Group Editing of Files and Web Sites

Introduction

Some research groups like to maintain a project/research group web page. We recommend that a single individual perform this task. Any member of the group may contribute and provide edits for existing documents, and add documents, but only the single responsible individual deposits these changes in the web site's directories. The other members of the group give their materials to this individual, and he or she in turn "publishes" them. This ensures the best quality and fewest problems, and is often faster.

However, not everyone wishes to follow our recommendations. This document outlines what needs to be done to permit group-editing access, and what should not be done. There are several prerequisites you need to understand how to maintain proper permissions. These are not difficult, even for those with little UNIX experience. We do include information for those that choose to eschew the more direct UNIX approach for a Windows or MacOS GUI-based approach. The UNIX-based concepts and tools you should know about are:

1. How to login to a UNIX system (rlogin, SecureShell, etc.) For more information on this, see the CNG's *Introduction to UNIX* document, http://www.seas.rochester.edu/CNG/docs/IntroUnix.html.
2. Change directory (the ***cd*** command). See the *Introduction to UNIX* document for more information.
3. Obtain and understand a directory listing (the ***ls*** command). See the *Introduction to UNIX* document for more information.
4. Understand groups and how to set/change the group of a file or directory (the ***chgrp*** command). See the *Introduction to UNIX* document for more information.
5. Understand file/directory permissions, and how to set/change them (the ***chmod*** command). More information on this can be found in the CNG's *Intermediate UNIX* document, http://www.seas.rochester.edu/CNG/docs/InterUnix.html.

These are not difficult skills to acquire. If you already have some facility with these prerequisites, you can skip down to the core material in the "Group editing of files" and/or "Work-arounds" sections below. Some of the topics are presented below in abbreviated form.

Groups

UNIX systems allow individuals to belong to a finite number of *groups*. The maximum number of groups to which one may belong varies with the particular version of UNIX. Eight (8) groups is roughly the smallest maximum number of groups to which one may belong.

Files and directories also have group membership. However, a file or directory can belong to one and only one group.

Group notation

The UNIX command '***ls***' lists directory contents. When using '*ls*' you will normally want to use the option flags '*-l*' and possible '*-g*' (usually '*-lg*'; '*ls –g*' is not useful) to show the full information about the object you are listing, e.g.,

```
42% ls -lg ECEFall03.csv
-rw-r--r--   1 gort     cngstaff    12025 Sep 15 08:40 ECEFall03.csv
43% ls -l ECEFall03.csv
-rw-r--r--   1 gort         12025 Sep 15 08:40 ECEFall03.csv
```

This shows the difference between using the '*-l*' and the '*–lg*' options to '*ls*'. However, Linux and the SysV version of 'ls' on Suns should use the '-l' only, as adding 'g' then shows only the group (no owner) information. If –lg yields only the group information, try again using '-l'.

Dissecting this a bit,

```
-rw-r--r--   1 gort     cngstaff    12025 Sep 15 08:40 ECEFall03.csv
```

we see the owner is 'gort'.

Then we can look again and see that

```
-rw-r--r--   1 gort     cngstaff    12025 Sep 15 08:40 ECEFall03.csv
```

the group membership of this file is 'cngstaff'.

Setting group membership

You may change the group membership of any file or directory you own to any group to which you belong, by using the '***chgrp***' command. For example, to change the example file from 'cngstaff' to 'staff', I would

```
chgrp staff ECEFall03.csv
```

so that the information shown by 'ls' would now look like:

```
-rw-r--r--   1 gort     staff    12025 Sep 15 08:40 ECEFall03.csv
```

Permissions

UNIX has three levels of access (*owner*, *group* and *other* or world) to files and directories. Each of these levels can be assigned different access types. For our purposes here, we need to consider *read*, *write*, *execute*, and *setgid* permission levels.

Permission notation

Permissions are symbolically displayed when using the UNIX directory lister 'ls'. Looking at our example from above

```
-rw-r--r--   1 gort     cngstaff    12025 Sep 15 08:40 ECEFall03.csv
```

we see the permissions are '-rw-r—r—'. These ten characters break down, in order, as follows:

1. An object type identifier. Here the identifier is a dash (**-**), which signifies an *ordinary file*. Other identifiers of interest to us are '**d**' for *directory* and '**l**' (that's the letter ell) for *symbolic links*.
2. Three permission markers for the owner. The three markers have a letter to indicate a permission is set, and a dash to signify the permission for that position is not set. The permissions are **r** for *read*, **w** for *write*, **x** for *execute*, and **s** for *setuid/setgid*.
3. Three permission markers – defined as above – for group access.
4. Three permission markers for everyone not the owner and not in the group to which this file belongs, i.e., 'other' or 'world' permissions.

So from above, the '-rw-r—r—' indicates that

- this is an ordinary file
- the owner (gort) can read and write to this file. It is not executable, nor is it setuid (we do not want any setuid file!).
- The group members can read this file, but they cannot write or execute this file, and it is not setgid.
- The world (those not gort and not members of cngstaff group) can read but not write to this file, nor can they execute it.

A mail directory is typically well-protected, e.g.,

```
45% ls -lgd Mail
drwx------   2 gort     cngstaff     5120 Sep 17 15:00 Mail
```

which shows that this is a directory, that gort (the owner) has read/write/execute privileges, and that group and other are shut out completely.

A web directory that is to be shared by a group needs to first of all belong to that group, and second needs to have correct permissions. "Correct" in this context is 'drwxrwsr-x'.

For example,

```
205% ls -lgd ECE221
drwxrwsr-x    2 liobe    ece221w      4096 Sep 17 14:51 ECE221
```

If this directory has any other permissions, then it will not work correctly as a group-shared web directory.

It is especially bad if the "other" has the write (w) permission on; this means anyone can edit/remove/replace anything in that directory. Not good at all, and not allowed by SEAS computing policy.

It is also bad (for the group trying to work in this directory) if either the group write (w) permission, or the setgid (s) permission are not set on shared directories. If the write permission is missing, your group members will be unable to recover from certain problems, and will not be able to create new files, or replace old files with new versions, in that directory. If the setgid permission is missing, then objects (e.g., files) created in that directory will not automatically belong to the same group as the directory. That means that at some point, those files may not be editable by other members of the group, even if the group write permission is set – which may in fact allow exactly the wrong set of people to have editing privileges. The 's' hides the fact that this directory's group permissions includes an execute permission. You don't really execute a directory, but you do need that set to be able to work in a directory. If the 'x' permission is missing, then the group members cannot work in that directory. An upper-case 'S' in the group permission set denotes that the setgid permission is present, but the execute permission is not (which is not good for shared-edit directories). Please note – do ***NOT*** apply the setgid permission to *files* – use it only with directories. Also the execute permission should be set only for directories and for files that are intended to be executed – programs and scripts. Documents, text files, images, PDF (etc.) files do not need execute permissions and should not have those permissions set.

Group Editing of Files and Websites

This is the core material for this document. Research groups and web page maintainers often work as teams. There is a need for all members of a team (or a selected subset of members) to be able to update a common set of files and directories. This is easily accomplished by carefully managing group membership of individuals and files and directories, and by carefully managing permissions of those objects. As long as all group members are careful and conscientious then file access works well and is safe. One person failing to exercise due diligence can make it difficult, cumbersome or impossible for other members to also work with the shared files.

For shared/group editing of files to work, all members of the editing team must ensure that when they work with or create files and/or directories in the shared area,

1. All directories they own in the shared area are in the correct group. If a file or directory in the shared area is incorrect, the owner of that object must use the ***chgrp*** command to correct the group membership of the object.
2. Directory permissions (for directories in the shared area) must be correct. "Correct" is defined as "**`drwxrwsr-x`**" or "**`drwxrws---`**" (the latter is more restrictive, and locks out all but group members from even looking into the directory). Note that neither "**`drwxrwSr-x`**" or "**`drwxrwxr-x`**" are correct. Deviations from the correct permissions can easily be corrected with the ***chmod***

command (on a UNIX/Linux host) if necessary. Directory permissions are even more critical than file permissions; if the directory permissions are correct, but the file permissions are wrong, there are work-arounds that can be used to apply critical updates when the file permissions cannot be corrected. But if the directory permissions (or group membership) are wrong, then you may be stuck.

3. File permissions (file files in the shared area) must be correct. "Correct" is defined as "`-rwxrwxr-x`" or "`-rwxrwx---`" for executable files, and "`-rw-rw-r-`" or "`-rw-rw----`" for non-executable files (the latter in both cases is more restrictive, preventing those not in the group from running or reading the files). This can be corrected with the ***chmod*** command.

Work-arounds (also Windows, MacOS)

As mentioned above, if you encounter problems, there may be work-arounds you can employ so that you can apply updates to the shared materials. Assume that you want to update the file "**bar**" in the shared directory "**foo**". You try to edit the file, but you get a "permission denied" message, or you try to copy a new version into place, and get a permission denied. You will find that the file either has incorrect permissions, or belongs to the wrong group (shame on the author of the original file for not making sure this was correct…). But if the directory ("folder for you MacOS and Windows users) "**foo**" (which contains the file "**bar**") is in the correct group, and has the correct permissions, then you can still apply your updates. Simply use one of the following methods (again, you only need do this if the target file has permissions that prevent you from directly editing or replacing this file):

Method 1 (dropping in a new version):

1. Rename "**bar**" to "**bar.old**" or some other appropriate name (e.g., "**bar.jones**" to indicate the owner). While you cannot alter the file, you can alter the directory, and renaming a file is just altering the directory.
2. If you are uploading a new version of bar, you can now do that (but do try to make sure that the group and permissions are correct). That's it.... you're done.

Method 2 (editing an existing version):

1. If you need to edit "bar", that is, you are not uploading a new version, then move the current version with incorrect permissions aside – i.e., rename it, and then create a copy of the existing (renamed) file, which you will own, and thus can edit. In UNIX, one would do the following. "`mv bar bar.old; cp bar.old bar; chmod 0664 bar`", which also ensures the permissions are correct. In Windows or MacOS, you can easily create a new copy of the **bar.old** file (in Windows, right click on the old version, and select '***Rename***' and change the name to **bar.old**. Then right-click that file again, and this time select "***Copy***"; then right-click again, and select "***Paste***". You will now have a file '***Copy of bar.old***', which you can then right-click, and Select '***Rename***', and type in the

new name of '**bar**'). Now you can edit your copy of "bar". You're almost done (really).

2. Decide what to do with the old version (e.g., "**bar.old**"). If your edits/updates were successful, it is probably best to delete the old version. Again, while you cannot edit the old version, you can delete it ("`rm bar.old`" in UNIX, and in Windows or MacOS, just drag **bar.old** into the Trash/Recycle bin). It might be nice to let the owner of the deleted version know that you've done this.

If the directory containing the materials is not correct (wrong permissions, wrong group membership), then you can only contact the file or directory owner and ask them to correct the permissions/group problems, or contact a system administrator for assistance. Neither offers very quick turn-around. Directory (folder) permissions are very important for group-editing.

In some cases we can create a program that traverses a directory tree correcting permissions and group membership, and does this on a schedule (e.g., every 4 hours). But this is limited to critical-need situations as it is too much of a CPU drain (slows the system down) to be used for all possible shared areas. It also requires that all areas it traverses have all files and directories with the same permissions and group membership. It is much better for individuals to exercise care when working in shared areas.

Some people will be using ftp, or UNIX-to-Windows or UNIX-to-Apple/Mac file sharing. These often allow for management of permissions and group membership; see the documentation for your ftp or file sharing client for more information on this.

One final consideration; ***please make sure your file and directory names consist only of letters a-z and A-Z (UNIX is case-sensitive) and digits 0-9***. Dashes and underscores are permitted also. Almost all other characters (including spaces and tabs) have special meanings in UNIX, and will make it difficult to properly manage your files.