Experiment 3

Aim: Study of file management and user management commands

Theory

A user is an entity, in a Linux operating system, that can manipulate files and perform several other operations. Each user is assigned an ID that is unique for each user in the operating system. In this post, we will learn about users and commands which are used to get information about the users. After installation of the operating system, the ID 0 is assigned to the root user and the IDs 1 to 999 (both inclusive) are assigned to the system users and hence the ids for local user begins from 1000 onwards. In a single directory, we can create 60,000 users.

The general format for the Linux command-line follows the verb > object model. So first is the command itself that indicates the operation the user wants to perform (e.g. copy, rename, delete, etc.). The command is then followed by what would be considered in grammatical terms the direct object (e.g. a file named August\_expenses that we need to copy or a directory called Expense Reports that needs to be removed).

User management includes everything from creating a user to deleting a user on your system. User management can be done in three ways on a Linux system.

Graphical tools are easy and suitable for new users, as it makes sure you'll not run into any trouble.

Command line tools includes commands like useradd, userdel, passwd, etc. These are mostly used by the server administrators.

Third and very rare tool is to edit the local configuration files directly using vi.

Every Linux system have three types of owner:

User: A user is the one who created the file. By default, whosoever, creates the file becomes the owner of the file. A user can create, delete, or modify the file.

Group: A group can contain multiple users. All the users belonging to a group have same access permission for a file.

Other: Any one who has access to the file other than user and group comes in the category of other. Other has neither created the file nor is a group member.

3A

Aim

To study Unix file system (tree structure), file and directory permission single and multiuser envieonment.

Theory

UNIX files Tree structure

The Unix file system is a methodology for logically organizing and storing large quantities of data such that the system is easy to manage. A file can be informally defined as a collection of (typically related) data, which can be logically viewed as a stream of bytes (i.e. characters). A file is the smallest unit of storage in the Unix file system.

By contrast, a file system consists of files, relationships to other files, as well as the attributes of each file. File attributes are information relating to the file, but do not include the data contained within a file. File attributes for a generic operating system might include (but are not limited to):

* a file type (i.e. what kind of data is in the file)
* a file name (which may or may not include an extension)
* a physical file size
* a file owner
* file protection/privacy capability
* file time stamp (time and date created/modified)

Additionally, file systems provide tools which allow the manipulation of files, provide a logical organization as well as provide services which map the logical organization of files to physical devices.

From the beginners perspective, the Unix file system is essentially composed of files and directories. Directories are special files that may contain other files.

The Unix file system has a hierarchical (or tree-like) structure with its highest level directory called root (denoted by /, pronounced slash). Immediately below the root level directory are several subdirectories, most of which contain system files. Below this can exist system files, application files, and/or user data files. Similar to the concept of the process parent-child relationship, all files on a Unix system are related to one another. That is, files also have a parent-child existence. Thus, all files (except one) share a common parental link, the top-most file (i.e. /) being the exception.

Below is a diagram (slice) of a "typical" Unix file system. As you can see, the top-most directory is / (slash), with the directories directly beneath being system directories. Note that as Unix implementaions and vendors vary, so will this file system hierarchy. However, the organization of most file systems is similar.

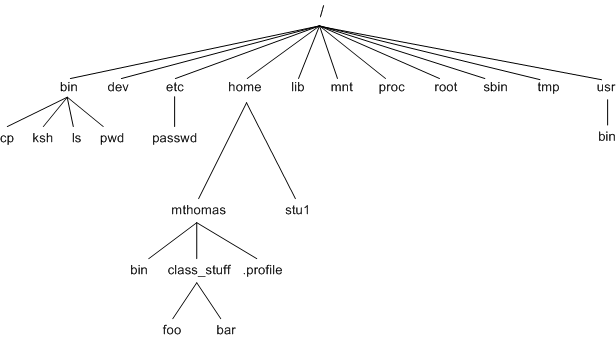


Fig 3a.1

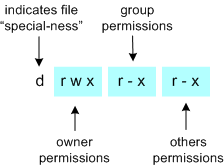
File manipulation in Linux

The first (leftmost) character indicates the "type" of the file. Another way to describe this is whether the file has any special attributes associated with it. If it is an ordinary file (i.e. no special attributes), it will have a dash in this first position. If it is a directory file, it will have the letter d in this position. Or, if it is a link to another file it will have the letter l (ell) in this first position. You can see examples of an ordinary file and a directory in the ls -l output above. Other special attributes exist but do not merit discussion here.

The next 9 characters are arranged in 3 groups of 3 characters each; that is 3 characters to describe the permissions for the owner of the file, 3 characters to describe the permissions for the group and 3 characters for all other users permissions. The 3 characters indicate whether the particular user has read (denoted by r), write (denoted by w), or execute (denoted by x) permissions on that particular file. Thus in the diagram above, the owner of the file has all available permissions (indicated by rwx), a user belonging to this group has read and execute (indicated by r-x) permissions, and everyone else also has read and execute permissions. Observe if a user does not have a particular permission, a dash will appear instead of the corresponding letter.

Changing the permission of an existing file is accomplished using the chmod command, that is to change file protection modes. Changing the permission for a file (or files) can only be done by the owner of the file (or the root user). The usage of this command (using octal mode 1) is

where ijk represent 3 octal numbers (0-7); where i selects the user permissions, j selects the group permissions and k selects the permissions for all others. To disable all (read, write and execute) permissions for any group, you set the respective i, j or k value to 0. To set read permission for any grouping (i.e. user, group or other) you add the value of 4 to the respective i, j, or k value, to set write permission, you add 2 and to set execute permission, you add 1. The values of 4, 2, and 1 are derived from the first three powers of two, i.e. 22, 21, 20 respectively. Thus to set the user permissions to rwx, you set i to 7 (4 for read + 2 for write + 1 for execute), to set the group and other permissions to r-x, you set j to 5 (4 + 1) and likewise for k. The command to select this would then look as follows:



Example of commands

* **chmod *ijk* file(s)**
* **chmod 540 file\_spec(s)**
* **chmod 755 file\_spec(s)**

To illustrate with another example, if the owner of a file wanted to set read and execute for the user permissions, read only for group permissions and disable all permissions for all others

We can see from these two examples that to allow all permissions, you use the maximum value of 7; to prohibit all permissions, you use the minimum value of 0. As a user, you will not have to remember every combination of 4, 2, and 1; typically you will standard combinations such as 755, 644, 700, etc.

* The permissions of read, write and execute take on a slightly different meaning with respect to directory files. Thus if a file is a directory:
* read permission determines if a user can view the files contained in a directory, i.e. list the files in it
* write permission determines if a user can create new files or delete files in the directory. This allows a user with write permission to a directory to have the ability to delete files in the directory even if they don't have write permissions for the file (see Note: on rm below). Watch out for this gotcha!
* execute permission determines if the user can move (i.e. cd) into the directory

Where chmod sets permissions on existing files, setting default file permissions set upon file creation is done using the umask command. This is somewhat similar to chmod, but the permissions are set on newly created files only. While similar to chmod, rather than setting which permissions are set, umask sets which permissions are "removed" or unset.

Conclusion : In the above experiment we studied about the unix file management system and how to modify file permission through chmod command.

3B

Aim :Execution of file system Management commands

Theory:

1.ls

ls is one of the basic commands that any Linux user should know.

The ls command lists files and directories within the file system, and shows detailed information about them. It is a part of the GNU core utilities package which is installed on all Linux distributions.

This article will show you how to use the ls command through practical examples and detailed explanations of the most common ls options.

**ls –a**

In Linux, hidden files start with . (dot) symbol and they are not visible in the regular directory. The (ls -a) command will enlist the whole list of the current directory including the hidden files.

**ls -l** It will show the list in a long list format.

OUTPUT

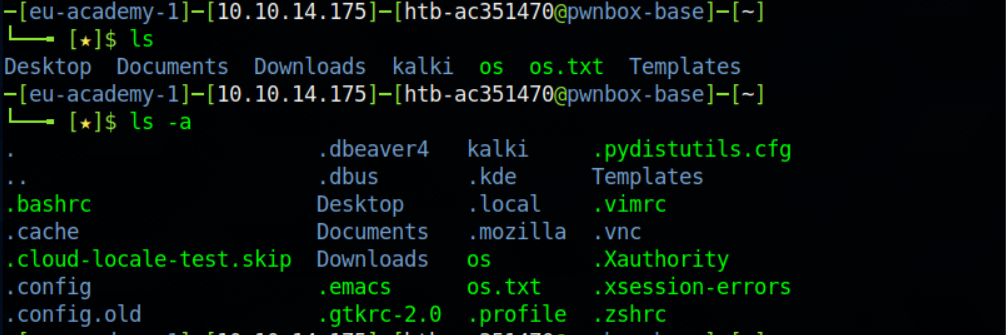


Fig 3.b.1

2.Less

Less command is a Linux utility that can be used to read the contents of a text file one page(one screen) at a time. It has faster access because if file is large it doesn’t access the complete file, but accesses it page by page.

For example, if it’s a large file and you are reading it using any text editor, then the complete file will be loaded to main memory. The less command doesn’t load the entire file, but loads it part by part which makes it faster.



Fig 3b.2



Fig 3b.3

3. Tail

It is the complementary of head command.The tail command, as the name implies, print the last N number of data of the given input. By default it prints the last 10 lines of the specified files. If more than one file name is provided then data from each file is precedes by its file name. 

4.Head

It is the complementary of Tail command. The head command, as the name implies, print the top N number of data of the given input. By default, it prints the first 10 lines of the specified files. If more than one file name is provided then data from each file is preceded by its file name.



Fig 3.b.6

5.Cat

Cat(concatenate) command is very frequently used in Linux. It reads data from the file and gives their content as output. It helps us to create, view, concatenate files. So let us see some frequently used cat commands.

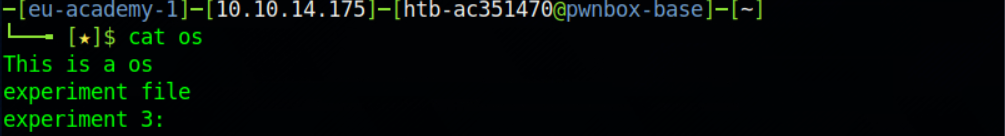


Fig 3.b.7

6.Echo

Echo command in linux is used to display line of text/string that are passed as an argument . This is a built in command that is mostly used in shell scripts and batch files to output status text to the screen or a file.

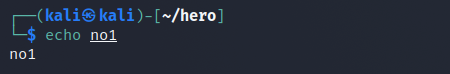


Fig 3.b.8

7.exit

exit command in linux is used to exit the shell where it is currently running. It takes one more parameter as [N] and exits the shell with a return of status N. If n is not provided, then it simply returns the status of last command that is executed.

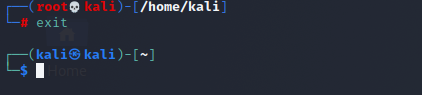


Fig 3.b.9

8.Find

The find command in UNIX is a command line utility for walking a file hierarchy. It can be used to find files and directories and perform subsequent operations on them. It supports searching by file, folder, name, creation date, modification date, owner and permissions. By using the ‘-exec’ other UNIX commands can be executed on files or folders found.

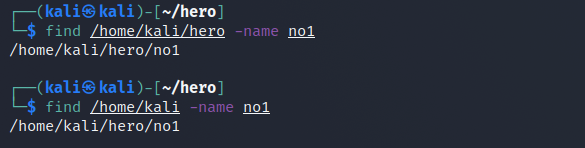
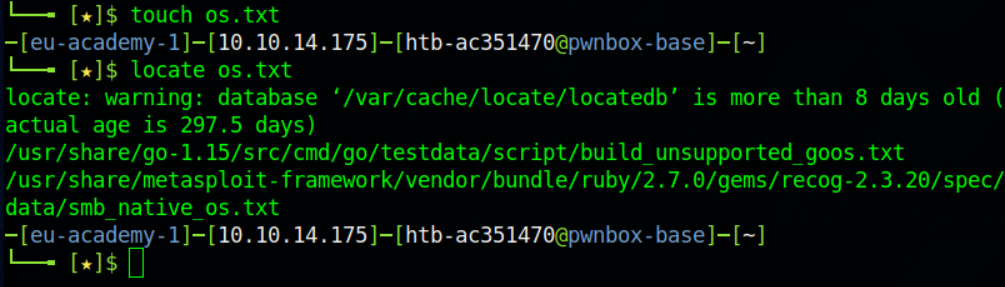


Fig 3.b.10

9.Locate

Locate command in Linux is used to find the files by name. There is two most widely used file searching utilities accessible to users are called find and locate. The locate utility works better and faster than find command counterpart because instead of searching the file system when a file search is initiated, it would look through a database. This database contains bits and parts of files and their corresponding paths on your system. By default, locate command does not check whether the files found in the database still exist and it never reports files created after the most recent update of the relevant database.



10.pwd

pwd stands for Print Working Directory. It prints the path of the working directory, starting from the root.pwd is shell built-in command(pwd) or an actual binary(/bin/pwd).

$PWD is an environment variable which stores the path of the current directory.

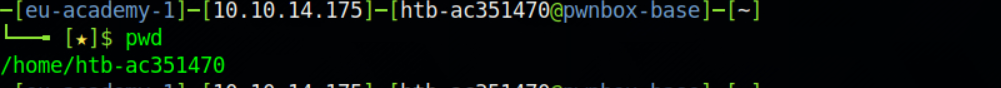


Fig 3b.12

11.mkdir

mkdir command in Linux allows the user to create directories (also referred to as folders in some operating systems ). This command can create multiple directories at once as well as set the permissions for the directories. It is important to note that the user executing this command must have enough permissions to create a directory in the parent directory, or he/she may receive a ‘permission denied’ error.

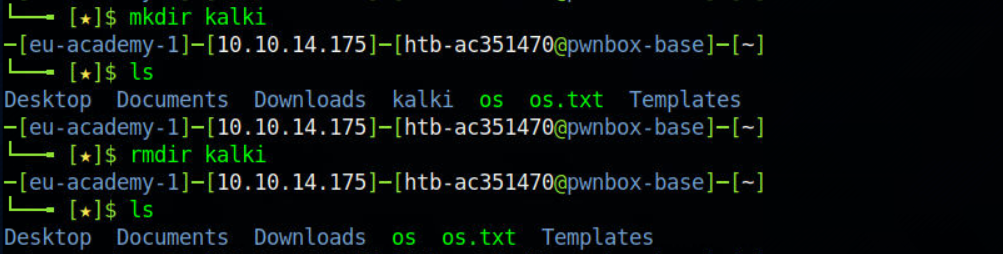
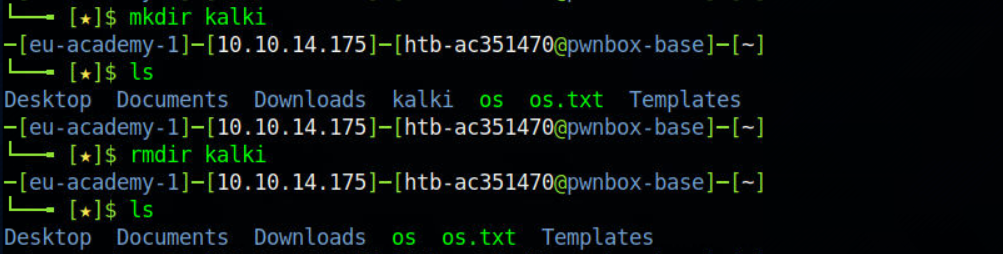


Fig 3b.13

12.rmdir

rmdir command is used remove empty directories from the filesystem in Linux. The rmdir command removes each and every directory specified in the command line only if these directories are empty. So if the specified directory has some directories or files in it then this cannot be removed by rmdir command.



13. More

more command is used to view the text files in the command prompt, displaying one screen at a time in case the file is large (For example log files). The more command also allows the user do scroll up and down through the page. The syntax along with options and command is as follows. Another application of more is to use it with some other command after a pipe. When the output is large, we can use more command to see output one by one.

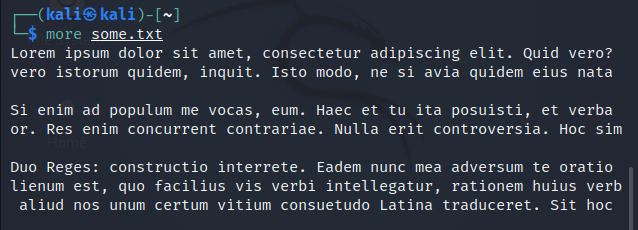


Fig 3b.15

14. Tar

The Linux ‘tar’ stands for tape archive, is used to create Archive and extract the Archive files. tar command in Linux is one of the important command which provides archiving functionality in Linux. We can use Linux tar command to create compressed or uncompressed Archive files and also maintain and modify them.

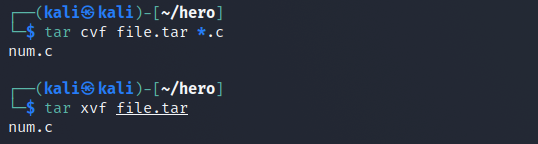


Fig 3b.16

15.zip

ZIP is a compression and file packaging utility for Unix. Each file is stored in single .zip {.zip-filename} file with the extension .zip.

* zip is used to compress the files to reduce file size and also used as file package utility. zip is available in many operating systems like unix, linux, windows etc.
* If you have a limited bandwidth between two servers and want to transfer the files faster, then zip the files and transfer.
* The zip program puts one or more compressed files into a single zip archive, along with information about the files (name, path, date, time of last modification, protection, and check information to verify file integrity). An entire directory structure can be packed into a zip archive with a single command.

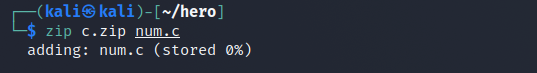


Fig 3b.17

16.Unzip

If you have a zip compressed file, you can unzip it in the Linux command line. The unzip command in Linux is quite versatile and you can use it do a lot more than just extracting zip file.

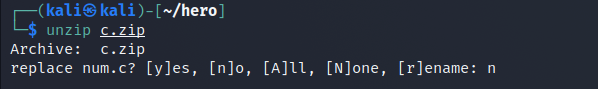
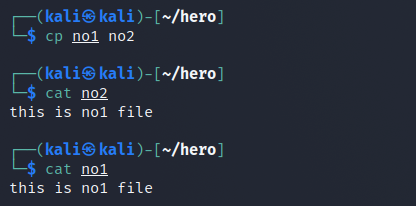


Fig 3b.18

17.CP

cp stands for copy. This command is used to copy files or group of files or directory. It creates an exact image of a file on a disk with different file name. cp command require at least two filenames in its arguments.

cp command works on three principal modes of operation and these operations depend upon number and type of arguments passed in cp command



18.Grep

Grep is an acronym that stands for Global Regular Expression Print.

Grep is a Linux / Unix command-line tool used to search for a string of characters in a specified file. The text search pattern is called a regular expression. When it finds a match, it prints the line with the result. The grep command is handy when searching through large log files.

The grep command consists of three parts in its most basic form. The first part starts with grep, followed by the pattern that you are searching for. After the string comes the file name that the grep searches through.

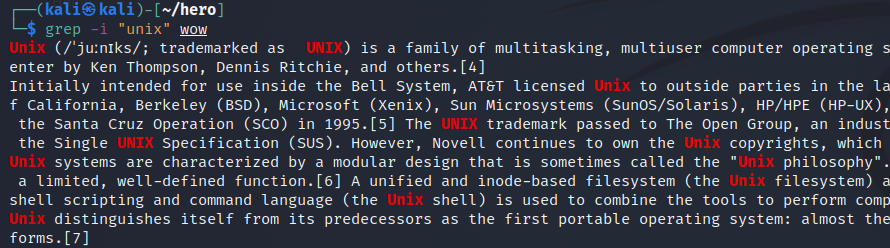
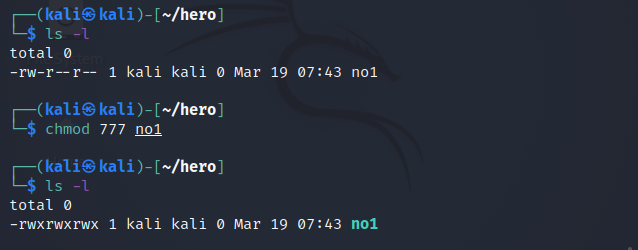


Fig 3b.20

19.Chmod

In Unix-like operating systems, the chmod command is used to change the access mode of a file. Types of permissions which we will be changing using chmod command In linux terminal, to see all the permissions to different files, type ls -l command which lists the files in the working directory in long format.



Conclusion : In the above experiment we performed a number of System management commands we also learned how to change file permission how to zip and unzip file and find certain elements in a file and how to find a specific file etc.

3.C

Aim : Execution of user management commands

Theory

1.Chgrp

chgrp command in Linux is used to change the group ownership of a file or directory. All files in Linux belong to an owner and a group. You can set the owner by using “chown” command, and the group by the “chgrp” command.

First we need to have administrator permission to add or delete groups. We can Login as root for this purpose or using sudo.

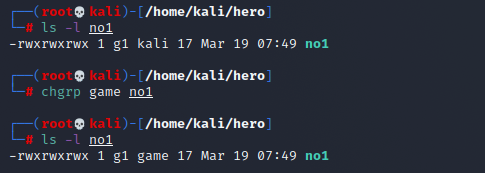


Fig 3c.1

2.Chown

Different users in the operating system have ownership and permission to ensure that the files are secure and put restrictions on who can modify the contents of the files. In Linux there are different users who use the system:

* Each user has some properties associated with them, such as a user ID and a home directory. We can add users into a group to make the process of managing users easier.
* A group can have zero or more users. A specified user can be associated with a “default group”. It can also be a member of other groups on the system as well.

To protect and secure files and directory in Linux we use permissions to control what a user can do with a file or directory.

Linux uses three types of permissions:

* Read: This permission allows the user to read files and in directories, it lets the user read directories and subdirectories stores in it.
* Write: This permission allows a user to modify and delete a file. Also it allows a user to modify its contents (create, delete and rename files in it) for the directories. Unless the execute permission is not given to directories changes does do affect them.
* Execute:Thispermission on a file allows it to get executed. For example, we have a file named php.sh so unless we don’t give it execute permission it won’t run.

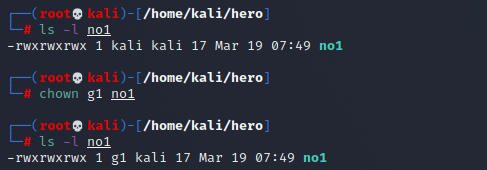


Fig 3c.2

3.Groupadd

Groups in Linux refer to the user groups. In Linux, there can be many users of a single system, (normal user can take uid from 1000 to 60000, and one root user (uid 0) and 999 system users (uid 1 to 999)). In a scenario where there are many users, there might be some privileges that some users have and some don’t, and it becomes difficult to manage all the permissions at the individual user level. So using groups, we can group together a number of users, and set privileges and permissions for the entire group. groupadd command is used to create a new user group.



Fig 3c.3

4.Sudo

sudo (Super User DO) command in Linux is generally used as a prefix of some command that only superuser are allowed to run. If you prefix “sudo” with any command, it will run that command with elevated privileges or in other words allow a user with proper permissions to execute a command as another user, such as the superuser. This is the equivalent of “run as administrator” option in Windows. The option of sudo lets us have multiple administrators.

These users who can use the sudo command need to have an entry in the sudoers file located at “/etc/sudoers”. Remember that to edit or view the sudoers file you have to use sudo command. To edit the sudoers file it is recommended to use “visudo” command. By default, sudo requires that users authenticate themselves with a password which is the user’s password, not the root password itself.

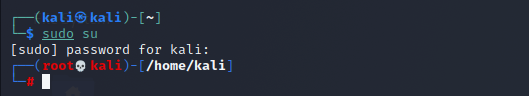


Fig 3c.3

4.Useradd

useradd is a command in Linux that is used to add user accounts to your system. It is just a symbolic link to adduser command in Linux and the difference between both of them is that useradd is a native binary compiled with system whereas adduser is a Perl script which uses useradd binary in the background. It make changes to the following files:

* /etc/passwd
* /etc/shadow
* /etc/group
* /etc/gshadow

creates a directory for new user in /home



Fig 3c.4

5.Userdel

userdel command in Linux system is used to delete a user account and related files. This command basically modifies the system account files, deleting all the entries which refer to the username LOGIN. It is a low-level utility for removing the users.

userdel -f: This option forces the removal of the specified user account. It doesn’t matter that the user is still logged in. It also forces the userdel to remove the user’s home directory and mail spool, even if another user is using the same home directory or even if the mail spool is not owned by the specified user.



Fig 3c.5

6.Usermod

usermod command or modify user is a command in Linux that is used to change the properties of a user in Linux through the command line. After creating a user we have to sometimes change their attributes like password or login directory etc. so in order to do that we use the Usermod command. The information of a user is stored in the following files:

* /etc/passwd
* /etc/group
* /etc/shadow
* /etc/login.defs
* /etc/gshadow
* /etc/login.defs

When we execute usermod command in terminal the command make the changes in these files itself.



Fig 3c.7

8.who

who command is used to find out the following information :

1. Time of last system boot

2. Current run level of the system

3. List of logged in users and more.

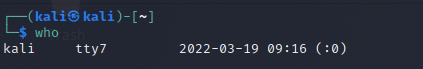


Fig 3c.8

9.whoami

whoami command is used both in Unix Operating System and as well as in Windows Operating System.

* It is basically the concatenation of the strings “who”,”am”,”i” as whoami.
* It displays the username of the current user when this command is invoked.
* It is similar as running the id command with the options -un.



Conclusion : In the above experiment we performed a number of user management commands the helped us to knoe hoe to create a new user , group and granting them permission as per our needs and many more commands to know which user is performing the tasks.