Title: Security Analysis of File Permissions and User Access in Linux Operating Systems

1. Experiment aim:

This experiment aims to, like, mess around with file stuff on Linux or whatever. Participants will do some random things with this chmod command, create groups without a clear purpose, and play with some logging tools just for the heck of it. The goal is to, I guess, let them do stuff with files? It's not super clear, but whatever.  
   
2. Theoretical background:

inux File Permissions: So, Linux has these permission things, and you can use this chmod command to change them. There's something about read, write, and execute, but who really cares about the details? It's just about letting people access files or not.

User Groups and File Security: User groups exist, and, like, you can put users in them. This might do something with files, maybe? We're not entirely sure, but it's a thing you can do. There might be some benefits or whatever, but it's not that important.

File Access Logging and Monitoring: There are tools like auditctl that you can use for logging. It records stuff about files, but honestly, it's not that exciting. You create log files, and I guess they show what users do, but who's really interested in that? It's just logging for the sake of logging.

3. Research:

**Ex. 1. Exploring chmod Commands Scenario:**

1. Create a new directory named "SecureFiles" in your home directory.

2. Inside "SecureFiles," create three text files: "file1.txt," "file2.txt," and "file3.txt."

3. Set the following permissions:

* "file1.txt" should be readable, writable, and executable by the owner, and readable by others.
* "file2.txt" should be readable and writable by the owner only.
* "file3.txt" should be readable, writable, and executable by the owner and the group.

# Scenario: Create a new directory named "SecureFiles" in your home directory.

mkdir ~/SecureFiles

# Scenario: Inside "SecureFiles," create three text files: "file1.txt," "file2.txt," and "file3.txt."

touch ~/SecureFiles/file1.txt ~/SecureFiles/file2.txt ~/SecureFiles/file3.txt

# Scenario: Set the following permissions:

# "file1.txt" should be readable, writable, and executable by the owner, and readable by others.

chmod u+rwx,o+r ~/SecureFiles/file1.txt

# "file2.txt" should be readable and writable by the owner only.

chmod u+rw ~/SecureFiles/file2.txt

# "file3.txt" should be readable, writable, and executable by the owner and the group.

chmod ug+rwx ~/SecureFiles/file3.txt

**Questions:**

What chmod commands did you use to set the specified permissions?

For "file1.txt": chmod u+rwx,o+r ~/SecureFiles/file1.txt

For "file2.txt": chmod u+rw ~/SecureFiles/file2.txt

For "file3.txt": chmod ug+rwx ~/SecureFiles/file3.txt

How do the permissions of each file affect user access?

For "file1.txt": The owner has read, write, and execute permissions, while others have only read permissions.

For "file2.txt": The owner has read and write permissions, while others have no permissions.

For "file3.txt": Both the owner and the group have read, write, and execute permissions, while others have no permissions.

**Ex. 2.  User Groups and File Access Scenario:**

1. Create a new user named "TestUser" on your Linux system.
2. Add "TestUser" to a group named "SecureGroup."
3. Ensure that "SecureGroup" has read and write access to all files inside the "SecureFiles" directory.
4. Log in as "TestUser" and attempt to modify "file3.txt" inside the "SecureFiles" directory.

# Scenario: Create a new user named "TestUser" on your Linux system.

sudo adduser TestUser

# Scenario: Add "TestUser" to a group named "SecureGroup."

sudo addgroup SecureGroup

sudo usermod -aG SecureGroup TestUser

# Scenario: Ensure that "SecureGroup" has read and write access to all files inside the "SecureFiles" directory.

chmod -R g+rw ~/SecureFiles

# Scenario: Log in as "TestUser" and attempt to modify "file3.txt" inside the "SecureFiles" directory.

su - TestUser

echo "Additional content" >> ~/SecureFiles/file3.txt

exit

**Questions:**

How did you add "TestUser" to the "SecureGroup"?

Used the command sudo usermod -aG SecureGroup TestUser to add the user "TestUser" to the group "SecureGroup."

Why was "TestUser" unable to modify "file3.txt"? What permissions were missing?

"TestUser" was unable to modify "file3.txt" because the write permissions for the group were not granted. The command chmod -R g+rw ~/SecureFiles was used to ensure read and write access for the group "SecureGroup" to all files inside the "SecureFiles" directory.

**Ex. 3. Logging and Monitoring File Access Scenario:**

1. Enable file access logging for the "SecureFiles" directory.
2. Create a log file to record all file access events.
3. Access "file1.txt" from another user account, and check the log file for the recorded event.

# Scenario: Enable file access logging for the "SecureFiles" directory.

sudo auditctl -w ~/SecureFiles -p rwxa

# Scenario: Create a log file to record all file access events.

sudo touch /var/log/file\_access.log

sudo chmod 666 /var/log/file\_access.log

# Scenario: Access "file1.txt" from another user account and check the log file for the recorded event.

# (Assuming another user account is named "AnotherUser")

su - AnotherUser

cat ~/SecureFiles/file1.txt

exit

cat /var/log/file\_access.log

**Questions:**

How did you enable file access logging for the directory?

Used the command sudo auditctl -w ~/SecureFiles -p rwxa to enable file access logging for the "SecureFiles" directory.

What information is logged when accessing "file1.txt" from another user account?

The log file /var/log/file\_access.log will record events associated with accessing "file1.txt," capturing details such as the username, timestamp, and the type of access (read, write, execute). The exact information can be extracted from the log file for further analysis

4. Conclusions:

So, like, after doing some random file stuff on Linux, we kind of figured out some things, I guess. Participants messed around with this chmod thing and made groups for some reason. It's not super clear why, but they did it. Oh, and there's this logging tool, auditctl, that creates log files or whatever. We checked them, and, like, users did some things with files, which was expected, I guess.

In the end, we did stuff, but it's not like it was mind-blowing or anything. File permissions, groups, and logging are just things you can do if you're into that sort of stuff. It's not clear if it really matters, but hey, we went through the motions. Maybe someone, somewhere will find it somewhat useful, or not. Whatever.