Linux File Permissions

Muhammad Moinur Rahman

<moin@bofh.network>

File Access Rights

- Types of Users:
  - Owner
  - Group
  - All/Other
- Types of Permissions:
  - Read
  - Write
  - Execute
- Types of Files
  - Directories
  - Other files

Summary of File Permissions in LINUX

User Type	Read (r)	Write (w)	Execute (x)
User (u)	X	X	X
Group (g)	X	X	X
Others (o)	X	X	X

Directory Permissions

- read = list files in the directory
- write = add new files to the directory
- execute = access files in the directory

Determining File Access Rights

```
Type User    Group Other Links  owner  group  size   date   hour  name
d    rwx     rwx   r-x   3      bofh bofh 4096   Feb 25 09:49 directory
-    rwx     r--   r--   12     bofh bofh 4096   Feb 16 05:02 file
```

Permission Values

r	w	x	Octal Value	Meaning
0	0	0	0	No Permission
0	0	1	1	Execute-only Permission
0	1	0	2	Write-only Permission
0	1	1	3	Write and Execute Permissions
1	0	0	4	Read-only Permission
1	0	1	5	Read and Execute Permissions
1	1	0	6	Read and Write Permissions
1	1	1	7	Read, Write and Execute Permissions

Changing the Access Rights

Purpose – to set/change permissions in files

- chmod [options] octal-mode filelist
- chmod [options] symbolic-mode filelist

Options

- -R recursively process subdirectories

Values for Symbolic Mode Components

Who	Operator	Privilege
u User	+ Add Privilege	r Read Bit
g Group	- Remove Privilege	w Write Bit
o Other	= Set Privilege	x Execute/Search Bit
a All		u User's Current Privilege
ugo All		g Group's Current Privilege
		o Other's Current Privilege
		l Locking Privilege Bit
		s Set User of Group ID mode bit
		t Sticky Bit

Examples of the chmod Commands and Their Purposes

Command	Purpose
`chmod 700 *`	Sets access privileges for all the files(including directories) in the current directory to read, write and execute for the owner and provides no access or privileges to any other users.
`chmod 740 directory`	Sets access privileges for directory to read, write and execute for the owner, read-only for the group and provides no access to others.
`chmod 751 ~/directory`	Sets access privileges for ~/directory to read, write and execute for the owner, read and search for the group, and search-only permissions for the others.
`chmod 700 ~`	Sets access privileges for the home directory to read, write and execute for the owner and no privileges for anyone else. read and search for the group, and search-only permissions for the others.

Examples of the chmod Commands and Their Purposes

`chmod u=rwx files`	Sets owners' access privilege to read, write and execute for files and keeps the group's and other's privileges to their current value.
`chmod ugo-rw files` `chmod a-rw files`	Does not let anyone read or write files.
`chmod a+x files.sh`	Lets everyone execute files.sh
`chmod g=u files`	Makes files group privileges match it's owner's privileges
`chmod go= files`	Removes all access privileges for groups and others on files

Position of file type and access privilege bits for LINUX files (as seen by “ls –l” command)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				SUID	SGID	Sticky	r	w	x	r	w	x	r	w	x
Bits of File Types				Bits for Special Access Privileges			Owner's Access Privileges			Group's Access Privileges			Other's Access Privileges		

Position of access privilege bits for LINUX files as specified in the chmod command

12	11	10	9	8	7	6	5	4	3	2	1
SUID	SGID	Sticky	r	w	x	r	w	x	r	w	x
Bits for Special Access Privilege			Owner's Access Privilege			Group's Access Privilege			Other's Access Privilege		

Default File Access Rights

- umask is a bitmap which tells which permissions to deny by default on new files
- 022 = 000 010 010 (deny write for g+o)
  rwx r-x r-x (new files permissions)
- umask with no parameters returns the current mask value
- umask newmask - sets new mask
- umask command usually used in a startup file

SUID Bit

- A special permission bit that allows executable files to run using the privileges of the owner of the files rather than the user of the file
- Can be set using commands:

  chmod u+s filelist

  chmod 4xxx filelist
- Shows up in ls - l in place of the user x bit as an s if the file is executable - (rwsrwxrwx)
- Very dangerous to use

SGID Bit

- A special permission bit that allows executable files to run using the privileges of the owner's group rather than the user of the file
- Set using the commands

  chmod g+s filelist

  chmod 2xxx filelist

Sticky Bit

- A special bit that can be used as follows:
- For a file: it directs the operating system to keep the program in memory if possible after it finishes execution (Early versions of UNIX)
- For a directory: it sets it up such that only the owner of the directory can delete (or rename) files from the directory, even if other users have write privilege (tmp)
- Can be set using the chmod command using the options:
  chmod +t filelist
- Shows up in “ls –l” as a t - (rwxrwxrwt)