Estonian Information Technology College

File permissions

Operating systems I800

Edmund Laugasson
edmund.laugasson@itcollege.ee

There has been used materials from Margus Ernits, Katrin Loodus when creating current slides.

Files and folders

- What is a file?
- Information in computer will be saved as a file
- in UNIX-like systems (including Linux) also devices are files
- user-level possibilities with files (F) and directories (D)
  - creation D: *mkdir* / F: *touch* or *nano* (etc *plain text editor*)
  - changing D or F: *mv*, *cp* / F: *nano* (etc text editor)
  - removing D *rmdir* or *rm* / F: *rm*

File permissions

- has been created as a part of the OS security model
- File permissions depend on the file system
  - FAT and NTFS have their own place
  - in UNIX-like OS there are many filesystems
- file permissions are not always wanted (public share etc)
- sometimes we want things that cannot be made with file permissions (give permission for certain people etc)
- also ACL (Access Control List) exist, needs “acl” mount option in /etc/fstab

https://en.wikipedia.org/wiki/File_system
http://www.tldp.org/LDP/sag/html/filesystems.html
https://en.wikipedia.org/wiki/List_of_file_systems
https://wiki.archlinux.org/index.php/Access_Control_Lists
https://help.ubuntu.com/community/FilePermissionsACLs

File permissions 2

- The most commonly used file systems can limit the user's actions
  - r(ead)
  - w(rite) – means also deleting, changing permissions
  - e(x)ecute
- enabling and disabling will be done with setting up permissions depending on the role of the user:
  - u(ser) – file owner
  - g(roup) – user who belongs to the group of the file
  - o(ther) – any other user in the system, belongs to different group than file owner

File permissions 3

- ... are checked when opening the file
  - when file is already opened then changing permissions do not affect existing process
- file permission check will be done by operating system
  - when file is not encrypted and you can boot device from external media then file permissions do not defend

Folder permissions

- read
- add (write)
- delete
- enter („execute”)
- changing permissions

UNIX-like file permissions

- using the command *ls -l* in $HOME folder there will be the following output

```
drwxr-xr-x  18 user group  4096 Aug  3 07:00 Downloads
drwxr-xr-x   2 user group  4096 Jul 21 13:54 Public
drwxr-xr-x   6 user group  4096 Aug  8 11:03 Documents
drwxr-xr-x   2 user group  4096 Jul 21 13:54 Templates
drwxr-xr-x   7 user group  4096 Jul 22 14:06 Music
drwxr-xr-x  14 user group  4096 Jul 22 14:08 Pictures
drwxr-xr-x   2 user group  4096 Aug  5 14:26 Desktop
drwxr-xr-x   9 user group  4096 Jul 28 14:26 Videos
-rwxrwxr-x   1 user group   211 Sep 21 09:46 script.sh
```

File or folder specification is on the first column:

- **-** it is a file
- **d** it is a folder (directory)

RWX

- **rwx** in case of file
  - **R**ead – can read
  - **W**rite – can write
  - e**X**ecute – can execute (run)
- **rwx** in case of folder
  - **R**ead – can read directory content
  - **W**rite – can add, change, delete, rename
  - e**X**ecute – can enter into directory
- minus means missing of appropriate permission

owner – group - others

- in UNIX-like systems there are folders and devices actually files
- every file and folder has permission for owner, group and other users
- e.g. in case of the following file the owner have permission to read, write and execute the file
- the group have permission to read and execute
- others have no permissions at all
-
```
Owner   Group   Others
rwx     r-x     ---
```

chmod with characters

- to change permissions there is a command **chmod**
- syntax: **chmod permissions object**
- the object is a file, folder or device
- permissions
  - can be presented with character combination

```
        u
        g  +  r
chmod       -  w   file1 file2 ...
        o  =  x
        a
```

  - where: u – user (owner); g - group, o - others, a - all, r - read, w – write, x -execute
  - e.g. u+x will add execute permission for user
  - o-rwx removes read, write, execute permission from other users
- for superuser (root, UID=0) these user restrictions have no effect

chmod with octal numbers

- can be presented also numerically by using octal numbers

permissions	owner			group			others		
chmod	read	write	execute	read	write	execute	read	write	execute
0777	4	2	1	4	2	1	4	2	1
0755	4	2	1	4	0	1	4	0	1
0500	4	0	1	0	0	0	0	0	0

- syntax to change: *chmod number file*
- 0 – SetUID (will be explained later)
- $7 = 4 + 2 + 1 = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
- $5 = 4 + 0 + 1 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
- $5 = 4 + 0 + 1 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

http://permissions-calculator.org/

chmod 2

- e.g.: chmod 640 file
- every number is a sum
  - 4 reading permission
  - 2 writing permission
  - 1 executing permission
- 6 = 4+2 this means that for the file there will be given reading and writing permission for the owner
- 4 reading permission for the group
- 0 others do not have any permission at all

Numerically

```
----------   0000 no permissions at all
---------x   0001 eXecute
--------w-   0002 Write
-------wx   0003 Write, eXecute
------r--   0004 Read
-------r-x   0005 Read, eXecute
-------rw-   0006 Read, Write
-------rwx   0007 Read, Write, eXecute
--------t   1000  sticky
-----S---   2000 setgid
--S------   4000 setuid
```

chmod 3

- `chmod a-x file1 file2 ...`
- `chmod u+x file1 file2 ...`
- **u (user)** there will be user (owner) permissions set up
- **g (group)** there will be group permissions set up
- **o (other)** there will be other users permissions set up
- **a (all)** there will be all permissions set up
- chmod u+x,o-r file.txt
  - execute permission will be added to the user
  - read permission will be removed for others

chmod 4

- - (minus) removes the permission
- + (plus) adds the permission
- = (equal) sets up only these permissions
  - `chmod a=r,u=w file.txt`
  - reading for all, writing only for owner
  - `--w-r--r-- file.txt`

Special permissions setuid and setgid

- Let us see the password changing program *passwd*
- `-rwsr-xr-x 1 root root  /usr/bin/passwd`
- to change the password there are more permissions needed
- in UNIX-like systems there are more permissions in use than rwx
  - s - setUID run in owner permissions
  - s – setGID run in group permissions. In case of folder the created files there will have same permissions as group
  - t – *sticky bit* in case of folder only owner can change or delete files

setuid setgid

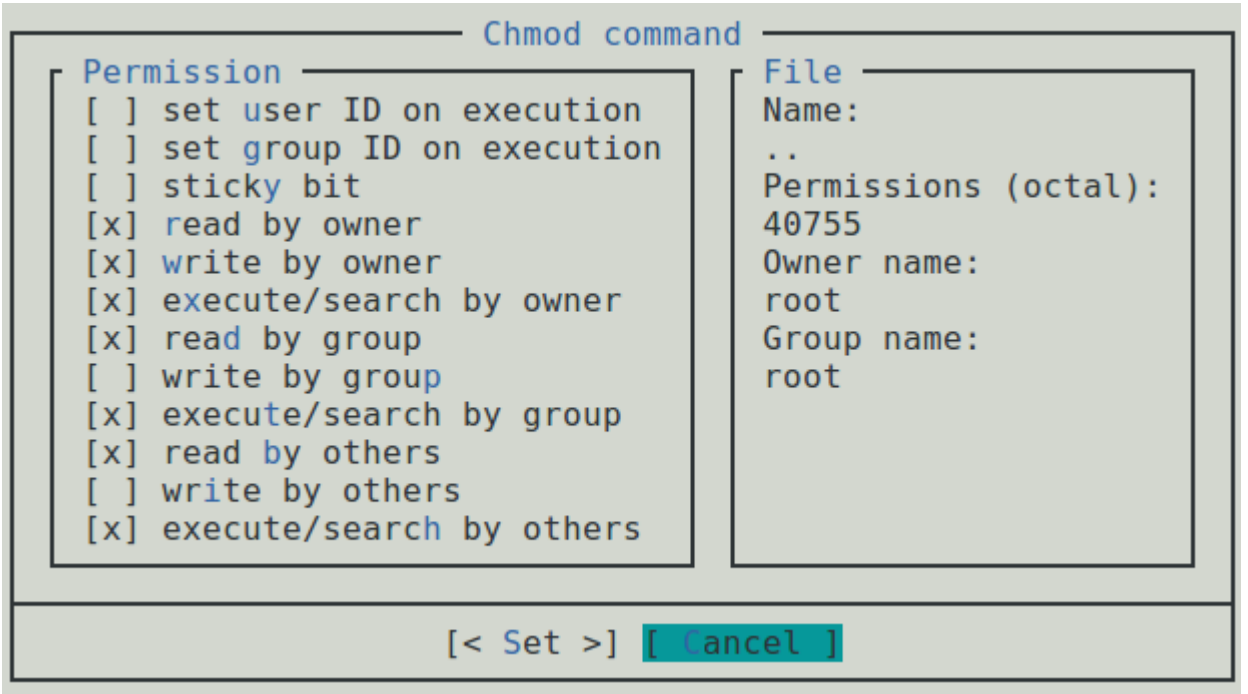
- `chmod u+s file`
  - allows by user run the program with owner permissions
- `chmod g+s file`
  - allows by user run the program with group permissions
- `chmod u+s folder`
  - nothing will happen
- `chmod g+s folder`
  - files in such folder have same group as the folder

http://permissions-calculator.org/

chmod numerically offline

```
chmod 777 file1.txt      rwxrwxrwx
chmod 755 folder         drwxr-xr-x
chmod 644 file2.txt      -rw-r--r--
chmod 4755 program       -rwsr-xr-x
```

to run program temporarily in US English:
LC_ALL=C mc
... where *mc* is the command name



When there is installed mc (Midnight Commander) then with that program there would be possible to see numbers with explanation. With space key can change the choice, with TAB and SHIFT+TAB can move between fields and buttons.

NTFS (MS Windows)

- File permissions in NTFS file system allow to give read -, write -, changing -, and other permissions
  - to multiple users
  - to multiple groups
  - for the group where some users are left out
- ACL (*Access Control List*) is connected with objects
  - specifies user/group or computer permission to specific object

Change owner and group

- To change file owner, group as superuser (root)
  - **chown** [options] user[:group] file
  - **chown** -R student:student folder/
    - -R will change recursively (including containing files and folders) the ownership as student and also group as student
  - chown student: file
    - will change ownership to student
  - chown :student file
    - will change group to student
- chgrp [parameters] group file
  - will change file, folder group

NTFS permissions in case of file

- in case of file
  - reading (R)
  - writing (W)
  - executing (X)
  - deleting (D)
  - changing permissions (P)
  - set user as owner: *Take Ownership* (O)

NTFS permissions in case of folder

- in case of folder
  - reading (R)
  - writing (W)
  - executing (X)
  - deleting (D)
  - setting up permissions (P)
  - set user as owner *Take Ownership* (O)
  - read the folder content
  - read permissions

NTFS

- allows set up special permissions
  - view attributes
  - view permissions
  - etc

permissions

- sometimes in scripts and programs there would be needed to set up for new files and folders same permissions
- this can be arranged with ***umask*** command
  - *umask* will set up permissions that cannot be in case of new file or folder
  - https://wiki.itcollege.ee/index.php/Umask
  - http://www.webune.com/forums/umask-calculator.html
- there is also *setfacl*, *getfacl*: (requires *acl* mount option in /etc/fstab and therefore uncomfortable to use and not very often used in practice)
  - `setfacl -m u:student:rw file.txt`
    Will add read and write permission for the user student
  - `getfacl file`
    retrieves the list of ACL rules
    https://help.ubuntu.com/community/FilePermissionsACLs
    https://wiki.archlinux.org/index.php/Access_Control_Lists

Searching

- *grep* will search from standard output or from file
  - *grep <search string> <location> (grep -rnw /etc/grub.d/ -e „set -e”)*
  - *grep <search string> (dmesg | grep usb)*
- *find* will search file, folder also by given attributes
  - *find <location> search parameter <search value>*
  - *find /etc/grub.d/ -type f -exec grep "set -e" {} \; -print*
- *locate* will search by name
  - *sudo updatedb* will update database in first place
  - *locate [arguments] search term*
- https://help.ubuntu.com/community/FindingFiles
- https://help.ubuntu.com/community/grep
- https://help.ubuntu.com/community/find

References

- File permissions
  - http://en.wikipedia.org/wiki/Filesystem_permissions
- in Estonian
  http://kuutorvaja.eenet.ee/kasutamine/os/failioigused.html
- http://permissions-calculator.org/
  calculate file permissions
- file permissions
  - http://catcode.com/teachmod/
  - in Estonian https://wiki.itcollege.ee/index.php/Osadmin_spikker#1.8._Faili.C3.B5igused_ja_nende_muutmine
- owner and group
  - in Estonian https://wiki.itcollege.ee/index.php/Osadmin_spikker#1.9._Failiomaniku_ja_grupi_muutmine

Questions?

Thank you for your attention!