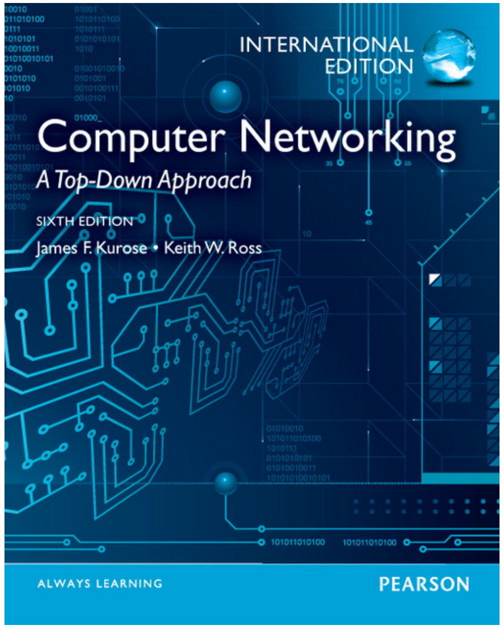
Chapter 5: Link Layer

IK1203

Peter Sjödin, psj@kth.se

Chapter 5
Link Layer



Computer Networking: A Top Down Approach
6th edition
Jim Kurose, Keith Ross
Addison-Wesley
March 2012

A note on the use of these ppt 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

Chapter 5: Link layer

our goals:

- understand principles behind link layer services:
  - error detection and correction
  - sharing a broadcast channel: multiple access
  - link layer addressing
  - local area networks: Ethernet, VLANs
- Instantiation and implementation of various link layer technologies

Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

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

5.5 link virtualization: MPLS

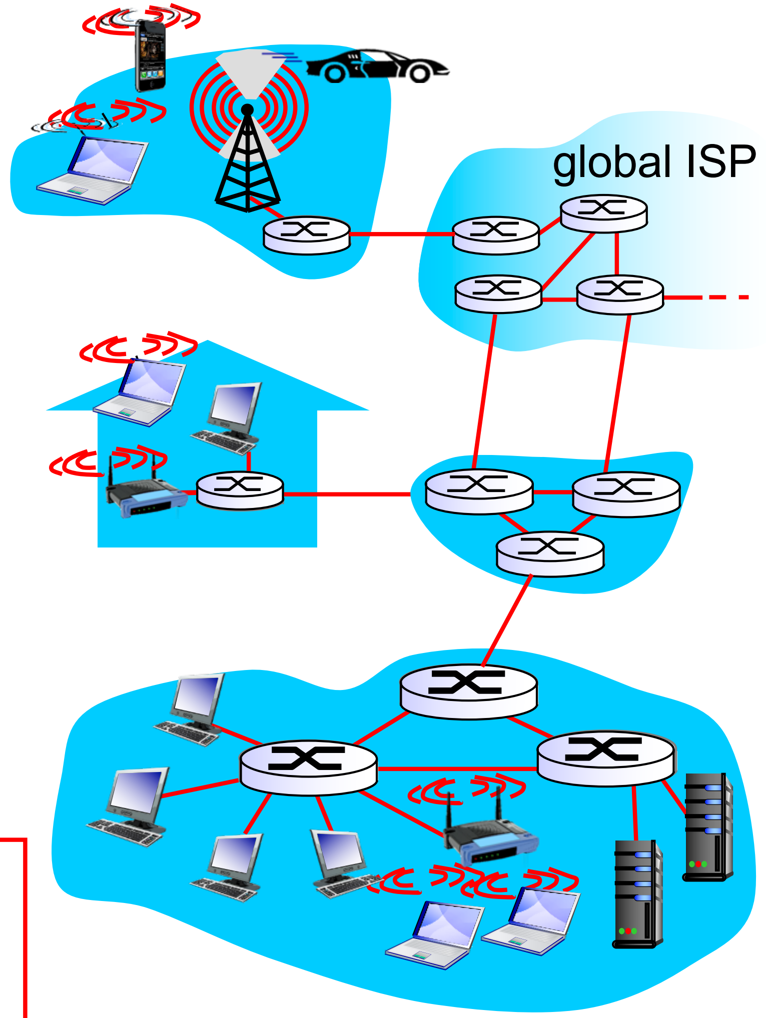
5.6 data center networking

5.7 a day in the life of a web request

Link layer: introduction

terminology:

- hosts and routers: nodes
- communication channels that connect adjacent nodes along communication path: links
  - wired links
  - wireless links
  - LANs
- layer-2 packet: frame, encapsulates datagram



link layer has responsibility of transferring datagram from one node to *physically adjacent* node over a link

Link layer: context

- datagram transferred by different link protocols over different links:
  - e.g., Ethernet on first link, frame relay on intermediate links, 802.11 on last link
- each link protocol provides different services
  - e.g., may or may not provide reliable data transfer over link

transportation analogy:

- trip from Princeton to Lausanne
  - limo: Princeton to JFK
  - plane: JFK to Geneva
  - train: Geneva to Lausanne
- tourist = datagram
- transport segment = communication link
- transportation mode = link layer protocol
- travel agent = routing algorithm

Link layer services

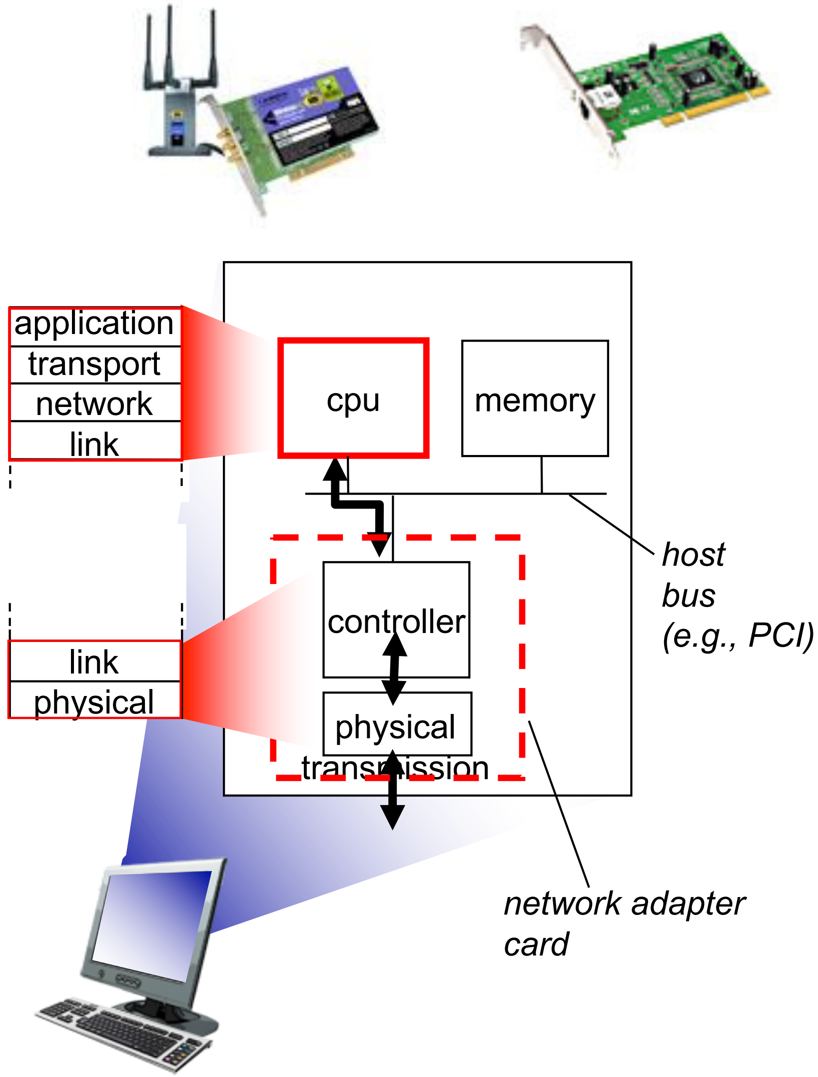
- *framing, link access:*
  - encapsulate datagram into frame, adding header and trailer
  - channel access, if shared medium
  - "MAC" addresses used in frame headers to identify source and destination
    - "Medium Access Control"
    - different from IP addresses!
- *reliable delivery between adjacent nodes*
  - we learned how to do this already (chapter 3)!
  - seldom used on low bit-error link (fiber, some twisted pair)
  - wireless links: high error rates
    - *Q:* why both link-level and end-end reliability?

Link layer services (more)

- *flow control:*
  - pacing between adjacent sending and receiving nodes
- *error detection:*
  - errors caused by signal attenuation and noise.
  - receiver detects presence of errors:
    - signals sender for retransmission or drops frame
- *error correction:*
  - receiver identifies *and corrects* bit error(s) without resorting to retransmission
- *half-duplex and full-duplex*
  - with half duplex, nodes at both ends of link can transmit, but not at same time

Where is the link layer implemented?

- in each and every host
- link layer implemented in "adaptor" (aka *network interface card* NIC) or on a chip
  - Ethernet card, 802.11 card; Ethernet chipset
  - implements link and physical layer
- attaches into host's system buses
- combination of hardware, software, and firmware



Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

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

5.5 link virtualization: MPLS

5.6 data center networking

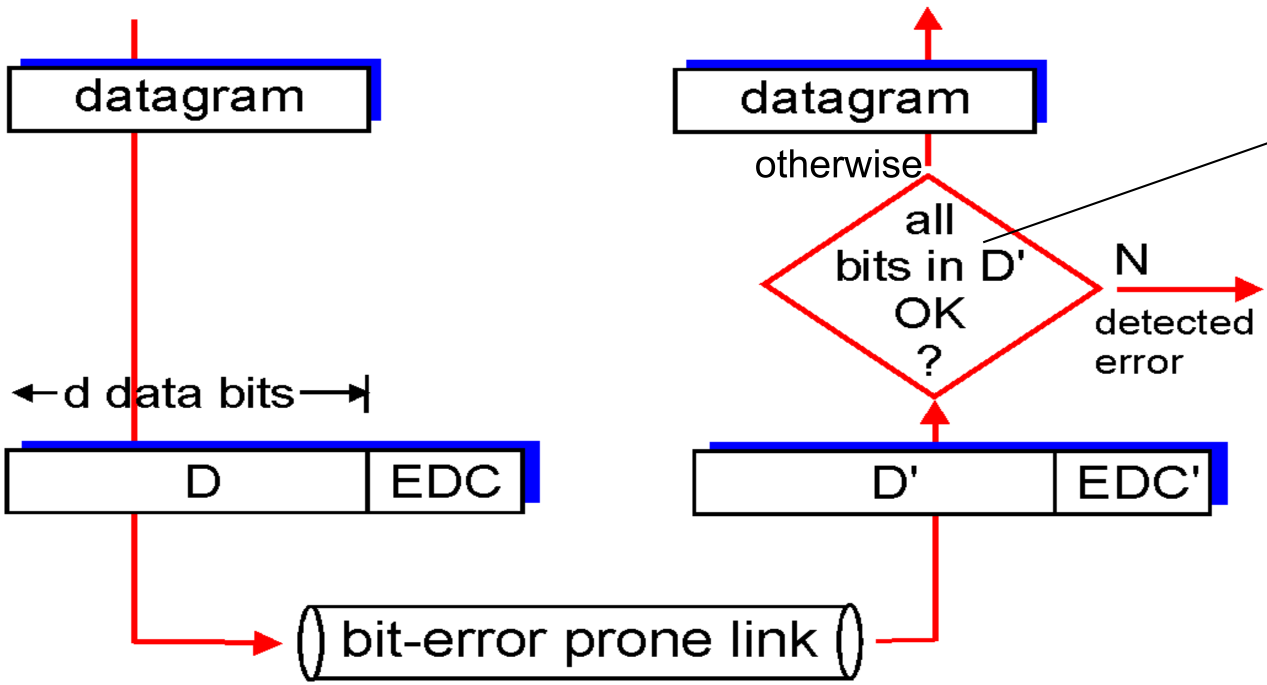
5.7 a day in the life of a web request

Error detection

- Receiver cannot tell by just looking at the data whether data has been corrupted
    - Need to transfer more information to help with this – redundancy

D: Data protected by error detection
E(): Error detection function
EDC: Error Detection and Correction bits, sent together with data, calculated as: EDC = *E*(D)

Check if EDC' = *E*(D')?
If not, it means error:
D' ≠ D, or EDC' ≠ EDC.



Error detection functions

- Error detection not 100% reliable!
  - protocol may miss some errors, but rarely
  - larger EDC field yields better error detection and correction

Name	EDC Length	Description	Examples
Even parity	1 bit	Set so total number of one's is even. Detects single bit errors.	Memory, RAID disks
Internet checksum	16 bits	16-bit sum (modular arithmetic)	IP, TCP, UDP
CRC-32	32 bits	Cyclic code (polynomial division). Detects all burst errors up to 32 bits long (and some longer).	Ethernet

Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

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

5.5 link virtualization: MPLS

5.6 data center networking

5.7 a day in the life of a web request

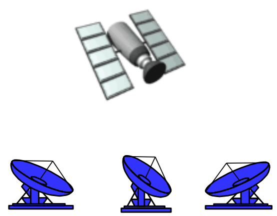
Multiple access links, protocols

two types of "links":

- point-to-point
  - PPP for dial-up access
  - point-to-point link between Ethernet switch and host
- *broadcast (shared wire or medium)*
  - old-fashioned Ethernet
  - upstream HFC
  - 802.11 wireless LAN

shared wire (e.g., cabled Ethernet)

shared RF (e.g., 802.11 WiFi)



shared RF (satellite)

humans at a cocktail party (shared air, acoustical)

Multiple access protocols

- single shared broadcast channel
- two or more simultaneous transmissions by nodes: interference
  - *collision* if a node receives two or more signals at the same time

multiple access protocol

- distributed algorithm that determines how nodes share channel, i.e., determines when a node can transmit

An ideal multiple access protocol

given: broadcast channel of rate R bit/s

desiderata:

1. when one node wants to transmit, it can send at rate R.
2. when M nodes want to transmit, each can send at average rate R/M
3. fully decentralized:
   - no special node to coordinate transmissions
   - no synchronization of clocks
4. simple

MAC protocols: taxonomy

three broad classes:

- *channel partitioning*
  - divide channel into smaller ”pieces” (time slots, frequency, code)
  - allocate piece to node for exclusive use
  - TDMA (Time Division Multiple Access), FDMA, CDMA
- *random access*
  - channel not divided
    - collisions could happen
    - ”recover” from collisions
  - Ethernet, IEEE 802.11
- *”taking turns”*
  - nodes take turns
    - but nodes with more to send can take longer turns
  - coordinated access
  - Bluetooth, token ring, IEEE 802.11 PCF (Point Coordination Function)

Random access protocols

- when node has packet to send
  - transmit at full channel data rate R.
  - no *a priori* coordination among nodes
- two or more transmitting nodes ➜ collision
- random access MAC protocol specifies:
  - how to detect collisions
  - how to recover from collisions (e.g., via delayed retransmissions)
- examples of random access MAC protocols:
  - slotted ALOHA
  - ALOHA
  - CSMA, CSMA/CD, CSMA/CA

CSMA (carrier sense multiple access)

CSMA: sense (listen) before transmit:

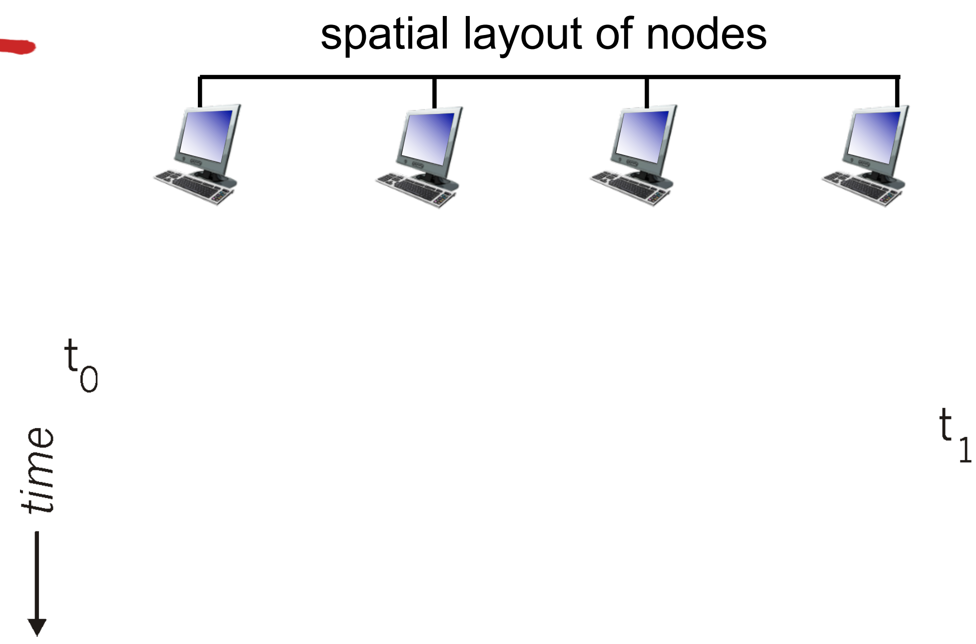
if channel sensed idle: transmit entire frame

- if channel sensed busy, defer transmission

- human analogy: don't interrupt others!

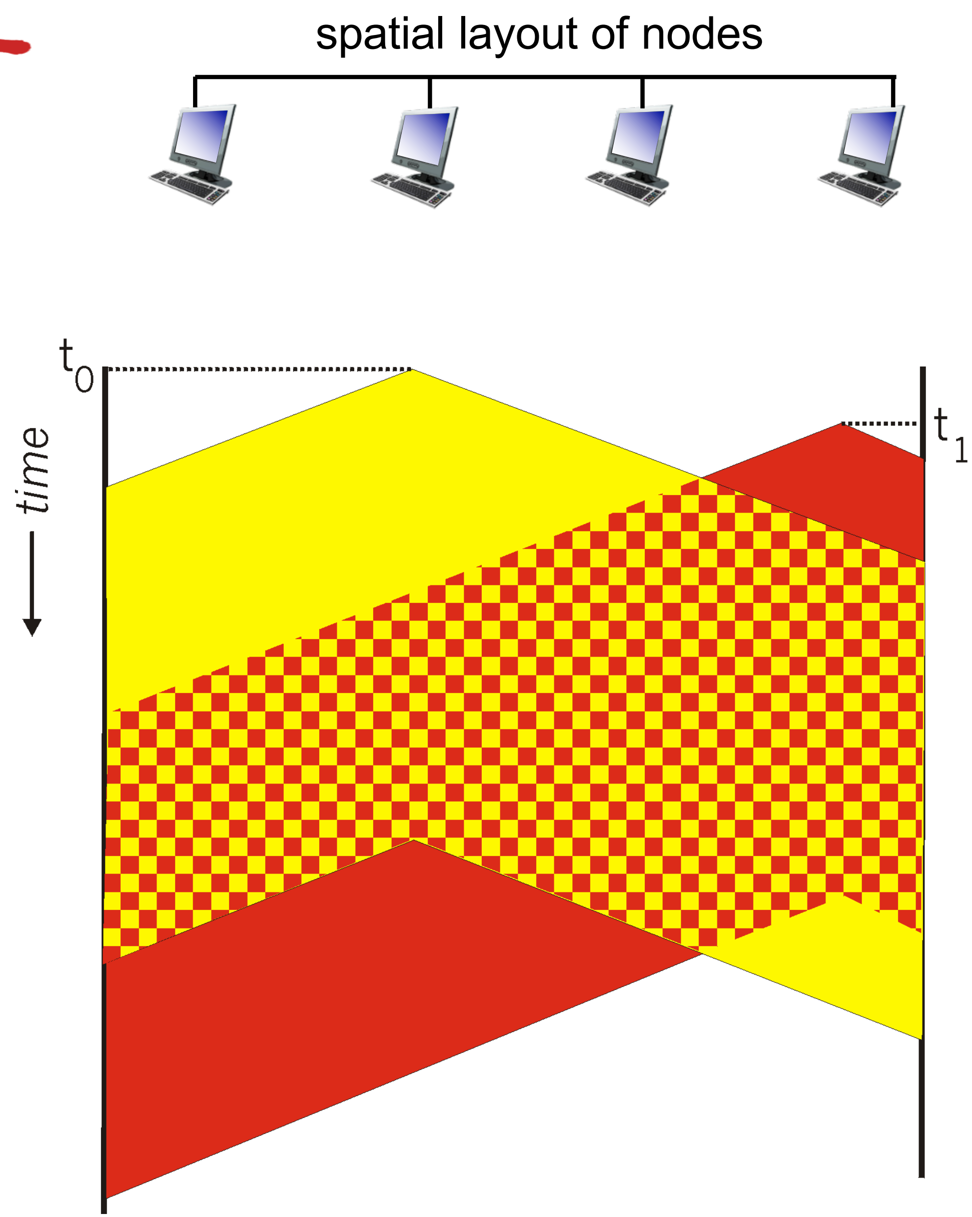
CSMA collisions

- collisions *can* still occur: propagation delay means two nodes may not hear each other's transmission
- collision: entire packet transmission time wasted
  - distance & propagation delay play role in in determining collision probability



CSMA collisions

- collisions *can* still occur: propagation delay means two nodes may not hear each other's transmission
- collision: entire packet transmission time wasted
  - distance & propagation delay play role in in determining collision probability

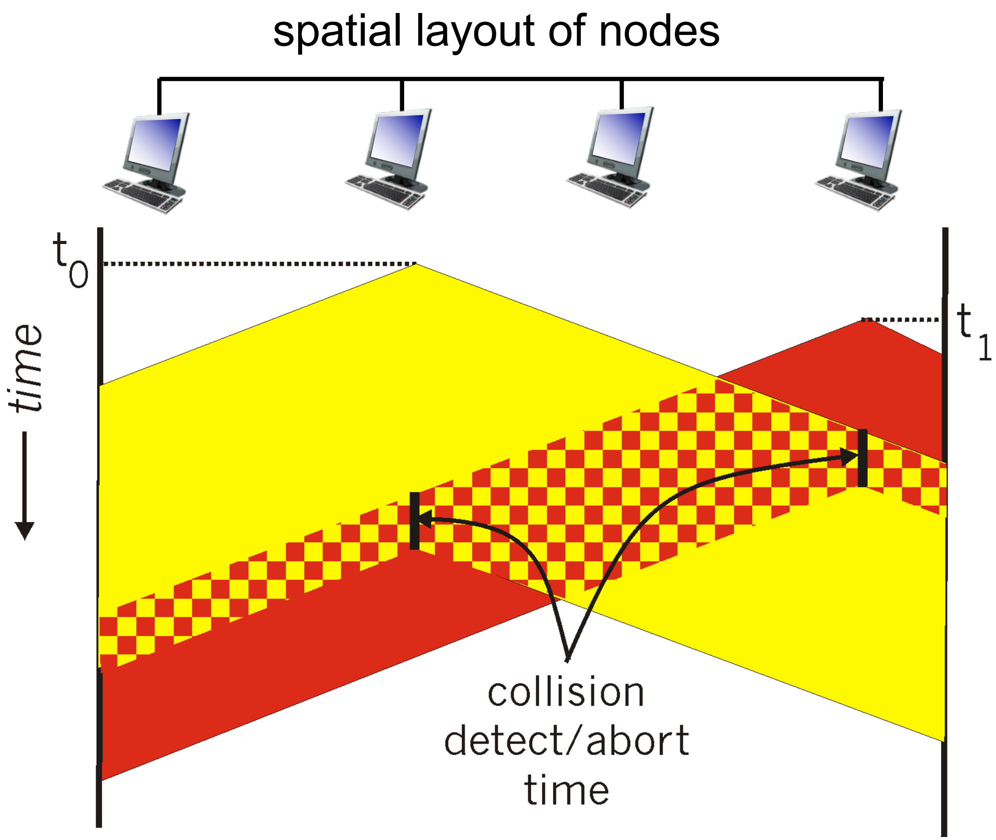


CSMA/CD (collision detection)

CSMA/CD: carrier sensing, deferral as in CSMA
- collisions *detected* within short time
- colliding transmissions aborted, reducing channel wastage

- collision detection:
  - easy in wired LANs: measure signal strengths, compare transmitted and received signals
  - difficult in wireless LANs: received signal strength overwhelmed by local transmission strength
- human analogy: the polite conversationalist

CSMA/CD (collision detection)



Summary of MAC protocols

- *channel partitioning,* by time, frequency or code
  - Time Division, Frequency Division
- *random access* (dynamic),
  - ALOHA, S-ALOHA, CSMA, CSMA/CD
  - carrier sensing: easy in some technologies (wire), hard in others (wireless)
  - CSMA/CD used in Ethernet
  - CSMA/CA used in 802.11
- *taking turns*
  - polling from central site, token passing
  - bluetooth, FDDI, 802.11 with PCF (Point Coordination Function), token ring

Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

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

5.5 link virtualization: MPLS

5.6 data center networking

5.7 a day in the life of a web request

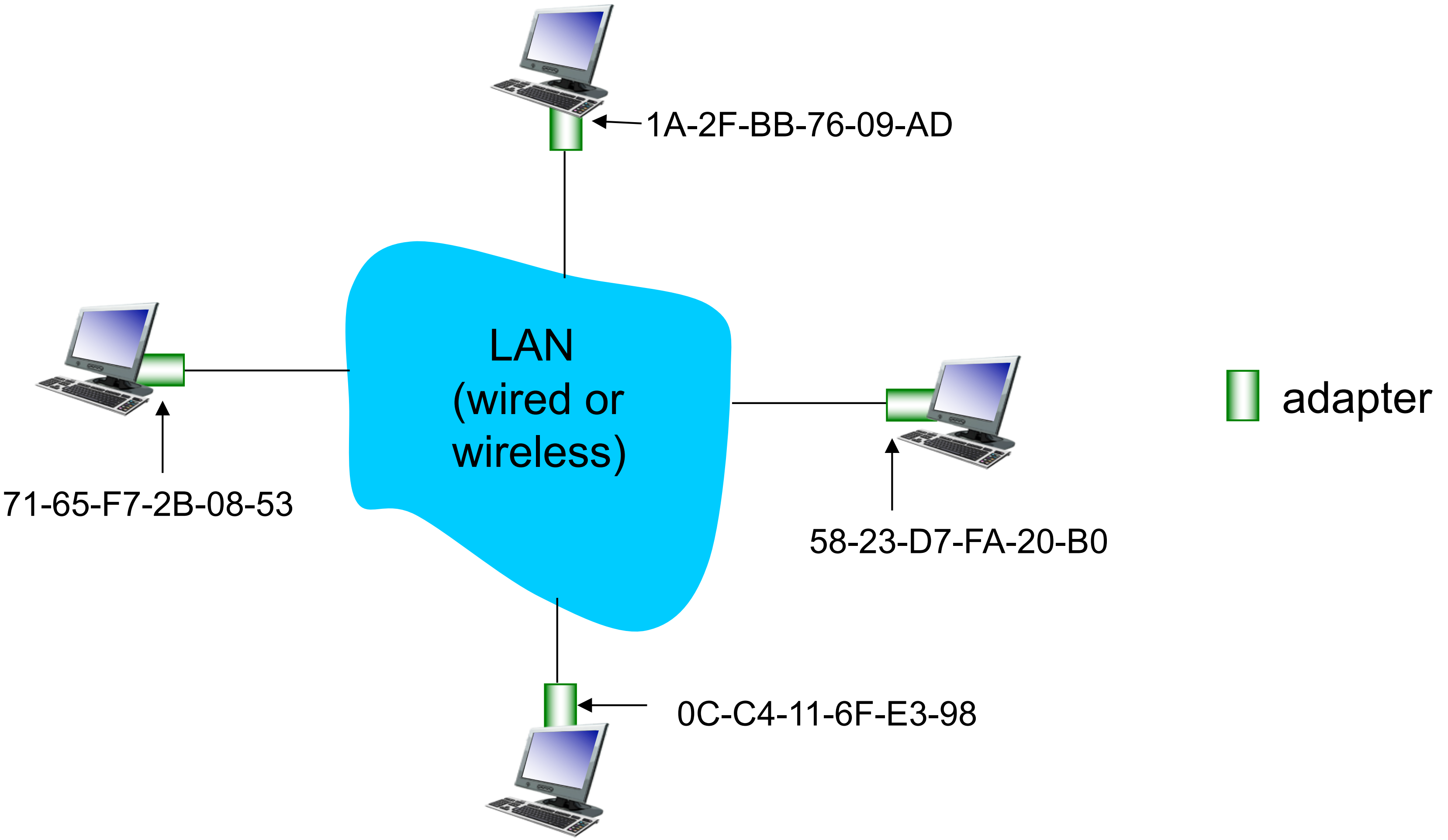
MAC addresses and ARP

- 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 configurable
  - e.g.: 1A-2F-BB-76-09-AD (also written 1A:2F:BB:76:09:AD)

hexadecimal (base 16) notation
(each "number" (letter/digit) represents 4 bits)

LAN addresses and ARP

each adapter on LAN has unique *LAN* address

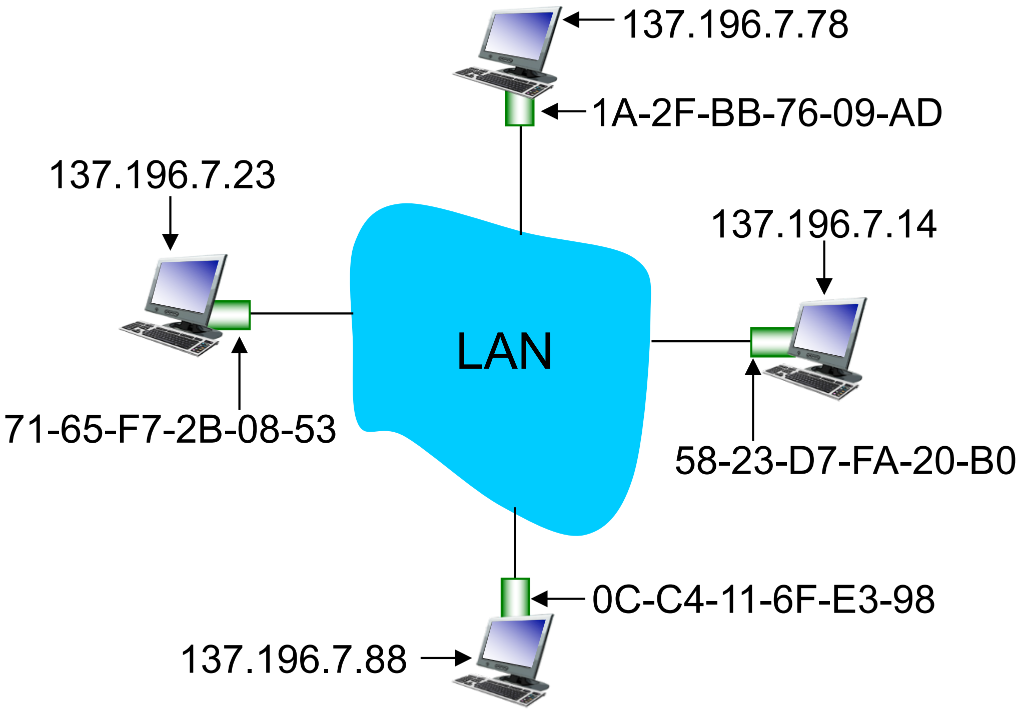


LAN addresses (more)

- MAC address allocation administered by IEEE
  - manufacturer buys portion of MAC address space
- MAC address bound to specific hardware (adapter)
- analogy:
  - MAC address: like Social Security Number
    - "personnummer"
  - 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 or router) on a LAN has a table:

- IP/MAC address mappings for nodes on the LAN:

  < IP address; MAC address; TTL>

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

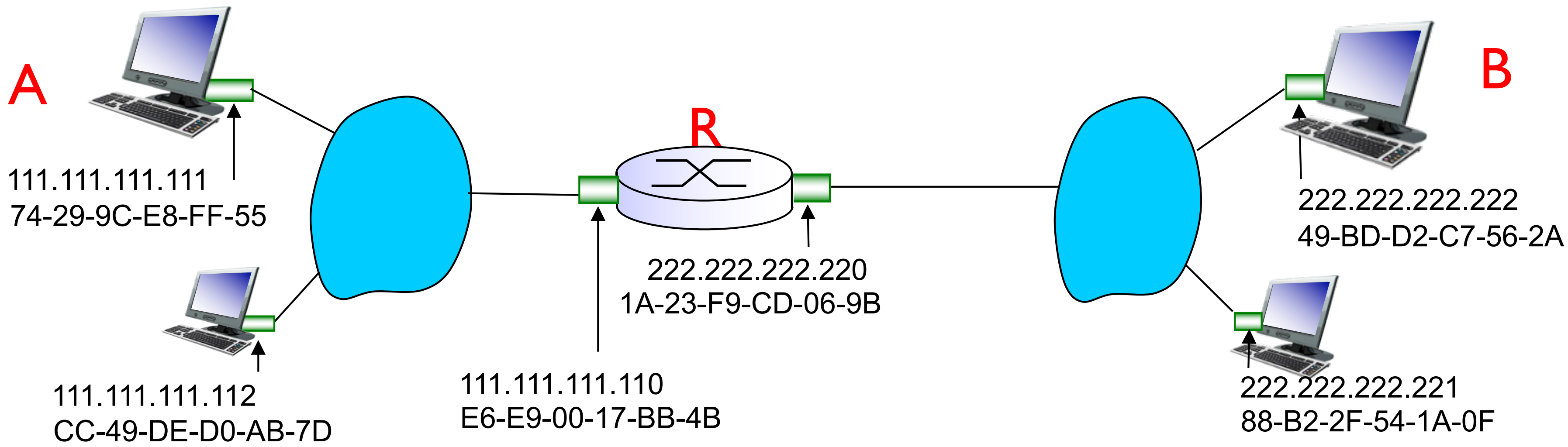
ARP protocol: same LAN

- A wants to send datagram to B
  - B's MAC address not in A's ARP table.
- A broadcasts ARP query, containing B's IP address
  - dest MAC address = FF-FF-FF-FF-FF-FF
  - all nodes on LAN receive the ARP query
- B receives ARP query, 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 configuration or intervention from network administrator*

Addressing: routing to another LAN

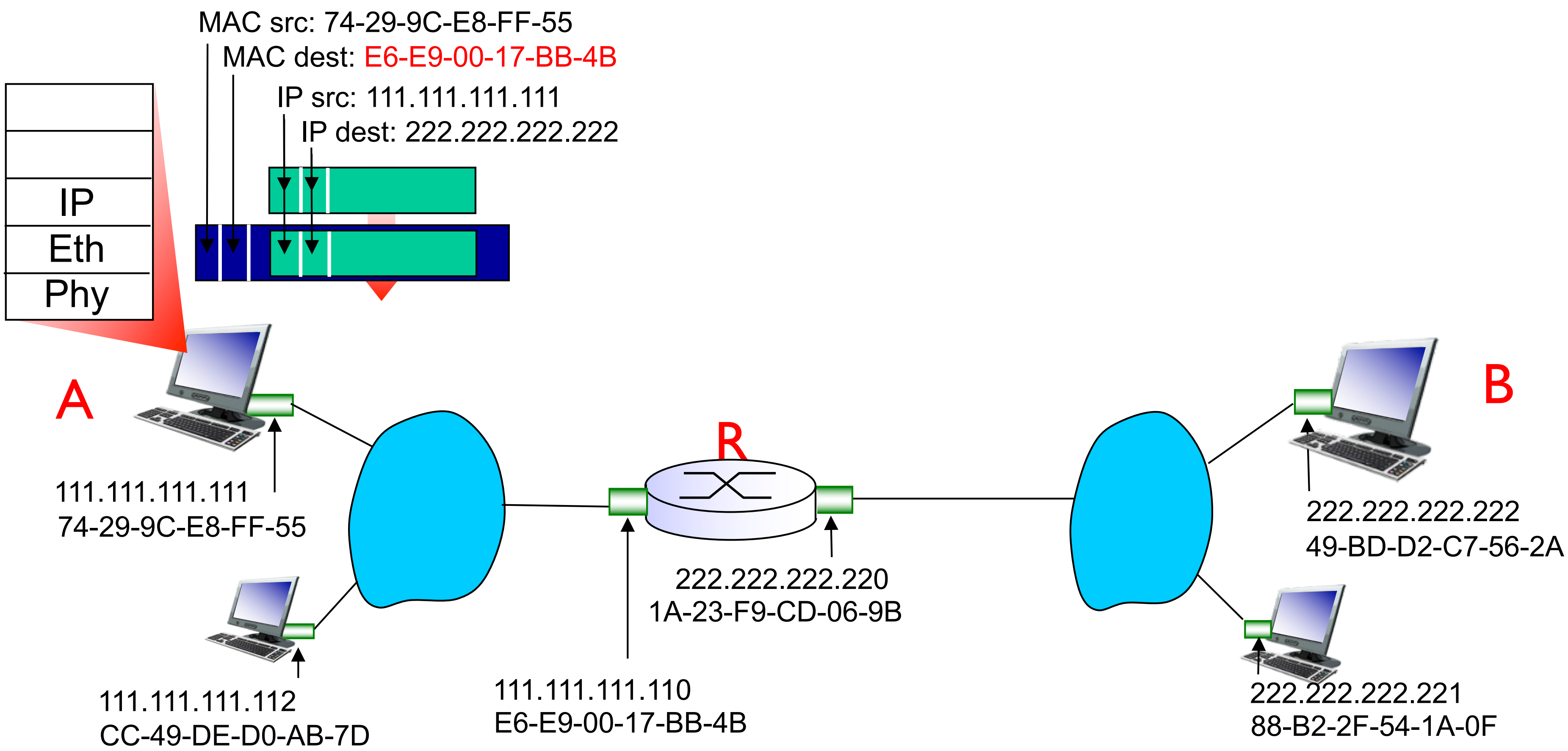
walkthrough: send datagram from A to B via R

- 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?)



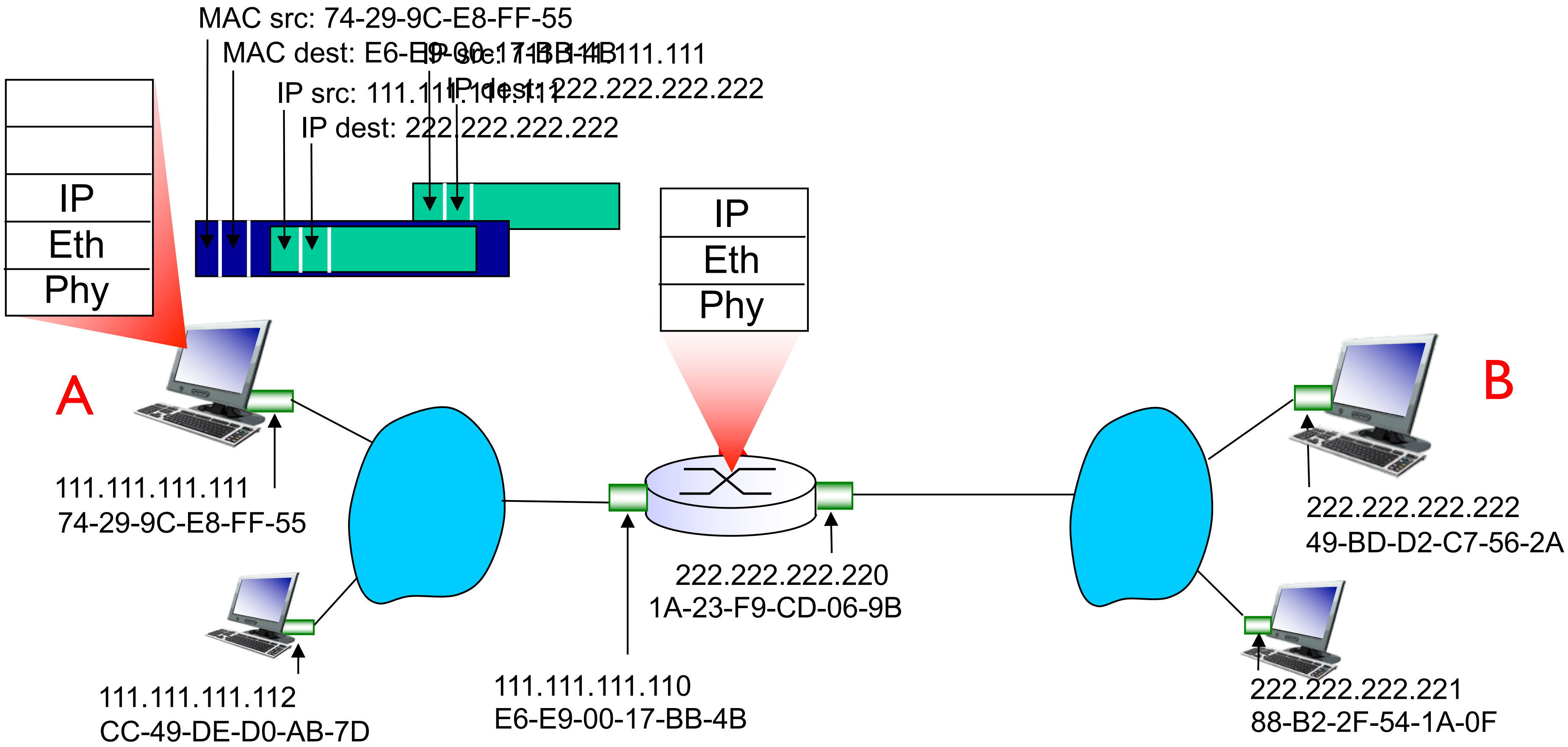
Addressing: routing to another LAN

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



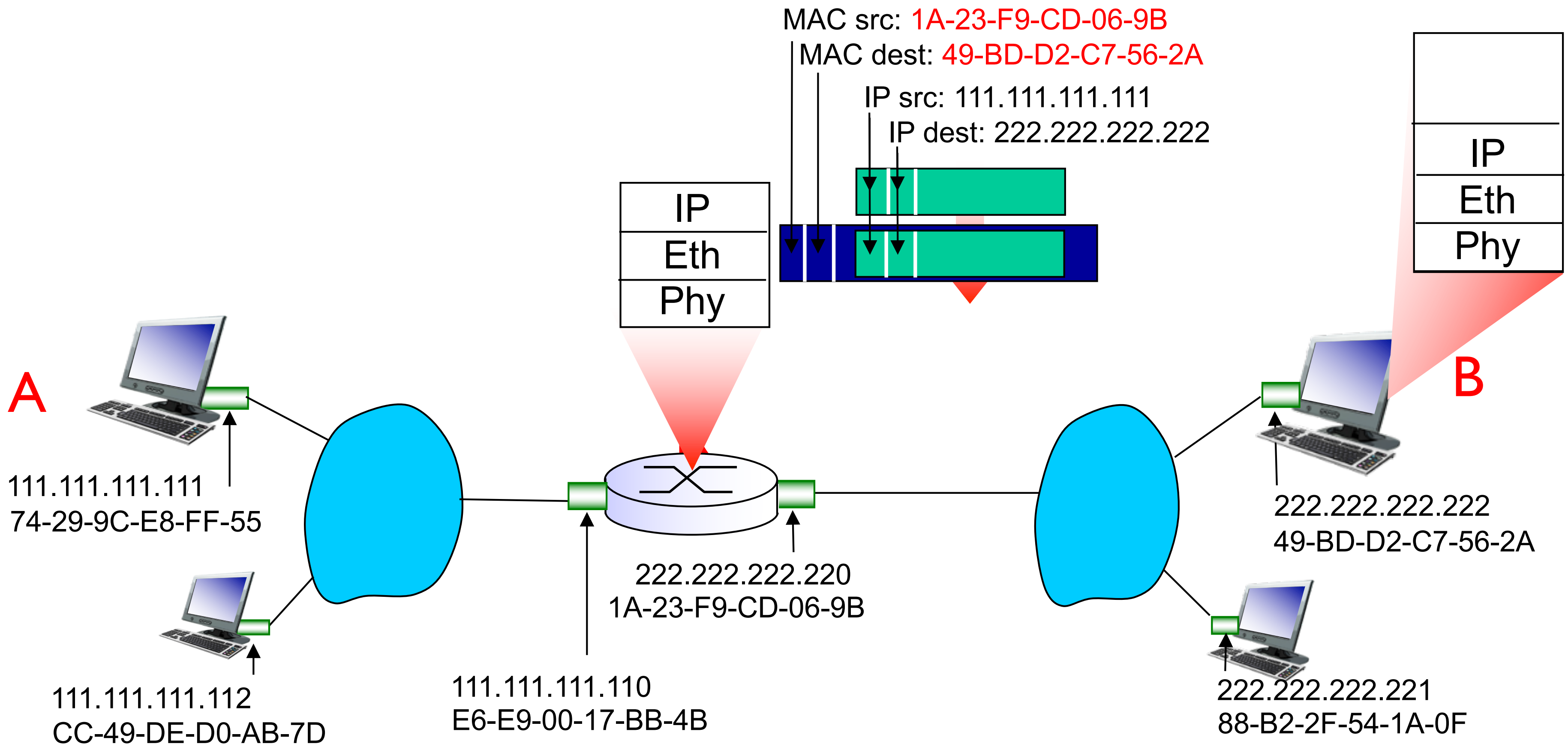
Addressing: routing to another LAN

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



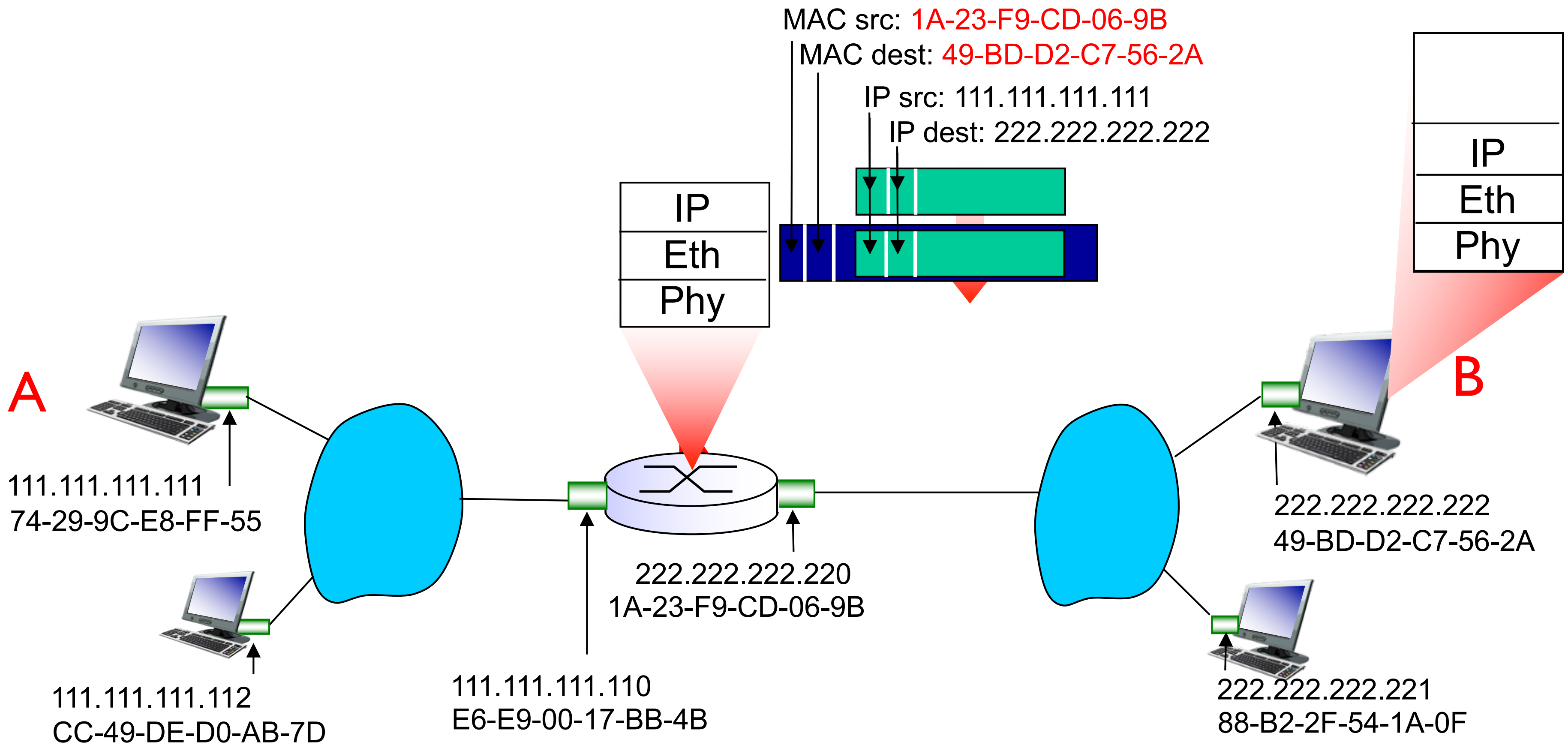
Addressing: routing to another LAN

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



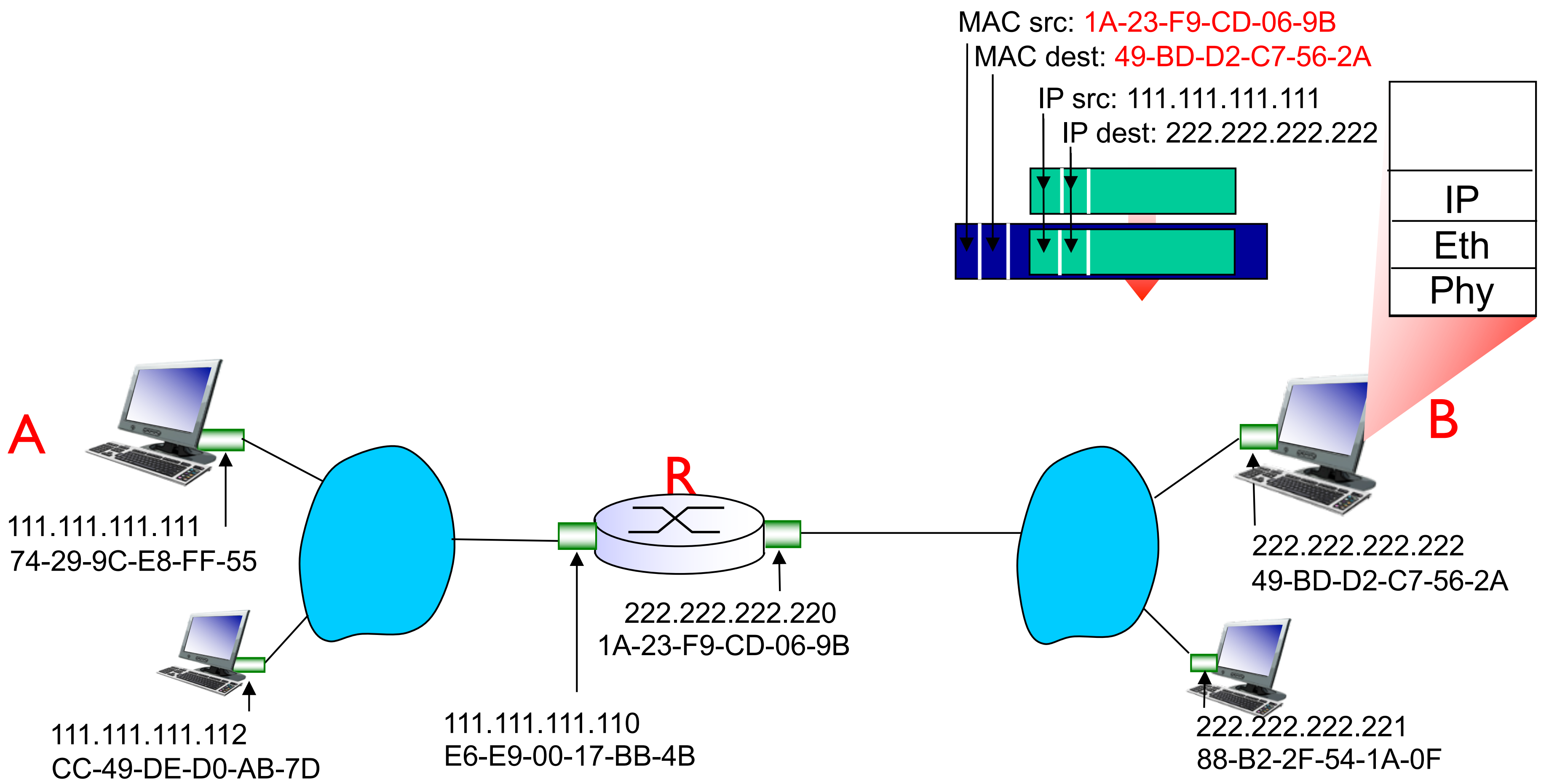
Addressing: routing to another LAN

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



Addressing: routing to another LAN

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



Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

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

5.5 link virtualization: MPLS

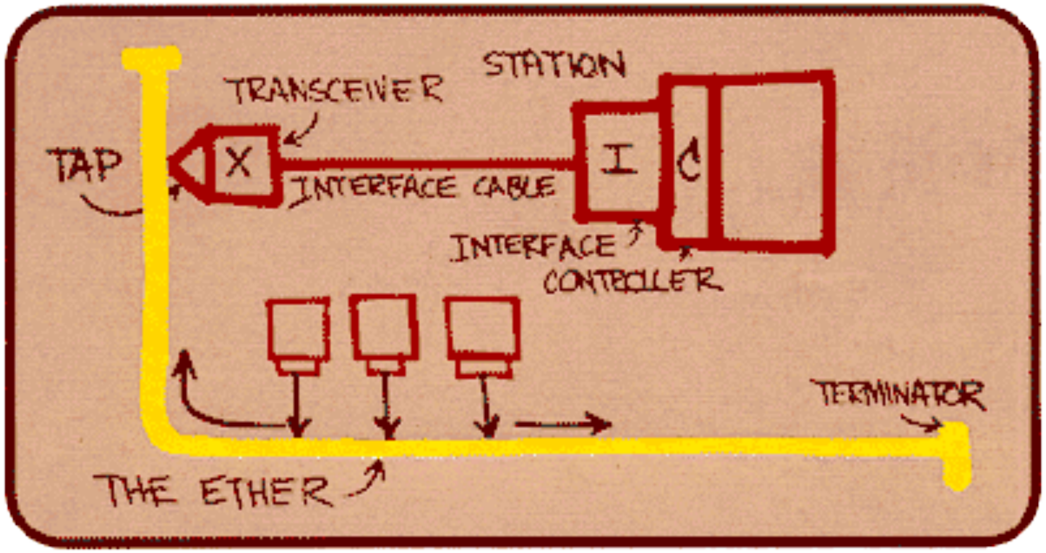
5.6 data center networking

5.7 a day in the life of a web request

Ethernet

"dominant" wired LAN technology:

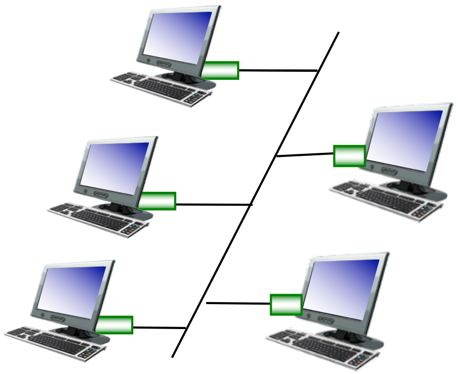
- cheap \$20 for NIC
- first widely used LAN technology
- simpler, cheaper than token LANs and ATM
- kept up with speed race: 10 Mbps – 100 Gbps



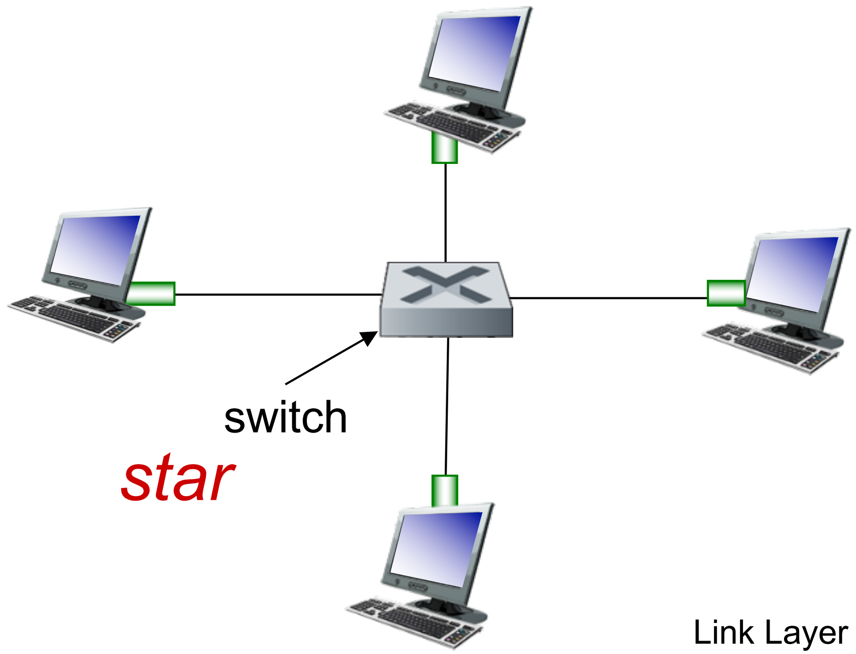
Metcalfe's Ethernet sketch

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)



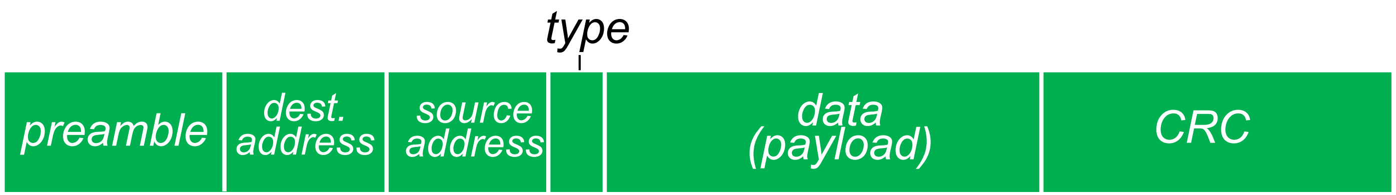
bus: coaxial cable



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 and sender clock rates

Ethernet frame structure (more)

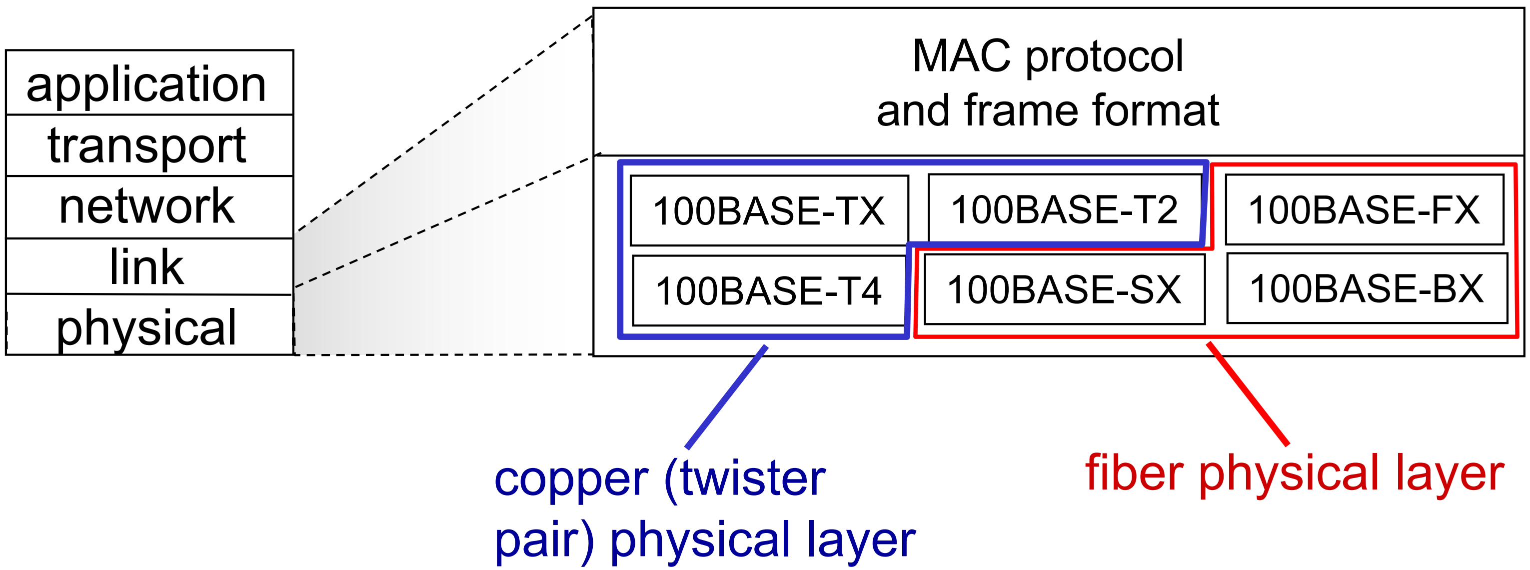
- *addresses:* 48-bit source and destination MAC addresses
  - if adapter receives frame with matching destination address, or with broadcast address (e.g. ARP packet), it passes data in frame to network layer protocol
  - otherwise, adapter discards frame
- *type:* indicates higher layer protocol (mostly IP but others possible, e.g., Novell IPX, AppleTalk)
- *CRC:* cyclic redundancy check at receiver
  - error detected: frame is dropped

Ethernet: unreliable, connectionless

- *connectionless:* no handshaking between sending and receiving NICs
- *unreliable:* receiving NIC doesn't send acknowledgements (positive or negative) to sending NIC
- Ethernet's MAC protocol: unslotted *CSMA/CD with binary back-off*
  - Back-off: wait before trying again after a collision
    - Time to wait is randomized
    - After *c* collisions, wait between 0 and 2^c-1 time units

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, 100 Gbps
  - different physical layer media: fiber, cable



Ethernet Standards (Selected)

Standard	Year	Description
Ethernet II (DIX)	1982	10 Mbit/s over thick coax cable
IEEE 802.3	1983	10 Mbit/s over thick coax cable
IEEE 802.3a	1985	10 Mbit/s over thin coax cable
IEEE 802.3i	1990	10 Mbit/s over twisted pair
IEEE 802.3j	1993	10 Mbit/s over fiber
IEEE 802.3u	1995	100 Mbit/s over twisted pair and fiber (Fast Ethernet)
IEEE 802.3x	1997	Full duplex, flow control, DIX framing
IEEE 802.3z	1998	1 Gbit/s Ethernet over fiber (Gigabit Ethernet)
IEEE 802.3ab	1999	1 Gbit/s Ethernet over twisted pair (Gigabit Ethernet)
IEEE 802.3ae	2003	10 Gbit/s Ethernet over fiber
IEEE 802.3af	2003	Power over Ethernet (12.95 W)
IEEE 802.3ah	2004	Ethernet in the First Mile
IEEE 802.3an	2006	10 Gbit/s Ethernet over twisted pair
IEEE 802.3av	2009	10 Gbit/s EPON
IEEE 802.3az	2010	Energy Efficient Ethernet
IEEE 802.3ba	2010	40 Gbit/s and 100 Gbit/s Ethernet over fiber and (short) copper

Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs

- addressing, ARP
- Ethernet
- switches
- ~~VLANS~~

5.5 link virtualization: MPLS

5.6 data center networking

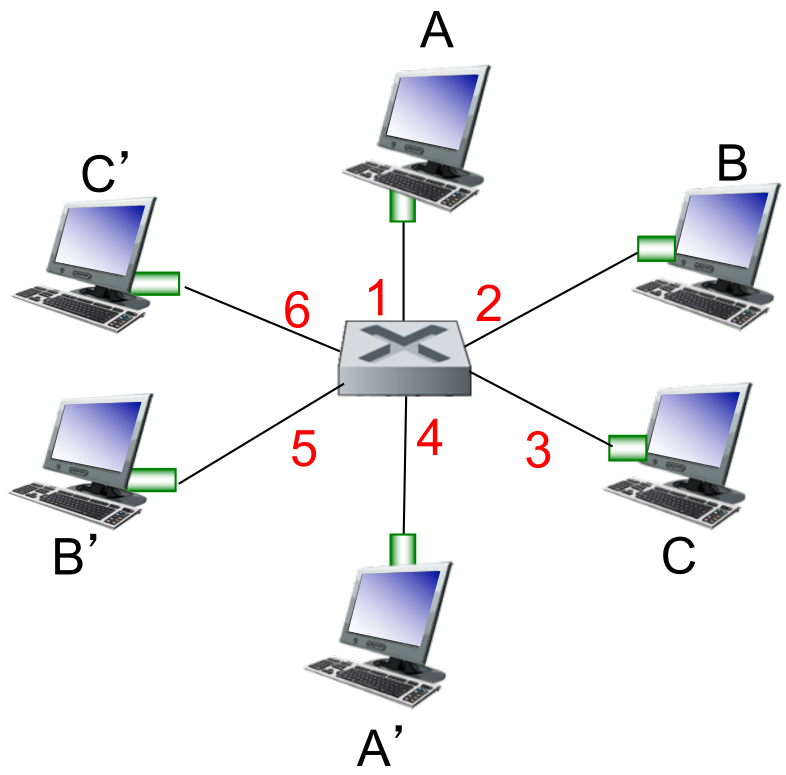
5.7 a day in the life of a web request

Ethernet switch

- link-layer device: takes an *active* role
  - store and 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 connections 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

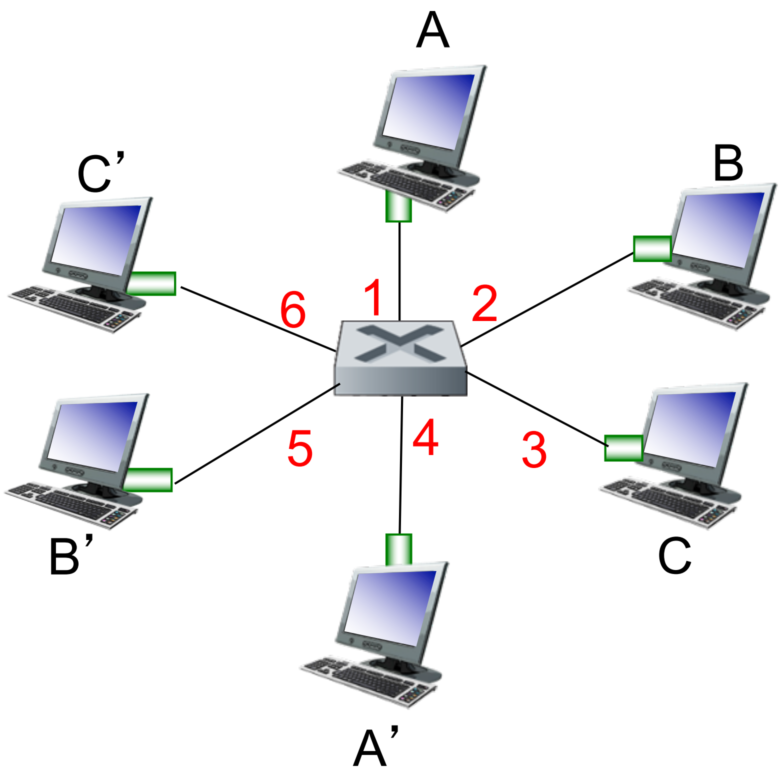


switch with six interfaces (1,2,3,4,5,6)

Switch forwading table

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

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



switch with six interfaces (1,2,3,4,5,6)

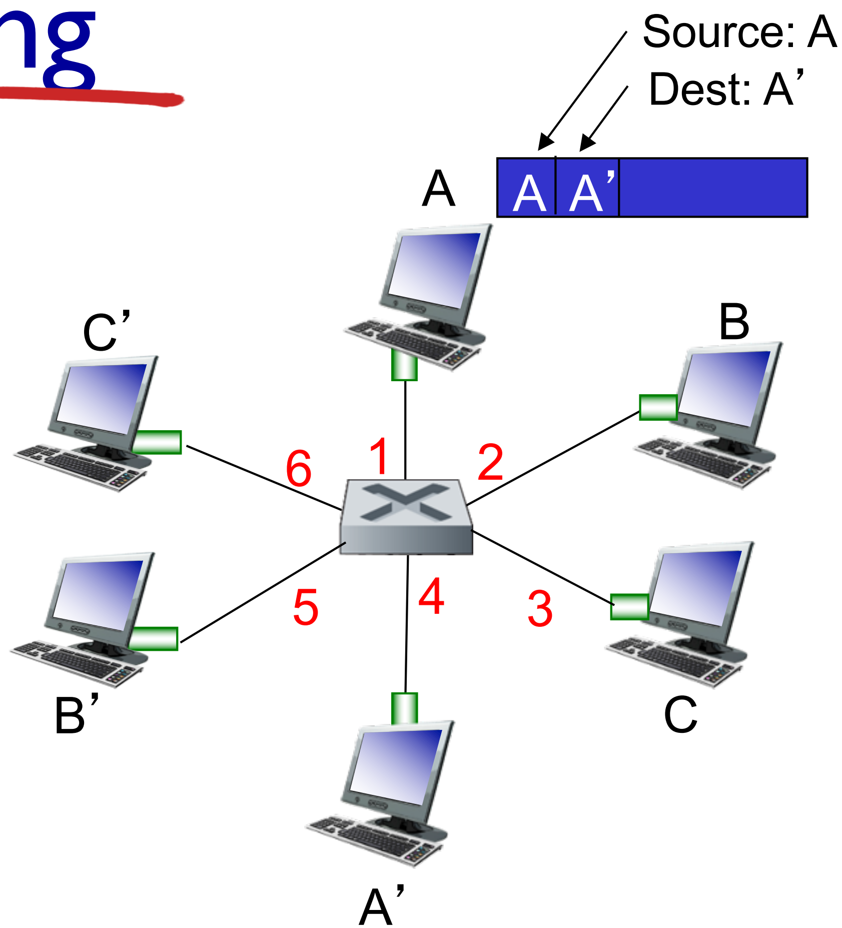
Q: how are entries created and maintained in switch table?

- *something like a routing protocol?*

- *A: No. Through self-learning*

Switch: self-learning

- 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



MAC addr	interface	TTL
A	*1*	*60*

Switch table (initially empty)

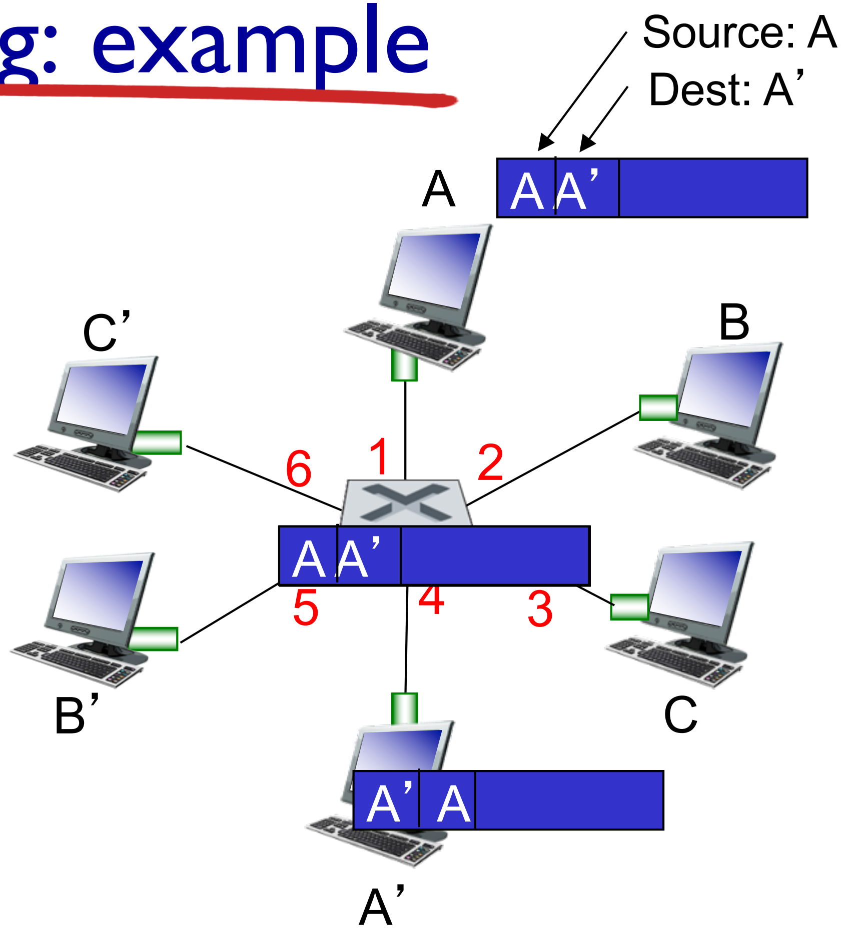
Switch: frame filtering/forwarding

when frame received at switch:

1. record incoming link and 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*

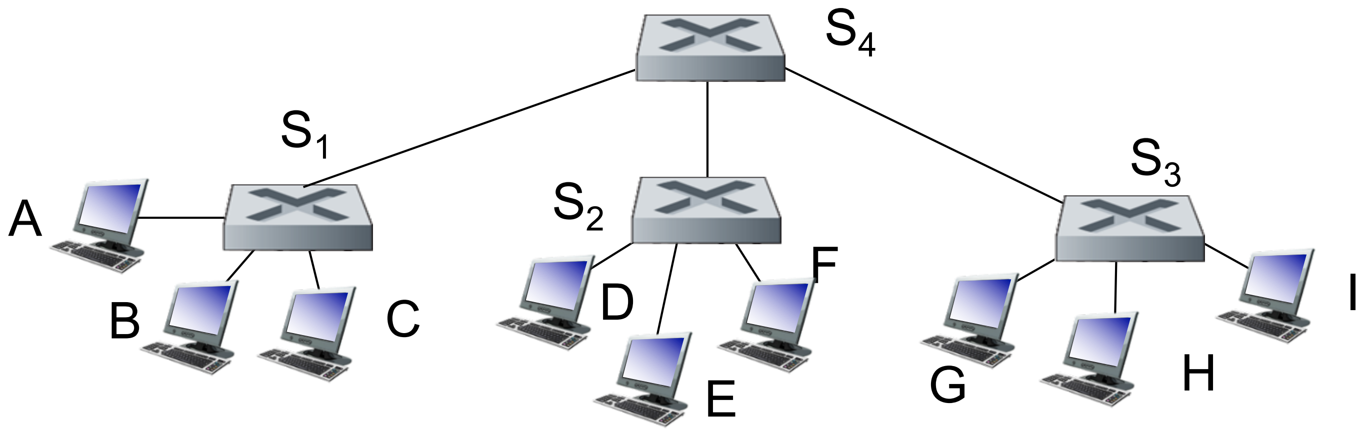


MAC addr	interface	TTL
A	*1*	*60*
A'	*4*	*60*

switch table (initially empty)

Interconnecting switches

- switches can be connected together

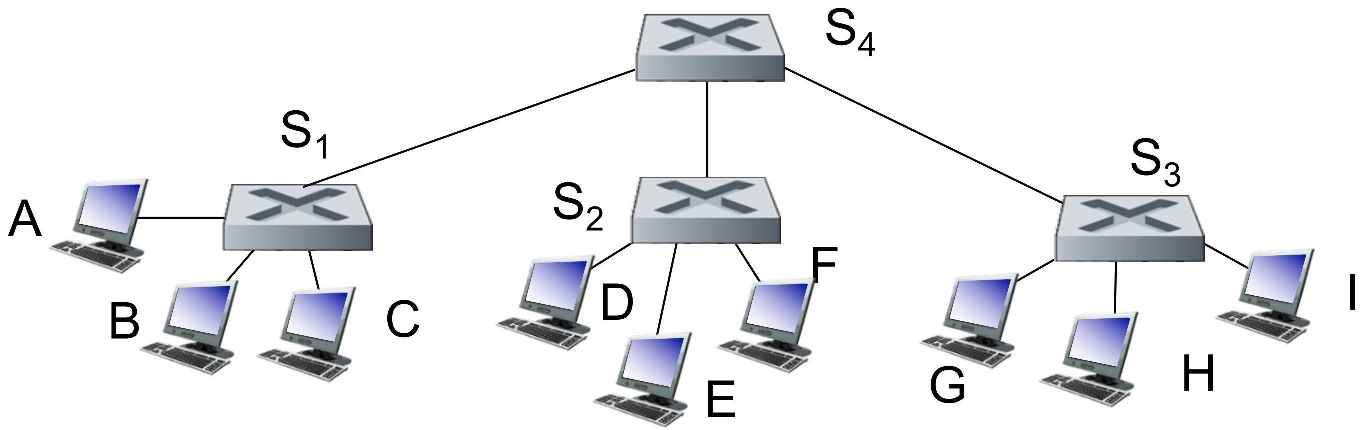


Q: sending from A to G - how does S_1 know to forward frame destined to F via S_4 and S_3?

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

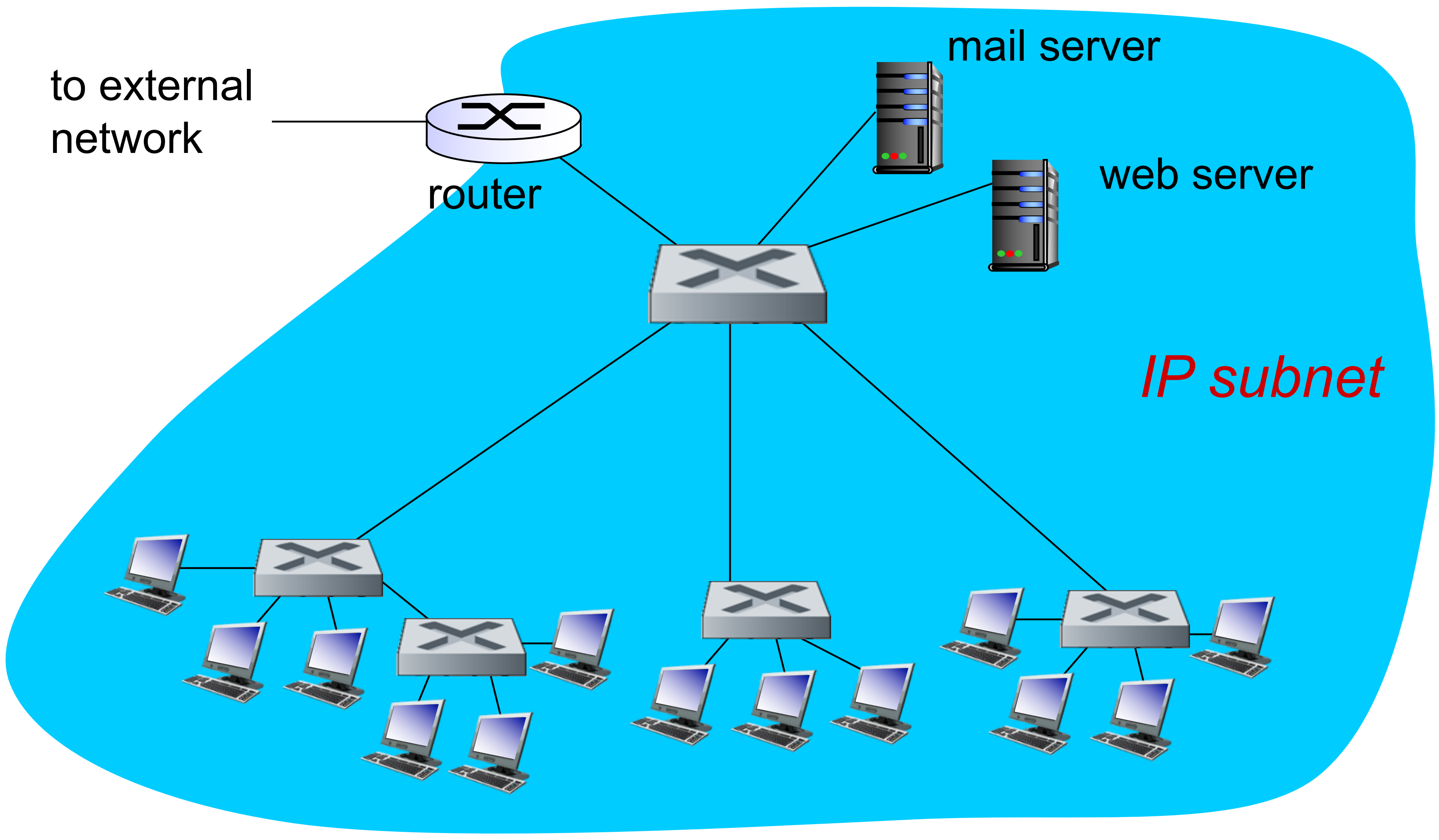
Self-learning multi-switch example

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



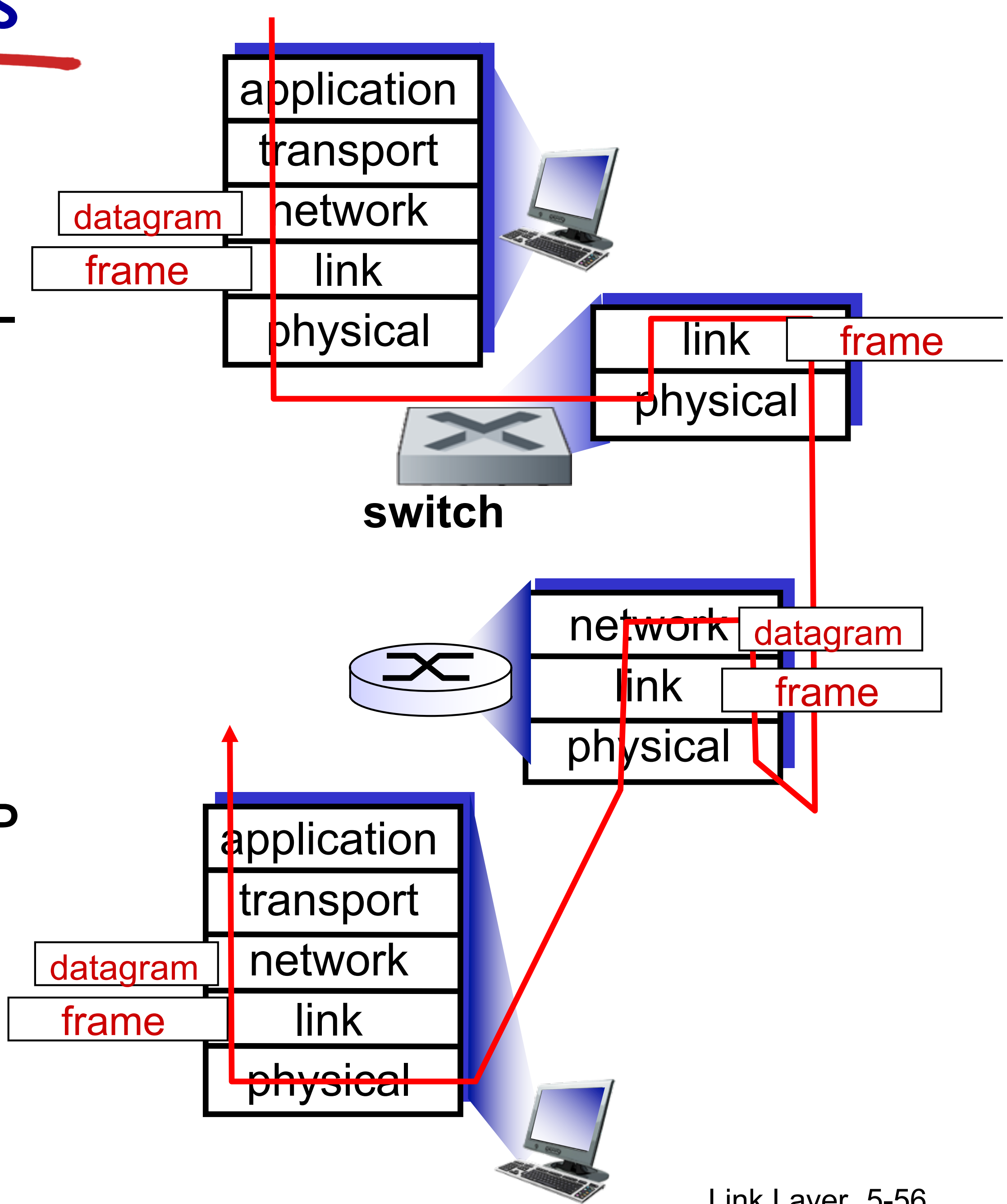
Switches vs. routers

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



Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

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

5.5 ~~link virtualization: MPLS~~

5.6 data center networking

5.7 a day in the life of a web request

Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

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

5.5 link virtualization: MPLS

5.6 data center networking

5.7 a day in the life of a web request

Data center networks

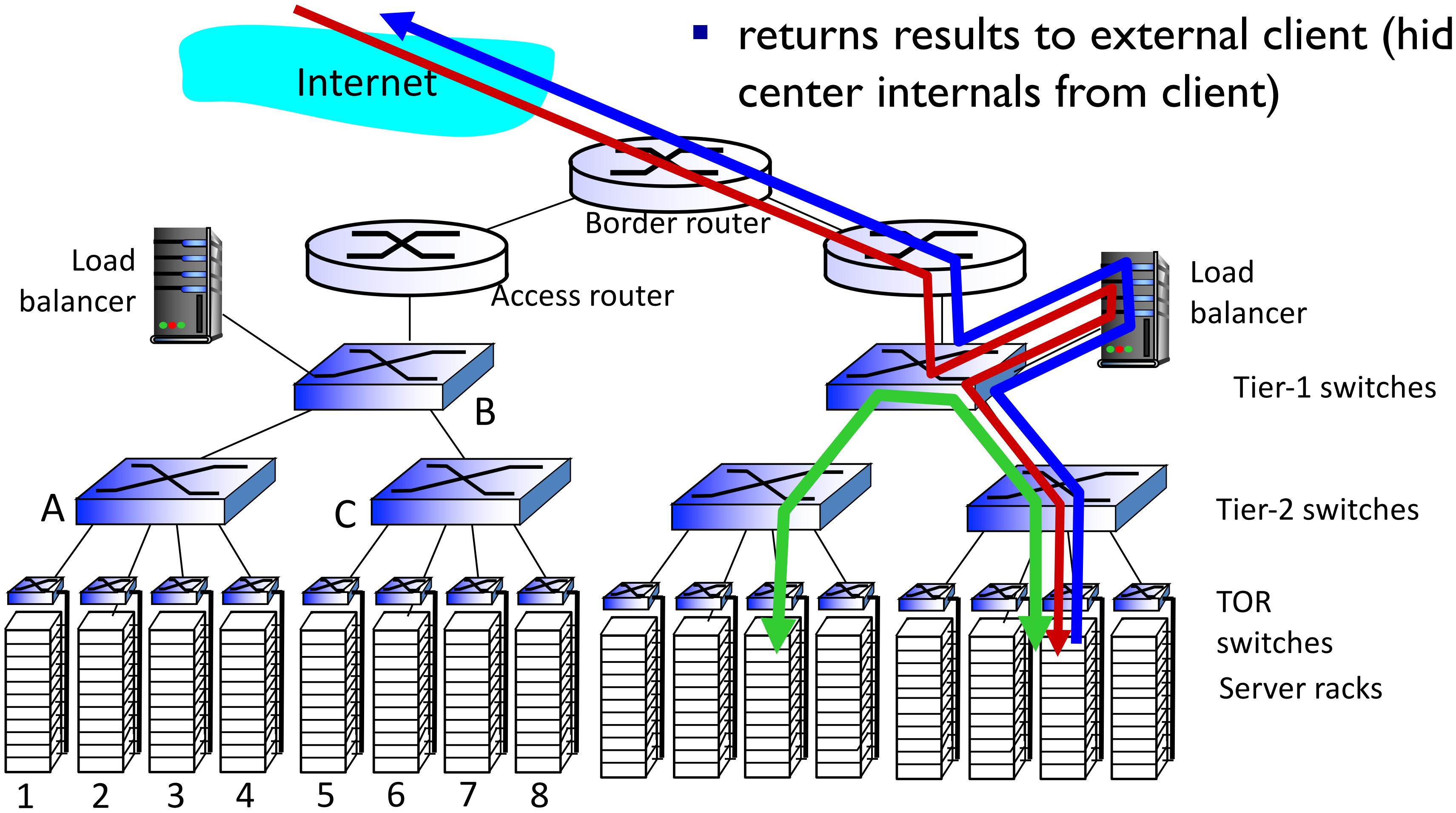
- 10's to 100's of thousands of hosts, often closely coupled, in close proximity:
  - e-business (e.g. Amazon)
  - content-servers (e.g., YouTube, Facebook, Akamai, Apple, Microsoft)
  - search engines, data mining (e.g., Google)
- challenges:
  - multiple applications, each serving massive numbers of clients
  - managing/balancing load, avoiding bottlenecks in processing, networking, and data access

Inside a 40-ft Microsoft container, Chicago data center

Data center networks

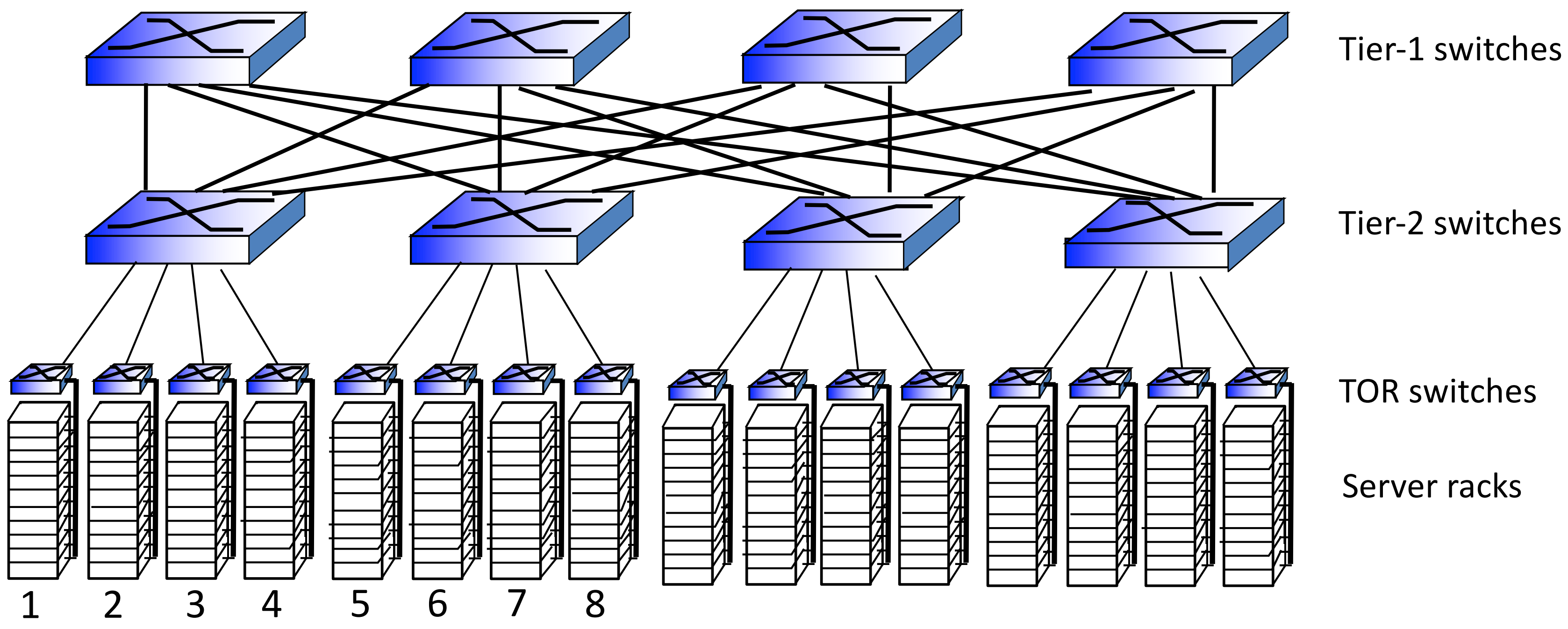
load balancer: application-layer routing

- receives external client requests
- directs workload within data center
- returns results to external client (hiding data center internals from client)



Data center networks

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



Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

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

5.5 link virtualization: MPLS

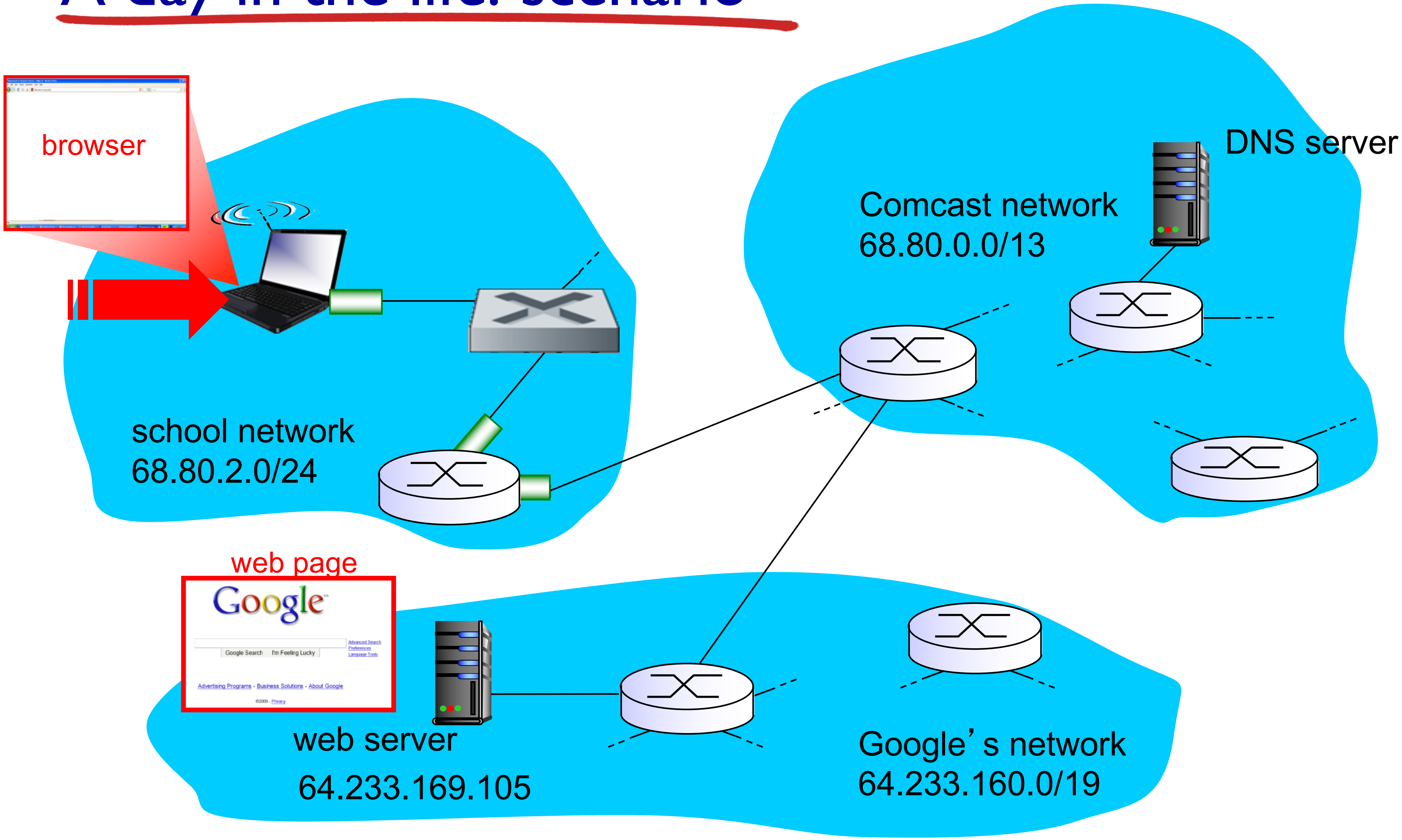
5.6 data center networking

5.7 a day in the life of a web request

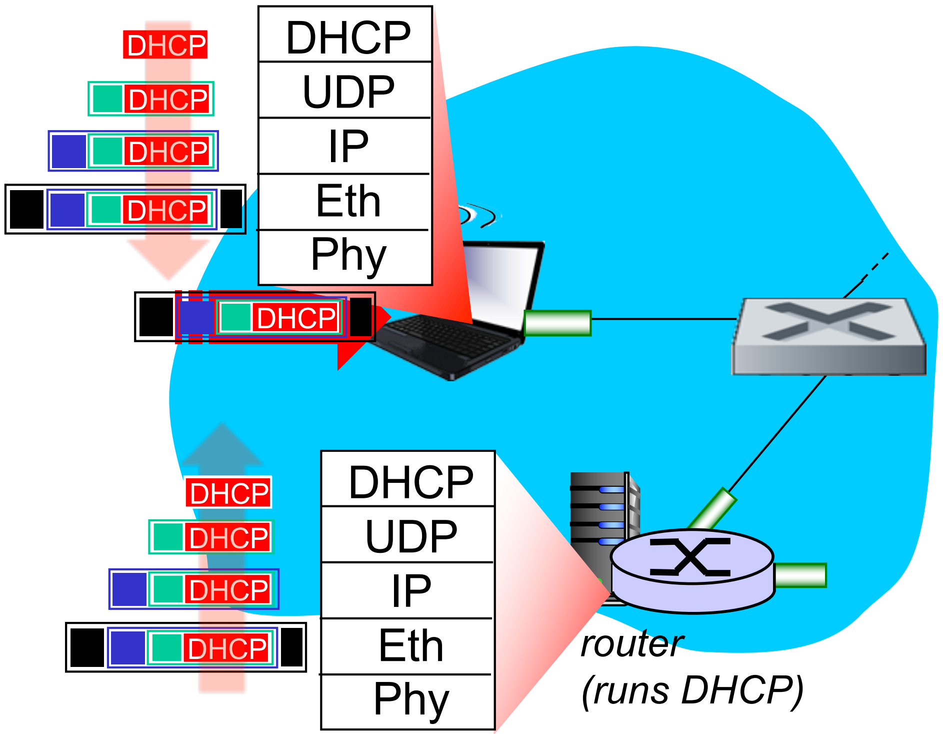
Synthesis: a day in the life of a web request

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

A day in the life: scenario

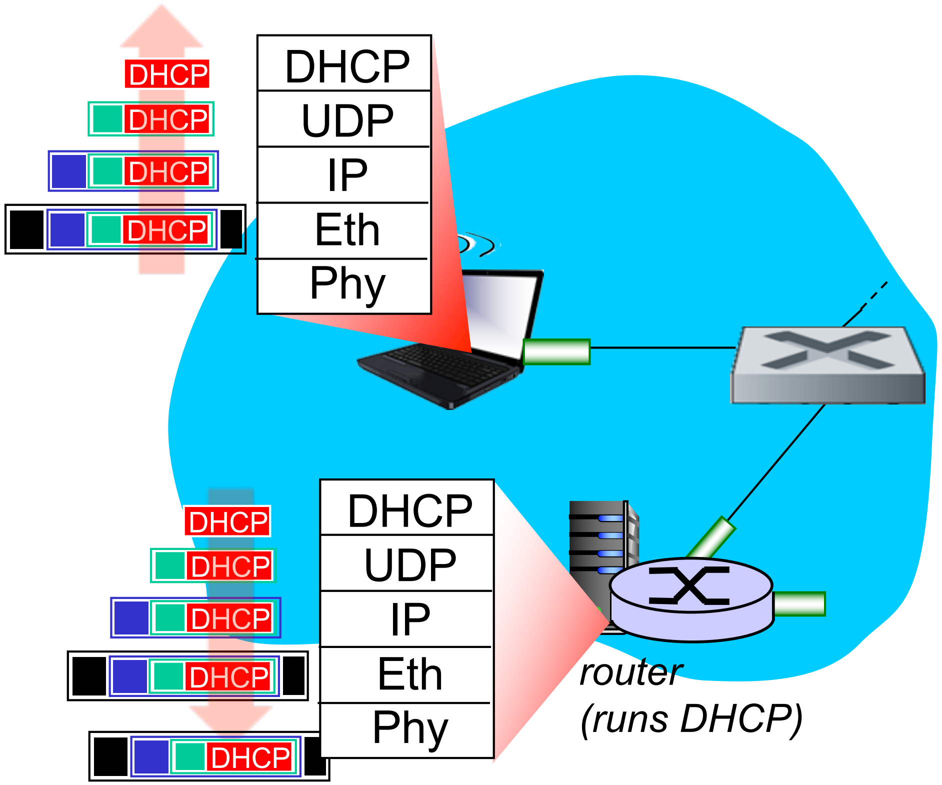


A day in the life… connecting to the Internet



- connecting laptop needs to get its own IP address, address of first-hop router, and address of DNS server: use *DHCP*
- DHCP request *encapsulated* in *UDP*, encapsulated in *IP*, encapsulated in *802.3* Ethernet
- Ethernet frame *broadcast* (destination: FF-FF-FF-FF-FF-FF) on LAN, received at router running *DHCP* server
- Ethernet *demultiplexed* to IP demultiplexed to UDP demultiplexed to DHCP

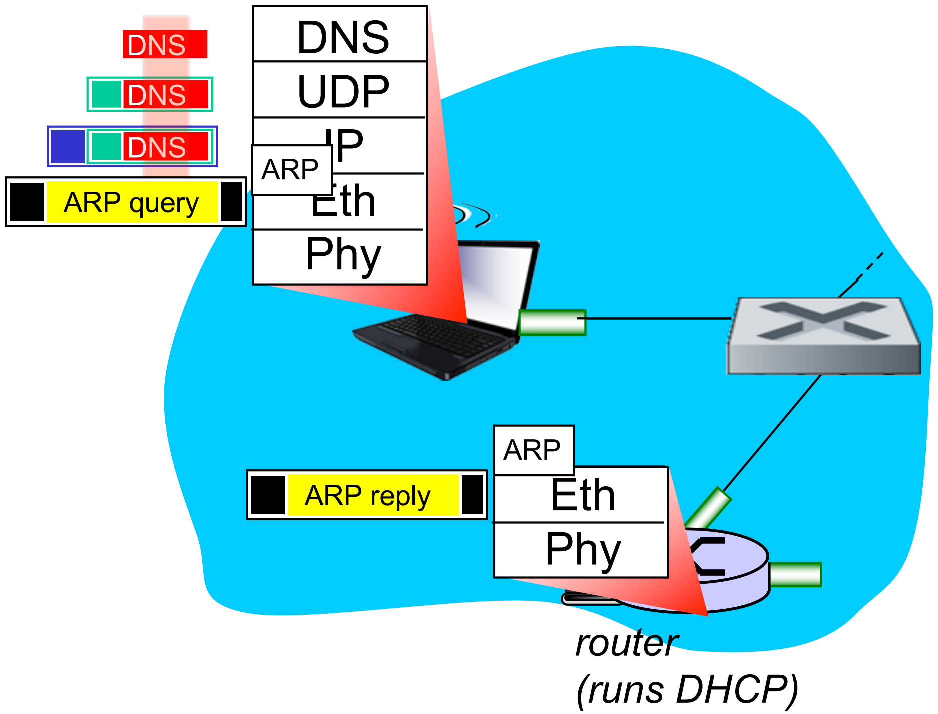
A day in the life... connecting to the Internet



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

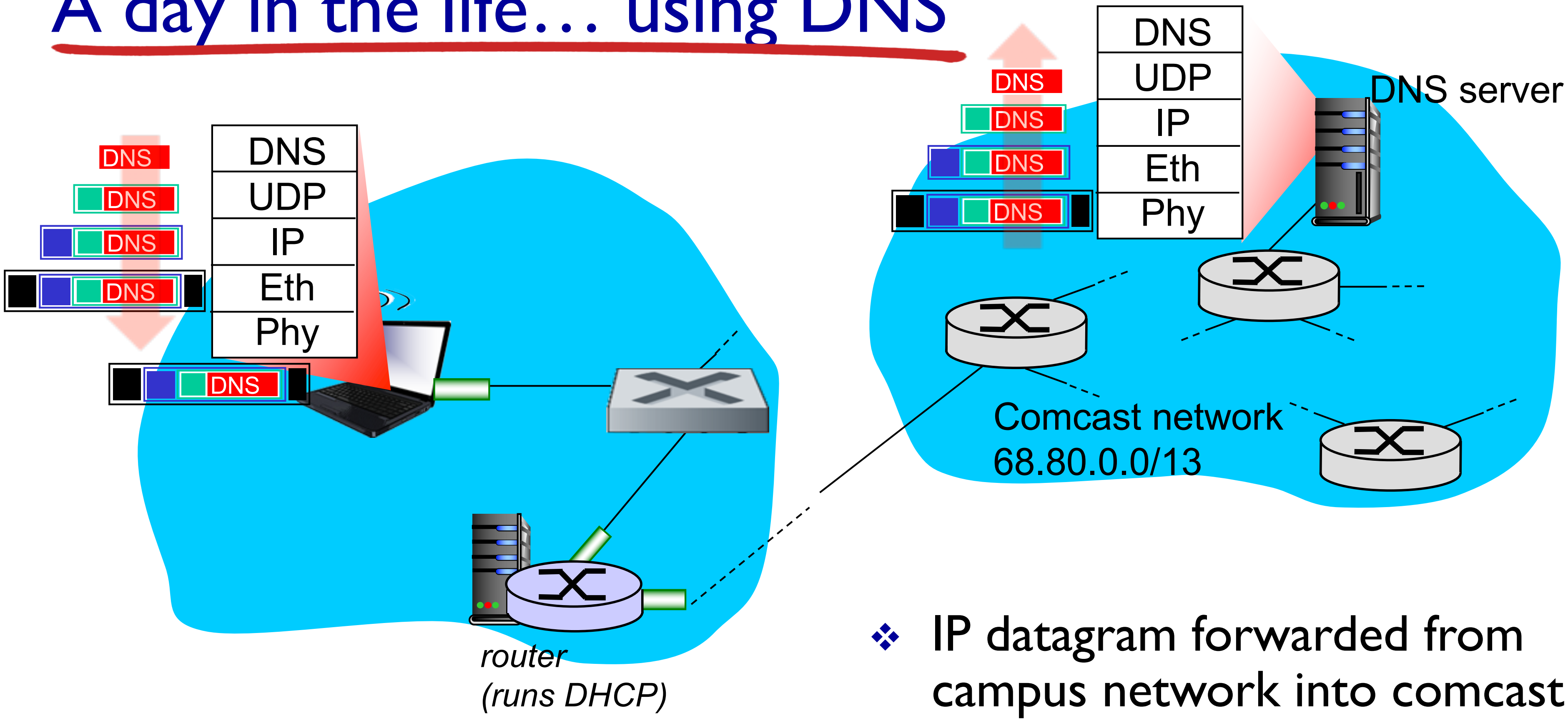
Client now has its IP address, knows name & address of DNS server, and IP address of its first-hop router

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



- before sending *HTTP* request, client needs IP address of www.google.com: *DNS*
- DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Ethernet. 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

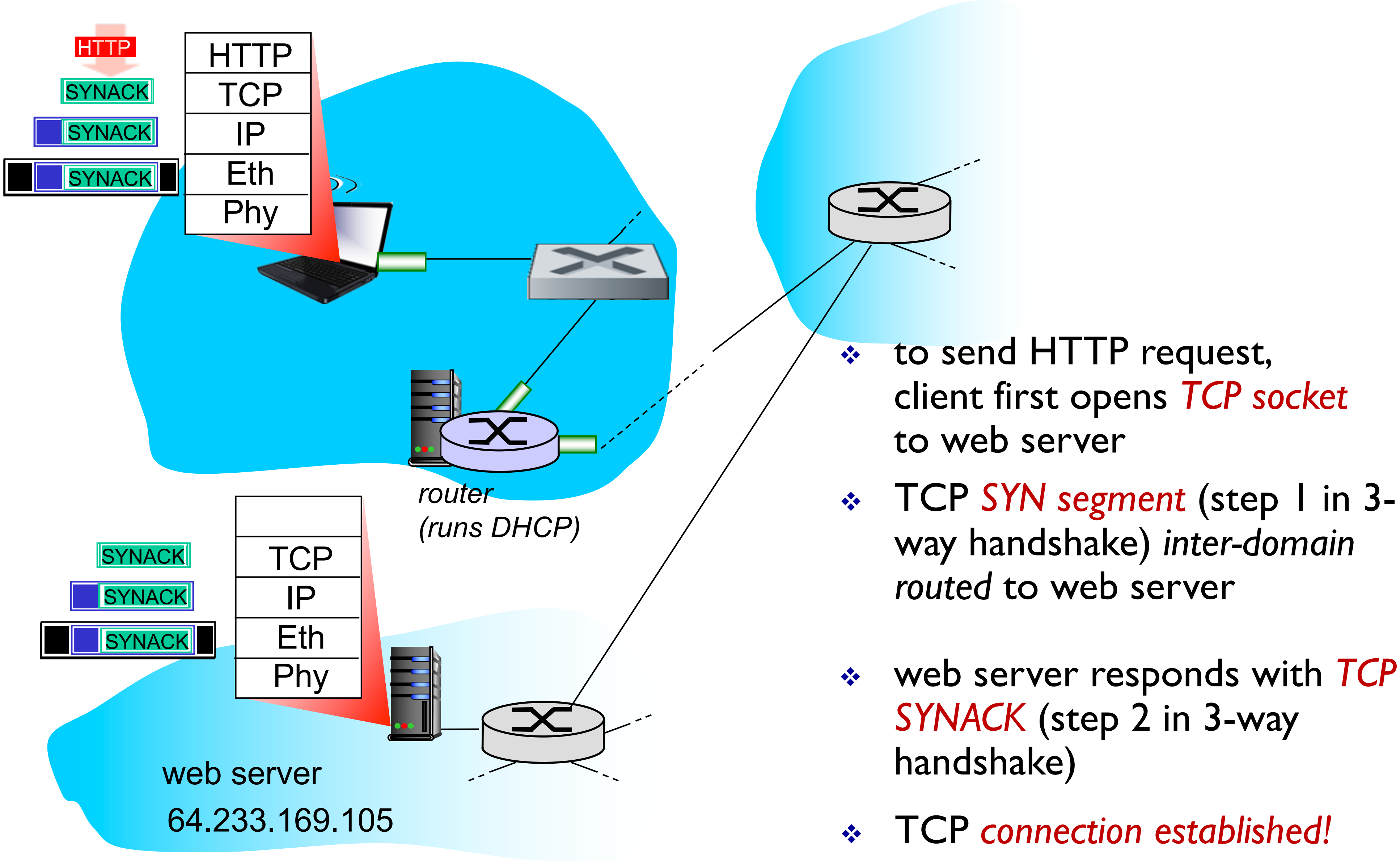
A day in the life… using DNS



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

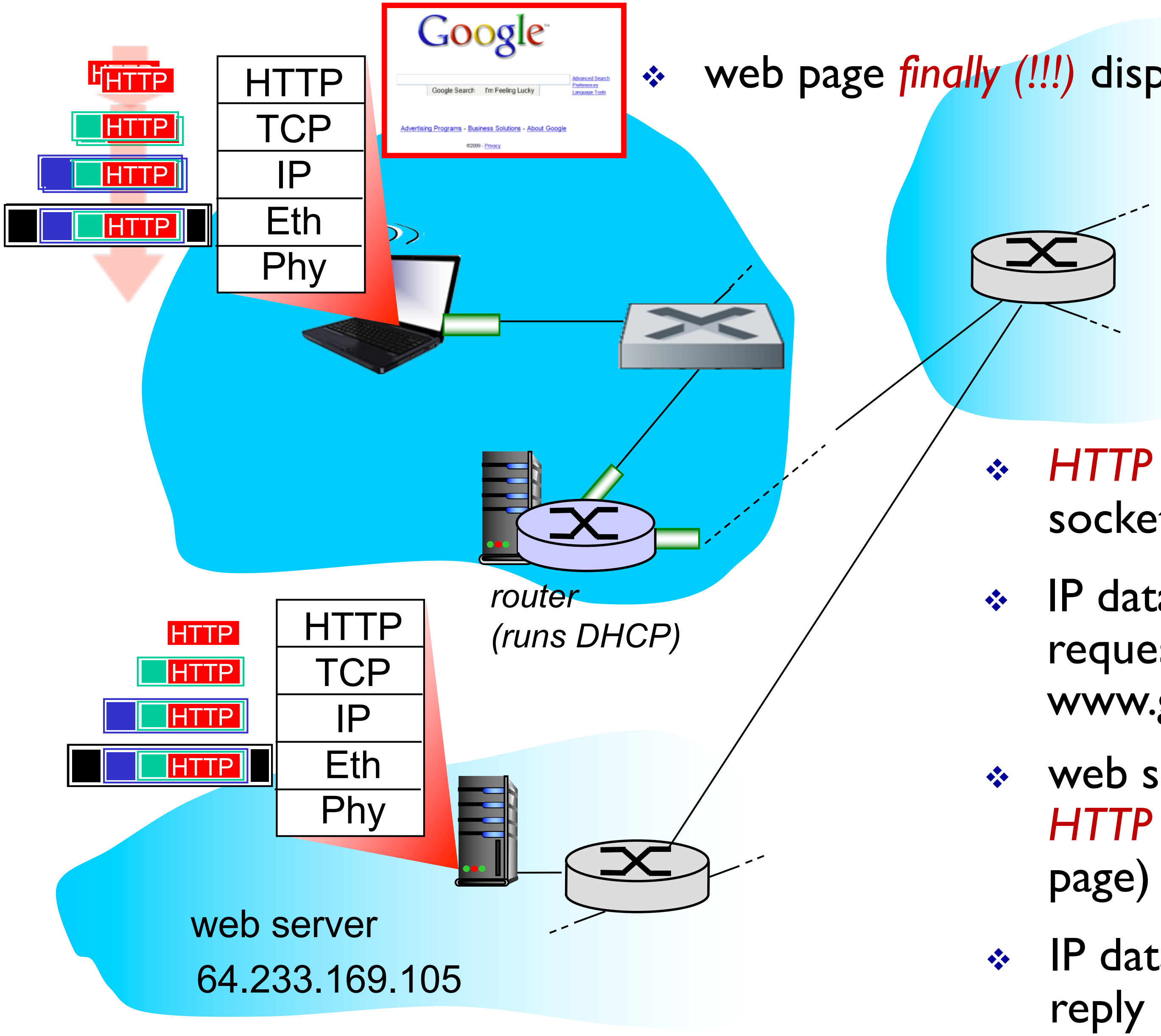
- IP datagram forwarded from campus network into comcast network, routed (tables created by *RIP, OSPF, IS-IS* and/or *BGP* routing protocols) to DNS server
- demultiplexed to DNS server
- DNS server replies to client with IP address of www.google.com

A day in the life…TCP connection carrying HTTP



- 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

- *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 5: 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 5: 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
  - network management