Advanced System Utilities of Linux

CSC 1153 – LABORATORY ASSIGNMENTS

K.D.C.G Kapugama
Department of Computer Science
Faculty of Science
University of Ruhuna

Grep & Find

- grep : “general regular expression parser”
- grep searches text files for the occurrence of a specified regular expression and outputs containing a match to standard outpu.
- The grep filter is famous among Unix / Linux

Common grep Command Options

- `grep [options] pattern [files]`
  - E.g. grep -i ‘hello world’ file.txt
- Options:
  - -c Display the number of matched lines.
  - -h Display the matched lines, but do not display the filenames.
  - -i Ignore case sensitivity.
  - -l Display the filenames, but do not display the matched lines.
  - -n Display the matched lines and their line numbers.
  - -v Display all lines that do NOT match.

grep and Regular Expression

- **^ (Caret)** : match expression at the start of a line, as in ^A.
- **$** : match expression at the end of a line, as in A$.
- **** : turn off the special meaning of the next character, as in \^.
- **[ ]** : match any one of the enclosed characters, as in [aeiou].
  - Use Hyphen "-" for a range, as in [0-9].
- **[^]** : match any one character except those enclosed in [ ], as in [^0-9].
- **. (dot)**: any single character
- *** :** zero or more occurrences of the previous character.
- **.* :** Nothing or any number of characters.

grep and Regular Expression

- **Examples.**
  - `grep bob files`
    - Search files for lines with ‘bob’
  - `grep ‘^bob’ files`
    - ‘bob’ at the start of a line
  - `grep ‘bob$’ files`
    - ‘bob’ at the end of a line
  - `grep ‘^bob$’ files`
    - lines containing only ‘bob’

grep and Regular Expression

- **Examples.**
  - `grep '\^b' files`
    - search files for lines with '^b'
  - `grep 'B[oO][bB]' files`
    - search for BOB, Bob. BOb or BoB
  - `grep '^$' files`
    - search for empty lines
  - `grep '[0-9][0-9]' files`
    - search for pairs of numeric digits

find command

- The find program searches a given directory (and its subdirectories) for files based on a variety of attributes.
  - `find [path] [options] [expression]`
- Examples
  - `find ~` : List the home directory
  - `find ~ -type d` : search the directories in the home directory
  - `find ~ -type f` : search the regular files in the home directory
  - `find ~ -name "*.txt"` : `-name` followed by regular expression with wildcards patterns

find command

- The find program searches a given directory (and its subdirectories) for files based on a variety of attributes.
  - `find [path] [options] [expression]`
- Examples
  - `find ~ -size +1M` : Search for files larger than the specified number
  - `find ~ -cmin -10` : Search for files last modified less than 10 minutes ago
  - `find ~ -mtime 10` : Search for files last modified n days ago.

Process Management in Linux

- **Process:** The running instance of a program

- **Associated commands:**
  - **top :** provides a real-time view of the system and only shows the number of process that fits on the screen.
  - **ps :** shows the processes running in the current terminal.
    - **ps aux :** shows a comprehensive list of all processes running o the system.

Background and Foreground Processes

- **Foreground processes:**
  - These processes are user dependent. If the foreground processes are run from the terminal, the shell prompt remains unavailable and will be available only when the foreground process is terminated or stopped.
- **Background processes:**
  - These processes are usually run independently from the user. If a background process is started from a terminal by the user, the shell prompt remains available. One must append **&** a symbol after the process name in the terminal to start it in the background.
  - E.g.: `gedit &`

Background and Foreground Processes

- **Placing a foreground job into a background job**
  - press `ctrl+z` to free the terminal (Suspending the job)
  - Use `bg` command
- **Bringing a background job to the foreground**
  - `fg %[job id]`
- **Viewing suspended jobs:** `jobs`
- **Killing a process**
  - `kill -9 [pid]`
  - `ctrl+c`

Shell Environment Variables

- Variables specific to a certain environment.
- Linux by default sets many environment variables for you
-
  - **env** : print a list of environment variables
  - Use **echo** to print value of an environment variable.
    - `echo $[variable_name]`

Shell Environment Variables

- Some useful environment variables
  i. HOME : The home directory of the current user.
  ii. SHELL : The path of the user’s current shell
  iii. USER : Current user
  iv. PWD : Current working directory
  v. PATH : The search path for commands.
  vi. PS1 : Shell prompt

Compressing Files in Linux

- **tar** : tape archive.
- Create compressed and archived files.

```
tar [options] [archive-file] [file or directory to be archived]
```

Compressing Files in Linux

- **Creating an archieve**
  - Example : archieving `file 1` and `file 2` to test. tar
  - `tar –cvf test.tar file1 file2`
    - c : create a new .tar archieve file
    - v : verbose show the tar file progress
    - f : file name of the archieve file.

Compressing Files in Linux

- **Creating a gzip archieve file**
  - `tar -cvzf test.tar.gz file1 file2`

    OR
  - `tar -cvzf test.tgz file1 file2`

Compressing Files in Linux

- **Extracting files from Archieve**
  - Use `x` option
    - `tar -xvf test.tar`
    - `tar -xvzf test.tgz`
  - Extracting to a specified directory : use `C` option
    - `tar -xvf test.tar -C temp` : Extract the files to temp directory.
  - Extracting a single file in from a tar file.
    - `tar -xvf test.tar file1`

Executing Multiple Commands

- **cmd1; cmd2; cmd3 :** executing cmd1, cmd2 and cmd3 sequentially.
- **cmd1 || cmd2 :** execute cmd2 only if cmd1 fails
- **cmd1 && cmd2:** execute only if cmd1 success