Lab Tasks

By

Hajira Imran(44594)

Subject: Operating System

Submitted to: Ma’am Kausar

Date:11/11/2024

BSCS SEMESTER – 5

RIPHAH INTERNATIONAL UNIVERSITY

ISLAMABAD, PAKISTA

Task 1:

Which command would you use to find the process ID (PID) of a process named OSLab without running it. After obtaining the PID, which command would you use to kill the process?

Solution:

Finding the Process ID (PID) of a Process Named "OSLab":

1. pgrep OSLab:

The pgrep command is used to search for processes based on their names. When you run pgrep OSLab, it checks for any running processes that match the name "OSLab" and returns their Process ID (PID).

Killing the Process after Finding the PID:

2. kill -l:

Ps –ef | grep file name

Kill -9 PID

Once you have the PID (e.g., from the pgrep output), you can use the kill command to send a termination signal to that process. The kill command doesn't necessarily "kill" the process immediately; it sends a signal, and by default, the signal is SIGTERM (terminate), which gracefully stops the process.

Syntax: kill -9 PID

Task 2:

How would you write a script that uses a signal trap to handle specific signals, and what is the purpose of a signal trap in such a script?

Solution:

Signal Trap in a Script

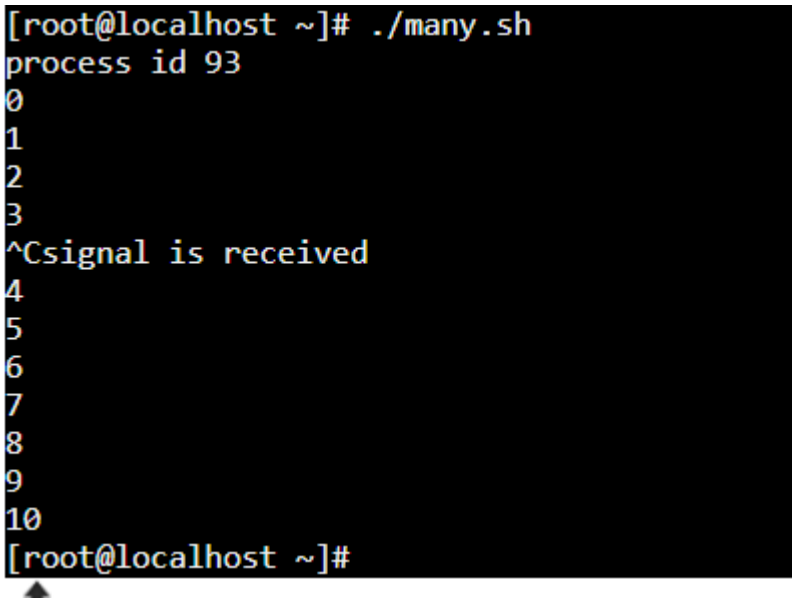
A signal trap in a script is used to catch and handle specific signals that are sent to a process. Signals are notifications sent to a process to tell it to perform some action (e.g., terminate, stop, or pause). By using a signal trap, you can control how the process reacts to these signals instead of letting the process terminate or behave in an unintended way.

Purpose of a Signal Trap:

Custom Signal Handling: Lets you define actions when a signal is received, such as cleaning up files or saving data, rather than letting the process handle it by default.

Prevent Unintended Termination: Helps stop the process from automatically quitting when it receives signals like SIGINT, allowing you to control the response.

Graceful Shutdown: Enables the script to handle signals like SIGTERM, so it can clean up resources or save progress before it ends.


[root@localhost ~]# ./many.sh
process id 93
0
1
2
3
^Csignal is received
4
5
6
7
8
9
10
[root@localhost ~]#

```
#!/bin/bash

echo "process id $$"
trap "echo signal is received" 2
count=0
while ((count<=10))
do
echo $count
((count++))
sleep 2
done
trap 2
exit 0

~
~
~
~
~
~
~
~
~
~
~
~
~
```

```
[root@localhost ~]# ./many.sh
process id 105
0
1
2
3
4
^Z
[1]+  Stopped(SIGTSTP)        ./many.sh
[root@localhost ~]#
```