



# Comprehensive Linux Operations

## Project Overview

This project spans various aspects of Linux system administration, including file management, user and group management, service control, process handling, and more. You will be completing tasks that simulate real-world scenarios, providing hands-on experience with Linux commands and configurations.

## Project Breakdown

### Part 1: Creating and Editing Text Files (20 minutes)

**Scenario:** You are tasked with documenting the configurations and settings for a new server. You'll use different text editors to create and update these documents.

#### 1. Using Nano

Create a file `server_config.txt` using Nano :

```
nano server_config.txt
```

```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ nano server_config.txt
einfochips@AHMLPT2603:~/Music/My_TASK 2$
```

Add the following content:

```
Server Name: WebServer01
IP Address: 192.168.1.100
```

OS: Ubuntu 20.04

- 
- Save and exit (Ctrl+O, Enter, Ctrl+X).

## 2. Using Vi

Edit the same file with Vi:

```
vi server_config.txt
```

- 

Append the following text:

```
Installed Packages: Apache, MySQL, PHP
```

- 
- Save and exit (Esc, :wq).

## 3. Using Vim

Further edit the file with Vim:

```
vim server_config.txt
```

- 

Add the following text:

```
Configuration Complete: Yes
```

- Save and exit (Esc, :wq).

## Part 2: User & Group Management (20 minutes)

**Scenario:** You need to set up user accounts and groups for a new team joining the project.

### 1. Adding/Removing Users

**Add a new user `developer`:**

```
sudo adduser developer  
Or sudo useradd developer
```

```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ sudo adduser developer
Adding user `developer' ...
Adding new group `developer' (1007) ...
Adding new user `developer' (1002) with group `developer' ...
The home directory `/home/developer' already exists. Not copying from `/etc/skel'.
adduser: Warning: The home directory `/home/developer' does not belong to the user you are currently creating.
New password:
BAD PASSWORD: The password is shorter than 8 characters
Retype new password:
passwd: password updated successfully
Changing the user information for developer
Enter the new value, or press ENTER for the default
    Full Name []: Developer
    Room Number []: 4
    Work Phone []: 4
    Home Phone []: 4
    Other []: 4
Is the information correct? [Y/n] Y
```

Remove the user **developer**:

```
sudo deluser developer
Or
sudo userdel developer
```

```
einfochips@AHMLPT2603:~$ sudo userdel developer
[sudo] password for einfochips:
```

## 2. Managing Groups

Create a group **devteam**:

```
sudo groupadd devteam
```

```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ sudo groupadd devteam
einfochips@AHMLPT2603:~/Music/My_TASK 2$
```

**Add the user `developer` to the `devteam` group:**

```
sudo usermod -aG devteam developer
```

`-a` (append): This option tells `usermod` to add the user to the supplementary group(s) without removing them from any other groups they are already a member of.

`-G` (groups): This option is used to specify the list of supplementary groups the user should be added to.

The `usermod` command in Linux is used to modify an existing user's account details. It allows administrators to change various attributes of a user, such as their username, home directory, shell, and group memberships.

Examples

```
Sudo usermod -l dev developer
```

```
Sudo usermod +L dev
```

```
Sudo usermod +U dev
```

```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ sudo groupadd devteam
einfochips@AHMLPT2603:~/Music/My_TASK 2$ sudo usermod -aG devteam developer
einfochips@AHMLPT2603:~/Music/My_TASK 2$
```

**Remove the user `developer` from the `devteam` group:**

```
sudo gpasswd -d developer devteam
```

```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ sudo gpasswd -d developer devteam
Removing user developer from group devteam
einfochips@AHMLPT2603:~/Music/My_TASK 2$
```

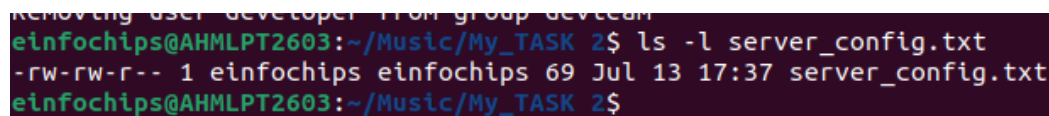
## Part 3: File Permissions Management (20 minutes)

**Scenario:** Ensure that only the appropriate users have access to specific files and directories.

### 1. Understanding File Permissions

View permissions for `server_config.txt`:

```
ls -l server_config.txt
```



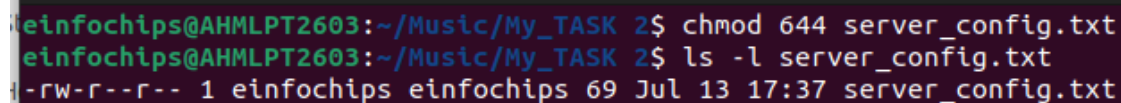
```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ ls -l server_config.txt
-rw-rw-r-- 1 einfochips einfochips 69 Jul 13 17:37 server_config.txt
einfochips@AHMLPT2603:~/Music/My_TASK 2$
```

- Discuss the output (e.g., `-rw-r--r--`).
- 

### 2. Changing Permissions and Ownership

Change permissions to read/write for the owner and read-only for others:

```
chmod 644 server_config.txt
```



```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ chmod 644 server_config.txt
einfochips@AHMLPT2603:~/Music/My_TASK 2$ ls -l server_config.txt
-rw-r--r-- 1 einfochips einfochips 69 Jul 13 17:37 server_config.txt
```

## Understanding File Permissions

File permissions are represented as a set of three groups, each containing three bits:

1. **User (owner)**
2. **Group**
3. **Others**

Each group of three bits represents:

- **Read (r):** Permission to read the file (4)
- **Write (w):** Permission to write to the file (2)
- **Execute (x):** Permission to execute the file (1)

For example, a permission set of `rwxr-xr--` means:

- **User:** read, write, and execute ( $rw x = 4 + 2 + 1 = 7$ )

- **Group:** read and execute (r-x = 4+1 = 5)
- **Others:** read only (r-- = 4+0+0 = 4)

## Using **chmod** with Numeric (Octal) Mode

In numeric mode, permissions are represented by a three-digit octal number, where each digit ranges from 0 to 7:

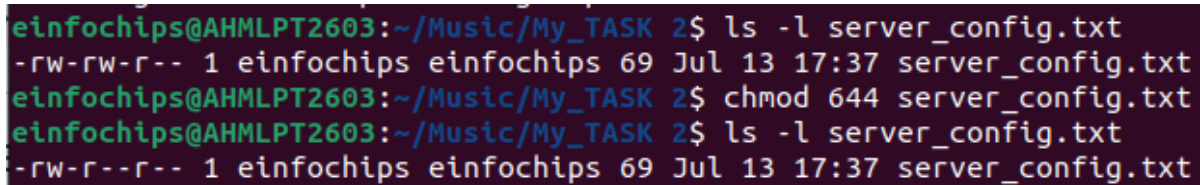
- **7:** read, write, and execute (rwx)
- **6:** read and write (rw-)
- **5:** read and execute (r-x)
- **4:** read only (r--)
- **3:** write and execute (wx)
- **2:** write only (w-)
- **1:** execute only (x)
- **0:** no permissions (---)

Examples:

- **chmod 755 file** sets permissions to **rw-r-xr-x**.
- **chmod 644 file** sets permissions to **rw-r--r--**.

Verify the change:

```
ls -l server_config.txt
```



```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ ls -l server_config.txt
-rw-rw-r-- 1 einfochips einfochips 69 Jul 13 17:37 server_config.txt
einfochips@AHMLPT2603:~/Music/My_TASK 2$ chmod 644 server_config.txt
einfochips@AHMLPT2603:~/Music/My_TASK 2$ ls -l server_config.txt
-rw-r--r-- 1 einfochips einfochips 69 Jul 13 17:37 server_config.txt
```

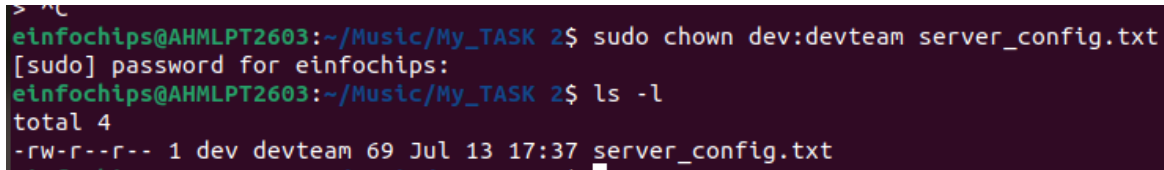
Change the owner to **developer** and the group to **devteam**:

The **chown** command in Linux is used to change the ownership of files and directories. Ownership includes both the user (owner) and the group associated with the file or directory.

```
sudo chown developer:devteam server_config.txt
```

Verify the change:

```
ls -l server_config.txt
```



```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ sudo chown dev:devteam server_config.txt
[sudo] password for einfochips:
einfochips@AHMLPT2603:~/Music/My_TASK 2$ ls -l
total 4
-rw-r--r-- 1 dev devteam 69 Jul 13 17:37 server_config.txt
```

## Part 4: Controlling Services and Daemons (20 minutes)

**Scenario:** Manage the web server service to ensure it is running correctly and starts on boot.

### 1. Managing Services with systemctl

Start the Apache service:

```
sudo systemctl start apache2
```

Stop the Apache service:

```
sudo systemctl stop apache2
```

Enable the Apache service to start on boot:

```
sudo systemctl enable apache2
```

Disable the Apache service:

```
sudo systemctl disable apache2
```

Check the status of the Apache service:

```
sudo systemctl status apache2
```

### 2. Understanding Daemons

- Discuss the role of the `sshd` daemon in providing SSH access to the server.

## Part 5: Process Handling (20 minutes)

**Scenario:** Monitor and manage processes to ensure the server is performing optimally.

### 1. Viewing Processes

List all running processes:

`ps aux`

```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ ps aux
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root            1  1.0  0.0 168804 13636 ?        Ss   09:39   0:33 /sbin/init splash
root            2  0.0  0.0      0     0 ?        S    09:39   0:00 [kthreadd]
root            3  0.0  0.0      0     0 ?        I<   09:39   0:00 [rcu_gp]
root            4  0.0  0.0      0     0 ?        I<   09:39   0:00 [rcu_par_gp]
root            5  0.0  0.0      0     0 ?        I<   09:39   0:00 [slub_flushwq]
root            6  0.0  0.0      0     0 ?        I<   09:39   0:00 [netns]
```

Use `top` to view processes in real-time:

`top`

```
top - 10:32:33 up 53 min, 1 user, load average: 1.55, 1.18, 1.08
Tasks: 448 total, 1 running, 446 sleeping, 0 stopped, 1 zombie
%Cpu(s): 3.9 us, 1.6 sy, 0.0 ni, 94.3 id, 0.1 wa, 0.0 hi, 0.0 st, 0.0 st
MiB Mem : 15744.6 total, 6924.8 free, 3869.3 used, 4950.5 buff/cache
MiB Swap: 15259.0 total, 15259.0 free, 0.0 used, 11004.3 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 2338 root        20   0 4935288 149768 62464 S   10.9   0.9   5:59.13 dockerd
 5598 einfoch+   20   0 5111164 296812 135952 S    8.3   1.8   5:10.72 gnome-shell
11207 einfoch+   20   0 565748 62776 43468 S    3.3   0.4   0:47.81 gnome-terminal-
1237  mysql     20   0 2373624 396808 35200 S    1.3   2.5   0:34.19 mysqld
 8521 einfoch+   20   0 11.6g 477444 226944 S    1.0   3.0   9:41.07 firefox
 9287 einfoch+   20   0 2922968 417800 130528 S    1.0   2.6   4:48.83 Isolated Web Co
55005 einfoch+   20   0 13476 4352 3328 R    1.0   0.0   0:00.11 top
 622  systemd+  20   0 26340 14080 9216 S    0.7   0.1   0:15.22 systemd-resolve
1075 root        20   0 2243564 50724 32256 S    0.7   0.3   0:22.32 containerd
   1 root        20   0 168804 13636 8004 S    0.3   0.1   0:33.11 systemd
  16 root        20   0      0      0      0 I    0.3   0.0   0:04.90 rcu_preempt
 112 root        0 -20      0      0      0 I    0.3   0.0   0:04.24 kworker/u17:0-i91
 334 root       19  -1 56676 26160 24880 S    0.3   0.2   0:11.23 systemd-journal
 621 systemd+  20   0 14836 6784 6016 S    0.3   0.0   0:06.20 systemd-oomd
 962 root        20   0 276236 11136 10240 S    0.3   0.1   0:07.10 thermald
1045 root        20   0 1124724 145504 59648 S    0.3   0.9   0:14.39 wdaemon
1084 root        20   0 285432 21584 19200 S    0.3   0.1   0:00.85 anydesk
1310 root        20   0 3712796 35876 17152 S    0.3   0.2   0:18.12 TaniumCX
1417 root        20   0 772520 24704 21632 S    0.3   0.2   0:04.14 TaniumCX
1750 root        20   0 772952 22912 20224 S    0.3   0.1   0:04.09 TaniumCX
3290 mdatp       20   0 862252 158948 61312 S    0.3   1.0   0:06.92 wdaemon
4231 einfoch+   20   0 9873444 635108 33896 S    0.3   3.9   1:18.41 java
6085 einfoch+   20   0 719560 88144 48196 S    0.3   0.5   0:07.11 snap-store
6420 einfoch+   20   0 3204452 71732 51316 S    0.3   0.4   0:02.82 gjs
53878 root        20   0      0      0      0 I    0.3   0.0   0:00.19 kworker/u16:1-i91
   2 root        20   0      0      0      0 S    0.0   0.0   0:00.01 kthreadd
   3 root        0 -20      0      0      0 I    0.0   0.0   0:00.00 rcu_gp
   4 root        0 -20      0      0      0 I    0.0   0.0   0:00.00 rcu_par_gp
   5 root        0 -20      0      0      0 I    0.0   0.0   0:00.00 slub_flushwq
   6 root        0 -20      0      0      0 I    0.0   0.0   0:00.00 netns
   8 root        0 -20      0      0      0 I    0.0   0.0   0:00.00 kworker/0:0H-even
```

### 2. Managing Processes

Identify a process to kill using `ps` or `top`, then kill it:

`kill <PID>`



## Zombie Process

```
ps aux | grep "Z"
```

Change the priority of a process (e.g., running `sleep` with a lower priority):

In Linux, the priority of a process determines its importance and how much CPU time it receives relative to other processes. Process priority is managed by the kernel scheduler and is represented by a numerical value called the "priority value" or "nice value".

### Nice Value (`nice`):

- The nice value ranges from -20 to +19.
- Lower nice values indicate higher priority (more favorable scheduling).
- Higher nice values indicate lower priority (less favorable scheduling).
- A process with a lower nice value gets more CPU time compared to processes with higher nice values.

### Static Priority (`priority`):

- The static priority ranges from 0 to 139 (in some older systems, it was 0 to 99).
- The static priority is calculated from the nice value and adjusted based on the scheduling policy and the current load on the system.
- Lower static priority values correspond to higher priorities.

```
nice -n 10 sleep 100 &
```

○

Change the priority of the process using `renice`:

```
renice +10 <PID>
```

```
einfochips@AHMLPT2603:~/Music/My_TASK 2$ nice -n 10 sleep 100 &  
[1] 68983
```

## Filesystem Hierarchy Standard (FHS):

- Linux follows the FHS, which defines the purpose and organization of directories in the filesystem.
- Key directories include:
  - **/bin**: Essential binaries (commands) for system boot and repair.
  - **/sbin**: System binaries (commands) for system administration (superuser mode).
  - **/usr**: User-related data including binaries, libraries, documentation, and more.
  - **/etc**: Configuration files for system-wide and application-specific settings.
  - **/home**: User home directories, where users store personal files and configurations.
  - **/opt**: Optional application software packages.

## System Binaries (**/bin**, **/sbin**, **/usr/bin**, **/usr/sbin**):

- These directories contain executable binaries similar to what you might find in "Program Files" in Windows.
- Most installed applications' binaries are located in **/usr/bin** or **/usr/sbin**.

# Mounting

In Ubuntu Linux, "mounting" refers to the process of making a filesystem available for access and use by associating it with a directory (referred to as the "mount point") in the host filesystem's directory tree. This allows the files and directories stored on the mounted filesystem to be accessed and manipulated as if they were part of the host system's own filesystem.

## Understanding Mounting

When you mount a filesystem:

- **Mount Point:** A directory on the host system where the contents of the mounted filesystem will be accessible.
- **Filesystem:** The partition or device containing the files and directories you want to access.

**find:** Search for files in a directory hierarchy.

**grep:** Print lines that match patterns.

grep "word" | filename

**locate:** Find files by name.

Locate filename

**which:** Locate a command.

Which apache2

Which nginx

**sort:** Sort lines of text files.

**uniq:** Report or omit repeated lines.

**wc:** Print newline, word, and byte counts for each file.

**Wc -w filename (count words)**

## System Monitoring and Performance

- **uptime:** Tell how long the system has been running.
- **free:** Display amount of free and used memory in the system.
- **vmstat:** Report virtual memory statistics.
- **iostat:** Report CPU and input/output statistics.
- **sar:** Collect, report, or save system activity information.

## Disk Usage and Storage

- **df**: Display disk space usage.
- **du**: Estimate file space usage.

## System Information

- **uname**: Print system information.
- **dmesg**: Print or control the kernel ring buffer.
- **lsb\_release**: Print distribution-specific information.
- **hostname**: Show or set the system's hostname.

## Creating and Deploying a Static Website with Apache2

### Preparation (5 minutes)

- Ensure you have access to a Linux environment (e.g., virtual machines, EC2 instances, or local installations) with sudo privileges.

### Activity Breakdown

#### Part 1: Installing Apache2 (5 minutes)

##### 1. Update Package Lists

Open the terminal and run:

```
sudo apt update
```

- 

## 2. Install Apache2

Install Apache2 by running:

```
sudo apt install apache2
```

- 

## 3. Start and Enable Apache2

Start the Apache2 service:

```
sudo systemctl start apache2
```

- 

Enable Apache2 to start on boot:

```
sudo systemctl enable apache2
```

- 

## 4. Verify Installation

- Open a web browser and navigate to [http://your\\_server\\_ip](http://your_server_ip). You should see the Apache2 default page.

## Part 2: Creating the Website (10 minutes)

### 1. Navigate to the Web Directory

Change to the web root directory:

```
cd /var/www/html
```

- 

### 2. Create a New Directory for the Website

Create a directory named `mystaticwebsite`:

```
sudo mkdir mystaticwebsite
```

- 

Change ownership of the directory:

```
sudo chown -R $USER:$USER /var/www/html/mystaticwebsite
```

### 3. Create HTML File

Create and edit the `index.html` file:

```
nano /var/www/html/mystaticwebsite/index.html
```

○

Add the following content:

```
<!DOCTYPE html>

<html>

<head>

  <title>My Static Website</title>

  <link rel="stylesheet" type="text/css" href="styles.css">

</head>

<body>

  <h1>Welcome to My Static Website</h1>

  <p>This is a simple static website using Apache2.</p>

  <script src="script.js"></script>

</body>

</html>
```

○

- Save and exit (Ctrl+O, Enter, Ctrl+X).

### 4. Create CSS File

Create and edit the `styles.css` file:

```
nano /var/www/html/mystaticwebsite/styles.css
```

○

Add the following content:

```
body {  
  
    font-family: Arial, sans-serif;  
  
    background-color: #f0f0f0;  
  
    text-align: center;  
  
    margin: 0;  
  
    padding: 20px;  
  
}
```

```
h1 {  
  
    color: #333;  
  
}
```

○

- Save and exit (Ctrl+O, Enter, Ctrl+X).

## 5. Create JavaScript File

Create and edit the `script.js` file:

```
nano /var/www/html/mystaticwebsite/script.js
```

○

Add the following content:

```
document.addEventListener('DOMContentLoaded', function() {  
  
    console.log('Hello, World!');  
  
});
```

○



- Save and exit (Ctrl+O, Enter, Ctrl+X).

## 6. Add an Image

Download or copy an image file (e.g., `logo.png`) to the website directory:

```
cp /path/to/your/logo.png /var/www/html/mystaticwebsite/logo.png
```

- 

Update `index.html` to include the image:

```
<body>
```

```
    <h1>Welcome to My Static Website</h1>
```

```
    
```

```
    <p>This is a simple static website using Apache2.</p>
```

```
    <script src="script.js"></script>
```

```
</body>
```

- 

## Part 3: Configuring Apache2 to Serve the Website (10 minutes)

### 1. Create a Virtual Host File

Create and edit the virtual host configuration file:

```
sudo nano /etc/apache2/sites-available/mystaticwebsite.conf
```

- 

Add the following content:

```
<VirtualHost *:80>
```

```
    ServerAdmin webmaster@localhost
```

```
    DocumentRoot /var/www/html/mystaticwebsite
```

```
    ErrorLog ${APACHE_LOG_DIR}/error.log
```

```
CustomLog ${APACHE_LOG_DIR}/access.log combined  
  
</VirtualHost>
```

- Save and exit (Ctrl+O, Enter, Ctrl+X)

## 2. Enable the New Virtual Host

Enable the virtual host configuration:

```
sudo a2ensite mystaticwebsite.conf
```

## 3. Disable the Default Site

Disable the default site configuration:

```
sudo a2dissite 000-default.conf
```

○

## 4. Reload Apache2

Reload the Apache2 service to apply the changes:

```
sudo systemctl reload apache2
```

○

## 5. Test the Configuration

- Open a web browser and navigate to [http://your\\_server\\_ip](http://your_server_ip). You should see the static website with the HTML, CSS, JS, and image.

By JASH SHAH