IT 240
Shell Scripting for Administrators

Chapter 6
Processing Text with sed

Stan J. Senesy
IT Program/CCS
New Jersey Institute of Technology

sed Versions

- *sed* comes in multiple flavors, each of which have peculiarities
- The most common versions are:
  - BSD sed (OS X, BSD)
  - GNU sed (Linux, Unix)
- You can check which version you have with the *$sed --version* command
- If you don't have the version you'd like, use your package manager to install it

How *sed* Works

- *sed* operates on a stream of data, usually received from the standard input
- The input to or output from *sed* may be redirected to a file if desired
- *sed* typically doesn't modify the original data, so saving the results in a file is usually necessary

sed Commands

- Some common commands for sed are:
  - ‘*d*’ - delete line(s)
  - -e filename - redirects input from a file
  - -*n* - suppresses printing of pattern space at the end of processing it’s edit commands
  - ‘*p*’ - prints the pattern space (default). When used with -*n* it suppresses duplicate lines

sed Commands and Addressing

- More sed commands:
  - *--quiet*, *--silent* - function the same as the *-n* flag
- You can add line numbers to *sed* commands to specify a particular line. For instance *sed '1d'* will delete only the first line
- The command *sed '1,5d'* will delete the first 5 lines of the input
- *sed '4,+5d'* will delete the 4th line and the next five lines

More *sed* Addressing

- You can negate addresses from deletion with the *!* symbol. For instance: *sed '1,5!d'* will delete all but the first five lines
- *sed '1~3d'* means to start deleting at 1 and delete every third line

Substitution

- You can configure *sed* to substitute text when it finds a match in the input
- The command *sed ‘s/root/toor/’* will replace the keyword root with the keyword toor for the first instance it finds.
- Make sure you don’t forget the trailing slash!
- If you want the replacement to be global, add g to the end of the string *sed ‘s/root/toor/g’*

More Substitution

- Other commands that may be used with substitution include:
    - *number* - replaces only a specific match
    - *p* - print pattern space if a substitution was made
    - *w filename* - outputs to a file if a substitution was made
    - *I* or *i* - case insensitive
    - *M* or *m* - causes ^ to match the empty string after a newline and $ to match the empty string before a newline

String Separators

- We’ve looked at using the / symbol as a string separator in our previous example, but this may not always be what is desired
- In the example *sed ‘s:/root:/toor:’* we’re looking for */root* and replacing it with */toor*. The colon is the string separator
- If you still want to use the a char that is in the string as the separator, you need to use the escape from char: *sed ‘s/ \ /root/ \ /toor/’*

Still More Substitution

- Strings may be replaced as well: *sed 's/:root user/:absolutely power corrupts/g'*
- An empty substitution string will allow the deletion of the selected string from the output: *sed 's/root//g'*
- Substitution may be performed on specific lines: *sed '10s/sh/quiet/g'*
- The same repetition rules shown in deletion apply with substitution

sed Scripts

- The *-f* command may be used to specify an input filename
- Comments in *sed* typically begin with the # symbol
- Two potential comment problems:
  - Non-POSIX implementations will have problems with the # symbol
  - If the first two characters of your script are #*n*, the *-n* option will be set

Still MORE *sed* Commands

- The insert and append commands (*i* and *a*) may be used to add text to an input stream
- *i* will put text into a file immediately, while a outputs the text after all commands
- The change command *c* replaces the current line in the pattern space with the text that you define
- Change works for the entire line

Regular Expression Addressing

- Regular expressions are (regex) are some of the most powerful scripting tools, but require a bit of effort to use
- For instance, we can remove all the comments from an input stream with the command: *sed '/^#/d'*
- In order to understand how this works, let's take a look at some common regex characters

Regex Characters

- You'll find the following characters very useful:
  - ^ - matches beginning of lines
  - $ - matches end of lines
  - . - matches any single character
  - * - matches zero or more instances of the previous character