

Module 4 – Linux Fundamentals

Command Syntax

- ▶ **Commands typically have the syntax:**

command option(s) argument(s)

- ▶ **Options:**

- ▶ Modify the way that a command works
- ▶ Usually consist of a hyphen or dash followed by a single letter
- ▶ Some commands accept multiple options which can usually be grouped together after a single hyphen

- ▶ **Arguments:**

- ▶ Most commands are used together with one or more arguments
- ▶ Some commands assume a default argument if none is supplied
- ▶ Arguments are optional for some commands and required by others

File Permissions

Every **file** and **directory** in your account can be **protected** from or made **accessible** to other **users** by changing its access **permissions**. Every user has responsibility for controlling access to their files.

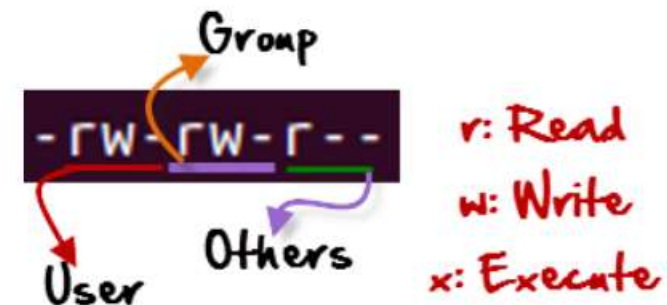
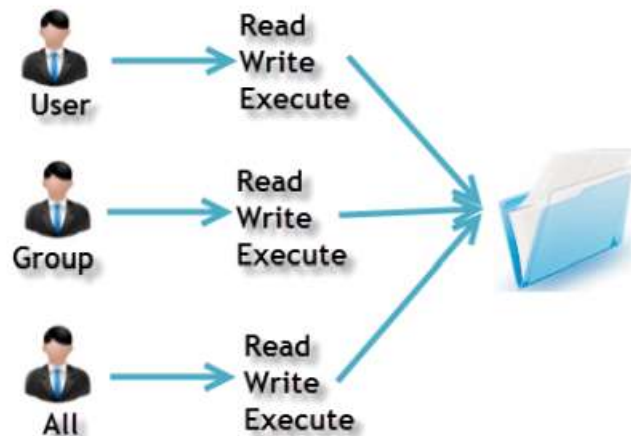
- ▶ There are 3 type of permissions
 - ▶ r -**read**
 - ▶ w -**write**
 - ▶ x -**execute** = running a program
- ▶ Each permission (rwx) can be controlled at three levels:
 - ▶ u -**user** = yourself
 - ▶ g -**group** = can be people in the same project
 - ▶ o -**other** = everyone on the system
- ▶ File or Directory permission can be displayed by running ls -l command
 - ▶ -rwxrwxrwx
- ▶ Command to change permission - **chmod**

Permissions using Numeric mode

- Permission to a file and directory can also be assigned numerically

chmod ugo+r File or chmod 444 File
-r--r--r--

Owners assigned Permission On Every File and Directory

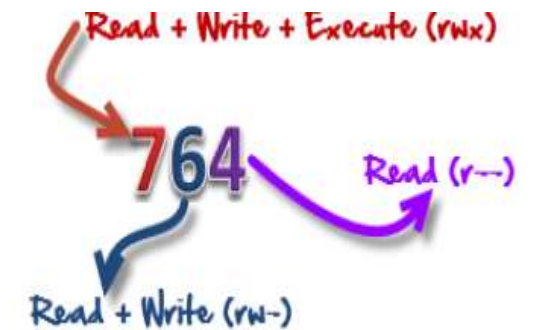


Permissions using Numeric mode

The table below assigns numbers to permissions types

Number	Permission Type	Symbol
0	No Permission	---
1	Execute	--X
2	Write	-W-
3	Execute + Write	-WX
4	Read	r--
5	Read + Execute	r-X
6	Read +Write	rw-
7	Read + Write +Execute	rwX

chmod 764 File



File Permissions - umask

- ▶ When creating a file or directory, a set of default permissions are applied.
- ▶ These **default** permissions are determined by the **umask**.
- ▶ The umask specifies permissions that you do not want set on by default.
- ▶ You can display the umask with the **umask** command, `umask -S`

```
[root@MyFirstLinux permission]# umask  
0022
```

- ▶ Full permission of File – 666 , Default permission of file = 666-022 → 644
- ▶ Full permission of Dir – 777 , Default permission of Dir = 777 -022 → 755
- ▶ You can set temporary umask value → `umask 034` (all files/dir will be created with this new default permission).it holds good until a session ends.

File Ownership

- ▶ There are 2 owners of a file or directory
 - ▶ **User** and **group**
 - ▶ The users and groups of a system can be locally managed in **/etc/passwd** and **/etc/group**
- ▶ Command to change file ownership – Only root user can perform
 - ▶ **chown** changes the ownership of a file
 - ▶ chown user Filename
 - ▶ chown user:group filename
 - ▶ **chgrp** changes the group ownership of a file
 - ▶ Chgrp grpname filename
 - ▶ **Recursive** ownership change option (Cascade) **-R**
 - ▶ chown -R user:grp dirname
 - ▶ chgrp -R grp dirname

Adding text to Files

3 Simple ways to **add text** to a file

- ▶ **vi**
- ▶ **Redirect command output > or >>**
- ▶ **echo > or >>**

Input and output redirects

- ▶ There are 3 redirects in Linux
 - ▶ Standard input (**stdin**) and it has file descriptor number as 0
 - ▶ Standard output (**stdout**) and it has file descriptor number as 1
 - ▶ Standard error (**stderr**) and it has file descriptor number as 2
- ▶ By default when running a command its output goes to the terminal
- ▶ The output of a command can be routed to a file using > or >> symbol
 - ▶ **ls -l > listings**
 - ▶ **Pwd >> findpath**

input redirection

- ▶ Redirecting stdin is done with < (short for 0<)
 - ▶ `cat < text.txt`
- ▶ **<< here document**
- ▶ The here document (sometimes called here-is-document) is a way to append input until a certain sequence (usually EOF) is encountered.
- ▶ The EOF marker can be typed literally or can be called with Ctrl-D
 - ▶ `cat <<EOF > filename` or `cat <<EOF >>filename`
 - >Hi
 - >Bye
 - >EOF

Redirect both stdout and stderr

- ▶ To redirect both stdout and stderr to the same file, use `2>&1`.
- ▶ Note that the order of redirections is significant.
- ▶ For example, the command **`ls > dirlist 2>&1`** directs both standard output (file descriptor 1) and standard error (file descriptor 2) to the file `dirlist`
- ▶ while the command **`ls 2>&1 > dirlist`** directs only the standard output to file `dirlist`

Filters or Text Processing commands

Tee ,Pipes and wc

▶ Tee

- ▶ The **tee** filter puts stdin on stdout and also into a file.
- ▶ So tee is almost the same as cat, except that it has two identical outputs.
 - ▶ `ls -l | tee filename`

▶ Pipes (|)

- ▶ A pipe is used by the shell to connect the output of one command directly to the input of another command. `Cmd1 | cmd2`

▶ wc

- ▶ Counting words, lines and characters is easy with wc.
 - ▶ `wc file ,wc -l file ,wc -m file ,wc -w file`

Cut

- ▶ The cut filter can select columns from files, depending on a delimiter or a count of bytes.
 - ▶ `cut -c1-3 filename` = List range of characters
 - ▶ `cut -c1-3,6-8 filename` = List specific range of character
- ▶ The example below shows -cut to filter for the username and userid in the /etc/passwd file.
- ▶ It uses the colon as a delimiter, and selects fields 1 and 3.
 - ▶ `cut -d: -f1,3 /etc/passwd | tail -4`

Grep

- ▶ The grep command which stands for “global regular expression print,” processes text line by line and prints any lines which match a specified pattern
- ▶ `grep keyword file` = Search for a keyword from a file
- ▶ `grep -i KEYword file` = Search for a keyword ignore case-sensitive
- ▶ `grep -v keyword file` = Display everything but keyword
- ▶ `cat /etc/passwd | grep “keyword”`
- ▶ `egrep “keyword1 | keyword2” file` = search for either keyword

sort,uniq Filter

- ▶ The **sort** filter will default to an alphabetical sort
 - ▶ `sort file`
 - ▶ `sort -r file` = Sort in reverse alphabetical order
 - ▶ `sort -k2 file` = Sort by field number (specify `-n` for numerical sort)
- ▶ **Uniq** command filters out the repeated or duplicate lines
 - ▶ `uniq file` = Removes duplicates
 - ▶ `sort file | uniq` = Always sort first before using uniq their line numbers
 - ▶ `sort file | uniq -c` = Sort first then uniq and list count