

CS-344 - Unix Operating System Fundamentals

Lecture 4 Using Basic UNIX Utilities

Based on slides created by
Dr. Bangalore for the
Spring 2005 offering of
the course

cut

- ❑ To cut out fields from each line of a file
- ❑ Each field must be separated by a delimiter
 - default delimiter is a tab
 - other delimiters use `-d<character>` (`-d' '` or `-d:`)
- ❑ Other options include:
 - By fields: `cut -f1,4 myfile`
 - By character position: `cut -c5-10 myfile`
- ❑ What do you see when you type:
 - `cut -f5,6 -d: /etc/passwd`
 - `ls -l | cut -c55-`

paste

- ❑ Puts data together by concatenating corresponding lines in specified files
- ❑ A tab character is added before the corresponding line from the 2nd file
- ❑ To change the default field separator use `-d` option (`paste -d':' file1 file1`)
- ❑ Paste can also be used to combine multiple lines in a single file using `-s` option (`paste -s -d'' myfile`)
- ❑ Multiple files can be specified as arguments for the paste utility (`paste file1 file2 file3 > newfile`)

9/15/2005

4

diff

- ❑ Compares two files and displays changes required to the first file to make it match the second file
- ❑ Lines unique to each file are marked between '`<`' and '`>`' characters
- ❑ No output is produced when the two files are identical
- ❑ Usage: `diff <options> file1 file2`
- ❑ Other useful options:
 - Ignore case: `-i`
 - Ignore all blanks: `-w`

9/15/2005

5

grep

(global regular expressions print)

- ❑ To search for a pattern in a file
- ❑ Usage: `grep <options> <pattern> <filename>`
- ❑ Each line with the matching pattern is displayed on the console by default
- ❑ To display lines not matching the specified pattern use `-v` option
- ❑ Specify pattern with in single quotes when using non-alpha numerical characters
- ❑ Other useful options:
 - Ignore case during comparison: `-i`
 - Display line number for matching lines: `-n`
 - Display only the filenames that contain the pattern: `-l` (useful when searching multiple files)

9/15/2005

6

sort (I)

- ❑ Sort all input lines based on specified order (default in ASCII order)
- ❑ Usage: `sort <options> <filename>`
- ❑ To sort:
 - Dictionary order: `sort -d myfile`
 - Ignore case: `sort -f myfile`
 - Numerical value: `sort -n myfile`
 - Reverse sort: `sort -r myfile`
- ❑ Output sent to standard output by default, use `-o` option to send output to a specified file
 - `sort myfile.in -o myfile.out`

9/15/2005

7

sort (II)

- ❑ To sort using data after specific fields use:
`sort -k n file`
- ❑ To sort using a specific field range use:
`sort +n -m file` (start after n delimiters and stop after m delimiters)
- ❑ To specify a secondary key for sorting use:
`sort +n -m +p -q file` (field range n-m for primary key, p-q for secondary key)
- ❑ A different sort order can be specified for the secondary key (see pages 265-266)
- ❑ Useful when sorting data based on a specific field and breaking ties with a secondary key

9/15/2005

8

join

- ❑ To combine two files based on a common field (input files must be sorted)
Usage: `join <options> file1 file2`
- ❑ By default all fields from both files are sent to the standard output after the common field
- ❑ To display only specific fields from each file the `-o` option can be used:
`join -o 2.2 1.2 1.1 file1 file2`
- ❑ By default field 1 is used as the common key, a different field can be specified using `-j` option:
`join -j1 n -j2 m file1 file2`

9/15/2005

9

sed

(stream editor)

- ❑ Stream editor – works on individual lines instead of reading the entire file
- ❑ Useful when working on large files or making changes from a script file
- ❑ Sample usage scenarios (output sent to standard output):
 - Replace all instances of a specific string:
`sed 's/abc/ABC/g' myfile`
 - Search specific string and then make a replacement:
`sed '/xyz/s/abc/ABC/g' myfile`
 - Delete lines:
`sed '/abc/d' myfile`

9/15/2005

10

tr

- ❑ Reads standard input, deletes or translate characters based on the input options, and displays output to standard output
- ❑ Usage: `tr <options> string1 string2`
- ❑ Examples:
 - `tr a A < myfile`
 - `tr abc XYZ < myfile` or `cat myfile | tr abc XYZ`
 - `tr -d 'xyz' < myfile`
 - `cat /etc/passwd | tr ':' ''`
 - `tr '\n' '' < myfile`

9/15/2005

11

tee

- ❑ Note that output redirection with `>` and `|` we can send output to either a file or a utility not both
- ❑ `'tee'` is used to send output to both a file and another utility (similar to a plumber's tee)
- ❑ The data is not modified in anyway by tee utility
- ❑ Example:
`ls -l | tee myfile | wc -l`

9/15/2005

12

uniq

- ❑ Uniq discards a duplicate line if adjacent lines in the input are same
- ❑ Duplicate lines in the input will not be detected if they are not adjacent
- ❑ If input is sorted only one copy of all duplicate and unique lines will be displayed
- ❑ Options:
 - To output unique lines only use `-u`
 - To output lines that have duplicates use `-d`
 - To output number of times a line is duplicated use `-c`
 - To ignore first `n` fields on each input line when comparing use `-f n`

9/15/2005

13

Miscellaneous

- ❑ Calculator
 - `bc` - uses a C language like syntax
 - `dc` - uses reverse polish notation
- ❑ `'file'` utility can be used to determine the file type

\$ bc	\$ dc
scale=10	12.0
12.0+2.1	2.1
14.1	+
\$ bc -l	p
12.0+2.1	14.1
14.1	\$

```
$ file /etc/motd
/etc/motd:  ascii text
$ file /bin/bash
/bin/bash:  ELF 32-bit MSB executable SPARC Version 1, dynamically
linked, stripped
$ file t.c
t.c:        c program text
```

9/15/2005

14
