Linux Basics

Yu Sheng

Center for Statistical Science

Fall 2018

Why should we learn Linux?

- ~~So that you can play cool and impress your friends.~~

- Most computing servers use Linux, so it's necessary to learn some basics.

- Linux is favored by open source communities, so the Linux platform usually gets the latest computing technologies first.

- Mac OS X is highly similar to Linux, which has a large user base (especially in the U.S.) – another reason why Linux gets the latest computing techs first.

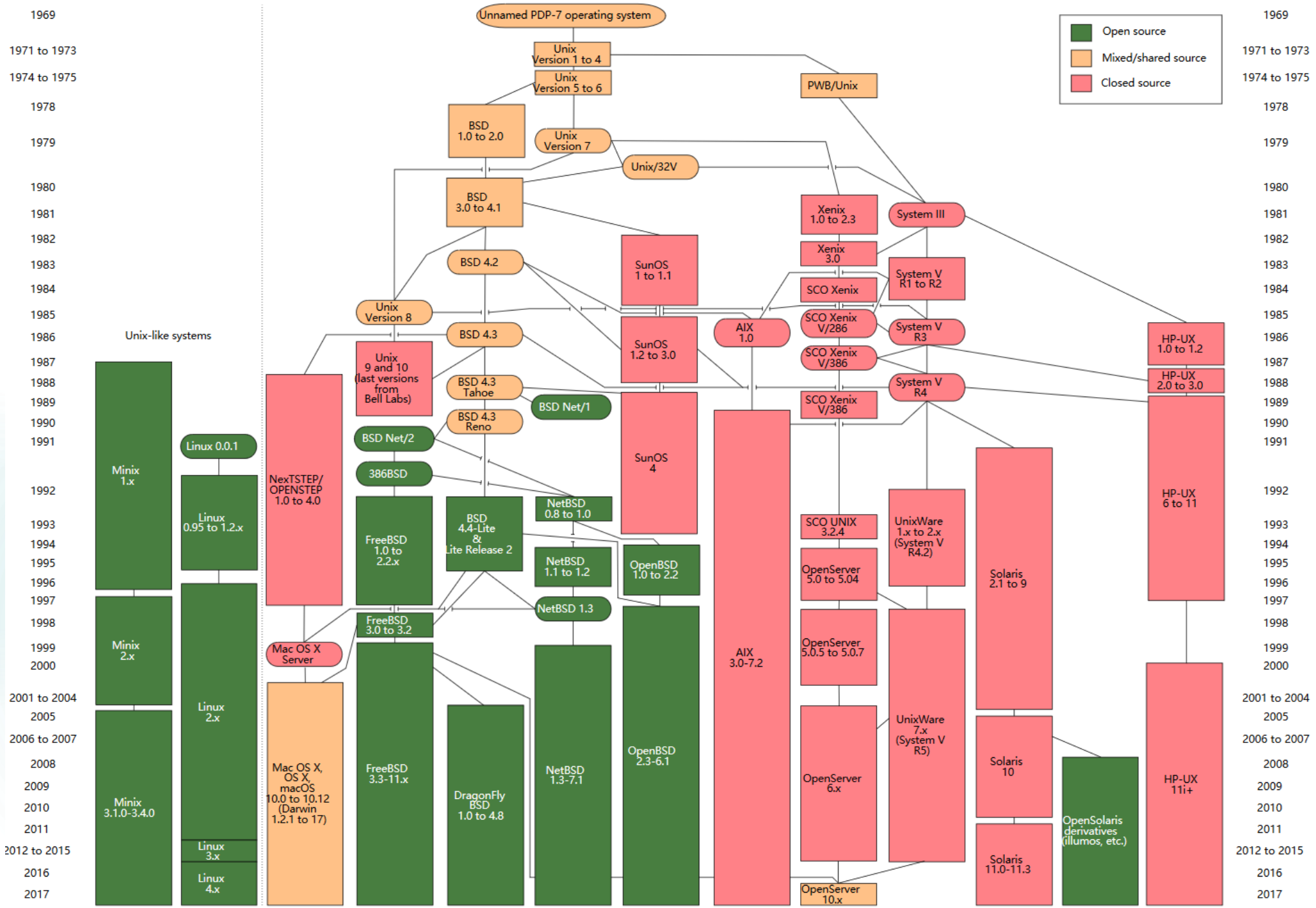
- Windows 10 has a Linux subsystem, so now you can use your Linux skills on Windows, too.

A bit of history…

- Linux is a clone of Unix – completely rewritten.
- Unix was originally developed at AT&T Bell Labs in the 1970s.
- Due to an earlier antitrust case, AT&T was required to license the operating system's source code to anyone who asked basically for free.
  - Unix grew quickly and became widely adopted by academic institutions and businesses.

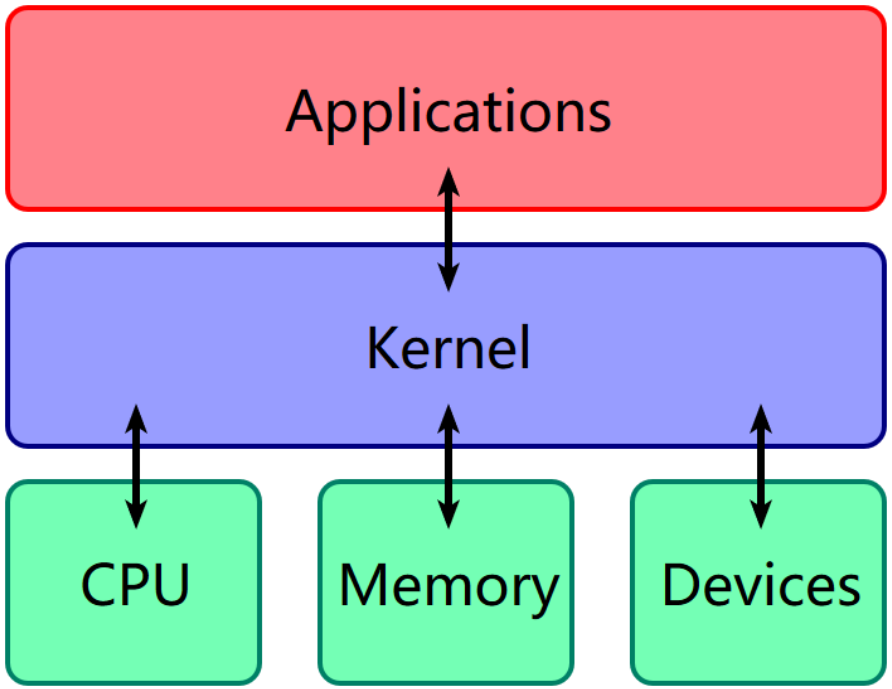
A bit of history…

- A second antitrust case in 1984 separated Bell Labs from AT&T.
  - Bell Labs began selling Unix as a proprietary product, where users were not legally allowed to modify Unix.
- Hobbyists became unhappy and started the Free Software Movement.
  - Unix branched into BSD, System V, etc.
  - The GNU Project had the goal of creating a "complete Unix-compatible software system" composed entirely of free software, and completed many components, such as compilers, text editors, a Unix shell, and a windowing system.
  - Linus Torvalds developed the Linux kernel in 1991 when he was a student.


Open source
Mixed/shared source
Closed source
Unix-like systems
1969
1971 to 1973
1974 to 1975
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001 to 2004
2005
2006 to 2007
2008
2009
2010
2011
2012 to 2015
2016
2017
Unnamed PDP-7 operating system
Unix Version 1 to 4
Unix Version 5 to 6
Unix Version 7
PWB/Unix
BSD 1.0 to 2.0
Unix/32V
BSD 3.0 to 4.1
Xenix 1.0 to 2.3
Xenix 3.0
System III
BSD 4.2
SunOS 1 to 1.1
SCO Xenix
System V R1 to R2
Unix Version 8
AIX 1.0
SCO Xenix V/286
System V R3
BSD 4.3
SunOS 1.2 to 3.0
SCO Xenix V/386
HP-UX 1.0 to 1.2
Unix 9 and 10 (last versions from Bell Labs)
BSD 4.3 Tahoe
BSD Net/1
System V R4
HP-UX 2.0 to 3.0
BSD 4.3 Reno
BSD Net/2
SunOS 4
386BSD
NetBSD 0.8 to 1.0
HP-UX 6 to 11
SCO UNIX 3.2.4
UnixWare 1.x to 2.x (System V R4.2)
Minix 1.x
Linux 0.0.1
NextSTEP/ OPENSTEP 1.0 to 4.0
Linux 0.95 to 1.2.x
FreeBSD 1.0 to 2.2.x
BSD 4.4-Lite & Lite Release 2
NetBSD 1.1 to 1.2
OpenBSD 1.0 to 2.2
OpenServer 5.0 to 5.04
Solaris 2.1 to 9
NetBSD 1.3
FreeBSD 3.0 to 3.2
Minix 2.x
Linux 2.x
Mac OS X Server
OpenServer 5.0.5 to 5.0.7
AIX 3.0-7.2
UnixWare 7.x (System V R5)
Mac OS X, OS X, macOS 10.0 to 10.12 (Darwin 1.2.1 to 17)
FreeBSD 3.3-11.x
DragonFly BSD 1.0 to 4.8
NetBSD 1.3-7.1
OpenBSD 2.3-6.1
OpenServer 6.x
Solaris 10
HP-UX 11i+
Minix 3.1.0-3.4.0
Linux 3.x
Linux 4.x
OpenSolaris derivatives (illumos, etc.)
Solaris 11.0-11.3
OpenServer 10.x

Basic concepts

- Linux may refer to
  - The Linux kernel



Basic concepts

- Linux may refer to
  - The Linux kernel
  - A Linux distribution/ecosystem

User mode	**User applications**	For example, bash, LibreOffice, GIMP, Blender, 0 A.D., Mozilla Firefox, etc.				
	Low-level system components:	**System daemons**: *systemd*, *runit*, *logind*, *networkd*, *PulseAudio*, ...	**Windowing system**: *X11*, *Wayland*, *SurfaceFlinger* (Android)	**Other libraries:** *GTK+*, *Qt*, *EFL*, *SDL*, *SFML*, *FLTK*, *GNUstep*, etc.	**Graphics**: *Mesa*, *AMD Catalyst*, ...	
	C standard library	`open()`, `exec()`, `sbrk()`, `socket()`, `fopen()`, `calloc()`, ... (up to 2000 subroutines) *glibc* aims to be POSIX/SUS-compatible, *uClibc* targets embedded systems, *bionic* written for Android, etc.				
Kernel mode	**Linux kernel**	`stat`, `splice`, `dup`, `read`, `open`, `ioctl`, `write`, `mmap`, `close`, `exit`, etc. (about 380 system calls) The Linux kernel System Call Interface (SCI, aims to be POSIX/SUS-compatible)				
		Process scheduling subsystem	IPC subsystem	Memory management subsystem	Virtual files subsystem	Network subsystem
		Other components: ALSA, DRI, evdev, LVM, device mapper, Linux Network Scheduler, Netfilter Linux Security Modules: *SELinux*, *TOMOYO*, *AppArmor*, *Smack*				

Basic concepts

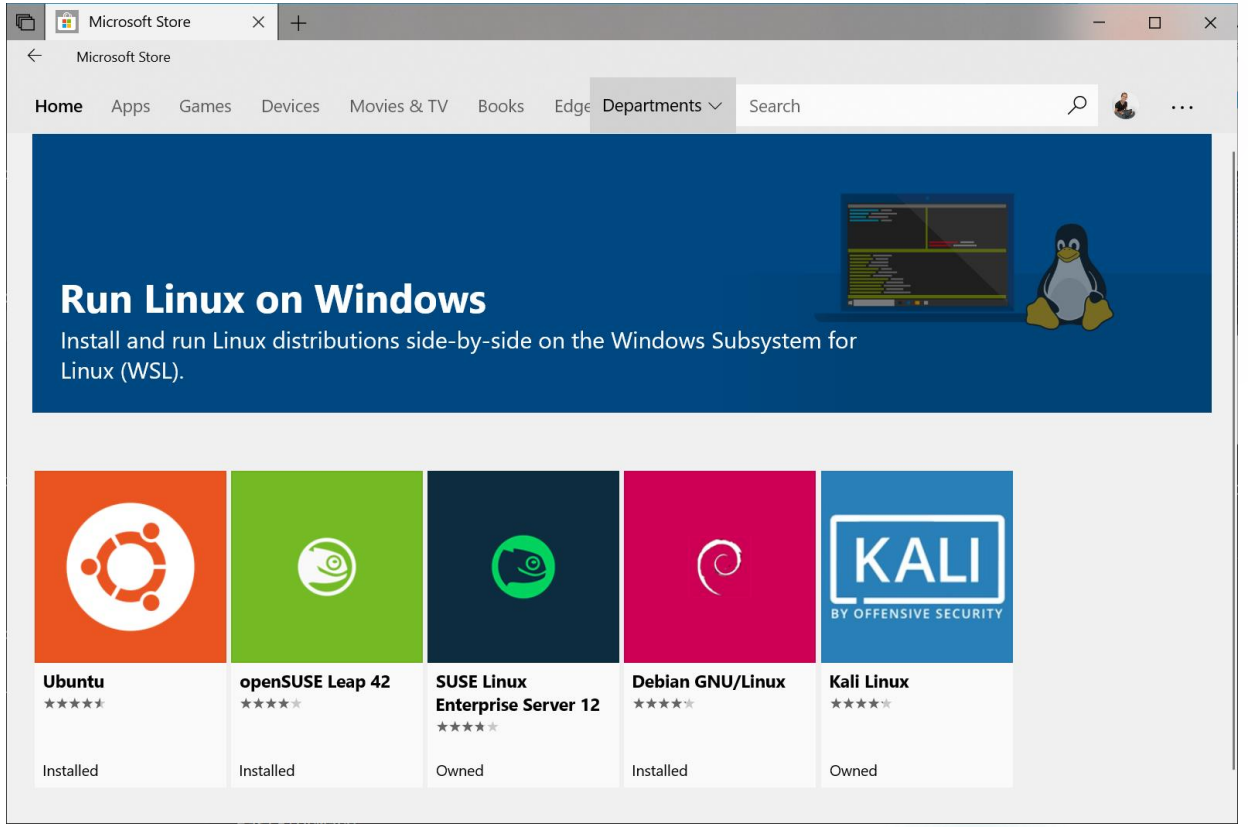
- Linux may refer to
  - The Linux kernel
  - A Linux distribution/ecosystem
  - The shell commands

Getting your copy of “Linux”

- For learning purposes, you can
  - Install a real Linux system (Ubuntu, CentOS, etc.) on your computer (dual systems, virtual boxes…).
  - Rent a cloud server for a small fee.
  - Mac OS X is already Unix-based.
  - Windows
    - Early systems – Cygwin
    - Windows 10 – Windows Subsystem for Linux (WSL)

Windows Subsystem for Linux (WSL)

1. Open PowerShell as Administrator and run:
   `Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux`
2. Restart your computer.
3. Install a Linux distro of your choice from Windows Store.



Directory structure

/ - Linux File System Root

/bin	User Binary Files
/boot	Boot Loader Files
/dev	Device Files
/etc	Configuration Files
/home	Home Directories
/lib	System Libraries
/media	Mount point for removable media
/mnt	Mount point for temporary file systems
/opt	Optional Addon Applications
/sbin	System Binaries
/srv	Service Data
/tmp	Temporary Files
/usr	User programs
/var	Variable Files
/root	Root User Directory
/proc	Process Information
/lost + found	

Navigating

- pwd – print working directory
- ls – show the content of the current directory
  - -a – show hidden files
  - -l – show more details
- cd – change directory
  - . – the current dir
  - .. – the parent dir
  - ~ - your home dir
  - cd without argument is the same as cd ~

Working with files and directories

- mkdir – create a folder at the current directory

用file来看一个文件的类型

- rm – remove files or directories
  - -r – remove directory recursively
  - -f – remove without prompt

Working with files and directories

- cp – copy files or directories
  - `cp picture.jpg picture-02.jpg`
  - `cp ~/pictures/picture.jpg ~/backup/picture.jpg`
  - `cp ~/pictures/picture-*.jpg ~/picture-backup`
  - `cp ~/pictures/picture-01.jpg ~/pictures/picture-02.jpg ~/picture-backup`
  - `cp -r ~/files ~/files-backup`
    - If the directory **files-backup** already exists, the directory **files** will be placed inside.
    - If **files-backup** does *not* already exist, it *will* be created and the *contents* of the **files** directory will be placed inside it.
- mv – move files or directories
  - Similar to cp
  - Can be used to rename files.

Viewing file contents

- cat – displays the file content
- less – works better with long files
  - Press u/d/space_bar to go up and down.
  - Press q to quit.
- head – display the first ten lines of a file
  - head -5 – the first 5 lines
- Tail – display the last ten lines of a file
  - head -5 – the last 5 lines

Searching the contents of a file

- In less, press '/', then type the pattern for search.
  - Press 'n' to search for the next occurrence.

- grep 'pattern' filename
  - **-i** ignore upper/lower case distinctions
  - **-v** display those lines that do NOT match
  - **-n** precede each matching line with the line number
  - **-c** print only the total count of matched lines

Text editing

- Linux comes with a powerful text editor `vi`
  - `vi` is also notoriously hard to learn
- We most likely only do light editing in a pure command line environment, so `nano` is more intuitive and recommended.

Redirection

Command	Meaning
>	redirect standard output to a file
>>	append standard output to a file
<	redirect standard input from a file
|	pipe the output of command1 to the input of command2

Redirection

- Example
  - Enter `cat`, then enter a few words, then press `^D` (end-of-file)
  - Do the same but with `cat > list1.txt`
  - Add more words with `cat >> list1.txt`
  - Similarly create `list2.txt`
  - Combine and sort and save the first 3 with `cat list1.txt list2.txt | sort | head -3 > top3.txt`

Wild cards

- * – matches 0 or more characters.
  - E.g., remove the previous lists with `rm *.txt`
- ? – matches exactly 1 character.

Get help

- man – check the manual of a command
- whatis – gives a one line description of a command
- whereis – locate the binary or source or manual pages for a command

Killing a process

- Killing means forcing a process to terminate. It is different from a normal quit.
- To kill a job running in the foreground, type ^C.
  - E.g., `sleep 1000`
- Killing a job running in the background with `kill`
  - `sleep 1000 &`
  - `ps` (find pid of `sleep`)
  - `kill [pid of sleep]`

Jobs

- The `&` at the end puts a process at the background.
- A processing running at the background is called a job.
- Use `jobs` to see the jobs running.
- Kill a job by job number: `kill %[job number]`

Connecting to a remote server

- On Windows, use PuTTY
- On Unix-like systems, use `ssh user_name@server_location`

Installing software

- The most common way to install a software is by using the *package management system*.

Operating System	Format	Tool(s)
Debian	`.deb`	`apt`, `apt-cache`, `apt-get`, `dpkg`
Ubuntu	`.deb`	`apt`, `apt-cache`, `apt-get`, `dpkg`
CentOS	`.rpm`	`yum`
Fedora	`.rpm`	`dnf`
FreeBSD	Ports, `.txz`	`make`, `pkg`

- It’s generally best to update package information before installing.
  1. sudo apt update
  2. sudo apt install [package_name]

Installing software

- Install a package from a local file

System	Command	Notes
Debian / Ubuntu	`sudo dpkg -i package.deb`	
	`sudo apt-get install -y gdebi && sudo gdebi package.deb`	Installs and uses `gdebi` to install `package.deb` and retrieve any missing dependencies.
CentOS	`sudo yum install package.rpm`	
Fedora	`sudo dnf install package.rpm`	
FreeBSD Packages	`sudo pkg add package.txz`	
	`sudo pkg add -f package.txz`	Installs package even if already installed.

Compile locally

- Some Linux software not in the package management system need to be compiled locally.

- To build a software, one must have compilers installed first.
  - `sudo apt update`
  - `sudo apt install build-essential`

- Download the files with `wget`
  - Or maybe you want to transfer the code to the server from your local computer. You can use ftp tools like FileZilla.

Example

- Create a Downloads folder at your home directory.
- Download http://www.ee.surrey.ac.uk/Teaching/Unix/units-1.74.tar.gz

Compressed files

- Linux compressed files are usually named like .tar.gz or .tar.bz2
  - .tar is an archive file, called a tarball, which is a single file that packages many.
  - .gz or .bz2 is the compression format, using gzip or bzip2.

- A single tar command can handle the archiving and zipping at the same time.

```
tar -czvf name-of-archive.tar.gz /path/to/directory-or-file
```

  - -c – create an archive
  - -z/j – compress the archive with gzip/bzip2
  - -v – verbose
  - -f – allow you to specify the filename of the archive

Compressed files

- tar can also used to unzip compressed files.
  - Replace -c with -x to extract.
  - tar -xzvf archive.tar.gz will extract locally
  - tar -xzvf archive.tar.gz -C /tmp will extract to the /tmp folder
- Linux also has `zip` and `unzip`

Example

- Unzip the downloaded file.

Configure and make

- When a program gets large, the compiling order and logic becomes complex. `Makefile` is a file provided with a program source code that tells the system which compilers to use, the compiling order, where to install, etc.
- When `Makefile` is present, use the command `make` to compile, and `make install` to install the program to specified location.
- Additionally, a program usually provides a standardized `configure` script instead of Makefile, which will check your environment (such as availability of compilers) and generate the Makefile.
- Normally you only need to specify the `--prefix` and the `--exec-prefix` options for installation location.

Example

- Compile and install the unit conversion program under ~/Programs/units
- Validate that the program has been installed correctly.