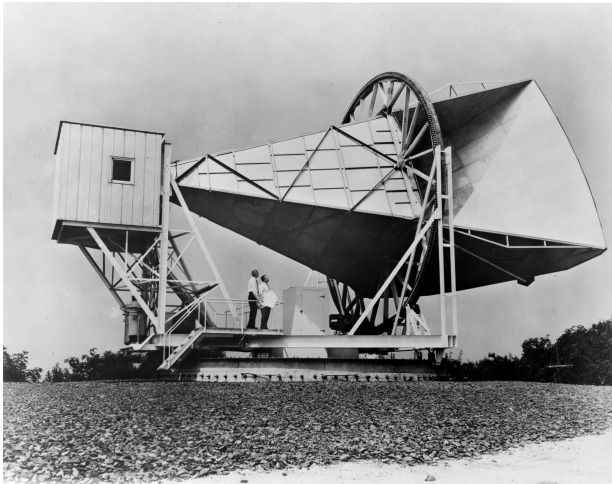


Why We Use Linux

(or macOS or *BSD or ...)

In the beginning ...

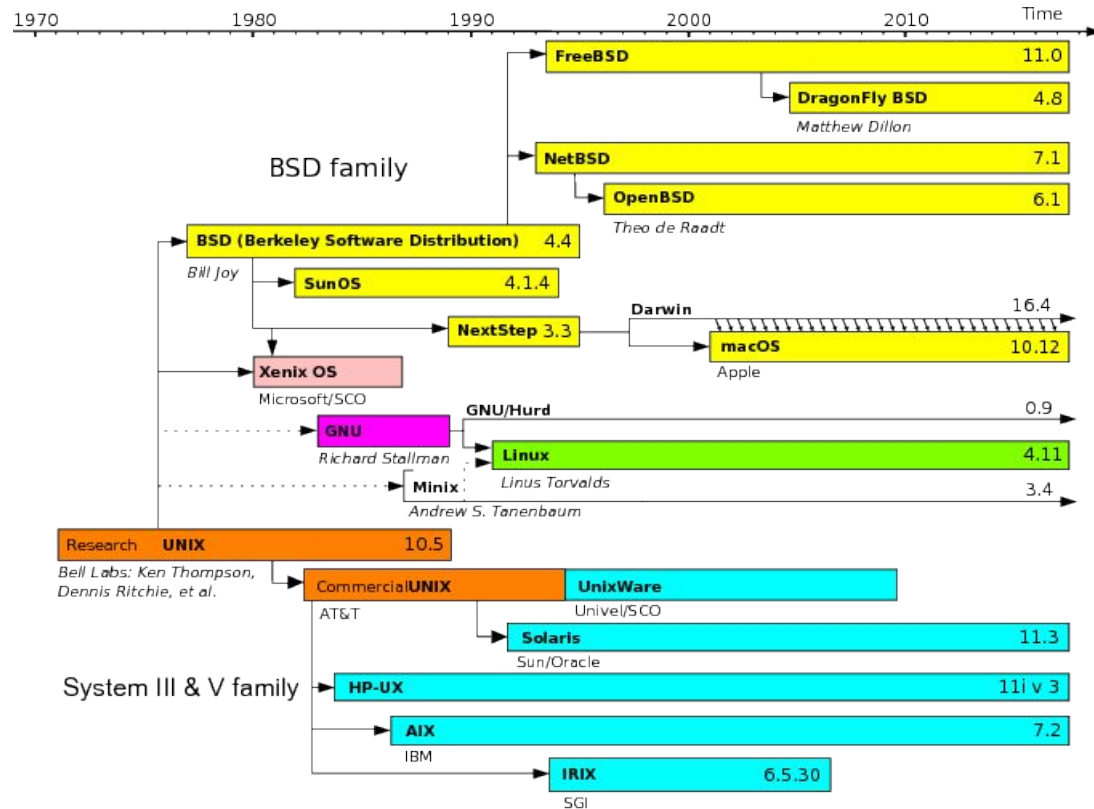
There was Bell Labs. All sorts of great things were happening, from detecting the CMB in the mid 1960s, to writing UNIX.



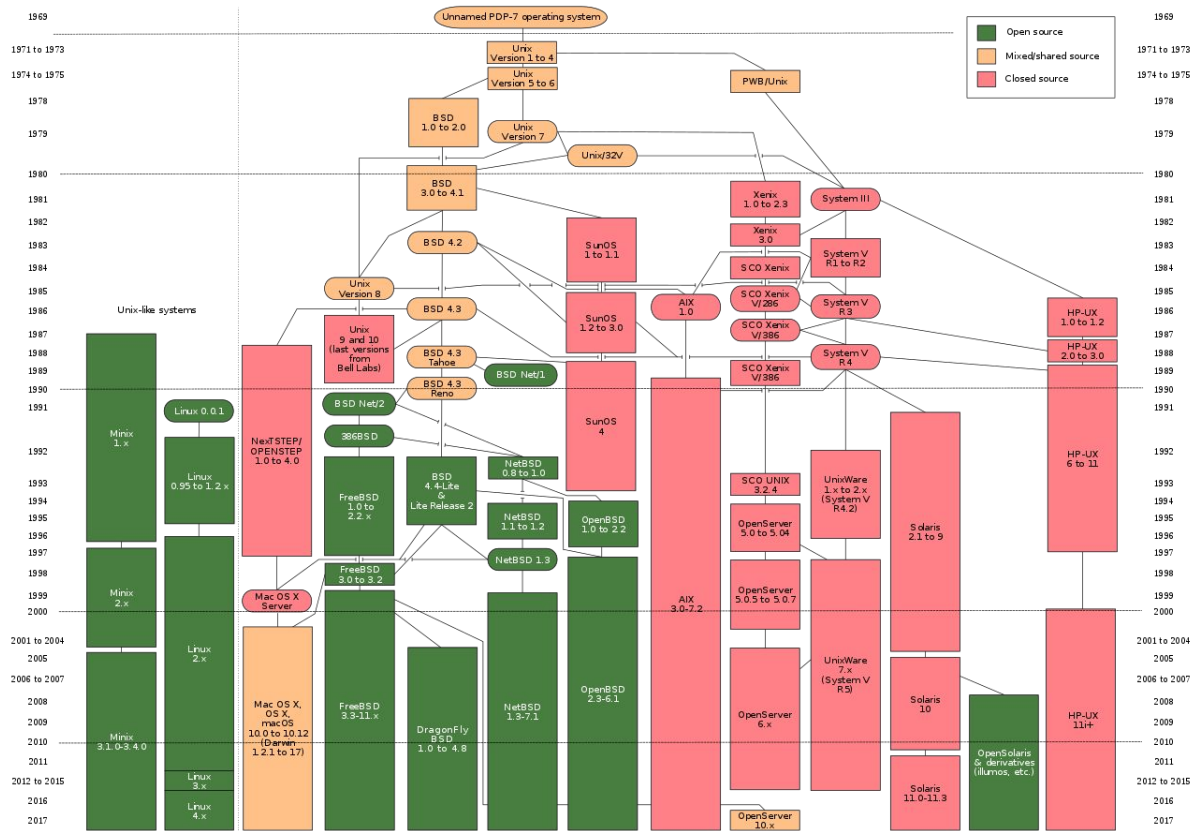
Ken Thompson and Dennis Ritchie wanted a multi-user operating system for their PDP-7 minicomputer, and so they wrote one. Along the way, Ritchie would also help develop the C programming language.

Bell Labs commercialized UNIX, and made it available to universities for a low cost. Consequently, a lot of people did computer research on UNIX systems. (Microsoft did this in the 1990s too)

Family Trees



Family Trees



Why is Nate talking to us about this?

I've been working with various flavors of Linux / UNIX / etc. for the past 19 years (yikes!)



My first laptop (Dual
Boot Win95 / Red Hat
Linux 5.2) 1998

Installing operating
systems on these
machines was
surprisingly difficult
and poorly
documented!



My first SPARC
workstation (SunOS 4.1 /
Red Hat 5.2 for Sparc)
1999

A Quote

"Within a month of his arrival, Randy solved some trivial computer problems for one of the other grad students. A week later, the chairman of the astronomy department called him over and said, "So, you're the UNIX guru."

"At the time, Randy was still stupid enough to be flattered by this attention, when he should have recognized them as bone-chilling words. Three years later, he left the Astronomy Department without a degree, and with nothing to show for his labors except six hundred dollars in his bank account and a staggeringly comprehensive knowledge of UNIX."

– Neal Stephenson, *Cryptonomicon* (p. 78). HarperCollins.

The quote applies to me!

That is to say, I was a physics undergrad who fell into computers. It has been a good ride. But two things to note:

- 1) I was never a CS person. If you try to out-nerd me you will probably win. I thoroughly enjoy nerd jamming, but as a pleasant pastime, not as a game of one-upmanship.
- 2) My knowledge in this area is like a reflecting pool. Broad, but shallow! I am here to help you solve problems.

What's motivating this talk?

- Your goal in life is **not** to be a SysAdmin
- You just want to get some science done!
- When you access remote systems at universities and observatories, it will involve Linux in some form.
- When you have a problem, **how do you ask the right questions to solve it?**



≠



What's motivating this talk?

- **SysAdmins can be difficult**
 - They forget that once upon a time, they were new.
- **There is a lot of bad advice on the internet!**
 - Asking the right questions helps you get the right answers.

Mainframes to PCs and Back Again

One CPU / Many users to One CPU / One User to Many CPUs / Many Users



PDP-7



IBM PC



Blue Waters

What options are there to try Linux!

Hardest: Install Linux on your own hardware

Moderately Hard: Install Virtualization software, install Linux on a Virtual Drive

Moderately Hard: Install Docker (needs virtualization enabled, can be thorny on old hardware)

Less Hard: Make a LiveCD (wait, people still have CD-ROM drives?) or LiveUSB, boot your computer from USB

First things first: What sort of Linux is this?

It is useful to know which “family” of linux you’re working with.
If it’s your laptop you probably already know this.

Most likely something in the **RedHat** or **Debian** Families

```
-bash-4.1# lsb_release -r -d -c
Description:    CentOS release 6.8
(Final)
Release:       6.8
Codename:      Final
```

```
ned@xps13:~$ lsb_release -r -d -c
Description:    Ubuntu 16.04.4 LTS
Release:       16.04
Codename:      xenial
```

First things first: What sort of Linux is this?

The distribution will give you a general idea of what software is installed, what software can be installed.

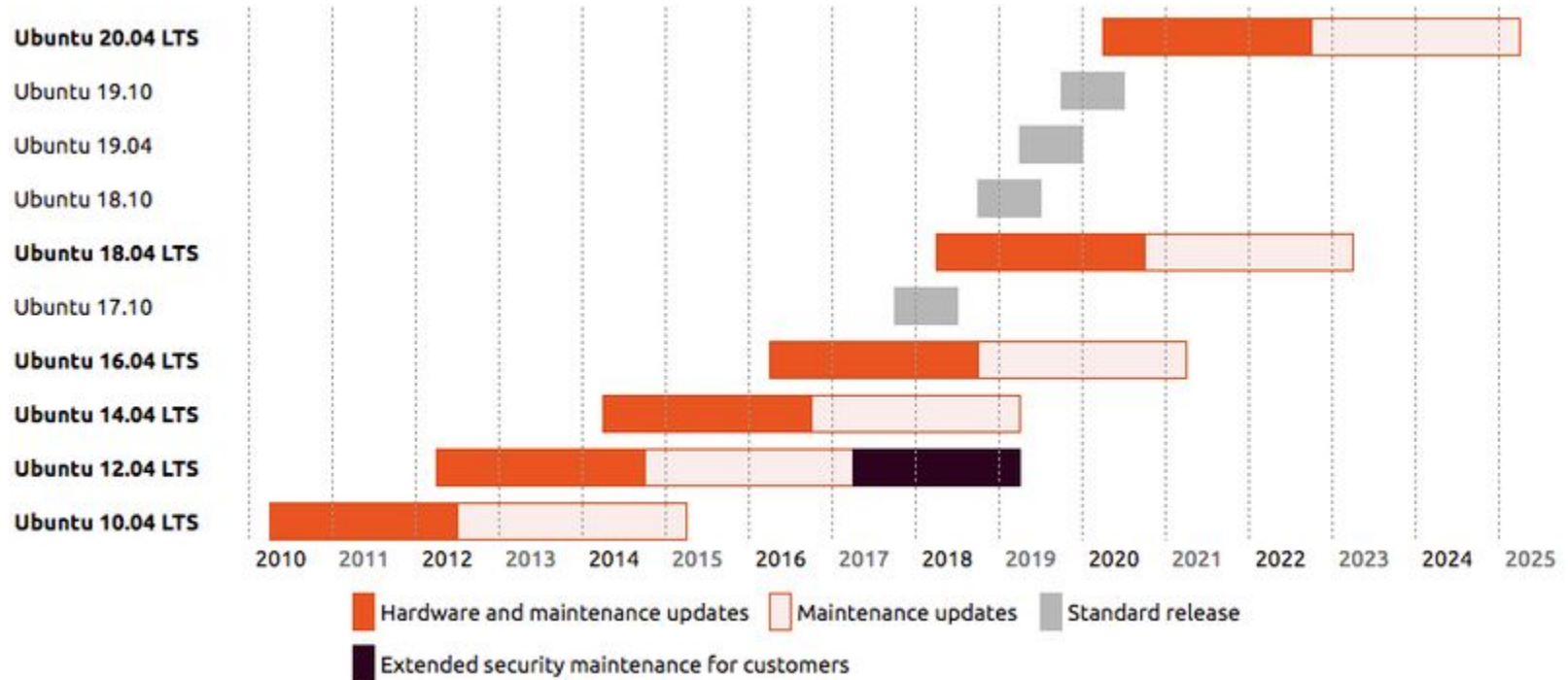
If it's on a server, *especially Red Hat* things move very slowly. Enterprise level distributions are cautious, and very conservative. By default you will likely be stuck with older versions of everything, from python to compilers to etc.

If it's on your personal machine, it most likely is running more closely to modern development, unless of course you don't update often.

First things first: What sort of Linux is this?

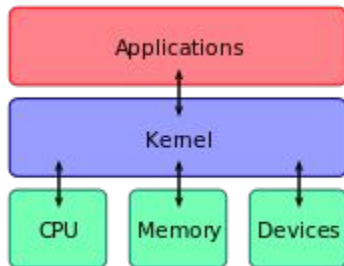
CentOS version	Release date	Full updates ^{[122][123]}	Maintenance updates ^{[122][123]}
3	19 March 2004	20 July 2006	31 October 2010
4	9 March 2005	31 March 2009	29 February 2012 ^[124]
5	12 April 2007	31 January 2014	31 March 2017 ^[125]
6	10 July 2011	10 May 2017	30 November 2020
7	7 July 2014	Q4 2020	30 June 2024
<div><div></div> Old version</div> <div><div></div> Older version, still supported</div> <div><div></div> Latest version</div>			

First things first: What sort of Linux is this?



Now that you know, what to do?

Linux is really just the kernel, the part of the operating system that acts as a bridge between the programs that you write and run and the hardware. To find out what version of the kernel you are running you can use `uname`



```
$ uname -a
```

```
Linux xps13 4.4.0-127-generic #153-Ubuntu SMP Sat May 19 10:58:46 UTC 2018  
x86_64 x86_64 x86_64 GNU/Linux
```

Why use Docker? What is it?

Docker is a container system. It shares kernel resources with other containers, on the system. It allows a lightweight layer to easily try out different software.

- Can be used on servers and on workstations / laptops
- Many software groups are using it for development work
- It is possible to tag docker builds with specific software versions, which allow you to troubleshoot problems.

Unfortunately, it assumes that you are running on a very modern system.

What sort of problem are you having?

Example: A Compiling Problem

You are frequently told “Installing software is as easy as `./configure && make && make install` but what if things go wrong?

1. Which compiler are you using? Something from the Gnu compiler family? (gcc/g++/gfortran), something in the LLVM family (Most likely on OSX)? One of the fancy commercial compilers? lcc (intel) , pgcc (Portland Group) ?
2. Is this fortran? Does it expect F77 compatibility?
3. Where does the software that this program needs live? Do you have a specific place that you like to install things or does it all live in /usr/local ?
4. Does the software that you are trying to compile also build a python module?

What sort of problem are you having?

Compiler Versions

Different compilers expect different standard options. Most free software assumes that you are going to be using the gnu compiler suite. Check the Makefile to see what what is set.

Common Variables in the file to check:

- CC, CXX, FC, CFLAGS, CXXFLAGS, FFLAGS (compilers and compiler options)
- CPPFLAGS, LDFLAGS (probably the most important. Paths to headers and libs)

What sort of problem are you having?

Where is the software

When you run `./configure` it goes through and checks to see if the software required to build the package exists. If there are errors, inspect “`configure.log`”. This will tell you what options were passed to the configure script and what sort of things might have failed.

Usually, you'll need to tell it the path to shared libraries, and then make sure that those paths are in `LD_LIBRARY_PATH`

You can check to see if a shared / dynamic library is missing with `ldd`

Some things to try with docker

`docker images` (shows images that you've downloaded)

`docker ps -a` (shows all containers)

`docker stop ipta` (stop your container)

`docker rm ipta` (remove your container)

`docker run -v /path/to/local/dir:/home/jovyan/work/data`

Get started with the exercises

```
docker exec -it -u jovyan ipita /bin/bash
```

This connects to the docker container as the jovyan user. Useful if you don't need to forward graphics (and perfectly acceptable for doing the exercise)

Go to <https://github.com/ipita/ipita-2018-workshop/tree/master/linux>

And click on “ipita-2018-workshop-linux.md”

You can also clone the repository for the python exercise by doing

```
git clone https://github.com/ipita/ipita-2018-workshop
```