Lecture # 3 – Files and Directories

- Hierarchical File Structure

  File system is basically a tree structure.
  Top node in the tree is the **root** directory (i.e. '/').
  Each directory can contain files and sub-directories.

  Files can have names containing any characters except '/' and NULL.
  Best to avoid special characters in filenames (i.e. stick to A-Z a-z 0-9 - . ,)
  Best to avoid spaces in filenames.

  Filenames were restricted to 14 characters in older versions of UNIX
  Filenames can now be fairly long (a limit >= 255 characters is not uncommon)

  Note: The UNIX directory structure is distinctly different than Windows since all disks belong to the same directory structure. When you need more space in a portion of a file system, you mount a new disk partition under a specific directory.

- mkdir – Create a new directory

  mkdir [-p] <directory name>

  <directory name> can be either absolute or relative directory name
  -p option creates intermediate directories if they do not exist

```
$  ls
file1 file2
$  mkdir newdir
$  ls
file1  file2  newdir

$  mkdir –p /u02/oradata/mydb
```

- Working Directories

  Current directory is the working directory
  pwd command will print the current directory

- Your home directory

  When you first login to the system, you can type **pwd** and to determine your home directory
  Your home directory is also stored in the HOME variable (i.e. echo $HOME)

- Absolute versus Relative Filenames

  Absolute filenames start with the root directory (/home/richj/file1.txt)
  Relative filenames start with current directory (working directory – richj/file1.txt)

- Special Files

  There are 2 special files in each directory (“.” And “..”)
  “.” (dot) represents the current directory
  “..” (dot dot) represent the parent directory

  ```
  $  cat file1
  $  cat ./file1

  $  cat ../tmp/file1
  $  cat ../../../file1
  ```

- cd – Change working directories

  cd <directory name>

  <directory name> can be either absolute or relative directory name.

  ```
  cd ..        (go to parent directory)
  cd ../home  (go up one directory, and down home)
  cd ./home   (go to the home subdirectory underneath the current directory)
  cd home     (same as last command)
  ```

  Note: The **cd** command without a directory takes you back to your home directory.

- rmdir – Remove directory

  rmdir <directory>
  Must be empty (i.e. No files or subdirectories)

  ```
  $  rmdir dir1
  $  rmdir ../tmp/dir2
  ```

- mv - Rename a file.

  mv <old> <new>

  You can move files to a new directory with “**mv** file newdir”.

- Access Permissions

  3 categories – owner (u), group (g), and other (o)
  3 permission types – read (r), write (w), and execute (x)

  Read permissions imply ability to read a file.
  Write permissions imply write and rename / delete.
  Execute permissions imply the ability to "attempt" to execute a file.

- ls –l shows file permissions

  ```
  -rwxrwxr-x    3     alex     pubs              2048    Aug 12 13:50   memo
  ```

  Columns in "ls –l" output are:

  1. File mode (combination of file type and permissions) – first character is file type, the remainder designates the file's permissions
  2. Number of hard links to the file
  3. File owner
  4. Group owner
  5. File size
  6. Modification date
  7. Filename

- chmod <permissions> <file> - Change access permissions on a file

  Read = 4
  Write = 2
  Execute = 1

  Can use letters or numbers (u+r, a-x, 777, 555, 744, etc)

  ```
  chmod 777 richj.txt                 (rwxrwxrwx)
  chmod 755 richj.txt                 (rwxr-xr-x)
  chmod 644 richj.txt                 (rw-r—r--)
  chmod 610 richj.txt                 (rw---x---)
  ```

- chown <owner> file – Change owner of a file

  ```
  chown rjohnson  richj.txt
  chown rjohnson:users richj.txt
  ```

  Note: You must be root or owner of file to use chown on a file.
  Note: On some versions of UNIX, only root can chown a file.

- Directory permissions

  Execute means search:
  ls does not work (no read permissions on ".")
  ls –ls for a particular file does work (because we are searching)
  more for a particular file does work (because we are searching)
  cd does work

  Read implies read for directory listing:

  Write means rename or delete for files in the directory (even if file permissions prevent).

  Note: 755 permissions are very common for directories.

- Links

  Hard links versus symbolic links
  Hard links are created with "ln origfile newfile"
  Symbolic links are created with "ln –s origfile newfile"
  Links can be deleted with rm just like normal files

- Find

  Used to "find" files matching specified criteria
  Very flexible; therefore, has many options (see man page for complete listing)

  Syntax: find <dir> options

  List all files starting in current directory named "file.txt"
  ```
  find . –name file.txt –print
  ```

  List all files under /usr named "ls"
  ```
  find /usr –name ls –print
  ```

  List all files under /usr owned by fred that have not been modified in last 7 days
  ```
  find /usr –user fred –mtime +7 –print
  ```

  List all files under /tmp larger than 500 blocks and perform a long listing
  ```
  find /tmp –type f –size +500 –exec ls –ls {} \;
  ```

  Determine files under /usr named core and delete them:
  ```
  find /usr –name core –exec rm –f {} \;
  ```

  Other useful options (-ctime, -atime, -perm)