Principles of Programming
CM10227

Lecture S.1.: Introduction to Linux/Unix

Dr. Marina De Vos
University of Bath
Ext: 5053

Academic Year 2012-2013

Outline

1. Shell Basics

Resources

- Unix for Beginners. Dirk Vermeir
- `http://osl.iu.edu/~lums/swc/www/index.html`

Outline

1. Shell Basics
   - The Shell
   - The File System
   - Some Linux Commands

Why Command-line?

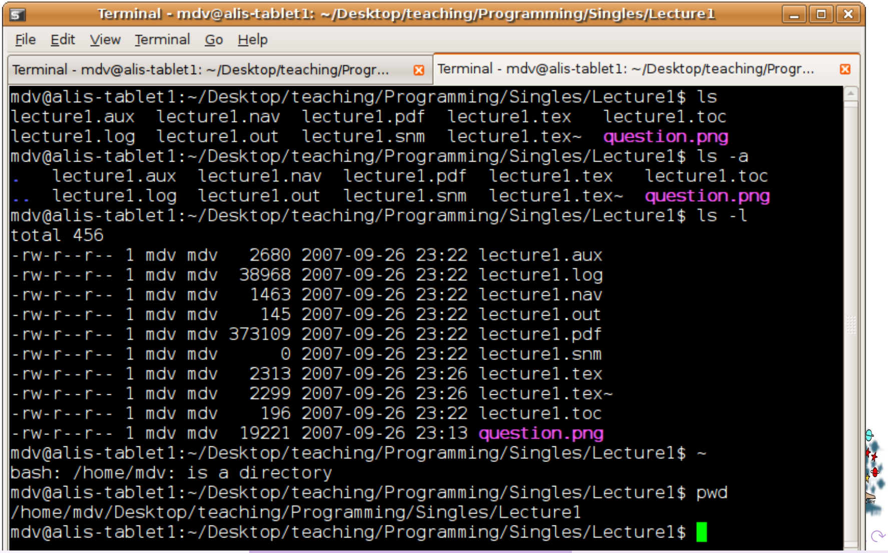
- Most modern tools have a graphical user interface (GUI)
  - Because they're easier to use
- But command-line user interfaces (CLUIs) still have their place
  - Easier (faster) to build new CLUI tools
    - Building a GUI takes time
    - Building a good GUI takes a lot of time
  - Higher action-to-keystroke ratio
    - Once you're over the (steeper) learning curve
  - Easier to see and understand what the computer is doing on your behalf
    - Which is part of what this course is about
  - Most important: it's easier to combine CLUI tools than GUI tools
    - Small tools, combined in many ways, can be very powerful

The Shell

The most important command-line tool is the command shell (often just called the shell)

- Manages a user's interactions with the operating system by:
  - Reading commands from the keyboard
  - Figuring out what programs the user wants to run
  - Running those programs
  - Displaying their output on the screen
- Looks (and works) like an interactive terminal circa 1980

The Terminal


Terminal - mdv@alis-tablet1: ~/Desktop/teaching/Programming/Singles/Lecture1
File Edit View Terminal Go Help
Terminal - mdv@alis-tablet1: ~/Desktop/teaching/Progr...
Terminal - mdv@alis-tablet1: ~/Desktop/teaching/Progr...

```
mdv@alis-tablet1:~/Desktop/teaching/Programming/Singles/Lecture1$ ls
lecture1.aux  lecture1.nav  lecture1.pdf  lecture1.tex   lecture1.toc
lecture1.log  lecture1.out  lecture1.snm  lecture1.tex~  question.png
mdv@alis-tablet1:~/Desktop/teaching/Programming/Singles/Lecture1$ ls -a
.   lecture1.aux  lecture1.nav  lecture1.pdf  lecture1.tex   lecture1.toc
..  lecture1.log  lecture1.out  lecture1.snm  lecture1.tex~  question.png
mdv@alis-tablet1:~/Desktop/teaching/Programming/Singles/Lecture1$ ls -l
total 456
-rw-r--r-- 1 mdv mdv   2680 2007-09-26 23:22 lecture1.aux
-rw-r--r-- 1 mdv mdv  38968 2007-09-26 23:22 lecture1.log
-rw-r--r-- 1 mdv mdv   1463 2007-09-26 23:22 lecture1.nav
-rw-r--r-- 1 mdv mdv    145 2007-09-26 23:22 lecture1.out
-rw-r--r-- 1 mdv mdv 373109 2007-09-26 23:22 lecture1.pdf
-rw-r--r-- 1 mdv mdv      0 2007-09-26 23:22 lecture1.snm
-rw-r--r-- 1 mdv mdv   2313 2007-09-26 23:26 lecture1.tex
-rw-r--r-- 1 mdv mdv   2299 2007-09-26 23:26 lecture1.tex~
-rw-r--r-- 1 mdv mdv    196 2007-09-26 23:22 lecture1.toc
-rw-r--r-- 1 mdv mdv  19221 2007-09-26 23:13 question.png
mdv@alis-tablet1:~/Desktop/teaching/Programming/Singles/Lecture1$ ~
bash: /home/mdv: is a directory
mdv@alis-tablet1:~/Desktop/teaching/Programming/Singles/Lecture1$ pwd
/home/mdv/Desktop/teaching/Programming/Singles/Lecture1
mdv@alis-tablet1:~/Desktop/teaching/Programming/Singles/Lecture1$
```

The Shell vs. the Operation System

- The shell is just one program among many
  - Many different ones have been written
  - sh was the first for Unix
    - Most others extend its capabilities in various ways
    - Which means that it's the lowest common denominator you can always rely on
  - We will use bash (the Bourne again shell)
    - Available just about everywhere
    - Even on Windows (thanks to Cygwin)

The Shell vs. the Operation System

- In contrast, the operating system is not just another program
  - Automatically loaded when the computer boots up
  - The only program that can talk directly to the computer's hardware
    - I.e., read characters from the keyboard, or send drawing commands to the screen
  - Manages files and directories on the disk
  - Keeps track of who you are, and what you're allowed to do
  - You can run many instances of the shell on a computer at once, but it can only run one operating system at a time

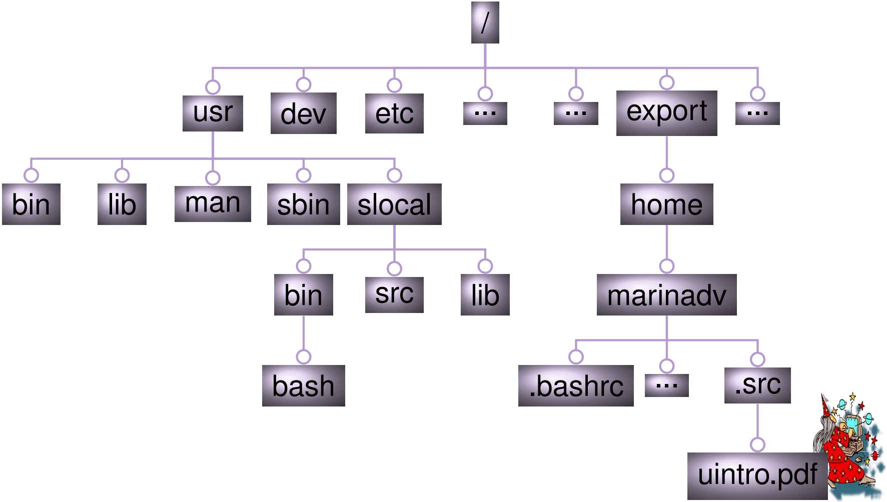
The File System

- The file system is the set of files and directories the computer can access

  Everything that stays put when you turn the computer off and restart it

- Data is stored in files
  - By convention, files have two part names, like notes.txt or home.html
  - Most operating systems allow you to associate a filename extension with an application
    - E.g., .txt is associated with an editor, and .html with a web browser
  - But this is all just convention: you can call files (almost) anything you want
- Files are stored in directories (often called folders)
  - Directories can contain other directories, too
  - Results in the familiar directory tree

Directory Tree



Drives

- On Unix, the file system has a unique root directory called /
  - Every other directory is a child of it, or a child of a child, etc.
- On Windows, every drive has its own root directory
  - So C:\home\mdv\notes.txt is different from J:\home\mdv\notes.txt
  - When you're using Cygwin, you can also write C:home\mdv as c :/ home/mdv
  - Or as /cygdrive/c/home/mdv
    - Some Unix programs give ":" a special meaning, so Cygwin needed a way to write paths without it

Paths

A path is a description of how to find something in a file system

- An absolute path describes a location from the root directory down
  - Equivalent to a street address
  - Always starts with ”/”
  - E.g., /home/mdv is my home directory, and /courses/swc/lec/shell.swc is this file
- A relative path describes how to find something from some other location
  - Equivalent to saying, Four blocks north, and seven east
  - E.g., from /courses/swc, the relative path to this file is lec/shell.swc

Special Paths

- Every program (including the shell) has a current working directory
  - Where am I?
  - Relative paths are deciphered relative to this location
  - It can change while a program is running
- Finally, two special names:
  - ”.” means the current directory
  - ”..” means the directory immediately above this one
    - Also called the parent directory
    - In /courses/swc/data, .. is /courses/swc
    - In /courses/swc/data/elements, .. is /courses/swc/data

File Systems

Most unix systems have several types of file systems

- Disk-based: UFS : to store all the files users create
- Netword-based: NFS: to connect to (mount) drives outside the machine
- tmpfs file system: supports simulating a file system in main memory, possibly backed up by swap storage. This is ideal for temporary files for which fast access is important.
- swap: file system is used to provide backup storage for processes that must temporarily be swapped out
- proc file space: provides a file view on the attributes of processes

pwd and ls

- pwd shows you the current directory

```
pwd
```

```
/home/mdv/Desktop/teaching/Programming/Singles/Lecture1
```

- ls shows you what's in the current directory

```
ls
```

```
lecture1.aux   lecture1.out   lecture1.tex    lecture1.vrb
lecture1.log   lecture1.pdf   lecture1.tex~   question.png
lecture1.nav   lecture1.snm   lecture1.toc    terminal.png
```

More on ls

What actually happens when I type ls is:

- The operating system reads characters from the keyboard
- Passes them to the shell (because it's the currently active window on my desktop)
- The shell breaks the line of text it receives into words
- Looks for a program with the same name as the first word (i.e., the command to run)
    - Describe in a moment how the shell knows where to look
- Runs that program
- Reads the program's output and sends it back to the operating system for display

Flags

- Flags are command-line option you can pass to commands
- Can tell ls to produce more informative output by giving it some flags
- By convention, flags start with "-", as in "-c" or "-l"
- For example: show directories with trailing slash

```
ls –F
```

```
bluej.png      code.sty~        copyright.tex~   rights.png       uintro.pdf
cm10192.tex    computer.jpg     Doubles/         Singles/
cm10192.tex~   computer.png     projects.zip     Stylefiles/
code.sty       copyright.tex*   python.png       template.tex~
```

- -a: gives you all files starting with ".", which are normally hidden
- -l: provides long listing format. provides permissions,size,latest access

Finding your way

- man pages: provide an overview of the functionality of a command.
  - man ls
- apropos: provides all commands related to a certain topic
  - appropos(permissions)
- −−help: provides support for a specific command
  - ls −−help

Manipulating Files and Directories

Lets work by example:

- let us create a temporary directory and play around in there

```
mkdir temp
```

- Note: no output
- The -v (verbose) flag tells mkdir to print a confirmation message
- Now go into that directory

```
cd temp
```

- Changes the shell's notion of our current working directory

```
pwd
```

```
/home/mdv/programming1/temp
```

Manipulating Files and Directories II

- No files there yet:

```
ls —a
```

```
. ..
```

- Use the editor of your choice (emacs,vim) to create a file called earth.txt with the following contents:

```
Name:  Earth
Period:  365.26 days
Inclination:  0.00
Eccentricity:  0.02
Object:  Planet
```

Manipulating Files and Directories III

- Easiest way to create a similar file venus.txt is to copy the one we have

```
cp earth.txt venus.txt
```

```
ls -t
```

```
venus.txt    earth.txt
```

- Note: the -t option tells ls to list newest first
- Check the contents of the file using cat (short for concatenate)
- Just prints the contents of a file to the screen
- You can also use more or less

Manipulating Files and Directories IV

- Edit the file so that looks like:

```
Name:   Venus
Period:   224.70 days
Inclination:   3.39
Eccentricity:   0.01
Object:   Planet
```

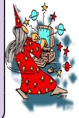
- Compare the sizes of the two files using wc (for word count) and Compare the two files using diff

```
wc earth.txt venus.txt

  4   9  69 earth.txt
  4   9  69 venus.txt
  8  18 138 total
```

```
diff earth.txt venus.txt

1,4c1,4
< Name: Earth
< Period: 365.26 days
< Inclination: 0.00
< Eccentricity: 0.02
---
> Name: Venus
> Period: 224.70 days
> Inclination: 3.39
> Eccentricity: 0.01
```



Manipulating Files and Directories V

- Linux does not care about filename extensions.
- cp earth. txt earth.pdf is valid although not a very sensible thing to do
- we can rename it using mv earth.pdf earth2. txt
- Removing a file can be done using rm, like for example rm earth2. txt
- A empty directory can be removed with rmdir or rm −r which recursively removed all files.

Wildcards

- Some characters (wildcards) mean special things to the shell
  - * matches zero or more characters
    - So ls *.f77 lists all the Fortran-77 files in a directory

```
    wc *.txt

      4    9   69 earth.txt
      4    9   69 venus.txt
      8   18  138 total
```

  - ? matches any single character
    - So ls ??.txt lists all the text files with two-letter prefixes
    - And ls ??.* lists all the files with two-letter prefixes, and any extension
  - on its own means the users home directory
  - harry means Harry's home directory
- Note: the shell expands wildcards before running commands
- Note: Be careful using rm in conjunction with *

Users

- Users have a user name and a password
- a user also has a home directory, and a shell program.
- Internally, the system uses so-called UID numbers to identify users.
- All this information is stored in the file /etc/passwd
- This also stores the user primary group id (GID) identifying a group to which the user belongs.
- A group is an arbitrary set of users
- A user can belong to several groups
- whoami,users,groups provide you with information regarding users and groups
- There is one special user with UID 0, called root
- This user is often called the super user because he can access all resources on the system, independently of any specific permissions

Ownership

- Each file has a user as owner and a group as group owner.
- Using chmod the owner can change permissions that determine the type of access (read, write or execute) ...
- allowed to three categories of users: the owner herself, the users belonging to the group owner group, and all other users.
- Note that ”execute” permission on a directory is interpreted as ”permission to traverse”

```
ls -l

-rw-r--r-- 1 mdv mdv      84 2007-09-27 23:08 earth.pdf

chmod g+w earth.txt
ls -l

-rw-rw-r-- 1 mdv mdv      84 2007-09-27 22:38 earth.txt
```