EECS 489
Computer Networks

Architecture and Applications

Agenda

- Application Requirements
- Network Layers
- A study of current design and choices

Announcements

- Anyone still on the waitlist please contact me after class
- Make sure you have access to:
  - AWS
  - The shared AMI for project 1
- Complete the mininet tutorial
- Take a look at the Ed post on Tips for using mininet

How to Design a Network

- Challenges
  - Has many users
  - Offers diverse services
  - Mixes diverse technology
  - Components build by different companies
  - Diverse ownership
  - Evolve over time

Review – OSI layers

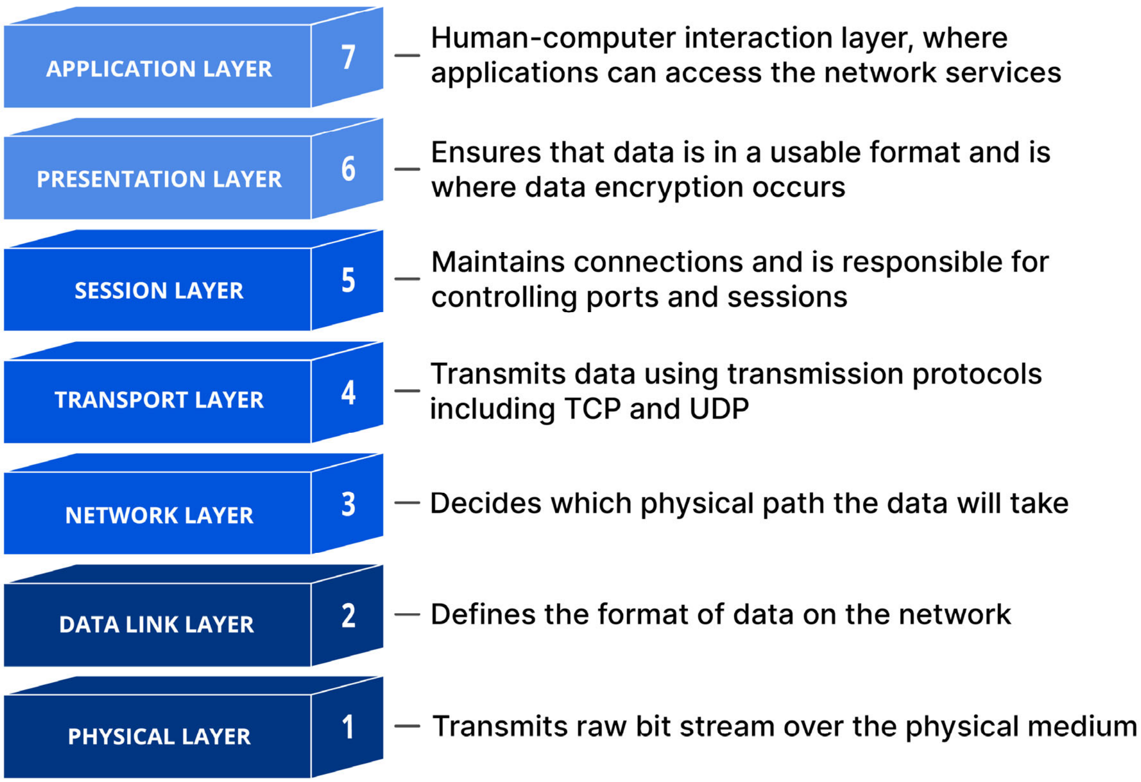
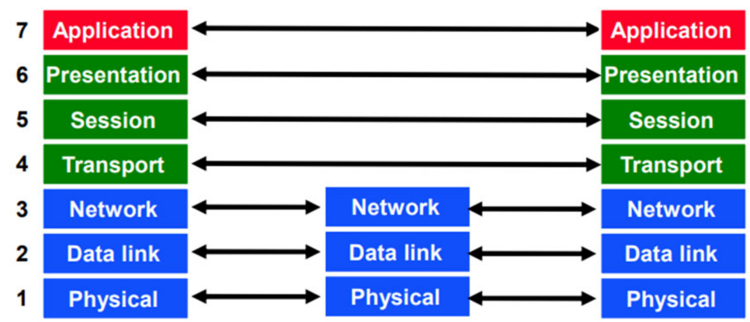


Image Source: https://www.cloudflare.com/learning/ddos/glossary/open-systems-interconnection-model-osi/

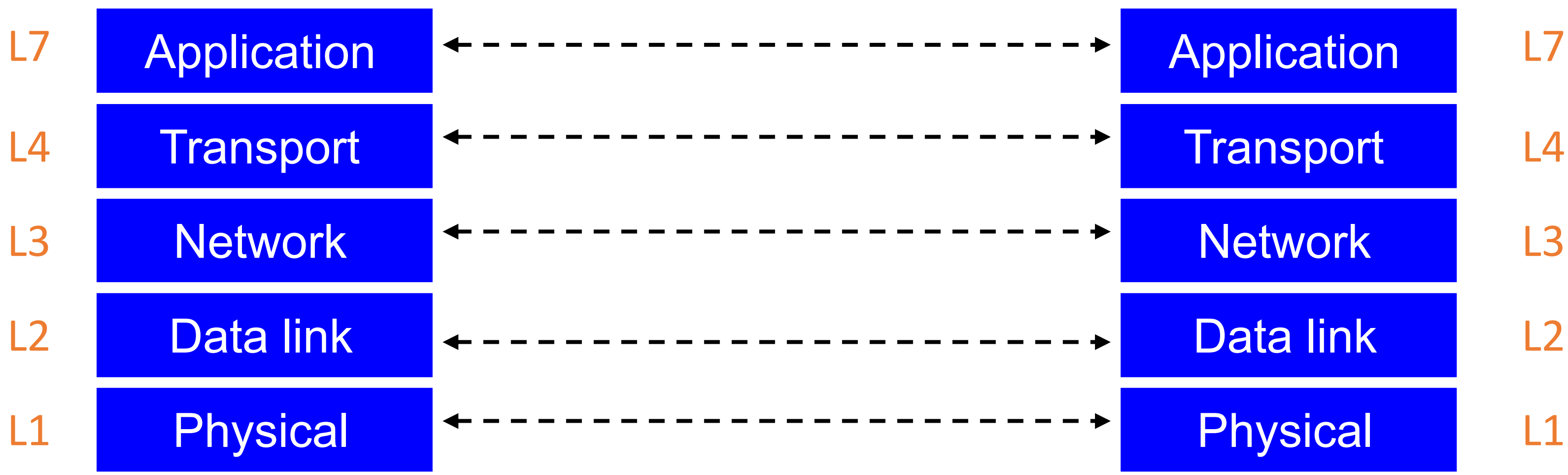
OSI Communications

- Only Horizontal and Vertical Communications
- Each Layer offers a service to the higher layer using the services of the lower layer
- “Peer” layers on different systems communicate via a protocol

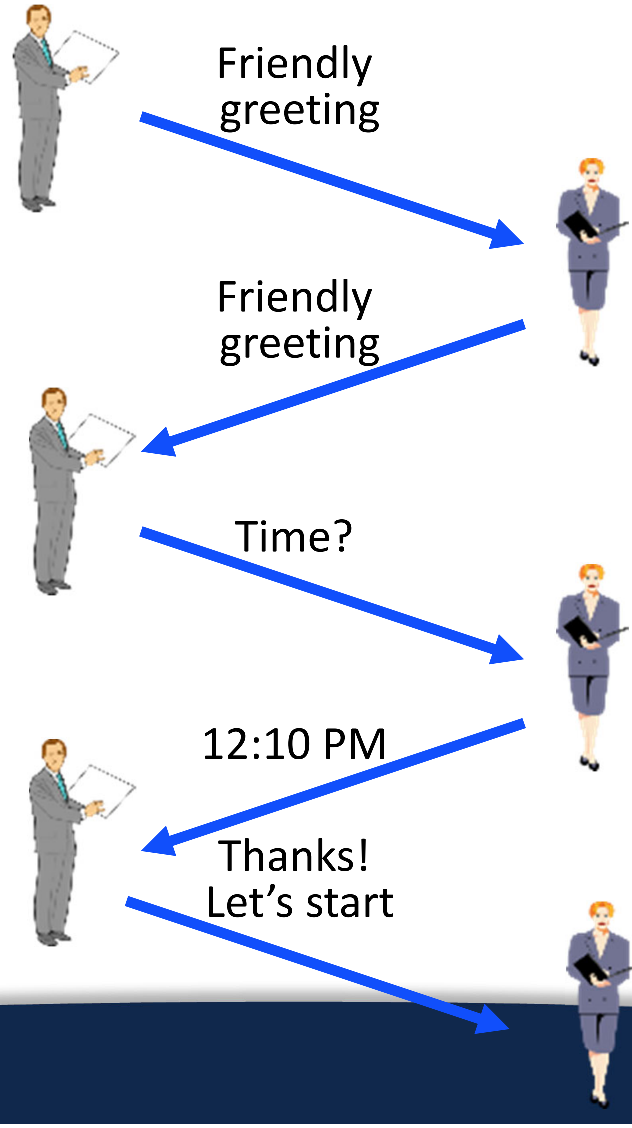


Layers and protocols

- Only Horizontal and Vertical Communications
- Each Layer offers a service to the higher layer using the services of the lower layer
- Communication between peer layers on different systems is defined by protocols



What is a Protocol?



What is a Protocol?

- An agreement between parties (in the same later) on how to communicate
- Defines the syntax of communication
  - Header → instructions on how to process payload
  - Each protocol defines the format of its headers
    - e.g., “the first 32 bits carry the destination address”

Header	Payload

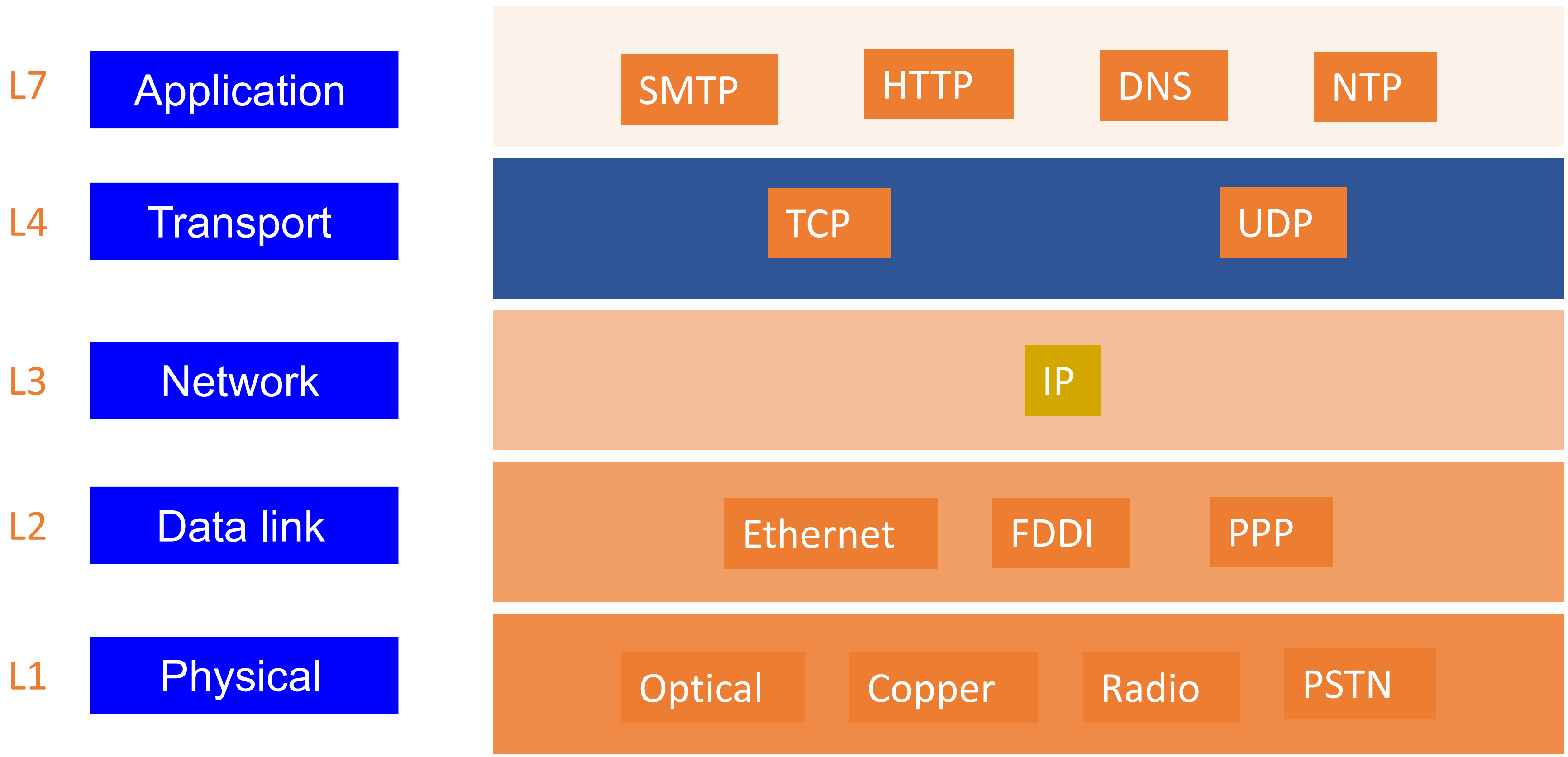
What is a Protocol?

- An agreement between parties on how to communicate
- Defines the syntax of communication
- And semantics
  - "First a hello, then a request…"
  - We will study many protocols later in the semester
- Protocols exist at many levels, hardware, and software
  - Defined by standards bodies like IETF, IEEE, ITU

Protocols at different layers

L7	Application	SMTP, HTTP, DNS, NTP
L4	Transport	TCP, UDP
L3	Network	IP
L2	Data link	Ethernet, FDDI, PPP
L1	Physical	Optical, Copper, Radio, PSTN

ONE network layer protocol



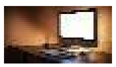
Starting out

- A Computer is switched on

2C:54:91:88:C9:E3

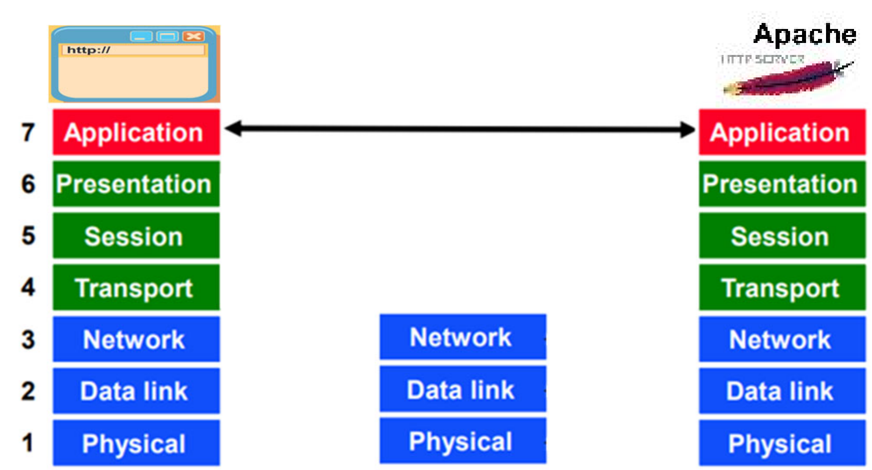
- We need a scalable way to:
  - Get an IP address
  - Get IP address of the default gateway
- DHCP (Lecture 19 – Switched Networks)

The first packet

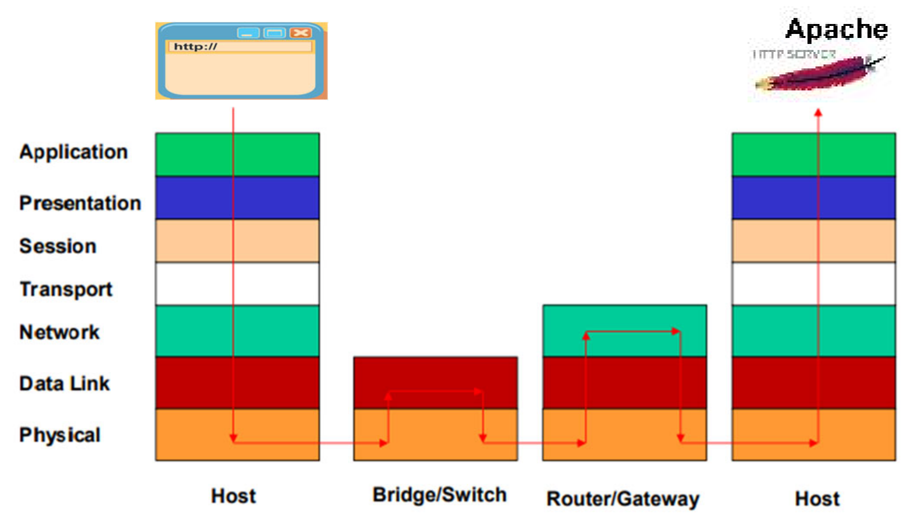


192.168.1.101
2C:54:91:88:C9:E3

- Who is communicating?



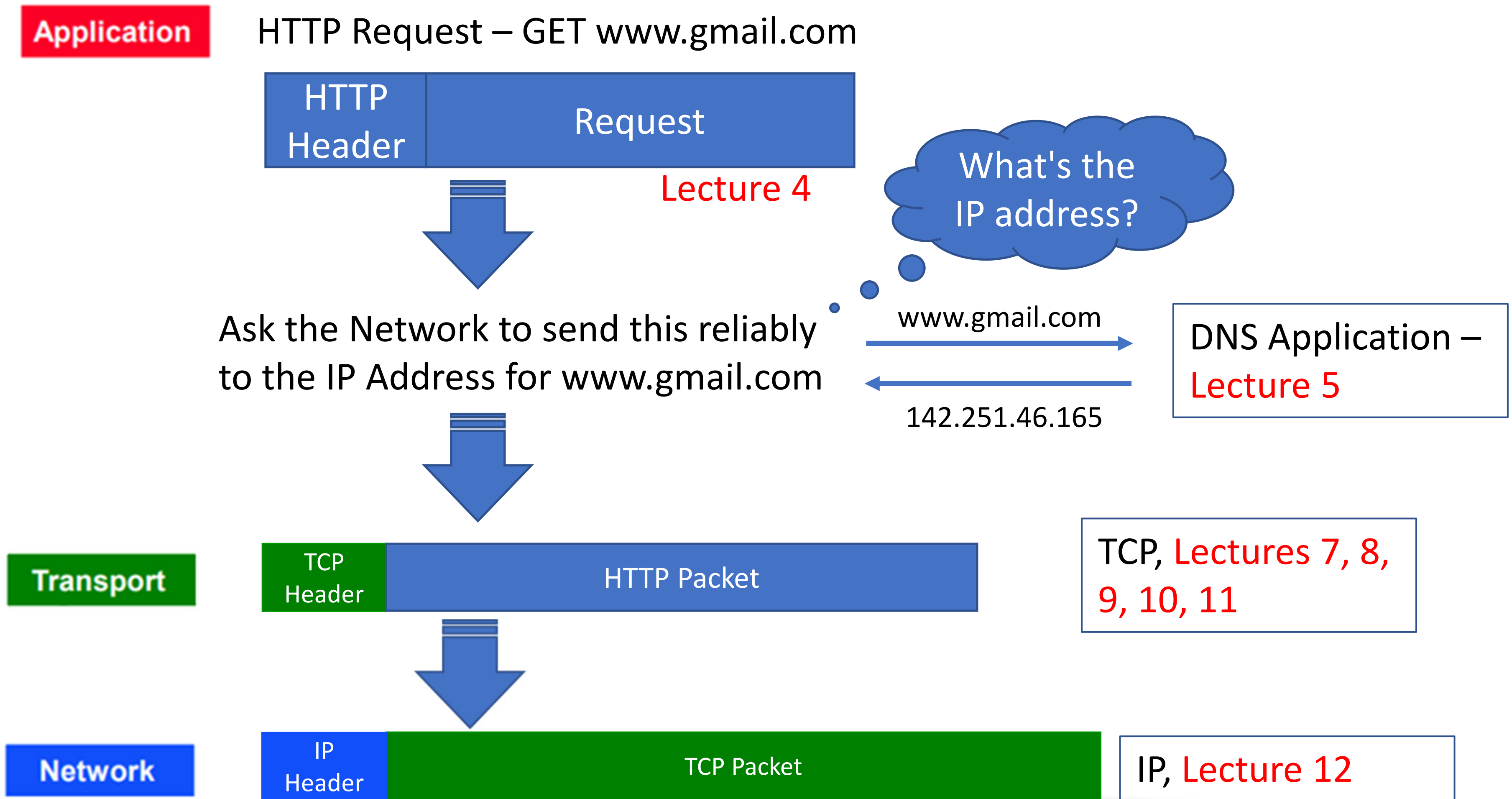
Actual Transfer of Data



Application Layer protocols

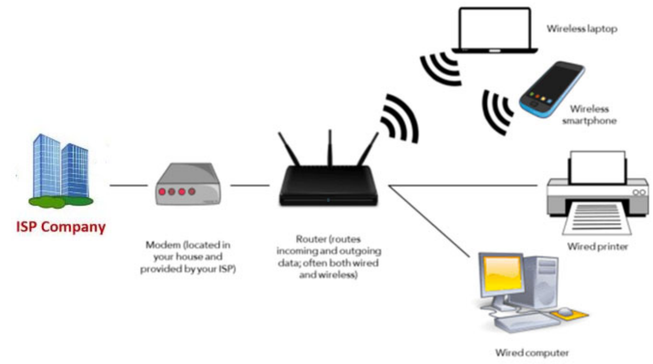
- Protocol between Web browser and Web server
  - HTTP – Lecture 4
- Other protocols?
  - FTP
  - SMTP
  - DNS
  - SSH
  - IMAP
  - SNMP
  - XMPP

Packet Generation

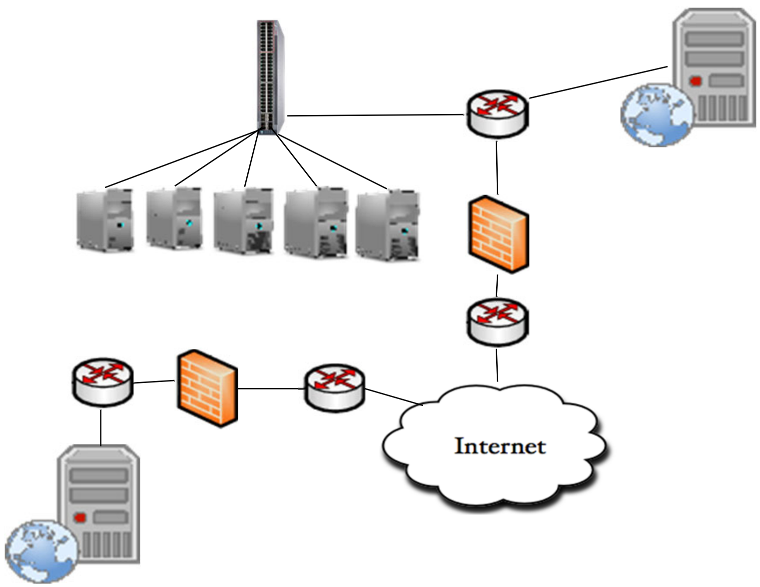


Packet Generation - continued

- How do we connect to the internet?

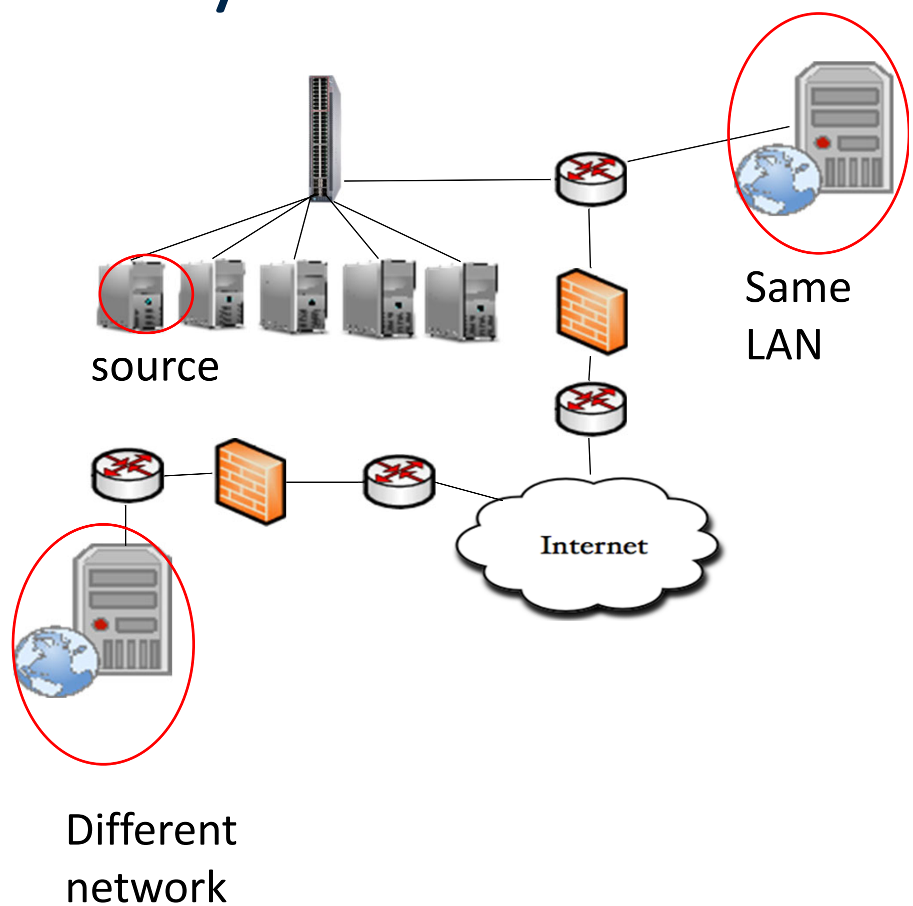


- More interesting case.

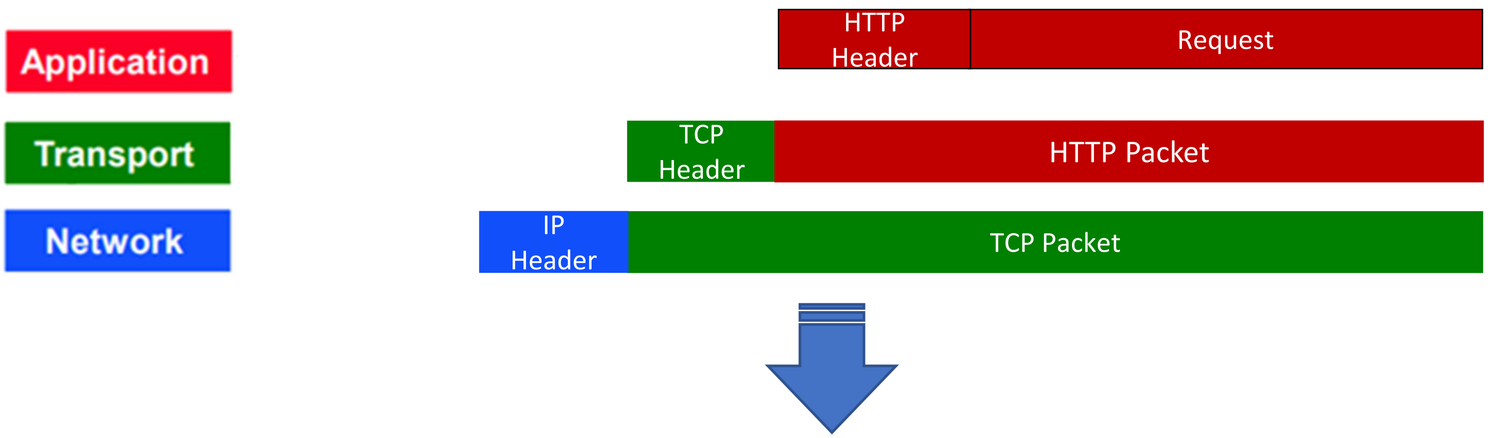


Packet Generation – IP Layer

- Where is the IP Address?
  - Same LAN
  - Different Network
- IP Addressing (Lecture 16)

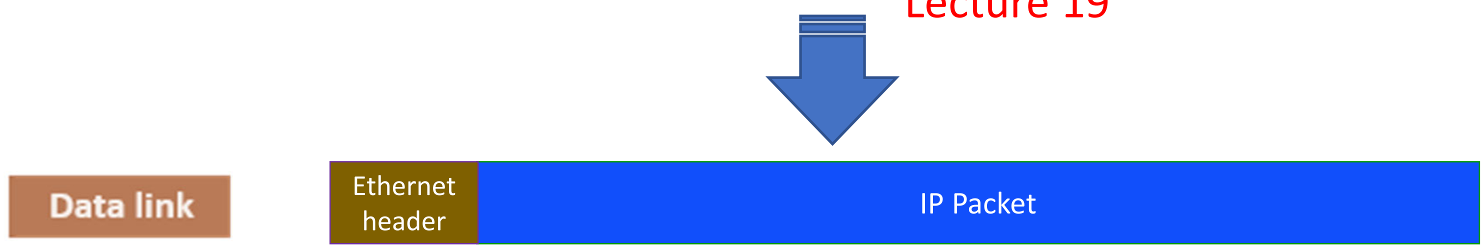


Data Link Layer

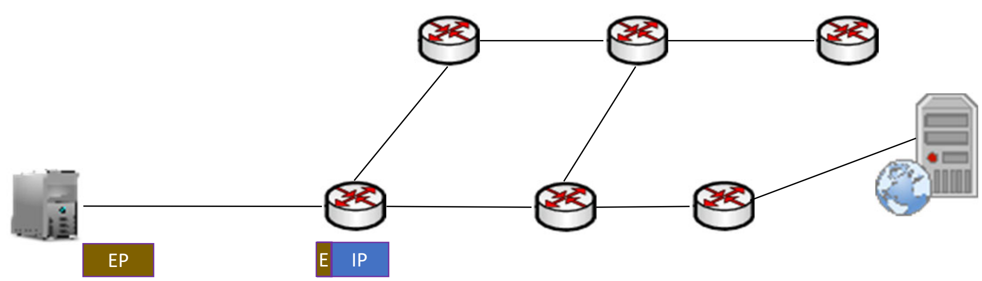


ARP to determine which hardware address to use

Lecture 19



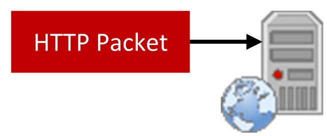
What happens in the network



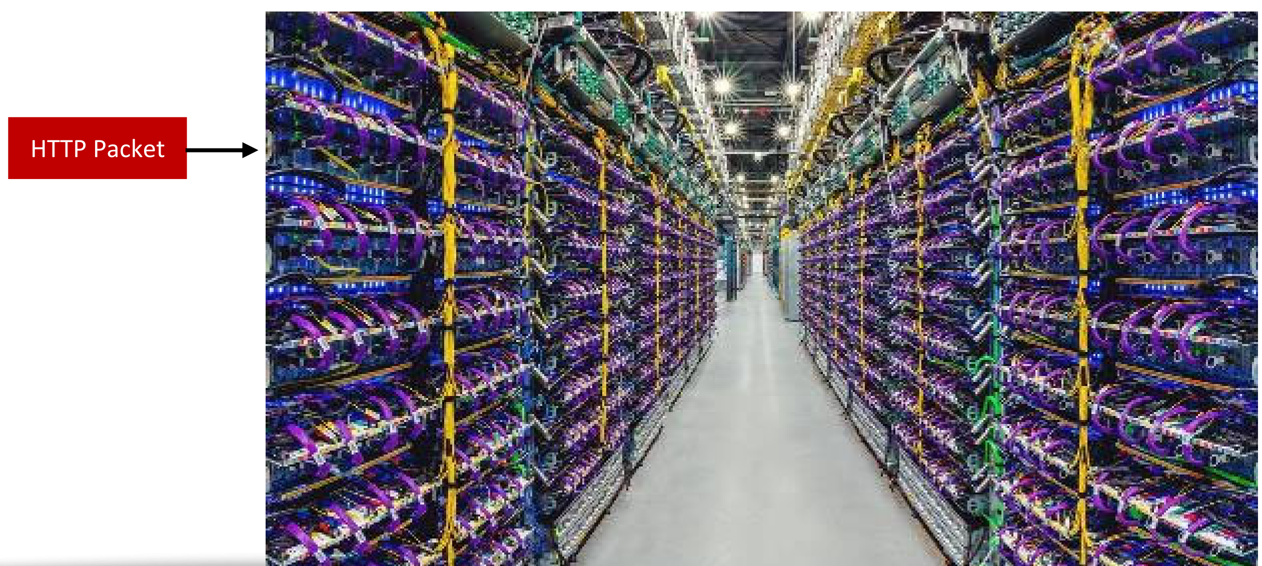
Routing and Forwarding
Lectures 14, 15, 16, and 17

At the destination

- What happens at the server?
- You are accessing gmail.com
- Option 1

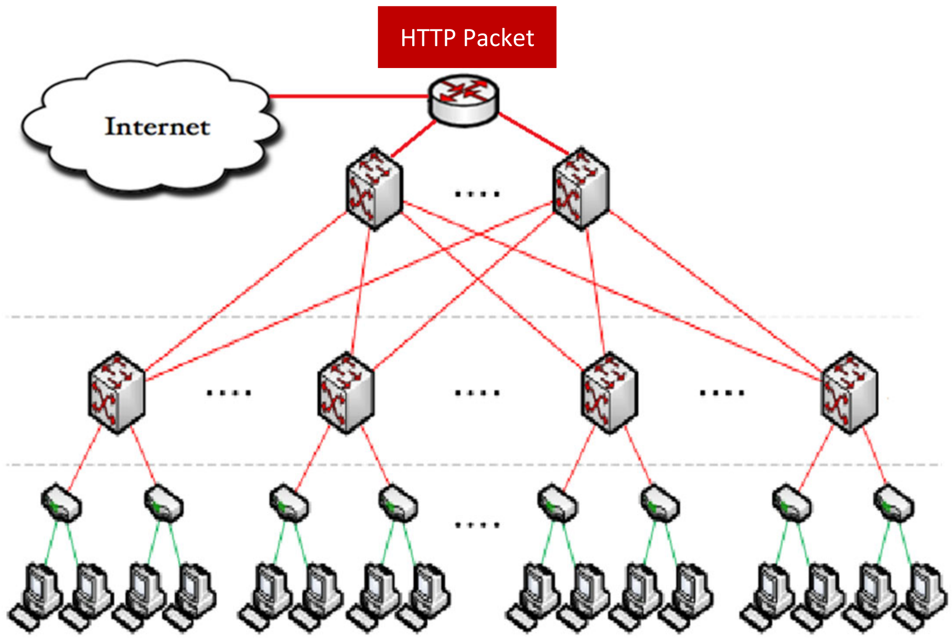


- Option 2



Data Centers – Lectures 6 and 23

- Tasks
  - Is the user logged in
  - Get user information
  - Get user emails
  - Get user calendar
  - Is Free account?
    - Get ads for this user



Optimizations

- What can we optimize?
  - Routes
  - Content
  - URL and IP Address pairs
  - IP and MAC address pairs

What gets implemented at the end systems?

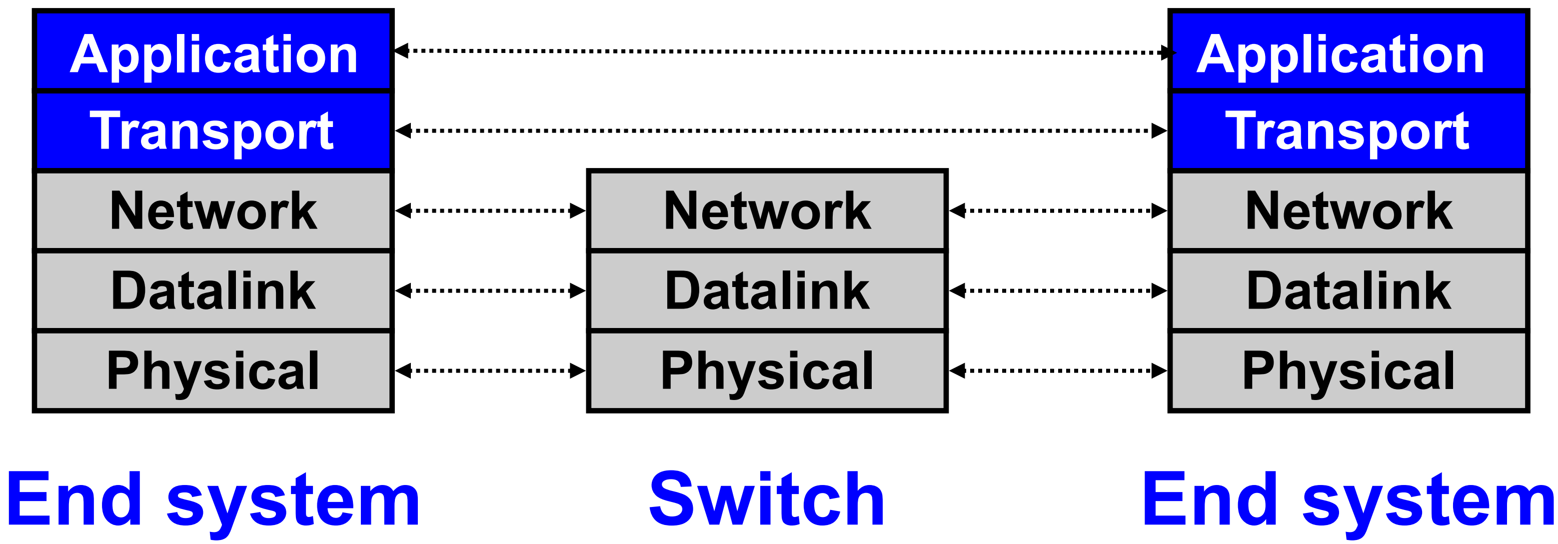
- Bits arrive on wire, must make it up to application
- Therefore, all layers must exist at host!

What gets implemented in the network?

- Bits arrive on wire → physical layer (L1)
- Packets must be delivered across links and local networks → datalink layer (L2)
- Packets must be delivered between networks for global delivery → network layer (L3)
- The network does not support reliable delivery
  - Transport layer (and above) not supported

Simple Diagram

- Lower three layers implemented everywhere
- Top two layers implemented only at hosts

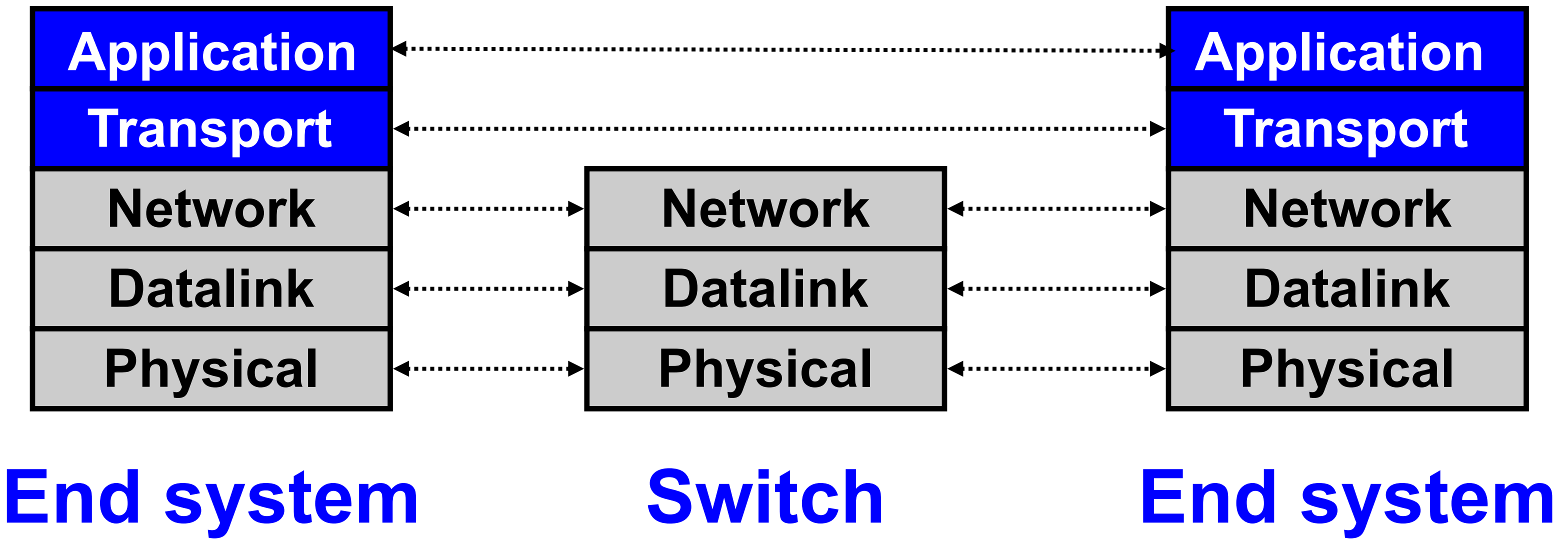


A closer look: End system

- Application
  - Web server, browser, mail, game
- Transport and network layer
  - typically part of the operating system
- Datalink and physical layer
  - hardware/firmware/drivers

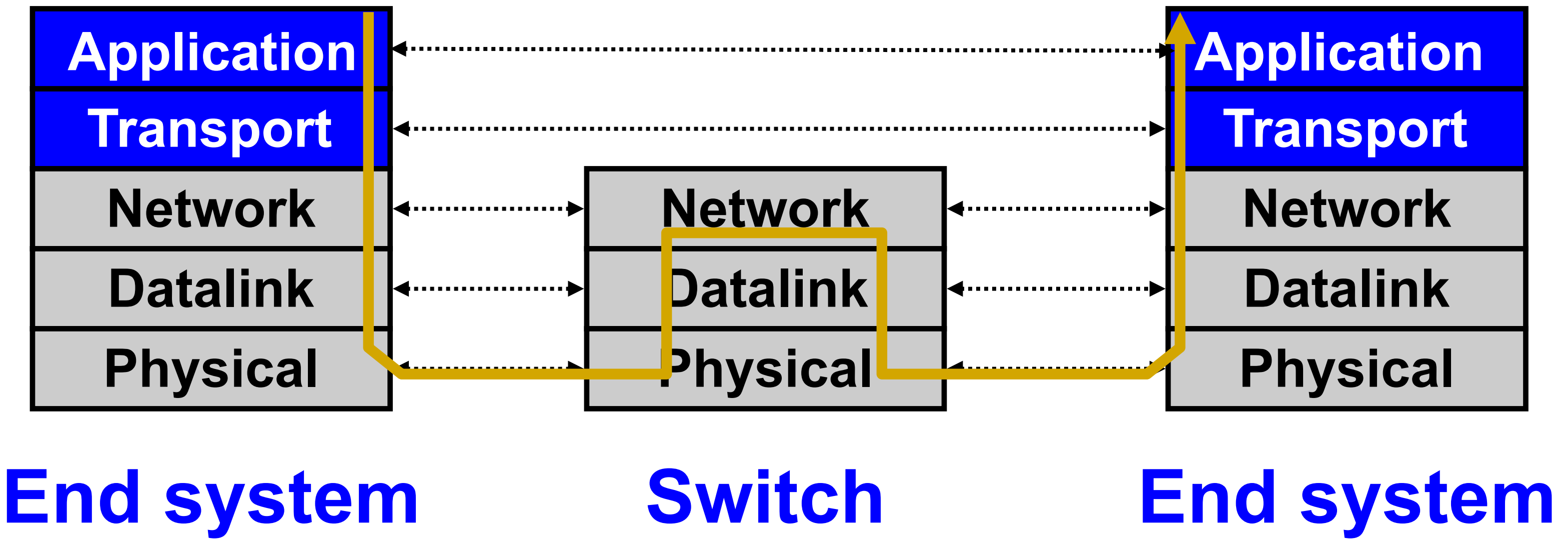
Logical communication

- A layer interact with its peers corresponding layer

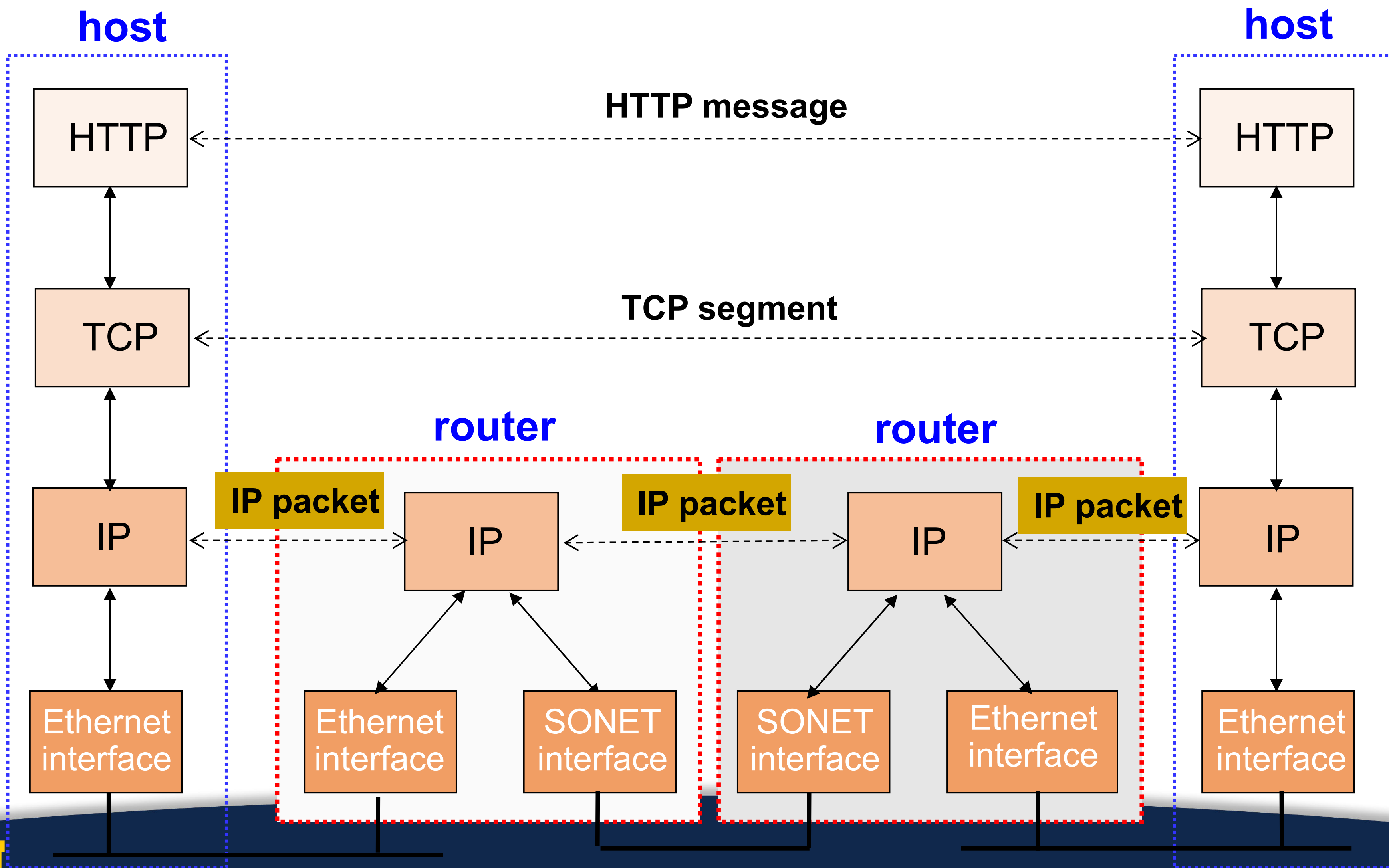


Physical communication

- Communication goes down to physical network
- Then up to relevant layer



A protocol-centric diagram

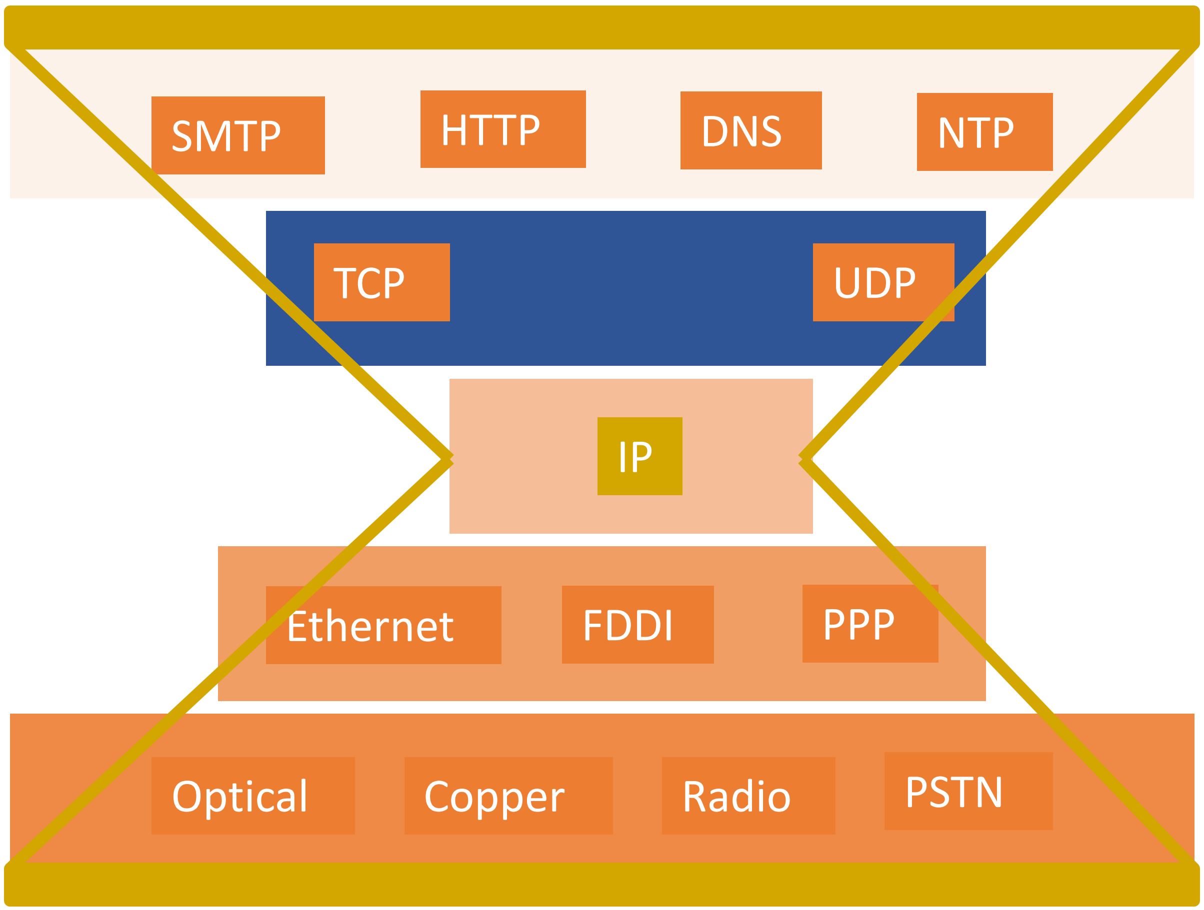


Pros and cons of layering

- Why layers?
  - Reduce complexity
  - Improve flexibility

- Why not?
  - Higher overheads
  - Cross-layer information often useful

IP is the narrow waist of the layering hourglass



Implications of hourglass

- Single network-layer protocol (IP)
- Allows arbitrary networks to interoperate
  - Any network that supports IP can exchange packets
- Decouples applications from low-level networking technologies
  - Applications function on all networks
- Supports simultaneous innovations above and below IP
- But changing IP itself is hard (e.g., IPv4 → IPv6)

Placing network functionality

- End-to-end arguments by Saltzer, Reed, and Clark
  - Dumb network and smart end systems
  - Functions that can be *completely* and *correctly* implemented *only* with the knowledge of application end host, should not be pushed into the network
  - Sometimes necessary to break this for performance and policy optimizations
  - Fate sharing: fail together or don't fail at all

Internet Architecture

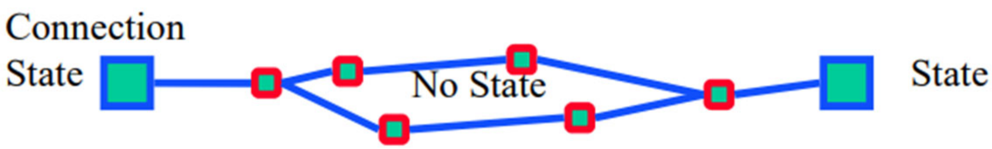
- Fundamental goal:
  - Effective network interconnection
- Goals, in order of priority:
  1. Continue despite loss of networks or gateways
  2. Support multiple types of communication service
  3. Accommodate a variety of networks
  4. Permit distributed management of Internet resources
  5. Cost effective
  6. Host attachment should be easy
  7. Resource accountability

Survivability

- If network disrupted and reconfigured
  - Communicating entities should not care!
  - No higher-level state reconfiguration
  - Ergo, transport interface only knows “working” and “not working.” Not working == complete partition.
- How to achieve such reliability?
  - Where can communication state be stored?

	Network	Host
Failure Handling	Replication	“Fate Sharing”
Network Engineering	Tough	Simple
Switches	Maintain State	Stateless
Host Trust	Less	More

Fate Sharing



- Lose state information for an entity if (and only if?) the entity itself is lost.
  - Example:
    - OK to lose TCP state if one endpoint crashes -
      - NOT okay to lose if an intermediate router reboots
    - Is this still true in today's network?
      - NATs and firewalls (More about those later)
- Survivability compromise: Heterogenous network -> less information available to end hosts and Internet level recovery mechanisms

Quiz 2 – open until 8pm

- https://forms.gle/fqMfo2fDFHkBkADz7

Types of Service

- Recall TCP vs. UDP
  - Elastic (in terms of delay) apps that need reliability: remote login or email
  - Inelastic, loss-tolerant apps: real-time voice or video
  - Others in between, or with stronger requirements
  - Biggest cause of delay variation: reliable delivery
    - Today's net: ~100ms RTT
    - Reliable delivery can add seconds.
- Original Internet model: "TCP/IP" one layer
  - First app was remote login…
  - But then came debugging, voice, etc.
  - These differences caused the layer split, added UDP
- No QoS support assumed from below
  - In fact, some underlying nets only supported reliable delivery
    - Made Internet datagram service less useful!
  - Hard to implement without network support
  - QoS is an ongoing debate…

Varieties of Networks

- Different types of networks
  - Interconnect the ARPANET, X.25 networks, LANs, satellite networks, packet networks, serial links…
- Minimum set of assumptions for underlying net
  - Minimum packet size
  - Reasonable delivery odds, but not 100%
  - Some form of addressing unless point to point
- Important non-assumptions:
  - Perfect reliability
  - Broadcast, multicast
  - Priority handling of traffic
  - Internal knowledge of delays, speeds, failures, etc.
- Much engineering then only has to be done once

Other goals

- Management
  - Today's Internet is decentralized - BGP
  - Very coarse tools. Still in the "assembly language" stage
- Cost effectiveness
  - Economies of scale won out
  - Internet cheaper than most dedicated networks
  - Packet overhead less important by the year
- Attaching a host
  - Not awful; DHCP and related autoconfiguration technologies helping. A ways to go, but the path is there

Accountability

- Huge problem.
- Accounting
  - Billing? (mostly flat-rate. But phones are moving that way too - people like it!)
  - Inter-provider payments
    - Hornet's nest. Complicated. Political. Hard.
- Accountability and security
  - Huge problem.
  - Worms, viruses, etc.
    - Partly a host problem. But hosts very trusted.
  - Authentication
    - Purely optional. Many philosophical issues of privacy vs. security.

Where do we go from here

- We start top-down and look at design choices, trade-offs, etc. at each layer