Project 3 – File Permissions

UNIX File attributes

- Every file in UNIX has several attributes
  - Owner
  - Create time
  - Last update time
  - Permissions
- UNIX maintains these attributes for us
  - When you create a file, you are the owner
  - When you create a file, time/date of creation is kept
  - When you edit and save a file, last update time is updated
  - When you create a file, “default” permissions are applied

File Permissions

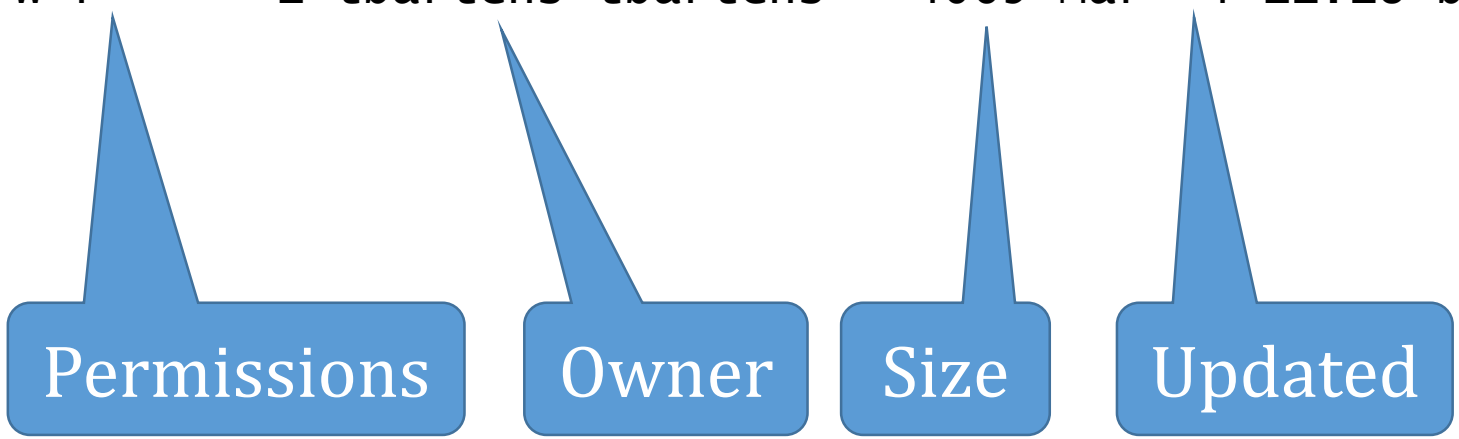
- Three separate permissions:
  - Permission to read a file
  - Permission to write to a file
  - Permission to execute a file

- Three separate audiences for those permissions:
  - User – what can you do with this file
  - Group – What can others in the same group as you do with this file
    - Groups are defined when your USERID is created by the administrators
  - Others – What can users outside your group do with this file

Listing Permissions

- Use the UNIX command “ls –l” (for “long listing”)

```
remote01:~/CS220/git/proj3-tbartens> ls -l
total 188
-rwxr-x--- 1 tbartens tbartens  28928 Mar 26 14:24 bomb
-rw-r----- 1 tbartens tbartens 108513 Mar 14 21:11 bomb_asm.txt
-rw-r----- 1 tbartens tbartens   4069 Mar  4 11:18 bomb.c
…
```



Decoding Permissions

- UNIX ls command prints out 10 characters
  - 1 for “Extended Permissions” (look it up if you’re curious)
  - 3 for “User Permissions” : rwx
  - 3 for “Group Permissions” : rwx
  - 3 for “Other Permissions” : rwx
  - r, w, or x indicates granting permission, dash (-) indicates lack of permission

```
-rwxr-x--- 1 tbartens tbartens  28928 Mar 26 14:24 bomb
```

- User may read, write, or execute the bomb
- Group members may read, or execute the bomb, but not write
- Non-group members cannot read, write, or execute the file

Changing Permission

- If you own a file, you can run the UNIX “chmod” command to change (modify) permissions
- Parameters tell chmod how to modify permissions
  - First, specify audience… u=user, g=group, o=others
    - Any combination is valid
    - No audience is assumed to be ugo (everybody)
  - Next, specify “+” (add permission) or “-” (subtract permission)
  - Then, specify permissions “r”=read, “w”=write, “x”=execute
    - Any combination is valid
  - Finally, specify the file (or files) to modify

```
>chmod go+r bomb.c
```

Execute Permission

- Without execute permission, you cannot run the binary instructions in the file

```
remote01:~/CS220/git/proj3-tbartens> chmod -x bomb
remote01:~/CS220/git/proj3-tbartens> ./bomb
./bomb: Permission denied.
remote01:~/CS220/git/proj3-tbartens> chmod u+x bomb
remote01:~/CS220/git/proj3-tbartens> ./bomb
Welcome to my fiendish little bomb. You have 6 phases
with
which to blow yourself up. Have a nice day!
_
```

Zip/Unzip and Permissions

- In UNIX, “zip” and “unzip” maintain (and recreate) permissions by default
- If you unzip outside of UNIX, permissions of stored directories are not restored!
- You may (for instance) lose the “execute” permission on a bomb executable

- Warning: GMAIL will not allow you to attach anything with “execute” permission turned on.

Line Ends

- If you use a Windows editor and transfer files from Windows to LDAP, you may have problems with line ends in your files
  - Windows editors use carriage return (/cr) line feed (/lf) to end lines
  - UNIX expects a newline (/n) at the end of a line

- On LDAP machines use…

```
dos2unix solution.txt
```

… to convert all /cr/lf into /n in the solution.txt file