Performance Monitoring and Analysis

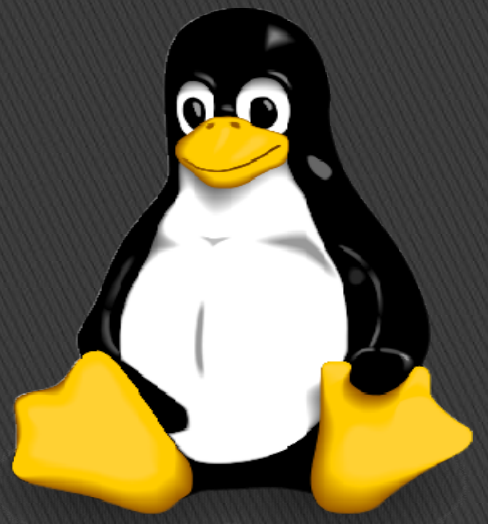
What is a Process in Linux?

- A process on a Linux system can be a running occurrence of an application or program. You can also refer to processes as tasks executing in the operating system.

- What is process ID:
- A Linux or Unix process is running instance of a program. For example, Firefox is a running process if you are browsing the Internet. Each time you start Firefox browser, the system is automatically assigned a unique process identification numb D). A PID is automatically assigned to each process when it is c on the system.

ps (processes status)

- **ps (processes status)** is a native Unix/Linux utility for viewing information concerning a selection of running processes on a system: it reads this information from the virtual files in /proc filesystem. It is one of the important utilities for system administration specifically under process monitoring, to help you understand whats is going on a Linux system.
- It has numerous options for manipulating its output, however you'll find a small number of them practically useful for daily usage.



ps

If you run **ps command** without any arguments, it displays processes for the current shell.

```
# ps
```

Print All Processes in Different Formats

Display every active process on a Linux system in generic (Unix/Linux) format.

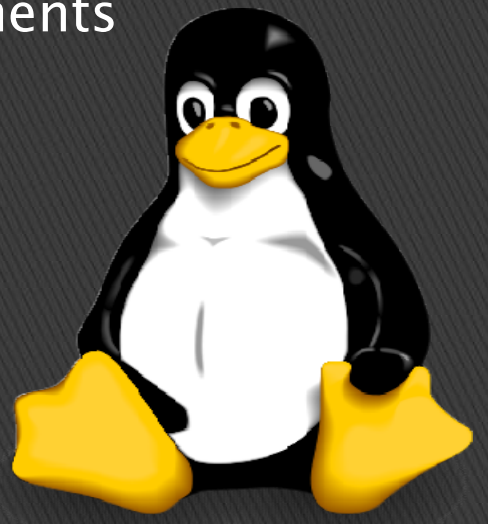
```
# ps -A
```

- OR

```
# ps -e
```

Print All Processes

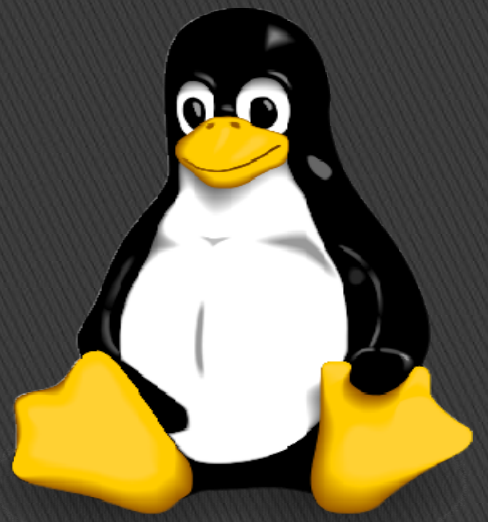
- UID: The username of the owner of the process
- PID: The unique process identification number of the process
- PPID: The parent process identification number of the process
- STIME:The time the process started (hh:mm:ss)
- TTY:The controlling terminal for the process. Note that system processes (daemons) display a question mark (?), indicating the process started without the use of a terminal.
- TIME: The cumulative execution time for the process
- CMD: The command name, options, and arguments



Display all processes

```
# ps axu
```

- a = show processes for all users
- u = display the process's user/owner
- x = also show processes not attached to a terminal



To perform a full-format listing, add the -f or -F flag.

```
# ps -ef
```

Display User Running Processes

- You can select all processes owned by you (runner of the **ps command**, root in this case), type:

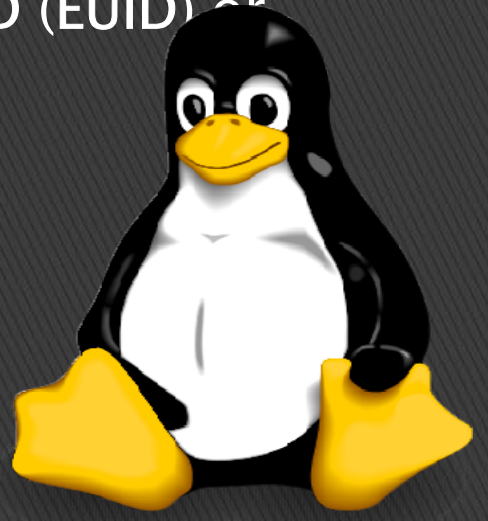
```
# ps -x
```

- To display a user's processes by real user ID (**RUID**) or name, use the **-U** flag.

```
# ps -fU  sachin
```

- To select a user's processes by effective user ID (**EUID**) or name, use the **-u** option.

```
# ps -fu sachin
```



Print All Processes Running as Root (Real and Effecitve ID)

The command below enables you to view every process running with **root** user privileges (real & effective ID) in user format.

```
# ps -U root -u root
```

Display Group Processes

- If you want to list all processes owned by a certain group (real group ID (**RGID**) or name), type.

```
# ps -fG apache
```

Display Processes by PID and PPID

- You can list processes by PID as follows.

```
# ps -fp 1178
```

- To select process by PPID, type.

```
# ps -f --ppid 1154
```

- Make selection using PID list.

```
# ps -fp 2226,1154,1146
```

Print Process Tree

- A process tree shows how processes on the system are linked to each other; processes whose parents have been killed are adopted by the init (or systemd).

```
# ps -e   --forest
```

- You can also print a process tree for a given process like this.

```
# ps   -f   --forest -C sshd
```

- OR

```
# ps   -ef    --forest | grep -v grep | grep sshd
```

Print Process Threads

- To print all threads of a process, use the -H flag, this will show the **LWP** (**light weight process**) as well as **NLWP** (**number of light weight process**) columns.

```
# ps -fL -C httpd
```

Specify Custom Output Format

- Using the **-o** or **-format** options, ps allows you to build user-defined output formats as shown below.

- To list all format specifies, include the L flag.

```
# ps L
```

- The command below allows you to view the **PID**, **PPID**, user name and command of a process.

```
# ps -eo pid,ppid,user,cmd
```

Display Parent and Child Processes

- To select a specific process by its name, use the -C flag, this will also display all its child processes.

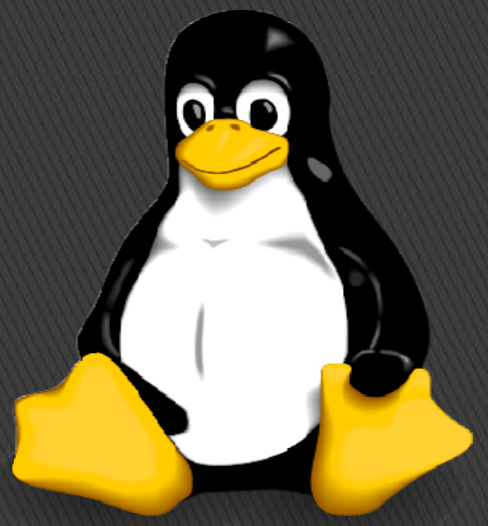
```
# ps -C sshd
```

- Find all PIDs of all instances of a process, useful when writing scripts that need to read PIDs from a std output or file.

```
# ps -C httpd -o pid=
```

- Check execution time of a process.

```
# ps -eo comm,etime,user | grep httpd
```



Troubleshoot Linux System Performance

```
# ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem | head
```

- OR

```
# ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%cpu | head
```

- To kill an Linux processes/unresponsive applications or any process that is consuming high CPU time.

- First, find the **PID** of the unresponsive process or application.

```
# ps -A | grep -i stress
```

- Then use the kill command to terminate it immediate

```
# kill -9 2583 2584
```

Print Security Information

- Show security context (specifically for **SELinux**) like this.

```
# ps -eM
```

- OR

```
# ps -context
```

- You can also display security information in user-defined format with this command.

```
# ps -eo euser,ruser,suser,fuser,f,comm,label
```

Perform Real-time Process Monitoring Using Watch Utility

Finally, since **ps** displays static information, you can employ the watch utility to perform real-time process monitoring with repetitive output, displayed after every second as in the command below (specify a custom **ps command** to achieve your objective).

```
# watch -n 1 'ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem
  | head'
```

Top Processes sorted by RAM or CPU Usage in Linux

- The following command will show the list of top processes ordered by RAM and CPU use in descendant form (remove the **pipeline** and **head** if you want to see the full list):

- # ps -eo pid,ppid,cmd,%mem,%cpu --sort=-%mem | head

Top Processes sorted by RAM or CPU Usage in Linux

- Brief explanation of above options used in above command.

- The -o (or –**format**) option of **ps** allows you to specify the output format. A favorite of mine is to show the processes' **PIDs** (**pid**), **PPIDs** (**pid**), the name of the executable file associated with the process (**cmd**), and the RAM and CPU utilization (%mem and %cpu, respectively).
- Additionally, I use --sort to sort by either %mem or %cpu. By default, the output will be sorted in ascendant form, but personally I prefer to reverse that order by adding a m sign in front of the sort criteria.

How to Find Process PID in Linux

- In Linux every process on a system has a **PID** (**Process Identification Number**) which can be used to kill the process.
- You can identify the **PID** of any process by using the **pidof** command as follows:

```
# pidof firefox
# pidof chrome
# pidof gimp-2.8
```

How to Kill Processes in Linux

- Once you find the process PID, let us now look at how to kill processes. In this first example, I am going to first get the PID of the process and then send a signal to it.

```
# pidof   httpd
# kill 9378
```

- To verify that the process has been killed, run the **pidof** command and you will not be able to view the **PID.**

```
# pidof   httpd
```

- Kill process by name
- #pkill   ssh

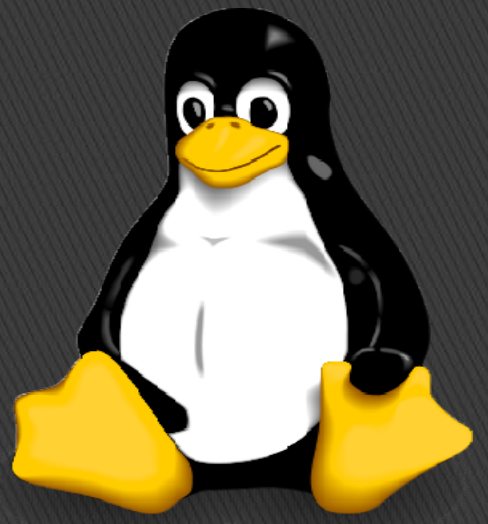
- For kill all process by using kill signal (-9)
- #Killall   -9   httpd

How to Kill Multiple Process PID's in Linux

- To kill more than one process, pass the PID(s) to the kill command as follows:

```
# pidof gimp-2.8
# pidof vlc
# pidof banshee

# kill -9 9734 9747 9762
```



Monitor Linux Performance

- Following command are also very useful to monitor system performance

1. Top – Linux Process Monitoring

- Linux **Top** command is a performance monitoring program which is used frequently by many system administrators to monitor Linux performance and it is available under many **Linux**/**Unix** like operating systems. The top command used to dipslay all the running and active real-time processes in ordered list and updates it regularly. It display **CPU usage**, **Memory usage**, **Swap Memory**, **Cache Size**, **Buffer Size**, **Process PID**, **User**, **Command**s and much more. It also shows high **memory** and **cpu** utilization of a running processess. The top command is much userful for system administrator to monitor and take correct action when required. Let's see top command in action.
- # top

2. VmStat – Virtual Memory Statistics

- Linux **VmStat** command used to display statistics of **virtual memory, kernerl threads, disks, system processes, I/O blocks, interrupts, CPU activity** and much more. By default vmstat command is not available under Linux systems you need to install a package called **sysstat** that includes a vmstat program. The common usage of command format is.
- **# vmstat**

3. Lsof – List Open Files

- **Lsof** command used in many **Linux/Unix** like system that is used to display list of all the open files and the processes. The open files included are **disk files**, **network sockets**, **pipes**, **devices** and **processes.** One of the main reason for using this command is when a disk cannot be unmounted and displays the error that files are being used or opened. With this commmand you can easily identify which files are in use. The most common format for this command is.
- # lsof

4. Tcpdump – Network Packet Analyzer

- **Tcpdump** one of the most widely used command-line **network packet analyzer** or **packets sniffer** program that is used capture or filter TCP/IP packets that received or transferred on a specific interface over a network. It also provides a option to save captured packages in a file for later analysis. tcpdump is almost available in all major Linux distributions.

```
# tcpdump   -i  enpos3
```

5. Netstat – Network Statistics

- **Netstat** is a command line tool for monitoring **incoming** and **outgoing network** packets statistics as well as interface statistics. It is very useful tool for every system administrator to monitor network performance and troubleshoot network related problems.

```
# netstat -a | more
```