**Lab 2 - Identifying Processes in Linux**

For this lab, you will be learning more about the “ps” command in Linux.

**What is a process?**

When we run a program, it contains a *series of instructions* that tells the computer what to do. These instructions are then used to figure out how much memory is needed, what computer resources are required, etc. These series of instructions are known as a *processes*.

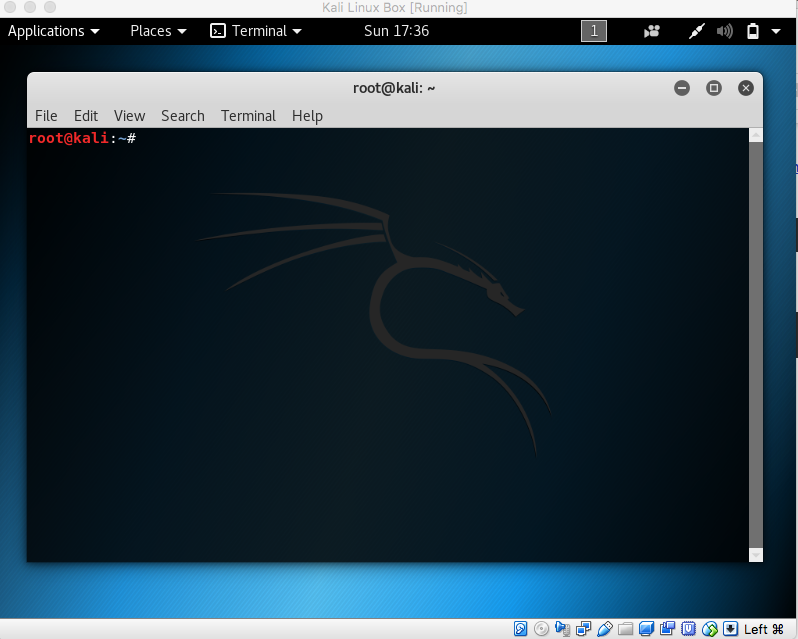
**What is ps?**

“PS” stands for “**Process Status**”. It can be used to display a list of processes that are currently running on your computer, the CPU usage, memory usage, command name, etc.

**Learning the Syntax**

**You can use...**

**Option 1:** The Kali Linux virtual machine that we have created from the previous labs and run these commands in terminal.



**Option 2:** If you have a MacOS, you can input this in terminal. **NOTE: Some commands may not work in terminal**

**Option 3:** This terminal online: <https://www.tutorialspoint.com/unix_terminal_online.php>

Type this first (without the ‘$’):

|  |
| --- |
| root@kali:~# ps |

The output should be look similar to this:

|  |
| --- |
| PID TTY TIME CMD  1377 pts/0 00:00:00 bash  1717 pts/0 00:00:00 ps |

The first line is a header that contains information for each row.

**PID –** the unique process ID

**TTY –** terminal type that the user is logged into

**TIME –** How long the process has been running

**CMD –** name of the command that launched the process.

**Now we will run through 10 examples of how to use the ps command. x**

**Table of Content**

1. **Displaying all processes**
2. **Displaying the process of users**
3. **Show process by name or ID**
4. **Sort processes by CPU or memory storage**
5. **Display process hierarchy in a tree style**
6. **Display child processes of a parent process**
7. **Display threads of a process**
8. **Change the columns to display**
9. **Display the elapsed time of processes**
10. **Real-time command view using ps**
11. **Displaying all processes**

This following command will display all processes:

|  |
| --- |
| root@kali:~# ps -ef |

Sample output:

|  |
| --- |
| UID PID PPID C STIME TTY TIME CMD  root 1 0 0 16:34 ? 00:00:01 /sbin/init  root 2 0 0 16:34 ? 00:00:00 [kthreadd]  root 3 2 0 16:34 ? 00:00:00 [kworker/0:0]  root 4 2 0 16:34 ? 00:00:00 [kworker/0:0H]  root 5 2 0 16:34 ? 00:00:00 [kworker/u4:0]  root 6 2 0 16:34 ? 00:00:00 [mm\_percpu\_wq] |

The new information that we see here contains:

**UID** - User ID

**PPID** - Parent Process ID.

**Parent & Child Processes -** One process can start another, thus creating a parent-child relationship.

**C** - CPU Usage

1. **Displaying processes by users**

In order to filter the processes by the user or UID, type in:

|  |
| --- |
| root@kali:~# ps -f -u [userid] |

For *[userid]*, replace it with your username or the UID from part 1. For instance, the UID can be ‘root’.

Sample output:

|  |
| --- |
| root@kali:~# ps -f -u root  UID PID PPID C STIME TTY TIME CMD  root 1 0 0 16:34 ? 00:00:01 /sbin/init  root 2 0 0 16:34 ? 00:00:00 [kthreadd]  root 4 2 0 16:34 ? 00:00:00 [kworker/0:0H]  root 6 2 0 16:34 ? 00:00:00 [mm\_percpu\_wq]  root 7 2 0 16:34 ? 00:00:00 [ksoftirqd/0]  root 8 2 0 16:34 ? 00:00:00 [rcu\_sched]  root 9 2 0 16:34 ? 00:00:00 [rcu\_bh] |

1. **Show process by name or id**

To search for a specific process, type:

|  |
| --- |
| root@kali:~# ps -C [process name] |

Use the process name from the row listed as “CMD”. For instance, a process name can be “bash”, resulting in this output:

|  |
| --- |
| root@kali:~# ps -C bash  PID TTY TIME CMD  1377 pts/0 00:00:00 bash |

Is it possible to search for multiple processes? Yes, but not with “-C”. For “-C”, the exact process name is required. It cannot search a partial name or wildcard.

Instead, use “-p”. Use a comma to separate the processes IDs/names.

For example, in order to search for processes with the PID 1828 and 1821, we can use this command:

|  |
| --- |
| root@kali:~# ps -f -p 1828,1831 |

Output:

|  |
| --- |
| UID PID PPID C STIME TTY TIME CMD  root 1828 2 0 17:06 ? 00:00:00 [kworker/0:2] |

1. **Sort processes by CPU or memory usage**

System admins often want to find out processes that are consuming lots of memory or CPU. This sort option will sort the process list based on a particular field or parameter.

|  |
| --- |
| root@kali:~# ps aux | sort |

The displays the processes in ascending order, according to the PID:

|  |
| --- |
| root@kali:~# ps aux | sort  62583 301 0.0 0.1 148272 3412 ? Ssl 16:34 0:00 /lib/systemd/systemd-timesyncd  colord 769 0.0 0.7 332788 15472 ? Ssl 16:34 0:00 /usr/lib/colord/colord  Debian-+ 587 0.0 0.3 76620 7784 ? Ss 16:34 0:00 /lib/systemd/systemd --user  Debian-+ 589 0.0 0.1 109196 2192 ? S 16:34 0:00 (sd-pam) |

To display the top 5 processes, simply add “head -5” to your command:

|  |
| --- |
| root@kali:~# ps aux | sort | head -5 |

You can also use to sort by CPU usage or memory:

|  |
| --- |
| root@kali:~# ps aux --sort=-pcpu,+pmem |

|  |
| --- |
| USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND root 699 49.2 1.3 528512 27952 ? Ssl 16:34 0:34 /usr/lib/packag root 978 32.9 14.1 3774384 289896 tty2 Sl+ 16:35 0:15 /usr/bin/gnome- Debian-+ 658 5.5 8.5 3362672 174328 tty1 Sl+ 16:34 0:04 /usr/bin/gnome- root 1210 4.9 2.6 1331536 55164 tty2 Sl+ 16:35 0:02 /usr/bin/gnome- root 825 3.5 2.6 507488 53808 tty2 Sl+ 16:35 0:01 /usr/lib/xorg/X root 1 2.0 0.4 159424 8712 ? Ss 16:34 0:01 /sbin/init root 1367 2.0 2.0 655580 42320 ? Ssl 16:35 0:00 /usr/lib/gnome- |

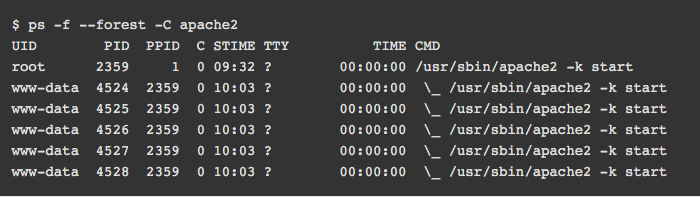
Which process is using the most CPU power in this example above?

What are the top 5 processes on your computer?

1. **Display process hierarchy in a tree style**

As mentioned before, processes can share a parent-child relationship. This means that one process can create another, thus making it important to be able to keep track of this hierarchy. The '--forest' option will construct an ascii art style tree view of the process hierarchy.

|  |
| --- |
| root@kali:~# ps -f --forest -C [name or ID of process] |

****

1. **Display child processes of a parent process**

To display the child processes of a particular parent, type:

|  |
| --- |
| root@kali:~# ps -o pid,uname,comm -C [process name] |

|  |
| --- |
| root@kali:~# ps -o pid,uname,comm -C gnome-session-binary  PID USER COMMAND  612 Debian-+ gnome-session-b  837 root gnome-session-b |

For this example, the name of the process is “gnome-session-binary”.

The first process that is owned by root is the main gnome-session-binary process and all others have been forked out of this main process.

The next command lists all child gnome-session-binary processes using the pid (which, in this case, is 612):

|  |
| --- |
| root@kali:~# ps --ppid 612  PID TTY TIME CMD  658 tty1 00:00:05 gnome-shell  698 tty1 00:00:00 gsd-xsettings  700 tty1 00:00:00 gsd-a11y-settin  702 tty1 00:00:00 gsd-clipboard  705 tty1 00:00:00 gsd-color  708 tty1 00:00:00 gsd-datetime  713 tty1 00:00:00 gsd-housekeepin  715 tty1 00:00:00 gsd-keyboard  720 tty1 00:00:00 gsd-media-keys  721 tty1 00:00:00 gsd-mouse |

1. **Display threads of a process**

Threads help processes perform their task by taking care of a specific tasks. **Threads** are used for small tasks, whereas **processes** are used for more 'heavyweight' tasks – basically the execution of applications.

|  |
| --- |
| root@kali:~# ps -p [process ID] -L |

This will display a new column, “LWP”, that shows all the threads belonging to that specific process:

|  |
| --- |
| root@kali:~# ps -p 715 -L  PID LWP TTY TIME CMD  715 715 tty1 00:00:00 gsd-keyboard  715 742 tty1 00:00:00 dconf worker  715 744 tty1 00:00:00 gmain  715 745 tty1 00:00:00 gdbus |

1. **Change the columns to display**

The following command shows only the pid, username, cpu, memory and command columns.

|  |
| --- |
| root@kali:~# ps -e -o pid,uname,pcpu,pmem,comm  PID USER %CPU %MEM COMMAND  1 root 0.0 0.4 systemd  2 root 0.0 0.0 kthreadd  4 root 0.0 0.0 kworker/0:0H  6 root 0.0 0.0 mm\_percpu\_wq  7 root 0.0 0.0 ksoftirqd/0 |

It is also possible to rename the column labels!

|  |
| --- |
| root@kali:~# ps -e -o pid,uname=USERNAME,pcpu=CPU\_USAGE,pmem,comm  PID USERNAME CPU\_USAGE %MEM COMMAND  1 root 0.0 0.4 systemd  2 root 0.0 0.0 kthreadd  4 root 0.0 0.0 kworker/0:0H  6 root 0.0 0.0 mm\_percpu\_wq  7 root 0.0 0.0 ksoftirqd/0 |

1. **Display the elapsed time of processes**

The elapsed time indicates how long the process has been running for:

|  |
| --- |
| root@kali:~# ps -e -o pid,comm,etime  PID COMMAND ELAPSED  1 systemd 02:12:11  2 kthreadd 02:12:11  4 kworker/0:0H 02:12:11  6 mm\_percpu\_wq 02:12:11  7 ksoftirqd/0 02:12:11  8 rcu\_sched 02:12:11  9 rcu\_bh 02:12:11 |

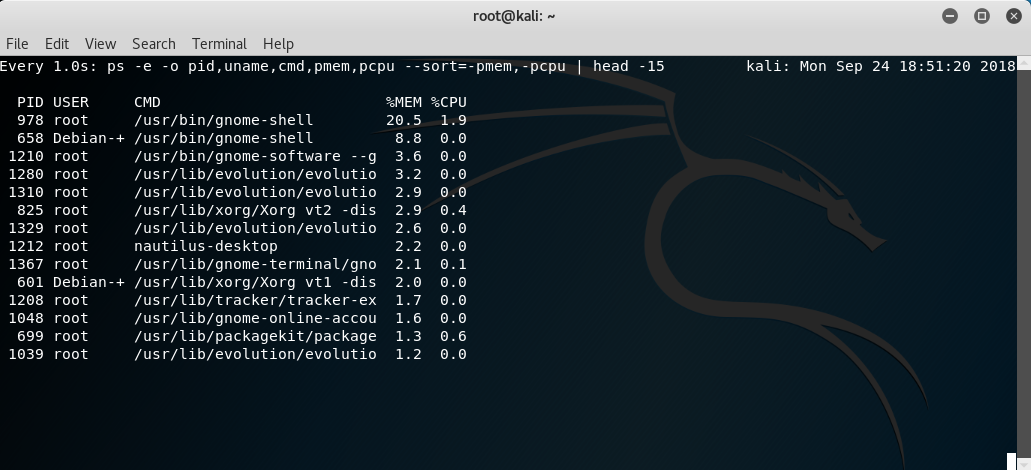
1. **Real-time command view using ps**

The watch command can be used to turn ps into a realtime process reporter.

Type this:

|  |
| --- |
| root@kali:~# watch -n 1 'ps -e -o pid,uname,cmd,pmem,pcpu --sort=-pmem,-pcpu | head -15' |

The output should look like this:



Congratulations! You have completed this lab.