Chapter 1
Introduction

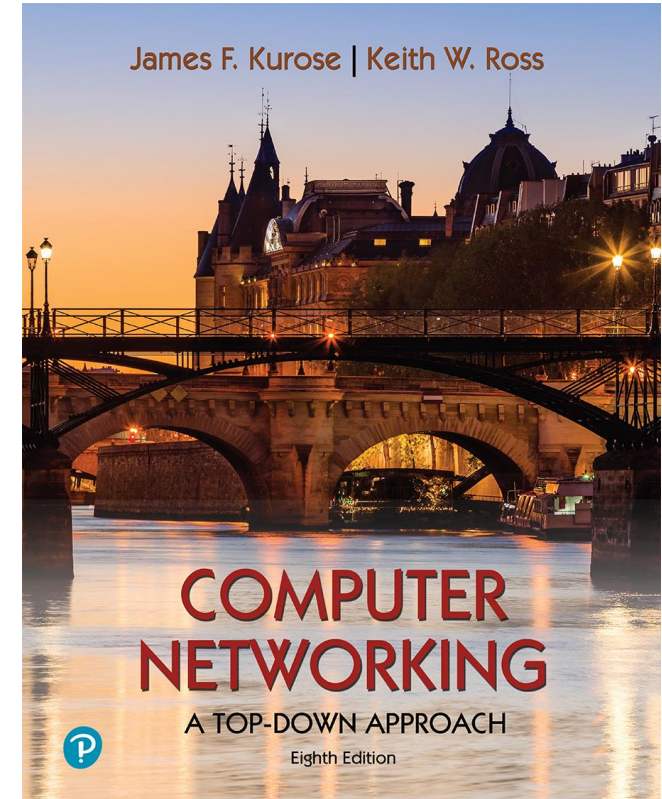A note on the use of these PowerPoint slides:
We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent 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

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
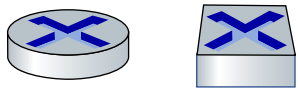

Overview/roadmap:

- What *is* the Internet? What *is* a protocol?
- Packet/circuit switching, internet structure
- Performance: loss, delay, throughput
- Protocol layers, service models

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)
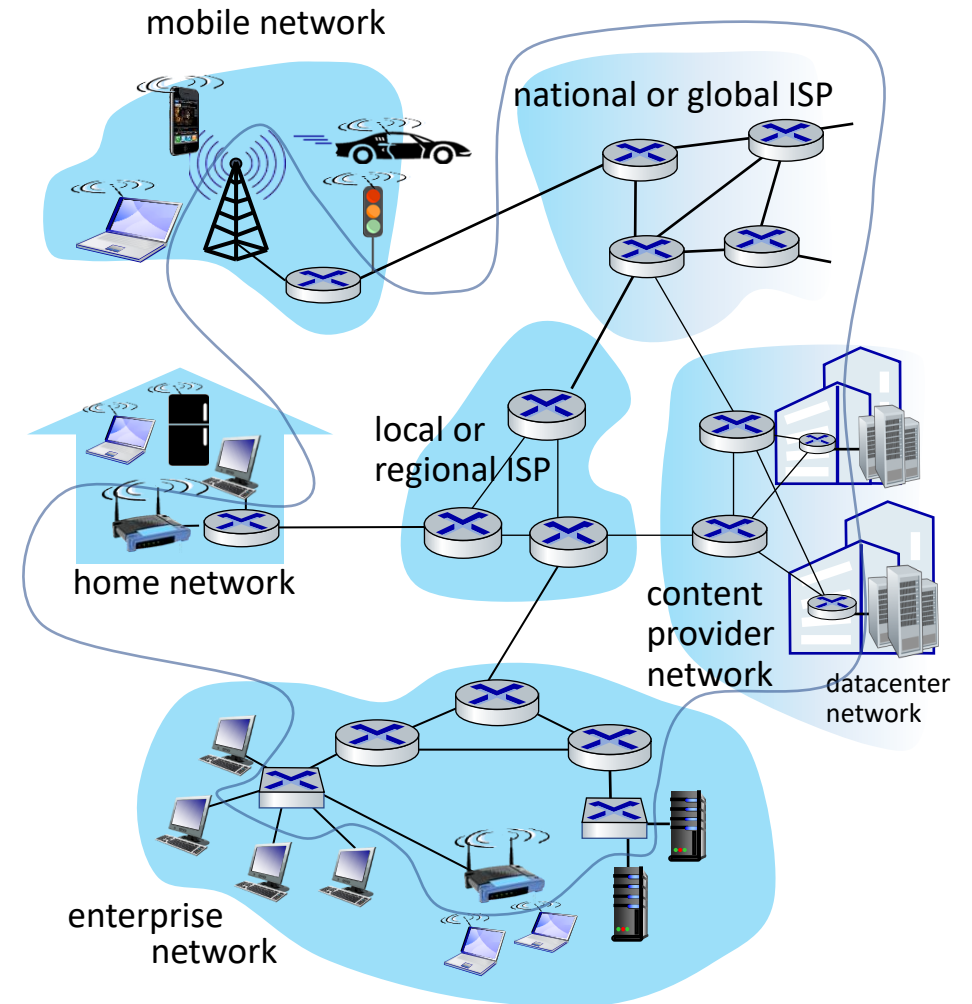
- *routers, switches*

Communication links

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

Networks

- collection of devices, routers, switches 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

AR devices

cars

scooters

Internet phones
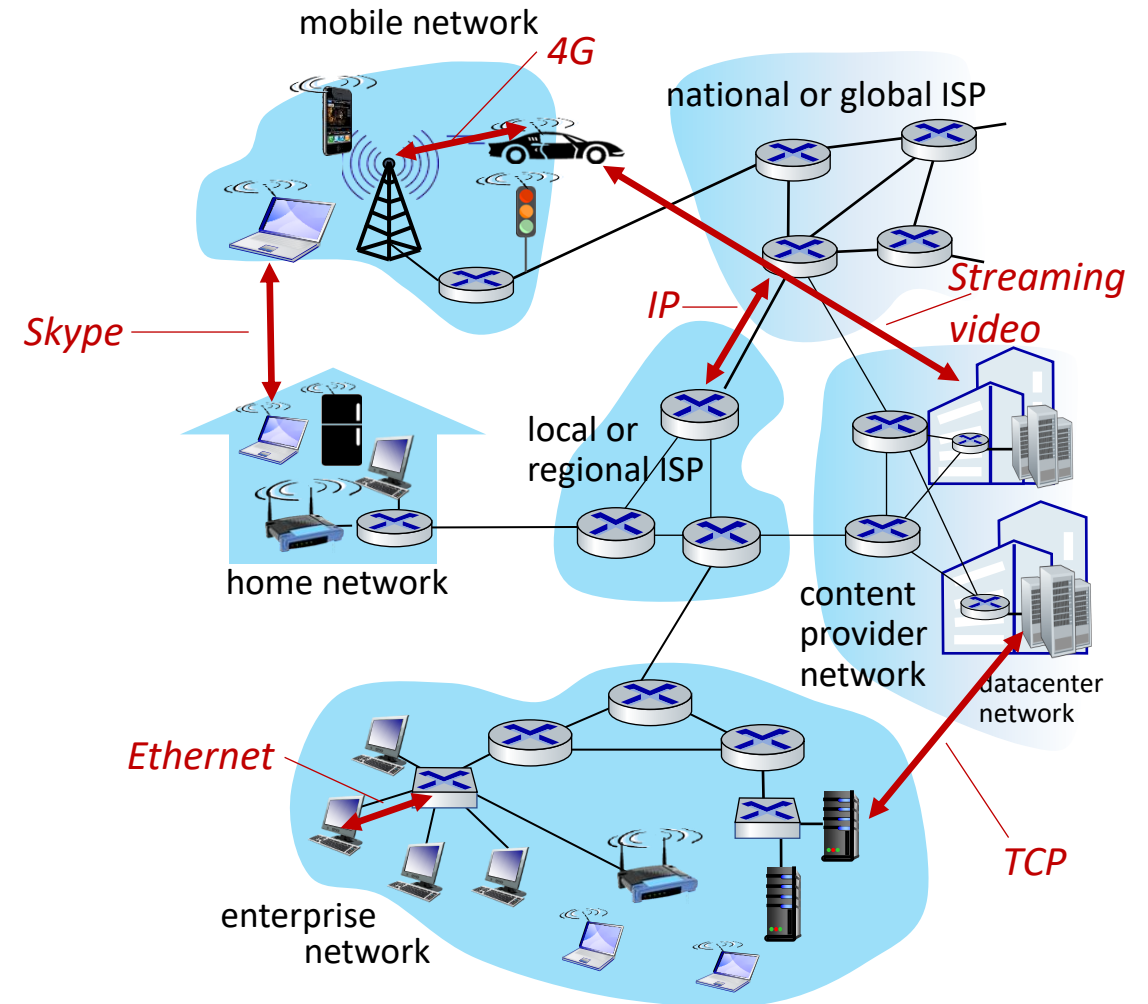
Gaming devices

sensorized, bed mattress

Fitbit

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, 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 distributed applications:
  - Web, streaming video, multimedia teleconferencing, email, games, e-commerce, social media, inter-connected appliances, …
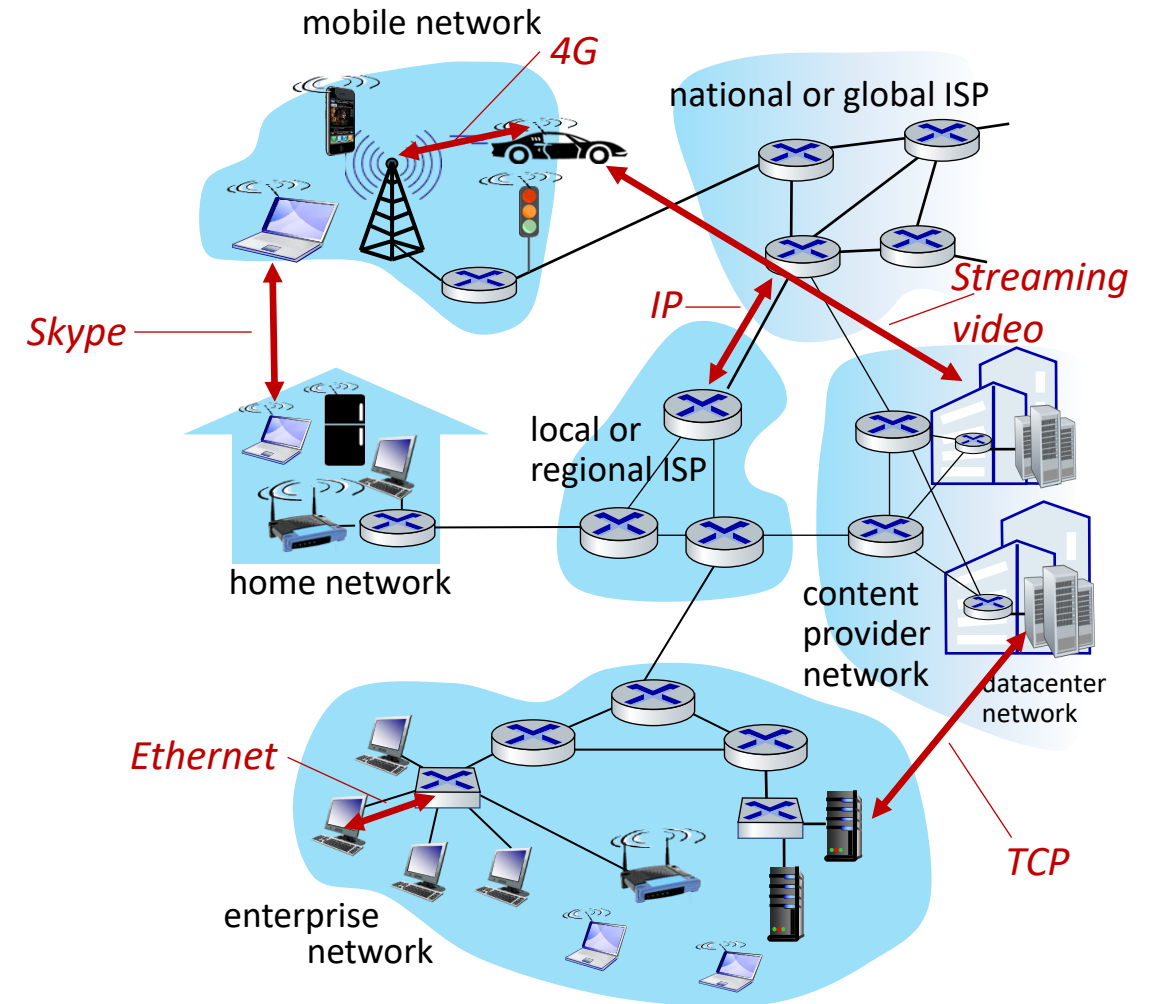
What's a protocol?

Human protocols:

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

Rules for:

... specific messages sent

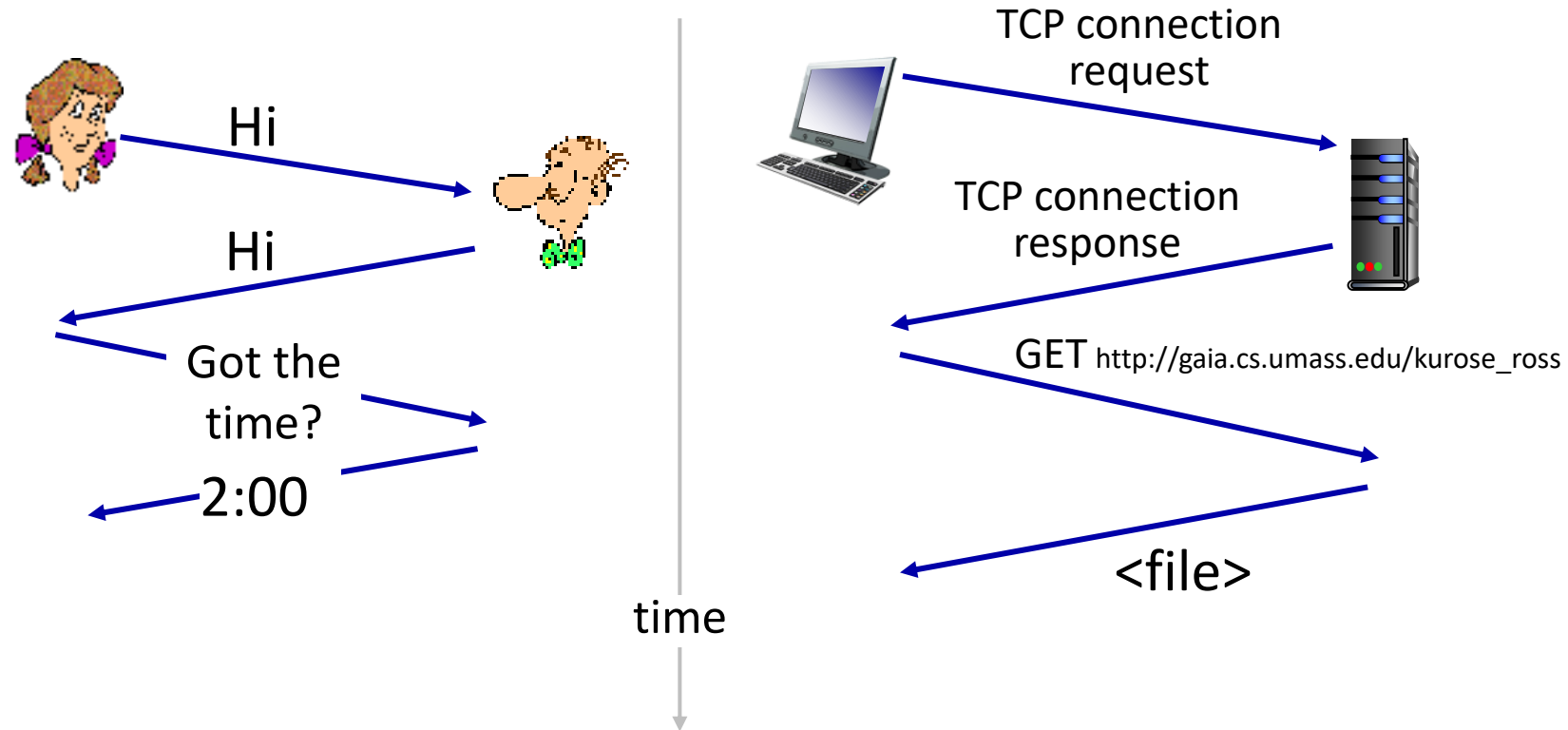... specific actions taken when message received, or other events

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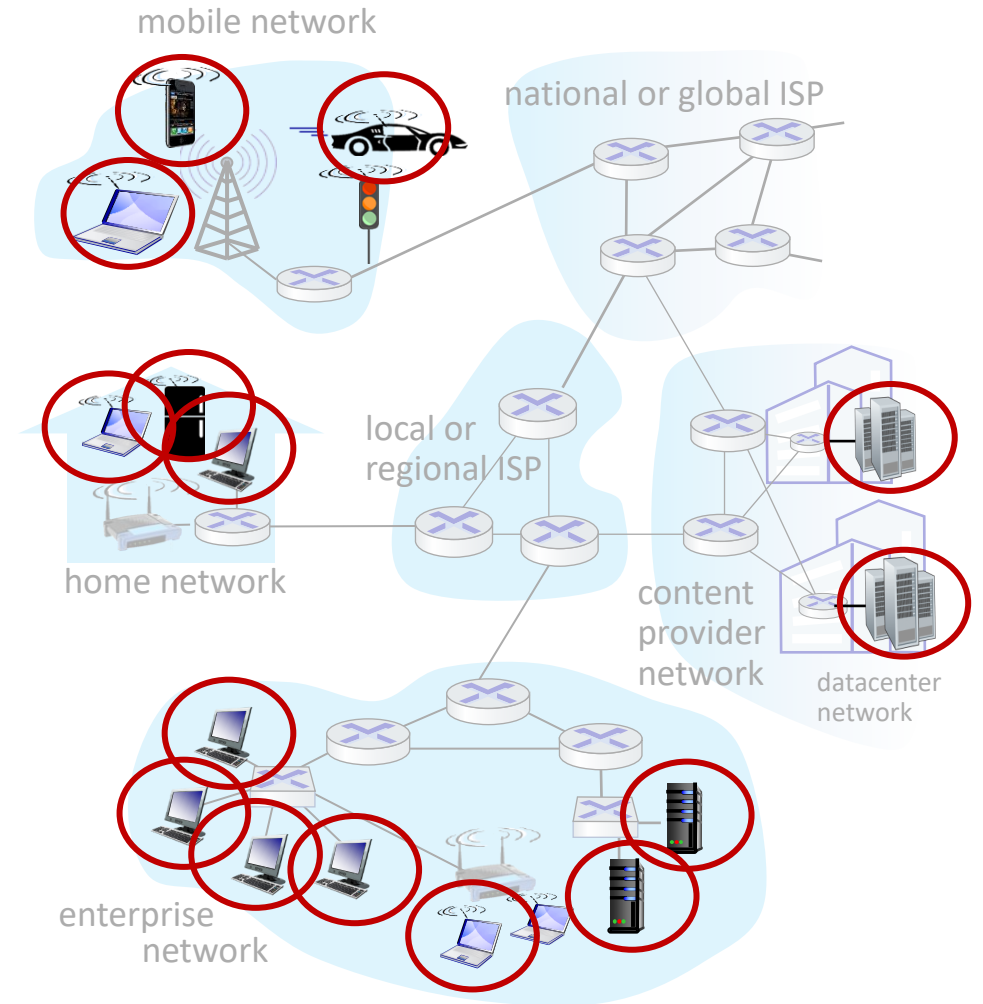
GET http://gaia.cs.umass.edu/kurose_ross

<file>

time

Q: other human protocols?

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:

- wired, wireless communication links

mobile network

national or global ISP

local or regional ISP

content provider network

datacenter network
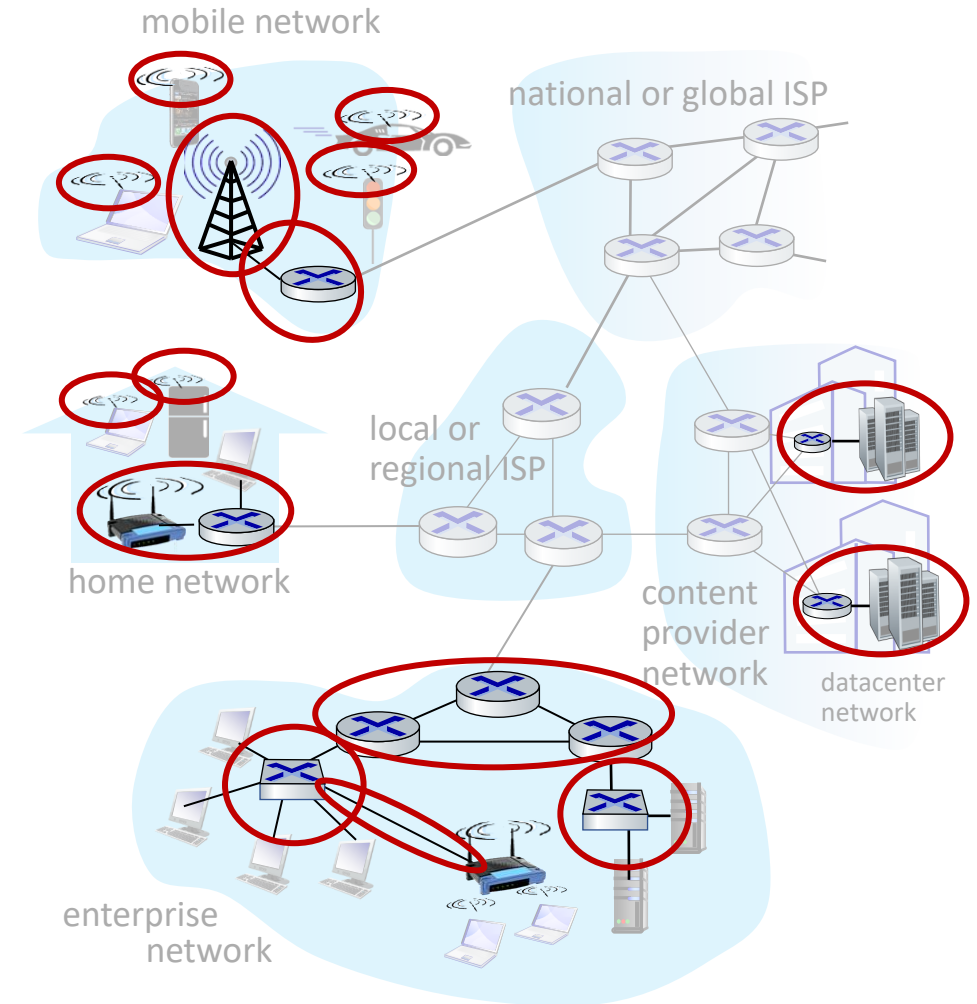
home network

enterprise network

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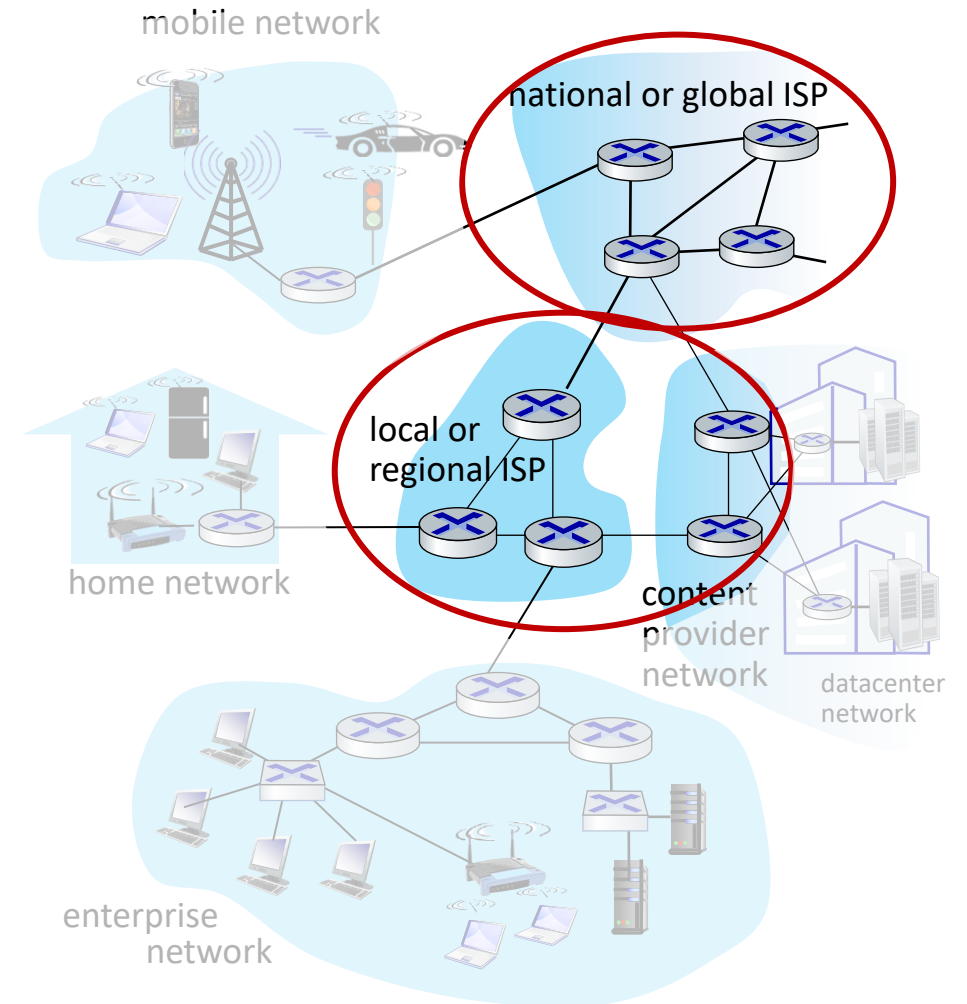
Network edge:
- hosts: clients and servers
- servers often in data centers

Access networks:
- 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

Network Edge
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*
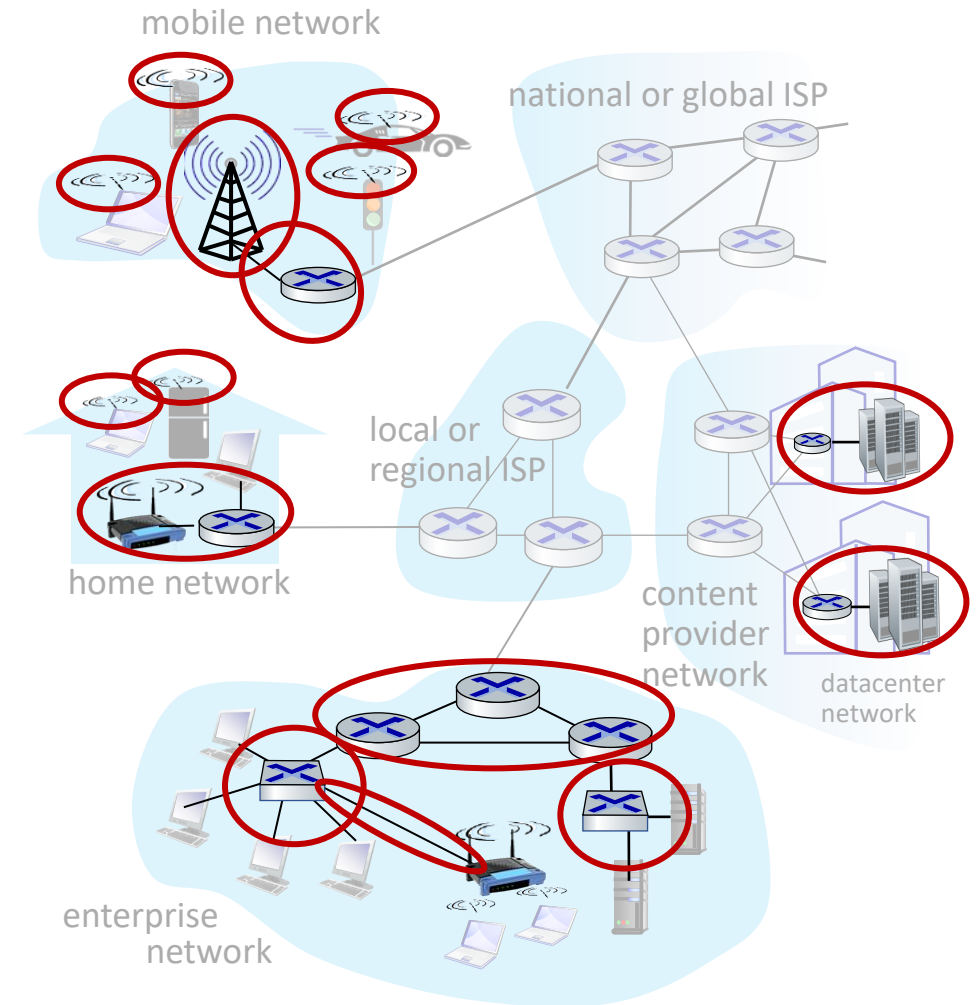 - link transmission rate, aka link *capacity, aka link bandwidth*

two packets, *L* bits each

host

R: link transmission rate
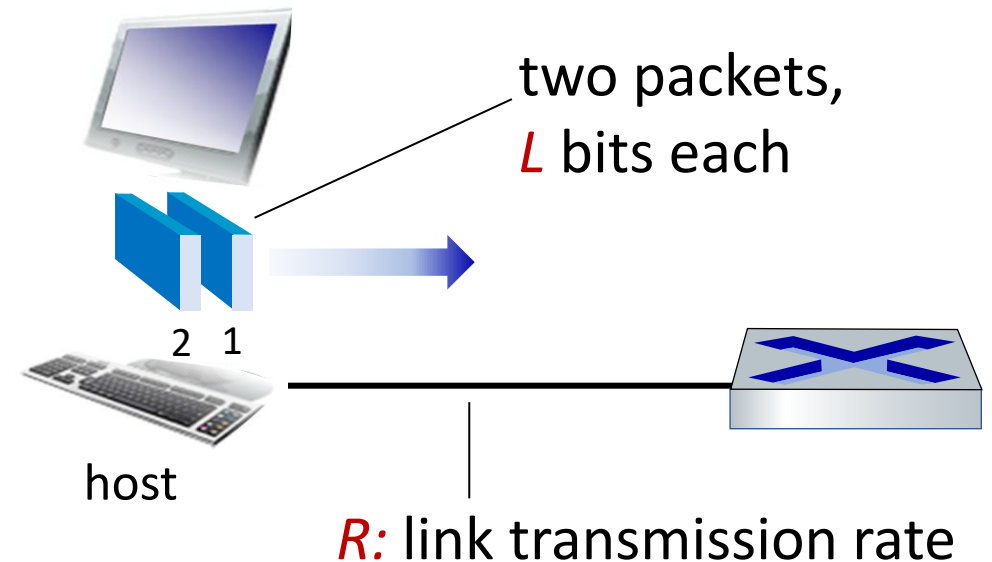
$$\text{packet transmission delay} = \text{time needed to transmit } L\text{-bit packet into link} = \frac{L \text{ (bits)}}{R \text{ (bits/sec)}}$$
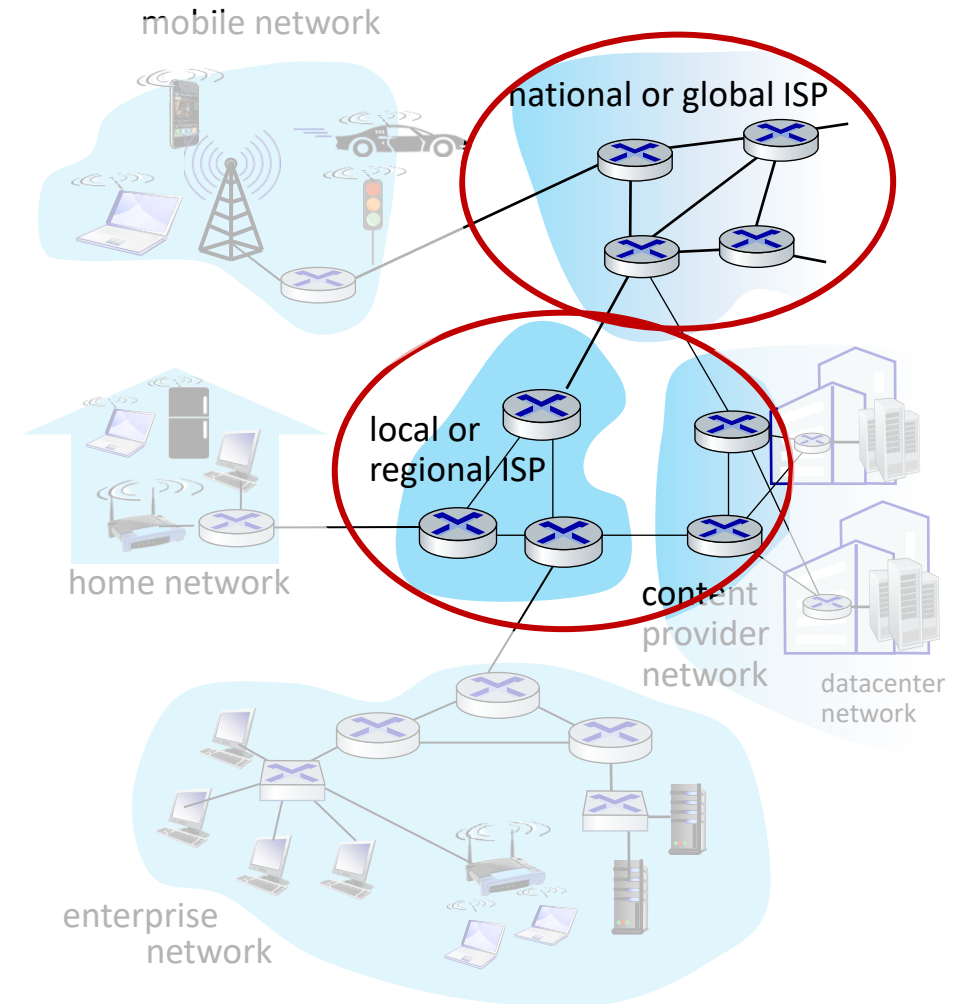
Access networks

Q: How to connect end systems to "Internet"?

- 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

Network core

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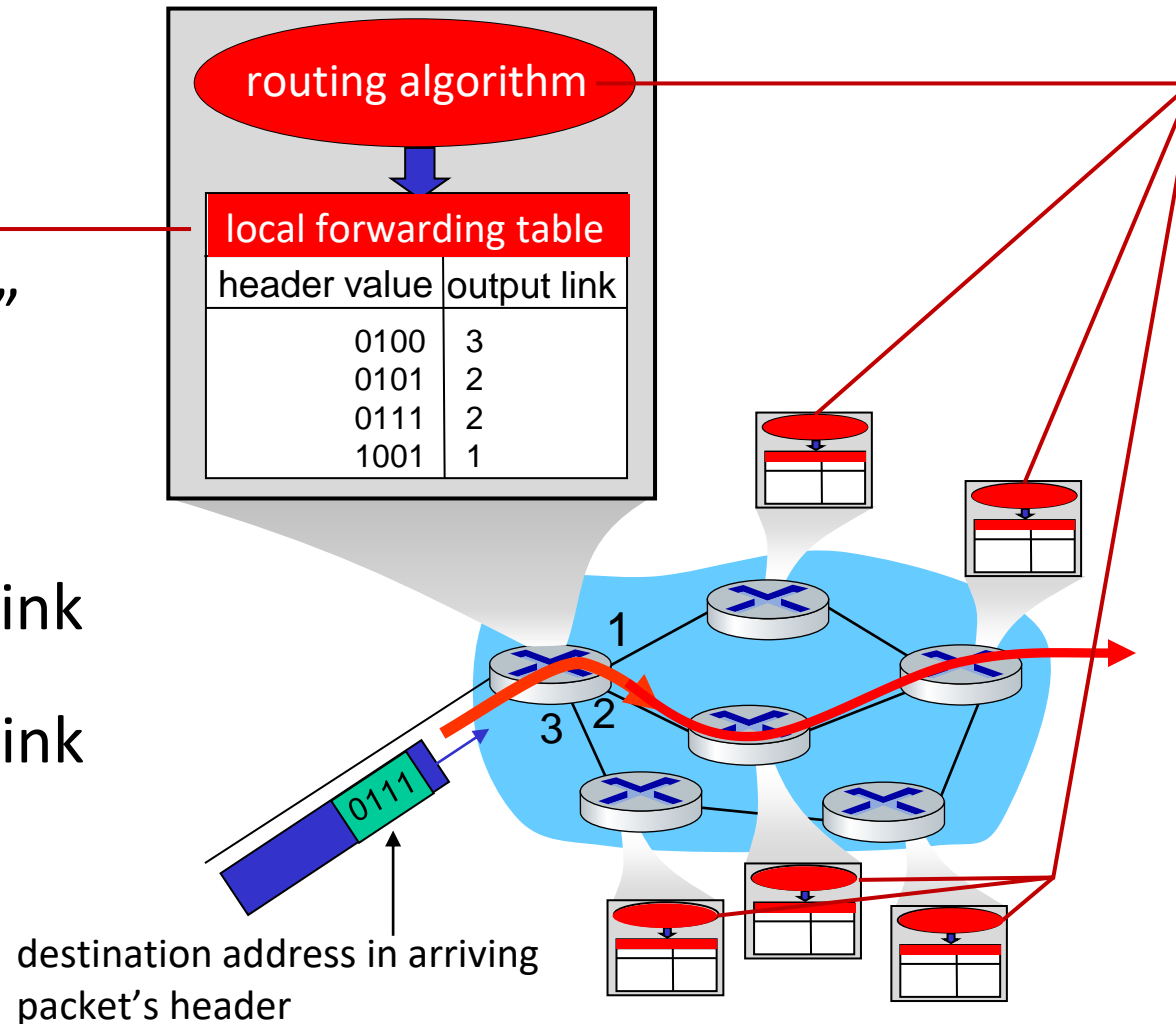 - network forwards packets from one router to the next, across links on path from source to destination

mobile network

national or global ISP

home network

local or regional ISP

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 algorithm

local forwarding table	
header value	output link
0100	3
0101	2
0111	2
1001	1

0111

destination address in arriving packet's header

Routing:

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

routing

forwarding

forwarding

Packet-switching: store-and-forward
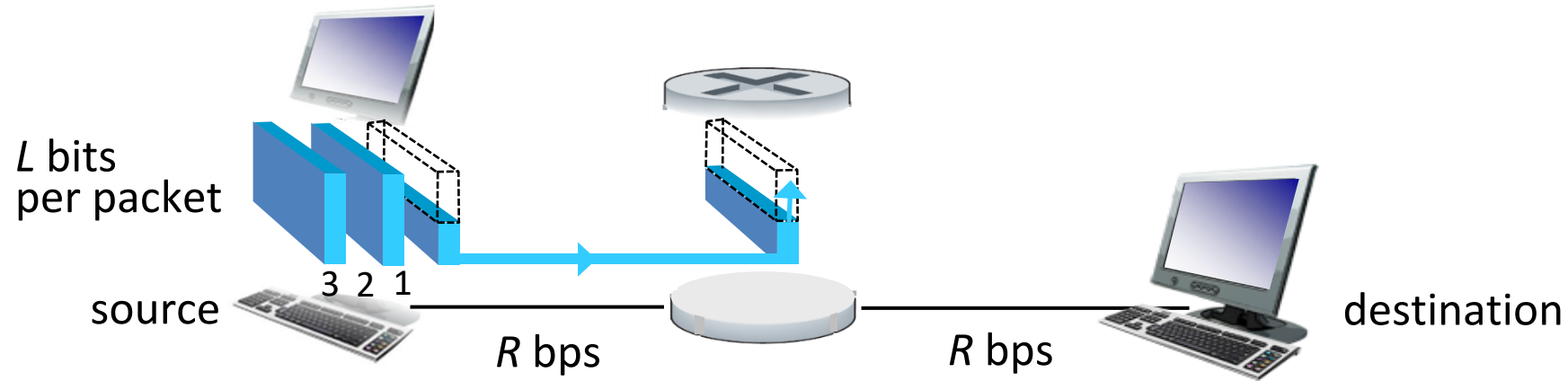
- **packet transmission delay:** takes L/R seconds to transmit (push out) L-bit packet into link at R bps

- *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
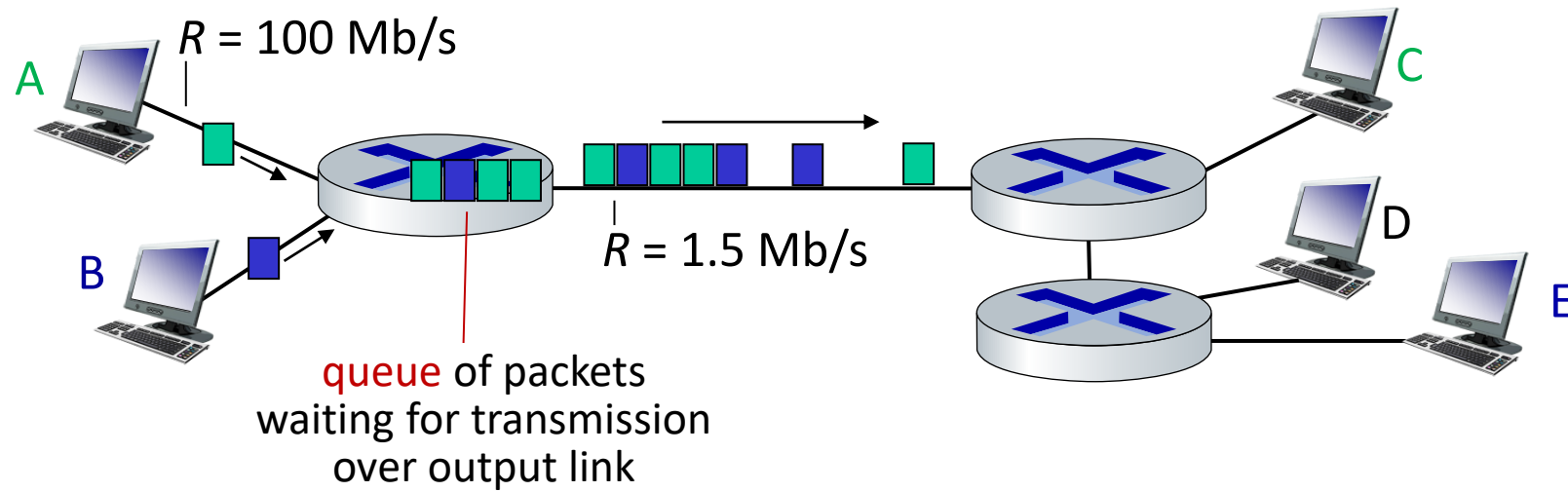- one-hop transmission delay = 0.1 msec

Packet-switching: queueing

R = 100 Mb/s

A

B

queue of packets
waiting for transmission
over output link

R = 1.5 Mb/s

C

D

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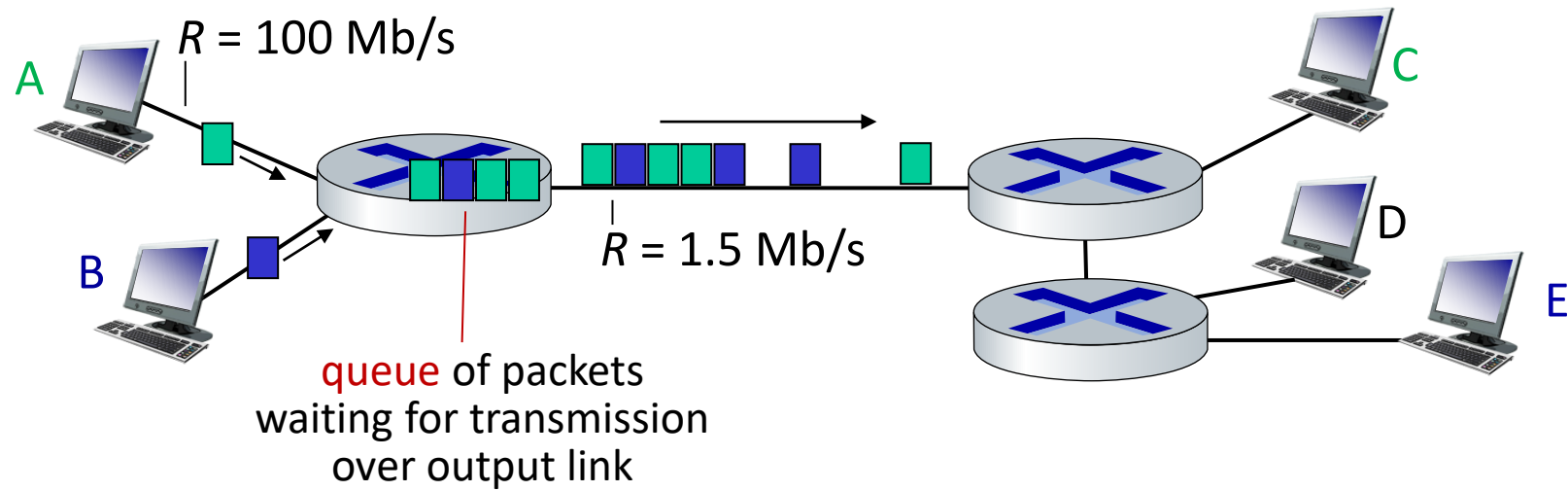
Queueing occurs when work arrives faster than it can be serviced:

Packet-switching: queueing

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)

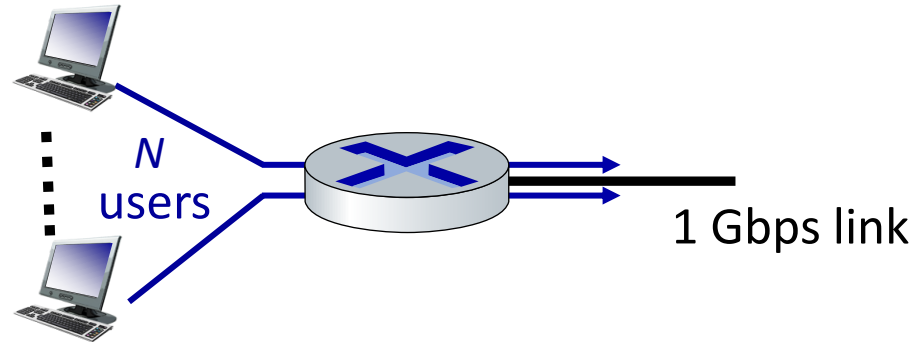- commonly used in traditional telephone networks
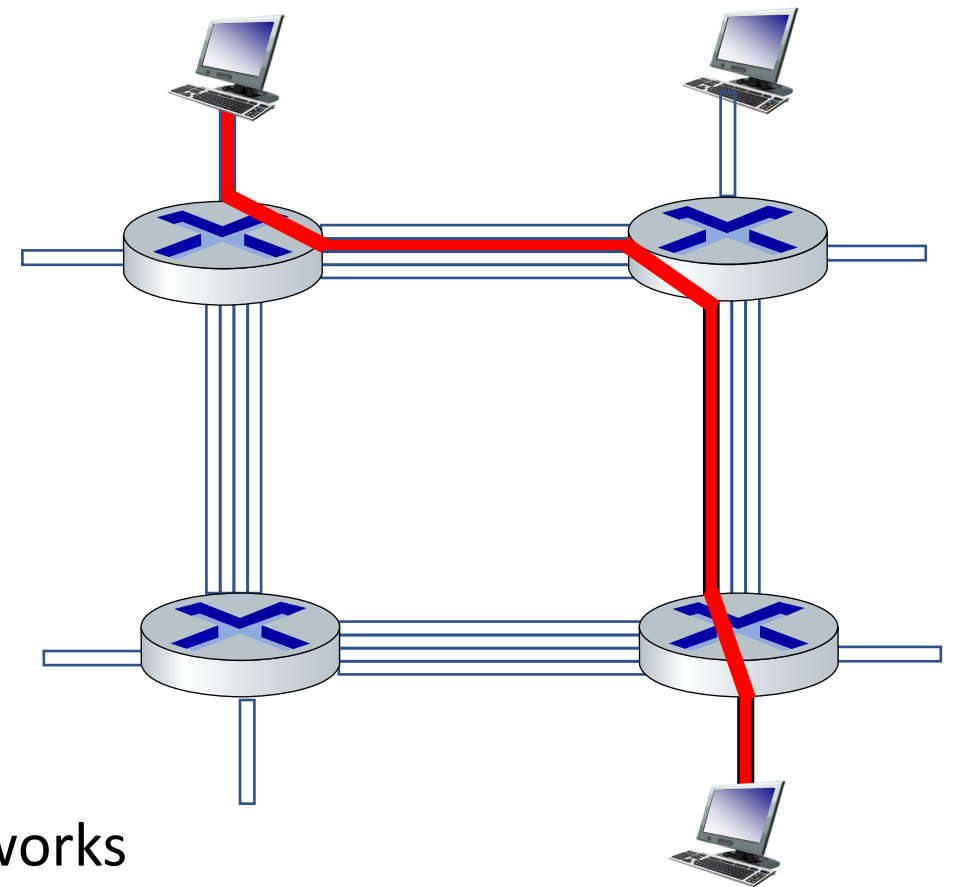
* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive

Packet switching versus circuit switching

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

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 0.0004 *

Q: how did we get value 0.0004?
A: HW problem

* 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." Various techniques that try to make packet switching as "circuit-like" as possible are possible (but we do not see them in this course)

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
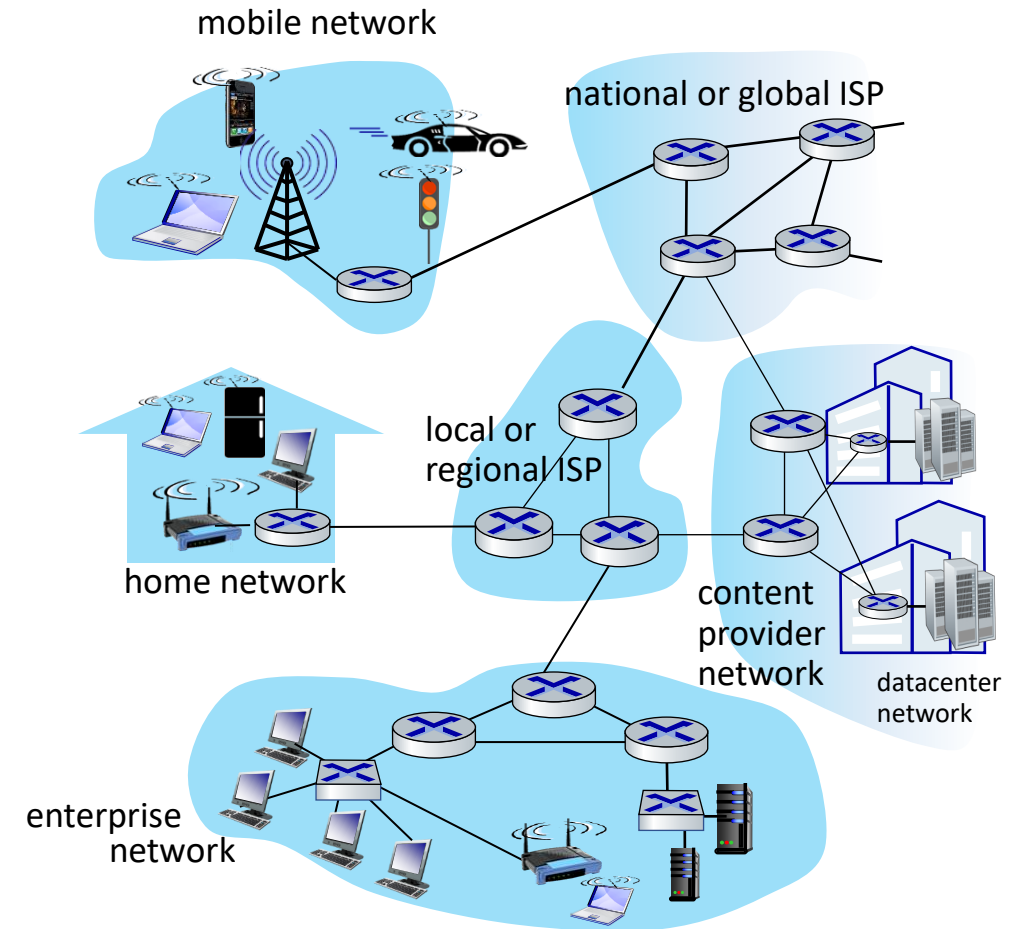
Let's take a stepwise approach to describe current Internet structure

Internet structure: a "network of networks"
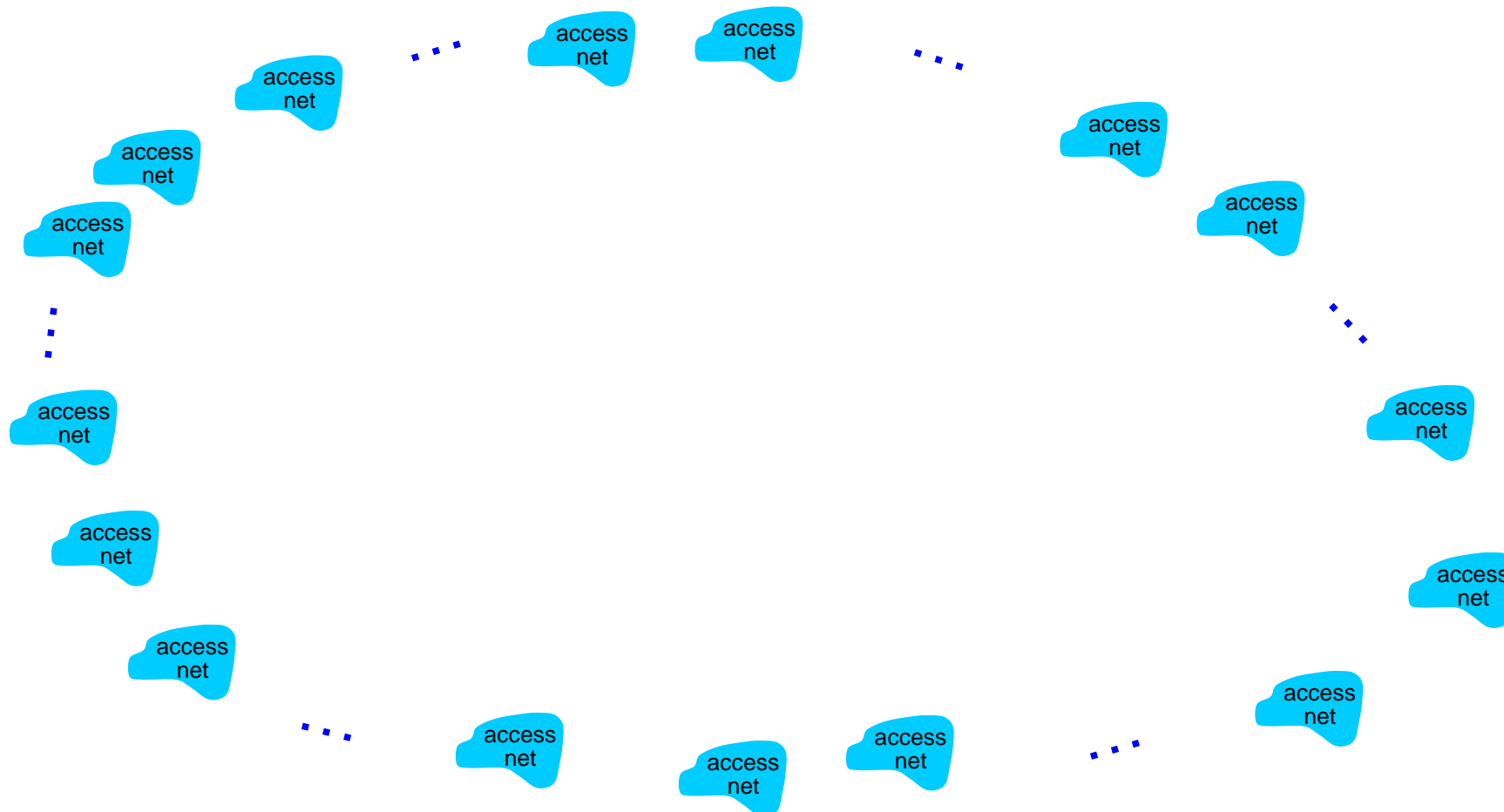
Question: given *millions* of access ISPs, how to connect them together?

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.
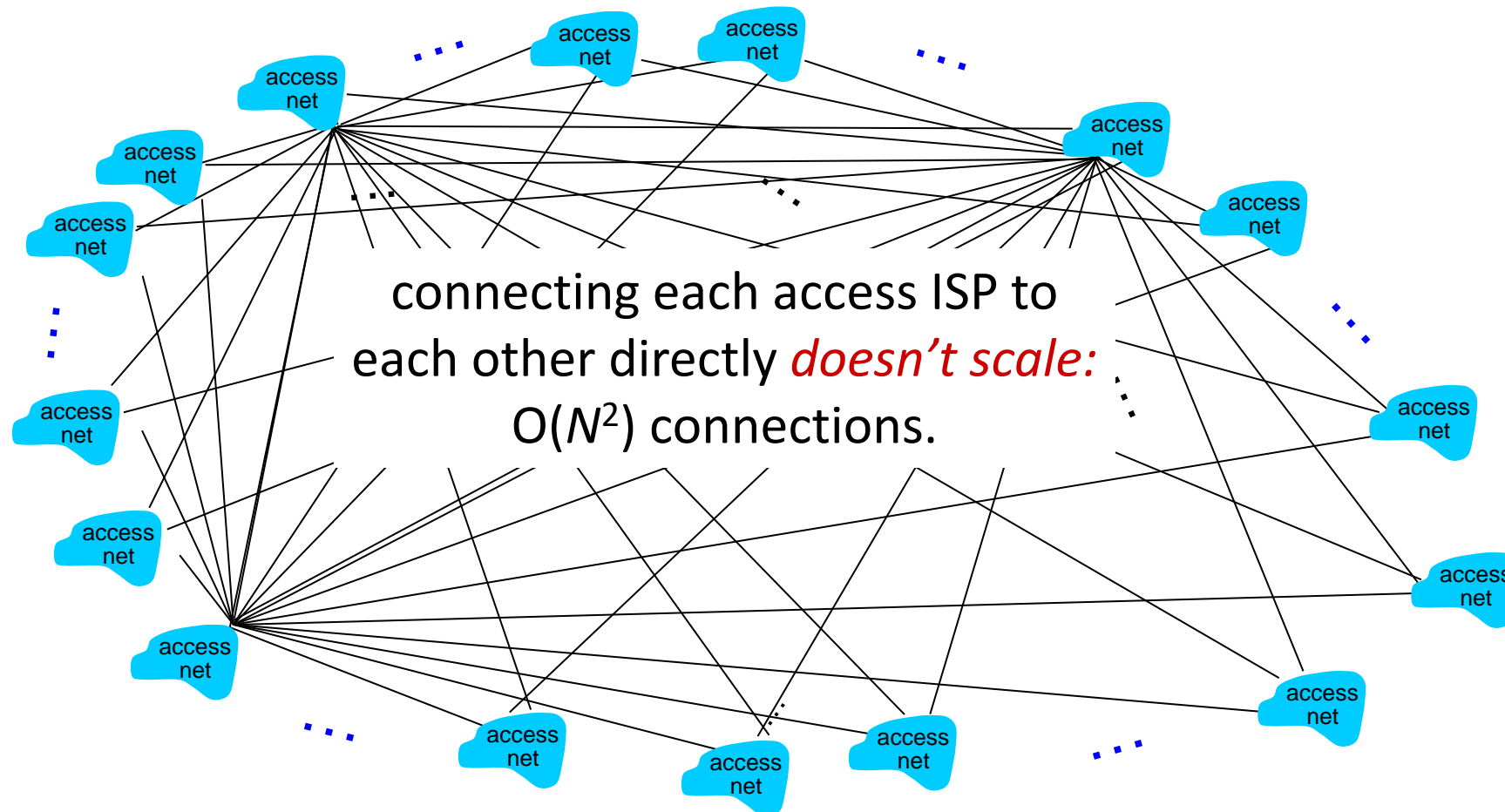
Internet structure: a "network of networks"

Option: connect each access ISP to one global transit ISP?
Customer and *provider* ISPs have economic agreement.
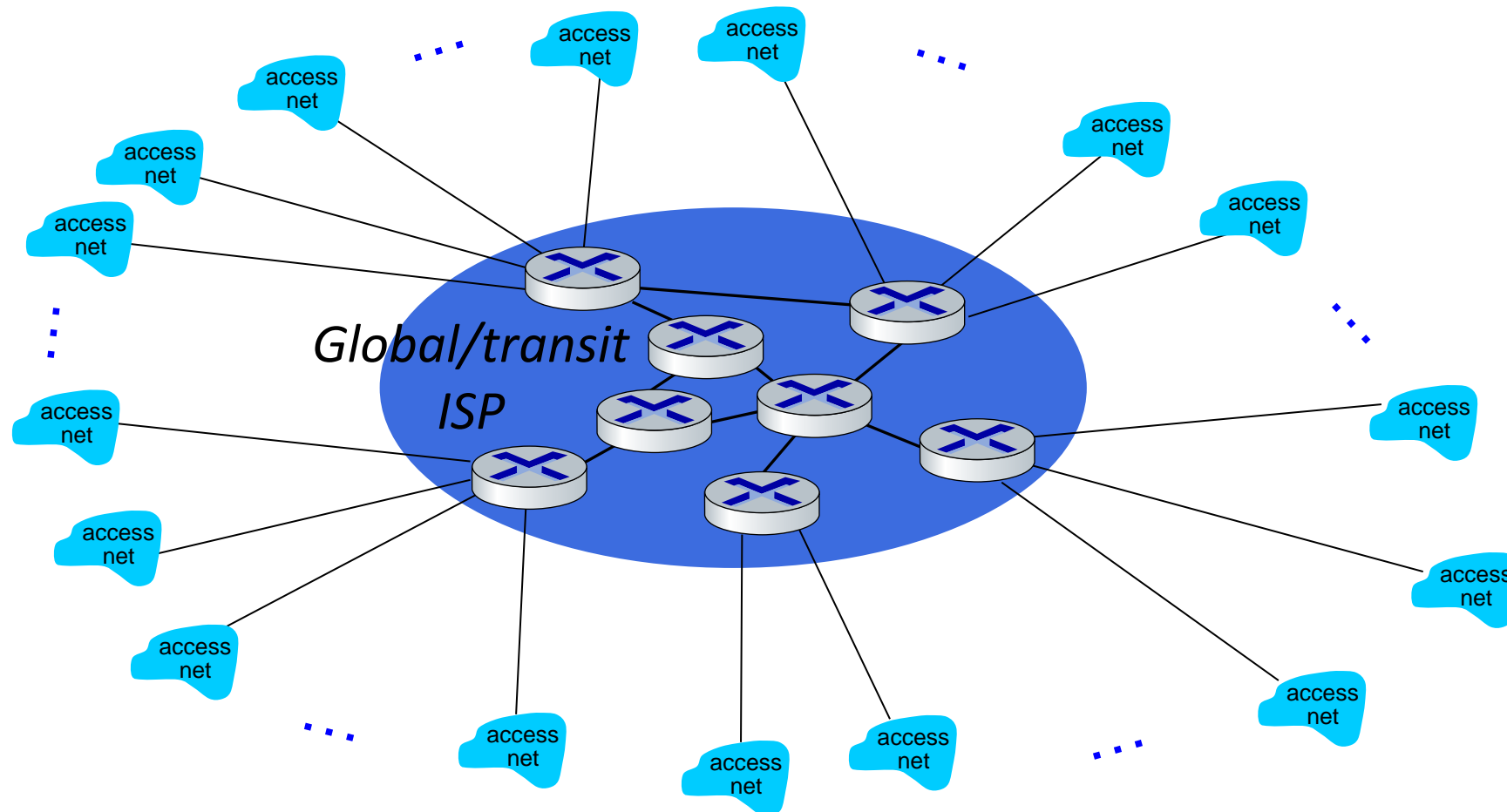
Global/transit ISP

Internet structure: a "network of networks"

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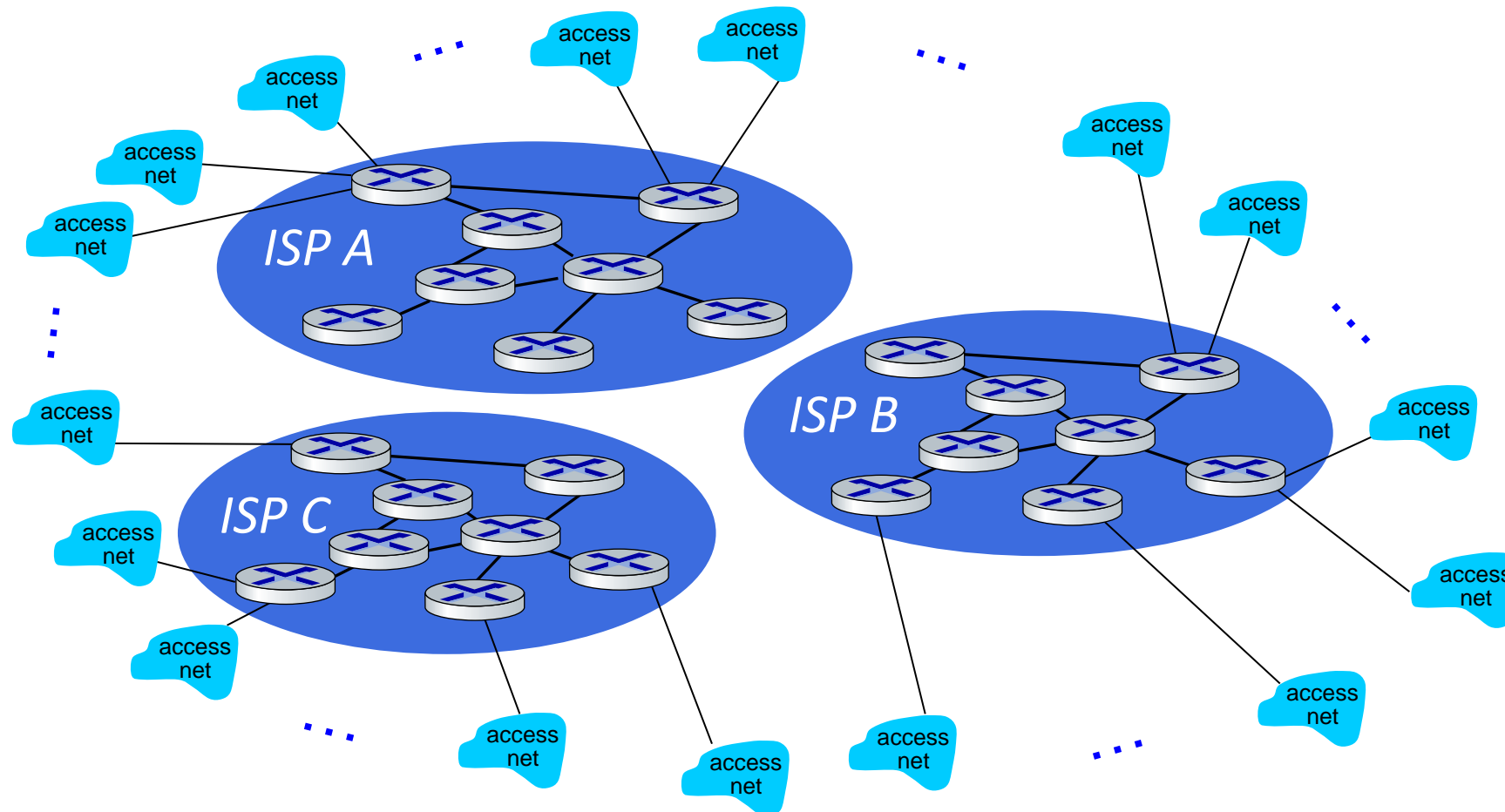
Internet structure: a "network of networks"

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

Internet exchange point

peering link
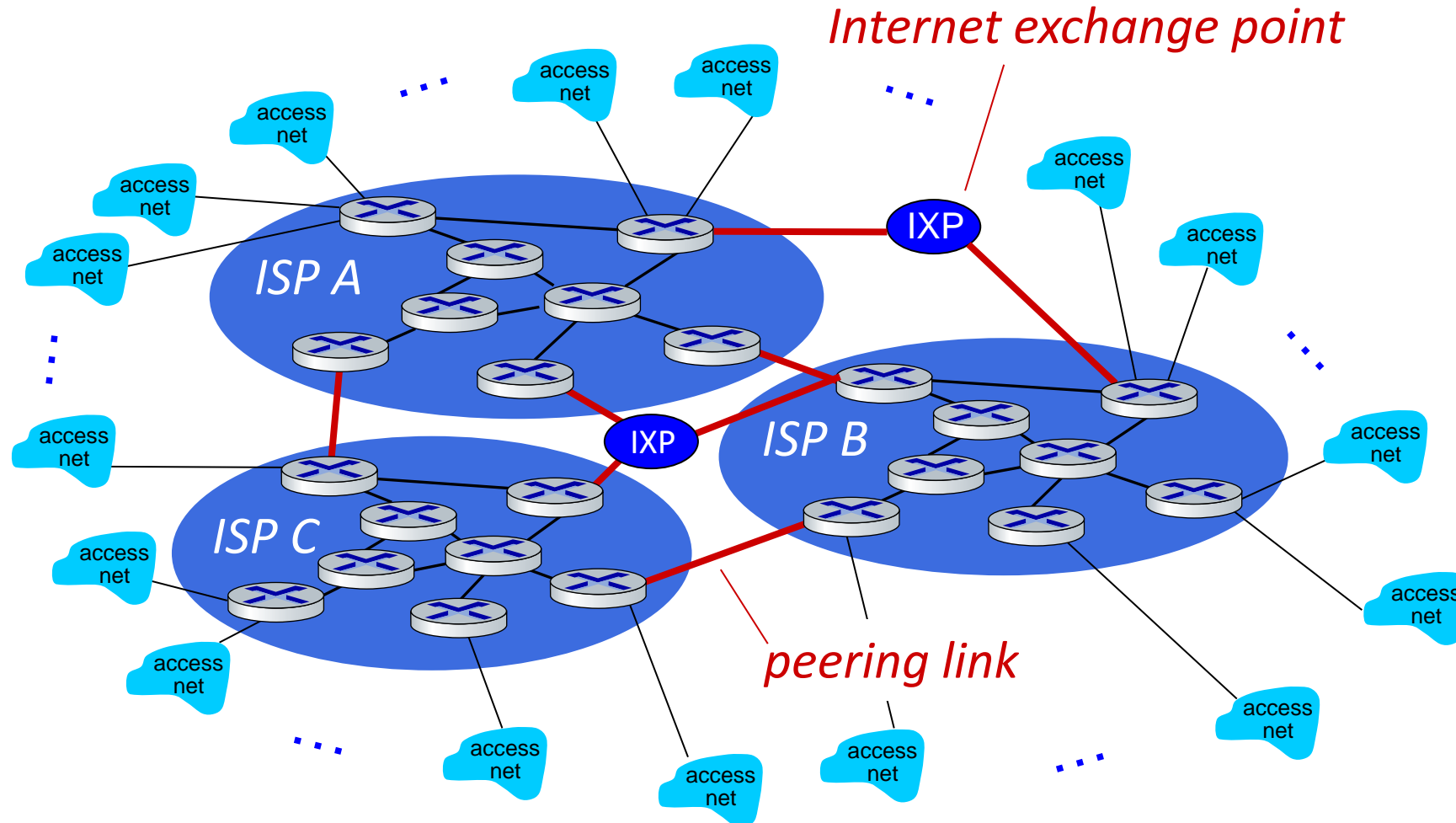
Internet structure: a "network of networks"

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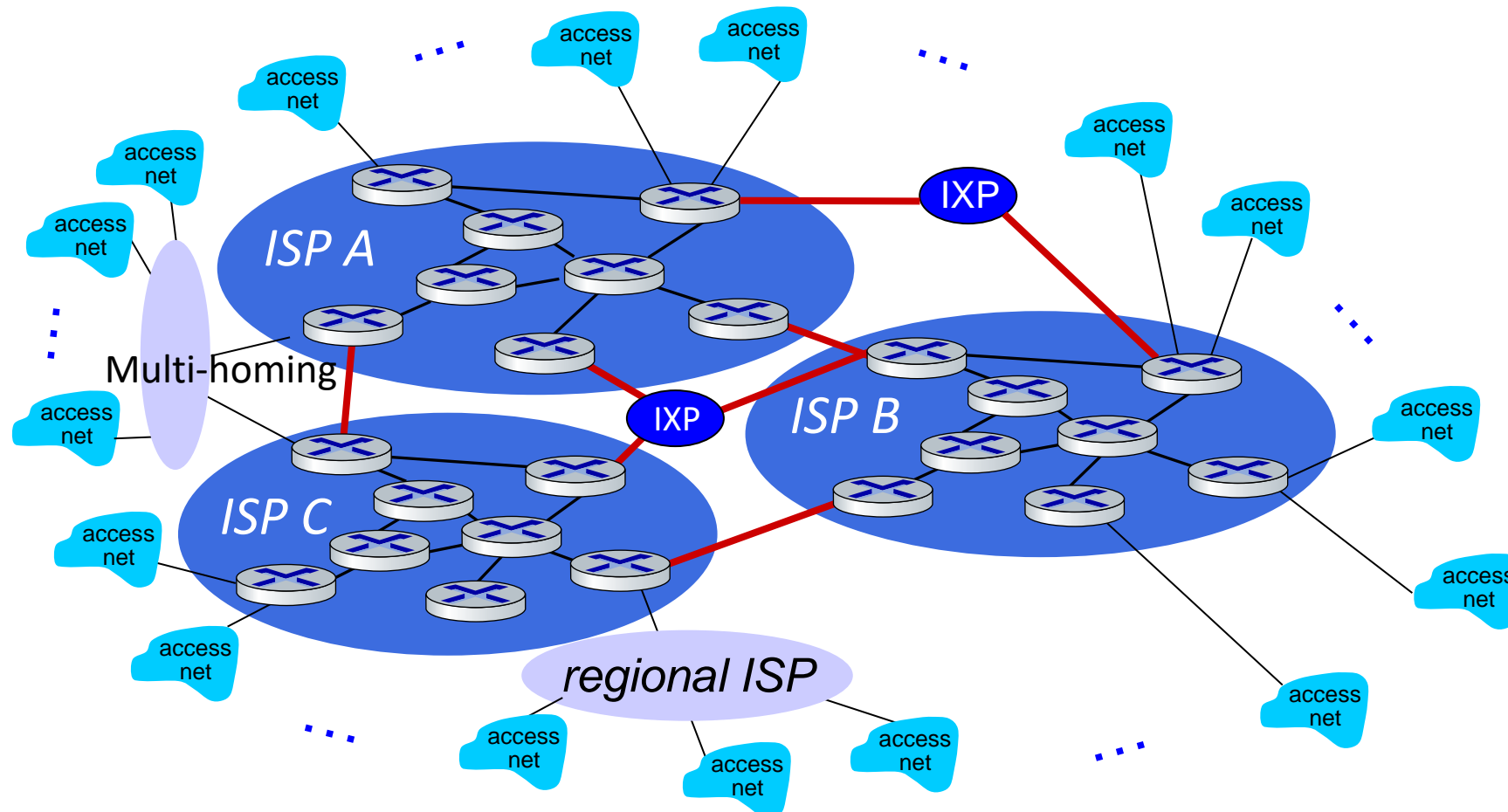
Internet structure: a "network of networks"

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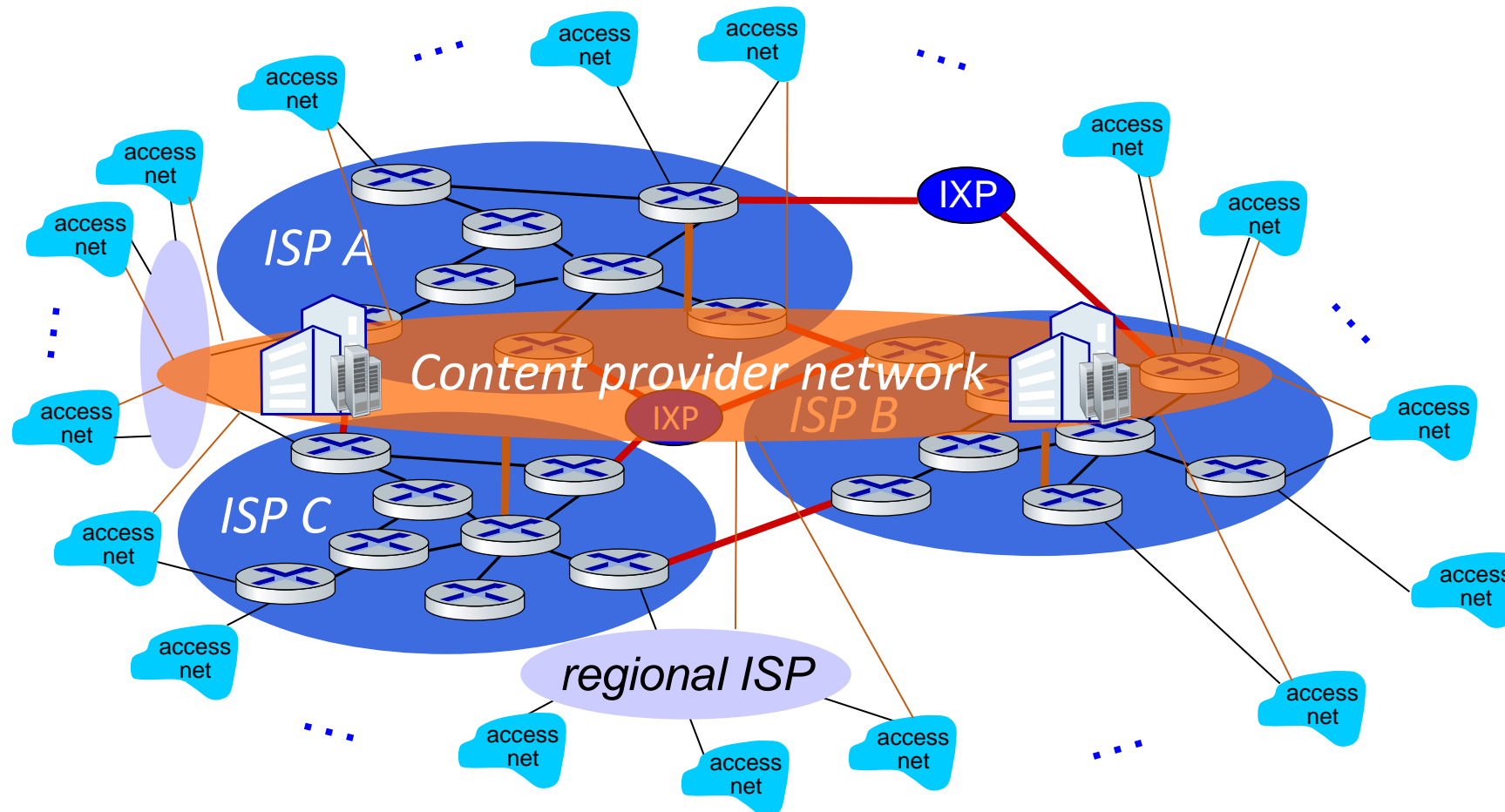
Internet structure: a "network of networks"
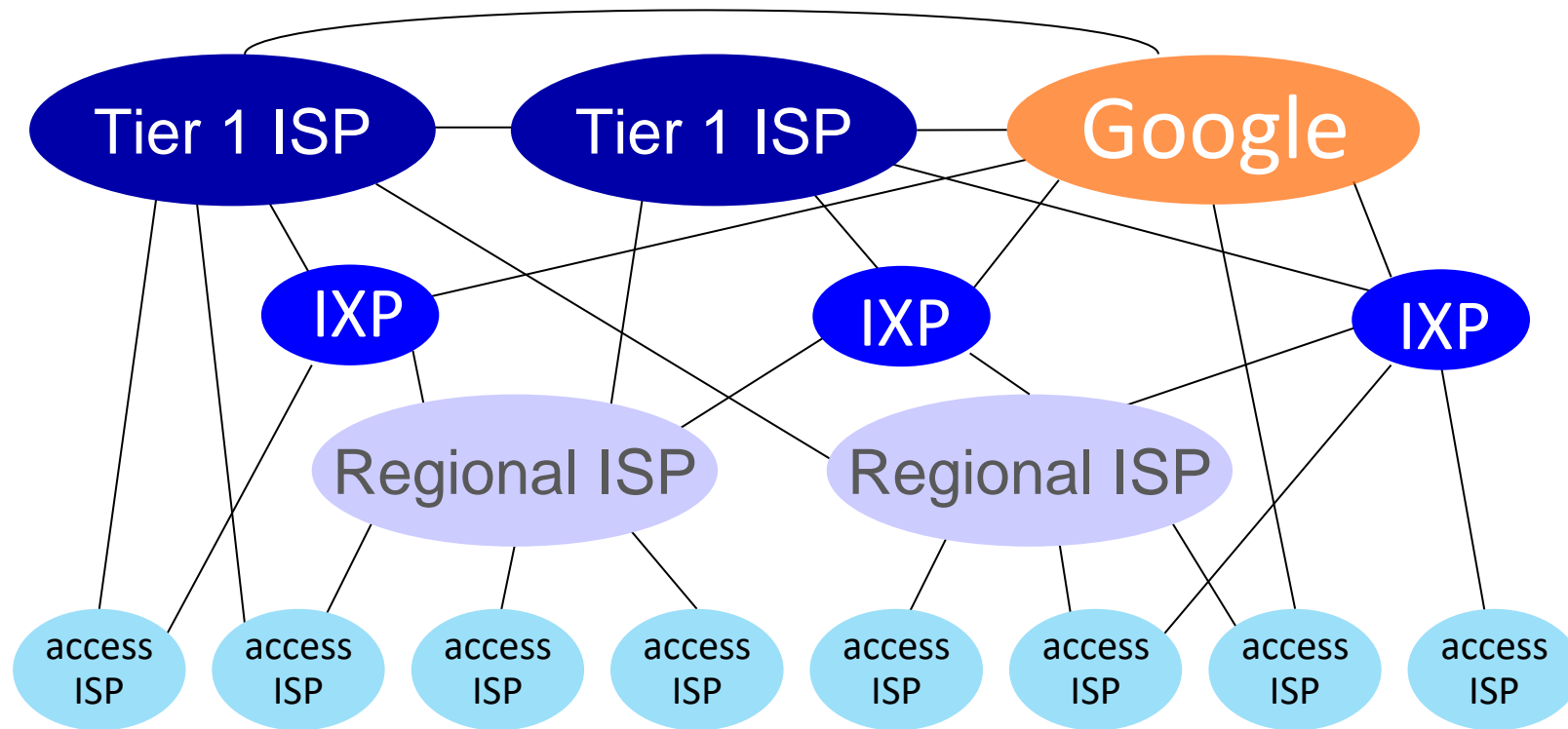
At "center": small # of well-connected large networks

- "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

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
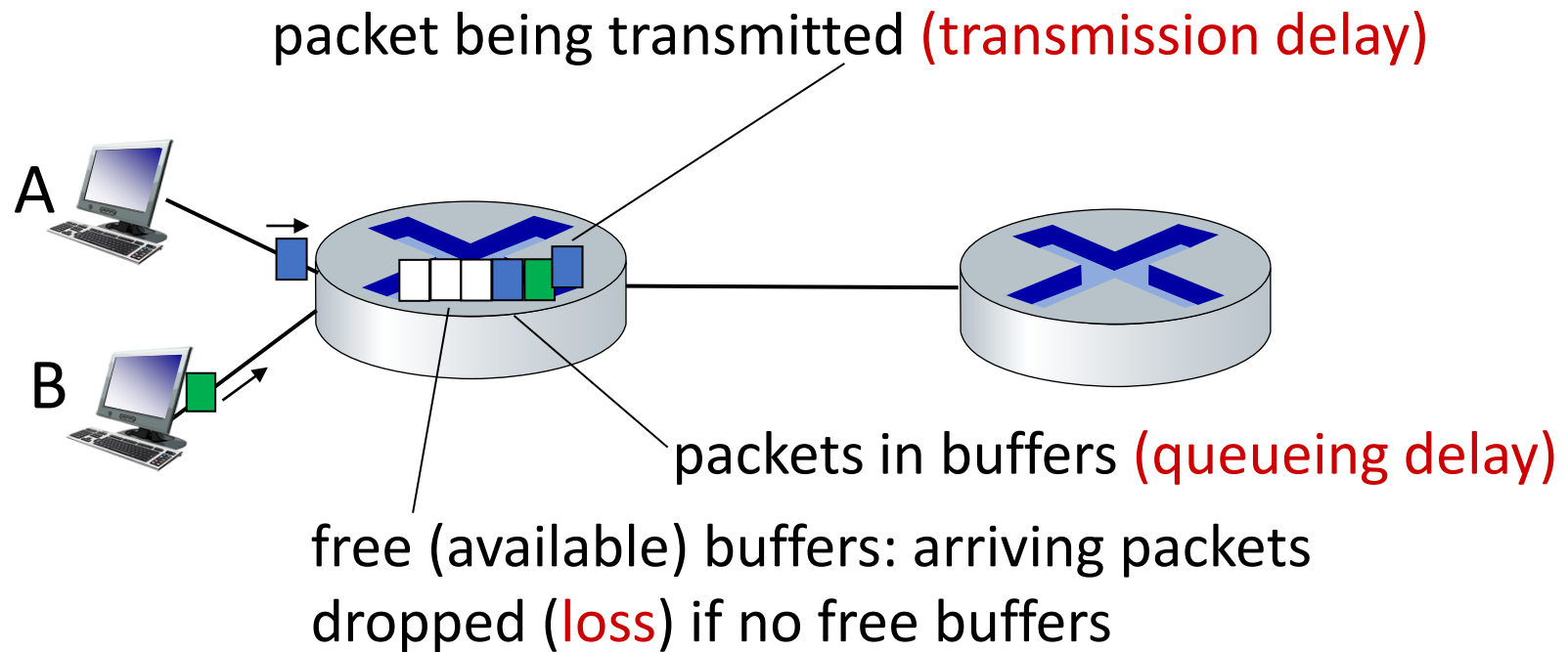
Packet delay: four sources
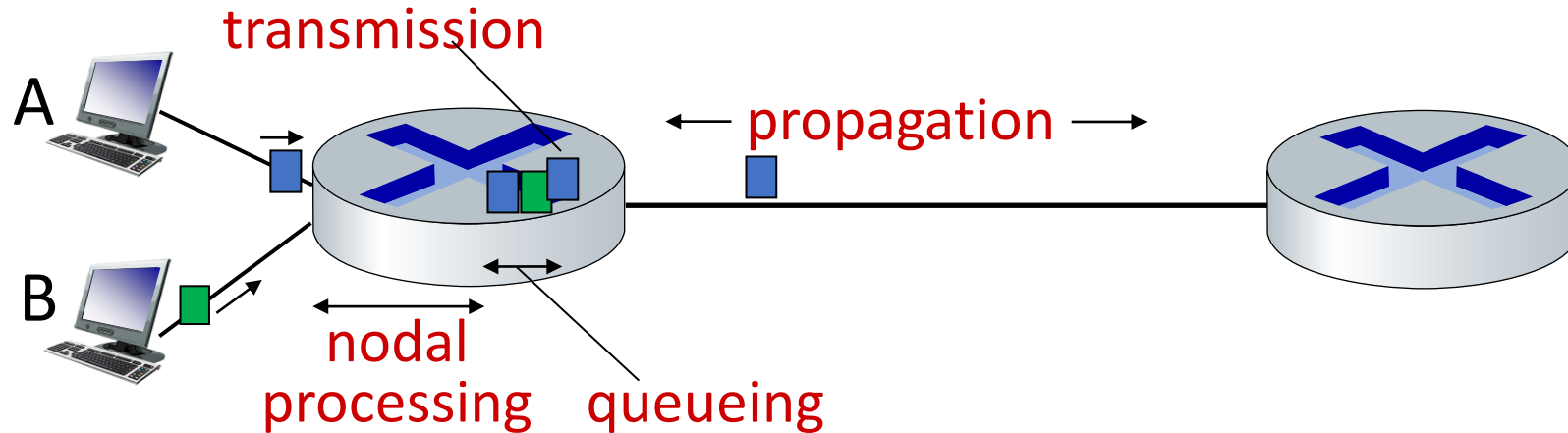
$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

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
- typically in the order of millisecs

Packet delay: four sources
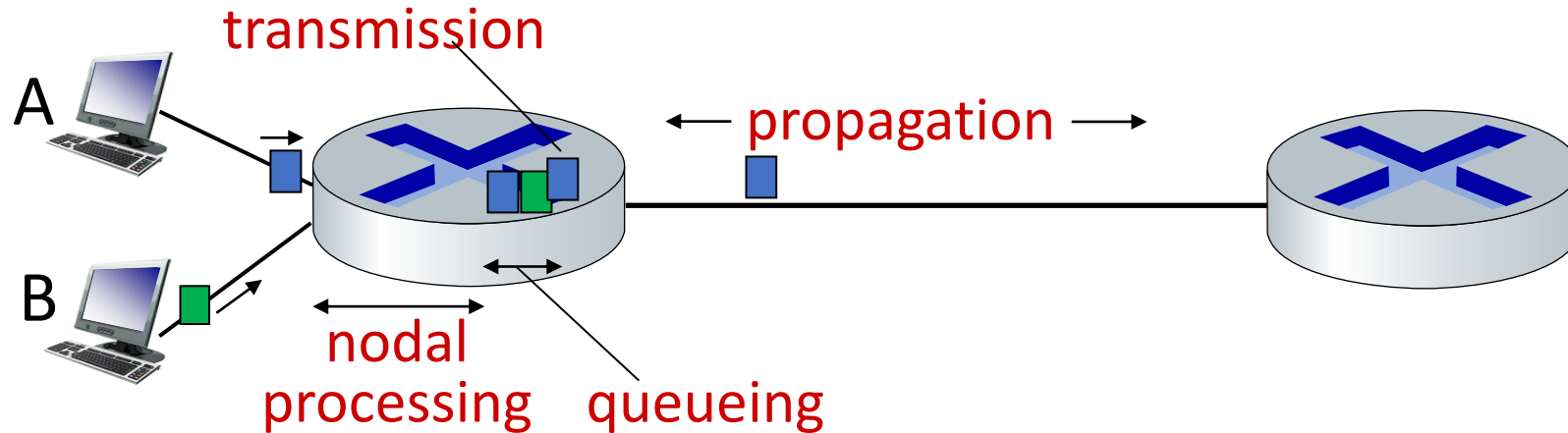
$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

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

Packet queueing delay (revisited)

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

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

traffic intensity = *La/R*   1

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

La/R $\sim$ 0

La/R $\rightarrow$ 1
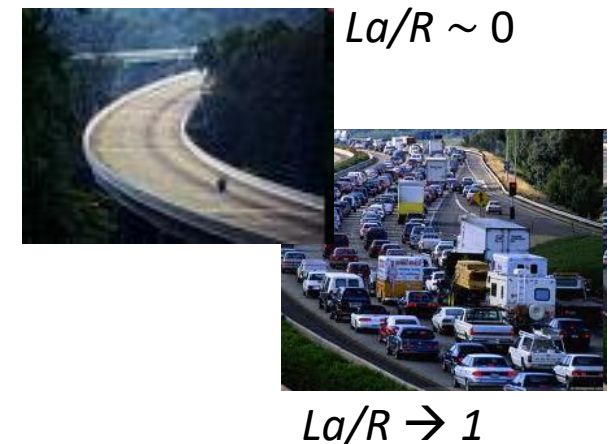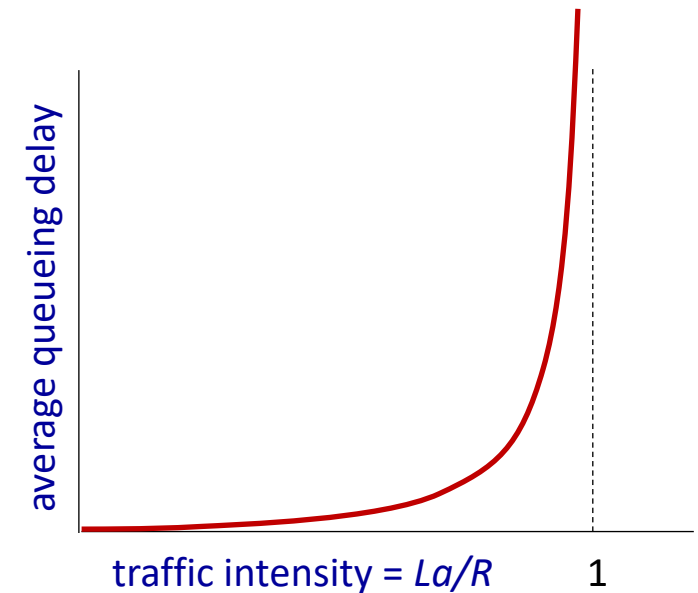
Packet loss

- 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*
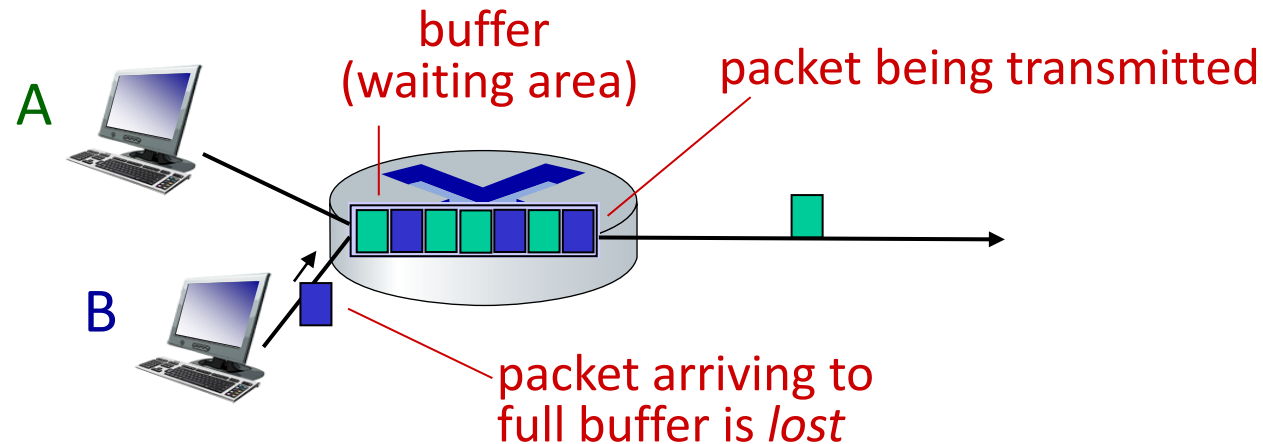
Throughput

- *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)
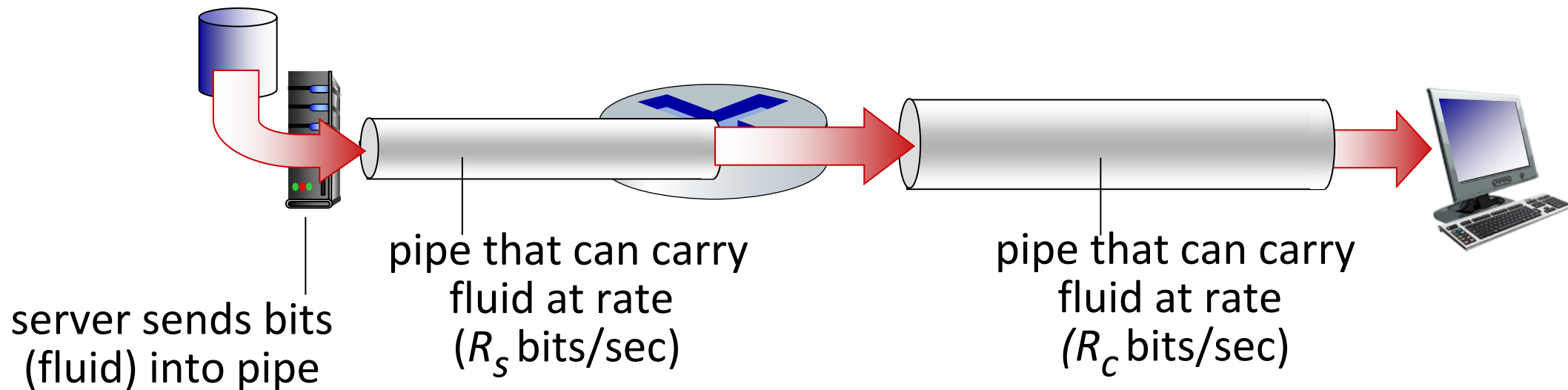
Throughput

$R_s < R_c$ What is average end-end throughput?
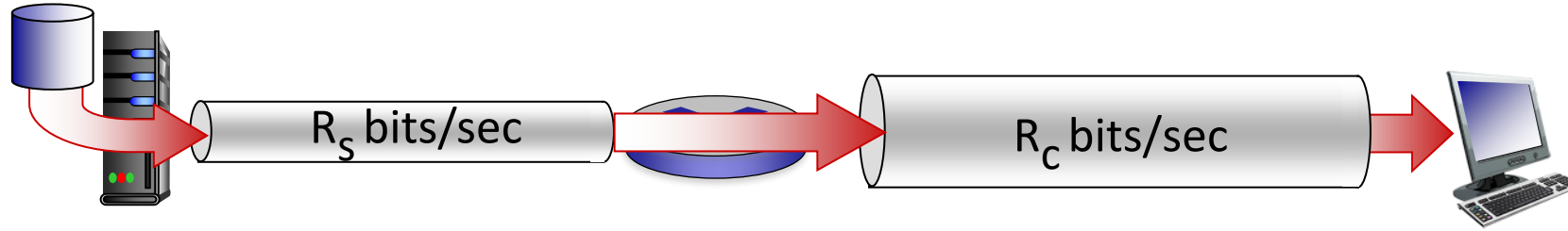
$R_s > R_c$ What is average end-end throughput?
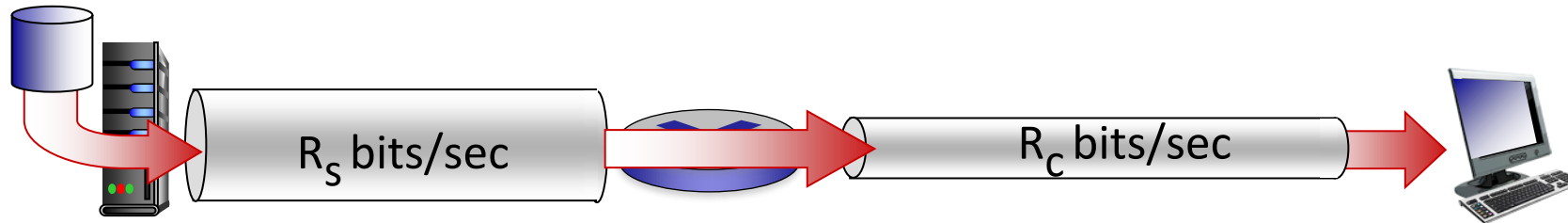
bottleneck link

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

Throughput: network scenario
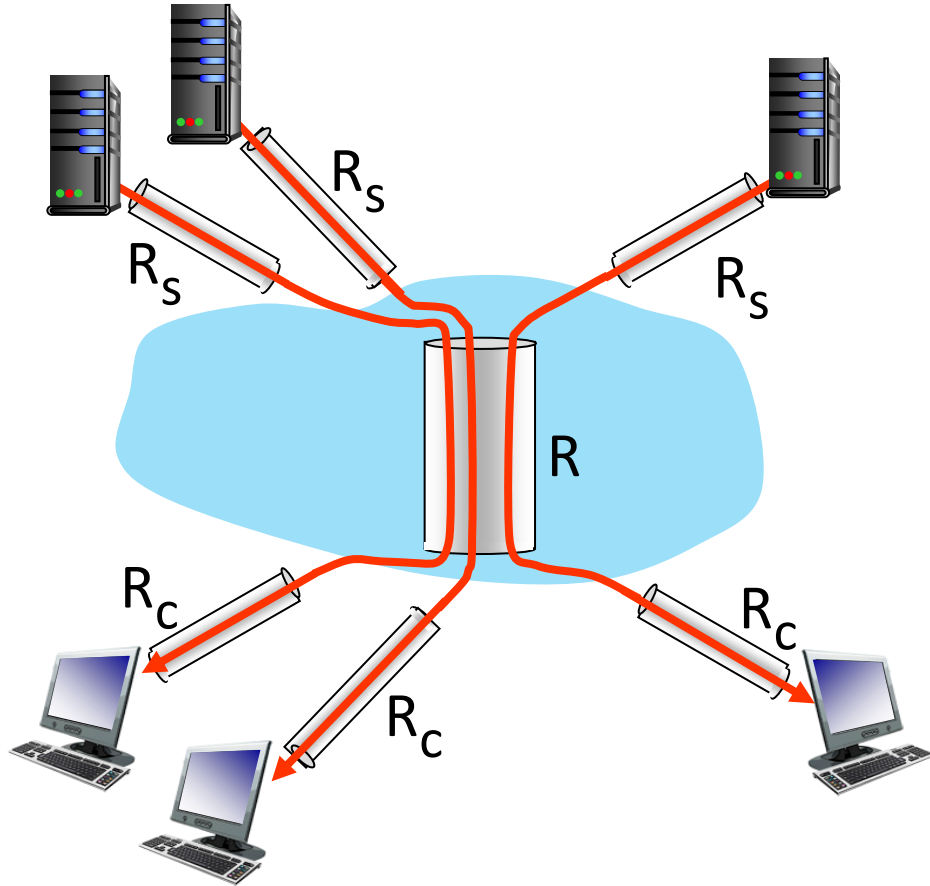
10 connections (fairly) share bottleneck link R bits/sec

- 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/

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
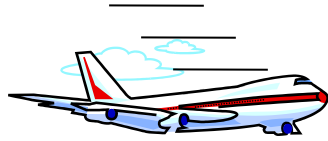
end-to-end transfer of person plus baggage

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, 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)
- *physical:* bits "on the wire"

application
transport
network
link
physical

Services, Layering and Encapsulation

application

transport

network

link

physical

source

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 (payload), 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

application

transport

network

link

physical

destination

Services, Layering and Encapsulation

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

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

source

destination

Services, Layering and Encapsulation

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

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

- 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

source		
application		
transport		
network		
link		
physical		

message

segment

datagram

frame

M

| H_t | M |

| H_n | H_t | M |

| H_l | H_n | H_t | M |

M

| H_t | M |

| H_n | H_t | M |

| H_l | H_n | H_t | M |

destination	
application	
transport	
network	
link	
physical	

source

destination

Encapsulation: an end-end view

message M

segment H_t M

datagram H_n H_t M

frame H_l H_n H_t M

source

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

destination

M

H_t M

H_n H_t M

H_l H_n H_t M

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

H_n H_t M

H_l H_n H_t M

| network |
| link |
| physical |

H_n H_t M
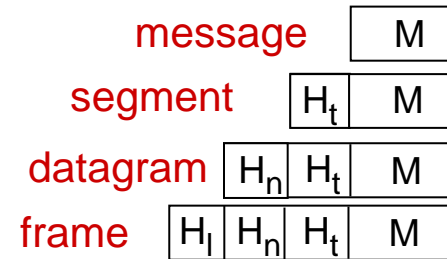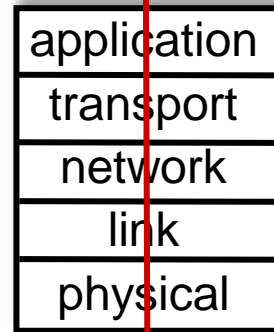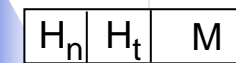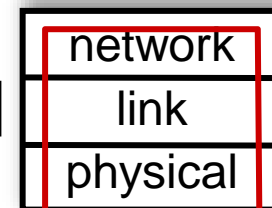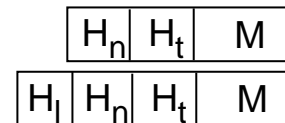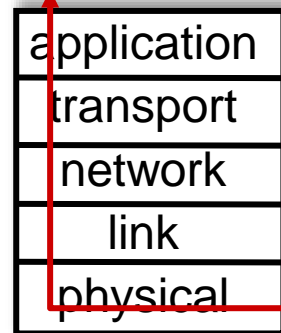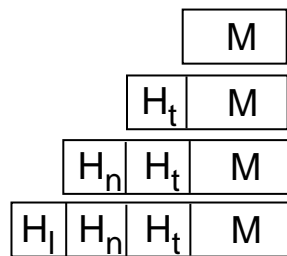
router

Chapter 2
Application Layer

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

Computer Networking: A Top-Down Approach

8th edition
Jim Kurose, Keith Ross
Pearson, 2020

DNS: Domain Name System

people: many identifiers:

- SSN, name, passport #

Internet hosts, routers:

- IP address (32 bit) - used for addressing datagrams
- "name", e.g., cs.umass.edu - used by humans

Q: how to map between IP address and name, and vice versa?

Domain Name System (DNS):

- *distributed database* implemented in hierarchy of many *name servers*
- *application-layer protocol:* hosts, DNS servers communicate to *resolve* names (address/name translation)
  - *note:* core Internet function, implemented as application-layer protocol
  - complexity at network's "edge"

DNS: a distributed, hierarchical database

Root DNS Servers *Root*

... ...

.com DNS servers   .org DNS servers   .edu DNS servers   *Top Level Domain*

...   .

yahoo.com   amazon.com   pbs.org   nyu.edu   umass.edu   *Authoritative*
DNS servers   DNS servers   DNS servers DNS servers DNS servers
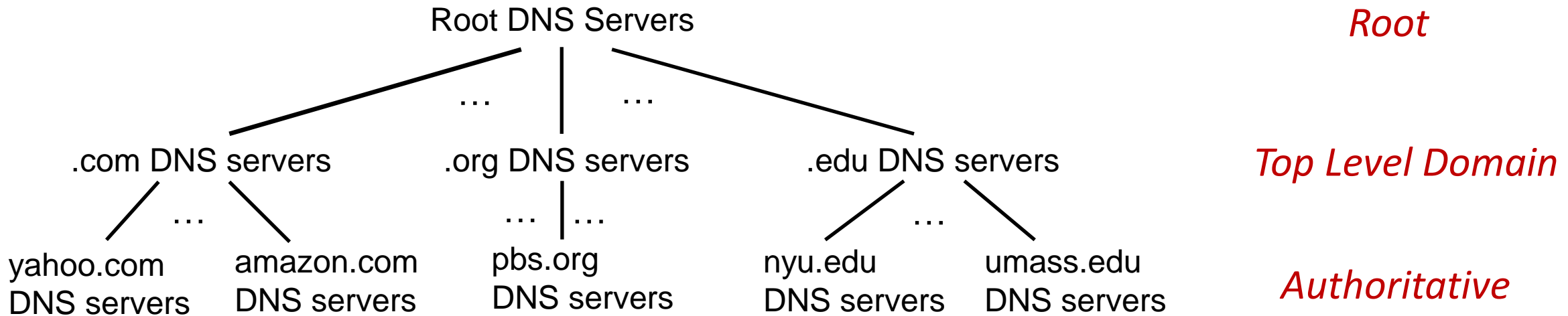
Client wants IP address for www.amazon.com; 1st approximation:

- client queries root server to find .com DNS server

- client queries .com DNS server to get amazon.com DNS server

- client queries amazon.com DNS server to get IP address for www.amazon.com

DNS: root name servers

- official, contact-of-last-resort by name servers that can not resolve name

Root DNS Servers

... ...

.com DNS servers   .org DNS servers   .edu DNS servers

...

yahoo.com   amazon.com   ... ...   nyu.edu   ... ...   umass.edu
DNS servers   DNS servers   pbs.org   DNS servers   DNS servers
                                 DNS servers
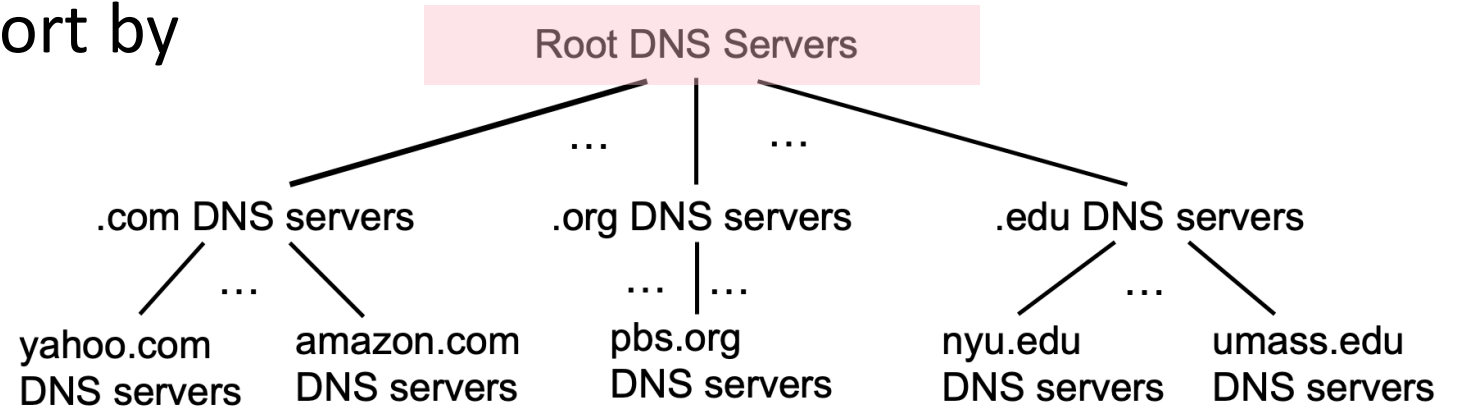
DNS: root name servers

- official, contact-of-last-resort by name servers that can not resolve name

- *incredibly important* Internet function
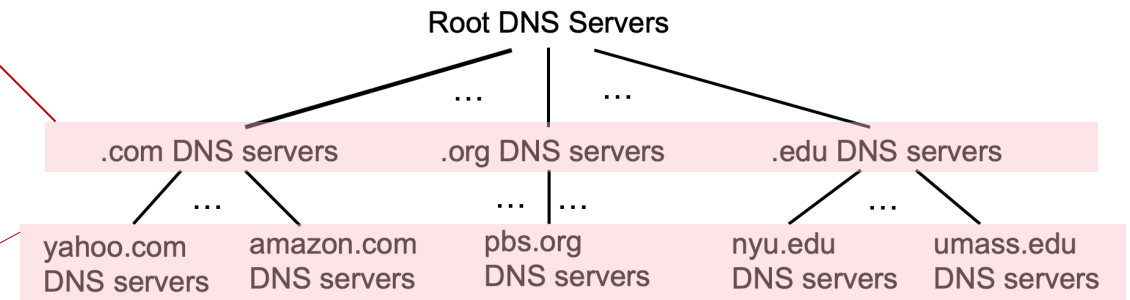 - Internet couldn't function without it!

13 logical root name "servers" worldwide each "server" replicated many times (~20 servers in Italy)

Key:
- ☐ 0 Servers
- 1–10 Servers
- 11–20 Servers
- 21+ Servers
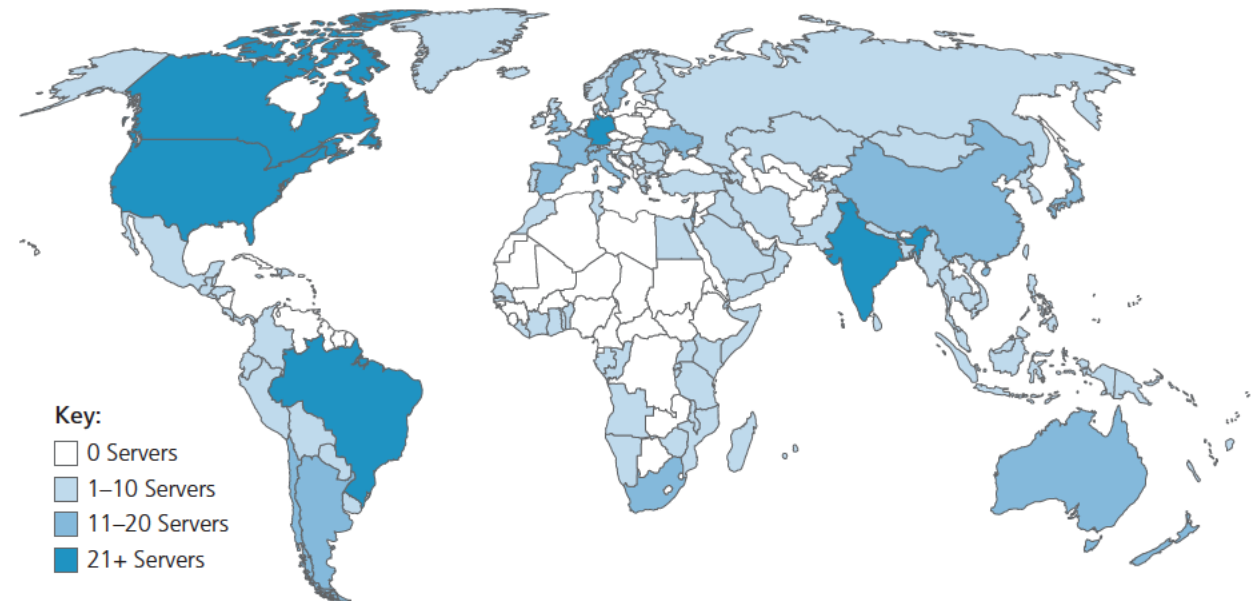
Top-Level Domain, and authoritative servers

Top-Level Domain (TLD) servers:

- responsible for .com, .org, .net, .edu, .aero, .jobs, .museums, and all top-level country domains, e.g.: .cn, .uk, .fr, .ca, .jp

Root DNS Servers

... ...

.com DNS servers   .org DNS servers   .edu DNS servers

...

yahoo.com   amazon.com   pbs.org   nyu.edu   umass.edu
DNS servers   DNS servers   DNS servers   DNS servers   DNS servers

authoritative DNS servers:

- organization's own DNS server(s), providing authoritative hostname to IP mappings for organization's named hosts
- can be maintained by organization or service provider
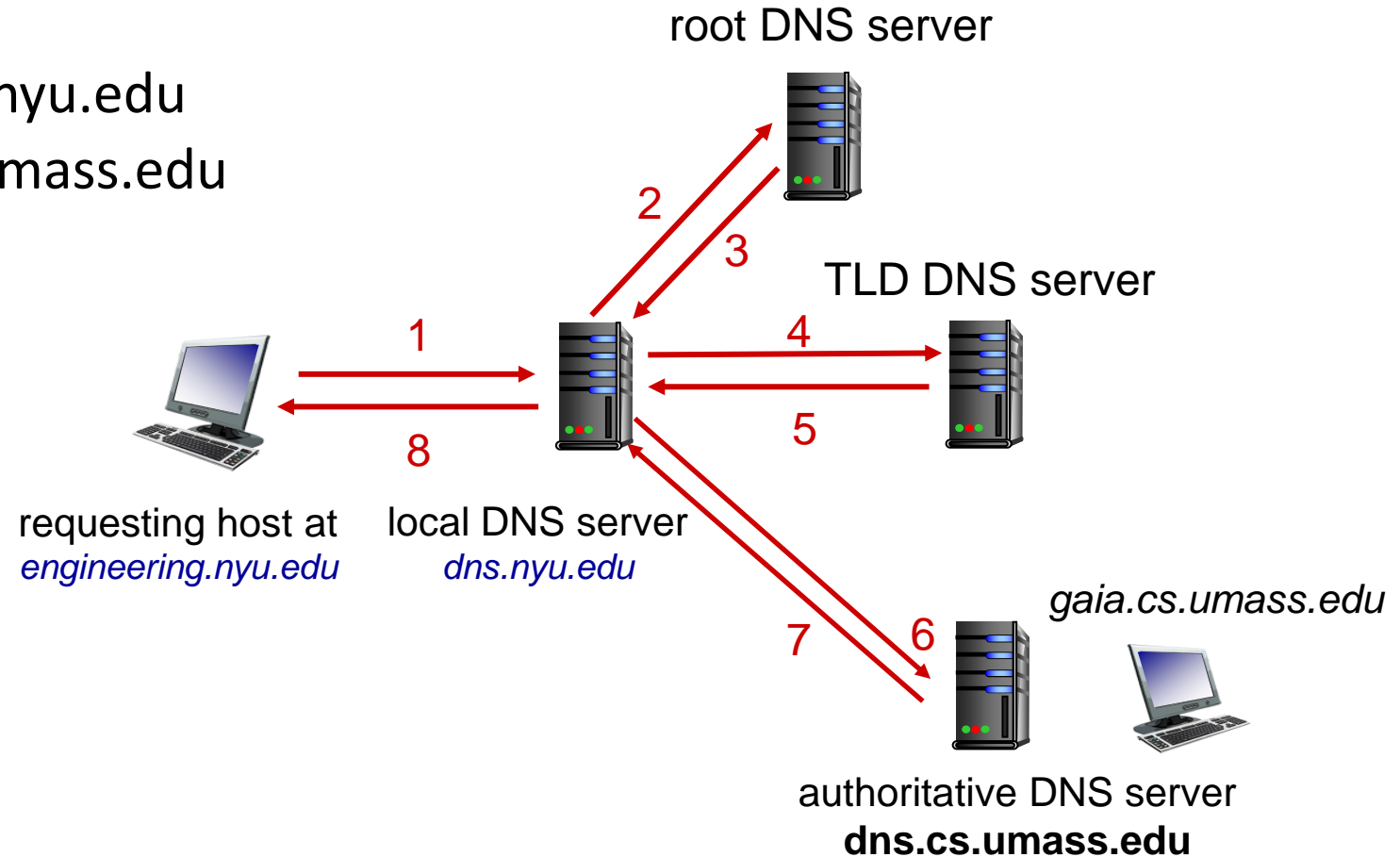
Local DNS name servers

- **when host makes DNS query, it is sent to its *local* DNS server**
  - Local DNS server returns reply, answering:
    - from its local cache of recent name-to-address translation pairs (possibly out of date!)
    - forwarding request into DNS hierarchy for resolution
  - each ISP has local DNS name server; to find yours:
    - MacOS: `% scutil --dns`
    - Windows: `>ipconfig /all`

DNS name resolution: iterated query

Example: host at engineering.nyu.edu
wants IP address for gaia.cs.umass.edu

Iterated query:

- contacted server replies with IP address of server to contact
- "I don't know this name, but ask this server"

root DNS server

TLD DNS server

requesting host at
engineering.nyu.edu

local DNS server
dns.nyu.edu

gaia.cs.umass.edu

authoritative DNS server
dns.cs.umass.edu

DNS name resolution: recursive query

Example: host at engineering.nyu.edu
wants IP address for gaia.cs.umass.edu

Recursive query:
- puts burden of name resolution on contacted name server
- heavy load at upper levels of hierarchy?

root DNS server

2
3
7
6

1
8

requesting host at
engineering.nyu.edu

local DNS server
dns.nyu.edu

TLD DNS server

5
4

gaia.cs.umass.edu

authoritative DNS server
dns.cs.umass.edu