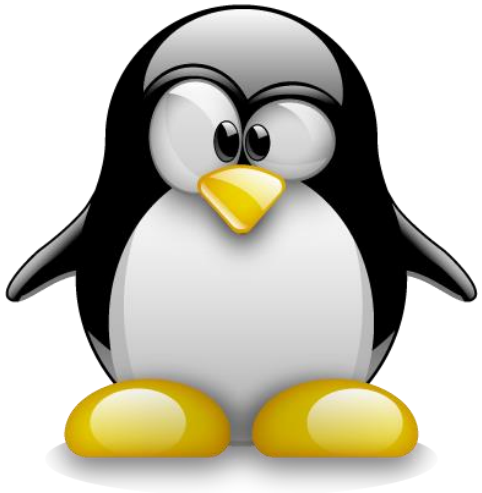
Linux File System Permissions

Outline

- File system security (Permissions)
- Changing access rights using chmod



File System Security

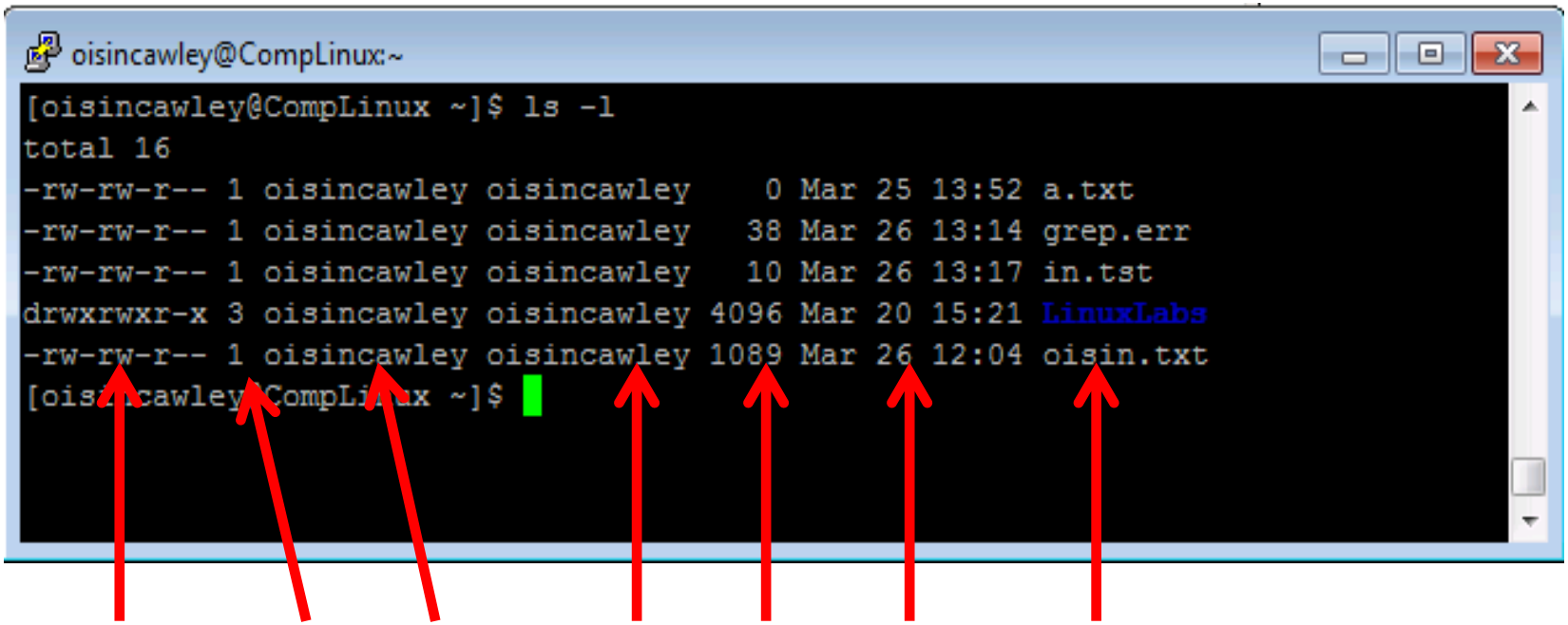
- Permissions

  - Permissions are an important feature of Linux.
    They provide security by limiting what actions a user can perform on a given file or directory.

  - Permissions control read, write, and execute access to a file for the file's owner, group and other users.
    This means that if you try to manipulate a file in a manner for which you do not have sufficient privileges, an error occurs.

File System Security

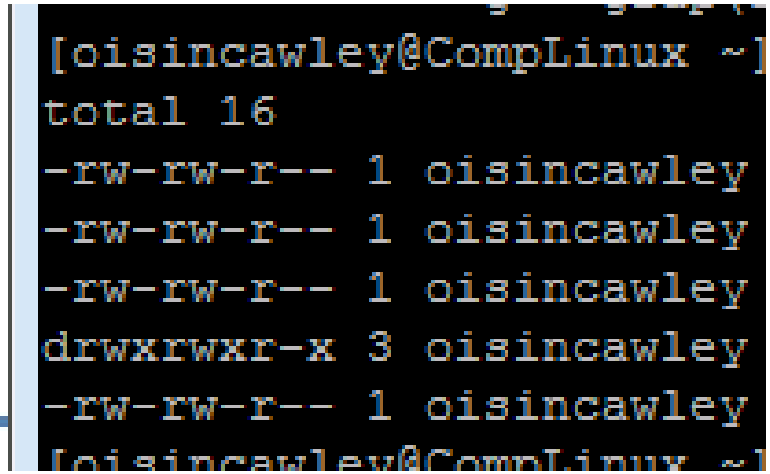
- File Permissions
  - Using the ls –l option we can see the long listing of files, which allows us to obtain more information about files in a directory.



File System Security

- Permissions Bits

In the left-hand column is a 10 symbol string consisting of the symbols d, r, w, x,
Example: `-rw-r--r--` or `drwxr-xr-x`



If **d** is present, it will be at the left hand end of the string, and indicates a directory.
otherwise **-** will be the starting symbol of the string, which indicates a file.

The 9 remaining symbols indicate the permissions, or access rights, and are taken as three groups of 3.

- The **left** group of 3 gives the file permissions for the user that owns the file (or directory)
- The **middle** group gives the permissions for the group of people to whom the file (or directory) belongs
- The **rightmost** group gives the permissions for all others ("world").

The symbols **r**, **w**, **x** have slightly different meanings depending on whether they refer to a simple file or to a directory.

See also: https://www.guru99.com/file-permissions.html

Access Rights on Files

- r (or -), indicates read permission (or otherwise), that is, the presence or absence of permission to read and copy the file
- w (or -), indicates write permission (or otherwise), that is, the permission (or otherwise) to change a file
- x (or -), indicates execution permission (or otherwise), that is, the permission to execute a file, where appropriate.
- What permissions do we have here
  - `-rwxr--r--`
  - `drwxr-xr-x`
  - `-r-x------`
  - `-rwxrwx---`

Access Rights on Files

- r (or -), indicates read permission (or otherwise), that is, the presence or absence of permission to read and copy the file
- w (or -), indicates write permission (or otherwise), that is, the permission (or otherwise) to change a file
- x (or -), indicates execution permission (or otherwise), that is, the permission to execute a file, where appropriate.
- What permissions do we have here
  - `-rwxr--r--` A File with User rwx; Grp r; Other r
  - `drwxr-xr-x` A Directory with user rwx; grp r and x; other r and x
  - `-r-x------` A File with just user having read and exec access
  - `-rwxrwx---` A File with User and Group rwx; Other has no access

Access Rights on Directories

- r allows users to list files in the directory
- w means that users may delete files from the directory or move files into it
- x means the right to access files in the directory. This implies that you may read files in the directory provided you have read permission on the individual files.

Access Rights on Directories

- So, in order to read a file,
  you must have execute permission on the directory containing that file, and hence on any directory containing that directory as a subdirectory, and so on, up the tree.
- Some examples:

-rwxrwxrwx	a file that everyone can read, write and execute (and delete).
-rw-------	a file that only the owner can read and write - no-one else can read or write and no-one has execution rights (e.g. your mailbox file).

Changing File Permissions

- **chmod** (changing a file mode)
- The default permissions for newly created files and directories are:
  - File: `-rw-rw-r--`
  - Directory: `drwxrwxr-x`

- To change access permissions on a file we need to change the file access rights.
- We do this using the **chmod** command for Changing File Permissions
- You can modify the access permissions on a file by granting or revoking (+or -) read, write, or execute (rwx) access to all, owner, group, or other users. Use with care. You can deny yourself access to your own files by setting incorrect permissions on them!

Changing File Permissions

The format of chmod is **chmod [mode] filename**

- Create **mode** by concatenating characters from ***who***, ***opcode*** and ***permission***
- **who**
  - u = owner of file
  - g = group
  - o = other users on the system
  - a = all (default)
- **opcode**
  - + = add permission
  - - = remove permission
  - = = clear permissions and set to mode specified
- **permission**
  - r = read
  - w = write
  - x = execute

Changing File Permissions

For example, to remove read, write and execute permissions on a file called **myList.txt** for the group and others, type

```
chmod go-rwx myList.txt
```

This will leave the other permissions unaffected.

To give read and write permissions on the file myList.txt to all,

```
chmod a+rw myList
```

Only the owner of a file can use chmod to change the permissions of a file.

Changing File Permissions

Can also use the bitmap, using binary.

i.e. consider `drwxrwxrwx`

We can treat the rwx permissions as bits (a 3 digit binary string).
Then set the permission for each category (user, group, other).
E.G.: chmod 705 test.txt

```
i.e.   7 in binary is 111 is all permissions set (rwx)
       0 in binary is 000 is all permissions unset (---)
       5 in binary is 101 is r and x bits set (r-x)
```

Therefore to set all permissions on for all users
You get 777 which is rwxrwxrwx

Therefore

To set rwx for all users, use	chmod 777 filename
To set rwx for just file owner, use	chmod 700 filename

Q 1 : What would chmod 555 filename do?
Q 2 : What would chmod 644 filename do?
Q 3 : What would **chmod 754 filename** do?

Summary

The **chmod** modes options are as follows:

Symbol	Meaning
u	user
g	group
o	other
a	all
r	read
w	write (and delete)
x	execute (and access directory)
+	add permission
-	take away permission

Summary

The **chmod** bitmap options are as follows:

Decimal	Binary	rwx	Permission
0	000	---	no permission
1	001	--x	execute only
2	010	-w-	write only
3	011	-wx	write and execute
4	100	r--	read only
5	101	r-x	read and execute
6	110	rw-	read and write
7	111	rwx	read, write, execute