CSE 333
Lecture 16 -- networks

Hal Perkins

Department of Computer Science & Engineering
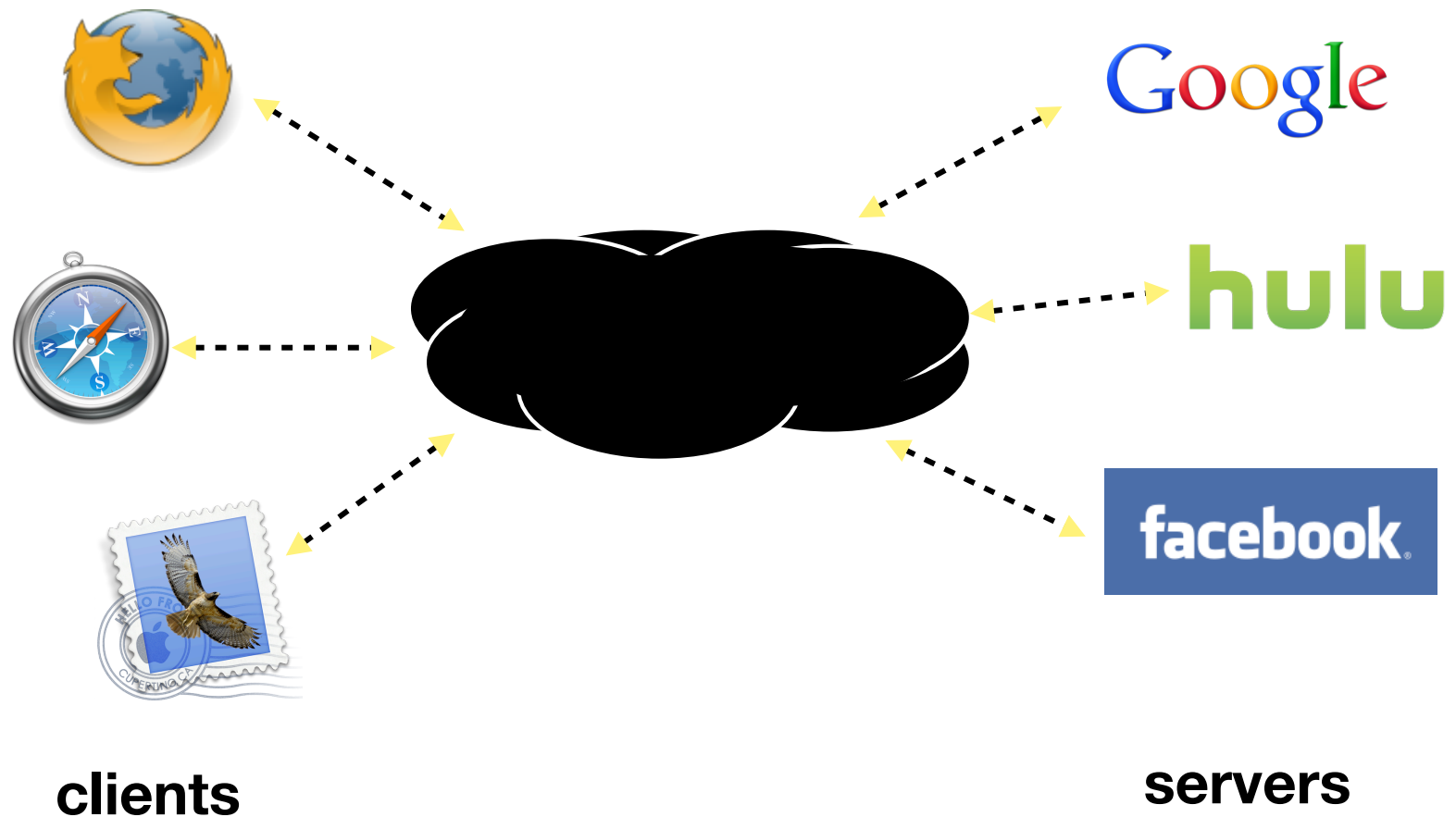
University of Washington

Administrivia

HW3 due Next Thursday

- No exercise over the weekend; probably one out Monday due Wednesday, then no more due next week

Today - overview of networking

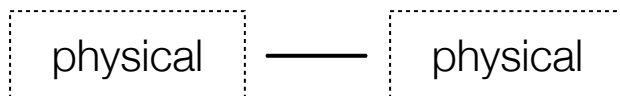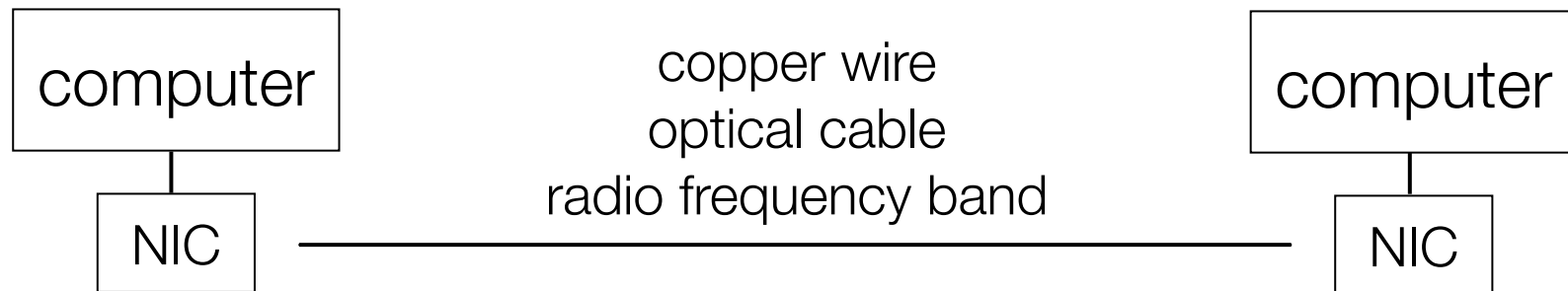Next week - client-side and server-side TCP sockets

Networks from 10,000ft

clients

servers
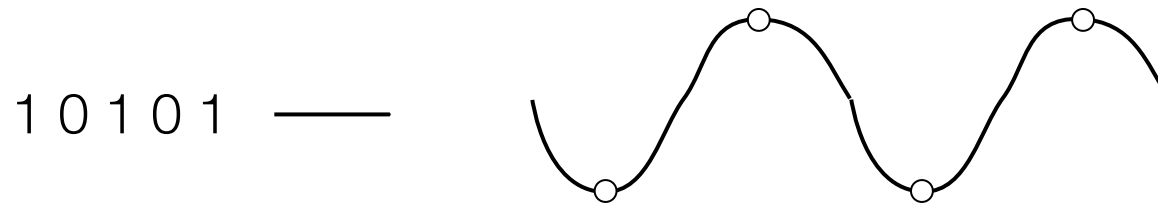
The "physical" layer
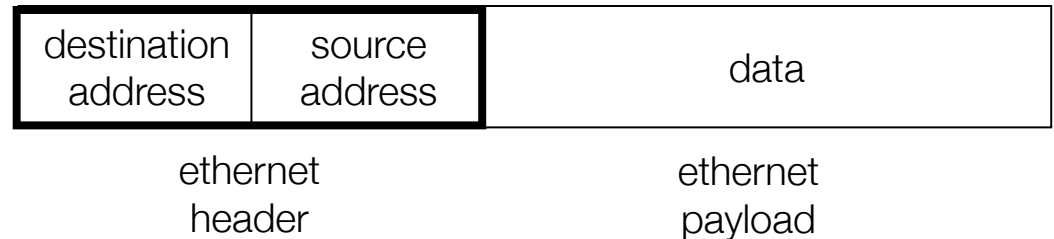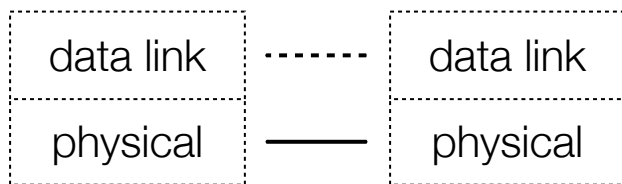
Individual bits are modulated onto a wire or transmitted over radio

‣ 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 ——

| computer |
| NIC |

copper wire
optical cable
radio frequency band

| computer |
| NIC |

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

00:1d:4f:47:0d:48	4c:44:1e:8f:12:0e	7a:37:8e:fc:1a:ea	de:ad:be:ef:ca:fe	01:23:32:21:ab:ba
computer	computer	computer	computer	computer
NIC	NIC	NIC	NIC	NIC

ethernet

data link	......	data link
physical	———	physical

destination address	source address	data
ethernet header		ethernet payload
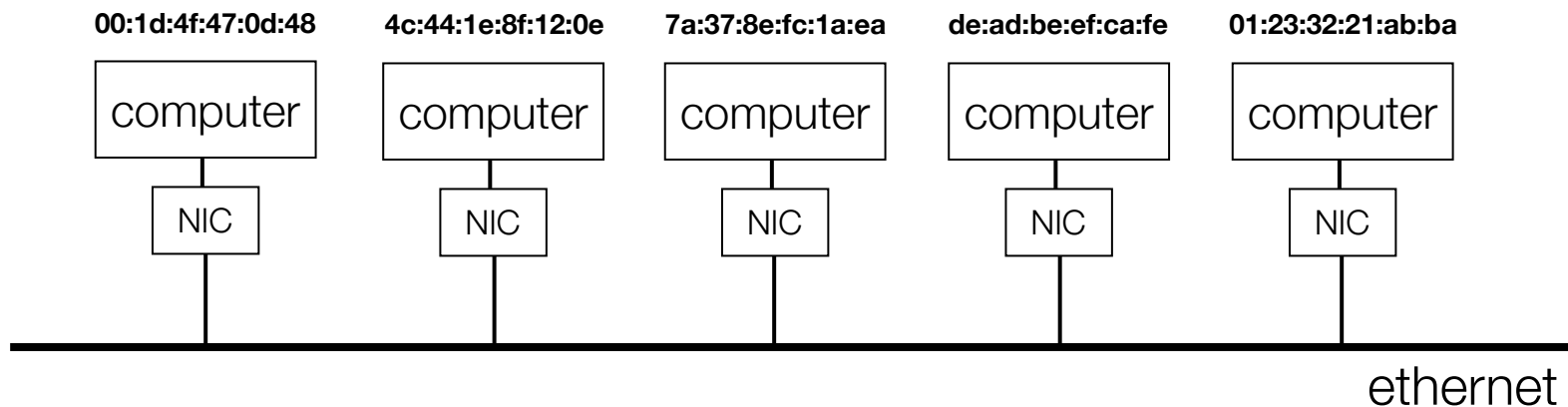
The "network" layer (IP)

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

128.95.10.55   **128.95.10.72**   **128.95.10.95**

host   host   host

ethernet

128.95.10.1

router

128.95.4.1

ethernet

host   host   host

128.95.4.3   **128.95.4.10**   **128.95.4.12**

network	....	network
data link	....	data link
physical	——	physical
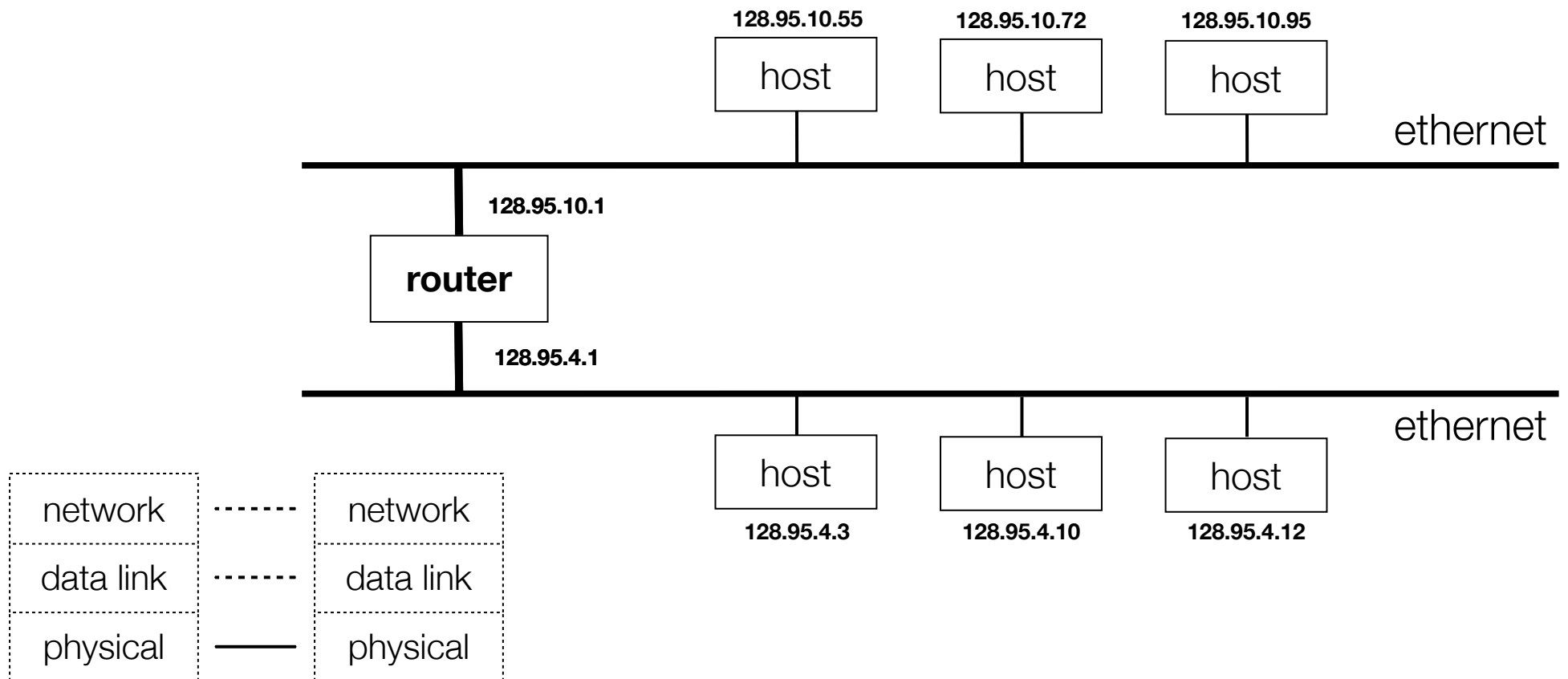
The "network" layer (IP)

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
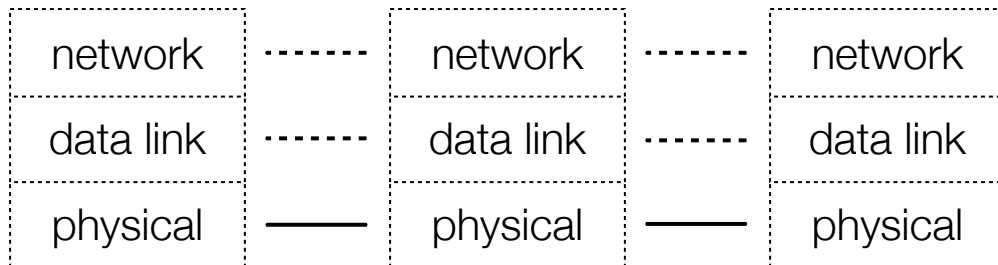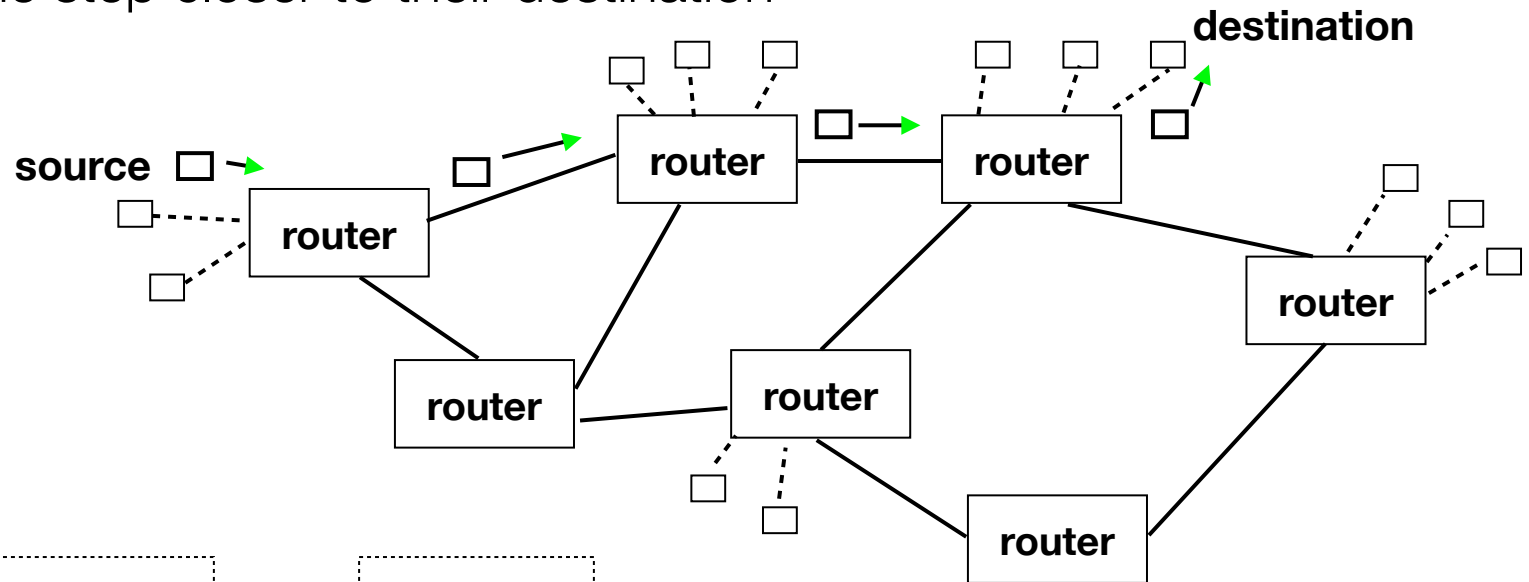
The "network" layer (IP)

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
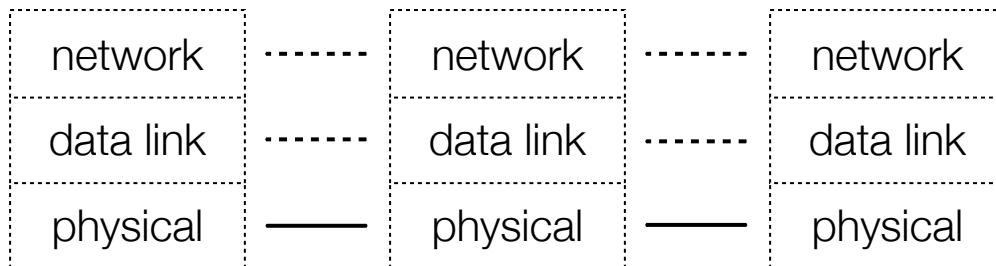
IP header	IP payload

destination address	source address	data

ethernet header   ethernet payload

network	------	network	------	network
data link	------	data link	------	data link
physical	——	physical	——	physical
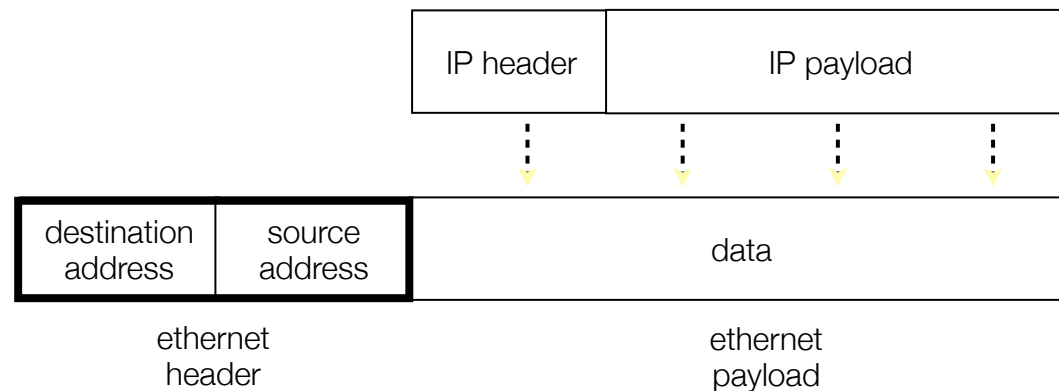
The "transport" layer (TCP, UDP)

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

transport	····························			transport
network	·······	network	·······	network
data link	·······	data link	·······	data link
physical	——	physical	——	physical
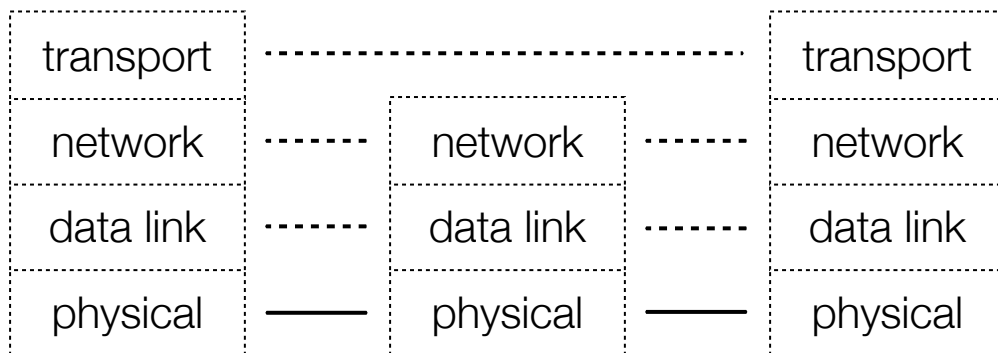
The "transport" layer (TCP, UDP)

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

transport	....			transport
network	......	network	......	network
data link	......	data link	......	data link
physical	——	physical	——	physical
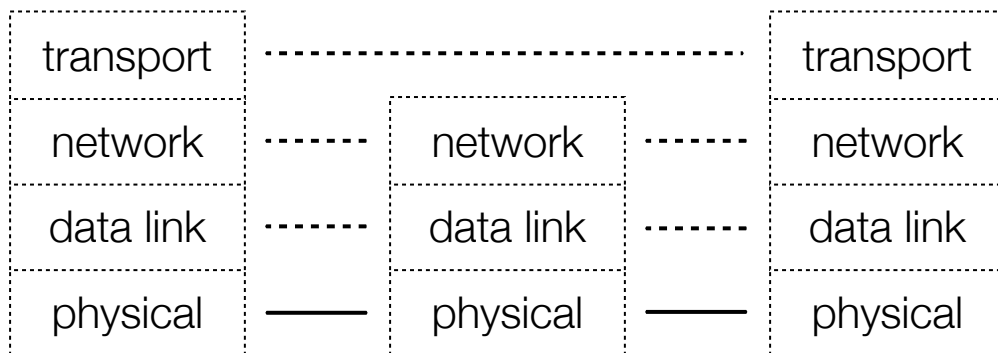
The "transport" layer (TCP)

Packet encapsulation -- same as before!

TCP header	TCP payload

IP header	IP payload

destination address	source address	data

ethernet header   ethernet payload

transport	....	transport
network	......	network	......	network
data link	......	data link	......	data link
physical	——	physical	——	physical
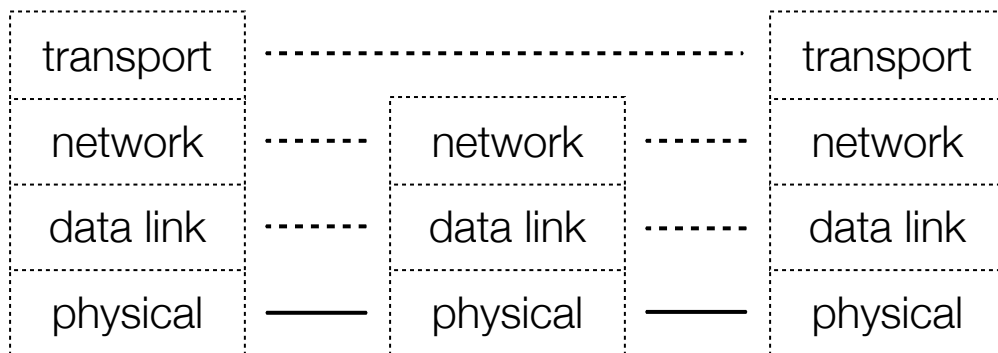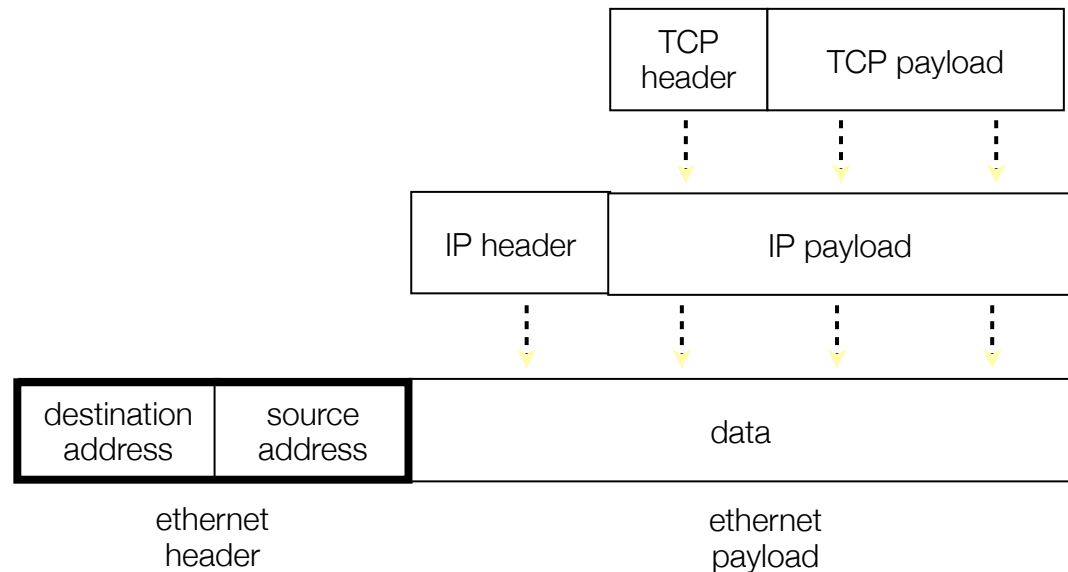
The "transport" layer (TCP)

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

transport	....			transport
network	......	network	......	network
data link	......	data link	......	data link
physical	——	physical	——	physical
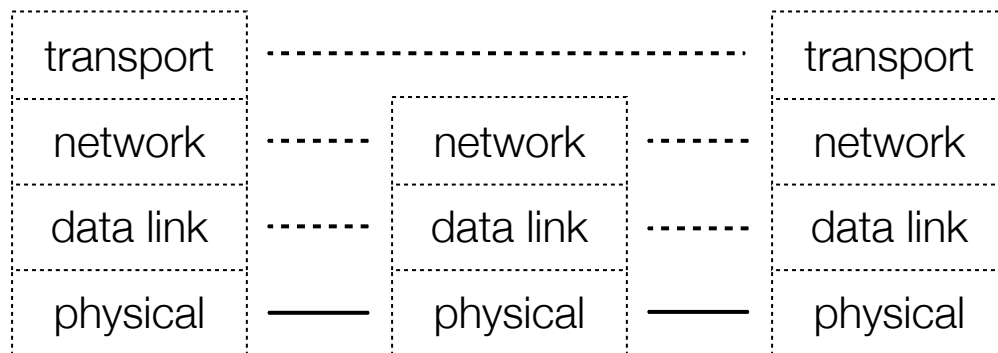
The "transport" layer (UDP)

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

transport	····························			transport
network	······	network	······	network
data link	······	data link	······	data link
physical	——	physical	——	physical
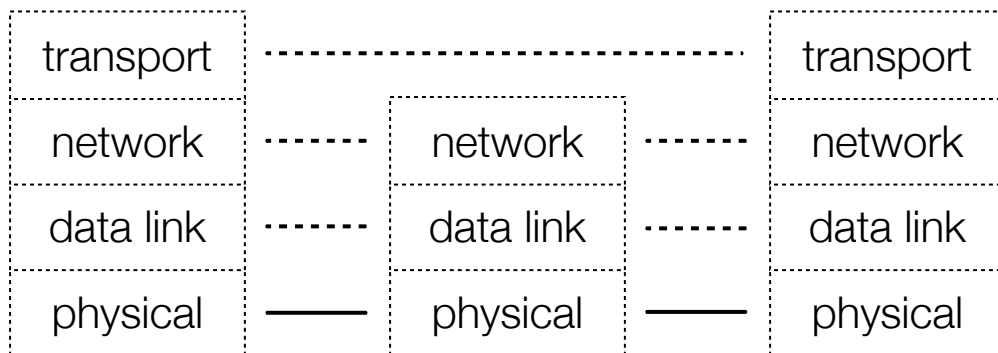
The (mostly missing) layers 5,6
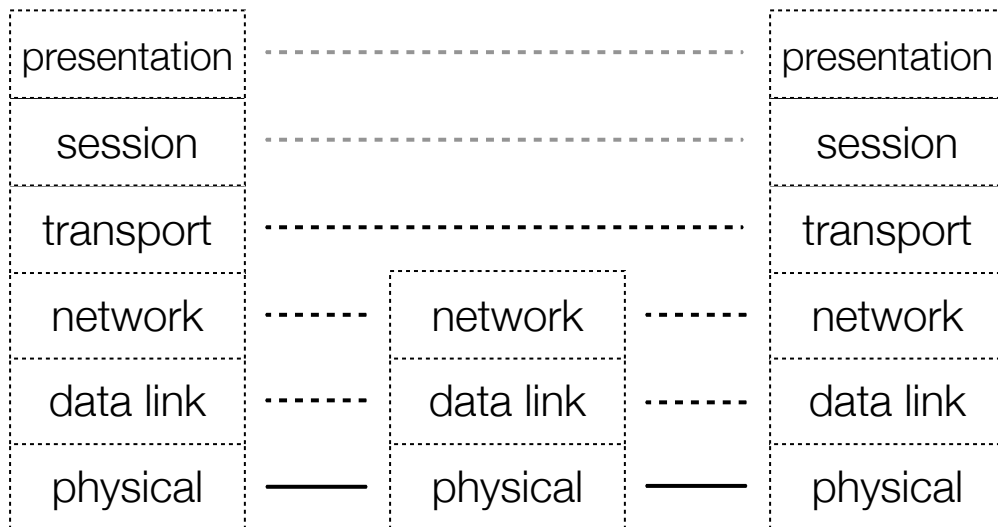
Layer 5: session layer

‣ supposedly handles establishing, terminating application sessions
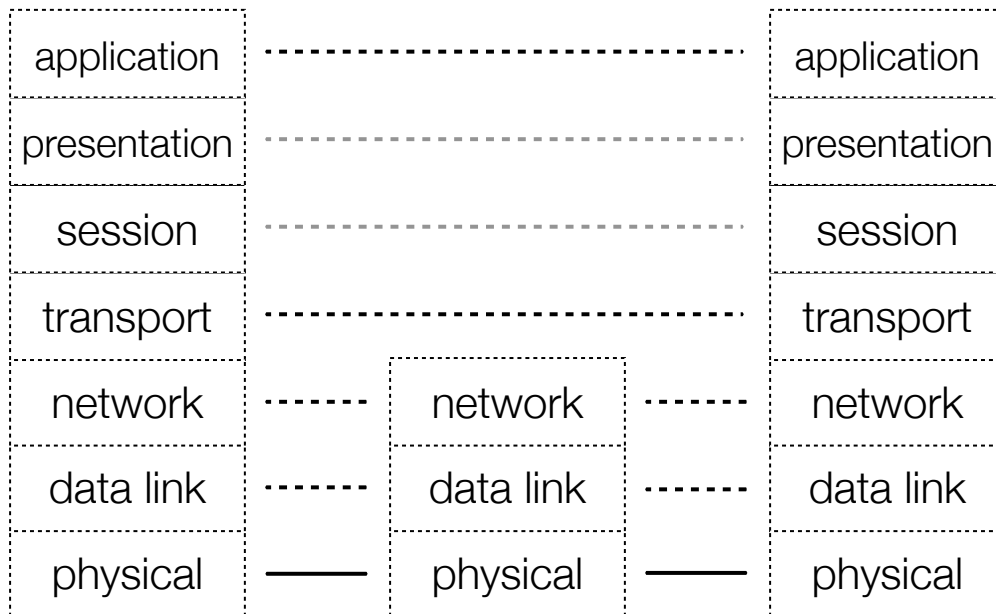
‣ RPC kind of fits in here

Layer 6: presentation layer

‣ supposedly maps application-specific data units into a more network-neutral representation

‣ encryption (SSL) kind of fits in here

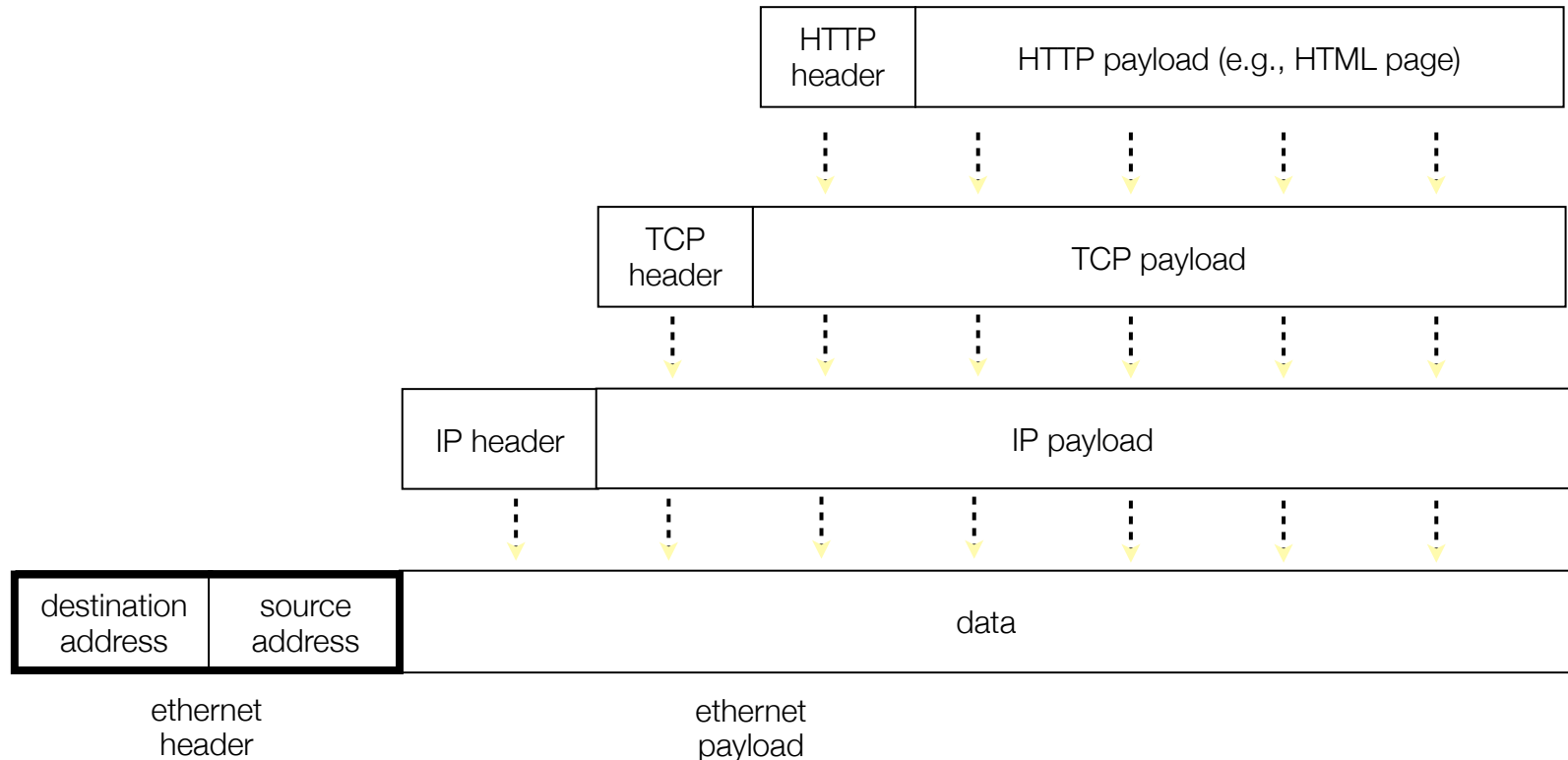| presentation | - - - - - - - - - - - - - - - - | | presentation |
session	- - - - - - - - - - - - - - - -		session
transport	- - - - - - - - - - - - - - - -		transport
network	- - - -	network	- - - -	network
data link	- - - -	data link	- - - -	data link
physical	——	physical	——	physical

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

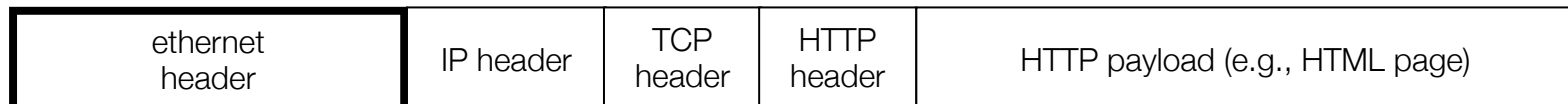| application	application
presentation	presentation
session	session
transport	transport

network	network	network
data link	data link	data link
physical	physical	physical

The "application" layer

Packet encapsulation -- same as before!

HTTP header	HTTP payload (e.g., HTML page)

TCP header	TCP payload

IP header	IP payload

destination address	source address	data

ethernet
header

ethernet
payload

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 Monday!