LINUS TASK

Task1: Flavors of Linux?

There are more than 600 Linux Distributions that are actively maintained. The very common for all these is the same Linux Kernel. Each distro has its own uniqueness in the Desktop environment, apps, package managers, and project goals.

Some of the commonly used Linux Distros are Ubuntu, Fedora, CentOS, Debian, Arch Linux, Linux Mint, openSUSE, Kali Linux, Manjaro, Gentoo, Elementary OS, Slackware, RHEL (Red Hat Enterprise Linux), MX Linux, Zorin OS.

Task2: Most used Linux commands in real time (Google)?

1. Ls: "List" - Lists the contents of a directory.

2. Cd: "Change Directory" - Moves you from one directory to another.

3. Grep: "Global Regular Expression Print" - Searches for text within files using regular expressions.

4. Cat: "Concatenate" - Displays the contents of a file.

5. Cp: "Copy" - Copies files or directories from one location to another.

6. Mkdir: "Make Directory" - Creates a new directory.

7. Pwd: "Print Working Directory" - Shows the current working directory.

8. Rm: "Remove" - Deletes files or directories.

9. Touch: Creates an empty file or updates the access and modification times of a file.

10. Head: Displays the beginning lines of a file.

11. Rmdir: "Remove Directory" - Deletes an empty directory.

12. Tail: Displays the last lines of a file.

13. Tar: "Tape Archive" - Used to create or extract files from an archive file.

14. Find: Searches for files and directories within a directory hierarchy.

15. Chmod: "Change Mode" - Modifies file permissions.

16. Mv: "Move" - Moves or renames files or directories.

17. Sort: Sorts lines of text files.

18. File: Determines the type of a file.

19. Mv: "Move" - Moves or renames files or directories.

20. Df: "Disk Free" - Shows disk space usage.

21. Echo: Displays text or variables.

22. Sed: "Stream Editor" - Processes and edits text.

23. Unzip: Extracts files from a ZIP archive.

24. Less: "Less is More" - Allows browsing of text files, providing backward and forward navigation.

Task3: Finding particular line in one file?

In Linux, there are several ways to read a particular line from an input file. Here are some common methods:

Using head and tail: You can combine head and tail commands to read a specific line. For example, to read the 10th line of a file named file.txt, you can use:

head -n 10 file.txt | tail -n 1

Using sed: The sed command can also be used to extract a specific line. For example, to read the 10th line of file.txt:

sed -n '10p' file.txt

Using awk: awk can be utilized to print a specific line. For instance, to read the 10th line of file.txt:

awk 'NR==10' file.txt

Using grep with line numbers: You can use grep with the -n option to display line numbers and then filter the desired line. For example, to read the 10th line of file.txt:

grep -n '' file.txt| grep '^10:' | cut -d: -f2-

Using tail with head: You can also use tail with the -n + option to start from a specific line and then use head to read the first line. For example, to read the 10th line of file.txt:

tail -n +10 file.txt | head -n 1

Task4 : sed?

The sed command, short for "stream editor," is a powerful text manipulation tool in Unix and Linux environments to edit files quickly and efficiently. The tool searches through, replaces, adds, and deletes lines in a text file without opening the file in a text editor.

Substitution (s): Replaces occurrences of a pattern within the text.

sed 's/old/new/' file.txt

Print (p): Prints lines that match a specified pattern.

sed -n '/pattern/p' file.txt

Delete (d): Deletes lines that match a specified pattern

sed '/pattern/d' file.txt

Insert (i): Inserts text before a specified line number or pattern.

sed '3i\This is a new line' file.txt

Append (a): Appends text after a specified line number or pattern.

sed '/pattern/a\This is a new line' file.txt

Substitution with global flag (g): Replaces all occurrences of a pattern within the text.

sed 's/old/new/g' file.txt

Print specific line (p) or range of lines (start,end)

sed -n '5p' file.txt

sed -n '5,10p' file.txt

Read commands from a file (-f option): Executes sed commands from a file.

sed -f script.sed file.txt

These are just a few examples of what sed can do. It's an extremely versatile tool and can be combined with other Unix commands to perform complex text manipulation tasks efficiently.

Task5: I have a file if the file contain the word of error need to print?

You can use the grep command to search for the word "error" in a file and print the lines containing that word. Here's how you can do it:

grep "error" filename

Replace filename with the name of your file. This command will print all lines in the file that contain the word "error".

If you want to ignore the case (i.e., search for both "error" and "Error"), you can use the -i option:

grep -i "error" filename

This will print all lines in the file that contain the word "error" regardless of the case.

If you want to also display the line numbers where the word "error" occurs, you can use the -n option:

grep -in "error" filename

This will print lines in the file along with their line numbers that contain the word "error".

Task6: chown (user id) (file name) change ownership of the file owner

The chown command in Linux stands for "change owner" and is used to change the ownership of files and directories. It allows you to change both the user and group ownership of a file or directory.

The basic syntax for the chown command is:

chown [options] new_owner:group file_name

options: Optional flags that modify the behavior of the command.

new_owner: The new owner of the file or directory. This can be specified as either a user name or a numeric user ID (UID).

group: The group that the file or directory should belong to. This can be specified as either a group name or a numeric group ID (GID).

file_name: The name of the file or directory whose ownership is being changed.

For example, to change the ownership of a file named "example.txt" to a user named "john" and a group named "users," you would use:

chown john:users example.txt

Or, if you prefer to use user and group IDs, you could use:

chown 1001:1000 example.txt

Task7: chown -R (user id) (dir name) change ownership of the dir owner

The `chown` command with the `-R` option is used to recursively change the ownership of directories and their contents. This is particularly useful when you want to change ownership for an entire directory tree, including all files and subdirectories within it.

The basic syntax for the `chown` command with the `-R` option is:

chown -R [options] new_owner:group directory_name

- `-R`: Recursively change ownership of directories and their contents.

- `options`: Optional flags that modify the behavior of the command.

- `new_owner`: The new owner of the directory and its contents. This can be specified as either a user name or a numeric user ID (UID).

- `group`: The group that the directory and its contents should belong to. This can be specified as either a group name or a numeric group ID (GID).

- `directory_name`: The name of the directory whose ownership is being changed.

For example, to recursively change the ownership of a directory named "example" and all its contents to a user named "john" and a group named "users," you would use:

chown -R john:users example

Or, if you prefer to use user and group IDs, you could use:

chown -R 1001:1000 example

Task8: chgrp (Grp id) (file name) change ownership of the group

The `chgrp` command in Linux is used to change the group ownership of files and directories. It allows you to specify a new group for a particular file or directory without affecting its user ownership.

The basic syntax for the `chgrp` command is:

chgrp [options] new_group file_name

- `options`: Optional flags that modify the behavior of the command.

- `new_group`: The new group that the file or directory should belong to. This can be specified as either a group name or a numeric group ID (GID).

- `file_name`: The name of the file or directory whose group ownership is being changed.

For example, to change the group ownership of a file named "example.txt" to a group named "users," you would use:

chgrp users example.txt

Or, if you prefer to use a numeric group ID, you could use:

chgrp 1000 example.txt

This command will change the group ownership of the file "example.txt" to the specified group.

Task9: chgrp (Grp id) (dir name) change ownership of the group dir

The `chgrp` command is used to change the group ownership of files and directories. If you want to change the group ownership of a directory, you can use the following syntax:

chgrp -R new_group directory_name

- `-R`: Recursively change ownership of directories and their contents.

- `new_group`: The new group that the directory and its contents should belong to. This can be specified as either a group name or a numeric group ID (GID).

- `directory_name`: The name of the directory whose group ownership is being changed.

For example, to change the group ownership of a directory named "example" and all its contents to a group named "users," you would use:

chgrp -R users example

Or, if you prefer to use a numeric group ID:

chgrp -R 1000 example

This command will change the group ownership of the directory "example" and all its contents, including subdirectories and files, to the specified group.

Task10: How to do permanent mount volume

To perform a permanent mount of a volume in Linux, you typically need to edit the **`/etc/fstab`** file, which stands for "file system table". This file contains information about all available file systems and how they should be mounted.

Here's a general outline of the steps:

1. Identify the volume: Determine the device name or UUID of the volume you want to mount. You can use commands like **`lsblk`** or `blkid` to list block devices and their UUIDs.

2. Choose a mount point: Decide where you want to mount the volume. This is typically a directory within the file system.

3. Edit **`/etc/fstab`:** Open the **`/etc/fstab`** file in a text editor with root privileges (e.g., `sudo nano /etc/fstab`).

4. Add an entry for the volume: Add a new line to the `/etc/fstab` file specifying the device or UUID of the volume, the mount point, filesystem type, mount options, and backup options.

Here's a general format for an entry:

UUID=<UUID> <mount_point> <filesystem_type> <mount_options> <dump> <pass>

For example:

UUID=12345678-1234-1234-1234-1234567890ab /mnt/data ext4 defaults 0 2

- `<UUID>`: The UUID of the volume.

- `<mount_point>`: The directory where the volume will be mounted.

- `<filesystem_type>`: The files ystem type of the volume (e.g., ext4, ntfs).

- `<mount_options>`: Mount options such as `defaults`, `rw` (read-write), `ro` (read-only), `noauto`, `user`, etc.

- `<dump>`: Backup frequency. Typically set to `0` for non-root filesystems.

- `<pass>`: Used by the `fsck` utility to determine the order in which file system checks are performed at boot time. Typically set to `2` for non-root file systems.

5. Save and exit the file: Save your changes and exit the text editor.

6. Test the configuration: You can test the **`/etc/fstab`** configuration by running:

sudo mount -a

This command will attempt to mount all filesystems listed in **`/etc/fstab`.**

7. Verify the mount: Check if the volume is mounted correctly by inspecting the mount point with **`df -h` or `mount`.**

The volume should now be mounted automatically every time the system boots.

Task11: Types of Zip?

Types of zip formats:

1. **ZIP**: The standard and widely used compressed archive format, supporting lossless data compression.

2. **ZIPX**: An extended version of ZIP with better compression ratios and additional compression methods.

3. **7z**: A high-compression format created by 7-Zip software, supporting various compression methods like LZMA, LZMA2, and BZIP2.

4. **RAR**: Developed by RARLAB, known for creating multi-volume archives and advanced features like password protection and error recovery.

5. **Gzip:** A file compression tool commonly used in Unix/Linux environments, often combined with tar to create tarballs (`tar.gz` or `.tgz`).

6. **Tar**: A file archiving format used to collect multiple files into a single archive, often combined with gzip or other compression tools to create compressed tar archives (`tar.gz` or `.tgz`).