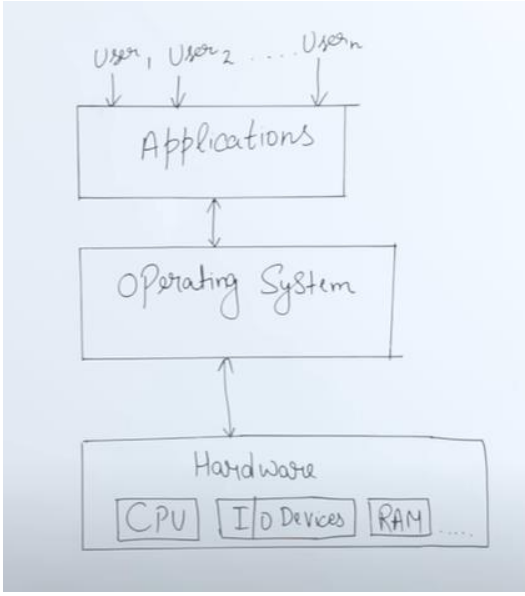
Author – Afiya Mulla **Topic - Linux**

What is an Operating System

Operating System is the software that directly manages a systems hardware and resources like CPU/Memory/Storage

Operating System can be used through 1] Command Line Interface 2] Graphical User Interface.

An Operating System sits between application and hardware , It makes the connections between all our software (application) and physical resources (Hardware) that do work



What is Linux and its Features

- Linux is an open-source operating system, The source code of Linux is freely available to anyone, developed by Linus Torvalds in 1991.
- Linux is known for its command line feature.
- Linux is known for its stability and security. It's widely used in servers and critical infrastructure due to its reliability.
- Linux comes in various distributions such as RedHat, Ubuntu, Fedora, Kali Linux , CentOS.
- Multiple users can access system resources such as memory, hard disk, and applications simultaneously without affecting each other's activities.
    - Linux to allows multiple users to access system resources simultaneously without affecting each other's activities

- Each user on a Linux system has a unique user account. User accounts include a username, user ID (UID), and a home directory. This segregation ensures that users have their own private space for files and settings.

- Processes in Linux run in isolated memory spaces. This means that the processes of one user cannot interfere with the processes of another user. Each user's applications and processes operate independently.

❖ Linux can run multiple programs at the same time. This is essential for server environments and useful for desktop users who need to run various applications concurrently.

- Each program running on a Linux system is a process

- Linux uses a process scheduler to manage the execution of multiple processes.

- scheduler allocates CPU time to each process in an efficient manner.

- Processes are assigned priorities, and the scheduler ensures that higher-priority processes get more CPU time while still allowing lower-priority processes to run.

- The operating system performs context switching to switch the CPU from one process to another. This happens so quickly that users perceive the programs as running simultaneously.

- background processes (data backup, logging, and monitoring etc) can run without affecting the performance of primary services.

- Linux ensures that CPU, memory, and I/O resources are used optimally, maximizing the server's performance.

- Servers often run multiple services such as web servers, database servers, and application servers simultaneously. Linux's ensures that all these services can run concurrently without interfering with each other.

.

❖ Linux can run on a variety of hardware platforms, from the smallest embedded devices to the largest supercomputers, making it extremely versatile.

- Eg - IoT Devices: Many Internet of Things (IoT) devices, like smart home appliances, use Linux-based operating systems due to their lightweight nature and ability to be customized for specific tasks.

- Routers and Switches: Network devices often run on embedded Linux because it
- provides robust networking capabilities and can be optimized for performance.
- Desktop and Laptops :Linux distributions like Ubuntu, Fedora, and Mint are tailored for desktop and laptop use. They come with user-friendly graphical interfaces and support a wide range of hardware:

- Data Centers and Web Servers: Linux powers a significant percentage of the world's web servers, including those running on platforms like Apache, Nginx, and cloud services.

Architecture of Linux

Linux has 2 parts

1. Kernel – It deals with program execution , memory management , Input output operations
2. Set of Commands – It deals with language compilers etc

Where is Linux used

Linux is well suited for all types of server applications

Web Servers

Database Servers

Email Servers

File Servers

IoT Devices: Many Internet of Things (IoT) devices, such as smart home appliances

Mobile Devices -Android: The Android operating system, which is based on the Linux kernel, is used in millions of smartphones, tablets, and other mobile devices worldwide.

Linux vs Windows

Linux	Windows
COST	
Free and open-source.	Proprietary and commercial.
No licensing fees for software installation.	Requires a purchased license for each copy. ,Licensing fees can be significant, especially for businesses.
PERFORMANCE	
Known for high performance and efficiency.	Performance can be more resource-intensive.
Can run on older hardware with minimal resources.	Requires relatively modern hardware to run smoothly.
SECURITY	

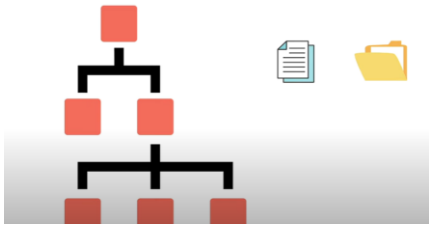
Generally considered more secure due to its open-source nature, which allows for constant security review, If incase it attacks then it will not get spread , it will remain in that particular folder and we can delete that folder to remove it	More targeted by viruses and malware. If virus attacks , It spreads through the entire operating system making it corrupt and needs a format as a solution
Fewer viruses and malware compared to Windows.	Regular updates and security patches are provided, but it can be vulnerable if not kept up-to-date.
Strong permission and user role features.	Security features have improved significantly with recent versions (e.g., Windows Defender, BitLocker).
USER FRIENDLYNESS	
Can be more challenging for beginners.	User-friendly and familiar to most users.
Command-line interface (CLI) proficiency is often required for advanced tasks.	GUI (Graphical User Interface) is intuitive and widely used in personal and business environments.
COMMUNITY	
Community support through forums, wikis, and IRC channels.	Official support from Microsoft.
Commercial support is available for enterprise distros like Red Hat and SUSE.	Extensive documentation, help forums, and professional support services.
Active and helpful user communities.	Many IT professionals are trained in Windows administration.
RELIABLE & SECURE	
It is considerec as reliable OS and is well supported with security patches	Reliability: Windows is widely used and trusted in various environments, from personal computers to enterprise systems. Security Patches: Microsoft provides regular security updates and patches through its Windows Update service to address vulnerabilities and improve system security.

Linux Files Systems and Linux Directory Structure

Linux relies on two key concepts for data organization: filesystems and directory structure.

Linux Filesystems:

- A filesystem is a method of organizing and storing data on a storage device like a hard disk drive (HDD) or solid-state drive (SSD). It defines how files and folders are stored, accessed, and managed.
- File Sytem stores data in hierarchy.

- 

ext2 (Second Extended File System):

- One of the earliest Linux file systems ,Does not support journaling, which means it is more vulnerable to data loss in case of sudden power loss or system crash.
- Simple and robust, suitable for older systems or non-critical data.

ext3 (Third Extended File System):

- Built on ext2 with added journaling support which helps recover the file system quickly after a crash, minimizing the risk of data corruption.
- Good balance of performance and reliability, widely used in Linux distributions for many years.

XFS (X File System):

- High-performance file system optimized for scalability and large files , Suitable for servers and high-performance computing environments where handling large amounts of data is critical.

Btrfs (B-Tree File System):

- Modern file system with features like snapshots, checksums, and integrated RAID.
- Designed for fault tolerance, easy administration, and support for large storage capacities.

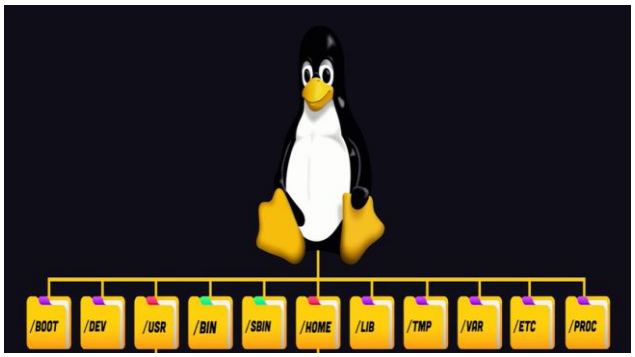
Linux Directory Structure

Linux organizes its file system into a hierarchical structure starting from the root directory /. The root directory (/) sits at the top, and all other directories branch out from there.

The directory structure organizes files and folders in a hierarchical manner, similar to a tree with branches.

/bin: Contains essential executable programs (commands) for common tasks.
/etc: Houses configuration files for the system and applications.
/home: This is where user accounts reside, with each user having a subdirectory for their personal files.
cd /home
[root@localhost home]# ls
Output
admin Afiya …folder gets created for each user

/lib: Stores system libraries needed by applications.
/sbin: Contains system administration tools typically only used by root (administrator) users.
/tmp: Holds temporary files used by applications.
/usr: This directory contains most user applications and libraries.



What is /etc/passwd and /etc/shadow file , why it exists

The /etc/passwd and /etc/shadow files are fundamental components of user authentication in Linux systems

They work together to securely manage user accounts and passwords.

/etc/passwd File:

- Function: This file stores basic user account information in a colon-separated format, one line per user.
- The /etc/passwd file is traditionally world-readable (accessible to all users). This allows system processes to identify users and their properties. However, it's important to note that the password itself is not stored in this file.

  /etc/shadow File

  This file stores the actual encrypted passwords for user accounts

  The `/etc/shadow` file is restricted to the root user (administrator) only. This enhances security by preventing unauthorized access to user passwords.

  --Basically for security purpose we have to different files

cd /

[root@localhost /]# ls

Output

bin boot dev etc home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var

Benefits of a structured directory system:

- Organization: Makes it easier to locate files and folders.

- Standardization: Ensures consistency across different Linux distributions.
- Permissions: Directory structures allow for setting permissions that control who can access and modify files.

LVM (Logical Volume Management) Explained

LVM (Logical Volume Management) is a powerful feature in Linux that allows you to manage physical storage space in a more flexible way

Physical Volumes (PVs): These are the actual physical storage devices that LVM uses, such as hard disk drives or partitions on those drives.
Volume Groups (VGs): A VG is a collection of one or more PVs that are grouped together under LVM for logical management.

Logical Volumes (LVs): These are virtual partitions created from the storage space within a VG. You can think of them as the usable partitions you see in your file system, but they are carved out of the combined space of the PVs in a VG.

Reducing the Size of an LVM Partition

While LVM offers flexibility, reducing the size of an existing logical volume (LV) requires careful steps to avoid data loss. Here's a general process:

1. Backup Your Data: This is crucial! Always ensure your data on the LV is fully backed up before proceeding.

2. Unmount the Filesystem: The LV you want to shrink must be unmounted before resizing.

3. Reduce the Filesystem Size: Use the resize2fs command (for ext2/ext3/ext4 filesystems) on the mounted LV to reduce its size. **Important:** This step reduces the size of the filesystem within the LV, not the LV itself.

4. Reduce the Logical Volume Size: Use the lvreduce command to shrink the size of the actual LV. This command will take the reduced size of the filesystem from step 3 into account.

5. (Optional) Reduce the Volume Group Size: If you want to reclaim unused space in the VG after shrinking the LV, you can use the vgreduce command. However, this step is more complex and typically not required for simply reducing a partition size.

Mounting and Unmounting Volumes in Linux

Mounting and unmounting volumes are fundamental tasks in Linux for managing access to storage devices. Here's a breakdown of the process:

Mounting:

- Purpose: Makes a storage device (partition, disk, filesystem) accessible to the system and allows you to interact with its contents.
- Command: The mount command is used for mounting.

Unmounting:

- Purpose: Detaches the mounted storage device from the system, making its contents inaccessible.
- Command: The umount command is used for unmounting.

Mounting Volumes

Identifying the volume.

Creating a mount point.

Using the mount command to attach the volume.

Using the umount command to detach the volume when done.

Step-by-Step Process

1. Identify the Volume Use the lsblk command to list available storage devices and partitions.

   Command - lsblk

2. **Create a Mount Point** Create a directory where the volume will be mounted.

   cd /mnt

   [root@newhostname mnt]# ls

   Command -sudo mkdir /mnt/mydata …. This will create a folder , where we can mount the volume

3. **Mount the Volume** Use the mount command to attach the volume to the mount point.

Command - sudo mount /dev/sdb1 /mnt/mydata … This mounts the filesystem on /dev/sdb1 to /mnt/mydata.

3. Verify the mount with the df -h or mount command.
4. df -h | grep /mnt/mydata
5. /dev/sdb1: Refers to the first partition on the second detected SCSI/SATA disk.
6. /mnt/mydata: The directory where you want to mount the filesystem.
7. Mounting: The process of attaching the filesystem on /dev/sdb1 to the directory /mnt/mydata, making the files on the partition accessible from that directory.

This command allows you to access and manage the files stored on /dev/sdb1 from the /mnt/mydata directory, integrating the storage device into the overall filesystem structure of your Linux system.

Most important Linux commands

Commonly used linux commands:

- ls: Display directory contents such as folders and files.
- mkdir: Used to create a new directory.
- pwd: Shows the current directory.
- top: Display system running processes and resource usage.
- grep: Search a specific pattern in a file.
- cat: Through this command, users can add multiple files and also display the content of the files.
- tar: Archives directories and files into a tarball.
- wget: Download files from the browser or web.
- free: Shows memory usage.
- df: Shows disk space usage.
- man: Gives a manual page for a specific command that displays instructions and details.

File permission in Linux.

There are three types of file permissions in Linux:

- Read: Users open and read files with this permission.
- Write: Users can open and modify the files.
- Execute: Users can run the file.

list all the processes running in Linux

You can list the currently running process in Linux through various commands such as:
ps Command:
The ps command displays brief information about the running processes. You can use the ps -f or ps -f command because the -f option shows the full-format result, and the -e option displays all processes. Moreover, you can use the ps auxf command to get a detailed list of processes.
top and htop Command:

- The top command displays the real-time details about the system process and the complete resource usage.
- The htop command is the improved version of the top command because it displays the color-coded list with additional features such as sorting, filtering, sorting, etc.

Check disk space usage

There are some simple commands you can use to check disk space usage, such as:
df Command:
The df or disk-free command shows the used and the available disk space. You can use the additional options to check disk space differently. For instance, you can use the df -h command to check the disk usage in the human-readable format.
du Command:
The du or disk usage command estimates and shows the disk space usage, so running the du command with no option shows the disk usage of your current directory. However, you can run the following command to check the disk usage of a specific directory:

```
du -sh ~/<directory>
```

Securing a Linux server

to secure the Linux server and protect it from data breaches, security threats, and unauthorized access. Here are some of these methods:

- Create a strong password
- Update the server and apply security patches.
- Use secured protocols like SSH and configure it to use key-based authentication for higher security.
- Use the intrusion detection system (IDS) to monitor network traffic and prevent malicious activities.
    - An Intrusion Detection System (IDS) is a security technology designed to detect unauthorized access or violations of policy within a computer network or system. IDSs monitor network traffic and system activities for signs of suspicious behavior, potential security breaches, or policy violations.
- Configure the firewall to limit the inbound and outbound traffic on the server.
- Disable all unused network services.
- Create regular backups.
- Review logs and perform regular security audits.
- Encrypt network traffic and enable monitoring.

Optimizing Linux system performance

- Updates the system as per the latest one available.
- Optimize the disk, enable the caching, and optimize the access pattern.
- Manage memory and CPU usage.
- Disable the necessary services and use lightweight alternatives of the tools.
- Monitor the system resources regularly.
- Perform the Kernel parameter tune-up.
- Use tools like Performance Co-Pilot (PCP) to monitor system-level performance.

Absolute and relative paths in Linux?

Absolute path

- An absolute path specifies the complete location of a file or directory starting from the root directory (/).
- It acts like a full address, guiding you from the top level of the filesystem down to the specific file or directory.
- Absolute paths always begin with a forward slash (/).
- Example: /home/user/documents/myfile.txt refers to the file "myfile.txt" located inside the "documents" folder within the "user" directory, which itself resides under the root directory.
- **Absolute paths:** Use absolute paths when you need to pinpoint the exact location of a file or directory regardless of your current working directory. This is useful when you know the full file path or want to access system-wide files in specific locations.
- Absolute paths are more specific and universally understood.

Relative Path

- Relative paths specify the location of a file or directory relative to your current working directory (the directory you're currently in).
- They are like instructions based on your current position.
- Relative paths do not start with a forward slash.
- Dot (.): Represents the current working directory.
- Double Dot (..): Refers to the directory one level up in the directory hierarchy, the parent directory.
- **Relative paths:** Use relative paths when working within a specific directory and want to reference files or directories relative to your current location. This is more concise and efficient when you're working within a project directory structure.
- Relative paths are more concise but depend on your current location.

sudo in Linux?

The word “sudo” is the short form of “Superuser Do” that allows you to run the command with system privileges. With this command, you can get the system’s administrative access to perform various tasks. The sudo command requires a password before the execution to verify the user’s authorization.

sudoers file in Linux, and how do you configure sudo access for users?

The sudoers file in Linux controls the sudo access permissions for users. It determines which users are allowed to run commands with superuser (root) privileges. To configure sudo access, you can edit the sudoers file using the visudo command.

For example:

```
sudo visudo
```

Now add this line anywhere in the file. For instance, if we want to grant a user full sudo access.

```
user_name ALL=(ALL) ALL
```

Create a file

1] cat - Primarily used to display the contents of a file. However, it can also be used for some file creation tasks with redirection.

By default, cat filename.txt simply displays the contents of the file "filename.txt" on your terminal. However, cat can be used for creating files with redirection. For instance, echo "Hello, world!" > newfile.txt uses redirection (>) to send the output of the echo command (the text "Hello, world!") to create a new file named "newfile.txt" containing that text.

Cat > filename ”content” //To come out of the file “Ctrl + d”

cat file.txt // displays the content

cat file1.txt file2.txt // Concatenate multiple files and display the output

cat file1.txt >> file2.txt // Append the contents of a file to another file:

cat file1.txt > file2.txt // Redirect the contents of a file to another file

cat -n file.txt //Number all lines of a file

2]touch - The `touch` command is used to create empty files or update the access and modification timestamps of existing files.

When you run touch filename.txt, it creates a new file named "filename.txt" with zero bytes of data (empty). You can create multiple files at once using the touch command with multiple filenames.

touch [options] [file...]

Options:

- -a : Change only the access time.
- -m : Change only the modification time.
- -t [[CC]YY]MMDDhhmm[.ss] : Use the specified time instead of the current time.
- -r file : Use the access and modification times of another file.

touch newfile.txt // Create an empty file

touch existingfile.txt // Update the timestamps of an existing file

touch -a file.txt // Change only the access time

touch -m file.txt // Change only the modification time

touch -t 202407091230.00 file.txt // Set a specific timestamp

/*

Touch

touch f3 f4 f5 --> created a filex

vi f3 --> It will open a vi editor , and add the content.

*/

Cat	Touch
Primarily used for displaying and concatenating the contents of files	Primarily used for creating empty files or updating the timestamps of existing files.
Useful for viewing file contents, merging files, and redirecting file data to other commands or files.	Useful for creating new files quickly and updating file timestamps for scripts or build processes.

Outputs the contents of files to the terminal or another file	No output to the terminal; it only creates or modifies files silently.
Can create new files if used with redirection, but typically reads and outputs file data.	Creates new empty files if they do not exist or updates the timestamps of existing files.
Cannot create multiple files	Can create multiple files at once

Creating a new file with content by using echo command

echo "This is some text content." > myfile.txt

Creating a new file with content by using text editor nano command

nano newfile.txt # This opens nano text editor. Write your content and save the file (Ctrl+O, then Enter).

Creating a new file with content by using text editor vi command

Vi newfile.txt

Remove a file

There are two primary commands for removing files in Linux:

rm (remove): This is the most common way to delete files. However, it's crucial to understand its behavior as it's permanent.

Syntax: rm [options] filename

Options:

-i: Enables interactive mode, prompting you for confirmation before deleting each file (recommended for beginners to avoid accidental deletion).

-f: Forces deletion without prompting, even for read-only files (use with caution).

-r: Recursively deletes directories and their contents (be very careful with this option).

Examples:

Rm myfile.txt: Removes the file "myfile.txt" without confirmation (not recommended unless you're sure).

rm -i important.txt: Prompts for confirmation before deleting "important.txt".

shred: This is a more secure option for deleting files, especially when dealing with sensitive data. It overwrites the file contents with random data multiple times before deletion, making data recovery significantly more difficult.

- Syntax: shred [options] filename
- Options:
    - -n: Number of passes to overwrite the file with random data (default is 3). Higher numbers provide greater security but take longer.

Example:

- shred -n 7 confidential.doc: Overwrites the file "confidential.doc" seven times with random data before deleting it.

Important Considerations:

- Deleted files are gone: Unlike some operating systems, Linux doesn't have a recycle bin. Once you delete a file, it's gone unless you have a backup.
- Use rm with caution: Especially when using rm -f or rm -r, be absolutely certain about the file or directory you're deleting to avoid accidental data loss.
- Consider shred for sensitive data: If you're deleting sensitive information, shred is a more secure option to make data recovery more challenging.

Create and Delete a directory

Creating a Directory:

- Use the mkdir (make directory) command to create new directories.
- Syntax: mkdir [options] directory_name
- Options (not always necessary):
    - -p: Creates all parent directories that don't exist in the path. Useful for creating nested directory structures at once.

mkdir documents # Creates a directory named "documents" in the current directory.

mkdir work/projects # Creates a directory named "projects" inside a directory named "work". If "work" doesn't exist, -p will create it automatically.

mkdir -v reports/sales/q1 # Creates "q1" inside "sales" which is created inside "reports", with verbose output showing each directory creation.

Deleting a Directory:

Mostly directories are blue in colour and files are black

- Use the rmdir (remove directory) command to delete empty directories.
- Syntax: rmdir directory_name
- Important: rmdir can only delete empty directories. It won't work if the directory contains files or subdirectories.
-

rmdir documents # Deletes the empty directory "documents".

To delete a directory with contents, you'll need to use the rm command with the -r option (see below).

Deleting a Directory with Contents

- Use the rm (remove) command with the -r (recursive) option to delete a directory and all its contents, including subdirectories and files.
- **Syntax:** rm -r directory_name
- **Warning:** This operation is permanent! Be absolutely sure you want to delete the directory and its contents before using rm -r.
-
- rm -r work # Deletes the directory "work" and all its contents (files and subdirectories). Use with caution!

Hidden Files

hidden files are files that don't show up by default when you use the `ls` command to list directory contents. These files typically start with a leading dot (.) character in their filename.Why hide files?

- System files: Linux uses many configuration and system files that users don't necessarily need to see or modify during regular operation. Hiding these files reduces clutter and helps prevent accidental changes to critical system settings.
- Personal preferences: Users can also create hidden files to store personal data or configuration files for specific applications, keeping them organized and separate from regularly accessed files.

Creating Hidden Files

`touch .mysecretfile` will create a hidden file named ".mysecretfile" in the current directory.

Viewing Hidden Files:

`ls -a` will show all files in the current directory, including those starting with a dot.

Unhiding Files

- While there's no single command to unhide a file, you can simply rename the file to remove the leading dot from its name.

- For example: `mv .hiddenfile normalfile` will rename the hidden file ".hiddenfile" to "normalfile", making it visible in directory listings.

Copy a file

`cp` (copy) command

```
cp [options] source_file destination
```

- s**ource_file:** The file you want to copy.

- **destination:** The location where you want to create a copy of the file. This can be a filename (to copy within the same directory) or a directory path (to copy to a different directory).

Cp myfile.txt myfile_copy.txt # Creates a copy of "myfile.txt" named "myfile_copy.txt" in the (same)current directory.

cp important.doc /home/user/documents # Copies "important.doc" to the "documents" directory within the user's home directory.

cp -r project_folder /home/user/backup # Copies the entire "project_folder" directory and its contents to the "backup" directory.

cp file1.txt file2.txt file3.txt /home/user/copied_files # Copies three files to the "copied_files" directory.

cp -rv /home/user/projects /backup # Copies the "projects" directory and its contents recursively to the "backup" directory, showing each file being copied.

The `-f` option, used with various Linux commands, stands for "force." The `-f` option instructs the command to **bypass certain safety checks** and proceed with the operation even if it might encounter issues.

`cp -f`: forces the copy operation to overwrite existing files at the destination **without prompting for confirmation**.

cp -f source_file destination

mv -f : forces the move operation, overwriting existing files at the destination without confirmation.

mv -f source_file destination

rm -f : bypasses this confirmation and deletes the file directly.

rm -f filename

Move or Rename a file

mv [options] source destination

mv file.txt /path/to/directory/ // move file.txt to a different directory

mv oldname.txt newname.txt // **Rename a file:**

The `mv` command in Linux is used to move or rename files and directories

How to check the currenly logged in user

whoami

how to check the id of the user

id
output
id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023

To see list of files

`ls`: This lists the contents of the current working directory by default. It displays filenames only, with minimal details.

`ls -l`:(List with details) This provides a long listing format, showing additional information for each file, including:

- File permissions (read, write, execute)
- Number of hard links
- File owner and group

- File size
- Last modification date and time

```
total 12
-rw-r--r-- 1 user  user  1024 May  2 10:30 file1.txt
drwxr-xr-x 2 user  group 4096 Jun  7 15:36 directory1
-rw-rw-r-- 1 user  user  2048 Jul  9 12:00 file2.txt
```

`ls -a`: This displays all files in the directory, including hidden files whose names begin with a dot (.).

`ls -t`: Sorts the listing by modification time, with the most recently modified files appearing at the bottom.

`ls -r`: Reverses the sorting order. For example, `ls -lt` sorts by modification time with the most recently modified files at the top.

`ls -h`: Displays file sizes in a more user-friendly format like KB, MB, or GB instead of bytes.

`ls -d directory_name`: Provides information about the specified directory itself, rather than its contents.

`ls -R directory_name`: Lists the contents of a directory and all its subdirectories recursively. This can be very useful for exploring directory structures.

`ls -lrth`

- **-l:** This stands for "long listing" and provides detailed information about each file, including permissions, owner, group, size, and modification time.
- **-r:** This stands for "recursive" and instructs `ls` to list the contents of the current directory and all subdirectories within it.
- **-t:** This stands for "sort by modification time" and arranges the file listing based on the last modification time, with the most recently modified files appearing at the bottom by default.
- **-h:** This stands for "human-readable" and displays file sizes in a more user-friendly format like KB, MB, or GB instead of bytes.

Difference between $ and

$ - Represents the current user
# - Represents the root user , currently operating with **root privileges**. The root user has the highest level of access and can modify the entire system.

Switch from one user to another

`su` Command - `su` switches you to the root user (`root`) if you don't specify a username.
su – username
Switches user within the current session
Efficient for temporary tasks under a different user

Requires knowing the password of the target user (be cautious with root)

Sudo su – 'super user do' switch to root user

File permission

File Permission Structure

-rwxr-xr—

File Type:

- `-` : Regular file
- `d` : Directory
- `l` : Symbolic link
- `c` : Character device
- `b` : Block device

There are 3 types of permission
Read : r
Write : w
Execute : x

There are 3 types of users in linux
Owner - u
Group - g
Others - o

File permissions are typically represented using a string of nine characters, like this: `rwxr-xr-x`.

- The first three characters represent the permissions for the owner (read, write, execute).
- The next three characters represent the permissions for the group.
- The last three characters represent the permissions for everyone else (others).
- A dash (-) in a position indicates that the corresponding permission is not allowed.

Example:
- `-rw-r--r--`: This grants read permission to the owner and group, and read permission only for others. This is a common permission setting for text files.

- `-rwx------`: This grants all permissions (read, write, execute) only to the owner, and no permissions to the group or others. This might be used for a private script file.
- `drwxr-xr-x`: This applies to directories. The `d` at the beginning indicates it's a directory. Here, the owner and group have read and execute permissions, and others have only read permission. This is a common permission setting for a directory containing shared resources.

Changing File Permissions

Albhabetic Notation

Owner : u

Group : g

Other : o

'+' : To add the permission

'-' : To remove the permission

chmod [who][operation][permissions] [file]

chmod u+x file.txt // Add execute permission for the owner

chmod g-w file.txt // Remove write permission for the group:

chmod a=rw file.txt // Set read and write permissions for all

Numeric (Octal) Mode

Permissions can also be set using numeric values where:

- `r` (read) = 4
- `w` (write) = 2
- `x` (execute) = 1

chmod [mode] [file]

chmod 755 file.txt

`755` translates to:

- Owner: `rwx` (4+2+1 = 7)
- Group: `r-x` (4+0+1 = 5)
- Others: `r-x` (4+0+1 = 5)

how to create a user with password in linux

- To create a new user with a password in Linux, you can use the `useradd` command followed by the `passwd` command
  Create the User Account (using sudo)

- sudo useradd username

  Set the Password (for the newly created user)
  sudo passwd username

Changing File Ownership

The `chown` command changes the owner and group of a file or directory.

chown [owner][:group] [file]

chown user1 file.txt // Change owner to `user1`

chown user1:group1 file.txt // Change owner to `user1` and group to `group1`

Pipe Command

The pipe (|) symbol in Linux is a powerful tool for chaining commands together. It allows you to send the output of one command as the input for another command.

Example: `ls | grep .txt` - This command would first use `ls` to list all files in the current directory. Then, it would pipe the output (list of filenames) to `grep .txt`, which would filter the list and only display filenames ending with ".txt".

head:

- Purpose: Displays the beginning (head) of a text file.
- Syntax: head [options] filename
- Options:
  - -n: Specifies the number of lines to display from the beginning (default: 10).
- Example:
  - headmyfile.txt: Displays the first 10 lines of "myfile.txt".
  - head -5 important.txt: Displays the first 5 lines of "important.txt".

tail:

- Purpose: Displays the ending (tail) of a text file.
- Syntax: tail [options] filename

- Options:
  - -n: Specifies the number of lines to display from the end (default: 10).
  - -f: Follows the file, displaying new lines as they are added (useful for live logs).
- Example:
  - tail system.log: Displays the last 10 lines of "system.log".
  - tail -f access.log: Continuously monitors the "access.log" file, showing new lines as they appear.

wc:

- Purpose: Counts lines, words, and characters in a text file.
- Syntax: wc [options] filename
- Options:
  - -l: Counts lines only.
  - -w: Counts words only.
  - -c: Counts characters only.
  - By default, wc provides counts for all three (lines, words, characters).
- `wc report.txt`: Displays the number of lines, words, and characters in "report.txt".
- `wc -l summary.txt`: Counts only the number of lines in "summary.txt".

Head	Tail	wc
Displays beginning of a text file	Displays ending of a text file	Counts lines, words, and characters
-n: Specify number of lines to display	-n: Specify number of lines, -f: follow the file	-l: Count lines only, -w: Count words only,
Example	Example	Example
head -3 data.csv	tail -f error.log	wc poem.txt

- Use `head` to peek at the initial contents of a file.
- Use `tail` to review the most recent parts of a file or monitor logs for changes.
- Use `wc` to get a quick overview of the size and content structure of a text file (number of lines, words, and characters).

Compress and decompress files in Linux

tar command along with gzip compression can be used.
If we want to create a file name “afiya” with gzip compression. We use the following command.

```
tar -czvf afiya.tar.gz files
```

This command will create a compressed archive file containg the specified “files”
To decompress the same, we use the following command.

```
tar -xzvf afiya.tar.gz
```

sed command used for in Linux

The sed command is used to perform text transformations on files. It can search for specific patterns and replace them with desired text.

For Example:

```
sed `s/foo/bar/g` file.txt
```

The sed command (stream editor) can be used to find and replace text in a file. The basic syntax is sed ‘s/pattern/replacement/g’ filename.

For example: to replace all occurrences of “true” with “False” in a file

```
sed 's/true/False/g' file_name
```

Repository in Linux

Repositories in Linux are servers that contain software packages. They allow users to download, install, and update software using package management systems. These repositories are typically maintained by the distribution developers

APT (Advanced Package Tool): Used by Debian and its derivatives like Ubuntu.
YUM (Yellowdog Updater, Modified): Used by RPM-based distributions like Fedora and CentOS.
DNF (Dandified Yum): The next-generation package manager for RPM-based distributions.

Some Important Commands

1. top: Display running processes and system resources.

Example: top

1. htop: Interactive process viewer.

Example: htop

1. vmstat: Display virtual memory statistics.

Example: vmstat -a

1. mpstat: Display CPU statistics.

Example: mpstat -a

1. iostat: Display I/O statistics.

Example: iostat -x

1. netstat: Display network socket statistics.

Example: netstat -an

1. ss: Display socket statistics.

Example: ss -tlnp

1. tcpdump: Capture and display network traffic.

Example: tcpdump -i eth0

1. strace: Trace system calls and signals.

Example: strace -p 1234

1. lsof: List open files and network connections.

Example: lsof -i :80

1. pgrep: Search for processes matching a pattern.

Example: pgrep -f mysql

1. pkill: Kill processes matching a pattern.

Example: pkill -f mysql

1. sysctl: Display and modify kernel parameters.

Example: sysctl -a

1. dmesg: Display kernel messages.

Example: dmesg | tail -50

1. auditctl: Display and configure audit rules.

Example: auditctl -l

**********PRACTICAL**********
----------File Creation----------
cat > file1
hi this is the eg to create a file from cat command
ctrl d

cat file1 // this will display the content of file1

cat file1.txt file2.txt // Concatenate multiple files and display the output
cat file1.txt >> file2.txt // Append the contents of a file to another file:
cat file1.txt > file2.txt // Redirect the contents of a file to another file(this will override the content of file2)
cat -n file1.txt //Number all lines of a file
---
touch fileT1 fileT2 fileT3 // create file with touch command , will create an empty file
touch fileT1 // this is the existing file , so it will change the timestamp

ll // can check the timestamp has been changed

touch -a f3 //// Change only the access time
check the output
ls -lu f3
stat f3

touch -m f3 // Change only the modification time
check the output
ls -lu f3
stat f3

touch -t 202407091230.00 f3
check the output
ls -lu f3
stat f3

---
echo "This is some text content." > echofile.txt
Check the output
cat echofile.txt

---
nano filebynano
Output
Nano Editor will be opened and u can write the content
write the content
Save: Ctrl + O, then Enter
Exit: Ctrl + X

Check the output
cat filebynano

---
vi filebyvi
Output
Vi Editor will be opened and u can write the content
cat filebyvi

---
rm filebyvi -- - remove a file

rm -i filebynano -- interactive terminal - remove a file , It will ask for a prompt

---

mkdir impdocuments # Creates a directory named "impdocuments" in the current directory.
mkdir work/projects -- # Creates a directory named "projects" inside "work". If "work" doesn't exist, -p will create it automatically.

mkdir work/projects -- If work folder doesn't exist
mkdir -p -v reports/sales/q1 // v , verbose - output will be shown
mkdir: created directory ‘reports’
mkdir: created directory ‘reports/sales’
mkdir: created directory ‘reports/sales/q1’

---
rmdir impdocuments .. delete empty directory
rmdir -r impdocuments .. delete the directory , and inside contents

---
touch .mysecretfile .. create hidden file
ls -a ..view hidden file
mv .mysecretfile myopenfile .. make it un hidden , by renaming it.

---

cp myfile.txt myfile_copy.txt   // copy within same folder
cp important.doc /home/user/documents // copy file "important.doc" to the /home/user/documents
mkdir copyfolder
cd copyfolder
pwd // current location
cd .. // moving one folder back
cp myopenfile /home/afiya/impwork/copyfolder //copying the file from the current loc to copyfolder
cd copyfolder
ls // checking the contents , whether files copied or not

cp -r project_folder /home/user/backup   // copy entire folder
cp -rv /home/user/projects /backup # Copies the "projects" directory and its contents recursively to the "backup" directory, showing each file being copied.

---
mv file.txt /path/to/directory // move a file , cut the file and paste at the given location

mv oldname.txt newname.txt // Rename a file
---
whoami //check current logged in user
id // check id of the current logged in user

---

ls ..display only the content
ls -l .. output - -rw-r--r--. 1 root root 0 Jul 11 13:36 myopenfile
ls -a .. will display hidden files
ls -t .. will sort and arrange the file based on the last modification time
ls -r ..list the contents of the current directory and all subdirectories within it.
ls -lh ..

Output
total 8.0K
-rw-r--r--. 1 root root 0 Jul 11 13:48 file1

ls -ld directoryname // List directories themselves, not their contents.(details abt the directory)
ls -ld insidecopy/
drwxr-xr-x. 2 root root 19 Jul 11 13:52 insidecopy/

ls -lR directoryname //-R: Recursively list subdirectories encountered.(details abt the directory with its content)
Output
ls -lR insidecopy/
insidecopy/:
total 4
-rw-r--r--. 1 root root 115 Jul 11 13:52 file1

---Group-----
sudo groupadd DevopsAdmin --create / Add a group
cat /etc/group --To list all groups on the system

sudo groupmod -n newgroupname oldgroupname -- change the group name
sudo groupmod -n DevopsAdmintrator DevopsAdmin -- change the group name
sudo groupdel DevopsAdmintrator -- delete the group

---Users----
sudo adduser Administrator --create user Administrator
sudo passwd Administrator -- password

cat /etc/passwd --To list all users on the system

sudo useradd -m Administrator -- add a home directory to the user

sudo usermod -l Admin Administrator --modify the login name

sudo usermod -d /new/home/directory Admin -- change user's home directory

sudo userdel -r Admin -- delete a user

sudo usermod -aG groupname username --Adding user to the group
sudo usermod -aG DevopsAdmintrator Afiya --Adding user to the group

NOTE-
/etc/passwd --This file contains user account information
/etc/shadow -- This file contains secure user account information, including encrypted passwords. It is readable only by the root user.
/etc/group -- This file contains group information

---
chmod [who][operation][permissions] [file]
Alphabetic
chmod u+x file1 .. u - owner will have execute rights ,g - group , o - other
chmod 755 file1 ..read = 4 , write =2 , execute =1 first number represent owner , second group , third other
7 means rwx for owner , 5 means read and execute

Note d : directory
- : regular file
first 3 character - owner , next three - group , next , other

---
chown [owner][:group] [file]
chown user1 file1.txt --- changing the owner of the file

chown user1:group1 file.txt ---changing owner to user1 and group to group1

---
grep "example" file.txt ---search for the pattern "example" in a file named file.txt

grep -i "example" file.txt ---search for the pattern "example" in a file named file.txt , case in sensitive

grep -r "example" --- Search for the pattern "example" in all files within the current directory and its subdirectories(Recursive search)

head -3 data.csv --- Displays beginning 3 lines of a text file
tail -f error.log ---Displays ending of a text file
wc poem.txt ---Counts lines, words, and characters

cat test
line no 1
line no 2
line no 3
line no 4
line no 5
line no 6

head -2 test
Output

line no 1
line no 2
[root@localhost copy]# tail -2 test
Output
line no 5
line no 6
[root@localhost copy]# wc test
Output
 6 18 60 test

---

ls | grep .txt - This command would first use ls to list all files in the current directory. Then, it would pipe the output (list of filenames) to grep .txt, which would filter the list and only display filenames ending with ".txt".

---
lsof---List All Open Files

lsof -u Ansible --- List Open Files by a Specific User

---
du -h insidecopy/ -du stands for disk usage , -h: Human-readable format (e.g., 1K, 234M, 2G).
Output

4.0K   insidecopy/

---
df -h ----df stands for Disk Free ,
Output
Filesystem   Size Used Avail Use% Mounted on
/dev/sda1   50G   10G   38G 21% /
tmpfs   16G   0   16G   0% /dev/shm
---

whatis --- it displays the purpose of the command

whatis ls
Output
ls (1)   - list directory contents

---

uname -a - prints all system information

---
netstat -tuln

-t: TCP connections.
-u: UDP connections.
-l: Listening sockets.
-n: Do not resolve names

Output
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp 0 0 0.0.0.0:111 0.0.0.0:* LISTEN
tcp 0 0 192.168.122.1:53 0.0.0.0:* LISTEN
tcp 0 0 0.0.0.0:22 0.0.0.0:* LISTEN
---

hostname //prints system's hostname
Output
localhost.localdomain

sudo hostname newhostname ////To set the hostname
[root@localhost copy]# hostname
newhostname

---
free -h //The free command displays the amount of free and used memory in the system
Output
total used free shared buff/cache available
Mem: 3.5G 2.9G 257M 30M 411M 380M
Swap: 3.9G 621M 3.3G

---
find /home/user -name "*.txt" //Search for files with .txt extension under /home/user.

find /var/log -mtime -7 //Find files modified in the last 7 days under /var/log.

find /home/user -size +10M //Find files larger than 10MB under /home/user.
---
zip -r archive.zip directory/ //zip a directory

tar -cvf archive.tar file1.txt file2.txt //Create archive.tar containing file1.txt and file2.txt.

tar -czvf archive.tar.gz directory/ ///Create archive.tar.gz containing directory (recursively).

[root@localhost copy]# vi test
[root@localhost copy]# cat test
line no 1
line no 2
line no 3

line no 4
line no 5
line no 6
Lord is True
Hi this is Afiya

sed -i 's/Afiya/Mafiya/g' test --sed is used for search and replace

Output
line no 1
line no 2
line no 3
line no 4
line no 5
line no 6
Lord is False
Hi this is Mafiya
---
cd - changed directory
cd .. - move one directory up

---Add this On Mail

ps - ps command is to check the currently running process , -f flag will give you detailed information.

```
  PID TTY          TIME CMD
 3577 pts/0    00:00:00 bash
 3626 pts/0    00:00:00 ps
```

ps -f

```
UID        PID  PPID  C STIME TTY          TIME CMD
afiya     3577  3556  0 08:51 pts/0    00:00:00 bash
afiya     3660  3577  0 08:51 pts/0    00:00:00 ps -f
```

---

sudo su - ---------- ‘super user do’ switch to root user

sudo: This stands for "Superuser DO." , This command allows a permitted user to execute a command as the superuser i.e with the privileges of the root user
With this command, you can get the system’s administrative access to perform various tasks.
The sudo command elevates your privileges to execute the following command (su -) as a superuser.

su: This stands for "switch user". It is used to switch from one user account to another.
The su command switches the current user to another user, which defaults to the root user if no user is specified

su Ansible ----------here i am trying to switch to Ansible user , it will ask me the password once pwd is provided it will switch to Ansible user
Password:
[Ansible@node1 afiya]$

[Ansible@node1 afiya]$ su ----------If you only write su , without specifying any user name it will switch to the root user
Password:
[root@node1 afiya]#

sudo su - ---------- sudo here is to elevate the privilege to run the switch user command
Last login: Tue Jul 16 09:16:16 IST 2024 on pts/0
[root@node1 ~]#

sudoers file : visudo -------- If you want to grant a user full sudo access ,It helps users to run commands with superuser (root) privileges.
sudo visudo
add this line in the file.
user_name ALL=(ALL) ALL

SOME IMPORTANT COMMANDS

1. top: Display running processes and system resources.

Example: top

1. htop: Interactive process viewer.

Example: htop

1. vmstat: Display virtual memory statistics.

Example: vmstat -a

1. mpstat: Display CPU statistics.

Example: mpstat -a

1. iostat: Display I/O statistics.

Example: iostat -x

1. netstat: Display network socket statistics.

Example: netstat -an

1. ss: Display socket statistics.

Example: ss -tlnp

1. tcpdump: Capture and display network traffic.

Example: tcpdump -i eth0

1. strace: Trace system calls and signals.

Example: strace -p 1234

1. lsof: List open files and network connections.

Example: lsof -i :80

1. pgrep: Search for processes matching a pattern.

Example: pgrep -f mysql

1. pkill: Kill processes matching a pattern.

Example: pkill -f mysql

1. sysctl: Display and modify kernel parameters.

Example: sysctl -a

1. dmesg: Display kernel messages.

Example: dmesg | tail -50

1. auditctl: Display and configure audit rules.

Example: auditctl -l

[root@localhost copy]# vi test
[root@localhost copy]# cat test
line no 1
line no 2
line no 3
line no 4
line no 5
line no 6
Lord is True
Hi this is Afiya

sed -i 's/Afiya/Mafiya/g' test --sed is used for search and replace

Output
line no 1
line no 2
line no 3
line no 4
line no 5
line no 6
Lord is False
Hi this is Mafiya