CSCI 4061: Unix Basics

Chris Kauffman

Week 1
Logistics

Reading

- Robbins and Robbins, Ch 1 and Ch 2
- OR Stevens and Rago, Ch 1 and Ch 7

Goals Today

- Warm-up C program
- Course Mechanics
- Unix Basics
Continue Mechanics

See remaining mechanics slides.
Finish Intro C Program
Access to Unix Machines

- CSE Labs
  - Via SSH
  - Via \texttt{http://vole.cse.umn.edu}
- Mac OS X: Terminal, development tools
- Windows: Install Cygwin
- Either: Install \texttt{VirtualBox} to host a Unix you like
Exercise: Quick Review

- Name two major functions of all operating systems
- Give some major components/abstractions that Unix provides as part of its interfaces
- What is a system call? How does it work?
Unix Standards: POSIX

POSIX defines what you an plausibly expect on unix-like systems. Includes

- C libraries for system calls, standard libraries
- Basic layout of file system and naming conventions
- Some Devices such as /dev/null

Distinction: C Standard vs Unix Library

- Lots of systems have a C compiler which has the C standard library: printf(), fopen(), exp() etc.
- Unix systems have additional, separate libraries for Unix-specific stuff like read(), fork(), select(), kill()
- Some branches of Unix have their own special, special versions of these like Linux clone()
## Command Line: Basic File System Navigation

<table>
<thead>
<tr>
<th>Command</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pwd</code></td>
<td>print the current directory</td>
</tr>
<tr>
<td><code>cd folder</code></td>
<td>change directory / folder</td>
</tr>
<tr>
<td><code>ls</code></td>
<td>list files in directory</td>
</tr>
<tr>
<td><code>cd ~</code></td>
<td>change to home directory</td>
</tr>
</tbody>
</table>

```bash
> pwd
/home/kauffman
> ls
1103-F2017  aurs  Downloads  Hello.class  Hello.java~  PathClassLoader.txt
4061-F2017  Desktop  Dropbox  Hello.java  misc  public_html
> cd 4061-F2017
> ls
exams  lectures  Makefile~  projects  schedule.html~  schedule.org~  textbook
labs  Makefile  misc  schedule.html  schedule.org  syllabus
> pwd
/home/kauffman/4061-F2017
> cd lectures
> pwd
/home/kauffman/4061-F2017/lectures
> ls
00-course-mechanics.org  00-course-mechanics.tex  01-introduction.org  01-introduction.tex
00-course-mechanics.org~  01-introduction-code  01-introduction.org~  02-unix-basic.c
00-course-mechanics.pdf  01-introduction-code.zip  01-introduction.pdf  02-unix-basics.org
> cd ~
> pwd
/home/kauffman
> ls
1103-F2017  aurs  Downloads  Hello.class  Hello.java~  PathClassLoader.txt
4061-F2017  Desktop  Dropbox  Hello.java  misc  public_html
```
Determining File Types

<table>
<thead>
<tr>
<th>Command</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>file something.ext</td>
<td>try to determine the type of given file</td>
</tr>
</tbody>
</table>

> file xxx
xxx: UTF-8 Unicode text, with very long lines

> file test.txt
test.txt: ASCII text

> file www
www: directory

> file 4061-F2017
4061-F2017: symbolic link to /home/kauffman/Dropbox/teaching/4061-F2017

> file 4061-F2017/
4061-F2017/: directory

> cd 4061-F2017/lectures/
> file 01-introduction-code.zip
01-introduction-code.zip: Zip archive data, at least v1.0 to extract

> file 02-unix-basics-code/no_interruptions.c
02-unix-basics-code/no_interruptions.c: C source, ASCII text

> file 02-unix-basics-code/no_interruptions.o
02-unix-basics-code/no_interruptions.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped

> file 02-unix-basics-code/a.out
02-unix-basics-code/a.out: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=ff87934737b0e48b891d27573ae8a2e5687c46a, not stripped

>
## Searching and Manipulating Text

<table>
<thead>
<tr>
<th>Command</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>cat file.txt</td>
<td>show contents of file in terminal</td>
</tr>
<tr>
<td>less file.txt</td>
<td>&quot;page&quot; text file, press &quot;q&quot; to quit</td>
</tr>
<tr>
<td>grep 'expression' file.txt</td>
<td>show lines matching expression in file</td>
</tr>
<tr>
<td>grep 'expression' *.txt</td>
<td>search every .txt file for lines</td>
</tr>
<tr>
<td>find .</td>
<td>show all files recursively from current directory</td>
</tr>
<tr>
<td><del>find . -name '*c'</del></td>
<td>find all C source files recursively</td>
</tr>
</tbody>
</table>

These may be covered in a future lab.
Editing Files

<table>
<thead>
<tr>
<th>Command</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>vi</td>
<td>modal editing, terse, powerful text manipulator, ALWAYS</td>
</tr>
<tr>
<td>emacs</td>
<td>modes for editing, extensible, almost always available</td>
</tr>
<tr>
<td>nano</td>
<td>simple, podunky, usually available</td>
</tr>
</tbody>
</table>

- Learn some vi or emacs
- Comes in real handy when you need to edit but there is no graphical login
## Permissions on Files

<table>
<thead>
<tr>
<th>Command</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ls -l</code></td>
<td>long listing of files</td>
</tr>
<tr>
<td><code>chmod u+x file.abc</code></td>
<td>make file executable by user</td>
</tr>
<tr>
<td><code>chmod o-rwx file.abc</code></td>
<td>remove permissions from other users</td>
</tr>
<tr>
<td><code>chmod 777 file.abc</code></td>
<td>everyone can do anything to file</td>
</tr>
</tbody>
</table>

```bash
> ls
a.out no_interruptions.c no_interruptions.c~ no_interruptions.o
> ls -l
  total 40K
  -rwxrwx--- 1 kauffman kauffman  8.5K Sep 7 09:55 a.out
  -rw-r--r-- 1 kauffman kauffman  955 Sep 7 09:55 no_interruptions.c
  -rw-r--r-- 1 kauffman kauffman  883 Sep 7 09:54 no_interruptions.c~
  -rw-rw---- 1 kauffman kauffman  2.4K Sep 7 11:59 no_interruptions.o
> chmod u-x a.out
> ls -l
  total 40K
  -rw-rwx--- 1 kauffman kauffman  8.5K Sep 7 09:55 a.out
  -rw-r--r-- 1 kauffman kauffman  955 Sep 7 09:55 no_interruptions.c
  -rw-r--r-- 1 kauffman kauffman  883 Sep 7 09:54 no_interruptions.c~
  -rw-rw---- 1 kauffman kauffman  2.4K Sep 7 11:59 no_interruptions.o
> ./a.out
bash: ./a.out: Permission denied
> chmod u+x a.out
> ./a.out
Ma-na na-na!
```
> man ls | cat

LS(1)  User Commands  LS(1)

NAME

`ls` - list directory contents

SYNOPSIS

`ls` [OPTION]... [FILE]...

DESCRIPTION

List information about the FILEs (the current directory by default). Sort entries alphabetically if none of `-cftuvSU` nor `--sort` is specified.

Mandatory arguments to long options are mandatory for short options too.

- `-a`, `--all`
  do not ignore entries starting with `.`

- `-A`, `--almost-all`
  do not list implied `.` and `..`

...
# Program Search PATH

<table>
<thead>
<tr>
<th>Command</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>echo $PATH</code></td>
<td>show where shell looks for programs</td>
</tr>
<tr>
<td><code>PATH=$PATH:/home/kauffman/bin</code></td>
<td>also look in my bin directory</td>
</tr>
<tr>
<td><code>PATH=$PATH:</code></td>
<td>also look in current directory</td>
</tr>
<tr>
<td><code>PATH=.</code></td>
<td>ONLY look in the current directory</td>
</tr>
</tbody>
</table>

```bash
> echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/bin:/usr/lib/jvm/default/bin:
/usr/bin/site_perl:/usr/bin/vendor_perl:/usr/bin/core_perl:/home/kauffman/bin:
/home/kauffman/Dropbox/bin:/home/kauffman/code/bin:/home/kauffman/code/utils:.
```

Search directories are separated by colons in Unix
Exercise: Compilation

- What command is typically used to compile C programs again?
- What function does a *runnable* C file need to have to make a program?
- Can you compile a C file without that special function function?
- What is the default name of a compiled program on Unix?
- How do you change the name of the file the compiler will produce?
make and Makefiles

- Example of a build system
- Very old system, many newer ones but a good starting point
- Will be discussed in Lab01 which will go up over the weekend
- Make sure to attend your first lab, the one you are registered for
How make and Makefile Work

Build up dependencies recursively

- A tree-like structure (actually a DAG)
- Run commands for the lowest level
- Then go up a level
- Then up another ...
- Can recurse to subdirectories to use other Makefiles as well
- Makefile describes dependencies between source/program files and commands to generate/compile

Makefile Format

target1 : dependency1 dependency2
do command 1
then do command 2

target2 : target1 dependency3
do command X
then do command Y
## Showing and Murdering Running Processes

<table>
<thead>
<tr>
<th>Command</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ps</code></td>
<td>show running processes associated with terminal</td>
</tr>
<tr>
<td><code>ps a</code></td>
<td>show ALL running processes</td>
</tr>
<tr>
<td><code>ps u</code></td>
<td>show all processes for me</td>
</tr>
<tr>
<td><code>kill 1234</code></td>
<td>send process 1234 the TERM signal</td>
</tr>
<tr>
<td><code>kill -9 1234</code></td>
<td>send process 1234 the KILL signal</td>
</tr>
<tr>
<td><code>pkill a.out</code></td>
<td>send process named a.out the TERM signal</td>
</tr>
<tr>
<td><code>pkill -9 a.out</code></td>
<td>send process named a.out the KILL signal</td>
</tr>
</tbody>
</table>

```
> ps
  PID TTY TIME CMD
8050 pts/1 00:00:00 bash
8061 pts/1 00:00:00 ssh
11033 pts/1 00:00:00 ps
> ps u
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
kauffman 724 0.0 0.0 201092 5520 tty2 Ssl+ Sep06 0:00 /usr/lib/gdm/gdm-x-session --run-script cinnamon-session-cinnamon
kauffman 726 0.1 0.5 691872 94388 tty2 R1+ Sep06 2:08 /usr/lib/xorg-server/Xorg vt2 -displayfd 3 -auth /run/user/1000/gdm/Xau
kauffman 737 0.0 0.3 603020 49496 tty2 S1+ Sep06 0:00 cinnamon-session --session cinnamon
kauffman 784 0.0 0.1 565264 23008 tty2 S1+ Sep06 0:00 /usr/lib/cinnamon-settings-daemon/csd-orientation
...```