Using the “CLI”

Unix / Linux Preparation Course

June 9, 2013
Lusaka, Zambia

The format of a command

```
command [options] parameters
```

“Traditionally, UNIX command-line options consist of a dash, followed by one or more lowercase letters. The GNU utilities added a double-dash, followed by a complete word or compound word.”

Two very typical examples are:

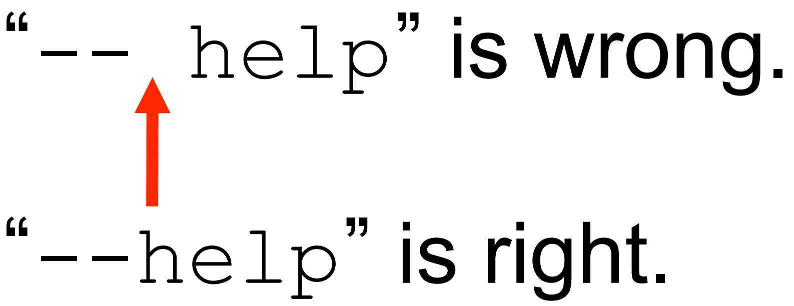
```
-h
--help
```

and

```
-v
--version
```

Command parameters

- The ***parameter*** is what a command ***acts upon***.
- Often there are multiple parameters.
- In Unix UPPERCASE and lowercase for both options and parameters matter.
- **Spaces** ___ are ___ critical ___



“`--  help`” is wrong.

“`--help`” is right.

Some command examples*

Let's start simple – *Follow along as we go*:

Display a list of files:

```
ls
```

Display a list of files in a long listing format:

```
ls -l
```

Display a list of all files in a long listing format with human-readable file sizes:

```
ls -alh
```

*do this! ☺

Some command examples cont.

Some equivalent ways to do “`ls -alh`”:

```
ls -lah
ls -l -a -h
ls -l –all --human-readable
```

Note that there is no double-dash option for “`–l`”. You can figure this out by typing:

```
man ls
```

Or by typing:

```
ls --help
```

Where's the parameter?

We typed the “`ls`” command with several options, but no parameter. Do you think “`ls`” uses a parameter?

Q.) What is the parameter for “`ls -l`”?

A.) It is “.” -- our current directory.

“`ls -l`” and “`ls -l .`”

are the same. We'll discuss files and directories later.

A disconcerting Unix feature

If a command executes successfully there is no output returned from the command execution. *this is normal*.

That is, if you type:

```
cp file1 file2
```

The result is that you get your command prompt back. *Nothing means success*.

Let's give this a try...

A disconcerting Unix feature cont.*

Try doing the following on your machine:

```
$ cd                       [cd = change dir]
$ touch file1              [touch = create/update]
$ cp file1 file2           [cp = copy]
```

- The “$” indicates the command prompt for a normal user.
- A “#” usually means you are the *root* user.

Using pipes

In Unix it is very easy to use the result of one command as the input for another.

To do this we use the pipe symbol “|”. For example:

```
ls /sbin | sort
ls /sbin | sort | more
```

What will these commands do? Give it a try. Press “q” to exit sort and more screen.

Stopping Command Output*

Stopping commands with continuous output:

Terminate foreground program: CTRL+C

```
$ ping yahoo.com
PING ds-any-fp3-real.wa1.b.yahoo.com (98.139.183.24) 56(84) bytes of data.
64 bytes from ir2.fp.vip.bf1.yahoo.com (98.139.183.24): icmp_req=1 ttl=46 time=610 ms
64 bytes from ir2.fp.vip.bf1.yahoo.com (98.139.183.24): icmp_req=2 ttl=47 time=541
  ms^C     ← here press CTRL + C
```

Terminate paging like “less <filename>”

```
$ less /etc//ssh/sshd_config

# Package generated configuration file
# See the sshd_config(5) manpage for details

# What ports, IPs and protocols we listen for
Port 22(END)    ← press the “q” key
```

Proper command line use

The command line in Unix is *much more powerful* than what you may be used to in Windows. ***You can...***

...easily edit long commands

...find and recover past commands

...quickly copy and paste commands.

...auto-complete commands using the tab key (in *bash* shell).

Edit long commands

STOP! Don't touch that keyboard!
Arrow keys are so very slooooooow...

- Use *Home* and *End* instead (ctrl-a, ctrl-e)
- Delete with *Backspace* not *Delete*.
- Press <ENTER> *as soon as the command is correct*. You *do not* need to go to the end of the command.
- Use "`history | grep string`", then `!NN` instead of lots of up-arrows.

Find and recover past commands*

As noted on the previous slide. Use:

```
$ history | grep “less”
```

Find command number in resulting list.

Execute the command by typing:

```
$ !number
```

Quickly copy and paste commands

In Unix/Linux once you highlight something it is *already* in your copy buffer.

To copy/paste do:

- Highlight text with left mouse cursor. It is now copied (like *ctrl-c* in Windows).
- Move mouse/cursor where you want (any window), and press the *middle* mouse button. This is paste (like *ctrl-v*).

Doesn’t work on a Mac…

Copy and paste commands

Do this!!!

Good system administrator

==

Lazy Person

Goal State

Don't try to type a long command if you can copy / paste it instead.

Auto-complete commands using tab

Very, very, very powerful

"The tab key is good", "the tab key is my friend", "press the tab key", "press it again" - This is your mantra.

Tab works in the *bash* shell. Note, the *root* user might not use the *bash* shell by default.

Use the tab key! You'll thank us later ☺

Auto-complete commands using tab

Core concept:

Once you type something unique, press TAB. If nothing happens, press TAB twice.

- If text was unique text will auto-complete. A command will complete, directory name, file name, command parameters will all complete.
- If not unique, press TAB twice. All possibilities will be displayed.
- Works with file types based on command!

Auto-completion

We’ll do this now:

```
$ cat /etc     (TAB twice quickly)
$ cat /etc/netw          (TAB)
$ cat /etc/network/in   (TAB)
```

Viewing files

Several ways to view a file:

1. `cat <filename>`
2. `more <filename`
3. `less <filename>`

- `cat` is short for *conCATenate*
- “`less` is `more`”

Viewing files*

Let’s do this now:

```
$ cat /etc/ssh/sshd_config
$ more /etc/ssh/sshd_config
$ less /etc/ssh/sshd_config
```

- `“q” to “q”uit from more, or less`
- Less: “b” for back, “f” for forward, “/” to search
- Less: /term, then “n” for next, “N” for previous

Obtaining help*

To get help explaining commands you can do:

```
- man <command>
- <command> --help
```

man stands for “man”ual.

More on “man”

```
- man man
```

More on Linux directory structure:

```
- man hier
```

Installing Software (Linux)

From the command line you use either apt or aptitude (root privileges needed)

- `apt-get install <PACKAGE>`
- `aptitude install <PACKAGE>`

Equivalent on FreeBSD would be (root privs):

- `pkg_add –r <PACKAGE>`

Finding a package (root not needed):

- `apt-cache search <NAME>`

Installing Software (Linux)

Let’s install two packages in Ubuntu:

```
$ sudo apt-get install postfix
```

Respond yes to all the defaults, then install another editor named “joe”:

```
$ sudo apt-get install joe
```

Your mission

Should you choose to accept it...

- Pay close attention to options and parameters.
- Use “`man command`” or “`command --help`” to figure out how each command works.
- Use command line magic to save lots and lots and lots and lots of time.
- A command acts upon its parameters based on the options you give to the command...