Lecture 5 (1-23-26)

Networking

Motivating Questions

1. What is my **IP Address**?
2. What **services** are running on my machine?
3. What is Notre Dame's **IP Address**?
4. How do I retrieve something from the **web**?
5. How do we measure **bandwidth** and **latency**?
6. How do I **transfer** files between machines?
7. What **services** are running on your machine?

- Probably won't cover all these today

How to view process on the machines

- `ps` shows processes for current shell session
- use `ps aux` to view all processes on the machine for HW1

Find:

- find has built in mechanism to filter
  - `find src -type d -iname '*20289*'`
    - `-type d` indicates directories
    - `-iname '*20289*'` uses wildcards
  - `iname` vs `name`
    - `iname` doesn't consider capitals
      - check the man page
  - the `find` command will be very useful in HW2

checking if so and so are writing trolls to the machine

- `ps aux | grep TROLL`
  - make sure you turn off your trolls after the HW
- `find . > /dev/null`
  - sends output to the black gole
  - but it is running

    - if you use control-Z then the job is paused
- if you use the `jobs` command then you can see the list of jobs
- if you use the command `bg` then it runs in the background
    - to check if its still running you can use
        - `jobs`
        - `ps ux` -> shows your running programs
- `find . > /dev/null &` -> starts the job in the background instead of foreground
- with `pkill name_of_program` you can kill the program by using the name instead of:
    - using the PID
    - or the job number
- **every job is a process**

What is the Internet?

- **loose, unstructured, chaotic, ad hoc collection** of **networks**, bound together by **standards**
    - Designed to tolerate failures
    - networks of networks
    - Its more than just a web

Analogy on internet

- pbui laptop named `deadpool`
- his wifes laptop is named `cable`
- in between the two laptops he has `hermis`
    - this is the **router**
- `deadpool` sends **packets** to `hermis` and `hermis` sends it to `cable`
- since this is all in pbui's house this is called a **LAN** (local)
- if pbui wants to watch "KPOP demon hunters" then he needs to access outside the the local network
- so he needs to access Youtube (owned by Google)
- so pbui uses his xfiniity service which talks to Google who talks to Youtube servers
- now pbui wants to text TA Sam (sam uses verizon)
- pbui phone named `kidpool`
    - so `kidpool` sends **packets** to `hermis` who sends **packets** to xfinity who sends to packets to verizon who sends packets to Sam's phone
- the internet is just a graph

What is the Client / Server model

- suppose you have a client
  - it wants to talk to the student machines
  - the person it makes a request to is called the **server**
    - client **server** can refer to machines and simply **processes**
      - in this class it's usually **processes**
- in order to talk to the **server** you need to know the **IP Address**
  - on the student machine i can use `ip addr`
    - but this gives a lot of info
      - `ip -br addr` will return a summary of `ip addr`
        - `-br` stands for brief
- there is an ip_address called **local host**
  - `127.0.0.1`
    - every machine has this ip address
    - it is used for applications to talk to the current machine without having to go over the network
  - `192.168.---.-`
    - all routers gives an ip address with this
    - this means we only have 2^32 ip addresses
      - we don't have enough ip addresses since each device needs one
- Analogy: I usually don't call parents with their phone number, usually call via contact
  - so ip addresses have a **hostname** (this is basically the content)
- if you do `host google.com`
  - you will get an ip address
    - if you enter this ip address in any browser it will take you to google
      - this is because **browsers** are client side applications
- so basically when you are using a browser
  - you enter a website
    - internally the browser converts the url to the 4 byte (32 bit) IP address using a program called DNS
- each individual application on a machine needs a separate **port**
  - so you need to know the ip address and the port
- use `ss -tlpn`
  - to see what is listening on the student machines
- This command is used to scan what ports its listening on:
  - `nmap -v -Pn -p 1000-1999 something_else_here`

To talk to another machine you need to know:

- its **IP Address**
- its **port**
- its **protocols**

All chrome does is:

- starts a socket connection with google (application)
- sends a couple of text lines
- the internet is simply unix philosophy