CS1101: Lecture 6
Basic File Security

Dr. Barry O'Sullivan
`b.osullivan@cs.ucc.ie`

Course Homepage
`http://www.cs.ucc.ie/˜osullb/cs1101`

Lecture Overview

- File Access Permissions;
- The Long Listing;
- Important Long Listing Information;
- Access Permissions: Files;
- Access Permissions: Directories;
- Categories of User;
- Examples;
- Changing File Permissions;
- Examples using the `chmod` command.

File Access Permissions

- UNIX is a multi-user operating system;
- Computers running UNIX are often used in a networked environment;
- Anything you can do to one of your own files you could potentially do to files belonging to another user;
- However, to prevent chaos, and to preserve privacy, UNIX allows users to restrict access to their files.
- File Access Permissions.

The Long Listing

- We have already used the `ls -l` command

```
drwxrwx---  2  you  stu  12   Apr  1  15:53  cs1101
-rwxrwx---  1  you  stu  997  Apr  1  15:54  fun
-rwxrwx---  1  you  stu  500  Apr  1  15:55  notes
```

- There are a few elements of each line that we need to be familiar with.

Important Long Listing Information

- **The File Type** – A d in the leftmost position indicates a directory. An ordinary file will have a - in this position;
- **Access Permissions** – These nine positions show who has permission to do what with the file or directory;
- **User** – This login of the person who owns the file;
- **User's Group** – A *group* is a collection of users to which the owner of the file belongs;
- **Size** – file-size in bytes;
- **Date/Time of last modification**
- **File name**

Access Permissions: Files

- The nine entries showing the access permissions deserve a closer look;
- Basically, there are 3 things that can be done to an **ordinary file** and there is a permission for each:
- **Read** – Examine (but not change) the contents of a file – Permission denoted by an `r`
- **Write** – Change the contents of a file – Permission denoted by an `w`
- **Execute** – If the file contains a program, run that program – Permission denoted by an `x`

Access Permissions: Directories

- Similarly, there are 3 things that can be done to a **directory** and there is a permission for each:
- **Read** – List the contents of the directory using the `ls` command – Permission denoted by an `r`
- **Write** – Change the contents of a directory by creating new files or removing existing files – To edit an existing file requires write permission on that file – Permission denoted by an `w`
- **Execute** – "Search" the directory using `ls` – Also, move to the directory from another directory, and copy files from the directory – Permission denoted by an `x`

Categories of User

- When deciding who can have access to a file, UNIX recognises three categories of users:
- **User** – The owner of the file for directory;
- **Group** – Other users belonging to the user's group;
- **Other** – All other users on the system;
- The first three permissions show what the **user** may do;
- The next three show what the **group** may do;
- The last three show what the **others** may do.

Example 1: Permissions

`rwxrwx---`

- The **user** has *read*, *write* and *execute* permissions only;
- The **group** has *read*, *write* and *execute* permissions only;
- The **others** have no privileges.

Example 2: Permissions

`rw-rw----`

- The **user** has *read* and *write* permissions only;
- The **group** has *read* and *write* permissions only;
- The **others** have no privileges.

Example 3: Permissions

`r--r-----`

- The **user** has *read* permissions only;
- The **group** has *read* permissions only;
- The **others** have no privileges.

Changing Access Permissions

- Access Permissions are sometimes called *modes* of the file or directory;
- To change the mode, you use the `chmod` ("change mode") command;
- `chmod` uses the following notation:

u	user (owner) of the file
g	group
o	others
a	all (owner,group,other)
=	assign a permission
+	add a permission
-	remove a permission
r	read permission
w	write permission
x	execute permission

Examples: chmod

- To give the owner execute permission without changing any other permissions, you would use:

```
chmod u+x <file>
```

- To remove read and write permissions from group members, you should use:

```
chmod g-rw <file>
```

- The following command will give everyone read permissions while removing any other permissions:

```
chmod a=r <file>
```

- To give everyone read and write permissions, you could use:

```
chmod a=rw <file>
```