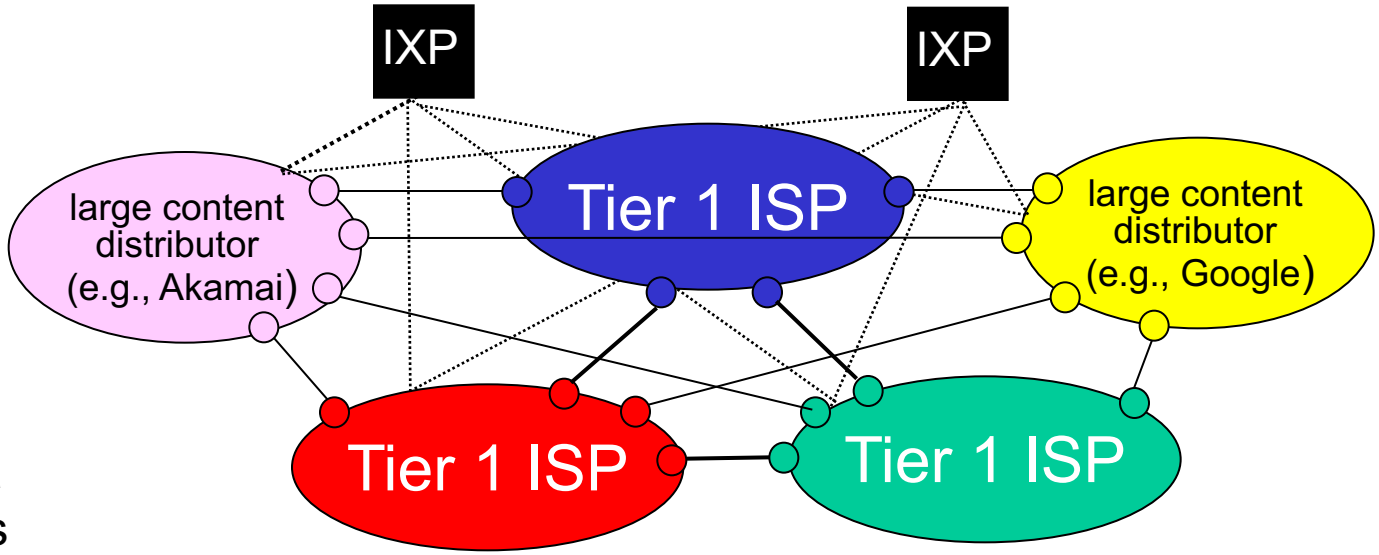
Introduction: roadmap

1.1 What *is* the Internet?
1.2 Network edge
1.3 Network core
1.4 Network access and physical media
1.5 Internet structure and ISPs
1.6 Delay & loss in packet-switched networks
1.7 Protocol layers, service models
1.8 History

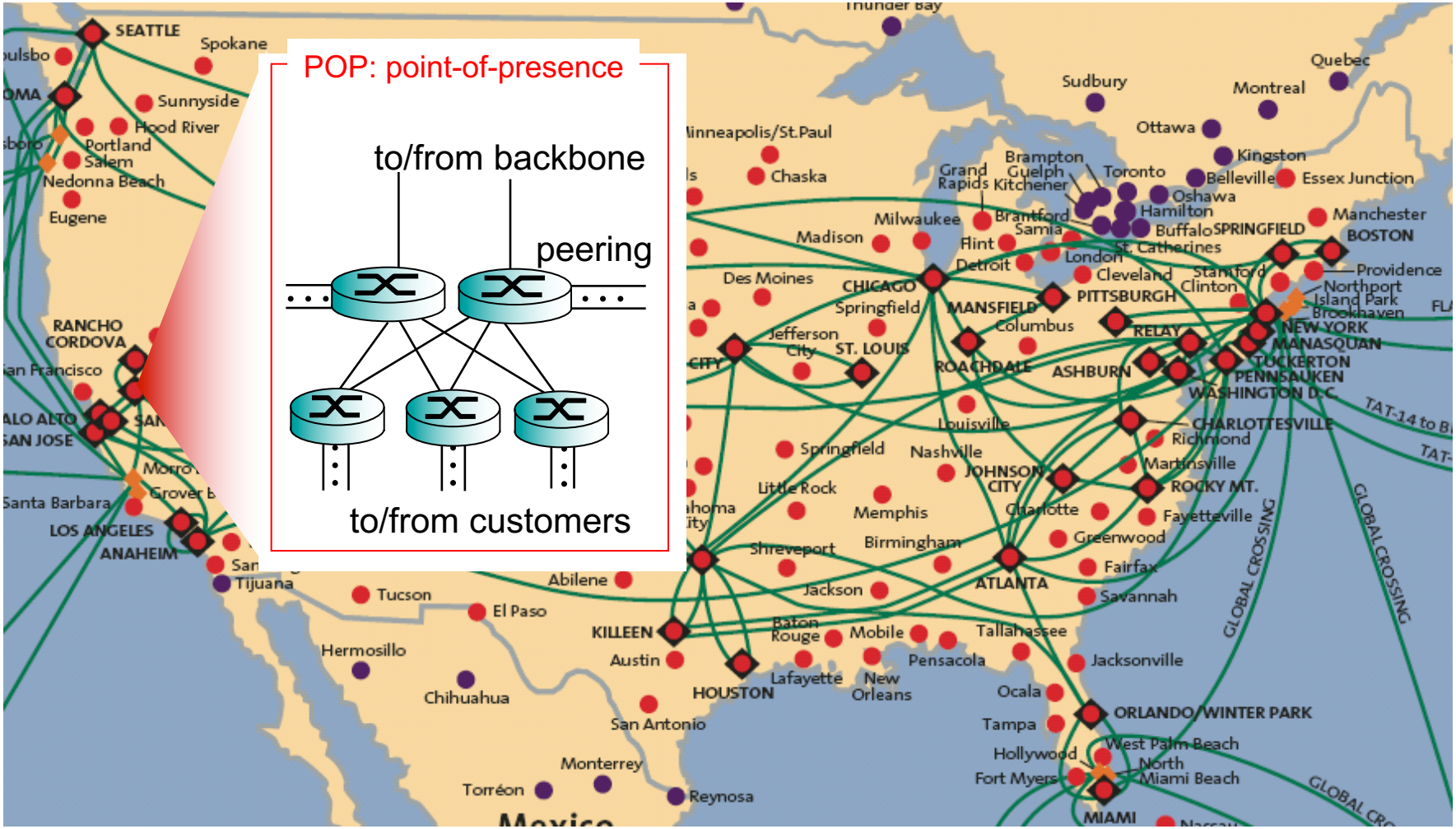
Internet structure: network of networks

- roughly hierarchical
- at center: small # of well-connected large networks
  - "tier-1" commercial ISPs (e.g., Verizon, Sprint, AT&T, Qwest, Level3), national & international coverage
  - large content distributors (Google, Akamai, Microsoft)
  - treat each other as equals (no charges)

tier-1 ISPs & content distributors, interconnect (peer) privately … or at Internet exchange points IXPs



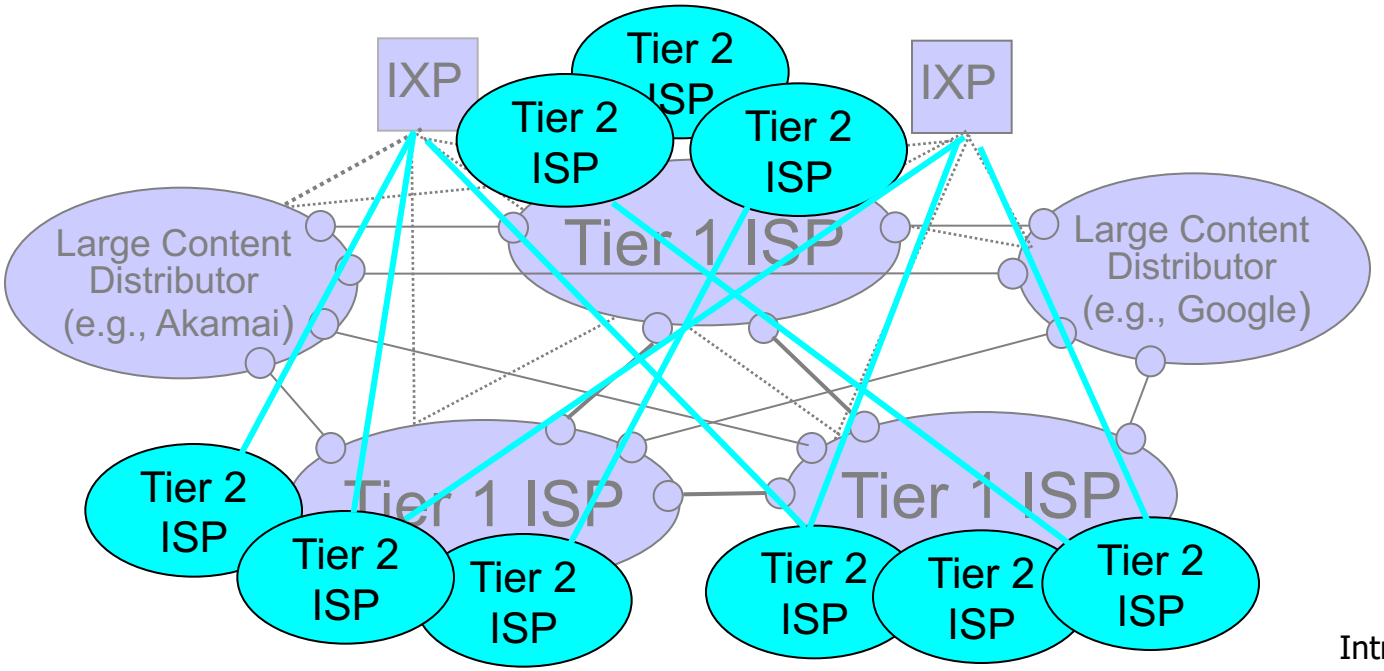
Tier-1 ISP: e.g., Sprint



Internet structure: network of networks

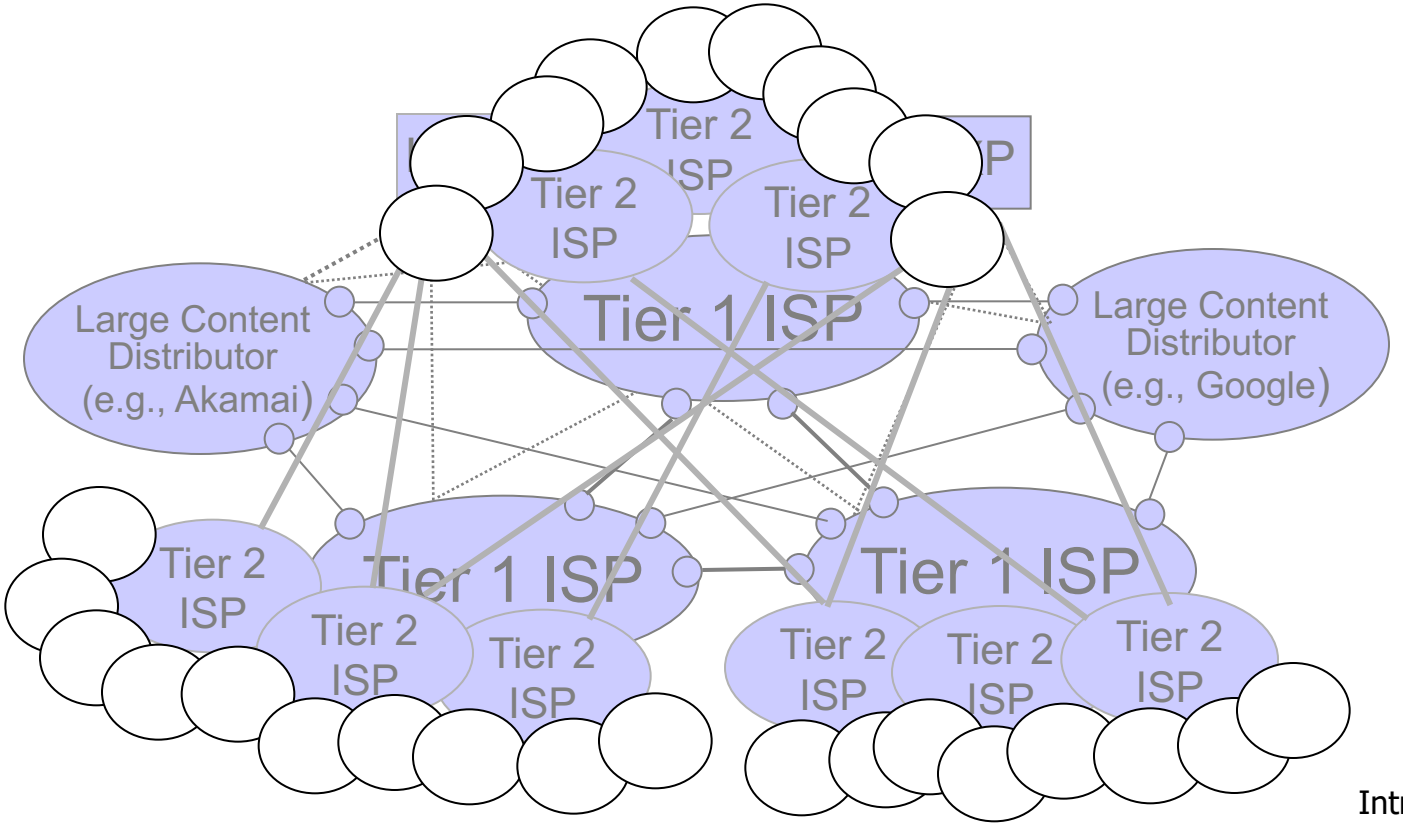
"tier-2" ISPs: smaller (often regional) ISPs

- connect to one or more tier-1 *(provider)* ISPs
  - each tier-1 has many tier-2 *customer nets*
  - tier 2 pays tier 1 provider
- tier-2 nets sometimes peer directly with each other (bypassing tier 1) , or at IXP



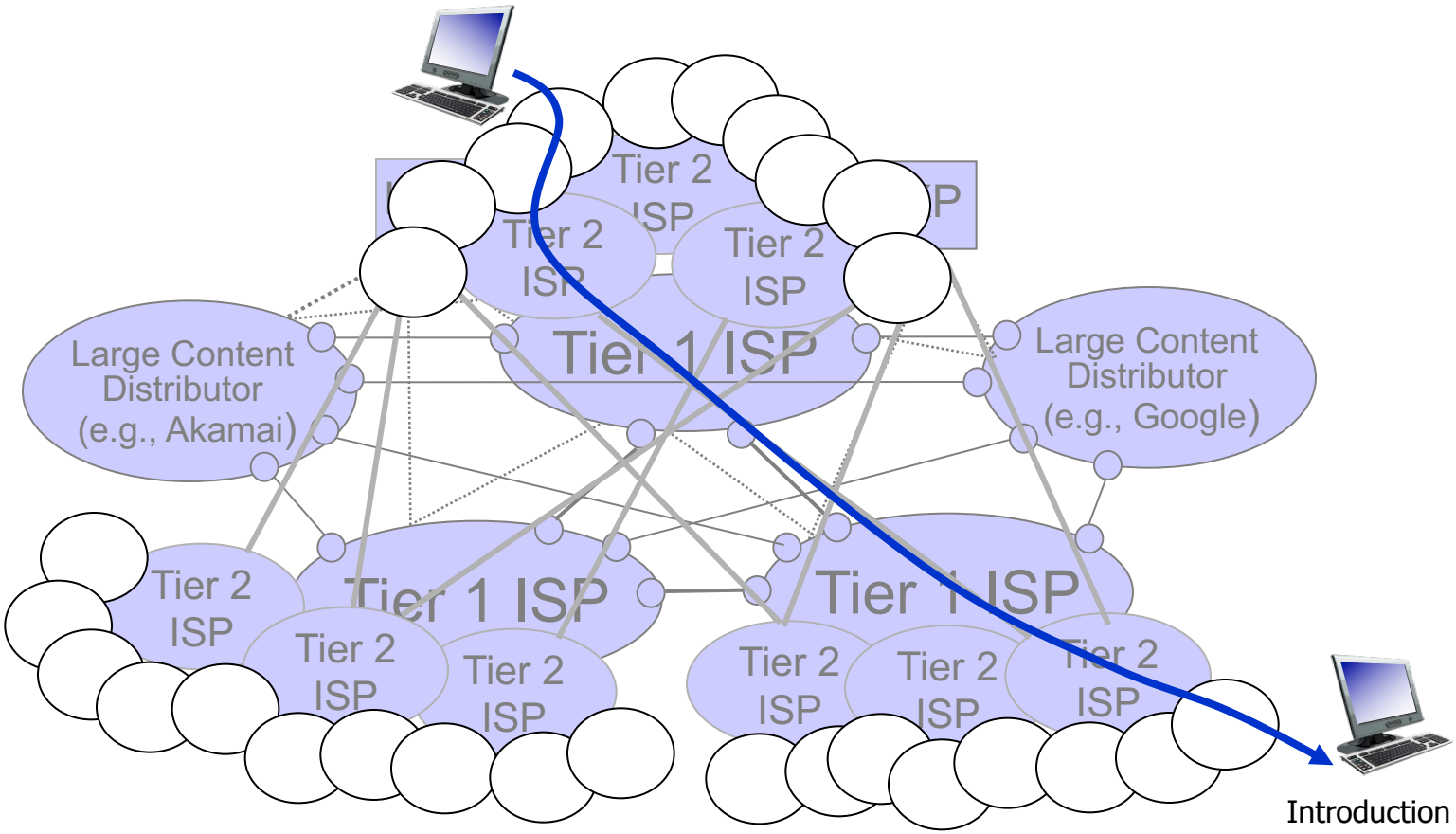
Internet structure: network of networks

- "tier-3" ISPs, local ISPs
- customer of tier 1 or tier 2 network
  - last hop ("access") network (closest to end systems)



Internet structure: network of networks

- a packet passes through *many* networks from source host to destination host



introduction: roadmap

1.1 what *is* the Internet?

1.2 network edge

- end systems, access networks, links

1.3 network core

- packet switching, circuit switching, network structure

1.5 Internet structure and ISPs

1.6 delay, loss, throughput in networks

1.7 protocol layers, service models

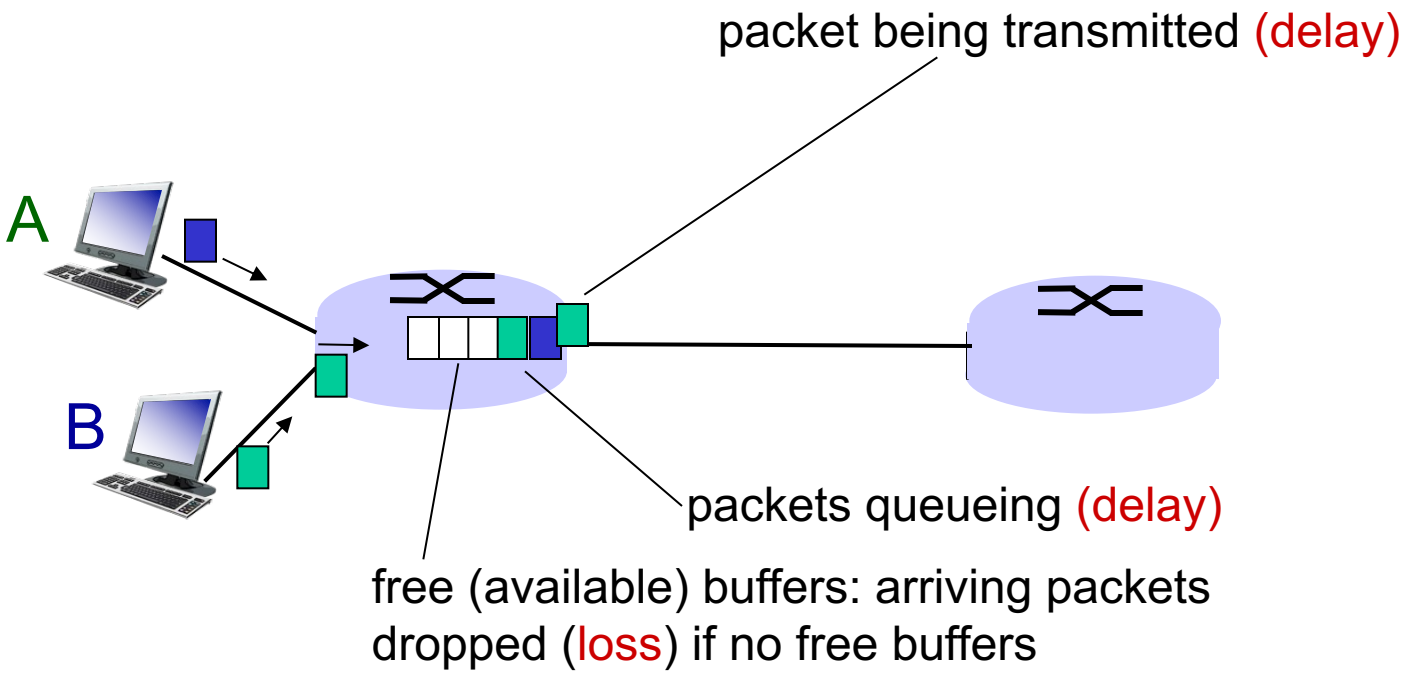
1.8 networks under attack: security

1.9 history

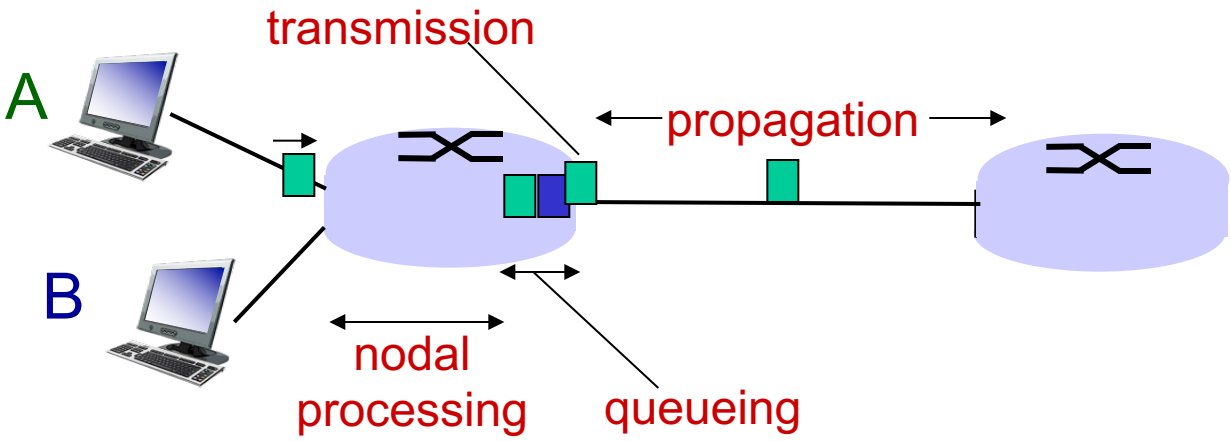
How do loss and delay occur?

packets *queue* in router buffers

- packet arrival rate to link (temporarily) exceeds output link capacity
- packets queue, wait for turn



Four sources of packet delay



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

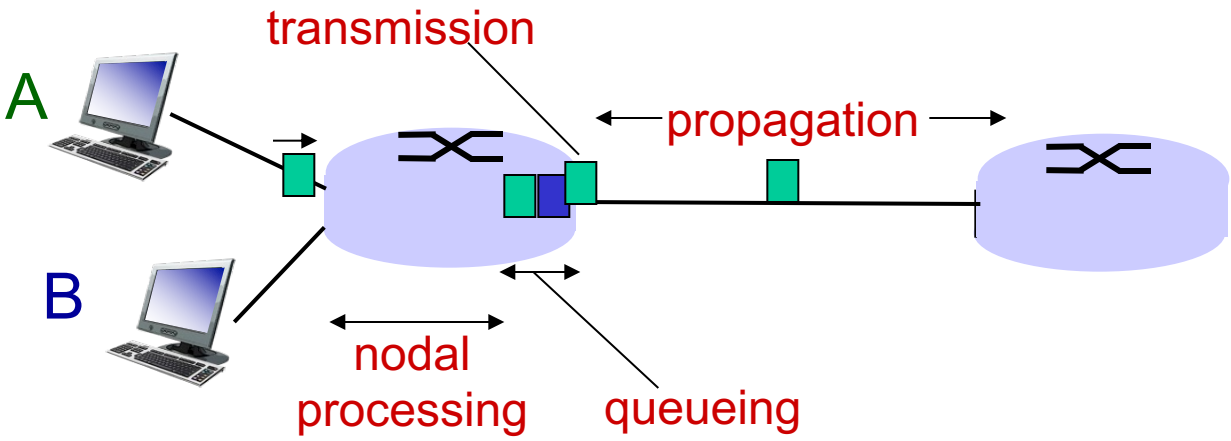
d_{proc}: nodal processing

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

d_{queue}: queueing delay

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

Four sources of packet delay



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

d_{trans}: transmission delay:

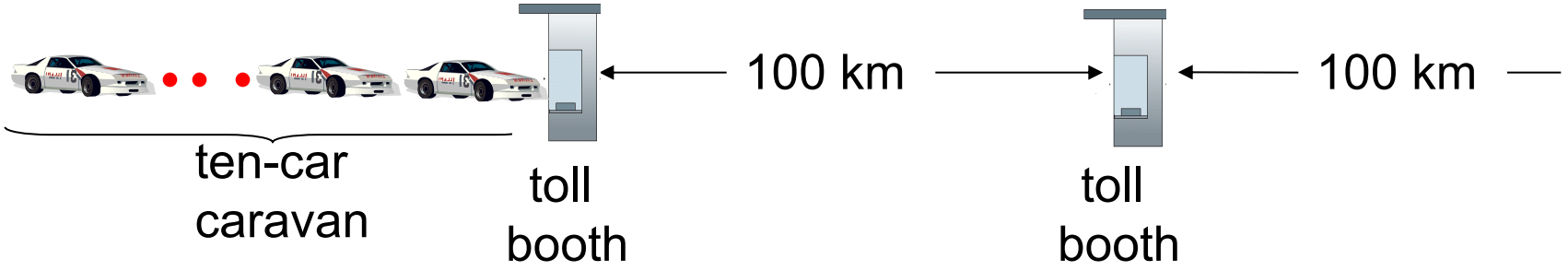
- L: packet length (bits)
- R: link bandwidth (bps)
- $d_{trans} = L/R$

d_{prop}: propagation delay:

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

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

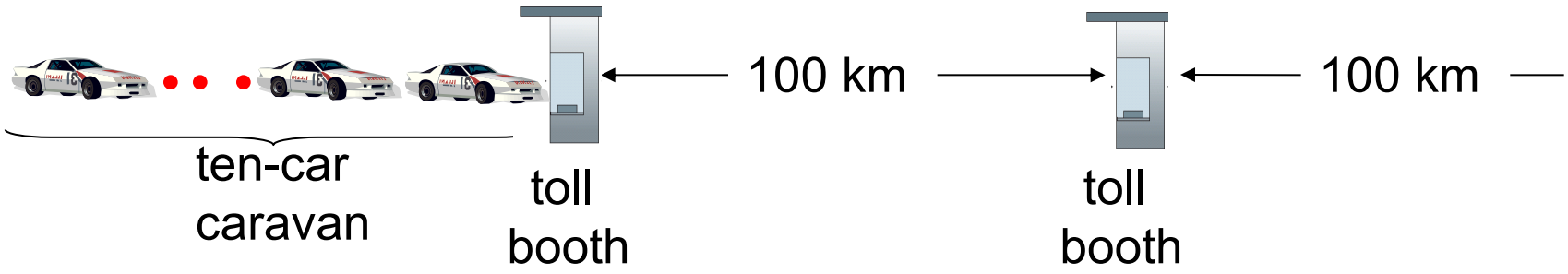
Caravan analogy



- cars "propagate" at 100 km/hr
- toll booth takes 12 sec to service car (bit transmission time)
- car~bit; caravan ~ packet
- *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*

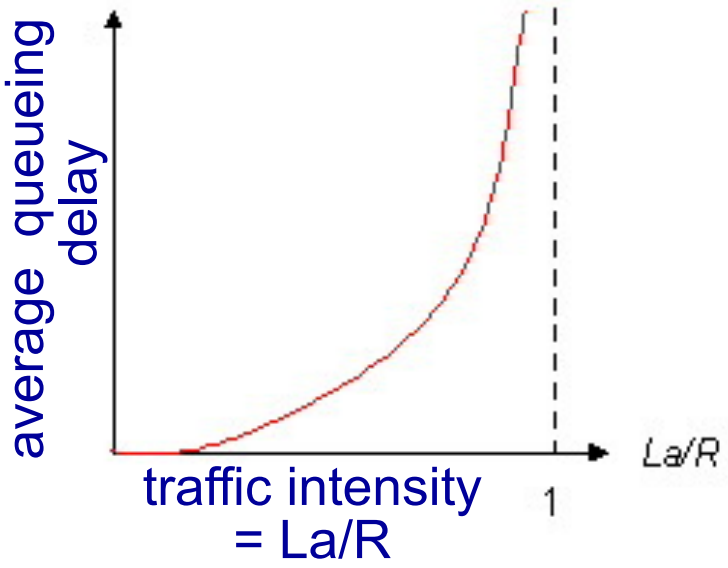
Caravan analogy (more)



- suppose cars now "propagate" at 1000 km/hr
- and suppose toll booth now takes 1 min to service a car
- *Q:* Will cars arrive to 2nd booth before all cars serviced at first booth?
  - *A: Yes!* after 7 min, 1st car arrives at second booth; three cars still at 1st booth.
  - first bit of packet can arrive at 2nd router before packet is fully transmitted at 1st router!

Queueing delay (revisited)

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



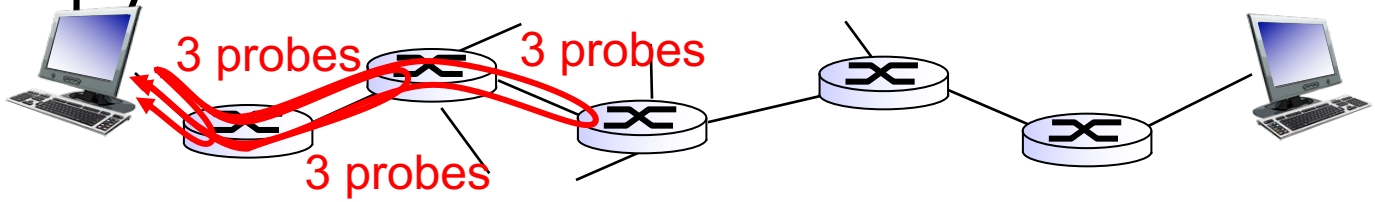
- *La/R* ~ 0: avg. queueing delay small
- *La/R* -> 1: avg. queueing delay large
- *La/R* > 1: more "work" arriving than can be serviced, average delay infinite!

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
  - router *i* will return packets to sender
  - sender times interval between transmission and reply.



"Real" Internet delays, routes

traceroute: .shanghai Unicom to www.usc.edu

3 delay measurements from Fudan to ww.usc.edu

```
1  10.51.207.185 (10.51.207.185)  1.870 ms  1.344 ms  1.134 ms
2  10.51.1.2 (10.51.1.2)  1.152 ms  1.095 ms  1.057 ms
3  210.13.73.185 (210.13.73.185)  14.952 ms  2.684 ms  4.454 ms
4  * * *
5  112.64.252.190 (112.64.252.190)  3.431 ms  6.079 ms  2.142 ms
6  218.105.2.149 (218.105.2.149)  4.693 ms  5.245 ms  3.858 ms
7  218.105.10.22 (218.105.10.22)  2.739 ms  2.710 ms  2.678 ms
8  pos3-0.ig1.lax7.alter.net (208.222.0.97)  157.196 ms  166.491 ms  171.185 ms
9  * * *
10 ae-7.r01.lsanca20.us.bb.gin.ntt.net (129.250.8.85)  168.629 ms  190.266 ms
     177.837 ms
11  ae-17.r01.lsanca07.us.bb.gin.ntt.net (129.250.4.207)  179.043 ms  179.474 ms
     180.199 ms
```

trans-oceanic link

Verizon

"Real" Internet delays, routes

traceroute: shanghai unicom to www.usc.edu

11 ae-17.r01.lsanca07.us.bb.gin.ntt.net (129.250.4.207) 179.043 ms 179.474 ms 180.199 ms

12 ntt-los-nettos-usc.ln.net (165.254.21.242) 175.243 ms 169.510 ms 170.251 ms

13 130.152.182.83 (130.152.182.83) 159.538 ms 169.551 ms 192.469 ms

14 fw6-rtr-border-cal.usc.edu (128.125.251.227) 194.779 ms 162.290 ms 158.598 ms

15 rtr6-fw6.usc.edu (128.125.255.146) 160.823 ms 161.753 ms 159.159 ms

16 v249-gw-33.usc.edu (68.181.194.69) 160.257 ms 158.741 ms 164.976 ms

17 fwc01.usc.edu (128.125.255.190) 161.820 ms 161.706 ms 164.768 ms

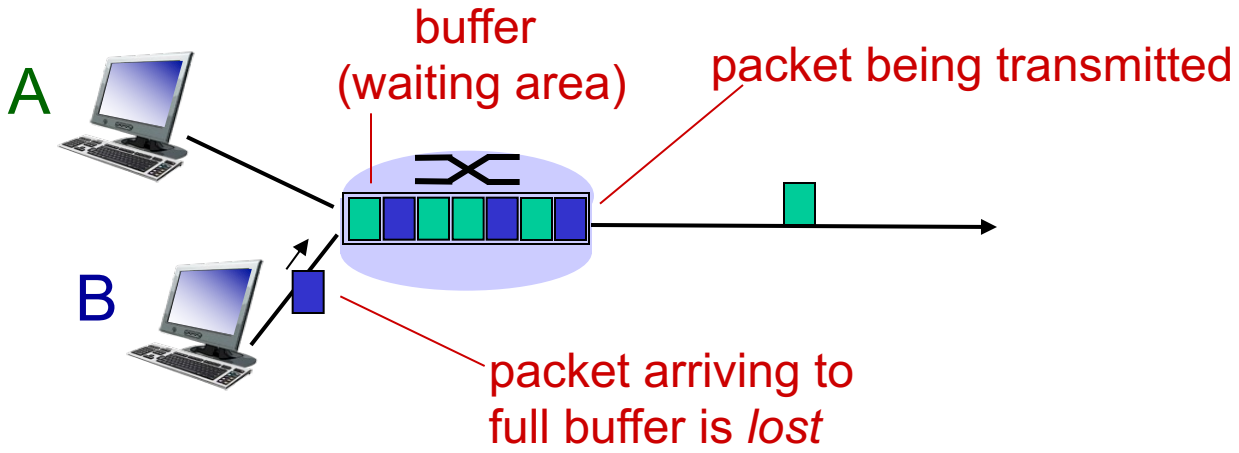
18 * * *

19 cwis.usc.edu (128.125.253.146) 166.461 ms 158.701 ms 163.325 ms

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

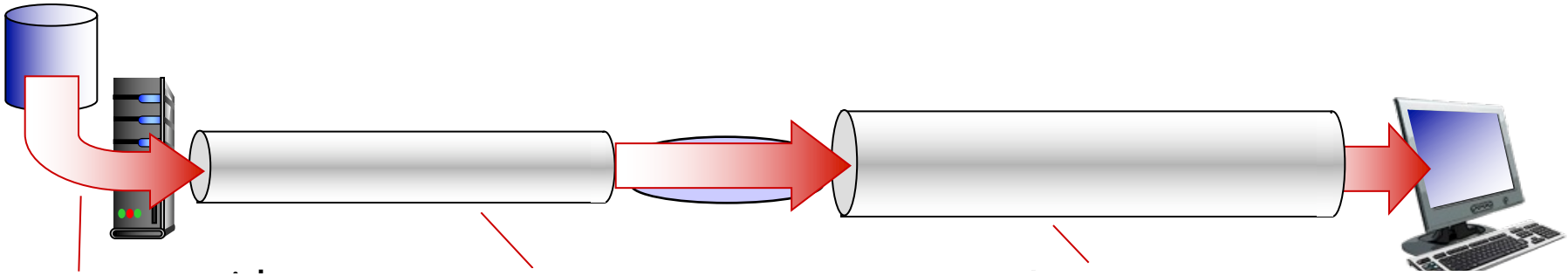
Packet loss

- 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



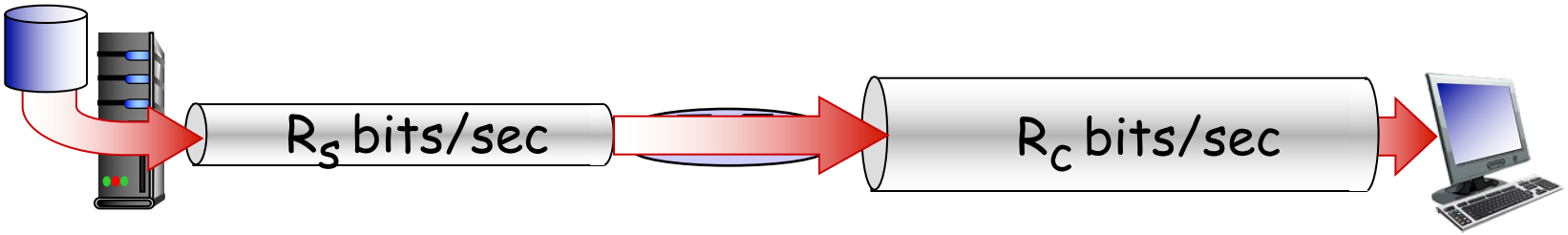
Throughput

- *throughput:* rate (bits/time unit) at which bits transferred between sender/receiver
  - *instantaneous:* rate at given point in time
  - *average:* rate over longer period of time

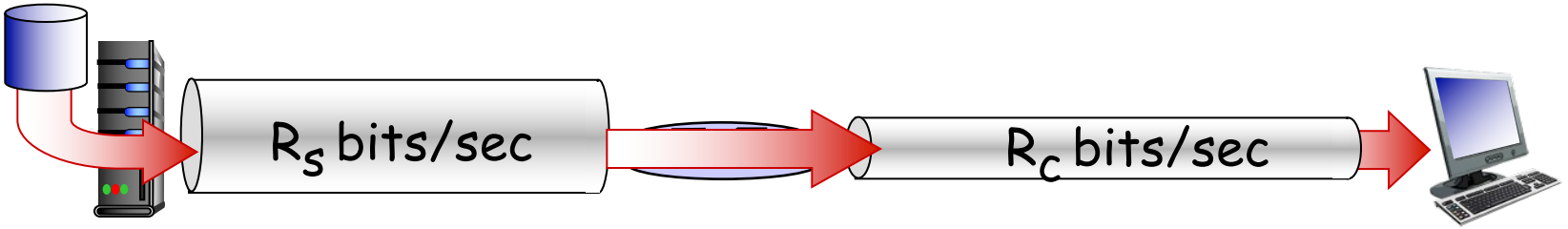


Throughput (more)

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



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

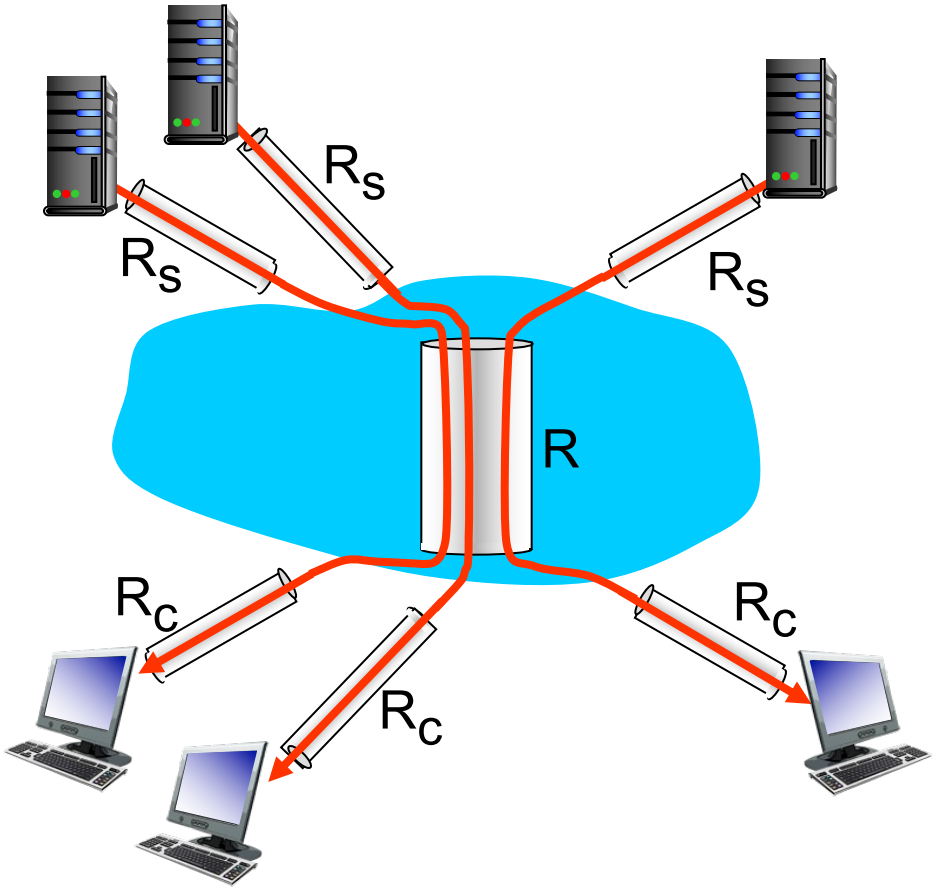


bottleneck link

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

Throughput: Internet 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

introduction: roadmap

1.1 what *is* the Internet?

1.2 network edge

- end systems, access networks, links

1.3 network core

- packet switching, circuit switching, network structure

1.5 Internet structure and ISPs

1.6 delay, loss, throughput in networks

1.7 protocol layers, service models

1.8 networks under attack: security

1.9 history

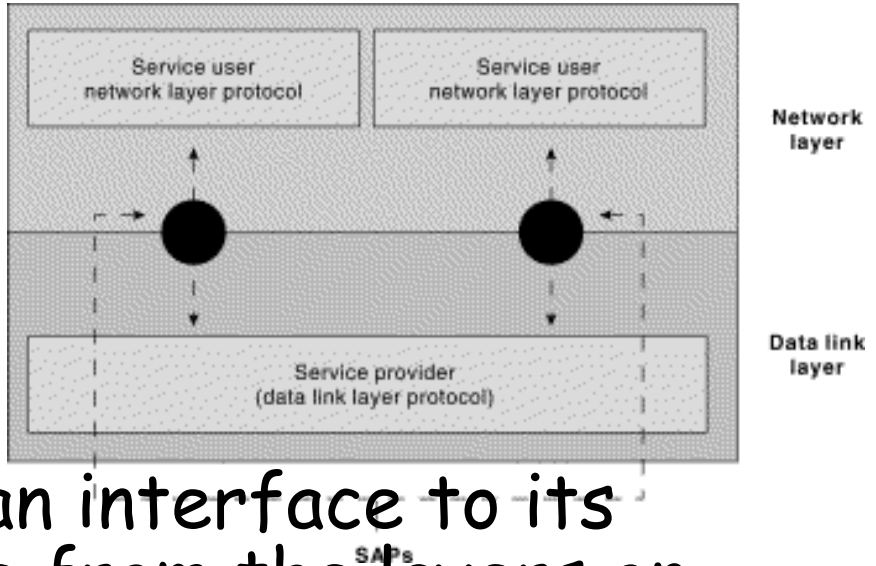
More on Protocols

- Protocols are the key to interoperability.
  - Networks are very heterogenous:

Computer: x86	**Hardware**
Ethernet: 3com	**Hardware/link**
Routers: cisco, etc.	**Network**
App: Email	**Application**

  - The hardware/software of communicating parties are often not built by the same vendor
  - Yet they can communicate because they use the same protocol
- Protocols exist at many levels.
  - Application level protocols, e.g. access to mail, distribution of files, web access, ..
  - Protocols at the hardware level allow two boxes to communicate over a link, e.g. the Ethernet protocol

Interfaces



- Each protocol offers an interface to its users, and expects one from the layers on which it builds
  - Syntax and semantics strike again
    - Data formats
    - Interface characteristics, e.g. IP service model
- Protocols build upon each other
  - Add value
    - E.g., a reliable protocol running on top of IP
  - Reuse
    - E.g., OS provides TCP, so apps don't have to rewrite

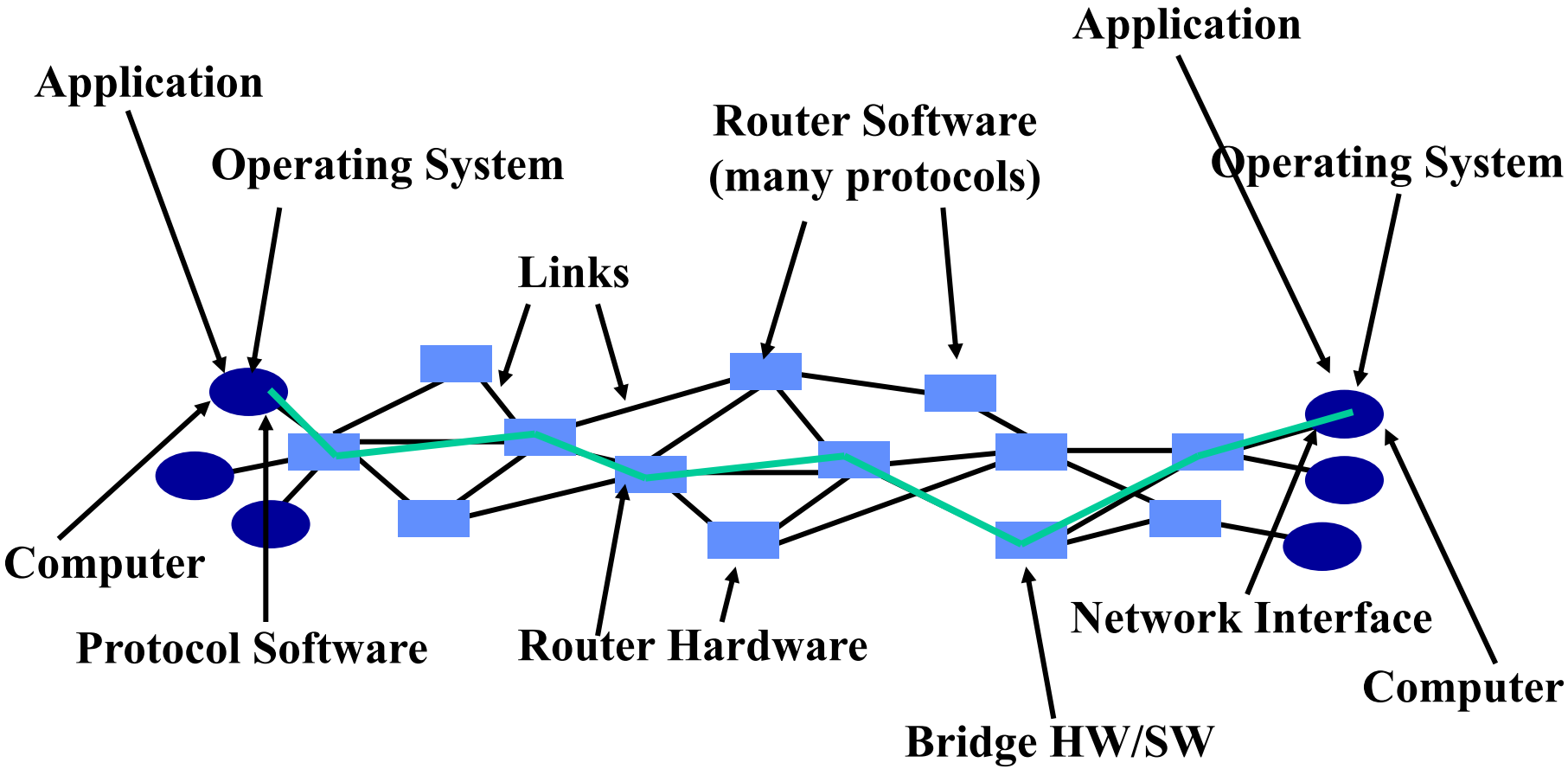
Implementing Protocol

- Requirements for packets:
  - Packet size limit?
  - Header information: Addresses, etc.
  - From where/who, to where ? Naming and address
    - POST, Telephone

- Almost in all lectures

Q: The relationship between Data link layer name and IP name?

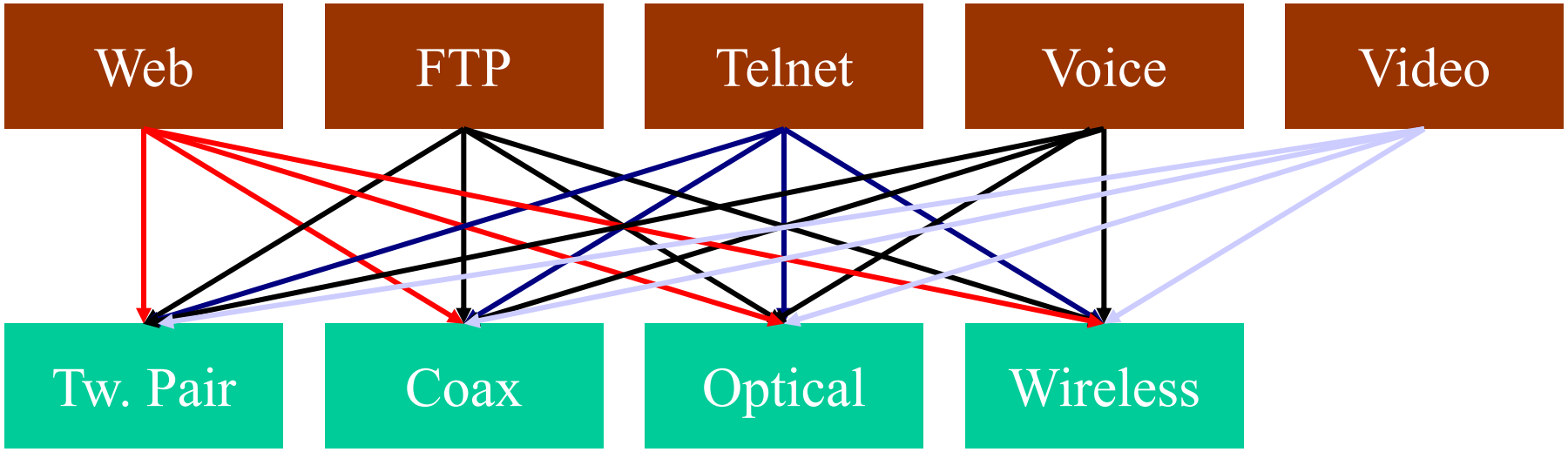
Too Many Network Components



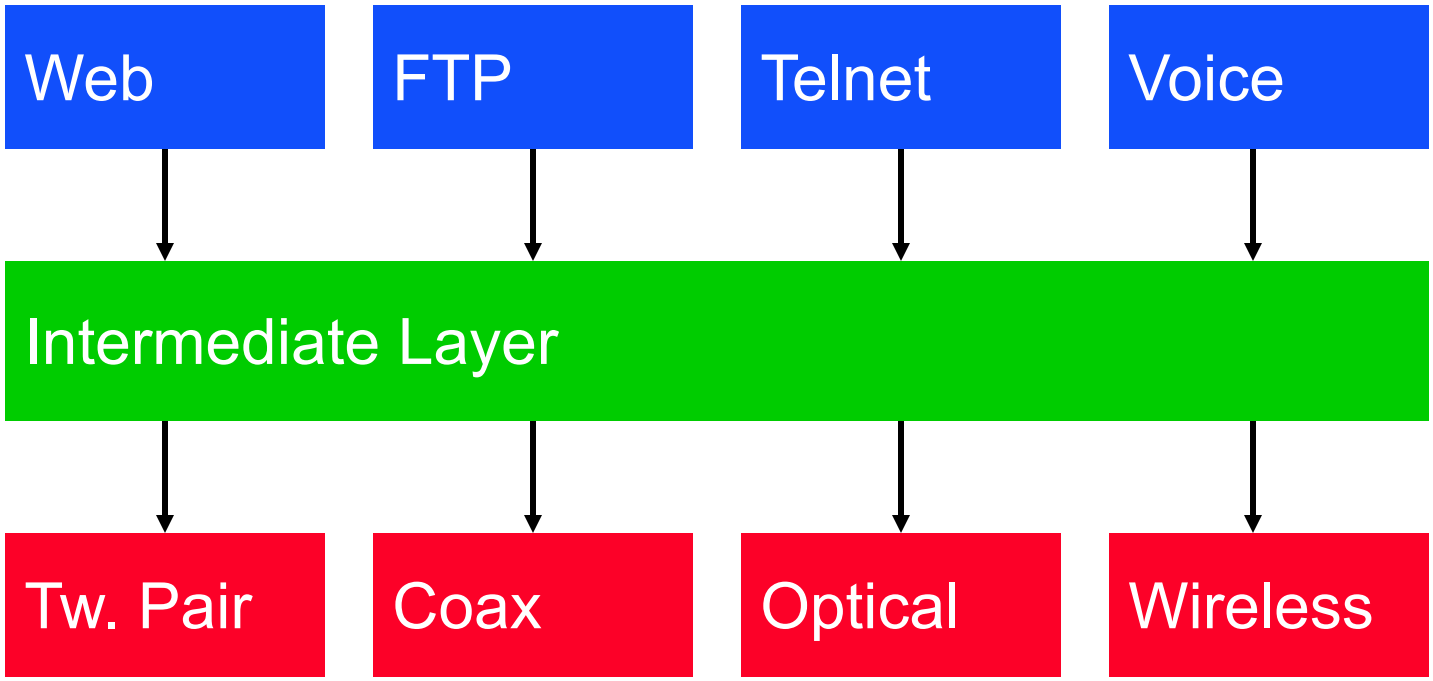
Too many components 2

- Links: copper, fiber, air
- Running ethernet, token ring, SONET, FDDI
- Routers speaking BGP, OSPF, RIP, …
- Hosts running FreeBSD, Linux, Windows, MacOS, …
- People using Mozilla, Explorer, Chrome, …
- and it changes all the time
- Protocol layers hide this stuff with simple abstractions.

Example: no abstraction



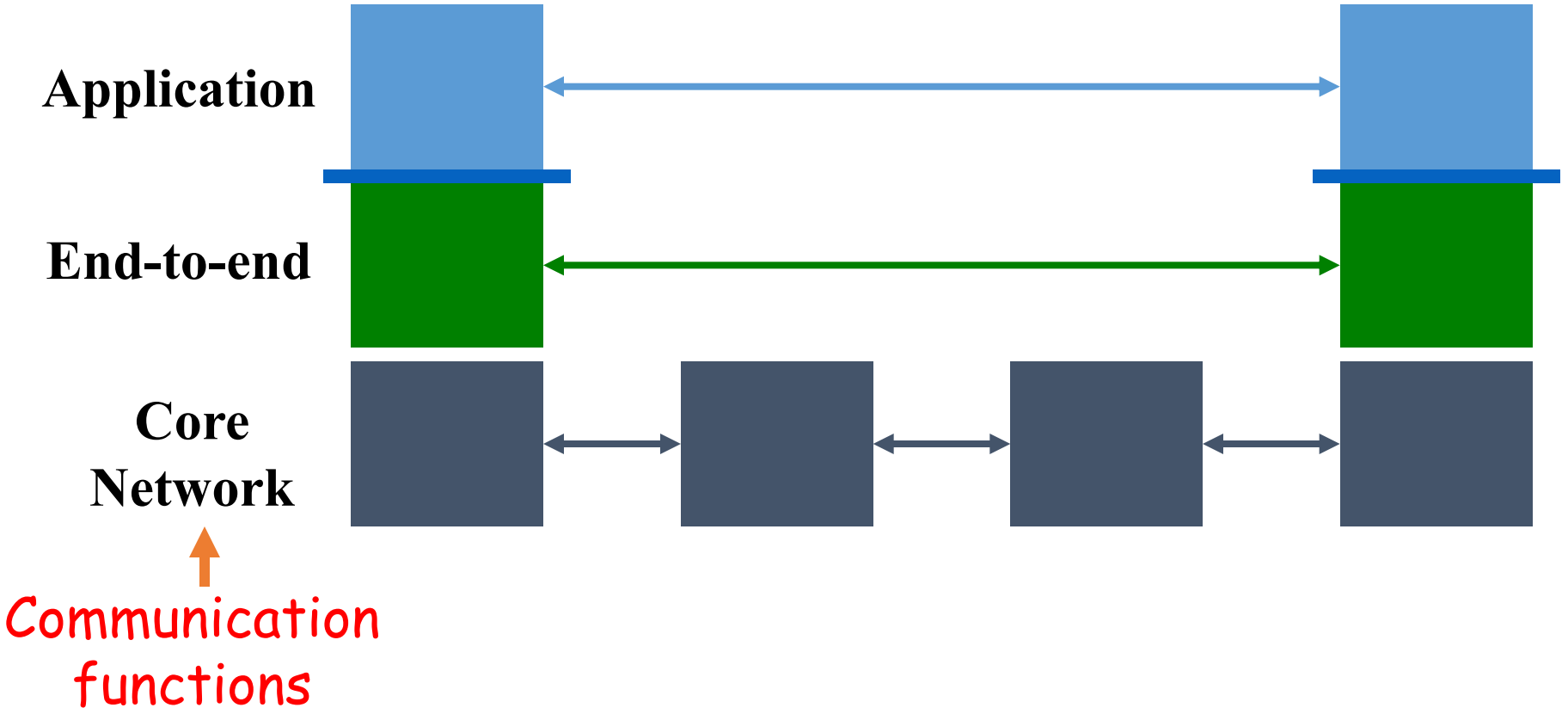
Three layers model : Hardware abstraction



Resources for Process | Communication Functions

How to do abstractions?

End-to-end abstraction



Looking at protocols

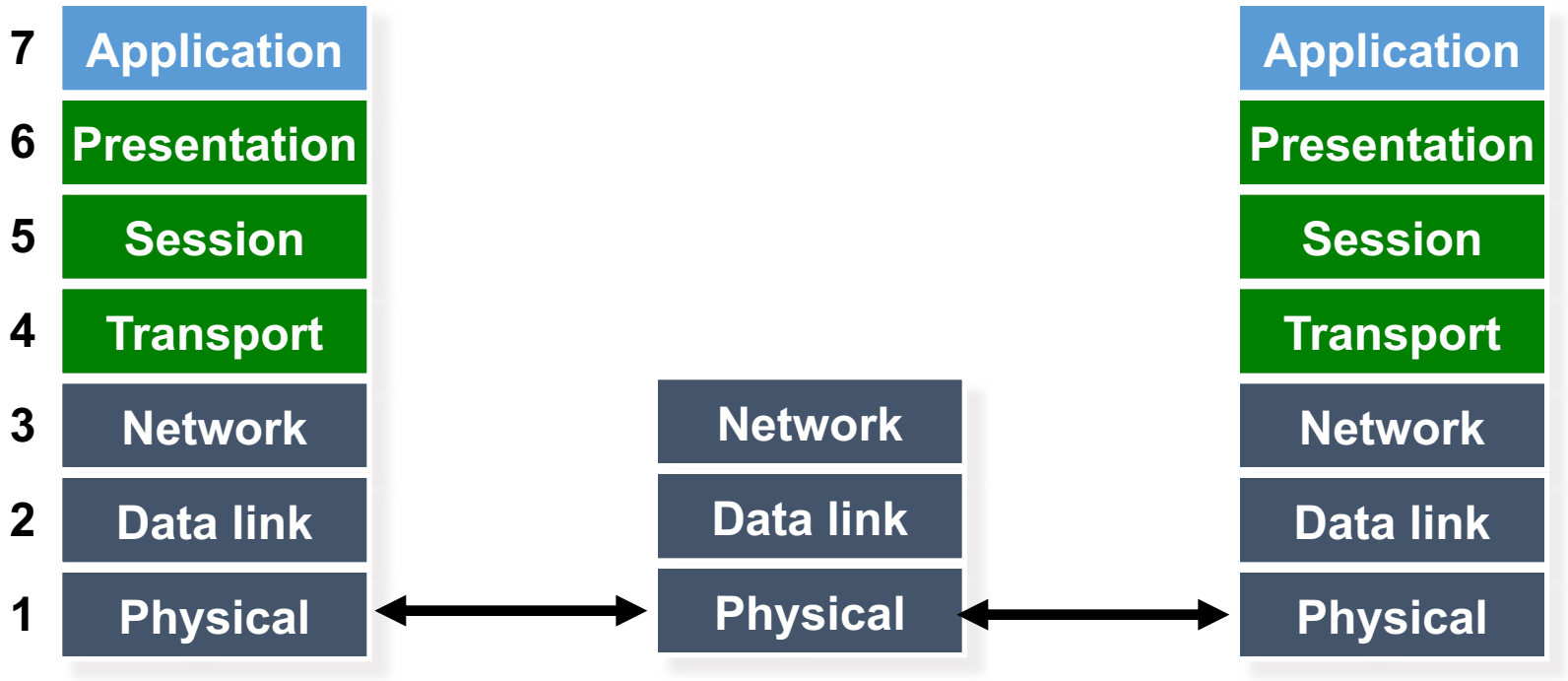
- Hop by hop / link protocols (communication related)
  - Ethernet
- End-to-end protocols (app related)
  - TCP, apps, etc.

Different way to abstraction

- Management / "control plane" protocols
  - Routing, etc.
    - Can be either link or e2e themselves
    - Definition somewhat vague.

A Layered Network Model

The Open Systems Interconnection (OSI) Model.



OSI Motivation

- Standard way of breaking up a system in a set of components, but the components are organized as a set of layers.
  - Only horizontal and vertical communication
  - Components/layers can be implemented and modified in isolation
- Each layer offers a service to the higher layer, using the services of the lower layer.
- "Peer" layers on different systems communicate via a protocol.
  - higher level protocols (e.g. TCP/IP, Appletalk) can run on multiple lower layers
  - multiple higher level protocols can share a single physical network
- "It's only a model!" - TCP/IP has been crazy successful, and it's not based on a rigid OSI model. But the OSI model has been very successful at shaping thought.

Advantages of layering

dealing with complex systems:

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

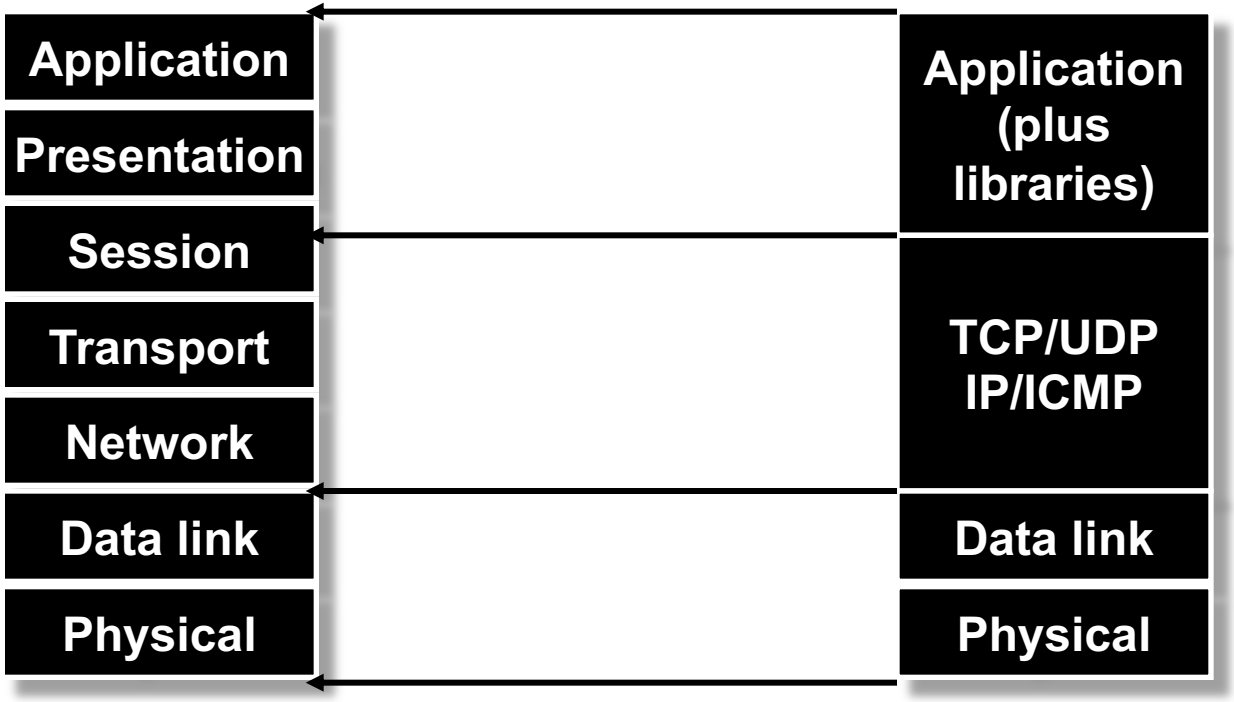
Limitations of the Layered Model

- Some layers are not always cleanly separated.
  - Inter-layer dependencies in implementations for performance reasons
  - Some dependencies in the standards (header checksums)
- Higher layers not always well defined.
  - Session, presentation, application layers
- Lower layers have "sublayers".
  - Usually very well defined (e.g., LAN, SONET protocol)
- Interfaces are not really standardized.
  - It would be hard to mix and match layers from independent implementations, e.g., windows network apps on unix (without compatability library)
  - Many cross-layer assumptions, e.g. buffer management

The TCP/IP Model

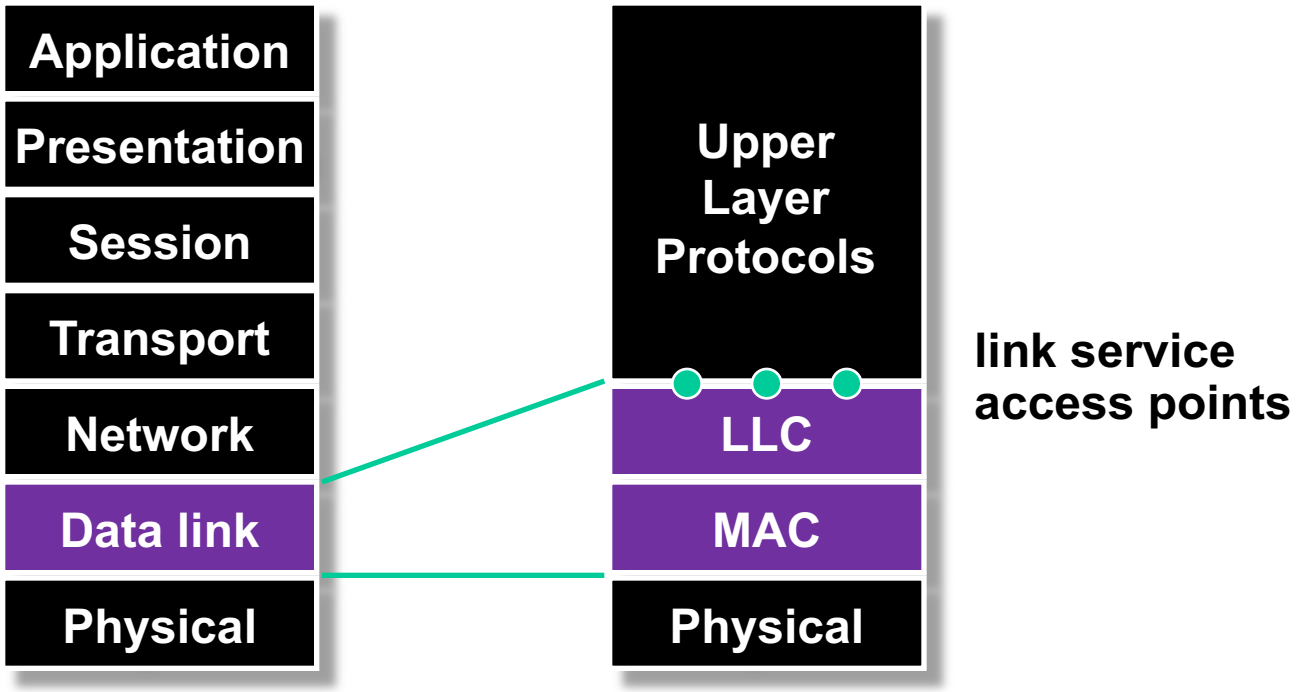
De facto industrial standard:

TCP/IP + IEEE802 link layer

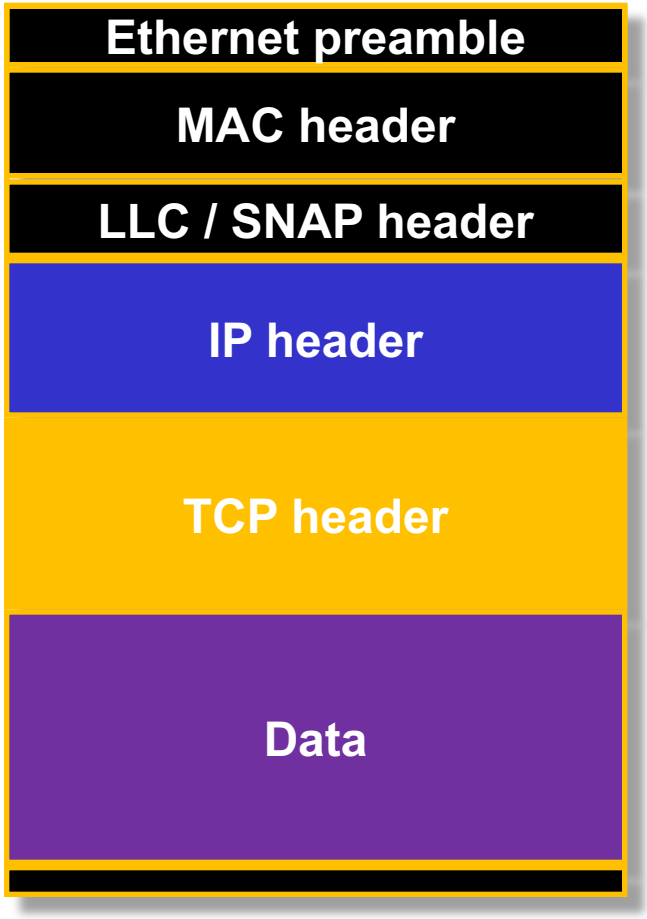


Local Area Network Protocols

IEEE 802 standards "refine" the OSI data link layer.

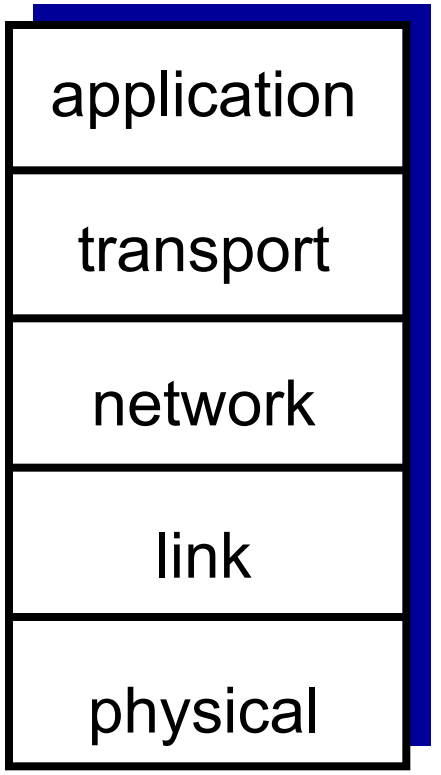


A TCP / IP / 802.3 Packet



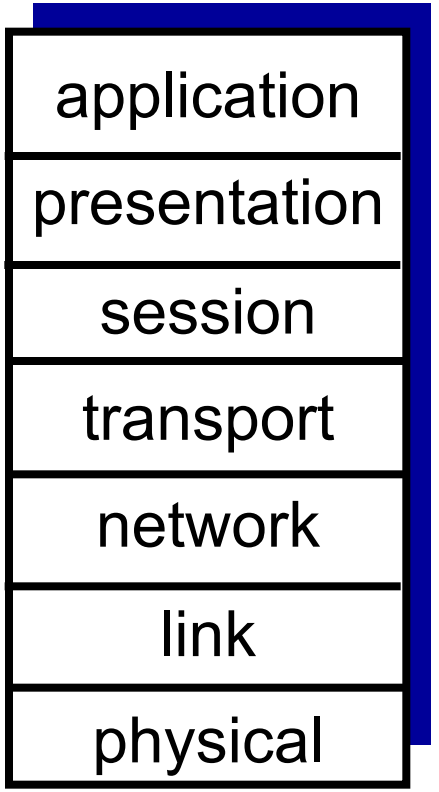
Internet protocol stack

- *application:* supporting network applications
  - FTP, SMTP, HTTP
- *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"



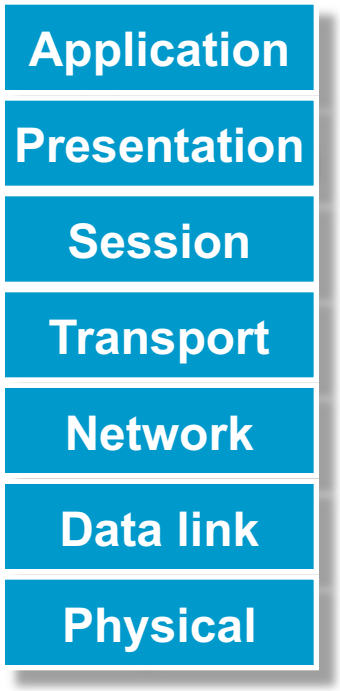
ISO/OSI reference model

- *presentation:* allow applications to interpret meaning of data, e.g., encryption, compression, machine-specific conventions
- *session:* synchronization, checkpointing, recovery of data exchange
- Internet stack "missing" these layers!
  - these services, *if needed*, must be implemented in application
  - needed?



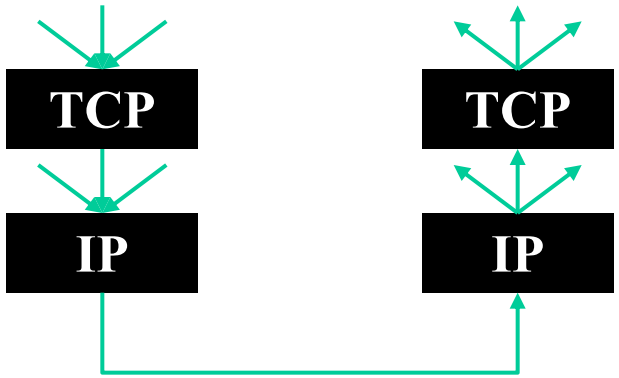
Different Sources of Components

- Application: web server/browser, mail, distributed game,..
- Presentation/session.
  - Often part of application
  - Sometimes a library
- Transport/network.
  - Typically part of the operating system
- Datalink.
  - Often written by vendor of the network interface hardware
- Physical.
  - Hardware: card and link



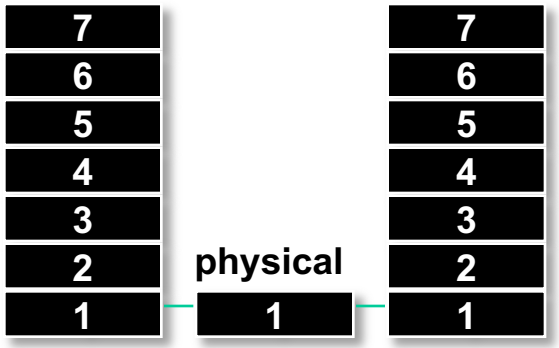
Multiplexing and Demultiplexing

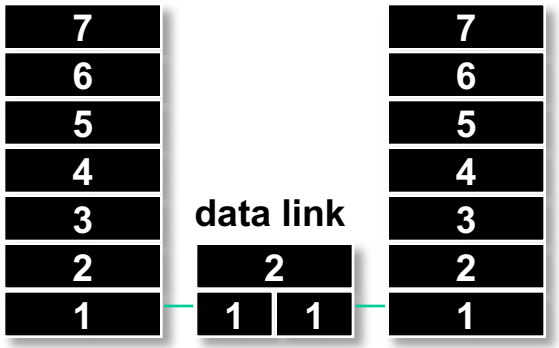
- There may be multiple implementations of each layer.
  - How does the receiver know what version of a layer to use?
- Each header includes a demultiplexing field that is used to identify the next layer.
  - Filled in by the sender
  - Used by the receiver
- Multiplexing ooccurs at multiple layers. E.g., IP, TCP, …

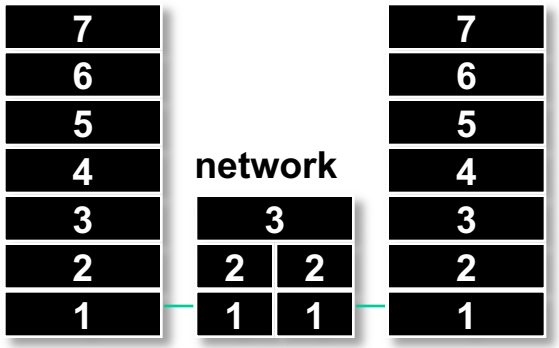


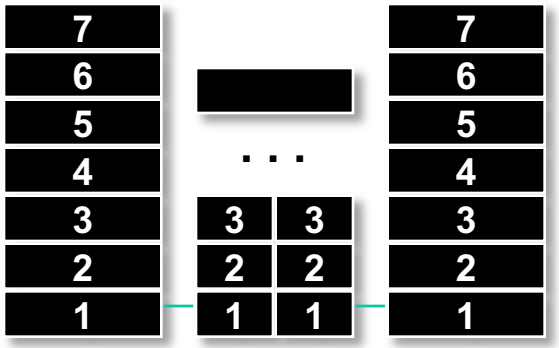
V/HL	TOS	Length
ID		Flags/Offset
TTL	Prot.	H. Checksum
Source IP address		
Destination IP address		
Options..		

Internetworking Options

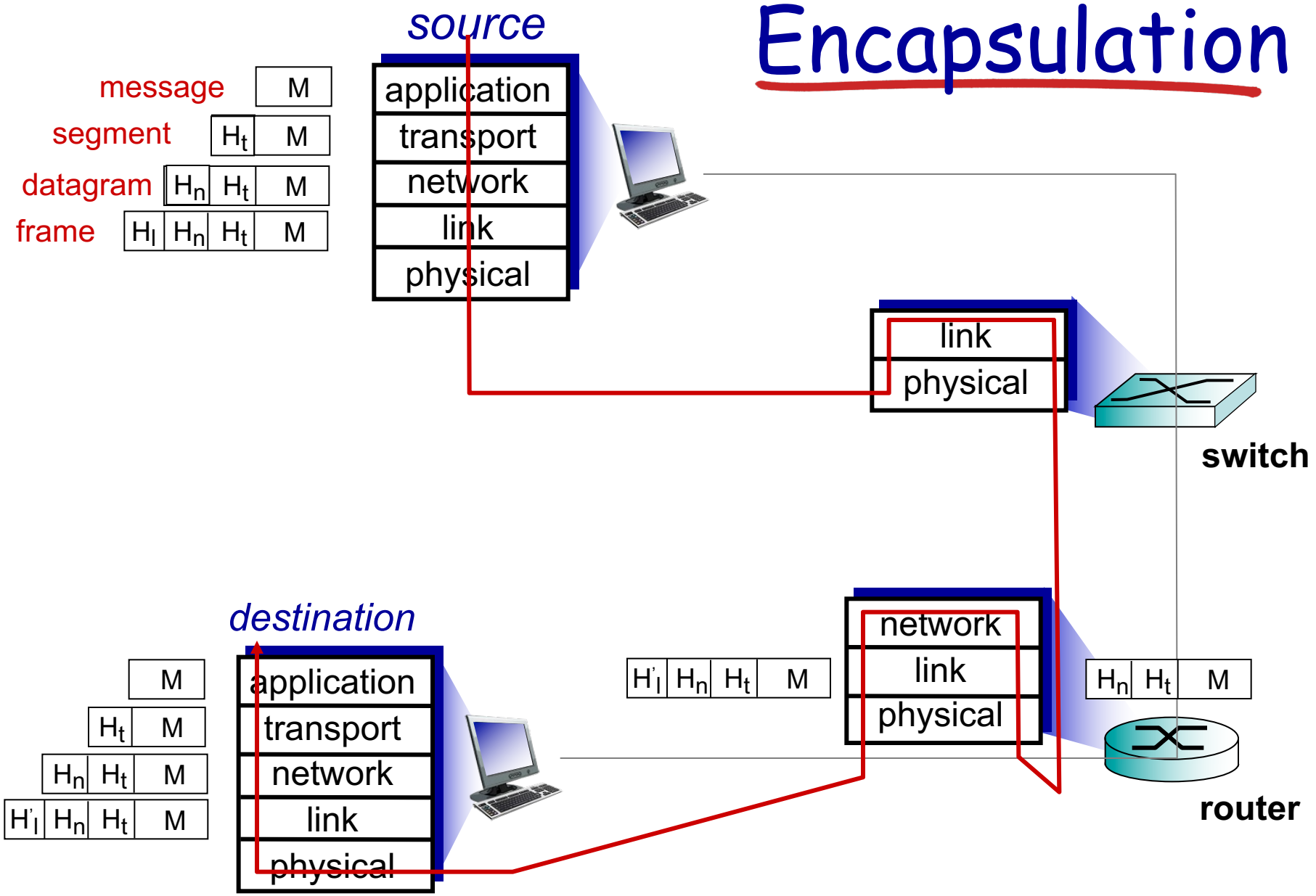




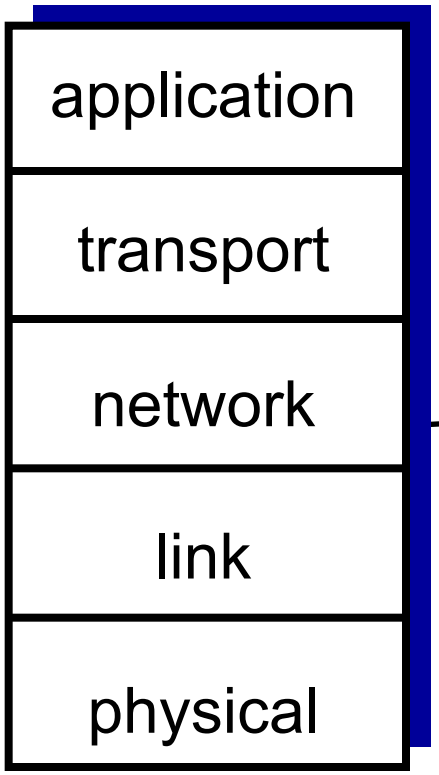




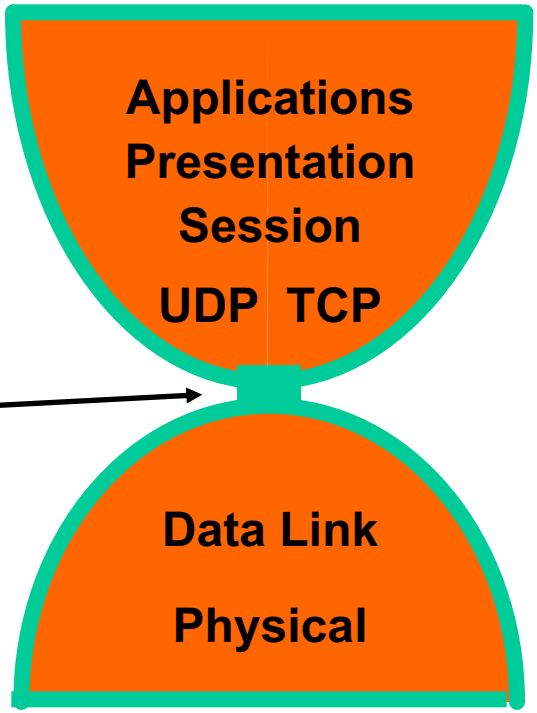
Encapsulation



The Internet Protocol Suite



Waist



The waist facilitates Interoperability.

The Hourglass Model

introduction: roadmap

1.6 delay, loss, throughput in networks
1.7 protocol layers, service models
1.8 networks under attack: security
1.9 history

Network security

- field of network security:
  - how bad guys can attack computer networks
  - how we can defend networks against attacks
  - how to design architectures that are immune to attacks
- 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!

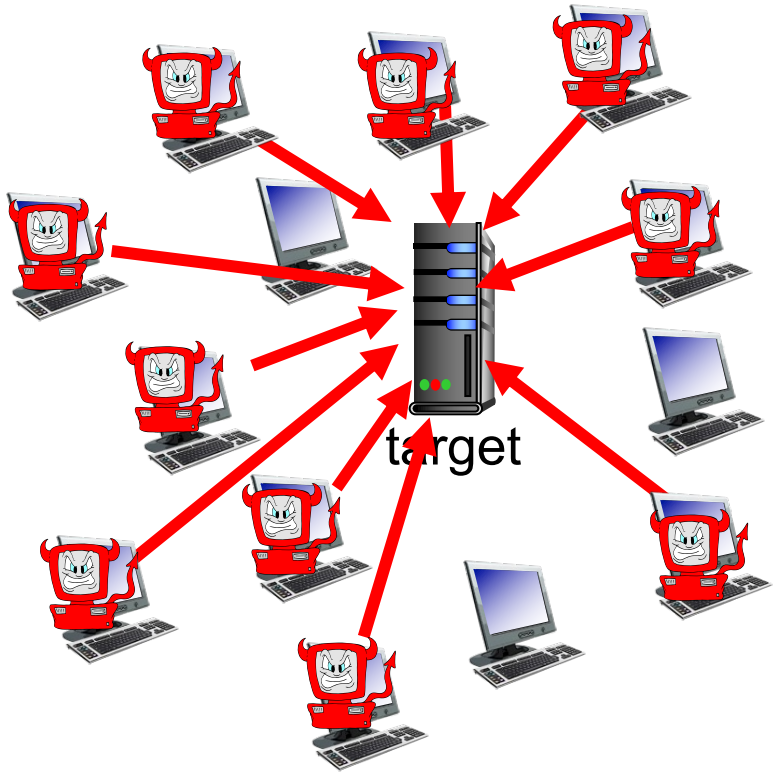
Bad guys: put malware into hosts via Internet

- malware can get in host from:
  - *trojan horse:* hidden in otherwise useful software
  - *virus:* self-replicating infection by receiving/executing object (e.g., e-mail attachment)
  - *worm:* self-replicating infection by passively receiving object that gets itself executed
- spyware malware can record keystrokes, web sites visited, upload info to collection site
- infected host can be enrolled in botnet, used for spam. DDoS attacks

Bad guys: attack server, network infrastructure

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

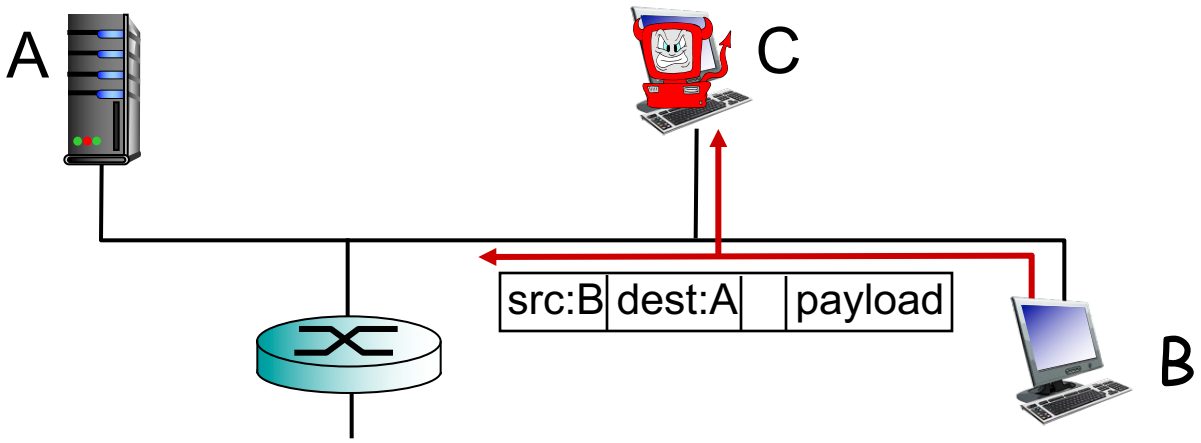
1. select target
2. break into hosts around the network (see botnet)
3. send packets to target from compromised hosts



Bad guys can sniff packets

packet "sniffing":

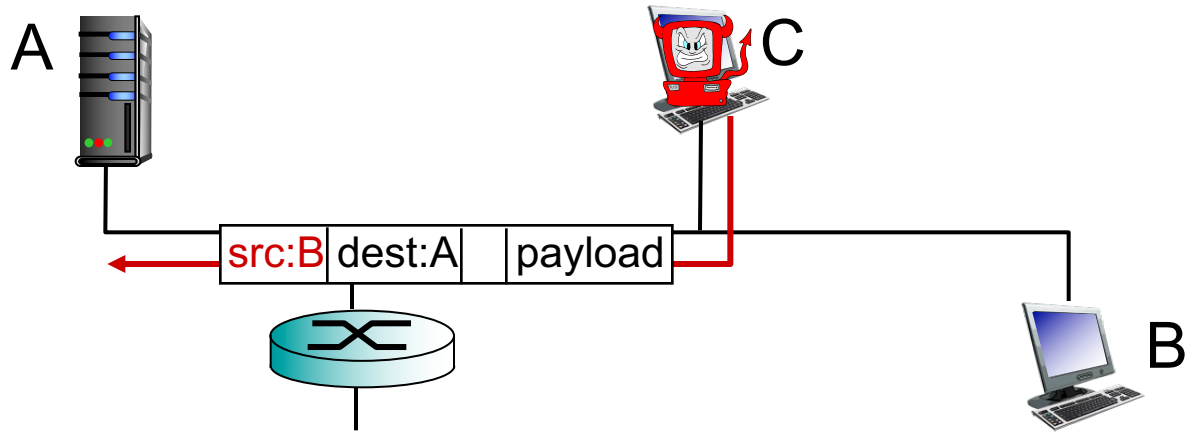
- broadcast media (shared ethernet, wireless)
- promiscuous network interface reads/records all packets (e.g., including passwords!) passing by



- wireshark software is a (free) packet-sniffer

Bad guys can use fake addresses

IP spoofing: send packet with false source address

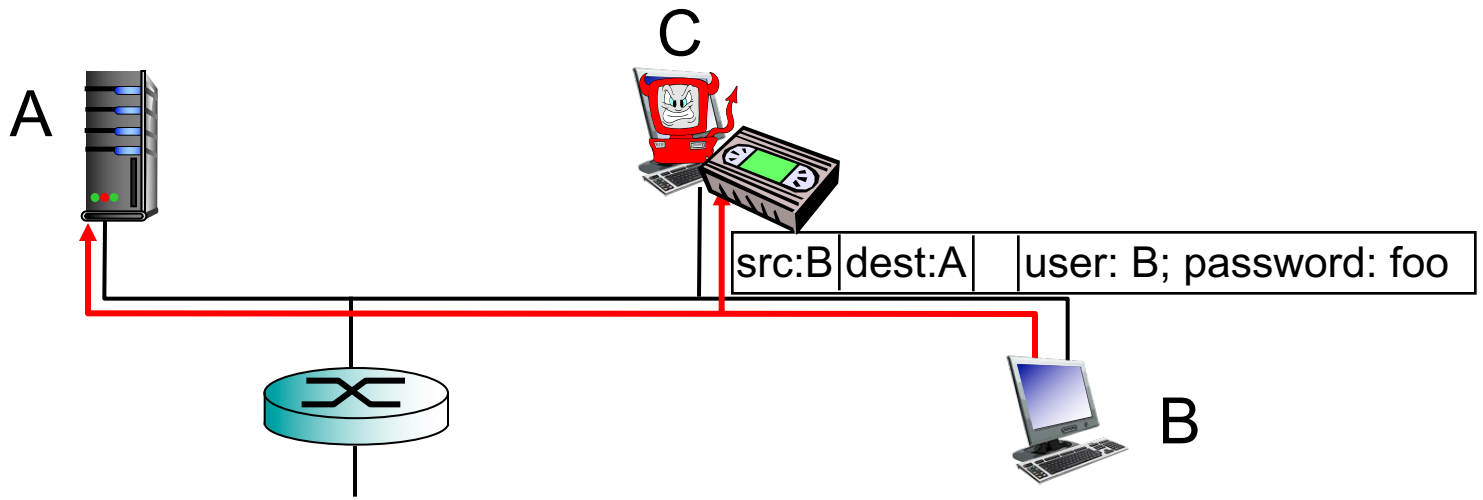


Combining with Dos attack;
Can design a new type Dos attack: Reflection DDos

Bad guys can record, playback

record-and-playback: sniff sensitive info (e.g., password), and use later

- password holder *is* that user from system point of view



introduction: roadmap

1.1 what *is* the Internet?
1.3 network core

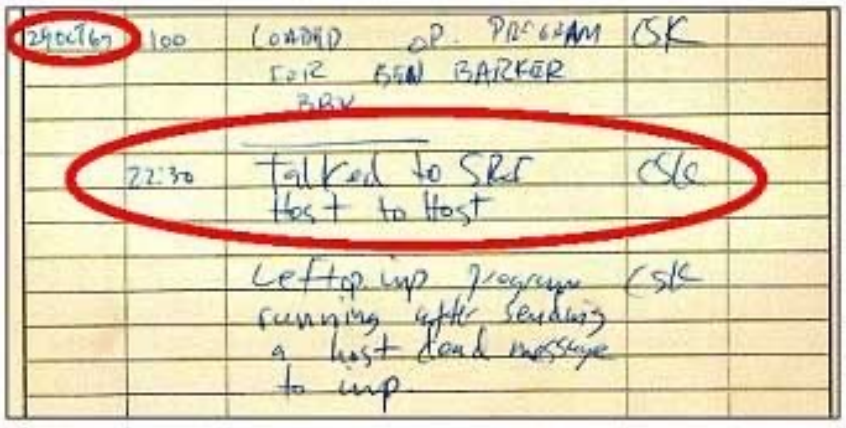
- packet switching, circuit switching, network structure

1.6 delay, loss, throughput in networks
1.7 protocol layers, service models
1.8 networks under attack: security
1.9 history

Internet history

1962-1972: Early packet-switching principles

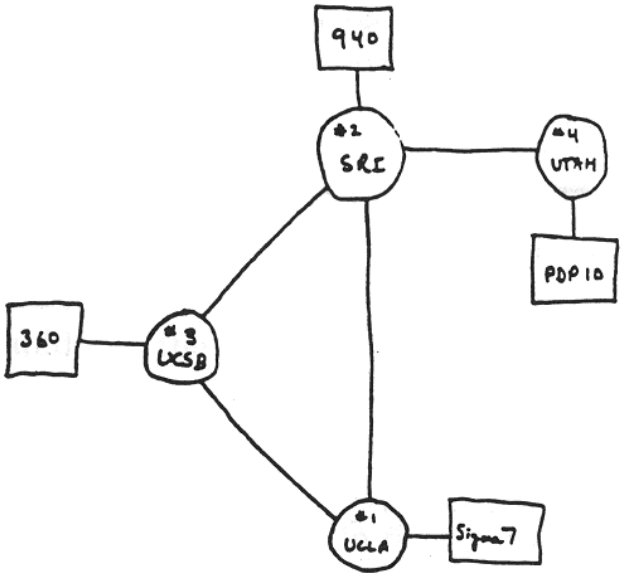
- 1961: Kleinrock - queueing theory shows effectiveness of packet-switching
- 1964: Baran - packet-switching in military nets
- 1967: ARPAnet conceived by Advanced Research Projects Agency
- 1969: first ARPAnet node operational
  - First link: UCLA-SRI



Internet history

1962-1972: Early packet-switching principles

- 1969: first ARPAnet node operational
  - Dec 5, 4 nodes



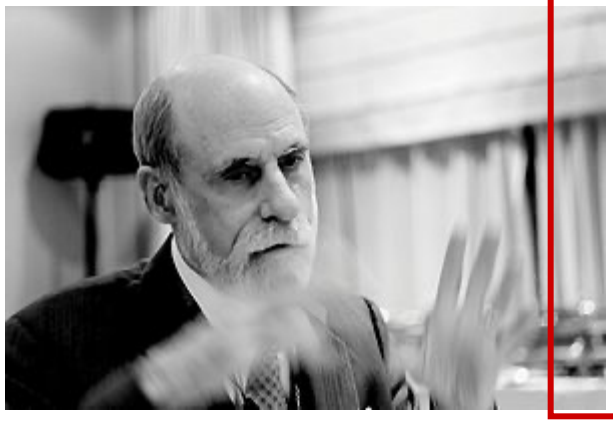
THE ARPA NETWORK

- 1972:
  - ARPAnet public demo
  - NCP (Network Control Protocol) first host-host protocol
  - first e-mail program
  - ARPAnet has 15 nodes

Internet history

1972-1980: Internetworking, new and proprietary nets

- 1970: ALOHAnet satellite network in Hawaii
- 1974: Cerf and Kahn - architecture for interconnecting networks



Cerf and Kahn's internetworking principles:

- minimalism, autonomy - no internal changes required to interconnect networks
- best effort service model
- stateless routers
- decentralized control

define today's Internet architecture

Internet history

1972-1980: Internetworking, new and proprietary nets

- 1976: Ethernet at Xerox PARC
- late70's: proprietary architectures: DECnet, SNA, XNA
- late 70's: switching fixed length packets (ATM precursor)
- 1979: ARPAnet has 200 nodes
- 1983: deployment of TCP/IP

Sun microsystems

Bill Joy: TCP/IP in BSD UNIX

Internet history

1980-1990: new protocols, a proliferation of networks

- 1982: smtp e-mail protocol defined
- 1983: DNS defined for name-to-IP-address translation
- 1985: ftp protocol defined
- 1983 !

Jan 1, TCP/IP deployment

new national network backbones: Csnet, BITnet, NSFnet, Minitel

- 1988:

-TCP congestion control

-First network worms

-100,000 hosts connected to confederation of networks

Robert Tappan Morris

Internet history

1990, 2000's: commercialization, the Web, new apps

- early 1990's: ARPAnet decommissioned
- 1991: NSF lifts restrictions on commercial use of NSFnet (decommissioned, 1995)
- early 1990s: Web
  - hypertext [Bush 1945, Nelson 1960's]
  - HTML, HTTP: Berners-Lee
  - 1994: Mosaic, later Netscape
  - late 1990's: commercialization of the Web

late 1990's – 2000's:

- more killer apps: instant messaging, P2P file sharing
- network security to forefront
- est. 50 million host, 100 million+ users
- backbone links running at Gbps

Internet history

2010~:

- >750 million hosts (2012)
- Over 2 billion PC, tablet and smartphone shipments worldwide in 2019 (Gartner)
- m2m traffic
- voice, video over IP
- P2P applications: BitTorrent (file sharing) Skype (VoIP), PPLive (video)
- more applications:
  - Social networks: facebook, Twitter
  - YouTube, gaming,
- wireless, mobility

Introduction: summary

covered a "ton" of material!

- Internet overview
- what's a protocol?
- network edge, core, access network
  - packet-switching versus circuit-switching
  - Internet structure
- performance: loss, delay, throughput
- layering, service models
- security
- history

you now have:

- context, overview, "feel" of networking
- more depth, detail *to follow!*