



Hervey Allen

Network Startup Resource Center

PacNOG 6: Nadi, Fiji

Using Commands in Linux

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 the command ***acts on***.
- 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:

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
```

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?

- What is the parameter for “ls -l”?
- It is “.” which represents 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

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 -l /sbin | sort
```

```
ls -l /sbin | sort | more
```

What will these commands do?

Press “q” to exit the “more” command.

Take advantage of the command line

The command line in Unix is *much more powerful* than what you may be used to with 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



! Don't touch that keyboard!

Arrow keys are so sloooooooooow...

- Use *Home* and *End* instead (or 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 "command string"
```

- Find command number in the resulting list, and execute the command by typing:

```
$ !number
```

- So, to find the ssh-keygen command you typed “many” commands ago you could do:

```
$ history | grep ssh-keygen
```

Quickly copy and paste commands

In Unix/Linux once you highlight something it is *already* in your copy buffer (except on the Mac).

To copy/paste do:

1. Highlight text with left mouse cursor. It is now copied.
(Like *ctrl-c* in Windows or *Apple-c* on the Mac)
2. Move the mouse or cursor where you want (any window), and press the *middle* mouse button.
(Like *ctrl-v* or *Apple-v*)

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.
- Remember, the tab key is your friend! 😊

Auto-complete commands using tab

Core concepts:

- 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!

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...

```
command [options] parameters
```