LINUX PROGRAMMING

ASSIGNMENT-5

NAME: V S SAIDATTA USN: ENG24CY0175 ROLLNO:28

1. What is a shell in Linux OS? How many categories of shell is currently exists in Linux? Why bash shell is very popular in Linux distribution?

A.

A shell in Linux is a command line interpreter that connects the user to the operating system kernel.

Type and run commands

- Run programs.
- Manage files and processes.
- Write shell scripts to automate tasks.

Shells in Linux divided into two categories:

- Command Line Shells.
  - These take input from the keyboard and show output on the terminal.
  - Examples: Bash (Bourne Again Shell), Sh (Bourne Shell), Csh (C Shell),Ksh (Korn Shell)
- Graphical Shells
  - These offer a graphical interface to interact with the system, which is built on top of the window system.
  - Examples: GNOME Shell, KDE Plasma Shell.
  - Why is the Bash shell (Bourne Again Shell) used so widely in Linux-based Operating Systems?

Bash is so widely used for the following reasons:

- It is usually the default shell in various popular Linux distributions like Ubuntu, Fedora.
- simultaneously, it is a combination of the traditional Bourne Shell and C Shell, resulting in a more powerful Shell with friendly usability and learning curves.
- It is meant to support common Unix commands by supporting command history, tab completion.

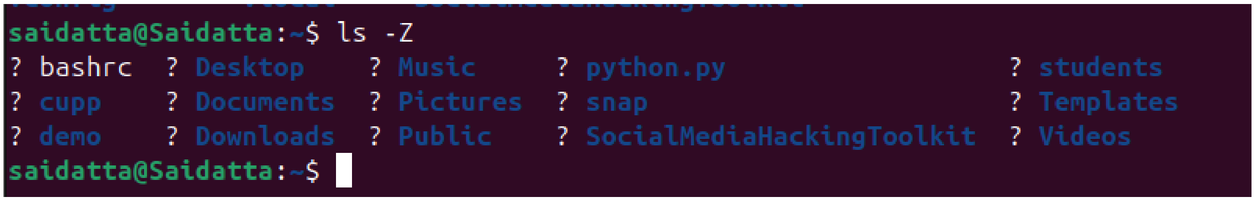
- Additionally, it can be applied for shell scripting, automation tasks, and sysadmin duties.

2. What does the ls -Z command display?

A.

The ls -Z command in Linux is used to display SELinux security context information for files and directories.

- ls ----> lists files and directories.
- -Z ----> shows SELinux (Security-Enhanced Linux) security context labels along with normal file details.



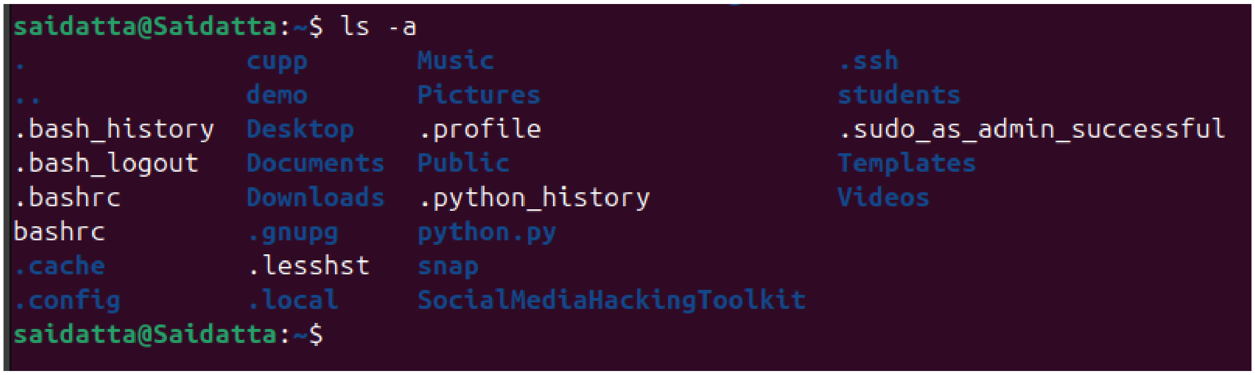
```
saidatta@Saidatta:~$ ls -Z
? bashrc  ? Desktop    ? Music      ? python.py                  ? students
? cupp    ? Documents  ? Pictures   ? snap                       ? Templates
? demo    ? Downloads  ? Public     ? SocialMediaHackingToolkit  ? Videos
saidatta@Saidatta:~$ 
```

3. Write a command to list all hidden files in the current directory.

A.

“ls -a”:

- “ls” -- list of files and directories.
- “-a” -- this command shows all files, even files such as starting with dot(.) which are hidden in linux.



```
saidatta@Saidatta:~$ ls -a
.               cupp       Music                      .ssh
..              demo       Pictures                   students
.bash_history   Desktop    .profile                   .sudo_as_admin_successful
.bash_logout    Documents  Public                     Templates
.bashrc         Downloads  .python_history            Videos
bashrc          .gnupg     python.py
.cache          .lesshst   snap
.config         .local     SocialMediaHackingToolkit
saidatta@Saidatta:~$
```

4. Explain the difference between hard links and soft links (symbolic links) in Linux.

A.

HARD LINK:

- A hard link serves as a direct reference to the same inode as the original file.
- A hard link will always have the same inode number as the original file.
- If the original file is deleted, the hard link will still work because it points to the same data.
- A hard link must be on the same filesystem (partition) as the original file.

SOFT LINK:

- A soft link is a reference or shortcut to the path of another file (like a Windows shortcut).
- Its inode number is different than that of the original file.
- If the original file is deleted, the soft link becomes broken (invalid).
- Can cross filesystems or partition.

5. A file has permissions -rwxr-x--x. Explain who can read, write, and execute it.

A.

The permission “-rwxr-x--x":

- 1st char: “-” represent for files and “d” represent for directory.
- 2nd to 4th char: represents user permissions.
- 5th to 7th char: represents group permissions.
- 8th to 10th char: represents other permissions.

Let's know about the symbols meaning:

‘r’ - Read ‘w’- Write ‘x’ - Execute ‘-’ - no permission

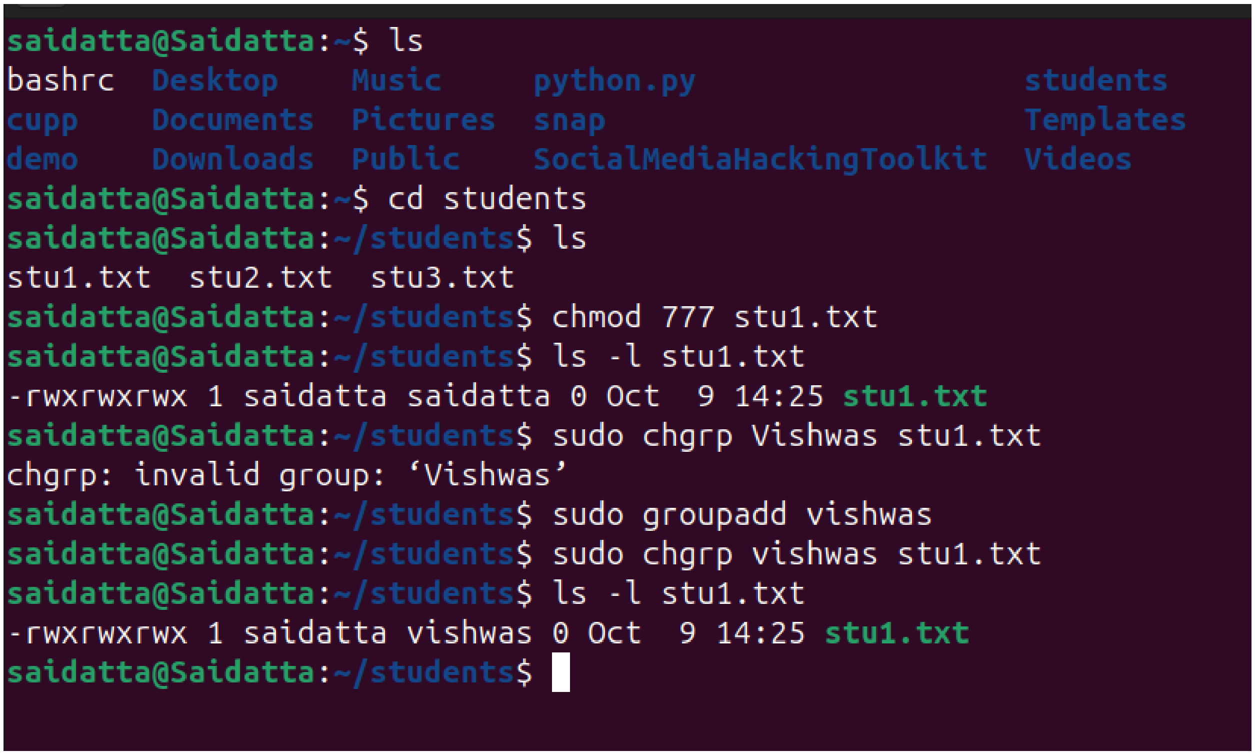
Let’s breakdown the file permission given in the question (-rwxr-x--x):

- Ower: full access for user. User can read, write, execute the file.
- Group members: members can read and execute the file, but they cannot edit it.
- Other: everyone one can only execute the file.

6. Write the command to change the group ownership of a file data.txt to group staff.

A.

The ‘chgrp’ used to change the group ownership of a file.



```
saidatta@Saidatta:~$ ls
bashrc  Desktop    Music      python.py                  students
cupp    Documents  Pictures   snap                       Templates
demo    Downloads  Public     SocialMediaHackingToolkit  Videos
saidatta@Saidatta:~$ cd students
saidatta@Saidatta:~/students$ ls
stu1.txt  stu2.txt  stu3.txt
saidatta@Saidatta:~/students$ chmod 777 stu1.txt
saidatta@Saidatta:~/students$ ls -l stu1.txt
-rwxrwxrwx 1 saidatta saidatta 0 Oct  9 14:25 stu1.txt
saidatta@Saidatta:~/students$ sudo chgrp Vishwas stu1.txt
chgrp: invalid group: ‘Vishwas’
saidatta@Saidatta:~/students$ sudo groupadd vishwas
saidatta@Saidatta:~/students$ sudo chgrp vishwas stu1.txt
saidatta@Saidatta:~/students$ ls -l stu1.txt
-rwxrwxrwx 1 saidatta vishwas 0 Oct  9 14:25 stu1.txt
saidatta@Saidatta:~/students$ 
```

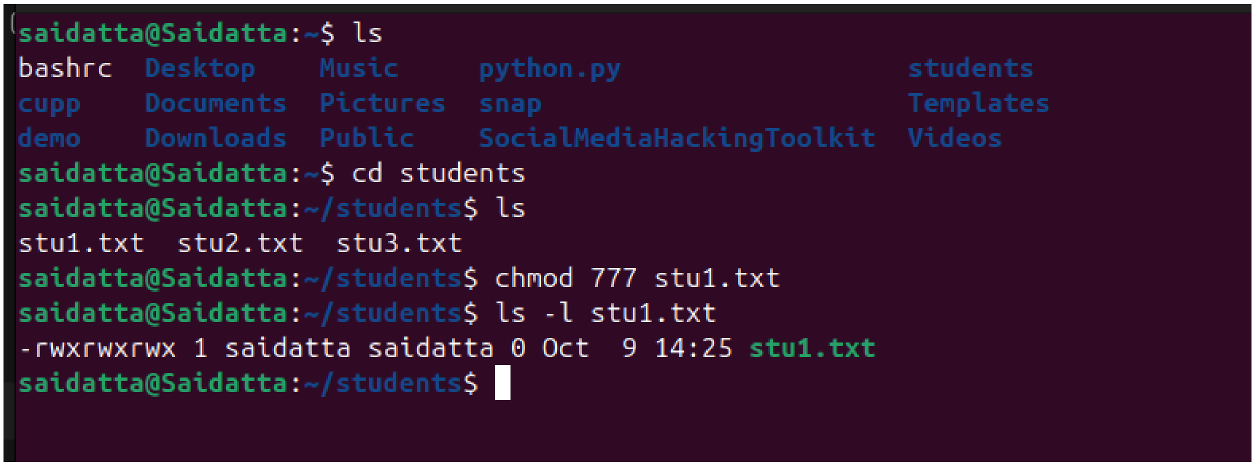
7. Why is it dangerous to give 777 permissions to a file? Explain with an example.

A.

Creating a file with 777 permissions allows everyone to read, write, and execute that file which is not a good idea as anyone can change, delete, or misuse it and you might lose your data or security. This poses **serious security risks**, especially on multi-user or server systems.

Never use 777 permissions unless you must.

- Instead, use this:
  - For most files, 644 (rw-r--r--) permissions will suffice.
  - For most scripts, 755 (rwxr-xr-x) permissions will suffice.



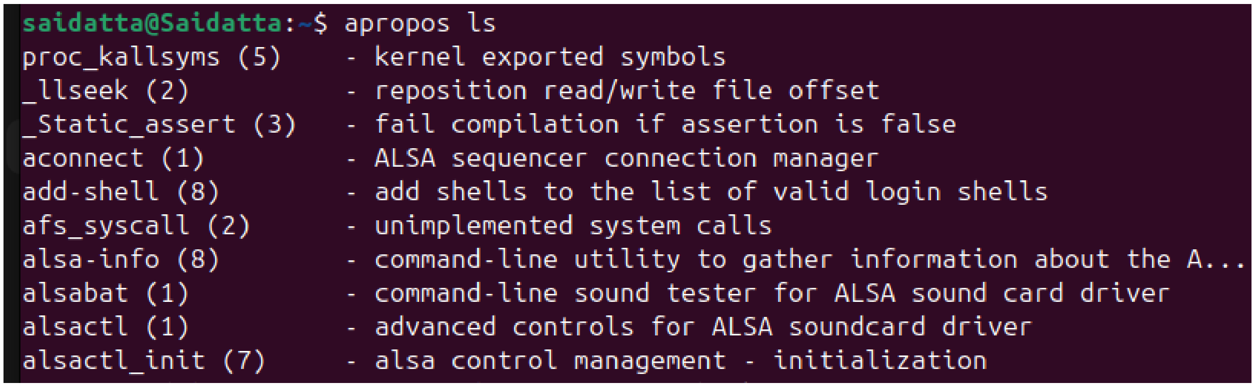
```
saidatta@Saidatta:~$ ls
bashrc  Desktop    Music      python.py                  students
cupp    Documents  Pictures  snap                       Templates
demo    Downloads  Public    SocialMediaHackingToolkit  Videos
saidatta@Saidatta:~$ cd students
saidatta@Saidatta:~/students$ ls
stu1.txt  stu2.txt  stu3.txt
saidatta@Saidatta:~/students$ chmod 777 stu1.txt
saidatta@Saidatta:~/students$ ls -l stu1.txt
-rwxrwxrwx 1 saidatta saidatta 0 Oct  9 14:25 stu1.txt
saidatta@Saidatta:~/students$ 
```

8. What is the difference between apropos (i.e., man -k) and whatis (i.e., man -f)?

A.

‘apropos <keyword> or man -k <keyword>’:

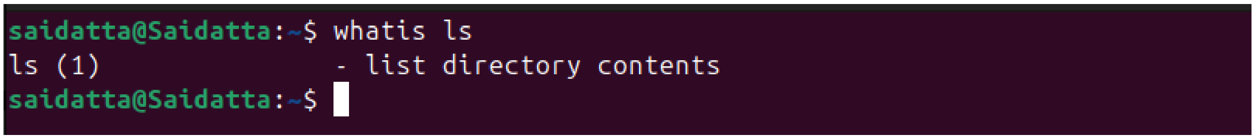
- Searches the manual page descriptions for any command that relates to the specified keyword.
- Keyword search -- provides a list of all commands that relate to the keyword.



```
saidatta@Saidatta:~$ apropos ls
proc_kallsyms (5)     - kernel exported symbols
_llseek (2)           - reposition read/write file offset
_Static_assert (3)    - fail compilation if assertion is false
aconnect (1)          - ALSA sequencer connection manager
add-shell (8)         - add shells to the list of valid login shells
afs_syscall (2)       - unimplemented system calls
alsa-info (8)         - command-line utility to gather information about the A...
alsabat (1)           - command-line sound tester for ALSA sound card driver
alsactl (1)           - advanced controls for ALSA soundcard driver
alsactl_init (7)      - alsa control management - initialization
```

‘whatis <command> or man -f <command>’:

- Presents a brief description of a command.
- Exact name match -- only provides the brief description of the specified command's name.



```
saidatta@Saidatta:~$ whatis ls
ls (1)                - list directory contents
saidatta@Saidatta:~$ 
```

9. Write a command to redirect the error output of a command to a file named error.log.

A.

To redirect the error output (also referred to as standard error or stderr) of a command to a file named error.log use:

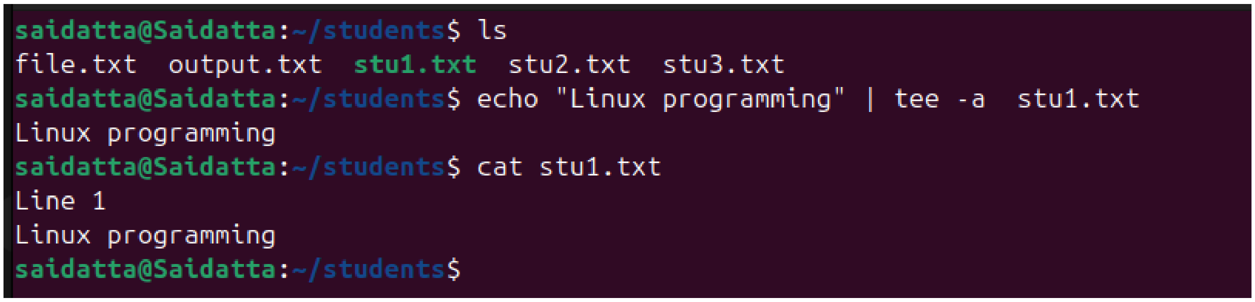
- command - the command you are executing.
- 2> - redirects standard error (stderr), which has file descriptor 2.
- error.log - this is the file that all error messages will be stored.

10. How can you use the tee command to append output to a file instead of overwriting it?

A.

To append output instead of overwriting with the tee command, use the ‘-a' option.

- ‘command’ (like ls, nmap, ping, etc) - The command that produces the output you want to capture.
- ‘|’ - Connects the output from the command to another command.
- ‘tee' - Reads and then writes the content to standard output (the terminal) and to a file.
- ‘-a’ - Appends to the indicated file instead of overwriting it.
- ‘filename' - The file where the output will be sent.



```
saidatta@Saidatta:~/students$ ls
file.txt  output.txt  stu1.txt  stu2.txt  stu3.txt
saidatta@Saidatta:~/students$ echo "Linux programming" | tee -a  stu1.txt
Linux programming
saidatta@Saidatta:~/students$ cat stu1.txt
Line 1
Linux programming
saidatta@Saidatta:~/students$
```