**7) Configuring Permissions**

* **Linux file system security.**

**Linux file system security restricts user access the files and directories. User requires permission to access files or directories. “ls -l “ or “ ll ” can be used to check security of any file or directory.**

**r :- READ**

**w :-WRITE**

**x :-EXECUTE**

**d rwx r-x r-x. 1 root root 4096 Jun 13 20:25**

**1 2 3 4 5 6 7 8 9**

**/root**

**10**

1. **File Type**
2. **Owner Permission**
3. **Group Permission**
4. **Other Permission**
5. **Link count**
6. **Owner of file / directory**
7. **Group owner of file / directory**
8. **Size**
9. **Creation date and time(Time Stamp)**
10. **Directory / File Name.**

* **File Types**
* **There are total 7 types of files. Out of which 3 are user defined are as follows:**
* **Normal File (-)**
* **Directory(d)**
* **Link file(l)**

**Remaining 4 files are system defined files.**

* **Block device file(b)**
* **Character device file(c)**
* **Socket files (s)**
* **Pipe file (p)**

1. **Normal file ( - ) :- The regular file is a most common file type found on the linux system. It governs all different files such as text files, images, binary files ,shared libraries etc.**
2. **Directory ( d ) :- Directory is the second most common file type found in linux. Directory can be created with mkdir command .**
3. **Link file ( l ) :- Link count is also called as refrance count. It shows count of link of files / directory.**

**Default link count of directory is 2 whereas default link count of file is 1.Whenever a new directory is created link count of it’s parent directory will increase by 1.**

**There are two types of symbolic link**

1. **Hard link**
2. **Soft link**

* **System Defined Files ( cd /dev )**

1. **Character Device Files ( c ):-Character and block device files allow users and programs to communicate with hardware peripheral devices .**
2. **Block Device File ( b ) :- Block devices are similar to character devices . They mostly govern hardware as hard drives, memory ,etc.**
3. **Socket files ( s ) :- Local domain socket are use for communication between processes. Generally ,they are used by services such as x windows ,syslog and firewall security etc.**
4. **Pipe File ( p ) :- Similarly as socket files named pipe allow communication between two local processes.( mknod :- creates pipe file)**

* **Inode Number :- Metadata(data of data).An inode is a data structure that keeps track of all the files and directories within a linux or unix based file system. Inode stores information about files and directories ,such as file ownership, access mode ( read , write , execute) and file type.**

**df -i /dev/sda1 :- to check how many inode no are available / free in the system.**

**ls -i :- Shows inode number.**

* **Difference between hardlink and softlink.**

|  |  |
| --- | --- |
| **HARD LINK** | **SOFT LINK** |
| **1)Inode no of hardlink is same as that of inode number of original file.** | **1)Inode number of softlink is different than inode number of original file.** |
| **2)Hardlink contain actual data from the file.** | **2)Softlink contain path of the original file and not the actual file** |
| **3)Thus size of hardlink and original file will be same.** | **3)Thus size of soft link depends on path length.** |
| **4)Creating or removing hardlink will increase or decrease link count by 1.** | **4)Creating or removing softlink will not affect link count.** |
| **5)removing original file will only reduce the link count and it will not affect any of its hardlink.** | **5)Removing original file will disable softlink,since softlink points to the non existing file.** |
| **6)Creating hardlink of directory is not possible.**  **Ln <original file> <hardlink name>** | **6)Creating softlink of directory is possible.**  **Ln -s <original file> <soft link name>** |
|  |  |

* **Link count**

**When we create a directory inside another directory the link count of parent directory will increase by 1.**

**File = 1**

**Directory =2**

**Eg. A = Parent directory**

**B = Sub /Children Directory**

**A/B = LINK COUNT OF A +1**

**=2+1**

**=3**

* **User and group ownership.**

**Basically ,by default ,user who creates the file / directory is the owner and his primary group acquires group ownership of that file / directory .User owner and group owner of that file will be shown in 6th and 7th number field .**

**Chown command is used to change owner and group where as using “chgrp” we can change group ownership.**

* **chown <user\_name> : <group\_name > < file/directory name>**
* **chown -R :- gives permission to all files inside user .**
* **chgrp <group\_ name> <file /directory name >**
* **Managing Permission**

**-rw-rw-r--**

**Field 2nd ,3rd and 4th represents permissions for owner, group and other users. Each of that field contain three basic permission which allow user to read ,write and executes files /directories . The effect of these permission differs when applied to file or directory. If applied to a file the read permission gives user right to open file for reading .Therefore user can read its contents “ chmod ” command is used to change these basic permissions.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **permission** | **Appied to files** | **Applied to directories** | **letter** | **number** |
| **Read** | **Open a file / directory** | **List contents of directories** | **r** | **4** |
| **Write** | **Change contents of a file** | **Create and delete file and modify permissions on file.** | **w** | **2** |
| **Execute** | **Run a program** | **Change to the directory** | **x** | **1** |

**Symbolic method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Symbol** | **Description** | **Symbol** | **Description** |
| **U** | **Owner permission** | **+** | **Add permission** |
| **G** | **Group permission** | **-** | **Remove permission** |
| **O** | **Other user Permission** | **=** | **Assign permission** |

* **Chmod <permission in numbers / letters> <file / directory name>**
* **Chmod g=x <file name /directory>:- gives group access to the file to execute.**
* **Chmod o-x <file name / directory>:- remove the execution access from other user.**
* **Chmod o-w <file name /directory>:- remove the write permission from other .**
* **Chmod ugo+rwx <file name /directory> :- add read, write and execute permission.**
* **Chmod g-rwx <file name /directory> :- remove read, write and execute permission for all members in group except file owner.**

**Numeric method**

|  |  |  |
| --- | --- | --- |
| **Octal** | **Binary** | **Permission** |
| **0** | **000** | **---** |
| **1** | **001** | **--x** |
| **2** | **010** | **-w-** |
| **3** | **011** | **-wx** |
| **4** | **100** | **r--** |
| **5** | **101** | **r-x** |
| **6** | **110** | **rw-** |
| **7** | **111** | **rwx** |

**Formula to calculate binary to octal = n\* 2^n**

**= 0\*2^0 + 0\*2^1 + 0\*2^2 =0**

**Syntax :- chmod < octal > < file name / directory >**

* **Full permission**
* **File =666(rw-rw-rw-)**
* **Directory = 777(rwxrwxrwx)**
* **Default permission**
* **When a root user create a file it has given permission “ rw-r--r-- “(644).**
* **A directory is given the permission “rwxrwxr-x”( 775 )**
* **For the root user file and directory permission are 664 and 755.**
* **These default values are determined by the value of umask.**
* **Umask**
* **Local user default umask = 002**
* **Root user default umask = 022**
* **The Linux umask command. umask (user file-creation mode) is**[**a Linux command**](https://phoenixnap.com/kb/linux-commands)**that lets you set up default permissions for newly created files and**[**folders**](https://phoenixnap.com/glossary/what-is-a-folder)**.**
* **A user-defined permissions ‘mask’. A user can choose how to restrict  permissions by using a permissions mask. A permission mask interacts with the default system permissions and changes them. The umask command is used to apply this mask.**
* **[-S]: Displays the current mask as a symbolic value.**
* **[-p]: Displays the current mask along with the umask command, allowing it to be copied and pasted as a future input.**
* **Vim /etc/profile :- configuration file of umask for permanent changes.**
* **Source /etc/profile :- to update the database.**
* **Calculating permission according to umask.**

|  |  |  |
| --- | --- | --- |
| **User** | **Calculation umask** | **Umask permission** |
| **Root** | **Dir 777-022**  **File 666-022** | **755**  **644** |
| **Local** | **Dir 777-002**  **File 666-002** | **775**  **664** |

* **Special permission**

**Special permissions are used to perform special task by any user with giving special permission. These special permission apply on user as well as executable file.**

* **SUID(Set user identity)**

**SUID is a special type of file permission given to a file.**

**Normally in Linux / UNIX when a program runs it inherits access permission from the logged in user. SUID is defined as giving temporary permissions to a user to run a program file with the permission of the file owner rather than the user who runs it. In simple words users will get file owners permission as well as owners UID and GID when executing file or program or command .**

**Chmod u+s <filename> or <path of command>**

**Eg. chmod u+s /sbin/dmidecode**

* **SGID(Set group identity)**

**SGID bit is a set of directory and if file created in that directory by root user or local user that files get directory group ownership automatically.**

**In other words when we applied SGID to particular directory, it inherit group permission to all files and directories that will create in SGID applied directory.**

**Chmod g+s <directory name>**

**S :- it shows group has no permission of execution**

**s :- it shows permission of execution in group.**

* **Sticky bit**

1. **Only owner of particular file can access or delete or rename the files.**
2. **The sticky bit is mainly used on directory in order to avoid deletion of a folder and its contents by other user’s through they having write permissions on the folder contents.**
3. **If sticky bit is enabled on a folder , the folder contents are deleted by only owner who created them and the root user.**

**chmod o+t <filename>**

**T :- others don’t have permission to delete or rename.**

**t :- others have permission.**

* **Sudo permission**

1. **Sudo (super user do)command in Linux is genrally used as prefix of some command that only super user are allow to run.**
2. **If you prefix sudo with any command it will run that command with elevated privileges or in other words allow a user with proper permission to execute a command as another user , such as the super user.**
3. **This is equivalent of “run as administrator ” option in windows .**
4. **The option of sudo lets us have multiple administrator.**
5. **To allow user to sudo command user must be listed in “ /etc/sudoers ” file or user should belong to wheel group .**
6. **Wheel group is a default group which allow users to used sudo command .**

**sudo useradd < name >**

**eg. edit config file sudoers , edit below the root permissions enter user name and your permission to that user /sbin /dimcode and save it.**

**To check => sudo demicode**

**gpasswd -a <username> wheel:-To give user sudo command acess .**

* **ACL(Access Control List)**

1. **ACL is use to set permission over file and directory to specific user or specific group.**
2. **We can assign multiple user with different permission on same file or directory.**
3. **Access control list (ACL) provides an additional , more flexible permission mechanism for file system. It is design to assist with UNIX file permission.**
4. **ACL allows you to give permission**

**for any user group to any disc resource.**

* **getfacl :- get file access control list .**

**Syntax :- getfacl <file\_name>**

* **setfacl :- set ,remove and change access control list.**
* **The command setfacl prepare to file access control list and getfacl to get file access control list .**
* **Each file and directory in linux file system is created with a specific set of file permission for its access.**
* **Each user can have different set of file access permission**
* **Syntax to apply ACL # setfacl -m u:<user name>:<permission> <file name>**
* **setfacl -m g:<group name>:<permission> <file name>**
* **# setfacl -m o:rwx /path :- to change other user permission.**
* **getfacl <filename> :- to check acl**
* **setfacl -b <directory name> :- remove all ACL from directory.**
* **Setfacl -x u:<username> </directory name> :- remove particular user.**

***-rwsr-xr-x 1 root root 34904 Mar 12  2014 /bin/su  
  
-rwsr-xr-x 1 root root 40760 Sep 26  2013 /bin/ping  
  
-rwsr-xr-x 1 root root 77336 Apr 28  2014 /bin/mount  
  
-rwsr-xr-x 1 root root 53472 Apr 28  2014 /bin/umount  
  
-rwsr-xr-x 1 root root 66352 Dec  7  2011 /usr/bin/chage  
  
-rwsr-xr-x 1 root root 30768 Feb 22  2012 /usr/bin/passwd  
  
---s--x--x 1 root root 123832 Nov 22  2013 /usr/bin/sudo  
  
-rwsr-xr-x 1 root root 51784 Nov 23* 2013 */usr/bin/crontab***