CSE160: Computer Networks

Lecture #02 – Protocols and Layering

2020-09-01

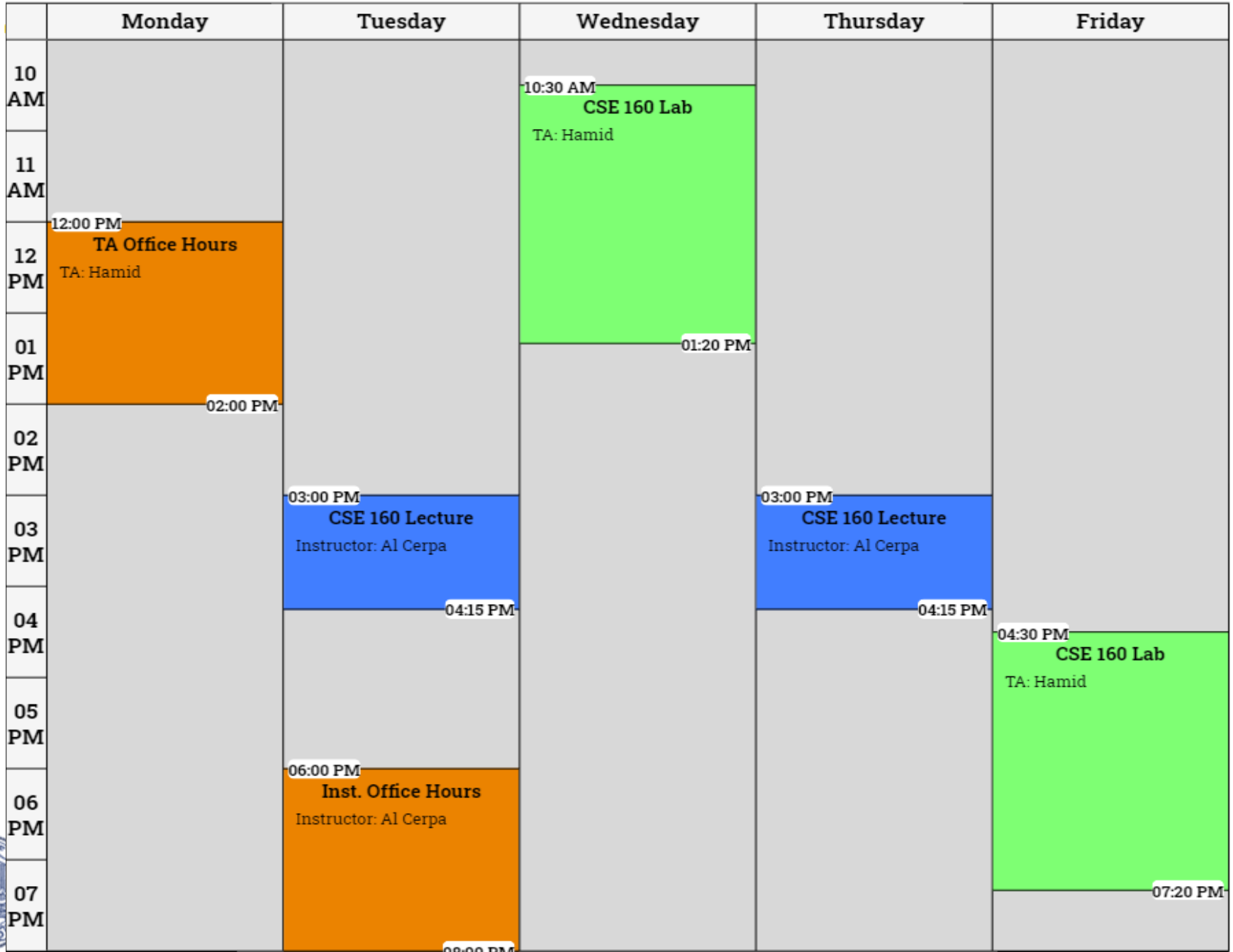
Professor

Alberto E. Cerpa

Contact Hours

- Contact time sometime **any** day of the week!
  - All labs are open for anyone that needs them (2 labs/week)
    - Preference is given to students registered for the lab in any particular day
  - Office hours on Mon, and Tue, covering all days of the week bw Lectures, Labs and Office Hours
  - Still working to find you tutors!

CSE 160 Contact Hours



Projects and Homework

- Both homework 1 and project 1 are already available on the class web page
- Check the full schedule in the syllabus for more information regarding due dates, etc.
- You should concentrate on getting the TinyOS development environment in place, so you can start working on project 1 asap!
- Proj 1 is due in 21 days, do not waste time!
- I have put 2 presentations in the Project directory
  - A brief tutorial of C for those that need a review
  - A brief intro to TinyOS and TOSSIM

Last Time …

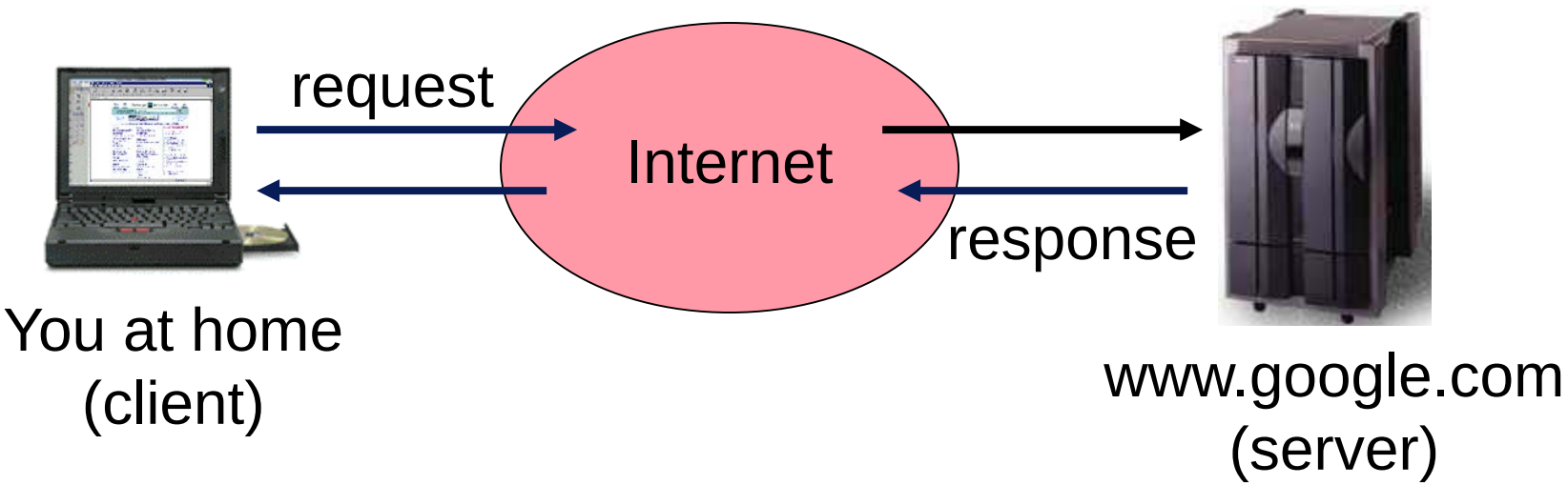
- Networks are used to share distributed resources
  - Key problems revolve around effective resource sharing
- Statistical multiplexing
  - It's well-suited to data communications

This Lecture

1. A top-down look at the Internet
2. Mechanics of protocols and layering
3. The OSI/Internet models

1. A Brief Tour of the Internet

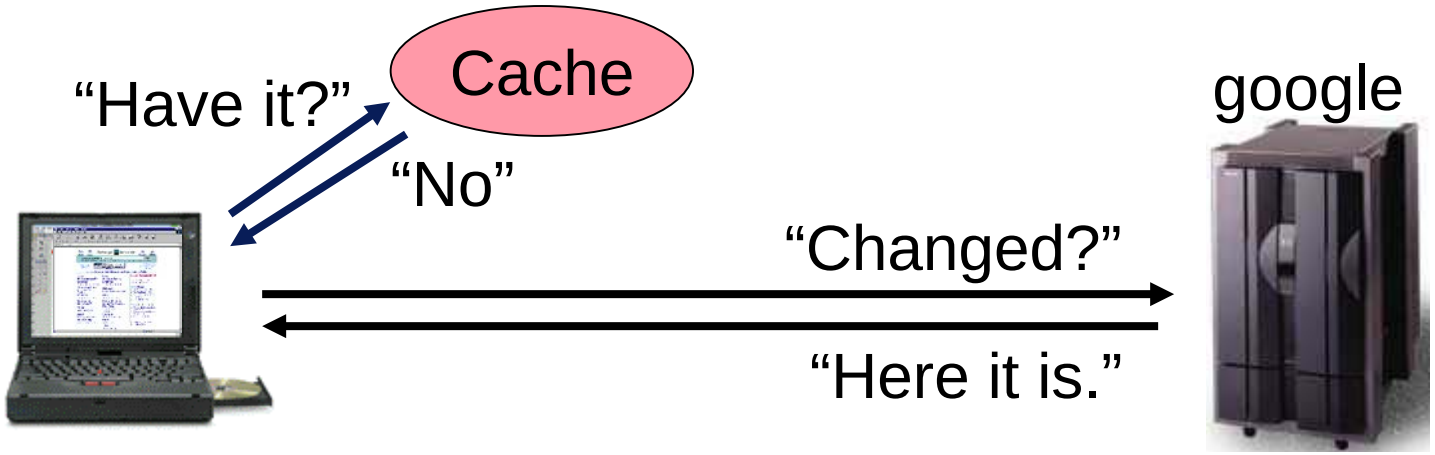
- What happens when you “click” on a web link?



- This is the view from 10,000 ft …

9,000 ft: Scalability

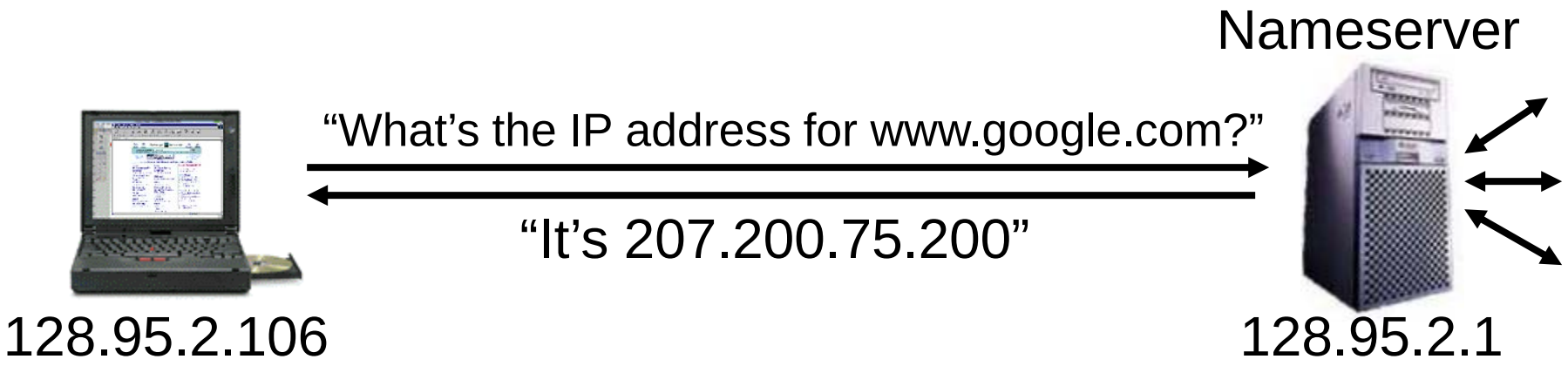
- Caching improves scalability



- Why?
- We cut down on transfers:
  - Check cache (local or proxy) for a copy
  - Check with server for a new version

8,000 ft: Naming (DNS)

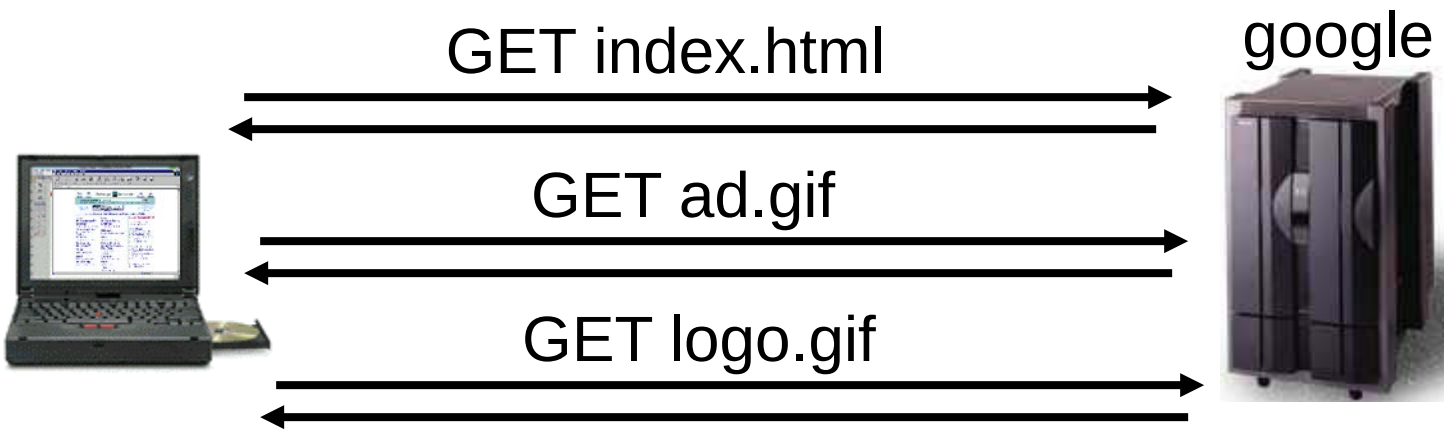
- Map domain names to IP network addresses



- All messages are sent using IP addresses
  - So we have to translate names to addresses first
  - But we cache translations to avoid doing it next time (why?)

7,000 ft: Sessions (HTTP)

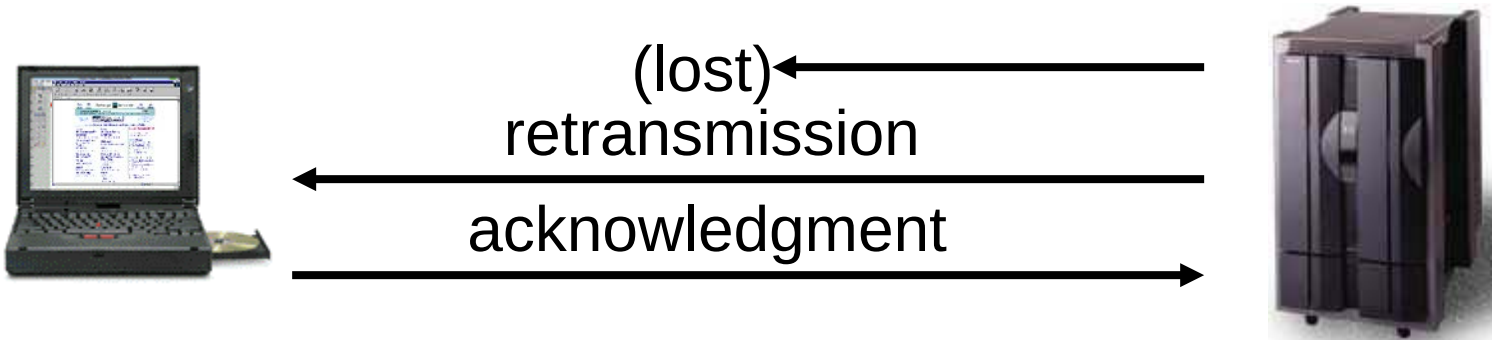
- A single web page can be multiple "objects"



- Fetch each "object"
  - either sequentially or in parallel

6,000 ft: Reliability (TCP)

- Messages can get lost



- We acknowledge successful receipt and detect and retransmit lost messages (e.g., timeouts)

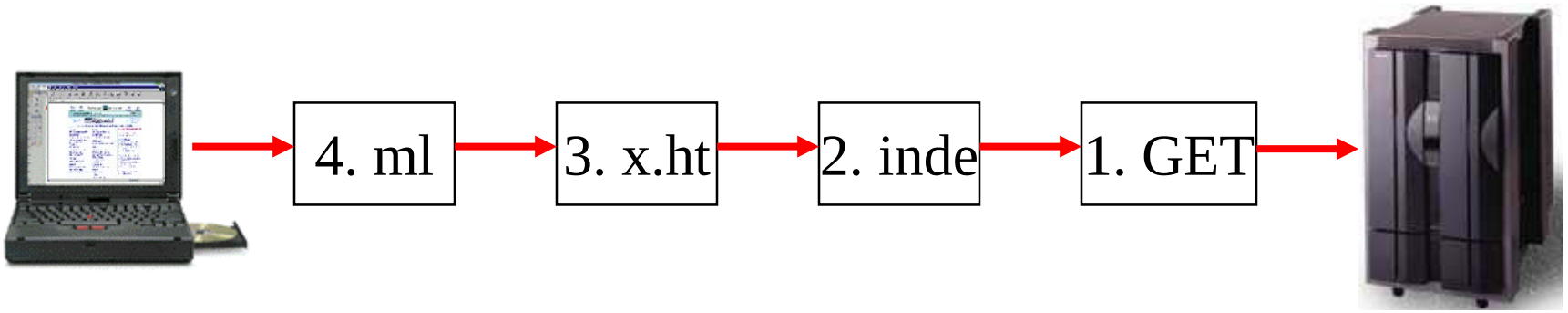
5,000 ft: Congestion (TCP)

- Need to allocate bandwidth between users

- Senders balance available and required bandwidths by probing network path and observing the response

4,000 ft: Packets (TCP/IP)

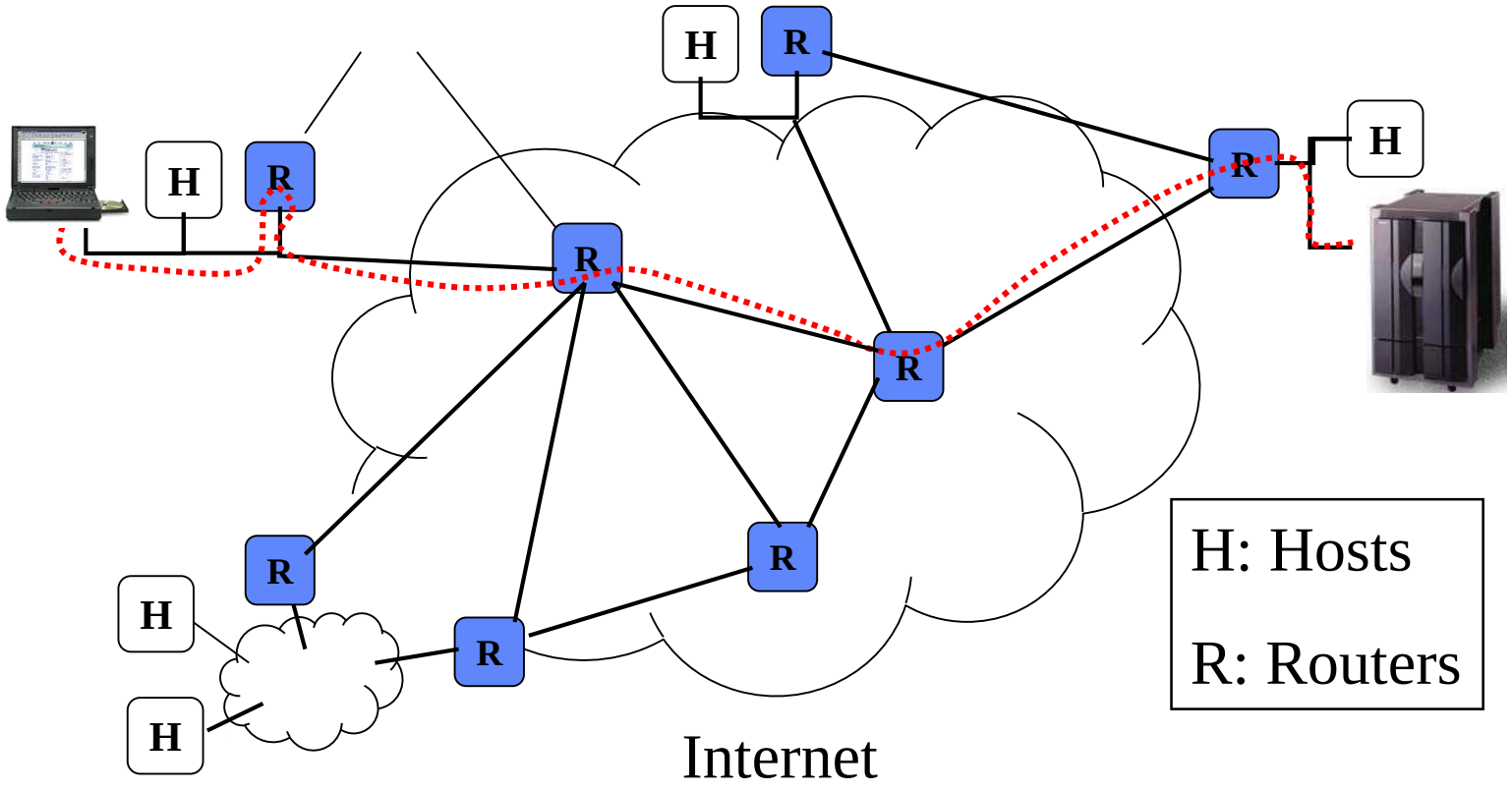
- Long messages are broken into packets
  - Maximum Ethernet packet is 1.5 Kbytes
  - Typical web page is 10 Kbytes



- Number the segments for reassembly

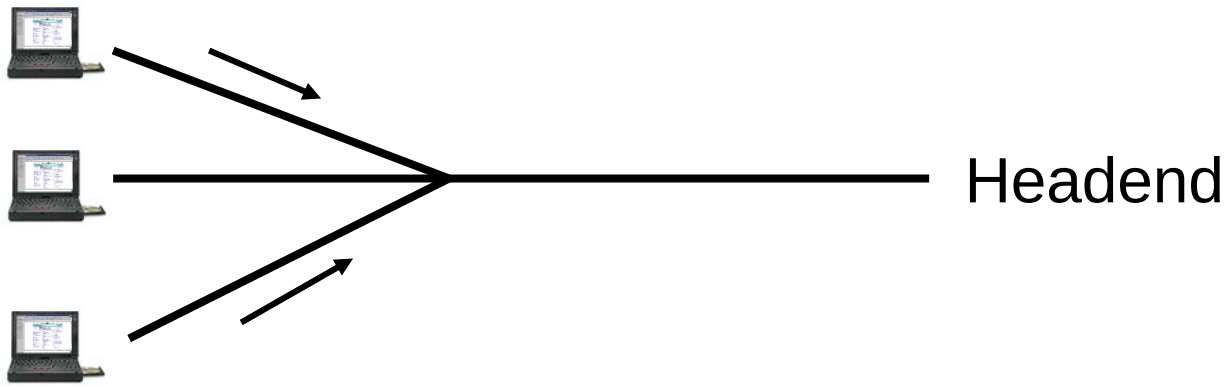
3,000 ft: Routing (IP)

- Packets are directed through many routers



2,000 ft: Multi-access (e.g., Cable)

- May need to share links with other senders



- Poll headend to receive a timeslot to send upstream
  - Headend controls all downstream transmissions
  - A lower level of addressing (than IP addresses) is used … why?

1,000 ft: Framing/Modulation

- Protect, delimit and modulate payload as signal

Sync / Unique	Header	Payload w/ error correcting code

- E.g, for cable, take payload, add error protection (Reed-Solomon), header and framing, then turn into a signal
  - Modulate data to assigned channel and time (upstream)
  - Downstream, 6 MHz (~30 Mbps), Upstream ~2 MHz (~3 Mbps)

Networks Need Modularity

- The network does much for apps:
  - Make and break connections
  - Find a path through the network
  - Transfers information reliably
  - Transfers arbitrary length information
  - Send as fast as the network allows
  - Shares bandwidth among users
  - Secures information in transit
  - Lets many new hosts be added
  - …
- We need a form of modularity, to help manage complexity and support reuse

2. Protocols and Layers

- We need abstractions to handle all this system complexity

  A protocol is an agreement dictating the form and function of data exchanged between parties to effect communication

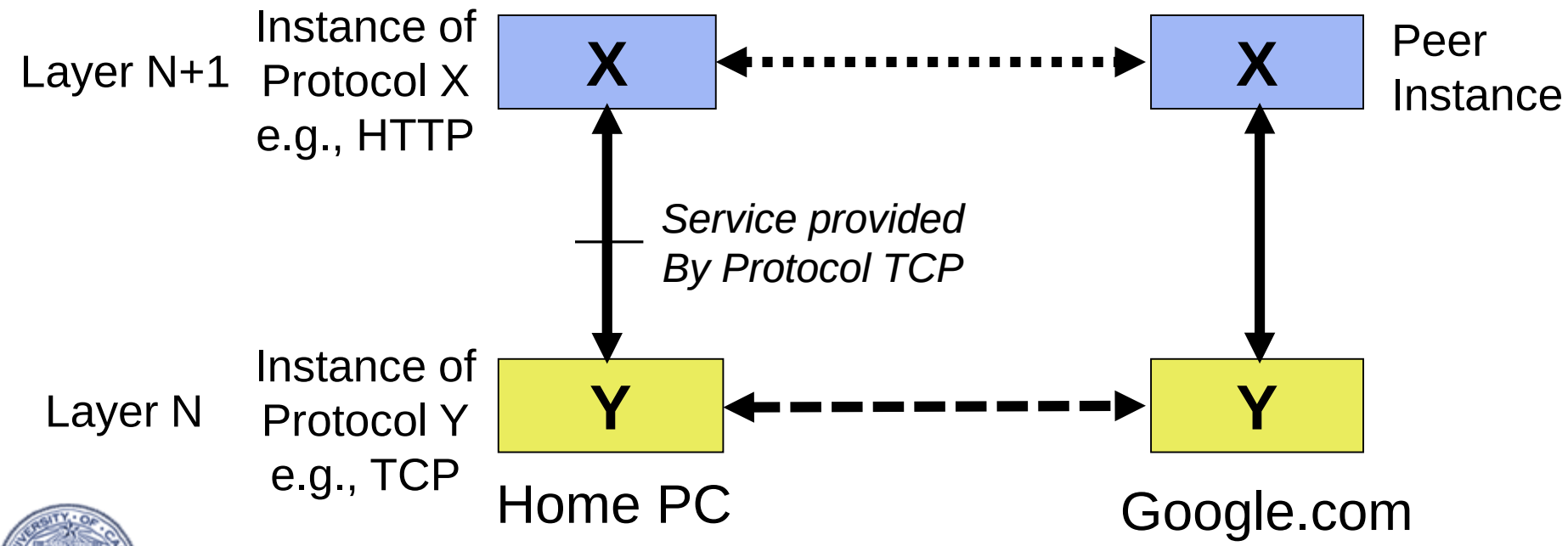
- Two parts:
  - Syntax: format -- where the bits go
  - Semantics: meaning -- what words mean, what to do with them
- Do you know any?
- Examples:
  - Ordering food from a drive-through window
  - IP, the Internet protocol
  - TCP and HTTP, for the Web

Protocols and Layers (2)

- Protocols and layering are the main structuring methods to divide up network functionality
  - Each instance of a protocol talks virtually to its peer using the protocol
  - Each instance of a protocol uses only the services of the lower layer

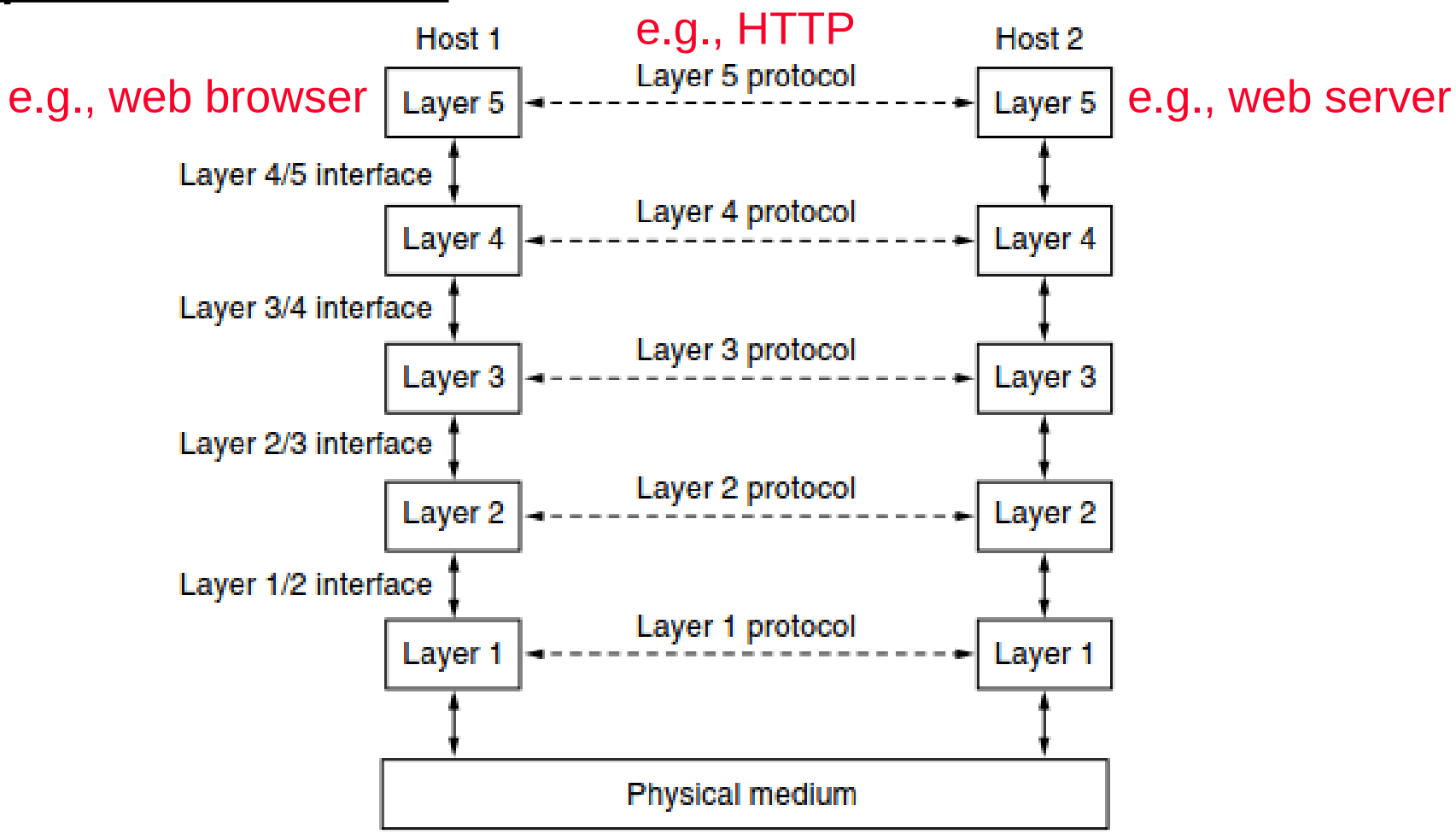
Layers and Protocol Stacks

- Layering is how we combine protocols
  - Higher level protocols build on services provided by lower level protocols
  - Peer layers communicate with each other
  - Protocols are horizontal, layers are vertical



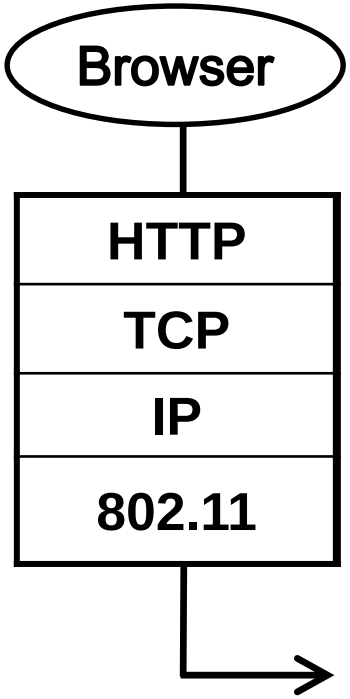
Layers and Protocol Stacks (2)

- Set of protocols in use is called a protocol stack

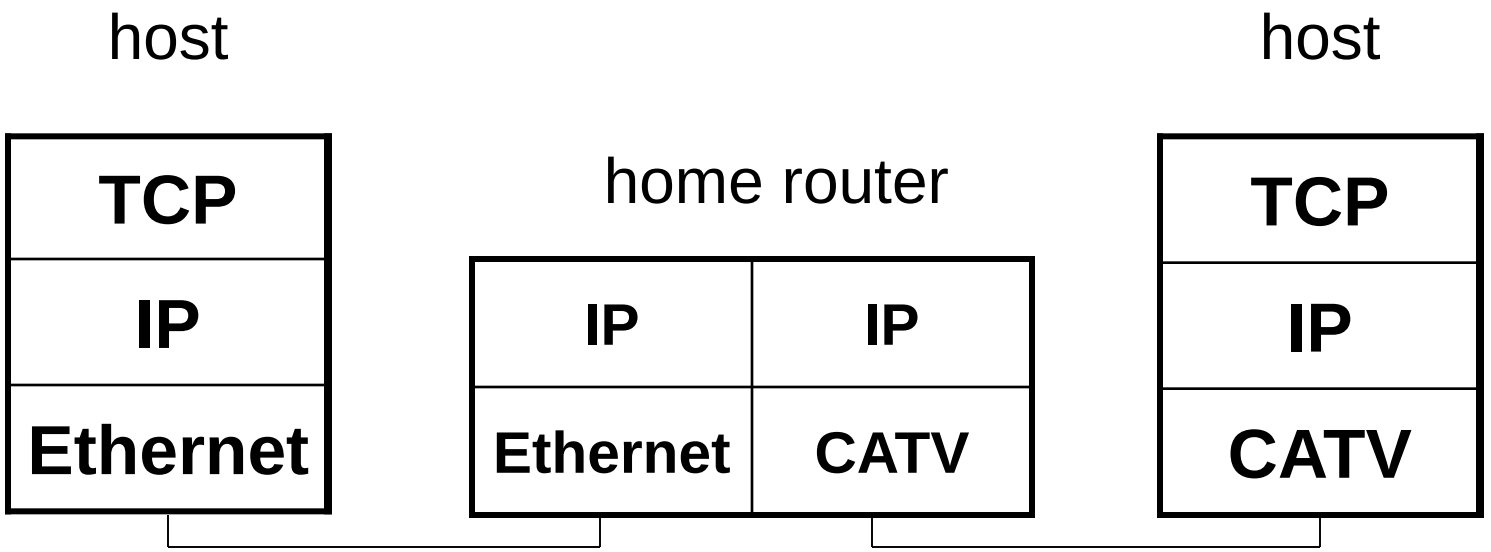


Protocols and Layers are everywhere

- Protocols you've probably heard of:
  - TCP, IP, 802.11, Ethernet, HTTP, SSL, DNS, … and many more
- An example protocol stack
  - Used by a web browser on a host that is wirelessly connected to the Internet

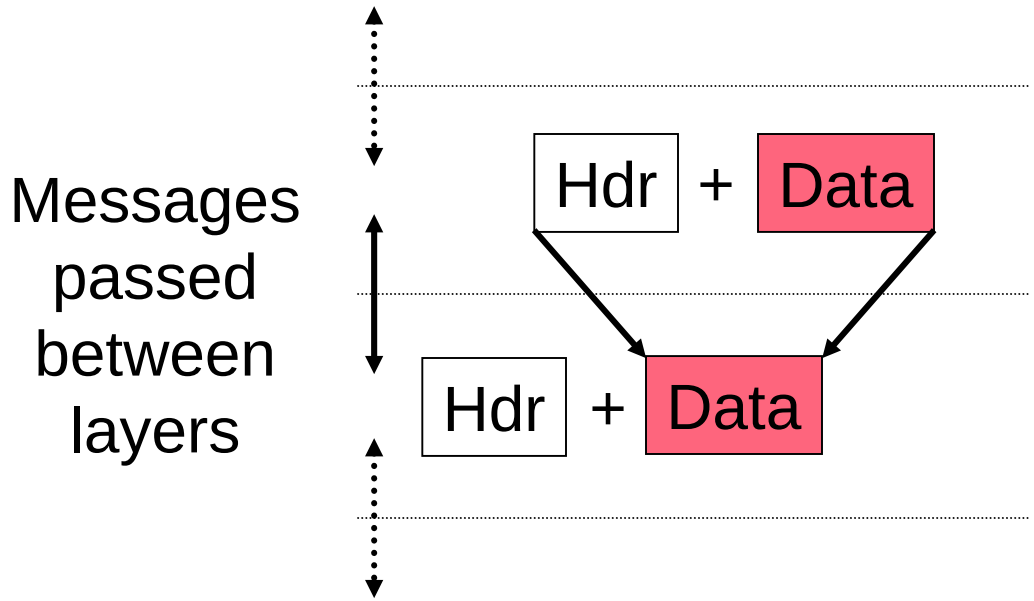


Example – Layering at work



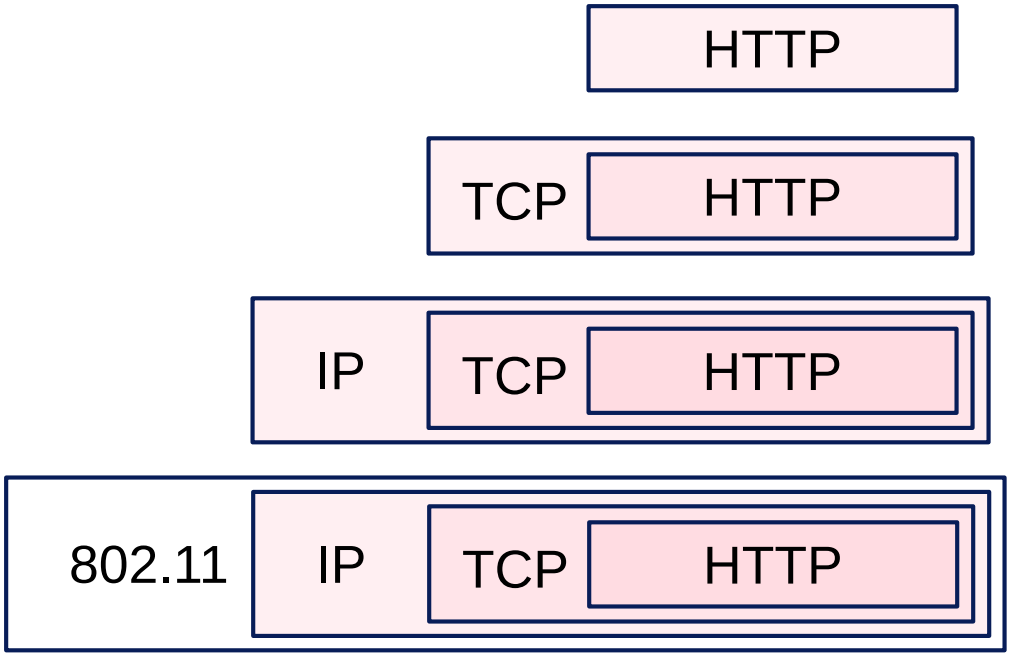
Encapsulation

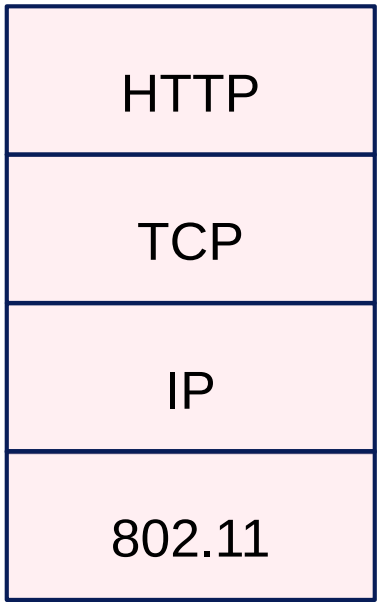
- Encapsulation is the mechanism used to effect protocol layering
  - Lower layer wraps higher layer content, adding its own information to make a new message for delivery
  - Like sending a letter in an envelope; postal service doesn't look inside



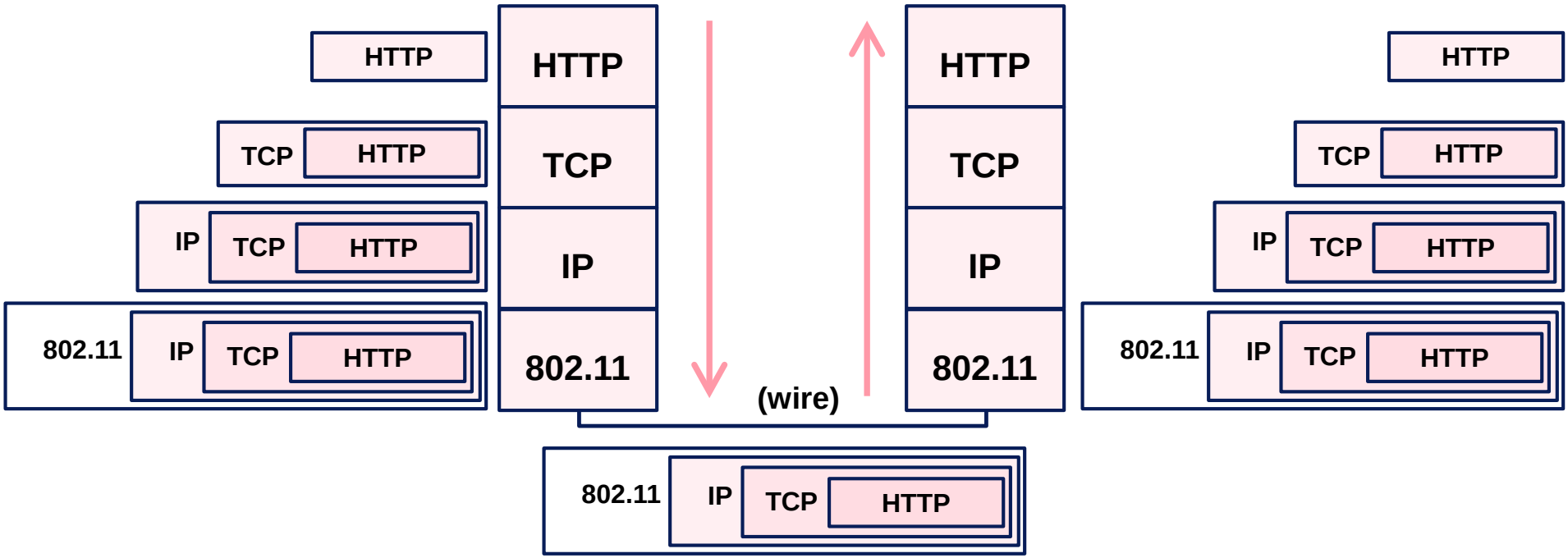
Encapsulation Example

- Lower layers are outermost



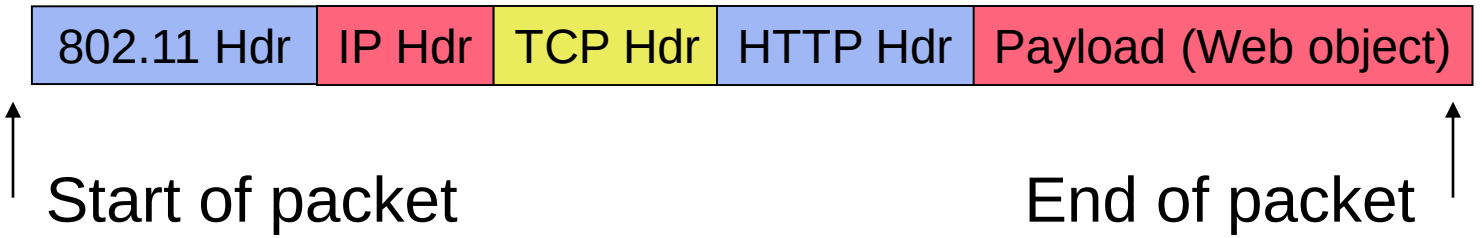


Encapsulation Example (2)



A Message on the Wire

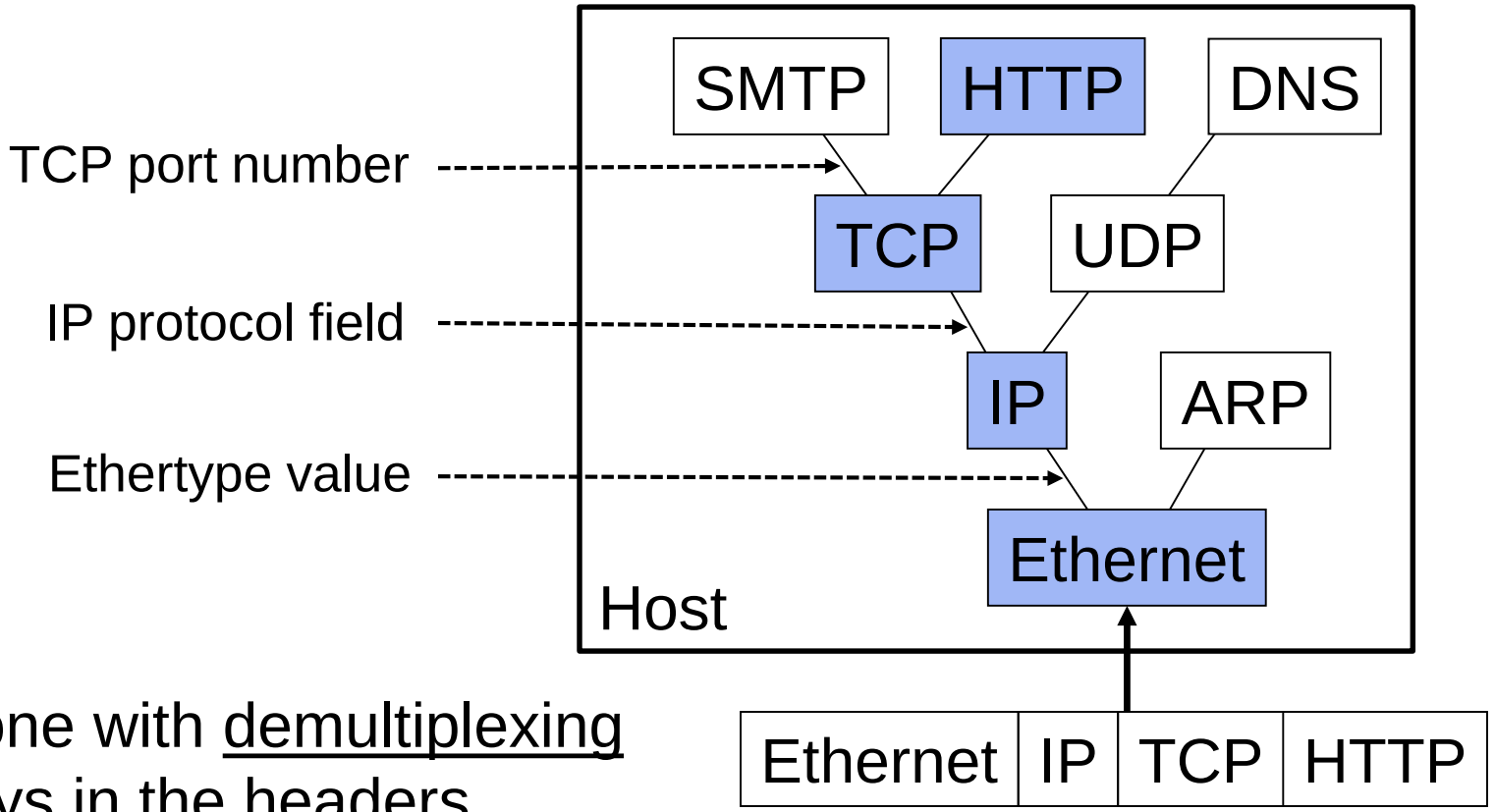
- Starts looking like an onion!



- This isn't entirely accurate
  - ignores segmentation and reassembly, 802.11 trailers, encrypt/compress contents, etc.
- But you can see that layering adds overhead

More Layering Mechanics

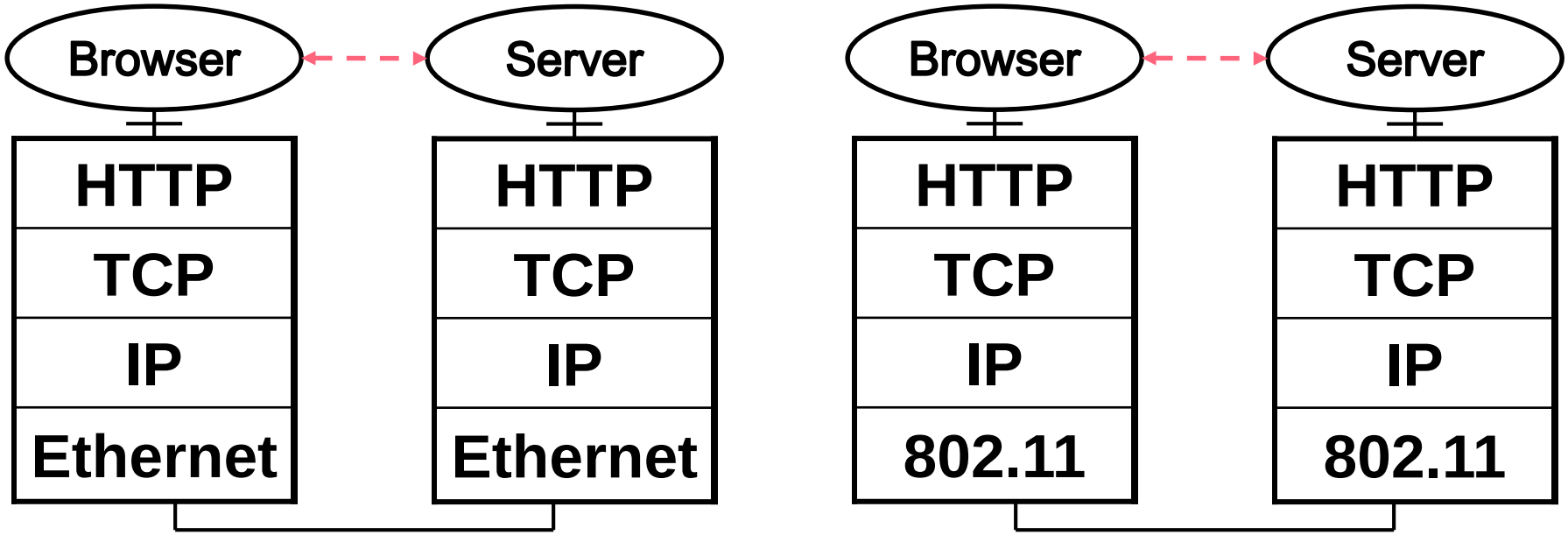
- Multiplexing and demultiplexing in a protocol graph
- Incoming message must be passed to the protocols that it uses. But how?



- Done with demultiplexing keys in the headers

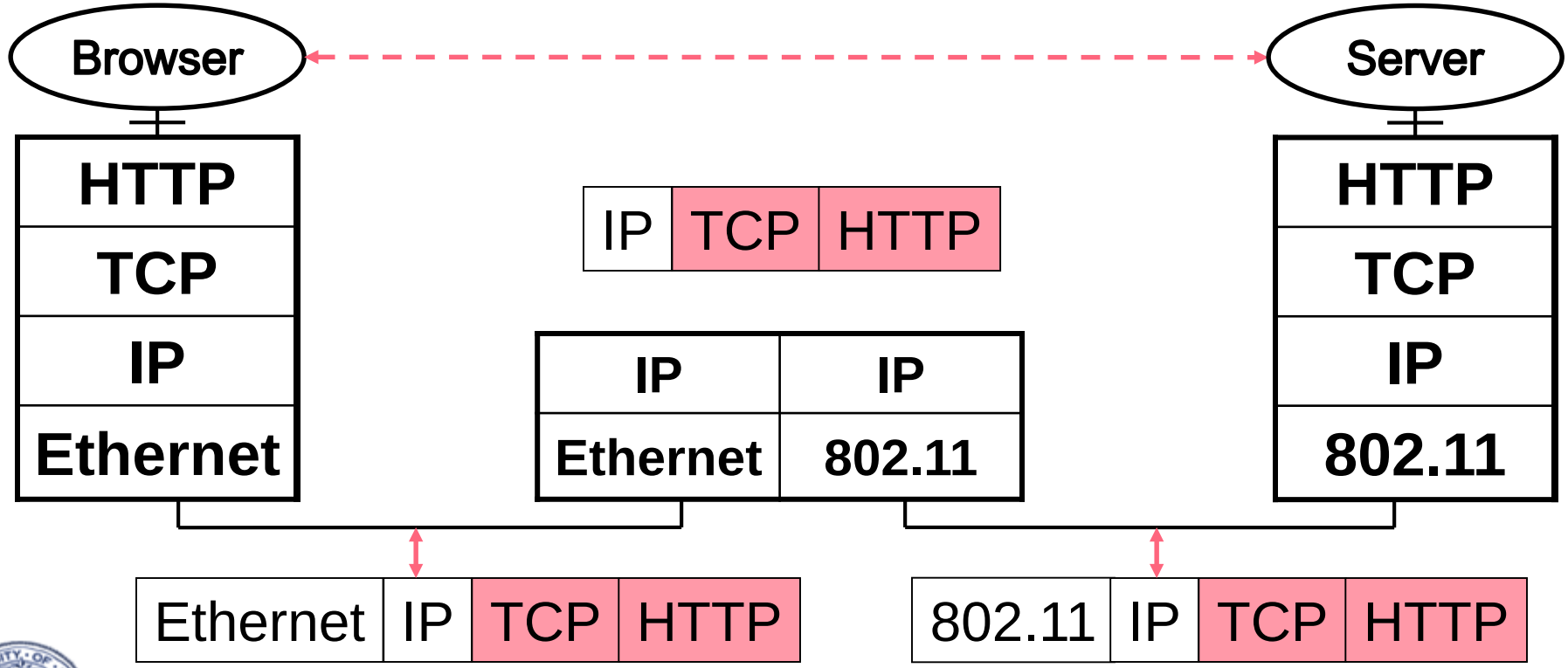
Advantage of Layering

- Information hiding and reuse
  - The browser does app/code does not know or care that you are transmitting over wired Ethernet or wireless 802.11 links!



Advantage of Layering (2)

- Use information hiding to connect different systems
  - We need a single layer that connects all things together
  - This is the job of IP



Disadvantage of Layering

- Adds overhead
  - But minor for long messages
- Hides information
  - App might care whether it is running over wired or wireless!

Reference Models

- A little guidance please…
- What functionality should we implement at which layer?
  - This is a key design question
  - Reference models provide frameworks that guide us

OSI "Seven Layer" Reference Model

- A principled, international standard, to connect systems
  - Influential, but not used in practice. (Woops!)

Application
Presentation
Session
Transport
Network
Link
Physical

- Provides functions needed by users
- Converts different representations
- Manage task dialogs
- Provides end-to-end delivery
- Sends packets over multiple links (routing)
- Framing, multiple access
- Symbol coding, modulation

Internet Reference Model

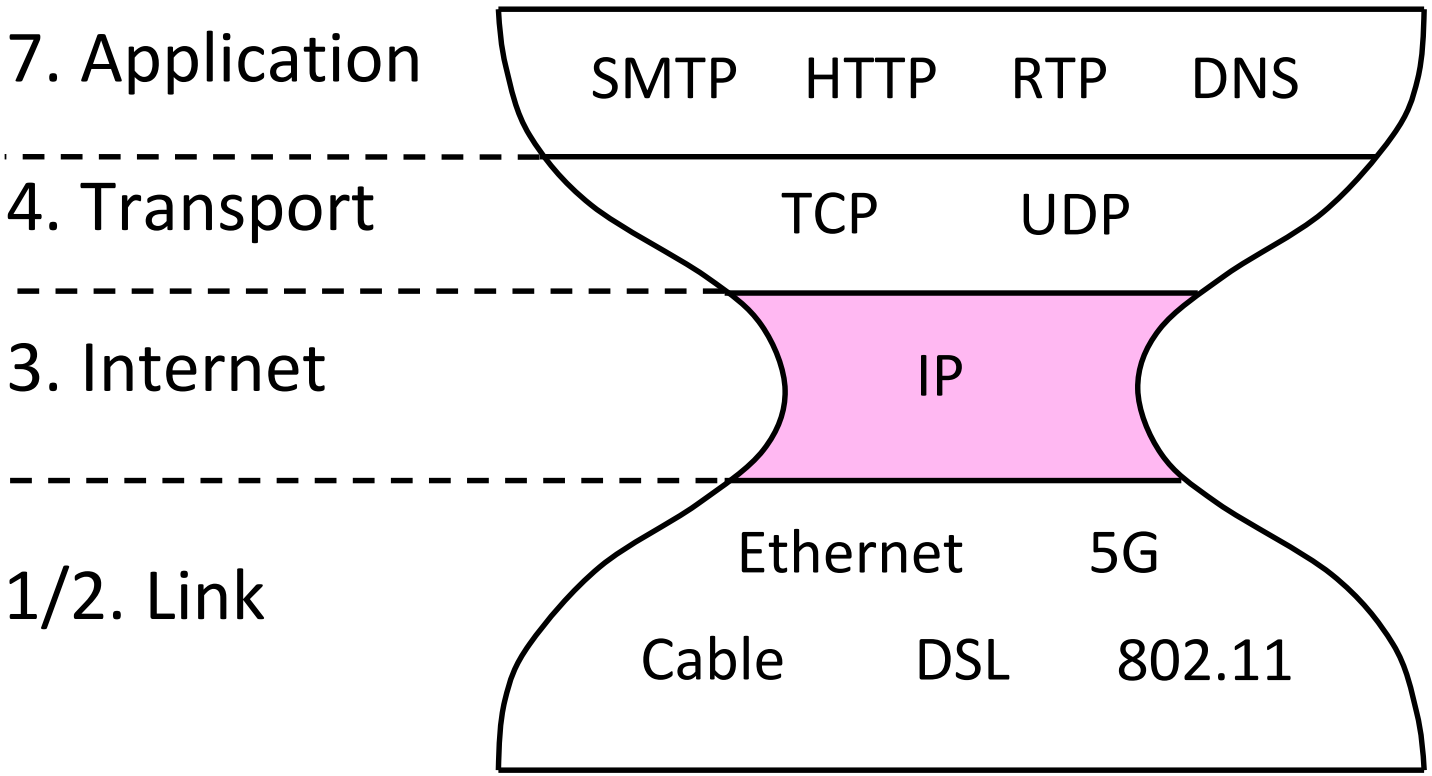
- A four layer model based on experience; omits some OSI layers and uses IP as the network layer.

7	**Application**	**Many (HTTP,SMTP)**
4	**Transport**	**TCP / UDP**
3	**Network**	**IP**
2,1	**Link**	**Many (Ethernet, …)**
OSI	Model	Protocols

- Programs that use network service
- Provides end-to-end data delivery
- Send packets over multiple networks
- Send frames over a link

Internet Reference Model (2)

- With examples of common protocols
- IP is the "narrow waist" of the Internet
  - Supports many different links below and apps above



Standards Bodies

- Where all the protocols come from?!
  - Focus on interoperability

Body	Area	Examples
ITU	Telecom	G.992, ADSL H.264, MPEG4
IEEE	Communications	802.3, Ethernet 802.11, WiFi
IETF	Internet	RFC 2616, HTTP/1.1 RFC 1034/35, DNS
W3C	Web	HTML5 standard CSS standard

- IETF (www.ietf.org) specifies Internet-related protocols
  - RFCs (Requests for Comments)
  - "We reject kings, presidents and voting. We believe in rough consensus and running code." – Dave Clark.

Layer-based Names

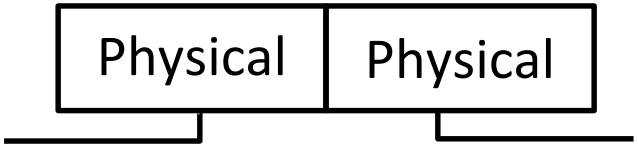
- For units of data

Layer	Unit of Data
Application	Message
Transport	Segment
Network	Packet
Link	Frame
Physical	Bit

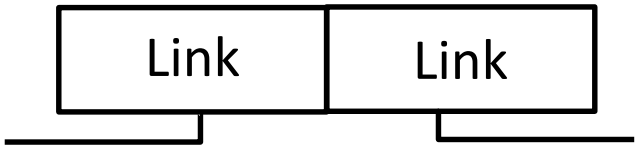
Layer-based Names (2)

- For devices in the network:

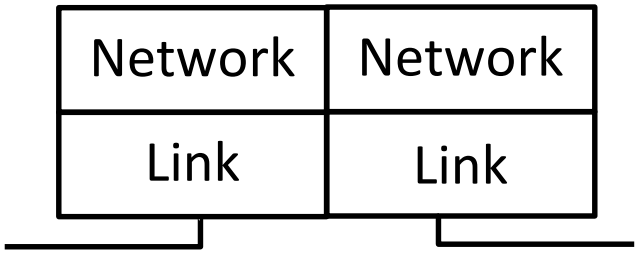
Hub, or repeater



Switch



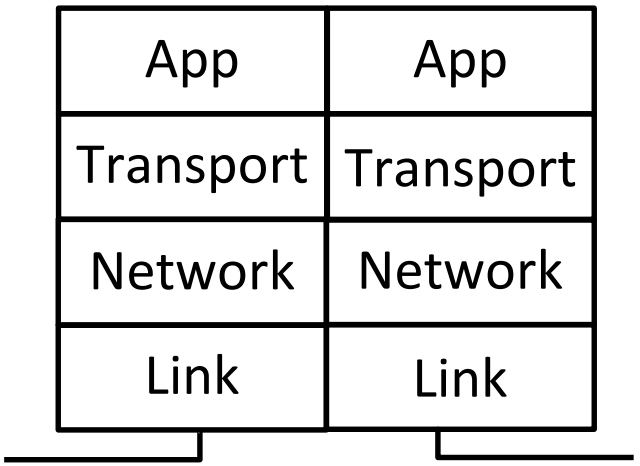
Router



Layer-based Names (3)

- For devices in the network:

Proxy or
middelbox
or gateway



But they all
look like this!

A Note About Layers

- They are guidelines, not strict
  - May have multiple protocols working together in one layer
  - May be difficult to assign a specific protocol to a layer

Functionality in Protocol Stacks

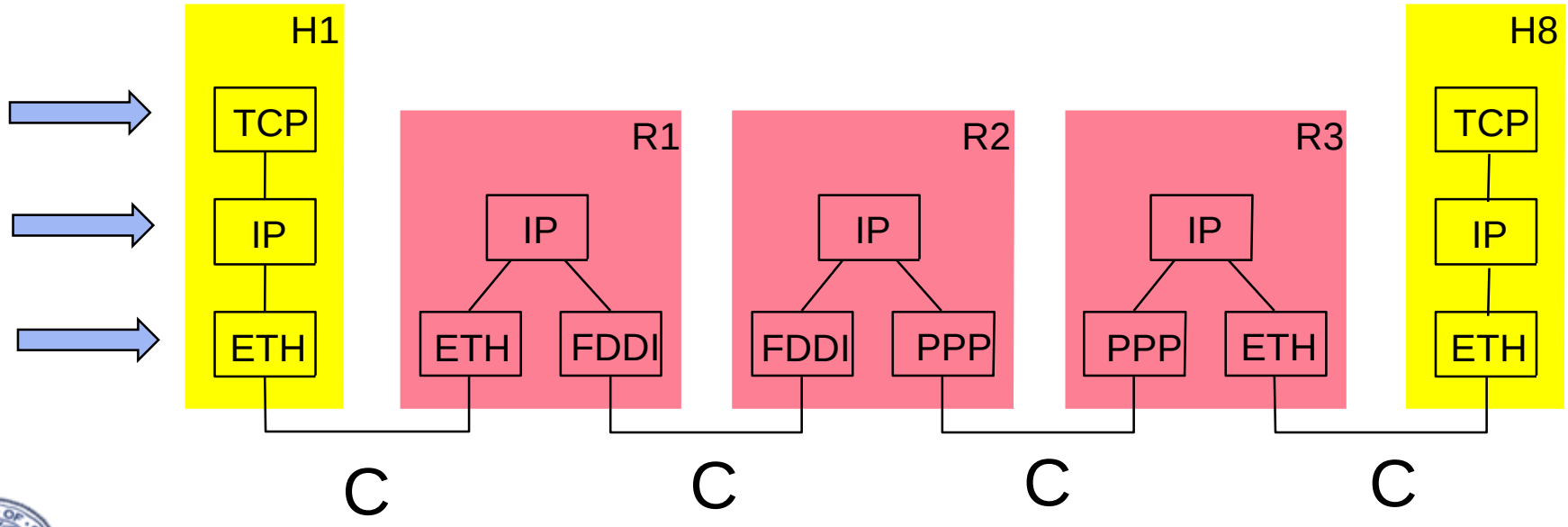
- Key Question: What functionality goes in which protocol?
- The “End to End Argument” (Reed, Saltzer, Clark, 1984):

Functionality should be implemented at a lower layer only if it can be correctly and completely implemented. (Sometimes an incomplete implementation can be useful as a performance optimization.)

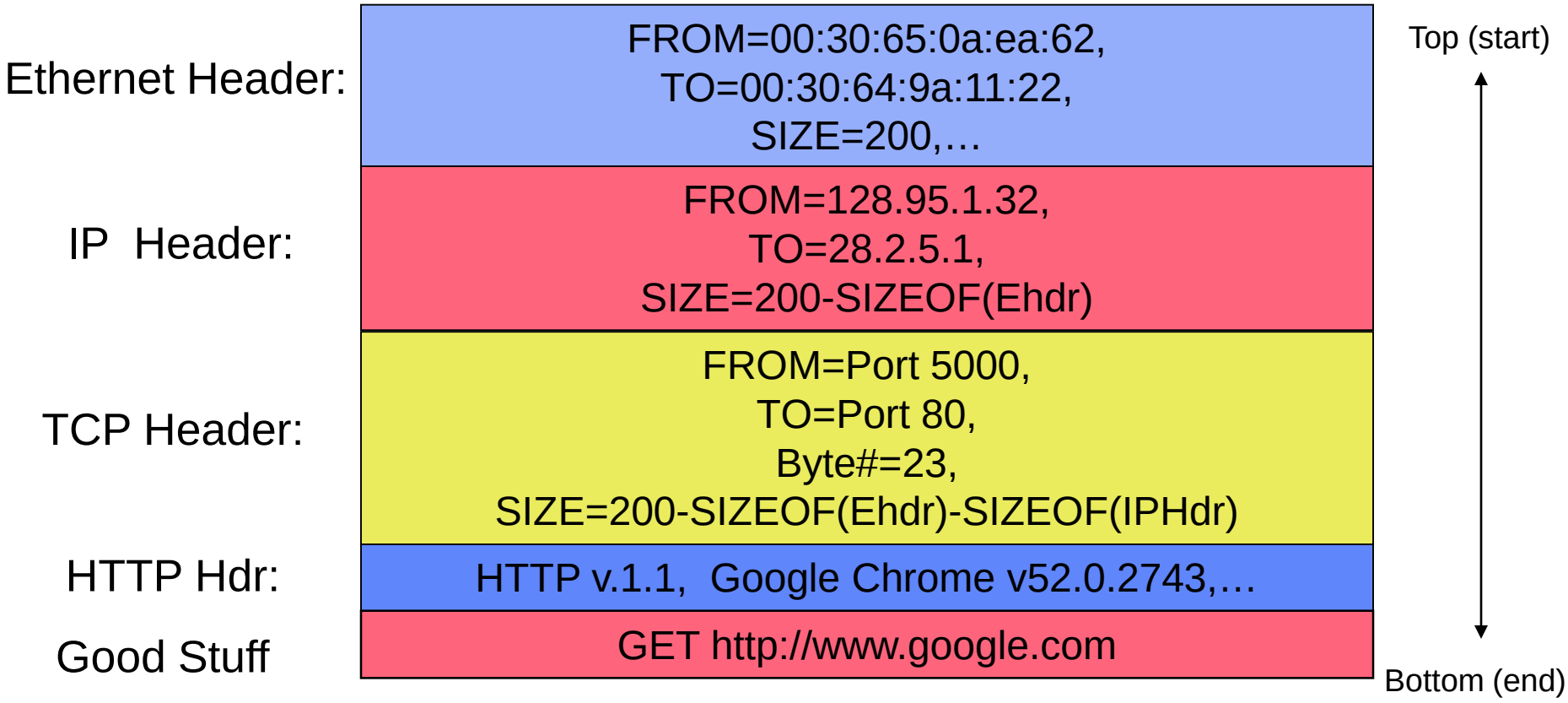
- Tends to push functions to the endpoints, which has aided the transparency and extensibility of the Internet.

Example: Reliability Functionality

- Basic Reliability function:
  - Encode E(M) = C
  - Decode D(C) = M, <good> || M', <bad>
- Which layer should have reliability function?



What's Inside a Packet



Key Concepts

- Protocol layers are the modularity that is used in networks to handle complexity
- The Internet/OSI models give us a roadmap of what kind of function belongs at what layer