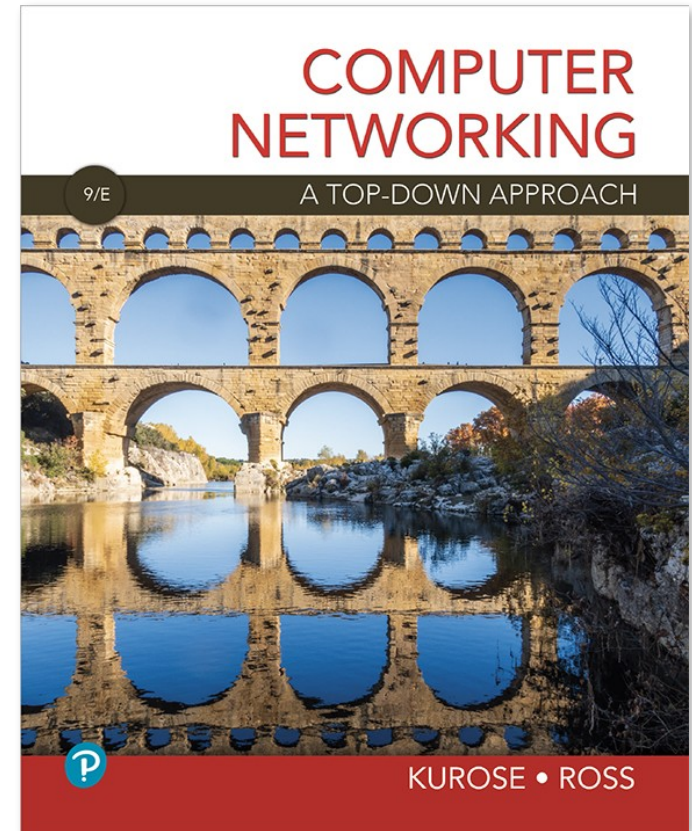# Chapter 1
Introduction

Computer Networking: A Top-Down Approach
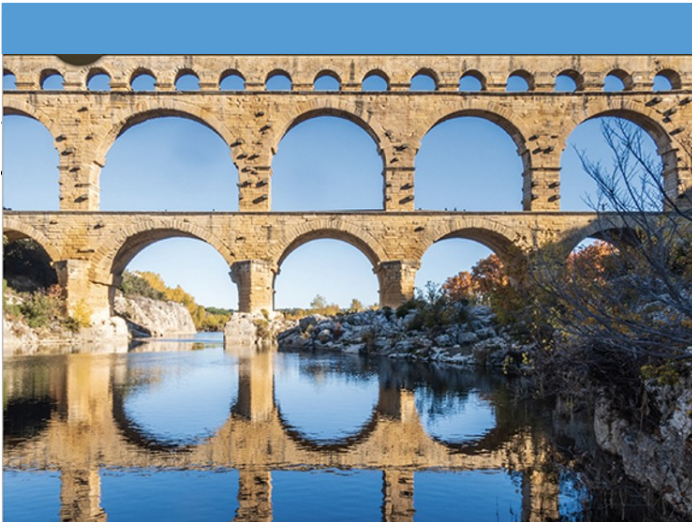
9th edition
Jim Kurose, Keith Ross
Pearson, 2025

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
- Security
- History

The Internet: a "nuts and bolts" view

Billions of connected computing *devices*:

- *hosts = end systems*
- running *network apps* at Internet's "edge"
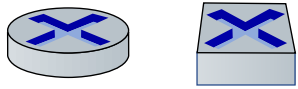
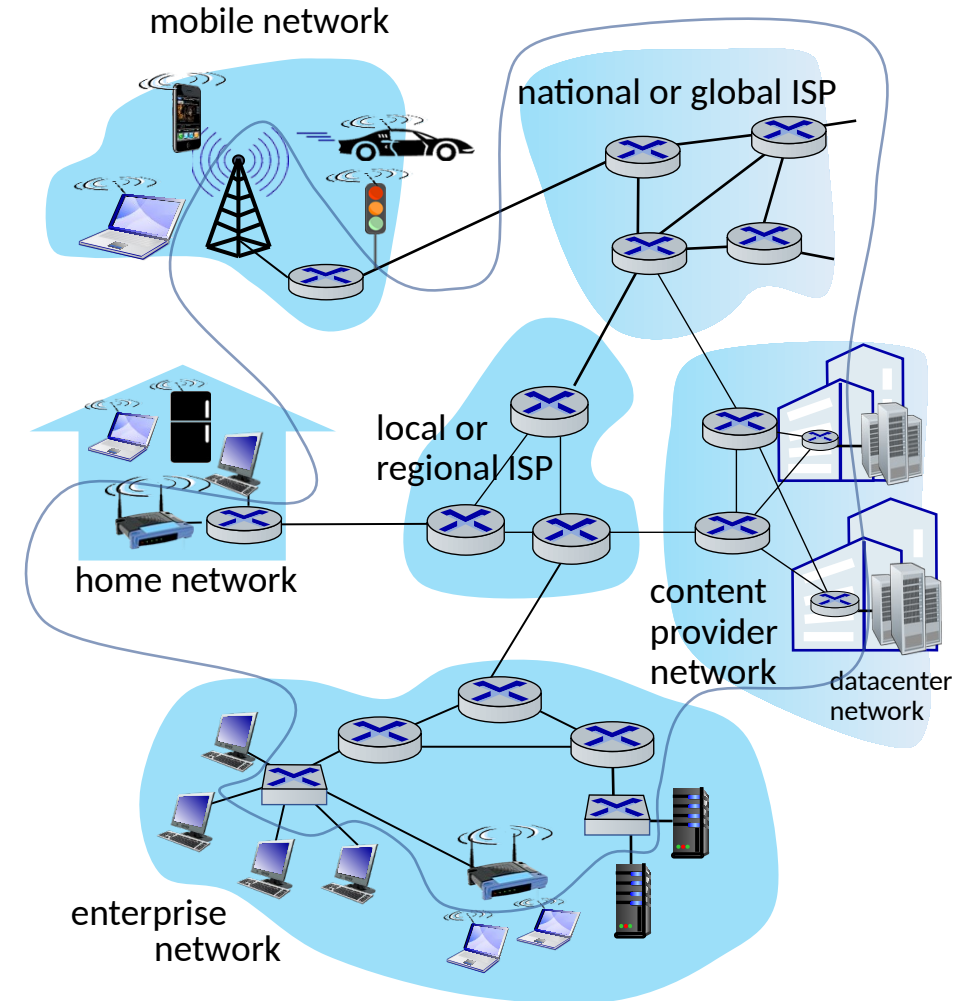*Packet switches*: forward packets (chunks of data)

- *routers, switches*

Communication links

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

Networks

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

mobile network

national or global ISP

local or regional ISP

home network

content provider network

datacenter network

enterprise network

"Fun" Internet-connected devices

Amazon Echo

Internet refrigerator

IP picture frame

Pacemaker & Monitor

Tweet-a-watt: monitor energy use

bikes

Security Camera

Slingbox: remote control cable TV

Web-enabled toaster + weather forecaster

cars

AR devices

scooters

Internet phones

Gaming devices

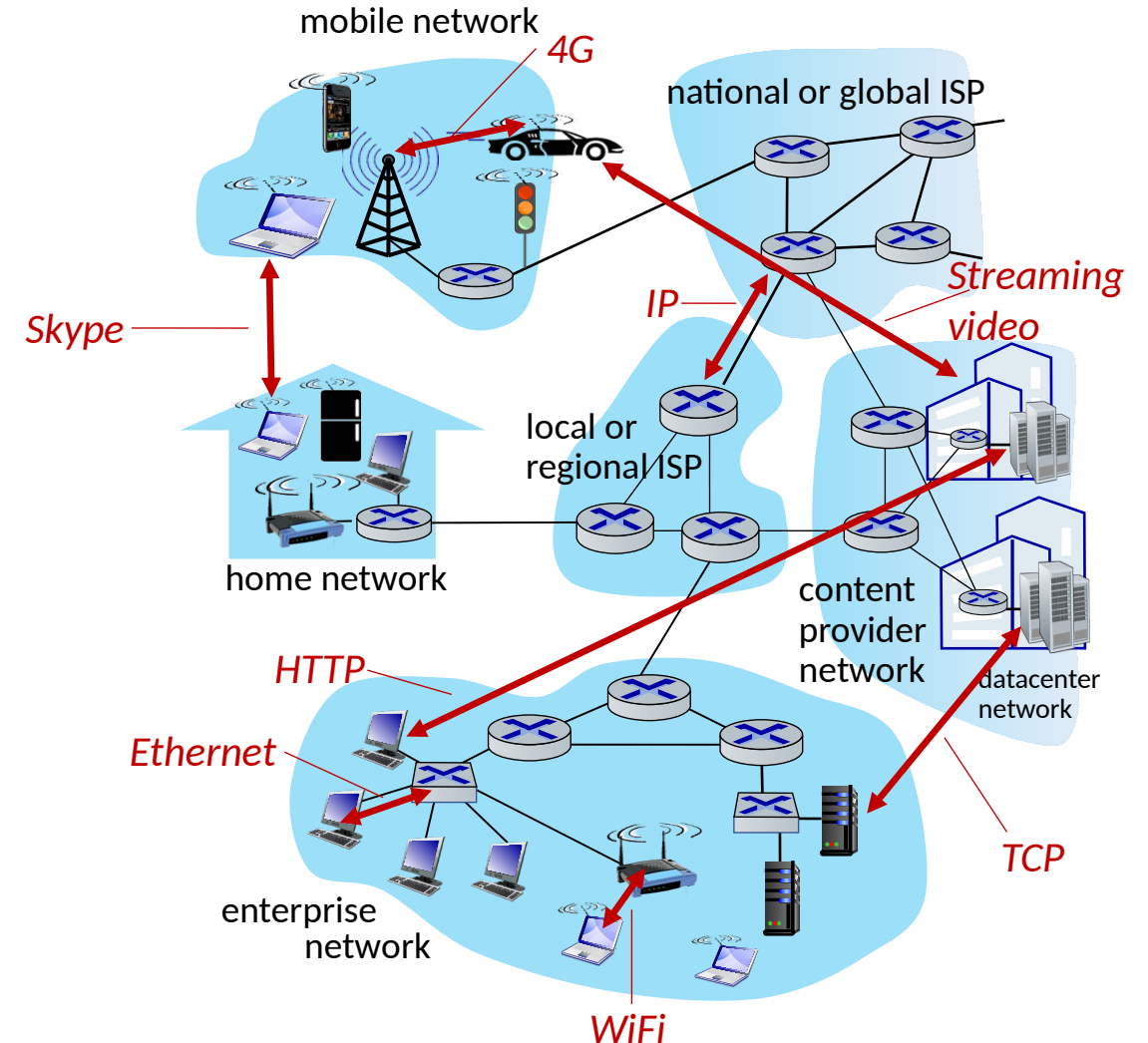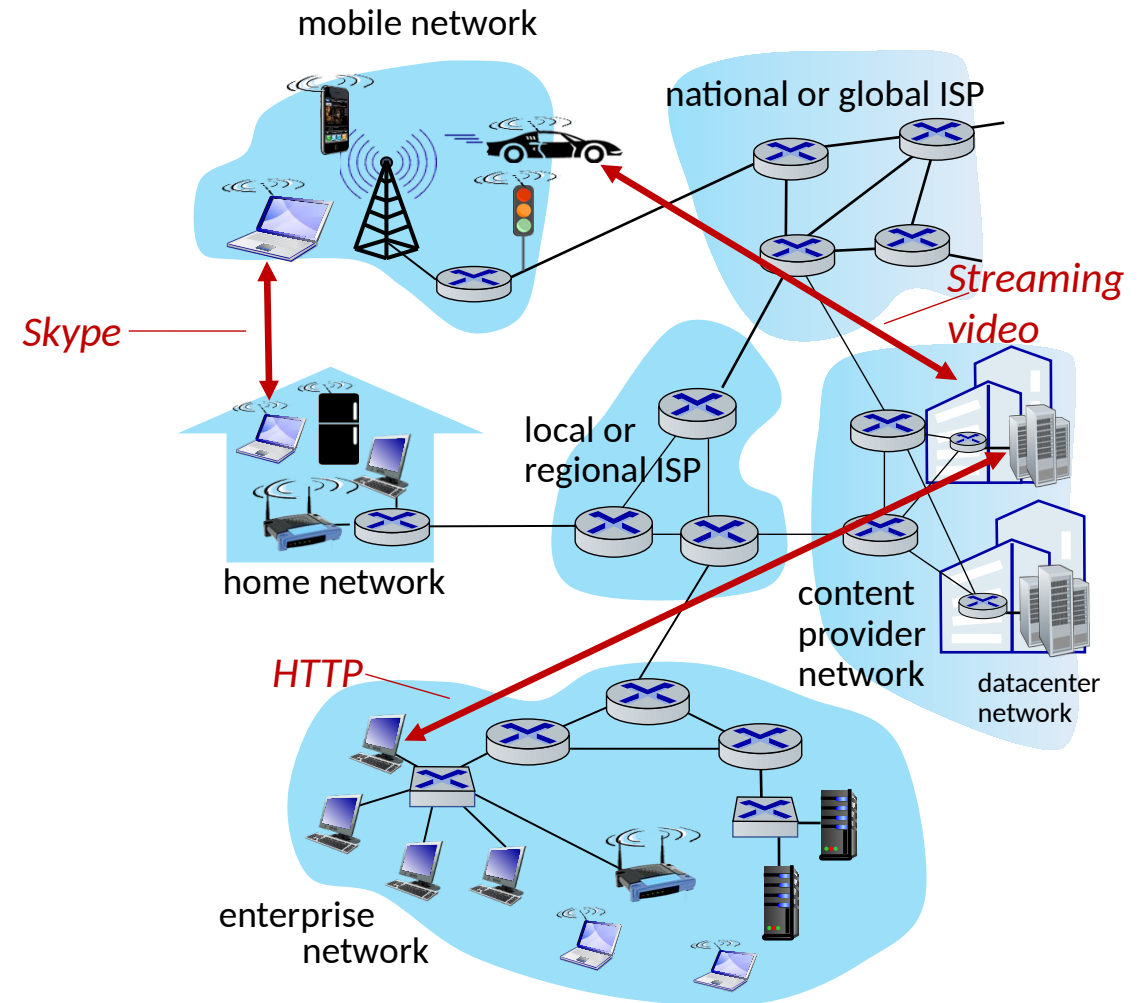sensorized, bed mattress

Fitbit

diapers

Others?

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, Zoom, TCP, IP, WiFi, 4/5G, 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

mobile network

national or global ISP

Skype

Streaming video

local or regional ISP

home network

content provider network

datacenter network

HTTP

enterprise network

What's a protocol?

Human protocols:

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

Rules for:

… specific messages sent

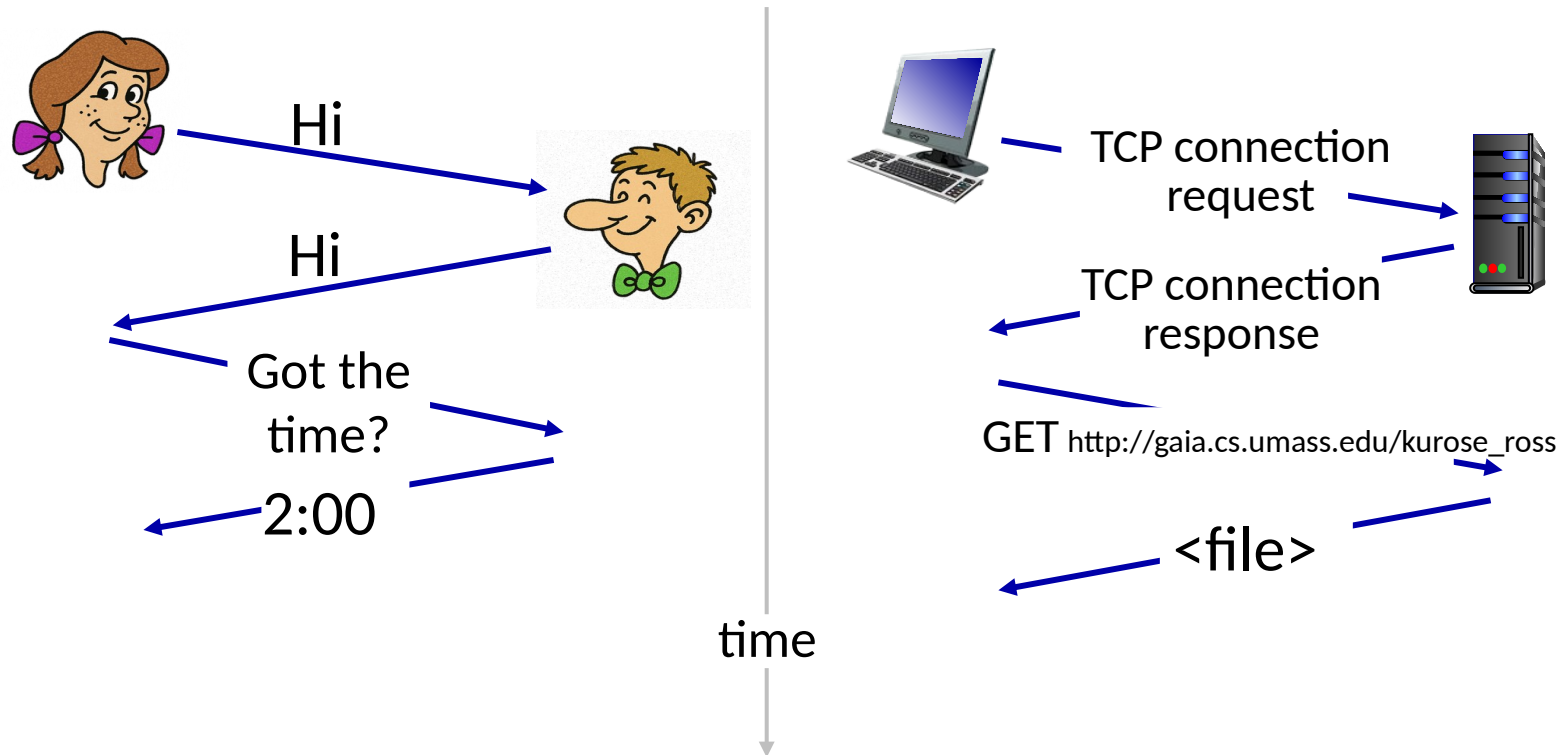… specific actions taken when message received, or other events

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:

Hi

Hi

Got the time?

2:00

TCP connection request

TCP connection response

GET http://gaia.cs.umass.edu/kurose_ross

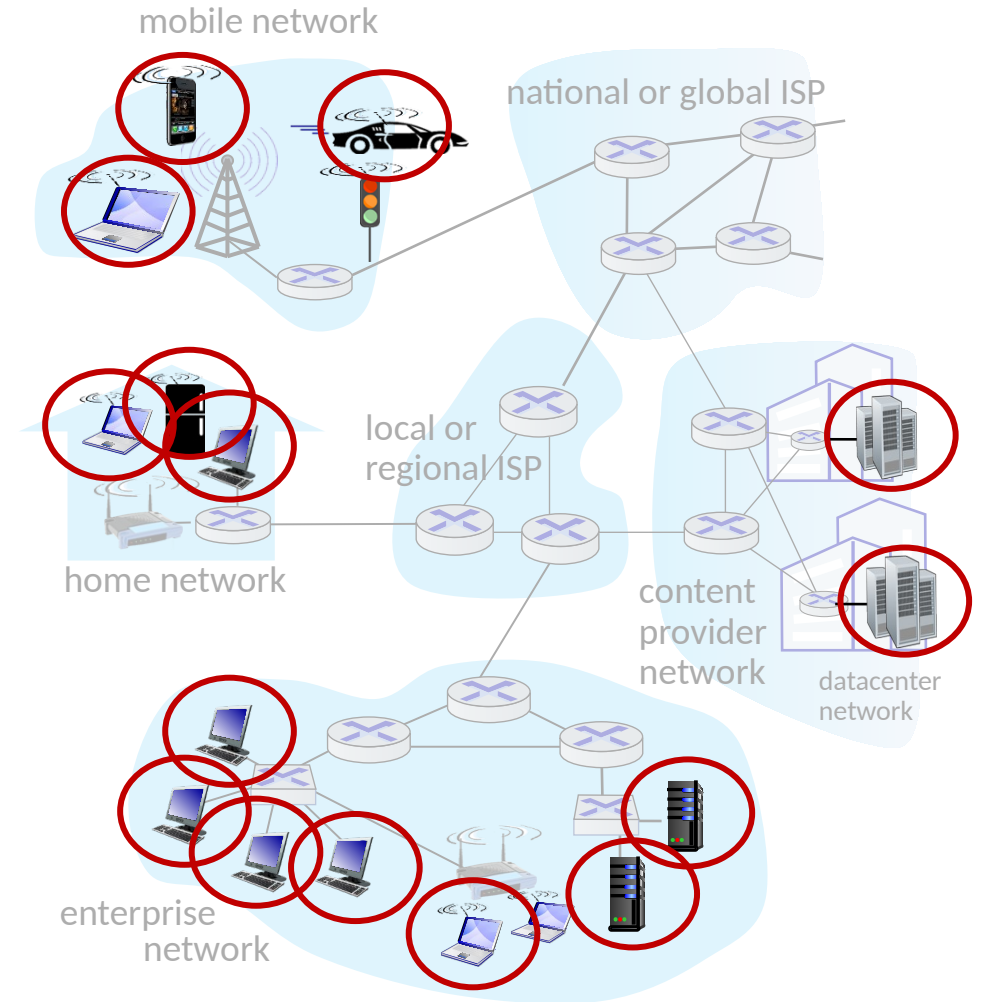<file>

time

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

mobile network

national or global ISP

local or regional ISP

home network

content provider network

datacenter network

enterprise network

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

mobile network

national or global ISP

local or regional ISP

home network

content provider network
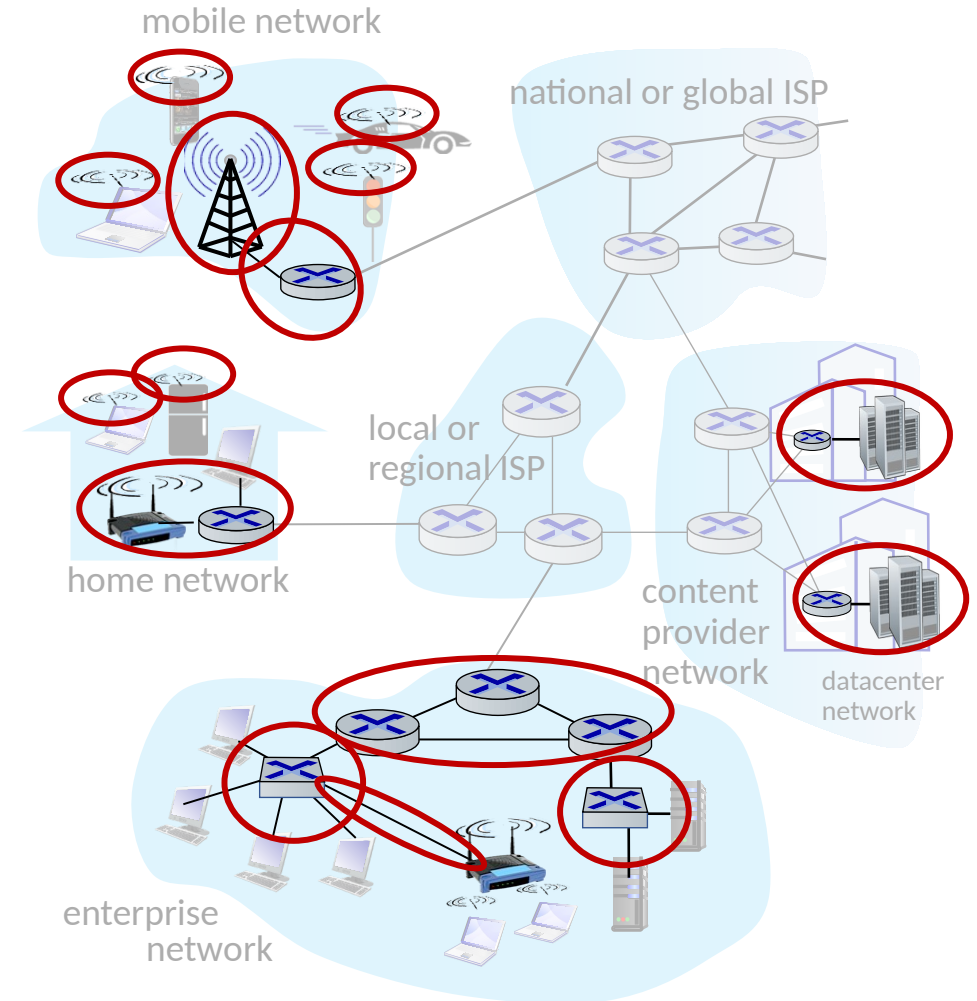
datacenter network

enterprise network

A closer look at Internet structure
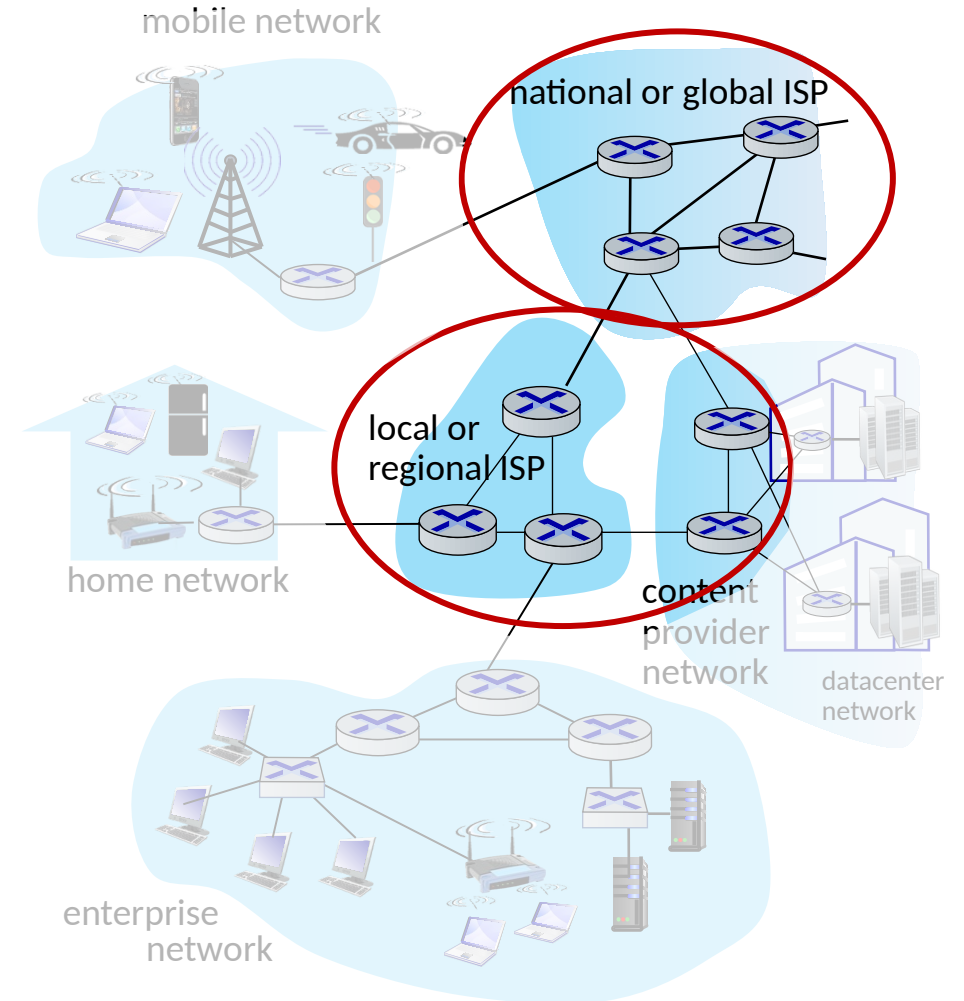
Network edge:
- hosts: clients and servers
- servers often in data centers

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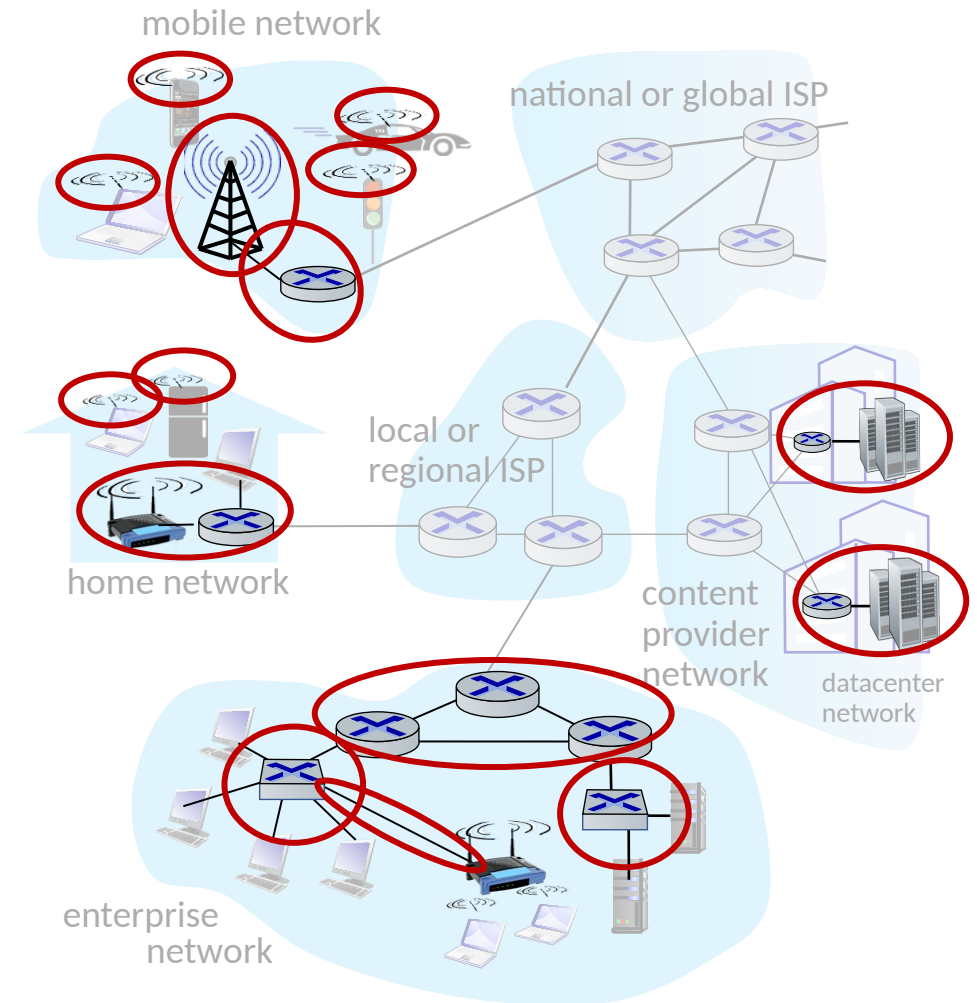- wired, wireless communication links

Network core:
- interconnected routers
- network of networks

mobile network

national or global ISP

local or regional ISP

home network

content provider network

datacenter network

enterprise network

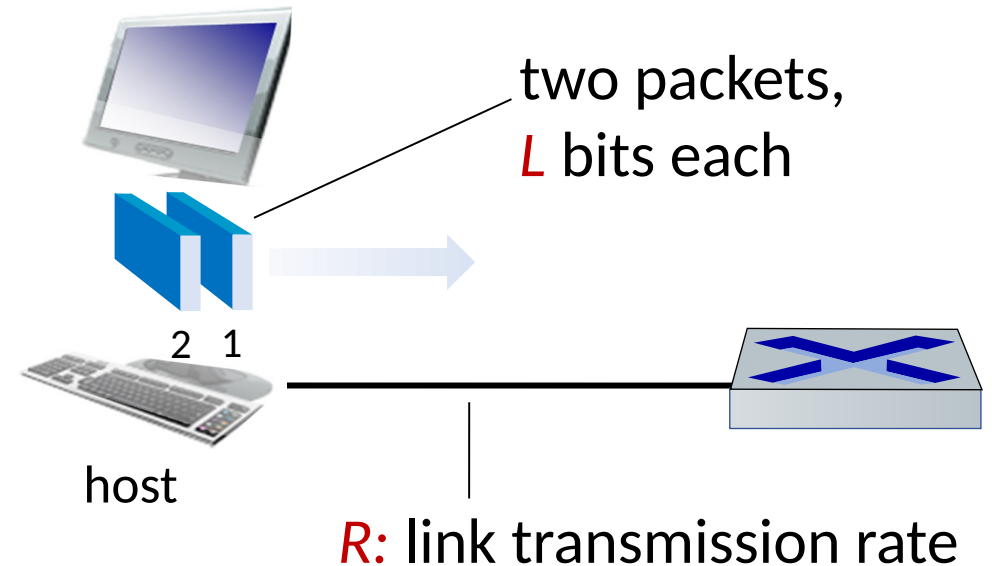# Access networks and physical media

Q: How to connect end systems to edge router?

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

mobile network

national or global ISP

local or
regional ISP

home network

content
provider
network

datacenter
network

enterprise
network

Host: sends *packets* of data
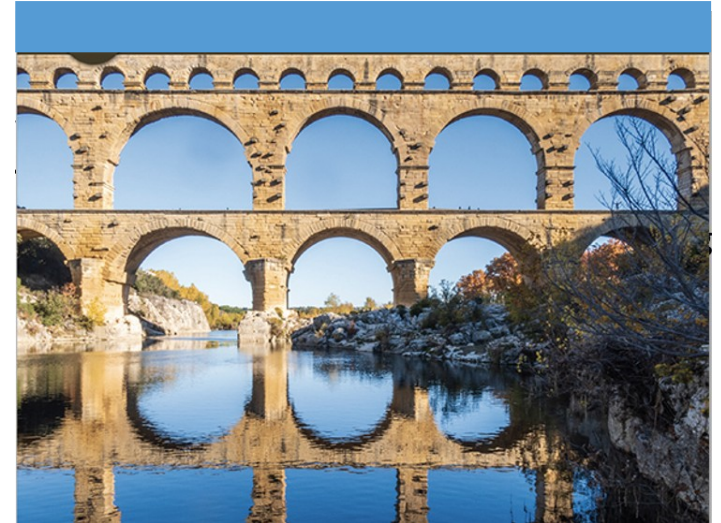
host sending function:

- takes application message

- breaks into smaller chunks, known as *packets*, of length *L* bits

- transmits packet into access network at *transmission rate R*
  - link transmission rate, aka link *capacity, aka link bandwidth*

two packets, *L* bits each

2 1

host

R: link transmission rate

$$
\begin{matrix}
\text{packet} \\
\text{transmission} \\
\text{delay}
\end{matrix}
\;=\;
\begin{matrix}
\text{time needed to} \\
\text{transmit } L\text{-bit} \\
\text{packet into link}
\end{matrix}
\;=\;
\dfrac{L \text{ (bits)}}{R \text{ (bits/sec)}}
$$

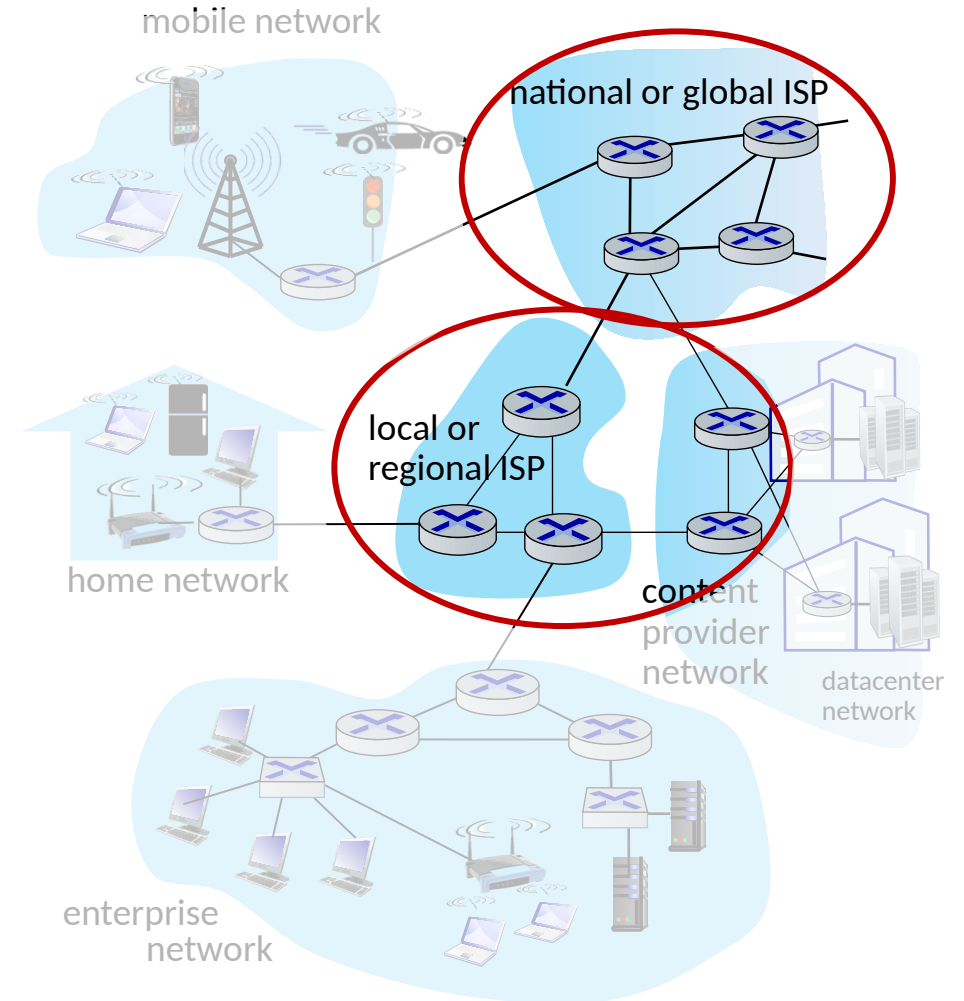# Chapter 1: roadmap

- 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

mobile network

national or global ISP

local or regional ISP

home network

content provider network

datacenter network

enterprise network

Two key network-core functions

Forwarding:
- aka "switching"
- *local* action: move arriving packets from router's input link to appropriate router output link
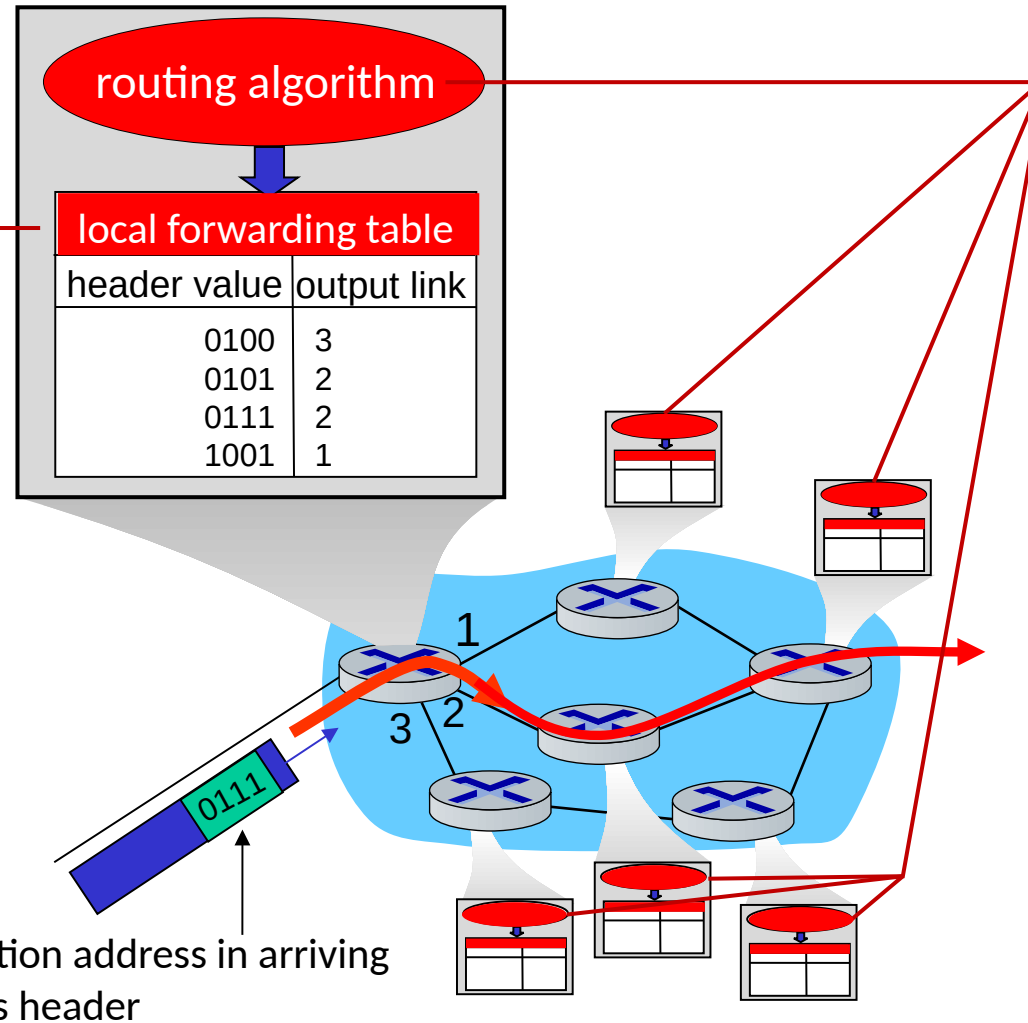
Routing:
- *global* action: determine source-destination paths taken by packets
- routing algorithms

routing algorithm

local forwarding table

header value	output link
0100	3
0101	2
0111	2
1001	1

1

3 2

0111

destination address in arriving packet's header

routing

forwarding

forwarding

forwarding

Packet-switching: store-and-forward

L bits per packet

3 2 1

source

R bps

R bps

destination
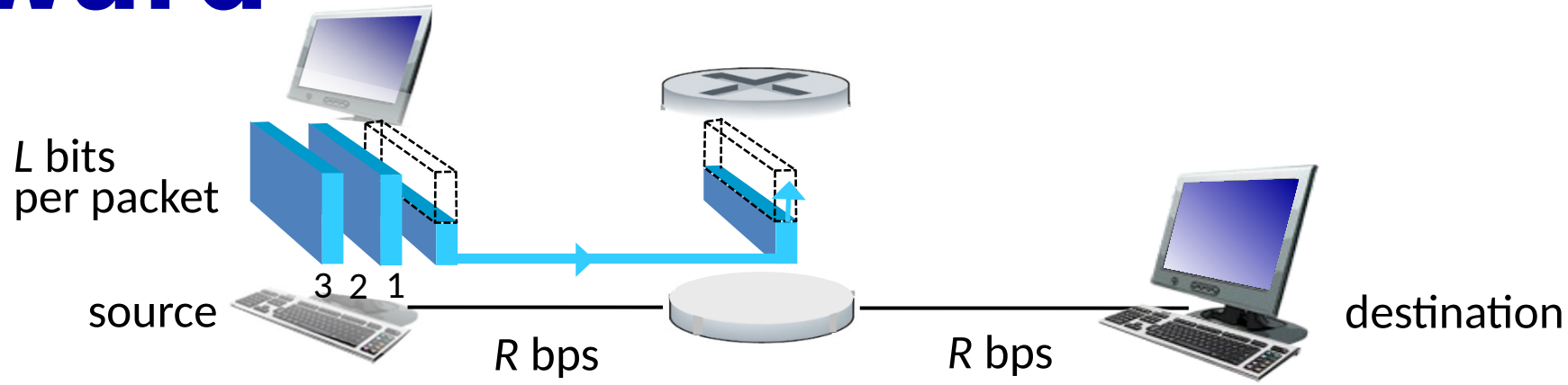
- **packet transmission delay:** takes *L/R* seconds to transmit (push out) *L*-bit packet into link at *R* bps

- *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
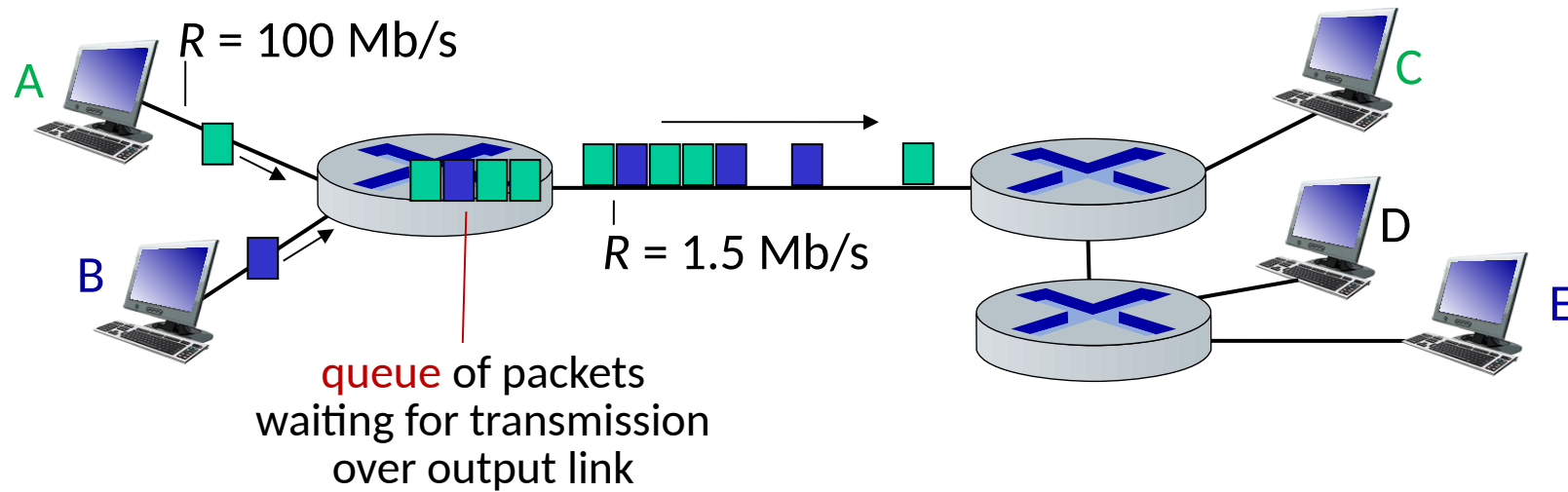- one-hop transmission delay = 0.1 msec

Packet-switching: queueing

R = 100 Mb/s

A

B

queue of packets
waiting for transmission
over output link

R = 1.5 Mb/s

C

D

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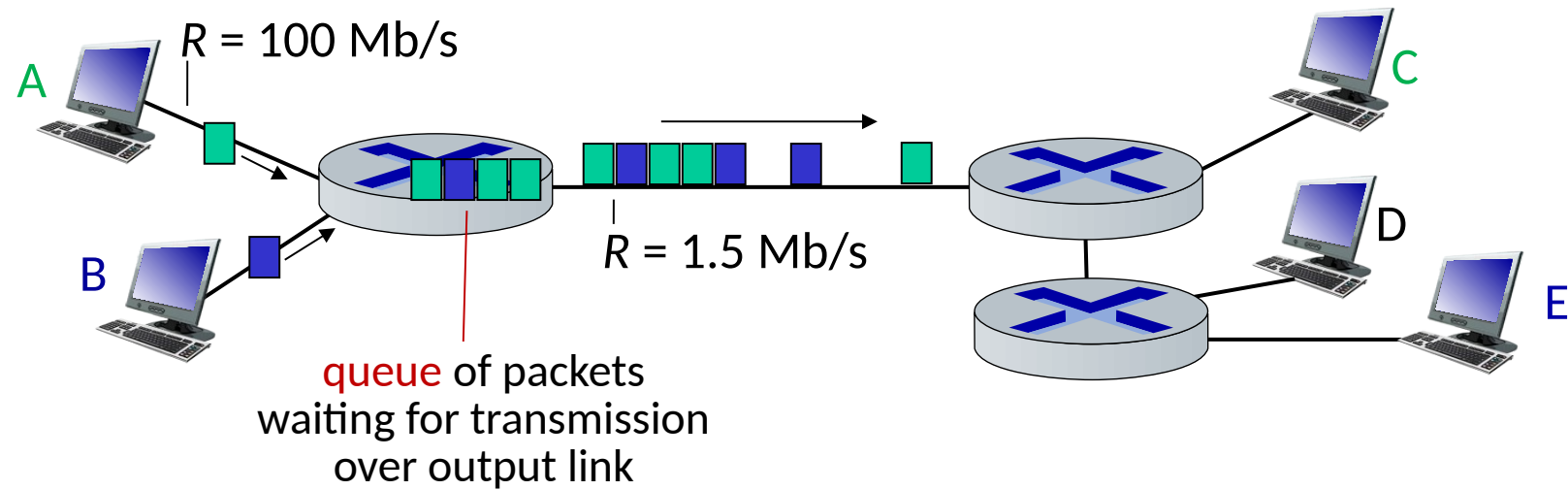
Queueing occurs when work arrives faster than it can be serviced:

Packet-switching: queueing

R = 100 Mb/s

A

C

R = 1.5 Mb/s

B

D

E

queue of packets
waiting for transmission
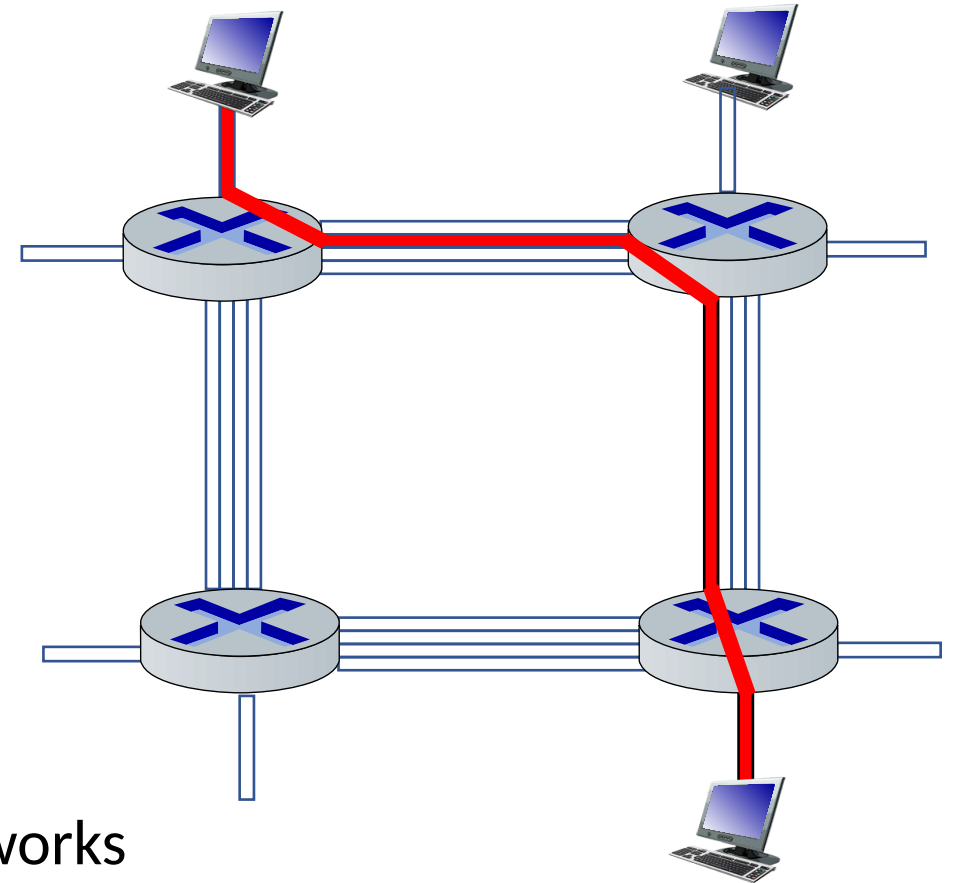over output link

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

Circuit switching: FDM and TDM

Frequency Division Multiplexing (FDM)

- optical, electromagnetic frequencies divided into (narrow) frequency bands
- each call allocated its own band, can transmit at max rate of that narrow band
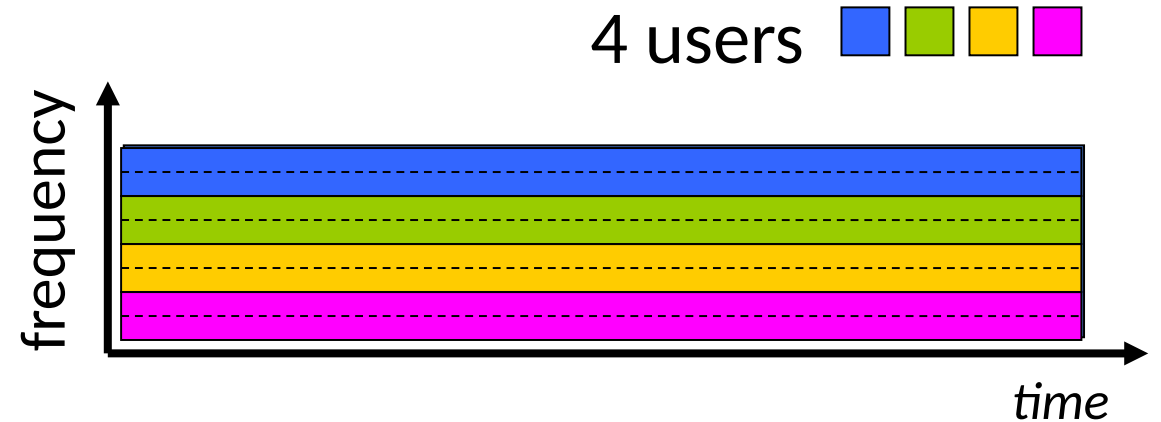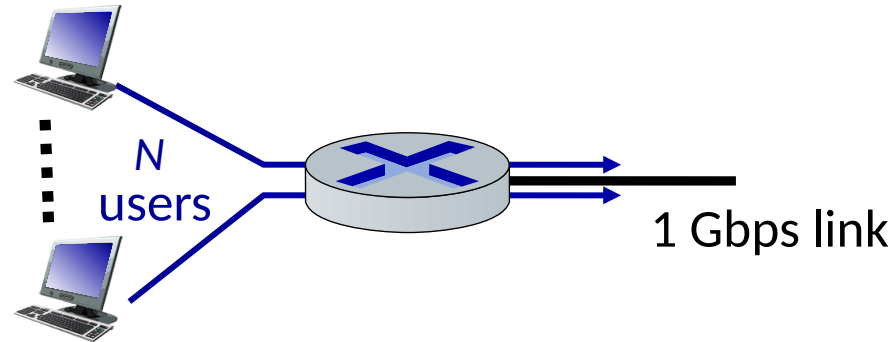
Time Division Multiplexing (TDM)

- time divided into slots
- each call allocated periodic slot(s), can transmit at maximum rate of (wider) frequency band (only) during its time slot(s)
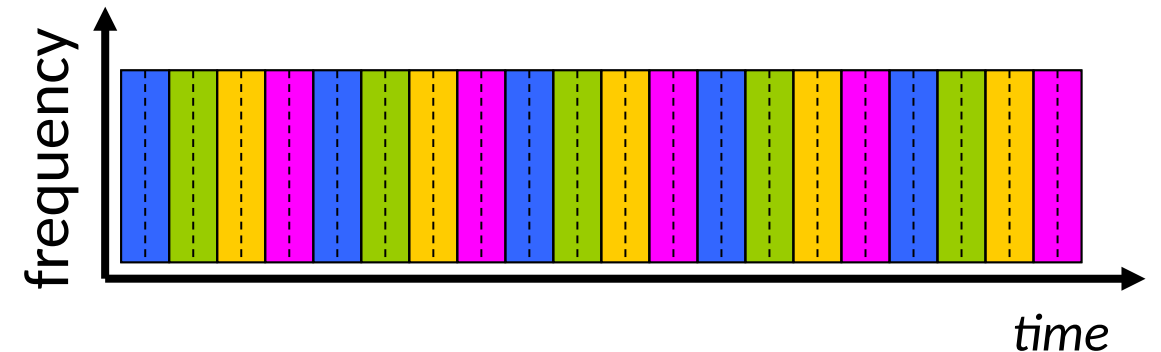
4 users

Packet switching versus circuit switching

example:

- 1 Gb/s link

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

N users

1 Gbps link

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: HW 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
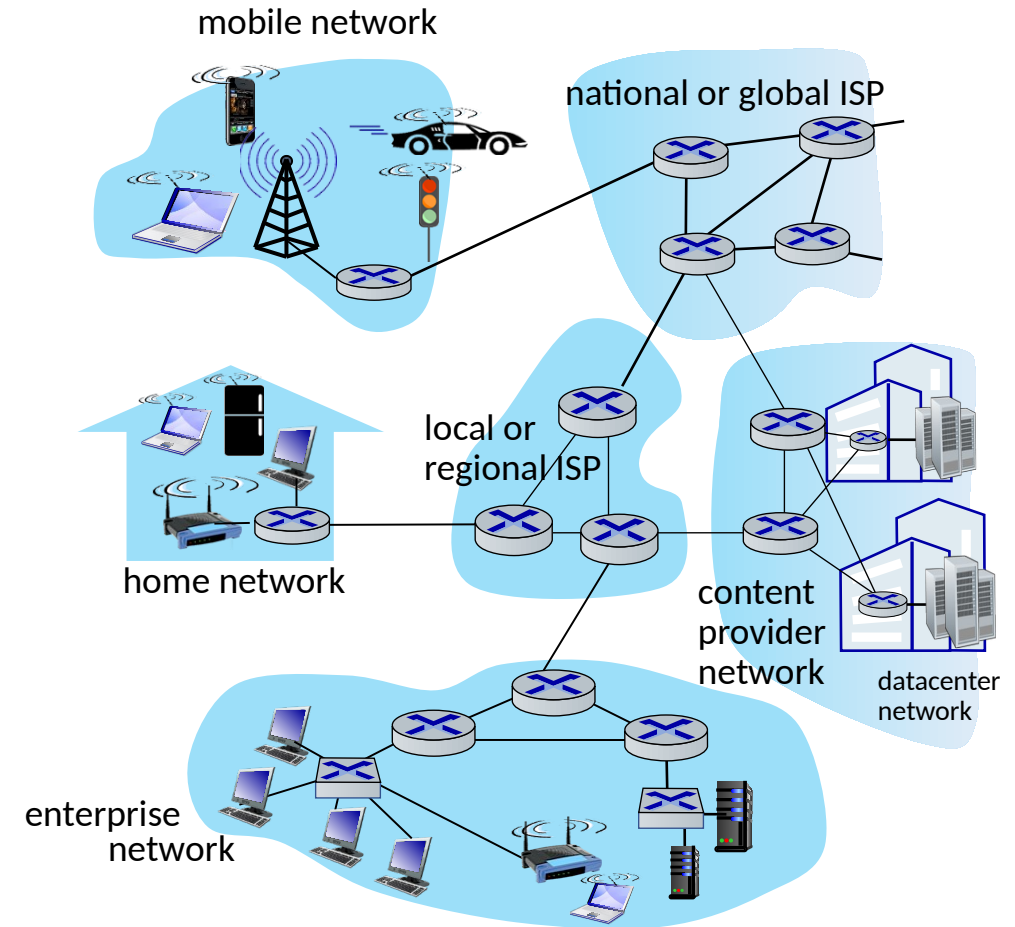
Packet switching versus circuit switching

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)?

Internet structure: a "network of networks"

- hosts connect to Internet via access Internet Service Providers (ISPs)

- access ISPs in turn must be interconnected
  - so that *any* two hosts *(anywhere!)* can send packets to each other

- resulting network of networks is very complex
  - evolution driven by economics, national policies

mobile network

national or global ISP

local or regional ISP

home network

content provider network

datacenter network

enterprise network

Let's take a stepwise approach to describe current Internet structure
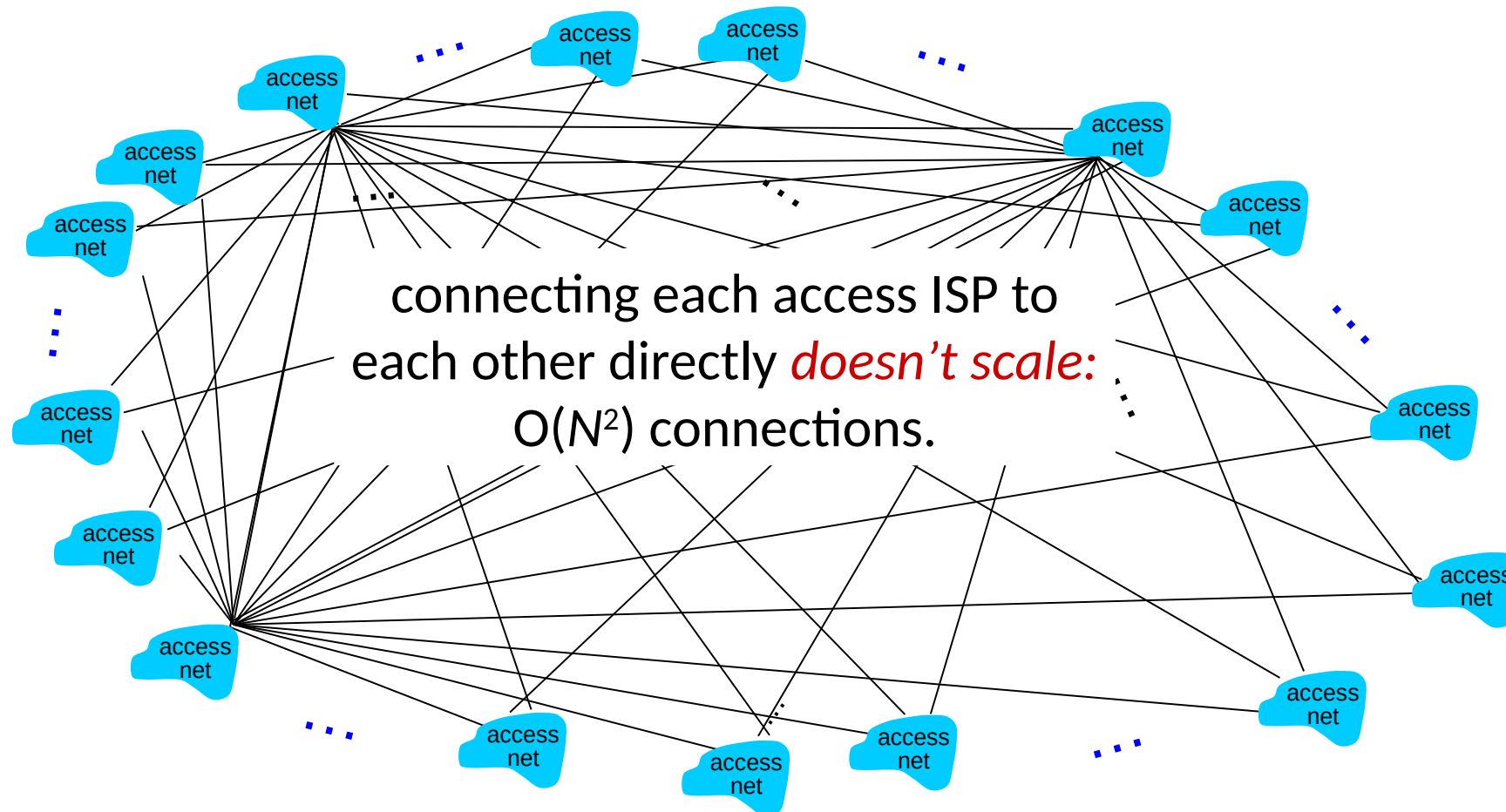
Internet structure: a "network of networks"
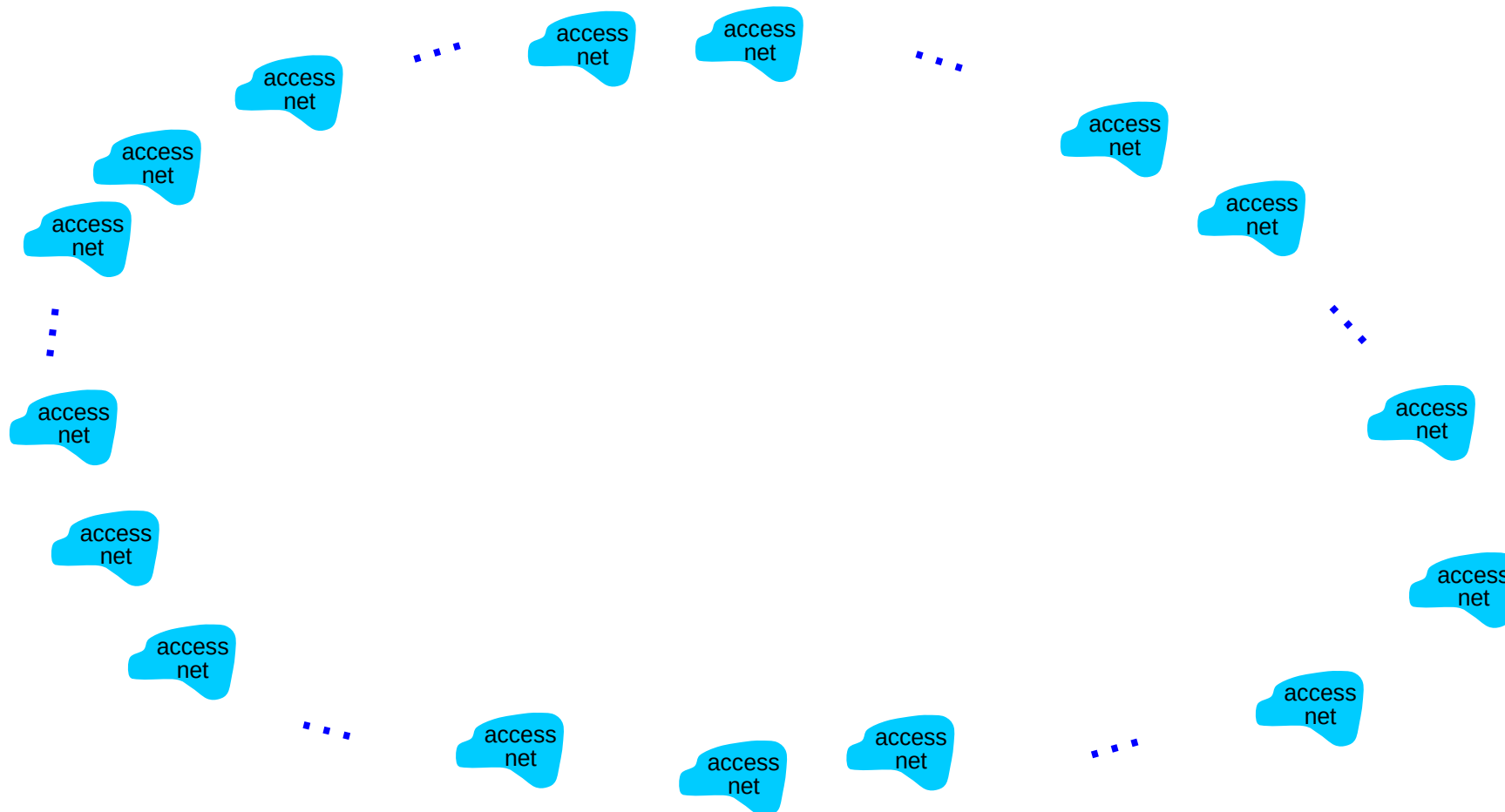
Question: given *millions* of access ISPs, how to connect them together?

Internet structure: a "network of networks"

Question: given *millions* of access ISPs, how to connect them together?

connecting each access ISP to each other directly *doesn't scale:* O(N^2) connections.

Internet structure: a "network of networks"

Option: connect each access ISP to one global transit ISP?
Customer and *provider* ISPs have economic agreement.
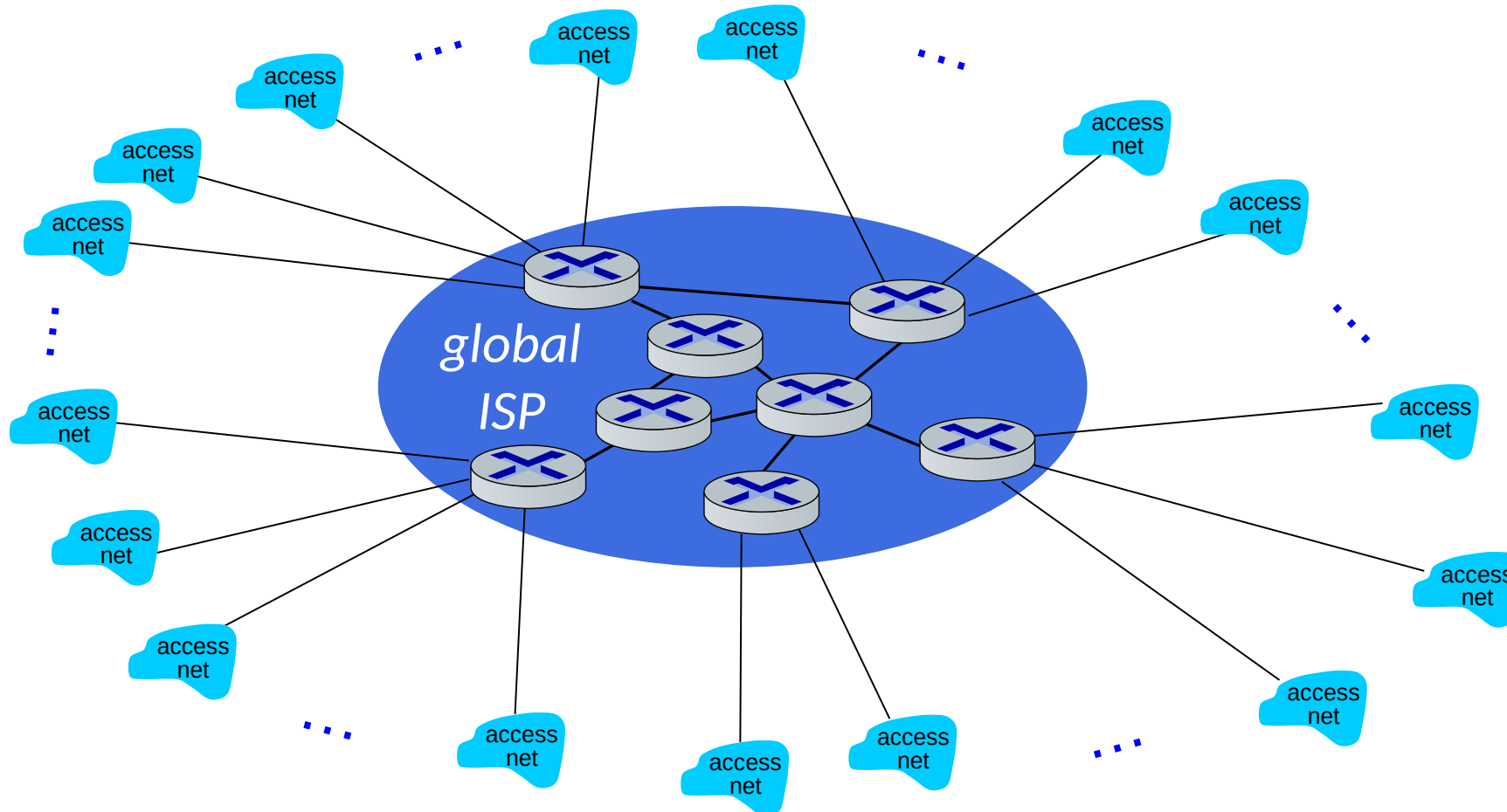
Internet structure: a "network of networks"

But if one global ISP is viable business, there will be competitors ....
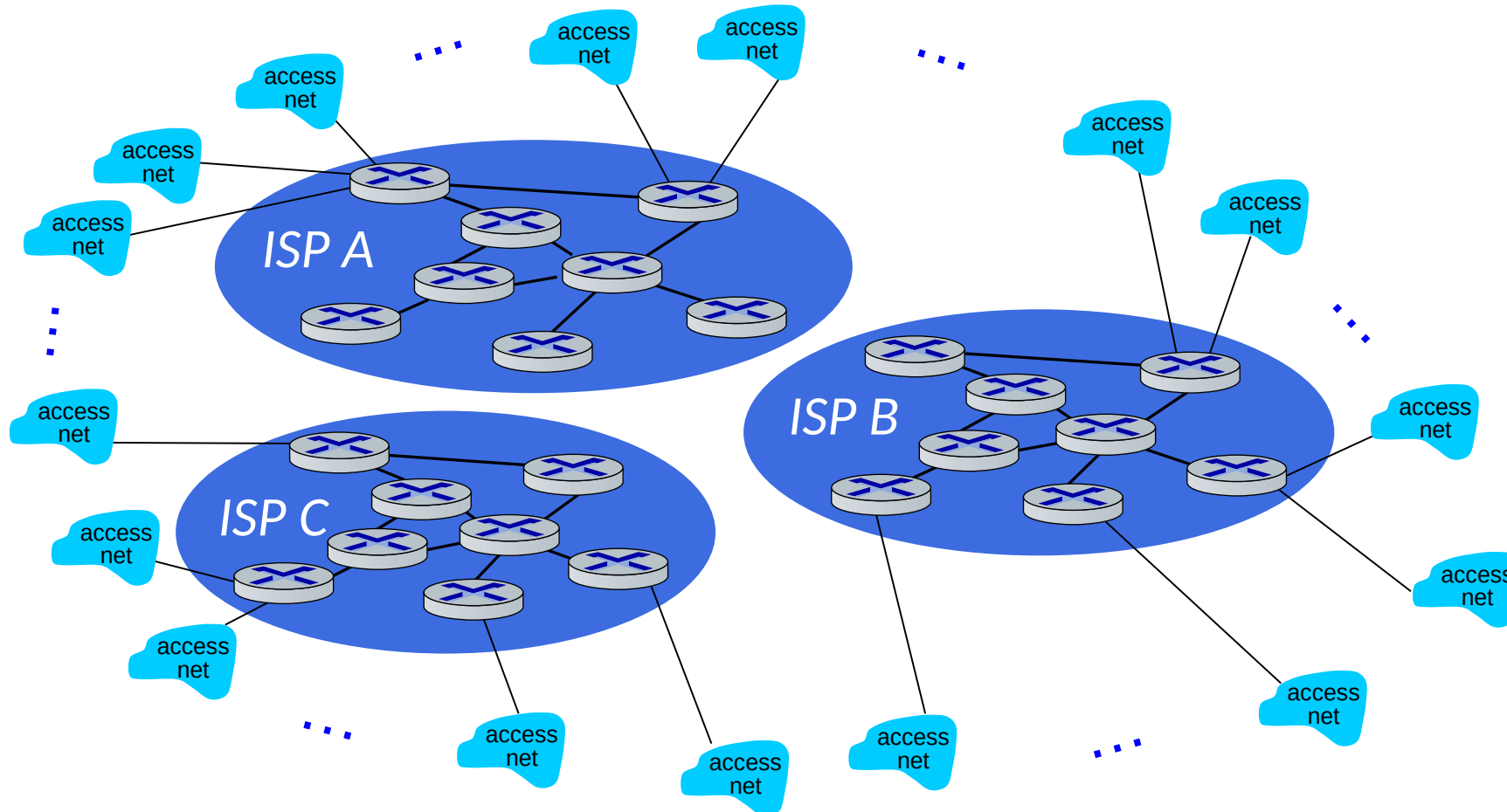
Internet structure: a "network of networks"

But if one global ISP is viable business, there will be competitors .... who will want to be connected

Internet exchange point

IXP

access net

ISP A

ISP B

IXP

ISP C

peering link
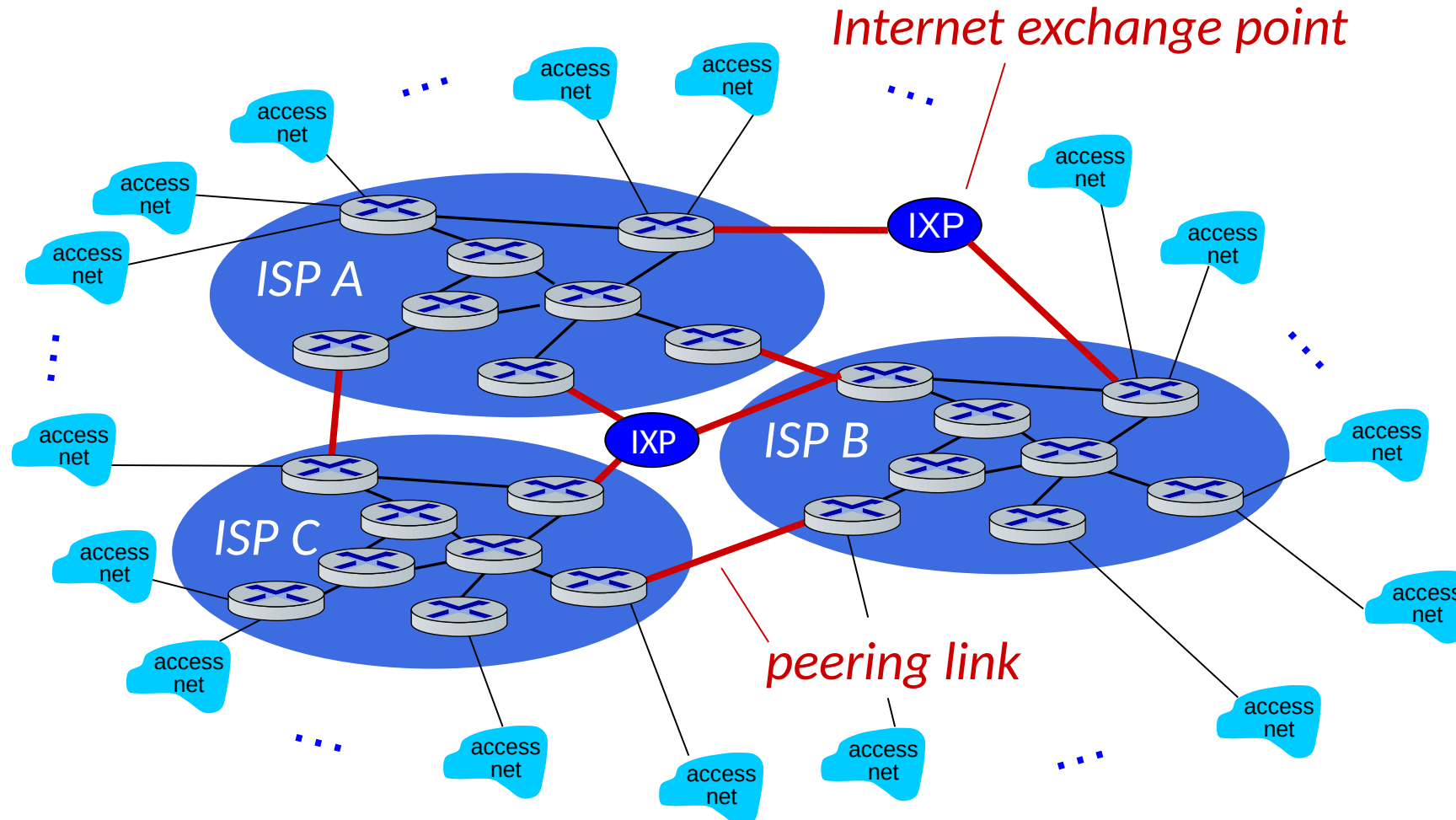
Internet structure: a "network of networks"

… and regional networks may arise to connect access nets to ISPs
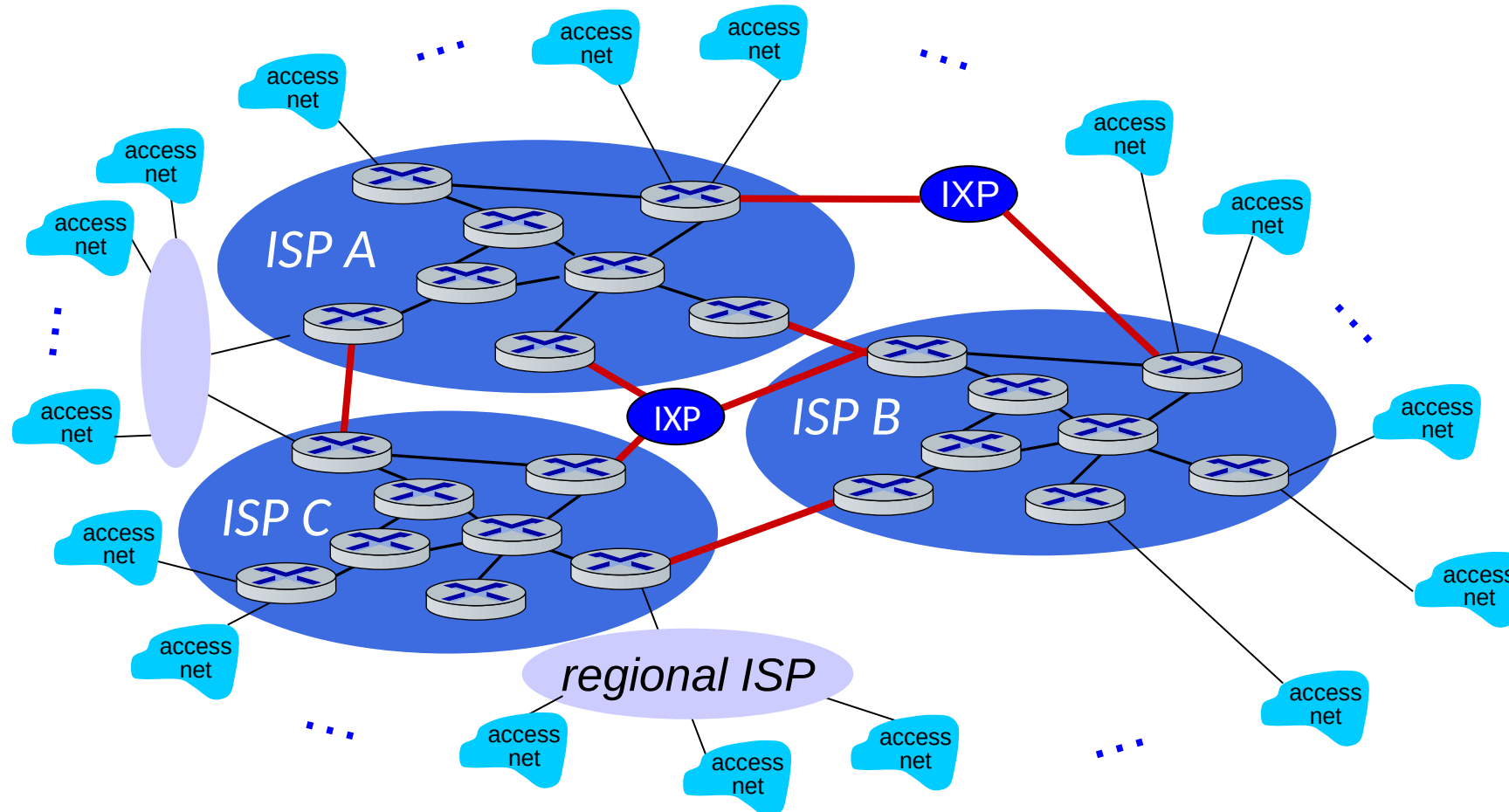
Internet structure: a "network of networks"

... and content provider networks (e.g., Google, Microsoft, Akamai) may run their own network, to bring services, content close to end users
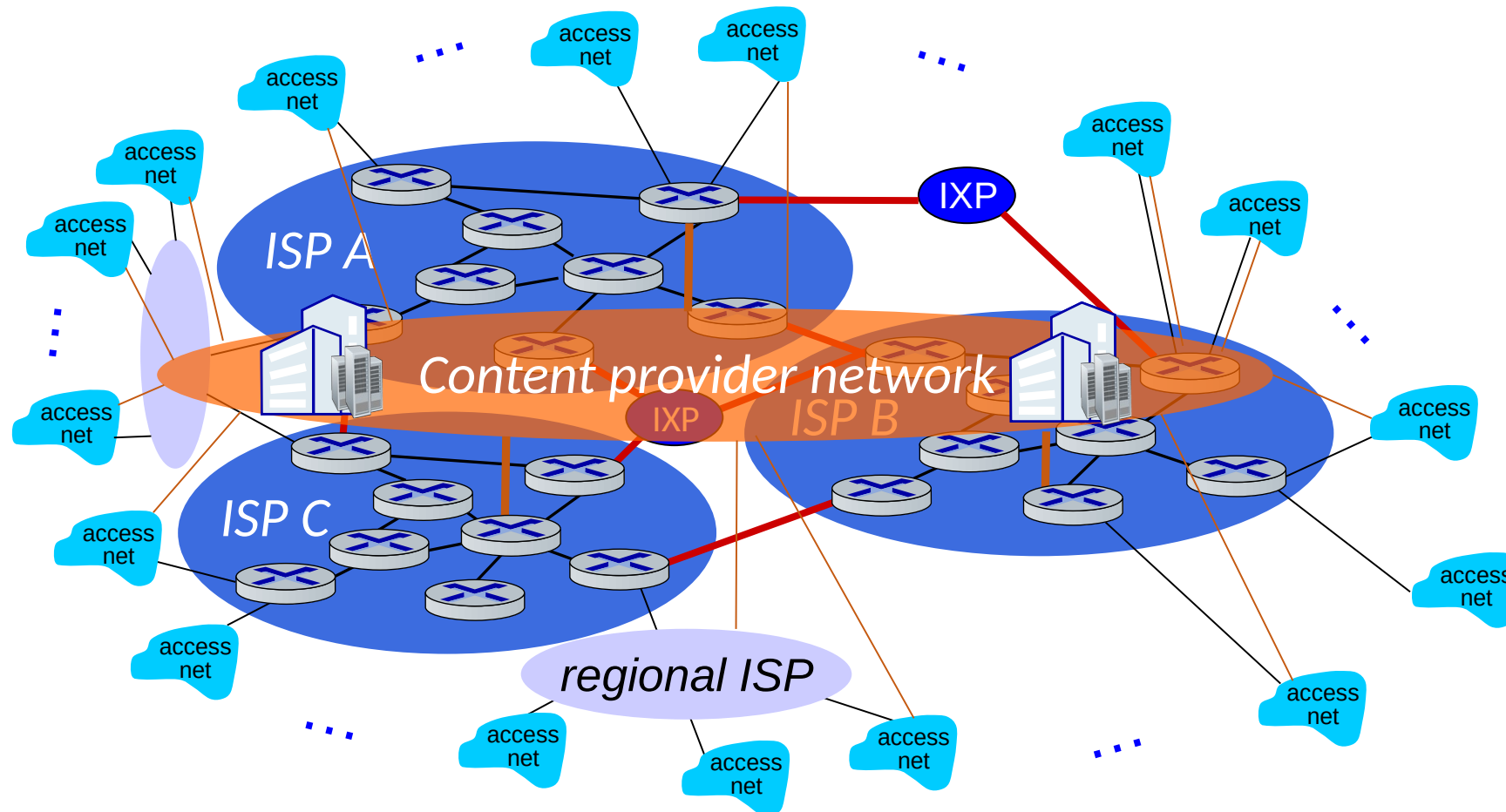
Internet structure: a "network of networks"
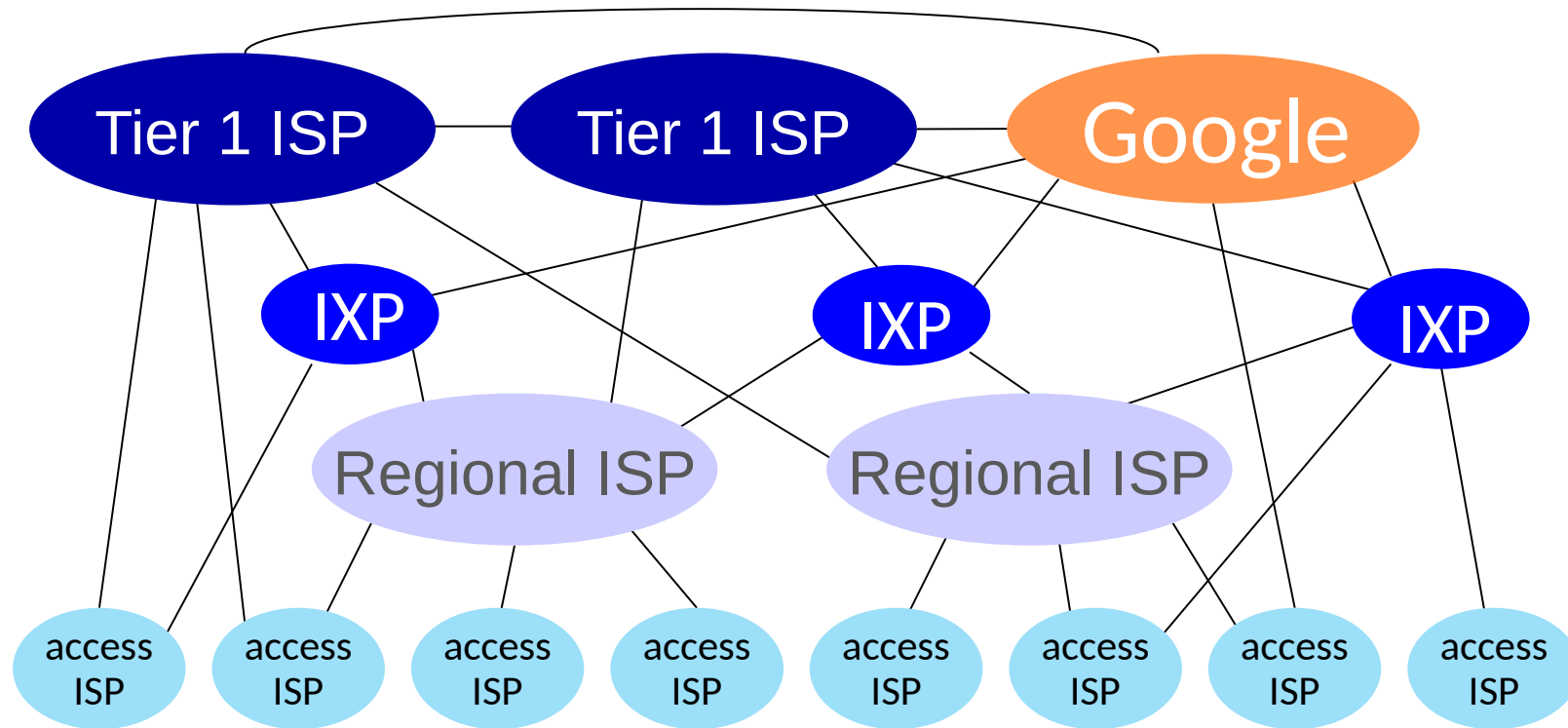
At "center": small # of well-connected large networks

- "tier-1" commercial ISPs (e.g., Level 3, Sprint, AT&T, NTT), national & international coverage
- content provider networks (e.g., Google, Facebook): private network that connects its data centers to Internet, often bypassing tier-1, regional ISPs

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 being transmitted (transmission delay)

A

B

packets in buffers (queueing delay)

free (available) buffers: arriving packets dropped (loss) if no free buffers
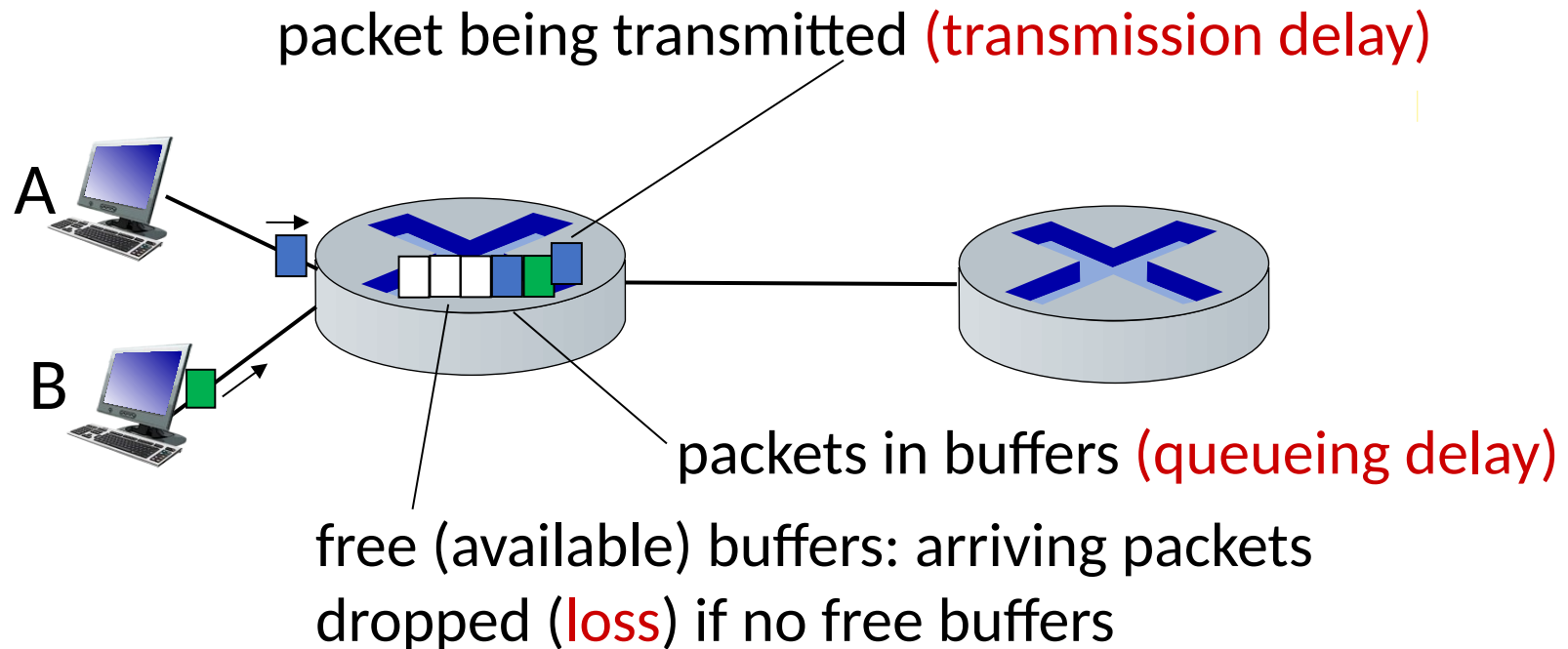
Packet delay: four sources
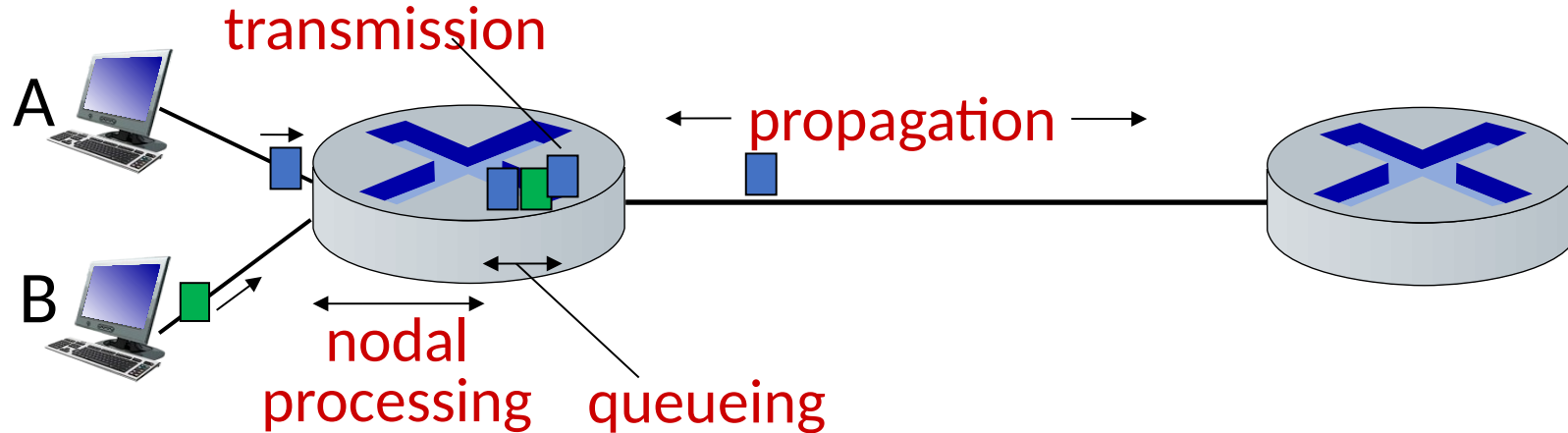
$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

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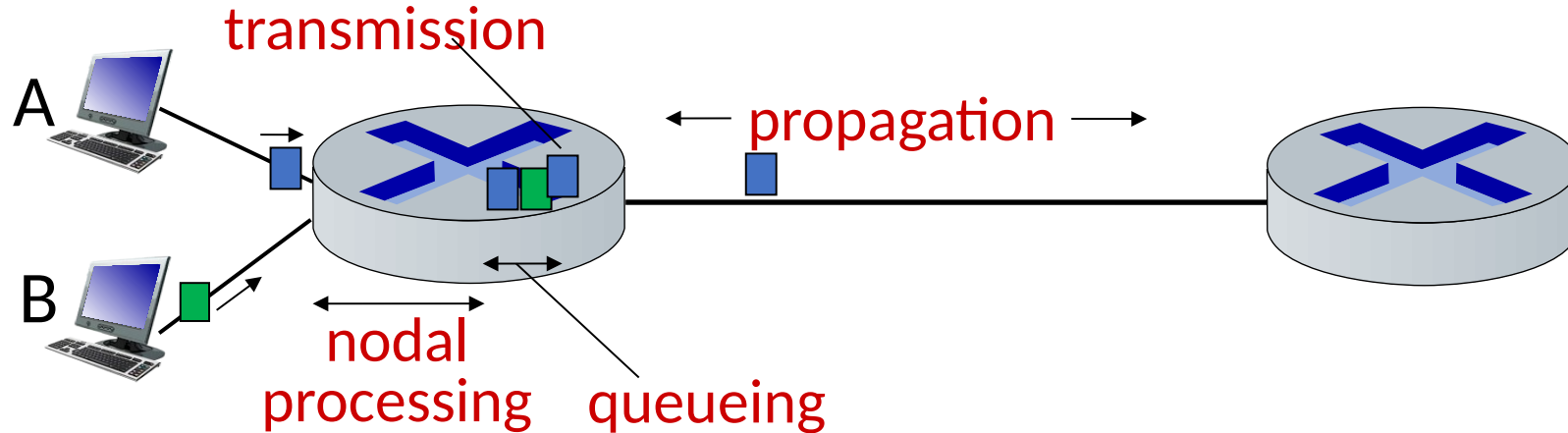
$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

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 (~2x10^8 m/sec)
- $d_{prop} = d/s$

d_{trans} and d_{prop}
very different

Caravan analogy

ten-car caravan
(aka 10-bit packet)

toll booth
(aka link)

100 km

toll booth
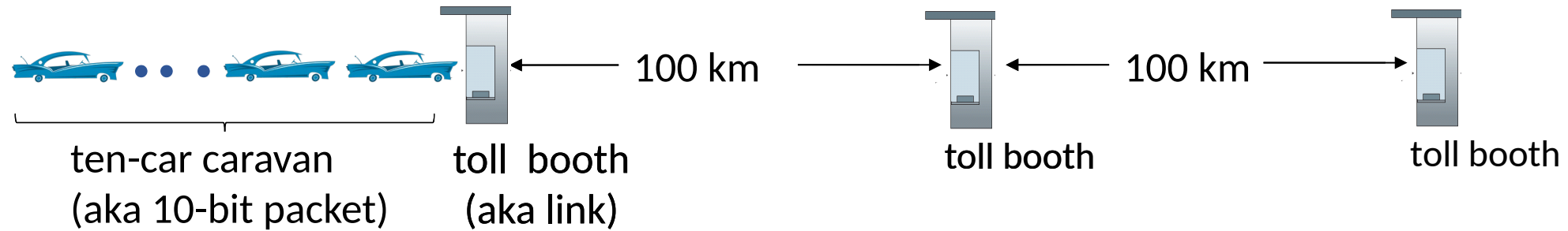
100 km

toll booth

- car ~ bit; caravan ~ packet; toll service ~ link transmission
- toll booth takes 12 sec to service car (bit transmission time)
- "propagate" at 100 km/hr
- *Q: How long until caravan is lined up before 2nd toll booth?*

- time to "push" entire caravan through toll booth onto highway = 12*10 = 120 sec
- time for last car to propagate from 1st to 2nd toll both: 100km/(100km/hr) = 1 hr
- *A: 62 minutes*

Caravan analogy

ten-car caravan
(aka 10-bit packet)

toll booth
(aka router)

100 km

toll booth

100 km
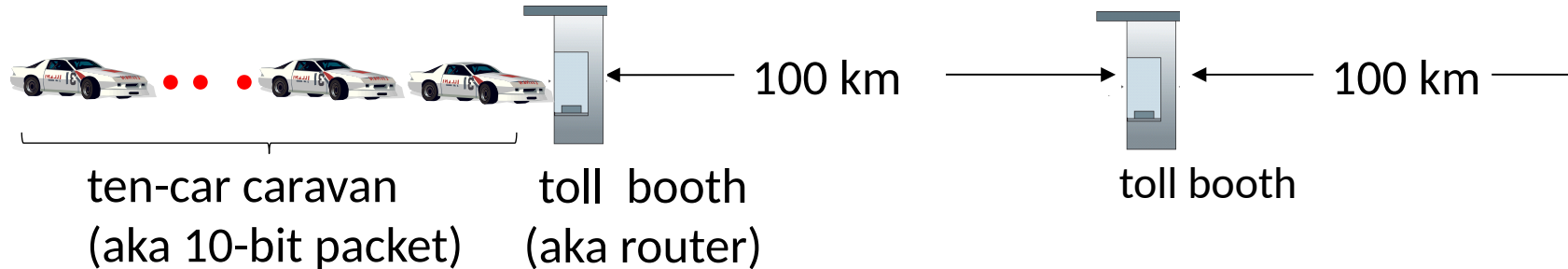
- suppose cars now "propagate" at 1000 km/hr

- and suppose toll booth now takes one min to service a car

- *Q: Will cars arrive to 2nd booth before all cars serviced at first booth?*

   A: Yes! after 7 min, first car arrives at second booth; three cars still at first booth

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"*

traffic intensity = *La/R*   1

- *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
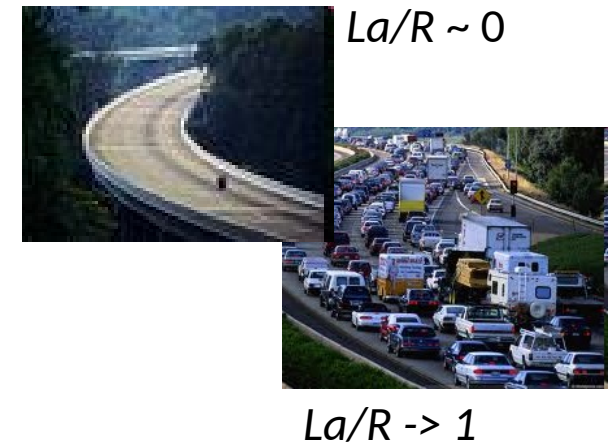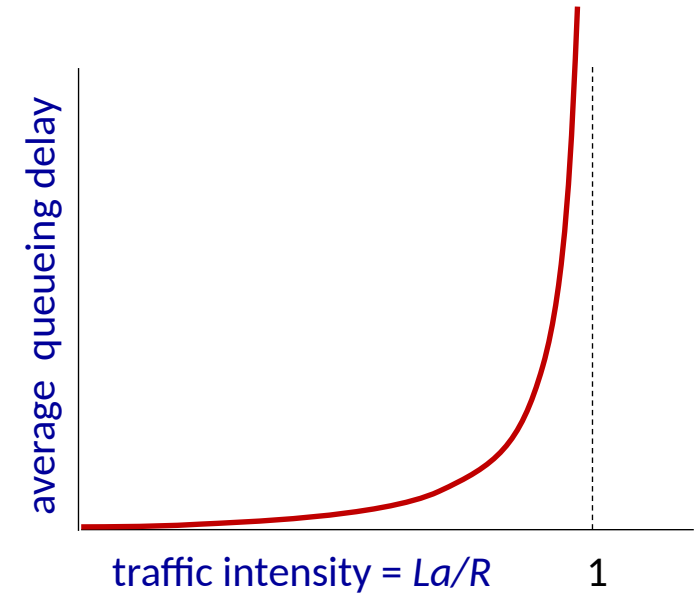
"Real" Internet delays and routes

- 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

3 probes

3 probes

3 probes
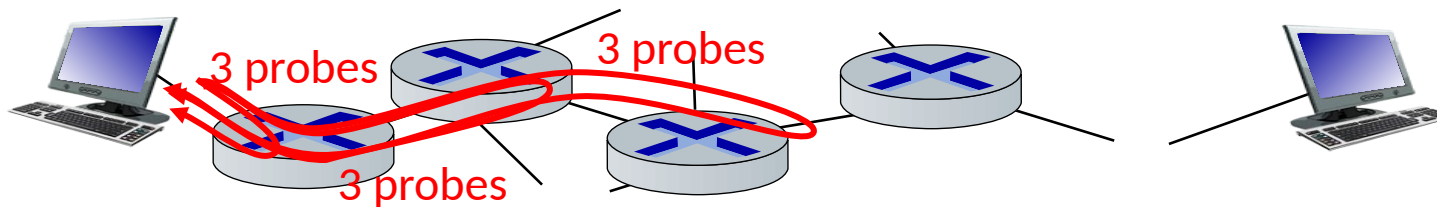
Real Internet delays and routes

traceroute: gaia.cs.umass.edu to www.eurecom.fr

3 delay measurements from
gaia.cs.umass.edu to cs-gw.cs.umass.edu

```
1  cs-gw (128.119.240.254)  1 ms  1 ms  2 ms
2  border1-rt-fa5-1-0.gw.umass.edu (128.119.3.145)  1 ms  1 ms  2 ms
3  cht-vbns.gw.umass.edu (128.119.3.130)  6 ms 5 ms 5 ms
4  jn1-at1-0-0-19.wor.vbns.net (204.147.132.129)  16 ms 11 ms 13 ms
5  jn1-so7-0-0-0.wae.vbns.net (204.147.136.136)  21 ms 18 ms 18 ms
6  abilene-vbns.abilene.ucaid.edu (198.32.11.9)  22 ms  18 ms  22 ms
7  nycm-wash.abilene.ucaid.edu (198.32.8.46)  22 ms  22 ms  22 ms
8  62.40.103.253 (62.40.103.253)  104 ms 109 ms 106 ms
9  de2-1.de1.de.geant.net (62.40.96.129)  109 ms 102 ms 104 ms
10 de.fr1.fr.geant.net (62.40.96.50)  113 ms 121 ms 114 ms
11 renater-gw.fr1.fr.geant.net (62.40.103.54)  112 ms  114 ms  112 ms
12 nio-n2.cssi.renater.fr (193.51.206.13)  111 ms  114 ms  116 ms
13 nice.cssi.renater.fr (195.220.98.102)  123 ms  125 ms  124 ms
14 r3t2-nice.cssi.renater.fr (195.220.98.110)  126 ms  126 ms  124 ms
15 eurecom-valbonne.r3t2.ft.net (193.48.50.54)  135 ms  128 ms  133 ms
16 194.214.211.25 (194.214.211.25)  126 ms  128 ms  126 ms
17 * * *
18 * * *
19 fantasia.eurecom.fr (193.55.113.142)  132 ms  128 ms  136 ms
```

3 delay measurements
to border1-rt-fa5-1-0.gw.umass.edu

trans-oceanic link

looks like delays *decrease*! Why?

* means no response (probe lost, router not replying)

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

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

buffer
(waiting area)

packet being transmitted

A

B

packet arriving to
full buffer is *lost*
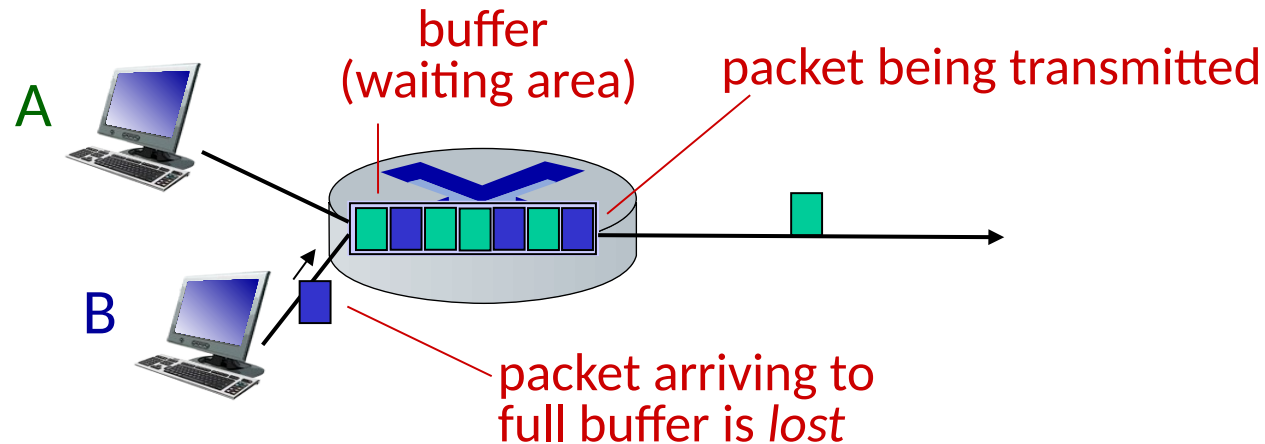
* Check out the Java applet for an interactive animation (on publisher's website) of queuing and loss

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

server sends bits
(fluid) into pipe

pipe that can carry
fluid at rate

$(R_s$ bits/sec)

pipe that can carry
fluid at rate

$(R_c$ bits/sec)
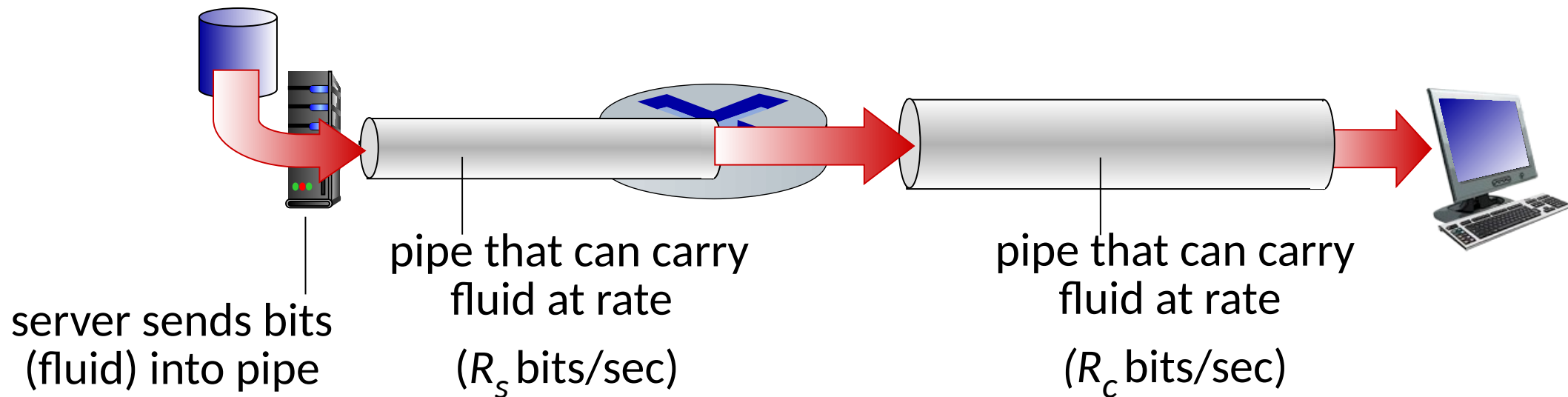
Throughput

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

R_s bits/sec

R_c bits/sec

$R_s > R_c$ What is average end-end throughput?
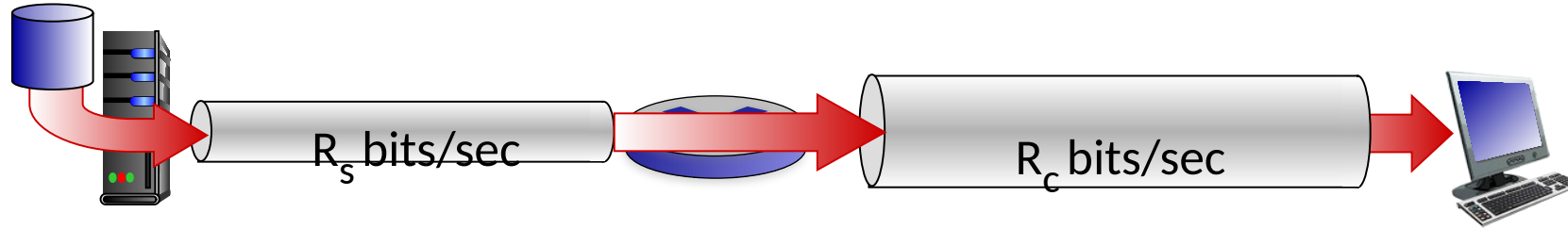
R_s bits/sec
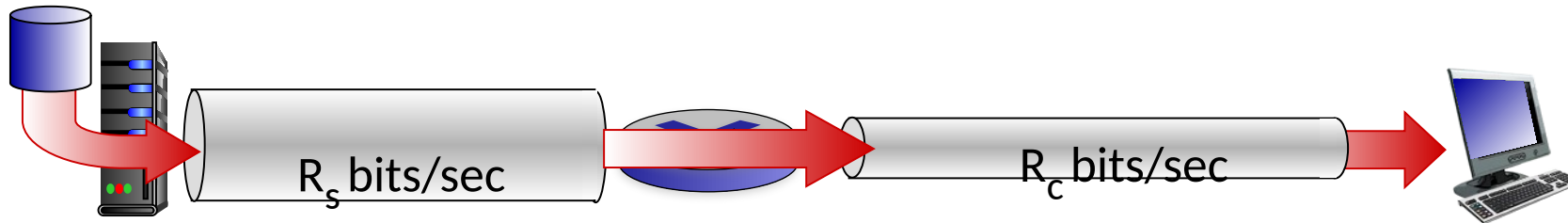
R_c bits/sec

bottleneck link

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

Throughput: network scenario

- per-connection end-end throughput: $min(R_c, R_s, R/10)$
- in practice: R_c or R_s is often bottleneck
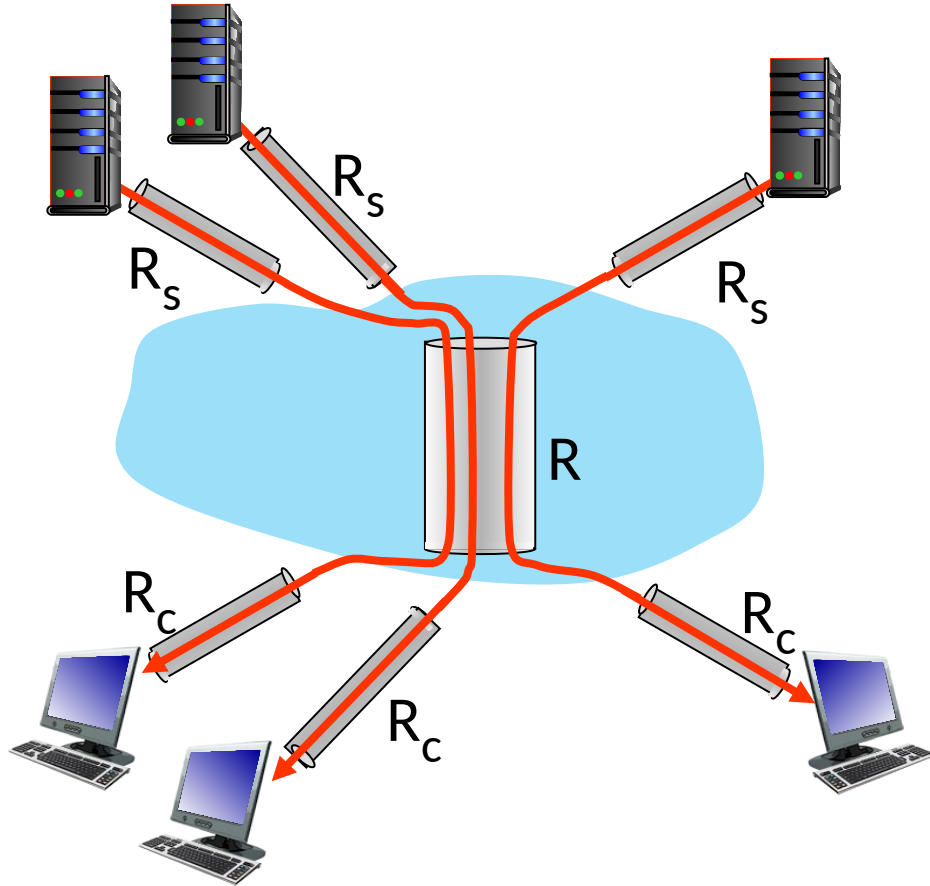
10 connections (fairly) share backbone bottleneck link R bits/sec

* 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

Network security

- Internet not originally designed with (much) security in mind
  - *original vision:* "a group of mutually trusting users attached to a transparent network" ^^
  - Internet protocol designers playing "catch-up"
  - security considerations in all layers!

- We now need to think about:
  - how bad guys can attack computer networks
  - how we can defend networks against attacks
  - how to design architectures that are immune to attacks

Network security

- Internet not originally designed with (much) security in mind
  - *original vision:* "a group of mutually trusting users attached to a transparent network" ⁀
  - Internet protocol designers playing "catch-up"
  - security considerations in all layers!
- We now need to think about:
  - how bad guys can attack computer networks
  - how we can defend networks against attacks
  - how to design architectures that are immune to attacks

Bad guys: packet interception

packet "sniffing":

- broadcast media (shared Ethernet, wireless)
- promiscuous network interface reads/records all packets (e.g., including passwords!) passing by

| src:B | dest:A | | payload |
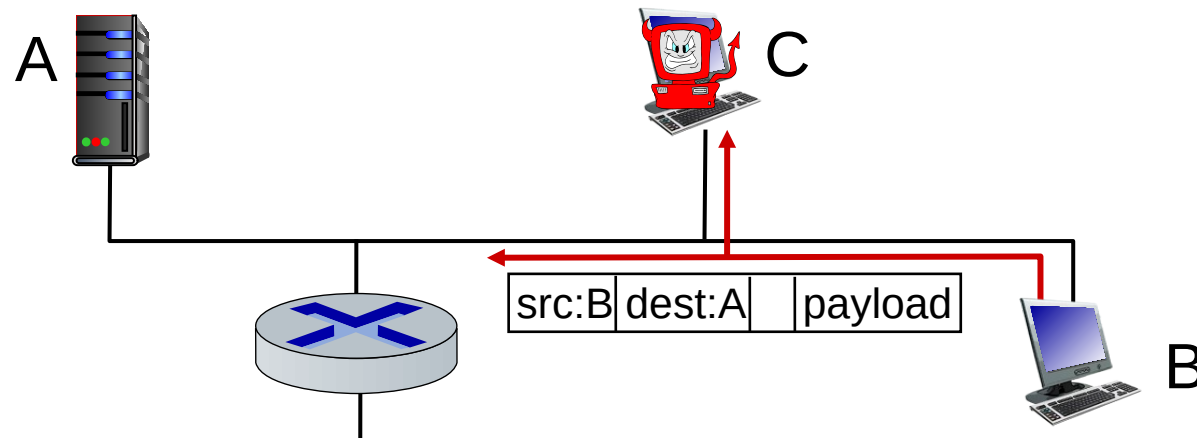
Wireshark software used for our end-of-chapter labs is a (free) packet-sniffer

Bad guys: fake identity

IP spoofing: injection of packet with false source address

A

C

src:B | dest:A | payload

B
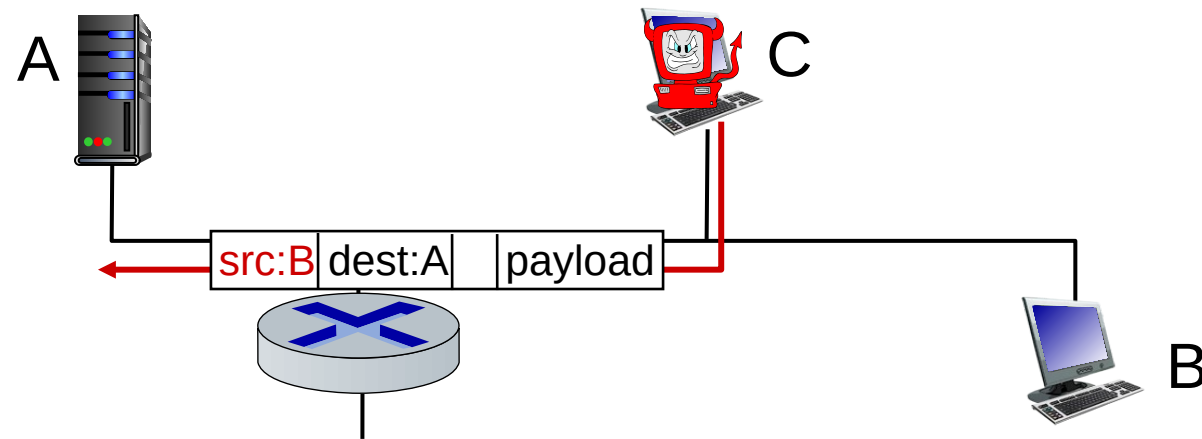
Bad guys: denial of service

Denial of Service (DoS): attackers make resources (server, bandwidth) unavailable to legitimate traffic by overwhelming resource with bogus traffic

1. select target

2. break into hosts around the network (see botnet)

3. send packets to target from compromised hosts

target
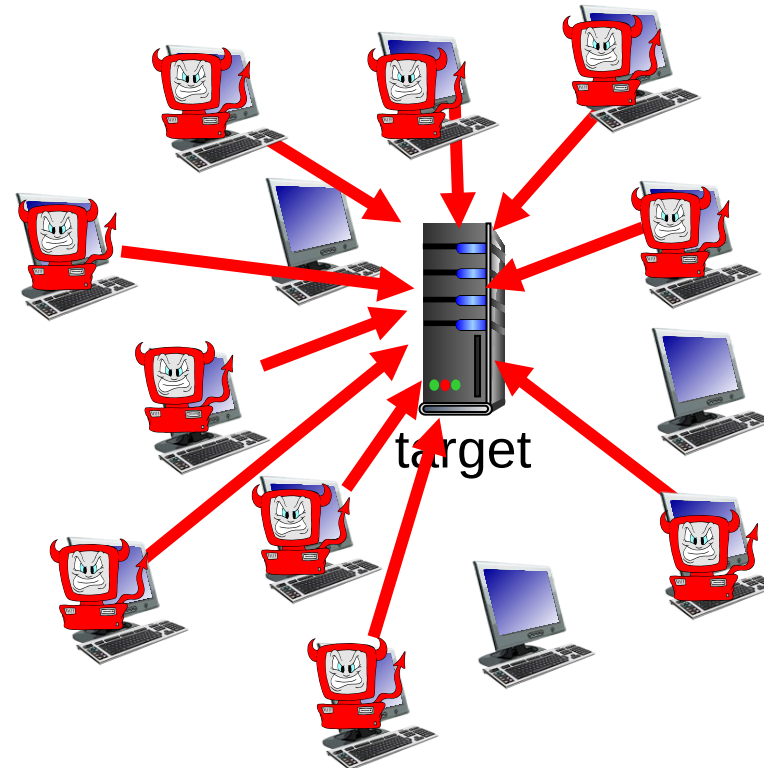
Lines of defense:

- authentication: proving you are who you say you are
  - cellular networks provides hardware identity via SIM card; no such hardware assist in traditional Internet

- confidentiality: via encryption

- integrity checks: digital signatures prevent/detect tampering

- access restrictions: password-protected VPNs

- firewalls: specialized "middleboxes" in access and core networks:
  - off-by-default: filter incoming packets to restrict senders, receivers, applications
  - detecting/reacting to DOS attacks

… lots more on security (throughout, Chapter 8)

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
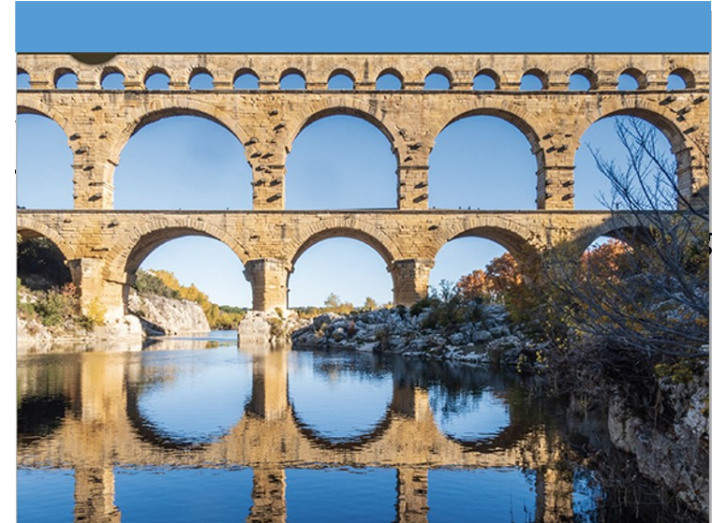
Protocol "layers" and reference models

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?

Example: organization of air travel
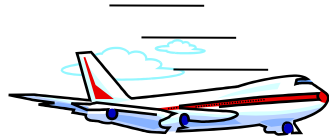
end-to-end transfer of person plus baggage

ticket (purchase)   ticket (complain)

baggage (check)   baggage (claim)

gates (load)   gates (unload)

runway takeoff   runway landing

airplane routing   airplane routing

airplane routing

How would you *define/discuss* the *system* of airline travel?
- a series of steps, involving many services

Example: organization of air travel

ticket (purchase)	*ticketing service*	ticket (complain)
baggage (check)	*baggage service*	baggage (claim)
gates (load)	*gate service*	gates (unload)
runway takeoff	*runway service*	runway landing
airplane routing	*routing service*	airplane routing

layers: each layer implements a service
- via its own internal-layer actions
- relying on services provided by layer below

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"

application
transport
network
link
physical

Services, Layering and Encapsulation

source		destination
application	M	application
	Application exchanges messages to implement some application service using *services* of transport layer	
transport	H_t M	transport
	Transport-layer protocol transfers M (e.g., reliably) from one *process* to another, using services of network layer	
network		network
link		link
physical		physical

- 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

M

application

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

H_t M

transport

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

H_n H_t M

network

link

physical

■ 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

source

application

transport

network

link

physical

destination

Services, Layering and Encapsulation

application	M	application
transport	H_t M	transport
network	H_n H_t M	network

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

link	H_l H_n H_t M	link

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

physical		physical

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

source

destination

Encapsulation

Matryoshka dolls (stacking dolls)

message   segment   datagram   frame

Services, Layering and Encapsulation

application

transport

network

link

physical

source

message | M |

segment | H_t | M |

datagram | H_n | H_t | M |

frame | H_l | H_n | H_t | M |

| M | application

| H_t | M | transport

| H_n | H_t | M | network

| H_l | H_n | H_t | M | link

physical
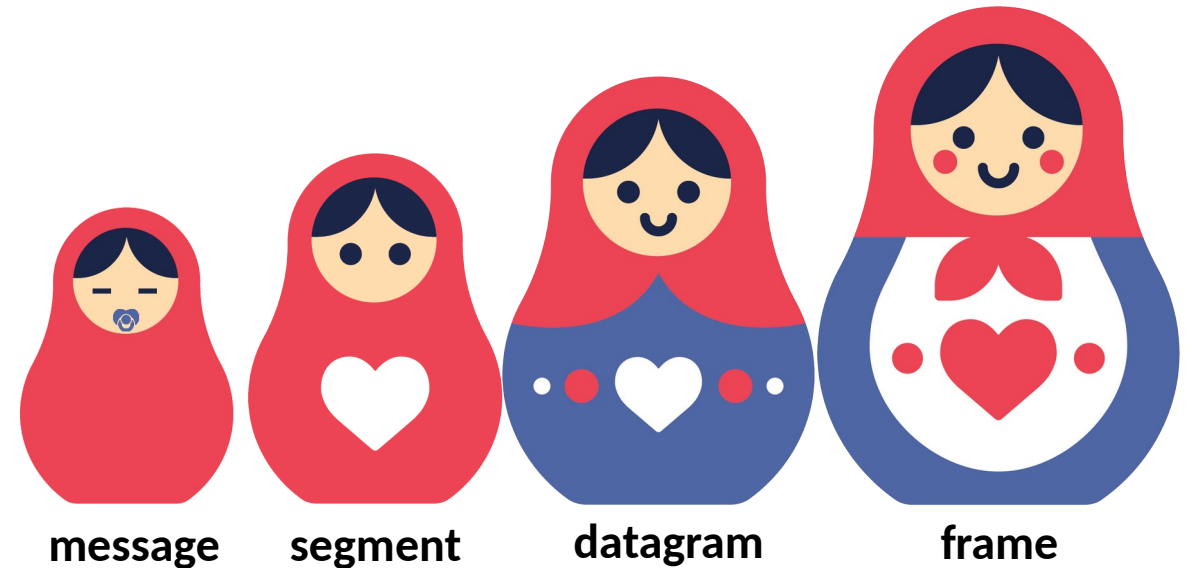
destination
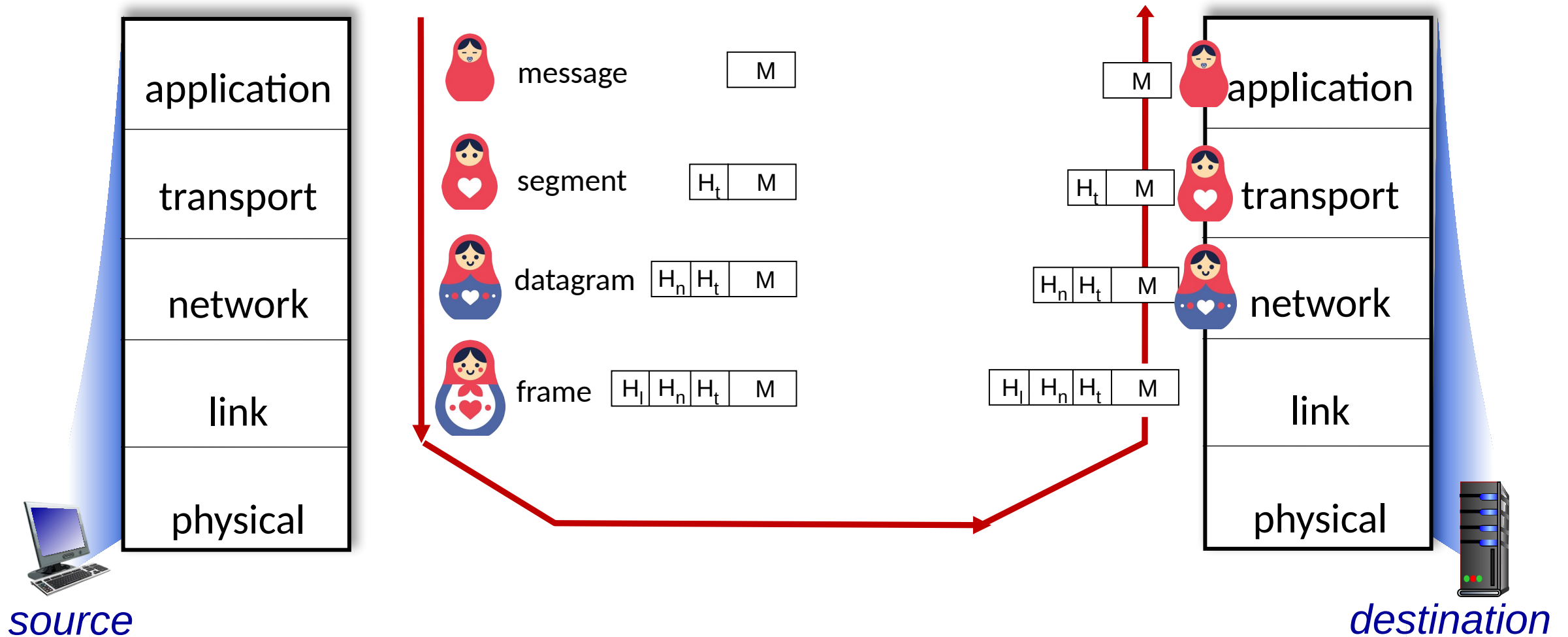
Encapsulation: an end-end view

source

message | M
segment | Ht | M
datagram | Hn | Ht | M
frame | Hl | Hn | Ht | M

| application |
| transport |
| network |
| link |
| physical |

| link |
| physical |

switch

destination

| M |
Ht	M
Hn	Ht	M
Hl	Hn	Ht	M

| application |
| transport |
| network |
| link |
| physical |

Hn | Ht | M
Hl | Hn | Ht | M

| network |
| link |
| physical |

Hn | Ht | M

router