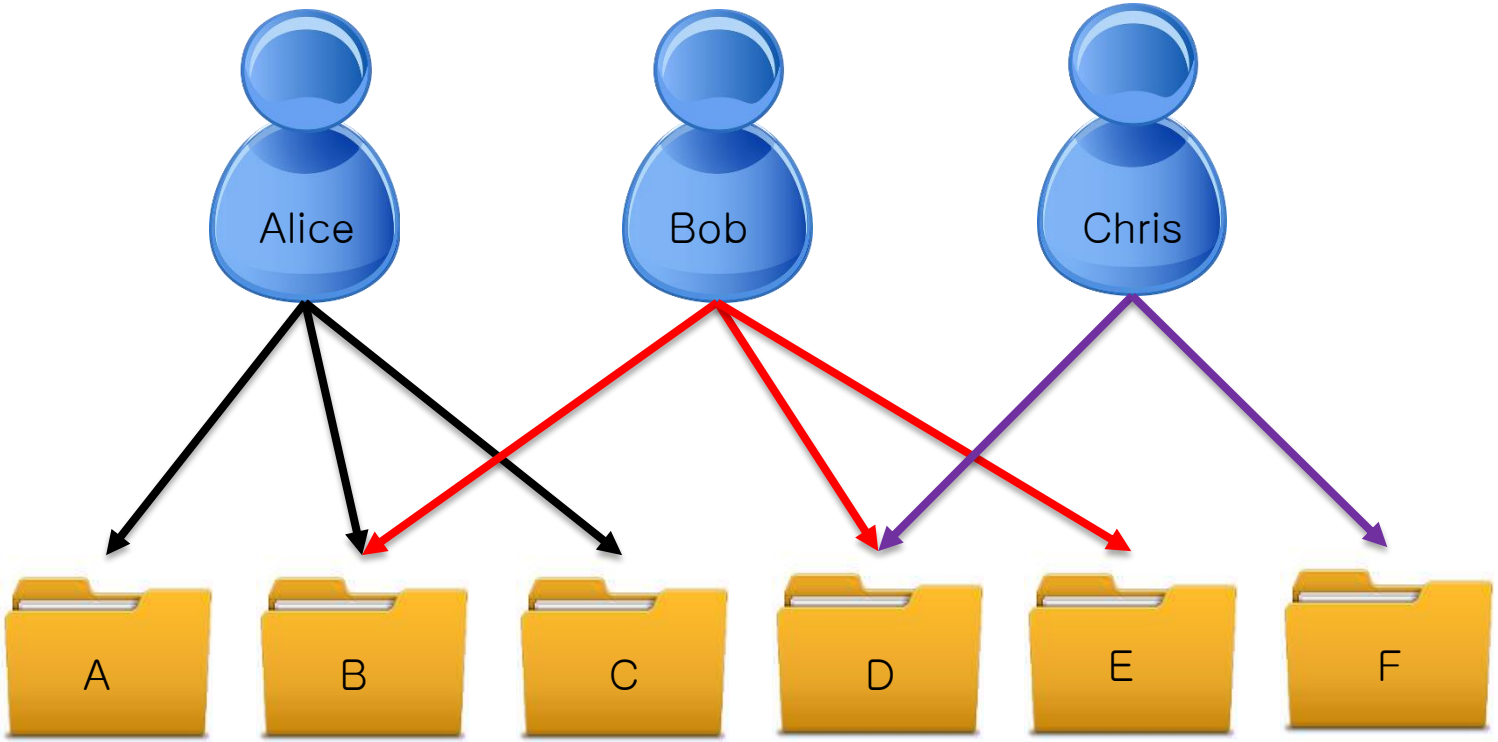
Access Control

Chonnam National University
School of Electronics and Computer Engineering

Kyungbaek Kim

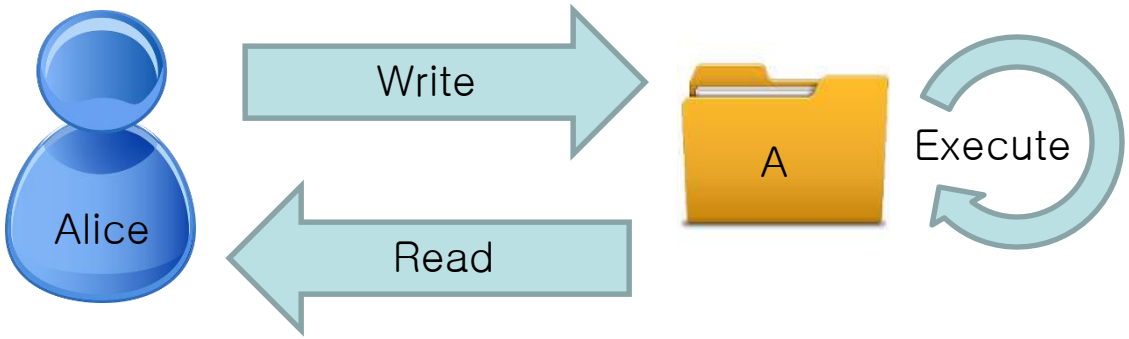
Access Control

- File Permissions
  - Hide secret contents from other users
  - Protect important contents from naïve users
  - Prevent invalid users from executing system programs
- Process permission
  - Determine user/group access privileges of a process
    - Effective UID and Effective GID


Alice
Bob
Chris
A
B
C
D
E
F

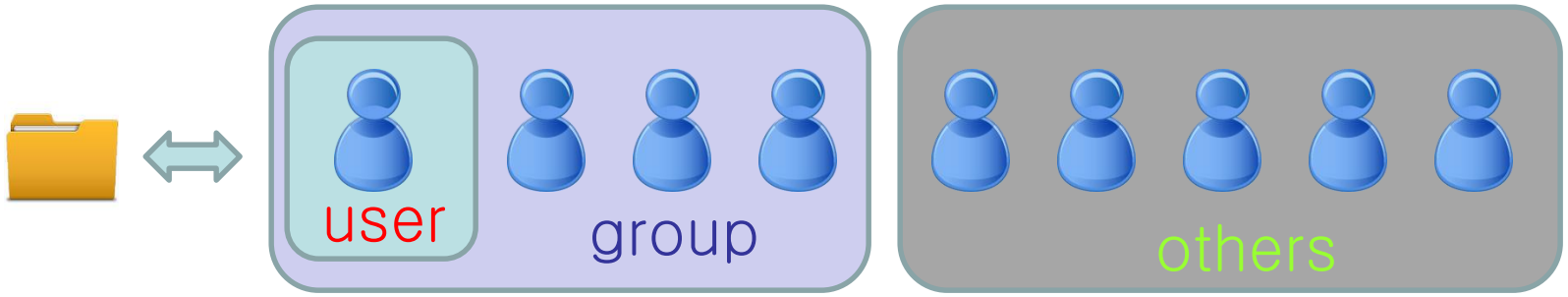
File access permissions

- There are three permissions for any file and directory
  - r → user can read a file
  - w → user can write a file
  - x → user can execute a file or enter the directory



File access categories

- Each of the tree permissions are assigned to three defined categories
  - User : the owner of the file
  - Group : the group that owns the file
  - Others : All other users



Example of Access permissions

- drwxr-x--- 10 kbkim dsm 4096 test
  - User “kbkim” owns the directory “test”
  - User “kbkim” reads/writes files under “test”
  - Users belong to group “dsm” enter the directory “test”
  - Users belong to group “dsm” can only read files under “test”
  - Other users can not access the directory “test”

Change Access Permission

- The permission of a file can be changed by using *chmod* command
  - chmod ### <file/directory name>
  - representation of #
    - Octet value (0 ~ 7)
    - r =4, w = 2, x = 1
    - Adding values according to permissions
  - E.g.) chmod 664 crisis-alert.sql

    → owner : 6 (rw-), group : 6 (rw-), other : 4 (r--)
- Only owner of the file or root can change the access permission

Octal notation of access permission

- 0 --- indicates no permissions
- 1 --x indicates execute permissions
- 2 -w- indicates write permissions
- 3 -wx indicates write and execute permissions
- 4 r-- indicates read permissions
- 5 r-x indicates read and execute permissions
- 6 rw- indicates read and write permissions
- 7 rwx indicates read, write, and execute permissions

Alternative way of chmod

- You can use “+” or “–” in combination with r, w, and x
  - to “add”(+) or “remove”(–) read, write and execute permission for a file
  - You can use “=“ for assignment
  - You can specify user (“u”), group (“g”), others (“o”) and all (“a”)
  - You can specify the operations : read (“r”), write(“w”), and execute (“x”)

Examples of chmod

- chmod +r test.txt
  - Add read permission for user, group and others
- chmod –w test.txt
  - Remove write permission from user, group and others
- chmod o+x test.txt
  - Add execute permission for others
- chmod g=rw test.txt
  - Change the permission to read and write for group
- Chmod u+x,go+w test.bat
  - Add execute permission to user, and Add write permission to group and others
- How about “chmod = test.txt” ?

Change Ownership

- The ownership of the file or directory can be changed using the command
  - **chown <owner> <file/directory name>**
- Only root can change the ownership

```
kbkim@cheetah:~/test$ ls –l
total 0
.-rw-r--r--  1 kbkim kbkim    0 2012-02-25 23:18 x
kbkim@cheetah:~/test$ chown mglee x
chown: changing ownership of `x': Operation not permitted
kbkim@cheetah:~/test$ chown dsm x
chown: invalid user: `dsm'
```

Change Group Ownership

- The group of the file or directory can be changed using the command
  - chgrp <group> <file/directory name>
- Only users included in the group or root can change the group ownership

Example of chgrp

```
kbkim@cheetah:~/test$ ls -l
total 0
-rw-r--r-- 1 kbkim kbkim 0 2012-02-25 23:18 x
kbkim@cheetah:~/test$ chgrp grads x
chgrp: changing group of `x': Operation not permitted
kbkim@cheetah:~/test$ chgrp grad x
chgrp: invalid group: `grad'
kbkim@cheetah:~/test$ chgrp dsm x
kbkim@cheetah:~/test$ ls -l
total 0
-rw-r--r-- 1 kbkim dsm 0 2012-02-25 23:18 x
kbkim@cheetah:~/test$
```

kbkim is a member of dsm

Recursive modification

- Use "–R" option for changing permission/ownership of subdirectories and files recursively
  - E.g) chgrp –R dsm dsm-work
    → change the group ownership of the directory "dsm-work"
    → change the group ownership of every contents under the directory "dsm-work"
    - Including subdirectories
    - Including files

Recursively modifying permissions of subdirectories

- To read the file entries in a directory, the directory has an executable permission and a read permission
- What happens "chmod –R u+x test"?
  - Lets assume that a directory "test" has a subdirectory "test1" and a normal file "t.txt".

  ➔ t.txt **unwillingly** acquires **an executable permission**
- Solution
  - "chmod –R u+X test"
  - X : executable or search only if the file is a directory or already has executable permission

Special file permission

- Setuid : Set user-ID on execution
  - With “s” for user, “u”
  - e.g., chmod u+s test
  - Related to processes
- Setgid : Set group-ID on execution
  - With “s” for group, “g”
  - e.g., chmod g+s group_dir
  - Related to directories/files
- Sticky bit : Restricted deletion flag
  - With “t” for others, “o”
  - e.g., chmod o+t tmp
  - Related to directories/files

setuid:special file permission

- Set user-id permission
- A process which runs from a file with setuid permission acquires the permission of the owner of the file
- Expressed with an 's' in 'user' position in a listing

```
kbkim@ubuntu:~$ ls -al /usr/bin/passwd
-rwsr-xr-x 1 root root 37140 2011-02-14 14:11 /usr/bin/passwd
```

- Enable setuid with:
  - $ chmod u+s /usr/local/bin/program

setuid and security hole

- Usually setuid is used to acquire root privilege to access the files owned by root
  - Example) passwd → needs to access /etc/shadow file to change user's password
- It means anyone can acquire root privilege during passwd is running
- It is possible to use setuid in malicious purposes
  - Such as "backdoor"

Example of setuid

peterpan@ubuntu: ~/tmp

File Edit View Terminal Help

```
peterpan@ubuntu:~/tmp$ ls -l
total 800
-rwsr-xr-x 1 root defender 818232 2012-04-01 06:59 backbash
peterpan@ubuntu:~/tmp$ id
uid=1001(peterpan) gid=1004(peterpan) groups=1001(defender),1003(neverland),1004(
peterpan),1007(family)
peterpan@ubuntu:~/tmp$ ./backbash -p
backbash-4.1# id
uid=1001(peterpan) gid=1004(peterpan) euid=0(root) groups=1001(defender),1003(nev
erland),1004(peterpan),1007(family)
backbash-4.1# chgrp root backbash
backbash-4.1# ls -l
total 800
-rwxr-xr-x 1 root root 818232 2012-04-01 06:59 backbash
backbash-4.1# chmod u+s backbash
backbash-4.1# ls -l
total 800
-rwsr-xr-x 1 root root 818232 2012-04-01 06:59 backbash
backbash-4.1# exit
exit
peterpan@ubuntu:~/tmp$ id
uid=1001(peterpan) gid=1004(peterpan) groups=1001(defender),1003(neverland),1004(
peterpan),1007(family)
peterpan@ubuntu:~/tmp$
```

setgid:special file permission

- If a directory has setgid ("set group-id") permission, files created within the directory acquire the same group ownership of the directory.
  - And directories created within the directory acquire both the same group ownership and setgid permission
- Useful for a shared directory where users of a group work on.

```
peterpan@ubuntu:~$ chmod g+s tmp
peterpan@ubuntu:~$ ls -l | grep tmp
drwxrwsrwx 2 peterpan defender  4096 2012-04-01 06:45 tmp
```

Example of setgid

peterpan@ubuntu: ~/tmp

File Edit View Terminal Help

```
peterpan@ubuntu:~$ chmod g+s tmp
peterpan@ubuntu:~$ cd tmp
peterpan@ubuntu:~/tmp$ mkdir test
peterpan@ubuntu:~/tmp$ touch a
peterpan@ubuntu:~/tmp$ ls -al
total 12
drwxrwsrwx 3 peterpan defender  4096 2012-04-01 06:49 .
drwxr-x--- 4 peterpan neverland 4096 2012-04-01 06:42 ..
-rw-r--r-- 1 peterpan defender     0 2012-04-01 06:49 a
drwxr-sr-x 2 peterpan defender  4096 2012-04-01 06:49 test
peterpan@ubuntu:~/tmp$
peterpan@ubuntu:~/tmp$
```

Sticky:special permission

- The "/tmp" directory must be world-writable, so that anyone may create temporary files within it
- But that would normally mean that anyone may delete any files within it
- A directory may have sticky permission
  - Only a file's owner or directory's owner delete it from a sticky directory
- Expressed with "t"

```
peterpan@ubuntu:~/tmp$ ls -l / | grep tmp
drwxrwxrwt  20 root root  4096 2012-04-01 07:02 tmp
```

Example of sticky

hook@ubuntu: /home/peterpan/tmpt

File Edit View Terminal Help

```
peterpan@ubuntu:~$ ls -al | grep tmpt
drwxrwxrwt 2 peterpan peterpan  4096 2012-04-01 07:07 tmpt
peterpan@ubuntu:~$ cd tmpt
peterpan@ubuntu:~/tmpt$ su wendy
Password:
wendy@ubuntu:/home/peterpan/tmpt$ echo "hello" | cat > a
wendy@ubuntu:/home/peterpan/tmpt$ exit
exit
peterpan@ubuntu:~/tmpt$ su hook
Password:
hook@ubuntu:/home/peterpan/tmpt$ ls -l
total 4
-rw-r--r-- 1 wendy wendy 6 2012-04-01 07:07 a
hook@ubuntu:/home/peterpan/tmpt$ rm a
rm: remove write-protected regular file `a'? y
rm: cannot remove `a': Operation not permitted
hook@ubuntu:/home/peterpan/tmpt$ exit
exit
peterpan@ubuntu:~/tmpt$ cd ..
peterpan@ubuntu:~$ chmod 777 tmpt
peterpan@ubuntu:~$ su hook
Password:
hook@ubuntu:/home/peterpan$ cd tmpt
hook@ubuntu:/home/peterpan/tmpt$ rm a
rm: remove write-protected regular file `a'? y
hook@ubuntu:/home/peterpan/tmpt$
```

Permissions as numbers

4000	Setuid	40	Readable by group owner
2000	Setgid	20	Writable by group owner
1000	'Sticky'	10	Executable by group owner
400	Readable by owner	4	Readable by anyone
200	Writable by owner	2	Writable by anyone
100	Executable by owner	1	Executable by anyone

chmod 1777 tempdirectory → sticky directory,
read/write/executable for all users
Chmod 4711 program → setuid program
read/write/executable for owner
executable for others

“umask” command

- When creating a new file or directory, the initial permission is given
  - 666 (rw- rw- rw-) for files
  - 777 (rwx rwx rwx) for directories
- By using umask, the initial permission of a file and a directory is given
  - File : 666 – umask
  - Directory : 777 – umask
- Example : umask 022
  - File creation : 666 – 022 = 644 (rw- r-- r--)
  - Directory creation : 777 – 022 = 755 (rwx r-x r-x)
- umask 0 : set mask to 0
- umask –S : display mask with symbols

Example

Umask Value :-

	File Permission			Directory Permission		
umask	user	group	other	user	group	other
000	rw-	rw-	rw-	rwx	rwx	rwx
002	rw-	rw-	r--	rwx	rwx	r-x
022	rw-	r--	r--	rwx	r-x	r-x
222	r--	r--	r--	r-x	r-x	r-x

File Permission :
666 – <umask value>

Directory Permission :
777 – <umask value>

Meaning of Umask