Introduction to Linux

Week 1 - Monday

A Brief History of Operating Systems

- The Dark Ages
  - No OS until 1960s
  - Manually loaded programs
  - Reboot after each program
- Batch OS
  - Unified application development across systems
  - Output via printer, later via monitor
  - I/O via magnetic tape or disk
  - Written in assembler (e.g., OS/360)
  - Multiprocess

A Brief History of Operating Systems

- Timesharing OS
  - Multiuser
  - Multics (1964)
    - Segmented memory
    - Paged virtual memory
    - Applications written in many languages
    - Shared multiprocess memory
- Personal Computer
  - Single machine for single user
  - OS must manage screen and input devices
  - Window, Icon, Menu, Pointing Device (WIMP, e.g., MacOS, 1984)
- Cutting-Edge OS
  - High performance computer (HPC) clusters (e.g., BlueGene/L at LLNL rated at 280.6 teraFLOPS)
  - Cell phones, video
  - Video games
  - Browsers

Why Knoppix?

- Multics (1964) → Unix (1970) → Minix (1987) → GNU/Linux (1991) → Knoppix (2003)
- Free software package via GNU
- Linux kernel (Unix-like OS)
- LiveCD Linux distribution
- Allows us to understand the workings of Unix
- Practice software construction via **command line interface** (CLI)

CLI vs. GUI

CLI	GUI
Steep learning curve	Intuitive
Pure control (e.g., scripting)	Limited Control
Cumbersome multitasking	Easy multitasking
Speed: Hack away at keys	Limited by pointing
Convenient remote access	Bulky remote access

Unix File System Layout

- Everything is a file (including devices)
- Tree structured hierarchy (with some exceptions)
- Lost?
  - man: get manual or man pages

The Basics: Moving Around

- pwd: print working directory
- cd: change working directory
- ~: home directory
- .: current directory
- /: root directory, or directory separator
- ..: parent directory

The Basics: Dealing with Files

- The basics continued…
  - mv: move a file (no "undo")
  - cp: copy a file
  - rm: remove a file
  - mkdir: make a directory
  - rmdir: remove a directory
  - ls: list contents of a directory
    - -d: list only directories
    - -a: list all files including hidden ones
    - -l: show long listing including permission info
    - -s: show size of each file, in blocks

The Basics: File Name Matching

- ?: matches any single character in a filename
- *: matches one or more characters in a filename
- []: matches any one of the characters between the brackets. Use '-' to separate a range of consecutive characters.

The Basics: History

- <up arrow>: previous command
- <tab>: auto-complete (tab twice to enumerate completions)
- !!: replace with previous command
- ![str]: refer to previous command with str
- ^[str]: replace with command referred to as str

The Basics: Look These Up

- cat
- head
- tail
- du
- ps
- kill
- diff
- cmp
- wc
- sort

The Basics: Redirection

- > file: write stdout to a file
- >> file: append stdout to a file
- < file: use contents of a file as stdin

The Basics: Changing File Attributes

- ln: create a link
  - Hard links: points to physical data
  - Soft links aka symbolic links (-s): points to a file
- touch: update access & modification time to current time
- chmod
  - read (r), write (w), executable (x)
  - User, group, others

The Basics: find

- -type: type of a file (e.g,, directory, symbolic link)
- -perm: permission of a file
- -name: name of a file
- -prune: don't descend into a directory
- -o: or
- -ls: list current file

vi

- Modes:
  - Normal: Enter commands
  - Insert: Insert text
  - Visual: Like normal, but you can highlight
  - Replace: Like insert, but you replace characters as you type
  - Recording: Record a sequence of keys

Emacs

- Almost like a Windows text editor, but much more powerful
- Usually easier to pick up
- Lots of extensions! The core of emacs contains a LISP interpreter

GNU Emacs Reference Card

Buffers

select another buffer	C-x b
list all buffers	C-x C-b
kill a buffer	C-x k

Transposing

transpose characters	C-t
transpose words	M-t
transpose lines	C-x C-t
transpose sexps	C-M-t

Spelling Check

check spelling of current word	M-$
check spelling of all words in region	M-x ispell-region
check spelling of entire buffer	M-x ispell-buffer

Tags

find a tag (a definition)	M-.
find next occurrence of tag	C-u M-.
specify a new tags file	M-x visit-tags-table
regexp search on all files in tags table	M-x tags-search
run query-replace on all the files	M-x tags-query-replace
continue last tags search or query-replace	M-,

Shells

execute a shell command	M-!
run a shell command on the region	M-|
filter region through a shell command	C-u M-|
start a shell in window *shell*	M-x shell

Rectangles

copy rectangle to register	C-x r r
kill rectangle	C-x r k
yank rectangle	C-x r y
open rectangle, shifting text right	C-x r o
blank out rectangle	C-x r c
prefix each line with a string	C-x r t

Abbrevs

add global abbrev	C-x a g
add mode-local abbrev	C-x a l
add global expansion for this abbrev	C-x a i g
add mode-local expansion for this abbrev	C-x a i l
explicitly expand abbrev	C-x a e
expand previous word dynamically	M-/

Regular Expressions

any single character except a newline	.	(dot)
zero or more repeats	*	
one or more repeats	+	
zero or one repeat	?	
quote regular expression special character c	\c	
alternative ("or")	|	
grouping	\(… \)	
same text as nth group	\n	
at word break	\b	
not at word break	\B	

	match start	match end
entity		
line	^	$
word	\<	\>
buffer	\`	\'

	match these	match others
class of characters		
explicit set	[…]	[^ …]
word-syntax character	\w	\W
character with syntax c	\sc	\Sc

International Character Sets

specify principal language	M-x set-language-environment
show all input methods	M-x list-input-methods
enable or disable input method	C-\
set coding system for next command	C-x RET c
show all coding systems	M-x list-coding-systems
choose preferred coding system	M-x prefer-coding-system

Info

enter the Info documentation reader	C-h i
find specified function or variable in Info	C-h C-i

Moving within a node:

scroll forward	SPC
scroll reverse	DEL
beginning of node	. (dot)

Moving between nodes:

next node	n
previous node	p
move up	u
select menu item by name	m
select nth menu item by number (1–9)	n
follow cross reference (return with l)	f
return to last node you saw	l
return to directory node	d
go to any node by name	g

Other:

run Info tutorial	h
quit Info	q
search nodes for regexp	M-s

Registers

save region in register	C-x r s
insert register contents into buffer	C-x r i
save value of point in register	C-x r SPC
jump to point saved in register	C-x r j

Keyboard Macros

start defining a keyboard macro	C-x (
end keyboard macro definition	C-x)
execute last-defined keyboard macro	C-x e
append to last keyboard macro	C-u C-x (
name last keyboard macro	M-x name-last-kbd-macro
insert Lisp definition in buffer	M-x insert-kbd-macro

Commands Dealing with Emacs Lisp

eval sexp before point	C-x C-e
eval current defun	C-M-x
eval region	M-x eval-region
read and eval minibuffer	M-:
load from standard system directory	M-x load-library

Simple Customization

customize variables and faces	M-x customize

Making global key bindings in Emacs Lisp (examples):

```
(global-set-key "\C-cg" 'goto-line)
(global-set-key "\M-#" 'query-replace-regexp)
```

Writing Commands

```
(defun command-name (args)
  "documentation" (interactive "template")
  body)
```

An example:

```
(defun this-line-to-top-of-window (line)
  "Reposition line point is on to top of window.
With ARG, put point on line ARG."
  (interactive "P")
  (recenter (if (null line)
                0
              (prefix-numeric-value line))))
```

The interactive spec says how to read arguments interactively. Type C-h f interactive for more details.