Advanced Unix Programming

Module 02

Raju Alluri
askraju @ spurthi.com

Advanced Unix Programming: Module 2

- Unix Utilities
  - Text Processing Utilities
  - Backup and Restore related utilities

Key prerequisits

- How to make several commands work in tandem
  - The pipe feature (`command_1 | command_2`) is used to direct the output of one command as input of another command
  - The input/output redirection helps a command to read/write input/output from/to a file

    `command <infile`

    - Reading input from a file

    `command >outfile`

    - Reading input from a file

Key prerequisites, #2

- How to view the manual pages (man pages) for commands

  `man <commandname>`

  - Shows you the manual page of the command
  - Includes information about syntax, commandline options etc.
  - Usually has some examples

Text Processing in Unix

- Unix Text Processing utilities are lot more powerful than they initially appear to be
- Several test processing utilities can be used in conjunction with each other to get higher order processing

Text Processing Utilities

- cat: concatenate files and show them

  `cat <enter>`

  – Just echo all the input text back to the terminal (Hit Ctrl+d to exit)

  `cat <filename>`

  – Display contents of <filename>

  `cat –n <filename>`

  – Diplay contents of <filename> along with line numbers

- echo: show the string on standard output

  `echo "hello world"`

  – Prints "hellow world"

Text Processing Utilities, #2

- tail: show the last part of the file

  `tail filename`

  - Show the last part of the file (typically 10 lines)

  `tail -n number filename`

  - Display the last <number> of lines from the filename

- head: show the beginning part of the file

  `head filename`

  - Show the first part of the file (typically 10 lines)

  `head -n number filename`

  - Show the first <number> of lines from the filename

Text Processing Utilities, #3

- sort: sort the lines in a file

  ```
  sort filename
  ```

  - Sort the lines of the file and print them

  ```
  sort -r filename
  ```

  - Sort the lines of the file and print them in reverse order

  ```
  sort -u filename
  ```

  - Sort the lines of the file and print only first of the similar lines

- nl: number lines in a file

  ```
  nl filename
  ```

  - Number the lines in filename and print them

Text Processing Utilities, #4

- wc: word count (count words, lines and characters)

```
wc filename
```

  - Print the number of words, lines and characters in filename
- uniq: print only the first occurance of the similar lines from a sorted file

```
uniq -c filename
```

  - Print the lines and give count (-c)

Text Processing Utilities, #5

- grep: match given pattern in a file

```
grep pattern filename
```

  - Look for pattern in filename and print all the lines that match with the pattern
  - Other options
    - -c: just print the count of the matches, but not the lines that match
    - -v: invert the pattern match: print only the lines that DON'T contain the pattern
    - -i: case insensitive: ignore case of the letters in the pattern

Text Processing Utilities, #6

- egrep: match extended patterns
- fgrep: Take multiple patterns, each separated by a new line

Text Processing Utilities, #7

- cut: cut selected fields from a file and print
  - 3 variants

```
cut -b5-10 filename
```

  - Print bytes 5-10 from each line of the filename

```
cut -c1-8 filename
```

  - Print characters 1-8 from each line of the filename

```
cut -f1,3 [-d delim] filename
```

  - Print 1st and 3rd field of each line. The field separator is TAB (or delim if given)

Text Processing Utilities, #9

- join: join fields from multiple files

```
join file1 file2
```

  - Join fields from file1 and file2, taking the first field as the key and whitespace as the field delimiter

- tee: read from standard input and send it to both stdout and the given files

```
tee file1
```

  - Reads from standard input and writes it to standard output and file1

```
grep -i student file1 | sort -u | tee file2
```

  - What does the above command do?

Text Processing Utilities, #8

- paste: concatenate lines from multiple files and print

  ```
  paste file1 file2
  ```

  - Concatenate lines of file1 and file2
  - Line from file1 will be added with a tab and a corresponding line from file2

  ```
  paste -d<chars> file1...
  ```

  - Concatenate lines from file1... by using <chars> as delimiters

Text Processing Utilities, #10

- more: page through text files

```
more file1
```

  - Show contents of file1 page by page on the terminal
    - <enter> takes you to next line, <space> takes to next page etc.

```
more file1 file2 file3
```

  - Page through each file, with a banner before each one

- pg: CRT handling of the files
  - Almost similar to more. Displaying is done slightly different.

```
pg file1 file2 file3
```

  - Gives an idea of how it is different from more

Text Processing Utilities, #11

- cmp: compare two files

```
cmp file1 file2
```

  - Compare file1 and file2, print the byte number and line number of the first ever difference

- diff: show the difference between two files

```
diff file1 file2
```

  - Shows how different the contents of file2 are from file1
    - Output with > indicates added lines
    - Output with < indicates deleted lines

Backup and Restore

- Backup to tapes has been a very common practice right from the early days of Unix
- Tape access is very sequential in nature

Backup and Restore Utilities

- tar: tape archive
  - 3 major variants

```
tar -c tarfile files/dirs
```

  - Copy files or directories into an archive "tarfile"

```
tar -t tarfile
```

  - List contents of the tarfile

```
tar -x tarfile files/dirs
```

  - Extract files or directories from the tarfile
  - Other options
    - -r: Append files at the end of archive
    - -u: Only append files that are newer than ones from archive

Backup and Restore Utilities, #2

- cpio: copy in and out of archives
  - 3 major variants

```
cpio –o ....
```

  - Take the list of files from standard input, prepare an archive and send it to the standard output

```
cpio –i ....
```

  - Take the archive from the standard input, write to files in the current directory

```
cpio –p ...
```

  - Pass Through: Combination of -o and -i; prepare an archive from first directory and dump that structure into second directory

Patterns

- Special Characters and their meaning
  - * indicates 0 or more occurances of preceding character
  - + indicates 1 or more occurances of preceding character
  - {n} indicates exact n matches of preceding character
  - {n,} indicates n or more matches
  - {n, m} indicates n or more, but less than m matches
  - [<characters>] indicates one of <characters> to match
    - "[Ss]tudent" matches with "Student" or "student"
  - - indicates a range of characters
    - [a-f] indicates abcdef
    - [0-9a-f] indicates hexadecimal digits
  - Special: [:alpha:], [:digit:], [:alnum:], [:lower:]