IP Address Commands

1. **`ip a`**
   Displays all network interfaces and their IP addresses.
   Example:
2. `ip a`

   Output shows details like interface names (`eth0`, `lo`), IP addresses, and status (`UP`, `DOWN`).

3. **`hostname -I`**
   Prints only the current IP address(es) of the machine.
   Example:
4. `hostname -I`

   Output:

   ```
   192.168.1.100 10.0.0.1
   ```

Working with `/etc/passwd`

3. **`cut -d ":" -f 1,6 /etc/passwd`**
   Extracts specific fields from the `/etc/passwd` file using `:` as the delimiter.
   Example:
4. `cut -d ":" -f 1,6 /etc/passwd`

   Output:

   ```
   root:/root
   user1:/home/user1
   ```

5. **`cat /etc/passwd`**
   Displays the entire contents of `/etc/passwd`.
   Example:
6. `cat /etc/passwd`
7. **`less /etc/passwd`**
   Opens `/etc/passwd` for viewing one screen at a time. Use `q` to quit.
   Example:
8. `less /etc/passwd`
9. **`head /etc/passwd`**
   Prints the first 10 lines of `/etc/passwd`.
   Example:
10. `head /etc/passwd`

11. **`head -n 12 /etc/passwd`**
Prints the first 12 lines of `/etc/passwd`.
Example:
```
12. head -n 12 /etc/passwd
```
13. **`tail /etc/passwd`**
Displays the last 10 lines of `/etc/passwd`.
Example:
```
14. tail /etc/passwd
```
15. **`tail -f /etc/passwd`**
Continuously monitors the file for changes and displays updates in real-time.
Example:
```
16. tail -f /etc/passwd
```

System Monitoring Commands

10. **`top`**
Displays running processes in real-time with resource usage.
Example:
```
11. top
```
12. **`ps aux`**
Lists all running processes with details like user, PID, and CPU usage.
Example:
```
13. ps aux
```
14. **`ps aux | grep <pname>`**
Searches for processes matching `<pname>`.
Example:
```
15. ps aux | grep apache
```
16. **`kill <pid>`**
Terminates a process by its Process ID (PID).
Example:
```
17. kill 1234
```
18. **`kill -9 <pid>`**
Forcibly terminates a process.
Example:
```
19. kill -9 1234
```

File Compression and Archiving

15. **File Types:**
    - `.tar`: Archive without compression.
    - `.tar.gz` or `.tgz`: Compressed with `gzip`.
    - `.tar.bz2`: Compressed with `bzip2`.

16.**Create a `.tar` archive:**

```
17. tar cvf file1.tar file1.txt
```

18.**Create a `.tar.gz` archive:**

```
19. tar cvfz file.tar.gz file.txt
```

20.**Extract files from a `.tar` archive:**

```
21. tar xvf file.tar
```

22.**Extract files from a `.tar.gz` archive:**

```
23. tar xvfz file.tar.gz
```

Disk Space and Usage

20.**`df -ht`**
Displays disk space in human-readable format, grouped by type.
Example:

```
21. df -ht
```

22.**`du -h`**
Shows disk usage of the current folder.
Example:

```
23. du -h
```

24.**`du -d2 -h`**
Displays disk usage up to 2 levels deep in human-readable format.
Example:

```
25. du -d2 -h
```

26.**`free`**
Displays memory usage and available free space.
Example:

```
27. free -h
```

Networking

24.**`netstat -ru`**
Displays the system’s routing table.
Example:

```
25. netstat -ru
```

26.**`netstat -an | grep LISTEN`**
Lists all listening ports.
Example:

```
27. netstat -an | grep LISTEN
```

Secure Copy Protocol (SCP)

26.**Copy a file from local to remote:**

```
27. scp file.txt user@remote_host:/path/to/destination
```

28. **Copy a file from remote to local:**
```
29. scp user@remote_host:/path/to/file.txt /local/path/
```
30. **Copy a directory recursively:**
```
31. scp -r /local/dir user@remote_host:/remote/dir
```
32. **Use a specific SSH port for SCP:**
```
33. scp -P 2222 file.txt user@remote_host:/path/
```

What is wget in Ubuntu?

wget is a command-line utility in Ubuntu (and other Linux distributions) used for **downloading files from the web**. It supports downloading over HTTP, HTTPS, and FTP protocols. wget is non-interactive, which makes it ideal for automating downloads and background tasks.

Why Use wget?

1. **Simple and Fast**: Easily download files without needing a browser.
2. **Supports Multiple Protocols**: Can handle HTTP, HTTPS, and FTP downloads.
3. **Resumes Interrupted Downloads**: Useful when a download is interrupted (e.g., due to network issues).
4. **Recursive Download**: Can download entire websites or directories.

Basic Syntax of wget

```
wget [options] <URL>
```

Common Usage Examples

1. Download a File

To download a single file:

```
wget https://example.com/file.zip
```

- This will download file.zip from example.com.

2. Download and Save the File with a Custom Name

You can specify the name to save the file as:

```
wget -O custom_filename.zip https://example.com/file.zip
```

- `-O`: Specifies the output file name.

3. Download a File in the Background

To download a file in the background (useful for large downloads):

```
wget -b https://example.com/largefile.zip
```

- `-b`: Runs the download in the background.

4. Resume an Interrupted Download

If a download was interrupted, you can resume it with the `-c` option:

```
wget -c https://example.com/largefile.zip
```

- `-c`: Resumes the download if it was previously stopped.

5. Download a Whole Website (Recursive Download)

To download an entire website or a directory recursively:

```
wget -r https://example.com
```

- `-r`: Downloads the entire website (recursively).

6. Limit Download Speed

To limit the download speed to a specific rate:

```
wget --limit-rate=200k https://example.com/file.zip
```

- `--limit-rate=200k`: Limits the download speed to 200 KB/s.

7. Download Multiple Files from a File

If you have a text file containing URLs, you can download all the URLs listed:

```
wget -i urls.txt
```

- `-i`: Takes a file (in this case, `urls.txt`) containing a list of URLs.

8. Download Over HTTPS with SSL Certificate Verification

You can ensure that `wget` verifies the SSL certificate:

wget --https-only https://example.com/file.zip

- --https-only: Ensures only HTTPS downloads.

Advanced Options

- **Set User-Agent**: Some websites may block downloads based on the user-agent. You can specify a custom user-agent.
- wget --user-agent="Mozilla/5.0" https://example.com/file.zip
- **Download From FTP**: If you're downloading from an FTP server:
- wget ftp://ftp.example.com/file.zip
- **Download with Authentication**: If the URL requires basic HTTP authentication:
- wget --user=USERNAME --password=PASSWORD https://example.com/file.zip

---

Summary of Common wget Options

Option	Description
-O <file>	Save the downloaded file with a specific name.
-c	Resume an interrupted download.
-b	Run the download in the background.
-r	Download a website recursively.
-i <file>	Download all URLs listed in a file.
--limit-rate	Limit download speed (e.g., 200k, 1M).
--https-only	Ensure the download is over HTTPS.
--user-agent	Specify a custom user-agent string.

Summary

- **What**: wget is a command-line tool for downloading files from the web.
- **Why**: It's fast, simple, and supports a variety of options like resuming downloads, recursive downloading, and limiting speed.
- **How**: Use wget <URL> to download a file, and customize with options like -c to resume or -r to download entire websites.

`sed` (Stream Editor) is a powerful text-processing tool in Linux. It is primarily used to search, replace, insert, and delete text in files or streams.

What is `sed`?

- **Stream Editor:** Processes text line by line.
- **Non-interactive:** Performs operations directly on the input without opening an editor.
- Commonly used for:
    - Substitutions
    - Deleting lines
    - Adding or modifying text
    - Extracting parts of a file

Basic Syntax

```
sed [options] 'command' file
```

- **Options:**
    - `-i`: Edits the file in place.
    - `-n`: Suppresses automatic printing of the pattern space.
    - `-e`: Allows specifying multiple commands.
    - `-f`: Reads commands from a file.

Key Commands

1. **Substitute (`s`)**
   Replaces occurrences of a pattern with a specified string.

   Syntax:

   ```
   sed 's/<pattern>/<replacement>/g' file
   ```

    - `g`: Global replacement (replace all occurrences).
    - Without `g`, only the first match on each line is replaced.

   Example:

   ```
   sed 's/Ubuntu/Linux/' file.txt
   ```

    - Replaces the first occurrence of "Ubuntu" with "Linux" on each line of `file.txt`.

   Global Replacement Example:

   ```
   sed 's/Ubuntu/Linux/g' file.txt
   ```

---

2. **In-Place Editing (`-i`)**
   Modifies the file directly without creating a backup.

   Example:

   ```
   sed -i 's/Ubuntu/Linux/g' file.txt
   ```

---

3. **Delete Lines (`d`)**
   Deletes specific lines.

   Syntax:

   ```
   sed '<line_number>d' file
   ```

   Examples:

   - Delete the 3rd line:
   - `sed '3d' file.txt`
   - Delete lines 2 to 5:
   - `sed '2,5d' file.txt`
   - Delete all lines containing "error":
   - `sed '/error/d' file.txt`

---

4. **Print Specific Lines (`p`)**
   Prints specified lines.

   Syntax:

   ```
   sed -n '<line_number>p' file
   ```

   Examples:

   - Print the 1st line:
   - `sed -n '1p' file.txt`
   - Print lines 2 to 4:
   - `sed -n '2,4p' file.txt`

---

5. **Insert Text (`i`)**
   Inserts a line of text before a specified line.

   Syntax:

   ```
   sed '<line_number>i <text>' file
   ```

Example:

```
sed '3i This is a new line' file.txt
```

- Adds "This is a new line" before the 3rd line.

---

6. **Append Text (`a`)**
   Appends a line of text after a specified line.

   Syntax:

   ```
   sed '<line_number>a <text>' file
   ```

   Example:

   ```
   sed '3a This is appended text' file.txt
   ```

   - Adds "This is appended text" after the 3rd line.

---

7. **Replace Line (`c`)**
   Replaces a specific line with new text.

   Syntax:

   ```
   sed '<line_number>c <text>' file
   ```

   Example:

   ```
   sed '3c This is the new content for line 3' file.txt
   ```

---

8. **Replace Using Regex**
   You can use regular expressions for advanced matching.

   Example: Replace all numbers with `#`:

   ```
   sed 's/[0-9]/#/g' file.txt
   ```

---

9. **Multiple Commands (`-e`)**
   Run multiple commands in a single execution.

   Example:

```
sed -e 's/Ubuntu/Linux/' -e '2d' file.txt
```

- Replaces "Ubuntu" with "Linux".
- Deletes the 2nd line.

---

10. **Read From File (`-f`)**
    Apply commands from a file.

    Example:
    Create a file named `commands.sed`:

    ```
    s/Ubuntu/Linux/
    3d
    ```

    Run:

    ```
    sed -f commands.sed file.txt
    ```

---

11. **Print Lines Matching a Pattern (`/pattern/p`)**
    Prints lines containing a specific pattern.

    Example:

    ```
    sed -n '/error/p' file.txt
    ```

    - Prints lines containing "error".

---

12. **Change Delimiter in `sed`**
    If your pattern contains `/`, you can change the delimiter (e.g., `|`).

    Example:

    ```
    sed 's|/home/user|/data/new|' file.txt
    ```

---

13. **Save Output to a New File**
    Use redirection to save changes to a new file.

    Example:

    ```
    sed 's/Ubuntu/Linux/' file.txt > newfile.txt
    ```

Practical Examples

1. Replace all occurrences of "foo" with "bar" in a file:
2. `sed 's/foo/bar/g' file.txt`
3. Delete empty lines:
4. `sed '/^$/d' file.txt`
5. Highlight matching patterns:
6. `sed 's/Ubuntu/[Ubuntu]/g' file.txt`
7. Extract lines containing "error" and save them to a new file:
8. `sed -n '/error/p' file.txt > errors.txt`
9. Insert a header at the top of a file:
10. `sed '1i # Header: File Info' file.txt`

---

1. Downloading Files

`wget`
`wget` is a command-line utility for downloading files from the internet.

- Example:
- `wget -O a.txt https://github.com`
  - Downloads the content of `https://github.com` and saves it as `a.txt`.

`curl`
`curl` is a more advanced tool than `wget`. It can download files and interact with APIs.

- Example:
- `curl https://get.docker.com -o get-docker.sh`
  - Downloads the file `get-docker.sh` from the specified URL.

---

2. Privilege Management

- **`sudo`**
  Temporarily grants administrative privileges to run commands as the root user.
  - Example:
  - `sudo apt update`
    - Runs the `apt update` command with elevated privileges.
- **`su`**
  Switches to another user account (including root).
  - Example:
  - `su username`
    - Switches to the user `username`.

---

3. Navigation & File Operations

- **`ls`**
  Lists directory contents.
  - Example:
  - `time ls -a`
    - Lists all files (including hidden ones) and measures the execution time.
- **`mv`**
  Moves or renames files.
  - Example:
  - `mv file1.txt file2.txt`
    - Renames `file1.txt` to `file2.txt`.

---

4. Searching

- **`locate`**
  Searches for files by name.
  - Example:
  - `sudo apt update -y && sudo apt install locate`
  - `locate file.txt`
    - Installs the `locate` command and searches for `file.txt`.
- **`grep` and `fgrep`**
  Searches for patterns in files.
  - Example:
  - `grep "error" file.txt`
    - Finds lines containing "error" in `file.txt`.
  - **`fgrep`** (fixed string grep) does not recognize regular expressions, making it faster for simple searches.

---

5. File Analysis

- **`cmp`**
  Compares two files byte by byte.
  - Example:
  - `cmp file1.txt file2.txt`
    - Checks for differences between the two files.
- **`diff`**
  Compares two files line by line.
  - Example:

   - `diff file1.txt file2.txt`
     - Displays the differences in text format.
- **`wc`**
  Counts lines, words, and characters in a file.
   - Example:
   - `wc file.txt`
     - Prints the number of lines, words, and characters in `file.txt`.

---

6. File Transformation

- **`sed`**
  Stream editor for text manipulation.
   - Example:
   - `sed 's/old/new/g' file.txt`
     - Replaces all occurrences of "old" with "new" in `file.txt`.
- **`cut`**
  Extracts specific fields from a file.
   - Example:
   - `cut -d':' -f1 /etc/passwd`
     - Displays the first field (username) of `/etc/passwd`.
- **`tr`**
  Translates or deletes characters.
   - Examples:
   - `echo "hello world" | tr '[:lower:]' '[:upper:]'`
     - Converts "hello world" to uppercase.
   - `echo "UST GLOBAL" | tr -d '[:space:]'`
     - Removes spaces from "UST GLOBAL".

---

7. Process Management

- **`ps`**
  Displays information about processes.
   - Example:
   - `ps aux`
     - Displays detailed process information.
   - Example with filtering:
   - `ps aux | grep docker`
     - Shows processes related to Docker.
- **`kill`**
  Terminates a process by its PID (Process ID).
   - Example:
   - `kill -9 PID`

        - Forcefully kills the process with the specified PID.

---

8. Disk & System Management

- **`df`**
  Displays disk space usage.
  - Example:
  - `df -ht`
    - Shows disk usage in a human-readable format, filtered by type.
- **`du`**
  Displays folder sizes.
  - Examples:
  - `du -h`
    - Shows folder sizes in human-readable format.
  - `du -d2 -h`
    - Shows folder sizes up to 2 levels deep.
- **`free`**
  Displays memory usage.
  - Example:
  - `free`

---

9. File Permissions

- **`chown`**
  Changes the owner of a file.
  - Example:
  - `sudo chown user:group file.txt`
- **`ln`**
  Creates links (hard or symbolic).
  - Example:
  - `ln -s /usr/bin/ls myls`
    - Creates a symbolic link to `/usr/bin/ls` named `myls`.

---

10. Scheduling Tasks

- **`crontab`**
  Automates tasks by scheduling commands.
  - Example:
  - `crontab -e`
    - Edits the cron jobs.

- Example to run every minute:
- `*/1 * * * * echo "Hello World" >> /home/user/hello.txt`

---

11. Network Commands

- **`ip a`**
  Displays IP addresses of network interfaces.
- **`ssh`**
  Securely connects to a remote system.
  - Example:
  - `ssh user@172.18.228.16`
    - Logs in to the remote system at the specified IP.

---

12. Docker Management

- **`systemctl`**
  Manages services.
  - Examples:
  - `systemctl status docker`
  - `systemctl stop docker`
  - `systemctl disable docker`
    - Displays the status, stops, and disables the Docker service.