37. Give commands for finding process ID.

PS:

The most common way to find out the Linux PID is to use the ps command: ps aux | grep name\_of\_process

In addition, the command will also display the PID for grep, because the process was started during the search. To remove it, add the following filter: ps aux | grep name\_of\_process | grep -v grep

Command PGREP:

If you do not need to see detailed information about the process, but only the PID is enough, you can use the pgrep utility:

PIDOF

The pidof command searches for the PID of a particular process by its name. No occurrences, the process name should only match the desired one:

Using the -s option, you can ask the command to display only one PID:

PSTREE:

The pstree command allows you to view a list of child processes for a specific process, as well as their pid identifiers. For example, look at the Apache process tree:

How to edit, rename and move file in Linux?

cat <file name>

This is the most popular and easy way to display the file content. It simply prints the file content to the terminal. It provides many options to make it more specific. To go in-depth with cat command, visit Linux cat

Rename a file

To rename a file there are other commands also like **'mv'**. But **'rename'** command is slightly advanced then others. This command will be rarely used and it works differently on different distros of linux. We'll work on Debian/Ubuntu examples.

Generally, renaming is not a big task, but when you want to rename a large group of files at once then it will be difficult to rename it with 'mv' command. In these cases, it is adviced to use 'rename' command. It can convert upper case files to lower case files and vice versa and cn overwrite files using perl expressions. This command is a part of perl script.

1. rename 's/old-name/new-name/' files
2. mv /old file name /new file name.

move file in Linux

mv file1(file to be moved.) file2(to the file where it needs to be moved)

Q3) Give 5 commands in Linux with explanation?

Sol

**1. pwd command**

Use the **pwd** command to find out the path of the current working directory (folder) you’re in. The command will return an absolute (full) path, which is basically a path of all the directories that starts with a forward slash **(/)**. An example of an absolute path is **/home/username**.

### 2. cd command

To navigate through the Linux files and directories, use the **cd** command. It requires either the full path or the name of the directory, depending on the current working directory that you’re in.

Let’s say you’re in **/home/username/Documents** and you want to go to **Photos**, a subdirectory of **Documents**. To do so, simply type the following command: **cd** **Photos**.

Another scenario is if you want to switch to a completely new directory, for example,**/home/username/Movies**. In this case, you have to type **cd** followed by the directory’s absolute path: **cd /home/username/Movies**.

There are some shortcuts to help you navigate quickly:

* **cd ..** (with two dots) to move one directory up
* **cd**to go straight to the home folder
* **cd-** (with a hyphen) to move to your previous directory

On a side note, Linux’s shell is case sensitive. So, you have to type the name’s directory exactly as it is.

**3. ls command**

The**ls** command is used to view the contents of a directory. By default, this command will display the contents of your current working directory.

If you want to see the content of other directories, type **ls** and then the directory’s path. For example, enter **ls** **/home/username/Documents** to view the content of **Documents**.

There are variations you can use with the **ls** command:

* **ls -R** will list all the files in the sub-directories as well
* **ls -a** will show the hidden files
* **ls -al** will list the files and directories with detailed information like the permissions, size, owner, etc.

**4. cat command**

**cat** (short for concatenate) is one of the most frequently used commands in Linux. It is used to list the contents of a file on the standard output (sdout). To run this command, type **cat** followed by the file’s name and its extension. For instance: **cat file.txt**.

Here are other ways to use the **cat** command:

* **cat > filename**creates a new file
* **cat filename1 filename2>filename3**joins two files (1 and 2) and stores the output of them in a new file (3)
* to convert a file to upper or lower case use, **cat filename | tr a-z A-Z >output.txt**

### 5. cp command

Use the **cp** command to copy files from the current directory to a different directory. For instance, the command **cp scenery.jpg** **/home/username/Pictures** would create a copy of **scenery.jpg** (from your current directory) into the **Pictures** directory.

40)

Sol

## Deadlock Detection

Deadlock can be detected by the resource scheduler as it keeps track of all the resources that are allocated to different processes. After a deadlock is detected, it can be handed using the given methods −

* All the processes that are involved in the deadlock are terminated. This approach is not that useful as all the progress made by the processes is destroyed.
* Resources can be preempted from some processes and given to others until the deadlock situation is resolved.

## Deadlock Prevention

It is important to prevent a deadlock before it can occur. So, the system checks each transaction before it is executed to make sure it does not lead to deadlock. If there is even a slight possibility that a transaction may lead to deadlock, it is never allowed to execute.

Some deadlock prevention schemes that use timestamps in order to make sure that a deadlock does not occur are given as follows −

* **Wait - Die Scheme**
* In the wait - die scheme, if a transaction T1 requests for a resource that is held by transaction T2, one of the following two scenarios may occur −
  + TS(T1) < TS(T2) - If T1 is older than T2 i.e T1 came in the system earlier than T2, then it is allowed to wait for the resource which will be free when T2 has completed its execution.
  + TS(T1) > TS(T2) - If T1 is younger than T2 i.e T1 came in the system after T2, then T1 is killed. It is restarted later with the same timestamp.
* **Wound - Wait Scheme**
* In the wound - wait scheme, if a transaction T1 requests for a resource that is held by transaction T2, one of the following two possibilities may occur −
  + TS(T1) < TS(T2) - If T1 is older than T2 i.e T1 came in the system earlier than T2, then it is allowed to roll back T2 or wound T2. Then T1 takes the resource and completes its execution. T2 is later restarted with the same timestamp.
  + TS(T1) > TS(T2) - If T1 is younger than T2 i.e T1 came in the system after T2, then it is allowed to wait for the resource which will be free when T2 has completed its execution.

## Deadlock Avoidance

It is better to avoid a deadlock rather than take measures after the deadlock has occurred. The wait for graph can be used for deadlock avoidance. This is however only useful for smaller databases as it can get quite complex in larger databases.

**Wait for graph**

The wait for graph shows the relationship between the resources and transactions. If a transaction requests a resource or if it already holds a resource, it is visible as an edge on the wait for graph. If the wait for graph contains a cycle, then there may be a deadlock in the system, otherwise not.