CS3640

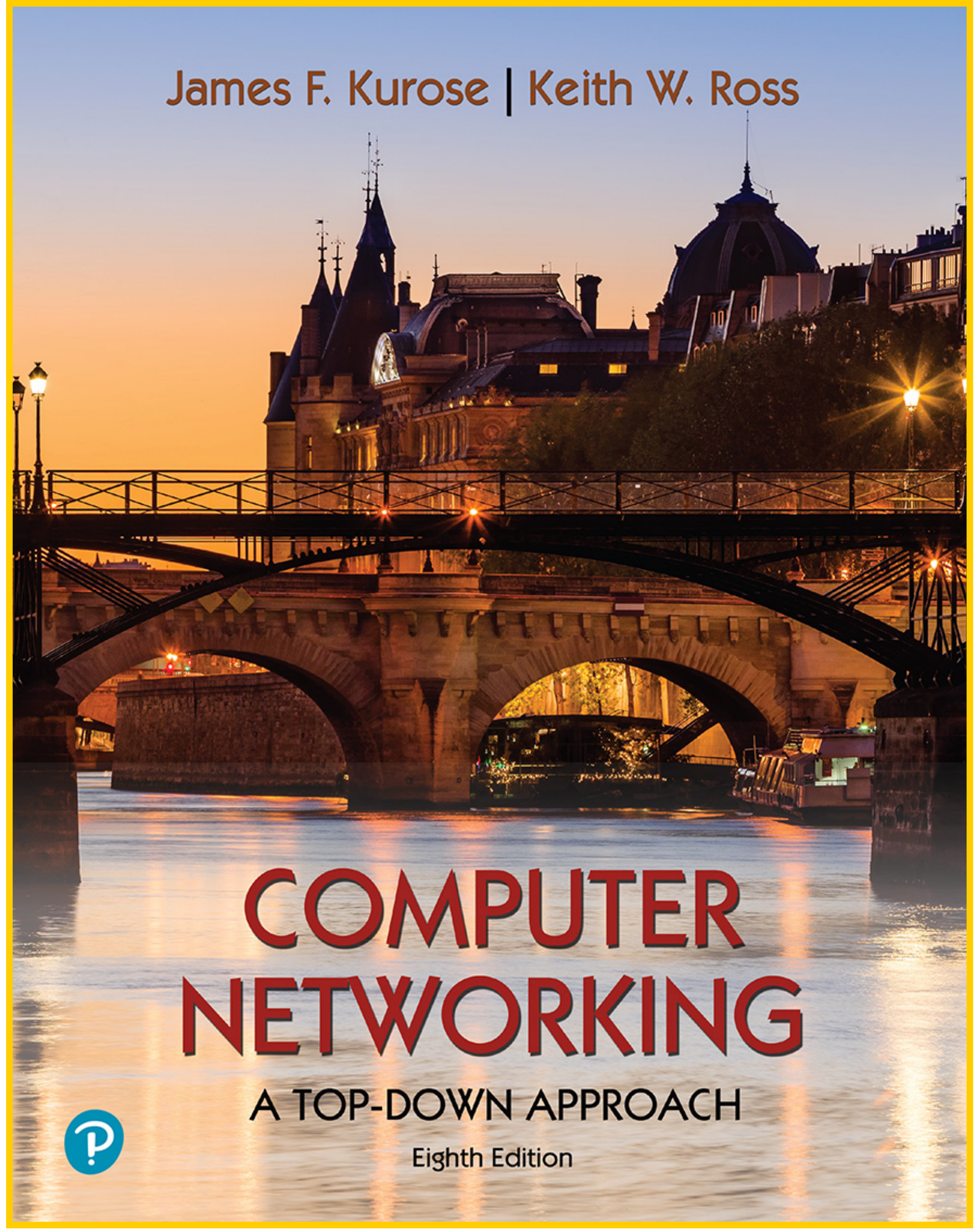
Overview (3): Performance & Protocols

Prof. Supreeth Shastri
Computer Science
The University of Iowa

Lecture goals

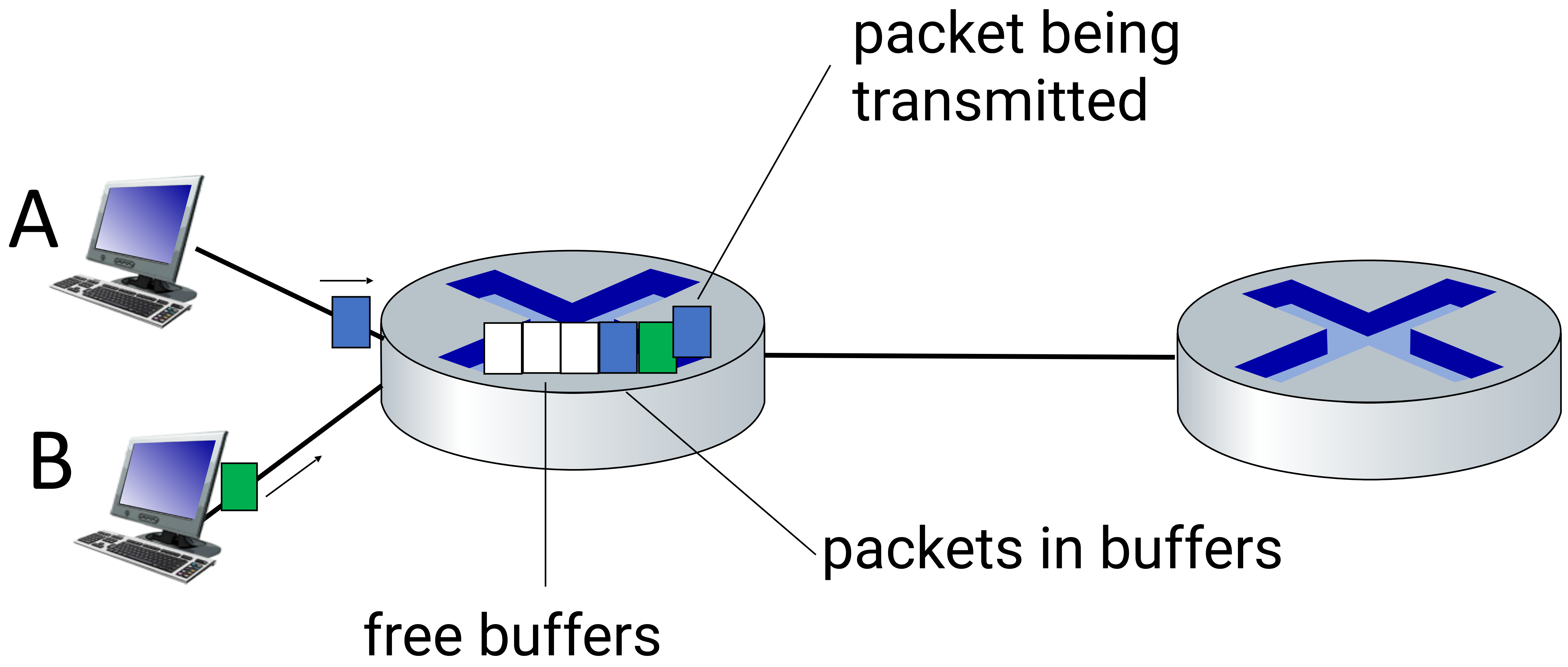
Continuing our in-depth exploration into the structure and functioning of the Internet

- *Performance: delay and throughput*
- *Protocol architecture*



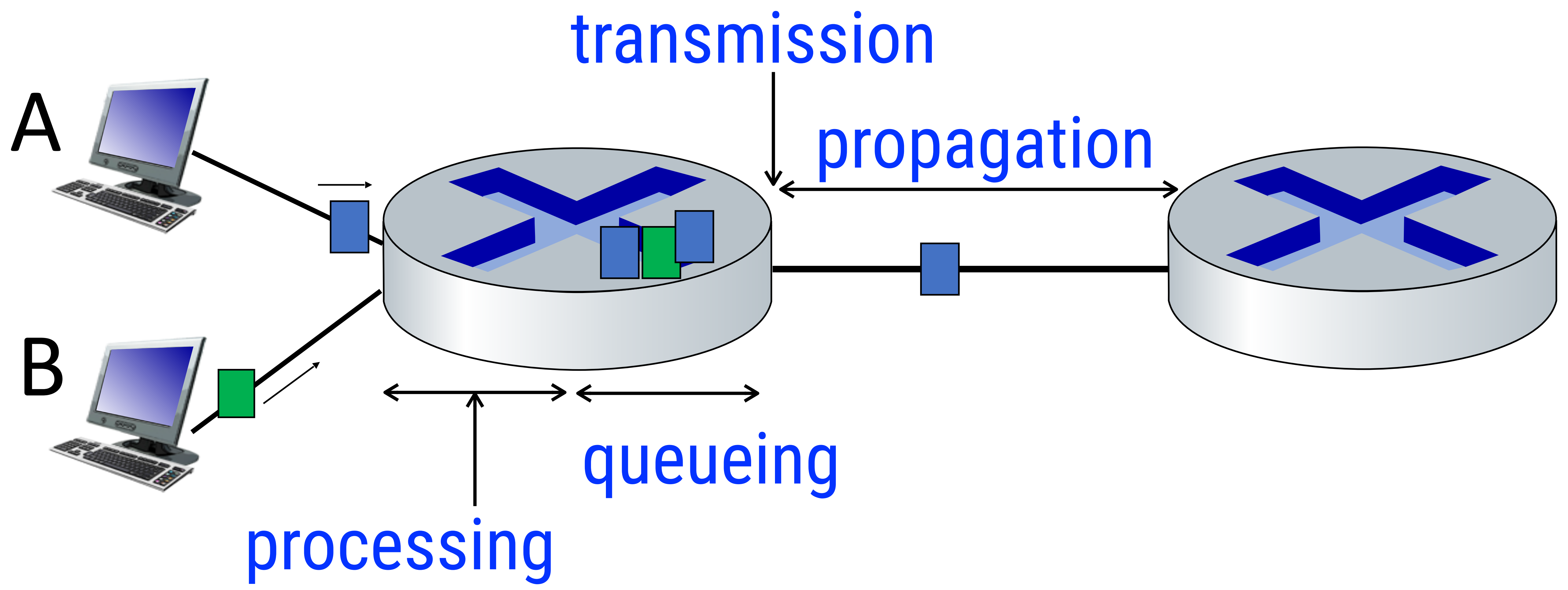
Chapter 1.4 - 1.5

Queueing and loss in packet switching



- packets are queued in when arrival rate for a link exceeds its output capacity
- packet are dropped when memory to hold queued packets fills up
  - *lost packet may be retransmitted by previous node, by source, or not at all*

Packet delay: four sources



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

d_{proc}: processing

- *check bit errors; determine output link*
- *typically < microseconds*

d_{queue}: queueing

- *time waiting at output link for transmission*
- *depends on router's congestion level*

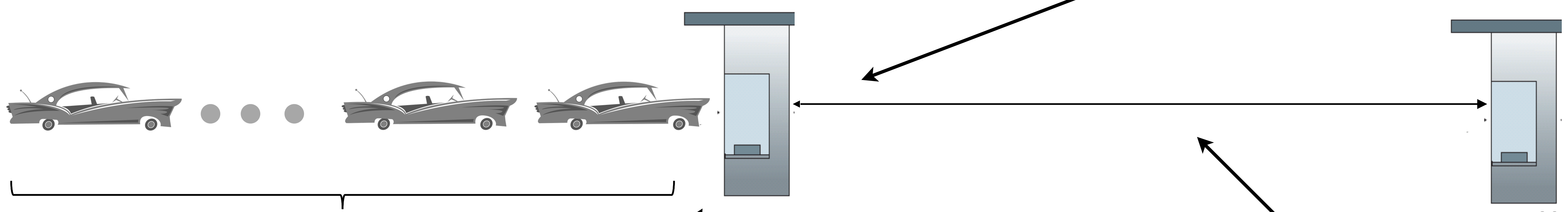
d_{trans}: transmission

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

d_{prop}: propagation

- *d: length of physical link*
- *s: propagation speed (~2x10^8 m/sec)*
- $d_{prop} = d/s$

Transmission vs. propagation delays



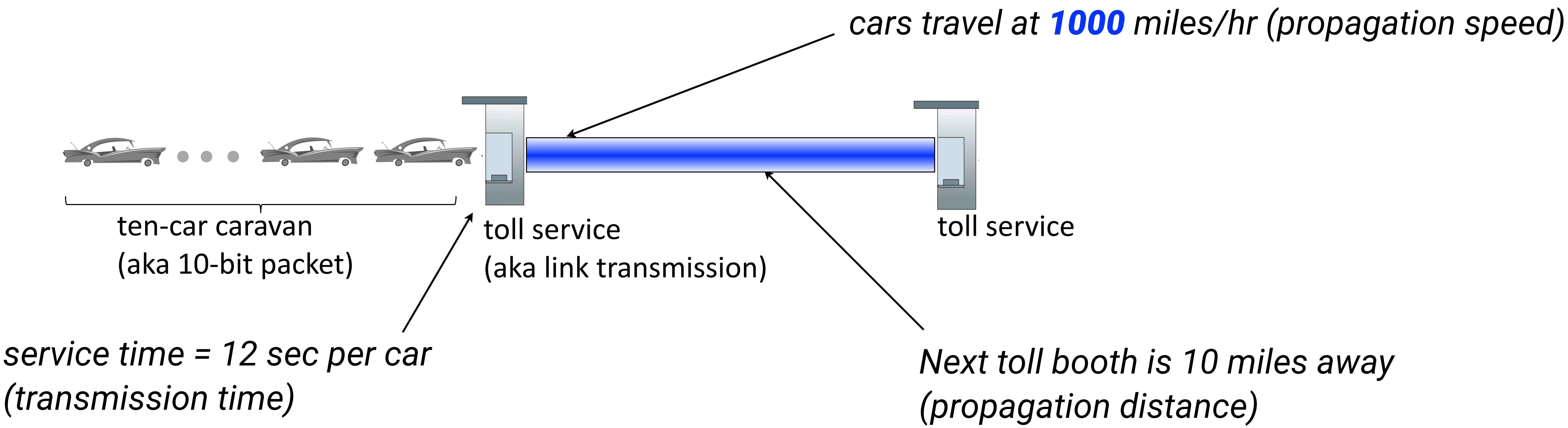
If the entire caravan arrives at the first toll booth at time t = 0, how long until caravan is lined up before the second toll booth?

*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: 10miles/(10miles/hr) = 1 hr

1 hr 2 m

Transmission vs. propagation delays



If the entire caravan arrives at the first toll booth at time t = 0, how long until caravan is lined up before the second 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: 10miles/(1000miles/hr) = 36 sec

156 s

Understanding traffic intensity

- a: avg. arrival rate (*packets/sec*)
- L: avg. packet length (*bits/packet*)
- R: link transmission rate (*bits/sec*)

traffic intensity

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

La/R ~ 0

Avg delay ≈ *none*

La/R → 1

Avg delay *depends on arrival distribution*

La/R > 1

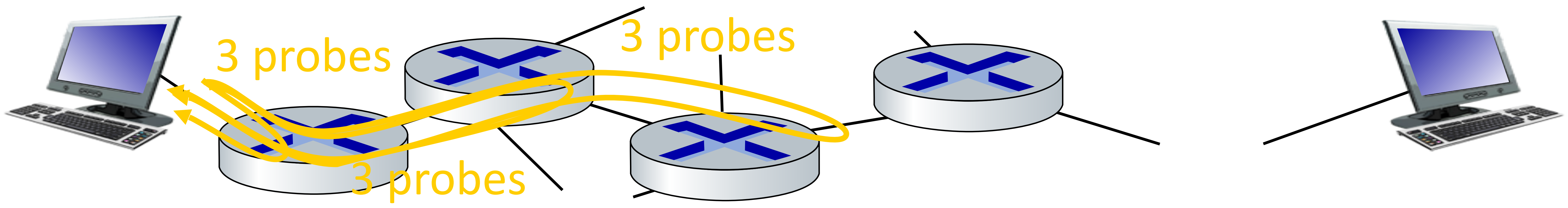
Avg delay → *infinity*

Quantifying delays in the “real” Internet

traceroute: *a tool that provides delay measurement from source to router along end-end Internet path towards destination.*

For all i

- *(i)* *send three packets that will reach router i on path towards destination (with time-to-live field set to i)*
- *(ii)* *router i will return packets to sender*
- *(iii)* *sender measures time interval between transmission and reply*



Quantifying delays in the “real” Internet

traceroute: from fastx01.divms.uiowa.edu to www.google.com

Router hierarchy within uiowa

uiowa’s ISP

3-delay measurements

```
[sshastri@fastx01 sshastri]$ traceroute www.google.com
traceroute to www.google.com (172.217.8.164), 30 hops max, 60 byte packets
 1  rtr-lc-1-production.net.uiowa.edu (128.255.58.1)  0.733 ms  0.776 ms  0.966 ms
 2  rtr-core-lc.net.uiowa.edu (128.255.2.44)  0.529 ms  0.626 ms  0.818 ms
 3  rtr-border-bsb.net.uiowa.edu (128.255.2.225)  0.377 ms  0.393 ms  0.406 ms
 4  r-equinix-isp-ae0-2236.wiscnet.net (216.56.50.73)  5.406 ms  5.406 ms  5.410 ms
 5  72.14.218.180 (72.14.218.180)  5.423 ms  5.426 ms  5.476 ms
 6  108.170.244.1 (108.170.244.1)  5.317 ms 108.170.243.225 (108.170.243.225)  6.289 ms  6.292 ms
 7  72.14.232.153 (72.14.232.153)  5.386 ms  5.346 ms  5.343 ms
 8  ord37s08-in-f4.1e100.net (172.217.8.164)  5.306 ms  5.296 ms  5.292 ms
[sshastri@fastx01 sshastri]$
```

Who is answering for google?

Why aren’t the delays strictly increasing?

traceroute demo

Quantifying delays in the “real” Internet

traceroute: from fastx01.divms.uiowa.edu to www.wimbledon.org

uiowa is using more then one ISP

```
[sshastri@fastx01 sshastri]$ traceroute www.wimbledon.org
traceroute to www.wimbledon.org (104.114.79.50), 30 hops max, 60 byte packets
 1  rtr-lc-1-production.net.uiowa.edu (128.255.58.1)  0.921 ms  0.921 ms  0.906 ms
 2  rtr-core-lc.net.uiowa.edu (128.255.2.44)  0.610 ms  0.872 ms  1.027 ms
 3  rtr-border-bsb.net.uiowa.edu (128.255.2.225)  0.415 ms  0.344 ms  0.404 ms
 4  et-5-1-5-102.cr1-min1.ip4.gtt.net (208.116.156.121)  8.262 ms  8.274 ms  8.261 ms
 5  ae19.cr9-chi1.ip4.gtt.net (141.136.108.189)  16.754 ms  27.958 ms  16.725 ms
 6  ip4.gtt.net (98.124.183.18)  25.264 ms  24.924 ms  24.907 ms
 7  ae3.ctl-ord3.netarch.akamai.com (23.203.151.229)  17.680 ms  17.637 ms  15.612 ms
 8  a104-114-79-50.deploy.static.akamaitechnologies.com (104.114.79.50)  11.331 ms  11.332 ms  11.329 ms
```

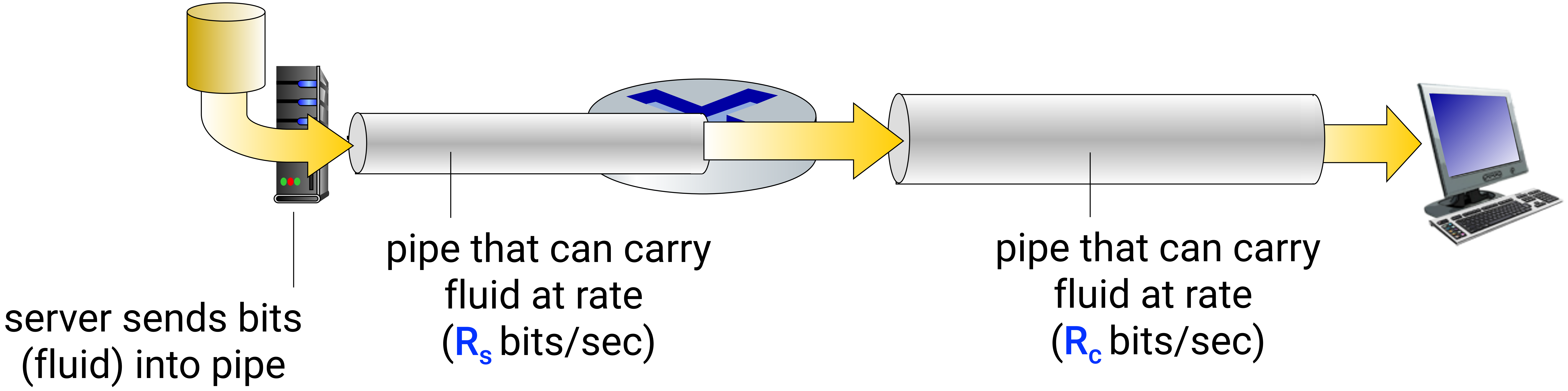
CDN! No cross Atlantic traffic

Delays are longer than google.com

Throughput

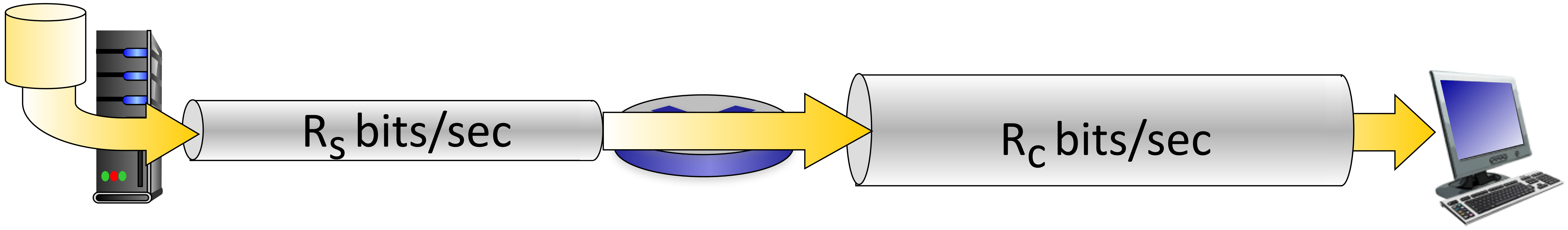
Rate (*measured in bits/sec*) at which bits are being sent from sender to receiver

- **instantaneous**: rate at a given point in time
- **average**: rate over a longer period of time

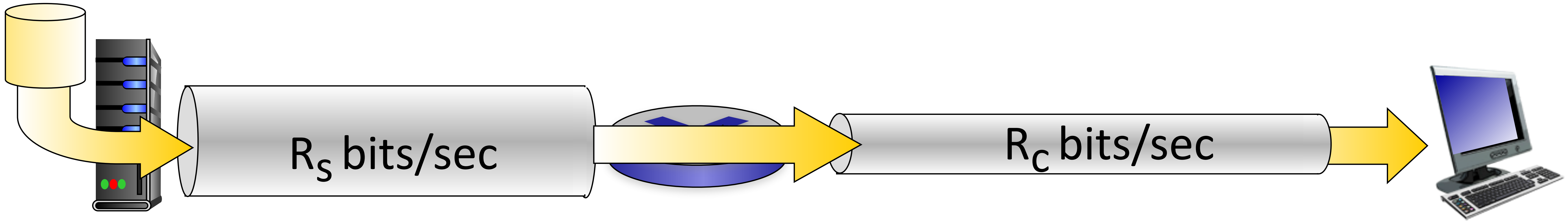


Throughput: *bottleneck link*

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



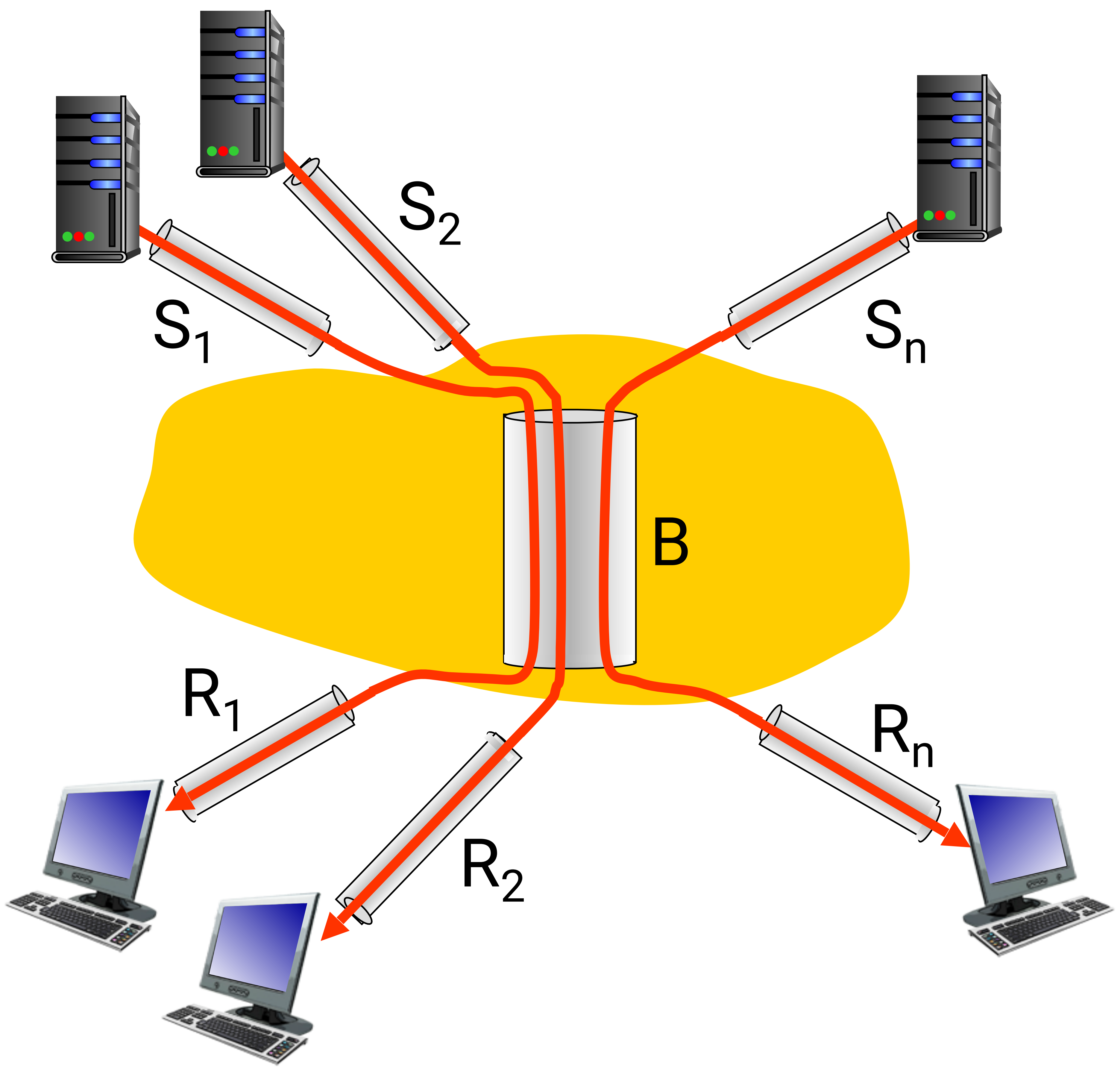
$R_s > R_c$ What is average end-end throughput?



bottleneck link

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

Throughput: *network scenario*



- *n connections fairly share the backbone link (B bits/sec)*
- *End-end throughput for connection k = min(S_k,R_k,B/n)*
- *in practice: S_k or R_k is often the bottleneck*

Protocol layering

What is **layering**?

An approach to **designing complex systems**

- *allows identifying system's components and explicitly defining their relationship*

Modularization eases maintenance and updating of system

- *change in layer's service implementation: transparent to rest of system*

Why is layering useful for **computer networks**?

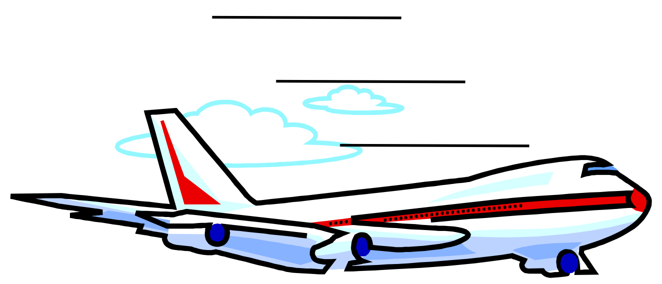
Computer networks have multitude of components interacting with each other

- *host devices, routers, links, protocols, applications, policies, and so on*

Internet is arguably the **largest engineered system** ever created by humans!

A layered system: air travel

a complex system involving people, goods, airplanes, airports, and services



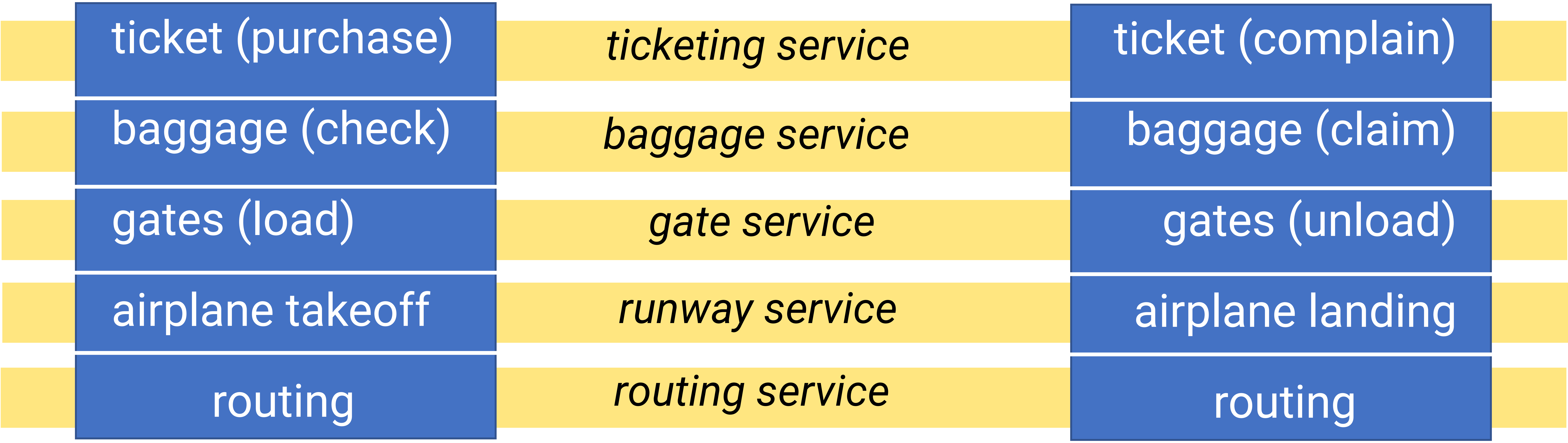
end-to-end transfer of person plus baggage

ticketing (purchase)	ticketing (complain)
baggage (check)	baggage (claim)
gates (load)	gates (unload)
airplane takeoff	airplane landing

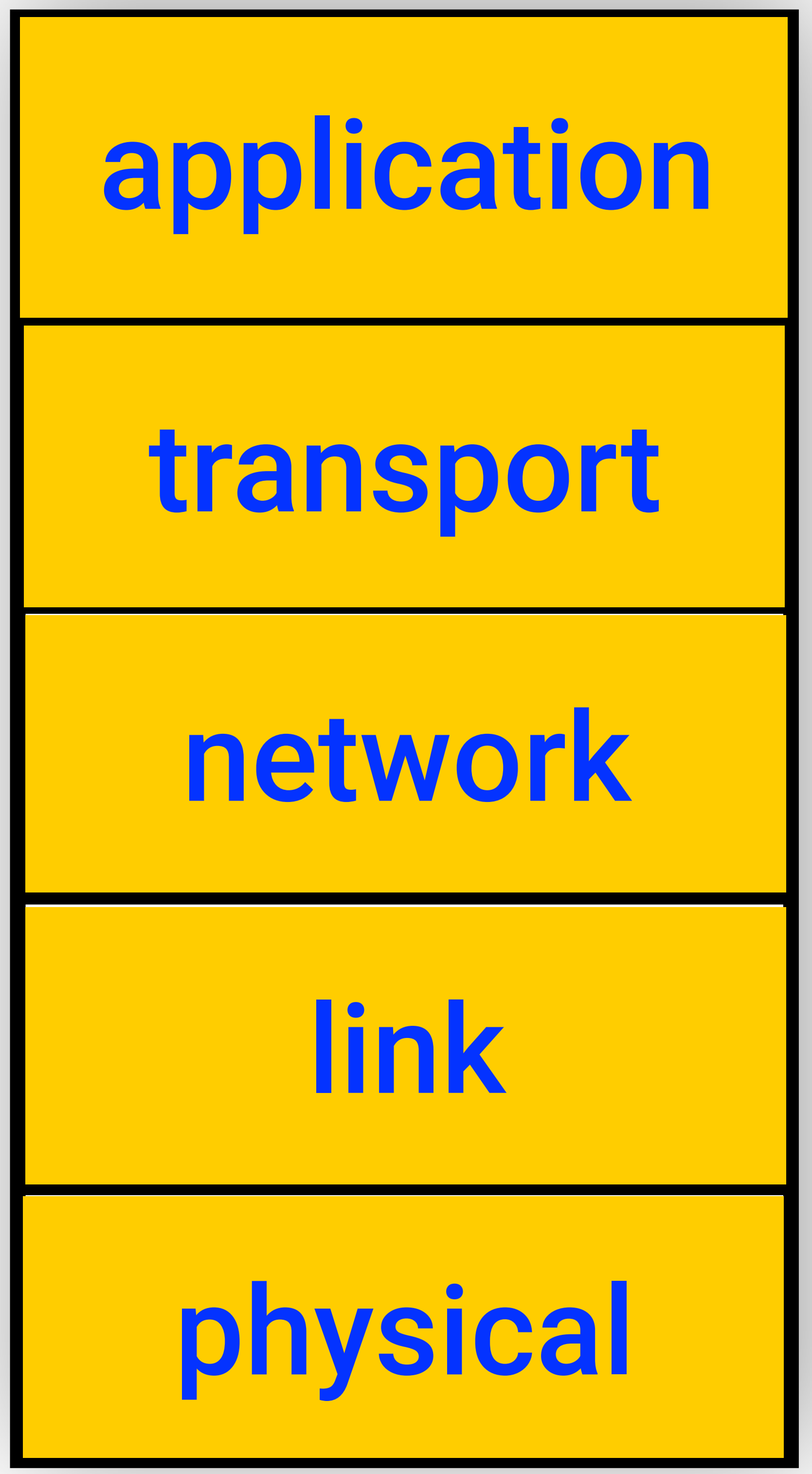
airplane routing

A layered system: air travel

each layer implements a service and relies on services provided by layer below

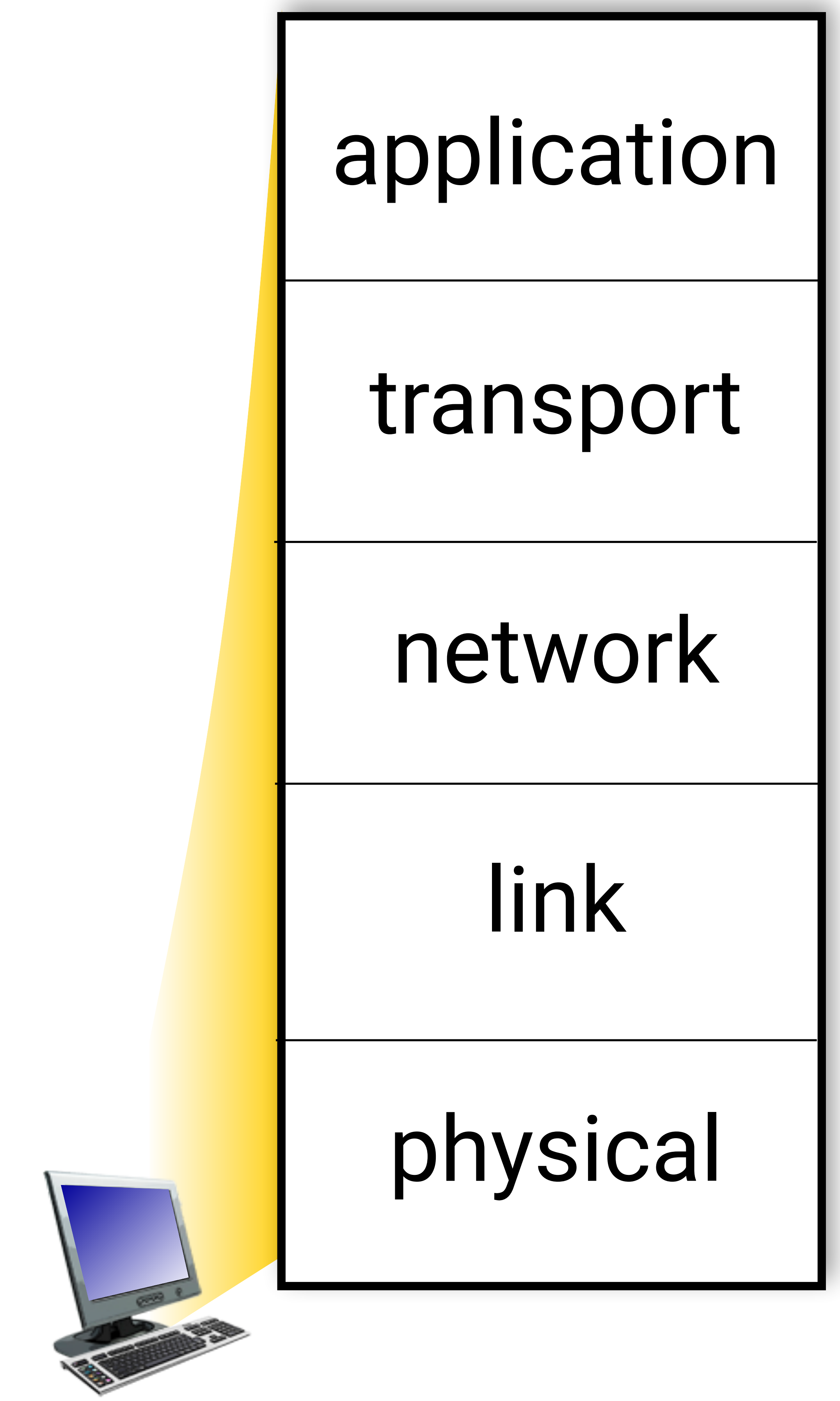


The five layer architecture of the Internet



- **Application:** *supporting network applications. E.g., HTTP, IMAP, SMTP, DNS*
- **Transport:** *process to process data transfer. E.g.,TCP, UDP*
- **Network:** *routing of datagrams from source machine to destination. E.g., IP, IPv6*
- **Link:** *deliver data between neighboring network elements. E.g., Ethernet, 802.11 (WiFi)*
- **Physical:** *bits "on the wire". E.g., 10BASE-T*

Protocol layering and services



source

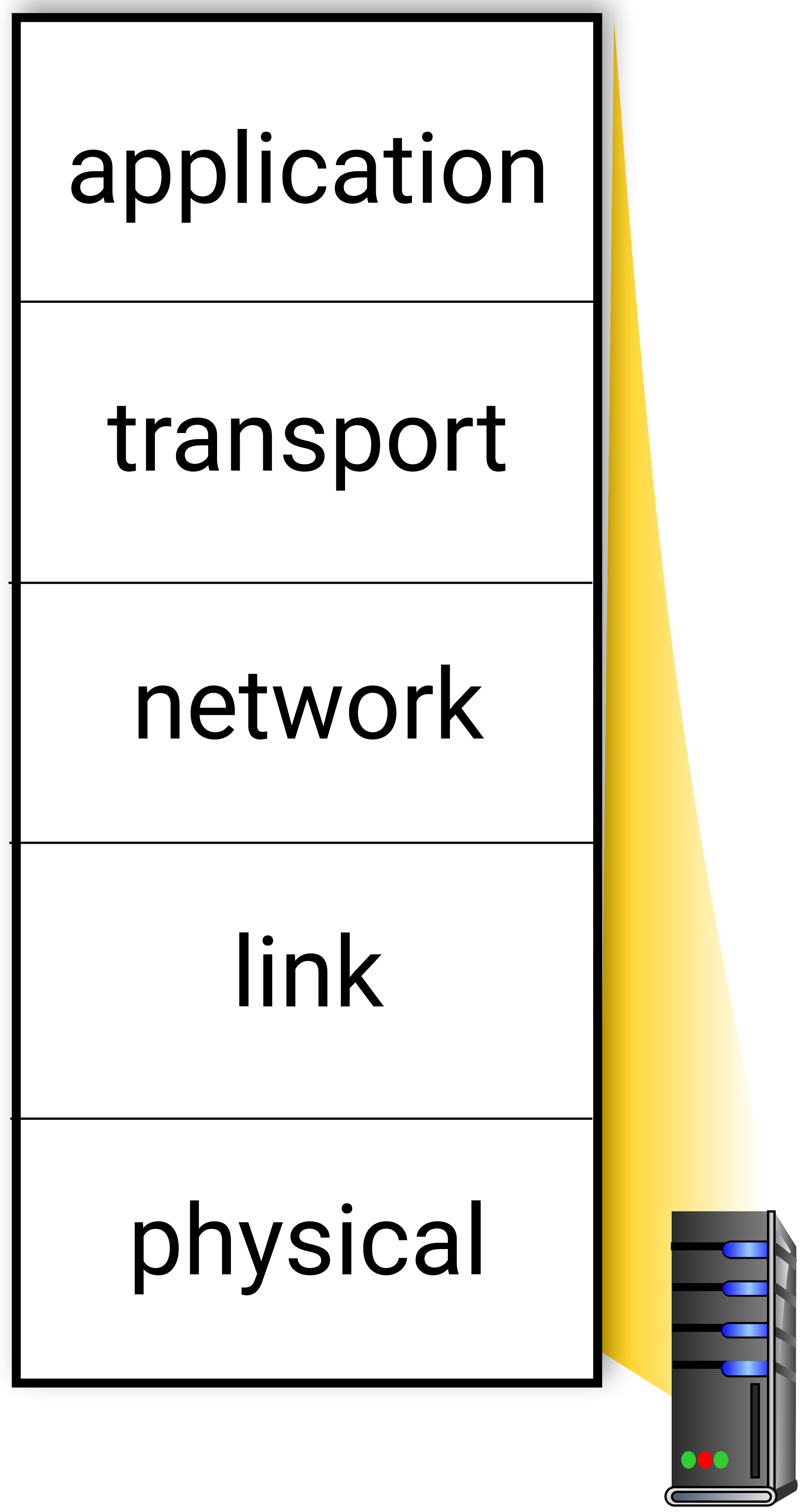
M

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

H_t M

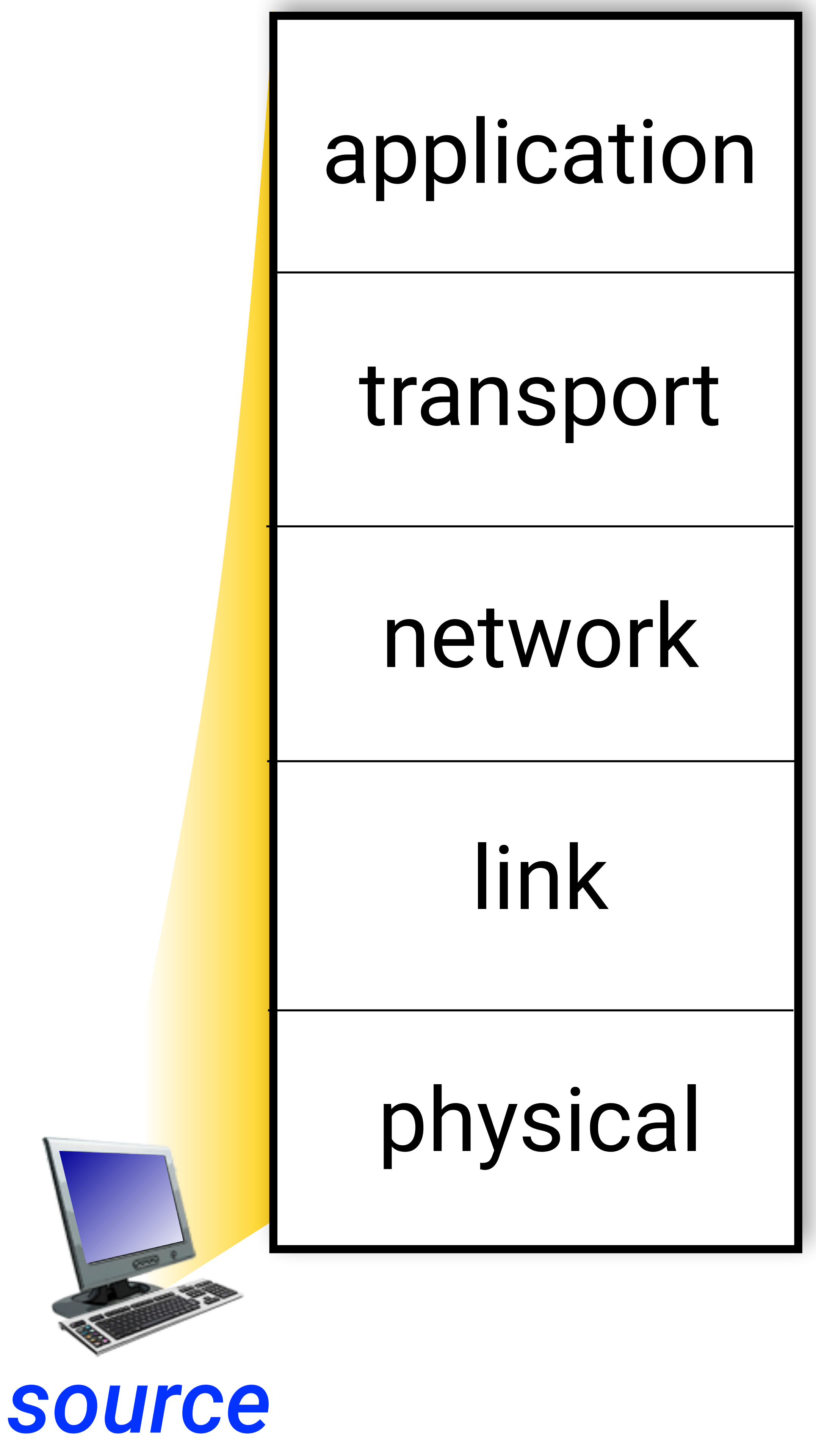
Transport-layer protocol transfers M from one *process* to another, using services of network layer

- 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



destination

Protocol layering and services



M

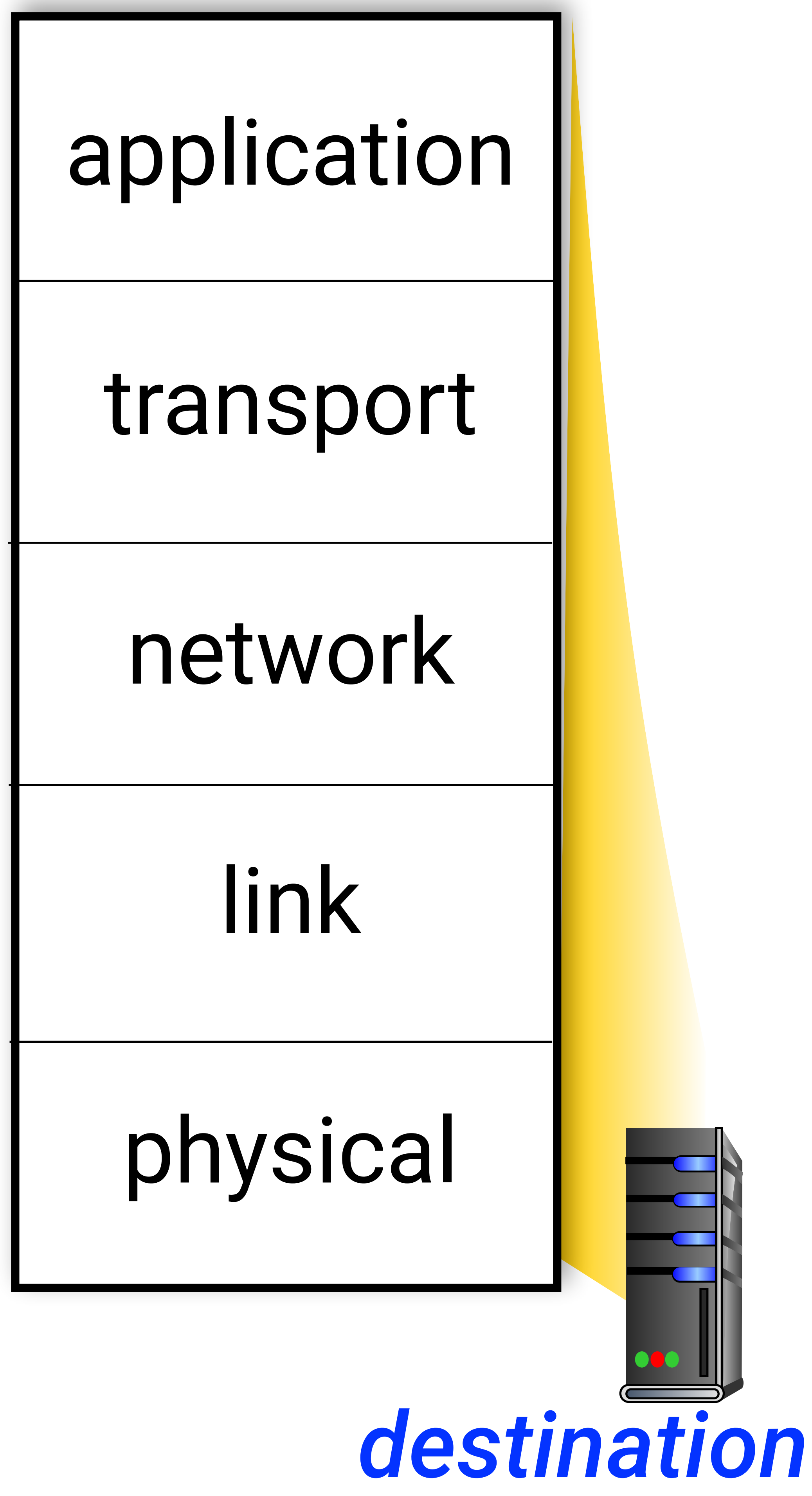
H_t M

Transport-layer protocol transfers M from one *process* to another, using services of network layer

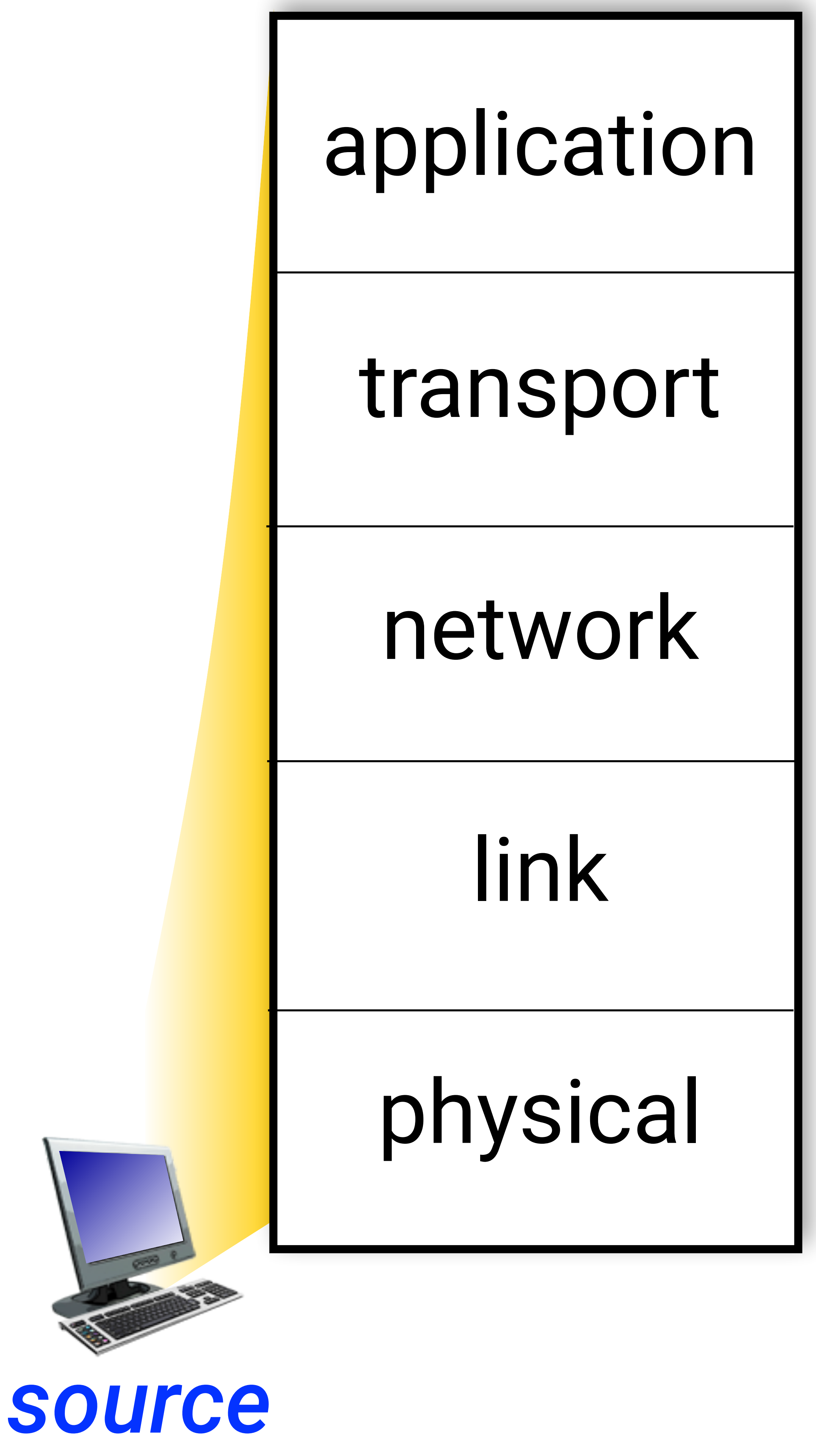
H_n H_t M

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



Protocol layering and services



M

H_t | M

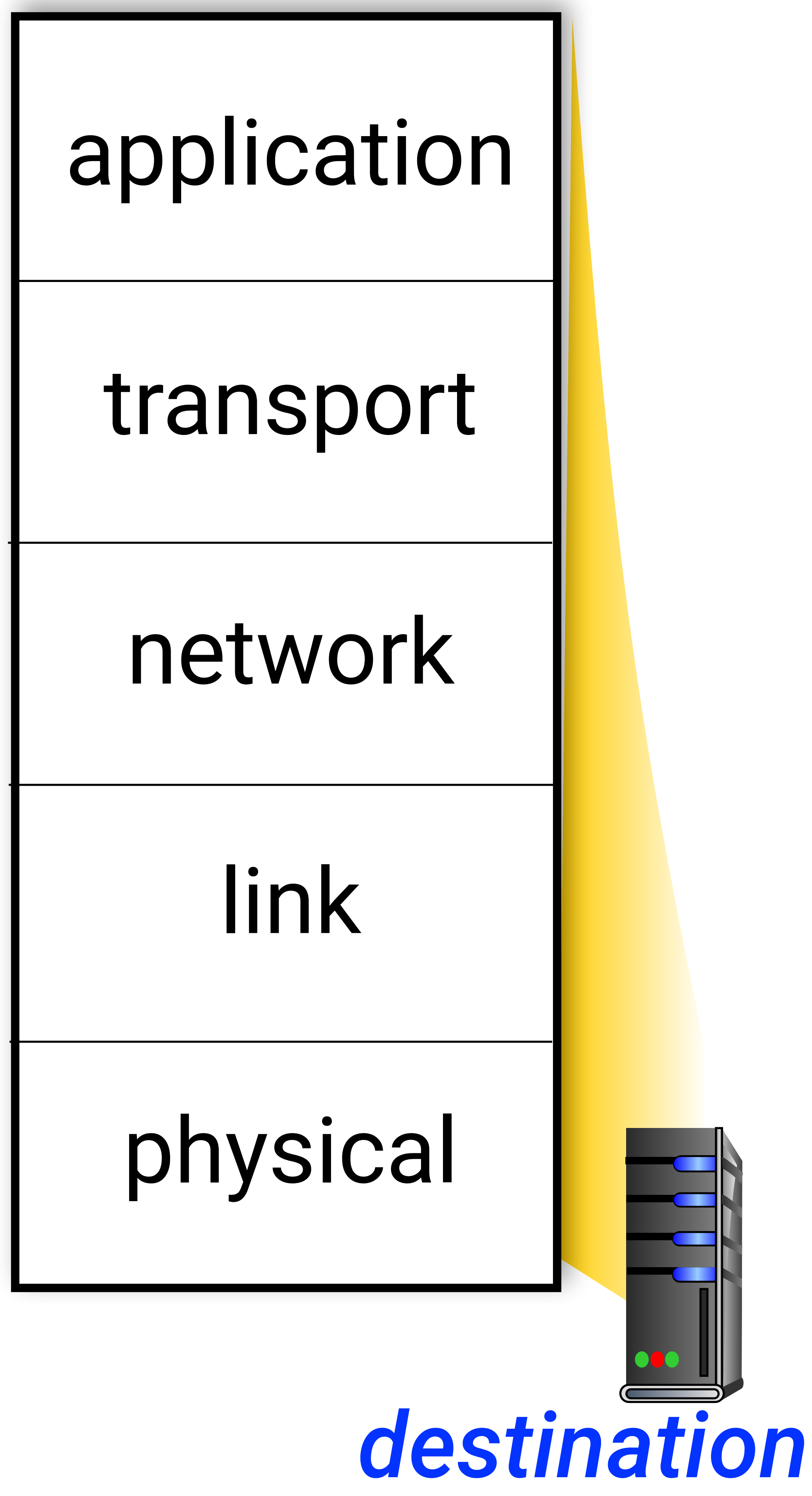
H_n | H_t | M

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

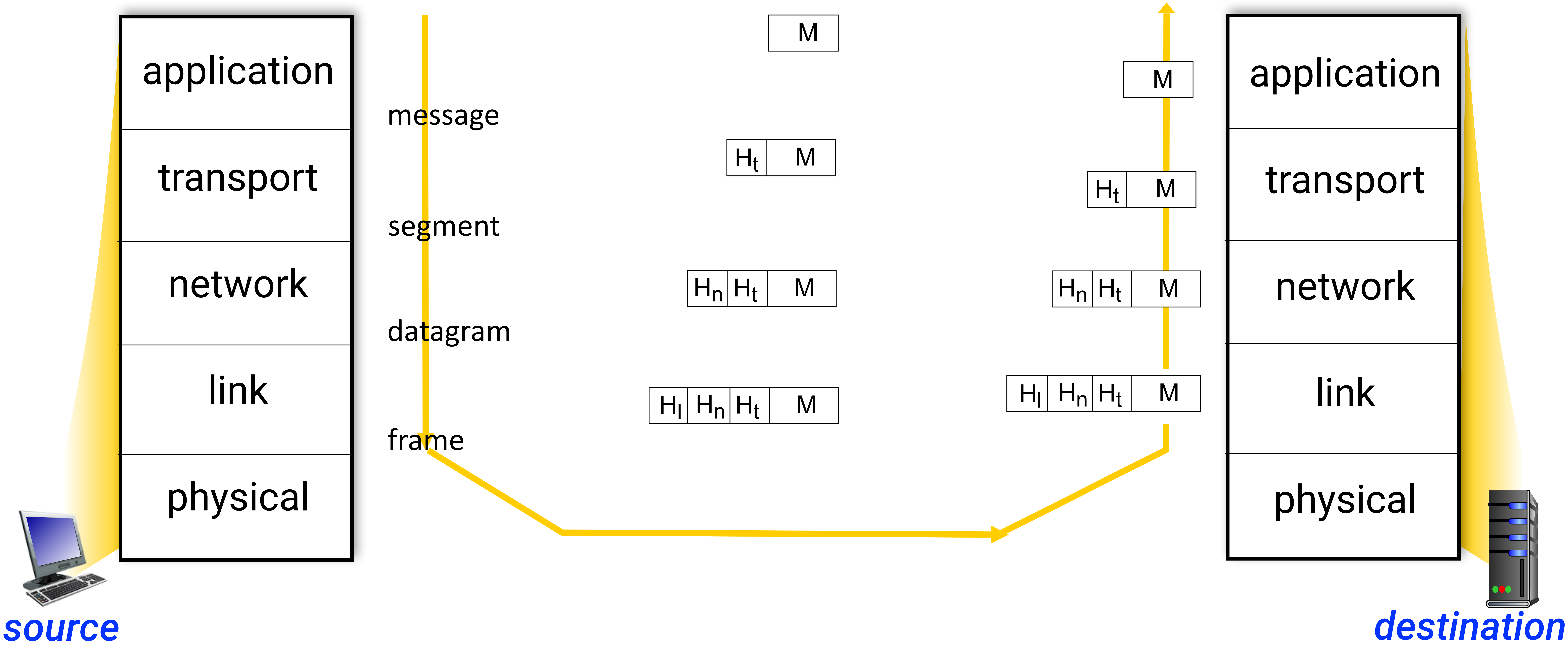
H_l | H_n | H_t | M

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

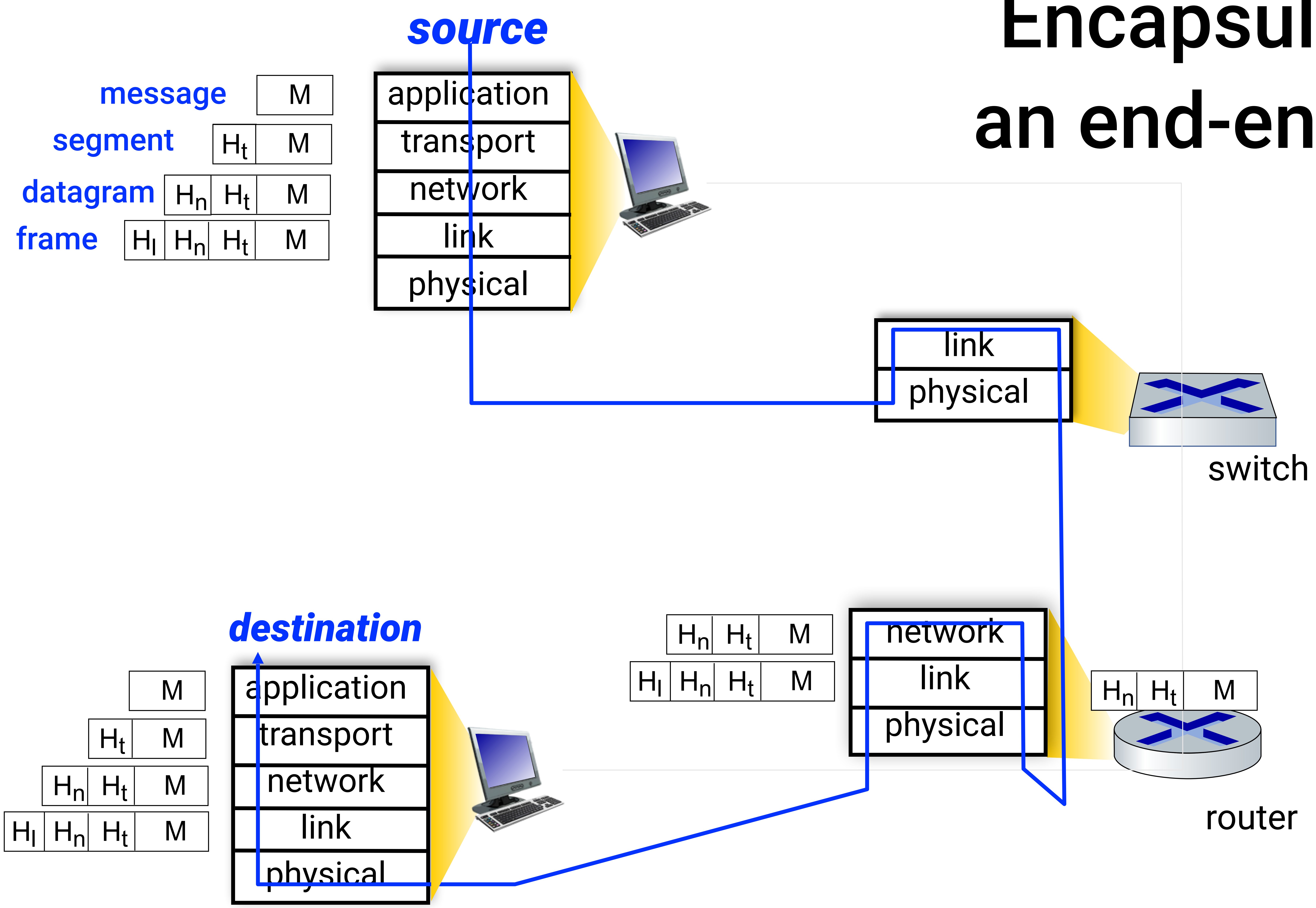
- link-layer protocol encapsulates network datagram [H_n| [H_t |M], with link-layer header H_l to create a link-layer frame



Protocol encapsulation



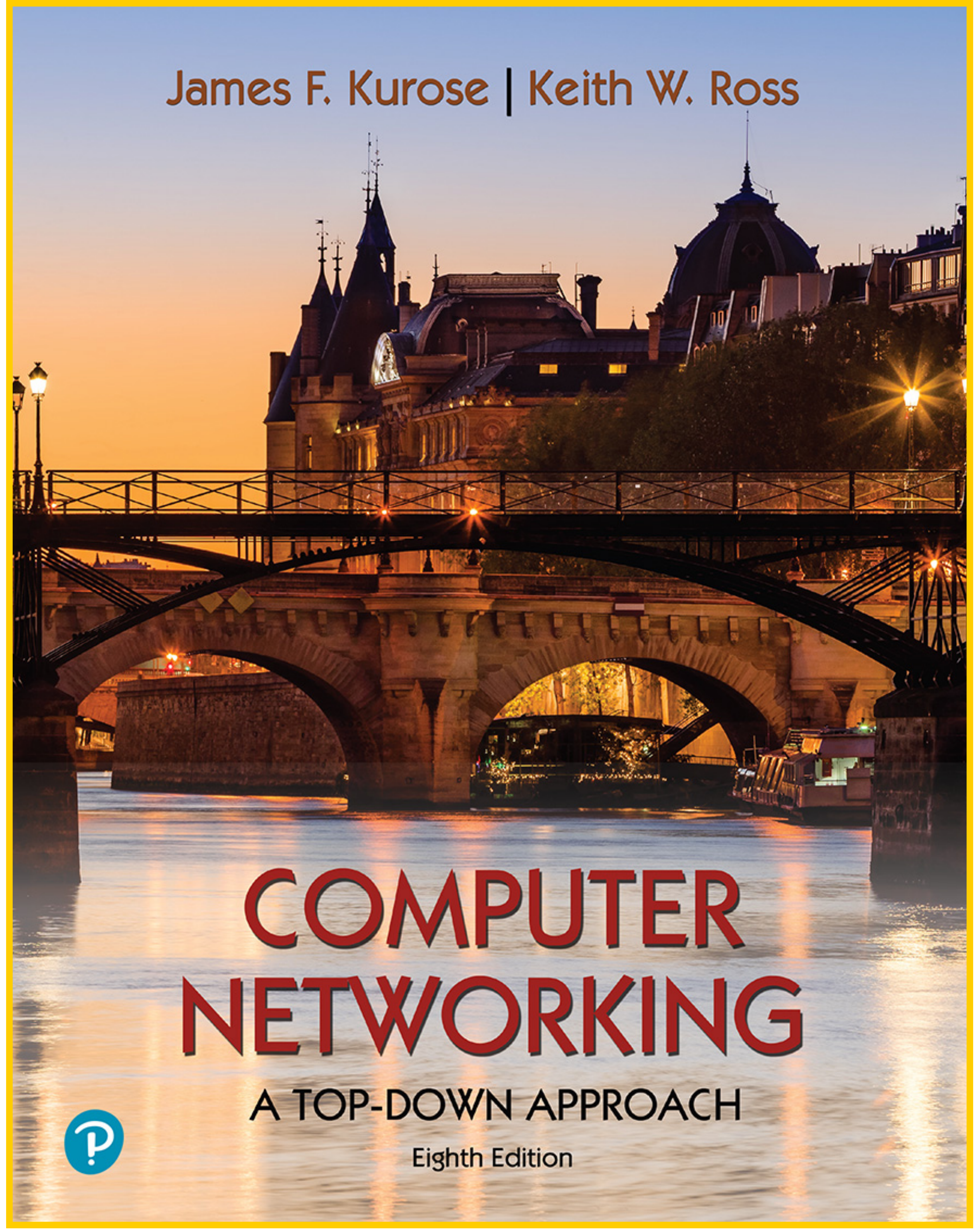
Encapsulation: an end-end view



Next Lecture

Continuing our in-depth exploration into the structure and functionality of the Internet

- *Network security*
- *Internet history and evolution*



Chapter 1.6 - 1.7

Spot Quiz (ICON)