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

1. Experiment aim:

**Student’s answer:**

This experiment aims to provide participants with hands-on experience in manipulating file permissions, user groups, and file access logging within a Linux operating system. Through a series of practical exercises, the goal is to deepen their understanding of the chmod command, user group management, and access monitoring mechanisms. By exploring these facets, participants will gain practical skills essential for securing files and directories in a Linux environment.  
   
2. Theoretical background:

**Student’s answer:**

Linux File Permissions: Understanding the foundational principles of Linux file permissions is crucial. This includes recognizing the three-tiered permission system (owner, group, others) and comprehending how read, write, and execute permissions influence user access.

User Groups and File Security: Delving into user groups sheds light on collaborative access control. Participants explore the significance of creating and managing user groups, observing how group permissions augment security and streamline file access.

File Access Logging and Monitoring: Implementing file access logging introduces participants to system auditing. The theoretical background includes grasping the role of tools like auditctl in enabling logging, and the importance of creating log files to track user interactions with files.

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.

**Student’s answer:**

# 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?

**Student’s answer:**

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?

**Student’s answer:**

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.

**Student’s answer:**

# 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"?

**Student’s answer:**

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?

**Student’s answer:**

"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.

**Student’s answer:**

# 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?

**Student’s answer:**

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?

**Student’s answer:**

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:

**Student’s answer:**

This series of exercises has equipped participants with practical skills essential for managing file permissions and access control in Linux. By successfully navigating the chmod command, comprehending user groups' impact on file security, and implementing file access logging, participants have gained valuable insights. The ability to tailor permissions, utilize user groups effectively, and monitor file access lays the foundation for a secure and well-organized Linux file system. These skills are applicable in real-world scenarios, empowering participants to contribute to the creation and maintenance of secure computing environments.