0.1 Linux File System Basics

Rules for Files & Directories

- File and directory names are case-sensitive. The filenames testout.txt, TestOut.txt, and TESTOUT.txt are three different files.
- File and directory names can be made up of upper and lowercase letters, numbers, the dot (.), and underscore (_) symbols.
- The dot (.) in a filename is not necessary. However, it can be helpful to use a dot-based filename extension to identify file type.
- A directory can hold files and other directories.
- Linux directories are arranged hierarchically as parent and child directories.
- The root directory does not have a parent directory and is represented by a single forward slash (/).
- The forward slash is also used as a delimiter when showing the directory path from the root directory to a file or directory. For example: /home/rtracy/bin/myaddresses.dat.

0.1 Basic Linux Directory Navigation

Command	Description
pwd	Displays the current working directory.
cd	Changes the present working directory. **cd ..** changes to the parent directory. **cd ../..** to changes two levels up in the directory. **cd /** changes to the root directory.
ls	Displays the contents of a directory. Options include: **-a** displays all directory contents, including hidden content. -l displays extended information, including the owner, modified date, size, and permissions **-R** displays the contents of a directory and all of its subdirectories. **-d** displays directories but not files. **-r** reverses the sort order.

0.1 Basic Linux Directory Management

Command	Description
mkdir	Creates a new directory. **-p** creates all directories within the specified path that does not already exist
cp	Copies directories. Copying leaves the source contents (directories and files) intact. **-r** Recursively copies subdirectories and files within the directory **-R** " "
mv	Moves or renames directories (and files). Moving directories removes the source directory and places it in the destination. Options include: **-f** overwrites a directory that already exist in the destination directory without prompting. **-I** prompts before overwriting a directory in the destination directory. **-n** never overwrites files in the destination directory.
rmdir	Deletes an empty directory.
rm	Removes the directory and file information from the file system, which makes the directories and files inaccessible. Options include: **-i** prompts before removing. **-r** removes directories, subdirectories, and files within them. **-f** eliminates prompt for read-only files and avoids an exit code error if a file doesn't exist.

0.1 Basic Linux File Management

Command	Function
touch	If the file does not exist, touch creates a blank version of the file. If the file does exist, this command updates the file's modification and last accessed time.
file	Shows the file type. The file command is useful because Linux does not require file extensions. The file command uses file signatures in: **/usr/share/misc/magic** **/usr/share/misc/magic.mgc** **/etc/magic**
stat	Displays file or file system status. Useful for tracking changes to files or creation dates. **-f** shows file system status instead of file information. **-L** follows links to the target file before displaying information.
cp	Copies files. Copying leaves the source file intact. **-f** overwrites files that already exist in the destination directory. **-i** prompts before overwriting a file in the destination directory.
mv	Moves or renames files (and directories). Moving files erases the source file and moves it to the destination. **-f** overwrites files that already exist in the destination directory. **-i** prompts before overwriting a file in the destination directory. **-n** never overwrites files in the destination directory.
rm	Removes a file or directory. **-i** prompts before deleting the file. **--dir** removes empty directories. **-r** to remove directories and their contents recursively. **-f** option to not prompt you when there are nonexistent files or arguments used with the command. The **rm** command deletes a file or directory's inode, but it does not actually delete its data. To permanently remove data, use the **shred** command.
shred	Deletes the file and overwrites the file's data on the hard disk. The shred command is useful when deleting files that contain proprietary information or other sensitive data. **-n** specifies the times to overwrite. The default is 25 times. **-u** deletes the inode. **-v** displays the progress of the file deletion. **-z** overwrites the filename with zeros.
chattr	Modifies file attributes. A + or - is used to add or remove attributes, respectively. For example, to make a file immutable (cannot be modified), using the +i flag sets the immutable file attribute, and -i removes the immutable file attribute. **-R** recursively changes attributes of directories and their content. **-V** displays verbose output and the program version. **-f** suppresses error messages. **-p** sets the file's project number. **-v** sets the file's version/generation number.
lsattr	Lists file attributes. **-R** recursively lists attributes of directories and their content. **-v** displays the program version. **-a** lists all files in directories. **-d** lists directories like other files rather than listing their content. **-v** lists the file's version/generation number.

0.1 Common nano Commands (text editing)

Shortcut	Function
^G	Displays the help text, which includes a list of all keyboard shortcuts.
^X	Closes the current buffer or exits from nano.
^O	Writes the current buffer (or the marked region) to disk.
M-Space **Alt+Space** **Esc+Space**	Goes back one word.
^Space	Goes forward one word.
M-A **Alt+A** **Esc+A**	Marks text starting from the cursor position.
M-6 **Alt+6** **Esc+6**	Copies current line (or marked region) and stores it in cutbuffer.
^K	Cuts current line (or marked region) and stores it in cutbuffer.
^U	Uncuts (paste) from the cutbuffer into the current line.
^W	Searches forward for a string or a regular expression.
^	Replaces a string or a regular expression.

^ = Control (Ctrl)

M-' = Meta Key (Alt or Esc are substitutions)

0.1 Common vi & vim Commands (text editing)

Command	Function	Mode
vi	Starts vi. Type the command at the shell prompt.	N/A
vi [file_name]	Starts vi and immediately begins working on the named file (either a new file or an existing file). Type the vi command at the shell prompt.	N/A
Insert key **i** **s**	Enters insert mode from command mode.	Command
Esc key	Enters command mode from edit mode.	Insert/Replace
Delete key	Deletes text.	Insert/Replace
Insert key	Toggles between the insert and replace modes while in edit mode.	Insert/Replace
#[line_number]	Goes to a specific line in the document while in command mode. For example, #94 moves the cursor to line 94.	Command
dw	Cuts a whole word and trailing space.	Command
de	Cuts a whole word but omits the trailing space.	Command
d$ **D**	Cuts all text following the cursor to the end of the line.	Command
dd	Cuts a line from the text.	Command
p	Places text in memory into the document.	Command
u	Undoes the last action.	Command
O	Opens a new line above the current line.	Command
o	Opens a new line below the current line.	Command
Ctrl+g	Displays the file name, the total number of lines in the file, and the cursor position.	Command
/[term]	Searches forward for all instances of a term. Press n to go to the next term and N to go to the previous term.	Command
?[term]	Searches backward for all instances of a term. Press n to go to the previous term and N to go to the next term.	Command
yy	Copies a line of text into memory.	Command
a	Appends text after the cursor.	Command
A	Appends text after the current line.	Command
C	Changes text from the current cursor position to the end of the line.	Command
cc	Changes text of the entire line.	Command
ZZ	Saves the current file and exits vi.	Command
h	Moves the cursor one space to the left.	Command
j	Moves the cursor down a line.	Command
k	Moves the cursor up a line.	Command
l	Moves the cursor one space to the right.	Command
z	Exits without saving.	Command
:	Enters command line mode from command mode.	Command
w	Saves the current document.	Command line
w [file_name]	Names and saves the file.	Command line
w![file_name]	Overwrites the file.	Command line
q	Exits vi. This produces an error if the text was modified.	Command line
q!	Exits vi without saving.	Command line
wq or exit	Saves the document and exits vi.	Command line
e!	Reloads the file from the last saved version. This discards all edits and reloads the last saved version of the file into vi.	Command line