Module: 1 - Linux server - Understand and use essential tools

1. What is the minimum number of partitions you need to install Linux?

➔ To install Linux, the minimum number of partitions you need is **two**:

1. **Root partition ("/")**: This is where the core files of the operating system will be installed.
2. **Swap partition**: This acts like virtual memory. It's used when our physical RAM gets full.

These two are the minimum for a basic Linux installation, but there are more options we can use for organization and performance (like separate partitions for home or boot), but these two are the essential ones.

2. . Explain About Chmod Command

➔ The `chmod` command in Linux is used to **change the permissions** of files or directories. Permissions determine who can read, write, or execute a file. We can control these permissions for three types of users:

1. **Owner (User)**: The person who owns the file.
2. **Group**: Users who are in the same group as the file.
3. **Others**: Everyone else.

Permissions are represented by three letters:

- **r** (read): Allows viewing the content of the file.
- **w** (write): Allows modifying the file.
- **x** (execute): Allows running the file if it's a program.

How `chmod` works:

- We can change the permissions by using either **letters** or **numbers**.

Using letters:

- `chmod u+x file.txt` → Adds **execute** permission for the **owner** (u stands for user/owner).
- `chmod g-w file.txt` → Removes **write** permission for the **group** (g stands for group).
- `chmod o+r file.txt` → Adds **read** permission for **others** (o stands for others).

Using numbers:

Permissions can also be set using numbers, where:

- **r = 4**, **w = 2**, **x = 1**.
- The numbers are added together to represent different combinations of permissions.

For example:

- `chmod 755 file.txt` means:

    - **Owner** gets to read, write, and execute (7 = 4+2+1).
    - **Group** and **Others** get read and execute permissions (5 = 4+1).

In simple terms, `chmod` is used to decide who can do what with a file or folder!

3. How to check Linux memory utilization

➔ To check memory utilization in Linux, we can use a few simple commands. Here are the most common ones:

1. `free` Command

This is the easiest way to check memory usage.

➔ Just type:
```
free -h
```
- The `-h` flag makes the output easier to read by displaying memory in human-readable units (like MB or GB).
  we'll see:
    - **Total**: The total memory available.
    - **Used**: The memory that's currently in use.
    - **Free**: The unused memory.
    - **Buffers/Cache**: Memory used for caching and buffers, which can be freed if needed.

2. `top` Command

This command gives us a dynamic, real-time view of your system's memory usage, along with CPU and processes.

➔ Just type:
css
Copy
```
top
```
- In the top part of the screen, we'll see memory-related info like:
    - **Mem**: Total memory, used memory, free memory, and more.
    - **Swap**: Information about swap space (virtual memory).

3. `htop` Command

`htop` is a more user-friendly, colored version of `top`. It shows memory usage in a graphical way.

➔ Type:
```
htop
```
- If it's not installed, we can install it with `sudo apt install htop` (on Ubuntu) or `sudo yum install htop` (on CentOS).

We can use these commands to quickly understand how much memory is being used and whether we need to worry about running out of memory!.

4. · Use grep to search for specific patterns in files.

➔ The `grep` command in Linux is used to **search for specific patterns** (like words or phrases) in files. It's really useful when we need to find something quickly in a large file or across multiple files.

Here's how it works in simple terms:

Basic Syntax:

```
grep [pattern] [file]
```

- **[pattern]**: This is the word or phrase you are looking for.
- **[file]**: The file (or files) where we want to search for the pattern.

Examples:

Search for a word in a file: If we want to find the word "apple" in a file called `fruits.txt`, we would use:

```
grep "apple" fruits.txt
```

1. This will show us all the lines in `fruits.txt` that contain the word "apple".

2. Search for a word in multiple files: To search for "apple" in all `.txt` files in the current directory, use:

```
grep "apple" *.txt
```

This will search through all `.txt` files and show any lines that contain the word "apple".

3.**Case-insensitive search**: If we don't care about whether the word is capitalized, use the `-i` option to make the search case-insensitive:

```
grep -i "apple" fruits.txt
```

This will find "apple", "Apple", "APPLE", and so on.

4.**Show line numbers with results**: To see the line numbers where the word appears, add the `-n` option:

```
grep -n "apple" fruits.txt
```

This will show the line number along with the matching line.

5.**Count the number of matches**: If we just want to know how many times the word appears in the file, use the `-c` option:

```
grep -c "apple" fruits.txt
```

```
So in simple terms, grep is a fast and powerful tool to search for
specific words or patterns in files. we can even combine it with
other commands to find exactly what we're looking for!
```