Linux Fundamentals: A Comprehensive Guide for Beginners and Enthusiasts (Part – 1)

Introduction

Linux is a powerful, open-source operating system that has become the backbone of modern technology. From powering servers and cloud infrastructure to running embedded systems and supercomputers, Linux is everywhere. Its flexibility, security, and cost-effectiveness make it a top choice for organizations worldwide. For IT professionals, developers, and system administrators, mastering Linux fundamentals is not just a skill, it’s a necessity.

Why Linux Matters in Cybersecurity and Technical Support

In the realm of cybersecurity, Linux is indispensable. Many cybersecurity tools and frameworks, such as Kali Linux, Metasploit, and Wireshark, are built on Linux. Security professionals rely on Linux for penetration testing, vulnerability assessment, and forensic analysis. Its robust permission system, logging capabilities, and open-source nature make it ideal for securing systems and identifying threats.

For technical support roles, Linux expertise is equally critical. Many enterprise systems, web servers, and networking devices run on Linux. Troubleshooting issues, managing servers, and ensuring system uptime often require a solid understanding of Linux commands and processes. Whether you’re diagnosing network problems, configuring firewalls, or automating tasks, Linux skills are a must-have.

What This Guide Covers

This guide is designed to help you build a strong foundation in Linux fundamentals. We’ll explore:

1. Installing Kali Linux: Set up Kali on a virtual machine to create your own home lab setup.
2. The Linux Environment: Discover how to use the terminal and explore Kali’s file system structure.
3. Navigating the File System: Learn the essential commands to move around and inspect directories with ease.

By the end of this guide, you’ll have the knowledge and confidence to tackle real-world challenges in cybersecurity and technical support using Linux. Let’s dive in!

1. Exploring Linux Environment:

In this section, we’ll take a closer look at the Linux environment. Whether you’re using a virtual machine, a cloud instance, or a physical machine, understanding the Linux environment is the first step toward mastering it.

1.1 Installing Kali Linux

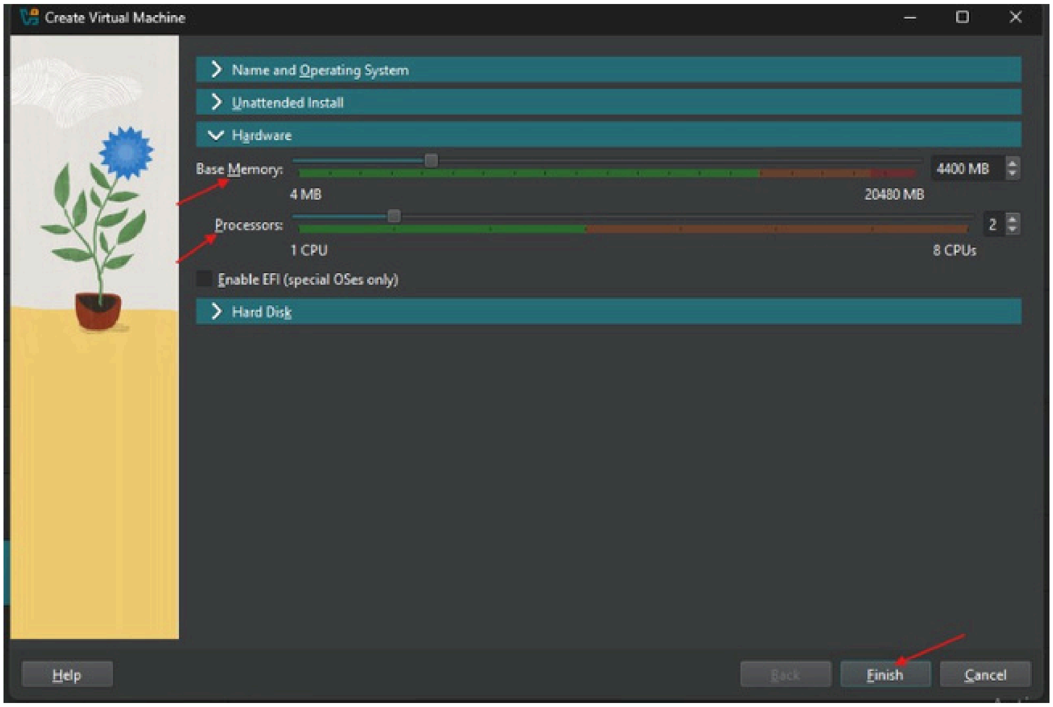
Before we dive into the Linux environment, let’s get Kali Linux up and run on your system! Kali is a specialized distribution for cybersecurity and setting it up is straightforward using a virtual machine (VM). We’ll use VirtualBox—a free and beginner-friendly virtualization tool. Follow these steps to install Kali Linux on VirtualBox:

1. Download Kali Linux and VirtualBox:

- Visit kali.org and download the latest Kali Linux ISO (e.g., "Kali Linux VirtualBox 64-bit").
- Download and install VirtualBox from virtualbox.org (available for Windows, macOS, and Linux).

2. Create a New Virtual Machine:

- Open VirtualBox, click "New," and name your VM (e.g., "Kali Linux").
- Set the type to "Linux" and version to "Debian (64-bit)."
- Allocate at least 2 GB of RAM (4 GB recommended) and create a virtual hard disk (20 GB recommended).

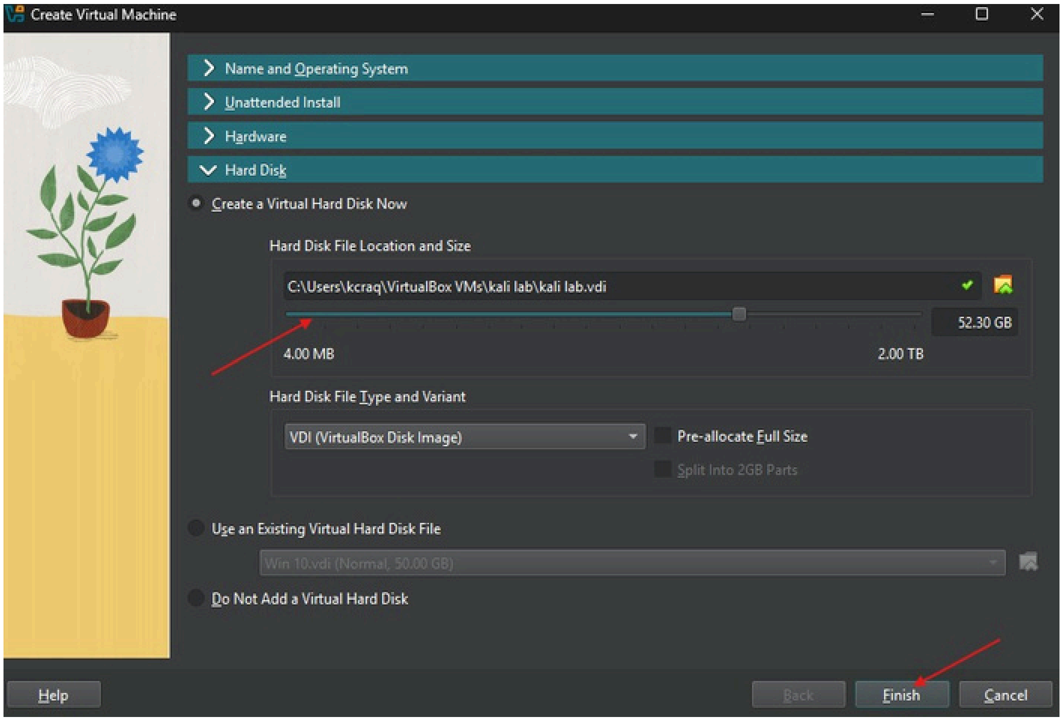


Base Memory: 4400 MB

Processors: 2

□ Enable EFI (special OSes only)

Creating a new virtual machine in VirtualBox for Kali Linux.



Create Virtual Machine

> Name and Operating System

> Unattended Install

> Hardware

∨ Hard Disk

● Create a Virtual Hard Disk Now

Hard Disk File Location and Size

C:\Users\kcraq\VirtualBox VMs\kali lab\kali lab.vdi

52.30 GB

4.00 MB — 2.00 TB

Hard Disk File Type and Variant

VDI (VirtualBox Disk Image)

□ Pre-allocate Full Size

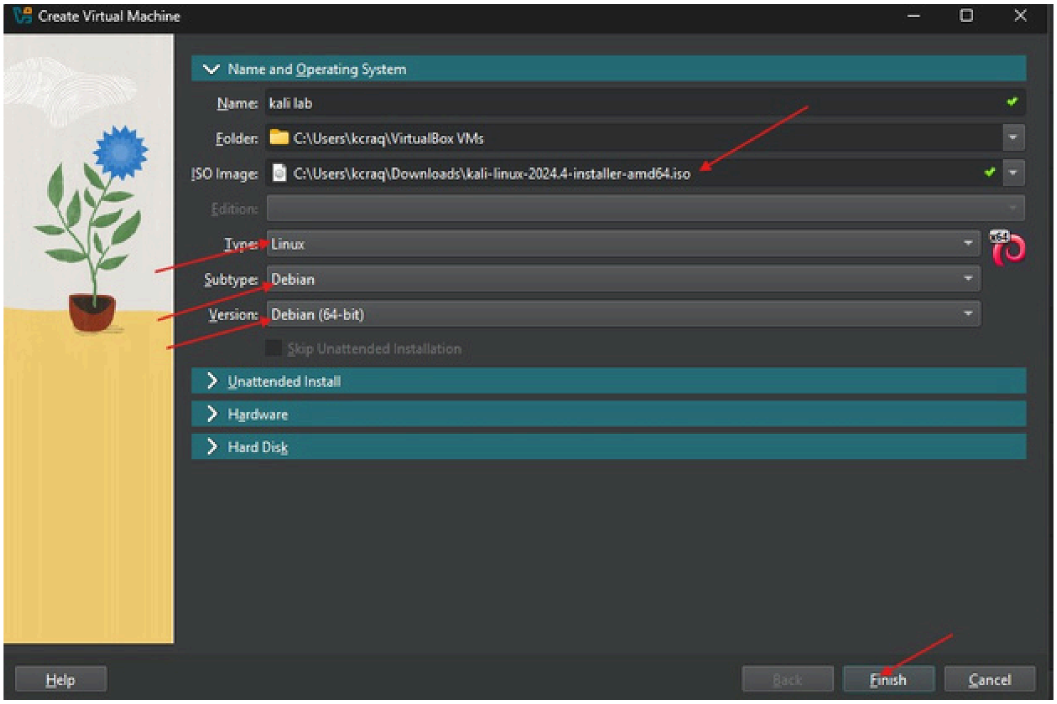
□ Split Into 2GB Parts

○ Use an Existing Virtual Hard Disk File

Win 10.vdi (Normal, 50.00 GB)

○ Do Not Add a Virtual Hard Disk

3. Attach the Kali ISO:
   - In the VM settings, go to "Storage," select the empty disk under "Controller: IDE," and choose the Kali ISO you downloaded.



Create Virtual Machine

∨ Name and Operating System

Name: kali lab

Folder: C:\Users\kcraq\VirtualBox VMs

ISO Image: C:\Users\kcraq\Downloads\kali-linux-2024.4-installer-amd64.iso

Edition:

Type: Linux

Subtype: Debian

Version: Debian (64-bit)

□ Skip Unattended Installation

> Unattended Install

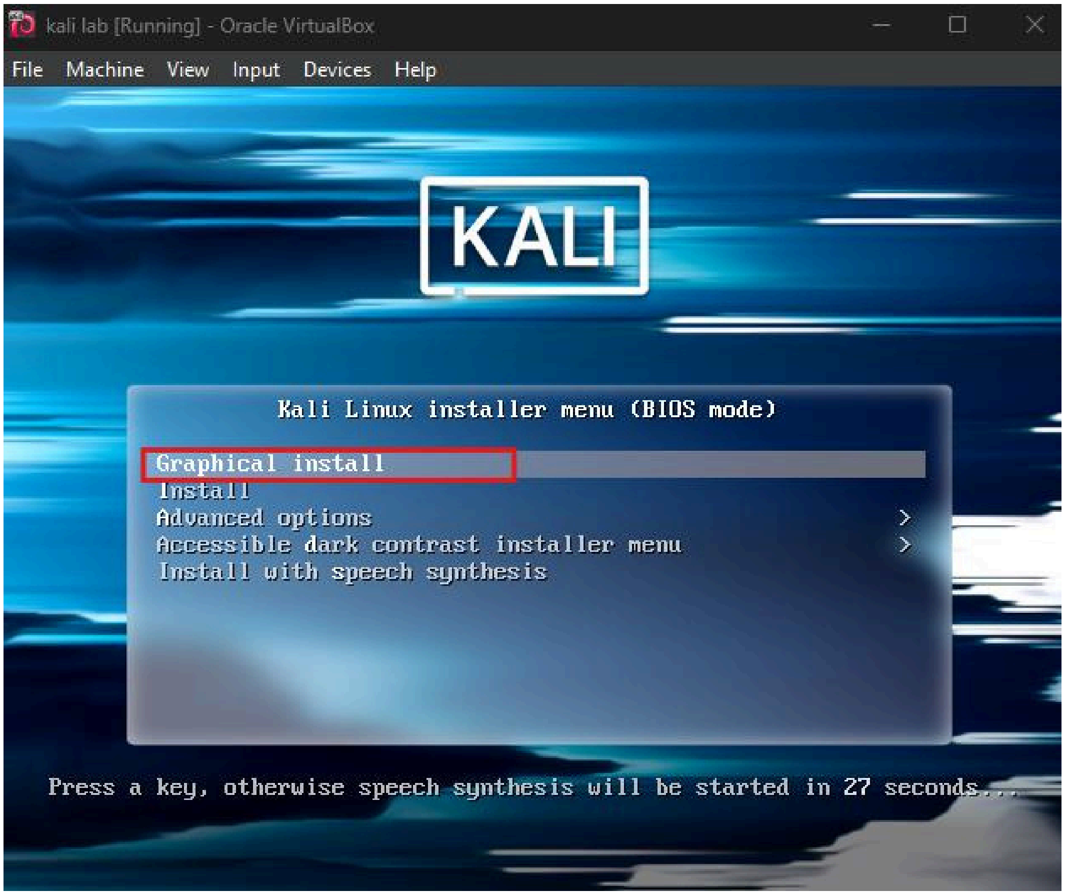
> Hardware

> Hard Disk

Attaching the Kali Linux ISO in VirtualBox settings.

4. Start the Installation:

- Start the VM, and you'll see the Kali boot menu. Select "Graphical Install" to begin.



kali lab [Running] - Oracle VirtualBox
File Machine View Input Devices Help
KALI

Kali Linux installer menu (BIOS mode)

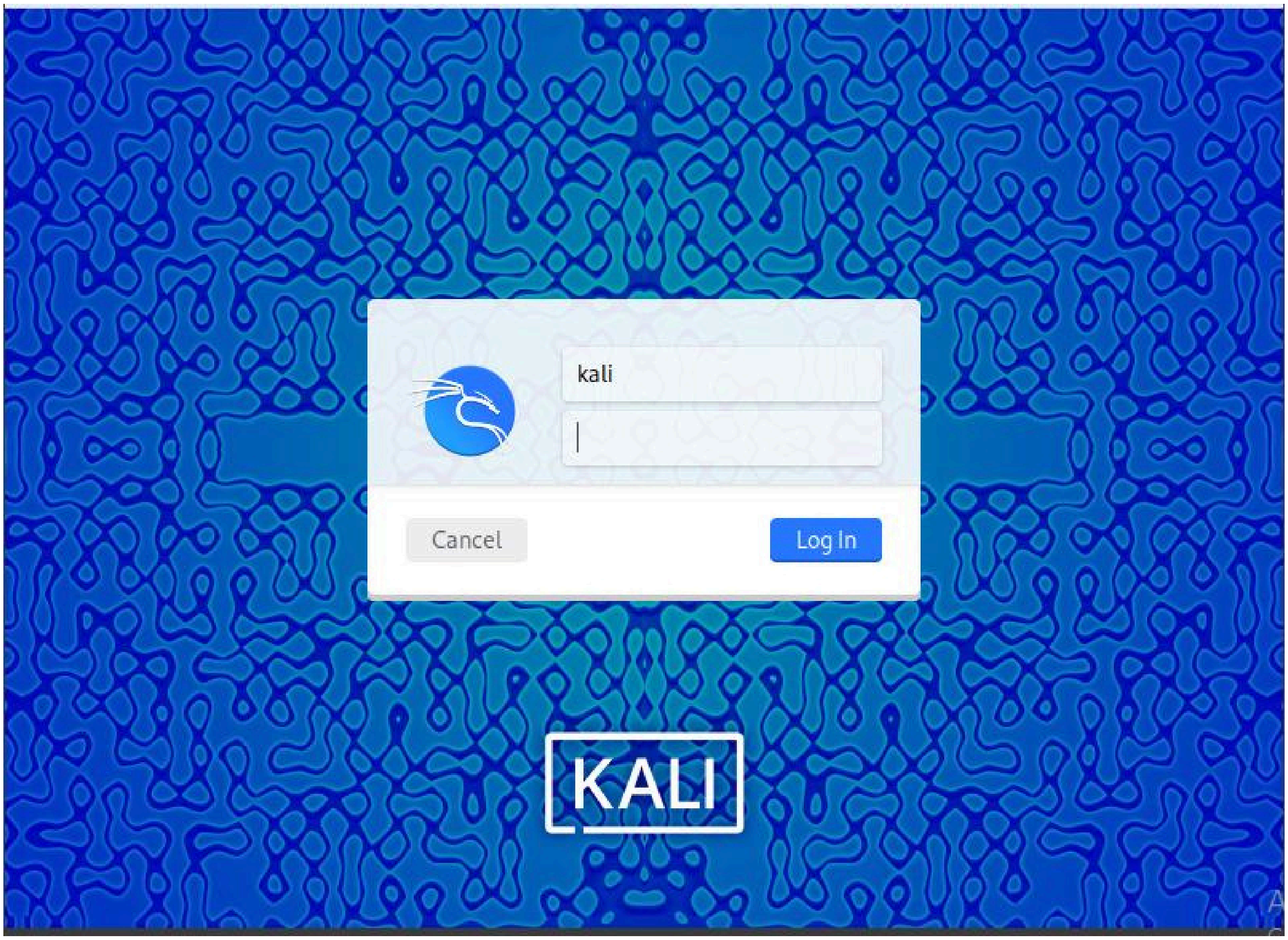
Graphical install
Install
Advanced options >
Accessible dark contrast installer menu >
Install with speech synthesis

Press a key, otherwise speech synthesis will be started in 27 seconds...

Selecting "Graphical Install" from the Kali Linux boot menu.

5. Follow the Installation Prompts:
- Choose your language, timezone, and hostname (e.g., "kali").
- Set up a user (e.g., username "kali" with a password).
- Select "Guided - use entire disk" for partitioning, then install the GRUB bootloader and finish the setup.

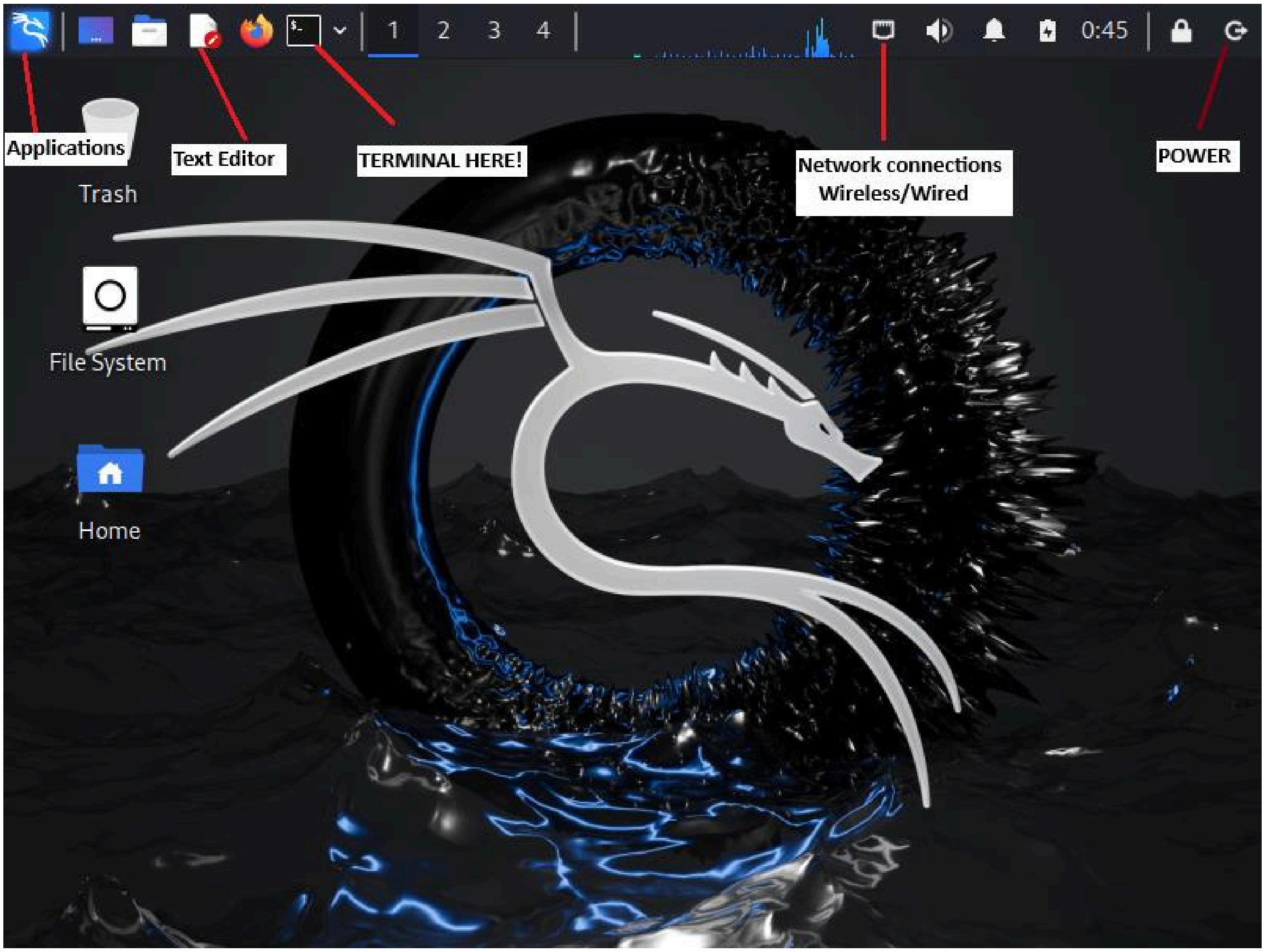
6. Boot Up and Log In:
- After installation, reboot the VM, log in with your user credentials, and you're ready to explore Kali!



*The Kali Linux login screen after installation.**

This setup gives you a safe, isolated environment to experiment with Kali Linux. Now that you're set up, let's explore the environment! Accessing the Terminal The terminal is the primary way to interact with Linux. It allows you to execute commands, manage files, and configure the system. most Linux servers are managed via the terminal, making it a critical skill.

Let's open the terminal here:



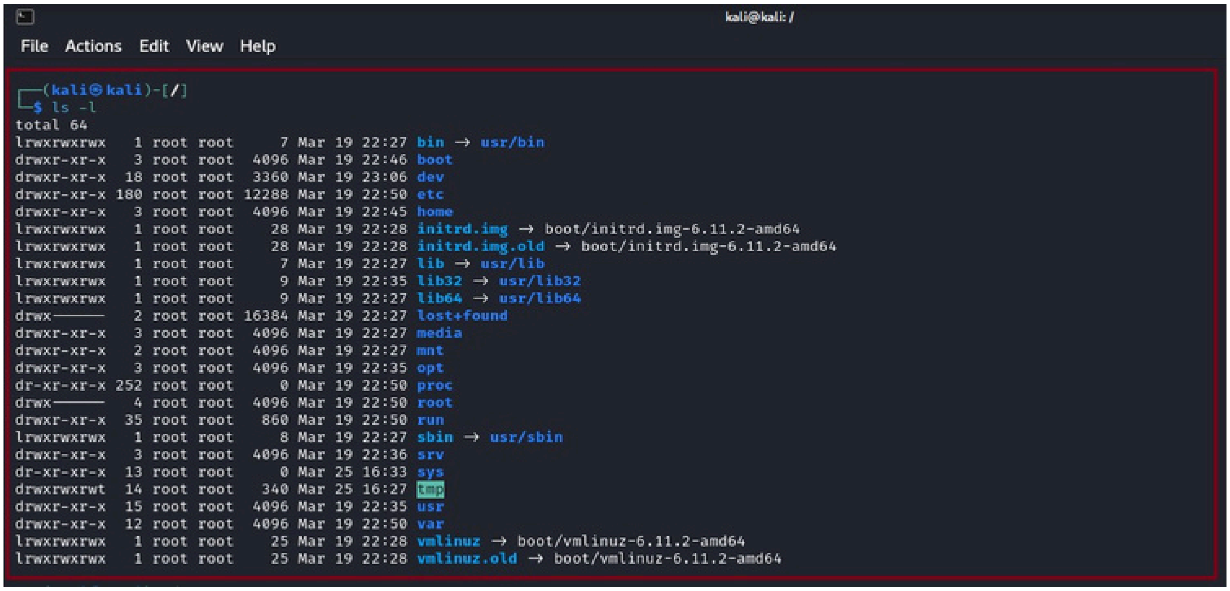
The First Screen in Kali linux

2. File System Layout

One of the first things you'll notice when working with Linux is its hierarchical file system. Unlike other operating systems, Linux organizes everything under a single root directory, represented by /. This structure is logical, consistent, and easy to navigate once you understand its key components.

Let's explore the root directory and its most important subdirectories.

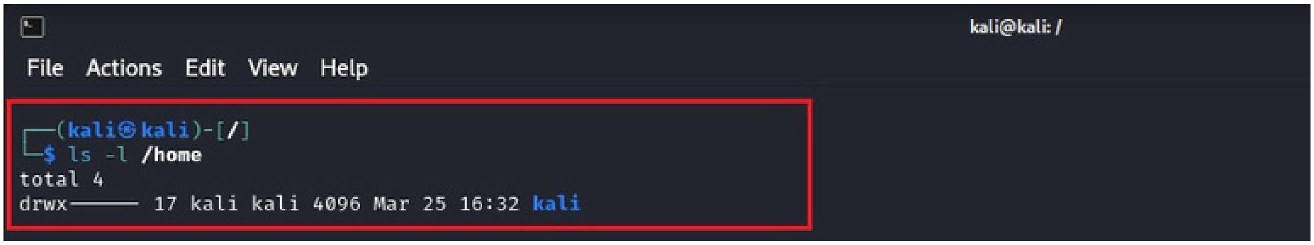
/ (Root Directory) The root directory is the starting point of the Linux file system. Every file, directory, and device in Linux is organized under this single hierarchy.



```
┌──(kali㉿kali)-[/]
└─$ ls -l
total 64
lrwxrwxrwx   1 root root     7 Mar 19 22:27 bin → usr/bin
drwxr-xr-x   3 root root  4096 Mar 19 22:46 boot
drwxr-xr-x  18 root root  3360 Mar 19 23:06 dev
drwxr-xr-x 180 root root 12288 Mar 19 22:50 etc
drwxr-xr-x   3 root root  4096 Mar 19 22:45 home
lrwxrwxrwx   1 root root    28 Mar 19 22:28 initrd.img → boot/initrd.img-6.11.2-amd64
lrwxrwxrwx   1 root root    28 Mar 19 22:28 initrd.img.old → boot/initrd.img-6.11.2-amd64
lrwxrwxrwx   1 root root     7 Mar 19 22:27 lib → usr/lib
lrwxrwxrwx   1 root root     9 Mar 19 22:35 lib32 → usr/lib32
lrwxrwxrwx   1 root root     9 Mar 19 22:27 lib64 → usr/lib64
drwx──────   2 root root 16384 Mar 19 22:27 lost+found
drwxr-xr-x   3 root root  4096 Mar 19 22:27 media
drwxr-xr-x   2 root root  4096 Mar 19 22:27 mnt
drwxr-xr-x   3 root root  4096 Mar 19 22:35 opt
dr-xr-xr-x 252 root root     0 Mar 19 22:50 proc
drwx──────   4 root root  4096 Mar 19 22:50 root
drwxr-xr-x  35 root root   860 Mar 19 22:50 run
lrwxrwxrwx   1 root root     8 Mar 19 22:27 sbin → usr/sbin
drwxr-xr-x   3 root root  4096 Mar 19 22:36 srv
dr-xr-xr-x  13 root root     0 Mar 25 16:33 sys
drwxrwxrwt  14 root root   340 Mar 25 16:27 tmp
drwxr-xr-x  15 root root  4096 Mar 19 22:35 usr
drwxr-xr-x  12 root root  4096 Mar 19 22:50 var
lrwxrwxrwx   1 root root    25 Mar 19 22:28 vmlinuz → boot/vmlinuz-6.11.2-amd64
lrwxrwxrwx   1 root root    25 Mar 19 22:28 vmlinuz.old → boot/vmlinuz-6.11.2-amd64
```

/home

This directory contains user home directories.
Each user on the system has a subdirectory here (e.g., /home/username), where they can store personal files, documents, and configurations.

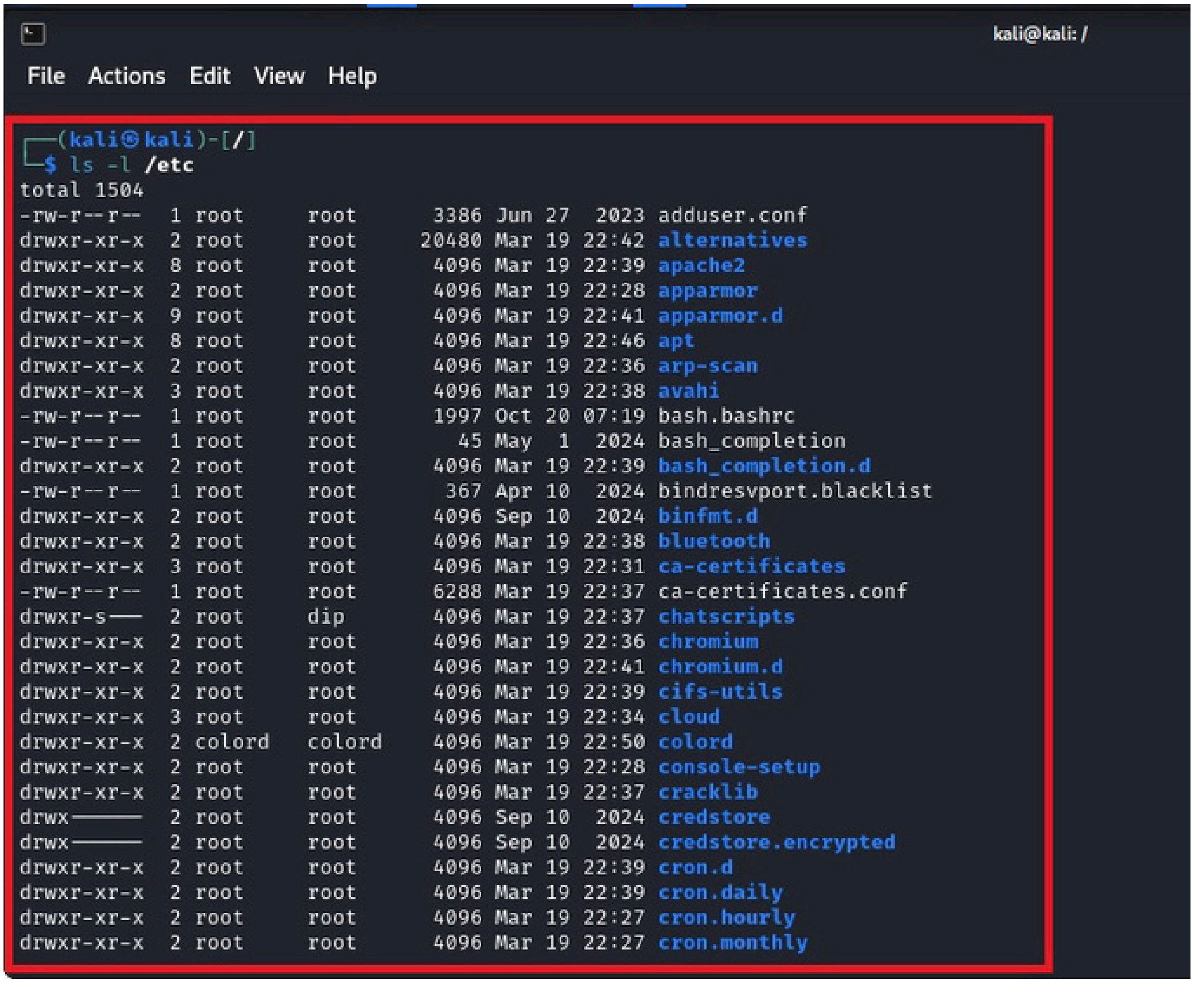


```
┌──(kali㉿kali)-[/]
└─$ ls -l /home
total 4
drwx────── 17 kali kali 4096 Mar 25 16:32 kali
```

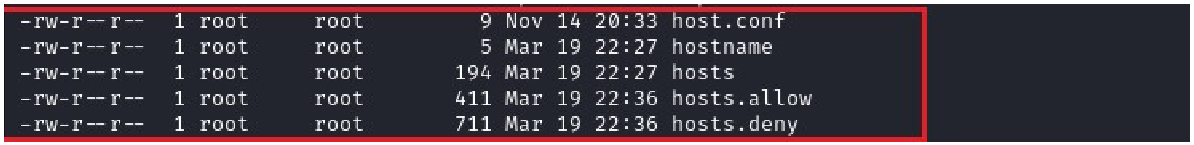
Example: If your username is kali, your home directory would be /home/kali.

/etc

The /etc directory stores system-wide configuration files.
This includes configuration files for installed applications, network settings, user accounts, and more.

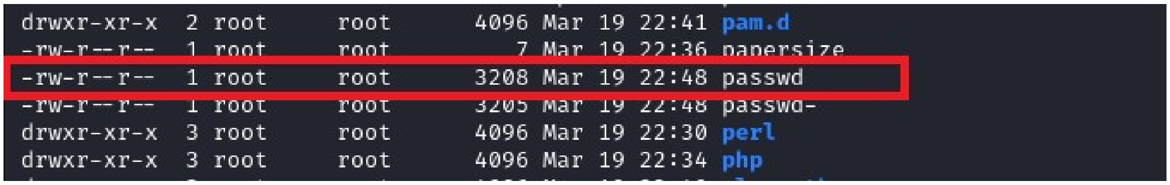


```
┌──(kali㉿kali)-[/]
└─$ ls -l /etc
total 1504
-rw-r--r--  1 root     root      3386 Jun 27  2023 adduser.conf
drwxr-xr-x  2 root     root     20480 Mar 19 22:42 alternatives
drwxr-xr-x  8 root     root      4096 Mar 19 22:39 apache2
drwxr-xr-x  2 root     root      4096 Mar 19 22:28 apparmor
drwxr-xr-x  9 root     root      4096 Mar 19 22:41 apparmor.d
drwxr-xr-x  8 root     root      4096 Mar 19 22:46 apt
drwxr-xr-x  2 root     root      4096 Mar 19 22:36 arp-scan
drwxr-xr-x  3 root     root      4096 Mar 19 22:38 avahi
-rw-r--r--  1 root     root      1997 Oct 20 07:19 bash.bashrc
-rw-r--r--  1 root     root        45 May  1  2024 bash_completion
drwxr-xr-x  2 root     root      4096 Mar 19 22:39 bash_completion.d
-rw-r--r--  1 root     root       367 Apr 10  2024 bindresvport.blacklist
drwxr-xr-x  2 root     root      4096 Sep 10  2024 binfmt.d
drwxr-xr-x  2 root     root      4096 Mar 19 22:38 bluetooth
drwxr-xr-x  3 root     root      4096 Mar 19 22:31 ca-certificates
-rw-r--r--  1 root     root      6288 Mar 19 22:37 ca-certificates.conf
drwxr-s---  2 root     dip       4096 Mar 19 22:37 chatscripts
drwxr-xr-x  2 root     root      4096 Mar 19 22:36 chromium
drwxr-xr-x  2 root     root      4096 Mar 19 22:41 chromium.d
drwxr-xr-x  2 root     root      4096 Mar 19 22:39 cifs-utils
drwxr-xr-x  3 root     root      4096 Mar 19 22:34 cloud
drwxr-xr-x  2 colord   colord    4096 Mar 19 22:50 colord
drwxr-xr-x  2 root     root      4096 Mar 19 22:28 console-setup
drwxr-xr-x  2 root     root      4096 Mar 19 22:37 cracklib
drwx------  2 root     root      4096 Sep 10  2024 credstore
drwx------  2 root     root      4096 Sep 10  2024 credstore.encrypted
drwxr-xr-x  2 root     root      4096 Mar 19 22:39 cron.d
drwxr-xr-x  2 root     root      4096 Mar 19 22:39 cron.daily
drwxr-xr-x  2 root     root      4096 Mar 19 22:27 cron.hourly
drwxr-xr-x  2 root     root      4096 Mar 19 22:27 cron.monthly
```



```
-rw-r--r--  1 root     root         9 Nov 14 20:33 host.conf
-rw-r--r--  1 root     root         5 Mar 19 22:27 hostname
-rw-r--r--  1 root     root       194 Mar 19 22:27 hosts
-rw-r--r--  1 root     root       411 Mar 19 22:36 hosts.allow
-rw-r--r--  1 root     root       711 Mar 19 22:36 hosts.deny
```

Here, we can configure this file i.e hosts.allow and hosts.deny to any hosts to allowed or deny to access specific services (e.g SSH)

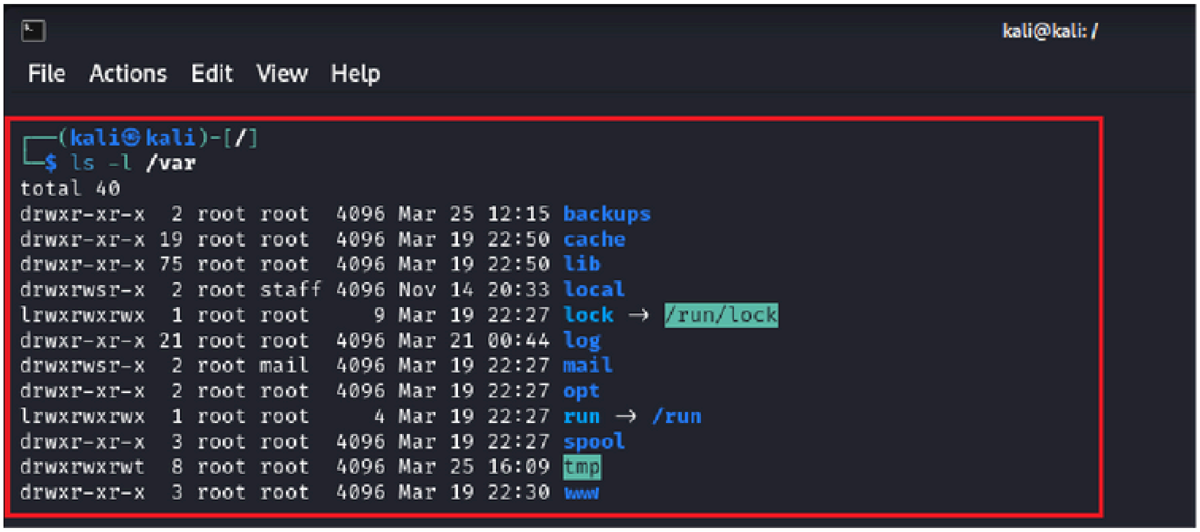


```
drwxr-xr-x  2 root     root      4096 Mar 19 22:41 pam.d
-rw-r--r--  1 root     root         7 Mar 19 22:36 papersize
-rw-r--r--  1 root     root      3208 Mar 19 22:48 passwd
-rw-r--r--  1 root     root      3205 Mar 19 22:48 passwd-
drwxr-xr-x  3 root     root      4096 Mar 19 22:30 perl
drwxr-xr-x  3 root     root      4096 Mar 19 22:34 php
```

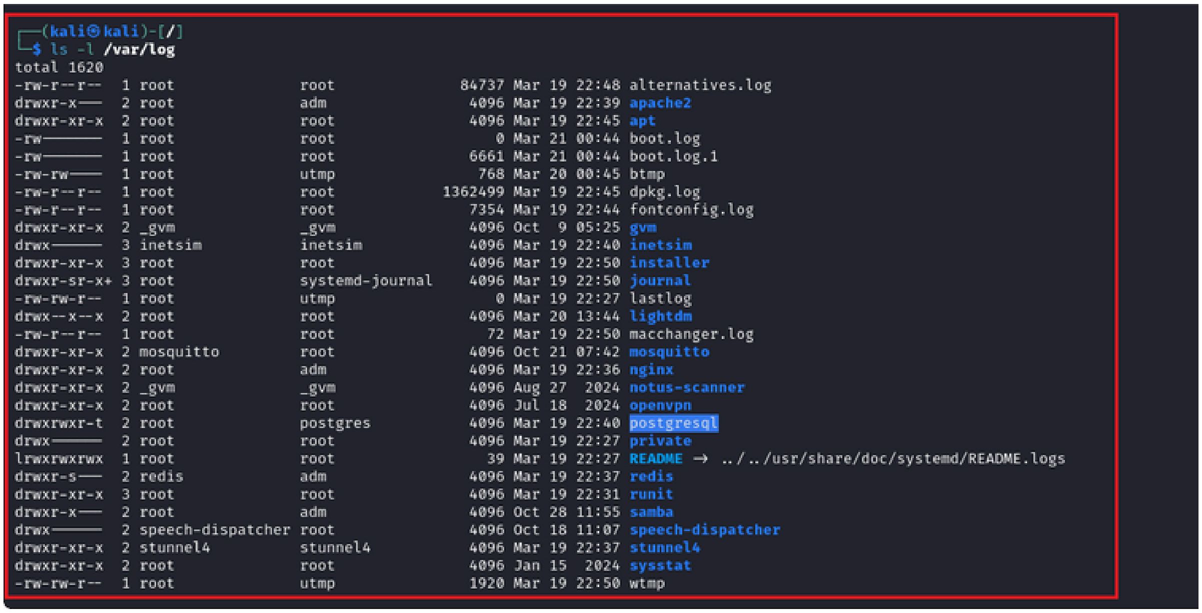
Example: The /etc/passwd file contains user account information, while /etc/network/interfaces stores network configuration.

/var

The /var directory holds variable data that changes frequently during system operation. This includes log files (/var/log), databases, email queues, and temporary files.



```
┌──(kali㉿kali)-[/]
└─$ ls -l /var
total 40
drwxr-xr-x  2 root root  4096 Mar 25 12:15 backups
drwxr-xr-x 19 root root  4096 Mar 19 22:50 cache
drwxr-xr-x 75 root root  4096 Mar 19 22:50 lib
drwxrwsr-x  2 root staff 4096 Nov 14 20:33 local
lrwxrwxrwx  1 root root     9 Mar 19 22:27 lock → /run/lock
drwxr-xr-x 21 root root  4096 Mar 21 00:44 log
drwxrwsr-x  2 root mail  4096 Mar 19 22:27 mail
drwxr-xr-x  2 root root  4096 Mar 19 22:27 opt
lrwxrwxrwx  1 root root     4 Mar 19 22:27 run → /run
drwxr-xr-x  3 root root  4096 Mar 19 22:27 spool
drwxrwxrwt  8 root root  4096 Mar 25 16:09 tmp
drwxr-xr-x  3 root root  4096 Mar 19 22:30 www
```

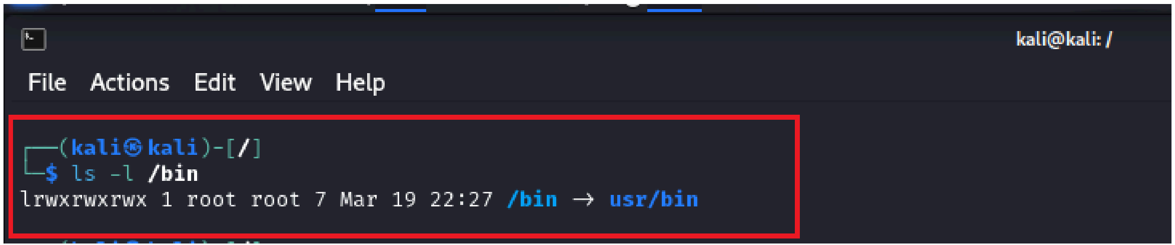


```
┌──(kali㉿kali)-[/]
└─$ ls -l /var/log
total 1620
-rw-r--r--  1 root              root              84737 Mar 19 22:48 alternatives.log
drwxr-x---  2 root              adm                4096 Mar 19 22:39 apache2
drwxr-xr-x  2 root              root               4096 Mar 19 22:45 apt
-rw-------  1 root              root                  0 Mar 21 00:44 boot.log
-rw-------  1 root              root               6661 Mar 21 00:44 boot.log.1
-rw-rw----  1 root              utmp                768 Mar 20 00:45 btmp
-rw-r--r--  1 root              root            1362499 Mar 19 22:45 dpkg.log
-rw-r--r--  1 root              root               7354 Mar 19 22:44 fontconfig.log
drwxr-xr-x  2 _gvm              _gvm               4096 Oct  9 05:25 gvm
drwx------  3 inetsim           inetsim            4096 Mar 19 22:40 inetsim
drwxr-xr-x  3 root              root               4096 Mar 19 22:50 installer
drwxr-sr-x+ 3 root              systemd-journal    4096 Mar 19 22:50 journal
-rw-rw-r--  1 root              utmp                  0 Mar 19 22:27 lastlog
drwx--x--x  2 root              root               4096 Mar 20 13:44 lightdm
-rw-r--r--  1 root              root                 72 Mar 19 22:50 macchanger.log
drwxr-xr-x  2 mosquitto         root               4096 Oct 21 07:42 mosquitto
drwxr-xr-x  2 root              adm                4096 Mar 19 22:36 nginx
drwxr-xr-x  2 _gvm              _gvm               4096 Aug 27  2024 notus-scanner
drwxr-xr-x  2 root              root               4096 Jul 18  2024 openvpn
drwxrwxr-t  2 root              postgres           4096 Mar 19 22:40 postgresql
drwx------  2 root              root               4096 Mar 19 22:27 private
lrwxrwxrwx  1 root              root                 39 Mar 19 22:27 README → ../../usr/share/doc/systemd/README.logs
drwxr-s---  2 redis             adm                4096 Mar 19 22:37 redis
drwxr-xr-x  3 root              root               4096 Mar 19 22:31 runit
drwxr-x---  2 root              adm                4096 Oct 28 11:55 samba
drwx------  2 speech-dispatcher root               4096 Oct 18 11:07 speech-dispatcher
drwxr-xr-x  2 stunnel4          stunnel4           4096 Mar 19 22:37 stunnel4
drwxr-xr-x  2 root              root               4096 Jan 15  2024 sysstat
-rw-rw-r--  1 root              utmp               1920 Mar 19 22:50 wtmp
```

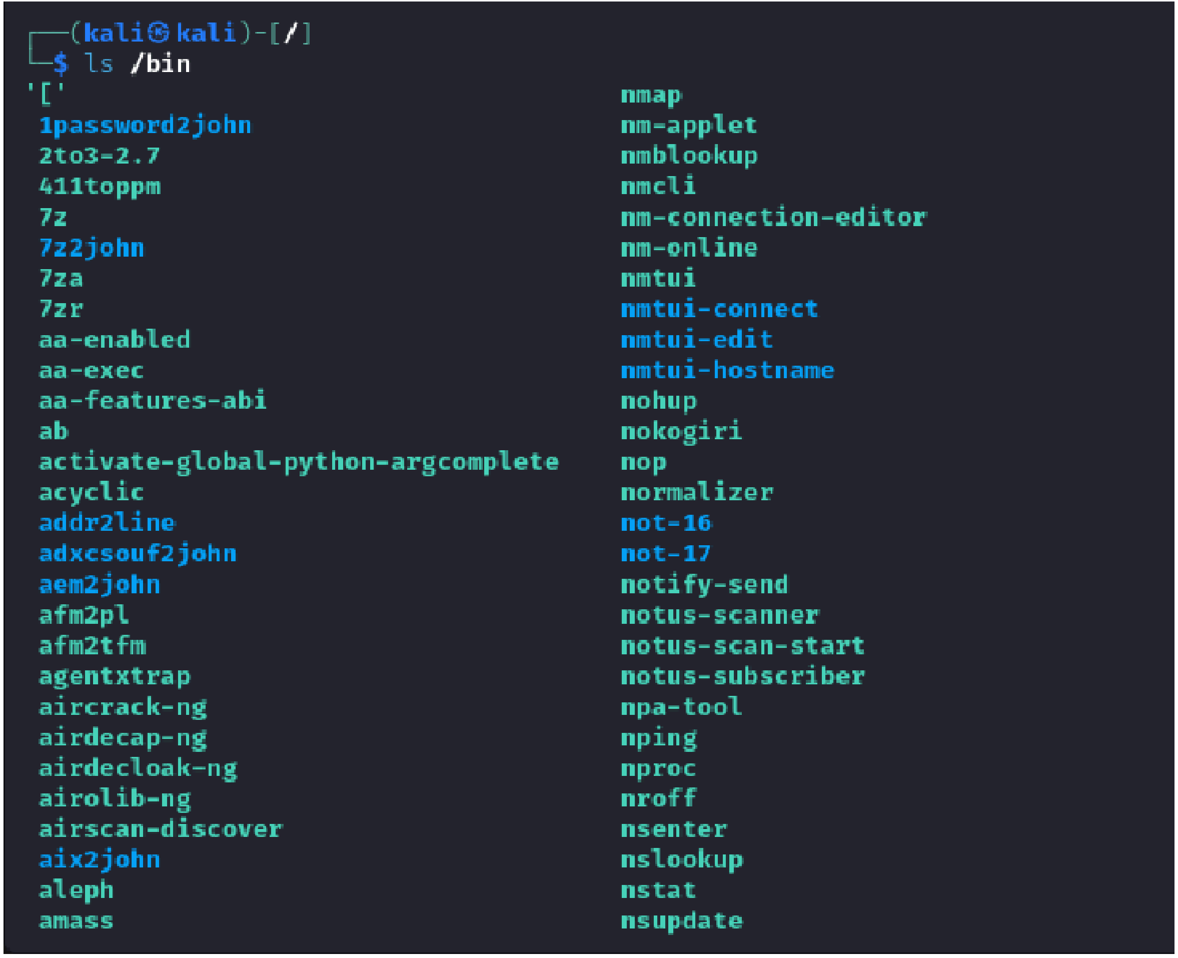
Example: System logs like syslog or auth.log are stored in /var/log and are crucial for troubleshooting.

/bin

The /bin directory contains essential command binaries (executable files) that are required for basic system functionality.
These commands are available to all users and are necessary for system recovery and maintenance.



```
┌──(kali㉿kali)-[/]
└─$ ls -l /bin
lrwxrwxrwx 1 root root 7 Mar 19 22:27 /bin → usr/bin
```



```
┌──(kali㉿kali)-[/]
└─$ ls /bin
'['                                       nmap
 1password2john                           nm-applet
 2to3-2.7                                 nmblookup
 411toppm                                 nmcli
 7z                                       nm-connection-editor
 7z2john                                  nm-online
 7za                                      nmtui
 7zr                                      nmtui-connect
 aa-enabled                               nmtui-edit
 aa-exec                                  nmtui-hostname
 aa-features-abi                          nohup
 ab                                       nokogiri
 activate-global-python-argcomplete       nop
 acyclic                                  normalizer
 addr2line                                not-16
 adxcsouf2john                            not-17
 aem2john                                 notify-send
 afm2pl                                   notus-scanner
 afm2tfm                                  notus-scan-start
 agentxtrap                               notus-subscriber
 aircrack-ng                              npa-tool
 airdecap-ng                              nping
 airdecloak-ng                            nproc
 airolib-ng                               nroff
 airscan-discover                         nsenter
 aix2john                                 nslookup
 aleph                                    nstat
 amass                                    nsupdate
```

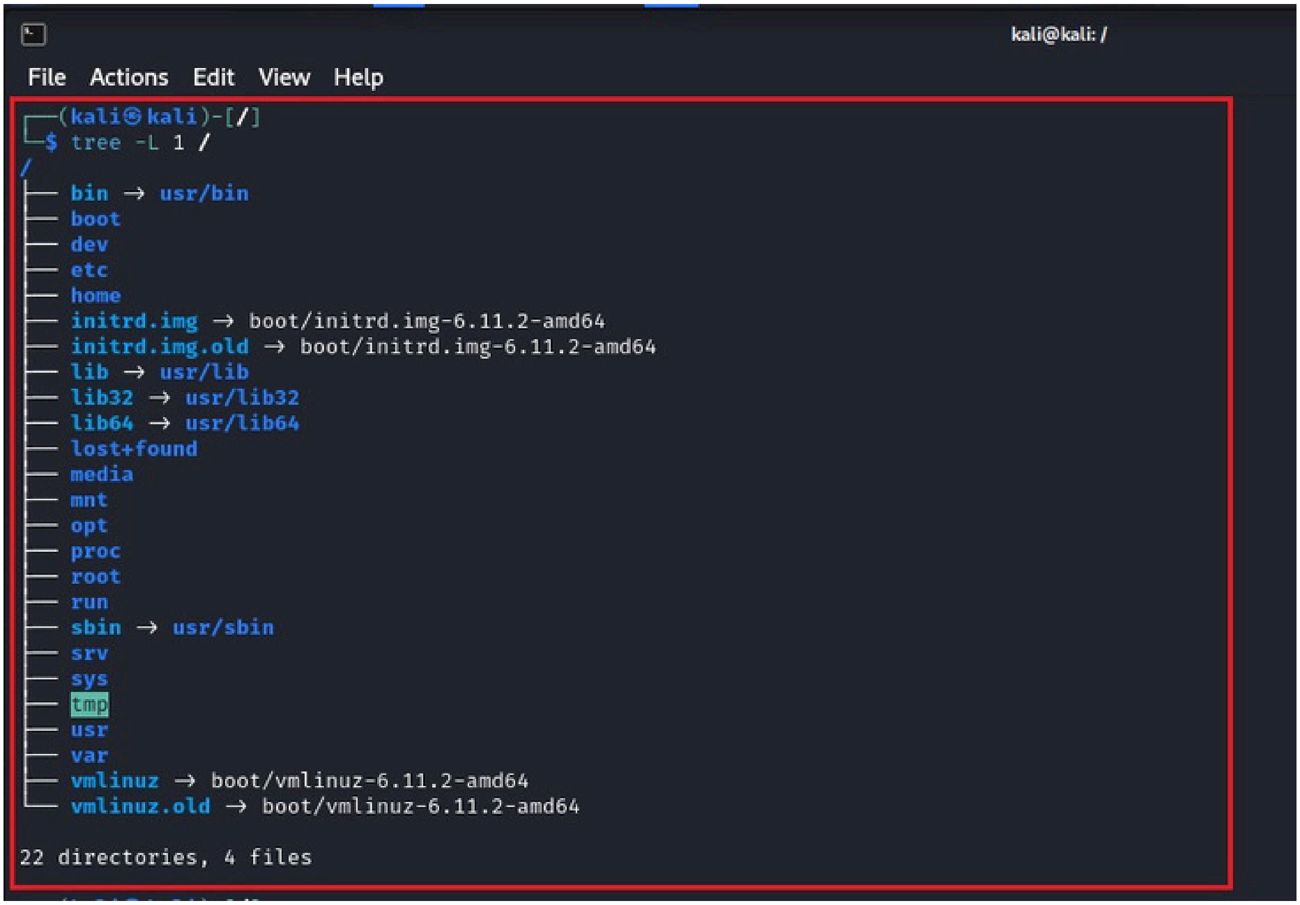
Example: Commands like ls, cp, mv, and rm are located here.

Why This Matters

- Understanding the Linux file system layout is crucial for system administration, troubleshooting, and security.
- For example, knowing where logs are stored (/var/log) helps you diagnose issues.
- Understanding /etc allows you to configure system settings effectively.
- Knowing /home helps you manage user data and permissions.
- This structure is consistent across most Linux distributions, making it a universal skill for Linux users.

Pro Tip

Use the **tree** command (if installed) to visualize the directory structure in a tree-like format.



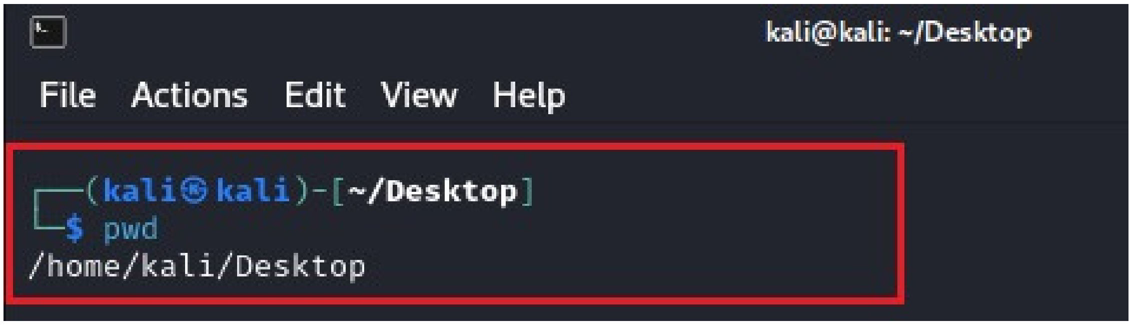
```
┌──(kali㉿kali)-[/]
└─$ tree -L 1 /
/
├── bin → usr/bin
├── boot
├── dev
├── etc
├── home
├── initrd.img → boot/initrd.img-6.11.2-amd64
├── initrd.img.old → boot/initrd.img-6.11.2-amd64
├── lib → usr/lib
├── lib32 → usr/lib32
├── lib64 → usr/lib64
├── lost+found
├── media
├── mnt
├── opt
├── proc
├── root
├── run
├── sbin → usr/sbin
├── srv
├── sys
├── tmp
├── usr
├── var
├── vmlinuz → boot/vmlinuz-6.11.2-amd64
└── vmlinuz.old → boot/vmlinuz-6.11.2-amd64

22 directories, 4 files
```

3.Navigating the file system.

Navigating the Linux file system is one of the most fundamental skills you'll need as a Linux user. Whether you're managing files, troubleshooting issues, or configuring services, knowing how to move around the file system efficiently is essential. In this section, we'll cover the basic commands for navigating directories and inspecting their contents.

pwd (Print Working Directory)

The pwd command displays the current directory you're in.
This is especially useful when you're deep in the directory structure and need to know your exact location.

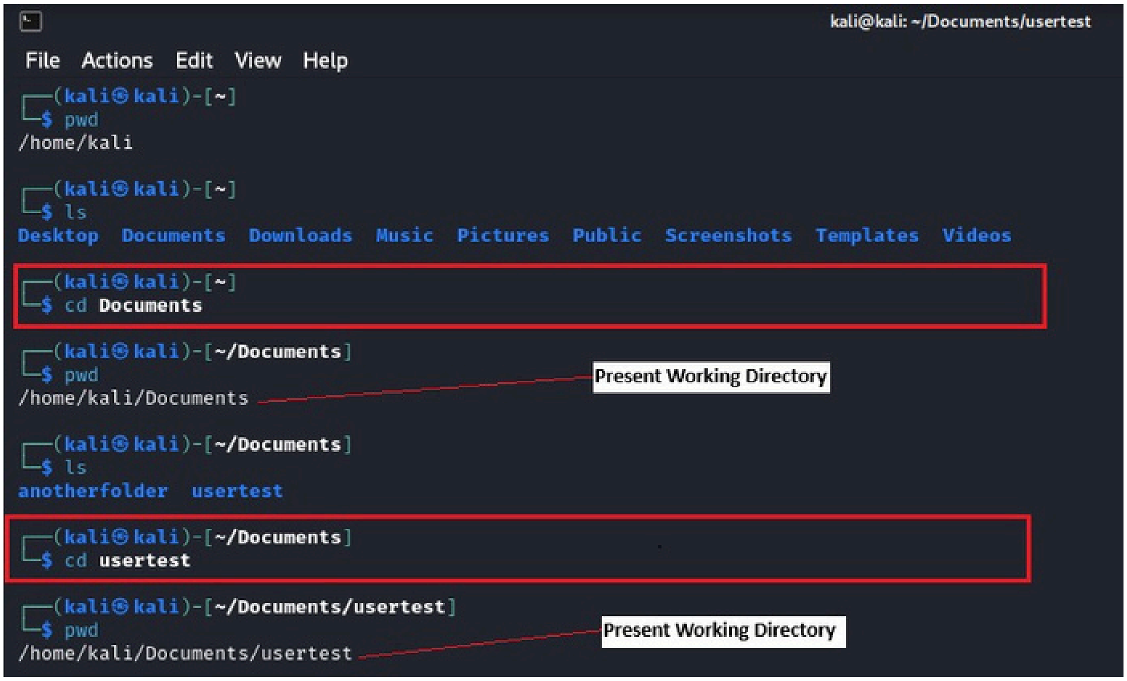


```
┌──(kali㉿kali)-[~/Desktop]
└─$ pwd
/home/kali/Desktop
```

This output shows that you're currently in the Desktop directory under the kali user's home directory.

cd (Change Directory)

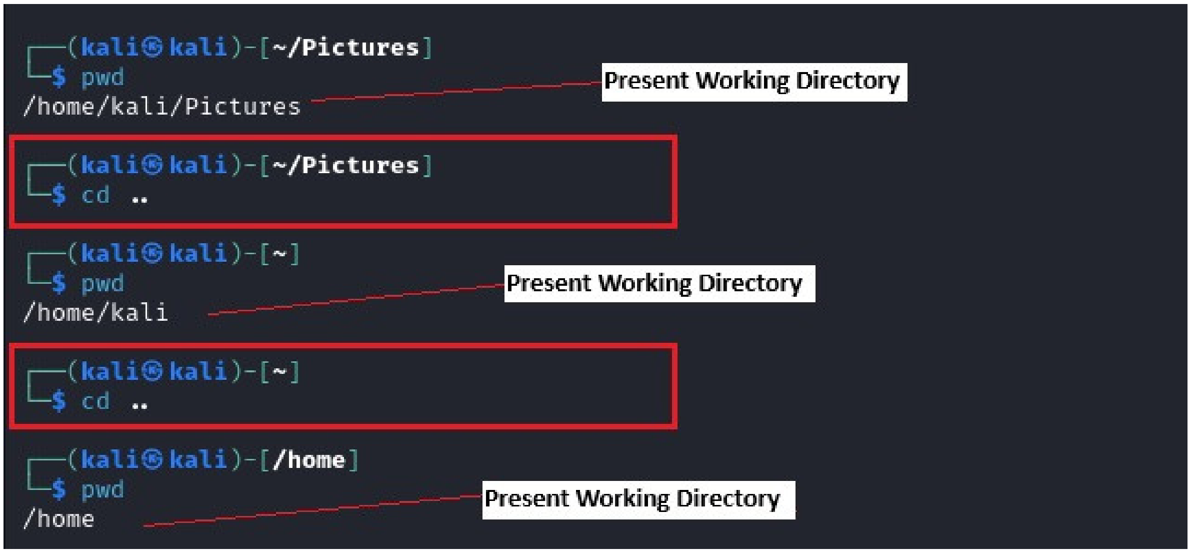
The cd command allows you to move between directories.
You can specify an absolute path (starting from /) or a relative path (relative to your current directory).


kali@kali: ~/Documents/usertest
File Actions Edit View Help
Present Working Directory
Present Working Directory

```
┌──(kali㉿kali)-[~]
└─$ pwd
/home/kali

┌──(kali㉿kali)-[~]
└─$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Screenshots  Templates  Videos

┌──(kali㉿kali)-[~]
└─$ cd Documents

┌──(kali㉿kali)-[~/Documents]
└─$ pwd
/home/kali/Documents

┌──(kali㉿kali)-[~/Documents]
└─$ ls
anotherfolder  usertest

┌──(kali㉿kali)-[~/Documents]
└─$ cd usertest

┌──(kali㉿kali)-[~/Documents/usertest]
└─$ pwd
/home/kali/Documents/usertest
```

cd .. (Move Up One Directory Level)

The .. symbol represents the parent directory.
Using cd .. moves you up one level in the directory hierarchy.

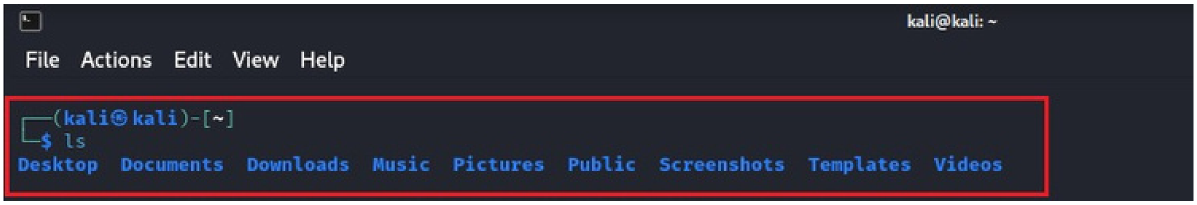


```
┌──(kali㉿kali)-[~/Pictures]
└─$ pwd
/home/kali/Pictures

┌──(kali㉿kali)-[~/Pictures]
└─$ cd ..

┌──(kali㉿kali)-[~]
└─$ pwd
/home/kali

┌──(kali㉿kali)-[~]
└─$ cd ..

┌──(kali㉿kali)-[/home]
└─$ pwd
/home
```

Present Working Directory
Present Working Directory
Present Working Directory

ls (List Directory Contents)

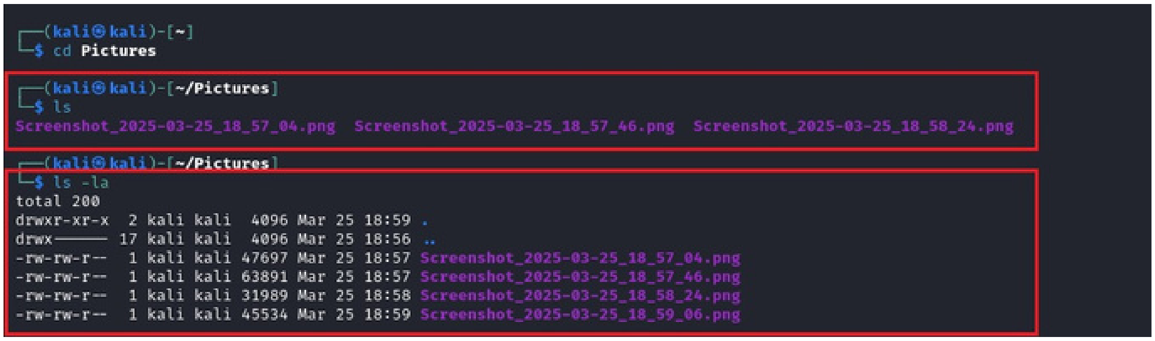
The ls command lists the files and directories in the current directory.
By default, it shows visible files and directories, but you can customize its output with options.



```
┌──(kali㉿kali)-[~]
└─$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Screenshots  Templates  Videos
```

ls -la (List Detailed File Information)

The ls -la command provides a detailed listing of files and directories, including hidden files.
The -l option shows details like permissions, ownership, size, and modification date.
The -a option includes hidden files (those starting with a .).
Example:



```
┌──(kali㉿kali)-[~]
└─$ cd Pictures

┌──(kali㉿kali)-[~/Pictures]
└─$ ls
Screenshot_2025-03-25_18_57_04.png  Screenshot_2025-03-25_18_57_46.png  Screenshot_2025-03-25_18_58_24.png

┌──(kali㉿kali)-[~/Pictures]
└─$ ls -la
total 200
drwxr-xr-x  2 kali kali  4096 Mar 25 18:59 .
drwx------ 17 kali kali  4096 Mar 25 18:56 ..
-rw-rw-r--  1 kali kali 47697 Mar 25 18:57 Screenshot_2025-03-25_18_57_04.png
-rw-rw-r--  1 kali kali 63891 Mar 25 18:57 Screenshot_2025-03-25_18_57_46.png
-rw-rw-r--  1 kali kali 31989 Mar 25 18:58 Screenshot_2025-03-25_18_58_24.png
-rw-rw-r--  1 kali kali 45534 Mar 25 18:59 Screenshot_2025-03-25_18_59_06.png
```

Key Differences Between ls and ls -la

ls:

Shows only visible files and directories.
No additional details like permissions or ownership.

ls -la:

Shows hidden files (those starting with a .).
Provides detailed information about each file and directory, including:

- File permissions (drwxr-xr-x).
- Number of links.
- Owner and group.
- File size.
- Last modification date

Why This Matters

Navigating the file system is the foundation of working with Linux.

- Whether you’re managing files, configuring services, or troubleshooting issues, you’ll constantly use these commands.

Understanding ls -la helps you:

- Identify hidden configuration files (e.g., .bashrc, .ssh).
- Check file permissions and ownership for security and troubleshooting.

Pro Tip

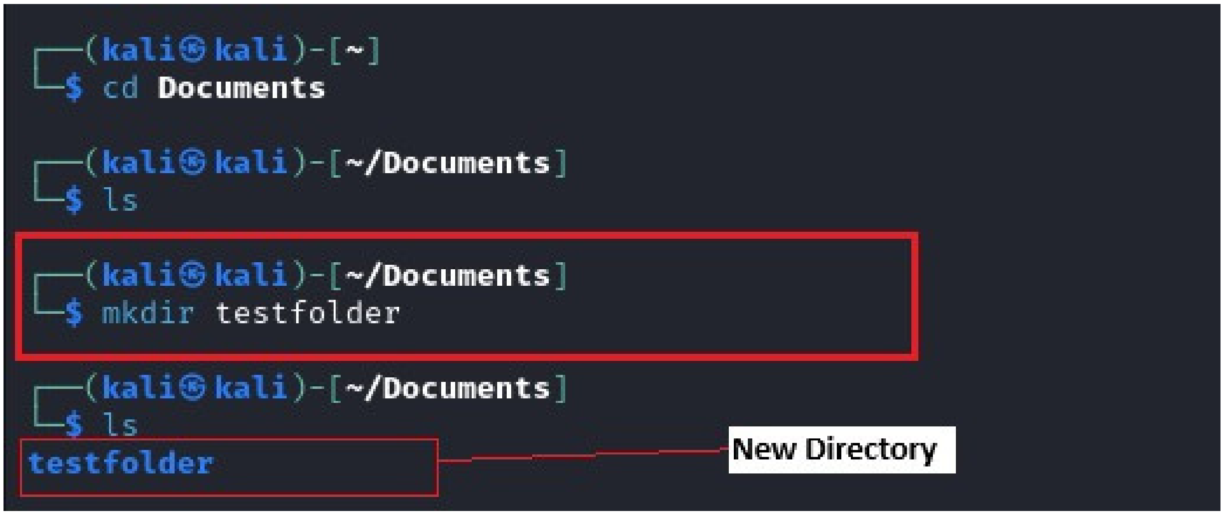
Use **tab** completion to save time when typing directory paths. For example:

1.4 Basic File Operations

Working with files and directories is a core part of using Linux. In this section, we'll cover the essential commands for **creating**, **viewing**, **copying**, **moving**, and **deleting** files and directories. These operations are the building blocks of file management in Linux.

mkdir (Make Directory)

The mkdir command creates a new directory.

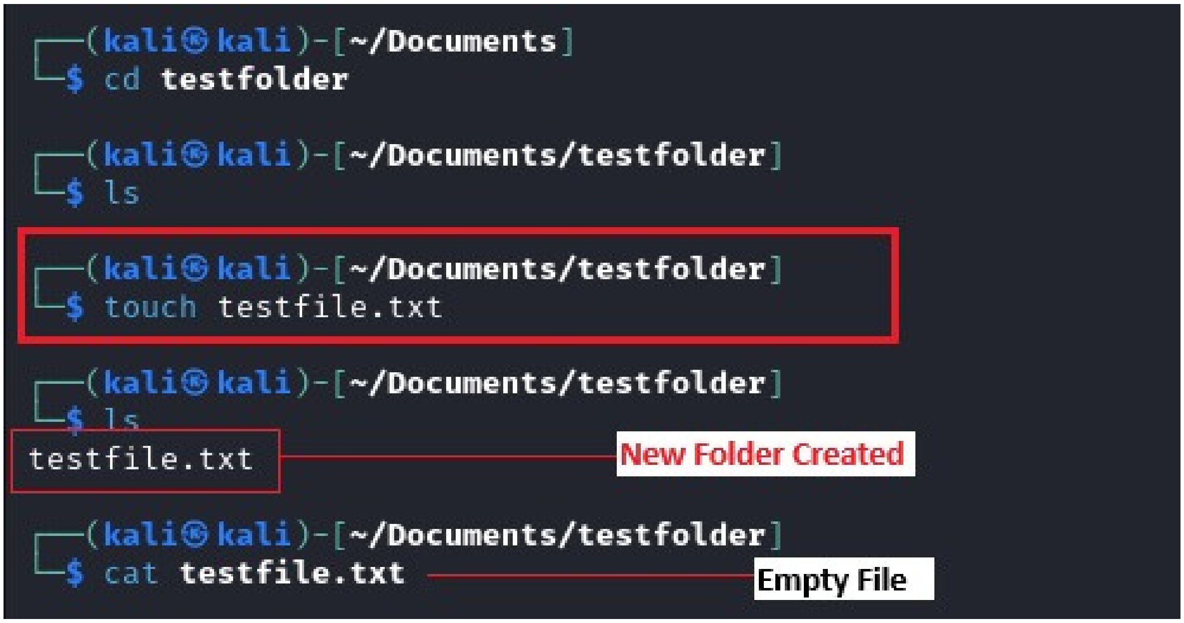


```
┌──(kali㉿kali)-[~]
└─$ cd Documents

┌──(kali㉿kali)-[~/Documents]
└─$ ls

┌──(kali㉿kali)-[~/Documents]
└─$ mkdir testfolder

┌──(kali㉿kali)-[~/Documents]
└─$ ls
testfolder
```

Syntax: mkdir directory_name

touch (Create an Empty File)

The touch command creates an empty file or updates the timestamp of an existing file.

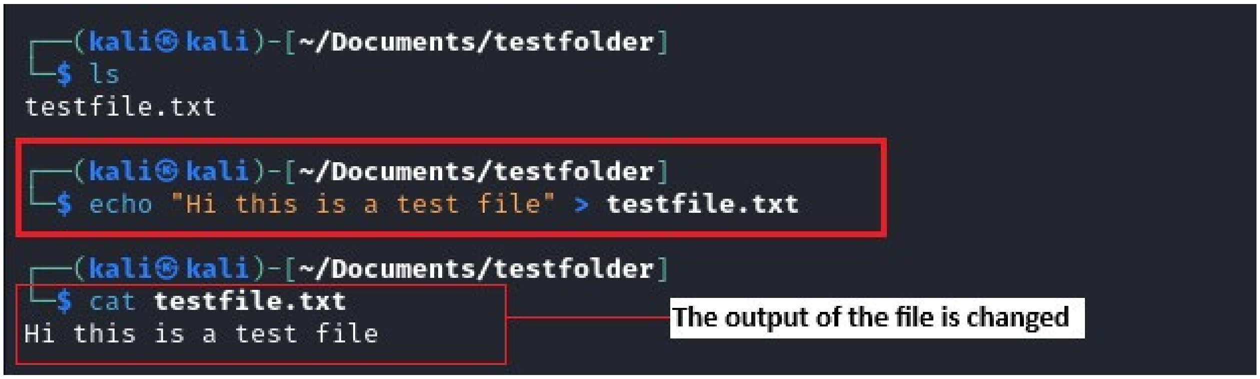


```
┌──(kali㉿kali)-[~/Documents]
└─$ cd testfolder

┌──(kali㉿kali)-[~/Documents/testfolder]
└─$ ls

┌──(kali㉿kali)-[~/Documents/testfolder]
└─$ touch testfile.txt

┌──(kali㉿kali)-[~/Documents/testfolder]
└─$ ls
testfile.txt

┌──(kali㉿kali)-[~/Documents/testfolder]
└─$ cat testfile.txt
```

Syntax: touch file_name

echo (Add Content to a File)

The echo command outputs text to the terminal or a file.
Use the > operator to redirect the output to a file.



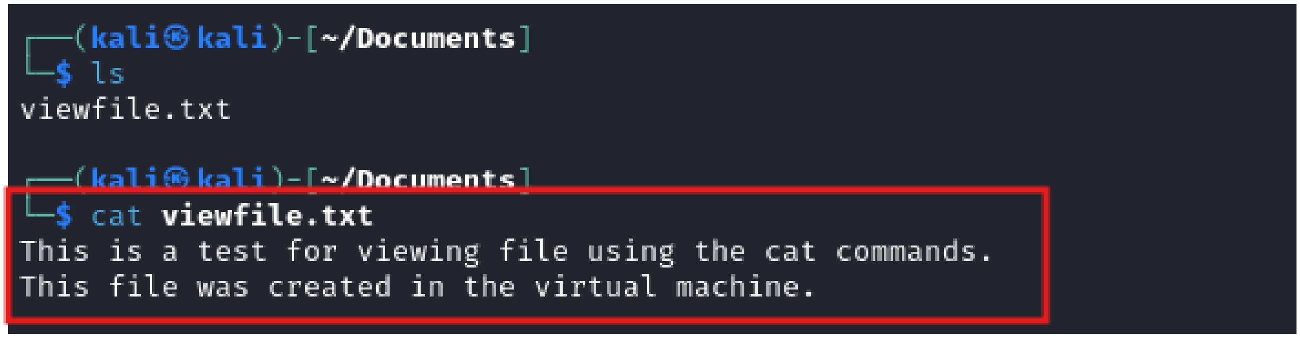
```
┌──(kali㉿kali)-[~/Documents/testfolder]
└─$ ls
testfile.txt

┌──(kali㉿kali)-[~/Documents/testfolder]
└─$ echo "Hi this is a test file" > testfile.txt

┌──(kali㉿kali)-[~/Documents/testfolder]
└─$ cat testfile.txt
Hi this is a test file
```

The output of the file is changed

Syntax: echo "text" > destination_file

cat (Display File Content)

The cat command displays the contents of a file.

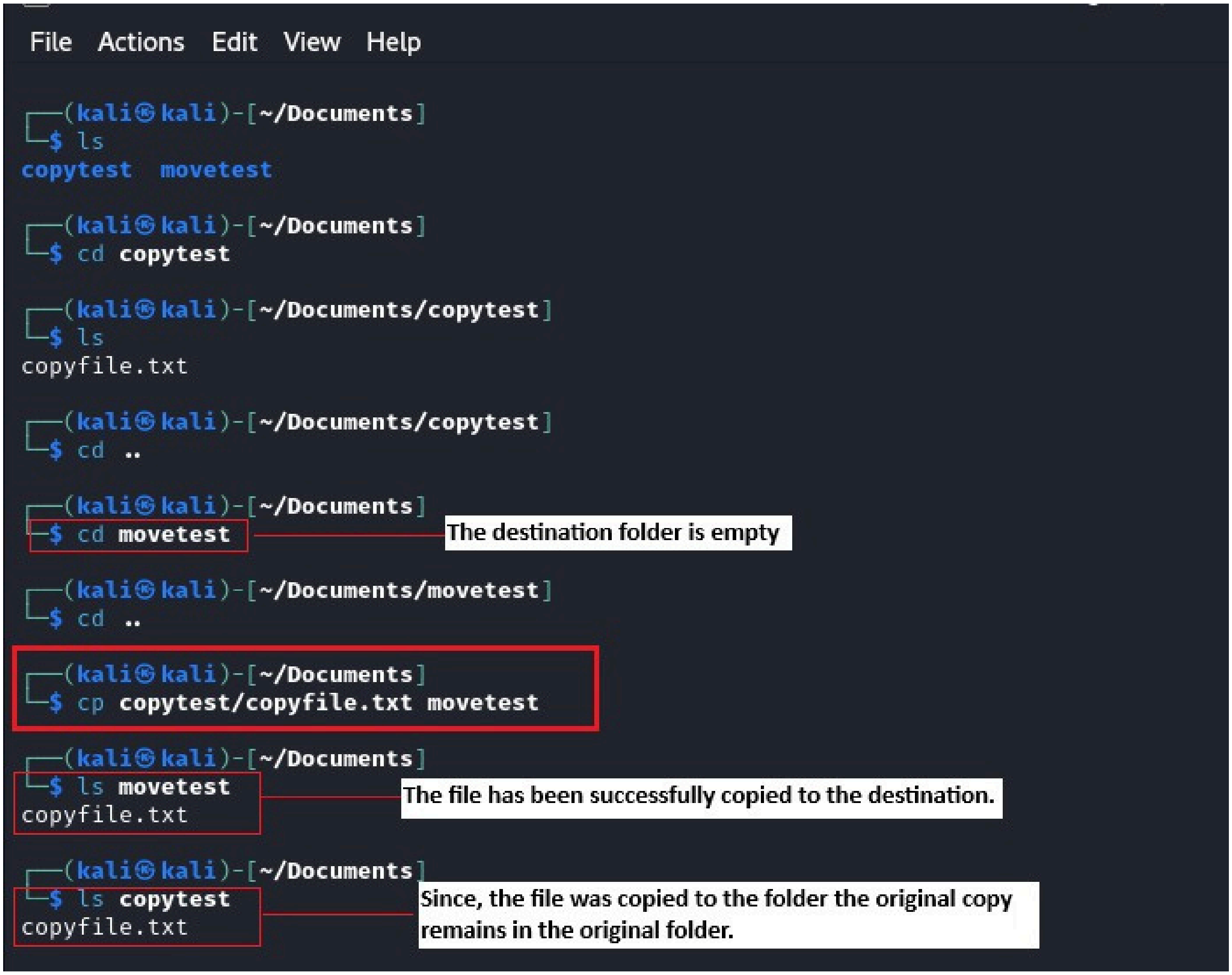


```
┌──(kali㉿kali)-[~/Documents]
└─$ ls
viewfile.txt

┌──(kali㉿kali)-[~/Documents]
└─$ cat viewfile.txt
This is a test for viewing file using the cat commands.
This file was created in the virtual machine.
```

Syntax: cat file_name

cp (Copy a File)

The cp command copies a file or directory to a new location.



```
┌──(kali㉿kali)-[~/Documents]
└─$ ls
copytest  movetest

┌──(kali㉿kali)-[~/Documents]
└─$ cd copytest

┌──(kali㉿kali)-[~/Documents/copytest]
└─$ ls
copyfile.txt

┌──(kali㉿kali)-[~/Documents/copytest]
└─$ cd ..

┌──(kali㉿kali)-[~/Documents]
└─$ cd movetest

┌──(kali㉿kali)-[~/Documents/movetest]
└─$ cd ..

┌──(kali㉿kali)-[~/Documents]
└─$ cp copytest/copyfile.txt movetest

┌──(kali㉿kali)-[~/Documents]
└─$ ls movetest
copyfile.txt

┌──(kali㉿kali)-[~/Documents]
└─$ ls copytest
copyfile.txt
```

The destination folder is empty

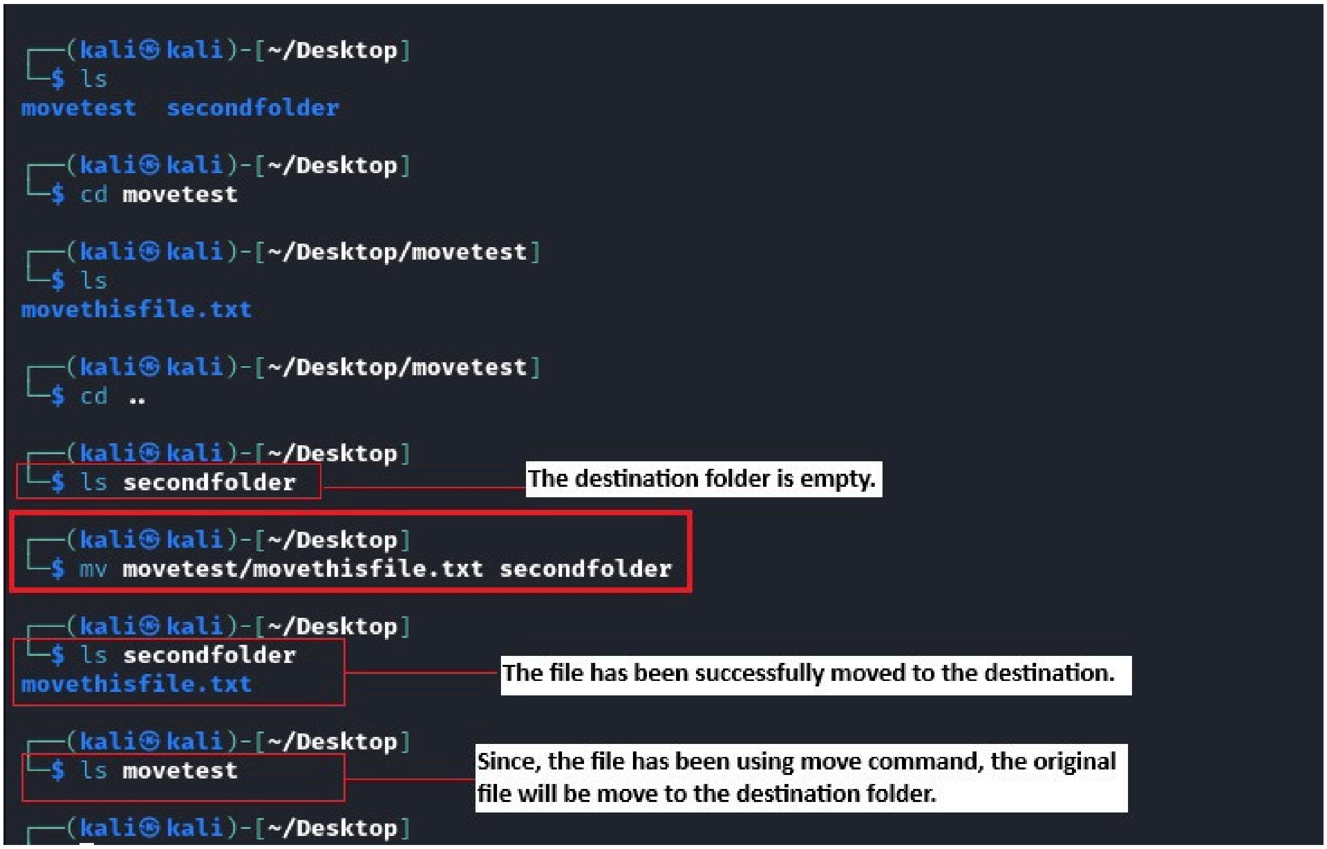
The file has been successfully copied to the destination.

Since, the file was copied to the folder the original copy remains in the original folder.

Syntax: cp source_location/file_name destination folder

mv (Move or Rename a File)

The mv command moves a file or directory to a new location or renames it.



```
┌──(kali㉿kali)-[~/Desktop]
└─$ ls
movetest  secondfolder

┌──(kali㉿kali)-[~/Desktop]
└─$ cd movetest

┌──(kali㉿kali)-[~/Desktop/movetest]
└─$ ls
movethisfile.txt

┌──(kali㉿kali)-[~/Desktop/movetest]
└─$ cd ..

┌──(kali㉿kali)-[~/Desktop]
└─$ ls secondfolder

┌──(kali㉿kali)-[~/Desktop]
└─$ mv movetest/movethisfile.txt secondfolder

┌──(kali㉿kali)-[~/Desktop]
└─$ ls secondfolder
movethisfile.txt

┌──(kali㉿kali)-[~/Desktop]
└─$ ls movetest

┌──(kali㉿kali)-[~/Desktop]
```

The destination folder is empty.

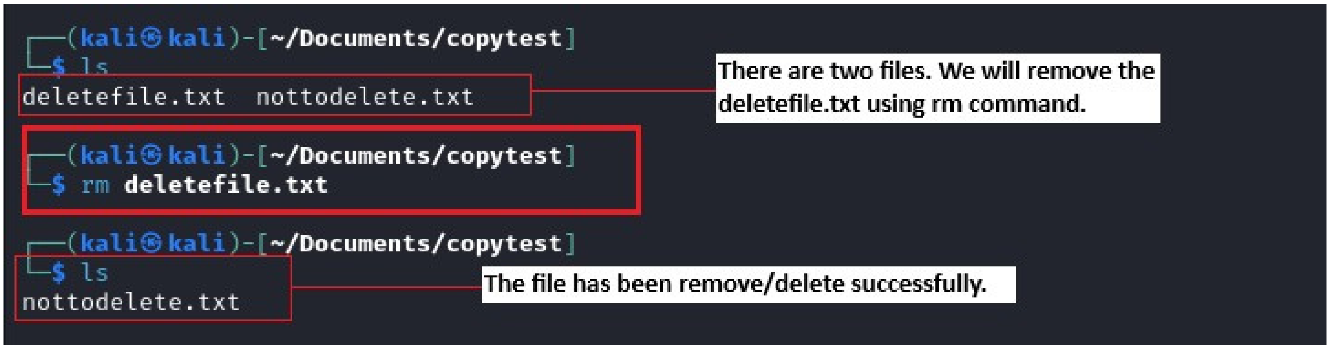
The file has been successfully moved to the destination.

Since, the file has been using move command, the original file will be move to the destination folder.

Syntax: mv source_destination/file_name destination folder

rm (Delete a File)

The rm command removes a file or directory.



```
┌──(kali㉿kali)-[~/Documents/copytest]
└─$ ls
deletefile.txt  nottodelete.txt

┌──(kali㉿kali)-[~/Documents/copytest]
└─$ rm deletefile.txt

┌──(kali㉿kali)-[~/Documents/copytest]
└─$ ls
nottodelete.txt
```

There are two files. We will remove the deletefile.txt using rm command.

The file has been remove/delete successfully.

Syntax: rm file_name

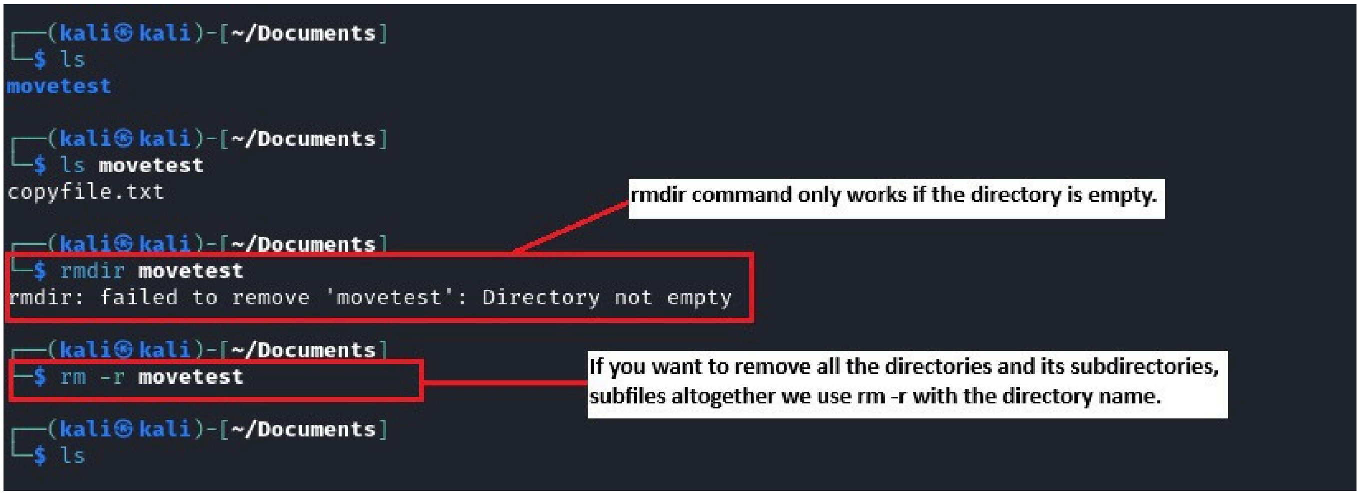
Caution: This action is irreversible!

Why This Matters

These commands are the foundation of file management in Linux.
Whether you’re organizing files, backing up data, or cleaning up your system, you’ll use these operations daily.

Pro Tip

Use the -r option with rm to delete directories and their contents recursively. For example:



```
┌──(kali㉿kali)-[~/Documents]
└─$ ls
movetest

┌──(kali㉿kali)-[~/Documents]
└─$ ls movetest
copyfile.txt

┌──(kali㉿kali)-[~/Documents]
└─$ rmdir movetest
rmdir: failed to remove 'movetest': Directory not empty

┌──(kali㉿kali)-[~/Documents]
└─$ rm -r movetest

┌──(kali㉿kali)-[~/Documents]
└─$ ls
```

rmdir command only works if the directory is empty.

If you want to remove all the directories and its subdirectories, subfiles altogether we use rm -r with the directory name.

Syntax: rm-r directory_name

Viewing and Editing Files

Linux offers a variety of tools for viewing and editing files, from simple command-line editors to graphical interfaces.

- echo: Creates or appends content to a file.
- cat: Displays file content.
- touch: Creates an empty file.
- nano: Beginner-friendly text editor.
- vim: Advanced text editor.
- vi: Legacy text editor.
- mousepad: Lightweight GUI editor.
- gedit: Default GNOME text editor.