Table Of Content

LINUX

Linux is an open-source, Unix-like operating system kernel originally developed by Linus Torvalds in 1991. It serves as the core component of various Linux distributions (commonly referred to as "distros"), which are complete operating system packages including the Linux kernel along with software applications, libraries, and utilities. Linux is renowned for its stability, security, and flexibility, and it powers a wide range of computing devices, from servers and desktop computers to embedded systems and mobile devices.

Operating System:

An operating system (OS) is a software system that manages computer hardware and provides services and utilities for software applications. It acts as an intermediary between the computer hardware and the user, facilitating the execution of programs, managing resources, and providing a user interface. Key functions of an operating system include process management, memory management, file system management, device management, and user interface management.

Types of Operating Systems:

There are several types of operating systems, each designed to serve specific computing environments and purposes. Some common types include:

1. **Single User, Single Operation Systems (SU-SO):**

- These systems support only one user at a time and allow for the execution of only one task or operation at a time.
- They are commonly found in simple embedded devices or basic single-tasking operating systems.
- Examples include simple microcontroller-based systems or early computing devices.

2. **Single User, Multi Operation Systems (SU-MO):**

- These systems support only one user at a time but allow for the execution of multiple tasks or operations concurrently.

- They are commonly found on personal computers and workstations.

- Examples include modern desktop operating systems like Microsoft Windows, macOS, and various Linux distributions.

3. **Multi User, Multi Operation Systems (MU-MO):**

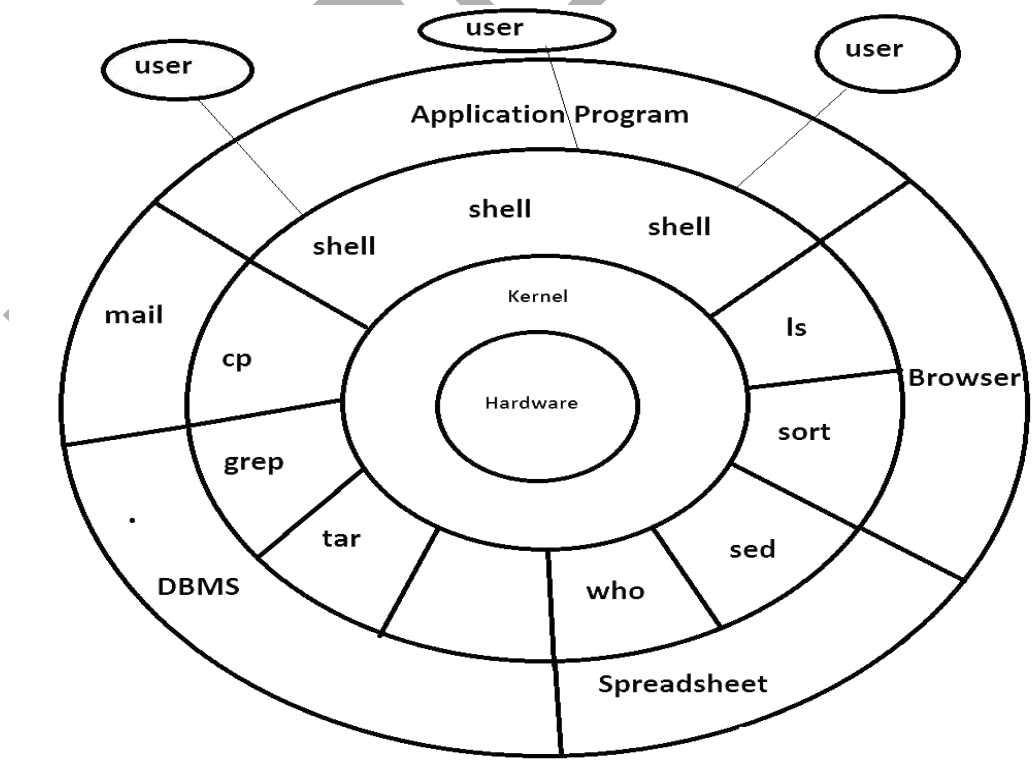
- These systems support multiple users concurrently and allow for the execution of multiple tasks or operations concurrently.

- They are commonly found in server environments or large-scale computing systems where multiple users need simultaneous access to resources.

- Examples include server operating systems like Linux distributions configured for server use, Unix-based systems, and some versions of Windows Server.

Linux Architecture:

The architecture of Linux refers to the way that the various components of the system are organized and interact with each other. At a high level, the architecture of Linux can be divided into the following layers:



Hardware: This is the physical components of the system, such as the CPU, memory, and storage devices.

Kernel: The kernel is the core of the operating system. It is responsible for managing the system's resources, such as memory and CPU time, and for providing a layer of abstraction between the hardware and the rest of the system. The kernel is responsible for tasks such as scheduling processes, handling interrupts, and managing device drivers.

Shell: The shell is a command-line interface that allows users to interact with the system. It provides a way to run commands, navigate the file system, and automate tasks. The most popular shell on Linux is Bash (Bourne Again SHell).

Libraries and system tools: These are collections of pre-written code that provide useful functionality to applications. Libraries can be linked to an application at compile time or runtime, while system tools are standalone programs that can be run from the command line.

Applications: These are the programs that users interact with directly, such as web browsers, text editors, and development tools.
Each of these layers builds on the layer below it, forming a cohesive and powerful operating system. The flexibility of Linux allows it to be used in a wide variety of settings, from small embedded devices to large enterprise servers.

Advantages of Linux

Linux, as an open-source operating system, offers numerous advantages across various domains, including flexibility, security, cost-effectiveness, and community support. Here are some key advantages of Linux:

1. **Open Source:** Linux is open-source software, which means its source code is freely available to anyone. This fosters collaboration, innovation, and customization within the Linux community.

2. **Flexibility and Customizability:** Linux provides users with a high degree of flexibility and customization options. Users can choose from a wide range of distributions (distros) tailored to specific use cases and preferences. Additionally, Linux allows users to customize virtually every aspect of the operating system, from the desktop environment to the kernel.

3. **Stability and Reliability:** Linux is known for its stability and reliability, particularly in server environments. Linux-based servers are renowned for their uptime and robust performance, making them ideal for mission-critical applications and services.

4. **Security:** Linux is inherently more secure than many other operating systems due to its strong security features and architecture. Features such as user permissions, file system encryption, and robust access controls help protect against malware, viruses, and unauthorised access.

5. **Cost-Effectiveness:** Linux is cost-effective, as it is free to use and distribute. Organizations can save significant costs on software licensing fees, making Linux an attractive option for businesses and individuals alike.

6. **Compatibility and Interoperability:** Linux supports a wide range of hardware architectures and platforms, making it highly compatible with diverse hardware configurations. Additionally, Linux seamlessly integrates with other open-source technologies and standards, facilitating interoperability and compatibility with third-party software and services.

7. **Performance:** Linux is renowned for its performance and efficiency, particularly in resource-constrained environments. Linux-based systems often outperform their counterparts in terms of speed, responsiveness, and scalability.

Disadvantages Of Linux:

1.**Lack of Standardization:** The Linux ecosystem lacks a standardized version, leading to fragmentation among different distributions. This can result in compatibility issues between software packages and difficulties in transferring skills between distributions.

2.**Complexity for Novice Users:** Linux is often perceived as less user-friendly compared to other operating systems like Windows or macOS, especially for novice users who may be accustomed to graphical user interfaces (GUIs) and may find the command-line interface (CLI) intimidating.

Linux vs Windows:

1. Cost:

- Linux: Typically free and open-source, with many distributions available at no cost.

- Windows: Requires purchasing licenses for most versions, although some editions may be available for free or at a lower cost for specific use cases.

2. User Interface:

- Linux: Offers various desktop environments with customizable interfaces.

- Windows: Utilizes the Windows Desktop interface with a Start menu and taskbar, providing a consistent user experience across versions.

3. Software Compatibility:

- Linux: Supports a wide range of open-source software and applications available through package managers, but may have limited compatibility with proprietary Windows software.

- Windows: Widely supported by commercial software vendors and has a vast ecosystem of applications, including popular productivity suites and specialized software.

4. Security:

- Linux: Generally considered more secure due to its open-source nature, frequent security updates, and user permissions model.

- Windows: Historically more susceptible to malware and security vulnerabilities, but recent versions have improved security features and updates.

5. Customization and Control:

- Linux: Highly customizable and offers extensive control over system configuration, with options to tailor the operating system to specific needs.

- Windows: Provides less customization options compared to Linux but offers more user-friendly controls and settings.

6. Hardware Support:

- Linux: Supports a wide range of hardware architectures and has drivers included in the kernel for many devices, particularly for servers and embedded systems.

- Windows: Offers broad hardware support for mainstream desktops and laptops, with official drivers available for many devices.

7. Usage Scenarios:

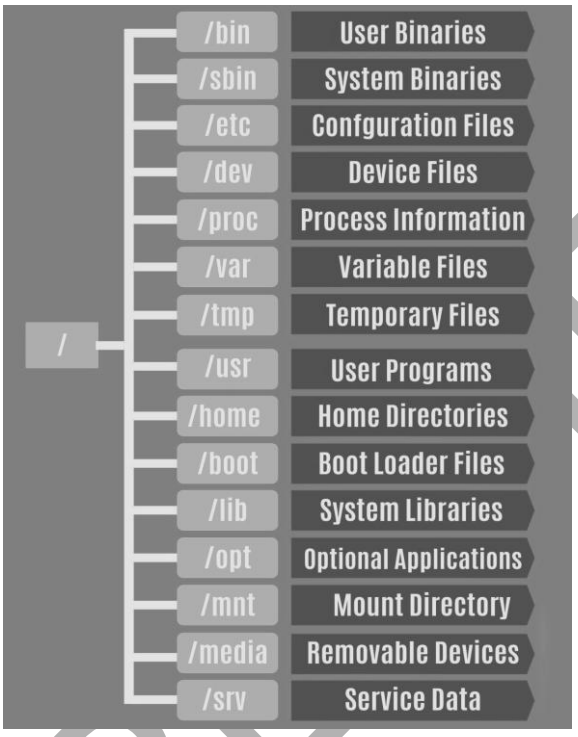
- Linux: Commonly used in server environments, cloud computing, embedded systems, and development environments.

- Windows: Widely used on desktops and laptops for personal computing, gaming, office productivity, and enterprise environments.

In summary, Linux and Windows have distinct characteristics and are suited for different use cases and preferences. Linux offers flexibility, security, and cost savings, while Windows provides ease of use, software compatibility, and broader hardware support. The choice between them often depends on specific requirements, familiarity, and personal preferences.

Linux Folder Structure:

The Linux filesystem follows a hierarchical structure, with directories organized in a tree-like format. Here's an overview of the main directories and their purposes in the Linux filesystem:



1. / (Root Directory):
   - The root directory serves as the top-level directory in the Linux filesystem hierarchy.
   - It contains all other directories and files on the system.
   - Only the root user has write permissions in this directory.

2. /bin (Binaries):
   - The /bin directory contains essential system binaries (executable files) that are required for system operation, such as ls, cp, mv, and mkdir.

3. /boot (Boot Loader Files):
   - The /boot directory contains boot loader files, kernel images, and other files required for the system's boot process.

4. /dev (Device Files):
   - The /dev directory contains device files representing hardware devices on the system, such as hard drives, partitions, terminals, and input/output devices.
   - These device files are used by the system to interact with hardware devices.

5. /etc (Configuration Files):
- The /etc directory contains system-wide configuration files and directories.
- It includes configuration files for various system services, network settings, user authentication, and more.

6. /home (User Home Directories):
- The /home directory contains user home directories, each typically named after the corresponding user.
- User-specific files and settings are stored within their respective home directories.

7. /lib and /lib64 (Shared Libraries):
- The /lib and /lib64 directories contain shared libraries required by programs and libraries on the system.
- These libraries provide common functionality used by multiple programs.

8. /media and /mnt (Removable Media Mount Points):
- The /media and /mnt directories are used as mount points for removable media devices, such as USB drives, optical discs, and network shares.
- Removable media devices are typically mounted under these directories when they are connected to the system.

9. /opt (Optional Software):
- The /opt directory is used for installing optional software packages.
- Third-party software packages and self-contained application directories are often installed under /opt.

10. /proc (Process Information):
- The /proc directory is a virtual filesystem that provides information about running processes and system configuration.

- It contains files and directories representing running processes, kernel configuration parameters, and system information.

11. /root (Root User Home Directory):
- The /root directory is the home directory for the root user (superuser).
- It contains configuration files and settings specific to the root user.

12. /sbin (System Binaries):
- The /sbin directory contains system binaries (executable files) that are essential for system administration tasks.
- These binaries are typically used by the root user for system maintenance and configuration.

13. /srv (Service Data):
- The /srv directory contains data files for system services provided by the system or installed software packages.
- It is used for storing data specific to services running on the system.

14. /tmp (Temporary Files):
- The /tmp directory is used for storing temporary files and directories created by system processes and users.
- Files in this directory are typically deleted automatically when the system reboots.

15. /usr (User Binaries and Data):
- The /usr directory contains user binaries (executable files), libraries, documentation, and other data files used by user applications.
- It is typically divided into subdirectories such as /usr/bin, /usr/lib, /usr/share, and /usr/include.

16. /var (Variable Files):
- The /var directory contains variable files that are expected to change during normal system operation.
- It includes system log files (/var/log), spool directories (/var/spool), and other variable data files.

This is a basic overview of the Linux filesystem structure and its main directories. Each directory serves a specific purpose in organising system files and directories, facilitating system administration and user interactions.

Linux Basic Commands

Here's a list of some basic Linux commands along with a brief description of their functionalities:

- ls: Lists files and directories in the current directory.

- cd: Changes the current directory.
- pwd: Prints the current working directory.
- mkdir: Creates a new directory.
- rm: Removes files or directories.

- cp: Copies files or directories.
- mv: Moves or renames files or directories.
- cat: Displays the contents of a file.
- ctrl+d: to save the file
- touch: to create multiple files
- whoami: Prints the current user.
- date: Displays or sets the system date and time.
- cal : Displays current month calendar

These are just a few examples of basic Linux commands. There are many more commands available, each serving specific purposes for managing and interacting with the Linux system. Familiarizing yourself with these commands is essential for effectively using and administering a Linux system.

Editors:

Editors in Linux are software tools used for creating and editing text files, configuration files, scripts, and code. They play a crucial role in system administration, software development, and everyday computing tasks.

Types Of Editors:

Vi (Visual Editor):

- Vi is a classic text editor originally developed in the 1970s for Unix-like operating systems.
- It provides powerful text editing capabilities and is highly efficient for editing files directly from the command line.

Vim (Vi Improved):
- Vim is an enhanced and modernized version of Vi, offering additional features and improvements.
- It maintains compatibility with Vi's commands and modes while providing a richer set of functionalities.

- Vim is highly configurable, extensible, and popular among Linux users and developers for its versatility and efficiency.

Types Of Modes:

1. **Command Mode:**
   - Command Mode is the default mode when you open Vi or Vim.
   - It allows you to navigate the file, perform editing operations, and execute commands.
   - You can move the cursor, delete(dd), copy(yy), paste(p), undo(u), search, replace, and more using keyboard shortcuts.

2. **Insert Mode:**
   - Insert Mode allows you to directly insert and edit text within the file.
   - You can enter Insert Mode by pressing `i` (for insert)
- In Insert Mode, you can type text as you would in any other text editor.

3. **Extended Command Mode:**
   - Extended Command Mode, also known as Command-Line Mode, lets you enter commands to perform various tasks.
   - You can enter Extended Command Mode by pressing `esc` from insert or Command Mode.
   - Here, you can save changes (`:w`), quit the editor (`:q`), , and force the command to execute, ignoring any warnings or restrictions. (`:!`).

Filter Commands:

Filter commands in Linux are used to process text input and produce output based on specific criteria or transformations. They enable users to manipulate,

extract, or filter data from files or command outputs, facilitating various text processing tasks efficiently.

- **head:** To display a specified number of lines from the top of the file.
- **tail:** To display a specified number of lines from the bottom of the file.

- **less:**Display content page by page. Less command allows users to scroll both forward and backward within the text file using arrow keys or other navigation keys.
- **more:** Display content page by page. More command does not allow users to scroll backward once they have advanced past a portion of the text.

Find

The find command in Linux is used to search for files and directories. It is a versatile tool that allows users to locate files or directories or users or groups based on their names

find / - name filename
find / - name foldername
find / - name groupname
find / - name username

File Permissions:

File permissions in Linux are a crucial aspect of the operating system's security model, determining who can access, modify, or execute files and directories. These permissions are primarily based on three categories of users: the file owner, the group associated with the file, and all other users on the system.

In Linux, file permissions are divided into three levels of access control, user (owner) level, group level, and other level. These levels determine who can access, modify, or execute files and directories on the system.

1. **User (Owner) Level:**
   - The user level refers to the permissions assigned to the owner of the file or directory.
   - Every file and directory in Linux has an associated user owner, which identifies the user account that owns the file or directory.

- The user owner has special privileges based on the permissions assigned to them. These privileges include the ability to read, write, and execute the file or directory.
- The user owner can also change the file's permissions, transfer ownership, and perform administrative tasks related to the file or directory.

2. **Group Level:**
- The group level refers to the permissions assigned to a specific group of users who share ownership of the file or directory.
- In addition to user ownership, each file and directory is associated with a group owner, which identifies a specific group of users.
- Group ownership allows multiple users belonging to the same group to access the file or directory with a common set of permissions.

- The group owner has privileges based on the permissions assigned to the group owner, which may be different from the user owner's permissions.
- Group ownership is useful for facilitating collaboration among users working on shared files or projects.

3. **Other Level:**
- The other level, also known as the world or everyone level, refers to permissions for all other users who are not the owner of the file or directory and do not belong to the group owner.
- It represents users who are not the owner of the file or directory and do not belong to the group associated with it.
- The other level permissions determine what actions users outside the user owner and group owner can perform on the file or directory.
- These permissions are relevant for users who are not the owner of the file or directory and do not belong to the group associated with it.

Each level of access control consists of three types of permissions: read (r), write (w), and execute (x). These permissions are assigned separately for the user owner, group owner, and other users. Properly configuring these permissions and ownership levels is essential for maintaining the security and integrity of the Linux file system.

Here's how file permissions work in Linux:

1. **Read Permission (`r`):** Allows a user to view the contents of a file.

2. **Write Permission (`w`):** Allows a user to modify the contents of a file, including adding, deleting, or modifying data within the file.

3. **Execute Permission (`x`):** Allows a user to execute (run) the file as a program or script. For directories, it allows the user to access the contents of the directory and navigate through it.

If a file has the executable permission (`x`), it means that the user has permission to run the file as a program or script. This permission is separate from the ability to modify the file's contents.

To modify a file, a user needs write (`w`) permission on that file. Without write permission, a user cannot make changes to the file, regardless of whether the file is executable or not.

Therefore, having the executable permission (`x`) on a file does not automatically grant permission to modify the file. Write permission (`w`) is

required for modifying the contents of a file, while execute permission (`x`) is necessary to run it as a program or script.

The permission string - --- --- --- can be interpreted as follows:

The first character (-) indicates that file is a regular file (not a directory or special file).
The next three characters (---) represent the permissions for the user owner (user). Since all permissions are set to ---, the user owner has no permissions.
The next three characters (---) represent the permissions for the group owner (user group). Similarly, all permissions are set to ---, meaning the group owner has no permissions.
The last three characters (---) represent the permissions for other users. Once again, all permissions are set to ---, indicating that other users have no permissions.

To see file permissions in Linux, you can use the **ls** command with the **-l** option, which displays detailed information about files and directories, including their permissions

To change the permissions of a file or directory in Linux, you can use the **chmod** command. The **chmod** command allows you to modify the permissions for the user owner, group owner, and other users.

Here are some examples of using `chmod`:

1. **Symbolic Representation:**
   - To grant read, write, and execute permissions to the user owner, and only read and execute permissions to the group owner and other users:

```
chmod u=rwx,g=rx,o=rx file.txt
```

2**. Numeric Representation:**
   - To grant read, write, and execute permissions to the user owner (7), and only read and execute permissions to the group owner (5) and other users (5):

```
chmod 755 file.txt
```

Remember to replace `file.txt` with the name of the file or directory you want to modify.

Additionally, if you want to recursively change permissions for all files and directories within a directory, you can use the `-R` option:

chmod -R [permissions] [directory]

For example:

chmod -R 755 directory

This command will recursively grant read, write, and execute permissions to the user owner, and only read and execute permissions to the group owner and other users, for all files and directories within the specified `directory`.

User Management:

User management in Linux involves creating, modifying, and deleting user accounts, as well as managing user permissions and access rights. This process is crucial for maintaining system security and controlling access to resources.

Creating Users:

The useradd command is used to create new user accounts in Linux.

sudo useradd username

This command creates a new user account named username with default settings.

Setting Passwords:
After creating a user account, a password should be set for the user to authenticate and log in.

sudo passwd username
This command prompts you to enter and confirm the password for the specified user account.

Linux families:

1. **Debian Family:**
   - Emphasizes stability, security, and free and open-source software principles.
   - Uses the Advanced Package Tool (APT) for package management.
   - Includes distributions like Ubuntu, Fedora, AVLinux, GRML.
   - Follows a stable release model with long development cycles.
   - Has a large and diverse community of developers and users.

2. **Red Hat Family:**
   - Designed for enterprise use cases with a focus on stability and security.
   - Uses the RPM Package Manager (RPM) for package management.
   - Includes distributions like CentOS,SuseLinux, Amazon Linux.
   - Follows stable release models with long-term support versions.
   - Provides commercial support for Red Hat Enterprise Linux (RHEL).

Both families offer a variety of distributions suited for different purposes, but they differ in their package management systems, release models, and support offerings. Users often choose between them based on their specific requirements and preferences.

Deployment Commands:

Deployment commands in Linux are used to install, configure, and manage software applications, services, or configurations on a server or computing environment. These commands facilitate the deployment process by automating tasks such as package installation, service management, and configuration updates.

Package Management Commands:

APT (Advanced Package Tool) Commands (Debian/Ubuntu):

apt-get: A command-line tool used for handling packages on Debian-based systems. It provides options to install, upgrade, and remove software packages, as well as to manage package repositories.

Example:
sudo apt-get install package_name

apt-get/apt: Used in Debian-based distributions to install, update, and manage software packages.

apt-get install java -y
apt-get install openjdk-8* -y
java --version

YUM (Yellowdog Updater Modified) Commands (Red Hat/CentOS):

yum: A command-line package management utility used in Red Hat-based distributions to install, remove, and update software packages. It handles dependency resolution automatically.

Example: sudo yum install package_name

yum install java -y
yum install git -y
git --version

To download packages from a URL onto a Linux server, you can use various command-line utilities such as `wget` or `curl`. Here's how to do it using each of these utilities:

1. **Using `wget`:**
   - `wget` is a command-line utility for downloading files from the web. Here's the basic syntax to download a file from a URL:

     wget [URL]

   - Replace `[URL]` with the actual URL of the package you want to download.
   - Optionally, you can specify a different name for the downloaded file using the `-O` option:

     wget -O [output_file] [URL]

     Replace `[output_file]` with the desired filename.
   Example:

   wget https://example.com/package.tar.gz

2. **Using `curl`:**
   - `curl` is another command-line utility for transferring data from or to a server. Here's the basic syntax to download a file from a URL:

     curl -O [URL]

   - Replace `[URL]` with the actual URL of the package you want to download.
   - Similar to `wget`, you can specify a different name for the downloaded file using the `-o` option:

     curl -o [output_file] [URL]

     Replace `[output_file]` with the desired filename.

   Example:

   curl -O https://example.com/package.tar.gz

After running one of these commands, the package will be downloaded from the specified URL and saved to the current directory (or the directory specified with `-O` or `-o` option) on your Linux server. You can then proceed to extract or install the package as needed.

To extract tar and zip files in Linux, you can use the `tar` and `unzip` commands, respectively. Here's how to do it for each format:

1. Extracting Tar Files:

- To extract a `.tar` file:

tar -xvf file.tar

This command will extract the contents of the `file.tar` archive in the current directory.

- To extract a compressed tarball (e.g., `.tar.gz`, `.tar.bz2`, `.tar.xz`):

tar -xzvf file.tar.gz For gzip-compressed tarball

Replace `file.tar.gz` with the appropriate filename of the compressed tarball.

- Options used:
  - `-x`: Extract files from the archive.
  - `-v`: Verbose mode to show progress.
  - `-f`: Specifies the filename of the archive to operate on.

2. Extracting Zip Files:

- To extract a `.zip` file:

unzip file.zip

This command will extract the contents of the `file.zip` archive in the current directory.

- To extract a zip file into a specific directory:

unzip file.zip -d /path/to/directory

Replace `/path/to/directory` with the desired destination directory.

- Options used:

  - `-d`: Specifies the destination directory for extracted files.

After executing the respective command, the contents of the tarball or zip file will be extracted into the specified directory or the current directory if no destination is provided. You can then access and work with the extracted files as needed.

Below Commands should not be used in Production Servers:

1. init 0

- `init 0` is a command used to shut down the system and power it off completely.
- When you run `init 0`, the system transitions to runlevel 0, which is reserved for system halt or shutdown. It stops all processes and then powers off the system.

2. init 1

- `init 1` is a command used to change the system's runlevel to single-user mode or rescue mode.
- When you run `init 1`, the system transitions to runlevel 1, which is also known as single-user mode. It boots the system into a minimal environment with only essential services running and provides a root shell for maintenance and repair tasks.

3. init 6

- `init 6` is a command used to reboot the system.
- When you run `init 6`, the system transitions to runlevel 6, which is dedicated to system reboot. It stops all processes, unmounts filesystems, and then reboots the system.

4. shutdown

- The `shutdown` command is used to initiate system shutdowns or reboots with various options.
- By default, running `shutdown` without any options will schedule a system shutdown after a specified delay.

5. poweroff

- The `poweroff` command is used to power off the system immediately without any delay.
- It is equivalent to running `shutdown -h now` or `init 0`.

Each of these commands provides a way to manage the system's state, whether it's shutting down, restarting, or transitioning to a different runlevel for maintenance or troubleshooting purposes. It's important to use them appropriately based on the desired outcome and the specific requirements of the situation.