ECS 98F - Advanced Command Line Usage

Noah Rose Ledesma

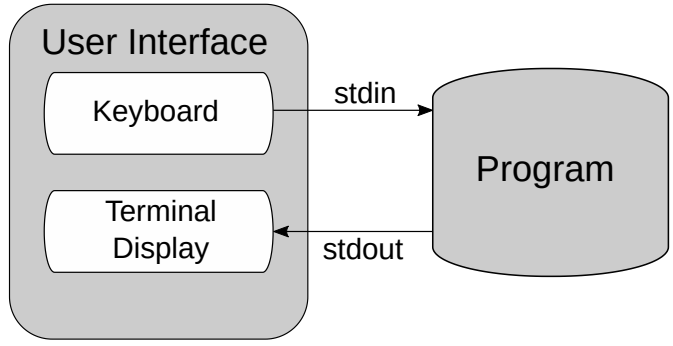
Agenda

- IO Streams
- Piping
- Unix `tail`, `grep`, and `cut`
- Root user & `sudo`
- Job control
- Exit codes & conditional execution
- Shell PATH variable
- SSH Keys & and moving files between machines
- Unix command conventions

Streams

Input and Output

- Programs have two primary streams
- `stdin` input stream
- `stdout` output stream



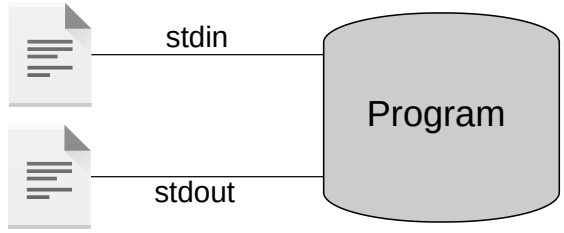
Stream Redirection

- Redirect `stdout` to a file:

```
$ echo "Hello World!" > hello.txt
$ cat hello.txt
Hello World!
```

- Redirect `stdin` to a file:

```
$ cat < hello.txt
Hello World!
```



- Append `stdout` to a file

```
$ cat < hello.txt >> hello_copy.txt
```

Data wrangling

```
$ ls -l /
total 61
lrwxrwxrwx   1 root root     7 Sep  2 15:30 bin -> usr/bin
drwxr-xr-x   4 root root   512 Dec 31  1969 boot
drwxr-xr-x   2 root root  4096 Jun 14  2019 bootloader
drwxr-xr-x  20 root root  3480 Nov 22 00:30 dev
drwxr-xr-x  85 root root  4096 Nov 20 15:51 etc
drwxr-xr-x   5 root root  4096 Oct 20 09:48 home
lrwxrwxrwx   1 root root     7 Sep  2 15:30 lib -> usr/lib
lrwxrwxrwx   1 root root     7 Sep  2 15:30 lib64 -> usr/lib
drwx------   2 root root 16384 Jun 14  2019 lost+found
drwxr-xr-x   3 root root  4096 Oct 23 13:56 media
drwxr-xr-x   2 root root  4096 May 23  2019 mnt
drwxr-xr-x  11 root root  4096 Oct 30 16:20 opt
dr-xr-xr-x 280 root root     0 Nov 22 00:29 proc
drwxr-x---  14 root root  4096 Oct 28 12:59 root
drwxr-xr-x  22 root root   600 Nov 22 00:32 run
lrwxrwxrwx   1 root root     7 Sep  2 15:30 sbin -> usr/bin
drwxr-xr-x   4 root root  4096 Jun 14  2019 srv
drwxr-xr-x   2 root root  4096 Oct 21  2019 switch
dr-xr-xr-x  13 root root     0 Nov 22 00:29 sys
drwxrwxrwt  12 root root   300 Nov 22 15:34 tmp
drwxr-xr-x  11 root root  4096 Nov 19 21:37 usr
drwxr-xr-x  13 root root  4096 Nov 20 15:51 var
```

Data wrangling

Tail command

- Prints the last `n` lines of its input
- E.g. `tail -n3`

Combining commands

- Make the output of `ls -l /` the input of `tail -n3`

```
$ ls -l / > temp.txt
$ tail -n3 < temp.txt
```

- Easier with piping

```
$ ls -l / | tail -n3
drwxrwxrwt  12 root root   300 Nov 22 15:41 tmp
drwxr-xr-x  11 root root  4096 Nov 19 21:37 usr
drwxr-xr-x  13 root root  4096 Nov 20 15:51 var
```

- Chain pipes for interesting manipulation

```
$ ls -l / | tail -n3 | grep usr | cut --delimiter=' ' -f10
21:37
```

grep & cut

grep

- Search the input stream for a string
- Outputs every line that contains the string

```
$ history | grep tail
407  ls -l / | tail -n3 | grep usr | cut --delimiter=' ' -f10
431  man tail
535  history | grep tail
```

cut

- Removes sections from each line of the input
- Useful when filtering columns from input

```
$ cat foo.txt
A,B,C,D
B,B,C,D
D,C,B,A
$ cat foo.txt | cut -d "," -f1,4
A,D
B,D
D,A
```

Demo

```
$ cat Unemployment_Rate_by_Age_Groups.csv \
    | tail -n+2 \
    | cut -d ',' -f4 \
    | sort -n \
    | uniq -d
```

- Remove the first line with `tail`
  - `-n+2` starts output at line 2
- Isolate fourth column from each line with `cut`
- Sort numerically with `sort`
- Remove duplicate years with `uniq`

With great power...

- Unix systems have a special `root` user with unrestricted permissions

```
$ ls -la /bin/ls
-rwxr-xr-x 1 root root 141936 Mar  6  2020 /bin/ls

$ rm -f /bin/ls
rm: cannot remove '/bin/ls': Permission denied
```

- Use `sudo` to perform actions as root
  - Short for "super user do"
  - Actions run as root can damage your system

```
$ sudo rm /bin/ls
$ ls
bash: /usr/bin/ls: No such file or directory
```

- System administration requires root privileges

```
$ sudo apt-get install python3
$ sudo reboot
$ sudo passwd noah
```

Sysfs

- Kernel parameters exposed in pseudo file system `/sys/`

```
$ cat /sys/class/input/mouse0/device/name
Logitech USB Optical Mouse
$ cat /sys/class/hwmon/hwmon3/temp1_crit
120000
$ cat /sys/class/leds/input2::scrolllock/brightness
0
```

- Some system variables can be changed *on-the-fly*

```
$ echo 1 > /sys/class/leds/input2::scrolllock/brightness
bash: /sys/class/leds/input2::scrolllock/brightness: Permission denied
```

```
$ sudo echo 1 > /sys/class/leds/input2::scrolllock/brightness
bash: /sys/class/leds/input2::scrolllock/brightness: Permission denied
```

```
$ sudo bash -c "echo 1 > /sys/class/leds/input2::scrolllock/brightness"
```

```
$ echo 1 | sudo tee /sys/class/leds/input2::scrolllock/brightness
```

- tee - read from standard input and write to standard output and files

Job Control

- Stop a running command with Ctrl+C

```
$ sleep 60
^C
```

- Pause a running command with Ctrl+Z

```
$ sleep 5
^Z
[1]+  Stopped                 sleep 5
$ echo "Hello"
Hello
```

- Resume in either foreground or background

```
$ fg
sleep 5



$
```

```
$ bg
[1]+ sleep 5 &
$ echo "What now?"
What now?
[1]+  Done                    sleep 5
```

Note: Ctrl+C is printed to the terminal as `^C`

Job Control

- Start a job in the background with `&`

```
$ cat /dev/random > /dev/null &
```

- List all jobs

```
$ jobs
[1]-  Stopped                 cat /dev/random > /dev/null
[2]+  Stopped                 sleep 10
```

- Terminate jobs with `kill`

```
kill %1
[1]+  Terminated              cat /dev/random > /dev/null
```

- Jobs are tied to the terminal session
  - Exiting terminal kills stopped & backgrounded jobs
- Protect background job from terminal exit:

```
$ nohup sleep 10 &
```

- Or for an already running job:

```
$ disown %1
```

Exit codes

- Exit code set when command finishes
- `0` indicates success. Any other value indicates error.

```
$ echo "Hello" > /dev/null
$ echo $?
0
$ cat missing.txt
cat: missing.txt: No such file or directory
$ echo $?
1
```

- Meaning of non-zero exit codes found in manpage
- Some commands specify meanings of exit codes in manpage

```
$ man ls
...
Exit status:
    0   if OK,

    1   if minor problems (e.g., cannot access subdirectory),

    2   if serious trouble (e.g., cannot access command-line argument).
...
```

Conditional execution

- Execute commands in series with `;`

```
$ echo -n "hello " ; echo "world"
hello world
```

- Conditionally on success with `&&`

```
$ cat / && echo "world"
cat: /: Is a directory
$ cd ~/slides/ && du -sh Makefile
4.0K    Makefile
```

- Conditionally on failure `||`

```
$ cat / || echo "Cat failed"
cat /: Is a directory
Cat failed
$ echo -n "hello" || echo "world"
hello
```

Path

- The shell searches $PATH when you enter a command

```
$ which ls
/usr/bin/ls
$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/bin:/opt/cuda-10.1/bin:/usr/lib/jvm/default/bin
 ...
$ echo $PATH | sed 's/:/\n/g'
/usr/local/sbin
/usr/local/bin
/usr/bin
/opt/cuda-10.1/bin
/usr/lib/jvm/default/bin
```

```
$ ./hello
Hello World!
$ hello
bash: hello: command not found
$ sudo mv hello /usr/local/bin
$ hello
Hello World!
```

More on SSH

- Run commands on remote machines & pipe the result

```
$ ssh noah@pc17.cs.ucdavis.edu ls | grep Downloads
```

- Passwordless authentication

```
$ ssh-copy-id -i ~/.ssh/id_ed25519.pub noah@pc17.cs.ucdavis.edu
```

- Copy files between machines

```
$ scp /path/to/local njrl@pc17.cs.ucdavis.edu:/path/to/remote
$ scp njrl@pc17.cs.ucdavis.edu:/path/to/remote /path/to/local
```

Useful Conventions

Many commands support these flags

- `-h` or `--help`
- `-V` or `--version`
- Configurable verbosity level: `-v` `-vv` `-vvv` or `--quiet`
- `-r` for recursive

Substitute filename for IO stream with `-`

```
$ ./hello | sed "s/Hello/Hewwo/" | diff hello.txt -
```

For fun: fortune & cowsay

```
$ sudo apt-get install fortune cowsay
...
$ fortune | cowsay
 _____________________________________
/ Why is the alphabet in that order? Is \
| it because of that song?              |
|                                       |
\ -- Steven Wright                      /
 -------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
$ fortune | cowsay -d
 ____________________________________
/ Most lectures have a happy ending. \
\ Everyone's glad when they're over. /
 ------------------------------------
        \   ^__^
         \  (xx)\_______
            (__)\       )\/\
             U  ||----w |
                ||     ||
```