GNU/Linux Architecture

Benjamin Brewster & Erich Kramer

Please boot up the "CentOS_Reference" VM in VirtualBox now

Why You Need to Care

- Because someday you'll have to:
  - Find a particular system file in Linux
  - Install Linux programs without fear
  - Manage packages like a boss
  - Understand the difference between /bin and /usr
  - Know how to pronounce GNU (guh-NEW)

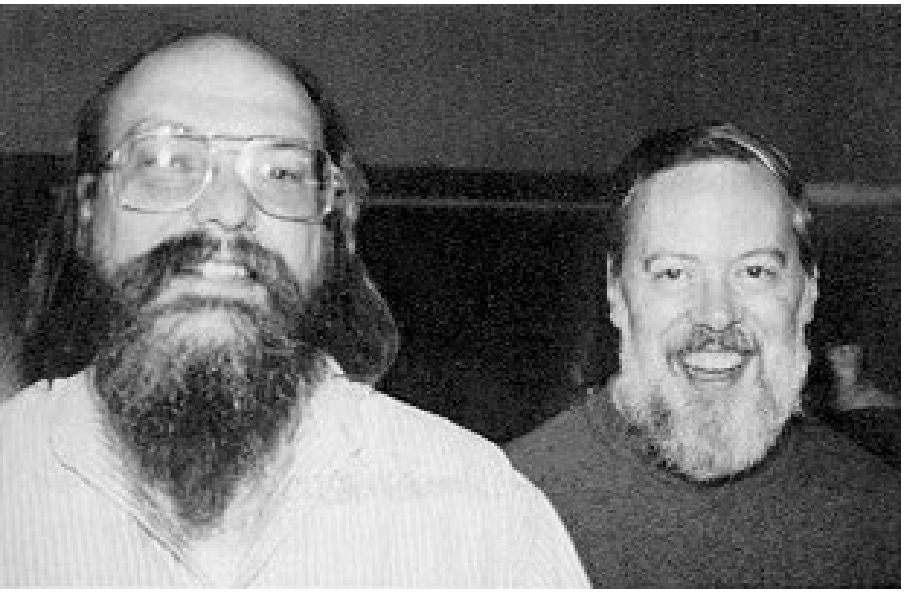
Linux Overview

- Let's say that an Operating System consists of two parts:
  - Programs that talk to other programs
    - GNU is a UNIX-like OS consisting of applications, libraries, dev tools, etc.
  - Programs that talk to the hardware
    - Linux is a kernel that handles machine resource allocation and communication with the hardware
    - First thing loaded into memory, stays resident while system runs
- Together, this is GNU/Linux
- We mostly just call it Linux, though it makes the GNU people mad

Other than beards, GNU is Richard Stallman's pet project

Ken Thompson – Unrestrained, yet directed

Bonus Dennis! Does this guy know how to party!



Gandalf the White

Richard Stallman

http://book.roomofthings.com/2012/05/15/enchanted_objects_IoT_for_humans

Family Trees

- Most modern GNU/Linux Operating System Distributions are children of Debian, Redhat, and Slackware
- We'll be studying and using CentOS, a descendent of Redhat
  - CentOS is very stable, and is widely used for running servers
  - Most OSU servers use CentOS

- Making your own distribution is as easy as swapping out the GNU "ls" utility for your own, and then giving it a snazzy name
- Distro name ideas from students:
  - Dolfins, JustABunchOfIllegalEmulators, etc.

Important Programs in Linux

Say

- Before we get into the contents of the Linux file system, let's talk about important processes
- The vmlinuz.* files are the Linux executable, stored in compressed form
  - The VM stands for Virtual Memory, as in, Linux supports it
  - The "z" means that it's compressed with gzip; it's not just compressed, though: the gzip decompressor is built into the front of the binary so that it can be decompressed by the bootloader!
  - Why compressed? Because it's slow to copy from external storage, but fast to work in RAM: make the copy be small
  - This is made from a complied kernel called "vmlinux", which is the uncompressed, unbootable image (the bootloader expects the compressed form)
- All the versions of the Linux kernel will be listed in this directory
- The kernel is loaded by the bootloader
- The kernel's first job is to start up systemd

Demonstrate

- `$ cd /boot`
- `$ ls -pla`

Important Programs and Services in Linux

Say	Demonstrate
• PID 0 is the scheduler and memory pages; part of the kernel (not a user-mode process)	• `$ ps -elf | sort -nr -k 4`
• PID 1 is systemd (or init), which starts up Linux, once it's been loaded by the bootloader, and then runs the system	•
• PID 2 is (usually) kthreadd, which spawns kernel threads as needed	•
• See how many other daemons are started up by PID 2?	•
• syslogd, in charge of logging everything Linux does	• `$ ps -elf | grep syslogd`
• crond, in charge of running scheduled tasks	• `$ ps -elf | grep crond`
• cupsd, in charge of printer	• `$ ps -elf | grep cupsd`
• Lots of network ones like httpd, ftpd, sshd, dhcpd, etc.	

Packages and Managers

Say

- While software can be compiled on any OS, and the files copied manually wherever you choose, modern Linux distros use the concept of Packages and Managers
- A Package is usually a collection of files and metadata related to a particular application, like apache or Docker
- The metadata tracks the particular version of the software, among other things
- Package Managers know how to:
  - Download and install new packages
  - Follow instructions in the packages to install the executables, and save instructions on later how to uninstall them
  - Update packages to a newer (or target) version

Demonstrate

- `$ yum list installed`
- `$ yum list installed | grep "grep"`

Packages and Managers Around the World

Say	Demonstrate
• Debian-based distros, like Debian, Ubuntu, etc., use packages stored as "dpkg" files	
• The dpkg manager is **apt**	• `$ apt`
• Arch-based distros, like Arch, Manjaro, etc., use packages stored as "libalpm" files	
• The libalpm manager is **pacman**	• `$ pacman`
• RedHat-based distros, like RedHat, Fedora, CentOS, etc., use packages stored as "rpm" files	
• The rpm manager is **yum**	• `$ yum`
• Valid and useful commands are install, update, and erase, which we'll talk about	

Finding Packages

Say

- Packages are stored in online repositories, but not all repositories have the same packages
- Sometimes, you won't know what a package is called, and you aren't looking in the right repo
- Use an online page to find stubborn ones
- Typically we'd install a repo with this command
- But EPEL has you install a package that includes the repo set up, so do it like this:

Demonstrate

- `$ sudo yum install cowsay`
- Show https://pkgs.org, search for cowsay in CentOS 7
- This says that the pkg is in EPEL x86_64
- `$ sudo yum-config-manager --add-repo ???`
- `$ sudo yum install epel-release`

Installing, Updating, and Removing

Say

- Now, we can install our program
- Do updates for cowsay
- See if there are any updates for anything
- Remove cowsay - 'tis a silly thing
- Put it back in, see how fast it installs since the dependencies are still there!

Demonstrate

- `$ sudo yum -y install cowsay`
  - This installs a lot of dependencies!
- `$ cowsay hi`
- `$ echo "HELLO WORLD" | cowsay`
- `$ sudo yum update cowsay`
- `$ yum check-update`
- `$ sudo yum erase cowsay`
- `$ sudo yum install cowsay`

What Dependencies?

Say

- Install some tools that will let us play with yum

- What are all those dependencies for cowsay?!
- That's a ridiculous amount of dependency

Demonstrate

- `$ sudo yup -y install yum-utils`

- `$ repoquery --tree-requires cowsay`

Something a Little More Useful: Lynx

Say	Demonstrate
• Let's install a text-based web browser. That's right.	• `$ sudo yum install lynx`
• You'll need to reject a ton of cookies unless you push V for "neVer"	• `$ lynx www.google.com`
• Up and Down move the cursor between links, while Left and Right move forward and back between pages in your history	• `elephant`

The UNIX File Hierarchy

- IN THE BEGINNING, there was only /
- In the past, no actual standard for file locations
- In 1994, the File System Standard (FSSTND), adapted into the Filesystem Hierarchy Standard (FHS), was adopted by the majority of distros

- Files have a few interesting characteristics and intentions:
  - Shareable or not: should these files be accessible to other hosts?
  - Should the files be read-only or are the modifiable?
  - Are the files physically stored on the disk, or are they virtual?

/usr

- As an example, consider `/usr` :: UNIX Systems Resource
- Contains binaries, docs, libraries, header files, and programs installed by users
- Has by far the largest share of data on the system
- These are intended to be **shareable**: people should see and use this data, but it should **not be modified**

`/usr/bin`	General purpose binaries
`/usr/local`	Locally (user) compiled binaries
`/usr/src`	Linux kernel source code

/bin

- Let's talk about binaries…

`/bin`	**Bin**aries that are essential and must be available `cd, kill, ping, mount, passwd, systemctl, vi`
`/sbin`	**S**ystem binaries essential for booting, restoring, etc. *On CentOS, simply a symbolic link to /usr/sbin*
`/usr/bin`	Contains most of the binaries for a system, very diverse `yum, chown, chmod, curl, dd, grep, ls, make`
`/usr/sbin`	Typically programs to be ran by root, sysadmin purposes `chroot, fdisk, fsck, parted, shutdown, useradd`
`/usr/local/bin`	Self-compiled binaries, typically
`/usr/local/sbin`	Self-compiled system binaries, typically

/proc

- This is a virtual file system! These files are created and destroyed when the system boots and shuts down
- Each process has a set of virtual files stored here, like stdin, stdout, stderr
- Lots of the data is hard to interpret, so other utilities do it for us:
  - `top`, `ps`, etc.

`/proc/sys`	Contains a lot of kernel settings you can examine
`/proc/1/stat`	Contains the state of process 1, which is systemd

/etc

- A critical location to know about: it's where most system config files are stored

`/etc/X11`	Where graphics settings are kept
`/etc/shells`	Allowed shells
`/etc/os-release`	What OS is installed
`/etc/fstab`	File system mapping table: where, what disks are
`/etc/sudoers`	Who can use sudo
`/etc/hosts`	Maps hostnames to IP addresses
`/etc/yum.repos.d/`	Stores all configured repos for packages

/dev

- Where physical devices are connected: hard drives, optical drives, GPUs, removable flash drives, mice, keyboards, and a few special ones:

`/dev/sda b…`	Hard drives
`/dev/random`	Generates random numbers
`/dev/zero`	Your source for all your null terminator needs
`/dev/null`	Throws away everything sent to it
`/dev/tty`	Your current terminal - try echo to it! `$ echo "YO" > /dev/tty`

/boot

- Where everything needed to boot your system lives

`/boot/vmlinuz*`	Actual Linux kernel binaries
`/boot/initramfs`	Default RAM-based file system for early boot
`/boot/efi`	Holds EFI variables for UEFI systems

Summary: Important Folders and Files in Linux

`/`	The beginning of everything, all files exist underneath root
`/boot`	Everything required for boot process; things needed *before* user-mode; may contain Linux exe itself
`/bin`	Essential binaries for configuring and using the OS
`/dev`	Device files; everything is a file: HDD partitions, mouse connections, etc.
`/etc`	Contains many critical config files; contains /etc/X11 and xorg configs
`/home`	Home file storage for users; contains user level configs
`/root`	Home file storage for root user
`/lib`	Application libraries and kernel modules
`/media`	Removable flash drives are usually mounted here, not cat pictures
`/opt`	Software and add-on packages provided by a third party
`/proc`	Process virtual filesystem
`/sbin`	System binaries for booting, restoring, and repairing the system; super admin stuff
`/usr`	Unix systems resource: installed programs, source code, etc.
`/var`	Variable data like logging, email, temp files, etc.

Conclusion

- Linux has a defined hierarchy of files
- Programs are easily installable and removable as separate packages, maintained by a package manager
- A key feature of package managers is to track the dependencies of a package: what it needs to be installed before it will function
- The Linux kernel is an executable itself!