LESSON – 12

MANAGING ADVANCED PERMISSIONS

LEARNING OUTCOMES

- Manage advanced permission of files
- Use setuid, setgid, sticky bit
- Demonstrate the usage of sticky bit in shared environment
- Work with ACLs

MANAGING ADVANCED PERMISSIONS

- Managing Advanced Permission:
    - It is a special permission set for files
    - It not set by default, but provide useful additions

Three advanced/special permissions

- Set user ID (SUID)
- Set group ID (SGID)
- Sticky bit

MANAGING ADVANCED PERMISSIONS

- SUID:
  - You can see the SUID permission in the file **/usr/bin/cat**
  - ls –l /usr/bin/cat

- ls –l /usr/bin/cat

  -rwxr-xr-x 1 root root 26968 Feb 7 2022 /usr/bin/cat

MANAGING ADVANCED PERMISSIONS

- Example

```
root@cst:/home/cst# whereis cat
cat: /usr/bin/cat /usr/share/man/man1/cat.1.gz

root@cst:/home/cst# ls -l /usr/bin/cat
-rwxr-xr-x 1 root root 26968 Feb  7  2022 /usr/bin/cat
```

Login to your other account and run the command as:
cat /etc/shadow

```
root@cst:/home/cst# chmod u+s /usr/bin/cat

root@cst:/home/cst# ls -l /usr/bin/cat
rwsr-xr-x 1 root root 26968 Feb  7  2022 /usr/bin/cat
```

MANAGING ADVANCED PERMISSIONS

- Example

```
pce@cst:$ cat /etc/shadow -> What do you notice?
cat: /usr/bin/cat /usr/share/man/man1/cat.1.gz
```

For security reasons, execute the following commands to exit out of the current user and revert the permissions of the cat command to its original state.

```
root@cst:/home/cst# chmod u-s /usr/bin/cat
```

MANAGING ADVANCED PERMISSIONS

- GUID:
    - If applied to executable file, it gives the user who executes the file the permission of the group owner of that file.
    - Hardly use. Normally, applied to some system files as a default setting.

MANAGING ADVANCED PERMISSIONS

- GUID:
  - when applied to a directory, SGID may be useful as you can use it to set default group ownership on files and subdirectories created in that directory

```
#mkdir mydir
#chmod g+s mydir
#mkdir mydir/subdir1
#ls –l mydir/
[you will see that the subdir1 has default set group ID]
```

MANAGING ADVANCED PERMISSIONS

- Example:

```
[root@cst ~]# ls -l /usr/bin/wall
-rwxr-sr-x 1 root tty 18592 Feb 21  2022 /usr/bin/wall
```

Open new terminal and send wall message from this terminal

```
cst@cst:~$ wall "TEST"

Broadcast message from cst@cst (pts/0) (Mon Sep 23 17:27:06 2024):

TEST
```

MANAGING ADVANCED PERMISSIONS

- The same message will appear in another terminal

  Remove SGID (chmod g-s /usr/bin/wall), and test it, this time it won’t work.

MANAGING ADVANCED PERMISSIONS

- Sticky bit:
  - This special permission protect files against accidental deletion. Only user owner of that file can be deleted.
  - Creates an environment for multiusers to work on to the file collaboratively.

MANAGING ADVANCED PERMISSIONS

- Example:

```
cst@cst:~$ ll -d /tmp
drwxrwxrwt 12 root root 4096 Sep 23 16:51 /tmp/
```

When applied sticky bit, a user can delete files only if either of the following is true:

- The user is an owner of the file
- The user is owner of the directory where the file exists

Now, create files in /tmp directory and try removing files (before removing sticky bit and after removing sticky bit) [Demonstration of working sticky bit]

MANAGING ADVANCED PERMISSIONS

- Applying Advanced Permissions:

  You can use chmod to apply special permission

  - For SUID, use chmod u+s
  - For SGID, use chmod g+s
  - For Sticky bit, use chmod +t

MANAGING FILE PERMISSION

- Applying Advanced Permissions

Permission	Numeric value	Relative value	On Files	On Directories
SUID	4	u+s	User executes file with permissions of file owner	No meaning
GUID	2	g+s	User executes file with permissions of group owner	Files created in directory get the same group owner
Sticky bit	1	+t	No meaning	Prevents users from deleting files from other users.

ACTIVITY I

- Working with Special Permissions:

1. Login with sudo user and make directory /Data/Sales in single line command
2. Add users u1 and u2 in sales group after creating users (Hint: create u1 and u2 users and sales group. Add u1,u2 into sales group]
3. Create two empty files in /Data/Sales directory as u1file1 and u1file2 and apply u1 user owner and sales group owner recursively to /Data directory
4. Change permission of /Data directory recursively to 775.
5. Switch to u2 , who is also a member of the sales group.
6. Use cd /Data/Sales and type ls –l (You will notice that there are two files that were created by user u1. Note down the permission of group owner. Remove files by command rm –f u1*. Did you succeed in removing it? If yes, why?
7. Escalate your current permission to root level [su -] and set SGID and Sticky bit on to /Data/Sales shared group directory.
8. Now switch to u2 user and create another two empty files as u2file1 and u2file2 in /Data/Sales directory [Make sure you are using u2 user shell]

1. Now switch to u1 (su – u1) and create two empty files u1file3 and u1file4. You can create these two files. Why? [Hint: SGID]
2. 10. Type rm –rf u2file*. Are you successful of removing these two files? Why? [Hint: Sticky bit]

MANAGING ACLS

- ACLs Introduction:
  - Even in advanced permission, we cannot give permissions to more than one user or one group on the same file/directory.
  - Access Control Lists do offer this feature.
  - The ACL also allow administrators to set default permissions in a sophisticated way where the permissions that are set can differ on different directories.
  - Think of a scenario in which a particular user is not a member of group created by you but still you want to give some read or write access, how can you do it without making user a member of group, here comes in picture Access Control Lists, ACL helps us to do this trick.

MANAGING ACLS

- Drawback – Not all utilities support it.
- ACL setting may lost at copying or moving files
- Backup software might not be able to backup ACL settings.

MANAGING ACLS

Commands for Managing ACLs

- getfacl
- setfacl

Backup and restore ACL file

```
$Backup: getfacl –R /directory > file.acls
$ Restore: setfacl - -restore=file.acl
```

MANAGING ACLS

Set ACLs

- Often applied to directories, not on individual file

Changing and Viewing ACL Settings

- Use getfacl and setfacl
- Use of ls –l does not show any existing ACL, instead show + after the listing of the permissions. [Solution: getfacl or treem(need to install)]

MANAGING ACLS

Example

```
cst@cst:~$ getfacl TESTTEST/
# file: TESTTEST/
# owner: jiwan
# group: jiwan
user::rwx
group::rwx
other::r-x
```

MANAGING ACLS

Setting read, write and execute permissions to user pce on TESTTEST directory

```
cst@cst:~$ sudo setfacl -m u:pce:rw TESTTEST
cst@cst:~$ getfacl TESTTEST/
# file: TESTTEST/
# owner: jiwan
# group: jiwan
user::rwx
user:pce:rw-
group::rwx
mask::rwx
other::r-x
```

DEMO

MANAGING ACLS

Setting read, write and execute permissions to group pce on TESTTEST directory

```
cst@cst:~$ sudo setfacl -m g:pce:rw TESTTEST
cst@cst:~$ getfacl TESTTEST/
# file: TESTTEST/
# owner: jiwan
# group: jiwan
user::rwx
user:pce:rw-
group::rwx
group:pce:rw-
mask::rwx
other::r-x
```

MANAGING ACLS

Working with Default ACLs

- ACL can give more permissions to more than one user or group
- Another benefit à can enable inheritance (permission
- can be set for all new items that are created in the directory)

MANAGING ACLS

Working with Default ACLs

- Configure multiusers or groups to the same directory, you have to set ACLs twice:

  1. setfacl –R –m (to modify the ACLs for current files)
  2. setfacl –m d: (to take care of all new items that will be created)

SUMMARY

- In this lesson, you have learnt that:

  - The special permission can be set by using SUID, SGID, and sticky bit
  - Using special permission may wrongly set the permission for files and directory.
  - Special permission cannot be set to multiusers where ACLs features help to do it.