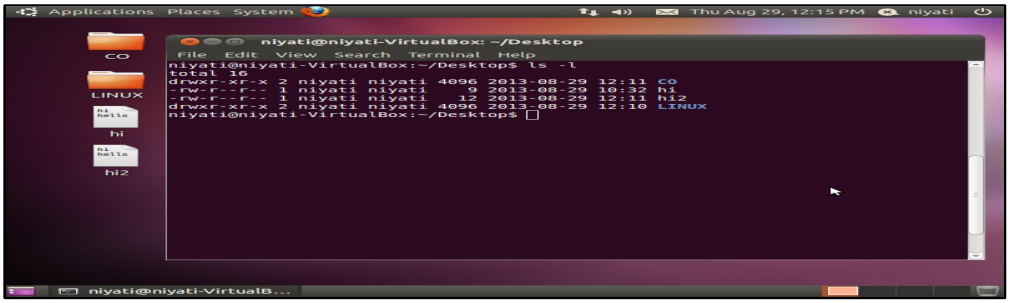
Topic # 6. Basic File Attributes

ls –l for listing file attributes



1st column

- There are 10 characters.
- 1st character:
  - ‘-‘ file is ordinary or regular file
  - ‘d’ file is directory file
  - ‘b’ or ‘c’ file is a device file
  - 'l' for soft link type of file
- 2nd to 10th character:
  - ‘r’ = read
  - ‘w’ = write
  - ‘x’ = execute for file, browse through for directory

2nd column

- It indicates the number of hard links associated with the file.
- A link count greater than one indicates that the file has more than one name.
- 'rm' command removes file only when it's count is 1 otherwise it is like the one of many name of file is removed.

But this does not mean that there are two copies of the file.

3rd column

- It shows the owner of the file.
- Owner is the user that had created the file.
- Owner can change contents and permissions.
- Owner of the directory can create, modify or remove files in that directory.

4th column

- It represents the group owner of the file.

5th column

- It shows the size of the file in bytes.
- File size is only character count.
- It is not the measure of disk space that it occupies.
- A directory maintains a list of filenames along with the inode number, for each file.
- The size of the directory file depends on the size of this list.

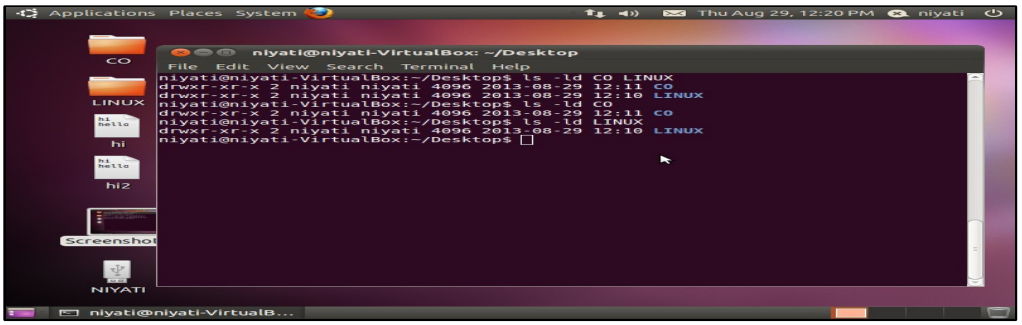
6th, 7th and 8th columns

- They indicate the last modification time of the file.
- Modification means change in content, not change in permissions or ownership.
- If file is less than a year old then year won't be displayed.

9th column

- It displays the filename.

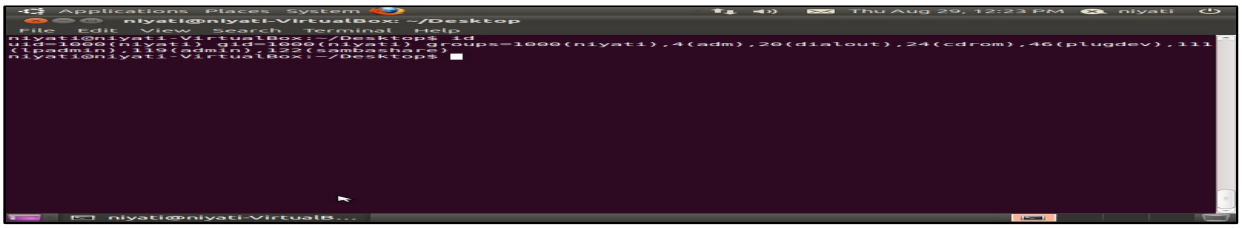
➢ The –d option: listing directory attributes



➢ File Ownership

- If you copy someone else's file, you are the owner of the copy.
- You can't create files in other user's home directory.
- The privileges of the group are set by the owner of the file and not by the group members.
- When the system administrator creates a user account, he has to assign these parameters to the user:
  - User-id (UID) both its name and numeric representation
  - Group-id (GID) both its name and numeric representation

- /etc/passwd : contains UID (number and name) and GID (number)
- /etc/group: contains GID (number and name)
- To view UID and GID use id command.



▯ chmod : changing file permissions

- chmod (change mode) command is used to set the permissions of one or more files for all three categories of users (user, group and others).
- This command can be run by user (owner) and superuser.
- This command can be used in two ways:
  - Relative manner : by specifying the changes to the current permissions
  - Absolute manner : by specifying the final permissions

- **Relative Permissions**
  - chmod only changes the permissions specified in the command line and leaves the other permissions unchanged.
  - $ chmod *category operation permission* filename(s)
    - ▯ Category :- user(u) or group(g) or others(o) or all(a)
    - ▯ Operation :- assign (+) or remove (-) permission
    - ▯ Permission :- read (r) or write (w) or execute(e)
  - chmod accepts multiple filenames in command line.
  - chmod accepts multiple expressions delimited by commas.
  - chmod can be used to set more than one permission.

- **Absolute Permissions**

Read	4
Write	2
Execute	1
Read and Write	4 + 2 = 6
Read and Execute	4 + 1 = 5
Write and Execute	2 + 1 = 3

Read, Write and Execute	4 + 2 + 1 = 7
No permissions	0

$ chmod 644 file1
Here, user has read and write permissions, group and others have only read permissions.

777 means all permissions to all. And 000 means no permission to all. Both these conditions are extreme and are not generally used as they would adversely affect the security of the system.

chmod can be used recursively with (-R) option
$ chmod –R a+r btech
In above, btech is a directory and we want to make all the files and directory readable for all categories of users.

- **Directory Permissions**

The default directory permissions are **rwxr-xr-x**. User (owner) can read, write and execute. Group and others can read and execute but they cannot write.

If the directory has write permission for group and others also then any user can remove every file in directory.

Please, note that we can not execute directory hence 'x' for directory itself means who all can browse through/search through the directory.

- **chown : changing file owner**

Changing ownership requires superuser permission
chown username filename

- **chgrp : changing group owner**

This does not require superuser permission.
$ chgrp groupname filename

Both chown and chgrp can be used with (-R) recursive option for directories and their content.

You may also use chown username:groupname filename to set both owner and group via single command.

For user management in linux as root you may do following exercise:

useradd username // this creates user with username and group username

groupadd groupname // this creates a group with no members enrolled

useradd -g groupname username // this creates user name username enrolled to group named gropuname as primary group of user.

usermod command can be used to update any user.

passwd username // this will let administrator set password for user named username.

userdel username // deletes username account from the system.

chage command let you change user password expiry information

p.s.

'groups' command displays all the group name any user is member of or enrolled in.

'id' command shows current user's UID and GID and other information.

ls -l displays last modification time field

ls -lu displays last access time field

ls -lc displays last inode modification time field

Using stat command for file or directory you can read information like all above timestamp and more.

```
[user1@centos ~]$ stat demo.txt
  File: `demo.txt'
  Size: 0               Blocks: 8          IO Block: 4096   regular empty file
Device: fd00h/64768d    Inode: 22053871    Links: 1
```

```
Access: (0664/-rw-rw-r--)  Uid: (  501/ user1)   Gid: (  502/ user1)
Access: 2013-09-13 11:36:04.000000000 +0530
Modify: 2013-09-13 11:36:04.000000000 +0530
Change: 2013-09-13 11:36:04.000000000 +0530
[user1@centos ~]$ mkdir btech
[user1@centos ~]$ stat btech
  File: `btech'
  Size: 4096            Blocks: 16         IO Block: 4096   directory
Device: fd00h/64768d    Inode: 22184128    Links: 2
Access: (0775/drwxrwxr-x)  Uid: (  501/ user1)   Gid: (  502/ user1)
Access: 2013-09-13 11:36:20.000000000 +0530
Modify: 2013-09-13 11:36:20.000000000 +0530
Change: 2013-09-13 11:36:20.000000000 +0530
[user1@centos ~]$
```

Wish to learn about ACLs and File Sytem mounting?

Let's say what will you do to mount your windows drives into linux?

Access Control Lists (ACL):

ACLs give users and administrators flexibility and direct fine-grained control over who can read, write, and execute files.

Access Control Lists (ACLs) are applied to files and directories. ACLs are an addition to the standard Unix file permissions (r,w,x,-) for User, Group, and Other for read, write, execute and deny permissions.

ls - show files which have acces control lists applied ("+" sign in last collumn)
Example: `-rw-rw-r--+`

Configuration for allowing the use of ACL on a filesystem:

File: `/etc/fstab`

```
...
...
LABEL=/home             /home                   ext3    rw,acl          1 2
...
...
```

Note:

- Note the addition of the attribute "acl" for the filesystem "/home/".
- Issue the following commands:
  - `umount /home`
  - Edit the file `/etc/fstab` and add the directive "acl".
  - `mount /home`

  or remount the command: `mount -v -o remount /home` which works on a drive even if in use.

You may either use mount and umount command everytime to mount drives into linux or you may put proper entry in /etc/fstab file for auto mounting of drives.

ACL Courtesy : http://www.yolinux.com/TUTORIALS/LinuxTutorialManagingGroups.html

Contributors (CE Department, DDU):

- Prof. Niyati J. Buch
- Prof. Jigar M. Pandya