 Aug22 pts/0    00:00:03 bash
arjun    28500 26544  1 Aug22 ?        15:23:58 /usr/lib/firefox/firefox -conten
```

2. Child process: The process created by another process (by its parent process). All child processes have a parent process.

The example is given above, the process having PID 28500(last row) is a child process of the process having PID 26544.

3. Orphan process: Sometimes when the parent gets executed before its own child process then the child process becomes an orphan process. The orphan process have “Init” process (PID 0) as their PPID (parent process ID)

4. Zombie process: The processes which are already dead but shows up in process status is called Zombie process. Zombie processes have Zero CPU consumption.

5. Daemon process: These are system-related processes that run in the background. A Daemon process can be recognized if it has “?” in its TTY field (6th column)

Managing running processes

The [ps] command is the command used to manage running processes and can be used for many things including viewing the status of your computer and getting a quick idea of how well the computer is performing.
Here are some common ps commands.
Command, Usefulness
ps View current interactive processes on this terminal.
ps –a All current processes on this terminal, by all users.
ps –x All processes not assigned to a terminal (daemons).
ps –aux Output all process running and include resource utilization information.
The man page for ps contains extensive documentation on how to modify and interpret the output of ps.
Top is a utility that can be used to display a live dataset of the currently running processes. Activate it by typing [top].

Thank you for your attention!