Ubuntu

Ubuntu is a popular open-source Linux distribution based on Debian. It's known for its ease of use, regular releases, and large community support. Ubuntu can be used for various purposes, including desktop computing, server management, and development.

Here's how you can use Ubuntu:

Installation:

You can download the Ubuntu ISO image from the official website and install it on your computer either as a standalone operating system or alongside other operating systems using dual-boot.

Desktop Usage:

After installation, you can log in to the Ubuntu desktop environment. Ubuntu provides a user-friendly interface similar to other operating systems like Windows or macOS. You can access applications, browse the internet, manage files, and perform various tasks using the graphical user interface (GUI).

Terminal Usage:

Ubuntu includes a terminal emulator, usually GNOME Terminal, which allows you to interact with the system using command-line instructions. The terminal provides powerful capabilities for tasks such as file management, software installation, system configuration, and automation.

To open the terminal, you can search for "Terminal" in the application launcher or use the keyboard shortcut Ctrl + Alt + T.

1:File management

1. ls (List Directory Contents):

- Syntax: **ls [options] [files or directories]**
- Description: Lists the contents of a directory.
- Example:
- **ls**: Lists files and directories in the current directory.
- **ls -l**: Lists files and directories in long format with detailed information.
- **ls -a**: Lists all files and directories, including hidden ones.
- **ls Documents**: Lists files and directories in the "Documents" directory.

2. cd (Change Directory):

- Syntax: **cd [directory]**
- Description: Changes the current working directory.
- Example:
- **cd Documents**: Changes the current directory to "Documents".
- **cd ..**: Moves up one directory.
- **cd** /: Changes to the root directory.
- **cd** ~: Changes to the user's home directory.

3. pwd (Print Working Directory):

- Description: Displays the current working directory.
- Example:
- **pwd**: Displays the current working directory, e.g., "/home/user/Documents".

4. mkdir (Make Directory):

- Syntax: **mkdir [directory_name]**
- Description: Creates a new directory.
- Example:
- **mkdir new_folder**: Creates a new directory named "new_folder".
- **mkdir -p path/to/new/directory**: Creates nested directories if they don't exist.

5. touch:

- Syntax: **touch [file_name]**
- Description: Creates an empty file.
- Example:
- **touch example.txt**: Creates a new empty file named "example.txt".

6. rm (Remove):

- Syntax: **rm [options] [file(s) or directory(ies)]**
- Description: Removes files or directories.
- Example:
- **rm file.txt**: Deletes a file named "file.txt".

- **rm -r directory_to_remove**: Deletes a directory and its contents recursively.

7. cp (Copy):

- Syntax: **cp [options] [source] [destination]**
- Description: Copies files or directories.
- Example:
- **cp file.txt /path/to/destination**: Copies a file named "file.txt" to the specified destination.
- **cp -r directory_to_copy /path/to/destination**: Copies a directory and its contents recursively to the specified destination.

8. mv (Move/Rename):

- Syntax: **mv [options] [source] [destination]**
- Description: Moves files or directories to a new location or renames them.
- Example:
- **mv file.txt /path/to/destination**: Moves a file named "file.txt" to the specified destination.
- **mv old_name.txt new_name.txt**: Renames a file from "old_name.txt" to "new_name.txt".

9. cat (Concatenate):

- Syntax: **cat [file(s)]**
- Description: Concatenates and displays the content of files.
- Example:
- **cat file.txt**: Displays the content of the file "file.txt".
- **cat file1.txt file2.txt > combined.txt**: Concatenates the content of two files and saves it to a new file named "combined.txt".

10. grep (Global Regular Expression Print):

- Syntax: **grep [options] [pattern] [file(s)]**
- Description: Searches for patterns in files.
- Example:
- **grep "pattern" file.txt**: Searches for the specified pattern in the file "file.txt".
- **grep -r "pattern" /path/to/directory**: Searches for the pattern recursively in all files within the specified directory.

11. chmod (Change Mode):

- Syntax: **chmod [options] [mode] [file(s)]**
- Description: Changes file permissions.
- Example:
- **chmod +x script.sh**: Adds execute permission to the script "script.sh".
- **chmod 644 file.txt**: Sets read and write permissions for the owner and read-only permissions for others on the file "file.txt".

12. chown (Change Owner):

- Syntax: **chown [options] [user:group] [file(s)]**
- Description: Changes the owner and group of files.
- Example:
- **chown user:group file.txt**: Changes the owner and group of the file "file.txt" to the specified user and group.

These commands provide essential functionality for managing files and directories, including copying, moving, renaming, displaying content, searching for patterns, and changing permissions and ownership. Mastering these commands allows for efficient file manipulation and organization in the terminal.

2: System information

1. uname (Print System Information):

- Syntax: **uname [options]**
- Description: Prints basic system information such as the kernel name, network node hostname, kernel release, kernel version, machine hardware architecture, and operating system.
- Example:
- **uname -a**: Displays all system information.
- **uname -r**: Prints the kernel release.
- **uname -m**: Shows the machine hardware architecture.

2. top (Display System Processes):

- Description: Displays real-time information about system processes, including CPU and memory usage.
- Example:
- **top**: Shows a dynamic view of system processes.
- Press **q** to quit.

3. df (Display File System Disk Space Usage):

- Syntax: **df [options] [file or directory]**
- Description: Shows the amount of disk space used and available on mounted filesystems.
- Example:
- **df -h**: Displays disk space in human-readable format.
- **df /dev/sda1**: Shows disk space usage for the specified filesystem.

4. du (Display Disk Usage):

- Syntax: **du [options] [file or directory]**
- Description: Displays disk space usage for files and directories.
- Example:

- **du -h**: Shows disk usage in human-readable format.
- **du -s /path/to/directory**: Displays total disk usage for the specified directory.

5. free (Display Memory Usage):

- Description: Shows the amount of free and used memory in the system.
- Example:
- **free**: Displays memory usage statistics.
- **free -h**: Shows memory usage in human-readable format.

6. uptime (Display System Uptime):

- Description: Shows how long the system has been running.
- Example:
- **uptime**: Displays system uptime along with load average.

7. lscpu (Display CPU Information):

- Description: Prints CPU architecture information.
- Example:
- **lscpu**: Shows detailed information about the CPU.

8. lspci (List PCI Devices):

- Description: Lists all PCI devices connected to the system.
- Example:
- **lspci**: Displays information about PCI devices.

9. lsusb (List USB Devices):

- Description: Lists all USB devices connected to the system.
- Example:
- **lsusb**: Displays information about USB devices.

10. dmidecode (Display System Hardware Information):

- Description: Retrieves hardware information from the system BIOS.
- Example:
- **sudo dmidecode**: Displays detailed hardware information.

3: Package management

1. apt (Advanced Package Tool):

- Syntax: **apt [options] [command] [package]**
- Description: A high-level package management utility for installing, updating, and removing software packages and their dependencies.
- Example:

- **sudo apt update**: Updates the package index to get the latest package information from repositories.
- **sudo apt install package_name**: Installs the specified package.
- **sudo apt remove package_name**: Removes the specified package.
- **sudo apt upgrade**: Upgrades installed packages to their latest versions.

2. dpkg (Debian Package Manager):

- Syntax: **dpkg [options] [command] [package]**
- Description: A low-level package management utility for working directly with Debian package files.
- Example:
- **dpkg -i package.deb**: Installs a local Debian package file.
- **dpkg -r package_name**: Removes the specified package.
- **dpkg -l**: Lists all installed packages.

3. apt-get:

- Syntax: **apt-get [options] [command] [package]**
- Description: A command-line interface for the APT package management system.
- Example:
- **sudo apt-get install package_name**: Installs the specified package.
- **sudo apt-get remove package_name**: Removes the specified package.
- **sudo apt-get update**: Updates the package index.
- **sudo apt-get upgrade**: Upgrades installed packages.

4. apt-cache:

- Syntax: **apt-cache [options] [command] [package]**
- Description: A command-line tool for querying APT's package cache.
- Example:
- **apt-cache search search_term**: Searches for packages matching the specified search term.
- **apt-cache show package_name**: Displays detailed information about the specified package.
- **apt-cache policy package_name**: Shows the installation candidates for the specified package.

5. snap:

- Syntax: **snap [options] [command] [package]**
- Description: A package management system for installing and managing snap packages, which are self-contained software packages.
- Example:
- **sudo snap install package_name**: Installs the specified snap package.
- **sudo snap remove package_name**: Removes the specified snap package.
- **snap find search_term**: Searches for snap packages matching the specified search term.
- **snap list**: Lists installed snap packages.

4: Network

1. ifconfig (Interface Configuration):

- Description: Displays information about network interfaces and configures them.
- Example:
- **ifconfig**: Displays information about all network interfaces.
- **ifconfig eth0**: Displays information about a specific network interface (e.g., eth0).
- **sudo ifconfig eth0 down**: Disables a network interface (e.g., eth0).
- **sudo ifconfig eth0 up**: Enables a network interface (e.g., eth0).

2. ping (Packet Internet Groper):

- Syntax: **ping [options] host**
- Description: Sends ICMP echo requests to a network host to check connectivity.
- Example:
- **ping google.com**: Sends ICMP echo requests to Google's servers.
- **ping -c 5 google.com**: Sends only 5 ICMP echo requests.
- **ping -t 10 google.com**: Sets a timeout of 10 seconds.

3. traceroute (Trace Route):

- Syntax: **traceroute [options] host**
- Description: Traces the route that packets take to reach a network host.
- Example:
- **traceroute google.com**: Traces the route to Google's servers.
- **traceroute -n google.com**: Does not perform DNS resolution for IP addresses.

4. netstat (Network Statistics):

- Syntax: **netstat [options]**
- Description: Displays network connections, routing tables, interface statistics, masquerade connections, and multicast memberships.
- Example:
- **netstat -tuln**: Displays listening TCP and UDP ports.
- **netstat -r**: Displays the kernel routing table.

5. nslookup (Name Server Lookup):

- Syntax: **nslookup [options] host**
- Description: Queries DNS servers to obtain domain name or IP address mapping.
- Example:
- **nslookup google.com**: Performs a DNS lookup for Google's IP address.
- **nslookup 8.8.8.8**: Performs a reverse DNS lookup for the IP address.

6. ssh (Secure Shell):

- Syntax: **ssh [user@]hostname [command]**
- Description: Connects to a remote server securely using the SSH protocol.

- Example:
- **ssh username@hostname**: Connects to a remote server.
- **ssh -p port_number username@hostname**: Specifies a custom SSH port.

7. scp (Secure Copy):

- Syntax: **scp [options] [source] [destination]**
- Description: Securely copies files between hosts on a network.
- Example:
- **scp file.txt user@hostname:/remote/directory**: Copies a local file to a remote server.
- **scp user@hostname:/remote/file.txt /local/directory**: Copies a file from a remote server to the local machine.

8. curl (Client for URLs):

- Syntax: **curl [options] [URL]**
- Description: Transfers data to or from a server.
- Example:
- **curl https://www.example.com**: Downloads the contents of a webpage.
- **curl -O https://www.example.com/file.txt**: Downloads a file and saves it with the same name.

9. wget (Web Get):

- Syntax: **wget [options] [URL]**
- Description: Downloads files from the internet using HTTP, HTTPS, or FTP protocols.
- Example:
- **wget https://www.example.com/file.txt**: Downloads a file from a URL.
- **wget -r -np https://www.example.com**: Downloads recursively without ascending to the parent directory.

These commands are essential for network troubleshooting, configuration, and management in Ubuntu and other Unix-like operating systems. They provide various functionalities, including network interface configuration, connectivity testing, DNS resolution, remote access, and file transfer.

5:Text Editor

1. nano:

- Description: Nano is a simple, easy-to-use command-line text editor.
- Syntax: **nano [filename]**
- Example:
- **nano example.txt**: Opens or creates a file named "example.txt" for editing in the nano editor.
- Inside nano:

- Use arrow keys to navigate.
- Use Ctrl + O to save the file.
- Use Ctrl + X to exit nano.

2. vi / vim (Vi Improved):

- Description: Vi and its improved version Vim are powerful command-line text editors with extensive features.
- Syntax: **vi [filename]** or **vim [filename]**
- Example:
- **vi example.txt** or **vim example.txt**: Opens or creates a file named "example.txt" for editing in vi or vim.
- Inside vi or vim:
- Press **i** to enter insert mode and start typing.
- Press **Esc** to exit insert mode.
- Use **:** followed by a command to execute commands (e.g., **:w** to save, **:q** to quit).

3. emacs:

- Description: Emacs is a highly customizable text editor with a wide range of features and extensions.
- Syntax: **emacs [filename]**
- Example:
- **emacs example.txt**: Opens or creates a file named "example.txt" for editing in Emacs.
- Inside Emacs:
- Use Ctrl + X, Ctrl + S to save the file.
- Use Ctrl + X, Ctrl + C to exit Emacs.

These text editors are commonly used in Ubuntu for editing configuration files, writing scripts, programming, and general text editing tasks. They offer different levels of complexity and customization options to suit various user preferences and requirements.

6:User management

. useradd:

- Syntax: **sudo useradd [options] username**
- Description: Creates a new user account.
- Example:
- **sudo useradd john**: Creates a new user account named "john".
- Options:
- **-m**: Creates the user's home directory if it doesn't exist.
- **-G group1,group2,...**: Adds the user to specified supplementary groups.

2. userdel:

- Syntax: **sudo userdel [options] username**

- Description: Deletes a user account.
- Example:
- **sudo userdel john**: Deletes the user account named "john".
- Options:
- **-r**: Removes the user's home directory and mail spool.

3. passwd:

- Syntax: **passwd [options] [username]**
- Description: Changes user password or sets password policies.
- Example:
- **passwd john**: Changes the password for the user account "john".
- Options:
- **-l**: Locks the password of the specified account.
- **-u**: Unlocks the password of the specified account.
- **-d**: Deletes the password for the specified account.

4. su (Switch User):

- Syntax: **su [username]**
- Description: Switches to another user account.
- Example:
- **su john**: Switches to the user account "john".

5. sudo (Superuser Do):

- Syntax: **sudo [command]**
- Description: Executes a command as another user, typically the superuser.
- Example:
- **sudo apt update**: Updates the package index with superuser privileges.
- **sudo -u john touch file.txt**: Creates a file named "file.txt" as the user "john".

6. groups:

- Syntax: **groups [username]**
- Description: Displays the groups a user belongs to.
- Example:
- **groups john**: Shows the groups the user "john" belongs to.

These user management commands allow you to create, delete, and manage user accounts on an Ubuntu system. They are essential for system administration tasks such as granting or revoking access permissions, managing user privileges, and maintaining user accounts.

7: Miscellaneous:

1. Echo:

- Syntax: **echo [options] [text]**
- Description: Prints text to the terminal.
- Example:
- **echo "Hello, World!"**: Prints "Hello, World!" to the terminal.
- **echo $PATH**: Prints the value of the PATH environment variable.

2. Date:

- Syntax: **date [options]**
- Description: Displays or sets the system date and time.
- Example:
- **date**: Prints the current date and time.
- **date "+%Y-%m-%d"**: Prints the current date in YYYY-MM-DD format.

3. History:

- Description: Displays the command history.
- Example:
- **history**: Lists previously executed commands along with their line numbers.
- **!n**: Executes the command with the line number n from the history.

4. tar (Tape Archive):

- Syntax: **tar [options] [files/directories]**
- Description: Creates or extracts tar archives.
- Example:
- **tar -cvf archive.tar file1 file2**: Creates a new tar archive named "archive.tar" containing "file1" and "file2".
- **tar -xvf archive.tar**: Extracts files from the tar archive "archive.tar".

5. gzip (GNU zip):

- Syntax: **gzip [options] [file(s)]**
- Description: Compresses or decompresses files.
- Example:
- **gzip file.txt**: Compresses "file.txt" to "file.txt.gz".
- **gzip -d file.txt.gz**: Decompresses "file.txt.gz" to "file.txt".

6. find:

- Syntax: **find [path] [options] [expression]**
- Description: Searches for files in a directory hierarchy.

- Example:
- **find /home/user -name "*.txt"**: Searches for all files with a ".txt" extension in the "/home/user" directory.
- **find / -type f -size +100M**: Finds files larger than 100 MB in the entire filesystem.

7. sort:

- Syntax: **sort [options] [file]**
- Description: Sorts lines of text files.
- Example:
- **sort file.txt**: Sorts the lines of "file.txt" alphabetically.
- **sort -n file.txt**: Sorts the lines numerically.

8. awk:

- Syntax: **awk [options] 'pattern {action}' [file(s)]**
- Description: A versatile pattern scanning and processing language.
- Example:
- **awk '{print $1}' file.txt**: Prints the first column of "file.txt".
- **awk '/pattern/ {print $2}' file.txt**: Prints the second column where a line matches the pattern.

9. sed (Stream Editor):

- Syntax: **sed [options] 'command' [file(s)]**
- Description: A stream editor for filtering and transforming text.
- Example:
- **sed 's/search/replace/g' file.txt**: Replaces all occurrences of "search" with "replace" in "file.txt".
- **sed -n '10,20p' file.txt**: Prints lines 10 to 20 of "file.txt".

These miscellaneous commands provide various functionalities for text processing, file manipulation, archiving, and system maintenance tasks. They are essential tools for efficient system administration and shell scripting in Ubuntu and other Unix-like operating systems.

- ## Users in Ubuntu or linux

1. **Root User:** Also known as the superuser, the root user has full access to all files and commands on a Linux system. The root user can perform any operation and has unrestricted control over the system. It is recommended to use the root account sparingly and only when necessary, as incorrect commands or actions executed as root can have serious consequences
   .
2. **System Users**: These are users created by the system for specific services or applications. They often have limited permissions and are used to run specific tasks or services on the system. System users typically do not have login privileges and are assigned unique user IDs (UIDs) for identification purposes.

3. **Regular Users:** These are standard users who interact with the system for everyday tasks. Regular users have restricted access to system files and commands, and they typically have a home directory where they can store their files and configurations. Regular users do not have administrative privileges by default and need to use tools like **sudo** to perform tasks that require elevated permissions.

What is absolute path and relative path to move in directory?

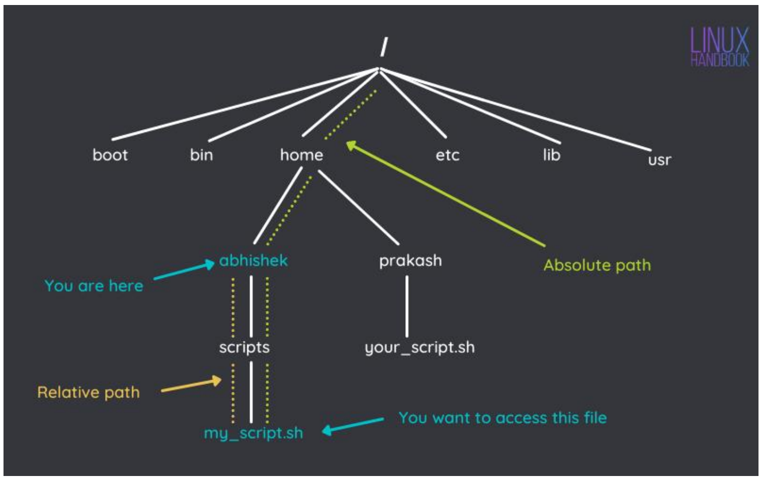
Absolute Path:

- An absolute path specifies the location of a file or directory from the root of the file system.
- It includes the entire directory path starting from the root directory ("/" in Unix-like systems).
- For example, in Unix-like systems, an absolute path might look like: /home/user/Documents/file.txt.
- Absolute paths are not dependent on the current working directory and always refer to the same location regardless of where they are used.

Relative Path:

- A relative path specifies the location of a file or directory relative to the current working directory.
- It does not start from the root directory; instead, it starts from the current directory.
- For example, if the current working directory is /home/user, and you want to refer to a file named file.txt located in the Documents directory within the current directory, you can use the relative path: Documents/file.txt.
- Relative paths are dependent on the current working directory. They specify the location of a file or directory in relation to where the command is being executed.

In summary, absolute paths provide the complete location of a file or directory from the root of the file system, while relative paths specify the location relative to the current working directory. Both types of paths have their uses depending on the context in which they are being used.


/
LINUX HANDBOOK
boot
bin
home
etc
lib
usr
abhishek
prakash
You are here
Absolute path
scripts
your_script.sh
Relative path
my_script.sh
You want to access this file