System Initialization and Processes

Introduction and/or Background

Linux is a multitasking and multi-user system. It allows multiple processes (program instances) to operate simultaneously without interfering with each other. Run levels are operational levels that describe the state of the system with respect to what services are available.

Objectives

In this project/lab the student will:

- Gain familiarity with Linux Operating System Kernel, Processes and Run levels.

Equipment/Supplies Needed

- As specified in Lab 0.0.1.

Procedure

Perform the steps in this lab in the order they are presented to you. Answer all questions and record the requested information. Use the Linux Virtual Machine to perform lab activities as directed. Unless otherwise stated, all tasks done as a non-root user. If root access is needed use the sudo command.

Assignment

System Kernel

The Linux kernel is the core component of a Linux operating system. It serves as the interface between the computer's hardware and its processes

1. Access the terminal.

2. View and record how long the system has been running and the system date/time, by entering the commands **uptime** and **date** in the terminal.

   Examples:

   ```
   User1@debian1:~$ uptime
   ```

```
22:15:29 up 2 min 1 user,   load average 0.36, 0.18, 0.07

User1@debian1:~$ date

Thu Jun 4 22:15:34 CDT 2020
```

3. Access the man page for the **uname** command from the command line.

   Ex: user1@debian1:~$ man uname

   a. What type of information does the **–s**, **–r**, and **-v** options provide?

   b. What do the –m, -p, -i, and –o provide?

   c. Click on ‘q’ to quit (exit out of the man page).

4. Enter the **uname** command with the appropriate option to answer the following. View/Record the command issued and the resulting output.

   a. What is the kernel-release?

   b. What is the kernel version?

   c. Is the hardware platform 32bit or 64bit?

System Process(es)

The Process Status command [**ps**] provides administrators a view of information related to processes running on the system.

1. Review the man page for the **ps** command in the man pages.

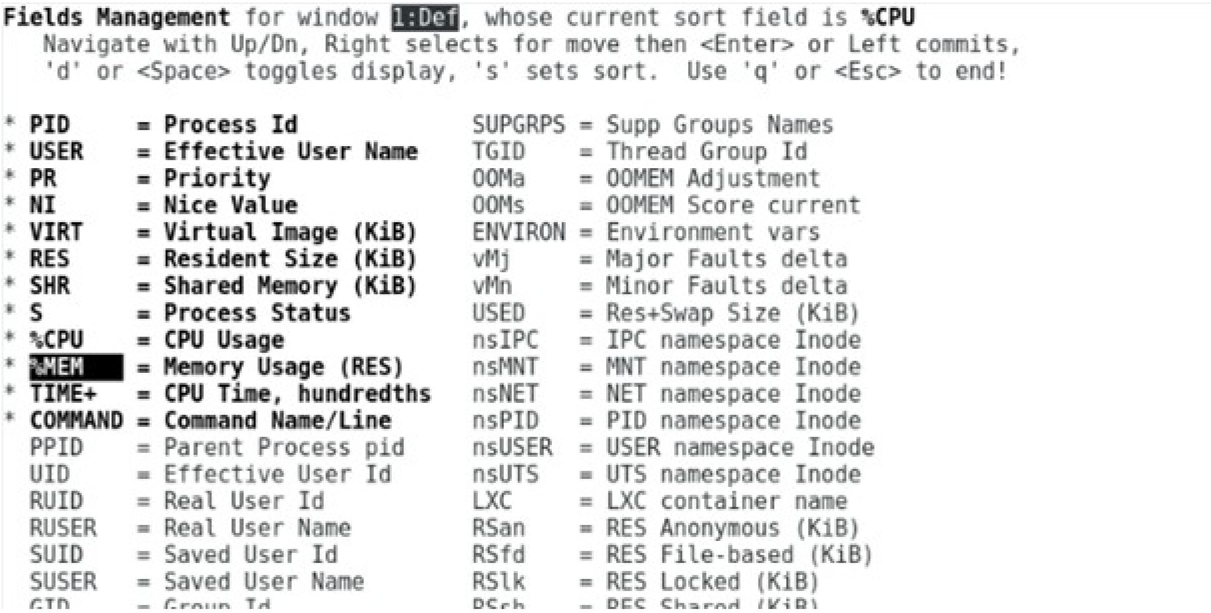
   a. Enter and Record a screenshot the first 10-15 lines of the output of the **ps** command with the **ux** option [**ps ux**]. This command shows only current logged in user processes. Who is logged in?

   b. Enter and Record the Output of the command **ps aux** display (shows all user processes). Who is logged in?

   c. Enter the command **pstree**. Record a partial screenshot of the output.

   d. Enter and Record a screenshot of the Output of the **top** command to check on processes. How many processes are running? Record a screenshot of the top processing running.

   e. Note the headings at the top of the list. Type the letter “**F**” in the terminal window. The Fields Management window should appear.



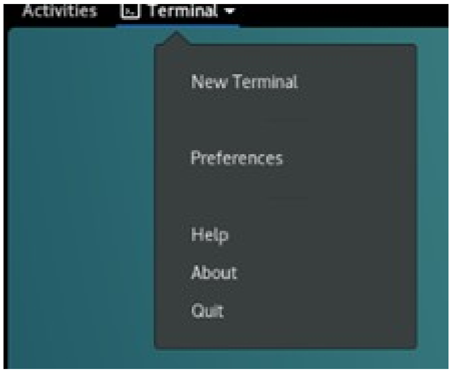
```
Fields Management for window 1:Def, whose current sort field is %CPU
   Navigate with Up/Dn, Right selects for move then <Enter> or Left commits,
   'd' or <Space> toggles display, 's' sets sort.  Use 'q' or <Esc> to end!

* PID     = Process Id              SUPGRPS = Supp Groups Names
* USER    = Effective User Name     TGID    = Thread Group Id
* PR      = Priority                OOMa    = OOMEM Adjustment
* NI      = Nice Value              OOMs    = OOMEM Score current
* VIRT    = Virtual Image (KiB)     ENVIRON = Environment vars
* RES     = Resident Size (KiB)     vMj     = Major Faults delta
* SHR     = Shared Memory (KiB)     vMn     = Minor Faults delta
* S       = Process Status          USED    = Res+Swap Size (KiB)
* %CPU    = CPU Usage               nsIPC   = IPC namespace Inode
* %MEM    = Memory Usage (RES)      nsMNT   = MNT namespace Inode
* TIME+   = CPU Time, hundredths    nsNET   = NET namespace Inode
* COMMAND = Command Name/Line       nsPID   = PID namespace Inode
  PPID    = Parent Process pid      nsUSER  = USER namespace Inode
  UID     = Effective User Id       nsUTS   = UTS namespace Inode
  RUID    = Real User Id            LXC     = LXC container name
  RUSER   = Real User Name          RSan    = RES Anonymous (KiB)
  SUID    = Saved User Id           RSfd    = RES File-based (KiB)
  SUSER   = Saved User Name         RSlk    = RES Locked (KiB)
  GID     = Group Id                RSsh    = RES Shared (KiB)
```

i. To sort the processes by memory, highlight **%MEM and enter the right arrow key (›). This will highlight %Mem.** Press **enter.** Press **s**. Press **q**. The columns should now be sorted by the %MEM column.
ii. Repeat the process, but sort by **PID**. Everything should be sorted by the PID column now.
iii. Record the output of each.
iv. Click on ‘**q’** again to get out the processes.
v. Enter the command **clear** to clear the screen.

2. Manage Linux Processes and Services.

a. As a non-root user, start the program **gedit**, by entering **gedit**. Notice that the terminal is not available to receive any new commands (no command prompt is displayed) because the **gedit** program is running in the foreground.

b. Open a new terminal window. Click on the terminal window. Click on Terminal in the top bar. Click on New Terminal.



c. In the new terminal enter the **top** command to view running processes. Exit top with ‘**q**’.

d. View only the processes run by you. Enter **top u <your_username>**. Exit **top**.

e. Look for and record the **pid** for the **gedit** process. Use the **pidof gedit** command.

f. View the process information for **gedit** by entering the following command: **ps <pid>** (where pid is the **gedit** pid recorded earlier). Record a screenshot.

g. Kill/terminate **gedit** application with the **kill <pid>** command. Note that the application is gone.

h. The **&** runs an application in the background. Run **gedit** in the "background".

   i. **gedit &** will run **gedit** in the background, however, it can still interact with the terminal window.
   ii. Try typing into the **gedit application** and in the terminal window. You should be able to type in both. Record a screenshot.
   iii. Close out the **gedit application**.

i. **Nice** values are user-space values that we can use to control the **priority** of a process. The **nice** value range is -20 to +19 where -20 is highest, 0 default and +19 is lowest. In the terminal window enter the command **gedit &**. Record the **pid** that is provided in the terminal again.

   i. Increase the priority of the **gedit** process to a nice value of -5 by entering the following command: **renice -5 –p <pid>** (Note: You must be root to change nice values. Switch to root by using the **sudo** command)

Example:

```
User1@debian:~$ su

Password: < Your password>

root@debian1:/home/user1# renice -5 –p -2665

2665 (proces ID) old prireniority 0, new priority -5

root@ debian1:/home/user1$
```

   ii. Use the renice command again to change the nice value for the **gedit** process to **0 [replace the -5 with 0]**.

iii. kill the **gedit** process by using the kill command: **kill <**pid>

iv. Exit root by typing **exit** in the terminal.

3. Manage Runlevels

In Linux, runlevels define how the system is started. The system starts as defined in the /etc/inittab file, in the initdefault line. Each runlevel has its own list of processes and services that are either turned on or off when the system boots. A runlevel is identified by a single number between 0 and 7. Note: There's rarely any actual need to change default runlevels, but they can come in handy when troubleshooting. (i.e switching to runlevel 1 for upgrades or modifications).

a. To identify the current run level of the computer, as root, use the **runlevel** command. Record the command output.

b. From the terminal, log in as root with the su command to try changing to another run level.

   Enter the **init** 6 command. What happened?

Lab Submissions Proof: Provide screenshots as indicated in the lab; upload your proof to Moodle for grading.

Rubric

Checklist/Single Point Mastery

Concerns Working Towards Proficiency	Criteria Standards for This Competency	Accomplished Evidence of Mastering Competency
	Criteria #1: Provide how long the system has been running and the system date/time (5 points)	
	Criteria #2: What type of information does the –s, –r, and -v options provide? (5 points)	
	Criteria #3: What do the –m, -p, -i, and –o provide? (5 points)	
	Criteria #4: What is the kernel-release? (5 points)	
	Criteria #5: What is the kernel version (5 points)	

	Criteria #6: Is the hardware platform 32bit or 64bit? (5 points)	
	Criteria #7: Enter and Record the first 10-15 lines of the output of the ps command with the ux option [ps ux]. (5 points)	
	Criteria #8: Who is logged in? (5 points)	
	Criteria #9: Enter and Record the Output of the command ps aux display (5 points)	
	Criteria #10: Who is logged in? (5 points)	
	Criteria #11: Provide a screenshot of the pstree command (5 points)	
	Criteria #12: Enter and Record the Output of the top command to check on processes. (5 points)	
	Criteria #13: How many processes are running? (5 points)	
	Criteria #14: Record pstree command sorted by %MEM Column (5 points)	
	Criteria #15: Record pstree command sorted by PID Column (5 points)	
	Criteria #16: Record the output of the ps command’s gedit output (5 points)	
	Criteria #17: Record a screenshot of the gedit application opened in terminal and proof the program opened (5 points)	
	Criteria #18: Record output of runlevel command (5 points)	
	Criteria #19: Record what happens when attempting to change runlevel as a root user	

	(10 points)	