# File Security and Permissions

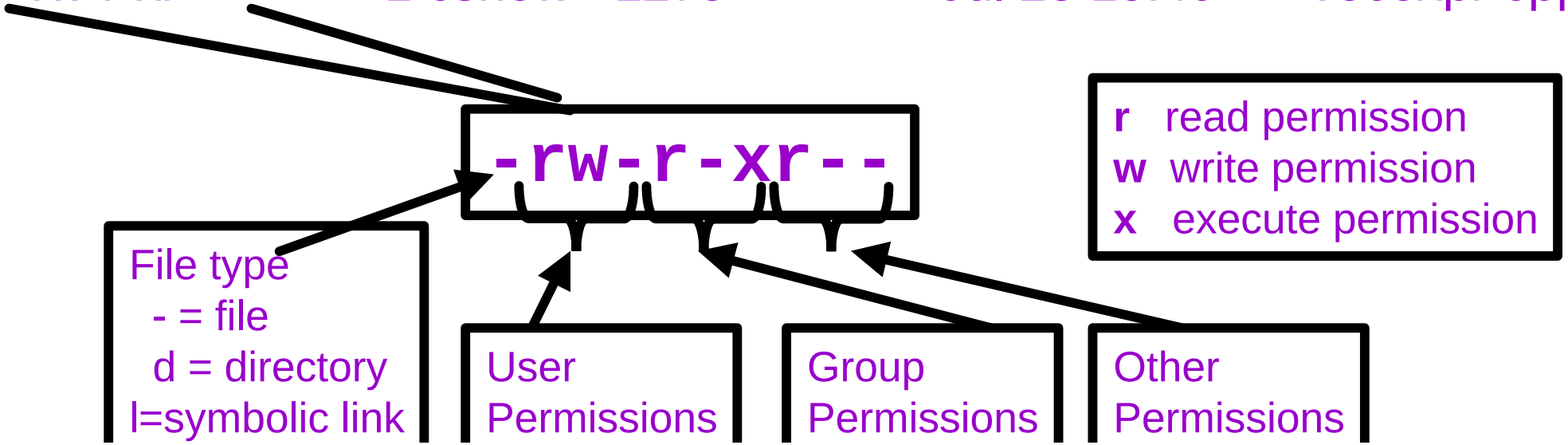
# File Permissions (1)

- With respect to a particular file, Unix divides the set of all users on a system into three categories:
  - user
    - The owner of the file.
  - group users
    - Most of you are in the group 2ndyr
    - Used for easier administration of access control.
    - Normally only the superuser can set up groups.
    - Users can be in more than one group.
  - others
    - Everyone else.

# File Permissions (2)

- Permissions can be viewed with the ls -l commanc

```
obelix[1] > ls -l
total 1247
-rw-------   1 csnow   1117   Jul 23 15:49   bad.cpp
drwx--x--x   2 csnow   2048   Jul 17 10:13   bibd/
drwxr-xr-x   2 csnow    512   Aug 27 23:18   cache/
-rw-------   1 csnow   2081   Jul 23 15:49   tst2.s
-rw-r-xr--   1 csnow   1275   Jul 23 15:49   vecexpr.cpp
```



# File Permissions (3)

- Permissions are changed with the chmod command.

- There are two syntaxes you can use:

  chmod DDD file [file ...]

  – DDD are 3 octal digits representing bits of protection

  – rwx rwx rwx can be thought of as 111 111 111 in binary

  rw- r-- r--

  110 100 100

  6 4 4 chmod 644 file

# File Permissions (4)

- chmod [ugoa][+-=][rwx] file [...]
  - This is the “symbolic” method.
  - chmod u+rwx file gives the User Read, Write, and eXecute
  - chmod g+rx file gives the Group Read and eXecute
  - chmod o-rwx file removes R, W, and X from Others
  - chmod a+x file gives All eXecute permission
  - chmod g=r file gives Group Read permission and makes sure it has nothing else

- Symbolic modes can be appended with commas
  - chmod u=rwx,g-w,o-rwx file for instance

# The umask command

- umask sets the default permissions for any file you will create

- Format is **backwards** to the chmod command
  - tells you which permissions will **NOT** be given
    - umask 077 means don't let anyone but the User do anything with my files by default

- Generally set umask once in your .cshrc file and never set it again

# Directory Permissions (1)

- Directory permissions are different from the file permissions
  - Requires execute permission to access files in the directory and its subdirectories
  - Requires read permission to list the contents of the directory (does not affect the subdirectory)
  - Requires write permission to create files in the directory (does not affect the subdirectory)

# Directory Permissions (2)

```
obelix[1] >  ls -l
drwx--x---        2048  Jul 17 10:13        bibd/
obelix[2] > ls -l bibd
-r--r--rwx         173   Jul 17 10:13        readme
```

- Files in bibd/ are accessible to user
- Files in bibd/ are accessible by name (if you know the name) for group users
- Files in bibd/ and subdirectories are not accessible to others.

# Directory Permissions (3)

- The -R option to chmod is useful when working with directories.
  - It recursively changes the mode for each chmod operanc that is a directory.
  - All files and directories would receive those permissions.
  - chmod -R a+rw dir gives everyone read and write permission to each file under dir (not execute though!!!)
  - chmod -R a+rwx dir gives the executable access to allow people to actually access the files under dir
    - Makes all files executable though ...
  - chmod -R a+rwX dir gives the executable access only to those files already executable (programs, directories, …)

# Exercise – File permission

- Create a directory dir1 in your home directory.
- Edit a file test.txt in dir1.
- Remove your own read permission of test.txt.
- *Try* to display the content of test.txt by cat.
- Remove your own write permission of test.txt
- Make some changes to test.txt with an editor and *try* to save.
- *Try* to delete the file test.txt

# Exercise – Directory Permission

- Create a directory dir2.
  - What is the permission of dir2?
  - What argument is provided to umask in your .cshrc file?
- Copy test.txt to dir2/test2.txt
- Remove your own ‘r’ permission of dir2.
  - *Try* to ls dir2.
  - cat dir2/test2.txt
  - cd dir2
  - ls
  - cd ..
- Set your own permission of dir2 to be r-x
  - cp test.txt dir2/test3.txt
  - rm dir2/test2.txt
  - edit the file dir2/test2.txt using an editor and save the changes
- Set your own permission of dir2 to be rw-
  - cd dir2
  - cat dir2/test2.txt
  - cp test.txt dir2/test3.txt
  - ‘ls’ dir2
  - ls dir2