Experiment No. 1

Title: Introduction to different Operating Systems and study file permissions in Linux

Batch: B-2 **Roll No: 16010422234** **Name: Chandana Galgali** **Date: 23/07/2024**

Experiment No: 1

Aim: Introduction to Operating Systems and implementation of file permission commands in Linux.

Resources needed: Any open source OS/ online CoCalc editor, internet

---

Theory:

Pre lab/Prior concepts:

Introduction of Operating System

An operating system acts as an intermediary between the user of a computer and computer hardware. The purpose of an operating system is to provide an environment in which a user can execute programs in a convenient and efficient manner.

An operating system is system software that manages the computer hardware. The hardware must provide appropriate mechanisms to ensure the correct operation of the computer system and to prevent user programs from interfering with the proper operation of the system.

Types of Operating System

Batch Operating System- Sequence of jobs in a program on a computer without manual interventions.

Time sharing operating System- allows many users to share the computer resources.(Max utilization of the resources).

Distributed operating System- Manages a group of different computers and makes them appear to be a single computer.

Network operating system- computers running in different operating systems can participate in a common network (It is used for security purposes).

Real time operating system- meant applications to fix the deadlines.

Linux Distribution

Linux distribution is an operating system that is made up of a collection of software based on Linux kernel or you can say distribution contains the Linux kernel and supporting libraries and software. And you can get a Linux based operating system by downloading one of the Linux distributions and these distributions are available for different types of devices like embedded devices, personal computers, etc. Around 600 + Linux Distributions are available and some of the popular Linux distributions are:

MX Linux, Manjaro, Linux Mint, elementary, Ubuntu, Debian, Solus, Fedora, openSUSE, Deepin

Ownership of Linux files

Every file and directory on your Unix/Linux system is assigned 3 types of owner, given below.

User

A user is the owner of the file. By default, the person who created a file becomes its owner. Hence, a user is also sometimes called an owner.

Group

A user- group can contain multiple users. All users belonging to a group will have the same access permissions to the file. Suppose you have a project where a number of people require access to a file. Instead of manually assigning permissions to each user, you could add all users to a group, and assign group permission to file such that only this group members and no one else can read or modify the files.

Other

Any other user who has access to a file. This person has neither created the file, nor does he belong to a usergroup who could own the file. Practically, it means everybody else. Hence, when you set the permission for others, it is also referred to as set permissions for the world.

Now, the big question arises how does Linux distinguish between these three user

types. Let us understand the Permission system on Linux.

Permissions

Every file and directory in your Linux system has the following 3 permissions defined for all the 3 owners discussed above.

Read: This permission gives you the authority to open and read a file. Read permission on a directory gives you the ability to list its content.

Write: The write permission gives you the authority to modify the contents of a file. The write permission on a directory gives you the authority to add, remove and rename files stored in the directory. Consider a scenario where you have to write permission on file but do not have write permission on the directory where the file is stored. You will be able to modify the file contents. But you will not be able to rename, move or remove the file from the directory.

Execute: In Windows, an executable program usually has an extension ".exe" and which you can easily run. In Linux, you cannot run a program unless the execute permission is set. If the execute permission is not set, you might still be able to see/modify the program code (provided read & write permissions are set), but not run it.

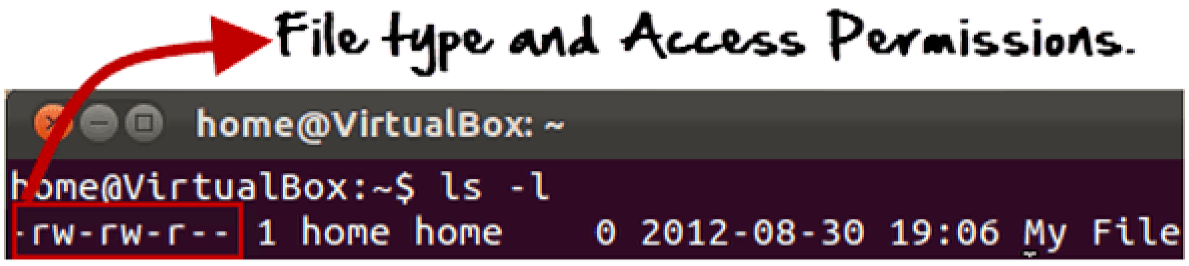
Access Permissions of files

Read access (mnemonic: r, binary weight: 4)
Permission to read the file; for directories this means the permission to list the contents of the file.
Write access (mnemonic: w, binary weight: 2)
Permission to modify the file; for directories, this means the permission to create or delete files.
Execute access (mnemonic: x, binary weight: 1)
Permission to execute the file; for directories, this means the permission to access files in the directory.
In the above output, these permissions appear in groups of three, discarding the very first character. Thus each triad is made of “rwx” and the absence of a permission is marked by the dash “-”. The three triads represent permissions for the file owner, the file group and the other groups respectively. The following interpretations can be thus be made about the file:
The file owner “sameer” has read and write permissions.
The file group “users” has only read permissions.
The other groups have only read permissions.

Owners can define the permissions on every file and

folder. ls - l on terminal gives

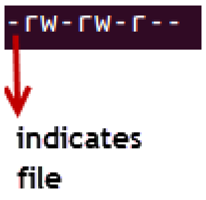
ls – l



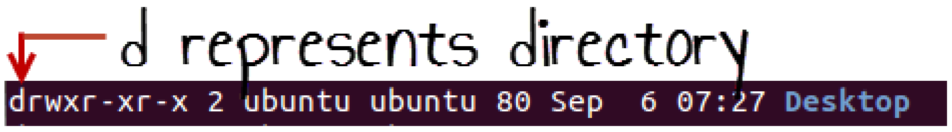
```
home@VirtualBox:~$ ls -l
-rw-rw-r-- 1 home home     0 2012-08-30 19:06 My File
```

Here, we have highlighted **'-rw-rw-r--'**and this weird looking code is the one that tells us about the permissions given to the owner, user group and the world.

Here, the first '-' implies that we have selected a file.p>



Else, if it were a directory, **d** would have been shown.



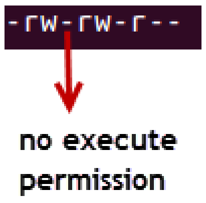
```
drwxr-xr-x 2 ubuntu ubuntu 80 Sep  6 07:27 Desktop
```

The characters are pretty easy to remember.

r = read permission
w = write permission
x = execute permission
- = no permission

Let us look at it this way.

The first part of the code is **'rw-'**. This suggests that the owner 'Home' can:



- Read the file
- Write or edit the file
- He cannot execute the file since the execute bit is set to '-'.

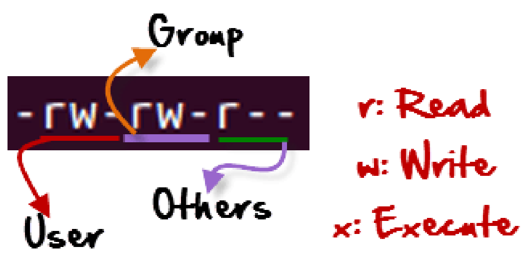
By design, many Linux distributions like Fedora, CentOS, Ubuntu, etc. will add users to a group of the same group name as the user name. Thus, a user 'tom' is added to a group named 'tom'.

The second part is **'rw-'.** It for the user group 'Home' and group-members can:

- Read the file
- Write or edit the file

The third part is for the world which means any user. It says **'r--'.** This means the user can only:

- Read the file



Changing file/directory permissions with 'chmod' command

Say you do not want your colleague to see your personal images. This can be achieved by changing file permissions.

We can use the '**chmod**' command which stands for 'change mode'. Using the command, we can set permissions (read, write, execute) on a file/directory for the owner, group and the world.

Syntax:
chmod permissions filename

There are 2 ways to use the command -

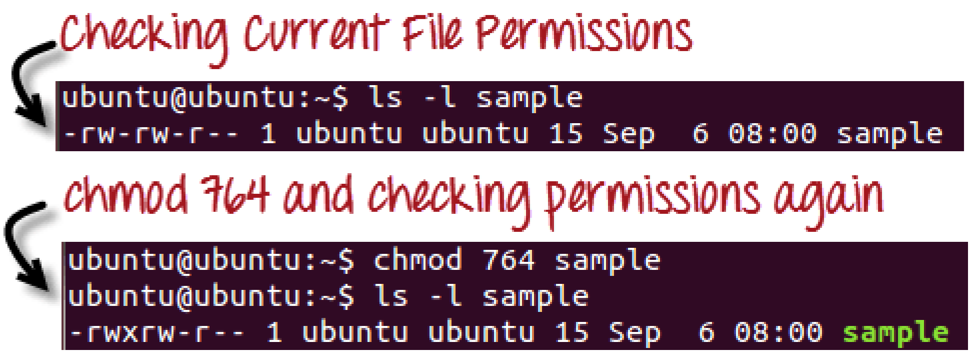
1. **Absolute mode**
2. **Symbolic mode**

Absolute (Numeric) Mode

In this mode, file permissions are not represented as characters but a three-digit octal number. The table below gives numbers for all permissions types.

	Permission Type	**Symbol**
0	No Permission	---
	Execute	--x
2	Write	-w-
	Execute + Write	-wx
4	Read	r--
	Read + Execute	r-x
6	Read +Write	rw-
	Read + Write +Execute	rwx

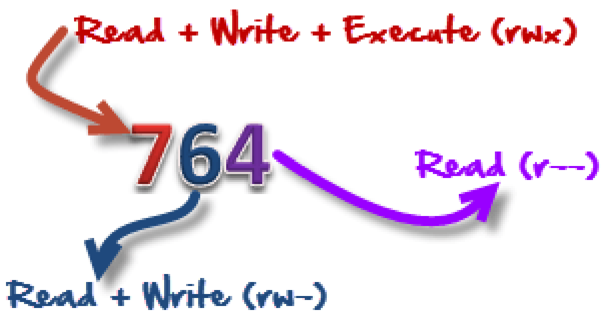
Let's see the chmod command in action.


Checking Current File Permissions
chmod 764 and checking permissions again

```
ubuntu@ubuntu:~$ ls -l sample
-rw-rw-r-- 1 ubuntu ubuntu 15 Sep  6 08:00 sample
```

```
ubuntu@ubuntu:~$ chmod 764 sample
ubuntu@ubuntu:~$ ls -l sample
-rwxrw-r-- 1 ubuntu ubuntu 15 Sep  6 08:00 sample
```

In the above-given terminal window, we have changed the permissions of the file ‘sample’ to '764'.



'764' absolute code says the following:

- Owner can read, write and execute
- Usergroup can read and write
- World can only read

This is shown as '-rwxrw-r-

This is how you can change the permissions on file by assigning an absolute number.

Symbolic Mode

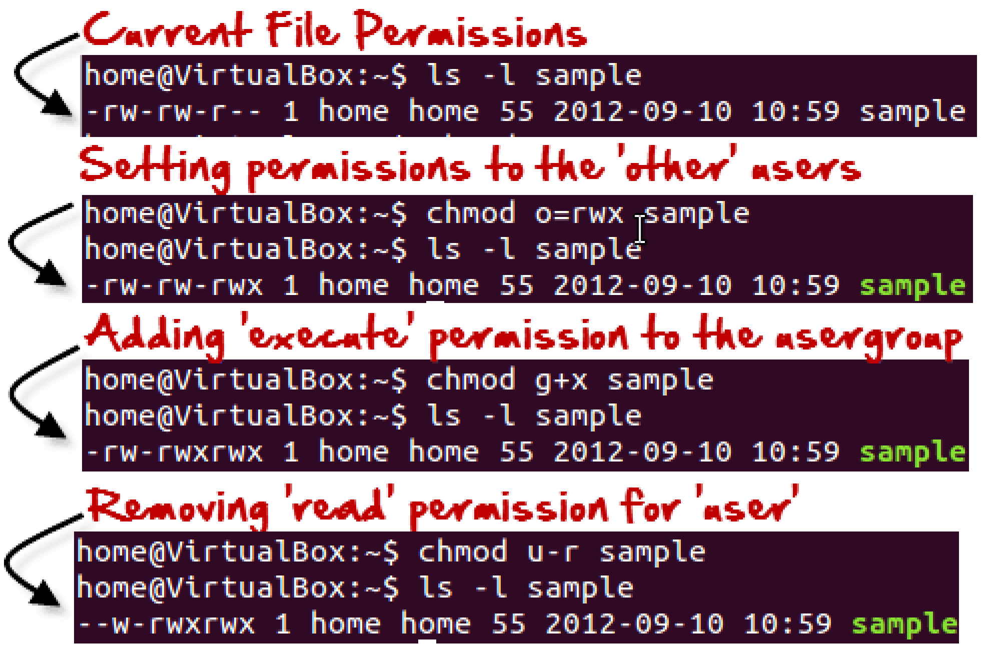
In the Absolute mode, you change permissions for all 3 owners. In the symbolic mode, you can modify permissions of a specific owner. It makes use of mathematical symbols to modify the file permissions.

Operato r	Descriptio n
+	Adds a permission to a file or directory
-	Removes the permission
=	Sets the permission and overrides the permissions set earlier.

The various owners are represented as:

User Denotations	
u	user/owner
g	group
o	other
a	all

We will not be using permissions in numbers like 755 but characters like rwx. Let's look into an example.


Current File Permissions
Setting permissions to the 'other' users
Adding 'execute' permission to the usergroup
Removing 'read' permission for 'user'

```
home@VirtualBox:~$ ls -l sample
-rw-rw-r-- 1 home home 55 2012-09-10 10:59 sample
```

```
home@VirtualBox:~$ chmod o=rwx sample
home@VirtualBox:~$ ls -l sample
-rw-rw-rwx 1 home home 55 2012-09-10 10:59 sample
```

```
home@VirtualBox:~$ chmod g+x sample
home@VirtualBox:~$ ls -l sample
-rw-rwxrwx 1 home home 55 2012-09-10 10:59 sample
```

```
home@VirtualBox:~$ chmod u-r sample
home@VirtualBox:~$ ls -l sample
--w-rwxrwx 1 home home 55 2012-09-10 10:59 sample
```

Activities:

Students have to explore different types of operating systems and differentiate them on the basis of pros and cons.

Batch Operating System:

Pros:

- Efficient for large jobs.
- Reduces idle time for the CPU.

Cons:

- No interaction with the user while the job is being processed.
- Difficult to debug.

Time-Sharing Operating System:

Pros:

- Multiple users can use the system simultaneously.
- Efficient utilization of resources.

Cons:

- Requires complex scheduling algorithms.
- Security and data integrity issues due to multiple users.

Distributed Operating System:

Pros:

- Resource sharing among multiple systems.
- Enhanced performance and reliability.

Cons:

- Network dependency.
- Complex to manage and configure.

Network Operating System:

Pros:

- Facilitates resource sharing across a network.
- Provides centralized control.

Cons:

- High cost of implementation.
- Dependency on the central server for operations.

Real-Time Operating System:

Pros:

- Guarantees a certain capability within a specified time constraint.
- Suitable for time-critical applications.

Cons:

- Limited multitasking capabilities.
- Expensive to implement.

Take any 5 domains like gaming, finance, banking etc and suggest which OS will be best suitable for these domains.

Gaming:

Best Suitable OS: Windows

Reason: Wide compatibility with games, extensive driver support, and optimized performance for gaming hardware.

Finance:

Best Suitable OS: macOS

Reason: High security, stability, and integration with financial software and tools.

Banking:

Best Suitable OS: Linux

Reason: Robust security features, stability, and the ability to run on various hardware configurations.

Education:

Best Suitable OS: Chrome OS

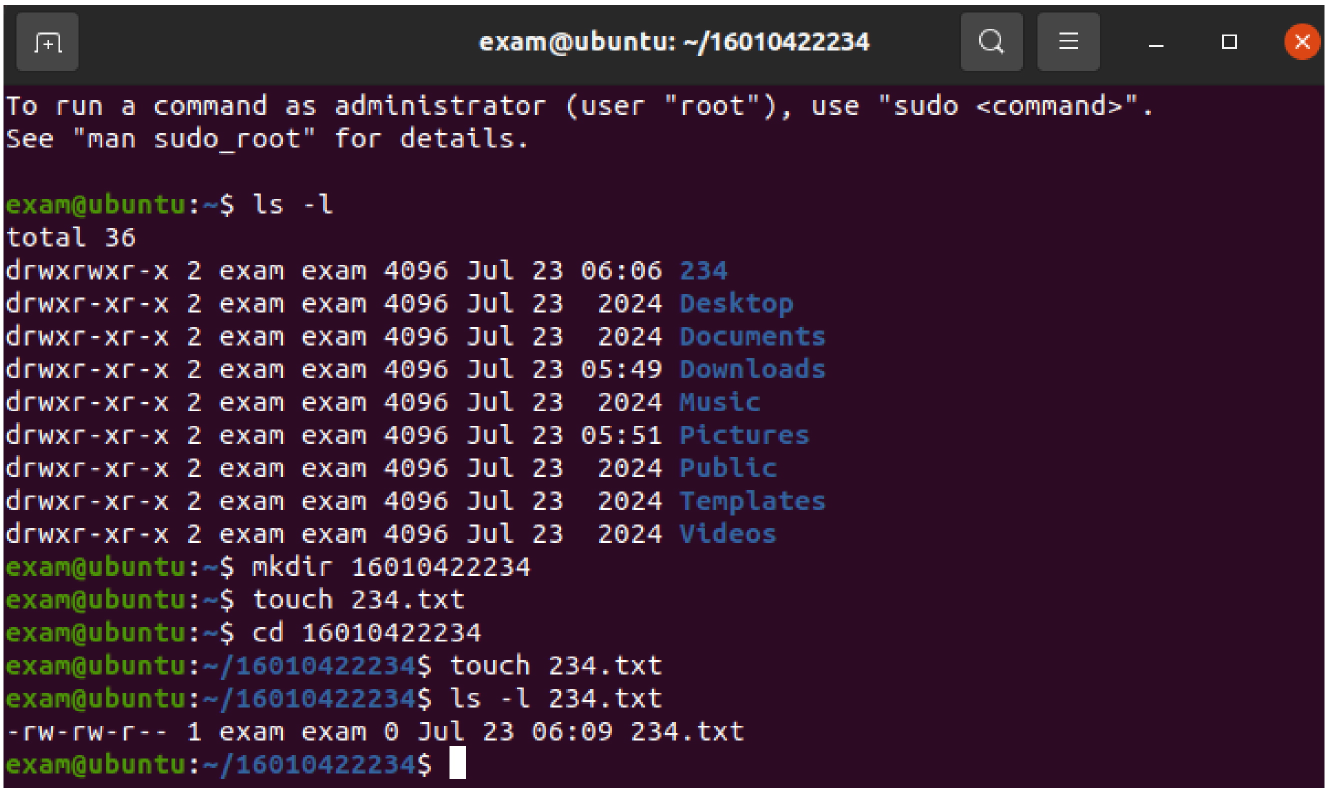
Reason: Simple to use, secure, and integrates well with educational tools and applications.

Software Development:

Best Suitable OS: Linux

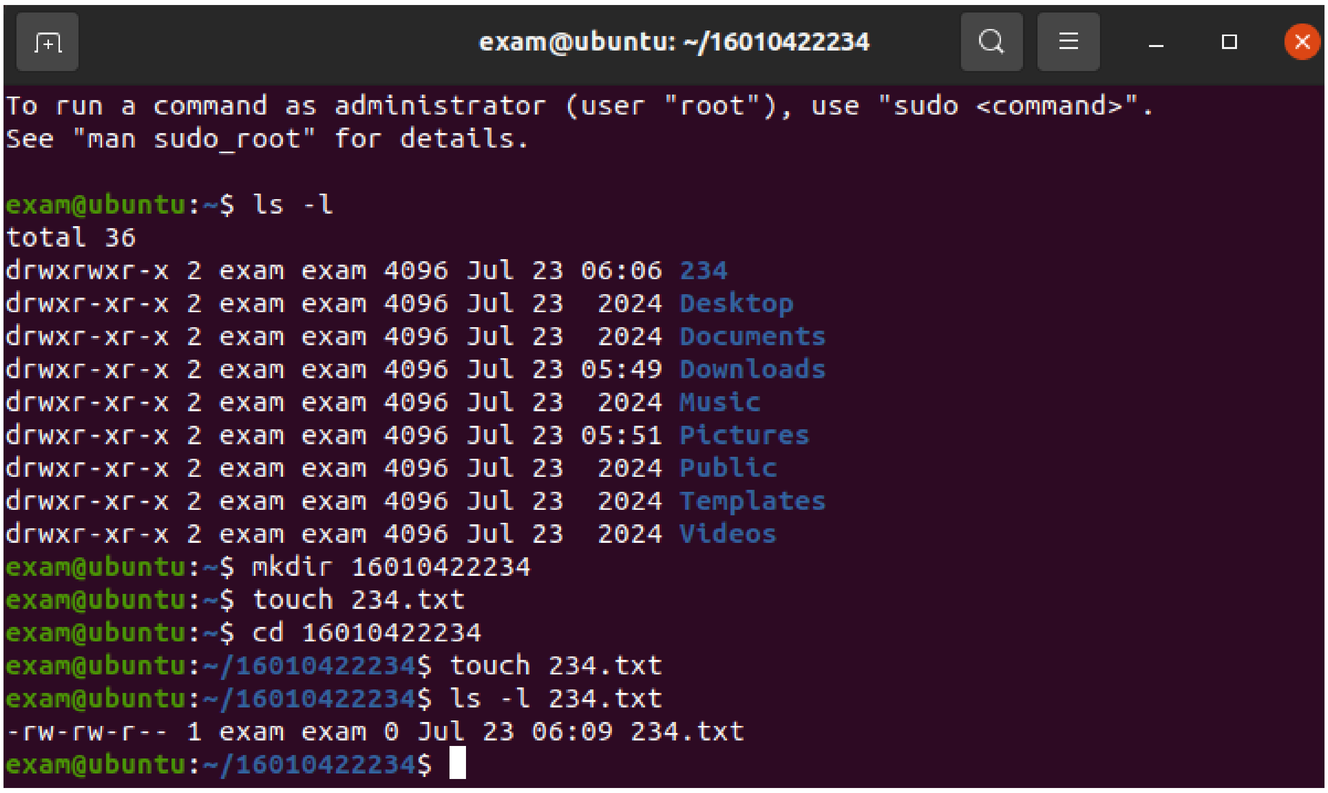
Reason: Flexibility, open-source nature, extensive development tools, and strong community support.

Results:



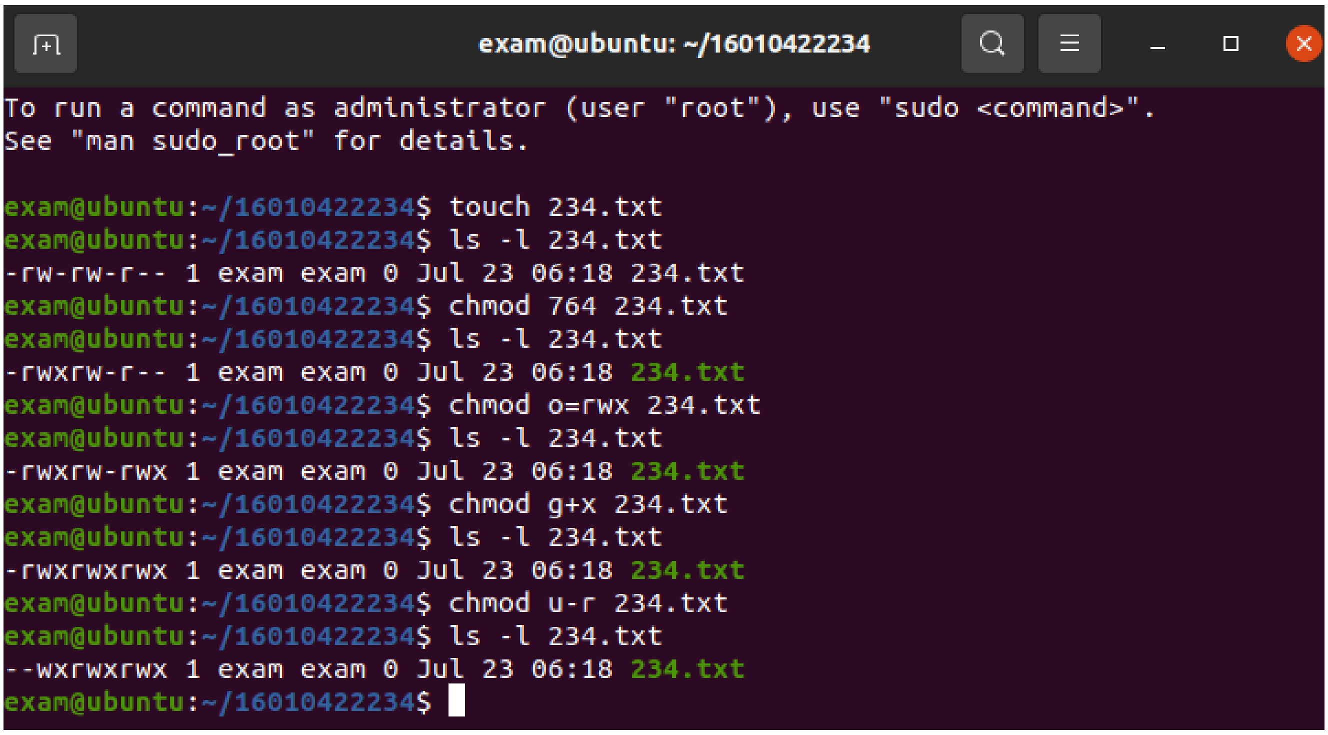
```
exam@ubuntu: ~/16010422234
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

exam@ubuntu:~$ ls -l
total 36
drwxrwxr-x 2 exam exam 4096 Jul 23 06:06 234
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Desktop
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Documents
drwxr-xr-x 2 exam exam 4096 Jul 23 05:49 Downloads
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Music
drwxr-xr-x 2 exam exam 4096 Jul 23 05:51 Pictures
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Public
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Templates
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Videos
exam@ubuntu:~$ mkdir 16010422234
exam@ubuntu:~$ touch 234.txt
exam@ubuntu:~$ cd 16010422234
exam@ubuntu:~/16010422234$ touch 234.txt
exam@ubuntu:~/16010422234$ ls -l 234.txt
-rw-rw-r-- 1 exam exam 0 Jul 23 06:09 234.txt
exam@ubuntu:~/16010422234$
```



```
exam@ubuntu: ~/16010422234
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

exam@ubuntu:~$ ls -l
total 36
drwxrwxr-x 2 exam exam 4096 Jul 23 06:06 234
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Desktop
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Documents
drwxr-xr-x 2 exam exam 4096 Jul 23 05:49 Downloads
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Music
drwxr-xr-x 2 exam exam 4096 Jul 23 05:51 Pictures
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Public
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Templates
drwxr-xr-x 2 exam exam 4096 Jul 23  2024 Videos
exam@ubuntu:~$ mkdir 16010422234
exam@ubuntu:~$ touch 234.txt
exam@ubuntu:~$ cd 16010422234
exam@ubuntu:~/16010422234$ touch 234.txt
exam@ubuntu:~/16010422234$ ls -l 234.txt
-rw-rw-r-- 1 exam exam 0 Jul 23 06:09 234.txt
exam@ubuntu:~/16010422234$
```



```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

exam@ubuntu:~/16010422234$ touch 234.txt
exam@ubuntu:~/16010422234$ ls -l 234.txt
-rw-rw-r-- 1 exam exam 0 Jul 23 06:18 234.txt
exam@ubuntu:~/16010422234$ chmod 764 234.txt
exam@ubuntu:~/16010422234$ ls -l 234.txt
-rwxrw-r-- 1 exam exam 0 Jul 23 06:18 234.txt
exam@ubuntu:~/16010422234$ chmod o=rwx 234.txt
exam@ubuntu:~/16010422234$ ls -l 234.txt
-rwxrw-rwx 1 exam exam 0 Jul 23 06:18 234.txt
exam@ubuntu:~/16010422234$ chmod g+x 234.txt
exam@ubuntu:~/16010422234$ ls -l 234.txt
-rwxrwxrwx 1 exam exam 0 Jul 23 06:18 234.txt
exam@ubuntu:~/16010422234$ chmod u-r 234.txt
exam@ubuntu:~/16010422234$ ls -l 234.txt
--wxrwxrwx 1 exam exam 0 Jul 23 06:18 234.txt
exam@ubuntu:~/16010422234$ 
```

Outcomes: CO1- Understand basic structure of modern operating system.

Conclusion:
This experiment provided an introduction to operating systems, focusing on their types, functionalities, and file permission management in Linux. Understanding file ownership and permissions is crucial for managing resources and maintaining security in a Linux environment. By differentiating various operating systems and their applications in different domains, we can choose the best-suited OS for specific needs, enhancing efficiency and performance.

References:

Books/ Journals/ Websites:

1. RichardBlumandChristineBresnahan, "LinuxCommandLine&ShellScripting", IInd Edition, Wiley, 2012.
2. Guru99. (11 August 2020). File Permissions in Linux/Unix with Example. Retrieved from https://www.guru99.com/file-Permissions.html