Acknowledgement: The source PowerPoint slides are public available, provided by Authors (JFK/KWR). They are revised for CS422@Purdue.

Chapter 1 Introduction

A note on the use of these PowerPoint slides:

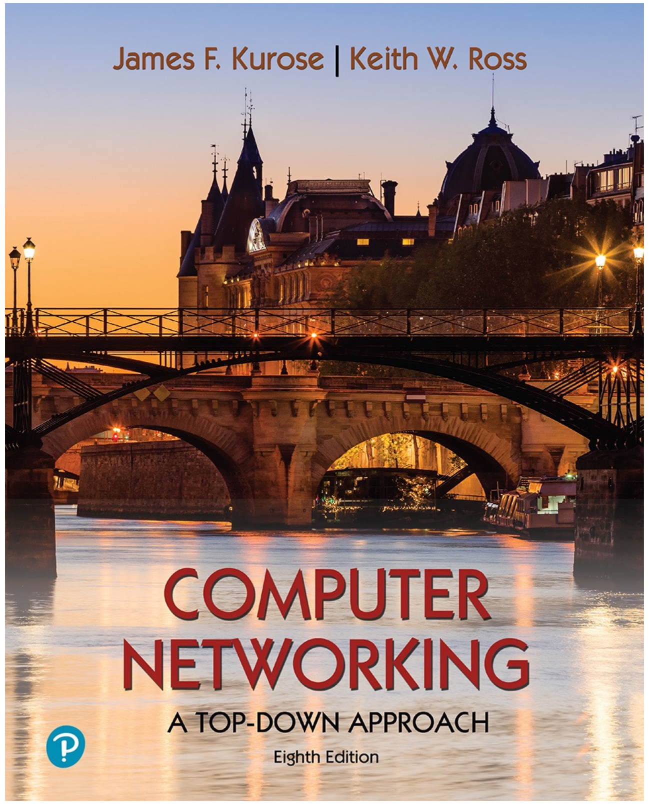
We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

For a revision history, see the slide note for this page.

Thanks and enjoy! JFK/KWR

All material copyright 1996-2020
J.F Kurose and K.W. Ross, All Rights Reserved



Computer Networking: A Top-Down Approach
8th edition
Jim Kurose, Keith Ross
Pearson, 2020

Chapter 1: introduction

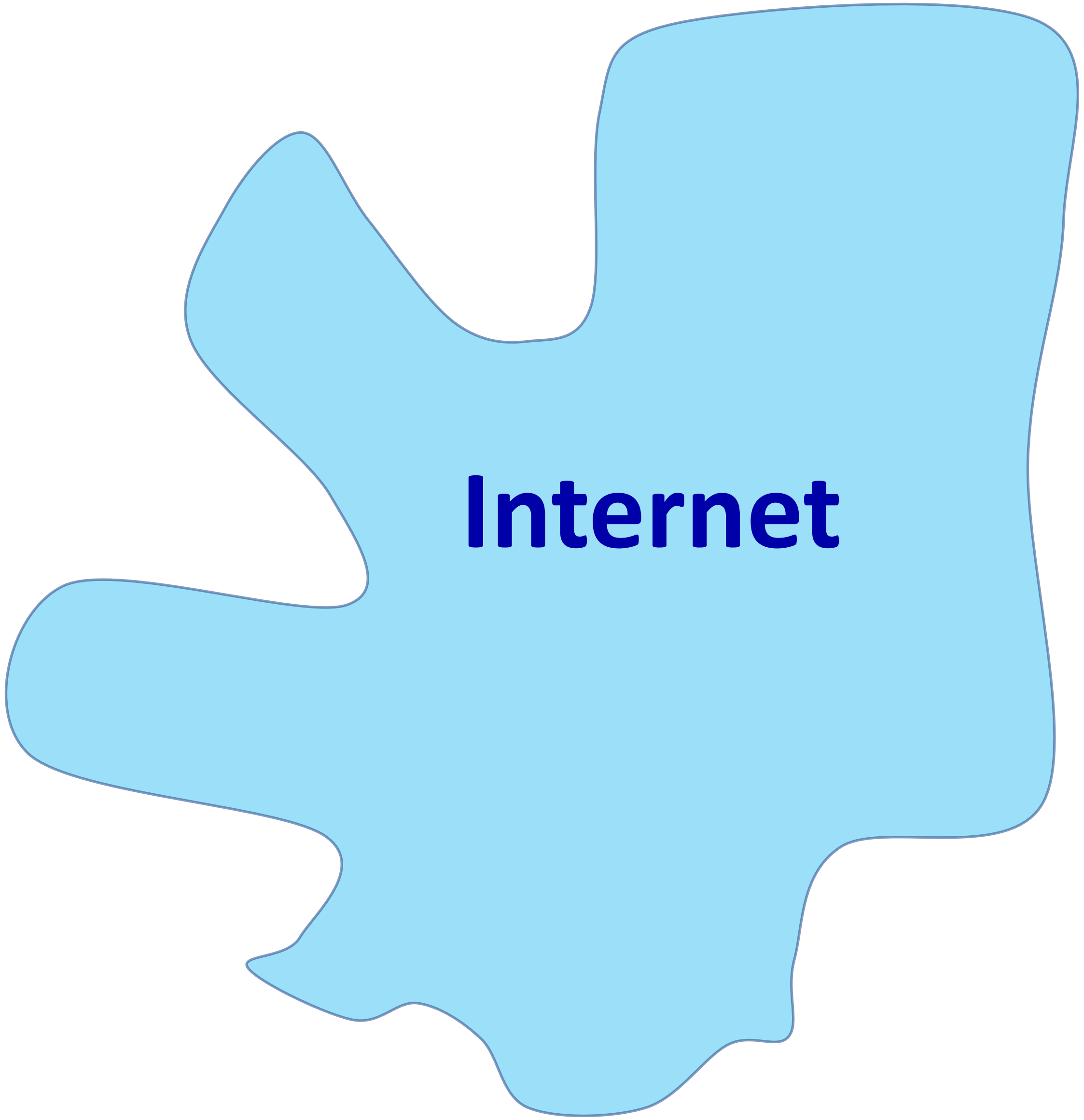
Chapter goal:

- Get "feel," "big picture," introduction to terminology
  - more depth, detail *later* in course

Overview/roadmap:

- What *is* the Internet? What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Protocol layers, service models

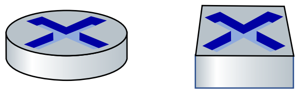
What is the Internet?



The Internet: a "nuts and bolts" view

Billions of connected computing *devices*:

- *hosts = end systems*
- running *network apps* at Internet's "edge"



Packet switches: forward packets (chunks of data)

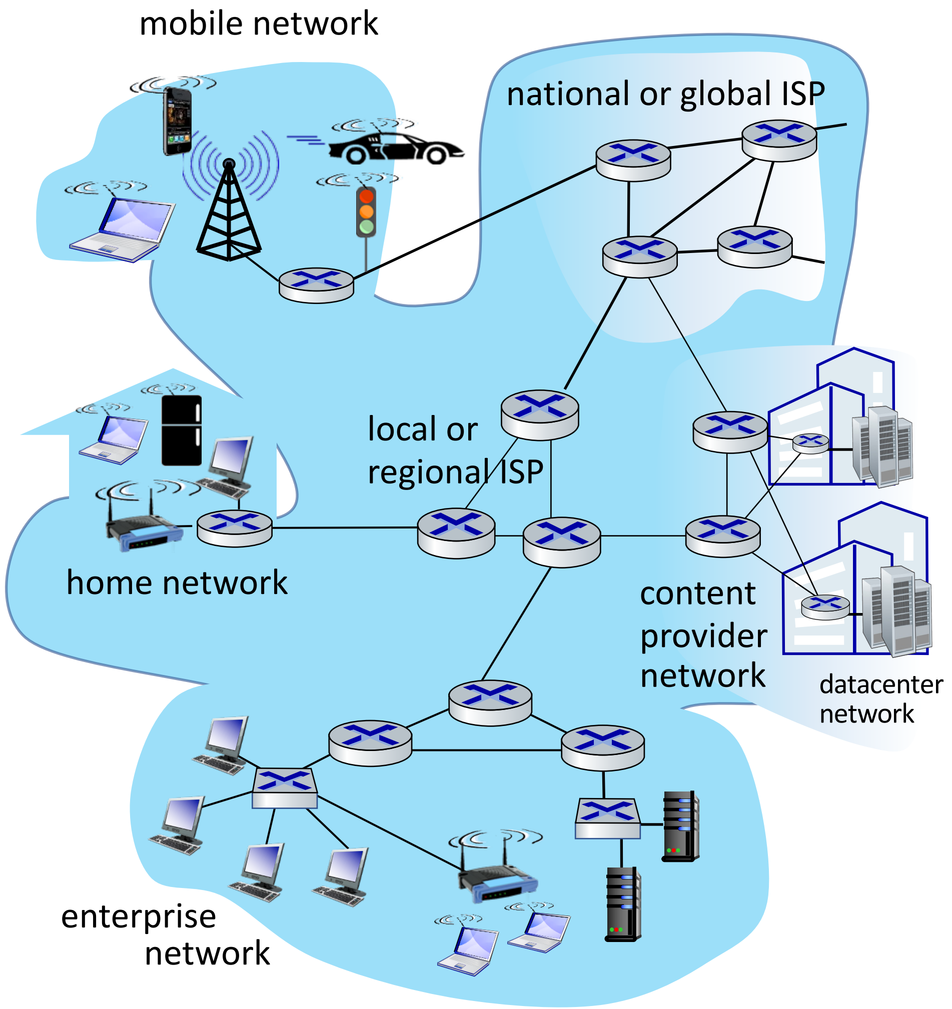
- *routers, switches*

Communication links

- fiber, copper, radio, satellite
- transmission rate: *bandwidth*

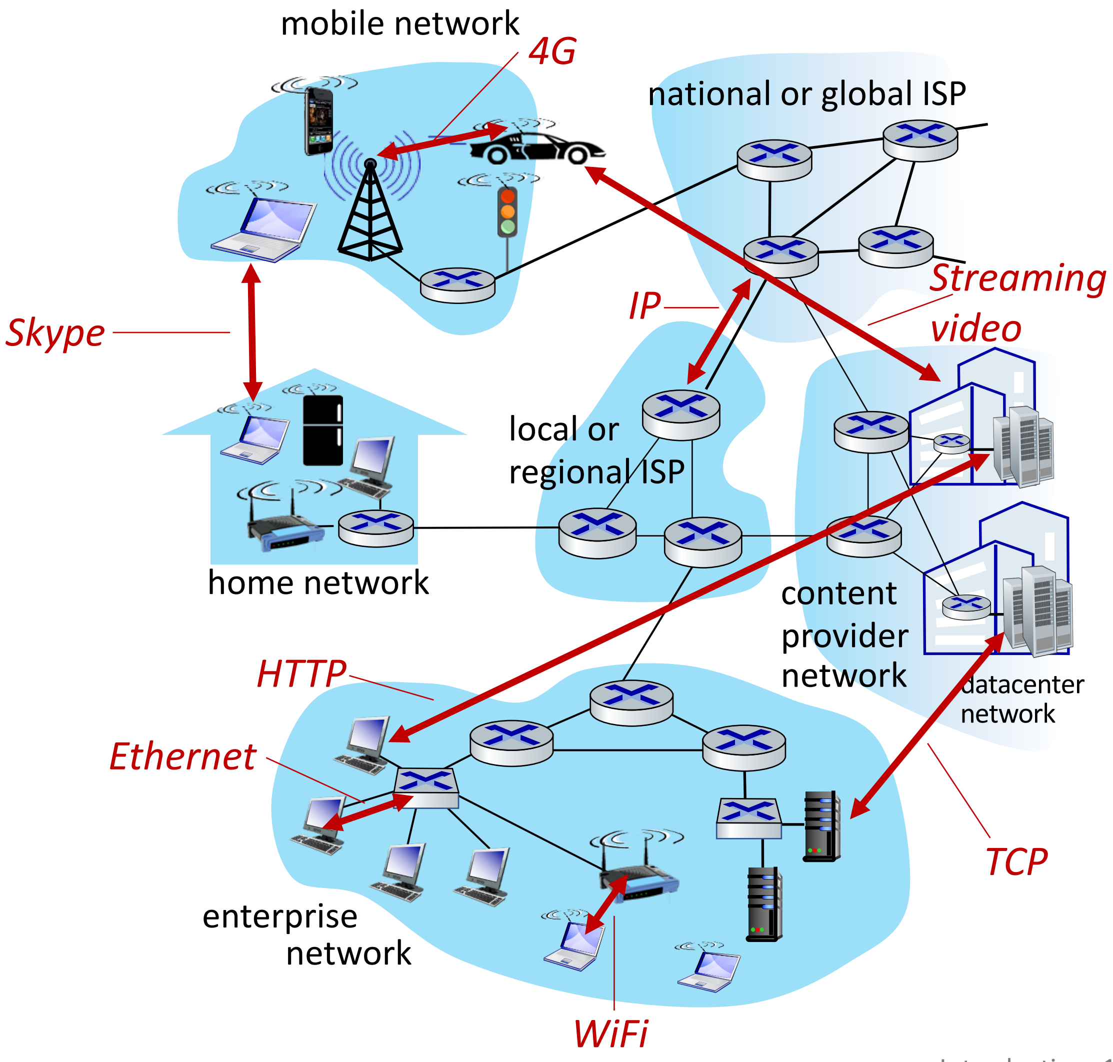
Networks

- collection of devices, routers, links: managed by an organization



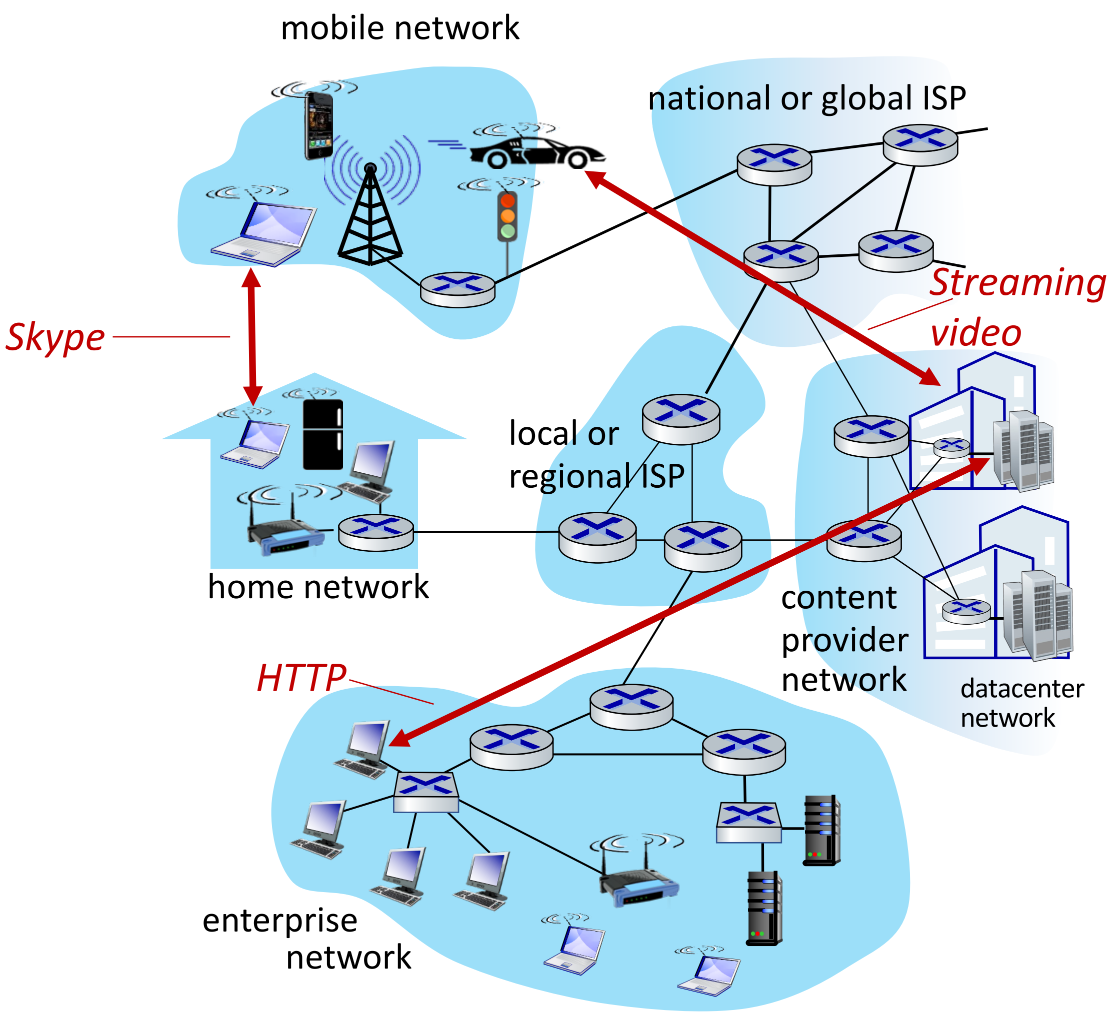
The Internet: a “nuts and bolts” view

- *Internet: “network of networks”*
  - Interconnected ISPs
- *protocols* are *everywhere*
  - control sending, receiving of messages
  - e.g., HTTP (Web), streaming video, Skype, TCP, IP, WiFi, 4G, Ethernet
- *Internet standards*
  - RFC: Request for Comments
  - IETF: Internet Engineering Task Force



The Internet: a “services” view

- *Infrastructure* that provides services to applications:
  - Web, streaming video, multimedia teleconferencing, email, games, e-commerce, social media, inter-connected appliances, …
- provides *programming interface* to distributed applications:
  - “hooks” allowing sending/receiving apps to “connect” to, use Internet transport service
  - provides service options, analogous to postal service



What's a protocol?

Human protocols:

- "what's the time?"
- "I have a question"
- introductions

Rules for:

… specific messages sent

… specific actions taken when message received, or other events

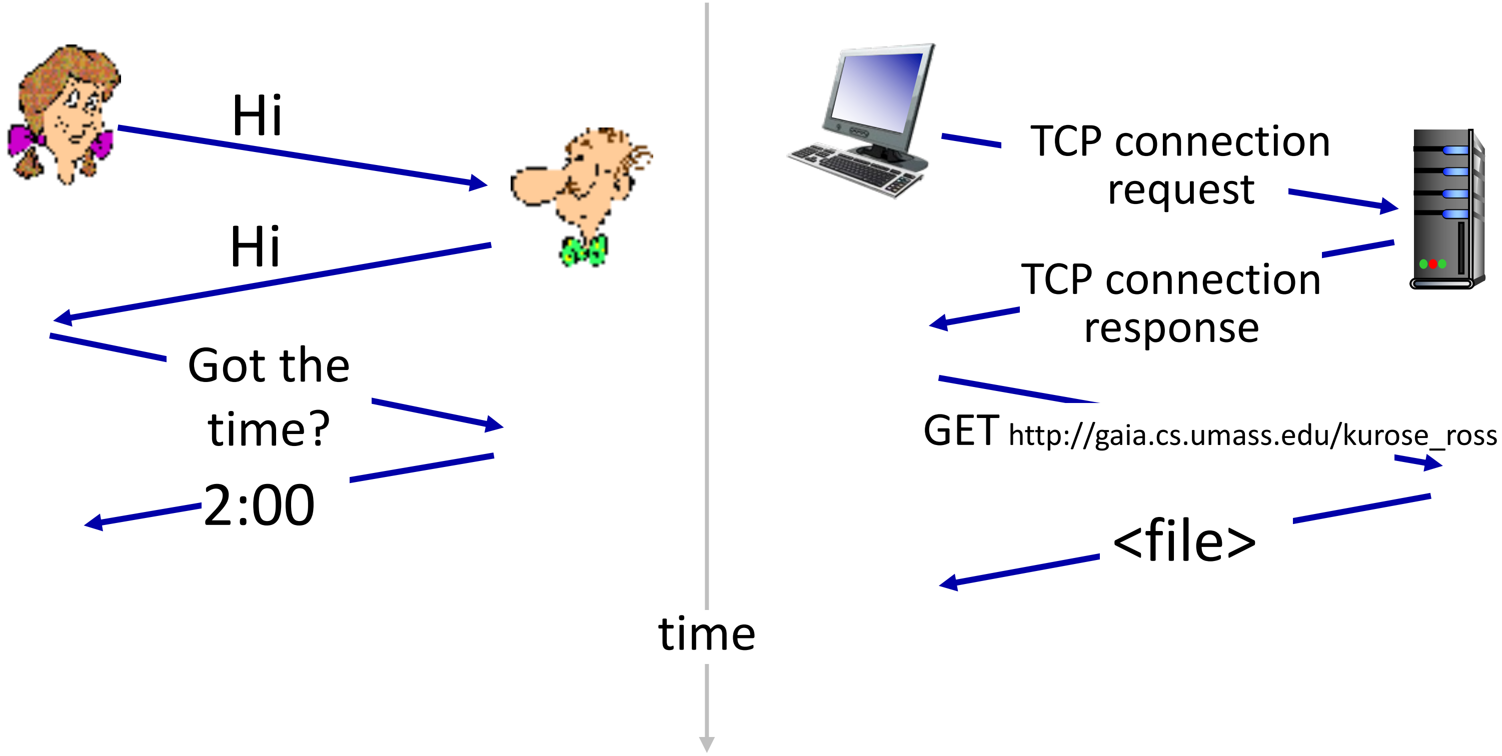
Network protocols:

- computers (devices) rather than humans
- all communication activity in Internet governed by protocols

Protocols define the format, order of messages sent and received among network entities, and actions taken on message transmission, receipt

What’s a protocol?

A human protocol and a computer network protocol:



Q: other human protocols?

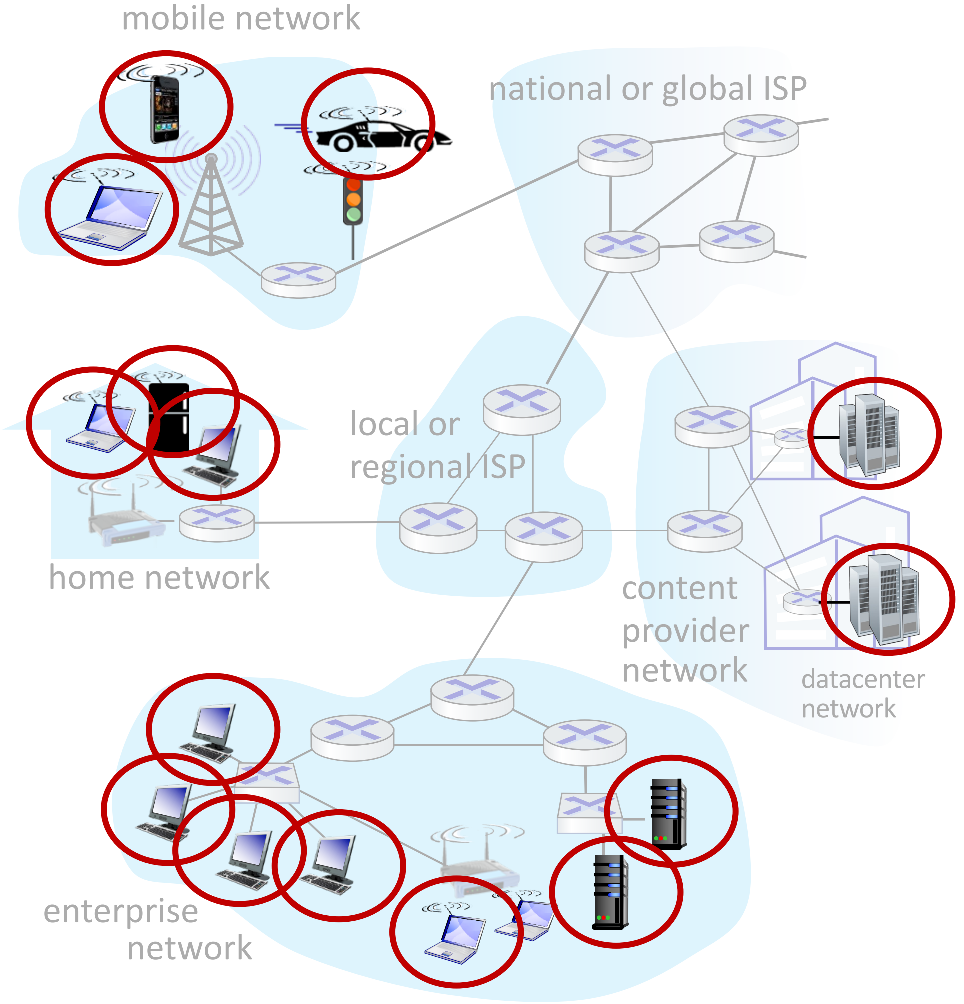
Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- ~~Security~~
- Protocol layers, service models
- ~~History~~

A closer look at Internet structure

Network edge:

- hosts: clients and servers
- servers often in data centers



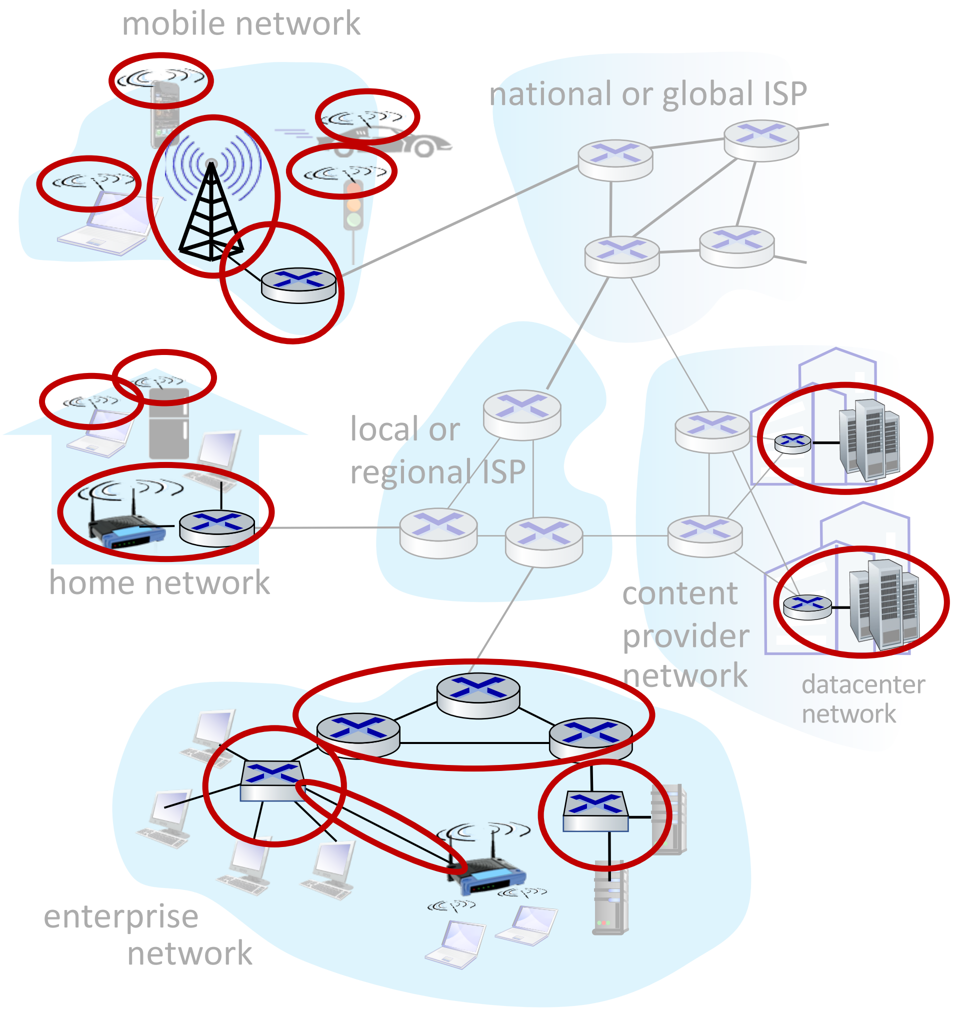
A closer look at Internet structure

Network edge:

- hosts: clients and servers
- servers often in data centers

Access networks, physical media:

- wired, wireless communication links

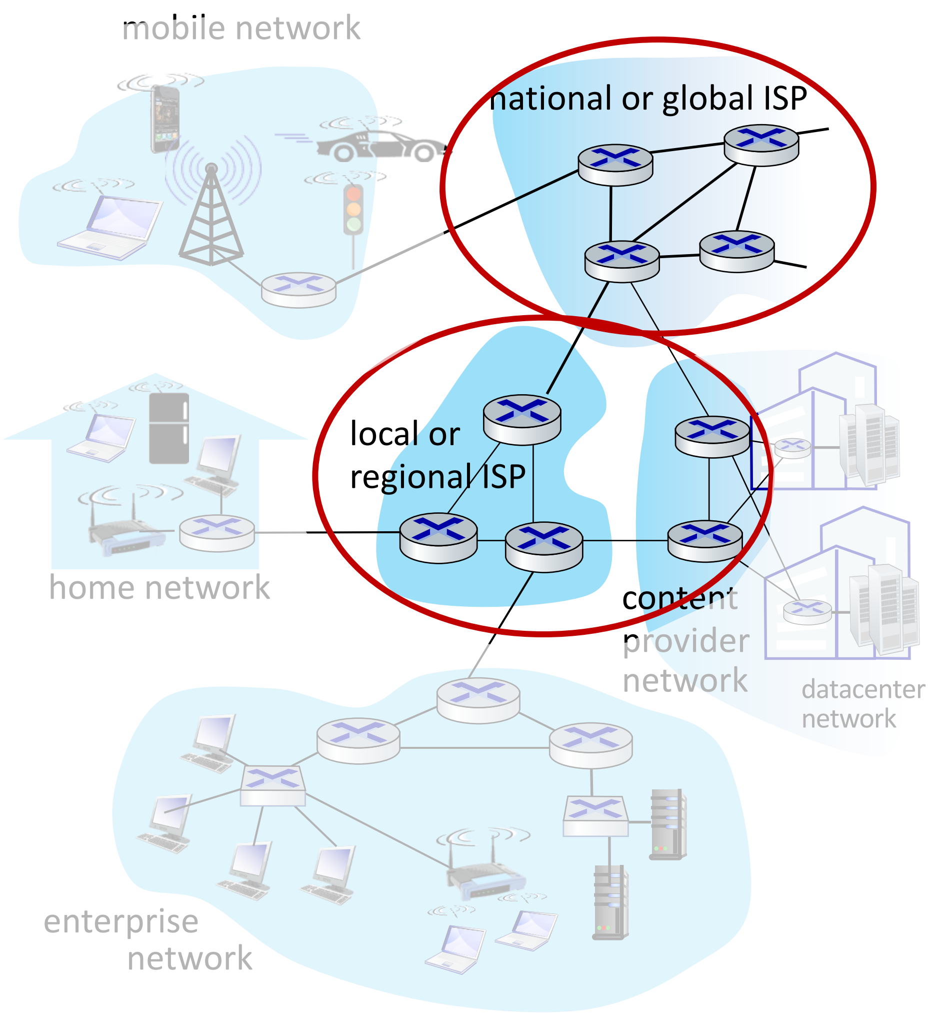


A closer look at Internet structure

Network edge:
- hosts: clients and servers
- servers often in data centers

Access networks, physical media:
- wired, wireless communication links

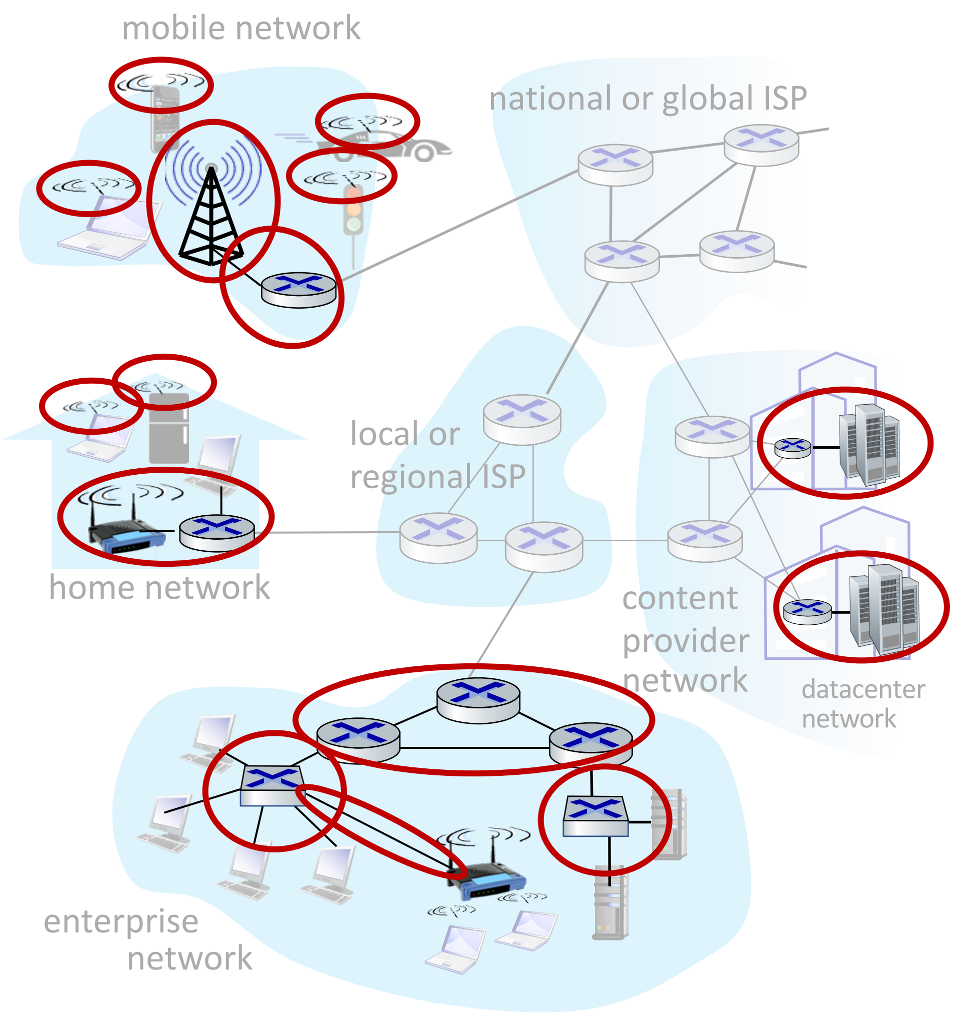
Network core:
- interconnected routers
- network of networks



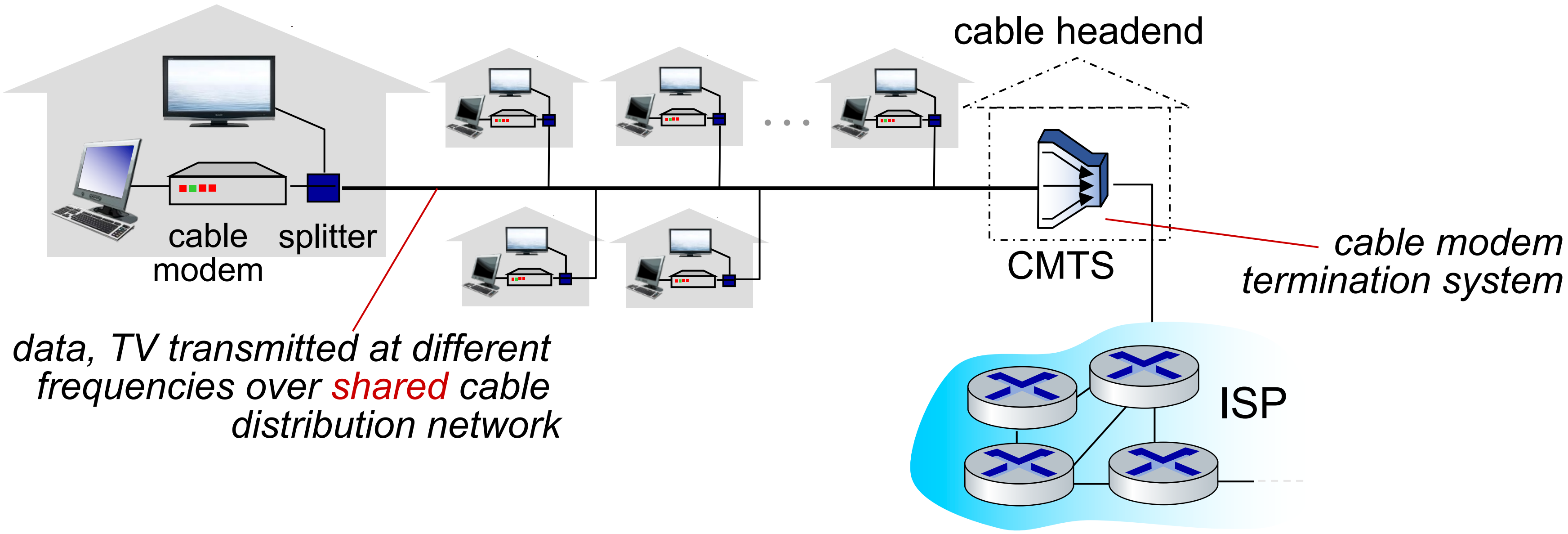
Access networks and physical media

Q: How to connect end systems to edge router?

- residential access nets
- institutional access networks (school, company)
- mobile access networks (WiFi, 4G/5G)



Access networks: cable-based access



- HFC: hybrid fiber coax
  - asymmetric: up to 40 Mbps – 1.2 Gbps downstream transmission rate, 30-100 Mbps upstream transmission rate
- network of cable, fiber attaches homes to ISP router
  - homes *share access network* to cable headend

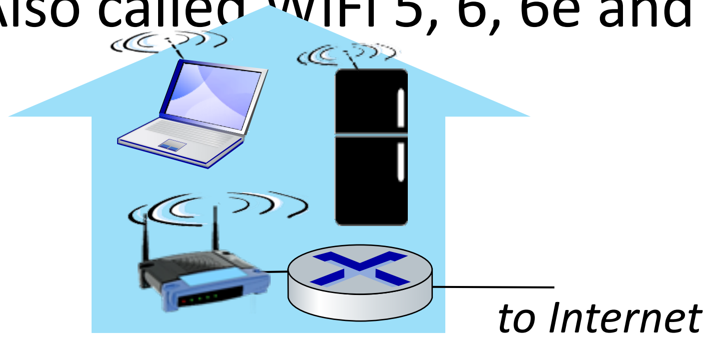
Wireless access networks

Shared *wireless* access network connects end system to router

- via base station aka "access point"

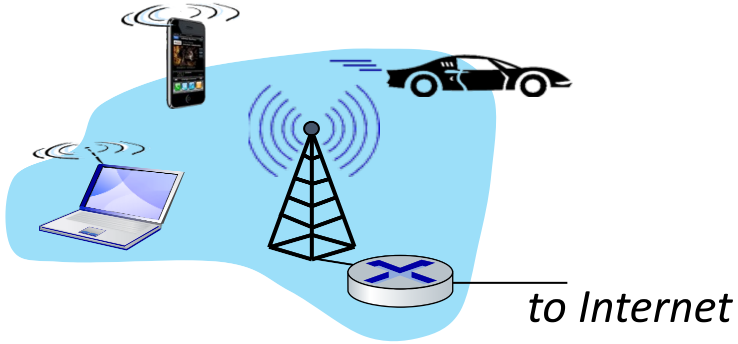
Wireless local area networks (WLANs)

- typically within or around building (~100 ft)
- 802.11b/g/n/ac/be (WiFi): up to 46 Gbps transmission rate
- Also called WiFi 5, 6, 6e and 7

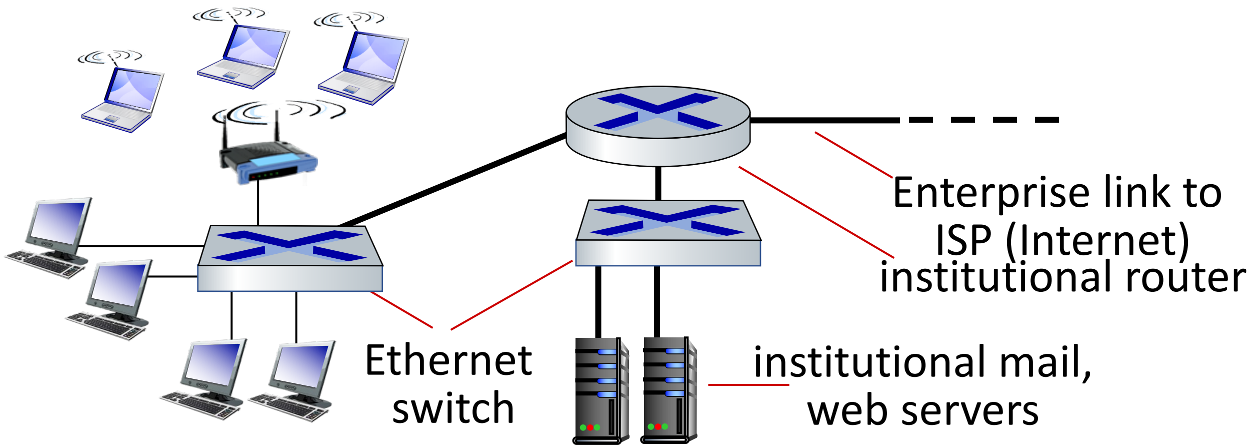


Wide-area cellular access networks

- provided by mobile, cellular network operator (10's km)
- 10 - 1000 Mbps
- 5G/4G cellular networks



Access networks: enterprise networks

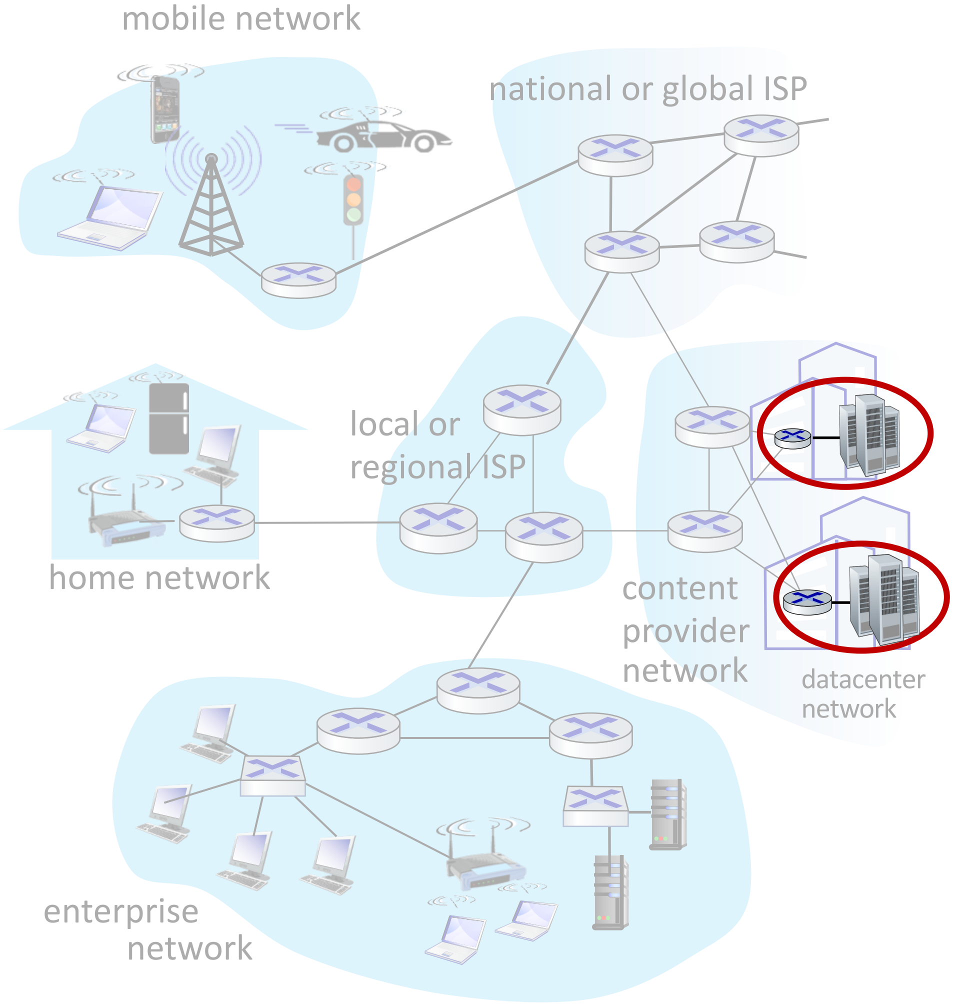


- companies, universities, etc.
- mix of wired, wireless link technologies, connecting a mix of switches and routers (we'll cover differences shortly)
  - Ethernet: wired access at 100Mbps, 1Gbps, 10Gbps
  - WiFi: wireless access points at 11, 54, 450 Mbps, 3.5Gbps, 46 Gbps

Access networks: data center networks

- high-bandwidth links (10s to 100s Gbps) connect hundreds to thousands of servers together, and to Internet

Courtesy: Massachusetts Green High Performance Computing Center (mghpcc.org)

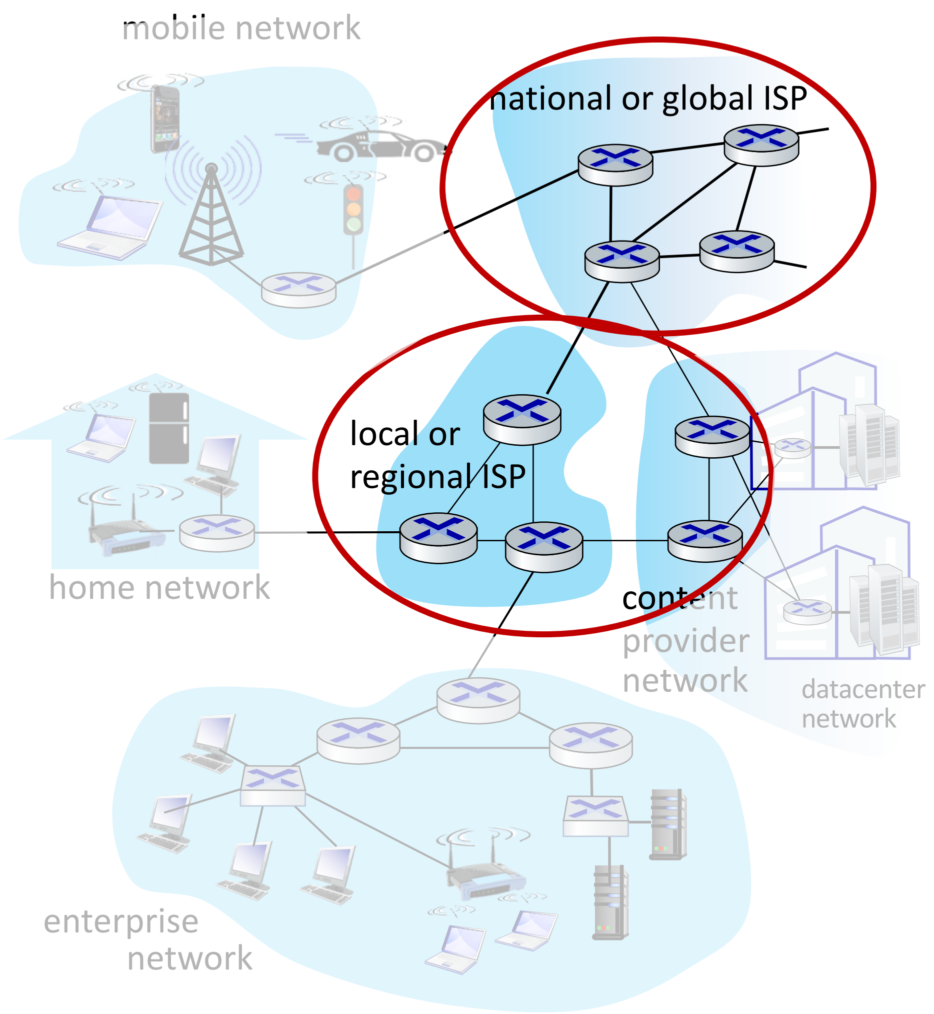


Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- **Network core: packet/circuit switching, internet structure**
- Performance: loss, delay, throughput
- ~~Security~~
- Protocol layers, service models
- ~~History~~

The network core

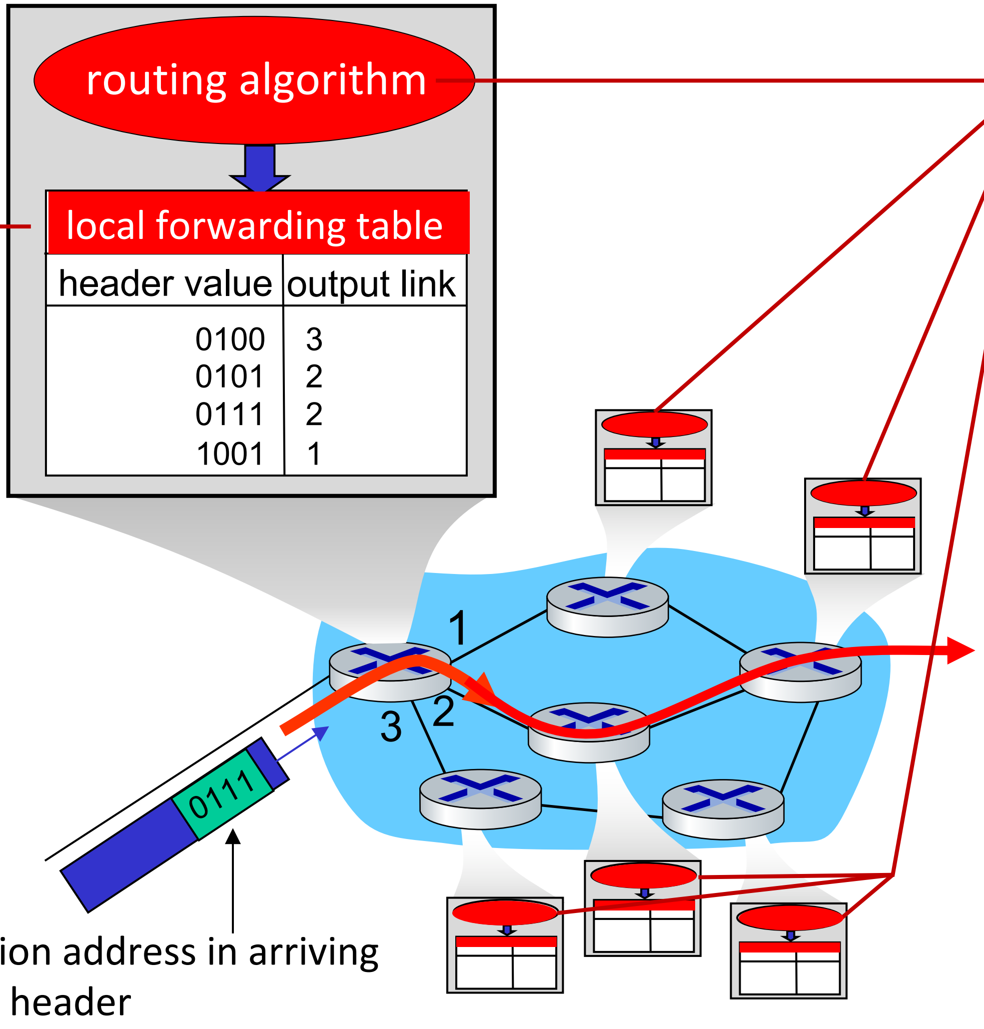
- mesh of interconnected routers
- packet-switching: hosts break application-layer messages into *packets*
  - network forwards packets from one router to the next, across links on path from source to destination



Two key network-core functions

Forwarding:

- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link



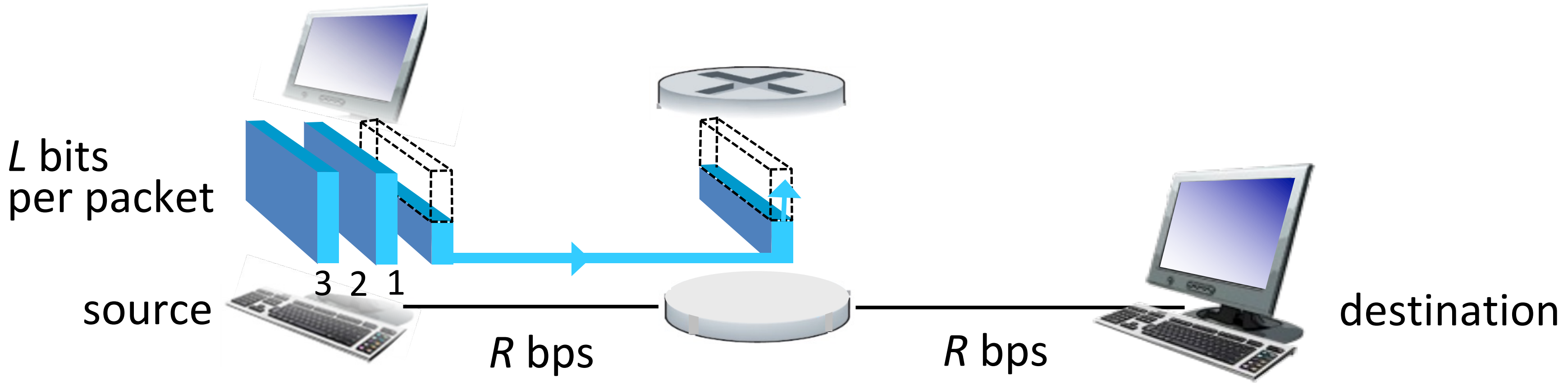
Routing:

- *global* action: determine source-destination paths taken by packets
- routing algorithms


routing
45 h
3,053 miles
48 h
3,309 miles
San Jose
Northampton
Victoria
Seattle
WASHINGTON
Portland
OREGON
IDAHO
MONTANA
WYOMING
NORTH DAKOTA
SOUTH DAKOTA
MINNESOTA
Minneapolis
WISCONSIN
IOWA
MICHIGAN
Chicago
Detroit
Toronto
Ottawa
Montreal
Quebec City
MAINE
VERMONT
NEW HAMPSHIRE
CT
RI
New York
Philadelphia
NJ
PENNSYLVANIA
MARYLAND
DE
OHIO
INDIANA
Indianapolis
ILLINOIS
Kansas City
United States
NEVADA
UTAH
COLORADO
Sacramento
San Francisco
CALIFORNIA
Las Vegas
WEST VIRGINIA
VIRGINIA
KENTUCKY
Nashville
TENNESSEE
NORTH CAROLINA
Charlotte
SOUTH CAROLINA
Atlanta
GEORGIA
ALABAMA
MISSISSIPPI
ARKANSAS
OKLAHOMA
Los Angeles
San Diego
ARIZONA
Phoenix
Tucson
NEW MEXICO
El Paso
Dallas
TEXAS
Austin
Houston
San Antonio
LOUISIANA
New Orleans
Jacksonville
Orlando
Tampa
FLORIDA
BAJA CALIFORNIA
SONORA
CHIHUAHUA
COAHUILA

forwarding
forwarding
45 h
3,053 miles
48 h
3,309 miles
United States
San Jose
Northampton
Sacramento
Cleveland

Packet-switching: store-and-forward

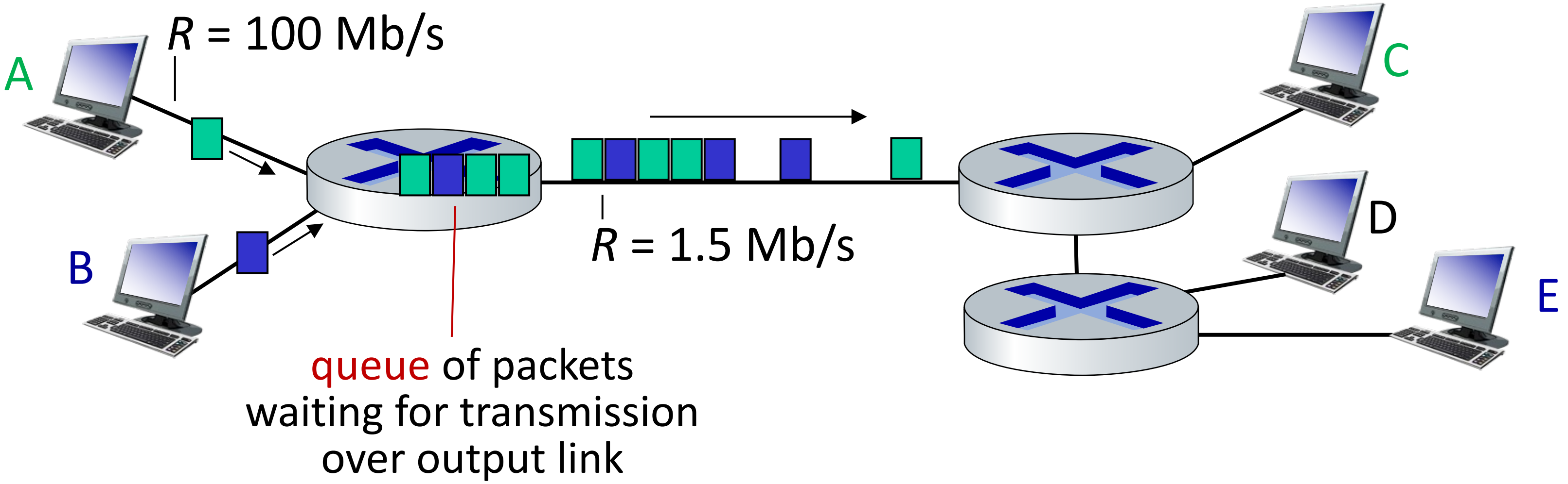


- packet transmission delay: takes L/R seconds to transmit (push out) L-bit packet into link at R bps
- *store and forward:* ***entire*** packet must arrive at router **before** it can be transmitted on next link

One-hop numerical example:

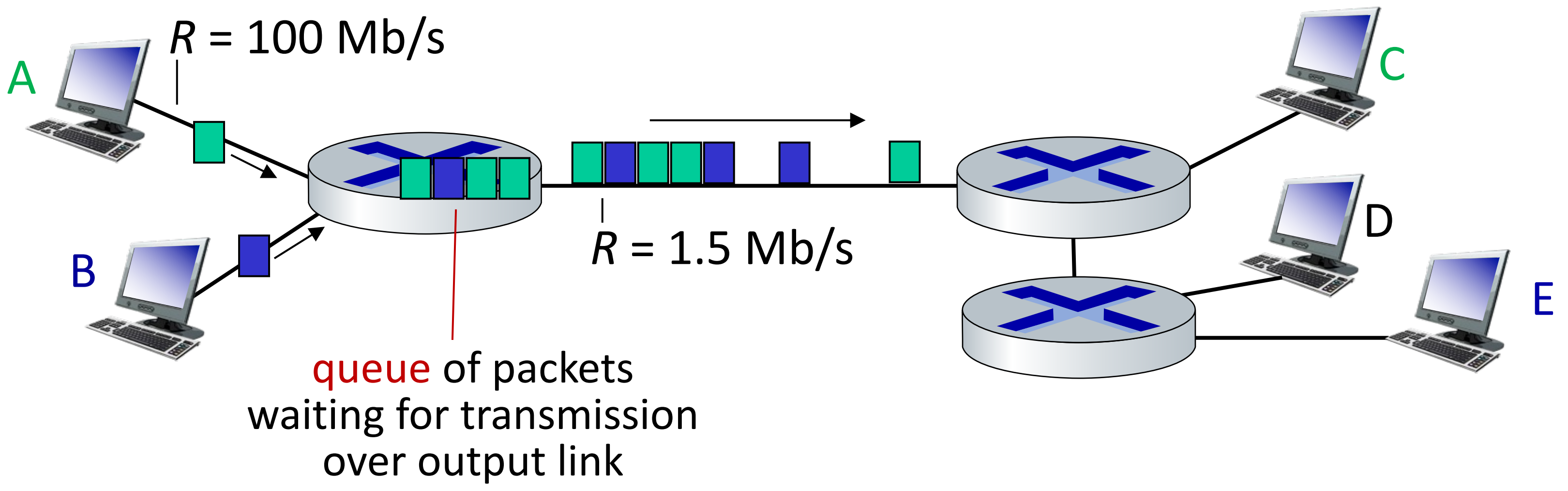
- L = 10 Kbits
- R = 100 Mbps
- one-hop transmission delay = 0.1 msec

Packet-switching: queueing



Queueing occurs when work arrives faster than it can be serviced:

Packet-switching: queueing



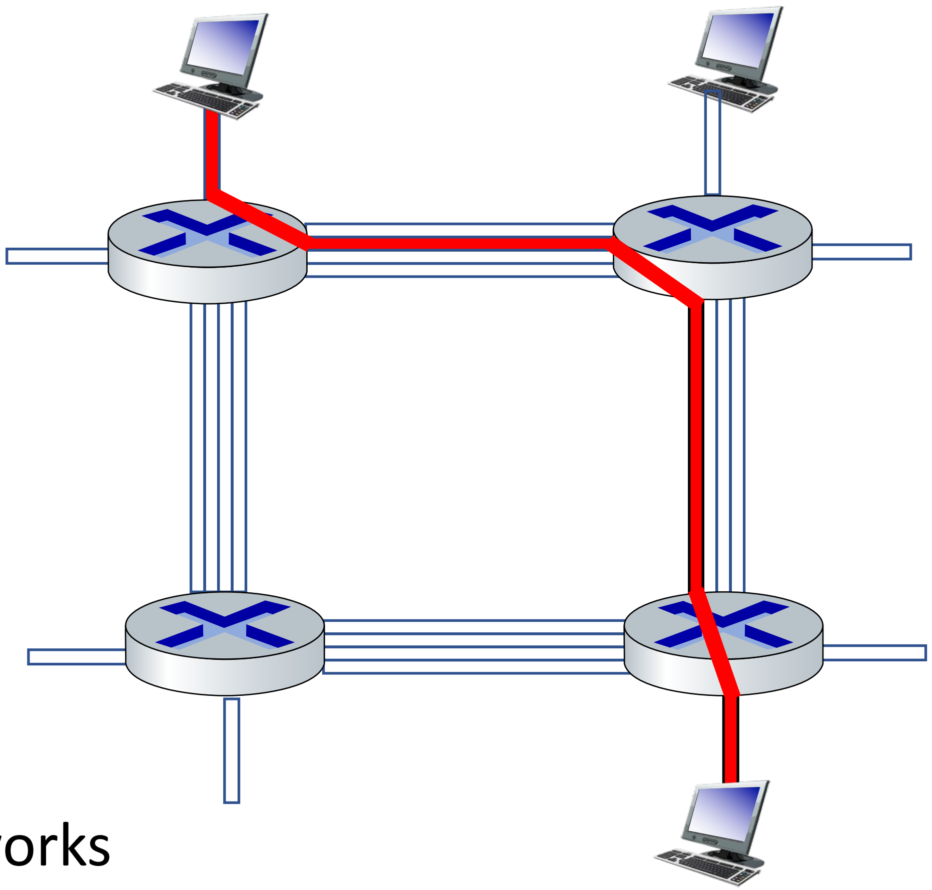
Packet queuing and loss: if arrival rate (in bps) to link exceeds transmission rate (bps) of link for some period of time:

- packets will queue, waiting to be transmitted on output link
- packets can be dropped (lost) if memory (buffer) in router fills up

Alternative to packet switching: circuit switching

end-end resources allocated to, reserved for “call” between source and destination

- in diagram, each link has four circuits.
  - call gets 2nd circuit in top link and 1st circuit in right link.
- dedicated resources: no sharing
  - circuit-like (guaranteed) performance
- circuit segment idle if not used by call (no sharing)
- commonly used in traditional telephone networks

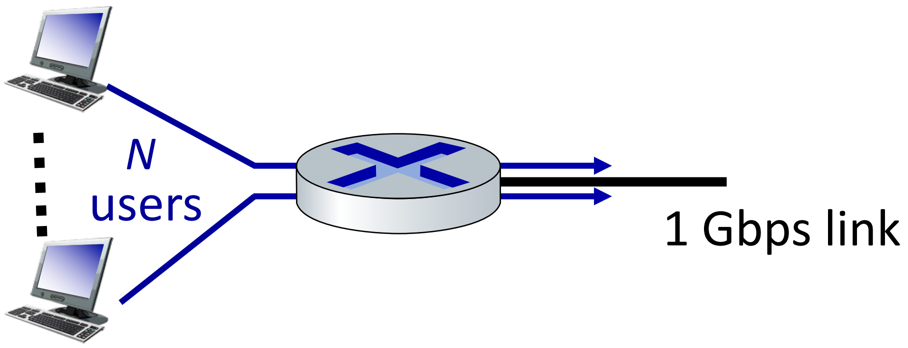


* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive

Packet switching versus circuit switching

example:

- 1 Gb/s link
- each user:
  - 100 Mb/s when "active"
  - active 10% of time



Q: how many users can use this network under circuit-switching and packet switching?

- *circuit-switching:* 10 users
- *packet switching:* with 35 users, probability > 10 active at same time is less than .0004 *

Q: how did we get value 0.0004?

A: A math problem (for those with course in probability only)

* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive

Packet switching versus circuit switching

Is packet switching a "slam dunk winner"?

- great for "bursty" data – sometimes has data to send, but at other times not
  - resource sharing
  - simpler, no call setup
- excessive congestion possible: packet delay and loss due to buffer overflow
  - protocols needed for reliable data transfer, congestion control
- *Q:* How to provide circuit-like behavior with packet-switching?
  - "It's complicated." We'll study various techniques that try to make packet switching as "circuit-like" as possible.

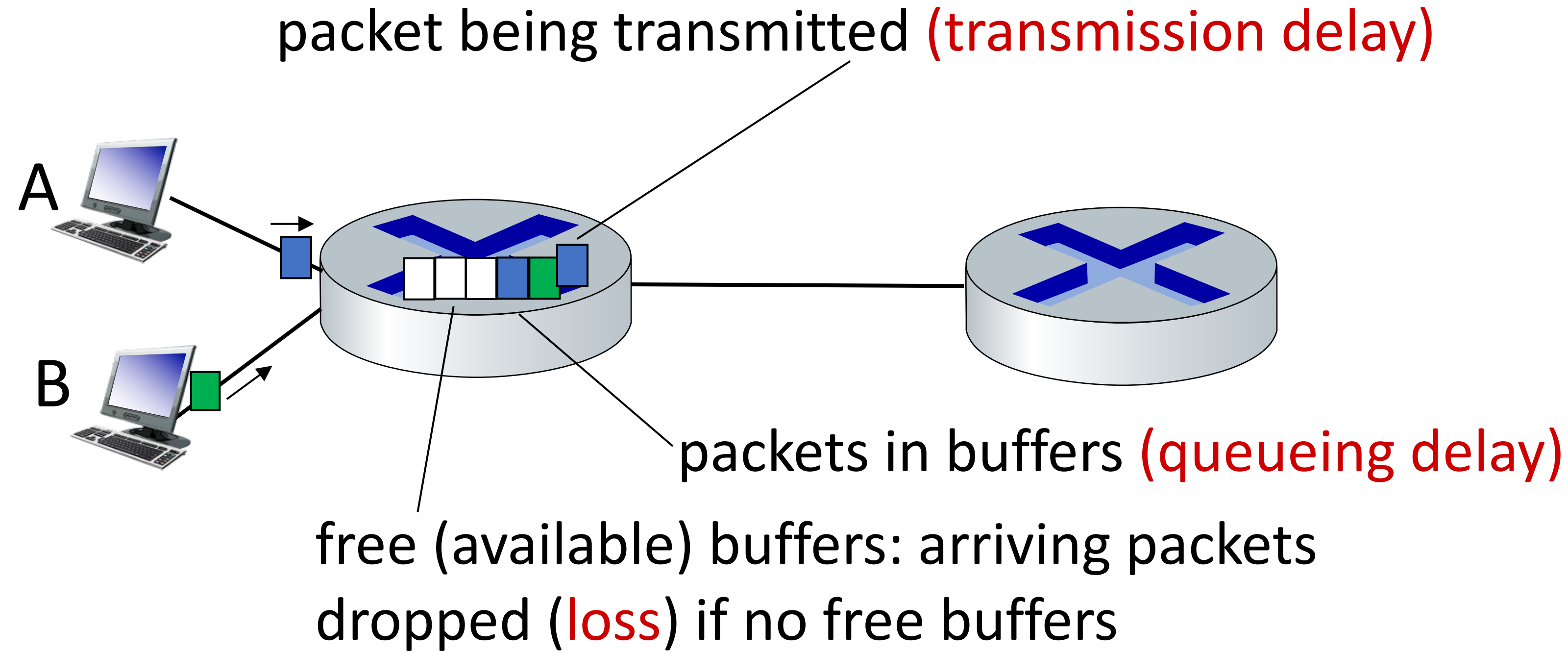
Q: human analogies of reserved resources (circuit switching) versus on-demand allocation (packet switching)?

Chapter 1: roadmap

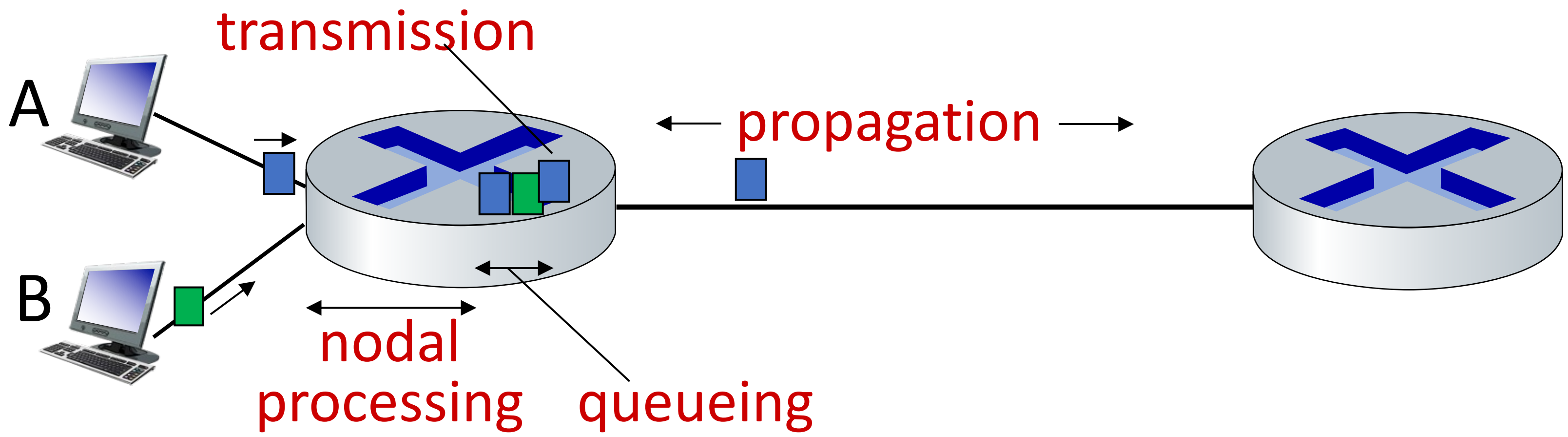
- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- ~~Security~~
- Protocol layers, service models
- ~~History~~

How do packet delay and loss occur?

- packets *queue* in router buffers, waiting for turn for transmission
  - queue length grows when arrival rate to link (temporarily) exceeds output link capacity
- packet *loss* occurs when memory to hold queued packets fills up



Packet delay: four sources



$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

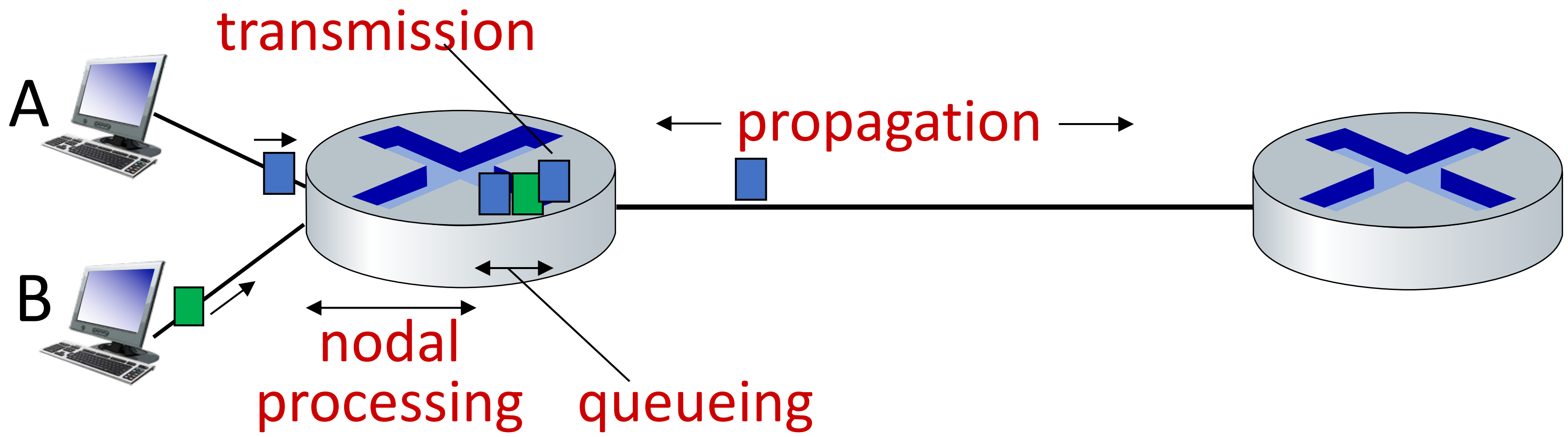
d_{proc}: nodal processing

- check bit errors
- determine output link
- typically < microsecs

d_{queue}: queueing delay

- time waiting at output link for transmission
- depends on congestion level of router

Packet delay: four sources



$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

d_{trans}: transmission delay:

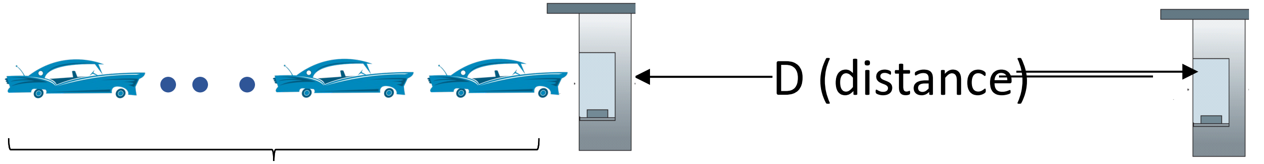
- L: packet length (bits)
- R: link *transmission rate (bps)*
- $d_{trans} = L/R$

d_{prop}: propagation delay:

- d: length of physical link
- s: propagation speed (~2×10^8 m/sec)
- $d_{prop} = d/s$

d_{trans} and d_{prop} *very* different

In-Class calculation: Caravan analogy



Scenario 1:

Scenario 2:

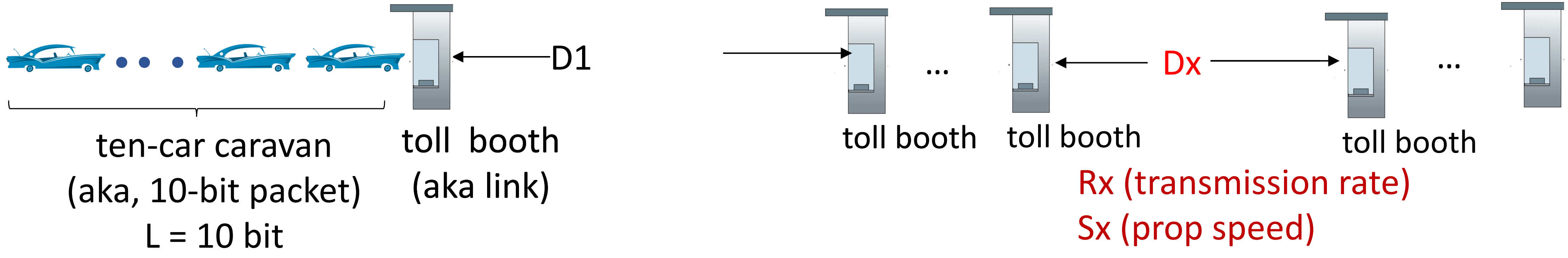
- toll booth takes 1 sec to serve one car (bit transmission time)
- D = 100m , "propagate" at 100 km/hr

- car ~ bit; caravan ~ packet; toll service ~ link t
- *Q:* How long until caravan is lined up before 2nd toll booth?

Settings	transmission delay d_{trans} = L/R	propagation delay d_{prop} = D/s	d_{total}
Scenario 1	L/R = 10 /0.1bps = 100 s	D/s = 100/100 = 1hr	1hr + 100s
Scenario 2	L/R = 10 /1bps = 10 s	D/s= 100m/100 = 3.6s	13.6s

Q: Will cars arrive to 2nd booth before all cars served at first booth? No!

In-Class calculation: More Booths …



- car ~ bit; caravan ~ packet; toll service ~ link transmission
- *Q:* How long until caravan passes the last toll booth?

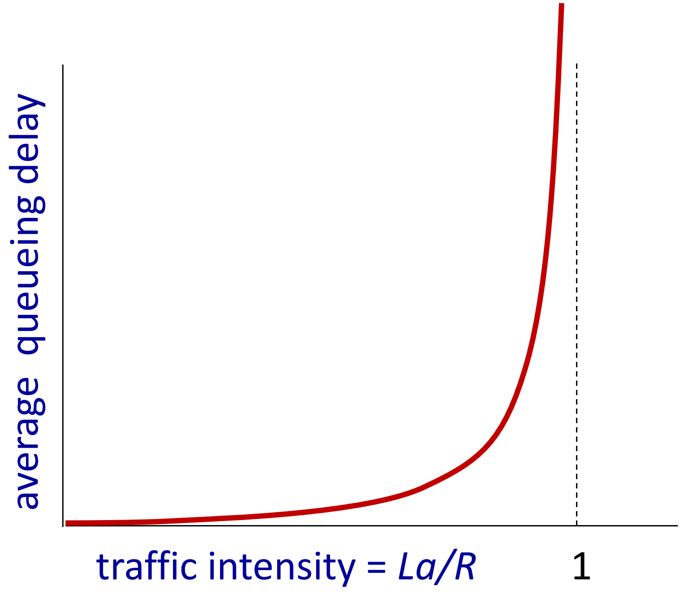
$$d_{total} = \sum d_{hop} \approx \sum (L/Rx + Dx/Sx)$$

(only transmission and prop delay considered)

Packet queueing delay (revisited)

- *a:* average packet arrival rate
- *L:* packet length (bits)
- *R:* link bandwidth (bit transmission rate)

$$\frac{L \cdot a}{R} : \frac{\text{arrival rate of bits}}{\text{service rate of bits}}$$ *"traffic intensity"*



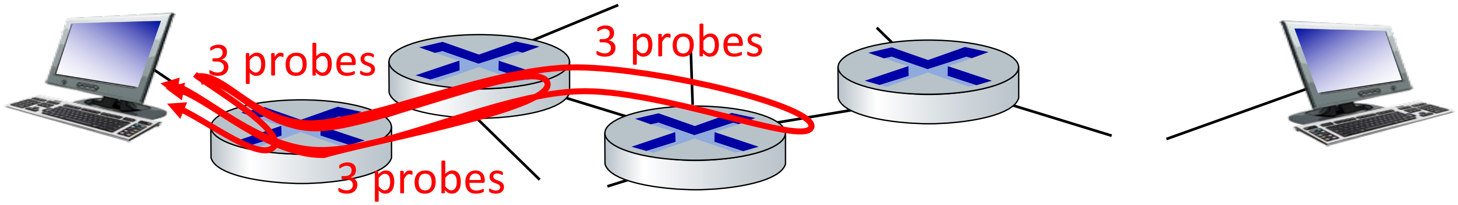
- *La/R* ~ 0: avg. queueing delay small
- *La/R* -> 1: avg. queueing delay large
- *La/R* > 1: more "work" arriving is more than can be serviced - average delay infinite!


La/R ~ 0


La/R -> 1

“Real” Internet delays and routes

- what do “real” Internet delay & loss look like?
- **traceroute** program: provides delay measurement from source to router along end-end Internet path towards destination. For all *i:*
  - sends three packets that will reach router *i* on path towards destination (with time-to-live field value of *i*)
  - router *i* will return packets to sender
  - sender measures time interval between transmission and reply



"Real" Internet delays, routes

3 delay measurements from PAL3.0

- traceroute: www.google.com

traceroute to www.google.com (142.250.191.132), 64 hops max, 72 byte packets
1 lamb-20-c7710-01-vlan1330.tcom.purdue.edu (10.184.0.10) 3.413 ms 5.932 ms 9.562 ms
2 lamb-20-c7710-03-vlan3014.tcom.purdue.edu (172.28.160.195) 4.965 ms 7.620 ms 7.767 ms
3 192.168.18.8 (192.168.18.8) 9.190 ms 8.347 ms 5.382 ms
4 tel-210-c7710-01-ptp-e1-11-1.tcom.purdue.edu (172.28.249.18) 19.250 ms 7.934 ms 4.958 ms
5 lamb-20-c7710-01-ptp-e1-3-1.tcom.purdue.edu (172.28.249.1) 5.298 ms 7.569 ms 5.606 ms
6 lamb-20-c7710-01-ptp-e10-2.tcom.purdue.edu (172.28.249.88) 5.270 ms 6.441 ms 6.710 ms
7 indiana-gigapop-lldc-internet-mx960.tcom.purdue.edu (192.5.40.187) 11.179 ms 6.225 ms 9.231 ms
8 lo-0.1.rtr.star.indiana.gigapop.net (149.165.255.11) 18.560 ms 14.497 ms 17.082 ms
9 et-0-2-2.2286.sw2.star.omnipop.btaa.org (149.165.183.86) 19.431 ms 15.515 ms 14.557 ms
10 r-equinix-isp-ae0-2401.ip4.wiscnet.net (140.189.9.133) 14.838 ms 15.049 ms 15.539 ms
11 72.14.218.180 (72.14.218.180) 14.289 ms 14.357 ms 17.355 ms
12 74.125.251.149 (74.125.251.149) 16.564 ms 12.532 ms 12.358 ms
13 142.251.60.7 (142.251.60.7) 12.217 ms 14.306 ms 13.943 ms
14 ord38s29-in-f4.1e100.net (142.250.191.132) 46.041 ms 36.942 ms 14.111 ms

* Do some traceroutes from exotic countries at www.traceroute.org

Demo in Class

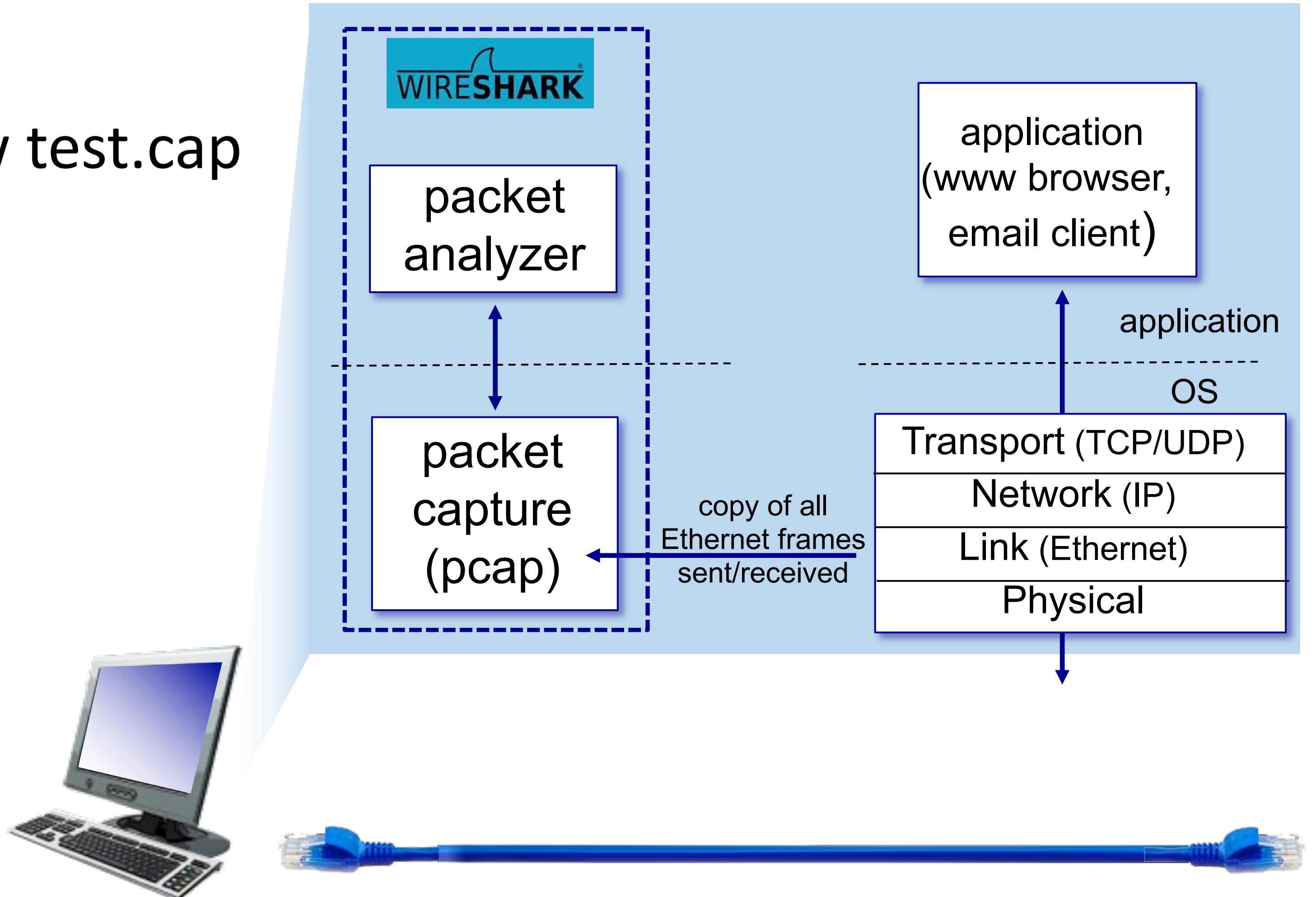
- traceroute -I www.cs.purdue.edu
- Traceroute www.google.com
- traceroute www.cam.ac.uk

(man traceroute)

- What are differences you can see?
- Can you see the link across the ocean? Why or why not?

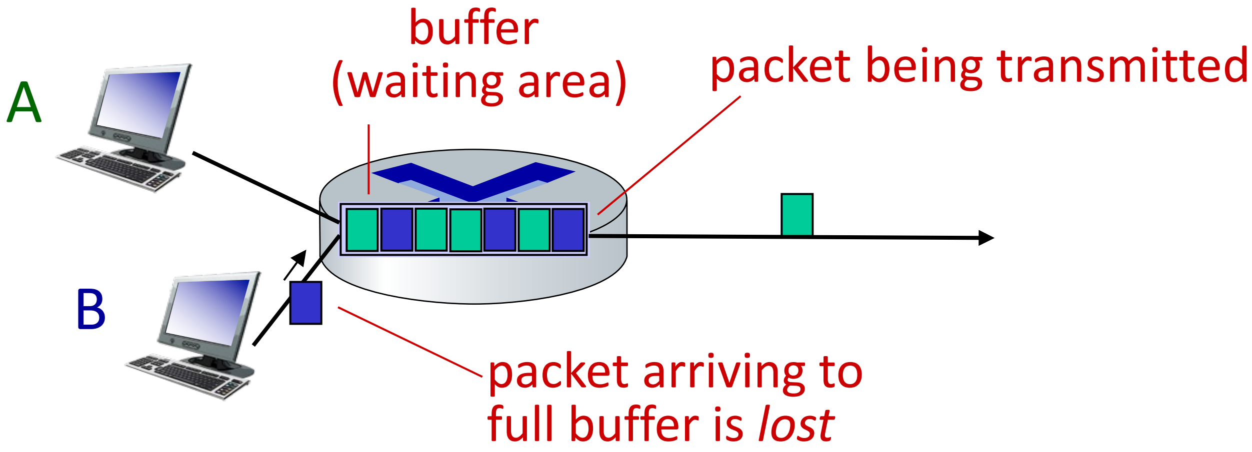
Networking tools: Packet sniffer and analyzer

- **tcpdump** (command)
  - > tcpdump –i en0
  - > tcpdump -i en0 -c 10 -w test.cap
  - > tcpdump –r test.cap
- **wireshark (UI)**



Packet loss

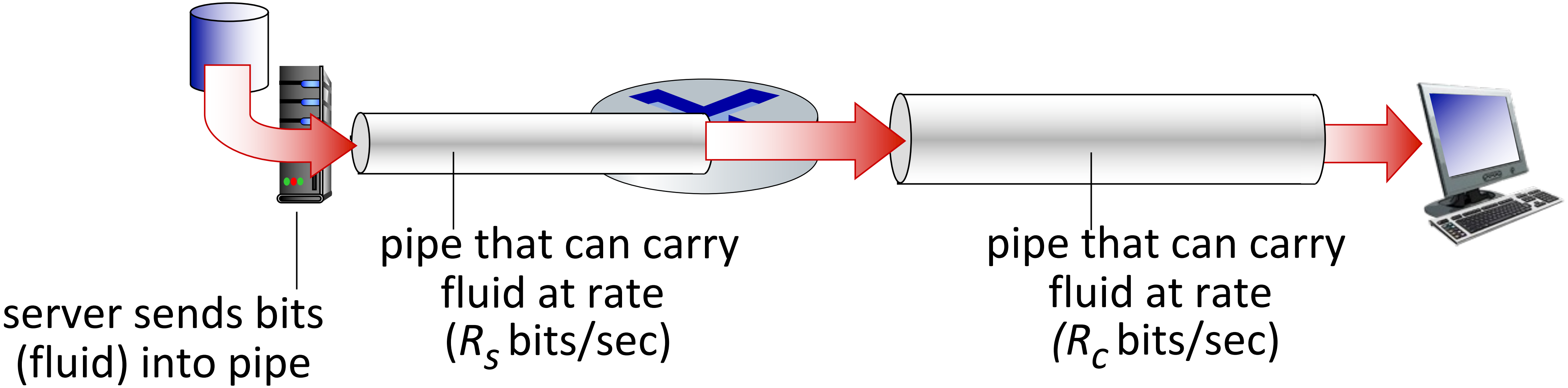
- queue (aka buffer) preceding link in buffer has finite capacity
- packet arriving to full queue dropped (aka lost)
- lost packet may be retransmitted by previous node, by source end system, or not at all



* Check out the Java applet for an interactive animation (on publisher's website) of queuing and loss

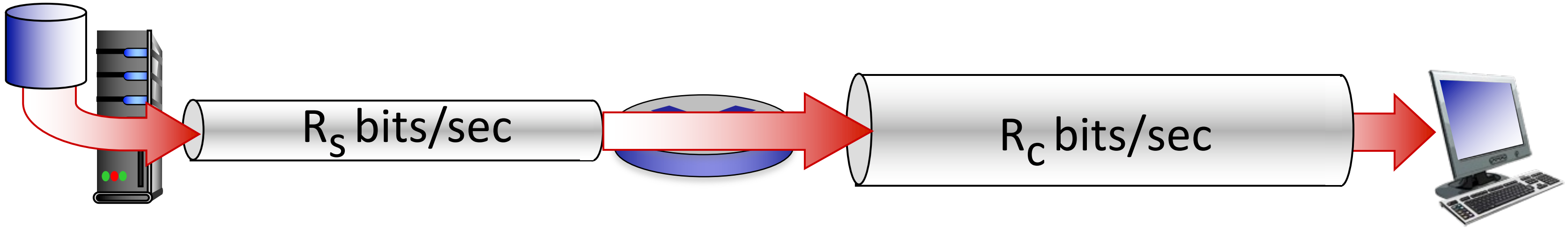
Throughput

- *throughput:* rate (bits/time unit) at which bits are being sent from sender to receiver
  - *instantaneous:* rate at given point in time
  - *average:* rate over longer period of time

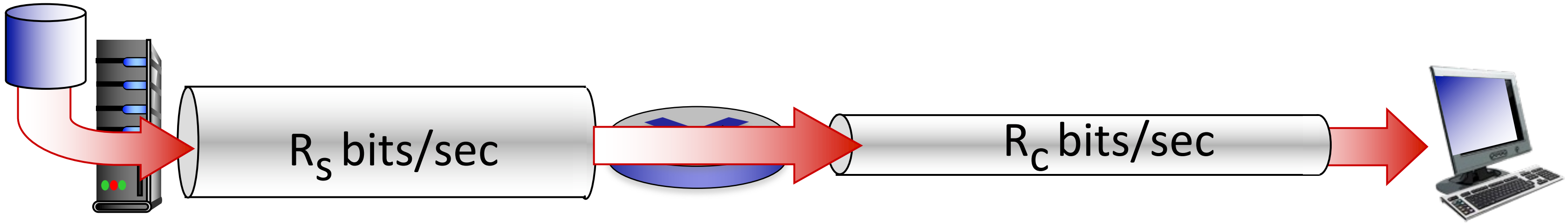


Throughput

$R_s < R_c$ What is average end-end throughput?



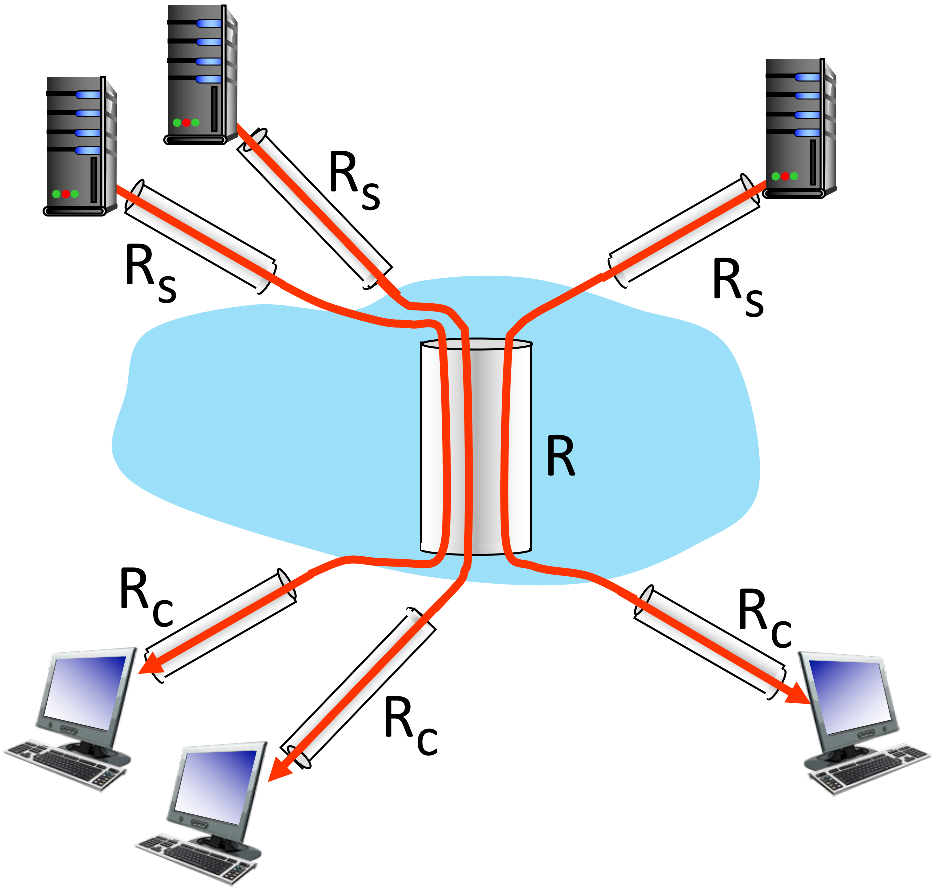
$R_s > R_c$ What is average end-end throughput?



bottleneck link

link on end-end path that constrains end-end throughput

Throughput: network scenario



10 connections (fairly) share backbone bottleneck link *R* bits/sec

- per-connection end-end throughput: $min(R_c,R_s,R/10)$
- in practice: R_c or R_s is often bottleneck

* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/

Chapter 1: roadmap

- What *is* the Internet?
- What *is* a protocol?
- Network edge: hosts, access network, physical media
- Network core: packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- ~~Security~~
- **Protocol layers, service models**
- ~~History~~

Protocol "layers" and reference models

Networks are complex, with many "pieces":

- hosts
- routers
- links of various media
- applications
- protocols
- hardware, software

Question: is there any hope of *organizing* structure of network?

- and/or our *discussion* of networks?

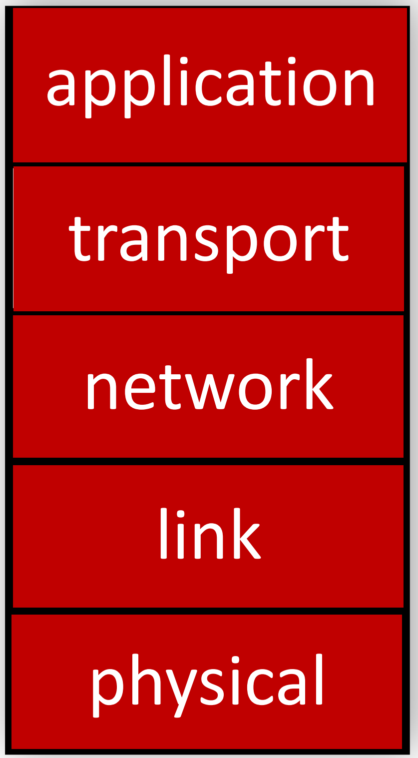
Why layering?

Approach to designing/discussing complex systems:

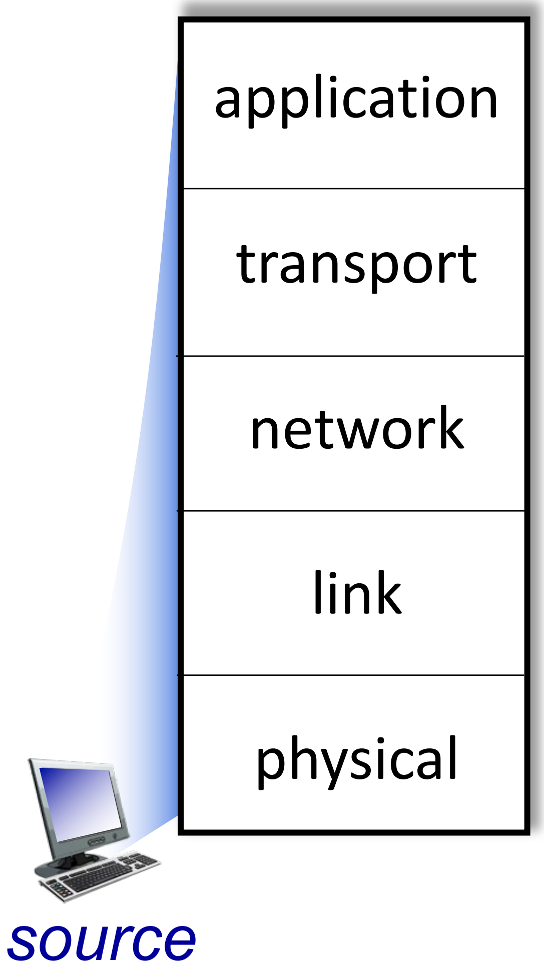
- explicit structure allows identification, relationship of system's pieces
  - layered *reference model* for discussion
- modularization eases maintenance, updating of system
  - change in layer's service *implementation*: transparent to rest of system
  - e.g., change in gate procedure doesn't affect rest of system

Layered Internet protocol stack

- *application:* supporting network applications
  - HTTP, IMAP, SMTP, DNS
- *transport:* process-process data transfer
  - TCP, UDP
- *network:* routing of datagrams from source to destination
  - IP, routing protocols
- *link:* data transfer between neighboring network elements
  - Ethernet, 802.11 (WiFi), PPP
- *physical:* bits "on the wire"



Services, Layering and Encapsulation



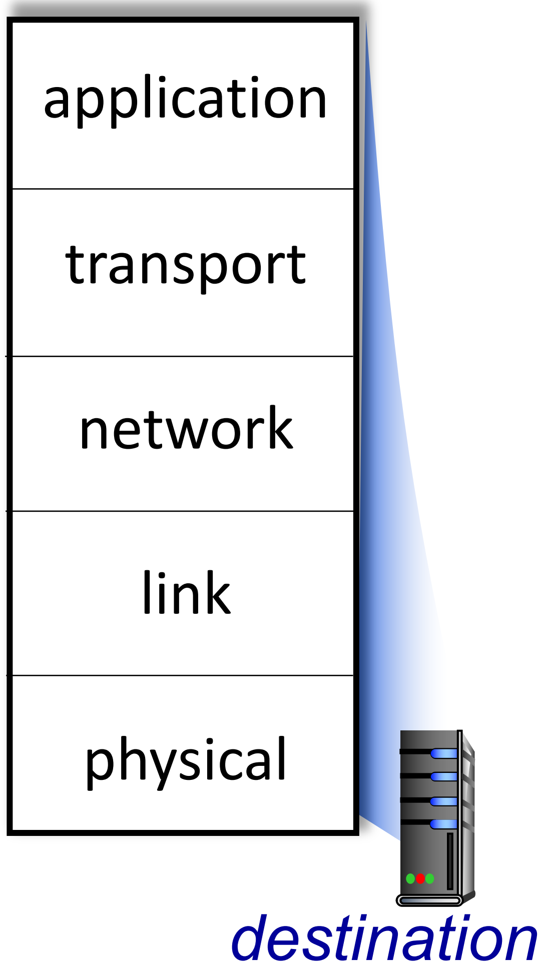
M

Application exchanges messages to implement some application service using *services* of transport layer

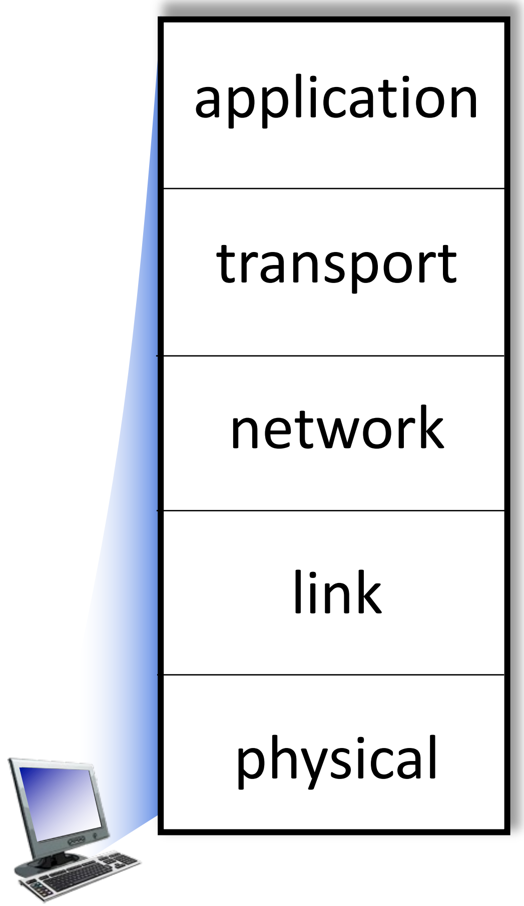
H_t M

Transport-layer protocol transfers M (e.g., reliably) from one *process* to another, using services of network layer

- transport-layer protocol encapsulates application-layer message, M, with *transport* layer-layer header H_t to create a transport-layer segment
  - H_t used by transport layer protocol to implement its service



Services, Layering and Encapsulation



source

M

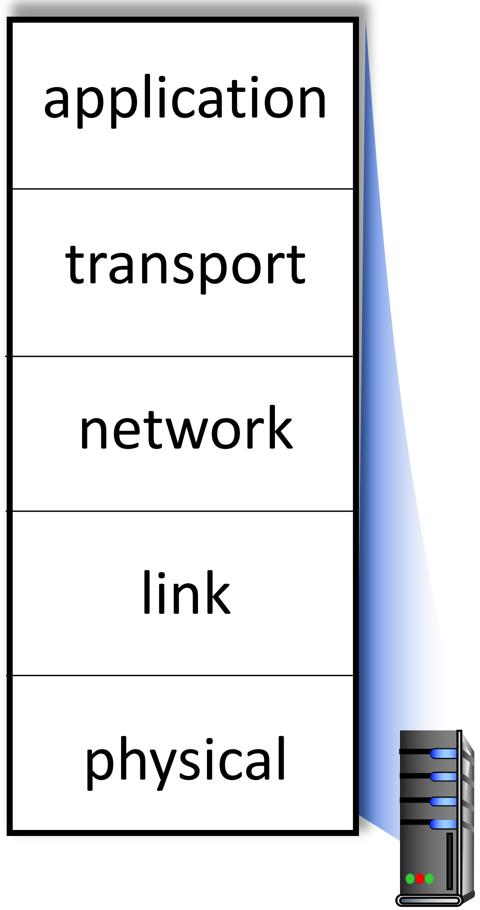
H_t M

Transport-layer protocol transfers M (e.g., reliably) from one *process* to another, using services of network layer

H_n H_t M

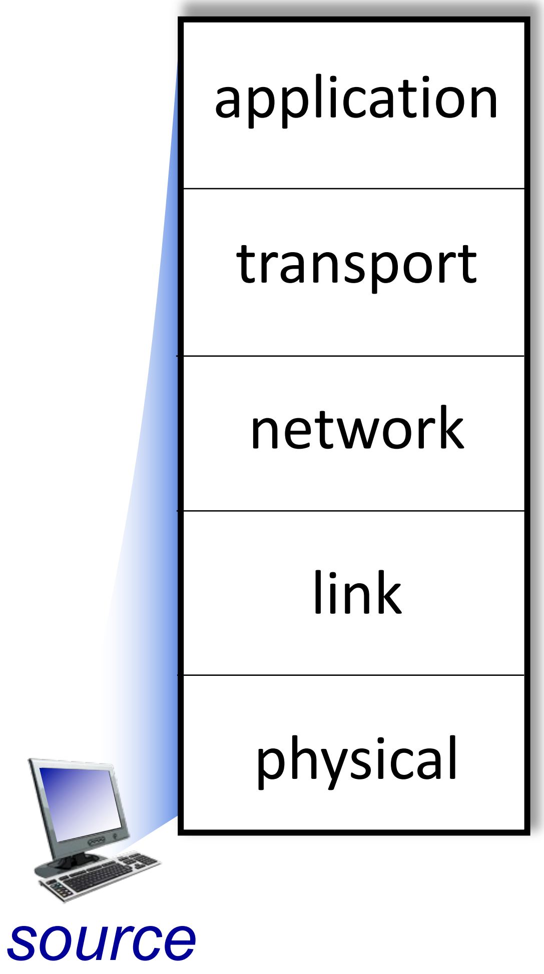
Network-layer protocol transfers transport-layer segment [H_t | M] from one *host* to another, using link layer services

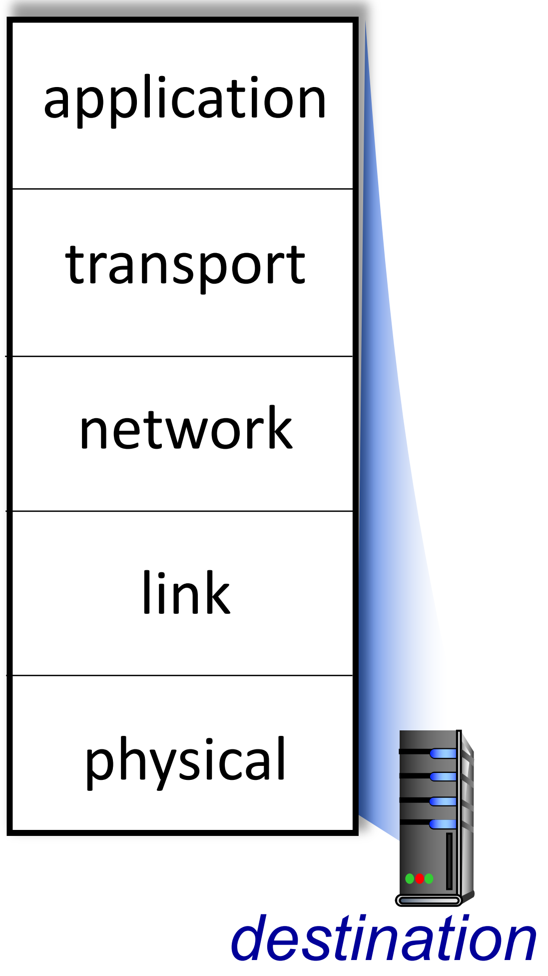
- network-layer protocol encapsulates transport-layer segment [H_t | M] with network layer-layer header H_n to create a network-layer datagram
  - H_n used by network layer protocol to implement its service



destination

Services, Layering and Encapsulation





M

H_t | M

H_n | H_t | M

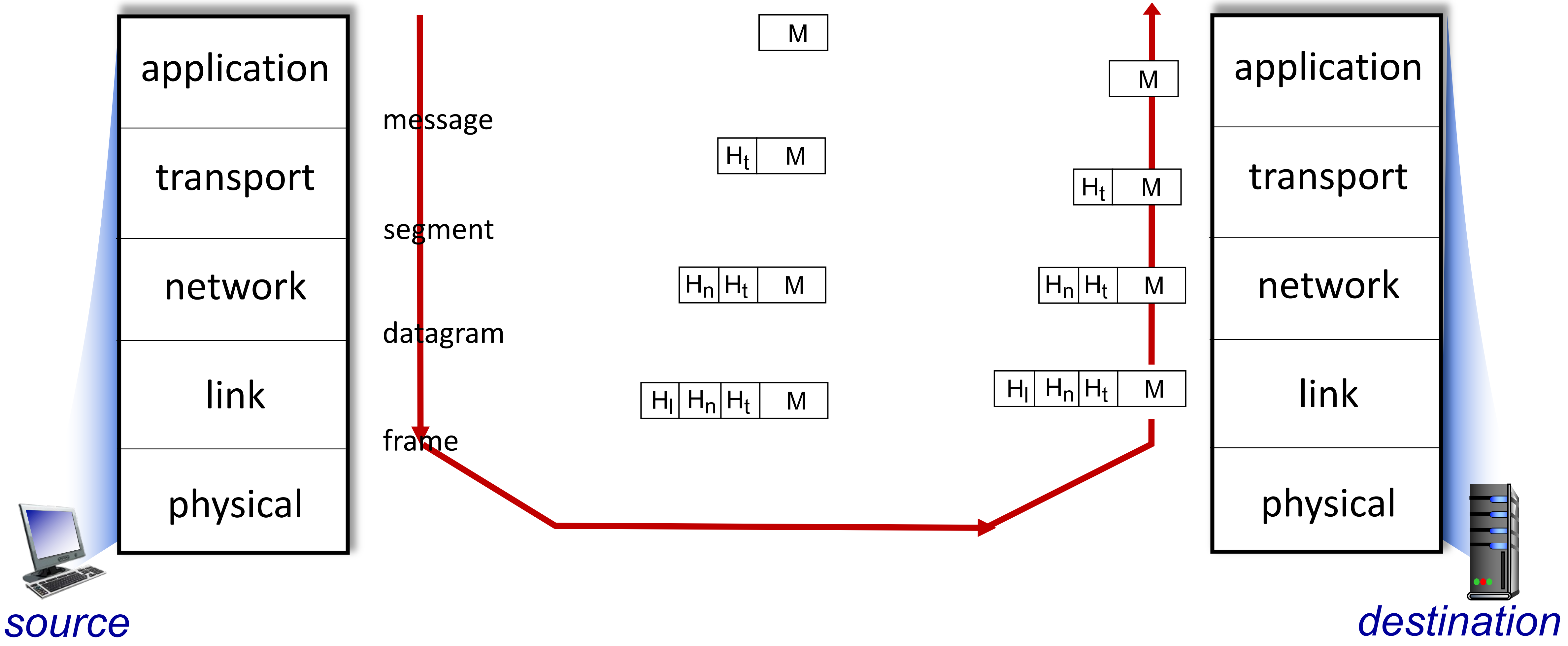
Network-layer protocol transfers transport-layer segment [H_t | M] from one *host* to another, using link layer services

H_l | H_n | H_t | M

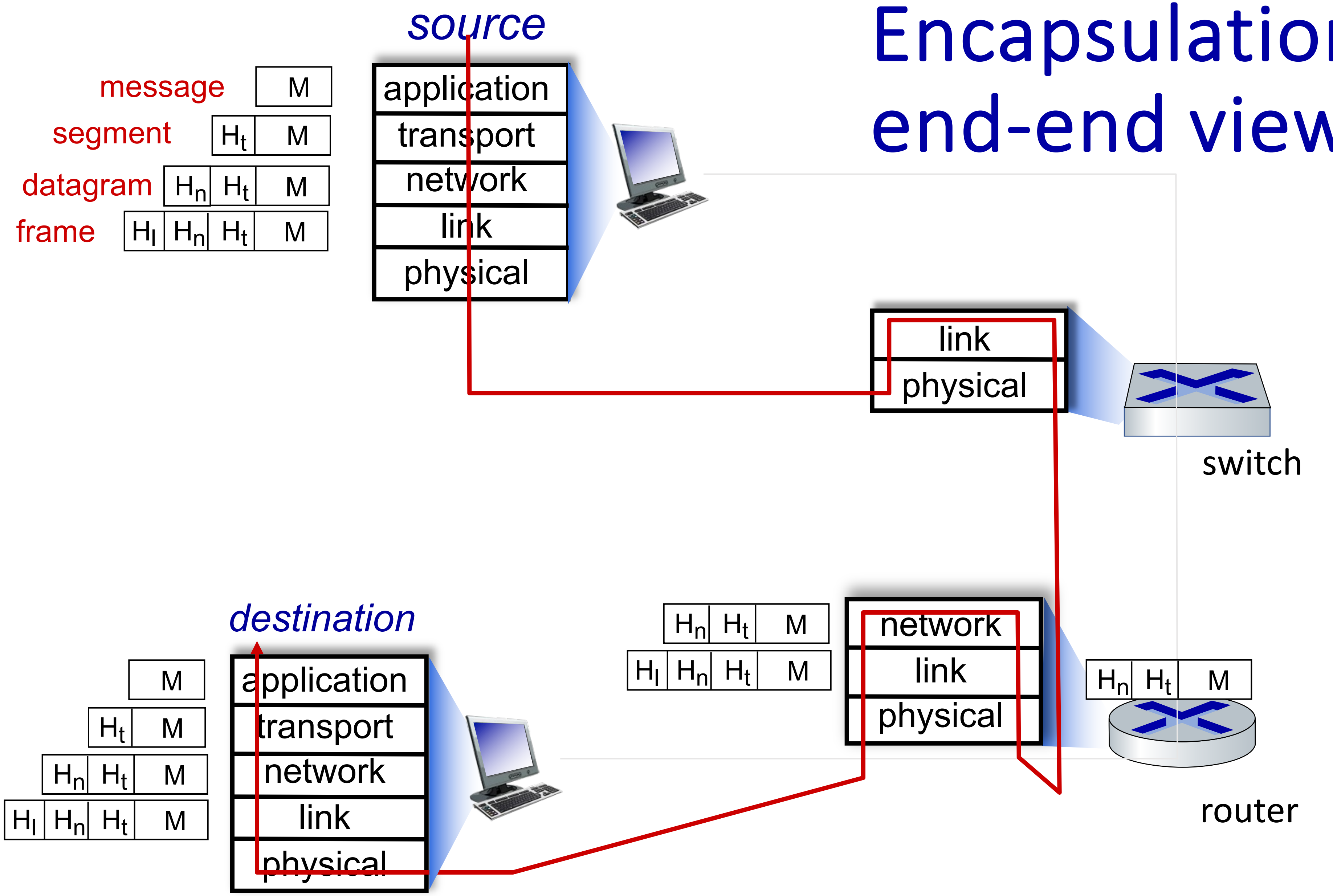
Link-layer protocol transfers datagram [H_n| [H_t |M] from *host* to neighboring host, using network-layer services

- link-layer protocol encapsulates network datagram [H_n| [H_t |M], with link-layer header H_l to create a link-layer frame

Services, Layering and Encapsulation



Encapsulation: an end-end view



Part I: Layering in Internet protocol stack

Applications

… built on …

Reliable (or unreliable) transport

… built on …

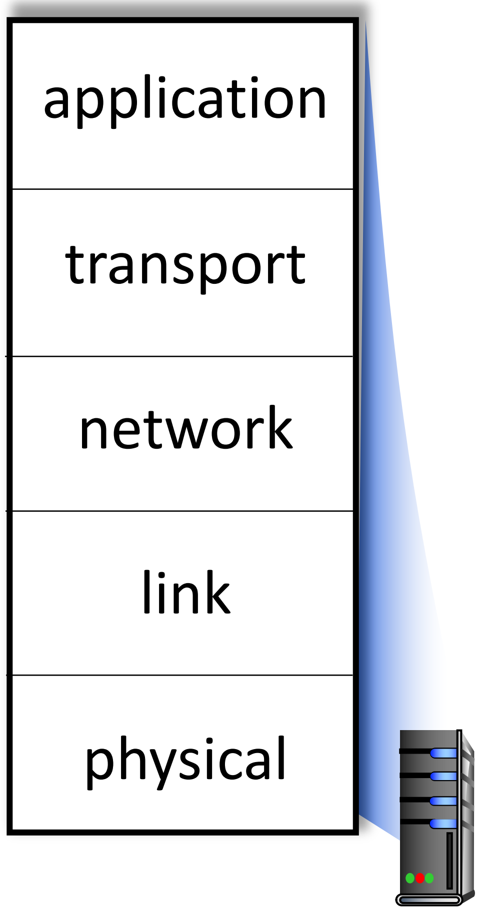
Best-effort global packet delivery

… built on …

Best-effort local packet delivery

… built on …

Physical transfer of bits



Source: Scott Shenker (UC Berkeley): slide 7 at The Future of Networking, and the Past of Protocols
https://www.youtube.com/watch?v=YHeyuD89n1Y&t=111s

Chapter 1: Summary

- what's the Internet?
- network edge
  - hosts, access network
- network core
  - Packet switching versus. circuit switching
- performance: loss, delay, throughput
- what's a protocol?
  - protocol layers, service models