Plan

* Recitation Qs
* Recap: Layers
* E2E Overview
* Breakout rooms post & corr
* Name that layer

Logistics

* Midterm on Mar 31
* Technical feedback on OPP: Apr 18
* Spring break next week!
Recitation Qs

1. What is the end-to-end argument?
   - Functionality should be provided at the highest “layer” possible

2. How is E2E argument used in practice?

3. Do you agree w/ E2E arg? Why?
Layering (Recap)

- Application: HTTP, SSH, ...
- Transport: TCP/UDP
- Network: IP
- Link: Ethernet, wifi, ...
- Physical: Cable, RF

One way to think about E2E:

It's the libertarian view of systems.

- Let the application handle it.
  - Your job (e.g. as the network) is to do the minimum necessary and get out of the way.
  - Full freedom for app
    - More work
    - More flexibility
    - More chance to get it wrong

"Dumb network"
Case study: Robust file Xfer
Where should acks happen?
Examples in computer systems

- Unix file system
  File is just a stream of bytes, not a Database

- Ethernet CRC check
  Good idea?

- Security in Ethernet, IP
  There is none? Left to application

- RISC versus CISC
  x86 - 64 x 3000 instructions
  RISC V = 50 (base)
  Good idea?
Small Groups

Benefits of E2E (“let application do it”) versus implementing functionality at lower layers? Drawbacks?

Examples of E2E not in computer systems?
- Car with radio?
Benefits of E2E

+ Flexibility — explain TCP versus UDP
  e.g. UDP for gaming, streaming, ...
+ Simplicity of impl
+ Separation of concerns
+ Less waste, less redundancy

Drawbacks

- Redundancy — every app has to reimplement it
  e.g. you computer has many crypto libraries
  must keep all patched
- Lose opportunities for optimization
  e.g. caching in network
  e.g. prioritization in network
  e.g. error correction in network (?)
Poll: Which layer? Why?

- Authentication
- Encryption
- Denial-of-service prevention
- Prioritization of traffic
- Guarantee of tx/rx rate
- Fault tolerance
- Internet censorship
- Firewalling/blocking malicious traffic
- Caching