1.Linux Commands

Here's a list of basic Linux commands that are commonly used in DevOps:

1. ls: List files and directories in the current directory.
2. cd: Change the working directory.
3. pwd: Print the working directory (current directory).
4. mkdir: Create a new directory.
5. rmdir: Remove a directory (only if it's empty).
6. rm: Remove files or directories.
7. cp: Copy files or directories.
8. mv: Move or rename files or directories.
9. touch: Create an empty file or update the timestamp of an existing file.
10. cat: Display the contents of a file.
11. less: View a file one page at a time (scrollable).
12. head: Display the beginning lines of a file.
13. tail: Display the ending lines of a file.
14. grep: Search for a pattern in files or output.
15. find: Search for files and directories based on various criteria.
16. chmod: Change file permissions.
17. chown: Change file ownership.
18. ps: Display information about running processes.
19. top: Monitor system processes and resource usage in real-time.
20. kill: Terminate processes by their PID (Process ID).
21. df: Display disk space usage on file systems.
22. du: Estimate file and directory space usage.
23. tar: Archive files and directories into a tarball.
24. gzip: Compress or decompress files using gzip compression.
25. scp: Securely copy files between local and remote systems over SSH.
26. ssh: Securely connect to a remote server over SSH.
27. ping: Test network connectivity to a host.
28. ifconfig / ip: Display or configure network interfaces (ip is a newer alternative to ifconfig).
29. netstat: Display network statistics and active connections.
30. systemctl: Control system services in systemd-based Linux distributions.
31. journalctl: View system logs (systemd journal).
32. cron: Schedule and automate tasks to run at specified intervals.
33. wget: Download files from the web via HTTP or FTP.

2.Linux Shell Scripting:

1. Variables: Declaring and using variables to store data and values.
2. Conditional Statements: Using if, elif, and else to perform actions based on conditions.
3. Loops: Employing for loops and while loops for repetitive tasks.
4. Functions: Creating functions to modularize code and reuse it.
5. Command Substitution: Capturing the output of commands and using it in scripts.
6. Input/Output: Handling standard input (stdin), output (stdout), and error (stderr).
7. File Operations: Creating, reading, writing, and manipulating files.
8. File Permissions: Changing and managing file permissions.

9. String Manipulation: Performing operations on strings (concatenation, substitution, etc.).
10. Arithmetic Operations: Doing arithmetic calculations.
11. Command-Line Arguments: Passing arguments to scripts from the command line.
12. Arrays: Working with arrays to store multiple values in a single variable.
13. Pipes: Using pipes to connect the output of one command to the input of another.
14. Redirects: Redirecting input and output streams (>, >>, <, 2>, 2>>, etc.).
15. grep, sed, and awk: Advanced text processing with these powerful commands.
16. cut: Extracting specific fields or columns from a line of text.
17. sort: Sorting lines of text.
18. uniq: Removing duplicate lines from sorted text.
19. tar: Archiving and extracting files and directories.
20. gzip: Compressing and decompressing files.
21. find: Searching for files and directories based on various criteria.
22. xargs: Taking input and executing commands based on that input.
23. curl: Interacting with URLs to transfer data.
24. jq: Processing JSON data on the command line.

3.Package Manager

apt (Advanced Package Tool) - Debian/Ubuntu:

1. sudo apt update: Update the package list to fetch the latest available versions.
2. sudo apt upgrade: Upgrade installed packages to the latest versions.
3. sudo apt install package_name: Install a new package.
4. sudo apt remove package_name: Remove a package while keeping its configuration files.
5. sudo apt purge package_name: Completely remove a package and its configuration files.
6. sudo apt search package_name: Search for packages matching the given name.
7. sudo apt show package_name: Display detailed information about a package.

4.Process Management

1. ps: Display information about running processes.

2. ps: Show information about the processes running in the current terminal session.
3. ps aux: Display a detailed list of all running processes on the system.
4. ps -ef: Another way to show a detailed list of all running processes.
5. ps -e | grep process_name: Search for a specific process by name.
6. top: Monitor system processes and resource usage in real-time.

7. top: Display a dynamic view of the system's processes, CPU usage, and memory usage.
8. Press 'q' to exit the top command.
9. htop: An alternative to top with a more user-friendly interface.

10. Install htop (if not already installed): sudo apt install htop (for Debian/Ubuntu) or sudo yum install htop (for Red Hat/Fedora/CentOS).
11. htop: Launch htop to monitor processes and resource usage.
12. kill: Terminate processes by their PID (Process ID).

13. kill PID: Terminate the process with the specified PID.
14. killall process_name: Terminate all processes with a specific name.
15. systemctl: Control system services in systemd-based Linux distributions.

16. sudo systemctl start service_name: Start a service.
17. sudo systemctl stop service_name: Stop a service.
18. sudo systemctl restart service_name: Restart a service.
19. sudo systemctl status service_name: Check the status of a service.
20. sudo systemctl enable service_name: Enable a service to start on boot.
21. sudo systemctl disable service_name: Disable a service from starting on boot.
22. service: An older command used to manage services on Linux.

23. sudo service service_name start: Start a service.
24. sudo service service_name stop: Stop a service.
25. sudo service service_name restart: Restart a service.
26. sudo service service_name status: Check the status of a service.
27. killall: Terminate processes by name.

28. killall process name: Terminate all processes with a specific name.

5.File Management

1. ls: List files and directories with their permissions and ownership.

2. ls -l: Display detailed file and directory listing with permissions, ownership, and other information.
3. ls -la: Display all files, including hidden files, with detailed information.
4. chmod: Change file permissions.

5. chmod permissions file: Change permissions of a specific file.
6. chmod permissions directory: Change permissions of a specific directory.
7. Permissions can be represented using numeric or symbolic notation.
8. Numeric notation: For example, chmod 644 file sets read and write permissions for the owner and read-only permissions for the group and others.

9. Symbolic notation: For example, chmod u+r file adds read permission to the owner, chmod g-w file removes write permission from the group, and chmod o+x file adds execute permission to others.
10. chown: Change file ownership.

11. chown owner:group file: Change both the owner and group of a file.
12. chown owner file: Change the owner of a file while keeping the group unchanged.
13. chown :group file: Change the group of a file while keeping the owner unchanged.
14. chgrp: Change file group ownership.

15. chgrp group file: Change the group of a file.
16. umask: Set the default permissions for new files and directories.

17. umask: Display the current umask value.
18. umask new_value: Set a new umask value. For example, umask 027 sets the default permissions for new files to 640 and new directories to 750.
19. su: Temporarily switch to another user.

20. su username: Switch to another user account.
21. Use su - to switch to another user along with their environment settings.
22. sudo: Execute a command with superuser (root) privileges.

23. sudo command: Run a command with root privileges.
24. sudo -u username command: Run a command as a specific user.
25. chattr: Change file attributes to make files immutable or append-only.

26. sudo chattr +i file: Make a file immutable (cannot be deleted or modified).
27. sudo chattr +a file: Make a file append-only (can only be opened in append mode).

6.Text processing

1. grep: Search for patterns in text.

2. grep pattern file: Search for a specific pattern in a file.
3. grep -r pattern directory: Recursively search for a pattern in all files within a directory.
4. grep -i pattern file: Perform a case-insensitive search.
5. grep -v pattern file: Invert the match and display lines that do not contain the pattern.
6. sed: Stream Editor for text manipulation.

7. sed 's/pattern/replacement/' file: Substitute the first occurrence of the pattern with the replacement in a file.
8. sed 's/pattern/replacement/g' file: Substitute all occurrences of the pattern with the replacement in a file.
9. sed -i 's/pattern/replacement/' file: Perform an in-place substitution, directly modifying the file.
10. awk: Text processing tool for data extraction and reporting.

11. awk '{print $1}' file: Print the first column of data in a file (space-separated by default).
12. awk -F',' '{print $2}' file: Set the field separator to comma and print the second column of data.
13. awk '/pattern/ {print $3}' file: Print the third column if a line matches the pattern.
14. cut: Extract specific columns from a file.

15. cut -d',' -f2 file: Extract the second column of data from a CSV file.
16. cut -c1-5 file: Extract the first five characters from each line.
17. sort: Sort lines of text.

18. sort file: Sort lines in ascending order (lexicographically).
19. sort -r file: Sort lines in descending order.
20. sort -n file: Sort lines numerically.
21. uniq: Remove duplicate lines from sorted text.

22. uniq file: Remove adjacent duplicate lines from a sorted file.
23. sort file | uniq: Sort and then remove duplicate lines from a file.
24. wc: Word, line, character, and byte count.

25. wc file: Count the number of lines, words, and characters in a file.
26. tr: Translate or delete characters.

27. tr 'A-Z' 'a-z' < file: Convert uppercase characters to lowercase in a file.
28. tr -d '\r' < file: Remove carriage return characters (Windows line endings) from a file.

7.Networking

1. ifconfig / ip: Display or configure network interfaces.

2. ifconfig: Show the configuration of all network interfaces (deprecated on some Linux distributions).
3. ip addr show: Show the configuration of all network interfaces (modern replacement for ifconfig).
4. ip addr show interface_name: Show the configuration of a specific network interface.
5. sudo ifconfig interface_name up/down: Enable or disable a network interface.
6. sudo ip link set interface_name up/down: Enable or disable a network interface (modern replacement for ifconfig).
7. ping: Test network connectivity to a host.

8. ping hostname: Send ICMP echo requests to the specified host.
9. ping -c count hostname: Send a specific number of echo requests.
10. ping -i interval hostname: Set the time interval between echo requests.
11. traceroute / tracepath: Trace the route packets take to a destination.

12. traceroute hostname: Print the route packets take to the specified host.
13. tracepath hostname: A simplified version of traceroute.
14. netstat: Display network statistics and active connections.

15. netstat -tuln: Display TCP and UDP listening ports.
16. netstat -ant: Display all TCP connections (active and listening).
17. netstat -anu: Display all UDP connections (active and listening).
18. nslookup / dig: DNS (Domain Name System) query tools.

19. nslookup hostname: Perform DNS lookups to translate hostnames to IP addresses.
20. dig hostname: Perform more detailed DNS lookups and retrieve additional information.
21. ssh: Securely connect to a remote server over SSH.

22. ssh user@hostname: Connect to a remote server with SSH.
23. scp: Securely copy files between local and remote systems over SSH.

24. scp local_file user@hostname:remote_directory: Copy a file from the local system to a remote system.
25. scp user@hostname:remote_file local_directory: Copy a file from a remote system to the local system.
26. curl: Interact with URLs to transfer data.

27. curl url: Retrieve data from a URL.
28. curl -O url: Download a file from a URL and save it with the original name.
29. iptables / firewalld: Manage firewall rules.

30. iptables: A powerful firewall utility (previously used, now being replaced by nftables on some distributions).
31. firewall-cmd: Command-line tool to configure firewalld, the default firewall management tool on newer distributions.
32. route: Display or modify the IP routing table.

33. route -n: Display the routing table with IP addresses (deprecated on some distributions).
34. ip route show: Display the routing table with IP addresses (modern replacement for route).

8.SSH: Securely connecting to remote servers using SSH and managing SSH keys.

SSH (Secure Shell) is a critical tool for DevOps engineers, as it allows them to securely connect to remote servers, transfer files, and manage server configurations. Here are some common SSH commands for DevOps:

ssh: Securely connect to a remote server over SSH.

1. ssh user@hostname: Connect to a remote server as a specific user.
2. ssh -p port user@hostname: Connect to a remote server on a non-default SSH port.
3. ssh-keygen: Generate SSH key pairs for secure authentication.

4. ssh-keygen: Generate a new SSH key pair (by default, RSA keys).
5. ssh-keygen -t key_type: Generate a specific type of key (e.g., RSA, DSA, ECDSA, or ED25519).
6. ssh-keygen -b bits: Set the number of bits for the key (e.g., 4096).
7. ssh-copy-id: Copy your public key to a remote server for passwordless login.

8. ssh-copy-id user@hostname: Copy your public key to the remote server's ~/.ssh/authorized_keys file.
9. ssh-agent: Manage SSH private keys.

10. ssh-agent: Start the SSH agent (an authentication agent that holds private keys).
11. ssh-add: Add your private key to the SSH agent for authentication.

12. scp: Securely copy files between local and remote systems over SSH.

13. scp local_file user@hostname:remote_directory: Copy a file from the local system to a remote system.
14. scp user@hostname:remote_file local_directory: Copy a file from a remote system to the local system.
15. sftp: Securely transfer files between local and remote systems over SSH.

16. sftp user@hostname: Start an interactive session for secure file transfer.
17. sshd: SSH server daemon, responsible for accepting incoming SSH connections.

18. sudo systemctl start sshd: Start the SSH server (use restart instead of start to restart it).
19. sudo systemctl stop sshd: Stop the SSH server.
20. ~/.ssh/config: Customize SSH client settings.

21. Edit the ~/.ssh/config file to set options for specific hosts, such as defining aliases, custom ports, and identity files.

9.Bash Shell

1. Aliases: Creating shortcuts for frequently used commands.

2. alias alias_name='command': Create an alias for a command.
3. alias ll='ls -alF': Create an alias 'll' for the 'ls -alF' command.
4. To make aliases permanent, add them to the ~/.bashrc or ~/.bash_aliases file.
5. Environment Variables: Setting variables that affect the behavior of programs and scripts.

6. export VAR_NAME=value: Set an environment variable.
7. export PATH=$PATH:/path/to/new/directory: Add a directory to the system's PATH variable.
8. To make environment variables permanent, add them to the ~/.bashrc or ~/.bash_profile file.
9. PS1: Customizing the shell prompt.

10. export PS1="your_prompt_here": Set a custom prompt.
11. Common placeholders for the prompt:
12. \u: Username
13. \h: Hostname
14. \w: Current working directory
15. \n: Newline
16. Example: export PS1="\u@\h:\w\n$ " will display the prompt as username@hostname:current_directory\n$.
17. ~/.bashrc: The Bash configuration file for interactive non-login shells.

18. Edit the ~/.bashrc file to set up aliases, environment variables, and custom functions that apply to interactive shells.
19. ~/.bash_profile (or ~/.bash_login): The Bash configuration file for login shells.

20. Edit the ~/.bash_profile file to set up environment variables that should only be set once during login.
21. source: Reload the current shell environment.

22. source ~/.bashrc: Reload the ~/.bashrc file to apply changes immediately without opening a new shell.
23. /etc/profile: The system-wide Bash profile configuration file.

24. Edit the /etc/profile file to set environment variables that apply to all users on the system.

10.Cron Jobs

1. crontab: The command to manage user-specific cron jobs.

2. crontab -e: Edit the user's crontab file to add or modify cron jobs.
3. crontab -l: View the user's current cron jobs.
4. crontab -r: Remove the user's crontab (all scheduled tasks).
5. Crontab Format: Understanding the cron schedule syntax.

6. A cron job is defined by five time and date fields, followed by the command to be executed.
7. The syntax is as follows: minute hour day_of_month month day_of_week command.
8. Each field can take specific values, including wildcards (*) and ranges (e.g., 0-59, 1-12, etc.).
9. Common Cron Scheduling Expressions:

   - * * * *: Run every minute.

10. 0 * * * *: Run at the beginning of every hour.
11. 0 0 * * *: Run once a day at midnight.
12. 0 2 * * 1: Run every Monday at 2:00 AM.
13. */5 * * * *: Run every 5 minutes.
14. 0 0 1 * *: Run on the first day of every month.
15. Logging: Redirecting cron job output to log files.

   - * * * * command >> /path/to/logfile.log 2>&1: Redirect standard output and error to a log file.

16. This helps in troubleshooting and monitoring cron jobs.

17. Anacron: A variation of cron that handles missed jobs.

18. anacron: A tool that ensures missed cron jobs are executed when the system is up.
19. System-Wide Cron Jobs:

20. For system-wide cron jobs, add scripts to the /etc/cron.hourly, /etc/cron.daily, /etc/cron.weekly, or /etc/cron.monthly directories.
21. These scripts will run hourly, daily, weekly, or monthly, respectively.
22. Cron Environment Variables:

23. Cron jobs often have a limited environment compared to the user's interactive shell. Set environment variables explicitly in the cron job script if needed.
24. User
    useradd: Add a new user account.

25. sudo useradd username: Create a new user account with the specified username.
26. passwd: Set or change a user's password.

27. sudo passwd username: Set or change the password for the specified user.
28. usermod: Modify user account attributes.

29. sudo usermod -aG groupname username: Add a user to an existing group.
30. sudo usermod -l new_username old_username: Rename a user.
31. userdel: Delete a user account.

32. sudo userdel username: Delete the specified user account.
33. sudo userdel -r username: Delete the user account and their home directory.
34. groupadd: Add a new group.

35. sudo groupadd groupname: Create a new group with the specified group name.
36. groupmod: Modify group attributes.

37. sudo groupmod -n new_groupname old_groupname: Rename a group.
38. groupdel: Delete a group.

39. sudo groupdel groupname: Delete the specified group.
40. id: Display user and group information.

41. id username: Display information about the specified user, including their groups.
42. chown: Change file ownership.

43. sudo chown username:groupname file: Change the owner and group of a file.
44. chmod: Change file permissions.

45. chmod permissions file: Change permissions of a specific file.
46. Permissions can be represented using numeric or symbolic notation.
47. su: Temporarily switch to another user.

48. su username: Switch to another user account.
49. Use su - to switch to another user along with their environment settings.
50. sudo: Execute a command with superuser (root) privileges.

51. sudo command: Run a command with root privileges.
52. sudo -u username command: Run a command as a specific user.

11.File Transfer

1. scp: Securely copy files between local and remote systems over SSH.

2. scp local_file user@hostname:remote_directory: Copy a file from the local system to a remote system.
3. scp user@hostname:remote_file local_directory: Copy a file from a remote system to the local system.
4. Example:

5. Copy a local file to a remote server: scp myfile.txt user@example.com:/path/to/destination/
6. Copy a remote file to the local system: scp user@example.com:/path/to/source/file.txt ~/destination/
7. Note: For larger file transfers and better sync capabilities, rsync is often preferred over scp.

8. rsync: Efficiently sync files between local and remote systems.

9. rsync options source destination: Synchronize files from source to destination.
10. Example:

11. Sync local files to a remote server: rsync -avz /path/to/source/ user@example.com:/path/to/destination/
12. Sync remote files to the local system: rsync -avz user@example.com:/path/to/source/ /path/to/destination/
13. Common options:

14. -a: Archive mode (Preserves permissions, ownership, timestamps, etc.).
15. -v: Verbose mode (Show details of the transfer).
16. -z: Compress files during transfer to reduce network usage

12.Monitoring and Alerting:

1. **Nagios**:
   - Nagios is a popular open-source monitoring system that can monitor hosts, services, and network devices.
   - Install Nagios:
     - Follow the installation instructions provided by the Nagios documentation for your specific operating system.
   - Configuration:
     - Customize monitoring configurations in Nagios by modifying the configuration files located in `/etc/nagios/`.
   - Monitoring Plugins:
     - Nagios uses plugins to check services and devices. Install monitoring plugins for the services you want to monitor.
2. **Prometheus**:
   - Prometheus is an open-source monitoring and alerting toolkit.
   - Install Prometheus:
     - Download Prometheus from the official website and follow the installation instructions.
   - Configuration:
     - Configure Prometheus by editing the `prometheus.yml` file to define targets (endpoints to scrape metrics) and other settings.
   - Alerting Rules:
     - Create alerting rules in the `rules` section of the `prometheus.yml` file.
3. **Grafana**:
   - Grafana is an open-source platform for visualizing and analyzing data.
   - Install Grafana:
     - Download Grafana from the official website and follow the installation instructions.
   - Integration with Prometheus:
     - Integrate Grafana with Prometheus as a data source to visualize and create dashboards for monitoring metrics.
4. **SystemD Service Management**:
   - Create SystemD service files to manage the monitoring and alerting components (e.g., Nagios, Prometheus, Grafana).
   - Start, stop, restart, and enable services using SystemD commands.
     - `sudo systemctl start service_name`: Start a service.
     - `sudo systemctl stop service_name`: Stop a service.
     - `sudo systemctl restart service_name`: Restart a service.
     - `sudo systemctl enable service_name`: Enable a service to start on boot.
5. **Alerting Tools**:

- Integrate alerting tools (e.g., AlertManager for Prometheus) to manage alerts and notifications.
- Configure alerting rules and receiver settings.

6. **Monitoring Agent Installation**:
   - For monitoring remote hosts, install monitoring agents (e.g., Node Exporter for Prometheus) on those hosts.
   - Configure the agent to expose metrics for monitoring.