CST 8102

Operating System Fundamentals (GNU/Linux)

What is a shell?

- A shell is simply a command interpreter that executes commands.
  - As a command interpreter, the shell provides the user interface to the rich set of utilities.
- A shell is also a programming language.
  - The programming language features allow these utilities to be combined
- Shells may be used interactively or non-interactively.
  - In interactive mode, they accept input typed from the keyboard. When executing non-interactively, shells execute commands read from a file.

What is BASH?

- BASH is an acronym of Bourne-Again Shell, referring to its initial conception, the Bourne shell (sh)

- It can be run on most Unix-like operating systems. It is the default shell on most systems built on top of the Linux kernel

- More information about BASH
  - Initial release: June 7 1989
  - written in C
  - Original author: Brian Fox
  - License: GPL

Shell

- To find all available shells in your system type following command:
  cat /etc/shells

- To find your current shell type following command
  echo $SHELL

- Shell builtins
  - A builtin is a utility that is built into a shell. Each of the shells has its own set of builtins.
  - A shell does not create a new process when you execute builtins, so they run more quickly.
  - The ***help*** bash-builtin command will list all builtins.

Basic Linux commands

- ***rmdir command***
- Remove empty directory (*only if empty*)
- Options: -p
  - Remove directory and its ancestors.
  - ***mkdir –p a/b/c ;rmdir -p a/b/c***

- ***rm command***
  - Remove file(s) and/or directories
    - ***rm file1 file2 file3***
  - Options:
    - **-r:** remove directories and their contents recursively
      - ***rm -r dir1***
    - **-f:** ignore nonexistent file, never prompt

Basic Linux commands

Some basic/useful commands:

- Change password

  passwd

- Find out who you are

  whoami

- What is the host name

  hostname

- Name of the operating system that is running

  uname

- Switch users

  su

Basic Linux commands

ls command

- display directory content
- can customize look & feel of directory listing based on options selected
  - Options: **-l, -a, -d**
    - `ls -la`
    - `ls -ld`

cd command

- Change to a directory: `cd /root`
- If used by itself, defaults to account's home directory: `cd`

Basic Linux commands

***pwd* command**

- Lists current working directory with full path

```
pwd
```

***mkdir* command**

- Create a directory

```
mkdir test
```

- Create both parent and child directories

```
mkdir -p parent/child
```

Basic Linux commands

***rmdir* command**

- Remove empty directory (*only if empty*)

```
rmdir parent/child
```

- Remove empty parent/child directories

```
rmdir -p parent/child
```

***more* command**

- Display text one screenful at a time

```
ls -al  /etc | more
```

Get Help, Find documentation

man** [options] **command

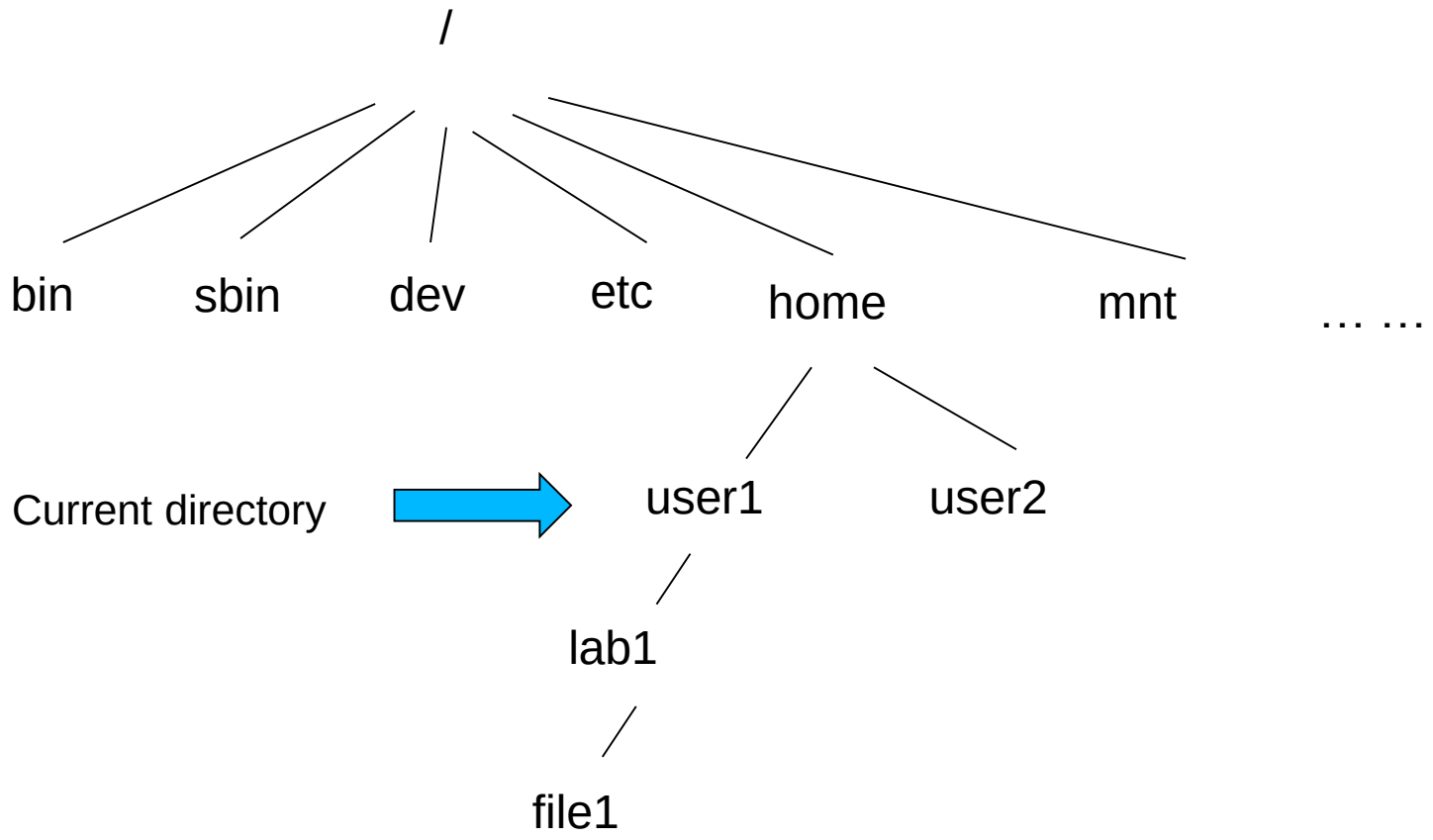
- Displays manual pages from system documentation
- Examples:
  - ***man ls***
  - ***man -a command***
    - display all man pages, not just the first one found
    - Press "*q*" to quit

info** [options] **command

- read, format & display info document for command
- slightly less formal than ***man*** in explanations

http://teaching.idallen.com/cst8207/16f/notes/140_man_page_RTFM.html

Absolute Path vs. Relative Path



/home/user1/lab1/file1	vs.	Lab1/file1

Absolute Path vs. Relative Path

Absolute path

- Also called full path, always starts with /, the name of the root directory
- Pathname of a file is built by tracing a path from the root directory through all the intermediate directories to the file
- It works no matter what the current directory is
- ~ (tilde): absolute path of current user's home directory

Relative path

- Traces a path from the current working directory to a file.
- When using a relative pathname, you need to know which directory is the current working directory
- Command used to find current working directory "*pwd*"

Absolute Path vs. Relative Path

- Two special operators used: " **.** " And "**..** "

"**.**" stands for current directory

"**..**" stands for parent directory

Examples:

- Login as root, and the current working directory is ***/home/user1***

`mkdir test` *(relative path, creating /home/user1/test)*

`mkdir /home/user1/test` *(absolute path)*

Absolute Path vs. Relative Path

- **`mkdir ./test`** *(relative path, creating /home/user1/test)*
- **`mkdir /home/user1/test`** *(absolute path)*

- **`mkdir ../test`** *(relative path, creating /home/test)*
- **`mkdir /home/test`** *(absolute path)*

- **`mkdir ../../test`** *(relative path, creating /test)*
- **`mkdir /test`** *(absolute path)*

teaching.idallen.com/cst8207/16f/notes/160_pathnames.html

Basic Linux commands

- ***cp command***
  - copy file(s) or directoires from one location to another
- ***cp file1 file2 /home/friend***
  - Options:
    - **-i:** prompt before overwrite
    - **-b**: make a backup of each destination file
    - **-u**: copy only when source file is newer or
      destinatin file is missing
    - **-r:** copy directories recursively
      - ***cp -r ~/dir1 ~/dir2***
    - **--parent:** append source path in destination directory
      - ***cp --parent /etc/passwd ~/***

Basic Linux commands

- ***mv command***
  - move file(s)/dir(s) to someplace different
    - ***mv file1 /dir1***
  - rename file(s)
    - ***mv file1 file2***
  - Options: ***-i, -b, -u***

- ***cat command***
  - Display text files to screen
    - ***cat /etc/fstab***
  - Concatenate files
    - ***cat /etc/fstab /etc/passwd***

Basic Linux commands

- ***touch* command**
  - Create a new file
    - `touch file1`
  - Change existing file timestamps

- ***tree* command**
  - List contents of directories in a tree-like format
  - Options: **-a, -d, -L**
  - Not available by default, to have it installed:
    - `sudo apt-get install tree`

Output Redirection

- Output Redirection Operator (>)

  ls > file1 (Redirect output to a file)

  cat /etc/fstab > file2

  (Redirect output to a file)

  A > B write file A into file B

- Append Redirection Operator (>>)

  Lets you redirect a output to the end of an existing file

  ls >> file3 (Add output to the end of a file)

Not to Overwrite Files

- Setting **noclobber** flag prevents you from accidentally overwriting a file when you redirect output to a file
  - To setup noclobber
    - ***set -o noclobber* or *set -C***
  - To unset noclobber
    - ***set +o noclobber* or *set +C***
- Examples:
  - ***date > f1***
  - ***set -o noclobber***
  - ***date > f1***
  - ***set + noclobber***
  - ***date > f1***

Pipe

- Using “|” to connect two commands
  - `ls /etc | more`
  - The output of the first command is the input of the second command

Customizing your prompt

- **Changing primary user prompt:**
  - PS1 environment variable
    - You can make your own prompt by using the following codes:

      \d, \h, \u, \n, \s, \t, \w

      PS1="\u@\h:\w$"
- **Changing secondary user prompt**
  - PS2 environment variable
  - Default secondary prompt is “>” for unfinished command in previous line, waiting for user to continue the command line.

    PS2="secondary prompt: "

wc command

- wc command prints newline, word and byte counts for files
- Options
  - w: word counts
  - l: line counts
  - c: character counts
- Examples:
  - `wc ~/*`
  - `wc -l /etc/fstab`

Displaying the Contents of Text Files

tail	Display the last 10 lines
head	Display the first 10 lines
more	Display text files page by page/line by line
less	Less is more
cat	Display the entire file
tac	Display the entire file in reverse order

cut command

- The ***cut*** command allows you to strip text out of files and display the cut text on the screen or redirect it to another file.
- Syntax: ***cut options filename***
  - Options
    - -d: is used to identify the delimiter
    - -f: is used to identify which field you want
- Examples:
  - ***cut -d: -f1 /etc/passwd***
  - ***cut -f1 /etc/passwd***
  - ***cut -d: -f1,3 /etc/passwd***

paste command

- The ***paste*** command pastes, or merges data from one file to another.
- Example: ***`paste file1 file2`***

vi file1
vi file2

  - The above command merges each line from file2 with each line from file1
  - The content of the original files remain the same.

file1

```
1
2
3
```

file2

```
Monday
Tuesday
Wednesday
```

output

```
1       Monday
2       Tuesday
3       Wednesday
```

History Command

- ***history*** **command**
  - It displays a list of the commands you have used in ***bash shell***
  - To clear history list
    - ***history -c***
  - Configuring history
    - HISTFILE: where the history is stored, by default, this file is ***~/.bash_history***
      - ***echo $HISTFILE***
    - HISTSIZE: the number of history commands could be saved. The default number is **1000** in Ubuntu.
      - ***echo $HISTSIZE***

/: to search keyword
.files are hidden files

History Event References

- You can use ***!*** Command to reference an history event
  - Re-execute any previous command**:** ***!n***
    - where ***n*** is the event number as listed in the **history** output
    - Examples:
      - ***!3***
      - ***!–4***
        - The reference can be an offset from the end of the history list
  - Re-execute the last command: ***!!***

Locating Commands

- ***bash shell*** keeps a list of directories in which it should look for commands in an environment variable called **$PATH**
- ***which*** command
  - It locates utilities by displaying the full pathnames of the commands which would be executed in the current environment.
  - It only searches the **$PATH**
  - Options: ***-a*** - which causes it to print all matches instead of stopping at the first one.
  - Examples:
    - `which cp`

Locating Commands

- ***whereis*** command
  - locate the binary, source code, and manual page files for a command
  - It searches for files related to a utility by looking in in a list of standard Linux places (*/bin*, */etc*, */usr/bin*,*/usr/local/bin/*, etc.) instead of using your searching path (**$PATH**)
  - Options:
    - **-b** - Search only for binaries
    - **-m** - Search only for manual pages
    - **-s** - Search only for source codes

Define Aliases

- Define your own commands
  - ***alias myls='ls -la'***
  - ***alias***
    - **To display all alias that have been defined**

- Remove Alias
  - ***unalias myls***