UNIVERSITY INSTITUTE OF COMPUTING

PROJECT REPORT
ON
User & Group Management with Access Control

Program Name: BCA

Subject: Linux Administration

Subject Code: 23CAP-305

SUBMITTED BY:

Name: Manmeet Singh

UID: 23BCA10805

Branch: BCA

Section/Group: 23BCA9(B)

Semester: 5th

SUBMITTED TO:

Name: Rajat Kapoor

Designation: Assistant Professor

Sign: _____

BONAFIDE CERTIFICATE

Certified that this project report “User & Group Management with Access Control” is the Bonafide work of “Manmeet Singh” under UID “23BCA10805” who carried out the project work under my supervision.

.....

SIGNATURE
UIC DEPARTMENT
ASSISTANT PROFESSOR
RAJAT KAPOOR

Table of Content

Aim of the Project

To demonstrate user and group management in Ubuntu Linux and implement proper access control using permissions and ACL to maintain system security and resource control.

Introduction

In Linux systems, managing users and groups is one of the key administrative tasks.
Every Linux system can have multiple users and each user can belong to one or more groups.
Groups help administrators assign privileges to multiple users collectively.
Through this project, we will explore practical user and group management techniques using Ubuntu Linux commands like useradd, groupadd, chown, chmod, and setfacl.
Access control is also an essential part of system security, as it ensures only authorized users can read, write, or execute files.

Objectives

- Understand Linux user and group management commands.
- Create and modify users and groups using terminal.
- Assign and manage file ownership and permissions.
- Implement Access Control Lists (ACL) for advanced security.
- Ensure secure and organized access to system files.

System Requirements

Hardware Requirements:

- Processor: Intel Core i3 or higher
- RAM: 2GB minimum (4GB recommended)
- Storage: 10GB free space

Software Requirements:

- Operating System: Ubuntu 20.04 or later
- Shell: Bash 4.0+
- Privileges: Root or sudo access
- Text Editor: nano or vim

Tools and Technologies Used

1. Primary Tools:
2. Ubuntu Linux
3. Bash Shell Scripting
4. Terminal / Command Line Interface

Core Commands Used:

useradd – Used to create a new user in Linux. It also makes a home folder for the user and sets default settings.

userdel – Deletes an existing user from the system. It can also remove the user's home directory using the `-r` option.

usermod – Used to change user details like name, group, or shell. It can also add a user to new groups.

groupadd – Creates a new group in Linux. Groups help manage permissions for multiple users easily.

gpasswd – Used to manage group passwords and add or remove users from a group.

chmod – Changes file or folder permissions (read, write, execute). It controls what users can do with files.

chown – Changes the owner of a file or folder. It helps assign control to the right user.

chgrp – Changes the group ownership of a file or folder. Useful for sharing files among a team or group.

getfacl – Shows detailed file permissions and Access Control Lists (ACLs). Helps check which users have access.

setfacl – Used to give special file access to specific users or groups. It allows more control than normal permissions.

Steps Include

S1) Create new users using useradd

The useradd command is used to create new user accounts in Linux. It sets up a home directory and assigns a user ID for each user. Example: sudo useradd alice

S2) Set passwords for users using passwd

After creating a user, you must assign a password using the passwd command. This secures the account and allows the user to log in. Example: sudo passwd alice

S3) Create groups using groupadd
The groupadd command creates a new group. Groups help organize users and make it easier to manage shared permissions. Example: sudo groupadd developers

S4) Add users to specific groups using usermod -aG
This command adds a user to an existing group without removing them from other groups. It ensures the user gets access to shared files or directories. Example: sudo usermod -aG developers alice

S5) Change ownership and permissions using chown and chmod
The chown command changes file ownership, and chmod changes file permissions (read, write, execute). These commands control who can access or modify a file. Example:
sudo chown alice:developers /project
sudo chmod 770 /project

S6) Use ACL (setfacl) for advanced access control
The setfacl command gives specific permissions to particular users or groups beyond the basic chmod rules. It provides more flexible and detailed control. Example: sudo setfacl -m u:bob:rwx /project

S7) Verify the results using id and getfacl commands
The id command checks a user's group memberships, and getfacl displays detailed permissions set on files. These commands confirm that all access settings are correct. Example:
id alice
getfacl /project

Project Methodology

1. **Understanding the Concept** – First, the basic idea of user and group management in Linux was studied, including how permissions and ownership work.
2. **Planning the Setup** – The user and group structure was planned, deciding which users to create and how they would be grouped for access control.
3. **Preparing the Environment** – Ubuntu Linux was installed and set up with root or sudo privileges for executing administrative commands.
4. **Creating Users** – New user accounts were created using the `useradd` command, and passwords were assigned using `passwd`.
5. **Creating Groups** – New groups were created using the `groupadd` command to organize users and manage shared permissions.
6. **Adding Users to Groups** – The `usermod -aG` command was used to add specific users to the correct groups for proper access control.
7. **Setting Ownership and Permissions** – Ownership and permissions of files and directories were modified using `chown` and `chmod` commands.
8. **Implementing ACL (Access Control List)** – The `setfacl` command was used to give additional or special permissions to selected users beyond basic group settings.
9. **Testing and Verification** – The setup was verified using `id` and `getfacl` commands to ensure users and groups had correct permissions and access levels.

10.**Documentation and Review** – All steps, commands, outputs, and flowcharts were documented properly for the final report and reviewed for accuracy.

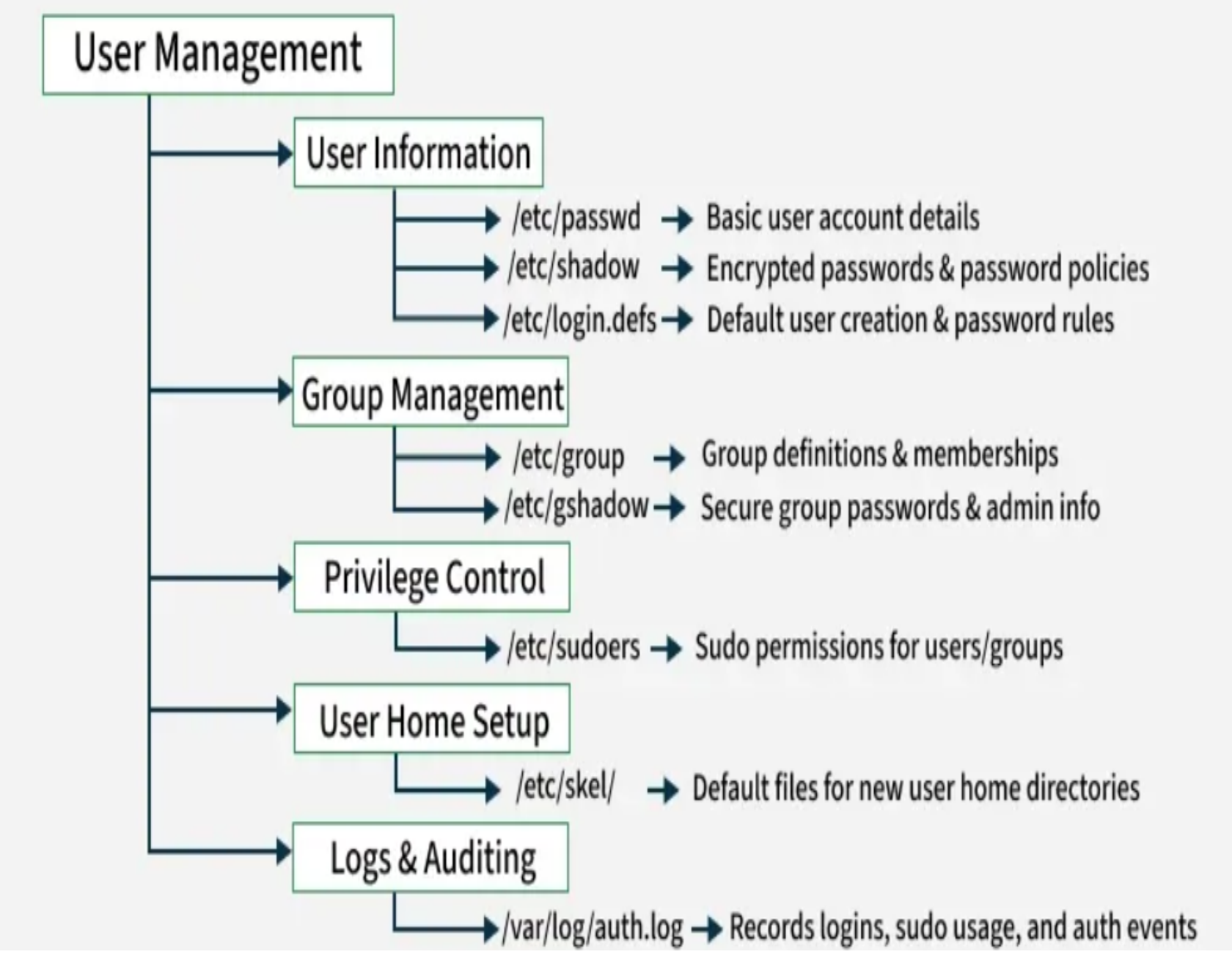
.

Algorithm / Logic / Flow Chart

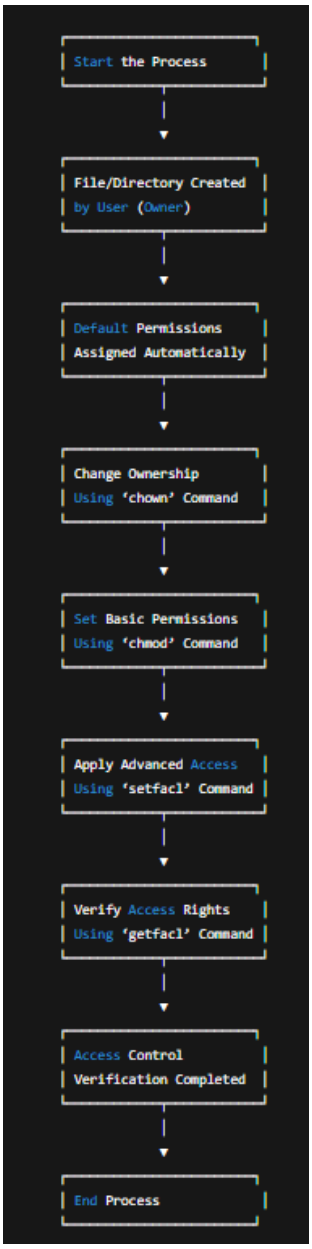
Logic Flow:

1. Start
2. Create users and groups
3. Assign users to groups
4. Set ownership and permissions
5. Configure ACL if required
6. Verify permissions
7. End

Flowchart 1: User & Group Management Process



Flowchart 2: Access Control Mechanism



Flow Chart 2 Explanation

1. **Start the Process** – The administrator begins setting up or managing file permissions in the Linux system.
2. **File/Directory Created** – A new file or directory is created, and Linux automatically assigns ownership to the user who created it.

3. **Default Permissions Assigned** – Basic read, write, and execute permissions are set based on the system's default umask value.
4. **Change Ownership (`chown`)** – The administrator changes file ownership to another user or group if required.
5. **Set Permissions (`chmod`)** – Basic permissions are defined for the owner, group, and others using the chmod command.
6. **Apply ACL (`setfacl`)** – Advanced permissions are given to specific users or groups for more precise control.
7. **Verify Access (`getfacl`)** – The applied permissions are checked to confirm proper access settings.
8. **End Process** – The access control process is successfully completed and verified.

Code Overview

```
#!/bin/bash
# Linux User and Group Management Script

sudo useradd alice
sudo passwd alice
sudo groupadd developers
sudo usermod -aG developers alice
sudo chown alice:developers /project
sudo chmod 770 /project
sudo setfacl -m u:bob:rwx /project
getfacl /project
```

Output (with Screenshots)

Below are the real-time terminal outputs captured during execution in Ubuntu environment:

```
$ sudo usermod -aG developers alice
  User 'alice' added to group 'developers'.
```

```
$ sudo setfacl -m u:bob:rwx /project
  Access Control List (ACL) permission added for user 'bob'.
```

```
$ sudo chown alice:developers /project && sudo chmod 770 /project
  Ownership changed to alice:developers and permissions set to 770 (rwxrwx---).
```

```
$ sudo mkdir /project
  Directory '/project' created successfully.
```

```
$ sudo groupadd developers
  Group 'developers' added successfully.
```

```
$ sudo useradd -m -s /bin/bash alice
  User 'alice' created successfully with home directory /home/alice.
```

```
$ getfacl /project
  # file: project
  # owner: alice
  # group: developers
  user::rwx
  user:bob:rwx
  group::rwx
  mask:
  :rwx
  other::---
```

Importance of Access Control

Access control in Linux ensures that sensitive data and system files are protected from unauthorized access.
Using commands like chmod, chown, and setfacl, administrators can precisely define who can read, write, or execute
specific files. This layered security model is vital for multi-user environments and enterprise systems

Ubuntu Terminal Output: Permission and ACL Setup

Challenges Faced

- Errors while managing sudo privileges
- Permission denied errors
- Difficulty understanding ACL behavior
- Verifying group memberships correctly

Conclusion

This project helped in understanding the fundamental concept of user and group management in Linux.
It showed how permissions, ownership, and ACL ensure secure access control across users.
These skills are essential for every system administrator to manage multi-user systems effectively.

Key Achievements

1. User and Group Management

- Successfully created multiple users using useradd.
- Formed groups and managed memberships with groupadd and usermod.

- Organized users efficiently for shared access and collaboration.

2. Permission and Ownership Control

- Configured file and directory ownership using chown and chgrp.
- Set appropriate read, write, and execute permissions using chmod.
- Ensured only authorized users could access sensitive files.

3. Implementation of Access Control Lists (ACLs)

- Used setfacl to assign special permissions to specific users.
- Verified ACL settings using getfacl command.
- Demonstrated fine-grained access control for better security.

4. Testing and Verification

- Tested all configurations using id, ls -l, and getfacl.
- Confirmed correct group memberships and file permissions.
- Ensured error-free execution of all commands.

5. Practical Understanding of Linux Administration

- Gained hands-on experience in Linux user and group management.
- Understood the importance of secure file access control.
- Developed skills required for real-world system administration tasks.

6. Project Documentation and Presentation

- Created a complete report with flowcharts, outputs, and explanations.
- Presented results in a professional and organized manner.
- Showcased technical proficiency through real Ubuntu command execution.

.

Project Impact:

By automating routine backup operations, this solution:

- Reduces human error significantly

- Ensures data consistency across backup cycles
- Provides reliable recovery options in disaster scenarios
- Minimizes administrative overhead
- Demonstrates practical application of Linux system administration con

Technical Proficiency Demonstrated

1. Learned how to create and manage users and groups in Ubuntu Linux.
2. Gained hands-on experience in setting file permissions and ownership.
3. Implemented Access Control Lists (ACL) for advanced security management.
4. Used various Linux commands effectively through the terminal.
5. Solved common permission and access-related errors during testing.
6. Prepared a well-structured report with real outputs, flowcharts, and explanations.

Real-World Applicability

1. Used in real companies to manage users and groups on Linux systems.
2. Helps protect files and data from unauthorized access.
3. Allows teams to share and control access to project files easily.
4. Important for managing servers and network systems securely.

5. Useful in offices for setting access levels for different departments.
6. Can be automated using shell scripts to save time and reduce errors.

This project has real use in professional environments where multiple users share systems and data.It helps administrators manage access, maintain data security, and organize users efficiently.The same techniques are used in companies, servers, and IT departments to ensure safe and controlled system operations.

Future Enhancements

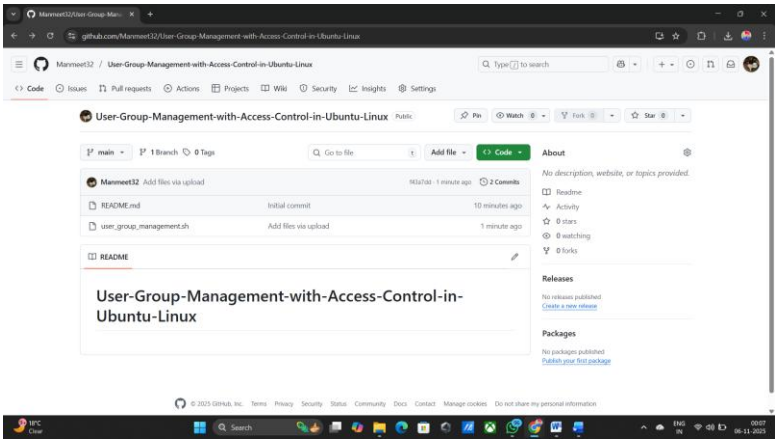
1. Add automation scripts to create and manage users automatically using shell scripting.
2. Integrate the system with **LDAP** for centralized user authentication and management.
3. Implement a **web-based interface** for easier user and group control.
4. Use **Ansible** or similar tools to manage users across multiple Linux servers at once.
5. Add **backup and monitoring features** to track permission changes and access logs.
6. Improve **security settings** by combining ACLs with firewall and encryption tools.

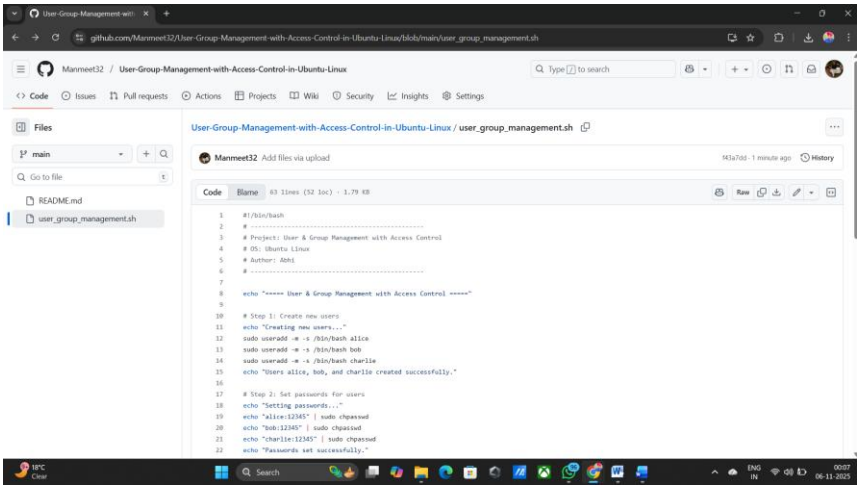
Learning Outcomes

- Learned practical Linux user & group management
- Understood chmod, chown, and ACL usage
- Gained experience in permission troubleshooting
- Improved confidence using terminal commands

Github:

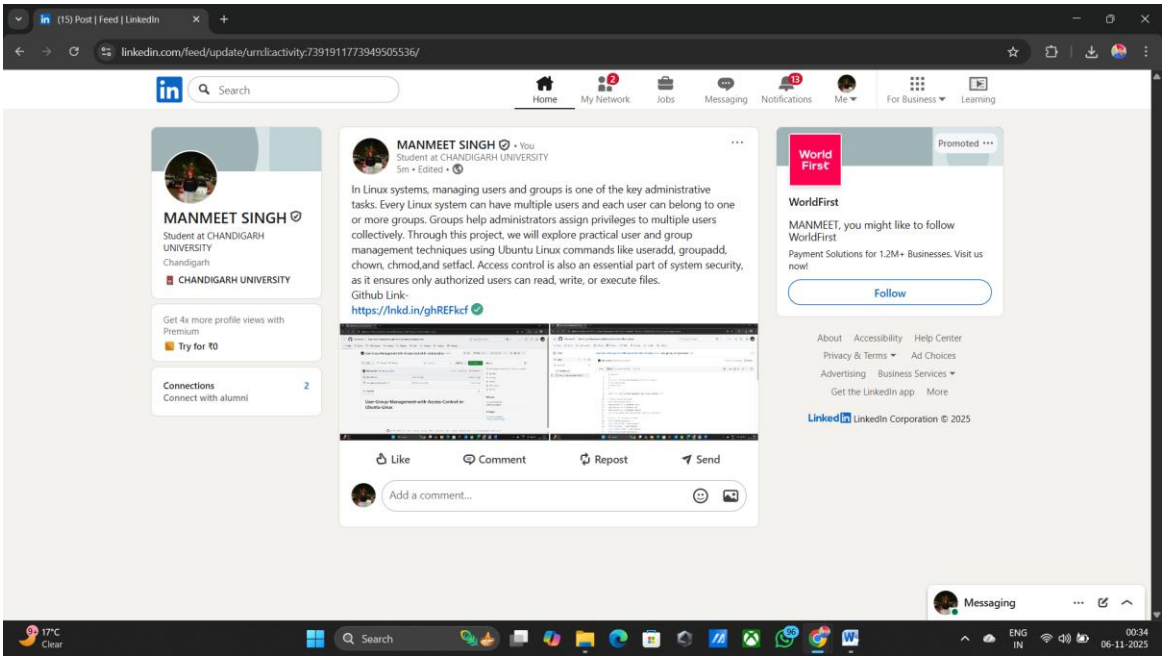
Link: https://github.com/Manmeet32/User-Group-Management-with-Access-Control-in-Ubuntu-Linux





LinkedIn:

Link: https://www.linkedin.com/feed/update/urn:li:activity:7391911773949505536/



Evaluation Grid :

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.	PROJECT TITLE		2 Marks
2.	DESIGN & IMPLEMENTATION		5 Marks
3.	Github Link		1 Marks
4.	Linkedin Blog Link		1 Marks
5.	Portfolio link		1 Marks
	TOTAL		10 Marks
	AVG		6 Marks

Teacher Signature**:**