Ganpat University | Institute of Computer Technology
॥ विद्यया समाजोत्कर्षः ॥

Name: Tushar Panchal

En.No: 21162101014

Sub: ITIM (IT Infrastructure & Management)

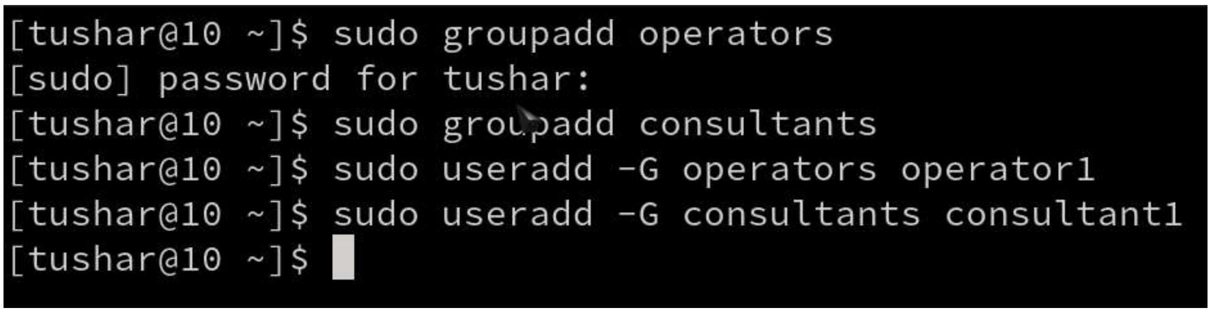
Branch: CBA

Batch:61

---------------------------PRACTICAL 02---------------------------

1. **Operators and Consultants are members of an IT support company. They need to start sharing information. servera contains a properly configured share directory located at /shares/content that hosts files. Currently, only members of the operators group have access to this directory, but members of the consultants group need full access to this directory. The consultant1 user is a member of the consultants group but has caused problems on many occasions, so this user should not have access to the directory.**
   Your task is to add appropriate ACL entries to the directory and its contents so that members of the consultants group have full access, but deny the consultant1 user any access. Make sure that future files and directories stored in /shares/content get appropriate ACL entries applied :

» Creating Users & Groups required :

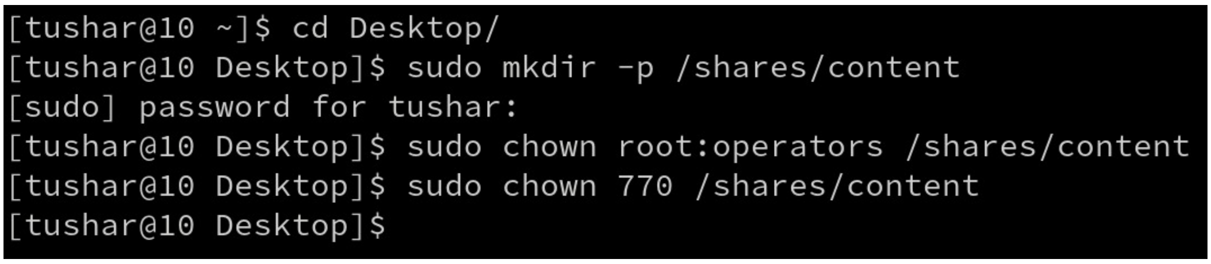


```
[tushar@10 ~]$ sudo groupadd operators
[sudo] password for tushar:
[tushar@10 ~]$ sudo groupadd consultants
[tushar@10 ~]$ sudo useradd -G operators operator1
[tushar@10 ~]$ sudo useradd -G consultants consultant1
[tushar@10 ~]$
```

Let's break down the commands we've executed:

1. **sudo groupadd operators**:
    - **sudo**: Stands for "superuser do" and is used to execute commands with superuser (administrative) privileges.
    - **groupadd**: Command to create a new group on the system.
    - **operators**: The name of the group being created. In this case, it's named "operators."
2. **sudo groupadd consultants**:
    - Similar to the first command, this creates another group named "consultants" using the **groupadd** command.
3. **sudo useradd -G operators operator1**:
    - **useradd**: Command to add a new user to the system.
    - **-G operators**: Specifies that the user should be added to the "operators" group.
    - **operator1**: The username of the user being created. In this case, it's "operator1."
4. **sudo useradd -G consultants consultant1**:
    - Similar to the previous user creation command, this adds a new user named "consultant1" to the "consultants" group.

Creating required folders with necessary permissions :

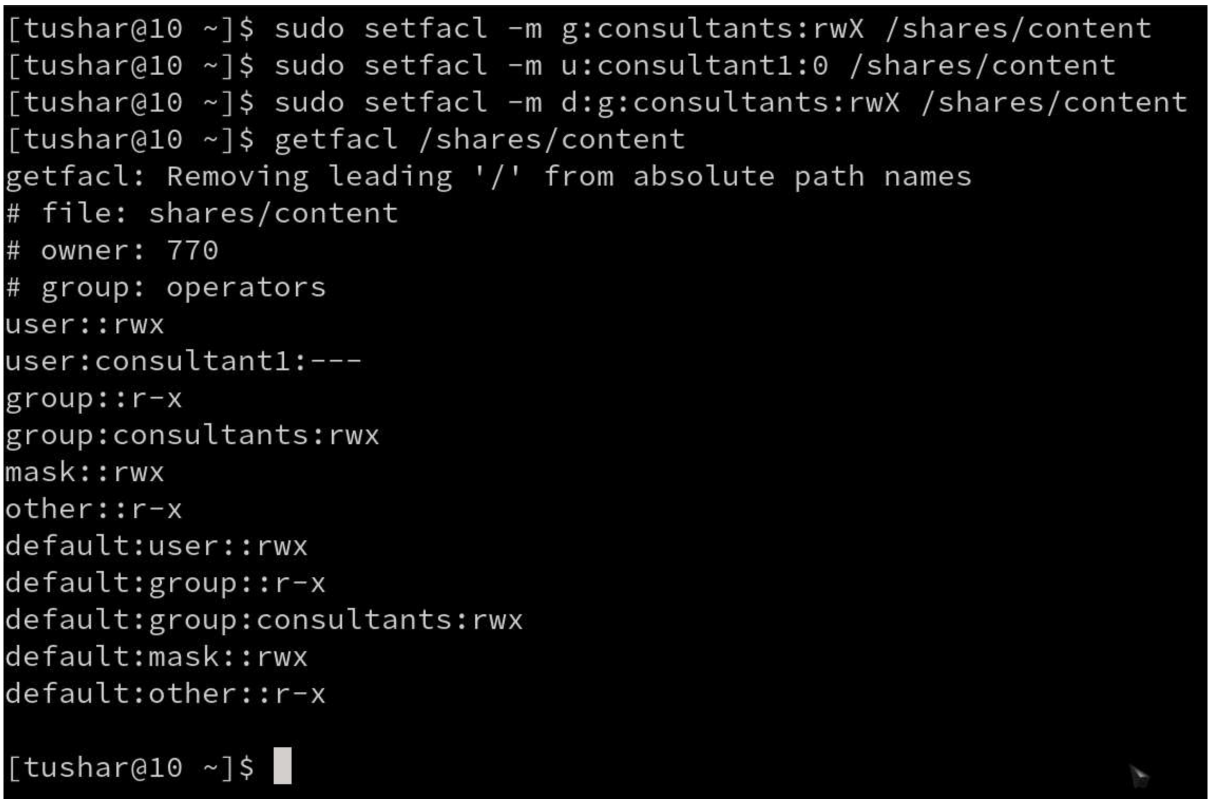


```
[tushar@10 ~]$ cd Desktop/
[tushar@10 Desktop]$ sudo mkdir -p /shares/content
[sudo] password for tushar:
[tushar@10 Desktop]$ sudo chown root:operators /shares/content
[tushar@10 Desktop]$ sudo chown 770 /shares/content
[tushar@10 Desktop]$
```

Let's break down the commands we've executed:

1. **cd Desktop/**: This command changes the current working directory to the "Desktop" directory.
2. **sudo mkdir -p /shares/content**: This command creates the directory path **/shares/content** with the **-p** option, which ensures that if any of the parent directories don't exist, they will be created as well. The command creates the necessary directory structure for your scenario.
3. **sudo chown root:operators /shares/content**: This command changes the ownership of the **/shares/content** directory to the user **root** and the group **operators**. The **chown** command stands for "change owner."
4. **sudo chmod 770 /shares/content**: This command sets the permissions for the **/shares/content** directory. The **chmod** command stands for "change mode." The three digits (770) represent the permission settings for the owner, group, and others. In this case:
   - The owner (**root**) has read, write, and execute permissions (7).
   - The group (**operators**) has read, write, and execute permissions (7).
   - Others have no permissions (0).

Setting File Permissions via setfacl command :



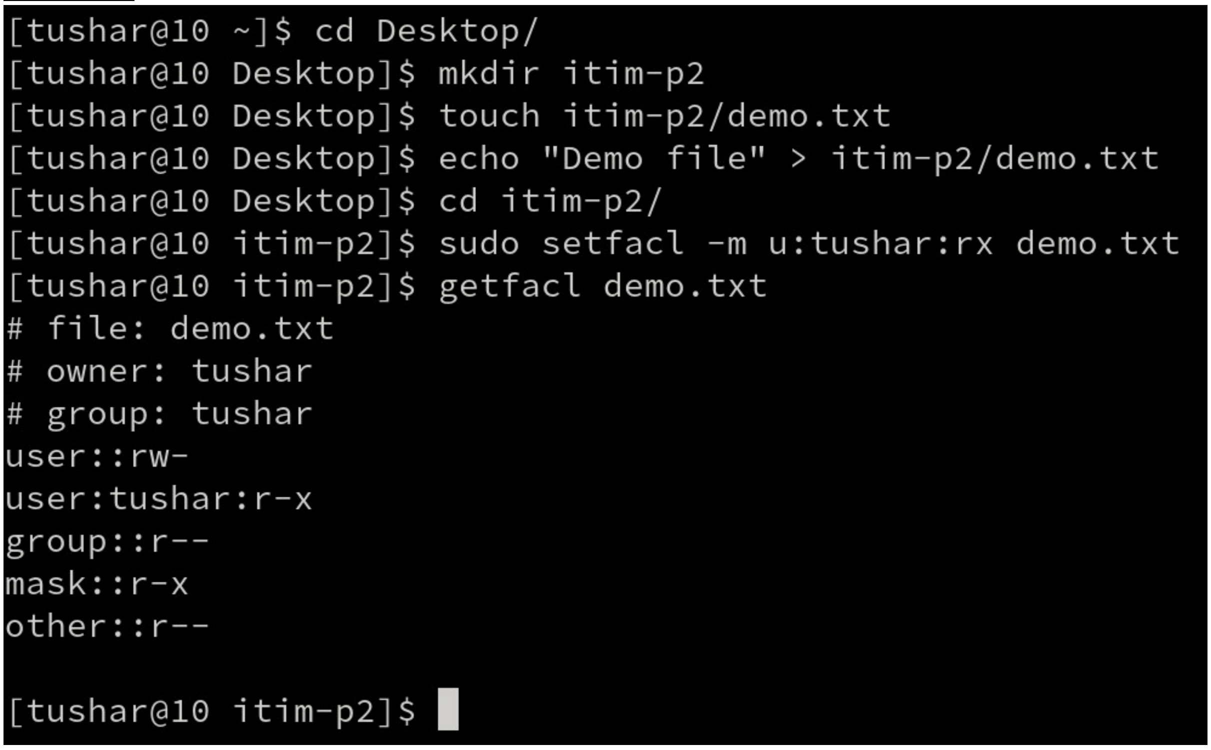
```
[tushar@10 ~]$ sudo setfacl -m g:consultants:rwX /shares/content
[tushar@10 ~]$ sudo setfacl -m u:consultant1:0 /shares/content
[tushar@10 ~]$ sudo setfacl -m d:g:consultants:rwX /shares/content
[tushar@10 ~]$ getfacl /shares/content
getfacl: Removing leading '/' from absolute path names
# file: shares/content
# owner: 770
# group: operators
user::rwx
user:consultant1:---
group::r-x
group:consultants:rwx
mask::rwx
other::r-x
default:user::rwx
default:group::r-x
default:group:consultants:rwx
default:mask::rwx
default:other::r-x

[tushar@10 ~]$ █
```

Let's break down the commands we've executed:

1. **sudo setfacl -m g:consultants:rwX /shares/content**:
   - **-m**: Modify the ACL (Access Control List).
   - **g:consultants:rwX**: Grant read (r), write (w), and execute (X) permissions to the **consultants** group.
   - **/shares/content**: The target directory for which you are modifying the ACL.
2. **sudo setfacl -m u:consultant1:0 /shares/content**:
   - **-m**: Modify the ACL.
   - **u:consultant1:0**: Deny (0) all permissions to the user **consultant1**.
   - **/shares/content**: The target directory for which you are modifying the ACL.

3. **sudo setfacl -m d:g:consultants:rwX /shares/content**:
    - **-m**: Modify the default ACL (for future files and directories).
    - **d:g:consultants:rwX**: Set default read, write, and execute permissions for the **consultants** group.
    - **/shares/content**: The target directory for which you are modifying the default ACL.
4. **getfacl /shares/content**:
    - **getfacl**: Get the ACL (Access Control List) information for the specified file or directory.
    - **/shares/content**: The target directory for which you want to retrieve the ACL information.

2. **<u>Create a txt file in a folder and allow only a specific user the read and execute access. Ensure that the user is not able to modify the content of the file :</u>**



```
[tushar@10 ~]$ cd Desktop/
[tushar@10 Desktop]$ mkdir itim-p2
[tushar@10 Desktop]$ touch itim-p2/demo.txt
[tushar@10 Desktop]$ echo "Demo file" > itim-p2/demo.txt
[tushar@10 Desktop]$ cd itim-p2/
[tushar@10 itim-p2]$ sudo setfacl -m u:tushar:rx demo.txt
[tushar@10 itim-p2]$ getfacl demo.txt
# file: demo.txt
# owner: tushar
# group: tushar
user::rw-
user:tushar:r-x
group::r--
mask::r-x
other::r--

[tushar@10 itim-p2]$ 
```

Let's break down the commands we've executed:

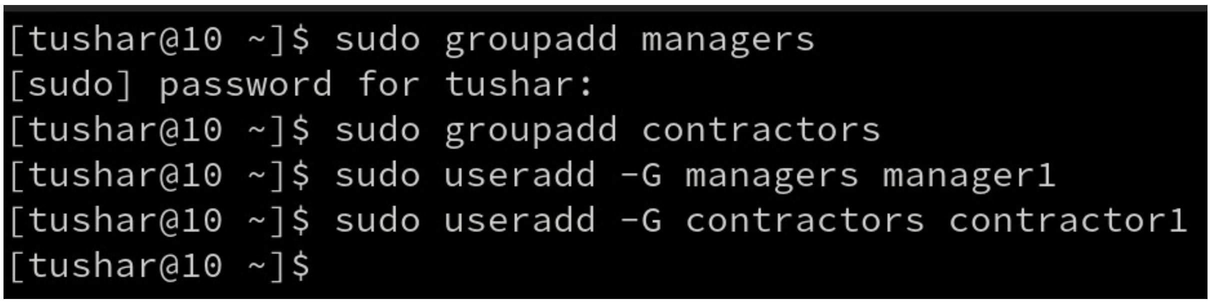
1. **cd Desktop/**: Change directory to the Desktop directory.
2. **mkdir itim-p2**: Create a new directory named **itim-p2** on the Desktop.
3. **touch itim-p2/demo.txt**: Create an empty file named **demo.txt** inside the **itim-p2** directory.
4. **echo "Demo file" > itim-p2/demo.txt**: Write the text "Demo file" into the **demo.txt** file.
5. **cd itim-p2/**: Change directory to the **itim-p2** directory.
6. **sudo setfacl -m u:tushar:rx demo.txt**: Set file access control list (ACL) for the user **tushar** on the **demo.txt** file, granting read and execute permissions.
7. **getfacl demo.txt**: Display the ACLs (Access Control Lists) for the **demo.txt** file.

Explanation of the ACLs displayed by **getfacl**:

- **user::rw-**: The owner (**tushar**) has read and write permissions.
- **user:tushar:r-x**: The specific user **tushar** has read and execute permissions.
- **group::r--**: The group (**tushar**) has read-only permissions.
- **mask::r-x**: The effective permissions mask based on the union of all entries. It is set to read and execute.
- **other::r--**: Others have read-only permissions.

3. **A stock finance agency is setting up a collaborative share directory to hold case files, which members of the managers group will have read and write permissions on. The co-founder of the agency, manager1, has decided that members of the contractors group should also be able to read and write to the share directory. However, manager1 does not trust the contractor3 user (a member of the contractors group), and as such, contractor3 should have access to the directory restricted to read-only. manager1 has created the users and groups, and has started the process of setting up the share directory, copying in some templates files. Because manager1 has been too busy, it falls to you to finish the job. Your task is to complete the setup of the share directory. The directory and all of its contents should be owned by the managers group, with the files updated to read and write for the owner and group (managers). Other users should have no permissions. You also need to provide read and write permissions for the contractors group, with the exception of contractor3, who only gets read permissions. Make sure your setup applies to existing and future files:**

Creating Users & Groups required :



```
[tushar@10 ~]$ sudo groupadd managers
[sudo] password for tushar:
[tushar@10 ~]$ sudo groupadd contractors
[tushar@10 ~]$ sudo useradd -G managers manager1
[tushar@10 ~]$ sudo useradd -G contractors contractor1
[tushar@10 ~]$
```

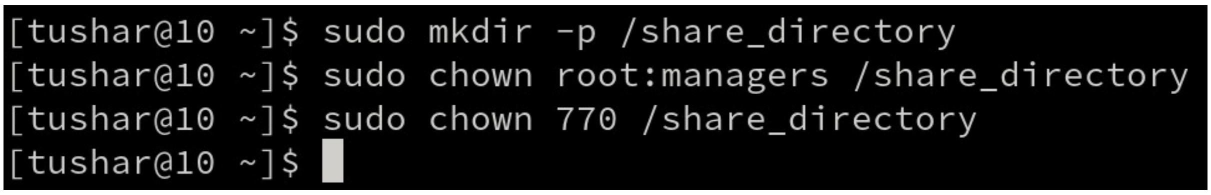
Let's break down the commands we've executed:

1. **sudo groupadd managers**:
   - This command creates a new group named "managers" using the **groupadd** command.
   - The **sudo** command is used to execute the following command with elevated privileges, typically requiring the user's password.
2. **sudo groupadd contractors**:

- Similar to the first command, this creates a new group named "contractors" using the **groupadd** command.

3. **sudo useradd -G managers manager1**:
    - The **useradd** command is used to add a new user to the system.
    - The **-G managers** option specifies that the user should be added to the "managers" group.
    - **manager1** is the username that is being added.
4. **sudo useradd -G contractors contractor1**:
    - Similar to the previous user addition, this command adds a user named **contractor1** to the "contractors" group.

» **<u>Creating directory required with necessary permissions :</u>**



```
[tushar@10 ~]$ sudo mkdir -p /share_directory
[tushar@10 ~]$ sudo chown root:managers /share_directory
[tushar@10 ~]$ sudo chown 770 /share_directory
[tushar@10 ~]$ 
```

Let's break down the commands we've executed:

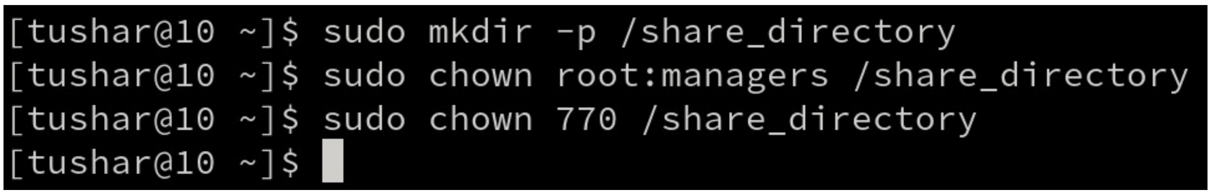
1. **sudo mkdir -p /share_directory**: This command creates a directory named **share_directory** in the root directory (**/**). The **-p** option ensures that the command creates the directory and any necessary parent directories that do not exist.
2. **sudo chown root:managers /share_directory**: This command changes the ownership of the **share_directory** to the user **root** and the group **managers**. The **chown** command stands for "change owner." It's common to set

the owner to **root** for administrative purposes and assign a specific group, in this case, **managers**.

3. **sudo chmod 770 /share_directory**: This command sets the permissions of the **share_directory**. The **chmod** command stands for "change mode." The **770** is a numeric representation of file permissions:
    - The first digit (**7**) represents the owner's permissions, which are **rwx** (read, write, and execute).
    - The second digit (**7**) represents the group's permissions, which are also **rwx**.
    - The third digit (**0**) represents the permissions for others, which are none (**---**).

So, with **770**, the owner and the group have full read, write, and execute permissions, while others have no permissions.

» Creating directory required with necessary permissions :



```
[tushar@10 ~]$ sudo mkdir -p /share_directory
[tushar@10 ~]$ sudo chown root:managers /share_directory
[tushar@10 ~]$ sudo chown 770 /share_directory
[tushar@10 ~]$ 
```

Let's break down the commands we've executed:

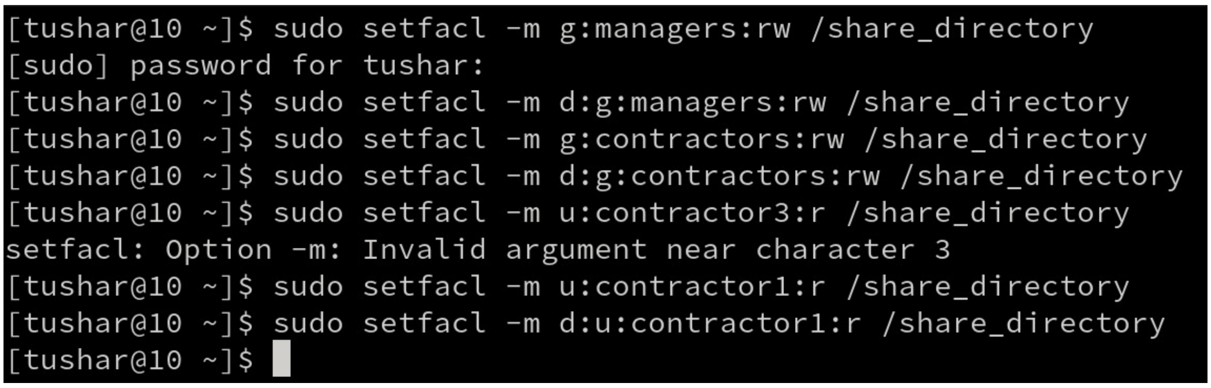
1. **sudo mkdir -p /share_directory**: This command creates a directory named **share_directory** in the root directory (**/**). The **-p** option ensures that the command creates the directory and any necessary parent directories that do not exist.
2. **sudo chown root:managers /share_directory**: This command changes the ownership of the **share_directory** to the user **root** and the group **managers**. The **chown**

command stands for "change owner." It's common to set the owner to **root** for administrative purposes and assign a specific group, in this case, **managers**.

3. **sudo chmod 770 /share_directory**: This command sets the permissions of the **share_directory**. The **chmod** command stands for "change mode." The **770** is a numeric representation of file permissions:
   - The first digit (**7**) represents the owner's permissions, which are **rwx** (read, write, and execute).
   - The second digit (**7**) represents the group's permissions, which are also **rwx**.
   - The third digit (**0**) represents the permissions for others, which are none (**---**).

So, with **770**, the owner and the group have full read, write, and execute permissions, while others have no permissions.

Setting & Modifying permissions via ACL :



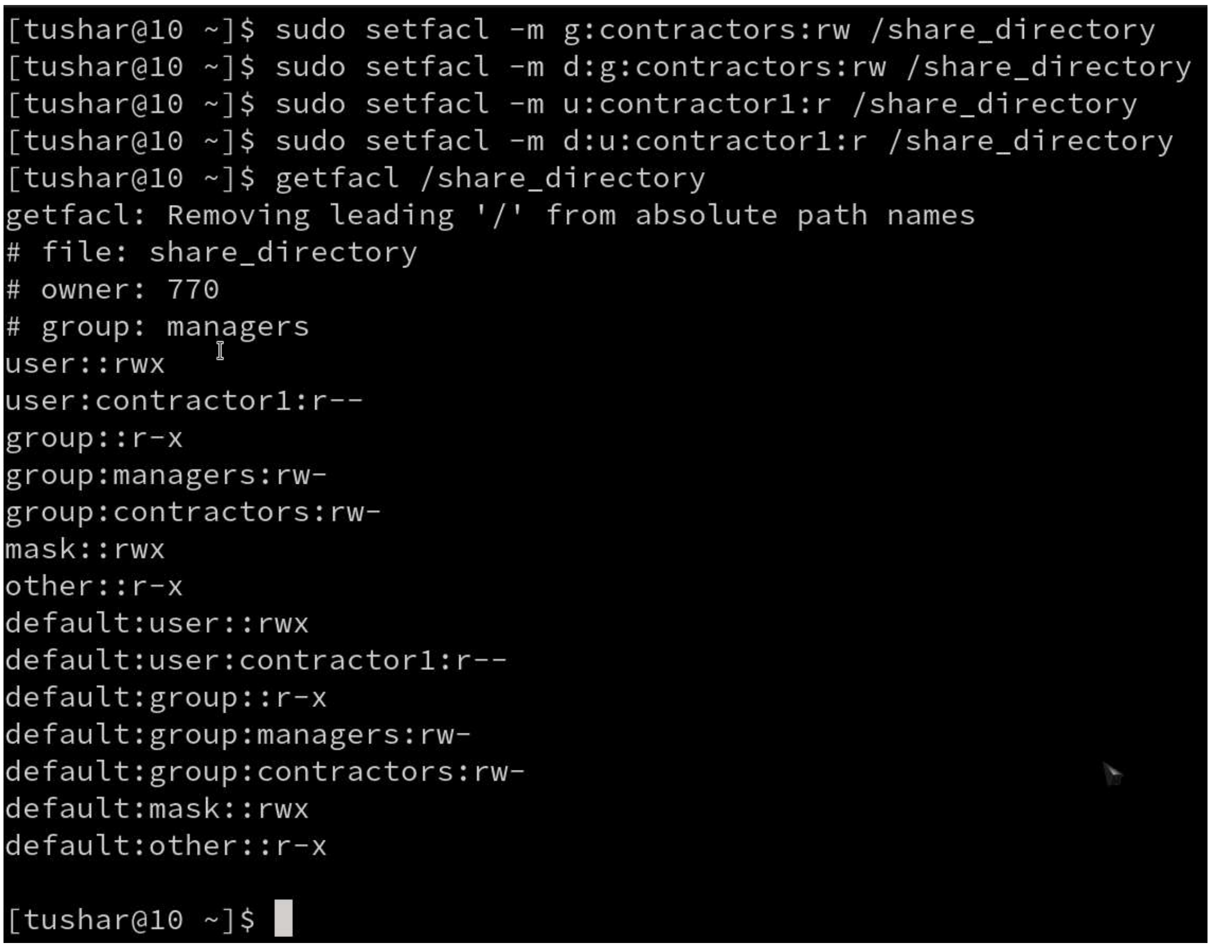
```
[tushar@10 ~]$ sudo setfacl -m g:managers:rw /share_directory
[sudo] password for tushar:
[tushar@10 ~]$ sudo setfacl -m d:g:managers:rw /share_directory
[tushar@10 ~]$ sudo setfacl -m g:contractors:rw /share_directory
[tushar@10 ~]$ sudo setfacl -m d:g:contractors:rw /share_directory
[tushar@10 ~]$ sudo setfacl -m u:contractor3:r /share_directory
setfacl: Option -m: Invalid argument near character 3
[tushar@10 ~]$ sudo setfacl -m u:contractor1:r /share_directory
[tushar@10 ~]$ sudo setfacl -m d:u:contractor1:r /share_directory
[tushar@10 ~]$ 
```

Let's break down the commands we've executed:

1. **sudo setfacl -m g:managers:rw /share_directory**
   - This command grants read and write (**rw**) permissions to the **managers** group for the **share_directory**.

2. **sudo setfacl -m d:g:managers:rw /share_directory**
    - This command sets the default ACL for the **managers** group, ensuring that any new files or directories created inside **share_directory** inherit the read and write permissions for the **managers** group.
3. **sudo setfacl -m g:contractors:rw /share_directory**
    - This command grants read and write (**rw**) permissions to the **contractors** group for the **share_directory**.
4. **sudo setfacl -m d:g:contractors:rw /share_directory**
    - This command sets the default ACL for the **contractors** group, ensuring that any new files or directories created inside **share_directory** inherit the read and write permissions for the **contractors** group.
5. **sudo setfacl -m u:contractor1:r /share_directory**
    - This command grants read-only (**r**) access to the specific user **contractor1** for the **share_directory**.
6. **sudo setfacl -m d:u:contractor1:r /share_directory**
    - This command sets the default ACL for the specific user **contractor1**, ensuring that any new files or directories created inside **share_directory** inherit the read-only access for **contractor1**.

Checking the permissions via ACL entries :



```
[tushar@10 ~]$ sudo setfacl -m g:contractors:rw /share_directory
[tushar@10 ~]$ sudo setfacl -m d:g:contractors:rw /share_directory
[tushar@10 ~]$ sudo setfacl -m u:contractor1:r /share_directory
[tushar@10 ~]$ sudo setfacl -m d:u:contractor1:r /share_directory
[tushar@10 ~]$ getfacl /share_directory
getfacl: Removing leading '/' from absolute path names
# file: share_directory
# owner: 770
# group: managers
user::rwx
user:contractor1:r--
group::r-x
group:managers:rw-
group:contractors:rw-
mask::rwx
other::r-x
default:user::rwx
default:user:contractor1:r--
default:group::r-x
default:group:managers:rw-
default:group:contractors:rw-
default:mask::rwx
default:other::r-x

[tushar@10 ~]$
```

Let's break down the commands we've executed:

getfacl /share_directory: This command retrieves and displays the ACL (Access Control List) for the **/share_directory**.

- The **user::rwx** entry indicates the owner has read, write, and execute permissions.
- The **user:contractor1:r--** entry indicates that **contractor1** has only read (**r**) permissions.
- The **group::r-x** entry indicates the default group permissions.
- The **group: managers:rw-** entry indicates that the **managers** group has read and write permissions.

- The **group:contractors:rw-** entry indicates that the **contractors** group has read and write permissions.
- The **mask::rwx** entry represents the maximum permissions that can be set for any entity in the ACL.
- The **other::r-x** entry indicates the default permissions for others.
- The **default** entries set default ACLs for future files and directories.