**OPS102 – Week 5 – Process Management - Sample Lab**

Student Name: <Abisek Gurung>

Student ID: <135741239>

**Introduction**

Both Linux and Windows, as powerful operating systems, provides robust process management capabilities. Understanding how to manage processes is crucial for effectively utilizing the operating system. A process refers to an executing program or task, whether it is a system service, a user application, or a background utility.

Here are some fundamental concepts related to process management:

* **Processes and Process IDs (PIDs):** Every process in Linux or Windows is assigned a unique identifier called a Process ID (PID). PIDs enable the system to track and manage processes effectively. You can view the PIDs of running processes using various commands and utilities.
* **Process States**: Processes can be in different states, such as running, sleeping, stopped, or terminated. Understanding these states helps in monitoring and controlling processes effectively. Commands like ps and top provide insights into process states.
* **Process Ownership**: Each process is associated with an owner, typically the user who initiated or owns the process. Process ownership is essential for managing permissions and access control.
* **Process Hierarchy**: processes follow a hierarchical structure. A process can create child processes, and those child processes can, in turn, spawn their own subprocesses. This hierarchical arrangement helps organize and manage related processes.
* **Process Control**: Linux provides various commands and tools to control processes. You can start, stop, pause, resume, or terminate processes using commands like kill, killall, pkill, and signals such as SIGSTOP and SIGCONT. Windows offers multiple methods to control processes. The Task Manager, a built-in Windows utility, allows you to view and manage running processes. It enables you to end processes, change process priorities, and analyze resource usage.
* F**oreground and Background Processes**: Both Linux and windows allow executing processes either in the foreground or background. Foreground processes run directly in the terminal, while background processes operate independently, freeing up the terminal for other tasks. You can switch between foreground and background using commands like &, fg, and bg commands in Linux. In windows, Task Manager and PowerShell provide options to manage processes in both modes.
* **Process Monitoring and Resource Usage**: Monitoring the performance and resource usage of processes is essential for system administrators. In Linux, tools like top, htop, and ps provide real-time information on CPU usage, memory consumption, and other vital statistics. In Windows, Task Manager provides real-time information on CPU usage, memory consumption, disk activity, and network utilization. Performance Monitor (PerfMon) is a powerful tool for in-depth process monitoring.

**Activity 1: Monitoring Linux Processes with ps command**

Perform the following steps:

1. Make certain that you are logged into your Matrix account
2. Issue a Linux command to confirm that you are located in your **home** directory.
3. The **ps**\_ command provides a list of processes that are running, or at least that were running at the time the command was called. Run the command ps in your terminal
4. What output you see, take a screenshot and paste below.

A screen shot of a computer

Description automatically generated

1. How many processes are currently running? What information is displayed for each process? Answer below.

= Two processes are currently running. The information displayed are the table list of PID (process ID), (TTY) Terminal Type, (Time) Execution Time and CMD (command that started the process) related with each process.

1. Use the ps command with the ‘-e’ option to display information about all processes in the system. Run the command **ps -e**

A screenshot of a computer

Description automatically generated

1. Analyze the output and identify the running processes on your system. Note the PID, TTY, and CMD columns. What do these column mean?

=

* PID = It stands for process ID, which helps to find the type of process.
* TTY = It stands for the type of terminal from which the process was started.
* CMD = It stands for the command that started the process

1. Use the 'ps' command with the '-f' option to display a full-format listing of the processes. Run the command **ps -f**

A screen shot of a computer screen

Description automatically generated

1. Examine the output, which provides detailed information about each process, including UID, PID, PPID, CPU%, MEM%, START, and CMD

* UID = It stands for User ID and shows the owner the process.
* PID = It stands for Process ID, a unique identifying number to identify the process.
* PPID = It stands for Parent Process ID, a unique identifying number to identify the parent processes.
* CPU% = It indicates the percentage of CPU used by the process
* MEM% = It indicates the percentage of physical memory (RAM) used by the process.
* START = It stands for the starting time of the process.
* CMD = It stands for Command, It indicates the name of the command.

1. Use the 'ps' command with the '-l' option to display a long listing format of processes. Execute the following command: **ps -l**

A black screen with white text

Description automatically generated

1. Analyze the output and observe the columns displayed, including F, S, UID, PID, PPID, PRI, NI, ADDR, SZ, RSS, WCHAN, STAT, TTY, TIME, and CMD.

* F = It stands for the flag.
* S = It stands for State pr process (R= Running and s= Sleeping/stopped).
* UID = It stands for User ID and shows the owner the process.
* PID = It stands for Process ID, a unique identifying number to identify the process.
* PPID = It stands for Parent Process ID, a unique identifying number to identify the parent processes.
* PRI = It stands for Priority; it indicates the priority of the process.
* NI = It stands for the nice value of the process. A negative value indicates high priority.
* ADDR = It
* SZ = It stands for Size; it indicates the total size of the memory.
* RSS = It stands for Resident Set Size; it indicates the amount of physical memory used by the process.
* WCHAN = It stands for Wait Channel, It represents the name of the kernel function if the process is in a sleeping state,
* STAT = It stands for Status; it shows additional information about the current state of the process.
* TTY = It stands for Teletypewriter; it indicates the type of terminal from when the process was started.
* TIME = It indicates the Total CPU time consumed by the process from where the process was started.
* CMD = It stands for Command, It indicates the name of the command.

1. Use the '**-u**' option followed by a username to display processes owned by that user.

A black background with white text

Description automatically generated

1. Use the '**-p**' option followed by a process ID (PID) to display information about a specific process.

A screen shot of a computer code

Description automatically generated

**Activity 2: Monitoring Linux Processes with top command**

The **top** command is a powerful tool in Linux used to monitor and manage system resources in real-time. It provides a dynamic view of CPU usage, memory utilization, running processes, and other essential system metrics.

In this activity, experiment with this command to understand resource usage.

1. Run the command **top** in your terminal. What output do you observe, below paste a screenshot of the terminal output?

A screenshot of a computer

Description automatically generated

1. Once the top command is running, you'll see a continuously updated display with various sections and columns.
2. Explain what information the following columns give.
   1. PR = It indicates the priority of the process. Where lower numerical values are given higher priority.
   2. NI = It indicates the nice value of the process. In which a negative number indicates a higher priority and a positive value indicates a lower priority.
   3. VIRT = It shows the total amount of virtual memory used by the processor.
   4. RES = It shows the amount of physical memory used by the processor for the current task.
   5. %CPU = It indicates the percentage of CPU used by the process.
   6. %MEM = It indicates the percentage of physical memory (RAM) used by the process.
   7. TIME+ = It indicates the total CPU time used by the task from its start. As the process runs the value will accumulate itself.
3. The top command provides interactive features to customize the display and perform actions. Press 'P' to sort processes by CPU usage, 'M' to sort by memory usage, and 'N' to sort by PID.

= P

A screenshot of a computer

Description automatically generated

= M

A screenshot of a computer

Description automatically generated

=N

A screenshot of a computer

Description automatically generated

1. To exit the top command, simply press 'q'. This will close the top display and return you to the terminal prompt.

A screenshot of a computer

Description automatically generated

**Activity 3: Sending signals to processes**

In Linux processes, system admins can send signals to communicate with processes and request specific actions. Signals are software interrupts delivered to a process by the operating system or another process. Signals allow processes to respond to various events, such as the termination of another process, user input, or changes in system conditions.

Signals are identified by unique numbers, known as signal numbers. Each signal number corresponds to a specific event or action.

Each signal has a default action associated with it, which determines what the process does when it receives that signal. Common default actions include termination, stopping, or ignoring the signal.

Common Signals: Linux systems have a set of standard signals defined, each with its own signal number. Some commonly used signals include:

* SIGTERM (Signal 15): This is the default signal sent by the kill command to request a process to terminate gracefully.
* SIGKILL (Signal 9): This signal immediately terminates a process. It cannot be caught or ignored.
* SIGSTOP (Signal 19): This signal pauses a process, suspending its execution until a SIGCONT signal is received.
* SIGCONT (Signal 18): This signal resumes the execution of a process that was previously stopped by a SIGSTOP signal.
* SIGHUP (Signal 1): This signal is typically sent to inform a process that the controlling terminal has been disconnected.

Signals can be sent to processes using the **kill** command.

Perform the following steps:

1. Issue the following command: **sleep 500**

The "sleep" command in Linux is a utility that allows you to pause the execution of a script or command for a specified amount of time. We will be using this command to simulate the behavior of a "long-running" process. This process will run for **500 seconds**, and is forcing the user to **wait** until this process finishes. A process that is **running in the terminal** is referred to as a **foreground process**.

1. Run the command: **psf**
2. Note the process id of sleep command.
3. Run the command: **kill PID** (replace PID with process id)

By default, the kill command sends the SIGTERM signal (signal number 15) to the process, requesting it to terminate gracefully. However, you can specify a different signal using the -s option followed by the signal number or signal name.

What output you see? Paste a screensot of the output below.

A computer screen shot of white text

Description automatically generated

Run the command sleep 500 another time and this time send the SIGKILL singnal to this process. What output you see? Paste a screensot of the output below.

A screenshot of a computer program

Description automatically generated

1. What difference you noticed in SIGTERM and SIGKILL singals?

= SIGTERM allows the process to be gracefully terminated while SIGKILL terminates the process forcefully.

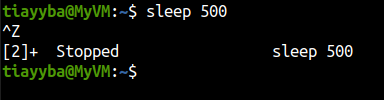
**Activity 4: Foreground and background processes**

1. Again ssue the following command:

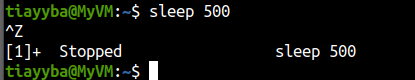
sleep 500

The Unix/Linux system is designed to allow users to send **preemptive signals** to manage those processes.

1. Press the following key combination to interrupt the process running on the terminal: **ctrl-z.** This sends a SIGSTOP signal to the process.
2. You should see output similar to what is displayed below:

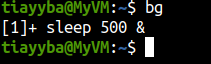


1. This indicates that this process has been placed into the **background**. This is useful in order to "**free-up**" the terminal to run other Linux commands
2. Issue the following Linux command: **jobs**  
   You should see the following output:



This display indicates that this process (that is now in the background) has **stopped**.  
In other words, the *sleep* command is NOT counting-down to zero to terminate.

1. The plus sign "+" indicates the most recent process placed into the background.
2. Sometimes you would like to run the process you stopped in the background. You can use bg command without arguments to run in background the most recent process that was stopped.
3. Run the command: **bg**
4. Issue the command: **jobs**
5. You should see the following output similar to what was displayed above



1. The & sign indicates that the process is now running in the backlground.
2. You can also bring this process to foreground using fg command.
3. Issue the command **fg.** This will make the sleep process run in foreground.  
   A screen shot of a computer code

   Description automatically generated

**Activity 5: Managing Windows Processes with PowerShell**

Mostly Task Manager application is used for managing processes on Windows. However, Windows PowerShell does provide some commands for process management. The main command used to get information about process is called ‘**Get-Process**’. In the following tasks use this command in Windows PowerShell to get information about the process.

1. Run the **Get-Process** command in PowerShell and explain the output

A screenshot of a computer

Description automatically generated

=The command lists all the process that is running in the local computer.

1. Explain the meaning of column headers of the information output by this command.

* Handles: - It indicates the number of handles that the process has opened.
* NMP(k): - It indicates the non-paged memory used by the process. It is measured in Kilobytes.
* PM(k): -It indicates the paged memory used by the process. It is measured in Kilobytes.
* WS(k): -It indicates the physical memory that the process is currently using. . It is measured in Kilobytes.
* CPU(s): - It indicates the CPU time that the process has consumed. It is measured in seconds.
* ID: - It indicates the Unique ID of the process.
* SI: - It indicates the session ID of the process. It is a unique identifier that a web server assigns to the user.
* ProcessName: - It is the name of the process where the process is currently running.

1. To get information about specific process you can use the syntax **Get-Process <process-name>**. For example, to get information about firefox process you can run command **Get-Process firefox**. Run this command to get information about a process of your choosing and show the screenshot below.

A screenshot of a computer screen

Description automatically generated

1. Using this same command describe with example how you can get information about multiple processes

A screenshot of a computer screen

Description automatically generatedA screenshot of a computer screen

Description automatically generated

= The Get-process command lists all the current processes which is running in the windows. We can also find the specific process by writing the name of the file with the command (Get-process -name chrome, WhatsApp).

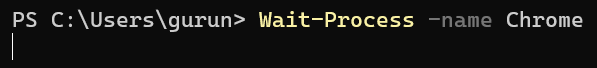
1. To stop a process you can use **Stop-Process** command with syntax **Stop-Process <process-name>**. In this task use this command to stop some process and show the screenshot below

A screenshot of a computer

Description automatically generated

1. There are two other commands of this class to manage processes. These commands are **Wait-Process** and **Debug-Process**. Search and read about these commands and provide examples.

= The command Wait-Process waits for one or more processes to be stopped before accepting the input. The command Debug-process attaches a debugger to a running process for debugging.

****

**A screenshot of a computer

Description automatically generated**

**Further Practice Questions.**

Answer the following questions based on your knowledge of process management in Linux.

1. What is a process in Linux? Answer:

=A process is referred to as the execution of a program or command on the operating system.

1. Name three different states a process can be in, and briefly describe each state.

a) State 1 Ready: - It is the state in which a process is ready and waiting for its execution.

b) State 2 Blocked/Waiting: - It is a state in which the process doesn't execute until some event occurs or could be waiting for an Input/Output operation.

c) State 3 Running: - It is a state in which the process is currently being carried out.

1. Which command is used to list processes in Linux? Provide an example of its usage. Command:

= In Linux ps command is used to list processes.

Example:

A black screen with white text

Description automatically generated

1. Explain the meaning of the following columns displayed by the ps command:

a) PID: It stands for the process ID, which helps to find the type pf process.

b) CPU%: It indicates the percentage of CPU used by the process.

c) MEM%: It indicates the percentage of physical memory (RAM) used by

the process.

1. How can you terminate a process in Linux? Describe two different methods.

Method 1: Using kill command and it terminates the process gracefully.

Method 2: Using pkill command and it terminate the program by its name. It terminates the multiple processes with the same name.

1. What is the purpose of the top command in Linux? How can you sort processes using top?

Purpose of top: it is used for monitoring processes and resource usage. It shows show process IDs (PID), memory usage, CPU usage, etc.

Sorting processes in top:

* ‘P’ to sort process by CPU usage
* ‘M’ to sort process by memory usage.
* ‘T’ to sort process by CPU time.
* ‘N’ to sort process by process IDs (PID)

1. Why is it important to exercise caution when terminating processes in Linux? Explain briefly.

= It is important to exercise caution when terminating processes in Linux because terminating processes may cause data loss or corruption that the process was handling and killing certain processes may lead to system crashes or freezes as some are necessary for the operating system to function properly.

1. Briefly explain the difference between the kill and killall commands in Linux.

Kill= it stops the process by its process IDs (IDS)

Killall = It stops the process forcefully stops a punch of processes when a specific name matches.

1. True or False: Terminating a process with SIGKILL allows it to perform cleanup operations before termination.

Answer: False

1. Name two signals that can be sent to a process using the kill command, and briefly describe their effects.

Signal 1 SIGTERM: This signal terminates the process gracefully as it allows the process to perform cleanup before terminating.

Signal 2 SIGKILL: SIGKILL is a fatal signal as it terminates the process without giving a chance for cleanup.