Linux Hands-on Practice Guide

What is Linux?

Linux is primarily the kernel of an operating system.It was created by Linus Torvalds in the 1990's .

Linux is a free and open-source operating system based on UNIX. It serves as the interface between user applications and computer hardware. Designed to be stable, secure, and flexible, Linux is widely used in servers, cloud infrastructure, mobile devices, networking equipment, and embedded systems.

Why Linux?

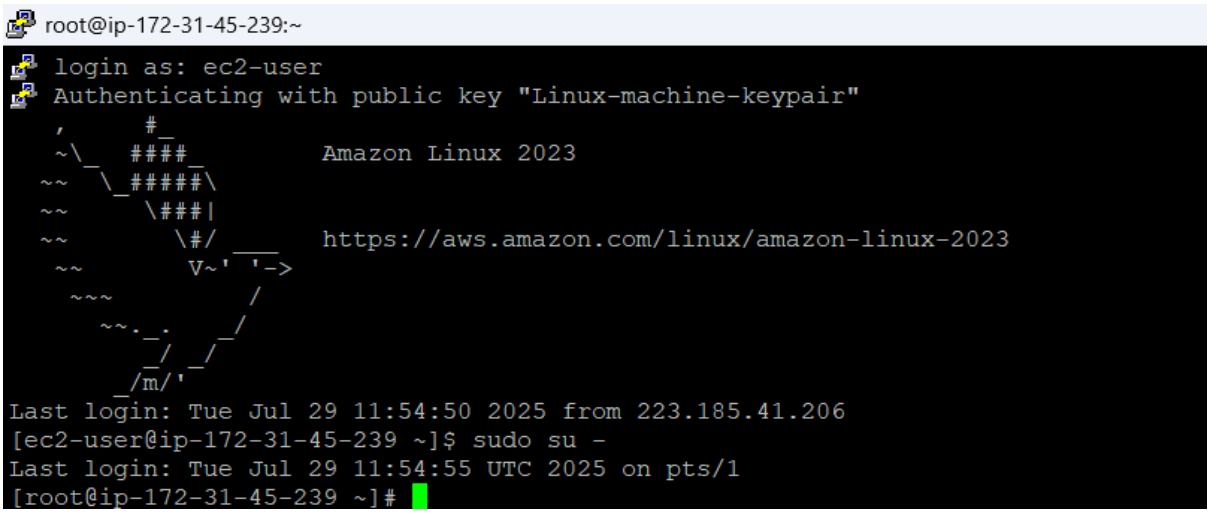
- **Security**: Built with strong permission and access control mechanisms
- **Free & Open Source**: No licensing cost; customizable source code
- **Stability & Performance**: Ideal for long-running, mission-critical systems
- **Package Ecosystem**: Offers thousands of tools, libraries, and services
- **Community & Enterprise Support**: Backed by both global communities and vendors like Red Hat, Canonical, etc.
- **Powerful CLI**: Efficient for automation, scripting, and remote administration

How Linux Works?

1. **User Interface**: Users interact via Graphical (GUI) or Command Line (CLI).
2. **Shell**: Receives user commands, interprets them, and communicates with the kernel.
3. **Kernel**: Core component that manages system resources (CPU, memory, I/O) and handles process control, file management, and hardware communication.
4. **Hardware**: The physical layer including memory, processors, disk drives, etc.

◈ Linux Overview

- Linux is a free, open-source, multi-user, multi-tasking operating system.
- It supports both GUI (Graphical User Interface) and CLI (Command Line Interface).
- Case-sensitive and security-focused with fine-grained control over user access and file permissions.



Screenshot 1: Linux Login Shell Output

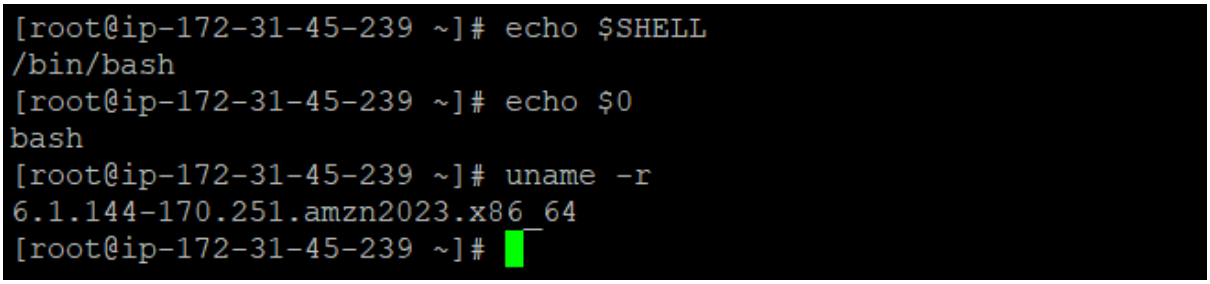
◈ Core Shell & Kernel Interaction

Shell: Interface between user and kernel.

- echo $SHELL — Shows default shell
- echo $0 — Shows current shell
- sh, bash, csh, tcsh — Switch between shell types

Kernel: Core engine interacting with hardware

- uname -r — View kernel version



Screenshot 2: Shell & Kernel Version Output

◈ File and Directory Operations

- touch file.txt — Create an empty file
- echo "Hello" > hello.txt — Create/write to file
- cat file.txt — Display file content
- mkdir newdir — Create directory
- mkdir -p /path/to/multiple/dirs — Create nested directories
- rm -rf file.txt — Delete file or directory forcefully

Bulk creation examples:

- touch file{1..100} — Create 100 files
- mkdir dir{1..5} — Create multiple directories

```
[root@ip-172-31-45-239 ~]# touch file{1..3}
[root@ip-172-31-45-239 ~]# mkdir dir1 dir2
[root@ip-172-31-45-239 ~]# ls -l
total 0
drwxr-xr-x. 2 root root 6 Jul 29 12:12 dir1
drwxr-xr-x. 2 root root 6 Jul 29 12:12 dir2
-rw-r--r--. 1 root root 0 Jul 29 12:12 file1
-rw-r--r--. 1 root root 0 Jul 29 12:12 file2
-rw-r--r--. 1 root root 0 Jul 29 12:12 file3
[root@ip-172-31-45-239 ~]# rm -rf *
[root@ip-172-31-45-239 ~]# ls -l
total 0
```

Screenshot 3: File & Directory Creation and Deletion Examples

◈ Permissions and Ownership

Viewing Permissions:

- ls -l — Long listing of files
- ls -lh — Human-readable format

Modify Permissions:

- chmod 755 file.sh — rwxr-xr-x
- chmod -R 700 dir/ — Recursive permission set

Ownership:

- chown user:group file — Change owner and group
- chgrp developers file — Change group only

Special Permissions:

- chmod g+s shared_folder/ — Set group ID on directory
- chmod +t /tmp — Sticky bit for secure shared directories

```
[root@ip-172-31-45-239 ~]# useradd jayshri
[root@ip-172-31-45-239 ~]# touch file1
[root@ip-172-31-45-239 ~]# ls -l
total 0
-r--------. 1 root root 0 Jul 29 12:22 file1
[root@ip-172-31-45-239 ~]# chmod 644 file1
[root@ip-172-31-45-239 ~]# ls -l
total 0
-rw-r--r--. 1 root root 0 Jul 29 12:22 file1
[root@ip-172-31-45-239 ~]# chown jayshri file1
[root@ip-172-31-45-239 ~]# ls -l
total 0
-rw-r--r--. 1 jayshri root 0 Jul 29 12:22 file1
[root@ip-172-31-45-239 ~]#
```

Screenshot 4: Permissions with chmod/chown Demonstration

◈ Linux File System Hierarchy

- / — Root directory
- /home — User home directories
- /etc — System config files
- /var — Logs, variable data
- /tmp — Temporary files
- /boot — Bootloader files
- /usr — User utilities

Navigation:

- pwd — Present working directory
- cd /path — Absolute path navigation
- cd .. — Go up one directory
- cd - — Switch to previous directory

```
[root@ip-172-31-35-183 etc]# cd /
[root@ip-172-31-35-183 /]# ls
bin   dev  home  lib64  media  opt   root  sbin  sys  usr
boot  etc  lib   local  mnt    proc  run   srv   tmp  var
[root@ip-172-31-35-183 /]# cd /home
[root@ip-172-31-35-183 home]# ls
ec2-user
[root@ip-172-31-35-183 home]# cd /tmp
[root@ip-172-31-35-183 tmp]# ls
systemd-private-ed1d5c6c6cd7493eb7e78b691e04afd5-chronyd.service-vlHCZp
systemd-private-ed1d5c6c6cd7493eb7e78b691e04afd5-dbus-broker.service-CmqaV1
systemd-private-ed1d5c6c6cd7493eb7e78b691e04afd5-policy-routes@enX0.service-DdiLUc
systemd-private-ed1d5c6c6cd7493eb7e78b691e04afd5-refresh-policy-routes@enX0.service-h9JU21
systemd-private-ed1d5c6c6cd7493eb7e78b691e04afd5-systemd-logind.service-t74jjF
systemd-private-ed1d5c6c6cd7493eb7e78b691e04afd5-systemd-resolved.service-I5636x
[root@ip-172-31-35-183 tmp]# cd /boot
[root@ip-172-31-35-183 boot]# ls
System.map-6.1.144-170.251.amzn2023.x86_64  initramfs-6.1.144-170.251.amzn2023.x86_64.img
config-6.1.144-170.251.amzn2023.x86_64      loader
efi                                         symvers-6.1.144-170.251.amzn2023.x86_64.gz
grub2                                       vmlinuz-6.1.144-170.251.amzn2023.x86_64
[root@ip-172-31-35-183 boot]# cd /var
[root@ip-172-31-35-183 var]# ls
account  cache  empty  games     lib    lock  mail  opt      run   tmp
adm      db     ftp    kerberos  local  log   nis   preserve  spool  yp
[root@ip-172-31-35-183 var]# cd /usr
[root@ip-172-31-35-183 usr]# ls
bin  games  include  lib  lib64  libexec  local  sbin  share  src  tmp
[root@ip-172-31-35-183 usr]# cd /etc
[root@ip-172-31-35-183 etc]# ls
DIR_COLORS              filesystems        machine-id       rsyncd.conf
DIR_COLORS.lightbgcolor  fstab             magic            rsyslog.d
```

Screenshot 5: Exploring File System with ls and cd

◈ User & Group Management

User Operations:

- useradd devuser — Add user
- passwd devuser — Set password
- userdel -r devuser — Remove user with home dir
- usermod -u 1101 devuser — Modify UID

Group Management:

- groupadd developers
- usermod -aG developers devuser — Add to group
- gpasswd -M user1,user2 devgroup — Set group members

Check Membership:

- id devuser
- groups devuser

Sudo Access:

- visudo → Add: devuser ALL=(ALL) ALL

```
[root@ip-172-31-43-49 ~]# useradd rohan
[root@ip-172-31-43-49 ~]# groupadd aws
[root@ip-172-31-43-49 ~]# id rohan
uid=1001(rohan) gid=1001(rohan) groups=1001(rohan)
[root@ip-172-31-43-49 ~]# useradd -G aws rohan
useradd: user 'rohan' already exists
[root@ip-172-31-43-49 ~]# usermod -G aws rohan
[root@ip-172-31-43-49 ~]# id rohan
uid=1001(rohan) gid=1001(rohan) groups=1001(rohan),1002(aws)
[root@ip-172-31-43-49 ~]# groups rohan
rohan : rohan aws
[root@ip-172-31-43-49 ~]#
```

Screenshot 6: Creating Users and Adding to Groups

◈ Text Editors (vi/vim)

Open File:

- vi file.txt
- vim file.txt

Modes:

- Press i for insert mode
- Press Esc then :wq to save and quit

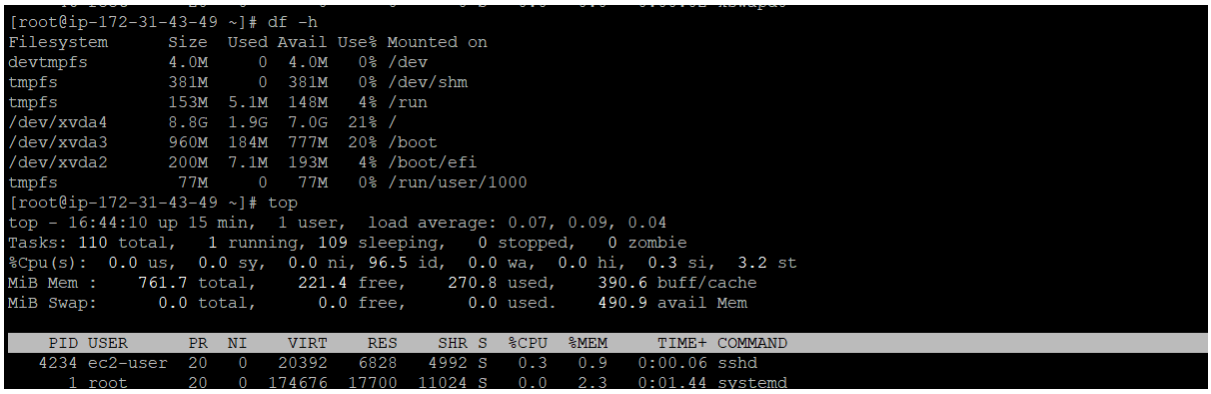
Navigation/Editing:

- gg — Start of file
- G — End of file
- :5 — Line 5
- dd, 2dd — Delete lines
- yy, p — Copy and paste
- :%s/old/new/g — Replace text

Screenshot 7: vi Editor Usage

◈ Monitoring and Diagnostics

- top, htop — Real-time resource monitoring
- ps aux — List processes
- df -h — Disk space
- du -sh folder/ — Folder size
- free -m — Memory usage
- uptime — System load



Screenshot 8: Output from top and df -h

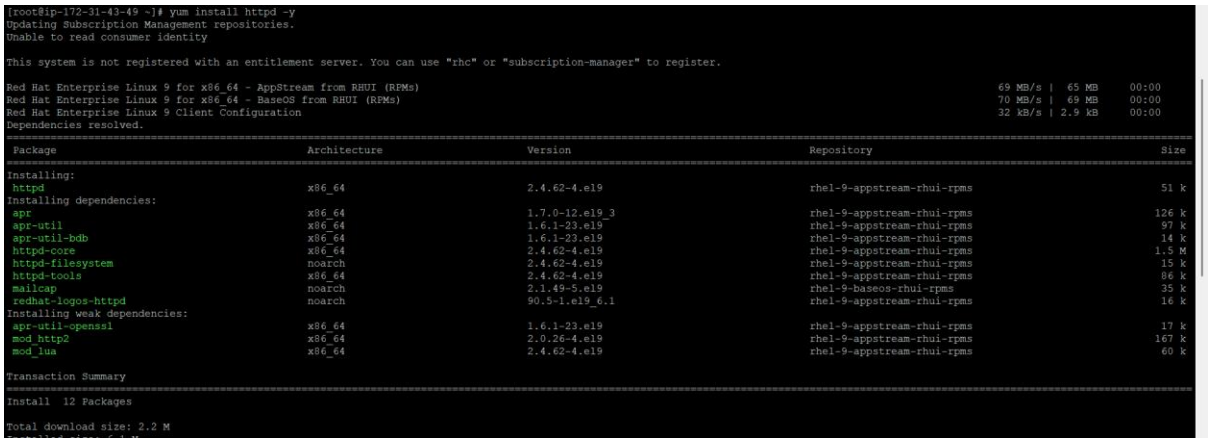
◈ Package Management

RPM:

- rpm -ivh pkg.rpm — Install package
- rpm -qa — List installed packages

YUM/DNF:

- yum install httpd -y
- yum remove git -y
- yum list installed



Screenshot 9: Installing Packages with YUM

◈ Cron Jobs (Scheduling)

- crontab -e — Edit crontab
- crontab -l — List jobs

Syntax:

┌──────────── minute (0 - 59)

│ ┌──────────── hour (0 - 23)

│ │ ┌──────────── day of month (1 - 31)

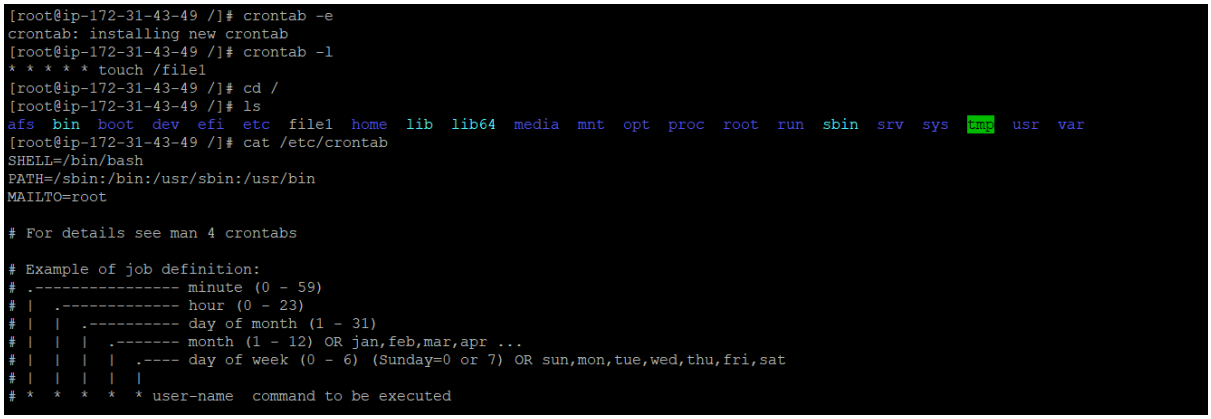
│ │ │ ┌──────────── month (1 - 12)

│ │ │ │ ┌──── day of week (0 - 6) (Sunday=0)

│ │ │ │ │

* * * * * command_to_execute

- Example: 05 11 * * * /home/user/script.sh



Screenshot 10: Viewing and Creating Cron Jobs

◈ Hard vs Soft Links

- ln file1 file_hard — Hard link (same inode)
- ln -s file1 file_soft — Soft link (symbolic)

Check: ls -li

Feature	Hard Link	Soft Link (Symbolic Link)
Definition	A direct reference to the physical data (inode) on disk.	A pointer (shortcut) to the file name, not the data itself.
Works Across File Systems?	No – must be on the same filesystem.	Yes – can point across filesystems or partitions.
Points To	The file's inode.	The file's path/name.
Effect if Original File is Deleted	Data still accessible (until all hard links are removed).	Link breaks (becomes a dangling link).
Can Link to Directories?	Not usually (restricted to avoid loops).	Yes (commonly used for directories).
File Permissions	Same as the original file.	Independent of the original file's permissions.
Inode Number	Same inode as the original file.	Different inode (points to the file name).
Creation Command	ln source target	ln -s source target
Example	ln file1 file2	ln -s file1 file2

Screenshot 11: Hard vs Soft Link Comparison

◈ Networking Basics

- ip a — Show IP addresses
- hostname — Show hostname
- hostnamectl set-hostname newname
- ping google.com — Connectivity test

```
[root@ip-172-31-35-183 ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enX0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc fq_codel state UP group default qlen 1000
    link/ether 0e:43:86:4c:01:27 brd ff:ff:ff:ff:ff:ff
    altname eni-086972c2d0898c37b
    altname device-number-0.0
    inet 172.31.35.183/20 metric 512 brd 172.31.47.255 scope global dynamic enX0
       valid_lft 2984sec preferred_lft 2984sec
    inet6 fe80::c43:86ff:fe4c:127/64 scope link proto kernel_ll
       valid_lft forever preferred_lft forever
[root@ip-172-31-35-183 ~]# hostnamectl set-hostname jayshri
[root@ip-172-31-35-183 ~]# hostname
jayshri
[root@ip-172-31-35-183 ~]# ping google.com
PING google.com (216.58.215.46) 56(84) bytes of data.
64 bytes from par21s17-in-f14.1e100.net (216.58.215.46): icmp_seq=1 ttl=118 time=0.695 ms
64 bytes from par21s17-in-f14.1e100.net (216.58.215.46): icmp_seq=2 ttl=118 time=0.739 ms
64 bytes from par21s17-in-f14.1e100.net (216.58.215.46): icmp_seq=3 ttl=118 time=0.700 ms
64 bytes from par21s17-in-f14.1e100.net (216.58.215.46): icmp_seq=4 ttl=118 time=0.680 ms
```

Screenshot 12: Output of ip a and hostname

◈ ACL (Access Control List)

Set/Modify Permissions:

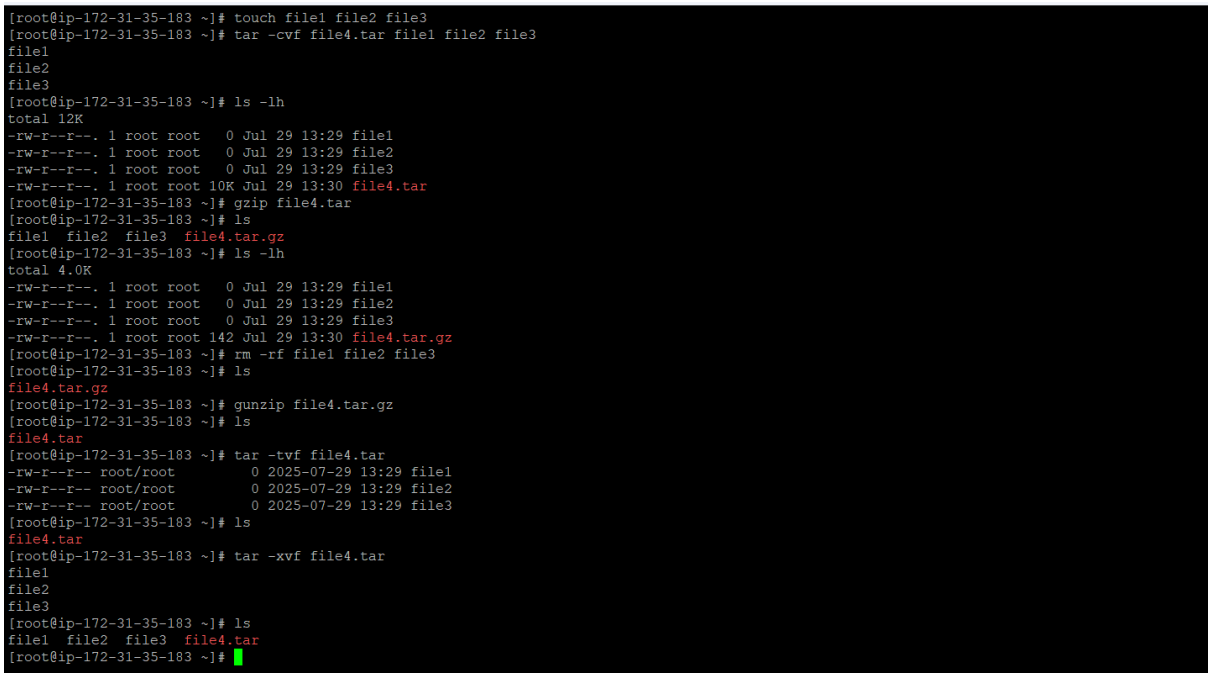
- setfacl -m u:devuser:rwx dir/
- setfacl -x u:devuser dir/ — Remove
- getfacl file/ — View ACL

```
[root@ip-172-31-35-183 ~]# ls -l
total 0
-rw-r--r--. 1 root root 0 Jul 29 14:00 file1
[root@ip-172-31-35-183 ~]# setfacl -m u:rohan:rw- file1
[root@ip-172-31-35-183 ~]# setfacl -m g:aws:rwx file1
[root@ip-172-31-35-183 ~]# getfacl file1
# file: file1
# owner: root
# group: root
user::rw-
user:rohan:rw-
group::r--
group:aws:rwx
mask::rwx
other::r--

[root@ip-172-31-35-183 ~]# ls -l
total 0
-rw-rwxr--+ 1 root root 0 Jul 29 14:00 file1
[root@ip-172-31-35-183 ~]# setfacl -x u:rohan file1
[root@ip-172-31-35-183 ~]# ls -l
total 0
-rw-rwxr--+ 1 root root 0 Jul 29 14:00 file1
[root@ip-172-31-35-183 ~]# getfacl file1
# file: file1
# owner: root
# group: root
user::rw-
group::r--
group:aws:rwx
mask::rwx
other::r--

[root@ip-172-31-35-183 ~]# setfacl -b file1
[root@ip-172-31-35-183 ~]# ls -l
total 0
-rw-r--r--. 1 root root 0 Jul 29 14:00 file1
[root@ip-172-31-35-183 ~]#
```

Screenshot 13: getfacl and setfacl Output

◈ Archiving and Compression

- tar -cvf archive.tar file1 file2
- tar -xvf archive.tar
- tar -czvf archive.tar.gz dir/
- tar -xzvf archive.tar.gz



Screenshot 14: tar Archive and Extraction Commands

◈ File Copying and Moving

- cp file1 file2
- cp -r dir1/ dir2/
- mv old.txt new.txt — Rename
- mv file.txt /tmp/ — Move file

Screenshot 15: Copy and Move File Commands Output

Conclusion

In conclusion, this guide encapsulates the essential Linux system administration competencies I have developed and validated through RHCSA certification. From managing users and groups to implementing permissions and Access Control Lists (ACLs), I have demonstrated the ability to maintain secure and organized systems. My clear understanding of hard and soft links, along with proficiency in copying and moving files and directories, reflects strong command over core file operations. Additionally, my knowledge of package management using YUM and RPM, job scheduling with crontab, and expertise in file compression and archiving techniques illustrate a comprehensive grasp of practical Linux administration tasks.

These skills collectively underscore my readiness to take on advanced server management, scripting, and automation responsibilities—establishing a strong, reliable foundation for future growth in professional Linux and DevOps environments.