Using the Shell

Amir Masoumzadeh

CSI 402 – Systems Programming

February 1, 2018

Administrivia

- member of the GitHub organization?
- late submissions
- homework 2 will be out today
- wireless network situation

I/O redirection

- Standard interfaces
    - standard input (stdin = 0)
        - default: keyboard
    - standard output (stdout = 1)
        - default: screen
    - standard error (stderr = 2)
        - default: screen
- can override defaults with I/O redirection
    1. `command > file`
    2. `command >> file`
    3. `command 2> file`
    4. `command > file 2>&1`
    5. `command &> file`
    6. `command < file`
    7. `command < file1 > file2`

I/O redirection: pipes

- pipe: output from one command as input for another
  - `command1 | command2`
- filters
  - related commands: less, sort, uniq, wc, grep, head, tail, tee

Expansion

- path expansion (wildcards, . . .)
- tilde expansion (~)
- arithmetic expansion
  - `$((2*3))`
- brace expansion
  - comma-separated list, range of integers/letters (with optional increment)
  - can be repeated, nested
  - unlike path expansion, files do not need to exist
  - `a{r,c,sse}t` gives `art act asset`
  - `{a..e}` gives `a b c d e`
  - `{a..z..4}` gives `a e i m q u y`

Expansion (cont.)

- parameter expansion
  - echo $PATH
- command substitution
  - `ls -l $(which cp)`
  - you can use back quotes too: `` ls -l `which cp` ``
- character escaping (\)
- quoting
  - single quote (')
    - suppress all expansions
  - double quote (")
    - suppress all expansions except $, \, `

Permissions

- each user has a UID and GID
    - also may belong to multiple other groups
- file ownership
    - each file belongs to a user (owner) and a group
- permissions
    - r: read (4)
    - w: write (2)
    - x: execute (1)
- file permissions (mode bits)
    - rwx rwx rwx for user, group, others
    - can use octal: e.g., 755
- setuid, setgid, sticky bit
- note that superuser root is not restricted by access control
- related commands: chmod, umask

User management

- each user has a primary group
  - also may belong to multiple other groups
- related commands: su, sudo, chown, chgrp, passwd

Processes

- types
  - interactive
  - automatic
  - daemons
- attributes
  - PID
  - parent (PPID)
  - owner (RUID), group (RGID), EUID, EGID, ...
- related commands: `ps`, `top`, `jobs`

Exercise

Write a one-line command that

- determines how many "chromium" processes is currently running, and
- stores the count and/or any errors in "chromium-count.txt"

Can you change your command so that

- it stores details of all "chromium" processes, and
- includes the count at the end?

Manipulating processes

- terminate foreground process (`ctrl-c`)
- run process in background (`command &`)
- return a process to foreground (`fg %jobspec`)
- stopping a process (`ctrl-z`)
- resuming a process in background (`bg %jobspec`)

Signals

- send signal to process (`kill [-siginal] PID`)
    - `HUP(1), INT(2), KILL(9), TERM(15), CONT(18), STOP(19), TSTP(20)`
- send signal to multiple processes
    - `killall`
- shutdown the system
    - `halt`, `reboot`, `poweroff`, `shutdown`

Environment

- data stored by shell session
    - shell variables
    - environment variables
    - shell aliases
    - shell functions
- env. variables: `SHELL`, `PATH`, `USER`, ...
- related commands: `set`, `printenv`, `source`