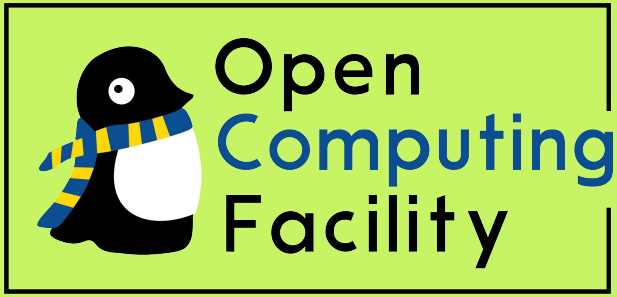


Lecture 3

Packaging

albert, adi

(slide credits dkessler, mdcha, ethanhs)

Course Resources

- Your facilitators!
- Ed, Gradescope
- OCF Slack (ocf.io/slack) or Discord (ocf.io/discord) #decal-general
- All materials available at decal.ocf.io
- Ask questions / work on lab with us during lab sessions! (Tuesday 8-9pm in OCF Lab)

Topics

- Intro to Distributions/Debian
- Packaging
- Compilation

Linux Distributions and Debian

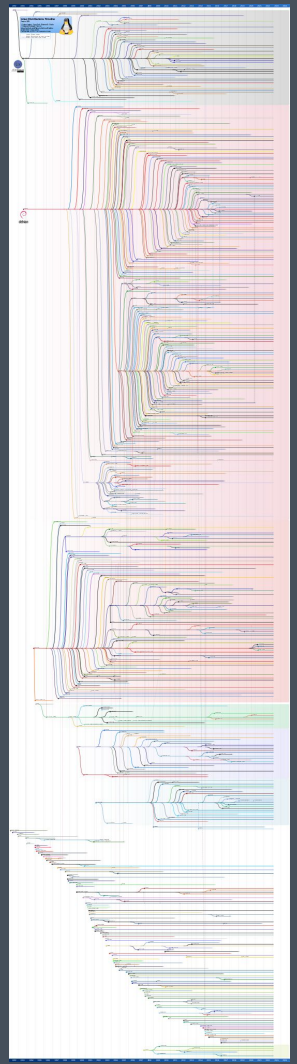
Distributions?

- Basically the Linux kernel + other software = operating system
- Because there are many different configurations of the kernel and other software, Linux OSes have a term called “distributions”

Distribution Family Tree

- Distributions are spun off a lot. “Derivatives”
  - Debian -> Ubuntu, RHEL -> Fedora, Arch -> Manjaro

https://upload.wikimedia.org/wikipedia/commons/1/1b/Linux_Distribution_Timeline.svg



Debian

- Debuted in late 1993
- Why use it?
  - Stable - new release every 2 yrs.
  - User-friendly - works out-of-the-box
  - Respects your privacy (cough cough pre-16.04 vanilla Ubuntu cough)
- ~~Cute naming scheme~~
- Widely used

Other Distributions?

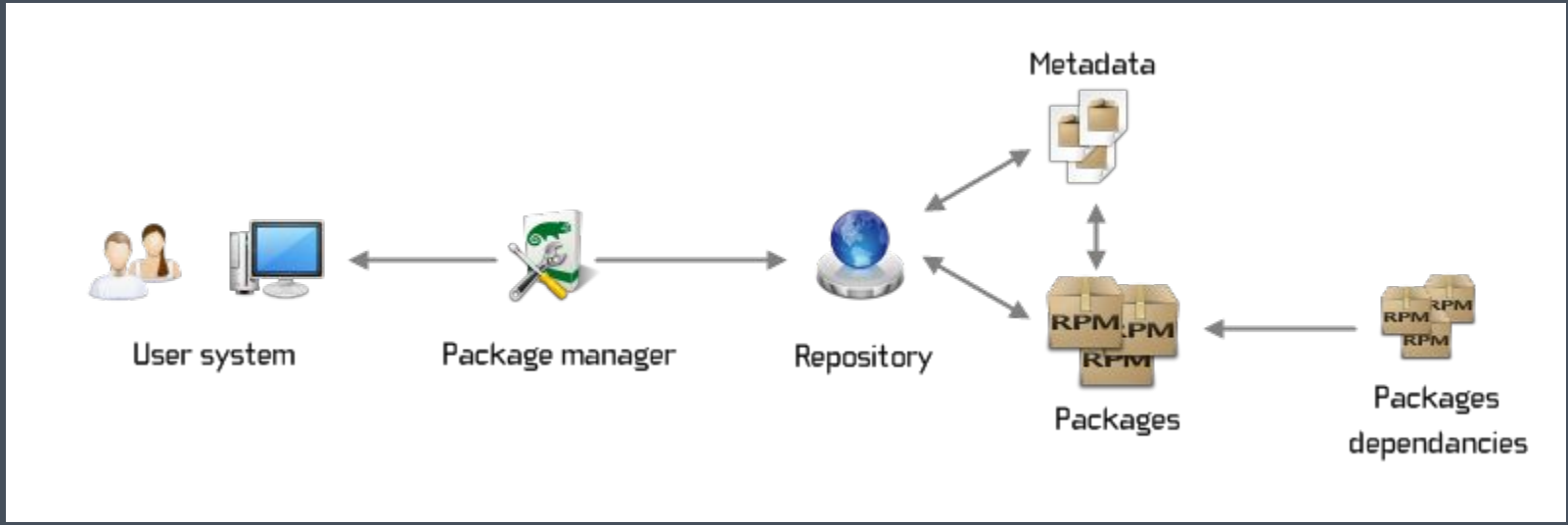
- Arch
  - Bragging rights
  - Newer, rolling-release software packages
  - Extensive documentation
- NixOS
  - Even more bragging rights
  - Easier to generate configurations – it's stored within one file

Installing software: package managers

What is a package?

- An archive containing binaries and libraries of an application
- Also includes some other metadata for the system about the application
- Used to install new applications onto a system
- Debian uses the `.deb` format

What's a package manager?



Why?

- Packages reviewed for malware
- No need to manually configure options
- Fast, easy way to install/update for user
- Updates and security patches

Package Manager != App Store

Packages:

- Maintainers vet updates
- Packages can depend on other packages (libraries)
- Users trust distribution to not package sneaky software

Apps

- Developers push updates directly to users
- Apps can only depend on underlying OS (sandboxing)
- Users hope developers won't be sneaky

Other Packaging Methods

Other forms of software distribution (that we won't discuss today):

- Webapps and the web platform
- Snaps/Flatpaks (kind of like "app stores" for desktop Linux)
- Helm: packages for Kubernetes
- Arch Linux (pacman, AUR, community repos, etc)
- Homebrew (cross-platform package manager for mac and linux)

Wait how do I use one then?

```
$ apt update
```

- Grabs a new list (catalogue) of what packages are available.

```
$ apt install <packagename>
```

- Installs the package.

```
$ apt remove <packagename>
```

- Uninstalls the package.

Wait how do I use one then?

```
$ apt dist-upgrade
```

Updates the packages and resolves package conflicts/removals.

```
$ apt search <packagename>
```

Searches for the package in the catalogue

```
$ apt install ./package.deb
```

Install local package (dangerous!)

A Demonstration of Package Installation

Install random package (cowsay).

What just happened

`apt install`:

a. Reads from the package lists
b. Finds out what **dependencies** the package needs
c. Checks what packages are already installed
d. **Download** the packages (if not installing a local .deb)
e. **Verify** integrity of packages
f. **Unpacks** them and copies the files over
g. Processes any remaining **triggers**
    i. Triggers are events such as scripts that run post-install
    An example is starting the application as a service

How does Debian know which package to use?

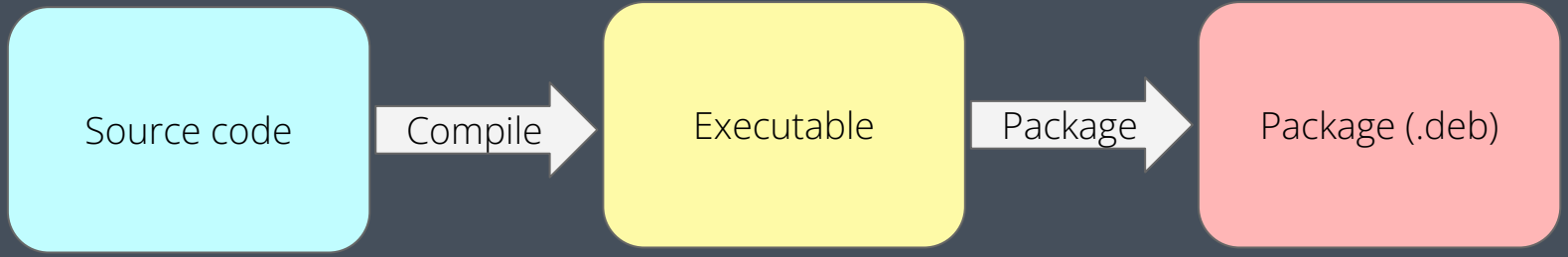
/etc/apt/sources.list and /etc/apt/sources.list.d/*

```
deb http://mirrors/debian/ stretch-backports main contrib non-free

deb http://mirrors/debian-security/ stretch/updates main contrib non-free

deb-src http://mirrors/debian-security/ stretch/updates main contrib non-free

deb http://mirrors/debian/ stretch-updates main contrib non-free

deb-src http://mirrors/debian/ stretch-updates main contrib non-free

deb http://mirrors/debian/ stretch main contrib non-free

deb-src http://mirrors/debian/ stretch main contrib non-free

# OCF

deb http://apt/ stretch-backports main

deb-src http://apt/ stretch-backports main

deb http://apt/ stretch main

deb-src http://apt/ stretch main

deb http://mirrors/puppetlabs/apt/ stretch puppet
```

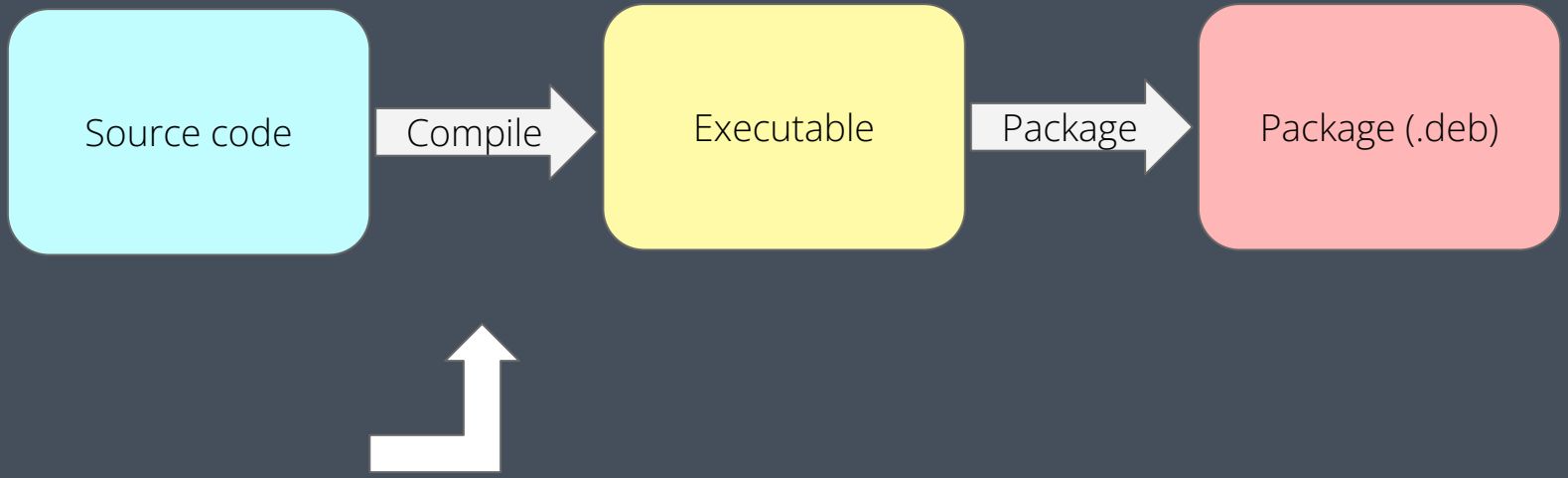
ocf note: unqualified URLs get .ocf.berkeley.edu appended
so http://mirrors/ is really http://mirrors.ocf.berkeley.edu/

Creating Packages

Package Creation 101



Package Creation 101



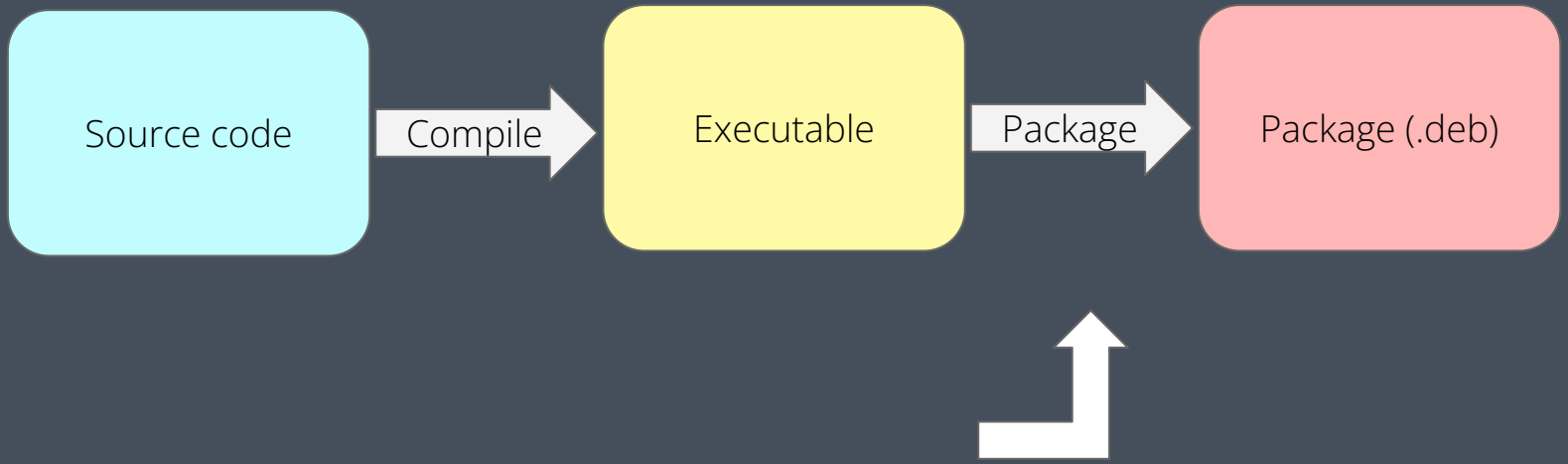
What is Compilation?

- Turns source code into a real executable.
- Turns ingredients into a dish.
- `$ gcc hellopenguin.c -o hellopenguin`
- Becomes complicated for large projects.

Manually Compiling Software

- Usually packages downloaded from the web have a **Makefile**.
- Basically they make the application for you as long you have the right things installed (like compilers like GCC).
- First `$ ./configure`
- Just run `make` and then `make install`.

Package Creation 101



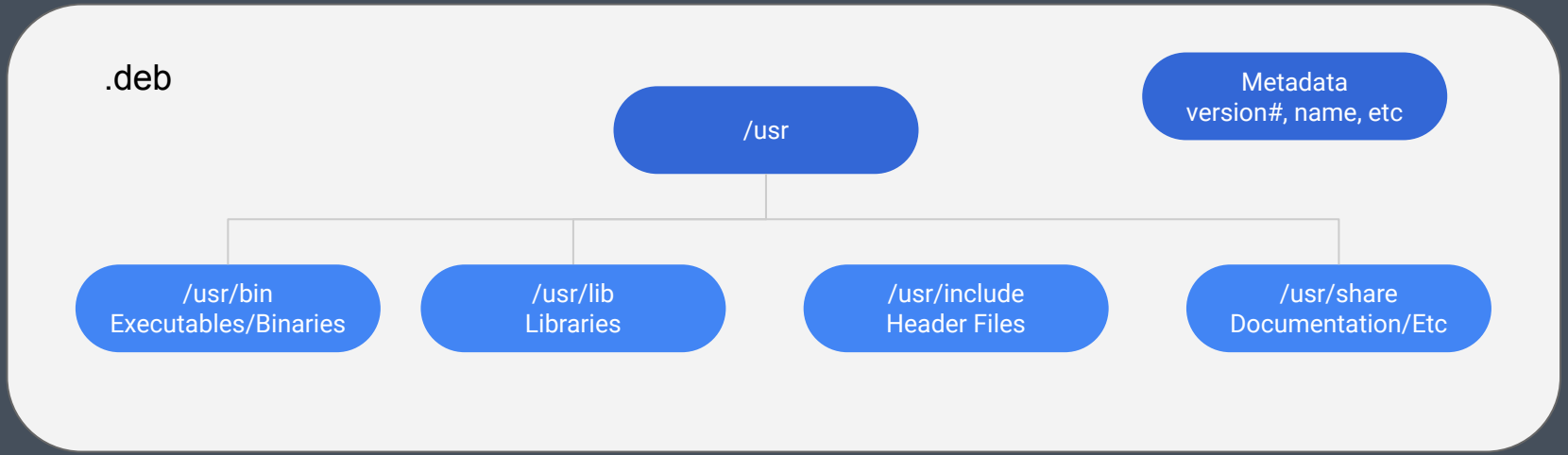
Ok now, how do I make a package?

- Huge pain in the butt
  - Download software/write it
  - Make sure everything configured
  - List dependencies, metadata (version, author, etc.)
  - Make sure linked libraries are in place
  - Use tool to make package

Packaging

- In the lab we're going to use Effing Package Management (FPM), which is a Ruby Gem and makes packaging (slightly) less painful
- `$ sudo apt install ruby-dev`
- `$ sudo gem install fpm`
- `$ fpm -s dir -t deb -n [name here] -v [version #] -C [the directory with the /usr folder]`

Packaging Basics - Package layout



Anatomy of a Debian package

Demo: Spelunking in a .deb

Spelunking in a .deb

- Downloading a .deb w/ `apt download` `wget`
- Breaking it open with `ar`
- Examining the control file
- Examining the data file
- Being lazy and using `apt show`

Anatomy of a Debian Package

- How can i view a package deb?
- .deb files are just ar archives
- apt download cowsay && ar x cowsay*_amd64.deb

Anatomy of a Debian Package

- control: Package metadata
- Size of package
- Package version
  - For package updates
- Dependencies of this package

Anatomy of a Debian Package

- /usr/bin/: Executable(s) the package provides
- Added to your $PATH
- /usr/share:
  - Documentation
  - Manpages
  - locales
- /etc: global configuration files
- md5sums: verify integrity of all files

Resources

- Reading List (accessible under decal.ocf.io/resources)

FPM documentation

- https://github.com/jordansissel/fpm/wiki

Debian documentation

- https://www.debian.org/doc/