

Basic Shell Commands

Bok, Jong Soon
javaexpert@nate.com
<https://github.com/swacademy/Ubuntu>

Focusing on Linux Commands

- These days, many important tasks in Linux can be done from both graphical interfaces and from commands.
- However, the command line has always been, and still remains, the interface of choice for Linux power users.

Focusing on Linux Commands (Cont.)

- For the following cases, Will probably need to rely on the command line:
 - Almost any time something goes wrong
 - Remote systems administration
 - Features not supported by GUI
 - GUI is broken or not installed

Finding Commands

■ `$ echo $PATH`

- Show the current `PATH`

■ `$ which`

- Locate a command

■ `$ find`

- Search for files in a directory hierarchy.

■ `$ whereis`

- Locate the binary, source, and manual page files for a command.

Finding Commands (Cont.)

■ `$ locate`

- List files in databases that match a pattern.

■ `$ apropos`

- Search the manual page names and descriptions.

■ `$ man 8 umount`

- View section 8 of the `man` page for `umount` (type `q` to quit)

which Command

- Shows the full path of commands.
- Syntax
 - **which [OPTION] command**
- Options
 - **-a, --all** : Print all matching executables in **PATH**
- Examples
 - **\$ which ls**

find Command

- Search for files in a directory hierarchy.

- Syntax

- `find [PATH] [OPTION] filename [EXPRESSION]`

Searching Files with `find`

- Searches the root file system (`/`) recursively for files named `*stat`:

```
$ find / -name '*stat' -print
```

- Can find files based on timestamps.
- Finds files in `/usr/bin/` that have been accessed in the past 2 minutes:

```
$ find /usr/bin/ -amin -2 -print
```

- Finds files that have not been accessed in `/home/chris` for more than 60 days:

```
$ find /home/chris/ -atime +60
```


Searching Files with `find` (Cont.)

- Use the `-type d` option to find directories.
- Finds all directories under `/etc`:

```
$ find /etc -type d -print
```
- Finds files in `/sbin` with permissions that match 750:

```
$ find /sbin/ -perm 750 -print
```
- The `xargs` option lets act on the files found:

```
$ find . -perm 644 -print | xargs ls -l
```

Searching Files with `find` (Cont.)

- Finds all files that are greater than 8 KB (+8k), lists those files from largest to smallest (`ls -lS`) and directs that list to a file (`/tmp/bigfiles.txt`):

```
$ find . -size +8k -print | xargs ls -lS > /tmp/bigfiles.txt
```

```
$ head -5 /tmp/bigfiles.txt
```

- Examples

- `$ find / -name passwd`
- `$ find / -name passwd 2> /dev/null`
- `$ find ~ -name 'core*' -exec rm {} \;`

Lab 1 : Using `find` command

find command

1. `$ find /bin -name ls`
2. `$ find /home -user instructor`

1. `$ cd temp`
2. `$ touch doc1 doc2 doc3 doc4`
3. `$ find . -user instructor`
4. `$ find . -user instructor -exec rm {} \;`

1. `$ touch doc1 doc2 doc3 doc4`
2. `$ find . -user instructor -ok rm {} \;`

whereis Command

- Locate the binary, source, and manual page files for a command.
- Syntax
 - **whereis [OPTION] filename**
- Options
 - **-b** : Search only for binaries.
 - **-m** : Search only for manual sections.
 - **-s** : Search only for sources.
 - **-B** : Change or limit the place for binaries.
 - **-M** : Change or limit the place for manual secs.
 - **-S** : Change or limit the place for sources.

Lab 2 : Using **whereis** command

whereis Command

- `$ whereis find`
- `$ whereis perl`
- `$ whereis -b find`
- `$ whereis -m find`

apropos Command

- Search the manual page names and descriptions.
- Syntax
 - `apropos keyword ...`
- Examples
 - `$ apropos zip`

locate Command

- List files in databases that match a pattern.
- Print out the complete path along with the filename containing the string.
- Before use, must build the index used by locate.
- This index makes it possible to find files quickly.
- The command to build this index is : **updatedb**
- Syntax
 - **\$ sudo -u root updatedb**
- The locate database is saved in the following directory in Ubuntu :
/var/lib/mlocate/mlocate.db

locate Command (Cont.)

■ Syntax

- `locate filename or pattern`

■ Pattern (with `-r` option)

- `^` : beginning of line
- `$` : end of line
- `[]` : match on a class of characters

■ Examples

- `$ locate passwd`
- `$ locate -r /passwd$`

grep Command

- Prints the matching lines.
- Syntax
 - `grep [options] pattern [filename]`
 - *`-i, --ignore-case`*
 - *`-l, --files-with-match`*
 - *`-n, --line-number`*
- Examples
 - `$ grep root /etc/passwd`
 - `$ grep -n unix ~/.txt`
 - `$ grep -l hello*.c`

Lab 3 : Using `grep` command

grep Command

- `$ cd temp`
- `$ cp /etc/services data`
- `$ grep SSL data`
- `$ grep -n SSL data`

Using **help** Messages

- Nearly all commands on a Linux system print some form of brief usage information if asked to.
- Frequently, the way to ask for this usage info is by way of the **-h** or **--help** argument to the command, and nothing more.

```
$ ls --help
```

- Since there is so much information printed by the **--help** flag, you can again use a pager to limit the output to one screen at a time.

```
$ ls --help | more
```

Using **man** Pages

- Can use the **apropos** command to search the **man** page database for any keyword or group of characters.
- The output will show **man** page sections which contain the word you supply to **apropos**.

```
$ apropos crontab
```

```
$ man man
```

man Page Sections

Number	Types of Pages
1	Executable programs or shell commands
2	System calls (functions provided by the kernel)
3	Library calls (functions within program libraries)
4	Special files (usually found in /dev)
5	File formats and conventions such as /etc/passwd
6	Games
7	Miscellaneous (including macro packages and conventions), such as man(7), groff(7)
8	System administration commands (usually only for root)
9	Kernel routines [Non standard]

Using **man** Pages (Cont.)

- Can view the **man** pages from those sections by passing the section number as an argument to the **man** command.

```
$ man 5 crontab
```

- If we omit the section number, **man** will return the **man** page from the first section it finds.

```
$ man crontab
```

man Command Options

Option	Description
<code>man -a crontab</code>	Shows all man page sections, in succession, for crontab
<code>man 5 crontab</code>	Shows the section 5 man page for crontab
<code>man crontab -P more</code>	Uses the pager program more for paging through the crontab man page
<code>man -f crontab</code>	Equivalent to the whatis command
<code>man -k crontab</code>	Equivalent to the apropos command

whatis Command

- Is another **man** page searching utility.
- It is different from **apropos** in that it only prints **man** page descriptions that match the keyword you type in.
- Running the **apropos** command for the **route** command returns three different **man** pages where a reference to the word **route** was found:

```
$ apropos route
```

whatis Command (Cont.)

- In running **whatis** for the **route** command, only the section 8 **man** page for the **route** command is returned.

```
$ whatis route
```

Using **info** Documents

- Read documentation in Info format.
- Can enter the info database by simply typing the **info** command or by opening a particular component (use the **q** key to quit the info utility).

```
$ info ls
```

shutdown Command

- Bring the system down.
- Syntax
 - **shutdown [OPTION] ... TIME [MESSAGE]**
- Options
 - **-k** : Don't really shutdown; only send the warning messages to everybody.
 - **-r** : Reboot after shutdown.
 - **-h** : Halt after shutdown.
 - **-H** : Requests that the system be halted after it has been brought down.

shutdown Command (Cont.)

■ Options

- **-P** : Requests that the system be powered off after it has been brought down.
- **-c** : Cancel an already running shutdown.

■ Example

- **\$ sudo shutdown -h +10**
- **\$ sudo shutdown +3 "System is going down"**
- **\$ sudo shutdown -r now**
- **\$ sudo shutdown -h 20:30 System will be halted**

halt Command

- Halt system.
- Syntax
 - **halt [OPTION]**
- Options
 - **-p** : When halting the system, do a poweroff. This is the default when halt is called as **poweroff**.
 - **-w** : Don't actually halt but only write the wtmp record (in the **/var/log/wtmp** file).
- Example
 - **\$ sudo halt -p**

poweroff Command

- System power off.
- Syntax
 - **poweroff [OPTION]**
- Options
 - **-p** : When halting the system, do a poweroff.
 - **-w** : Don't actually halt but only write the wtmp record (in the **/var/log/wtmp** file).
- Example
 - **\$ sudo poweroff -p**

reboot Command

- Reboot system.
- Syntax
 - `reboot`
- Example
 - `$ sudo reboot`

init Command

- Is the parent of all processes.
- Refer to <https://askubuntu.com/questions/1090882/where-is-the-default-runlevel-on-ubuntu-18-04>
- **\$ systemctl list-units --type=target --all | cat**

```
instructor@Ubuntu-Desktop:~$ systemctl list-units --type=target --all | cat
UNIT                                LOAD    ACTIVE SUB    DESCRIPTION
● all.target                        not-found inactive dead    all.target
basic.target                       loaded active active Basic System
cryptsetup.target                 loaded active active Local Encrypted Volumes
emergency.target                 loaded inactive dead Emergency Mode
getty-pre.target                 loaded inactive dead Login Prompts (Pre)
getty.target                     loaded active active Login Prompts
graphical.target                 loaded active active Graphical Interface
local-fs-pre.target              loaded active active Local File Systems (Pre)
local-fs.target                  loaded active active Local File Systems
multi-user.target                loaded active active Multi-User System
network-online.target            loaded active active Network is Online
network-pre.target              loaded inactive dead Network (Pre)
network.target                   loaded active active Network
nss-lookup.target                loaded active active Host and Network Name Lookups
nss-user-lookup.target           loaded active active User and Group Name Lookups
paths.target                     loaded active active Paths
remote-fs-pre.target             loaded inactive dead Remote File Systems (Pre)
```



init Command (Cont.)

■ Default runlevel

runlevel	Description
0	System halt ; no activity, the system can be safely powered down.
1	Single user; rarely used.
2	Multiple users, no NFS
3	Multiple users, command line interface; the standard runlevel for most Linux-based server hardware.
4	Unused.
5	Multiple users, GUI; the standard runlevel for most Linux-based desktop systems.
6	Reboot; used when restarting the system.

■ `$ sudo init 0` : System shutdown

■ `$ sudo init 6` : System reboot



init Command (Cont.)

- For desktop systems the default is *graphical.target* :
 - \$ **systemctl get-default**
- Can switch the target whenever you want with *systemctl isolate* command.
- The example below will switch to text-based *multi-user.target*
 - \$ **sudo systemctl isolate multi-user.target**
- Default target can be changed the following way :
 - \$ **sudo systemctl set-default multi-user.target**

who Command

- Show who is logged on.

- Syntax

- **who [OPTION]**

- Options

- **-a, --all**: Same as **-b -d --login -p -r -t -T -u**
 - **-b, --boot**: Time of last system boot.
 - **-d, --dead**: Print dead processes
 - **--login**: Print system login processes.

who Command (Cont.)

■ Options

- **-p, --process**: Print active processes spawned by init
- **-r, --runlevel**: Print current runlevel.
- **-t, --time**: Print last system clock change
- **-u, --users**: List users logged in.
- **-H, --heading**: Print line of column headings
- **-q, --count**: All login names and number of users logged on

Lab 4 : Using **who** command

who Command

- \$ who
- \$ who -H
- \$ who -b
- \$ who -r
- \$ who -a

uname Command

- Print system information, print information about the machine and operating system it is run on.
- Refer to `/proc/version`
- Syntax
 - `uname [OPTION]`
- Options
 - `-a`, `--all` : Print all information, in the following order :
KERNEL-NAME NODENAME KERNEL-RELEASE
KERNEL-VERSION MACHINE PROCESSOR
HARDWARE-PLATFORM OPERATING-SYSTEM

uname Command

■ Options

- **-s** : Print the kernel name
- **-n** : Print the network node hostname
- **-r** : Print the kernel release
- **-v** : Print the kernel version
- **-m** : Print the machine hardware name
- **-p** : Print the processor type
- **-i** : Print the hardware platform
- **-o** : Print the operating system

last Command

- Show listing of last logged in users.
- By default it shows a log of the file `/var/log/btmp`
- Syntax
 - `last [OPTION]`
- Options
 - `-n` : This is a count telling **last** how many lines to show.
 - `-t YYYYMMDDHHMMSS`: Display the state of logins as of the specified time.

date Command

- Display the current time in the given FORMAT, or set the system date.
- Syntax
 - `date [OPTION]... [+FORMAT1]`
`date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]`
`]`
- Options
 - `-s, --set=STRING` : Set time described by STRING.
 - `-u, --utc, --universal` : Print or set Coordinated Universal Time

1. FORMAT refer to <http://linux.die.net/man/1/date>

Lab 5 : Using `date` command

date Command

- `$ date`
- `$ date '+%A %B %d %G'`
- `$ date '+The date today is %F.'`
- `$ date --date='4 weeks'`
- `$ date --date='8 months 3 days'`
- `$ date 052112452016`
- `$ date -s '2016-10-20 23:34:21'`

cal Command

- Displays a simple calendar.

- Syntax

- `cal [-smjy13] [[month] year]`

- Options

- `-1` : Display single month output. (This is the default.)
 - `-3` : Display prev/current/next month output.
 - `-s` : Display Sunday as the first day of the week. (This is the default.)
 - `-m` : Display Monday as the first day of the week.
 - `-j` : Display Julian dates (days one-based, numbered from January 1).
 - `-y` : Display a calendar for the current year.

whoami Command

- Print effective userid.
- Syntax
 - **whoami**

pwd Command

- Print the full filename of the current working directory.
- Syntax
 - `pwd`

clear Command

- Clear the terminal screen.
- Syntax
 - **clear**