LINUX FILE PERMISSIONS - WHO GOES WHERE

Permissions are very important. On any system they represent what a particular user can do and more importantly what they cannot do. Linux is certainly no different in that it has permissions, but it is different in the really cool way that it does it.

Where to start
When it comes to Linux systems, they place users into three broad categories - owner, group and everyone else. This relates to the owner of the file, the group of users who can access the file and everyone else. Let's take a look at how it does this, lets take a look at a detailed listing of a file and break it down into segments;

```
-rw-r--r-- 1 Laser users 5 Jan 24 16:40 test
```

The Permissions
The actual permissions are set out in as "-rw-r--r--". You will notice that there are 10 columns, they are split into the first column, the next three, the next three and then the last three. The first column is special indicator flag which can show whether it is a directory, or device, or something else - we'll go through this later. The last 9 are grouped into 3 groups of 3;

```
- rw- r-- r--
^ ^ ^
| | |
owner group everyone
```

The way the permissions are set out in each group dictates what the each class of user can do with that file. Each group has three switches which can be set, these are *rwx* ,which equates to *r*ead *w*rite e*x*ecute. If the switch exists then that right is invoked, so in our example the owner has read and write rights to the test file, while the group has read rights only and the same applies to everyone else.

Now the rights can either be expressed *symbolically* where the actual letters r, w or x or used to denote the rights, or they can be expressed in an *octal* format where the each right in the group of three is represented by a number. The octal values for the three rights are as follows;

```
r w x
^ ^ ^
| | |
4 2 1 , the octal value is the sum of the assigned rights
```

For example, our test file had the assigned rights of "-rw-r--r--", the owner group has the r and w rights assigned, the sum of the numeric value for those rights is 6. So applying that logic to the other groups, we can see that the octal value for the rights of our test file is 644.

Working with permissions
In all possibility you will need to change the permissions of a file sometime. And for that we use the *chmod* command. This command allows you to change the permissions of a file, and this can be done either symbolically or with an octal value. First we will use change the rights symbolically by adding the e*x*ecute right to each group where *o* signifies the owner rights, *g* signifies the group rights and *a* changes the rights that anyone else has to the file;

Command	*chmod o+x,g+x,a+x ./test*
Result	*-rwxr-xr-x 1 Laser users 5 Jan 24 16:40 test*

We could have also used symbolic rights allocation to change a right for all the groups;

Command	*chmod -x ./test*
Result	*-rw-r--r-- 1 Laser users 5 Jan 24 16:40 test*

And last, but not least, we could have used the octal values to change the rights;

Command	*chmod 755 ./test*
Result	*-rwxr-xr-x 1 Laser users 5 Jan 24 16:40 test*

But wait, there's more..
Sometimes you not only want to change the permissions, but you want to change the owner or group of the file. If you look at our example, you would see that soon after the rights display are two names

```
"Laser users"
```

These names are the name of the owner of the file and the group of the file respectively. Now to change these attributes, linux gives you the capability to do so with the *chown* and *chgrp* commands. Both need the what you want the owner/group to be as the first part of the command, followed by the file's name which you want to change;

Command	*chown bob ./test*
Result	*-rwxr-xr-x 1 bob users 5 Jan 24 16:40 test*

Command	*chgrp dba ./test*
Result	*-rwxr-xr-x 1 bob dba 5 Jan 24 16:40 test*

File types
Remember the first column by the display of rights, I said I would get back to it, and here we are. Well as mentioned earlier the character found in this column tells you what type of file we are looking at, see the chart below for some of the more common types;

Indicator	Definition
-	Normal file
d	Directory
l	Symbolic link
s	Socket

Well thats it, as always I hope that helped and also showed that linux does have a very simple, but powerful, rights assignment methodology. As always read the up on the other uses, and have fun fiddling.