Lab 3: The Unix Shell

Benedict Reuschling

February 19, 2019

1 Shell commands

For this part of the lab exercise, create a separate folder in your home directory. Create a few files in that folder to have an environment where you can show the results of the commands listed below. Use the standard Shell (`/bin/sh`) for this exercise, as some other shells might handle them differently.

What do the following command sequences do? Note that they do not depend on each other.

1. `echo x*y`
2. `ls x*y`
3. `date|cut -c 1-3`
4. ``echo `who|wc -l` > count``

2 Word Count - Chain of Pipes

In this exercise, we want to create a chain of pipes on the shell that will echo the ten most frequently appearing words in a given text at the end. The file should be piped to others to format it in a way that makes it suitable for counting, and lastly present the results in descending order (ten most frequent words first) on the screen. An example output is given below:

```
12 the
10 and
 7 or
```

Capital letters should be transformed to lowercase before counting. Special characters (sentence stops, exclamation marks, etc.) must be filtered out. Split the problem into logical steps: transforming/filtering, counting, and echoing the result.

Use only the following programs to solve this exercise:

`cat(1)`, `fmt(1)`, `head(1)`, `sort(1)`, `tr(1)`, `uniq(1)`

Read the man pages for these commands and look for options that will help you achieve the result you need. Commands can be used multiple times in the chain of pipes. Use only those programs, don't write your own to solve this assignment. Use the attached poem "Be glad your nose is on your face" by Jack Prelutsky as test input (separate file provided).

When you're done, pick another text of your choice and see if it your chain of pipes works for that one, too.

3 Visualizing graphs

In the daily life of a programmer, sometimes there is a need for visualizing certain things like program flows, dependencies, and other graph-like structures to make them more comprehensive. To do that without having to learn a complicated graphics program, we will use the command line to generate graphs. Such a program for creating graphs is called GraphViz (`http://graphviz.org`). Install it by using either `pkg install` in FreeBSD (when you are using the VM from our first lab) or the package manager that your distribution provides.

Graphviz reads a file with instructions on how nodes are connected to each other and how they look like (shape, size, color, etc.). It then uses different layout engines to arrange the nodes and connections between them. Browse through the dot Guide[1] (one of the layout engines) to get a basic idea on how dot works. Then, create a graph similar to the one shown in Figure 1:

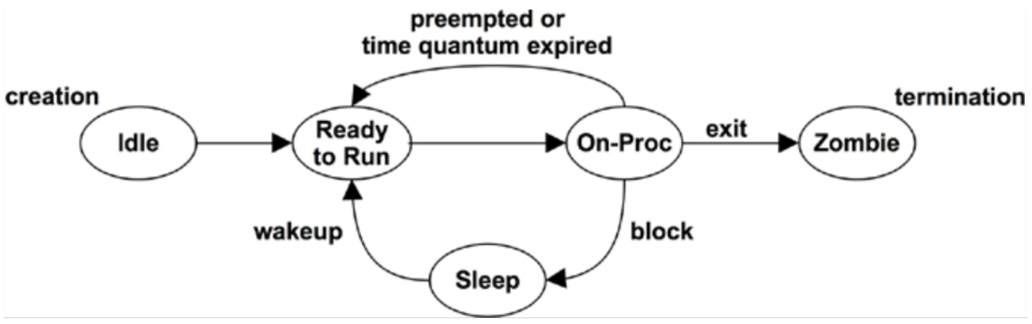

Figure 1: Process life cycle from “Systems Performance: Enterprise and the Cloud” by Brendan Gregg, Homepage: `http://www.brendangregg.com/sysperfbook.html`

Experiment with different layout engines and create a PNG file with the resulting image. If you don't have a graphical interface installed, use `scp` to copy the file to your local machine and view it there. You can also install a browser like firefox or chromium and view the PNGs there.

Note: your graph does not have to look exactly like Figure 1, just make sure that you include all the nodes, connections, and descriptions.

Show me your dot file and the resulting graph when you're finished.

[1] `http://graphviz.org/pdf/dotguide.pdf`