Administrivia

Midterms “returned” via gradescope yesterday
- Everyone able to get theirs?
- Will enable regrades after class — use gradescope links

HW3 due Thursday night

Today - overview of networking
Then - client-side and server-side TCP sockets
Rest of the quarter

HW4 out Friday; due last Wednesday of the quarter, 8/16 (plus late days if you have & need them)

A few more exercises, but nothing due until next Mon.

Final exam (i.e., 2nd midterm) in class last Friday, 8/18

- Review in section the previous day
Networks from 10,000ft

Internet

clients

servers

Google

hulu

facebook
The “physical” layer

Individual bits are modulated onto a wire or transmitted over radio

- physical layer specifies how bits are encoded at a signal level
- e.g., a simple spec would encode “1” as +1V, “0” as -1V

1 0 1 0 1 →

```
<table>
<thead>
<tr>
<th>computer</th>
<th>NIC</th>
<th>copper wire</th>
<th>optical cable</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>radio frequency band</td>
<td></td>
</tr>
</tbody>
</table>
```

physical physical
The “data link” layer

Multiple computers on a LAN contend for the network medium

- media access control (MAC) specifies how computers cooperate
- link layer also specifies how bits are packetized and NICs are addressed
The “network” layer (IP)

The Internet Protocol (IP) routes packets across multiple networks

- every computer has a unique Internet address (IP address)
- individual networks are connected by routers that span networks
The “network” layer (IP)

Protocols to:

- let a host find the MAC address of an IP address on the same network
- let a router learn about other routers and figure out how to get IP packets one step closer to their destination
The “network” layer (IP)

Packet encapsulation

- an IP packet is encapsulated as the payload of an Ethernet frame
- as IP packets traverse networks, routers pull out the IP packet from an ethernet frame and plunk it into a new one on the next network
The “transport” layer (TCP, UDP)

**TCP**

- the “transmission control protocol”
- provides apps with reliable, ordered, congestion-controlled byte streams
- fabricates them by sending multiple IP packets, using sequence numbers to detect missing packets, and retransmitting them
- a single host (IP address) can have up to 65,535 “ports”
  - kind of like an apartment number at a postal address
The “transport” layer (TCP, UDP)

**TCP**

- useful analogy: how would you send a book by mail via postcards?
- split the book into multiple postcards, send each one by one, including sequence numbers that indicate the assembly order
- receiver sends back postcards to acknowledge receipt and indicate which got lost in the mail
The “transport” layer (TCP)

Packet encapsulation -- same as before!

- Ethernet header
- IP header
- TCP header
- TCP payload
- IP payload
- Data
- Source address
- Destination address
- Ethernet payload
The “transport” layer (TCP)

Applications use OS services to establish TCP streams

- the “Berkeley sockets” API -- a set of OS system calls
- clients `connect()` to a server IP address + application port number
- servers `listen()` for and `accept()` client connections
- clients, servers `read()` and `write()` data to each other
The “transport” layer (UDP)

UDP

- the “user datagram protocol”
- provides apps with unreliable packet delivery
- UDP datagrams are fragmented into multiple IP packets
  - UDP is a really thin, simple layer on top of IP
Layer 5: session layer
- supposedly handles establishing, terminating application sessions
- RPC kind of fits in here

Layer 6: presentation layer
- supposedly maps application-specific data units into a more network-neutral representation
- encryption (SSL) kind of fits in here
The “application” layer

Application protocols

- the format and meaning of messages between application entities
- e.g., HTTP is an application level protocol that dictates how web browsers and web servers communicate
  - HTTP is implemented on top of TCP streams
The “application” layer

Packet encapsulation -- same as before!
The “application” layer

Packet encapsulation -- same as before!

```
| ethernet header | IP header | TCP header | HTTP header | HTTP payload (e.g., HTML page) |
```
The “application” layer

Popular application-level protocols:

- **DNS**: translates a DNS name (**www.google.com**) into one or more IP addresses (74.125.155.105, 74.125.155.106, ...)
  - a hierarchy of DNS servers cooperate to do this
- **HTTP**: web protocols
- **SMTP, IMAP, POP**: mail delivery and access protocols
- **ssh**: remote login protocol
- **bittorrent**: peer-to-peer, swarming file sharing protocol
See you on Wednesday!