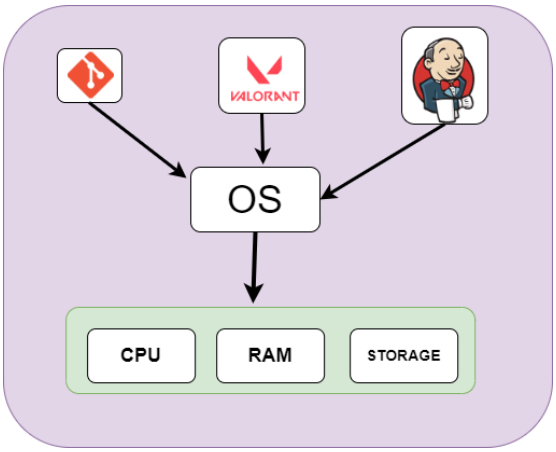
Session-2

Linux For DevOps

Linux is an open-source, Unix-like operating system kernel that serves as the foundation for various operating systems, collectively known as "Linux distributions" or "Linux distros."

Linux Components:

1. **Kernel:** The core component that interacts directly with hardware, manages resources, and provides essential services.
2. **Shell:** The command interpreter that lets users interact with the system through the command line. Popular shells include Bash (Bourne Again SHell) and Zsh.
3. **File System:** Linux uses a hierarchical file system structure, with directories (folders) organized under the root directory ("/"). Common directories include "/bin" (system binaries), "/home" (user home directories), and "/etc" (system configuration files).
4. **Packages and Package Managers:** Linux distributions use package managers like "apt" (Debian/Ubuntu), "yum" (Red Hat/CentOS), and "pacman" (Arch Linux) to install, update, and manage software packages.
5. **GUI and Desktop Environments:** While Linux is known for its command line, many distributions offer graphical user interfaces (GUIs) and desktop environments, such as GNOME, KDE Plasma, and XFCE.

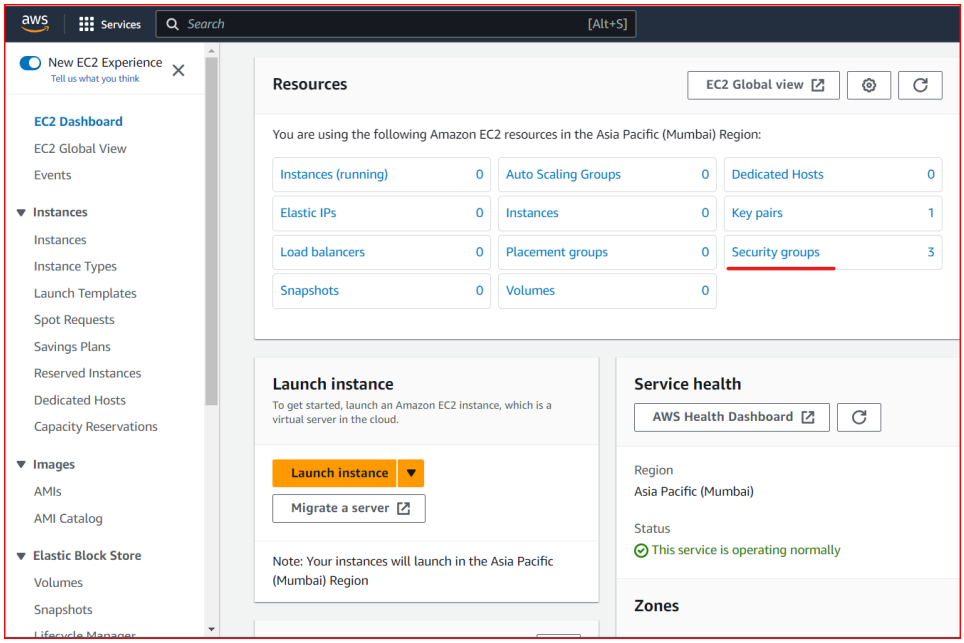


Linux is an operating system (OS) that serves as an intermediary between hardware and software applications, enabling them to communicate and work together seamlessly.

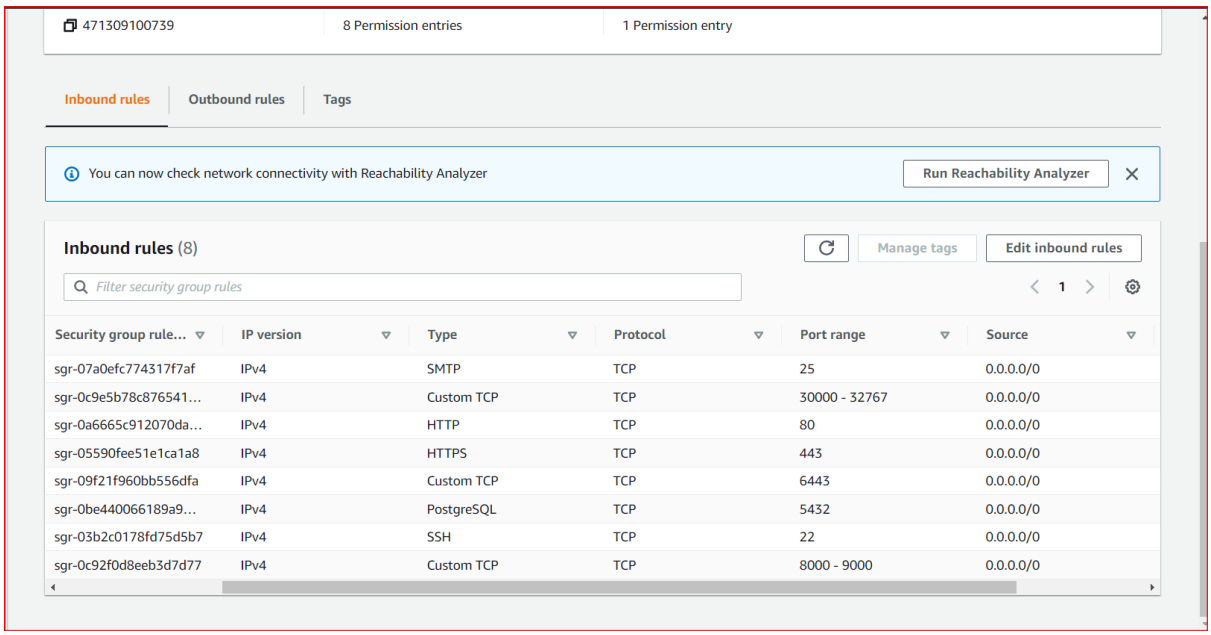
Creating a Linux Machine in AWS

1. Create an AWS Account and log in to AWS Management console.
2. Go to search, and type EC2

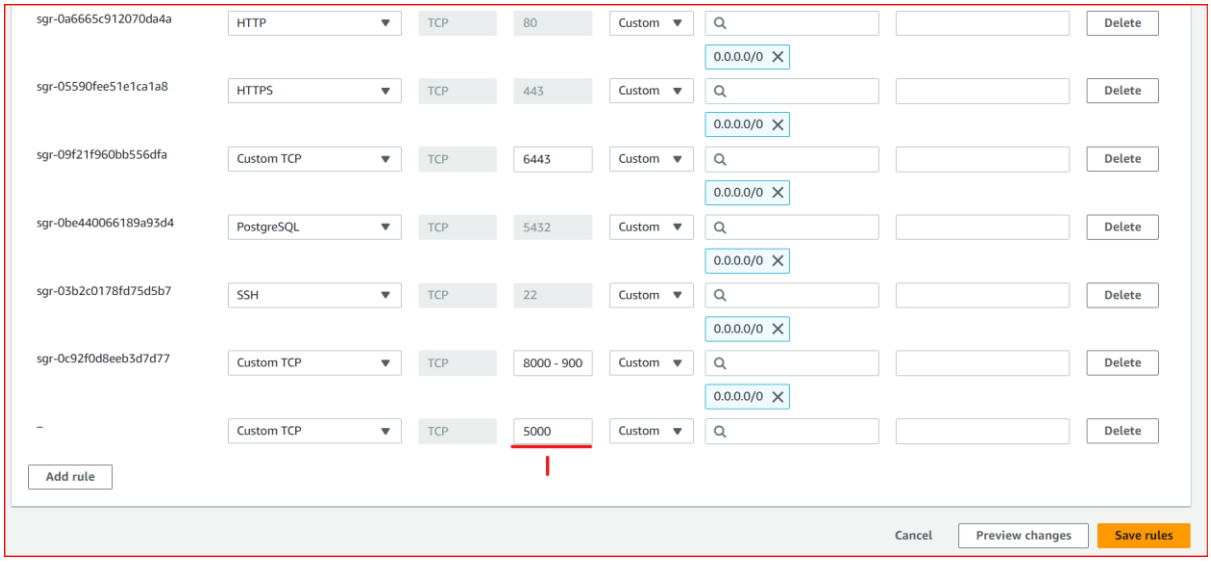
3. Click on EC2 and below page will open , in it click on security groups to open the ports on the VM.



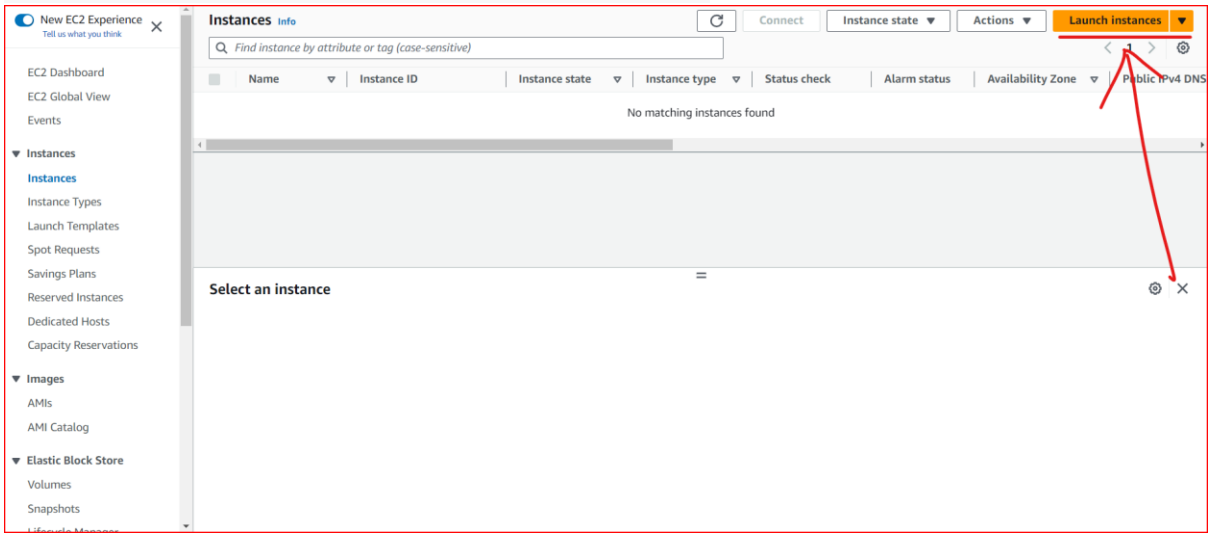
4. Inside the security group, I would request you to open below ports, as we will be needing those.



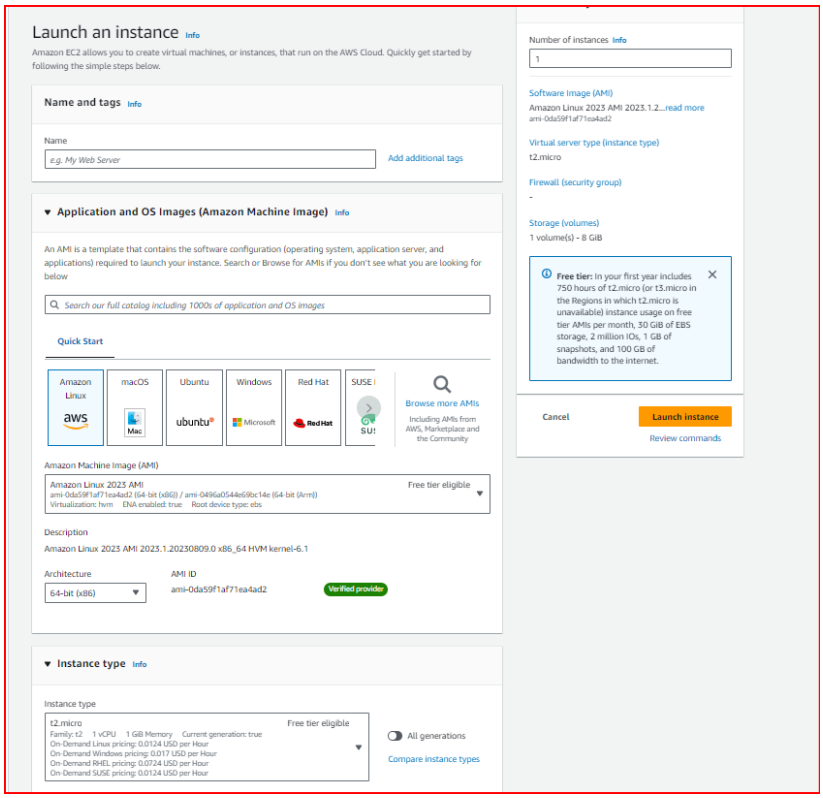
4. To Open a port you provide details as below.
   →Click create security group and click on add rule and add as below.
   →If editing an already existing group then click on edit inbound rules.

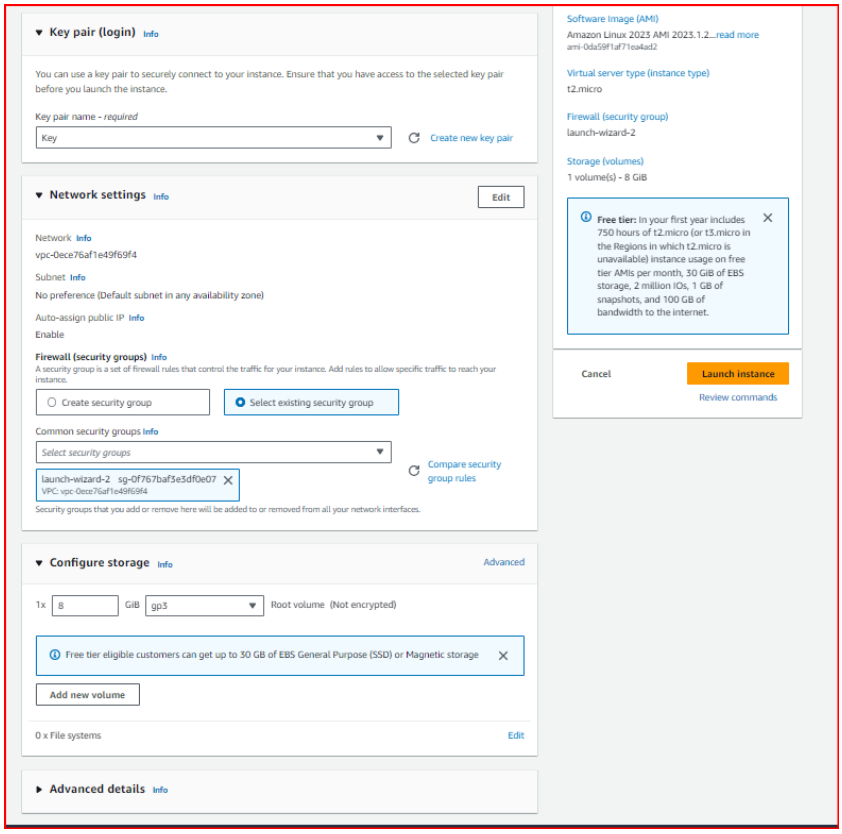


5. Once done, then click on instances or launch instance .

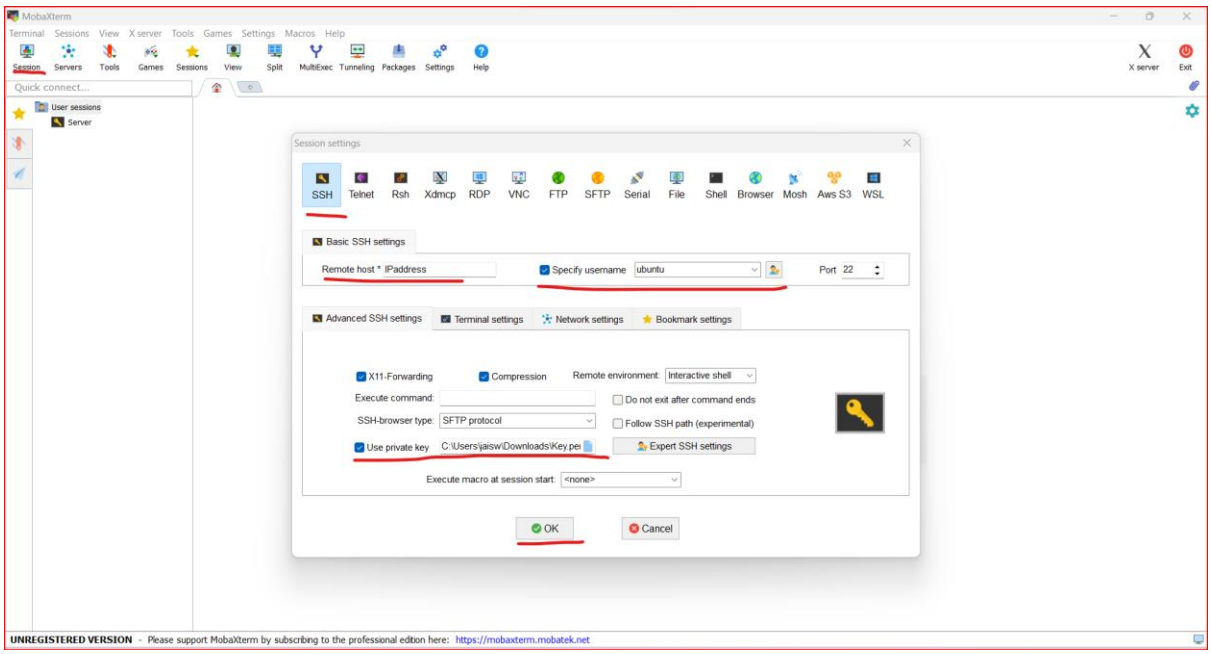


6. Next up, we will select the configuration for our VM. If doing for the first time then make sure to create a new key pair and keep the key safe.





7. Once your VM is created, you can access it using a tool Mobaxterm(https://download.mobatek.net/2322023060714555/MobaXterm_Portable_v23.2.zip)
   - Remote host-> Public IP
   - Username-> ubuntu if created ubuntu machine
   - Use private key-> select the pem file downloaded while creating the new key



Linux File system:

1. **/home:** This directory contains user home directories. Each user typically has a subdirectory here where they can store their personal files and configuration settings.

2. **/root:** This is the home directory for the root user, which is the superuser or administrator of the system.

3. **/bin:** Short for "binary," this directory contains essential system binaries (executable files) that are required for basic system operations, such as system maintenance and recovery.

4. **/sbin:** Similar to "/bin," this directory contains system binaries, but these are typically used by the system administrator for system maintenance and configuration tasks.

5. **/lib:** This directory holds essential libraries (shared files) needed for the system and applications to run properly.

6. **/usr:** This directory contains a variety of subdirectories including:

    - /usr/bin: User-level binaries and executables.
    - /usr/sbin: System binaries for administrative tasks.
    - /usr/lib: Libraries for user-level programs.
    - /usr/local: Locally installed software and libraries (not managed by the package manager).

7. **/opt:** This is often used for installing optional or third-party software. Software packages are installed in their own subdirectories within "/opt."

8. **/etc:** This directory contains system-wide configuration files. Configuration settings for various applications and services are stored here.

9. **/tmp:** Temporary files are stored in this directory. The contents of this directory are often cleared upon system reboot.

10. **/boot:** Contains files related to the boot process, such as the Linux kernel and bootloader configuration.

11. **/dev:** This directory contains device files that represent hardware devices on the system. These files allow programs to interact with hardware without needing to know specific details about the hardware.

12. **/var:** This directory contains variable data files that are expected to grow during the system's operation. This includes log files, spool directories, and other transient data.

13. **/media:** Mount point for removable media devices like USB drives and CD/DVD-ROM drives.

14. **/mnt:** A traditional location to temporarily mount additional filesystems or devices. This directory can also be used to manually mount filesystems.

Linux Must known concepts:

Changing Permissions

In Linux, you can change file and directory permissions using the `chmod` command. Permissions control who can read, write, and execute files and directories. Permissions are typically divided into three levels: owner, group, and others. Each level has three permissions: read (r), write (w), and execute (x). Here's how you can use the `chmod` command to change permissions:

1. Symbolic Method:

The symbolic method uses letters to represent permission changes. The syntax is as follows:

```
chmod who=permissions filename/directory file
```

name/directory

- **who:** Can be a combination of the following letters:
  - `u`: Owner (user)
  - `g`: Group
  - `o`: Others
  - `a`: All (equivalent to `ugo`)
- **permissions:** Use the following letters to specify the permission changes:
  - `r`: Read
  - `w`: Write

- `x`: Execute

Examples:

```
# Grant read and write permissions to the owner of the file
chmod u+rw filename

# Remove execute permission for the group and others
chmod go-x filename

# Give all permissions to the owner and read permission to the group and others

chmod ugo+rwx filename
```

2. Numeric Method:

The numeric method assigns a numeric value to each permission. The values are as follows:

- `4`: Read permission
- `2`: Write permission
- `1`: Execute permission

You can add these values to represent the desired permissions. For example:

- `7` grants read (4), write (2), and execute (1) permissions.
- `6` grants read (4) and write (2) permissions.

Examples:

```
# Grant read and write permissions to the owner, and read permission to the group and
others
chmod 644 filename

# Give full permissions to the owner and group, but no permissions to others
chmod 770 directory
```

Remember that changing permissions without proper consideration can affect the security and functionality of your system. It's important to only give permissions to users and groups that truly need them. Additionally, system files and directories should be handled with caution to avoid unintended consequences.

Changing Ownership

Linux, you can change the ownership of files and directories using the **chown** command. Changing ownership is useful when you need to transfer ownership of a file or directory to a different user or group. Here's how you can use the **chown** command to change ownership:

Syntax:

```
chown [OPTIONS] new_owner:group file/directory
```

- **new_owner:** The new owner's username.
- **group:** The new group. If not specified, the group remains unchanged.
- **file/directory:** The file or directory whose ownership you want to change.

Options:

- **-R**: Recursively change ownership for all files and subdirectories within the specified directory.

Examples:

```
# Change ownership of a file
chown newuser: newfile.txt
# Changes the owner to "newuser" and leaves the group unchanged

# Change ownership of a directory and its contents recursively
chown -R newuser:newgroup mydirectory
# Changes the owner to "newuser" and the group to "newgroup" for all files and
subdirectories in "mydirectory"
```

Keep the following points in mind:

1. **Superuser Privileges:** You typically need superuser (root) privileges to change ownership of files and directories that you don't own yourself.
2. **Existing User and Group:** Ensure that the new owner and group exist on the system.
3. **Recursion:** If you use the **-R** option, be cautious, especially when changing ownership of system directories, as it can have unintended consequences.
4. **Permissions:** Changing ownership might also impact permissions. Make sure to review and adjust permissions if needed after changing ownership.
5. **Backup:** Before performing ownership changes, it's a good practice to create a backup of the files or directories being modified, especially when using the **-R** option.

Always use the **chown** command with care, especially when changing ownership of system files and directories. Incorrect ownership changes can disrupt system functionality and security.

User Management & Group management

Some common commands and examples for creating, managing, and manipulating users and groups in a Linux environment.

1. Creating a User: You can use the `useradd` command to create a new user. For example, to create a user named "john":

```
sudo useradd john
```

2. Setting a Password for the User: Use the `passwd` command to set a password for the newly created user:

```
sudo passwd john
```

3. Creating a Group: To create a new group, you can use the `groupadd` command. For instance, to create a group named "developers":

```
sudo groupadd developers
```

4. Adding a User to a Group: You can add a user to a group using the `usermod` command. For example, adding the user "john" to the "developers" group:

```
sudo usermod -aG developers john
```

5. Changing User's Primary Group: If you want to change a user's primary group, you can use the `usermod` command with the `-g` option:

```
sudo usermod -g new_primary_group john
```

6. Changing User's Supplementary Groups: To modify a user's supplementary groups (groups other than the primary group), you can use the `usermod` command with the `-G` option:

```
sudo usermod -G group1,group2 john
```

7. Listing User's Groups: To see a list of groups a user belongs to, you can use the `groups` command:

```
groups john
```

8. Deleting a User: To delete a user, you can use the `userdel` command:

```
sudo userdel john
```

9. Deleting a Group: To delete a group, use the `groupdel` command:

```
sudo groupdel developers
```

10. Managing User Information: You can use the `usermod` command to modify various user properties such as username, home directory, shell, etc. For example, changing the shell for the user "john" to bash:

```
sudo usermod -s /bin/bash john
```

11. Display User Information: To view detailed information about a user, you can use the `id` command:

```
id john
```

Top Linux Commands :

Navigation:

- ls: List files and directories.
  - Example: ls -l /path/to/directory
- cd: Change directory.
  - Example: cd /path/to/directory
- pwd: Print working directory.
  - Example: pwd
- mkdir: Create a directory.
  - Example: mkdir new_folder

File Operations:

- cp: Copy files or directories.
  - Example: cp file.txt /path/to/destination
- mv: Move or rename files or directories.
  - Example: mv file.txt new_name.txt
- rm: Remove files or directories.
  - Example: rm file.txt
- touch: Create an empty file.

    - Example: touch new_file.txt

Text Manipulation:

- cat: Concatenate and display file content.
    - Example: cat file.txt
- grep: Search for a pattern in files.
    - Example: grep "pattern" file.txt
- sed: Stream editor for text manipulation.
    - Example: sed 's/old/new/' file.txt
- awk: Text processing tool.
    - Example: awk '{print $1}' file.txt

Process Management:

- ps: Display information about running processes.
    - Example: ps aux
- top: Display live system resource usage.
    - Example: top
- kill: Terminate processes by ID or name.
    - Example: kill PID

Package Management:

- apt: Advanced Package Tool (Debian-based systems).
    - Example: apt install package_name
- yum: Package manager (RHEL-based systems).
    - Example: yum install package_name
- dnf: Next-generation package manager (Fedora).
    - Example: dnf install package_name

Networking:

- ping: Send ICMP echo requests to a host.
    - Example: ping google.com
- ifconfig / ip: Network interface configuration.

    - Example: ifconfig or ip addr
- netstat: Network statistics.
    - Example: netstat -tuln
- ssh: Secure shell for remote access.
    - Example: ssh username@hostname
- curl: Transfer data using URLs.
    - Example: curl https://example.com

Compression and Archiving:

- tar: Create and extract archive files.
    - Example: tar -cvf archive.tar files
- gzip / gunzip: Compress or decompress files.
    - Example: gzip file.txt

Disk Usage:

- df: Display disk space usage.
    - Example: df -h
- du: Display file and directory space usage.
    - Example: du -sh /path/to/directory

System Information:

- uname: Display system information.
    - Example: uname -a
- lsb_release: Display distribution-specific information.
    - Example: lsb_release -a
- uptime: Display system uptime.
    - Example: uptime

User and Group Management:

- useradd: Add a new user.
    - Example: useradd newuser
- passwd: Change user password.

    - Example: passwd username
- groupadd: Add a new group.
    - Example: groupadd newgroup

Disk and Filesystem:

- fdisk: Manipulate disk partition table.
    - Example: fdisk /dev/sdX
- mount: Mount a filesystem.
    - Example: mount /dev/sdb1 /mnt
- umount: Unmount a filesystem.
    - Example: umount /mnt

File Searching:

- find: Search for files and directories.
    - Example: find /path/to/search -name "*.txt"
- locate: Quickly find files using a prebuilt index.
    - Example: locate file.txt

System Control:

- reboot: Reboot the system.
    - Example: reboot
- shutdown: Shutdown the system.
    - Example: shutdown -h now
- init: Change system runlevel (SysV init systems).
    - Example: init 3 (switch to text mode)

Text Editors:

- nano: Simple text editor.
    Example: nano file.txt
- vim / vi: Advanced text editor.
    Example: vim file.txt

File Transfer:

- scp: Securely copy files between hosts.
  - Example: scp file.txt remoteuser@remotehost:/path
- rsync: Efficiently transfer and synchronize files.
  - Example: rsync -av source/ destination/

Printing:

- lp: Print files.
  - Example: lp file.txt
- lpstat: Print queue status.
  - Example: lpstat -p

Monitoring:

- htop: Interactive process viewer.
  - Example: htop

Shell Features:

- history: Display command history.
  - Example: history
- !!: Execute the last command.
  - Example: !!
- Ctrl + R: Search command history.
  - Example: Press Ctrl + R, type search term, press Enter.

Remote Access:

- ssh: Secure shell for remote access.
  - Example: ssh user@host

- scp: Securely copy files between hosts.
    - Example: scp file.txt user@host:/path

File Permissions:

- chmod: Change file permissions.
    - Example: chmod 755 file.txt
- chown: Change file ownership.
    - Example: chown user:group file.txt

Compression and Archiving:

- tar: Create and extract archive files.
    - Example: tar -cvf archive.tar files
- gzip / gunzip: Compress or decompress files.
    - Example: gzip file.txt

Process Management:

- ps: Display information about running processes.
    - Example: ps aux
- kill: Terminate processes by ID or name.
    - Example: kill PID

Package Management:

- apt: Advanced Package Tool (Debian-based systems).
    - Example: apt install package_name
- yum: Package manager (RHEL-based systems).
    - Example: yum install package_name
- dnf: Next-generation package manager (Fedora).
    - Example: dnf install package_name

Networking:

- ping: Send ICMP echo requests to a host.
    - Example: ping google.com
- ifconfig / ip: Network interface configuration.
    - Example: ifconfig or ip addr
- netstat: Network statistics.
    - Example: netstat -tuln

Text Manipulation:

- cat: Concatenate and display file content.
    - Example: cat file.txt
- grep: Search for a pattern in files.
    - Example: grep "pattern" file.txt
- sed: Stream editor for text manipulation.
    - Example: sed 's/old/new/' file.txt
- awk: Text processing tool.
    - Example: awk '{print $1}' file.txt

Disk Usage:

- df: Display disk space usage.
    - Example: df -h
- du: Display file and directory space usage.
    - Example: du -sh /path/to/directory

System Information:

- uname: Display system information.
    - Example: uname -a
- lsb_release: Display distribution-specific information.
    - Example: lsb_release -a
- uptime: Display system uptime.
    - Example: uptime

User and Group Management:

- useradd: Add a new user.
  - Example: useradd newuser
- passwd: Change user password.
  - Example: passwd username
- groupadd: Add a new group.
  - Example: groupadd newgroup

Disk and Filesystem:

- fdisk: Manipulate disk partition table.
  - Example: fdisk /dev/sdX
- mount: Mount a filesystem.
  - Example: mount /dev/sdb1 /mnt
- umount: Unmount a filesystem.
  - Example: umount /mnt

File Searching:

- find: Search for files and directories.
  - Example: find /path/to/search -name "*.txt"
- locate: Quickly find files using a prebuilt index.
  - Example: locate file.txt

System Control:

- reboot: Reboot the system.
  - Example: reboot
- shutdown: Shutdown the system.
  - Example: shutdown -h now
- init: Change system runlevel (SysV init systems).
  - Example: init 3 (switch to text mode)

Text Editors:

- nano: Simple text editor.
  - Example: nano file.txt
- vim / vi: Advanced text editor.
  - Example: vim file.txt

File Transfer:

- scp: Securely copy files between hosts.
  - Example: scp file.txt remoteuser@remotehost:/path
- rsync: Efficiently transfer and synchronize files.
  - Example: rsync -av source/ destination/

Printing:

- lp: Print files.
  - Example: lp file.txt
- lpstat: Print queue status.
  - Example: lpstat -p

Please note that this is a comprehensive list, and you might not need to use all these commands in your daily tasks. It's also recommended to consult the respective command's manual pages (man command) for detailed information and options.