# Chapter 6
The Link Layer and LANs

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.

Thanks and enjoy! JFK/KWR

© All material copyright 1996-2016
J.F Kurose and K.W. Ross, All Rights Reserved
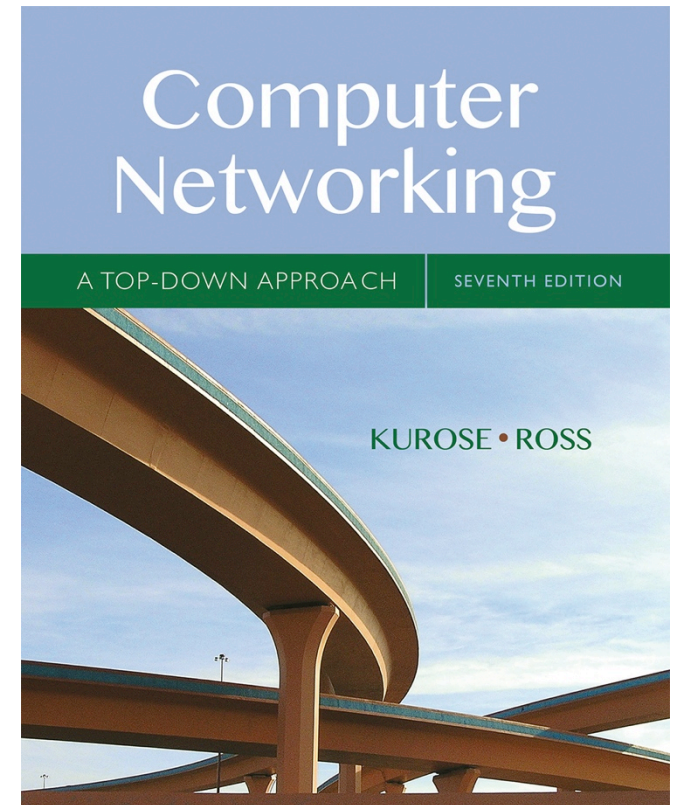
Computer Networking: A Top Down Approach

7th edition
Jim Kurose, Keith Ross
Pearson/Addison Wesley
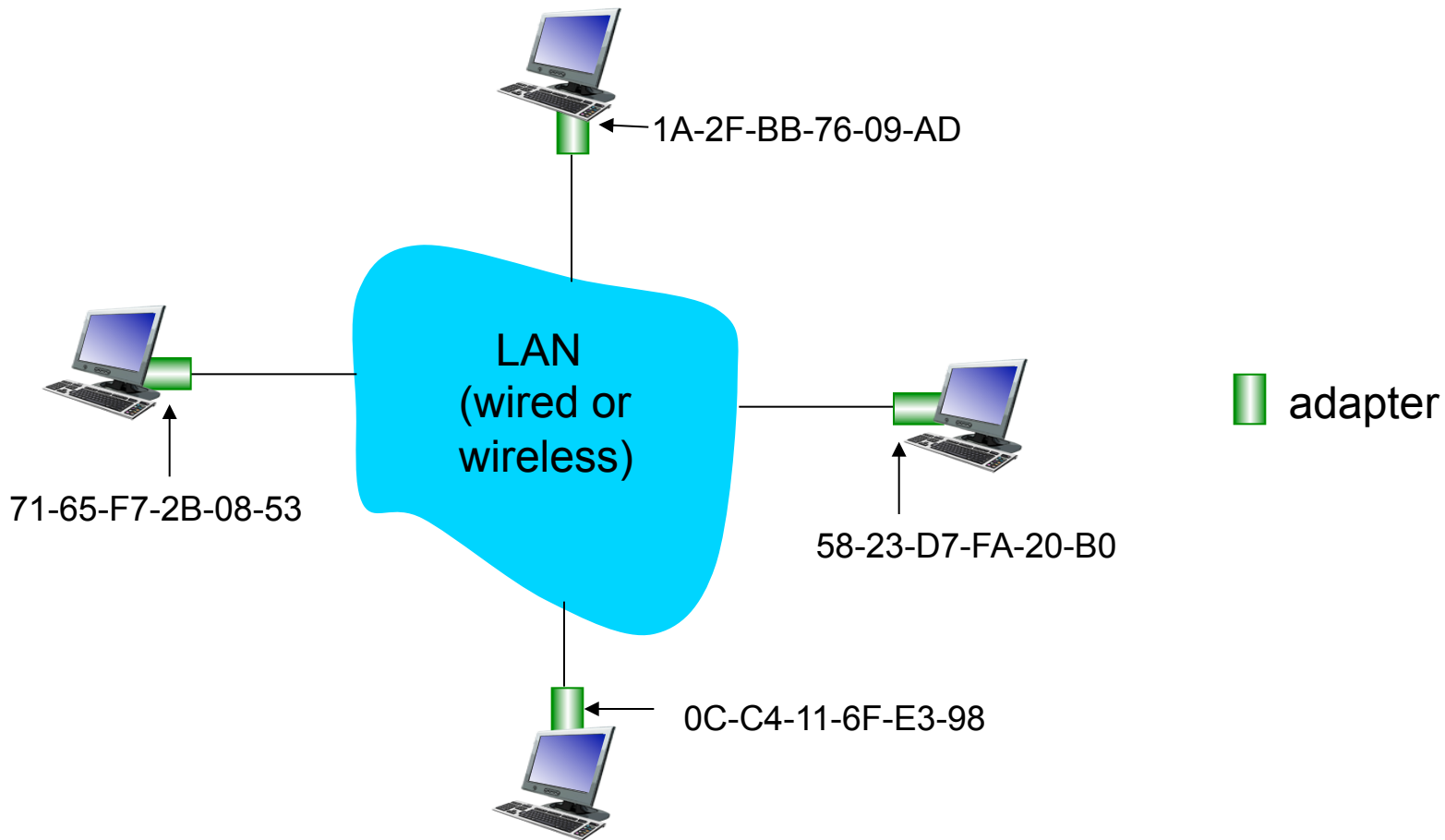April 2016

Link layer, LANs: outline

MAC addresses and ARP

- **32-bit IP address:**
  - *network-layer* address for interface
  - used for layer 3 (network layer) forwarding

- **MAC (or LAN or physical or Ethernet) address:**
  - function: *used 'locally'' to get frame from one interface to another physically-connected interface (same network, in IP-addressing sense)*
  - 48 bit MAC address (for most LANs) burned in NIC ROM, also sometimes software settable
  - e.g.: 1A-2F-BB-76-09-AD

hexadecimal (base 16) notation
(each "numeral" represents 4 bits)

LAN addresses and ARP

each adapter on LAN has unique *LAN* address

1A-2F-BB-76-09-AD

LAN
(wired or
wireless)

adapter

71-65-F7-2B-08-53

58-23-D7-FA-20-B0

0C-C4-11-6F-E3-98

LAN addresses (more)

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness)
- analogy:
  - MAC address: like Social Security Number
  - IP address: like postal address
- MAC flat address ➜ portability
  - can move LAN card from one LAN to another
- IP hierarchical address *not* portable
  - address depends on IP subnet to which node is attached

ARP: address resolution protocol

Question: how to determine interface's MAC address, knowing its IP address?
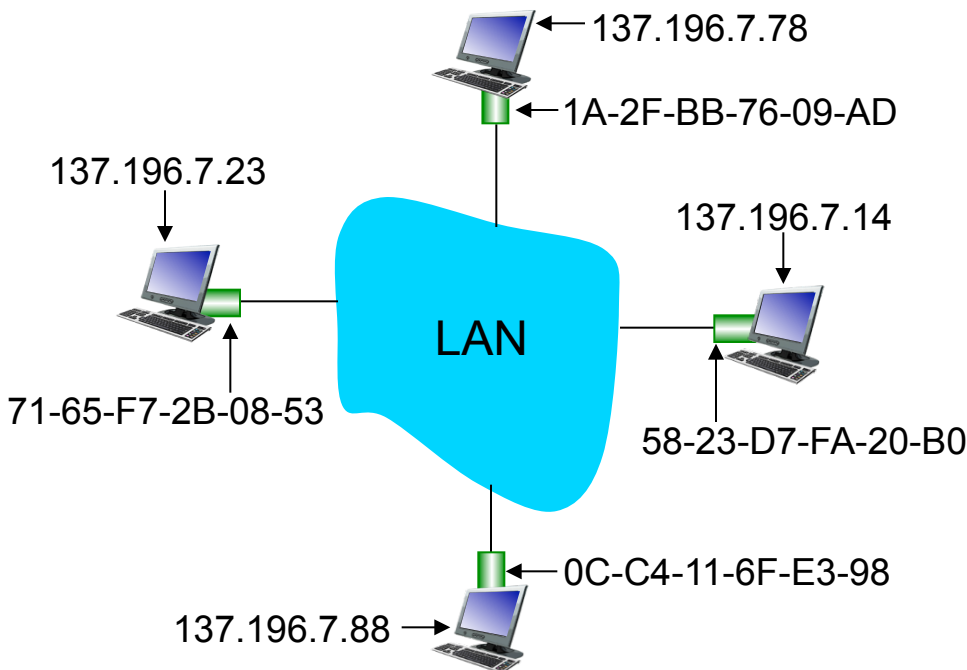
ARP table: each IP node (host, router) on LAN has table

- IP/MAC address mappings for some LAN nodes:

< IP address; MAC address; TTL>

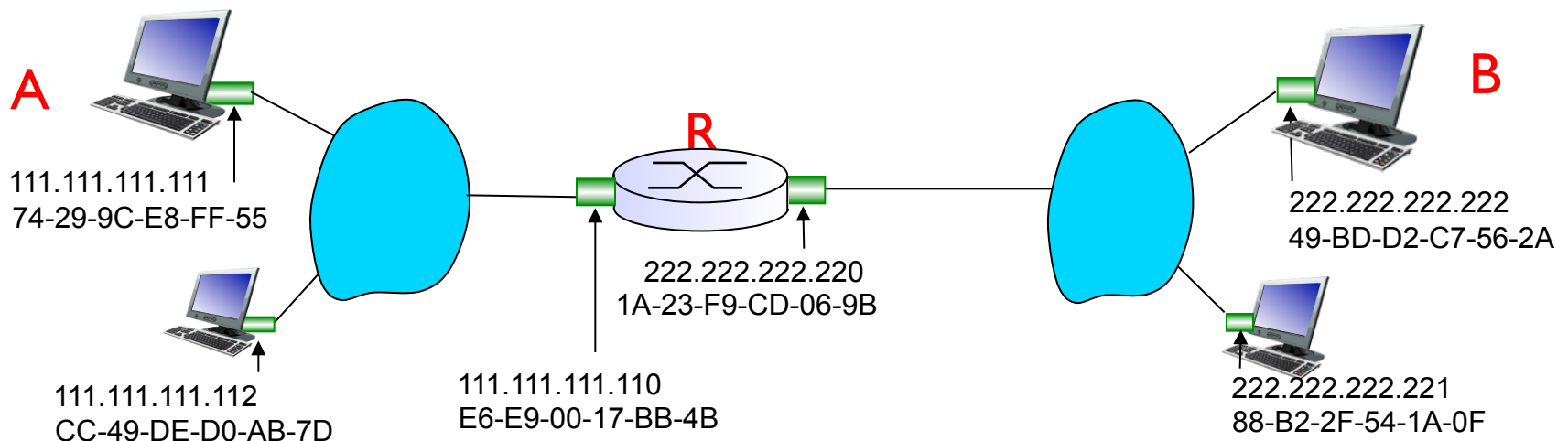- TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)

137.196.7.78

1A-2F-BB-76-09-AD

137.196.7.23

137.196.7.14

71-65-F7-2B-08-53

LAN

58-23-D7-FA-20-B0

0C-C4-11-6F-E3-98

137.196.7.88

ARP protocol: same LAN

- A wants to send datagram to B
  - B's MAC address not in A's ARP table.
- A broadcasts ARP query packet, containing B's IP address
  - destination MAC address = FF-FF-FF-FF-FF-FF
  - all nodes on LAN receive ARP query
- B receives ARP packet, replies to A with its (B's) MAC address
  - frame sent to A's MAC address (unicast)

- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
  - soft state: information that times out (goes away) unless refreshed
- ARP is "plug-and-play":
  - nodes create their ARP tables *without intervention from net administrator*
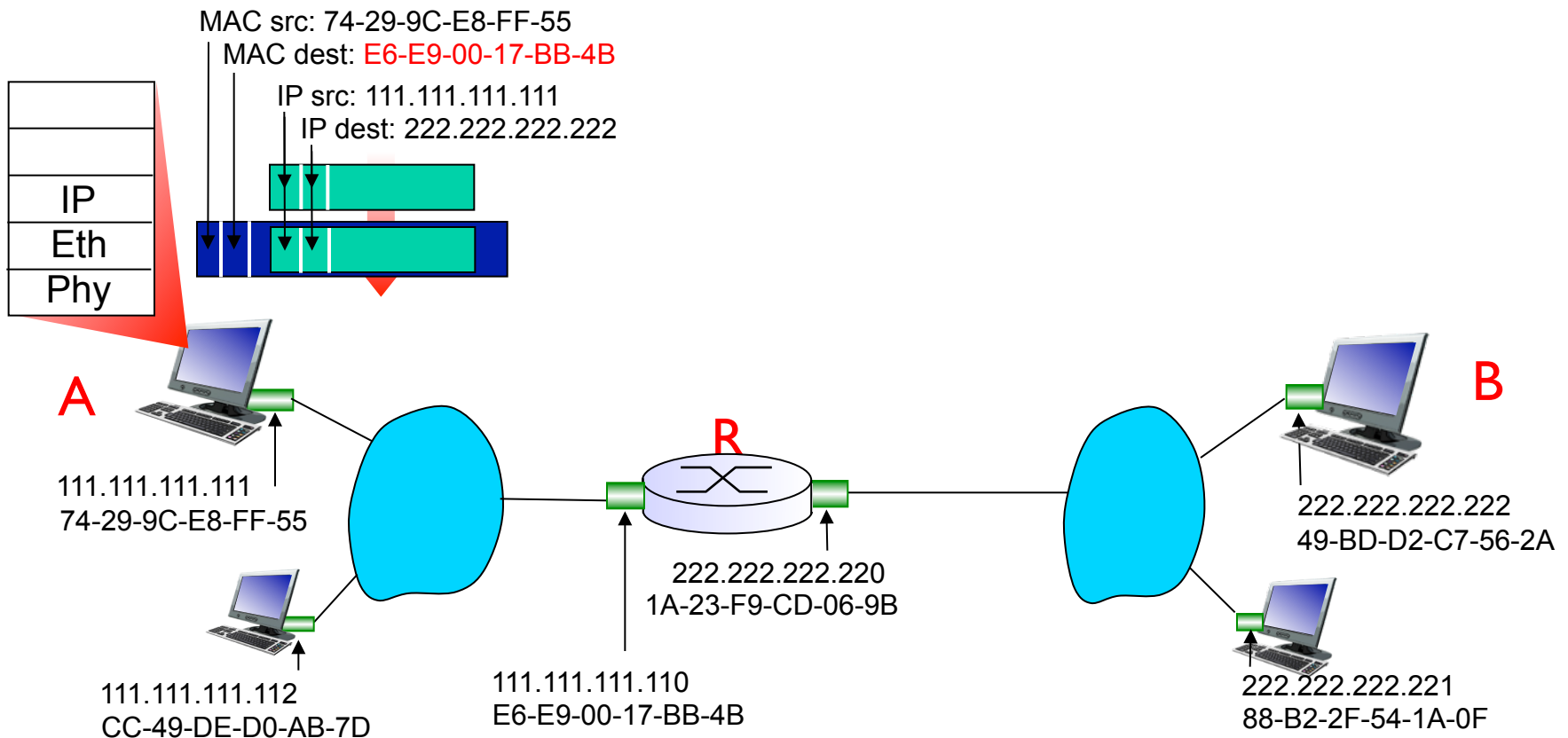
Addressing: routing to another LAN

walkthrough: send datagram from A to B via R
- focus on addressing – at IP (datagram) and MAC layer (frame)
- assume A knows B's IP address
- assume A knows IP address of first hop router, R (how?)
- assume A knows R's MAC address (how?)

A

111.111.111.111
74-29-9C-E8-FF-55

111.111.111.112
CC-49-DE-D0-AB-7D

R

222.222.222.220
1A-23-F9-CD-06-9B

111.111.111.110
E6-E9-00-17-BB-4B

B

222.222.222.222
49-BD-D2-C7-56-2A
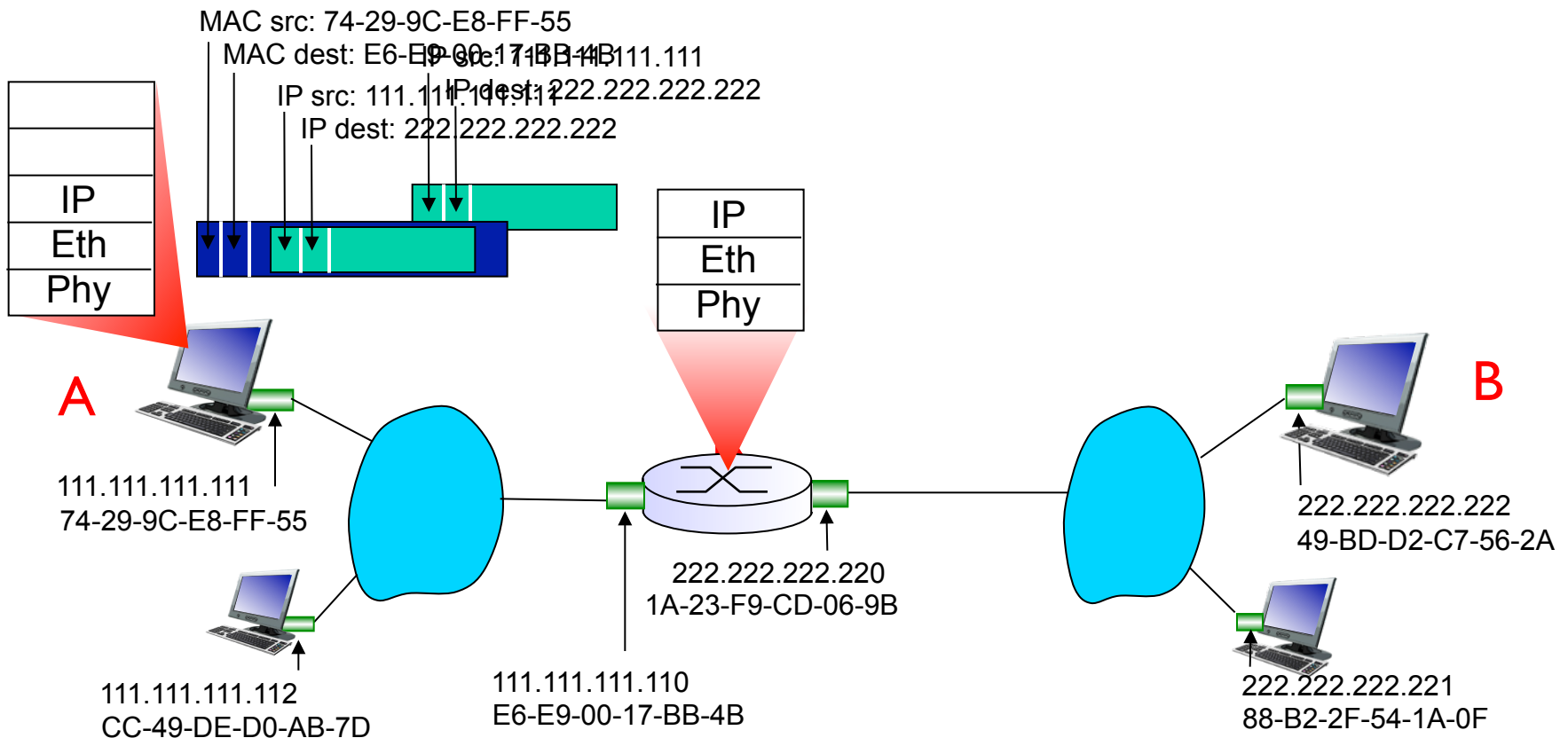
222.222.222.221
88-B2-2F-54-1A-0F

Addressing: routing to another LAN

- A creates IP datagram with IP source A, destination B
- A creates link-layer frame with R's MAC address as destination address, frame contains A-to-B IP datagram
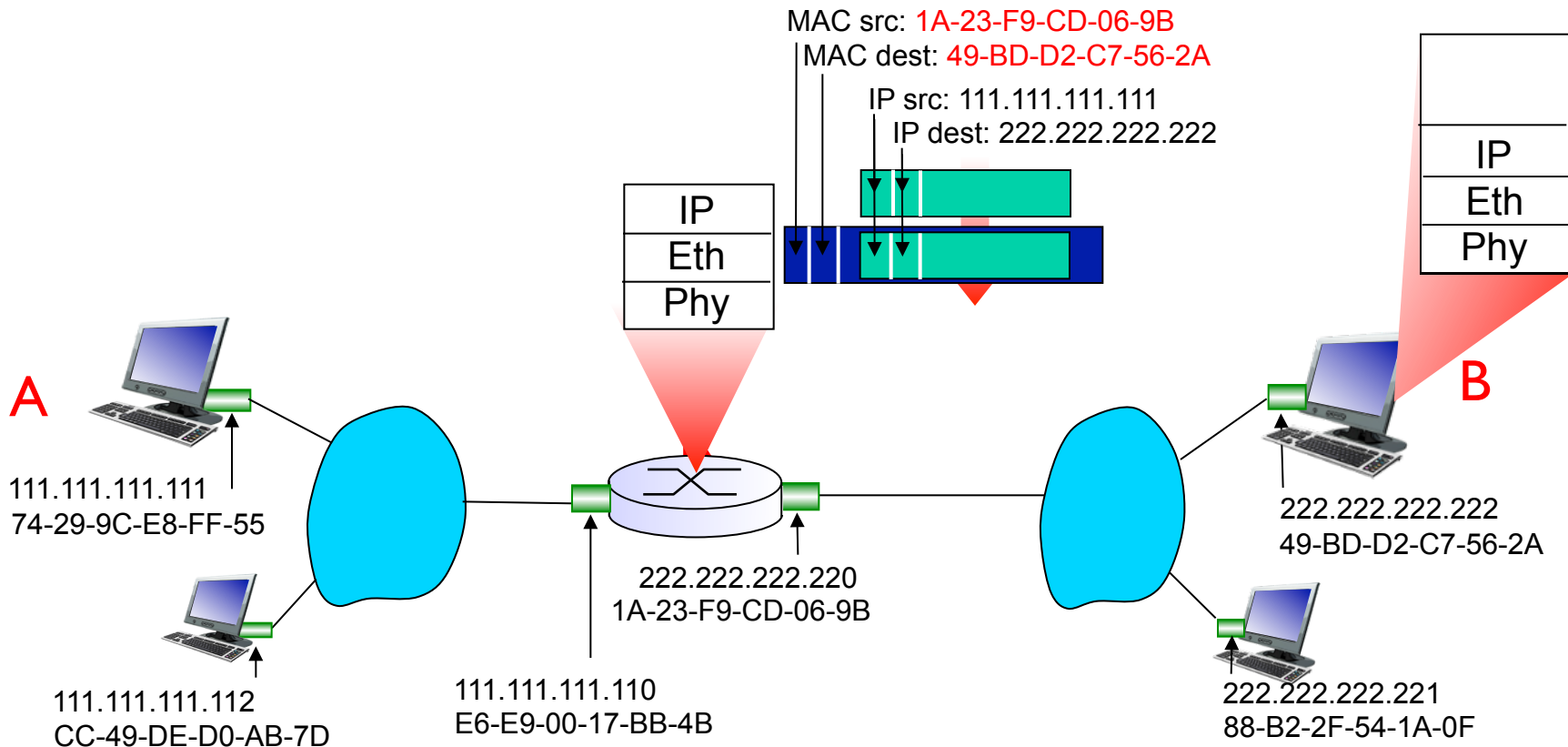
MAC src: 74-29-9C-E8-FF-55
MAC dest: E6-E9-00-17-BB-4B

IP src: 111.111.111.111
IP dest: 222.222.222.222

IP
Eth
Phy

A

111.111.111.111
74-29-9C-E8-FF-55

111.111.111.112
CC-49-DE-D0-AB-7D

R

222.222.222.220
1A-23-F9-CD-06-9B

111.111.111.110
E6-E9-00-17-BB-4B

B

222.222.222.222
49-BD-D2-C7-56-2A

222.222.222.221
88-B2-2F-54-1A-0F

Addressing: routing to another LAN

- frame sent from A to R
- frame received at R, datagram removed, passed up to IP

MAC src: 74-29-9C-E8-FF-55
MAC dest: E6-E9-00-17-BB-4B

IP src: 111.111.111.111
IP dest: 222.222.222.222

IP src: 111.111.111.111
IP dest: 222.222.222.222

| IP |
| Eth |
| Phy |

| IP |
| Eth |
| Phy |

A

B

111.111.111.111
74-29-9C-E8-FF-55

111.111.111.112
CC-49-DE-D0-AB-7D

222.222.222.220
1A-23-F9-CD-06-9B

111.111.111.110
E6-E9-00-17-BB-4B

222.222.222.222
49-BD-D2-C7-56-2A
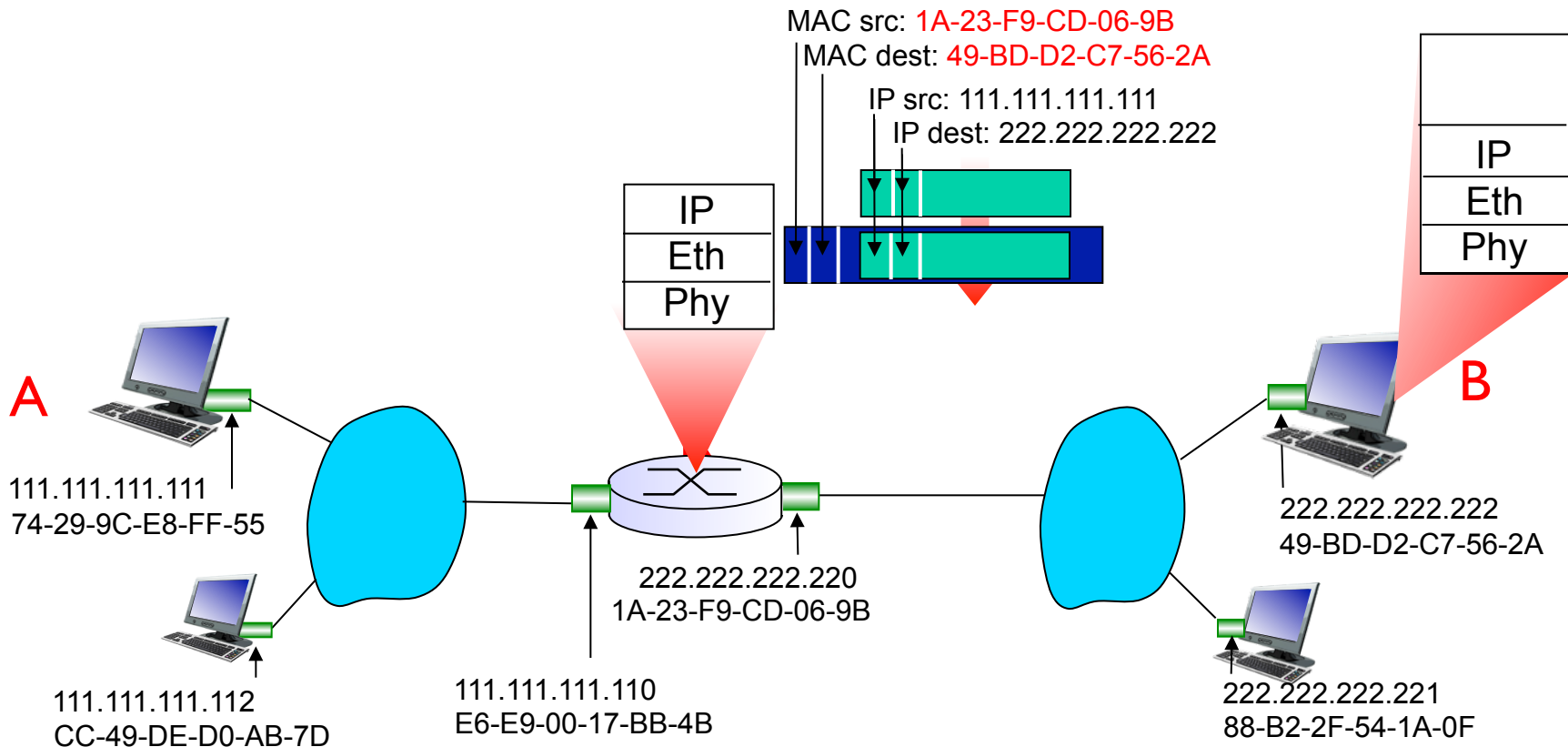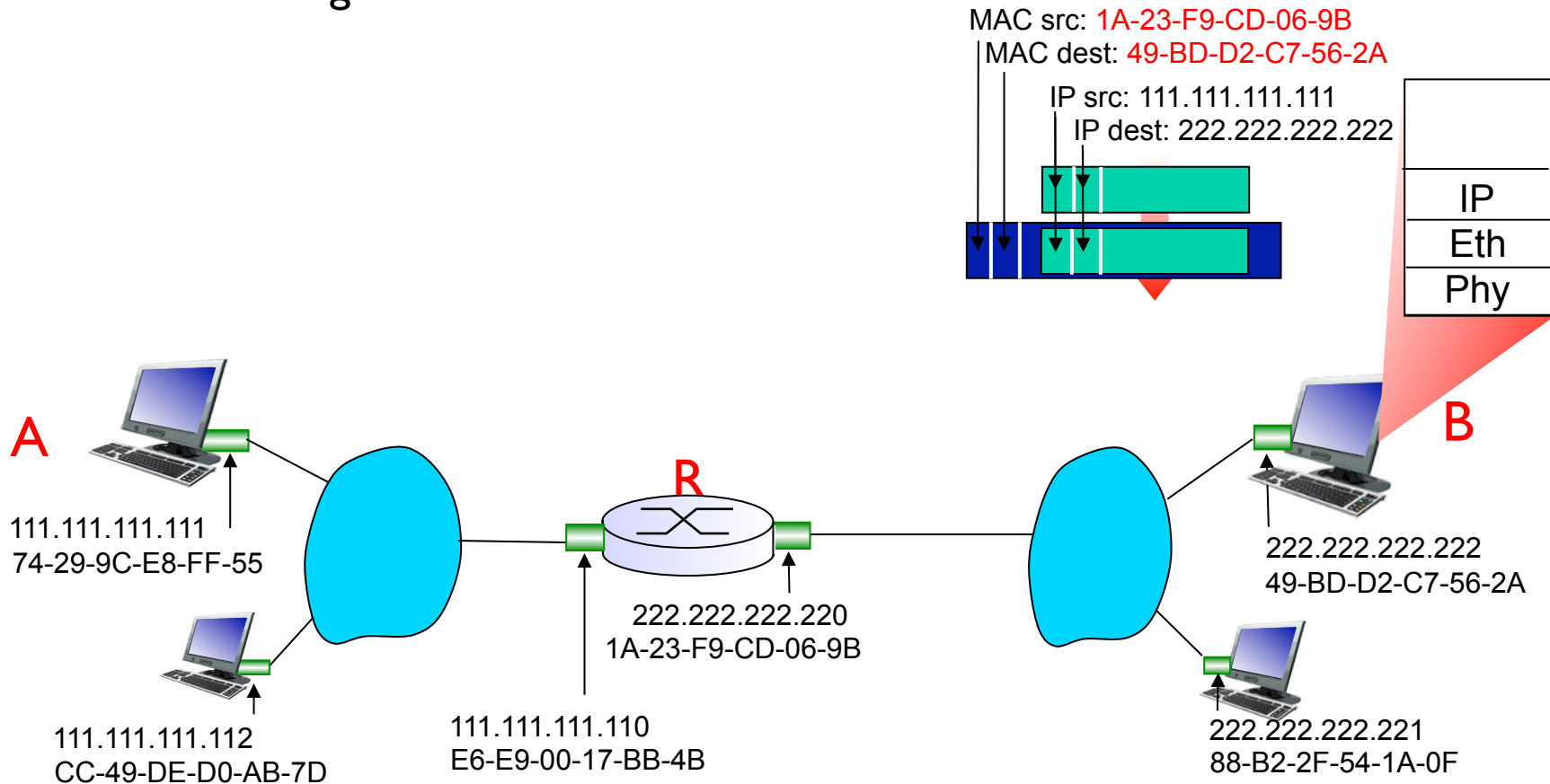
222.222.222.221
88-B2-2F-54-1A-0F

Addressing: routing to another LAN

- R forwards datagram with IP source A, destination B

- R creates link-layer frame with B's MAC address as destination address, frame contains A-to-B IP datagram

MAC src: 1A-23-F9-CD-06-9B
MAC dest: 49-BD-D2-C7-56-2A

IP src: 111.111.111.111
IP dest: 222.222.222.222

IP
Eth
Phy

IP
Eth
Phy

A

B

111.111.111.111
74-29-9C-E8-FF-55

111.111.111.112
CC-49-DE-D0-AB-7D

222.222.222.220
1A-23-F9-CD-06-9B

111.111.111.110
E6-E9-00-17-BB-4B

222.222.222.222
49-BD-D2-C7-56-2A

222.222.222.221
88-B2-2F-54-1A-0F

Addressing: routing to another LAN

- R forwards datagram with IP source A, destination B
- R creates link-layer frame with B's MAC address as destination address, frame contains A-to-B IP datagram

MAC src: 1A-23-F9-CD-06-9B
MAC dest: 49-BD-D2-C7-56-2A

IP src: 111.111.111.111
IP dest: 222.222.222.222

IP
Eth
Phy

IP
Eth
Phy

A

B

111.111.111.111
74-29-9C-E8-FF-55

111.111.111.112
CC-49-DE-D0-AB-7D

222.222.222.220
1A-23-F9-CD-06-9B

111.111.111.110
E6-E9-00-17-BB-4B

222.222.222.222
49-BD-D2-C7-56-2A

222.222.222.221
88-B2-2F-54-1A-0F

Addressing: routing to another LAN

- R forwards datagram with IP source A, destination B
- R creates link-layer frame with B's MAC address as dest, frame contains A-to-B IP datagram

MAC src: 1A-23-F9-CD-06-9B
MAC dest: 49-BD-D2-C7-56-2A

IP src: 111.111.111.111
IP dest: 222.222.222.222

IP
Eth
Phy

A

B

111.111.111.111
74-29-9C-E8-FF-55

R

222.222.222.222
49-BD-D2-C7-56-2A

111.111.111.112
CC-49-DE-D0-AB-7D

222.222.222.220
1A-23-F9-CD-06-9B

111.111.111.110
E6-E9-00-17-BB-4B

222.222.222.221
88-B2-2F-54-1A-0F

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

Link layer, LANs: outline

6.1 introduction, services

6.2 error detection, correction

6.3 multiple access protocols

6.4 LANs
- addressing, ARP
- Ethernet
- switches
- VLANS

6.5 link virtualization: MPLS

6.6 data center networking

6.7 a day in the life of a web request

Ethernet

"dominant" wired LAN technology:

- single chip, multiple speeds (e.g., Broadcom BCM5761)
- first widely used LAN technology
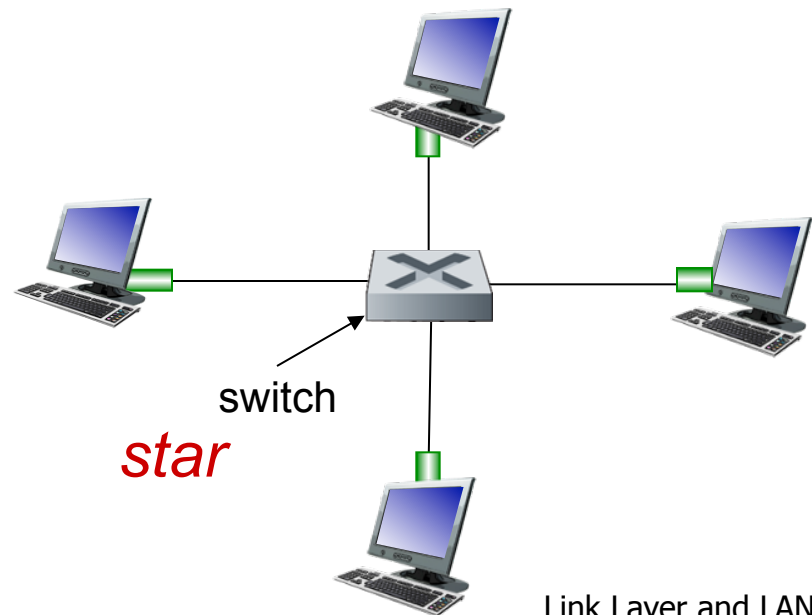- simpler, cheap
- kept up with speed race: 10 Mbps – 10 Gbps
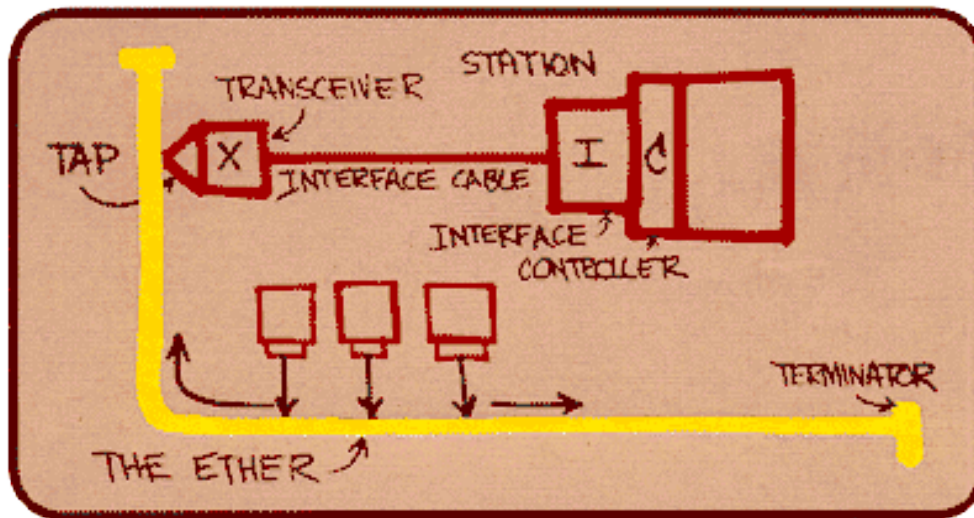
Metcalfe's Ethernet sketch

Ethernet: physical topology

- *bus:* popular through mid 90s
  - all nodes in same collision domain (can collide with each other)
- *star:* prevails today
  - active *switch* in center
  - each "spoke" runs a (separate) Ethernet protocol (nodes do not collide with each other)

bus: coaxial cable

switch

star
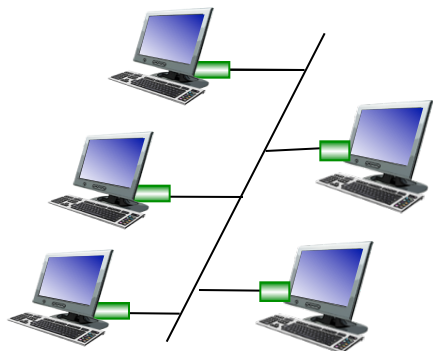
Ethernet frame structure

sending adapter encapsulates IP datagram (or other network layer protocol packet) in Ethernet frame

preamble:

- 7 bytes with pattern 10101010 followed by one byte with pattern 10101011
- used to synchronize receiver, sender clock rates

Ethernet frame structure (more)

- *addresses:* 6 byte source, destination MAC addresses
  - if adapter receives frame with matching destination address, or with broadcast address (e.g. ARP packet), it passes data in frame to network layer protocol
  - otherwise, adapter discards frame
- *type:* indicates higher layer protocol (mostly IP but others possible, e.g., Novell IPX, AppleTalk)
- *CRC:* cyclic redundancy check at receiver
  - error detected: frame is dropped

type

| preamble | dest. address | source address | | data (payload) | CRC |

Ethernet: unreliable, connectionless

- *connectionless:* no handshaking between sending and receiving NICs

- *unreliable:* receiving NIC doesn't send acks or nacks to sending NIC
  - data in dropped frames recovered only if initial sender uses higher layer rdt (e.g., TCP), otherwise dropped data lost

- Ethernet's MAC protocol: unslotted *CSMA/CD with binary backoff*

802.3 Ethernet standards: link & physical layers

- *many* different Ethernet standards
  - common MAC protocol and frame format
  - different speeds: 2 Mbps, 10 Mbps, 100 Mbps, 1Gbps, 10 Gbps, 40 Gbps
  - different physical layer media: fiber, cable
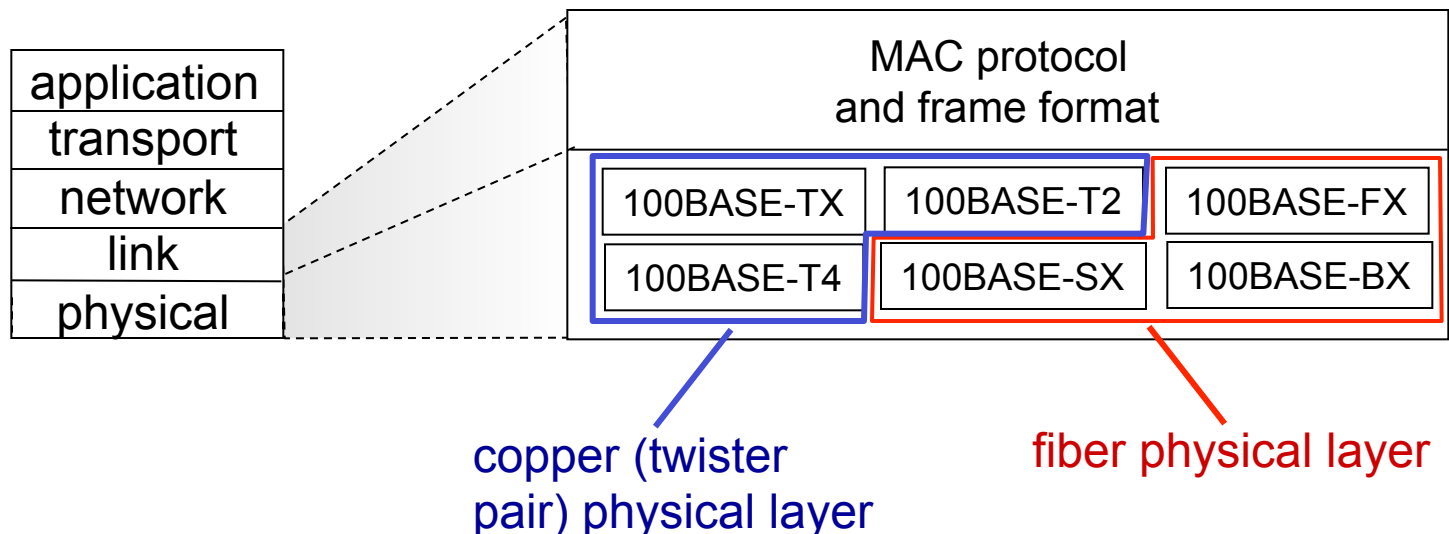
| application |
| transport |
| network |
| link |
| physical |

MAC protocol and frame format		
100BASE-TX	100BASE-T2	100BASE-FX
100BASE-T4	100BASE-SX	100BASE-BX

copper (twister pair) physical layer

fiber physical layer

Link layer, LANs: outline

Ethernet switch

- **link-layer device: takes an *active* role**
  - store, forward Ethernet frames
  - examine incoming frame's MAC address, selectively forward frame to one-or-more outgoing links when frame is to be forwarded on segment, uses CSMA/CD to access segment
- *transparent*
  - hosts are unaware of presence of switches
- *plug-and-play, self-learning*
  - switches do not need to be configured

Switch: *multiple* simultaneous transmissions

- hosts have dedicated, direct connection to switch
- switches buffer packets
- Ethernet protocol used on *each* incoming link, but no collisions; full duplex
  - each link is its own collision domain
- *switching:* A-to-A' and B-to-B' can transmit simultaneously, without collisions
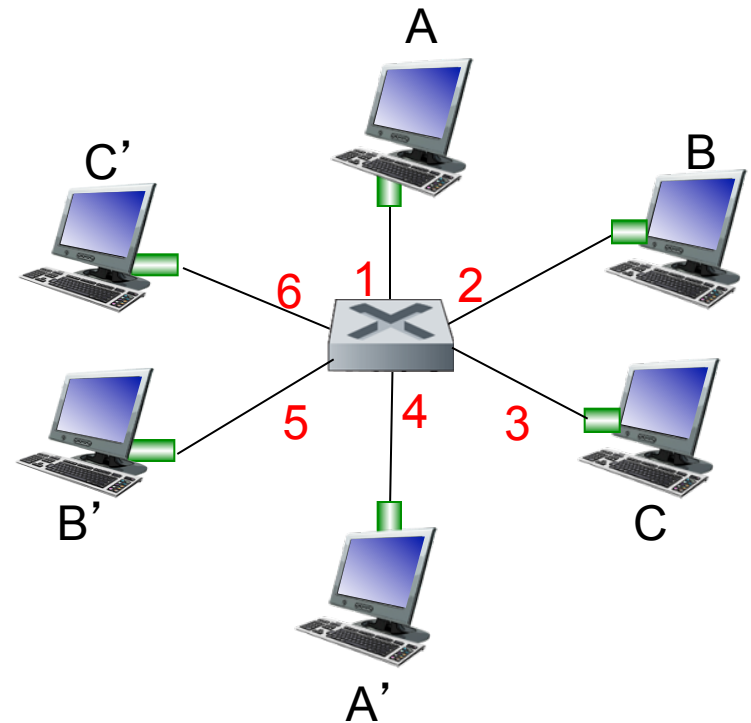
switch with six interfaces
(*1,2,3,4,5,6*)

Switch forwarding table

Q: how does switch know A'
reachable via interface 4, B'
reachable via interface 5?

- *A:* each switch has a switch
table, each entry:
  - (MAC address of host, interface
    to reach host, time stamp)
  - looks like a routing table!

Q: how are entries created,
maintained in switch table?
  - something like a routing
    protocol?
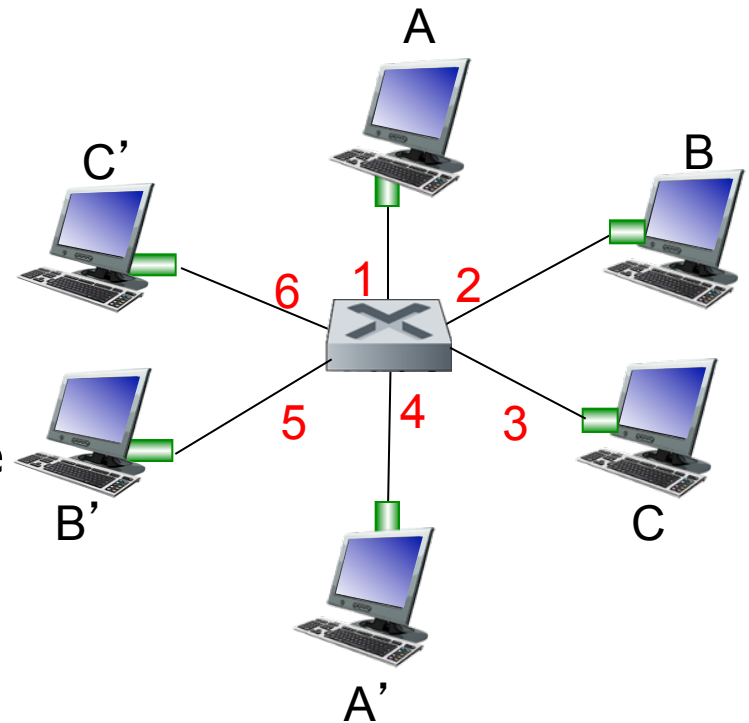
*switch with six interfaces
(1,2,3,4,5,6)*

Switch: self-learning

Source: A
Dest: A'

A | A A' |

- switch *learns* which hosts can be reached through which interfaces
  - when frame received, switch "learns" location of sender: incoming LAN segment
  - records sender/location pair in switch table

C'

B

6 1 2

5 4 3

B'

C

A'
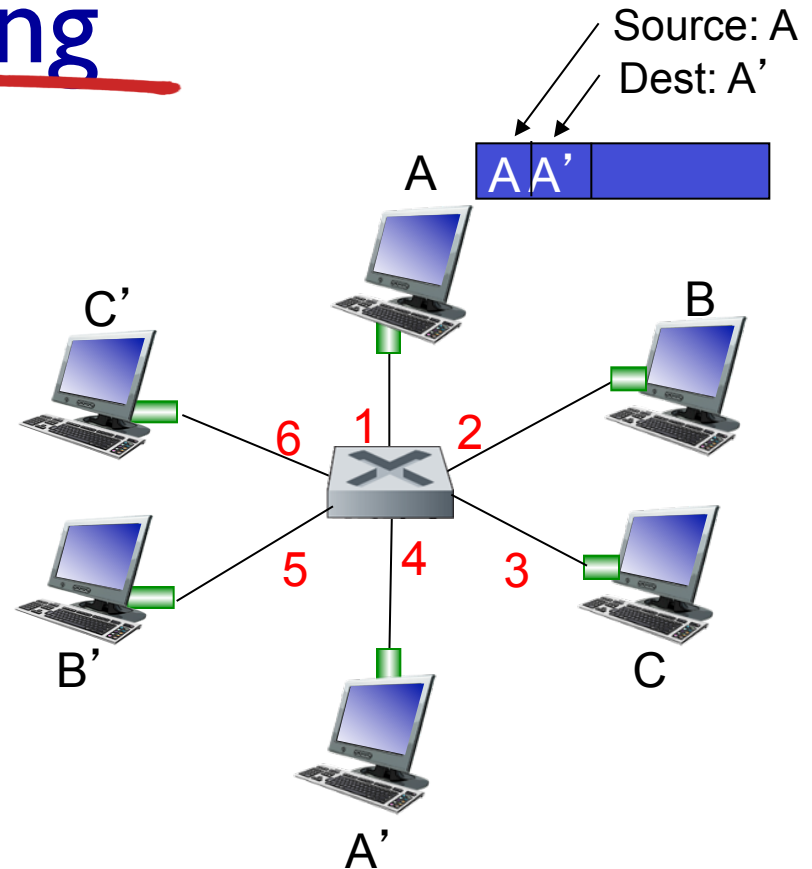
MAC addr	interface	TTL
A	1	60

Switch table (initially empty)

Switch: frame filtering/forwarding

when frame received at switch:

1. record incoming link, MAC address of sending host
2. index switch table using MAC destination address
3. if entry found for destination
        then {
        if destination on segment from which frame arrived
            then drop frame
            else forward frame on interface indicated by entry
        }
        else flood /* forward on all interfaces except arriving
                        interface */

Self-learning, forwarding: example

- frame destination, A', location unknown: *flood*

- destination A location known: selectively send on just one link
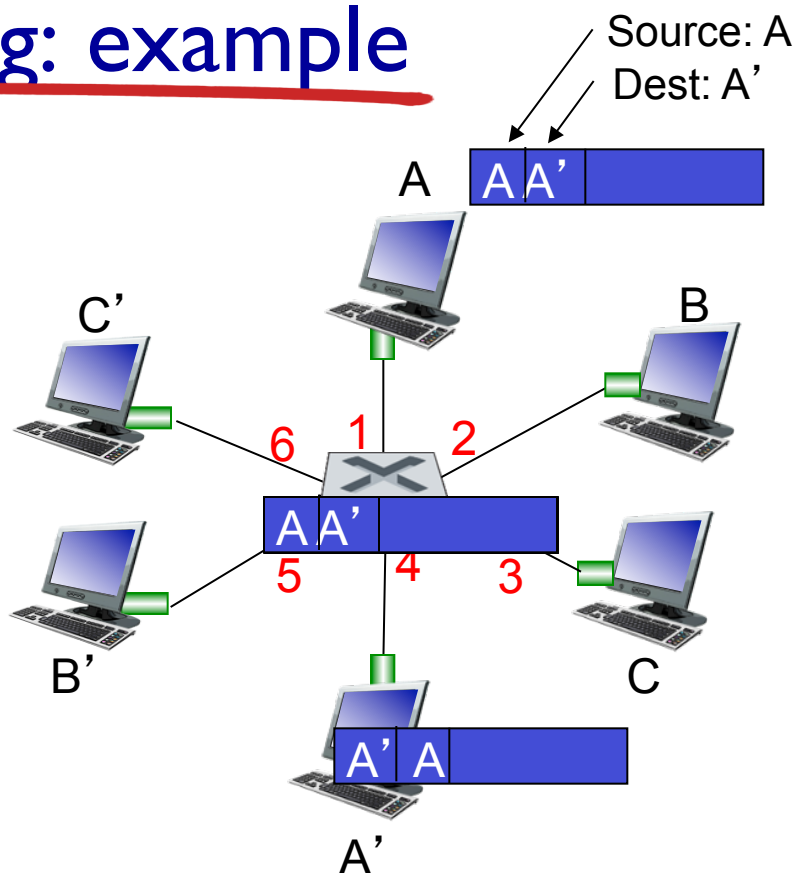
MAC addr	interface	TTL
A	1	60
A'	4	60

switch table (initially empty)

Interconnecting switches

self-learning switches can be connected together:
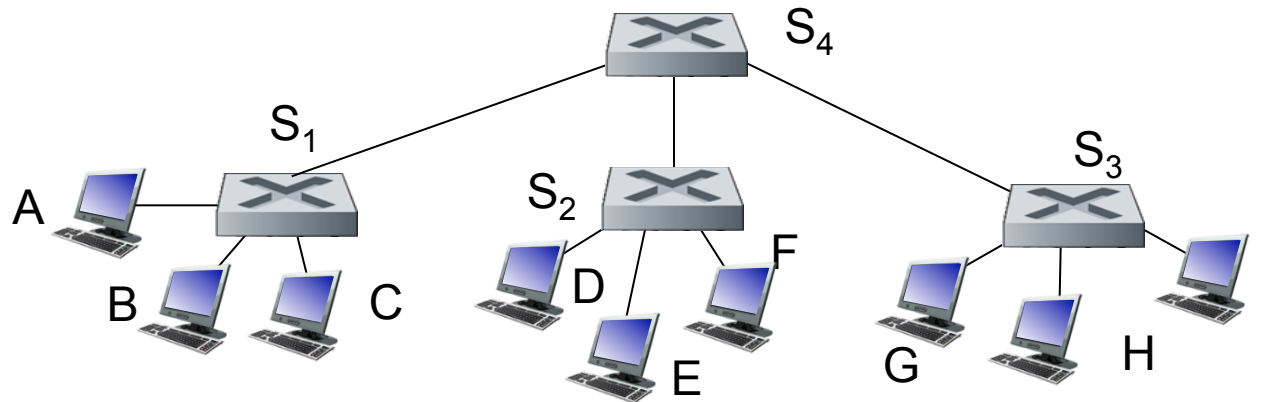
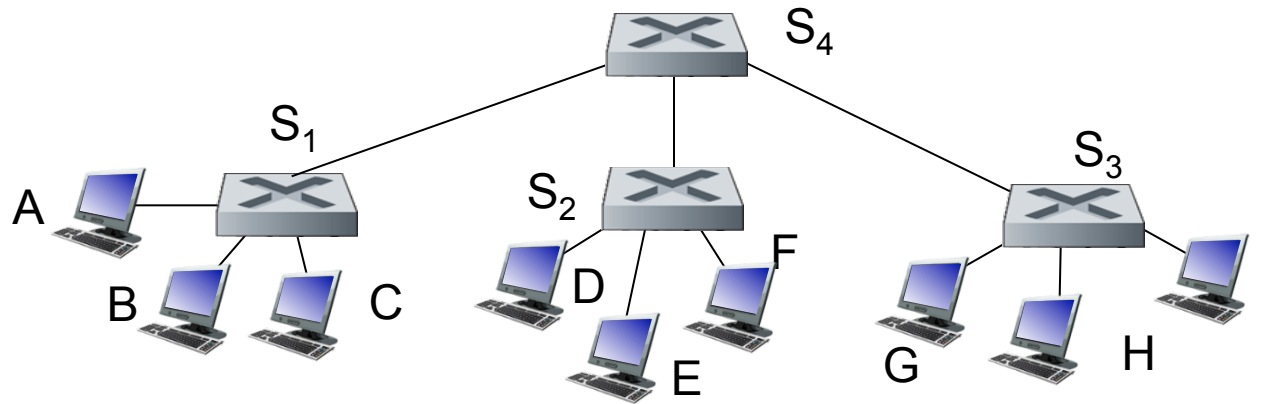*Q:* sending from A to G - how does S_1 know to forward frame destined to G via S_4 and S_3?

- *A:* self learning! (works exactly the same as in single-switch case!)

Self-learning multi-switch example

Suppose C sends frame to I, I responds to C

- Q: show switch tables and packet forwarding in S_1, S_2, S_3, S_4

Institutional network

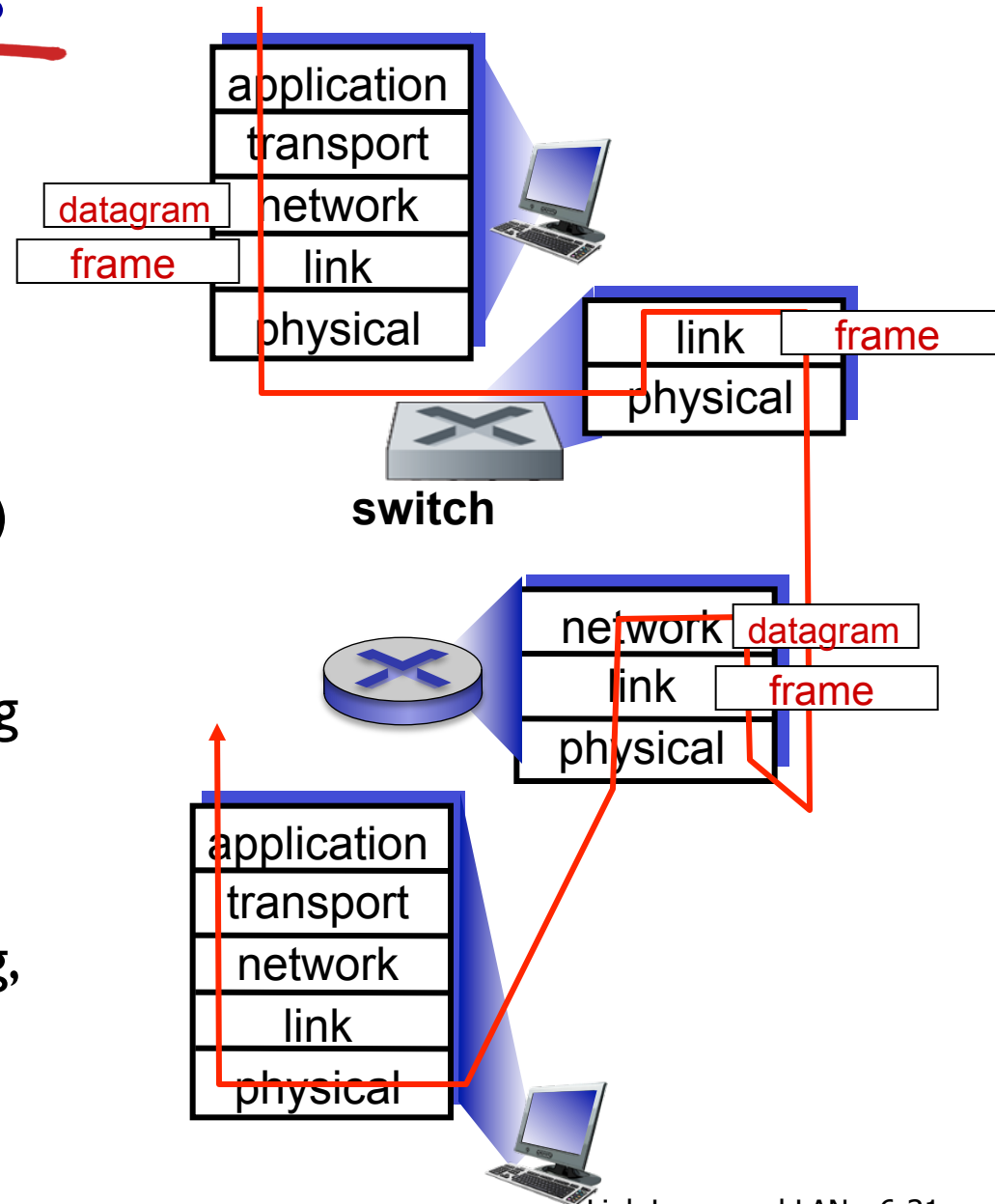to external network

router

mail server

web server

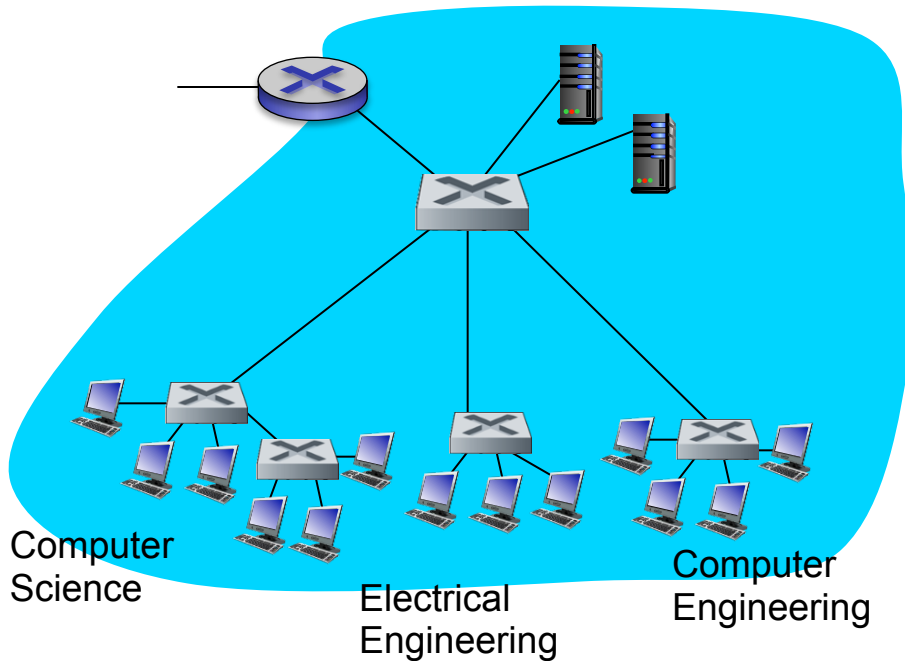*IP subnet*
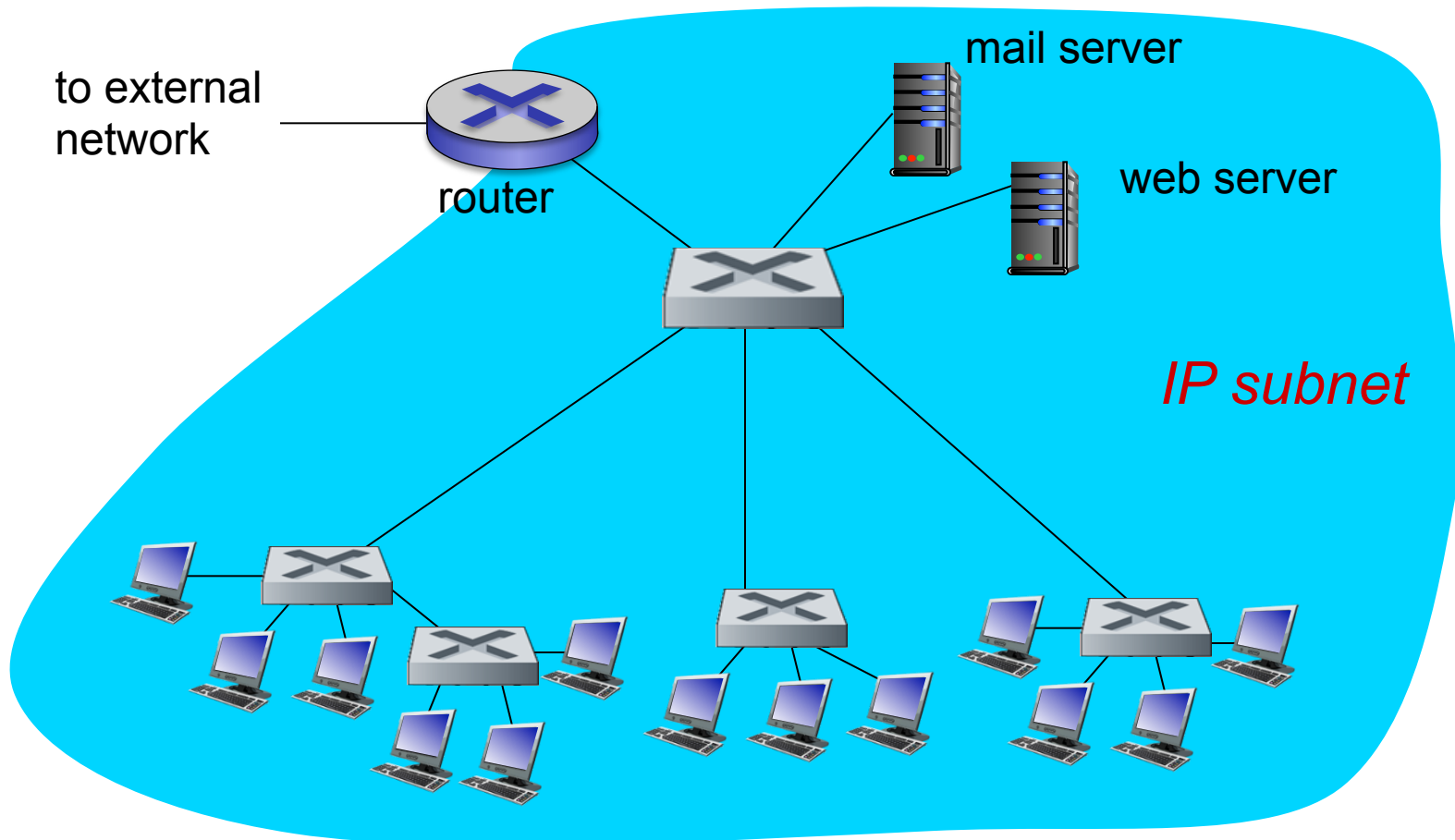
Switches vs. routers

both are store-and-forward:

- *routers:* network-layer devices (examine network-layer headers)

- *switches:* link-layer devices (examine link-layer headers)

both have forwarding tables:

- *routers:* compute tables using routing algorithms, IP addresses

- *switches:* learn forwarding table using flooding, learning, MAC addresses

application
transport
datagram | network
frame | link
physical

link | frame
physical

switch

network | datagram
link | frame
physical

application
transport
network
link
physical

VLANs: motivation

Computer
Science

Electrical
Engineering

Computer
Engineering

consider:

- CS user moves office to EE, but wants connect to CS switch?

- single broadcast domain:
  - all layer-2 broadcast traffic (ARP, DHCP, unknown location of destination MAC address) must cross entire LAN
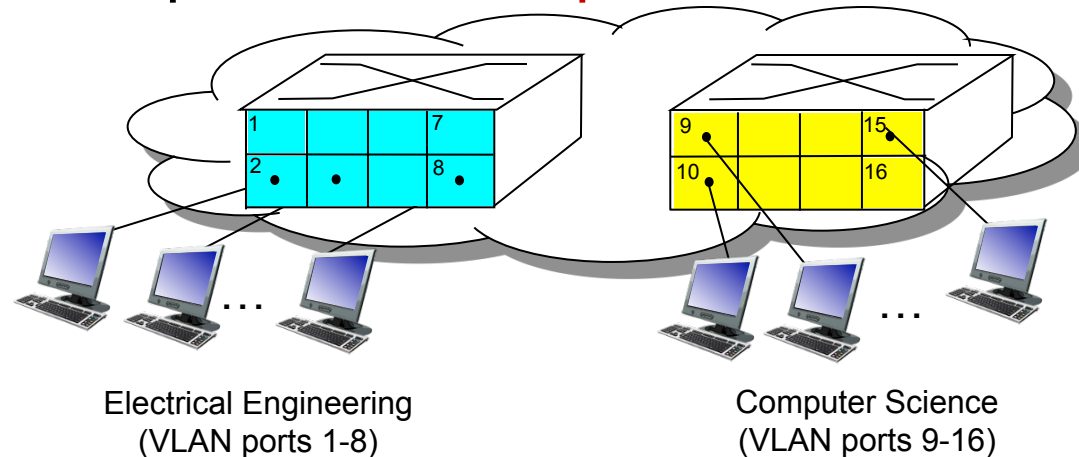 - security/privacy, efficiency issues

VLANs

Virtual Local Area Network

switch(es) supporting VLAN capabilities can be configured to define multiple **_virtual_** LANS over single physical LAN infrastructure.

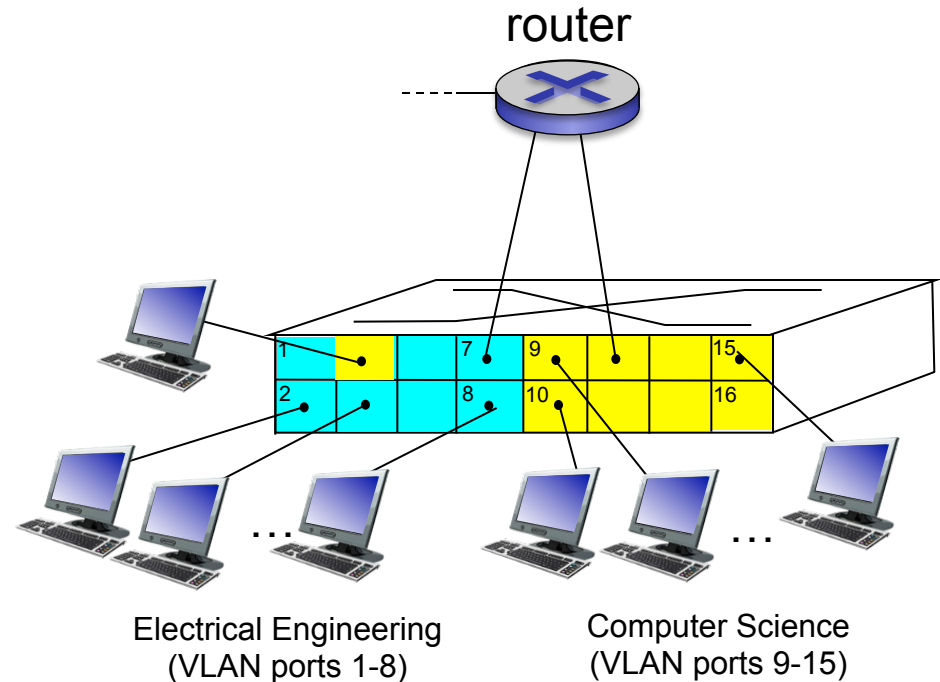**port-based VLAN:** switch ports grouped (by switch management software) so that *single* physical switch ……

Electrical Engineering
(VLAN ports 1-8)

Computer Science
(VLAN ports 9-15)
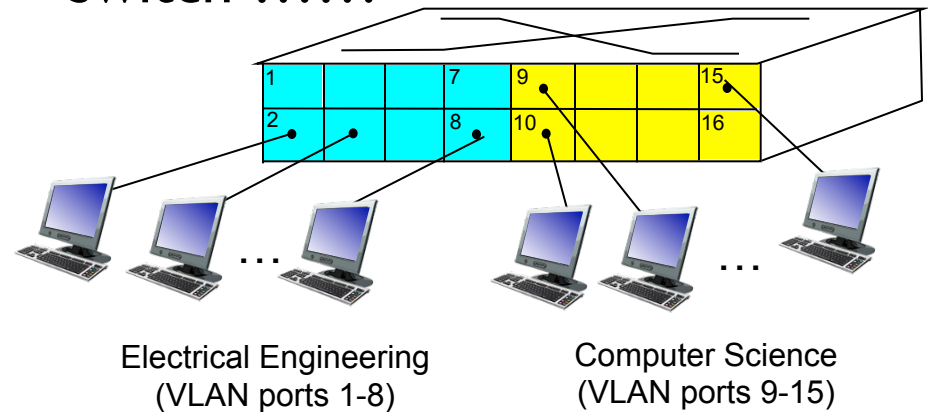
… operates as multiple virtual switches

Electrical Engineering
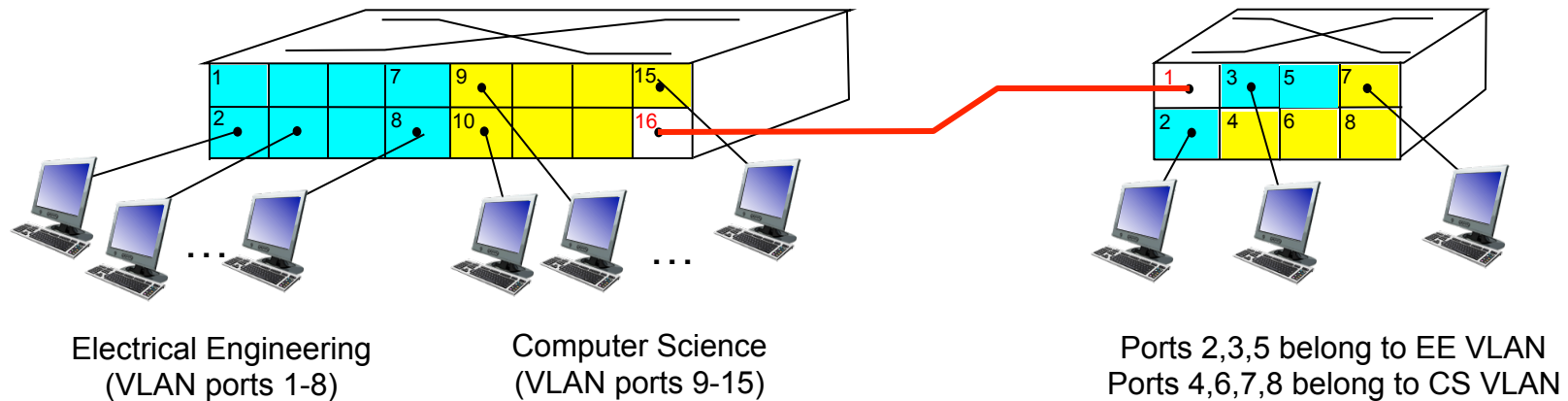(VLAN ports 1-8)

Computer Science
(VLAN ports 9-16)

Port-based VLAN

- *traffic isolation:* frames to/from ports 1-8 can *only* reach ports 1-8
  - can also define VLAN based on MAC addresses of endpoints, rather than switch port

- dynamic membership: ports can be dynamically assigned among VLANs

- forwarding between VLANS: done via routing (just as with separate switches)
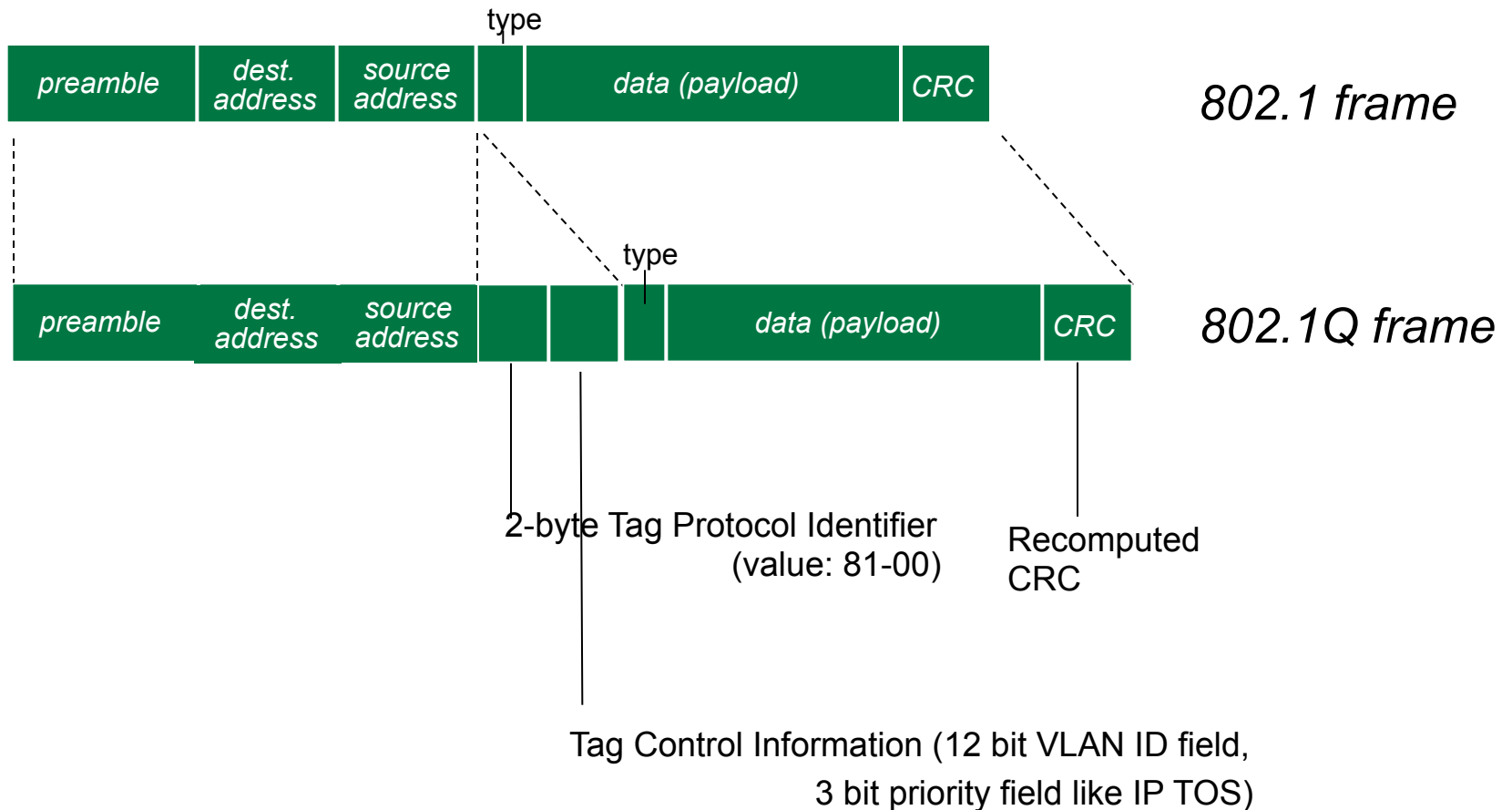 - in practice vendors sell combined switches plus routers

router

Electrical Engineering
(VLAN ports 1-8)

Computer Science
(VLAN ports 9-15)

VLANS spanning multiple switches

Electrical Engineering
(VLAN ports 1-8)

Computer Science
(VLAN ports 9-15)

Ports 2,3,5 belong to EE VLAN
Ports 4,6,7,8 belong to CS VLAN

- *trunk port:* carries frames between VLANS defined over multiple physical switches
  - frames forwarded within VLAN between switches can't be vanilla 802.1 frames (must carry VLAN ID info)
  - 802.1q protocol adds/removed additional header fields for frames forwarded between trunk ports

802.1Q VLAN frame format

type

| preamble | dest. address | source address | | data (payload) | CRC |

802.1 frame

type

| preamble | dest. address | source address | | | | data (payload) | CRC |

802.1Q frame

2-byte Tag Protocol Identifier (value: 81-00)

Recomputed CRC

Tag Control Information (12 bit VLAN ID field, 3 bit priority field like IP TOS)

Link layer, LANs: outline

6.1 introduction, services

6.2 error detection, correction

6.3 multiple access protocols

6.4 LANs
- addressing, ARP
- Ethernet
- switches
- VLANS

6.5 link virtualization: MPLS

6.6 data center networking

6.7 a day in the life of a web request

Multiprotocol label switching (MPLS)

- initial goal: high-speed IP forwarding using fixed length label (instead of IP address)
  - fast lookup using fixed length identifier (rather than shortest prefix matching)
  - borrowing ideas from Virtual Circuit (VC) approach
  - but IP datagram still keeps IP address!

PPP or Ethernet header	**MPLS header**	IP header	remainder of link-layer frame

label		Exp	S	TTL
20		3	1	5
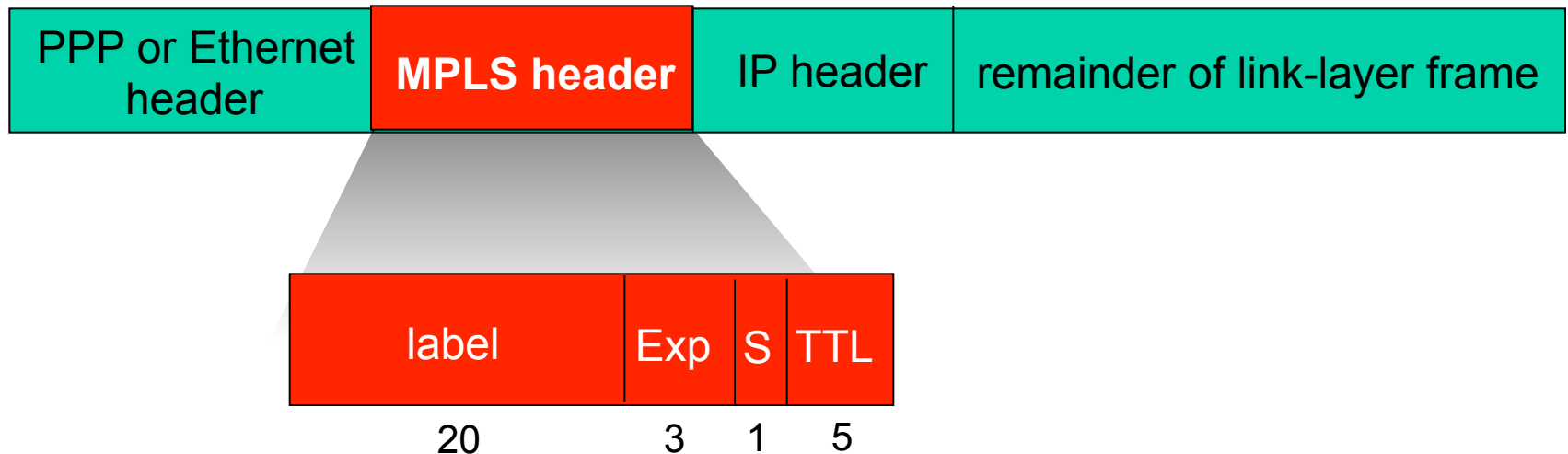
MPLS capable routers

- a.k.a. label-switched router
- forward packets to outgoing interface based only on label value (*don't inspect IP address*)
  - MPLS forwarding table distinct from IP forwarding tables
- *flexibility:* MPLS forwarding decisions can *differ* from those of IP
  - use destination *and* source addresses to route flows to same destination differently (traffic engineering)
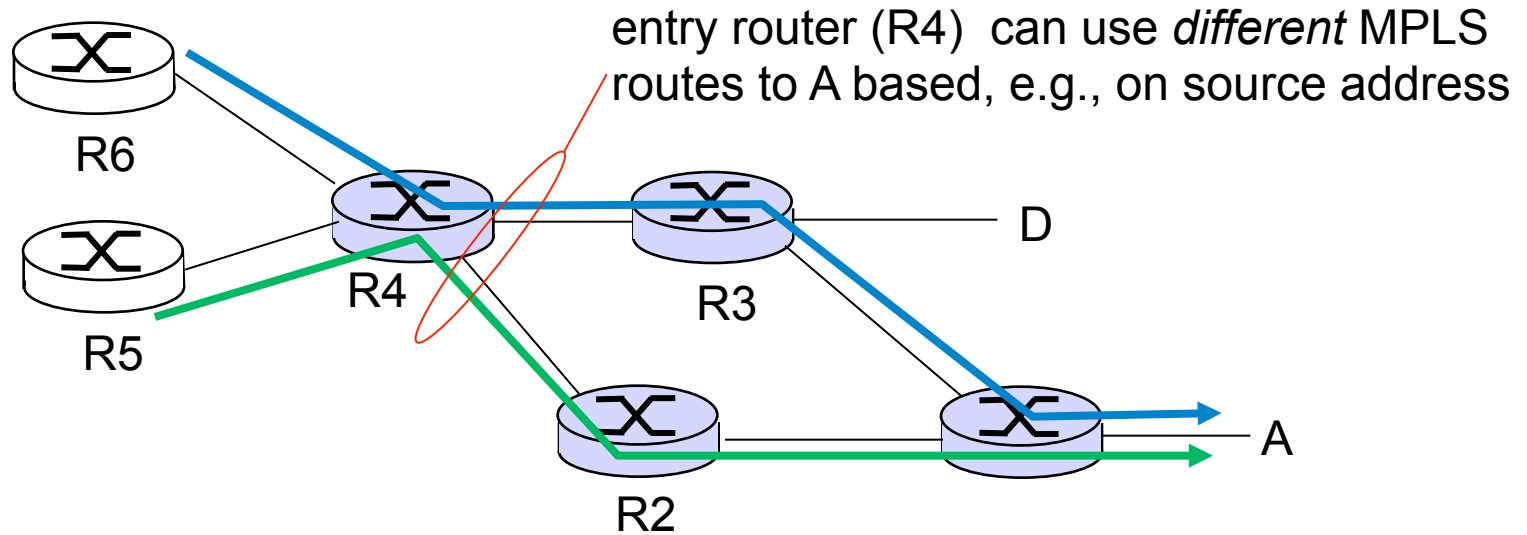 - re-route flows quickly if link fails: pre-computed backup paths (useful for VoIP)
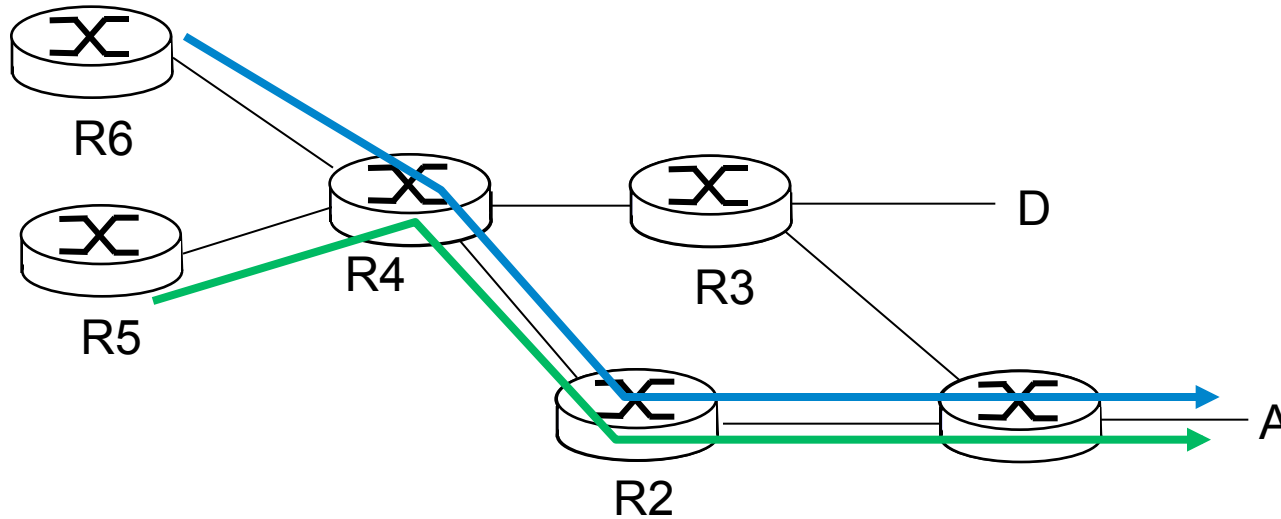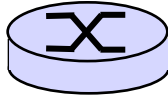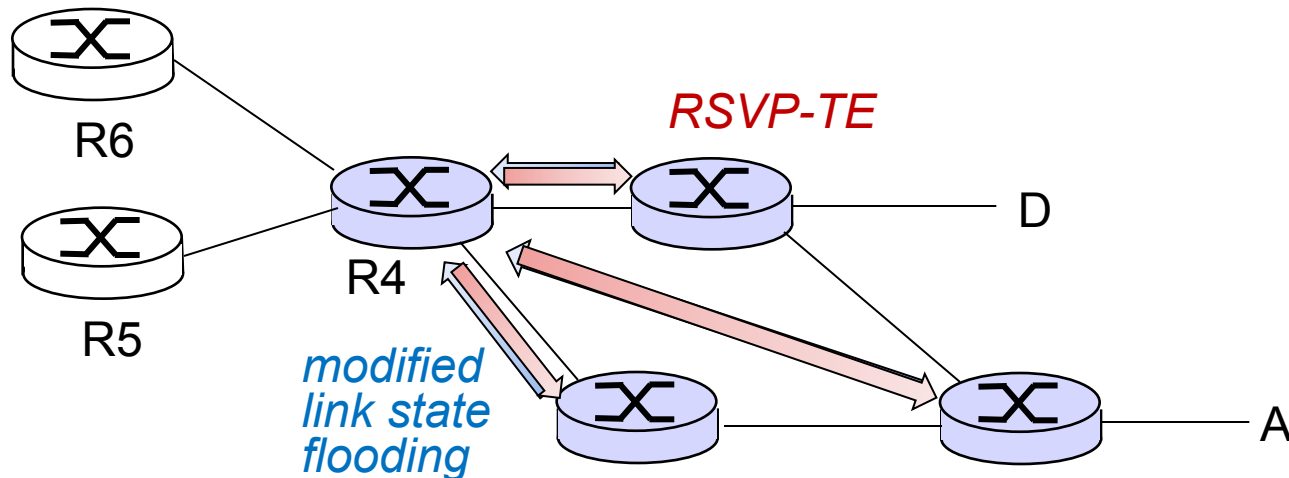
MPLS versus IP paths

- *IP routing: path to destination determined by destination address alone*

IP router

MPLS versus IP paths

entry router (R4) can use *different* MPLS routes to A based, e.g., on source address
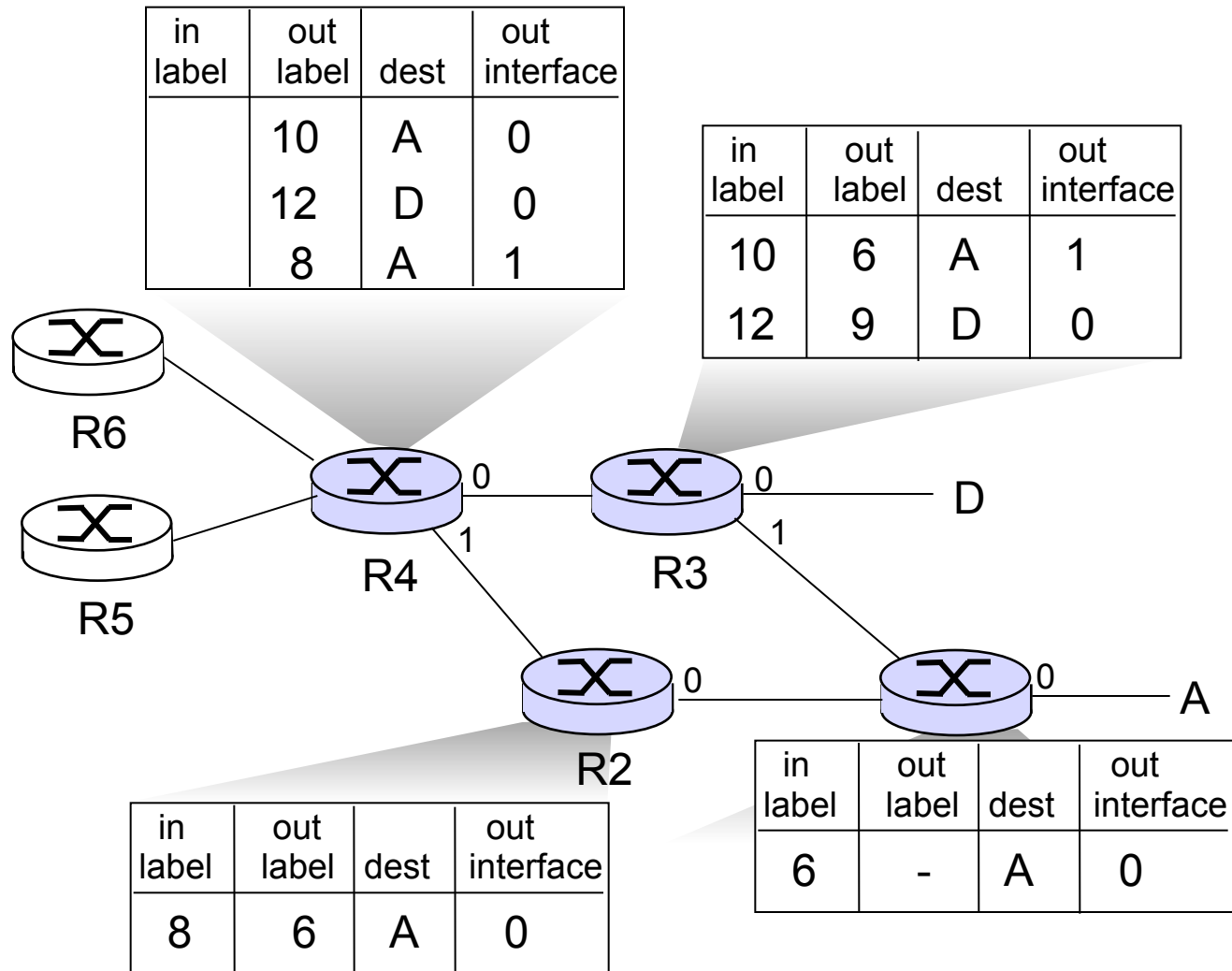
R6

R5

R4

R3

R2

D

A

- *IP routing:* *path to destination determined by destination address alone*

  IP-only router

- *MPLS routing:* path to destination can be based on source *and* destination address

  MPLS and IP router

  - *fast reroute:* precompute backup routes in case of link failure

MPLS signaling

- modify OSPF, IS-IS link-state flooding protocols to carry info used by MPLS routing,
  - e.g., link bandwidth, amount of "reserved" link bandwidth
- *entry MPLS router uses RSVP-TE signaling protocol to set up MPLS forwarding at downstream routers*

MPLS forwarding tables

in label	out label	dest	out interface
	10	A	0
	12	D	0
	8	A	1

in label	out label	dest	out interface
10	6	A	1
12	9	D	0

R6

R5

R4 0

R3 0 ———— D

1

1

R2 0

A 0 ———— A

in label	out label	dest	out interface
6	-	A	0

in label	out label	dest	out interface
8	6	A	0

Link layer, LANs: outline

Data center networks

- 10's to 100's of thousands of hosts, often closely coupled, in close proximity:
    - e-business (e.g. Amazon)
    - content-servers (e.g., YouTube, Akamai, Apple, Microsoft)
    - search engines, data mining (e.g., Google)

- challenges:
    - multiple applications, each serving massive numbers of clients
    - managing/balancing load, avoiding processing, networking, data bottlenecks
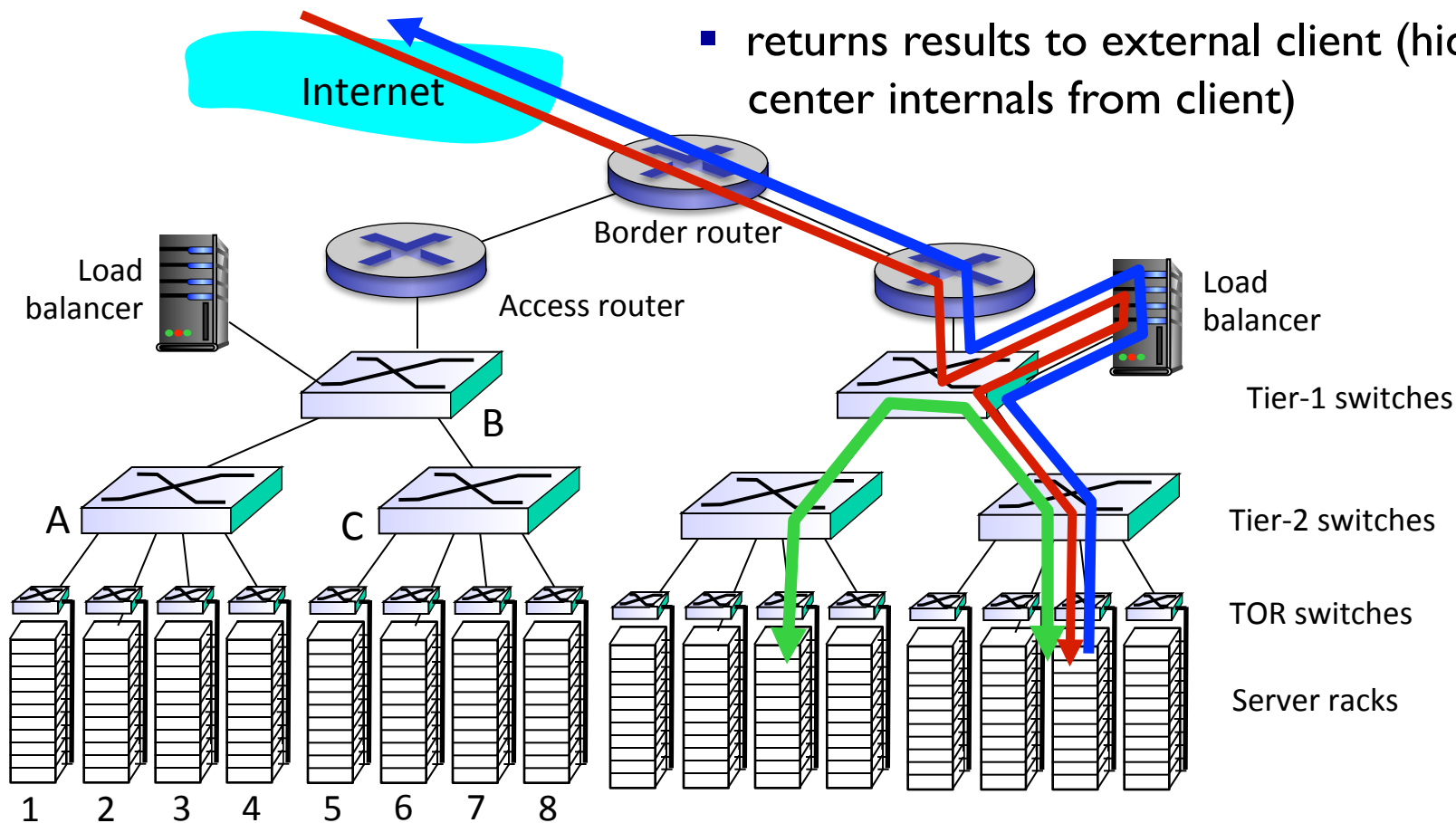

Inside a 40-ft Microsoft container, Chicago data center
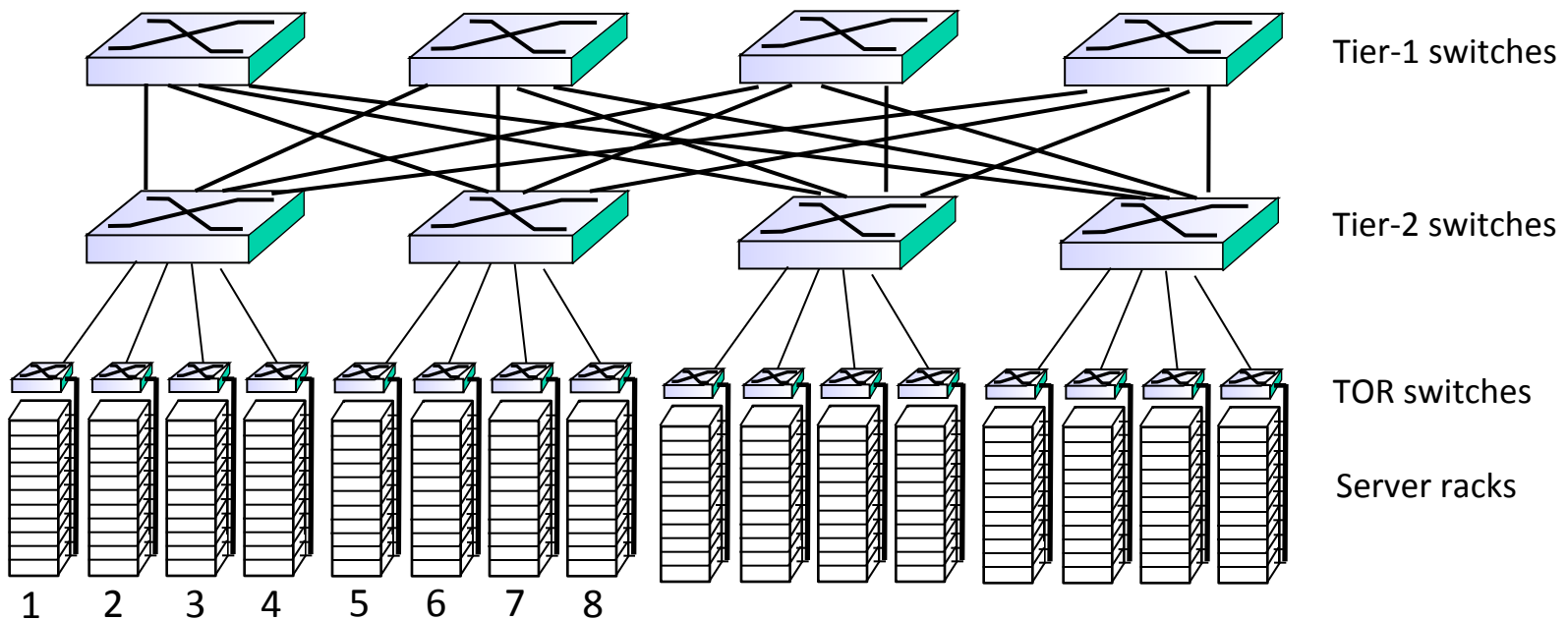
Data center networks

load balancer: application-layer routing
- receives external client requests
- directs workload within data center
- returns results to external client (hiding data center internals from client)

Internet

Border router

Access router

Load balancer

Load balancer

Tier-1 switches

A   B   C   Tier-2 switches

TOR switches

Server racks

1   2   3   4   5   6   7   8

Data center networks

- rich interconnection among switches, racks:
  - increased throughput between racks (multiple routing paths possible)
  - increased reliability via redundancy

Tier-1 switches

Tier-2 switches

TOR switches

Server racks

1 2 3 4 5 6 7 8

Link layer, LANs: outline

6.1 introduction, services

6.2 error detection, correction

6.3 multiple access protocols

64 LANs
- addressing, ARP
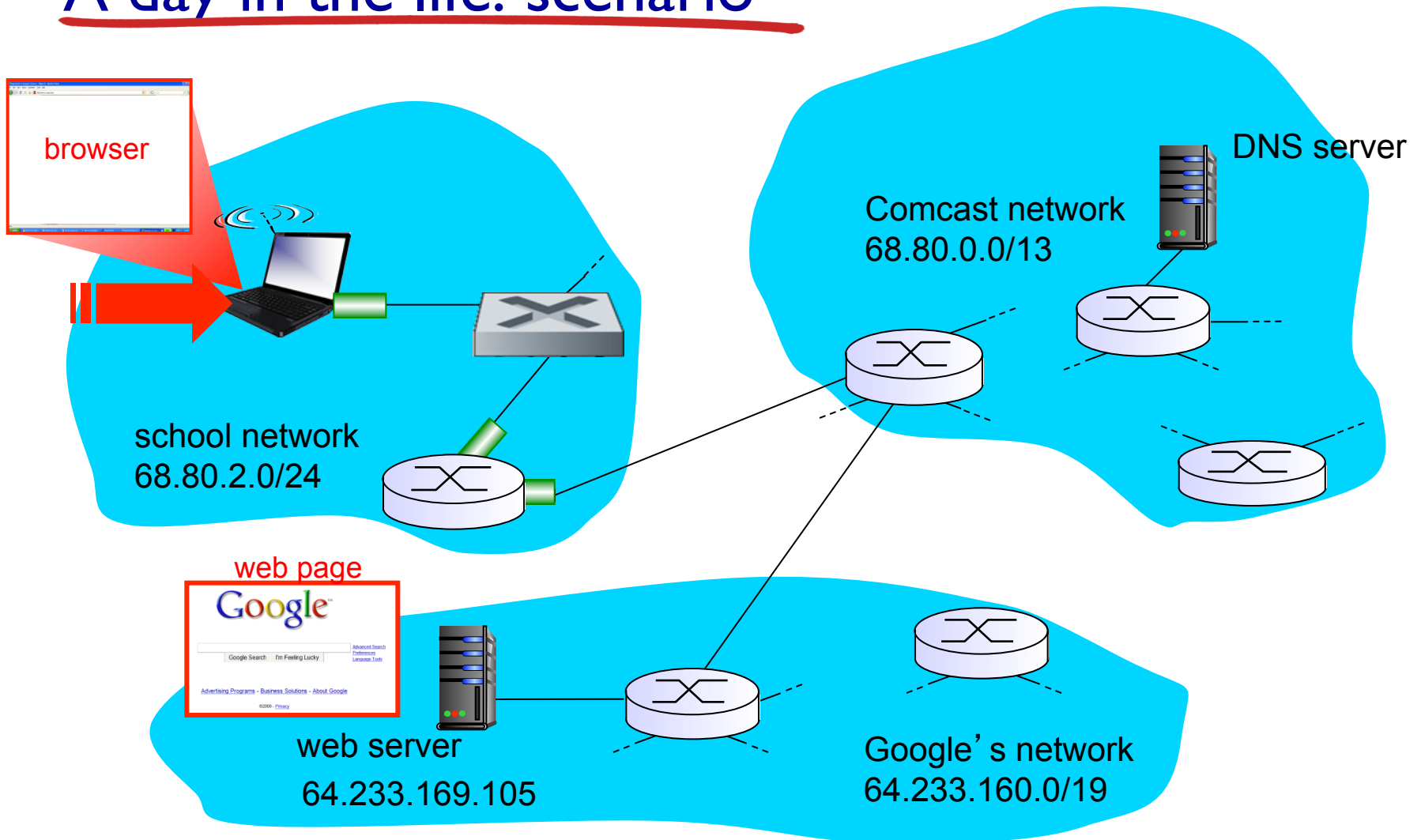- Ethernet
- switches
- VLANS

6.5 link virtualization: MPLS

6.6 data center networking
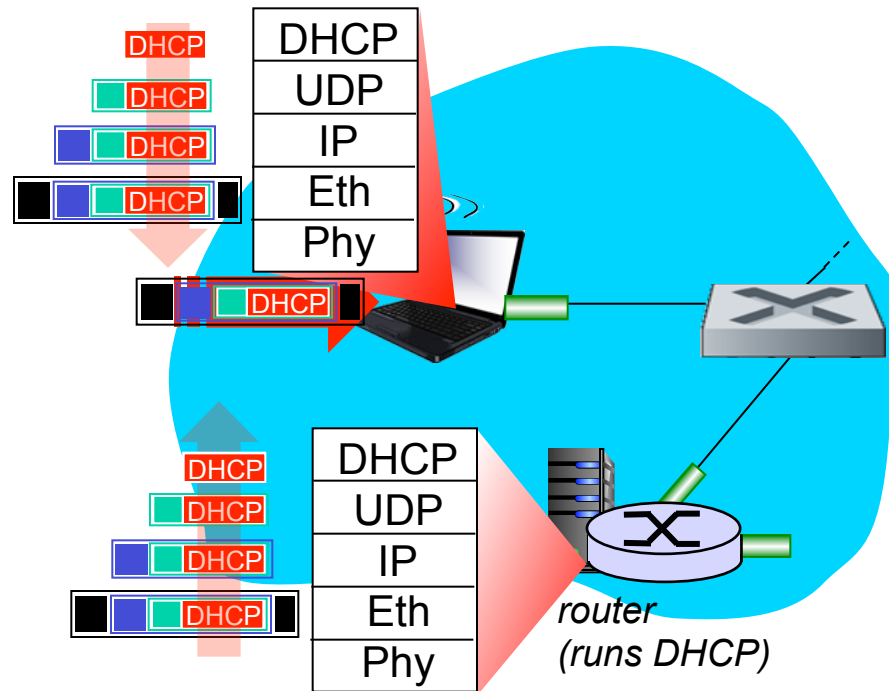
6.7 a day in the life of a web request

Synthesis: a day in the life of a web request

- journey down protocol stack complete!
  - application, transport, network, link
- putting-it-all-together: synthesis!
  - *goal:* identify, review, understand protocols (at all layers) involved in seemingly simple scenario: requesting www page
  - *scenario:* student attaches laptop to campus network, requests/receives www.google.com

A day in the life: scenario

browser

DNS server

Comcast network
68.80.0.0/13

school network
68.80.2.0/24

web page

Google

web server
64.233.169.105

Google's network
64.233.160.0/19

A day in the life… connecting to the Internet

router
(runs DHCP)

- connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use *DHCP*

- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.3 Ethernet

- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server

- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

# A day in the life… connecting to the Internet
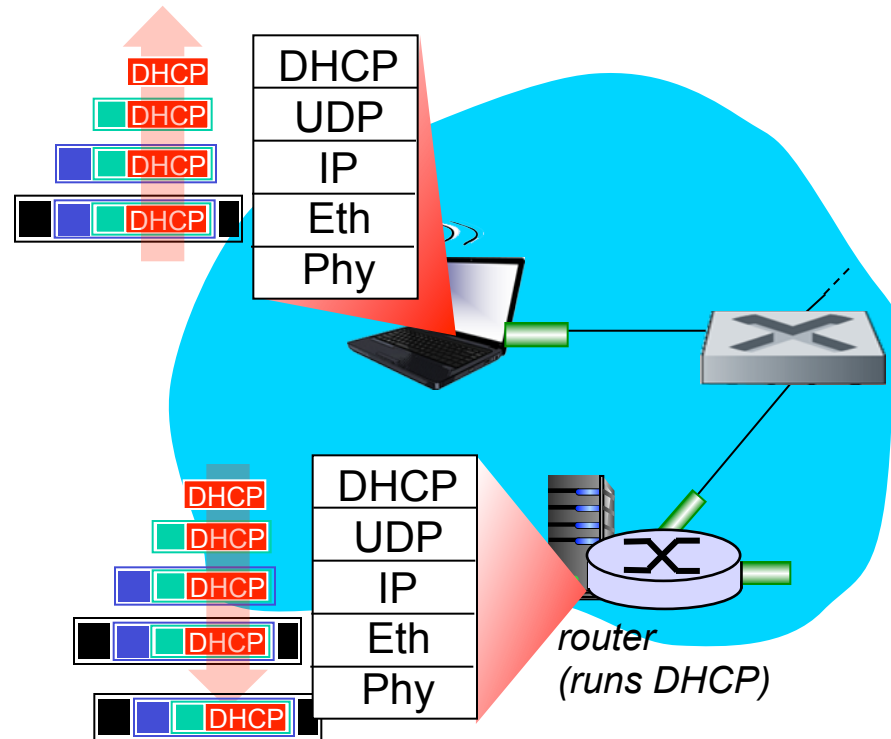
router
(runs DHCP)

- DHCP server formulates *DHCP ACK* containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server

- encapsulation at DHCP server, frame forwarded (switch learning) through LAN, demultiplexing at client

- DHCP client receives DHCP ACK reply

Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

A day in the life… ARP (before DNS, before HTTP)
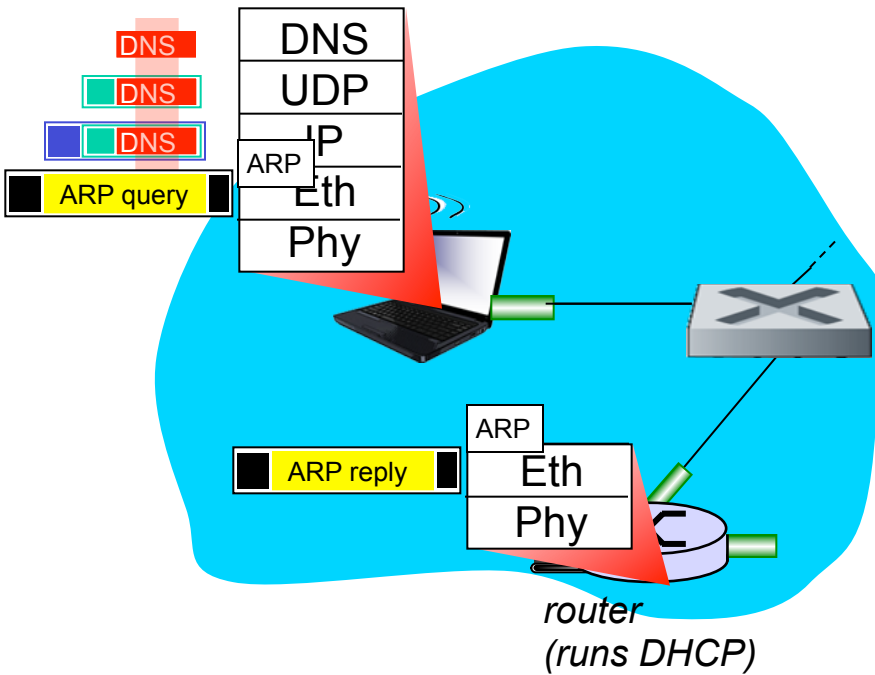
router
(runs DHCP)

- before sending *HTTP* request, need IP address of www.google.com: *DNS*

- DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: ARP

- ARP query broadcast, received by router, which replies with ARP reply giving MAC address of router interface

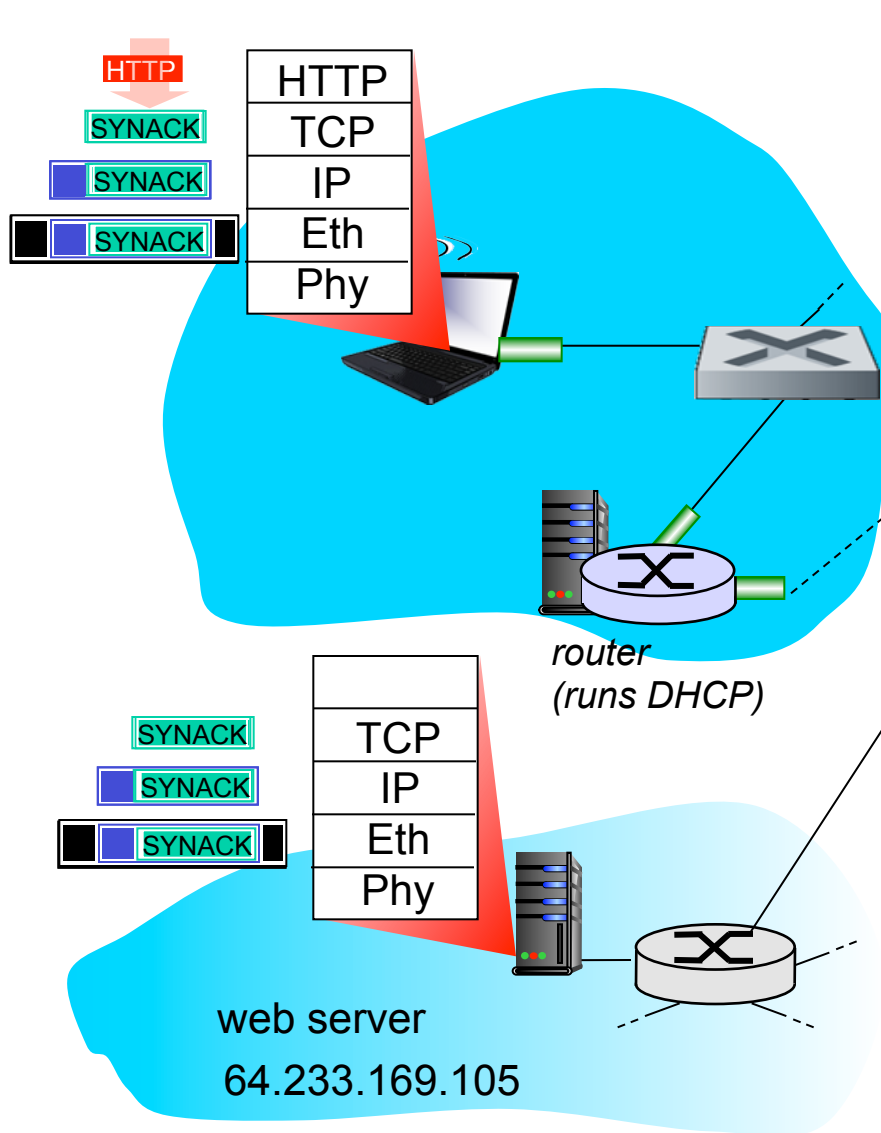- client now knows MAC address of first hop router, so can now send frame containing DNS query

A day in the life… using DNS

router
(runs DHCP)

Comcast network
68.80.0.0/13
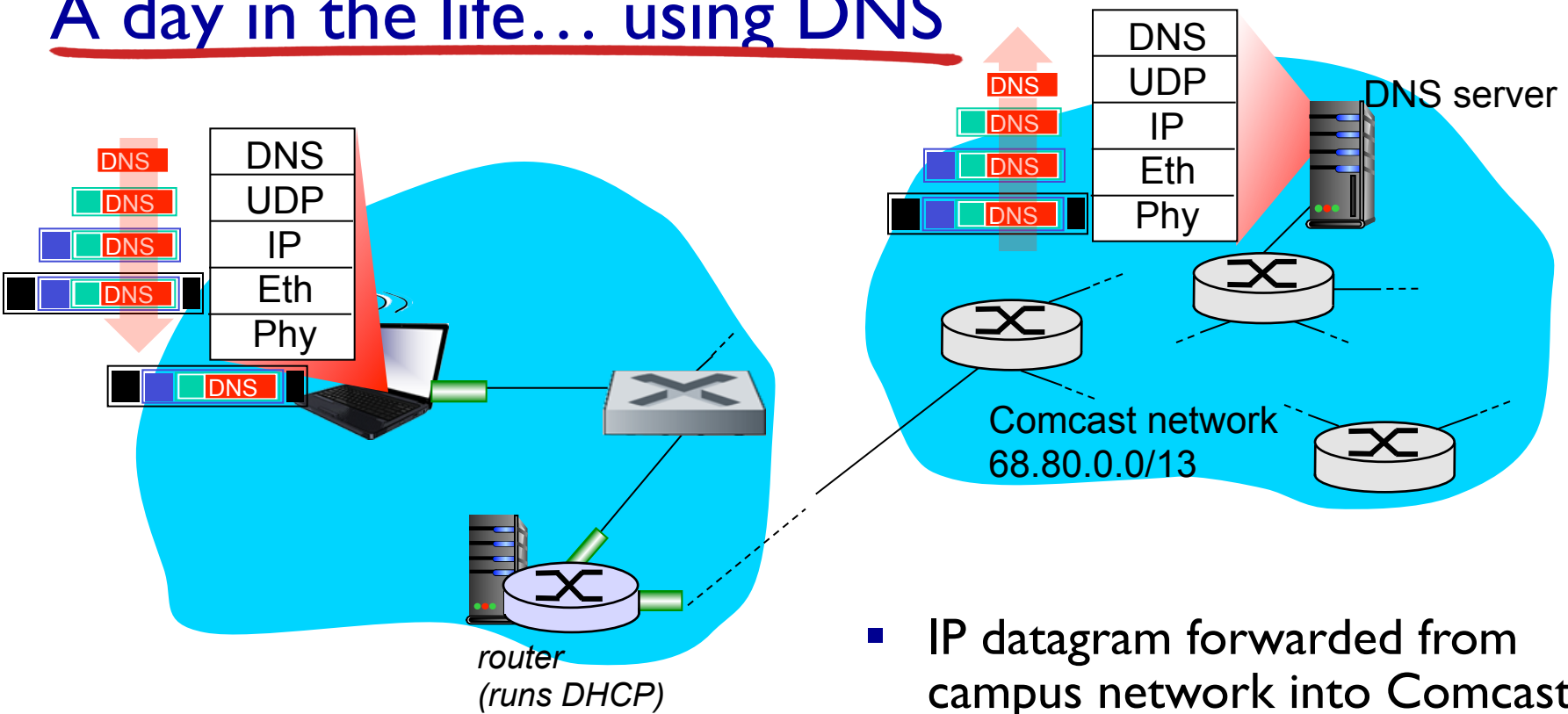
DNS server

- IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router

- IP datagram forwarded from campus network into Comcast network, routed (tables created by RIP, OSPF, IS-IS and/or BGP routing protocols) to DNS server

- demuxed to DNS server

- DNS server replies to client with IP address of www.google.com

A day in the life…TCP connection carrying HTTP

HTTP
SYNACK
SYNACK
SYNACK

HTTP
TCP
IP
Eth
Phy

router (runs DHCP)

SYNACK
SYNACK
SYNACK

TCP
IP
Eth
Phy

web server
64.233.169.105

- to send HTTP request, client first opens TCP socket to web server

- TCP SYN segment (step 1 in 3-way handshake) inter-domain routed to web server

- web server responds with TCP SYNACK (step 2 in 3-way handshake)

- TCP connection established!

A day in the life… HTTP request/reply

- web page finally (!!!) displayed
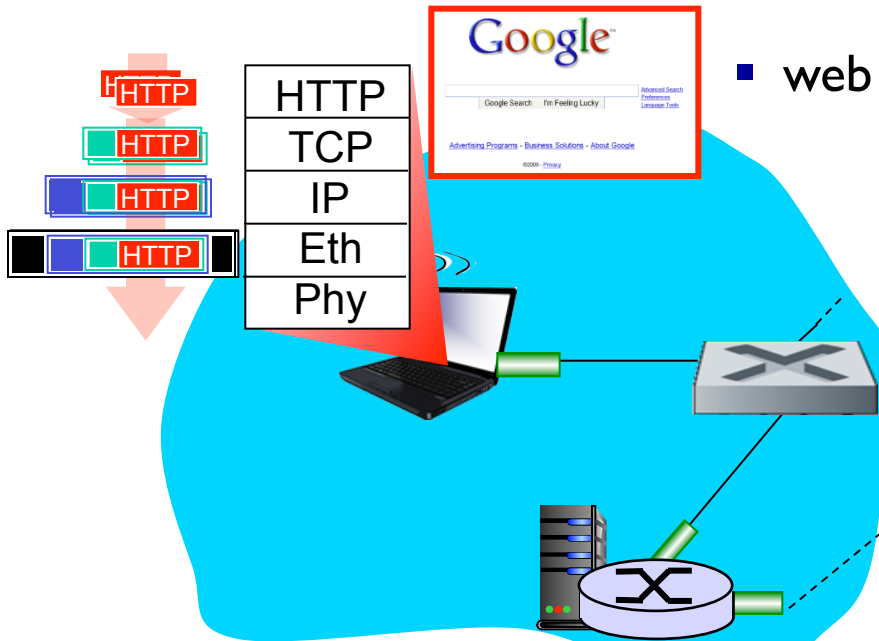
router (runs DHCP)
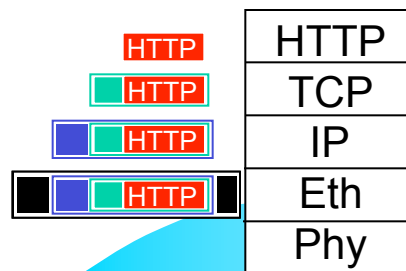
web server
64.233.169.105

- **HTTP request** sent into TCP socket

- IP datagram containing HTTP request routed to www.google.com

- web server responds with **HTTP reply** (containing web page)

- IP datagram containing HTTP reply routed back to client

Chapter 6: Summary

- principles behind data link layer services:
  - error detection, correction
  - sharing a broadcast channel: multiple access
  - link layer addressing
- instantiation and implementation of various link layer technologies
  - Ethernet
  - switched LANS, VLANs
  - virtualized networks as a link layer: MPLS
- synthesis: a day in the life of a web request

Chapter 6: let's take a breath

- journey down protocol stack *complete* (except PHY)

- solid understanding of networking principles, practice

- ….. could stop here …. but *lots* of interesting topics!
  - wireless
  - multimedia
  - security