CS2043 - Unix Tools & Scripting

Cornell University, Spring 2014[1]

Instructor: Bruno Abrahao

February 19, 2014

[1] Slides evolved from previous versions by Hussam Abu-Libdeh and David Slater

Plan for today

- **Screen**
- **Gnuplot**

Part I

More on process execution and user session.

Recap: Starting a Job in the background

To run a job in the background, we will use a new command-line operator:

&

`<command> [arguments] &`

- Runs the specified command as a background job
- Unless told otherwise, will send output to the terminal!
- In some systems your job ends if you log out!

The Other Way

If you have a **noninteractive** batch job, you can also allow it to continue to run after you logout by using `nohup`

`nohup`

`nohup command`

- command will continue to run after you logout
- output is sent to `nohup.out` if not otherwise redirected
- can be combined with `nice`

Example:

```
nohup nice -15 math < BatchJob.m &
```

screen

User Session

There are a few limitations with your basic BASH session. Some of these you may even have encountered already:

- Your session isn't preserved if you close your `ssh` connection
- It's a pain to switch back and forth between files/the prompt.
- Sometimes using two or three shells at once would be really convenient!

All of these complaints can be resolved by using `screen`.

Intro To Screen

The `screen command`

`screen` - a screen manager with terminal emulation

Can be used just as your terminal window. However, special commands can be used to allow you to save your session, create extra shells, or split the window into multiple independent panes.

Passing Commands to screen

Each `screen` commands consists of a CTRL-a (hereafter referred to as C-a) followed by another character.

Using Screen

Attach a screen

`screen [options]`

- Opens a new screen for use
- -a : include all capabilities

Resume a screen

`screen -r [pid.tty.host]`

- Resumes a detached screen session

`screen -x [pid.tty.host]`

- Attach to a non-detached screen session

If you only have one screen, the [pid.tty.host] string is unnecessary.

Identifying Screen Sessions

Screen Listing

`screen -ls` or `screen -list`

- Lists your screen sessions and their statuses

These screen sessions are the [pid.tty.host] strings required for resuming

Resuming a screen

If `screen -ls` returns `15829.pts-9.rumman (Detached)`

- `screen -r 15829.pts-9.rumman` to resume the screen

Note: You only need to specify the full "name" of the session if you have multiple sessions open. If you just have one session, just use `screen -r`

Creating More Shells

Creates a New Shell Window

`C-a c`

- Creates a new shell in a new window and switches to it
- Useful for opening multiple shells in a single terminal
- Similar to tabbed browsing/tabbed IMs

But how do we switch between windows? (hint: every window is numbered by order of creation)

Window Selection

`C-a 1` - switches to window 1 `C-a 9` - switches to window 9

Splitting Screen

Split Screen Computing

`C-a S` - splits your terminal area into multiple panes

`C-a tab` - changes the input focus to the next pane

- The 'S' is case-sensitive
- Each split results in a blank pane
- Use `C-a c` to create a new shell in a pane
- Use `C-a <num>` to move an existing window to a pane

Now lets put this together to do something useful

Suppose you are doing some serious scientific computing and want to run it on a remote server. We can put together what we have learned to do this efficiently:

- `ssh` into the remote machine

```
ssh <netid>@csug01.csuglab.cornell.edu
```

- start `screen`

```
screen
```

- start mathematica

```
math < BatchJob.m
```

- renice the math process so other uses can use the machine

```
renice -20 PID
```

- Detach the screen, logout, and come back 8 hours later when it is done

Part II

So far we have been only talking about text processing. However that is not all what we can do and automate! For example, we can use tools to automate graph plotting (`gnuplot`)

gnuplot

- A command-line program to generate 2D and 3D plots of functions, data, and data fits.
- Publication quality graphics as well as educational purposes.
- Available on all major operating systems (GNU/Linux, Unix, Windows, Mac OS X, and others).

Gnuplot

- Run `gnuplot` with no command line arguments to get an interactive console.
- Run `gnuplot plot_script` to execute the plotting commands from the given script file.

Important gnuplot commands

- `help` : great help and documentation for the many `gnuplot` features and commands
- `plot` : plot data or a function

Plot the function `sin(x)`

```
plot sin(x)
```

- `set xrange / yrange` : set the range on the x or y axis.

```
set xrange [0:5]
```

- `set xlabel / ylabel` : set the text label on the x or y axis

```
set xlabel ‘‘Time (s)’’
```

- `set title` : set the graph title

```
set title ‘‘Awesome performance graph!’’
```

Plotting options

- `gnuplot` can read data files and automatically separate data from multiple columns

Plot data using the first and second column from a file

```
plot ‘‘data.csv’’ using 1:2
```

- Plot multiple data series

```
plot ‘‘data.csv’’ using 1:2, ‘‘data.csv’’ using 1:3
```

- Plot the series using a line

```
plot ‘‘data.csv’’ using 1:2 with lines
```

- Plot the series using boxes

```
plot ‘‘data.csv’’ using 1:2 with boxes
```

gnuplot references

There are many things that gnuplot can do. To learn more about a particular command or function, use the built in help (which is searchable).

To learn more about the plot command

```
help plot
```

- Many very useful gnuplot tips here:
  http://t16web.lanl.gov/Kawano/gnuplot/index-e.html
- gnuplot project homepage:
  http://www.gnuplot.info/