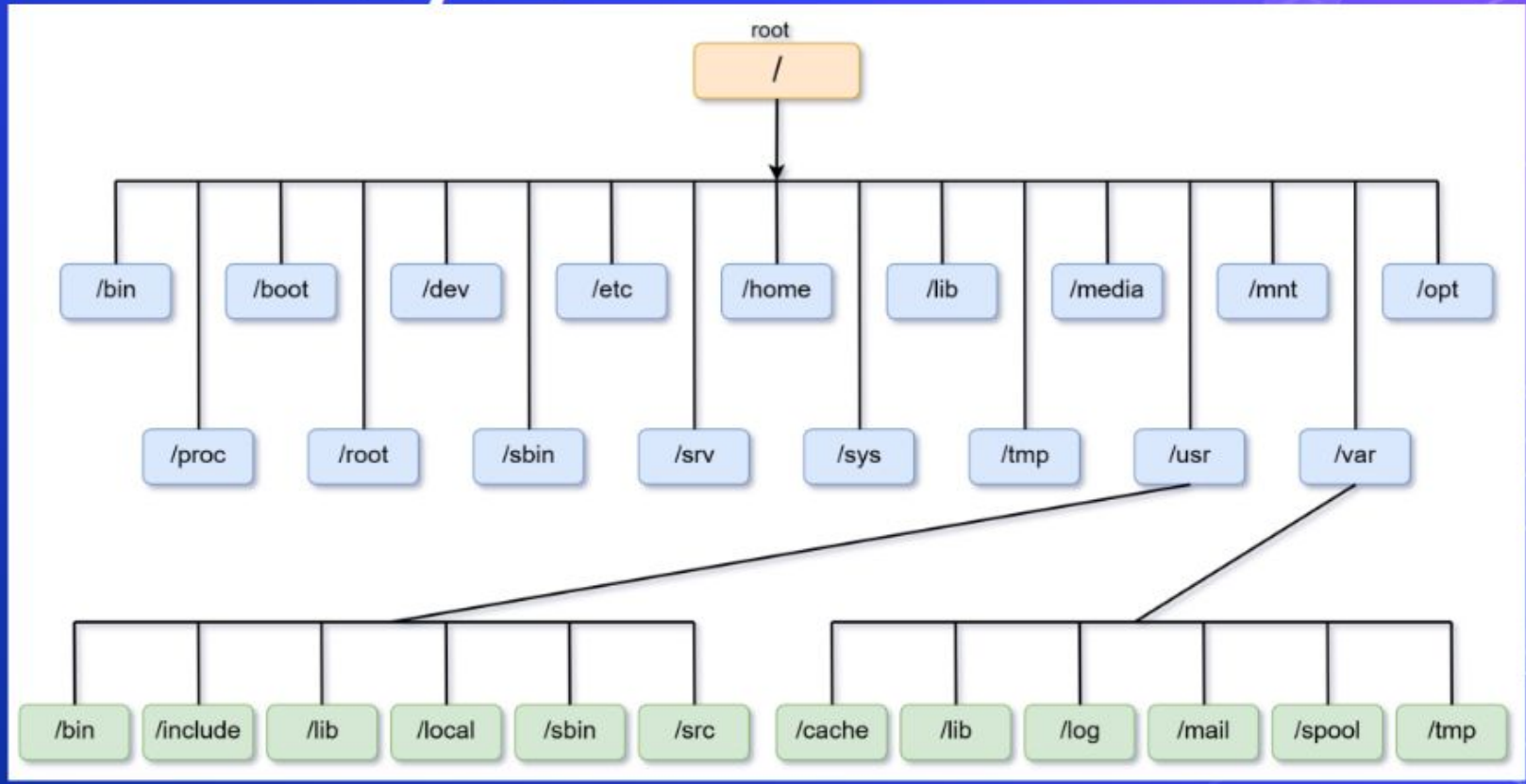
Linux Day 1

System Settings

What is Linux?

- Linux: an operating system (like Windows, Mac OS, etc.)
  - but not really
  - is just the kernel for the family of operating system distros
    - kernel: manages CPU, memory, other stuff
- open-source, options/"flavors" to suit different skill levels
- terminal: contains the "shell"
- operating system based on the idea that everything is a file

Linux Filesystem



Directories and Files

- almost everything in Linux is some sort of file
- / - root directory (contains every subdirectory and file under it)

Subdirectory Examples:

/home
directories for each individual user

/etc
contains system configurations

/var
variable; stores files and logs made during system

/bin
common executable programs in system

/boot
contains kernel and startup files

/lib
contains shared libraries and files

/mnt
mount; used to temporarily mount devices & filesystem

/proc
virtual filesystem w/ kernel and process files

```
serveruser@ubuntu: /
serveruser@ubuntu:~$ cd /
serveruser@ubuntu:/$ ls
bin    Documents   initrd.img.old  media  root  srv  var
boot   etc         lib            mnt    run   sys  vmlinuz
cdrom  home        lib64          opt    sbin  tmp  vmlinuz.old
dev    initrd.img  lost+found     proc   snap  usr
serveruser@ubuntu:/$
```

Basic Terminal Navigation

- **pwd**- print working directory (prints wherever you are located)
- **cd**- change directory (moves you to different directory)
  - cd [ABSOLUTE PATH]; cd [RELATIVE PATH]
  - **cd ..** (brings you back one directory)
  - **cd .** (brings you to the current one)
  - **cd -** (brings you to the previously used directory)
- **ls**- list (lists all files in current working directory)
  - **ls -l** (lists files as well as details in a long-listing format)
  - **ls -a** (lists ALL files in the directory)
  - **ls -la** (combines previous effects)
- **sudo**- “super user do” (gives root access for commands)

File & Directory Manipulation

- touch [file_name]
  - creates a file
- cp [file_name] [filename_of_copy or destination_directory]
  - copies a file under a different name if you put the filename of the copy
  - makes a copy in a different directory if you write the destination directory
- mv [source_of_file] [destination_of_file]
  - moves a file to the specified location
  - also can be used to rename file

File Editing & Viewing

- **nano** [file_name]
  - edits file in the command terminal
- **gedit** [file_name]
  - edits file in default text editor (graphical)
- **cat** [file_name]
  - prints out file contents in the terminal
- **less** [file_name]
  - displays content by pages; scroll through file

File & Directory Manipulation

- rm [file_name]
  - removes the file
- **mkdir** [directory_name]
  - creates directory
- **rmdir** [directory_name]
  - removes directory

Users

- **adduser** [user_name]
  - adds the user into the system w/ standard permissions
- **deluser** [user_name]
  - deletes user from the system
- **su** [user_name]
  - switch user to the user given (will prompt for password)
  - **sudo su**: switches to a **root** user
- **passwd** [user_name]
  - changes password of user
  - **passwd -l root** → locks root account
- **chpasswd**
  - changes multiple passwords
  - user1:user1_password (this format)

Groups

- **groupadd** [new_group]
  - creates a new group
- **gpasswd -a** [user_name] [group]
  - adds a user into a group
- **gpasswd -d** [user_name] [group]
  - deletes a user from the group
- **gpasswd -A** [users,users2]
  - Set administrator list
- **gpasswd** [group]
  - Prompts for new password when called by group administrator

Config File /etc/passwd

- sudo nano /etc/passwd: brings you to

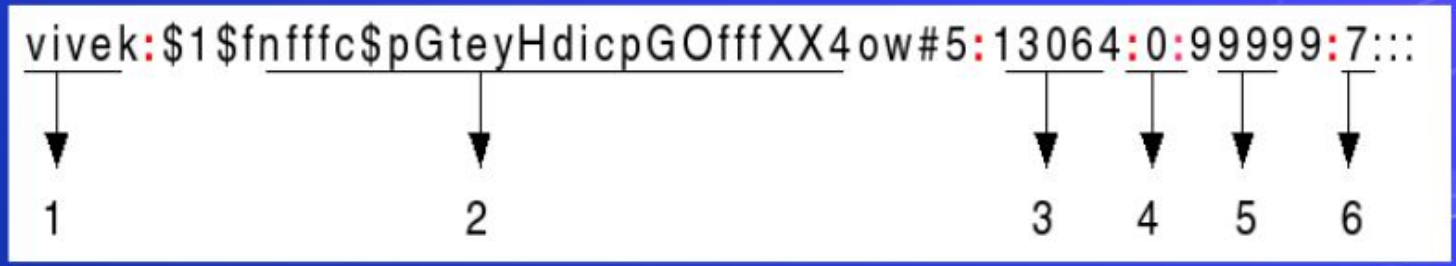
1. name of user
2. password (x.....?)
3. User ID (UID)
4. Group ID
5. Comments/GECOS (extra info)
6. Home directory
7. Shell directory

What to check for in /etc/passwd

- Only users have /home directories (/home/user_name)
- Only users have the shell /bin/bash. System users will have /bin/false or /usr/sbin/nologin.
- any user with a user ID of less than 1000 is most likely either a hidden user or a system user
  - hidden users are not safe and should be deleted
- If a user has a ! where the password should be, that indicates they don't have one, which is a problem.

Config file /etc/shadow

- sudo nano /etc/shadow: configures shadow file
  - contains the hash of the password and other password and account information



1. name of user
2. encrypted password
3. Last password change
4. Minimum password age
5. Max password age
6. warning period
7. Inactivity period

chage: a command we can use to edit /etc/shadow

- The **chage** command changes shadow file parameters on a per user basis
  - Changes only affect a certain user.
  - Global changes belong in the /etc/login.defs file, which we'll go into next

How to use chage

- chage [options] [username]
  - chage -m 6 [user]
    - Changes min days of user to 6
  - chage -M 15 [user]
    - Changes max days of user to 15
  - chage -W 7 [user]
    - Warn days
  - chage -I 5 [user]
    - Inactive days

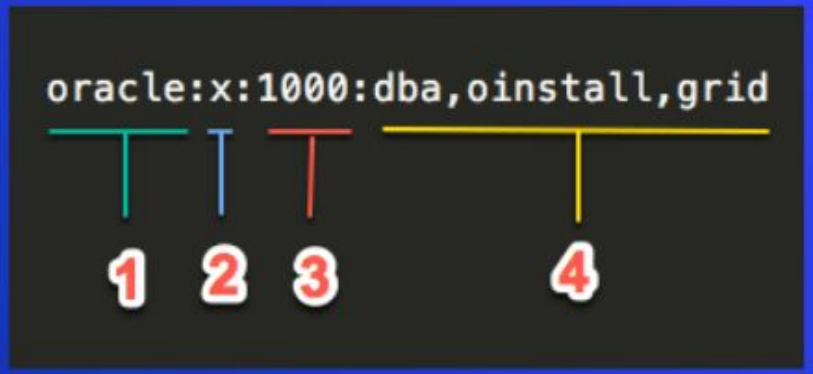
Config file /etc/login.defs

- **sudo nano /etc/login.defs**: brings you to login.defs file to edit information
  - configures settings for password aging as well as general user account stuff
  - PASS_MIN_DAYS (minimum # of days a password can be used) → **8 days**
  - PASS_MAX_DAYS (max # of days for certain password to be used) → **15 days**
  - PASS_WARN_AGE (# of days before password is expired, gives warning) → **7 days**

```
#
# Password aging controls:
#
#       PASS_MAX_DAYS   Maximum number of days a password m
#       PASS_MIN_DAYS   Minimum number of days allowed betw
#       PASS_WARN_AGE   Number of days warning given before
#
PASS_MAX_DAYS   99999
PASS_MIN_DAYS   0
PASS_WARN_AGE   7
```

Config File /etc/group

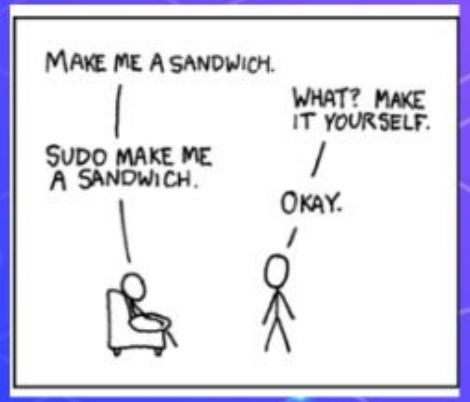
- **sudo nano /etc/group**: brings you to the group file to configure it
  - contains info about groups



1. group name
2. password
3. Group ID (GID)
4. Users in the group

Config file /etc/sudoers

- deals with users and capacity for them to use sudo
  - 3 types of users:
- **sudo visudo** (DO NOT USE NANO)
  - creates a duplicate file of /etc/sudoers(.tmp) to be edited
  - notifies you of syntax errors and logs edits

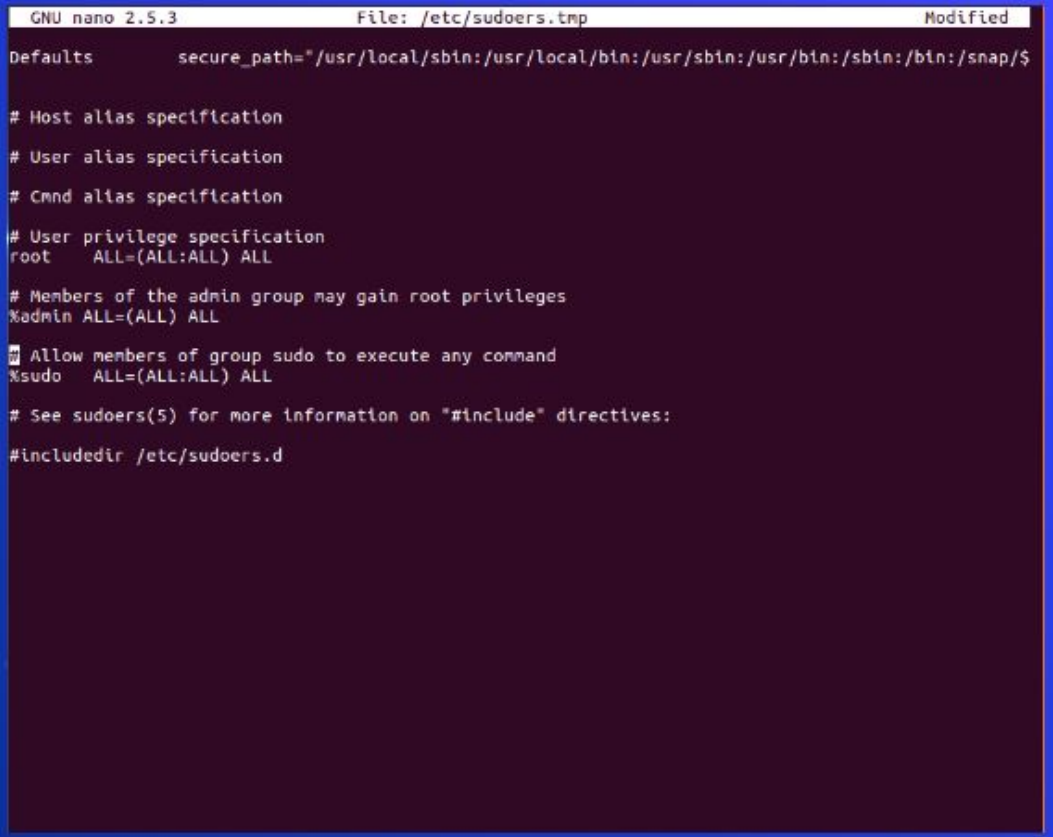



```
>>> /etc/sudoers: syntax error near line 31 <<<
What now?
Options are:
  (e)dit sudoers file again
  e(x)it without saving changes to sudoers file
  (Q)uit and save changes to sudoers file (DANGER!)

What now?
```

-BE CAREFUL when editing this file

Default Config for /etc/sudoers



- root ALL=(ALL:ALL) ALL
  - gives the root user unlimited privileges on the system and can use any command
- %admin ALL=(ALL) ALL
  - %: notates a group
  - admin group has root privileges
- make sure this is how the file looks like

Common problem in sudoers

- Does anyone know what and “!” means in coding?
  - Knowing that, what does “Default !authenticate” mean?
- If you ever see that line in /etc/sudoers or the /etc/sudoers.d directory, remove the !
  - Why would we do that?

Guest Account

- Ubuntu has a default guest account that people can log in to
- security purposes: must **disable** the guest account
- Ubu14: **/etc/lightdm/lightdm.conf** (configures the guest file)
- Ubu16: make a file called **50-no-guest-conf** in the LightDM config folder **/usr/share/lightdm/lightdm.conf.d/50-unity-greeter** for Ubuntu 14
- LightDM: display manager for Linux, disables guest account but also can be used for other user interface configs at login screen

Default File for /etc/lightdm/lightdm.conf

```
[Seat:*]
autologin-user=serveruser
allow-guest=true
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

- [Seat:*] or [SeatDefaults]
  - should be the first line
- autologin-user=serveruser
  - sets system to start the specified user to be logged in
  - autologin-guest=true → **BAD**
- allow-guest=true
  - enables guest account (default)
  - change to **false** to disable

nope not this one

PAM in Linux

- PAM (Pluggable Authentication Modules): large API that contains information about authentication and dictates the user's authentication to certain services
  - **account**: checks for account verification in relation to passwords or services
  - **authentication**: authenticates user and establishes credentials
  - **password**: updates user passwords, sets certain guidelines
  - **session**: manages actions taken at the beginning and end of sessions (after you log in)
- access all of them by going to the **/etc/pam.d** directory

```
serveruser@ubuntu:/etc/pam.d$ ls
chfn                common-session-noninteractive  login      runuser-l
chpasswd            cron                           newusers   su
chsh                cups                           other      sudo
common-account      gnome-screensaver              passwd     systemd-user
common-auth         lightdm                        polkit-1   unity
common-password     lightdm-autologin              ppp        vmtoolsd
common-session      lightdm-greeter                runuser
```

Format of PAM config files

```
module-type     control         module-path     arguments
```

- module-type: the different modules in PAM (the four mentioned in the last slide)
- control: determines how PAM will behave when a module fails
  - **required**: requests fails = denies authentication but runs through the stack
  - **requisite**: request fails = immediate denial of service/authentication
  - **optional**: request succeeds or fails = only significant if only one there
  - **sufficient**: request succeeds, nothing else checked, fail → continue checking
- module-path: module you are referencing (pam_<module>.so)
- arguments: certain options available from the selected module path to make authentication more secure
  - minlen=12, ocredit=-1, gecoscheck, etc.

Config for /etc/pam.d/common-auth

- auth [success=1 default=ignore] pam_unix.so nullok_secure
- auth optional pam_tally.so deny=5 unlock_time=900 onerr=fail audit even_deny_root_account silent

- auth required pam_tally2.so oneer=success audit silent deny=5 unlock_time=900

Config for /etc/pam.d/common-password

- install package **libpam-cracklib** (enforces password complexity)
- password requisite pam_cracklib.so retry=3 minlen=12 lcredit=-1 ucredit=-1 dcredit=-1 ocredit=-1 difok=4 reject_username minclass=3 maxrepeat=2 gecoscheck enforce_for_root
- password sufficient pam_unix.so use_authtok obscure rounds=80000 sha512 shadow remember=7
  - use_authtok→ takes the previous successful password from another module & apply here
  - rounds=80000 → rounds of encryption for password
  - sha512 → uses sha512 algorithm to encode the password
  - remember=7 → system remembers certain amount of previous passwords

Parameters In Detail

- pam_cracklib.so = references the cracklib module (compares w/ dictionary words)
- retry=3 → # of times user can retype password
- minlen=12 → minimum length
- lcredit=-1 → # of lowercase letters in password
- ucredit=-1 → # of uppercase letters in password
- dcredit=-1 → # of digits in password
- ocredit=-1 → # of characters in password
- difok=4 → # of character changes from old password

Package Management

- package: collection of files meant to run a certain task
- **apt-get** [command] [package]: retrieves packages from source to use
  - install [package] → installs and/or upgrades packages
  - autoremove [package] → removes extra dependencies & packages
  - remove [package] → removes only the **main** package
  - purge[package] → removes the packages and any config files along with it
    - add **--autoremove** to add another layer of removing
  - apt-get update → notifies the system of new versions of packages if available
  - apt-get upgrade → **actually** updates the system's packages if available
-

Package Management Pt.2

- dpkg -l: lists out all packages installed on the system
  - dpkg -l | grep [package] → searches the package list
  - packages to look for: john, medusa, hydra, netcat, etc.

Quick Review of Configuring the Software Update Center

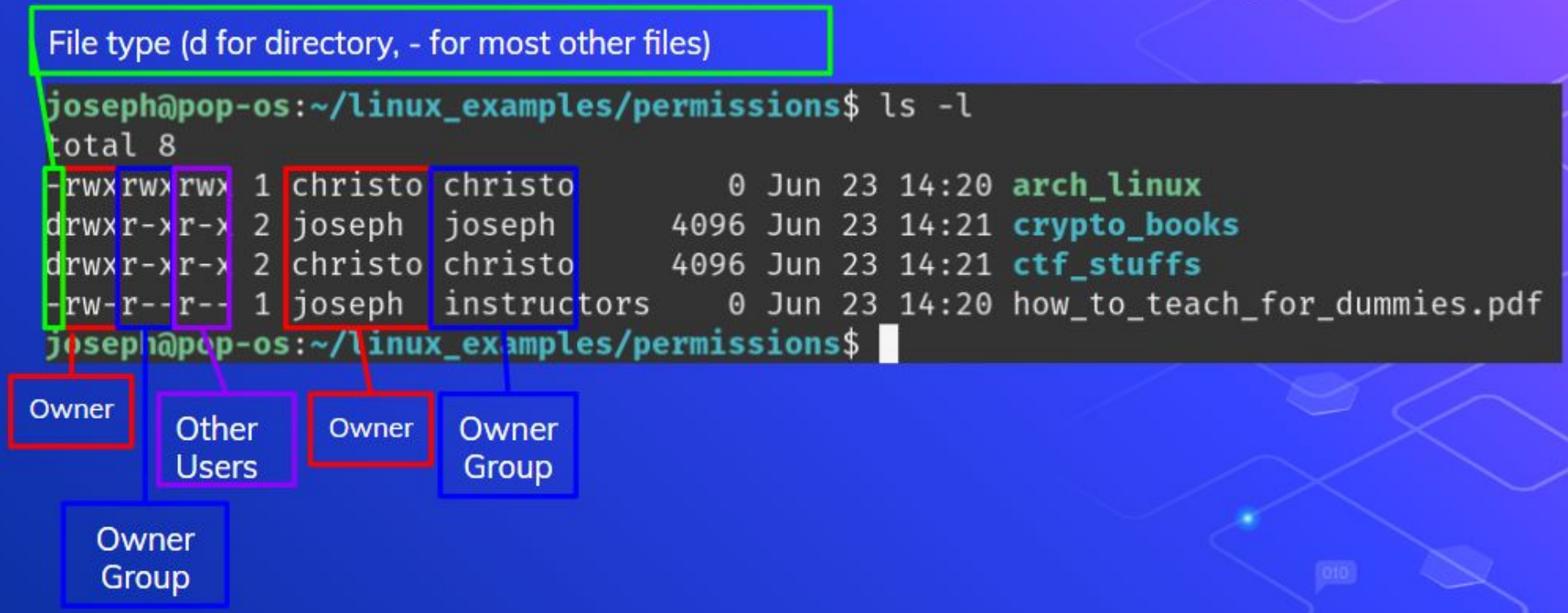
- There is a way to do this in /etc, but let's go over the GUI way, because it's easy to use.
- System Settings -> Software & Updates
- Ubuntu Software (Click two of them, nothing else)
  - Canonical-supported free and open-source software (main)
  - Community-maintained free and open-source software (universe)
  - Download from: Server for the United States

More on the Software Update Center

- Other Software
  - Canonical Partners
  - NEVER cdrom
- Updates (Check two)
  - Important security updates
  - Recommended updates
  - Check for updates Daily
  - When there are security updates: Download and install automatically
  - When there are other updates: Display weekly
  - Notify me of new Ubuntu version: Never
- apt-get update
  - Updates the packages based on your new changes
- apt-get upgrade

On to File Permissions...

Permissions



The Simple Way to Change Permissions

sudo chmod [permission_set] <filename>

sudo chown <new_owner> <filename>

sudo chgrp <new_group> <filename>

To set permission set:

u,g,o +/-/= rwx

Ex: sudo chmod o-wx a_file

Ex: sudo chmod u+rwx,o=r a_file

The Harder, but Faster Way to Set Permissions

- Let's start with a lesson on binary.
- binary has 2 digits: 1 & 0 (true or false)

$000_2=0_{10}$
$001_2=1_{10}$
$010_2=2_{10}$
$011_2=3_{10}$
$100_2=4$, etc...

2^2 2^1 2^0

Now tell me: What is 110_2?

how to convert binary to decimal:

- start at the rightmost digit
- that digit is 2^0
- the next digit is 2^1
- then 2^2,... and so on
- 1 means that the term exist. For instance, if the number is 001_2, that means that the 2^0 term exists. 2^0 is 1, so this number equals 1.
- So let's work through $101_2=$
- Remember, start from the right. 2^0 exists, because there is a 1 there. 2^1 doesn't. There is a 0 there. 2^2 exists because there is a 1 there. So... $2^2+0^1+2^0=4+0+1=5_{10}$

The Harder, Faster Way, continued

So basically Linux can represent a permission set (rwx) as either yes-or-no (true or false) so 0 or 1. So if a file had rwxrwxrwx it would be like 111111111. But remember! A permission set is just a group 3 at a time so it's more like 111 111 111. Based on the last slide, we can therefore turn it into $111_2=7_{10}$. So if we wanted to set rwxrwxrwx to a file we can use the command:

chmod 777 <file name>

Instead of chmod =rwxrwxrwx <file name>

So what would it be if I wanted it to be rw-r--r--?

Permissions

```
joseph@pop-os:~/linux_examples/permissions$ ls -l
total 8
-rwxrwxrwx 1 christo christo        0 Jun 23 14:20 arch_linux
drwxr-xr-x 2 joseph  joseph      4096 Jun 23 14:21 crypto_books
drwxr-xr-x 2 christo christo     4096 Jun 23 14:21 ctf_stuffs
-rw-r--r-- 1 joseph  instructors    0 Jun 23 14:20 how_to_teach_for_dummies.pdf
joseph@pop-os:~/linux_examples/permissions$
```

Why Do We Care About Permissions?

- If the wrong people have access to some of our important files, like /etc/passwd for instance, they can make changes that will leave our system vulnerable.
- Knowing permissions also helps with Forensics questions like, “Who owns the image in the /home directory?”
- To view both hidden files and permissions of files at the same time, use the command ls -la.

Forensics

- forensics: finding information about a computer or system (broad)
  - CyPat: specifically information about OS, cryptography, or info about files
- Google = friend because there's always going to be something to research every round
- Here are some tips and tricks:
  - check for hidden files (ls and its options are quite helpful)
  - understand your system
  - recognize certain patterns (hex, binary, etc.): http://rumkin.com/tools/cipher/
  - look into find and grep commands and their many options

Find command

- looks for file names and directories
- General Syntax: **find <filename/path>**
- to find a specific filetype: **find <directory> -name *.<file extension>**
  - -name: matches the information that comes after it
  - * : displays EVERY file with said .<file extension>
  - directory: / (to be able to search through the whole system)
- to find hidden files: **find ~ -type f -name "*."**
  - -type: helps you to search by what you're looking for (f= file, d= directory)
  - *. → shows hidden files (shown with a . before the filename)
- There are A LOT more options and configurations, so do some research to find out more about them!

Grep command

- looks for patterns and expressions inside files and text
- General Syntax: **grep <pattern> <file>**
- to find certain file type: **find <directory> | grep [.]<file extension>**
  - |: pipe- funnels the output from find command to be used by grep
  - searches for the file extension after all the files in the directory are shown
- find + grep together = pretty epic combination
- bunch of things you can do with grep (can be used to search many different configuration files) so research!