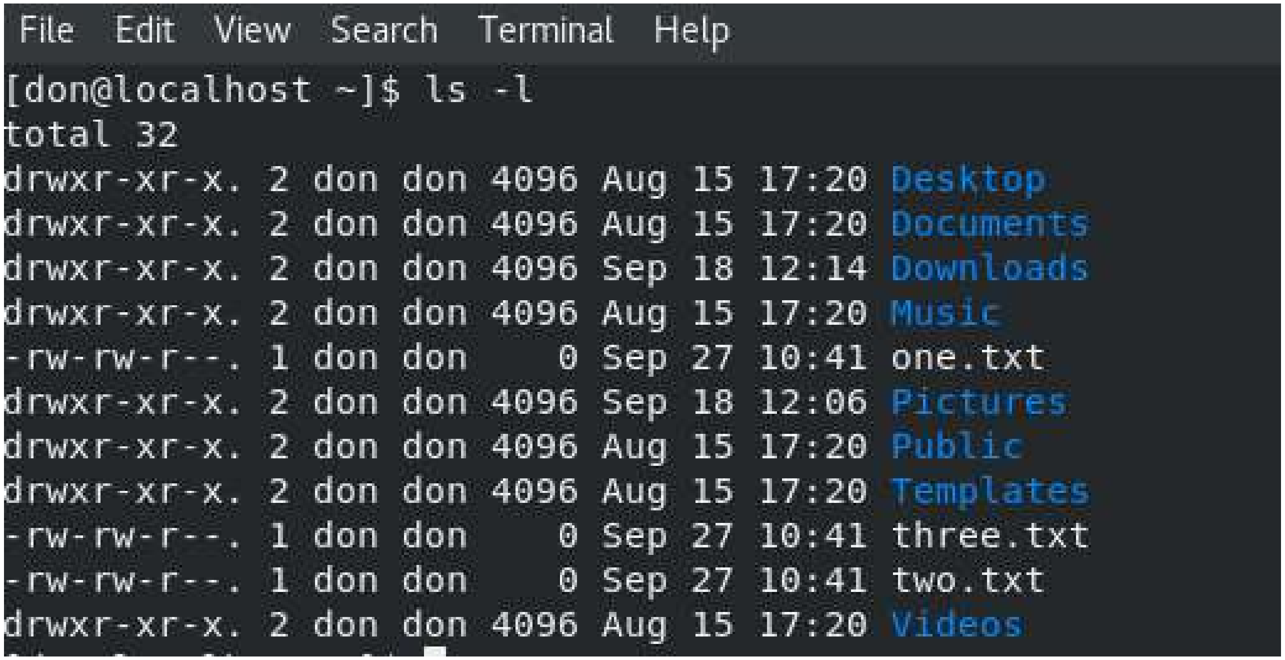
LINUX 作業系統實務
08. File Attributes

2020 TKU

Sherry Yin

ls -l: listing file attributes

$ ls -l



```
[don@localhost ~]$ ls -l
total 32
drwxr-xr-x. 2 don don 4096 Aug 15 17:20 Desktop
drwxr-xr-x. 2 don don 4096 Aug 15 17:20 Documents
drwxr-xr-x. 2 don don 4096 Sep 18 12:14 Downloads
drwxr-xr-x. 2 don don 4096 Aug 15 17:20 Music
-rw-rw-r--. 1 don don    0 Sep 27 10:41 one.txt
drwxr-xr-x. 2 don don 4096 Sep 18 12:06 Pictures
drwxr-xr-x. 2 don don 4096 Aug 15 17:20 Public
drwxr-xr-x. 2 don don 4096 Aug 15 17:20 Templates
-rw-rw-r--. 1 don don    0 Sep 27 10:41 three.txt
-rw-rw-r--. 1 don don    0 Sep 27 10:41 two.txt
drwxr-xr-x. 2 don don 4096 Aug 15 17:20 Videos
```

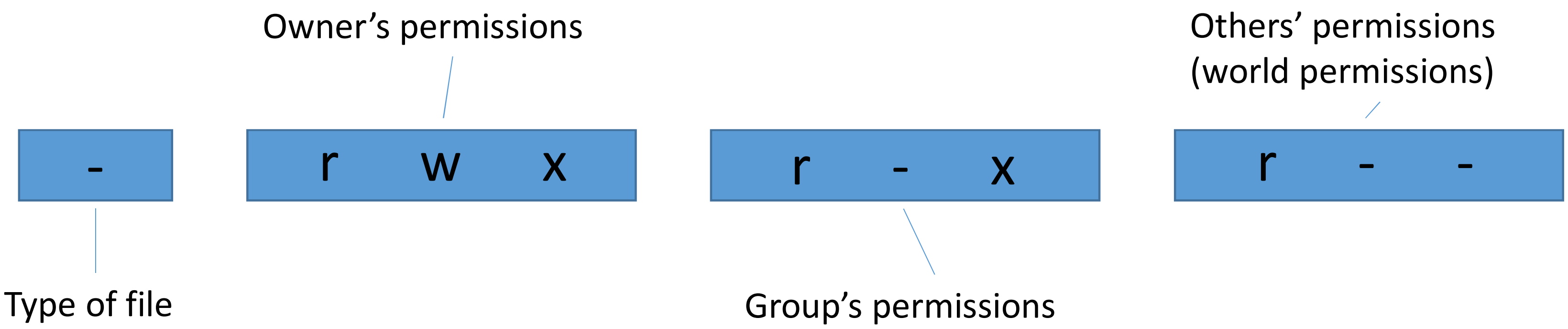
Details of the file attributes

#	Symbol	Description	Notes
1	- (ordinary file), d (directory), l (symbolic link)	Type and permissions	The remaining 9 characters form a string of permissions: r, w, x, and -
2	1, 2, …	Links, number of links associated with the file.	Directories have a link count of at least two. UNIX lets a file have multiple names.
3-4	root, romeo; metal	Ownership and group ownership	
5	13921	Size in bytes. The kernel allocates space in blocks of 1024 bytes or more. So for 163 bytes it could actually occupy 1024 on this disk	It reflects the character count and not the disk space consumption of the file.
6	Aug 10 10:42	Last modification time	The year is displayed only when the file was modified more than 6 months ago in Linux.
7	backup.sh	Filename	

-ld: listing directory attributes

- -ls (with directory name) will show the contents of the directory, not the listing with directory names
- ls -ld (with directory name): list the directories
- ls -ld: if without any specific directory name, this command will not show the listing of all subdirectories in the current directory. ls has no option to list only directories.

Structure of a file’s permissions string



Owner’s permissions

Others’ permissions
(world permissions)

- | r w x | r - x | r - -

Type of file

Group’s permissions

r: the read permission, which means cat can display the file
w: the write permission, you can edit such a file with an editor
x: execute permission
-: absence of the corresponding permission

umask: setting of a default set of permissions

- $ umask 022 (set the permission settings before creating a file)
- $ echo date > date.sh ; ls -l date.sh

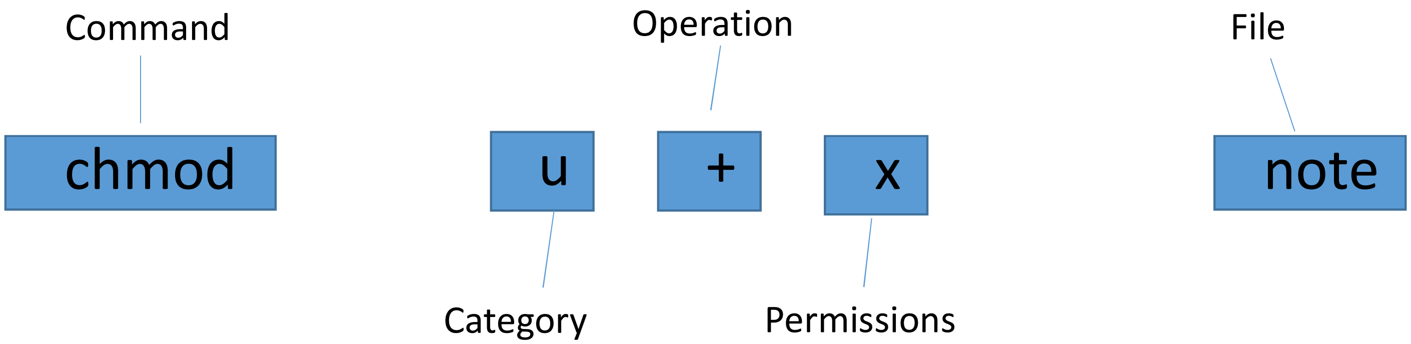
```
-rw-r—r--    1   romeo    metal     5 Aug 16 16:05     date.sh
```

chmod: changing file permissions

chmod [-R] mode file ...

- mode:
  - 1. in a relative manner by specifying the changes to the current permissions, it has 3 components:
    - User category (user, group, others)
    - The operation to be performed (assign or remove a permission)
    - The type of permission (read, write, execute)
  - 2. in an absolute manner by specifying the final permissions

Structure of a chmod Command



Command

Operation

File

chmod

u

+

x

note

Category

Permissions

Abbreviations used by chmod: relative manner

Category	Operation	Permission
u -- User	+ -- Assigns permission	r – Read permission
g -- Group	- -- Removes permission	w – Write permission
o -- Others	= -- Assigns absolute permission	x – Execute permission
a – All (ugo)	Example: $ chmod u+x date.sh (adds execute permission for user) $ chmod u-x, go+r date.sh	

Absolute assignment of chmod

- For example, if a file is to be made read-only to all:
  - $ chmod ugo=r date.sh
  - $ chmod a=r date.sh
  - $ chmod =r date.sh
- Represent the permissions each by one octal digit:
  - Read permission – 4 (Octal 100)
  - Write permission – 2 (Octal 010)
  - Execute permission – 1 (Octal 001)

Permission tables

Binary	Octal	Permissions	Significance
000	0	---	No permissions
001	1	--x	Executable only
010	2	-w-	Writable only
011	3	-wx	Writable and executable
100	4	r--	Readable only
101	5	r-x	Readable and executable
110	6	rw-	Readable and writable
111	7	rwx	Readable, writable, and executable

```
$  chmod 644 date.sh (rw-r--r--)
$  chmod 600 .netrc (-rw------- unreadable to group and others)
```

Can we delete a file with permissions 000? or prevent a file with permission 777 from being deleted? Yes, we can.

Recursive operation (-R)

- $ chmod –R : descends a directory hierarchy and applies the expression to every file and subdirectory it finds.
  - $ chmod –R a+x shell_scripts
  - $ chmod –R 755 . (works on hidden files also)
  - $ chmod –R a+x * (leaves out hidden files)

The directory

- The size of a directory file is determined by the number of files housed by it and not by the size of the files.
- $ umask 022 (read only for group and others); mkdir progs ; ls –ld progs
  - drwxr-xr-x   2   romeo   metal   512   Aug 16 09:24   progs
- Read permission for a directory means **ls** the directory contents will work. However, you can still read the files separately if you know their names.

Write permission

- For directory, write permission means you can create or remove files in it.
- Directory’s write permission off and file’s write permission on, means you cannot remove that file, but can edit it.
  - $ chmod u-w . ; ls –ld . ; rm date.sh
    - permission denied error
- Directory’s write permission on and file’s write permission off, means you can remove that file after confirmation.
  - $ chmod u+w . ; chmod u-w date.sh ; ls –ld . date.sh
  - $ rm date.sh
    - override protection 444 (yes/no)?

Execute permission

- $ cat /home/romeo/progs/date.sh
- You need to have execute permission for each of the directories in the pathname to “pass through” in searching for subdirectories.
- The execute privilege of a directory is often = the search permission
- $ chmod 666 progs ; ls –ld progs
- $ cd progs
  - bash: cd: progs: Permission denied

umask: default file and directory permissions

- UNIX has the following default permissions for all files and directories:
  - rw-rw-rw-(octal 666) for regular files
  - rwxrwxrwx(octal 777) for directories
- $ umask

  022
- http://linux.vbird.org/linux_basic/0320bash/csh/no3-8-01.html
- umask means user mask

inode (index node)

- Every file is associated with a table called the inode
- It is accessed by the inode number and contains the following attributes of a file:
  - File type (regular, directory, device, etc)
  - File permissions (the nine permissions and three more)
  - Number of links (the number of aliases the file has)
  - The UID of the owner
  - The GID of the group owner
  - File size in bytes
  - Date and time of last modification, last access and last change of the inode
  - An array of pointers that keep track of all disk blocks used by the file

inode cont.

- The inode does not store either the name of the file or the inode number (they are stored in the directory)
- $ ls -i date.sh (displays the inode number)

```
254414 date.sh
```

- The inode stores 12 permission bits, not 9. The extra three are set-user-id (SUID) and set-group-id(SGID), they determine whether a process will run with the UID and GID of the owner and group owner. The last bit is the sticky bit, mostly for directories.

How cat and ls work

- cat foo
  1. The kernel first locates the inode number of the file from the current directory
  2. It reads the inode for the file to fetch the file size and the addresses of the disk blocks that contains the file’s data
  3. It then goes to each block and reads the data until the number of characters displayed is equal to the file size
- ls – progs
  1. The kernel looks up the directory progs and reads all entries
  2. For every entry, the kernel looks up the inode to fetch the file’s attributes

ln: creating hard links

- Why is the filename not stored in the inode? – So that it can have multiple filenames (links).
- $ ln date.sh who.sh
  - The kernel adds an entry to the directory for the filename who.sh, but with the same inode number.
  - It also updated the link count in the inode from one to two
- $ ls -il date.sh who.sh
  - same information except for the filename
- rm reduce link count by one, this command is actually the system call **unlink**
- When a file’s link count drops to zero, it is completely removed from the system.

Why hard links?

1. If programs use this file in some specific directory but you want to move this file to another.
2. Provides some protection against accidental deletion
3. Save disk spaces

Symbolic link

- It is the fourth file type.
- A symbolic link is a separate file with its own inode number.
- $ ln –s date.sh date.sym
- $ ls –li date.sh date.sym
- Windows shortcuts are more like symbolic links (soft links)
- To copy a symlink, run $cp –P foo.sym foo.sym2