Linux Shell Commands

Overview of a Few Common Commands

Pete Nesbitt
May 2006

Below are a few examples of Linux commands often seen in scripts. Of course they may be incorporated into scripts or used at the command line. The intention is to provide a basic example of these commands, you may find the man page, the command followed by “--help” or google useful for furthering your understanding. They are presented in no particular order.

Although these commands may appear simple in function, most have many options. The real power of scripting or complex command lines comes when these simple functions are joined with pipes and redirects which can cause the output from one command to become the input of the next. In most cases spaces have been added around the pipe symbol '|', this is for readability. Such spaces are optional and do not impact the command string.

IMPORTANT NOTE: If you try and cut and paste the examples in this document, you will probably need to manually replace the special characters since the word processor modifies them for presentation. Ones copied from the command line will be okay, but ones typed directly into this document will need to be corrected. These include, but are not limited to ' ” ` / \ |

Command: grep
Description: matches a string from input
Examples:

1) find all files that end in .tar
   ls |grep ".tar$"
2) find all lines in a file that do not contain the string “error”, regardless of case
   grep -iv error messages.log
3) match one or more strings (this can be egrep or grep -E also see fgrep or grep -F)
   egrep “123|abc|wxy” somefile.txt

Command: cat
Description: display or join test files
Examples:

1) print a file to standard output (the screen)
   cat /etc/hosts
2) join file1 and files2 saving them as file3

Linux Shell Commands

Overview of a Few Common Commands

Pete Nesbitt
May 2006

cat file1 file2 > file3

Command: sed

Description: replace string1 with string2

Examples:

1) replace all instances of New with the Old from file1 and save as file2
   sed s/New/Old/g file1 > file2
2) remove the first instance of the string **123** from a string of numbers and just print to screen
   echo 123456123 | sed s/123// (outputs 456123)

Command: tr

Description: used to translate, delete or manipulate text strings

Examples:

1) change a string from lower to upper case
   echo abcdef | tr [:lower:] [:upper:] (outputs ABCBEF)
2) remove any characters (b, c or g) anywhere in a string
   echo abcdefg | tr -d bcg (outputs adef)

Command: cut

Description: separate a string into pieces, output certain ones

Examples:

1) extract the 2nd set of column separated values
   echo “123:456:789” | cut -d: -f2 (outputs 456)
2) extract the 1st and 3rd set of space separated values
   echo “123 456 789” | cut -d' ' -f1,3 (outputs 123 789)

Command: awk

Description: manipulates fields from a string. Unlike cut, the Field Separator can be multiple characters.

Examples:

1) extract the 2nd set of values
   echo "abc123def123ghi" | awk -F123 '{print $2}' (outputs def)

Linux Shell Commands

Overview of a Few Common Commands

Pete Nesbitt
May 2006

2) extract and reorder some part of a string
   echo "abc def ghi" | awk '{print $2,"\t",$1}' (outputs def abc)

Command: sort

Description: sorts data.

Examples:

1) reverse the natural number order of a series of digits. Without the -n, values such as 161 and 18872 would be placed together in the list below. (ls is used just to get us a quick list of numbers)
   ls -l|awk '{print $5}'|sort -nr
   - outputs a vertical list:
     51200
     19349
     18903
     18872
     8993
     6607
     5810
     ...
     1368
     1345
     274
     222
     195
     161
     141

There are many more ways to use the commands shown here and many more commands available. This is simple a primer to help you start to utilize some of the power of both scripting and the Linux command line environment.

EOF