2.3.4 Text Editor Facts

Traditionally, many Linux system configurations are made by editing text files. Therefore, knowing how to use a text editor is an important Linux skill. This is especially important when configuring a Linux servers that are not configured with a GUI.

This lesson covers the following topics:

- The vi editor
- The nano editor
- Choosing a text editor

The vi Editor

The vi or vim editor is a utility that creates and modifies text files. It is the defacto command line text editor included with most Linux distributions.

> Most Linux distributions actually include the vim editor instead of vi. The name vim is an acronym for "Vi Improved," which indicates that vim extends the vi functionality. Due to the vim interface being virtually identical to vi, the names vi and vim have become interchangeable, and the shorter name vi is used when describing both editors.

The vi editor uses the following operational modes:

- Command mode is the initial mode vi uses when started. It provides commands that can cut and replace text. It is also the mode from which you access the other vi modes.
- Command line mode is used to load files and to save files after editing them in the file system.
- Edit mode is the mode that vi uses to write and edit text in the file. It has two operation modes:
    - Insert mode adds text between the preceding and subsequent text.
    - Replace mode overwrites subsequent text.

The table below lists some of the most common vi commands:

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	N/A

	a new file or an existing file). Type the **vi** command at the shell prompt.	
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$ or **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 current cursor position to the end of the line.	Command

cc	Changes text of the entire line.	Command
ZZ	Saves 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 save 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

The nano Editor

The nano editor is included in most Linux distributions.

- To start the editor, type **nano** at the shell prompt.
- The nano editor is simpler to use than the vi editor.
    - Common keystroke shortcuts are listed at to the bottom of the nano interface.
        - A caret (^) in the shortcut means press and hold the Ctrl key.

        - 'M-' listed in the shortcut means press and hold the Meta key. On modern keyboards, the Alt or Esc key is substituted for the Meta key.
    - Ctrl+G (listed as ^G) displays the full help text, which includes a full list of shortcuts.
    - Prompts are displayed when user input is needed.
    - The Home, End, PgUp, PgDn and arrow keys move the cursor in the same was as other common editors.

The table below lists some common nano shortcuts:

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

Choosing a Text Editor

There are a host of text editors available for the Linux platform. The vi and nano editors have proven to be the most popular, largely due to their availability. A highlight of the differences

between vi and nano may help when choosing a text editor.

Attribute	The vi Editor	The nano Editor
Availability	Included in virtually every Linux distribution	Included in most Linux distributions
Licensing	BSD License or CDDL (free and open source software)	GNU General Public License (GPL) (free software license)
Interface Complexity	Non-intuitive, especially due to the different operational modes	Intuitive, mostly due to the onscreen display of keystroke shortcuts
Feature set	Full and complex feature set	Basic feature set
Learning curve	May take a prolonged time to learn due to its complex feature set	Usually learned quickly, especially for users having experience with other editors

If you are just starting out in Linux and only need a simple editor, choose nano. If you are already proficient using the vi editor or you need its more powerful features, choose vi.