

CS2094 Data Structures Lab

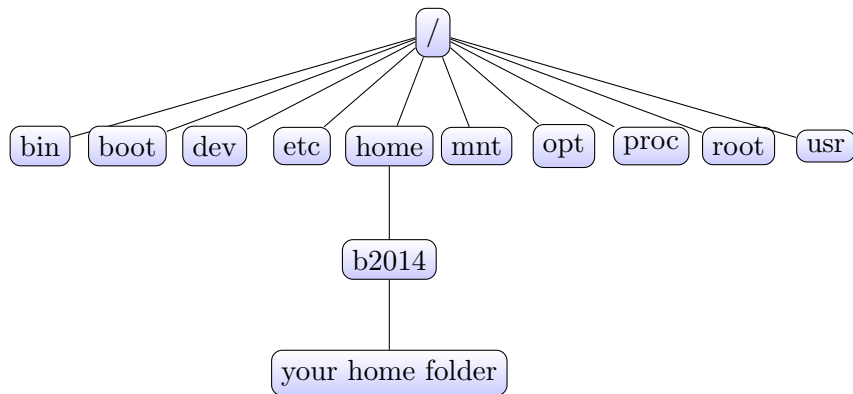
Introduction to Linux Environment

M. P. Giles, Rakesh, Sreekar Jayendra



Department of Computer Science & Engg
National Institute of Technology Calicut

Linux File System



Your home folder is `/home/b2014/yourname_rollno/`

Upon login the prompt in bash shell would be typically:

`username@hostname:~$`

[`~` → your home folder, `$` → a user, `#` → root user]

Basic Commands I


<code>man <i>command</i></code>	Shows the manual page of the command
<code>\$pwd</code>	Prints the working directory.
<code>\$ls [-la] or \$dir</code>	lists the contents of a folder in column major order, lexicographically sorted.
<code>\$ mkdir <i>dir_name</i></code>	Create a folder or directory named <i>dir_name</i> in working directory.
<code>\$cd <i>dir_name</i></code>	Switch to <i>dir_name</i> in working directory. <i>dir_name</i> = . for current directory. <i>dir_name</i> = .. for the parent. <i>dir_name</i> is empty for home folder. Give complete path for an arbitrary directory.
<code>\$touch <i>filename</i></code>	creates an empty file <i>filename</i> .
<code>\$cat <i>filename</i></code>	Prints the contents of <i>filename</i>
<code>\$cp [-r] <i>src dst</i></code>	Copy the <i>src</i> to <i>dst</i> . if <i>src</i> is a folder, use -r. If <i>dst</i> is a folder, adds <i>src</i> to it. Else, creates or overwrites the <i>dst</i> .

Basic Commands II

<code>\$mv src dst</code>	Move the <i>src</i> to <i>dst</i> . If <i>dst</i> is a folder, adds <i>src</i> to it. Else, creates or overwrites the <i>dst</i> .
<code>\$rm [-rf] name</code>	remove a file. Use -r to delete folders.
<code>\$grep string</code>	Search for the pattern <i>string</i> in a text in standard output. Usually used with pipes. Eg. <code>cat file grep pattern</code>
<code>\$cat "Text" > filename</code>	add "Text" to <i>filename</i> . Use >> to append.
<code>\$ssh user@host¹</code>	Login to a remote machine(host). <i>host</i> may be the host name or ip address.
<code>\$scp [-r]src user@host:dir</code>	Copy <i>src</i> to folder <i>dir</i> relative to user's home folder in <i>host</i> . Use -r if <i>src</i> is a folder.

Basic Commands III











<code>\$chmod [-R] ddd <i>name</i></code>	Change the permissions of file/folder. <code>d</code> is a 3-bit number corresponding to <code>r,w,x</code> permissions. Use <code>-R</code> for recursive mode.
<code>\$chmod u+x,g+w,o-r <i>name</i></code>	Add execute permission to user, write permission to group, and remove read permission from others.
<code>\$chown [-R] <i>user:group name</i></code>	Change the ownership of file/folder to user.

¹if local username is same as remote, `ssh host` is enough 

Vi / Vim I

- ▶ To start create/edit a file
`$vi` - opens an empty buffer.
`$vi filename` - opens a buffer with name `filename`
`$vi -r filename` - recover and open the file `filename` which had crashed earlier.
- ▶ There are two modes in vi - command mode and insert mode.
- ▶ To go to command mode : Press `Esc` Key
- ▶ For insert mode : press `i` / `I` / `a` / `A` / `o` / `O`
insert before cursor / start of current line / append after cursor / append at end of line / in new line after / in new line before
- ▶ Command Mode (after you press `Esc` key)
 - ▶ `:q` - Exit from editor.
 - ▶ `:w` - Write buffer to current file [Similar to Save]

Vi / Vim II

- ▶ `:wq [:x]` - Write buffer to current[original] file and exit.
- ▶ `:w filename` - Write buffer to filename [Similar to Save As].
- ▶ `:q!` - Exit from editor without saving.
- ▶ `:r filename` - to open a new file without exiting vi.
- ▶ Cursor Movement
 - ▶ `Nj` /  /  - move cursor down N/1/1 lines.
 - ▶ `Nk` /  - move cursor up N/1 line.
 - ▶ `Nh` /  /  - move cursor left N/1 character.
 - ▶ `Nl` /  - move cursor right N/1 character.
 - ▶ `0` - move cursor to start of current line (the one with the cursor)
 - ▶ `$` - move cursor to end of current line
 - ▶ `w` - move cursor to beginning of next word
 - ▶ `b` - move cursor back to beginning of preceding word
 - ▶ `:o  / 1G` - move cursor to first line in file
 - ▶ `:n  / nG` - move cursor to line n
 - ▶ `:$  / G` - move cursor to last line in file
- ▶ Text Manipulation
 - ▶ `u` / `:u ` - **undo the last action. Its a toggle.**
 - ▶ `x` - delete single character under cursor

Vi / Vim III

- ▶ N x - delete N characters, starting with character under cursor
- ▶ d w - delete the single word beginning with character under cursor
- ▶ d N w - delete N words beginning with character under cursor
- ▶ D - delete the remainder of the line, starting with current cursor position.
- ▶ d d / N d d / d N d - Delete(cut) one / N / N lines beginning with current line.
- ▶ y y / N y y / y N d - Yank(copy) one / N / N lines beginning with current line.
- ▶ p - paste the cut /yanked lines after current line
- ▶ Miscellaneous
 - ▶ /**str** - search forward for the string *str*. Use n / N to search for next /previous occurrence.
 - ▶ ?**str** - search backward for the string *str*. Use n / N to search for next /previous occurrence

Vi / Vim IV

- ▶ `:s/str/new/g[cI]` - replace all occurrences of *str* in current line with *new*. *new* may be empty. [c] is for explicit confirmation. [I] for case sensitive.
- ▶ `:%s/str/new/g` - replace all occurrences of *str* in all lines with *new* pty. Use [c] is for explicit confirmation
- ▶ `:%s/\<str\>/new/gc` - match whole word *str*.
- ▶ `:set nu` - to display line numbers. Use `:set nonu` to disable.

GCC I

- ▶ After creating a c/c++ program file eg. **add.c**, use the following command in the program directory to create executable **add**.

```
$gcc -o add add.c
```

- ▶ To execute the program, use the command
\$./add
- ▶ Use the following command to list all warnings. It is useful for clean codes.

```
$gcc -Wall -o add add.c
```

- ▶ Use the following for optimizing your code. If $n \in (1..3)$

```
$gcc -O $n$  -o add add.c
```

- ▶ To see the assembly code of the program

```
$gcc -S add.c
```

GCC II

- ▶ Use the following command to enable the debugging info.
Use `[$ulimit -c unlimited]` to enable file dump of any size.

```
$gcc -g -o add add.c
```

- ▶ To debug with gdb `$gdb ./add`
- ▶ To set break point in gdb
`(gdb) break add.c:8/ add.c:main / *0x1f7b`
- ▶ To start execution `(gdb) run`
- ▶ To step through the code `(gdb) s [or n]`
(use `print/display varname` to print value of varname).
- ▶ To fast forward from one break point to another
`(gdb) continue`
- ▶ To create combinable object files for multi-file projects
`$gcc -c add.o add.c`

- ▶ To create executable `final` in a multi-file project with “file1.c and file2.c”²

```
$gcc -c file1.o file1.c
```

```
$gcc -c file2.o file2.c
```

```
$gcc -o final file1.o file2.o
```

²Assuming that proper header files are created

*Thank
You*