

Internet Protocols EBU5403

The Data Link Layer part 2

Michael Chai (michael.chai@qmul.ac.uk)

Richard Clegg (r.clegg@qmul.ac.uk)

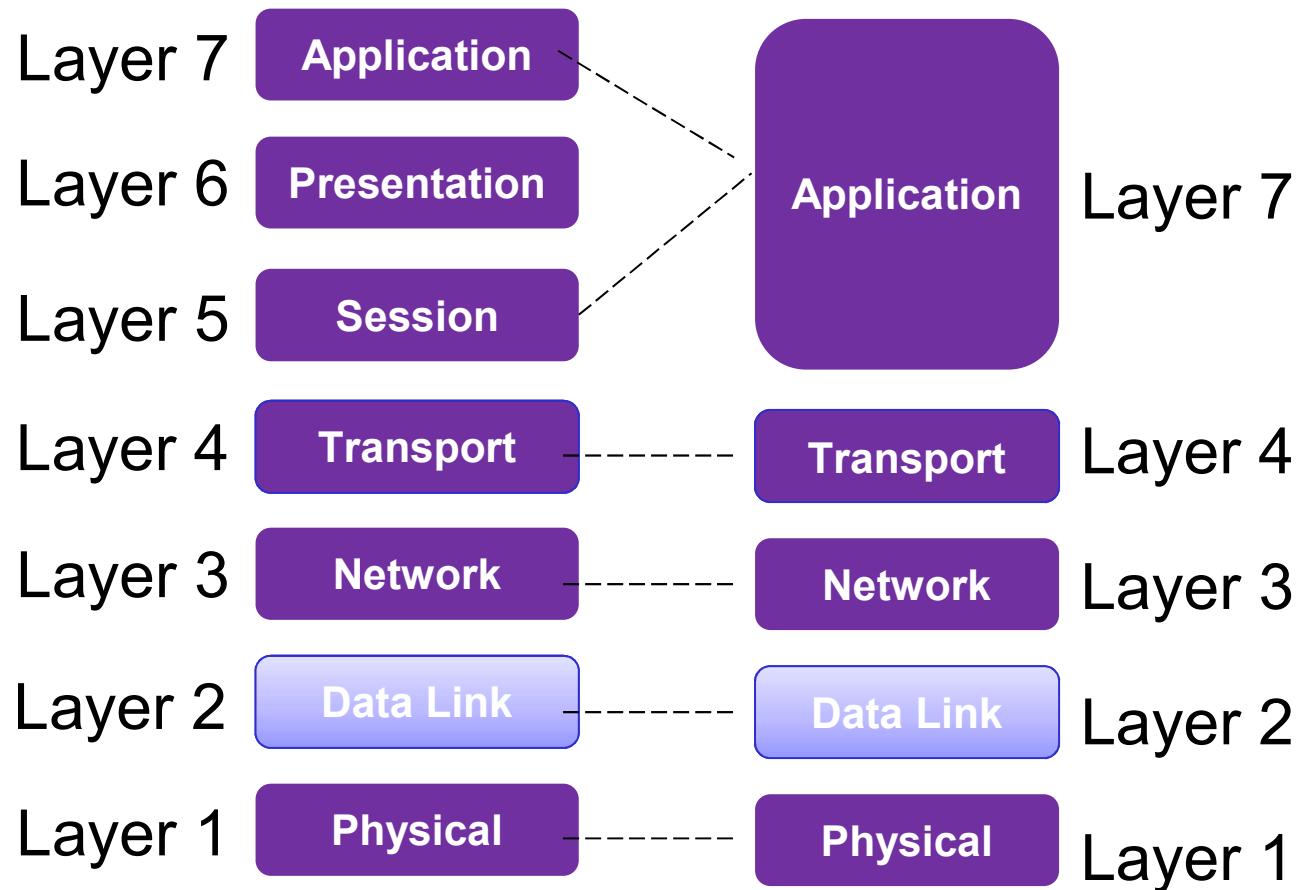
Adnan Kiani (adnankhal@googlemail.com)

	Week 1	Week 2	Week 3	Week 4
Telecom	Adnan Kiani	Michael Chai		Adnan Kiani
E-Commerce	Richard Clegg	Michael Chai	Richard Clegg	

Structure of course

- Week 1 (25th-29th September)
 - Introduction to IP Networks
 - The Transport layer (part I)
- Week 2 (16th-21st October)
 - The Transport layer (part II)
 - The Network layer (part I)
 - Class test (open book exam in class)
- Week 3 (20th-24th November)
 - The Network layer (part II)
 - The Data link layer (part I)
 - Router lab tutorial (assessed labwork after this week)
- Week 4 (18th-22nd December)
 - The Data link layer (part II)
 - Security and network management
 - Class test

Data Link Layer



Link layer, LANs: outline

6.1 introduction, services

6.2 error detection,
correction

6.3 multiple access
protocols

6.4 LANs

- addressing, ARP
- Ethernet
- WiFi
- switches
- VLANs

6.5 link virtualization:
MPLS

6.6 data center
networking

6.7 a day in the life of a
web request

Multiple access links, protocols

two types of “links”:

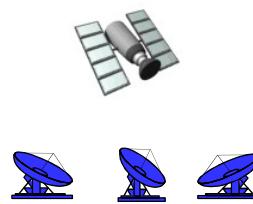
- **point-to-point**
 - PPP for dial-up access
 - point-to-point link between Ethernet switch, host
- **broadcast (shared wire or medium)**
 - old-fashioned Ethernet
 - upstream HFC (hybrid fibre coaxial)
 - 802.11 wireless LAN



shared wire (e.g.,
cabled Ethernet)



shared RF
(radio frequency)
(e.g., 802.11 WiFi)



shared RF
(satellite)



humans at a
cocktail party
(shared air, acoustic)

Multiple access protocols

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

multiple access protocol

- distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
- communication about channel sharing must use channel itself!
 - no out-of-band channel for coordination

An ideal multiple access protocol

given: broadcast channel of rate R bps

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, slots
4. simple

MAC protocols: taxonomy

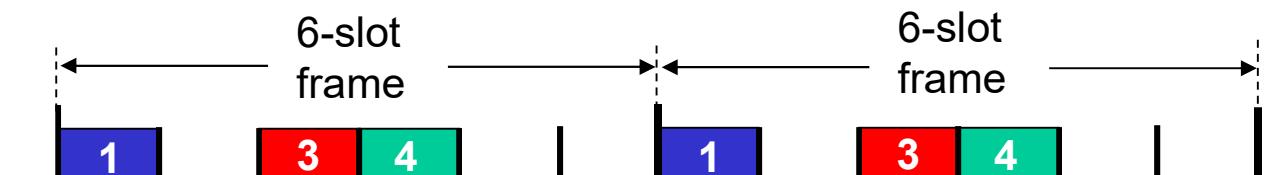
three broad classes:

- *channel partitioning*
 - divide channel into smaller “pieces” (time slots, frequency, code)
 - allocate piece to node for exclusive use
- *random access*
 - channel not divided, allow collisions
 - “recover” from collisions
- *“taking turns”*
 - nodes take turns, but nodes with more to send can take longer turns

Channel partitioning MAC protocols: TDMA

TDMA: time division multiple access

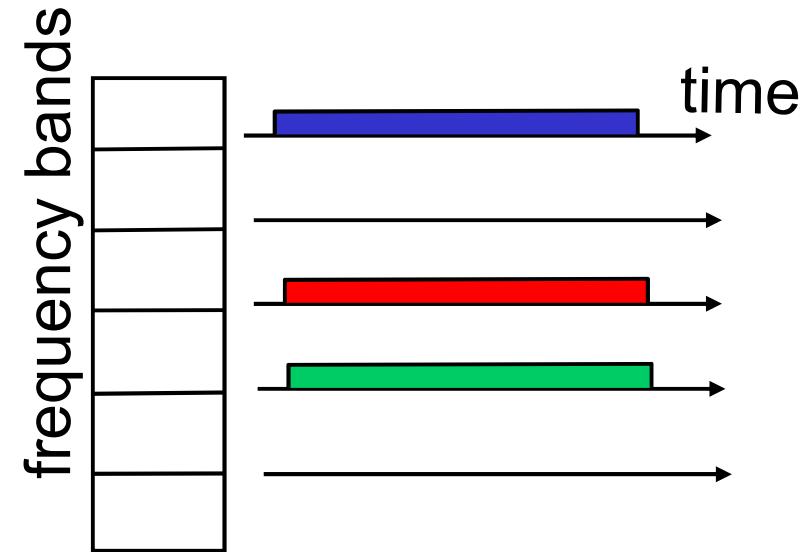
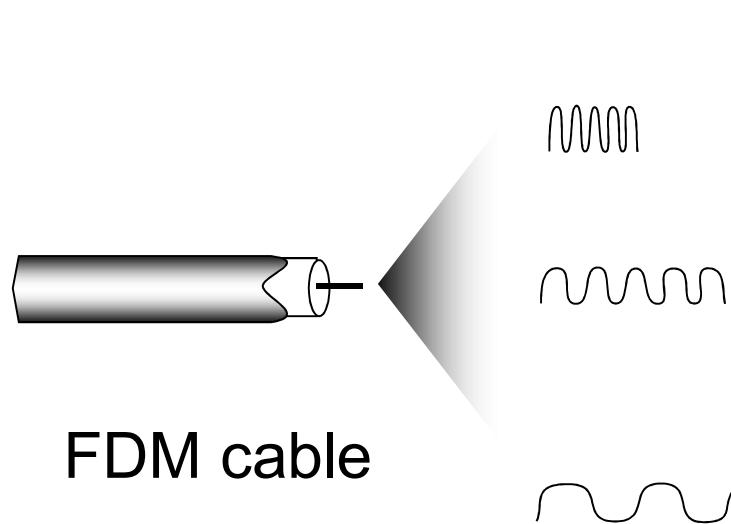
- access to channel in "rounds"
- each station gets fixed length slot (length = packet transmission time) in each round
- unused slots go idle
- example: 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle



Channel partitioning MAC protocols: FDMA

FDMA: frequency division multiple access

- channel spectrum divided into frequency bands
- each station assigned fixed frequency band
- unused transmission time in frequency bands go idle
- example: 6-station LAN, 1,3,4 have packet to send, frequency bands 2,5,6 idle



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 (not an acronym, means “hello” in Hawaii)
 - ALOHA
 - CSMA, CSMA/CD, CSMA/CA
 - Carrier Sense Multiple Access (collision detection/collision avoidance)

Slotted ALOHA

assumptions:

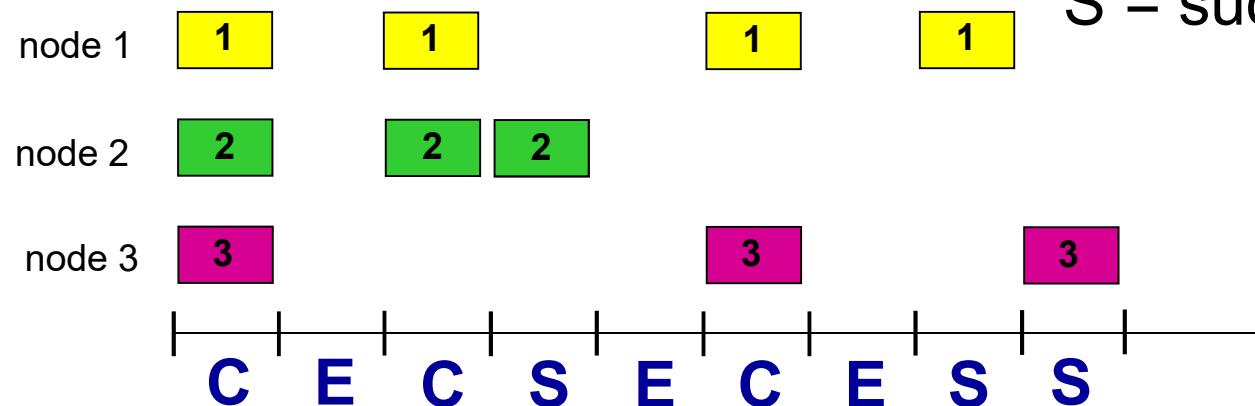
- all frames same size
- time divided into equal size slots (time to transmit 1 frame)
- nodes start to transmit only slot beginning
- nodes are synchronized
- if 2 or more nodes transmit in slot, all nodes detect collision

operation:

- when node obtains fresh frame, transmits in next slot
 - *if no collision:* node can send new frame in next slot
 - *if collision:* node retransmits frame in each subsequent slot with prob. p until success

Slotted ALOHA

C = collision
E = empty
S = successfully sent



Pros:

- single active node can continuously transmit at full rate of channel
- highly decentralized: only slots in nodes need to be in sync
- simple

Cons:

- collisions, wasting slots
- idle slots
- nodes may be able to detect collision in less than time to transmit packet
- clock synchronization

Slotted ALOHA: efficiency

efficiency: long-run fraction of successful slots (many nodes, all with many frames to send)

- suppose: N nodes with many frames to send, each transmits in slot with probability p
- prob that given node has success in a slot = $p(1-p)^{N-1}$
- prob that *any* node has a success = $Np(1-p)^{N-1}$

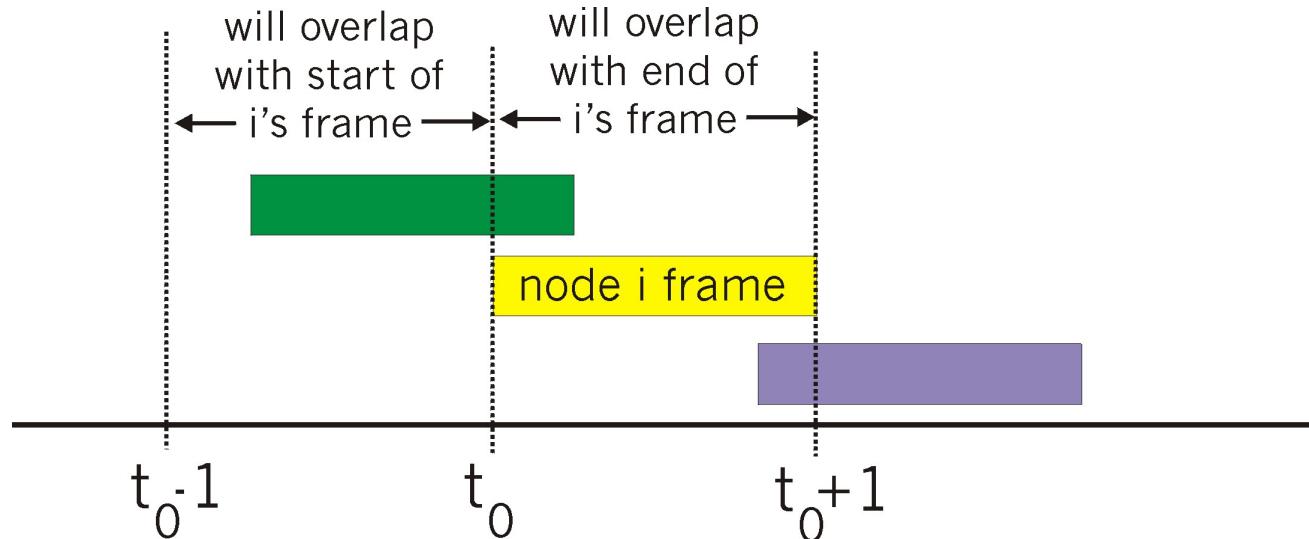
- max efficiency: find p^* that maximizes $Np(1-p)^{N-1}$
- for many nodes, take limit of $Np^*(1-p^*)^{N-1}$ as N goes to infinity, gives:
max efficiency = $1/e = .37$

at best: channel used for useful transmissions 37% of time!

!

Pure (unslotted) ALOHA

- unslotted Aloha: simpler, no synchronization
- when frame first arrives
 - transmit immediately
- collision probability increases:
 - frame sent at t_0 collides with other frames sent in $[t_0 - l, t_0 + l]$



Pure ALOHA efficiency

$P(\text{success by given node}) = P(\text{node transmits}) \cdot$

$P(\text{no other node transmits in } [t_0-l, t_0] \cdot$

$P(\text{no other node transmits in } [t_0, t_0+l]$

$$= p \cdot (1-p)^{N-l} \cdot (1-p)^{N-l}$$

$$= p \cdot (1-p)^{2(N-l)}$$

... choosing optimum p and then letting $N \rightarrow \infty$

$$= 1/(2e) = .18$$

even worse than slotted Aloha!

CSMA (carrier sense multiple access)

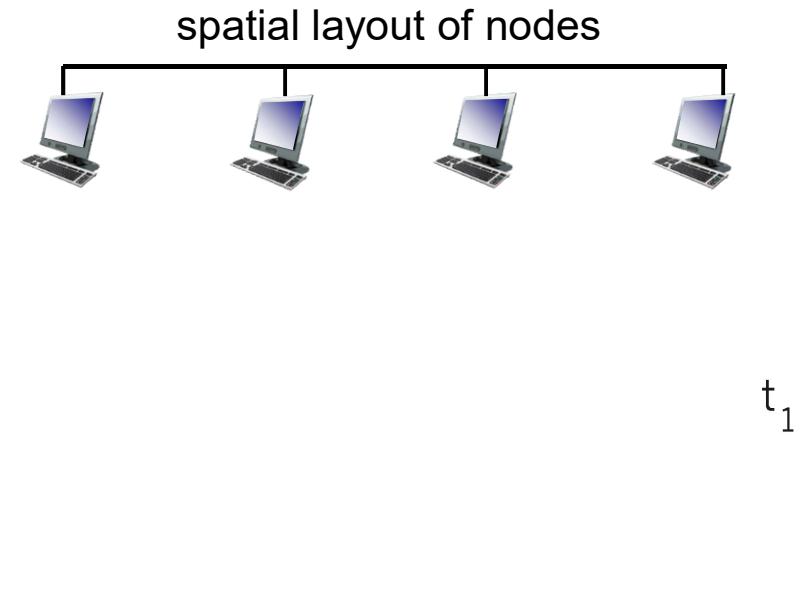
CSMA: listen before transmit:

if channel sensed idle: transmit entire frame

- if channel sensed busy, defer transmission
- human analogy: don't interrupt others!

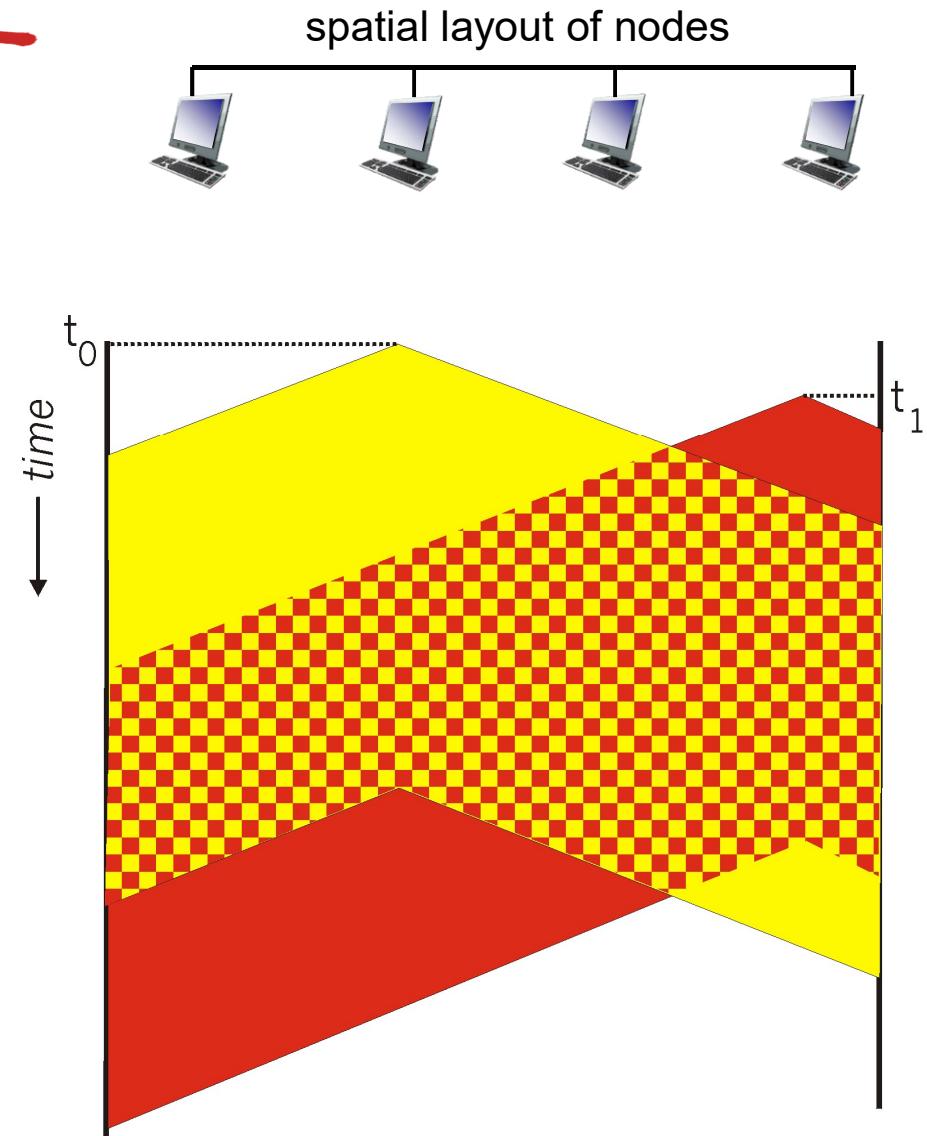
CSMA collisions

- **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 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 determining collision probability

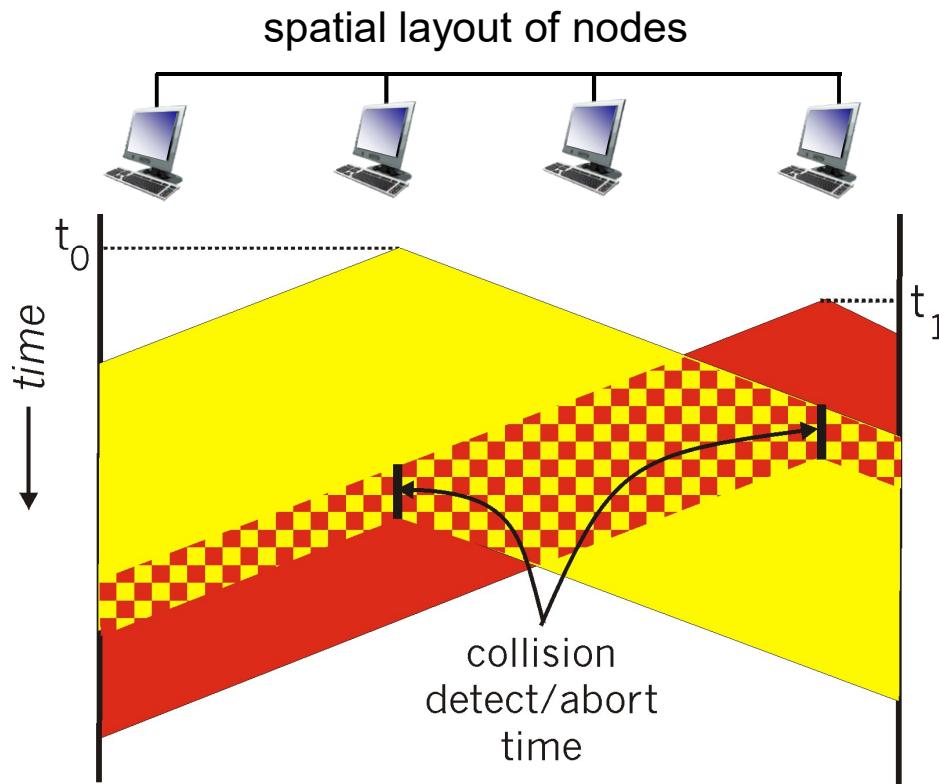


CSMA/CD (collision detection)

CSMA/CD: carrier sensing, deferral (backs off transmission) 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, received signals
 - difficult in wireless LANs: received signal strength overwhelmed by local transmission strength
- human analogy: polite talk where people wait for each other

CSMA/CD (collision detection)



Ethernet CSMA/CD algorithm

1. Network Interface Card (NIC) receives datagram from network layer, creates frame
2. If NIC senses channel idle, starts frame transmission. If NIC senses channel busy, waits until channel idle, then transmits.
3. If NIC transmits entire frame without detecting another transmission, NIC is done with frame !
4. If NIC detects another transmission while transmitting, aborts and sends jam signal
5. After aborting, NIC enters *binary (exponential) backoff*:
 - after m th collision, NIC chooses K at random from $\{0, 1, 2, \dots, 2^m - 1\}$. NIC waits $K \cdot 512$ bit times, returns to Step 2
 - longer backoff interval with more collisions

CSMA/CD efficiency

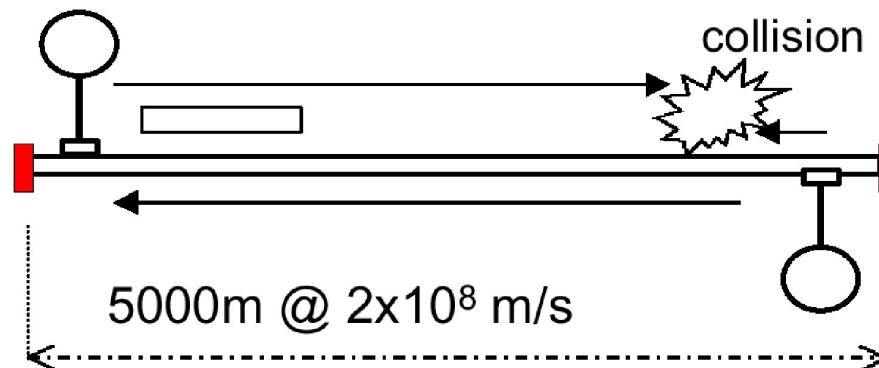
- T_{prop} = max prop delay between 2 nodes in LAN
- t_{trans} = time to transmit max-size frame

$$efficiency = \frac{1}{1 + 5t_{prop}/t_{trans}}$$

- efficiency goes to 1
 - as t_{prop} goes to 0
 - as t_{trans} goes to infinity
- better performance than ALOHA: and simple, cheap, decentralized!

CSMA/CD: Frame size

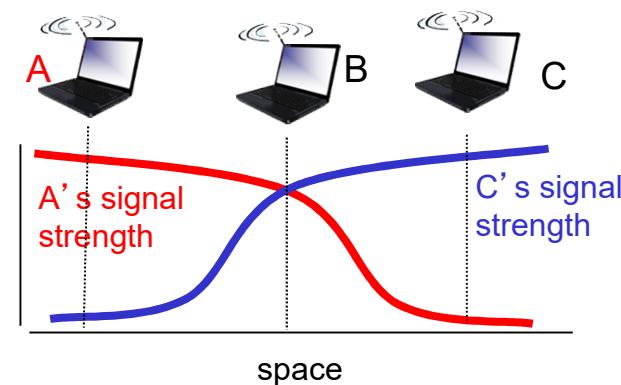
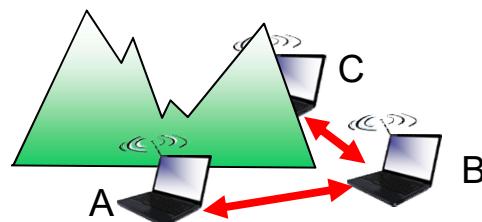
- Collision Detection
- Collision Window
 - Related to end-to-end propagation delay
- Minimum packet size must be greater than collision window
 - For 5000m (5km) bus @ 10 Mbits/sec,
 - $RTT = 2 \times 5000m / 2 \times 10^8 \text{ m/s} = 0.00005\text{s}$
 - min frame size = $0.00005\text{s} \times 10,000,000 \text{ bits/sec}$
 $= (500 \text{ bits}) \sim 64 \text{ octets}$



$2 \times 10^8 \text{ m/s} =$
2/3 speed of light =
speed of signal
in copper wire

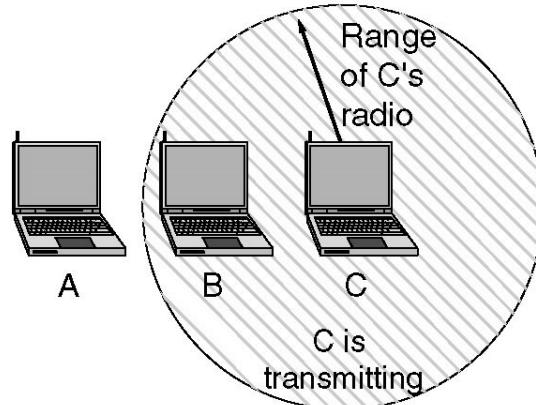
CSMA/CA (Collision Avoidance)

- avoid collisions: 2^+ nodes transmitting at same time
- 802.11: CSMA - sense before transmitting
 - don't collide with ongoing transmission by other node
- 802.11: no collision detection!
 - difficult to receive (sense collisions) when transmitting due to weak received signals (fading)
 - can't sense all collisions in any case: hidden terminal, fading
 - goal: *avoid collisions*: CSMA/C(ollision)A(voidance)



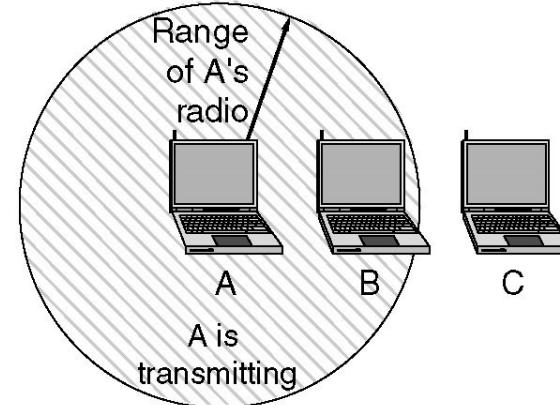
Hidden and Exposed Station problems

A wants to send to B
but cannot hear that
B is busy



(a)

B wants to send to C
but mistakenly thinks
the transmission will fail



(b)

(a) The hidden station problem.

A and C are hidden from each other

(b) The exposed station problem.

B is exposed to transmission from A

IEEE 802.11 MAC Protocol: CSMA/CA

802.11 sender

1 if sense channel idle for **DIFS** then

 transmit entire frame (no CD)

2 if sense channel busy then

 start random backoff time

 timer counts down while channel idle

 transmit when timer expires

 if no ACK, increase random backoff interval,
 repeat 2

802.11 receiver

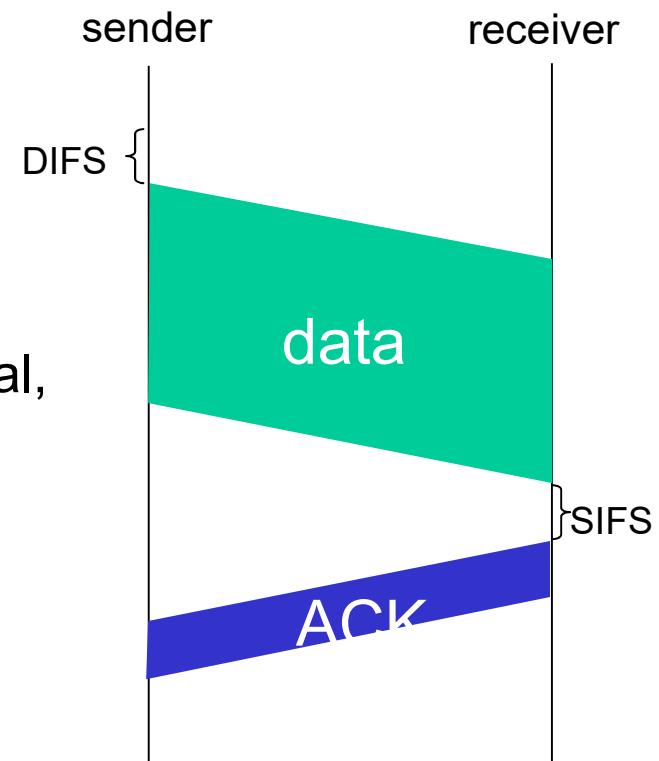
- if frame received OK

 return ACK after **SIFS** (ACK needed due to
 hidden terminal problem)

DCF = Distributed Coordination Function

DIFS = DCF InterFrame Space

SIFS = Short InterFrame Space



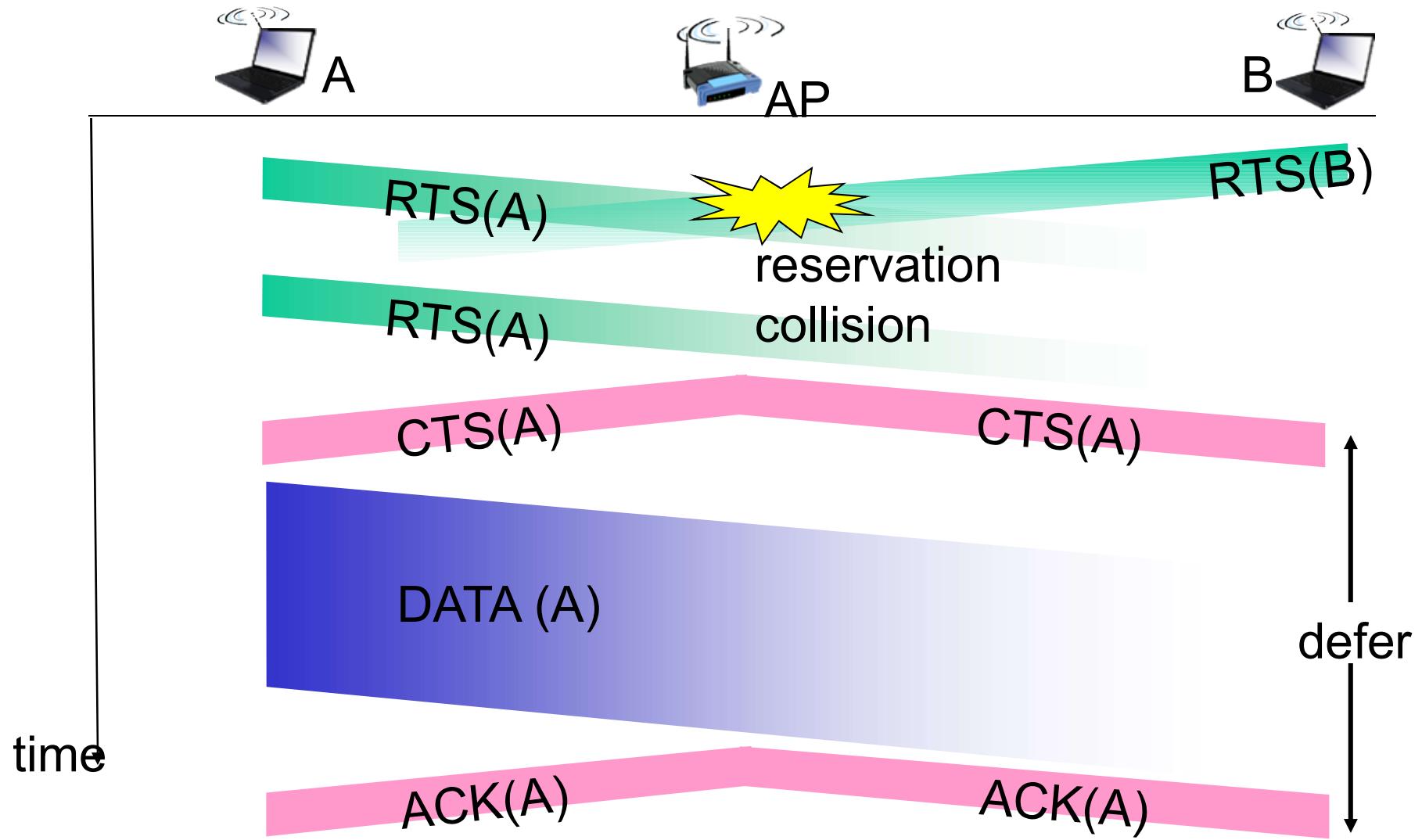
Avoiding collisions (more)

idea: allow sender to “reserve” channel rather than random access of data frames: avoid collisions of long data frames

- sender first transmits *small* request-to-send (RTS) packets to base station (BS) using CSMA
 - RTSs may still collide with each other (but they’re short)
- BS broadcasts **clear-to-send** CTS in response to RTS
- CTS heard by all nodes
 - sender transmits data frame
 - other stations defer transmissions

*avoid data frame collisions completely
using small reservation packets!*

Collision Avoidance: RTS-CTS exchange



“Taking turns” MAC protocols

channel partitioning MAC protocols:

- share channel *efficiently* and *fairly* at high load
- inefficient at low load: delay in channel access, I/N bandwidth allocated even if only 1 active node!

random access MAC protocols

- efficient at low load: single node can fully utilize channel
- high load: collision overhead

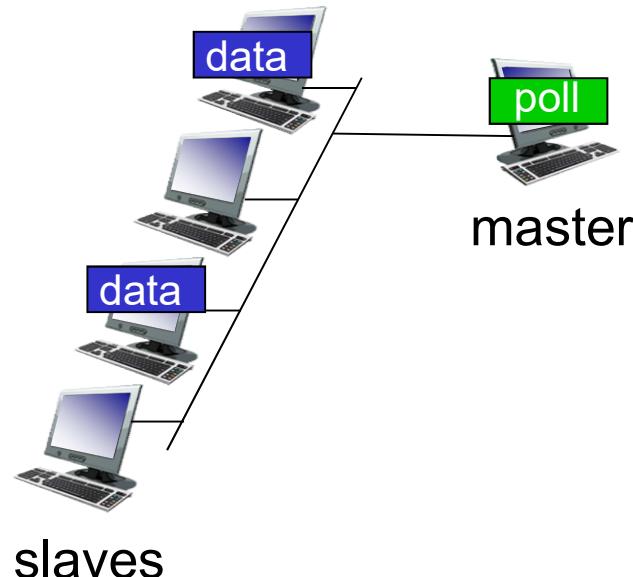
“taking turns” protocols

look for best of both worlds!

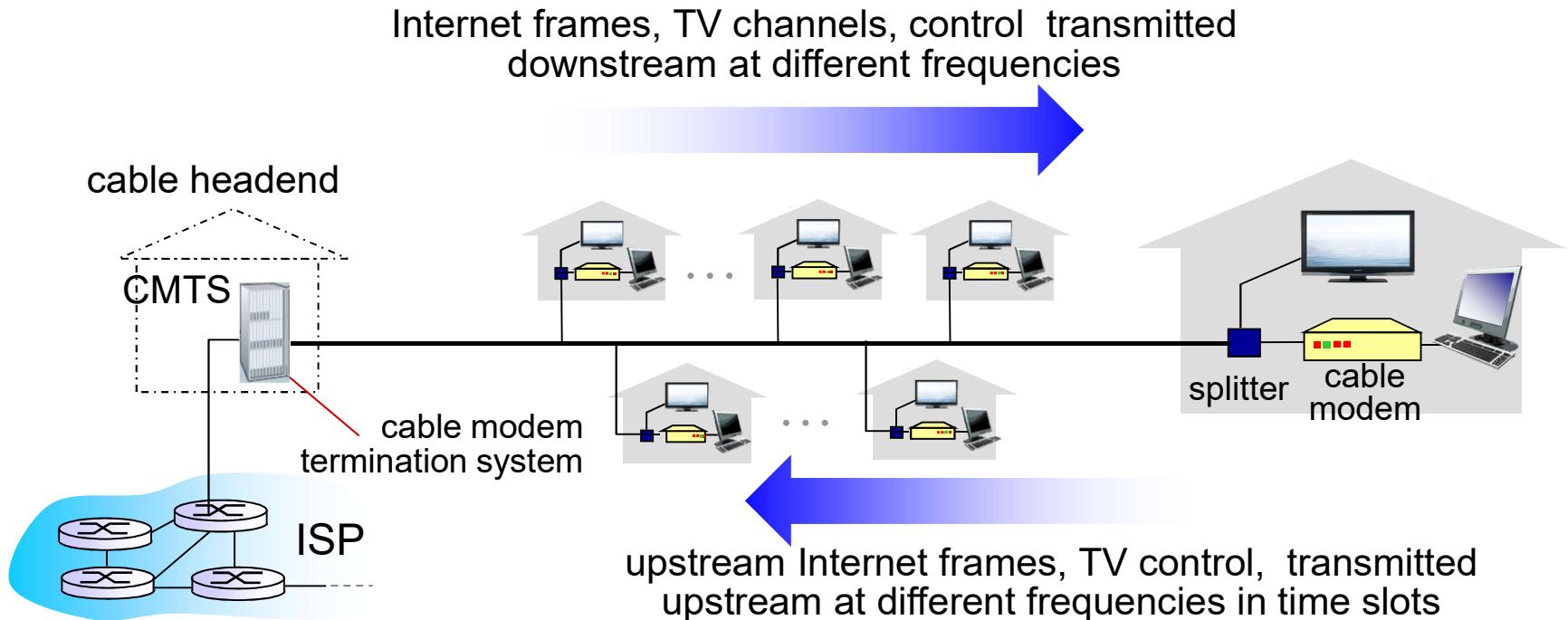
“Taking turns” MAC protocols

polling:

- master node “invites” slave nodes to transmit in turn
- typically used with “dumb” slave devices
- concerns:
 - polling overhead
 - latency
 - single point of failure (master)

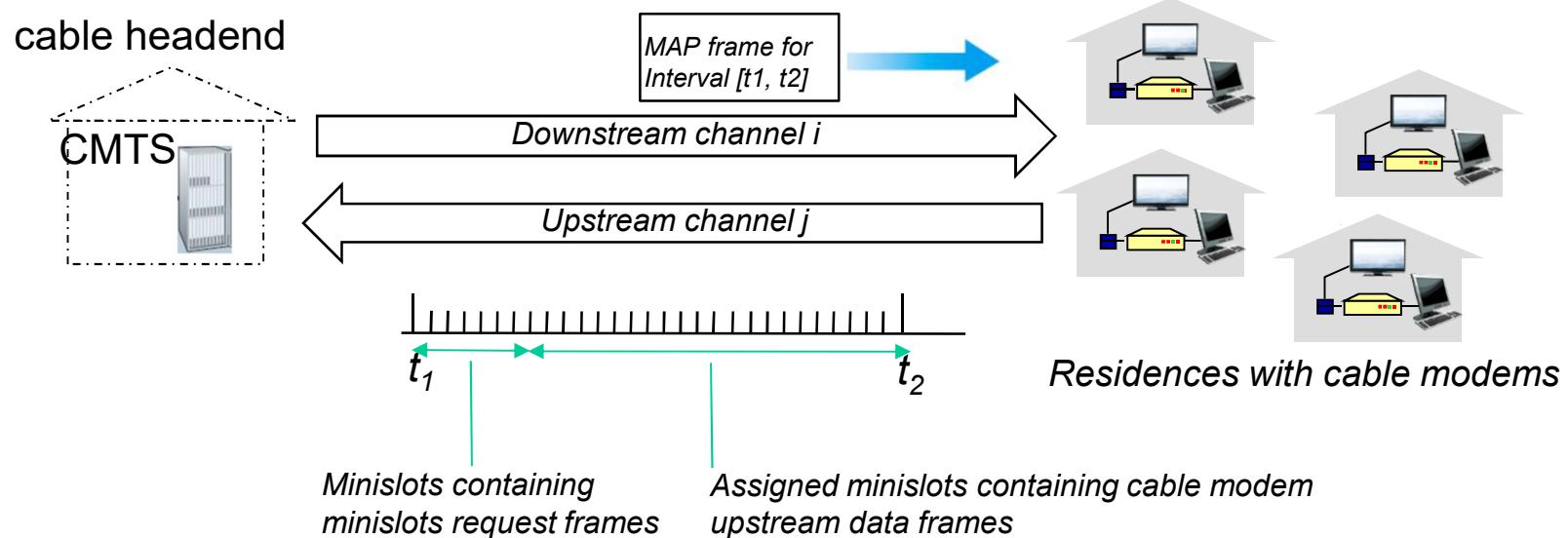


Cable access network



- **multiple 40Mbps downstream (broadcast) channels**
 - single CMTS transmits into channels
- **multiple 30 Mbps upstream channels**
 - **multiple access:** all users contend for certain upstream channel time slots (others assigned)

Cable access network



DOCSIS: data over cable service interface spec

- FDM over upstream, downstream frequency channels
- TDM upstream: some slots assigned, some have contention
 - downstream MAP frame: assigns upstream slots
 - request for upstream slots (and data) transmitted random access (binary backoff) in selected slots

What have we learned?

- *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
 - Bluetooth, Fibre Distributed Data Interface (FDDI)

Link layer, LANs: outline

6.1 introduction, services

6.2 error detection,
correction

6.3 multiple access
protocols

6.4 LANs

- addressing, ARP
- Ethernet
- WiFi
- switches
- VLANs

6.5 link virtualization:
MPLS

6.6 data center
networking

6.7 a day in the life of a
web request

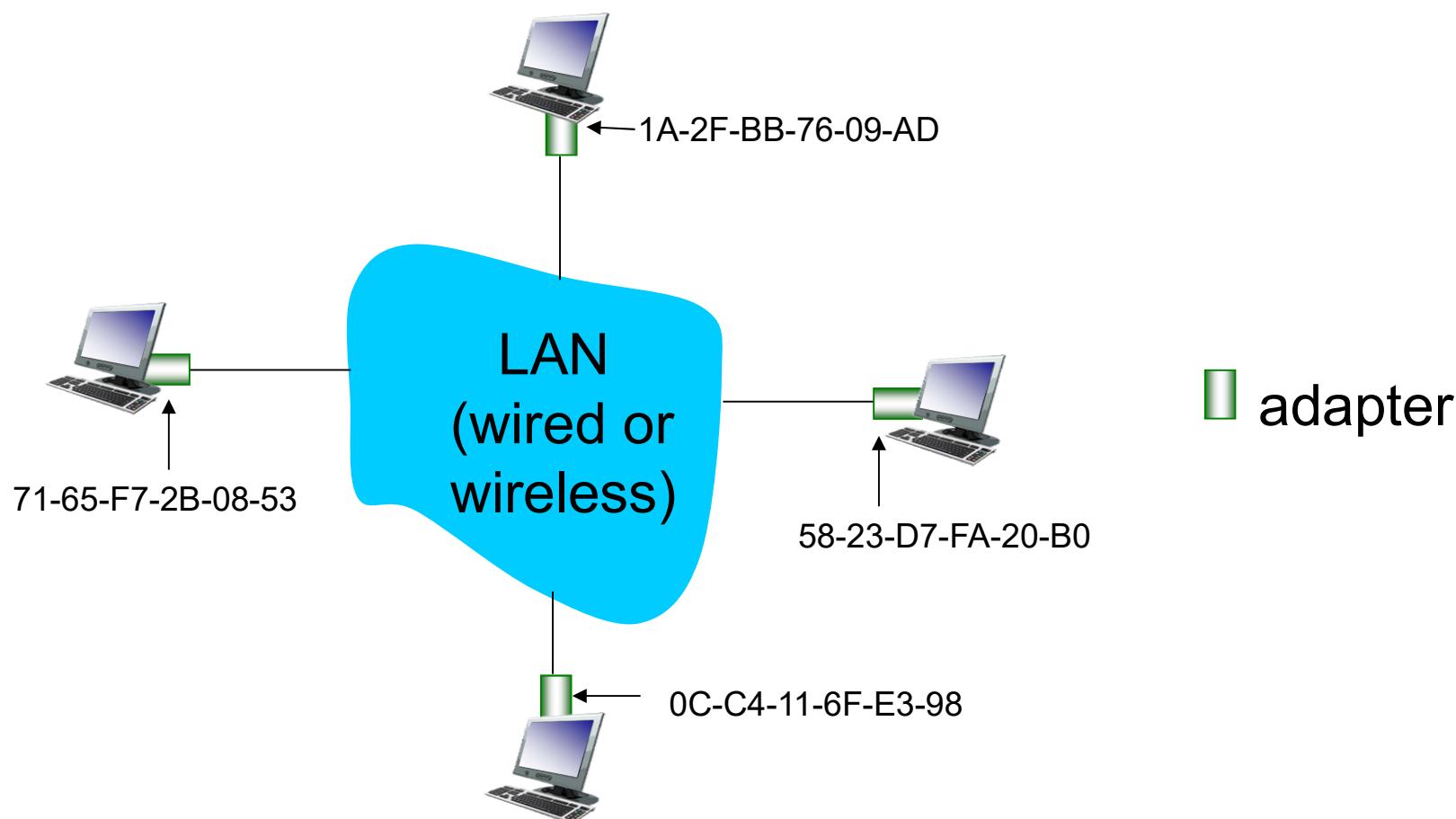
MAC addresses and ARP

- 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 settable
 - e.g.: IA-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

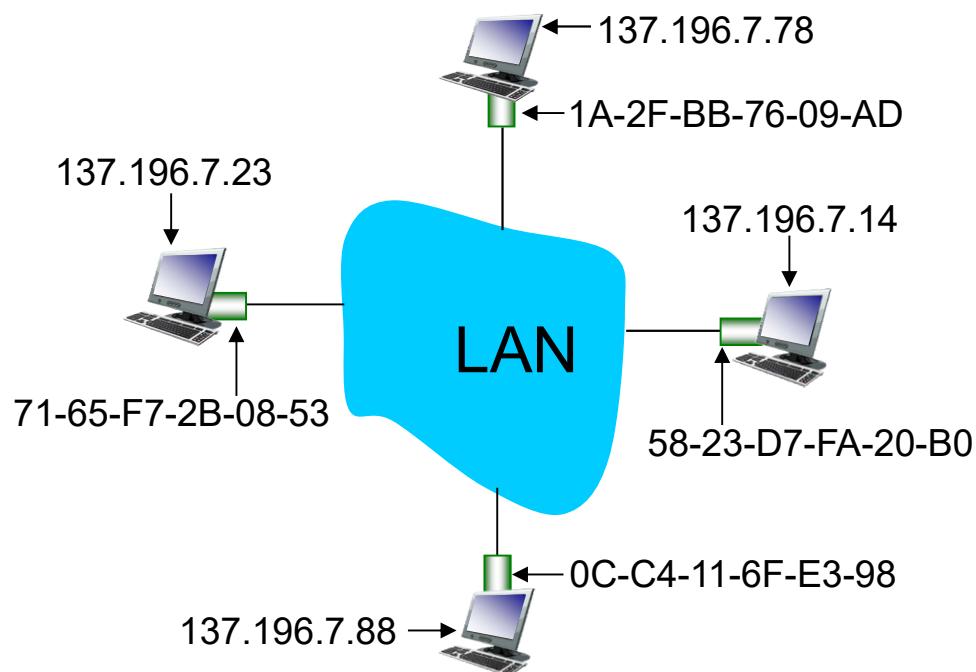


LAN addresses (more)

- 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)

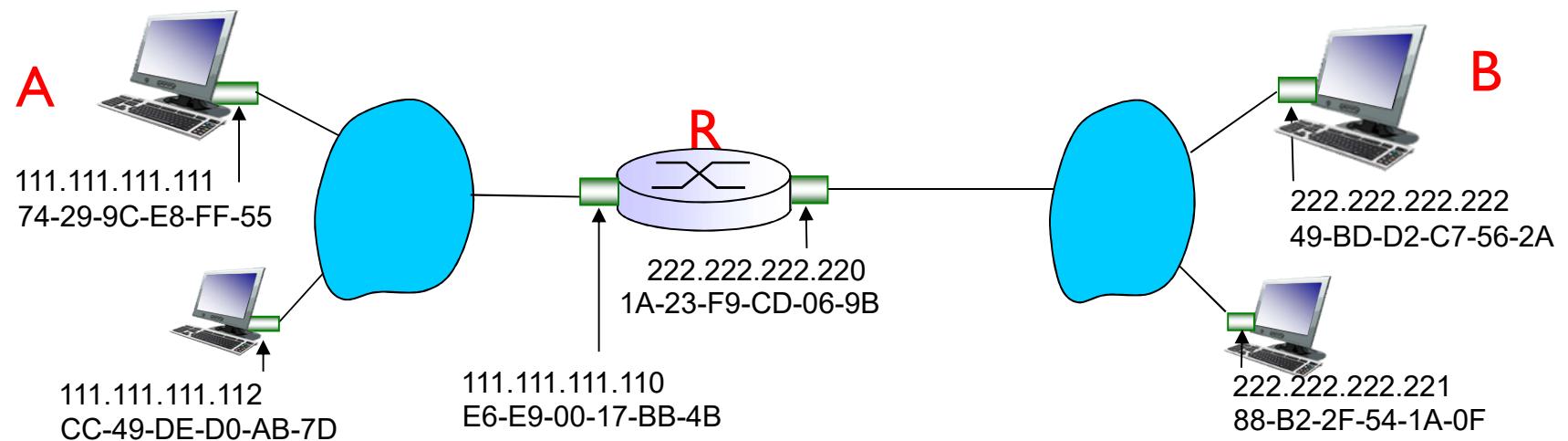
ARP protocol: same LAN

- 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

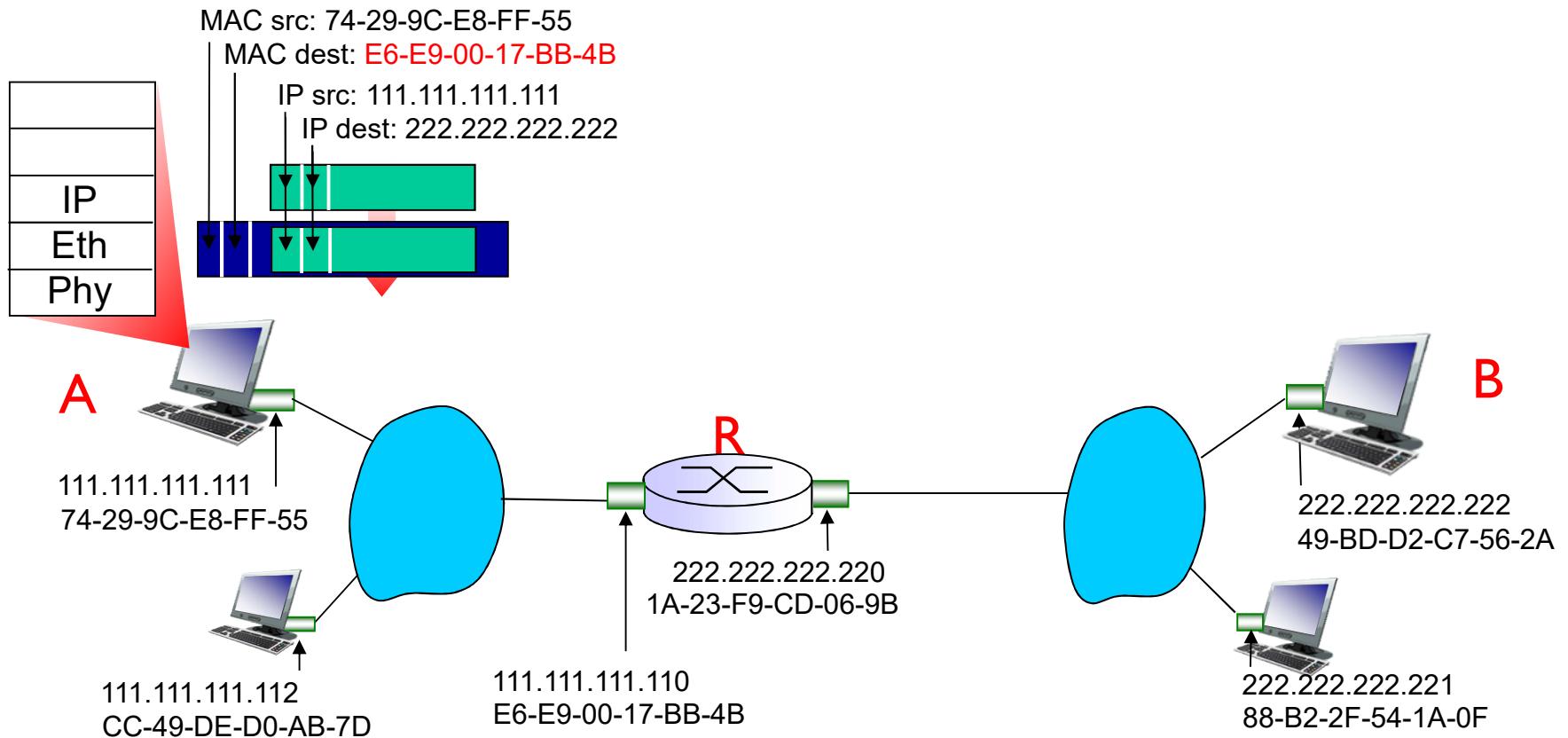
walkthrough: send datagram from A to B via R

- 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?)



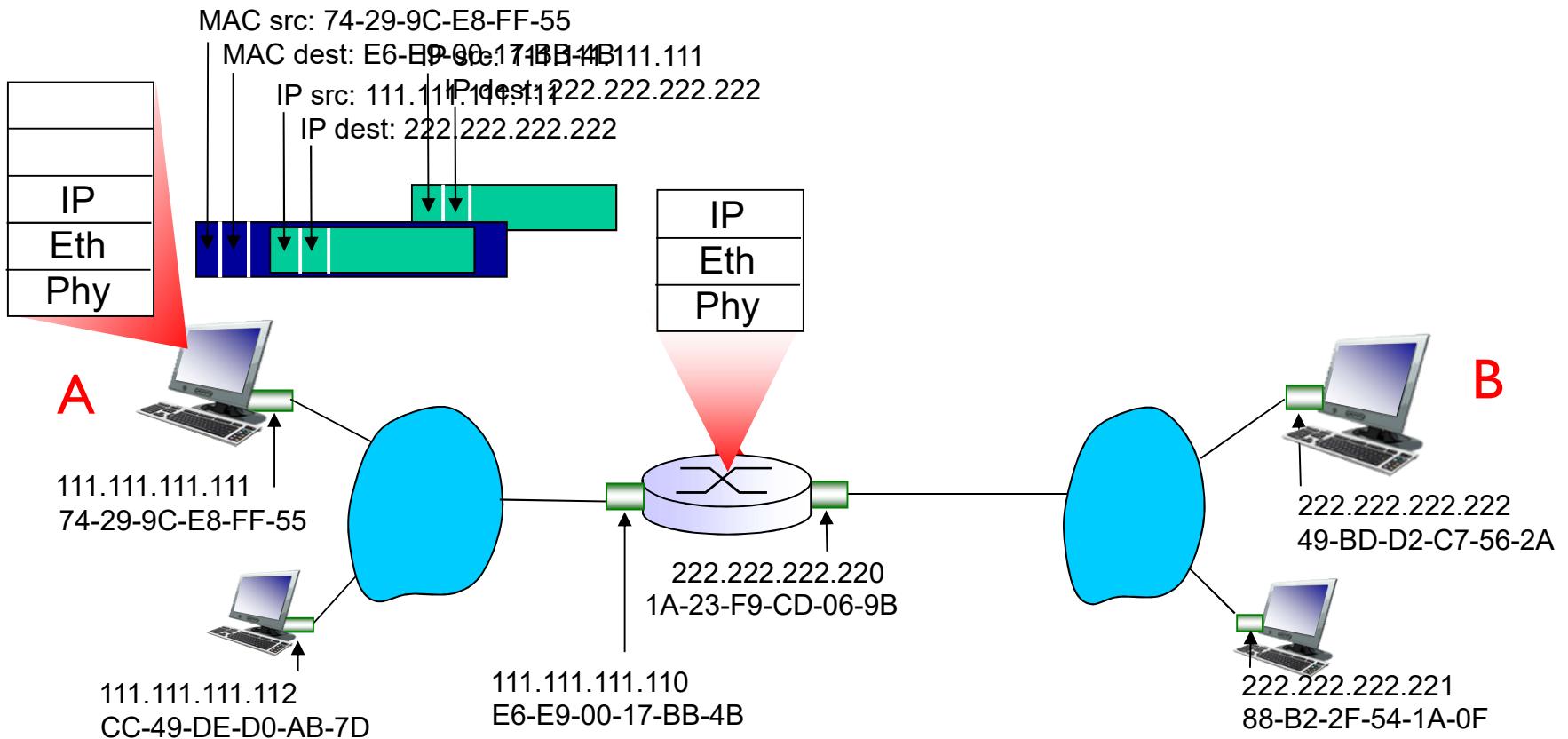
Addressing: routing to another LAN

- 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



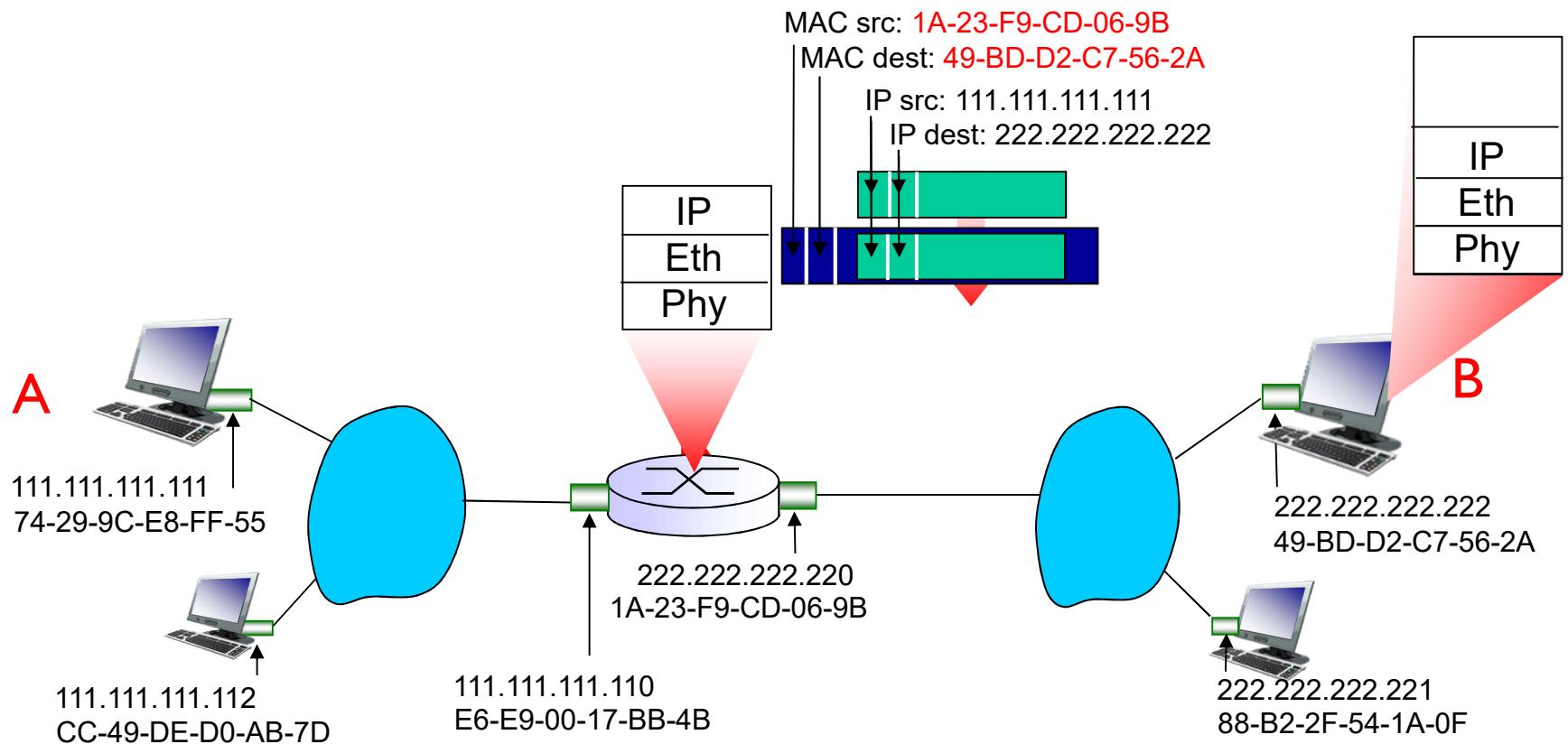
Addressing: routing to another LAN

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



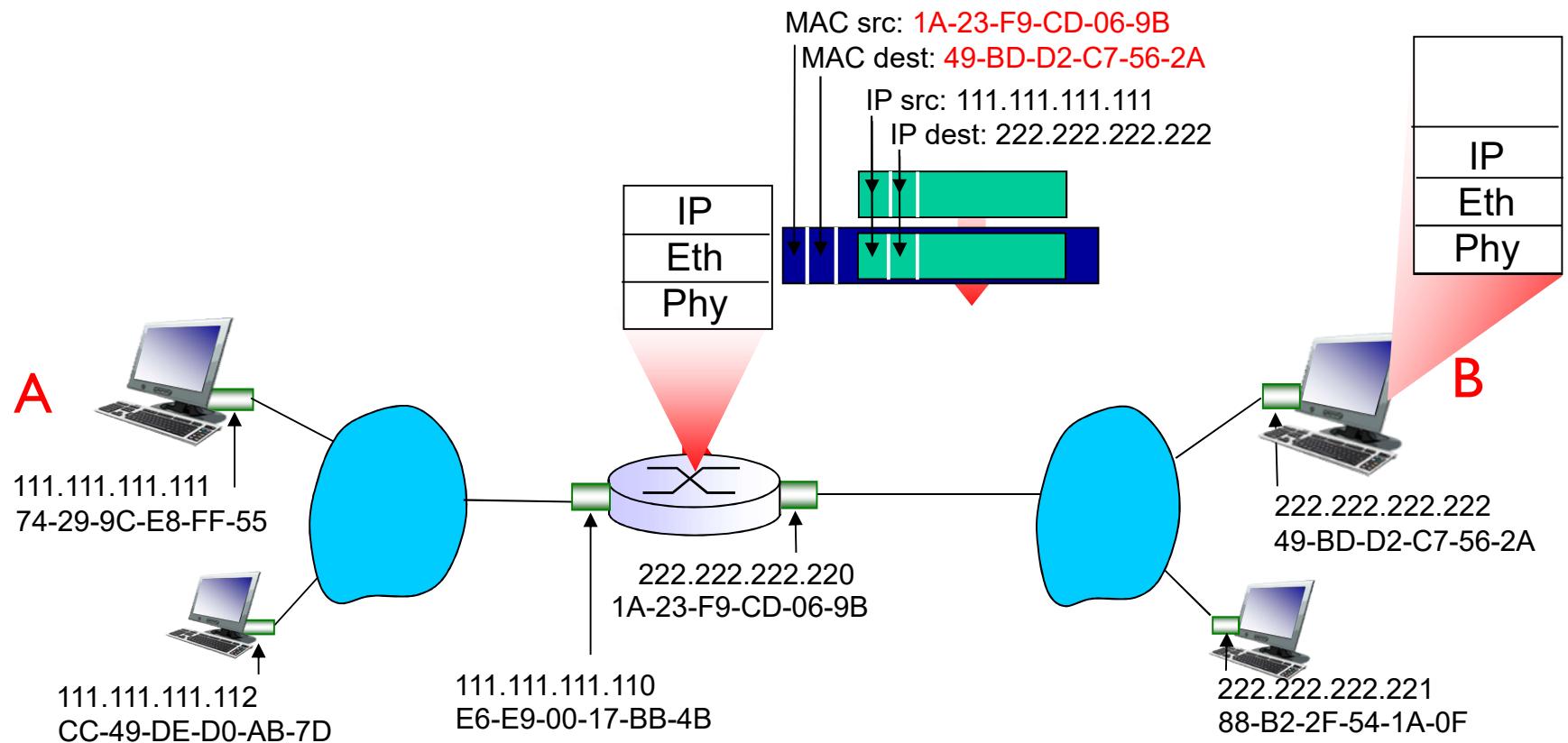
Addressing: routing to another LAN

- R forwards 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



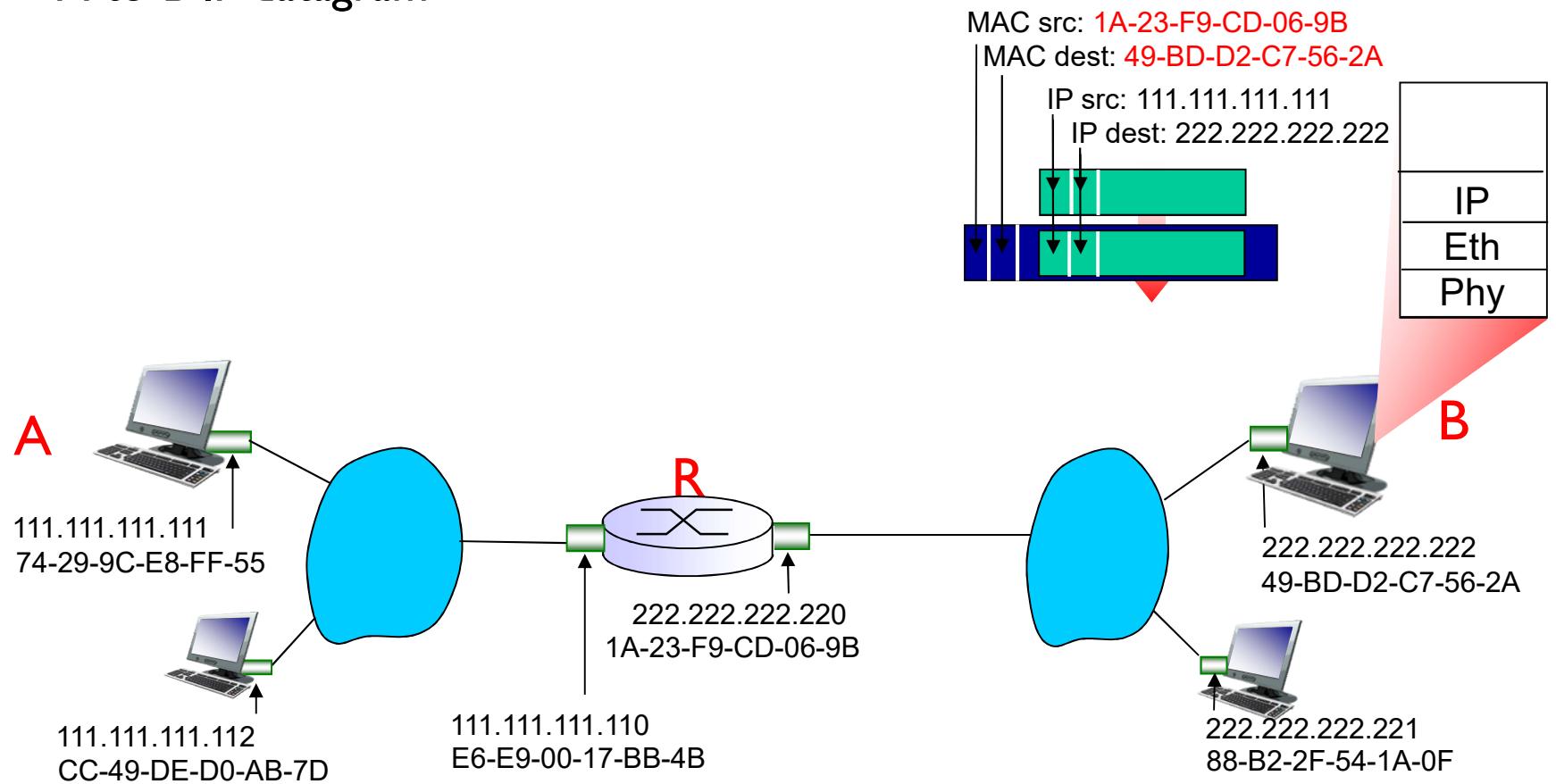
Addressing: routing to another LAN

- 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



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



* 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

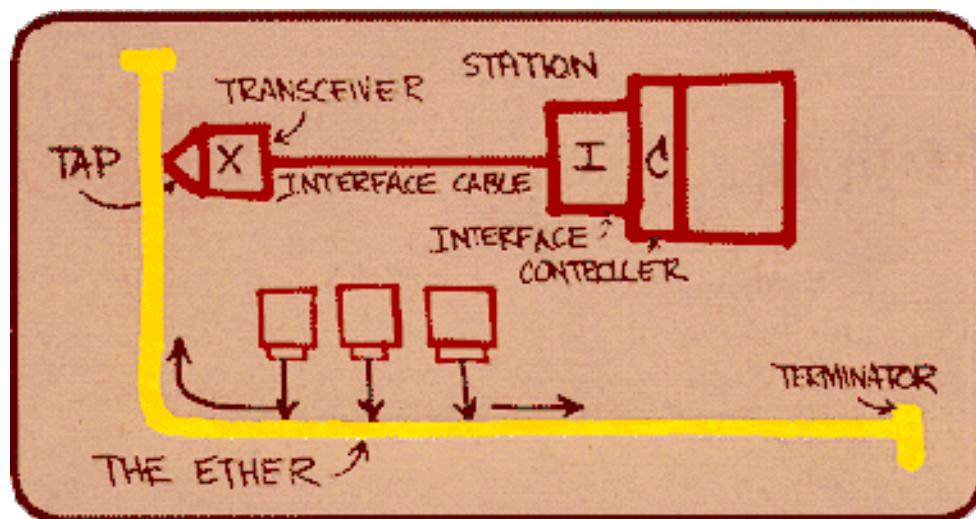
6.6 data center
networking

6.7 a day in the life of a
web request

Ethernet

“dominant” wired LAN technology:

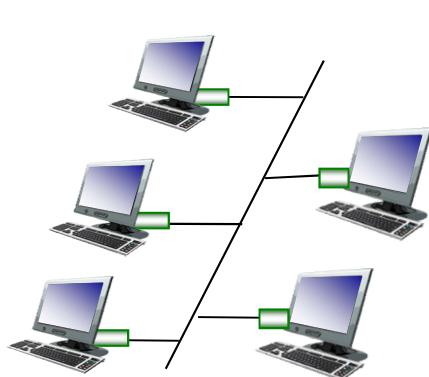
- single chip, multiple speeds (e.g., Broadcom BCM5761)
- first widely used LAN technology
- simpler, cheap
- kept up with speed race: 10 Mbps – 10 Gbps



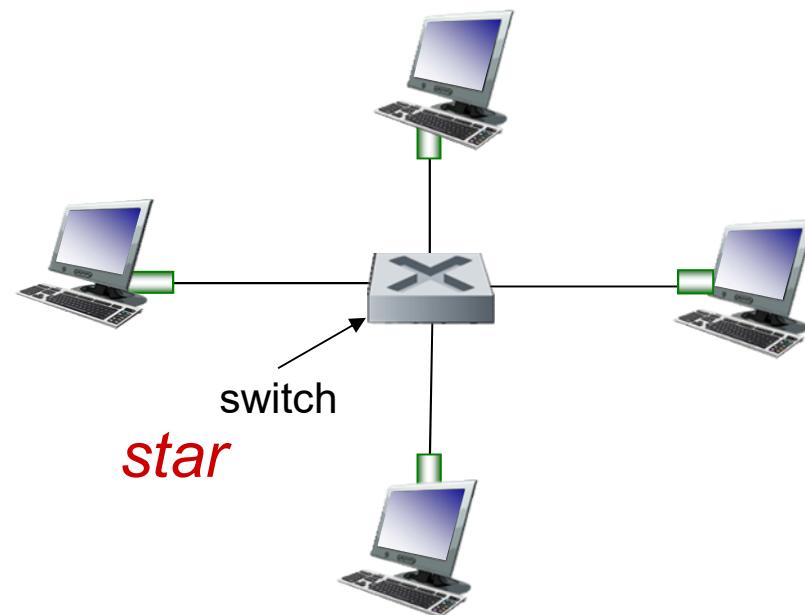
*Metcalfe's Ethernet
sketch*

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



Ethernet frame structure

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

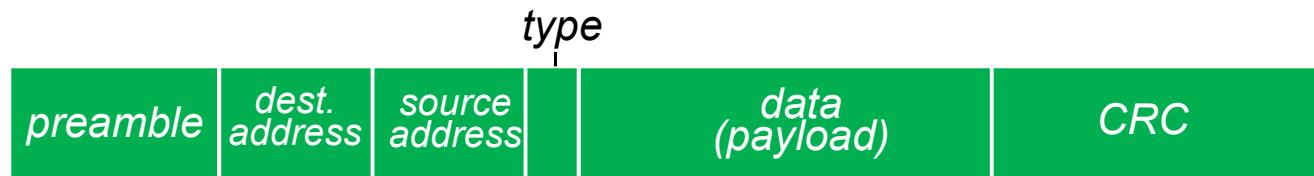


preamble:

- 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

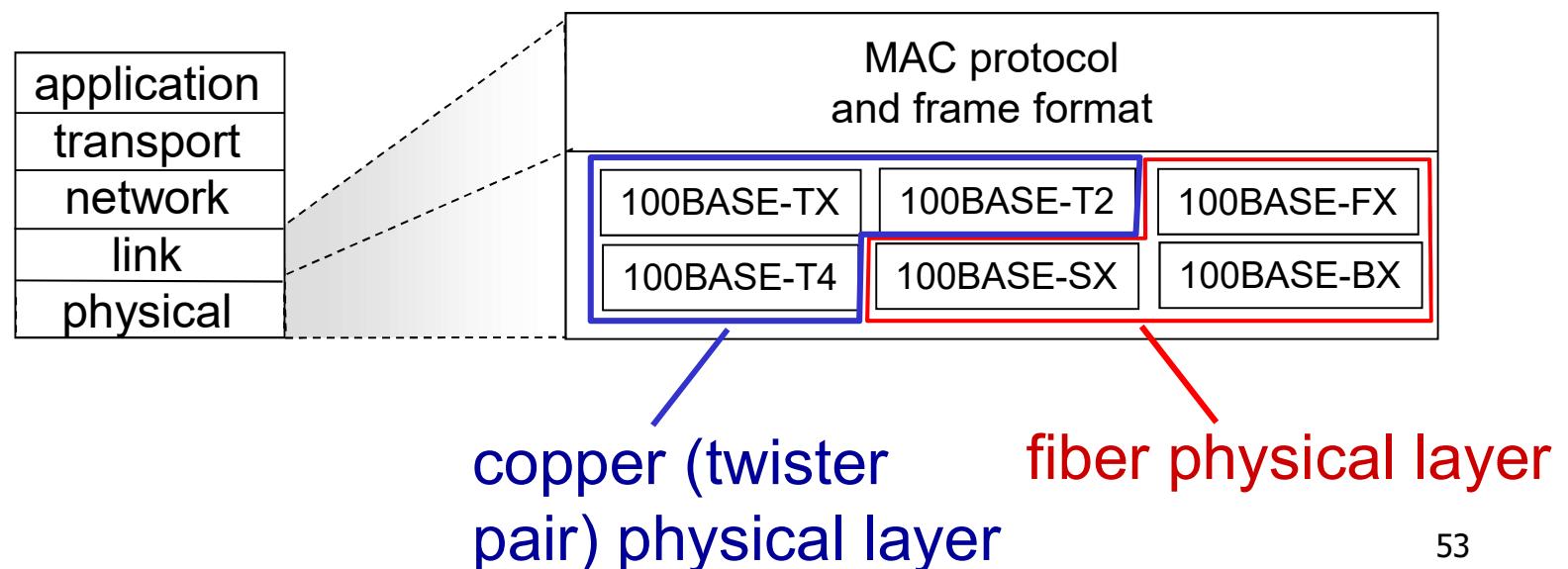


Ethernet: unreliable, connectionless

- ***connectionless***: no handshaking between sending and receiving NICs
- ***unreliable***: receiving NIC doesn't send ACK or NACK 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, 1 Gbps, 10 Gbps, 40 Gbps
 - different physical layer media: fiber, cable



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

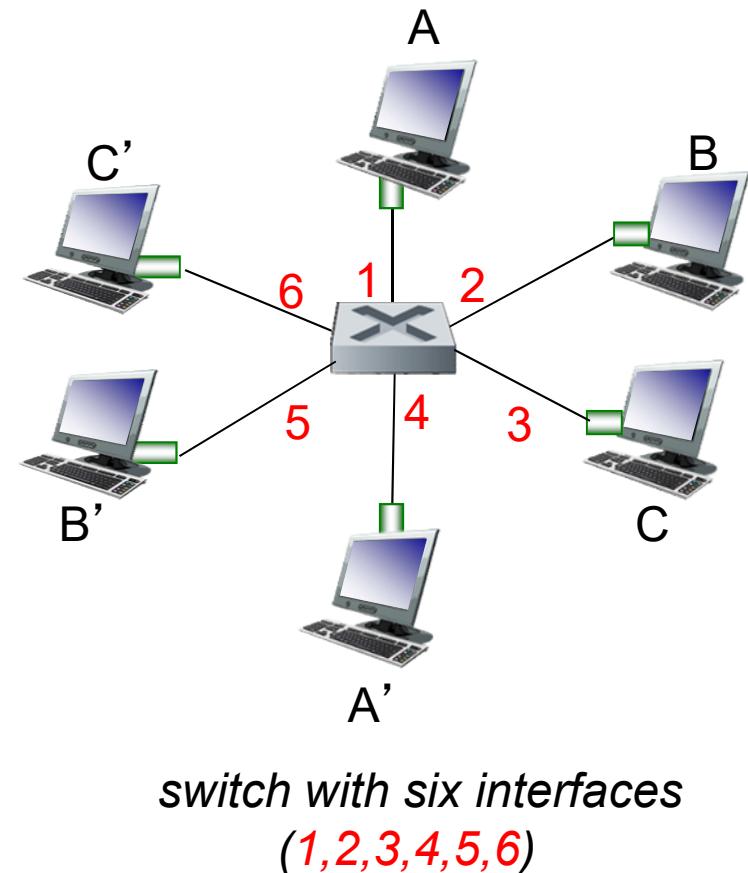
6.7 a day in the life of a
web request

Ethernet 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

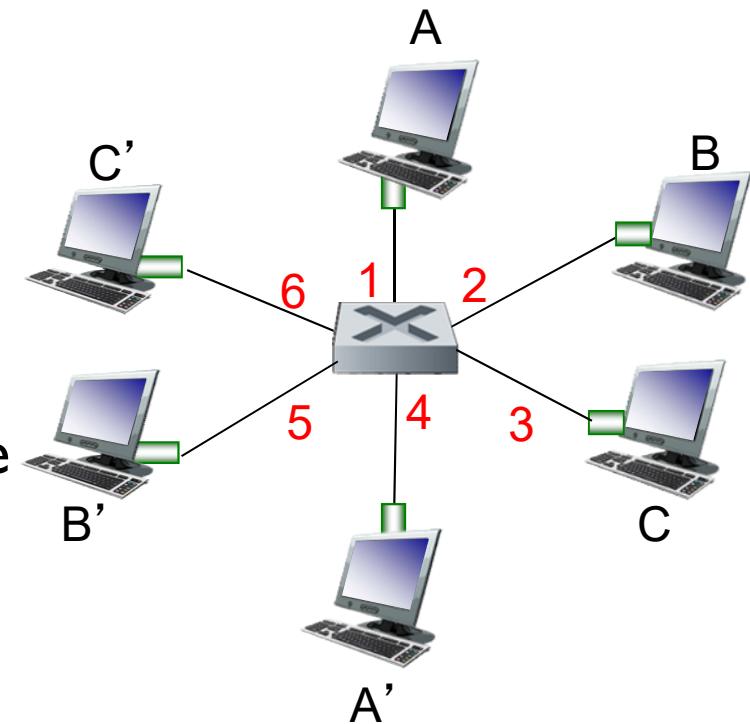
- hosts have dedicated, direct connection to switch
- switches buffer packets
- Ethernet protocol used on each incoming link, but no collisions; full duplex
 - each link is its own collision domain
- **switching:** A-to-A' and B-to-B' can transmit simultaneously, without collisions



Switch 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!



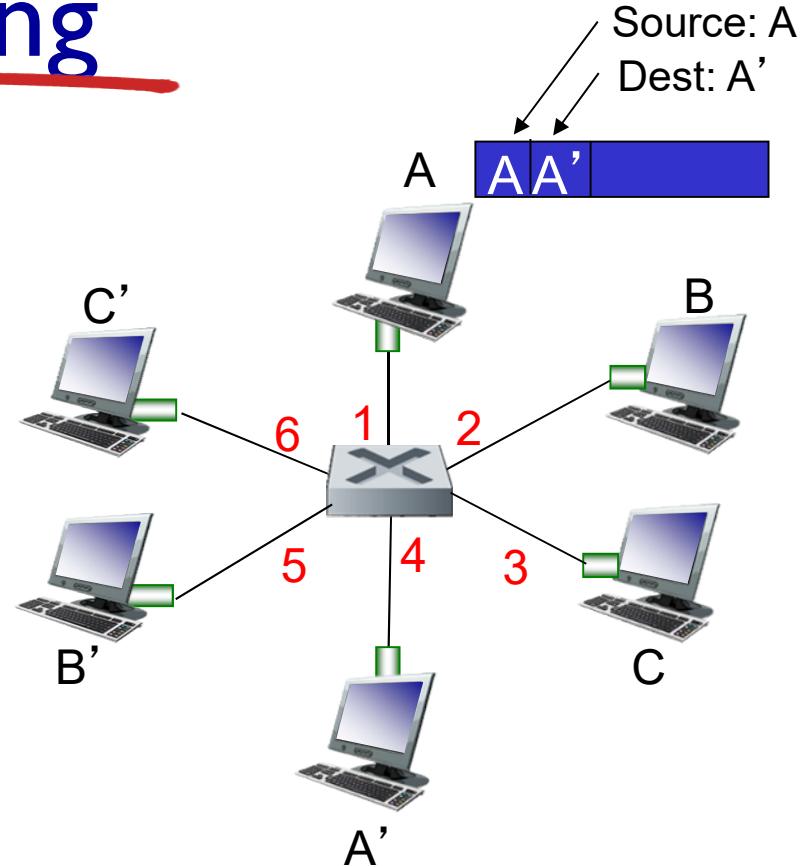
Q: how are entries created, maintained in switch table?

- something like a routing protocol?

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

Switch: self-learning

- 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)*

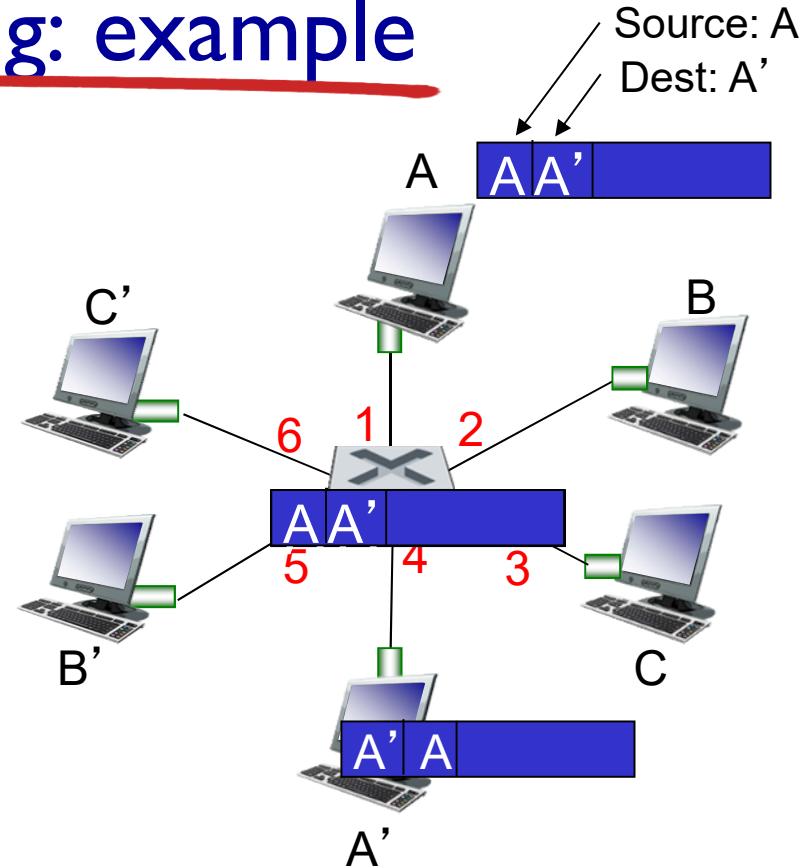
Switch: frame filtering/forwarding

when frame received at switch:

1. record incoming link, 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*

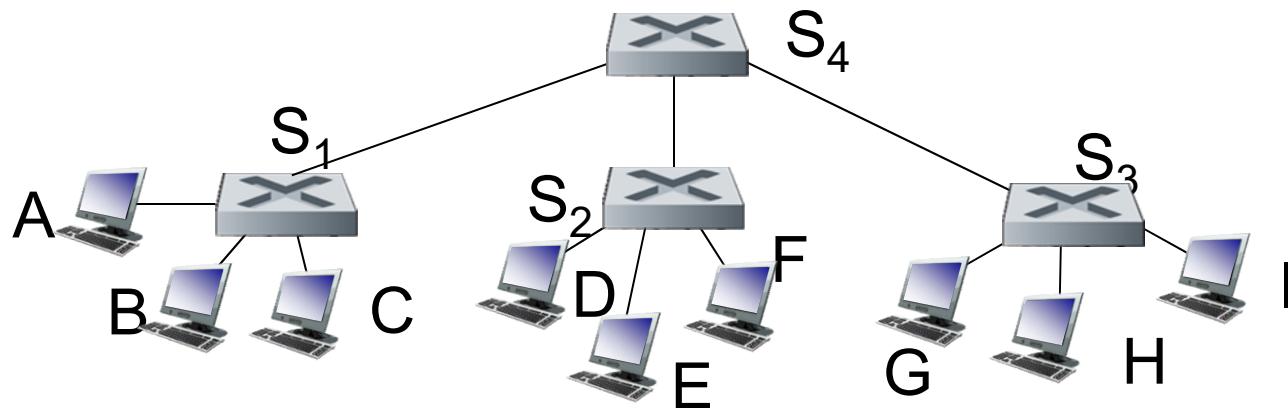


MAC addr	interface	TTL
A	1	60
A'	4	60

*switch table
(initially empty)*

Interconnecting switches

self-learning switches can be connected together:

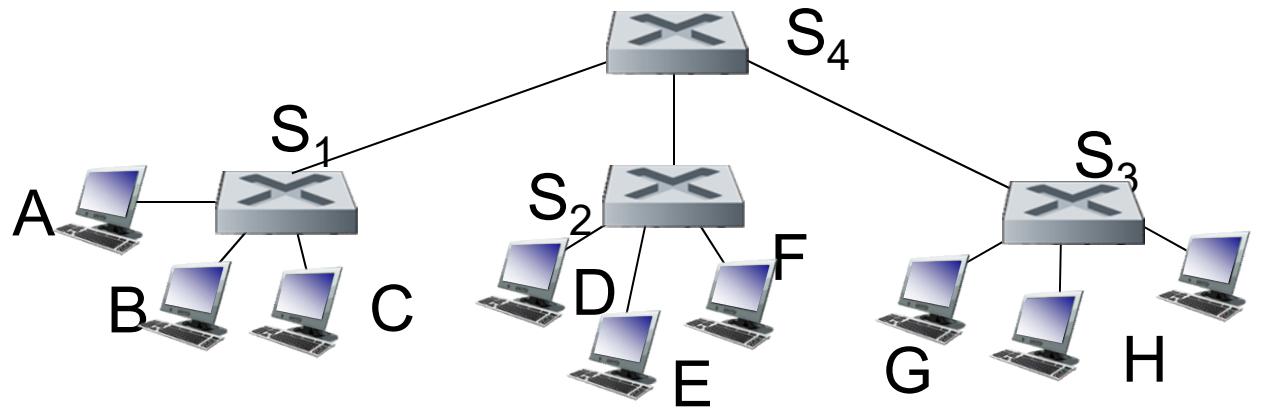


Q: sending from A to G - how does S_1 know to forward frame destined to G via S_4 and S_3 ?

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

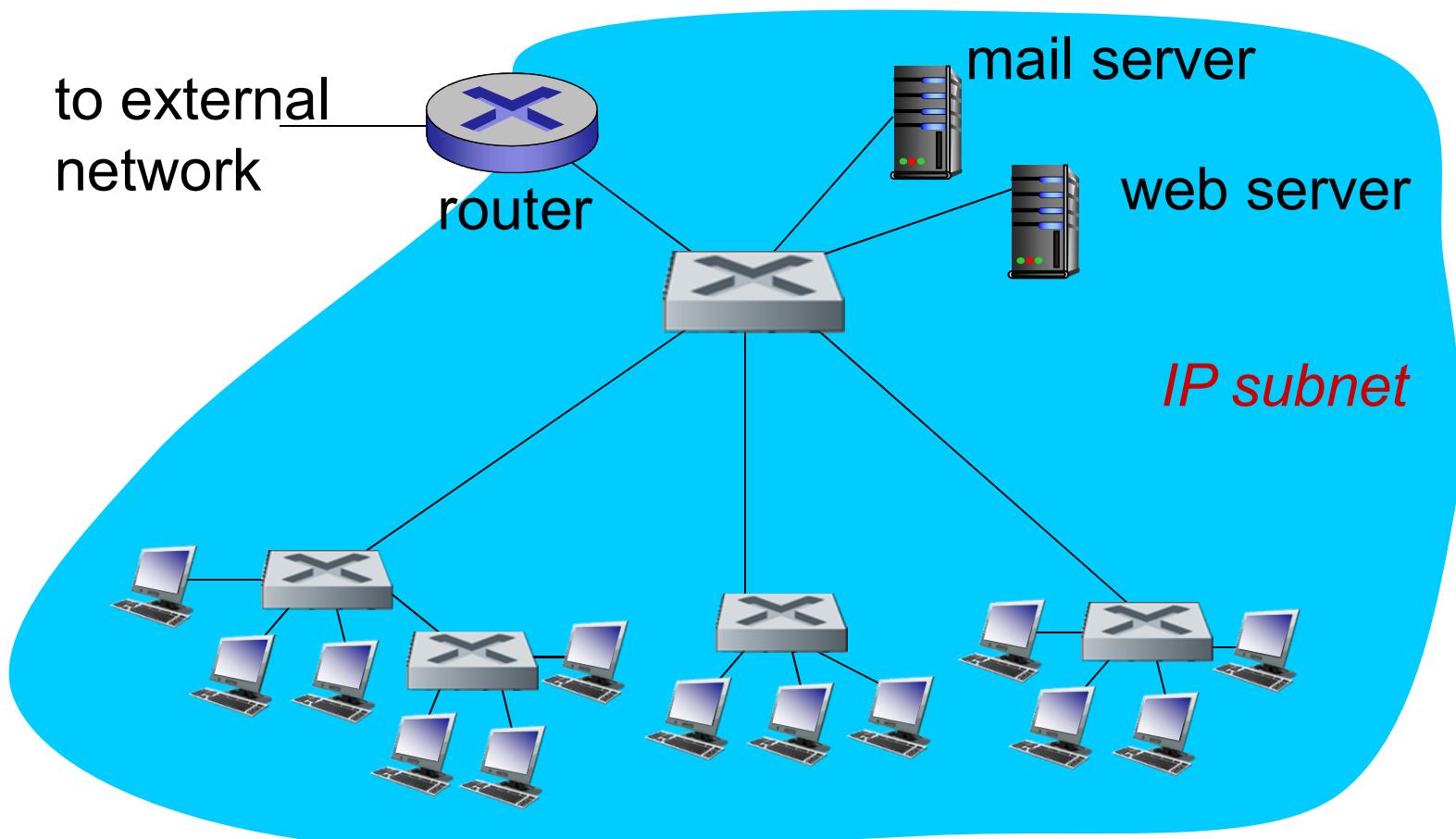
Self-learning multi-switch example

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



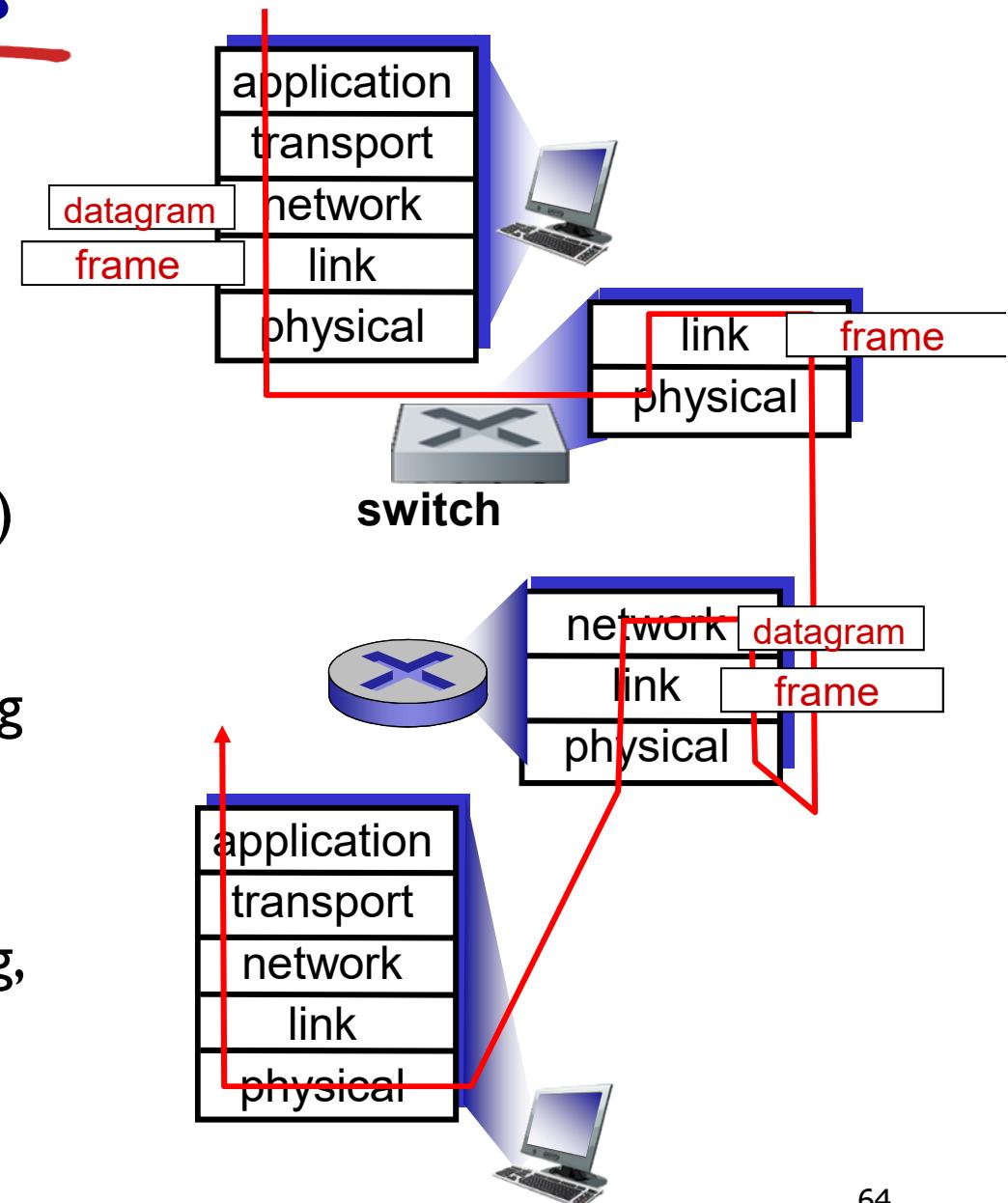
Switches vs. routers

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



What have we learned?

- *MAC address*
 - Format
- *ARP* : Determine MAC address, knowing its IP address
 - Addressing when routing to another networks
- *Ethernet Protocol*,
 - Physical topology
 - Fast Ethernet and Gigabit Ethernet
 - Frame structure and Frame bursting
- *Ethernet Switches*
 - Forwarding table
 - Self learning
 - Interconnecting switches

Link layer, LANs: outline

6.1 introduction, services

6.2 error detection,
correction

6.3 multiple access
protocols

6.4 LANs

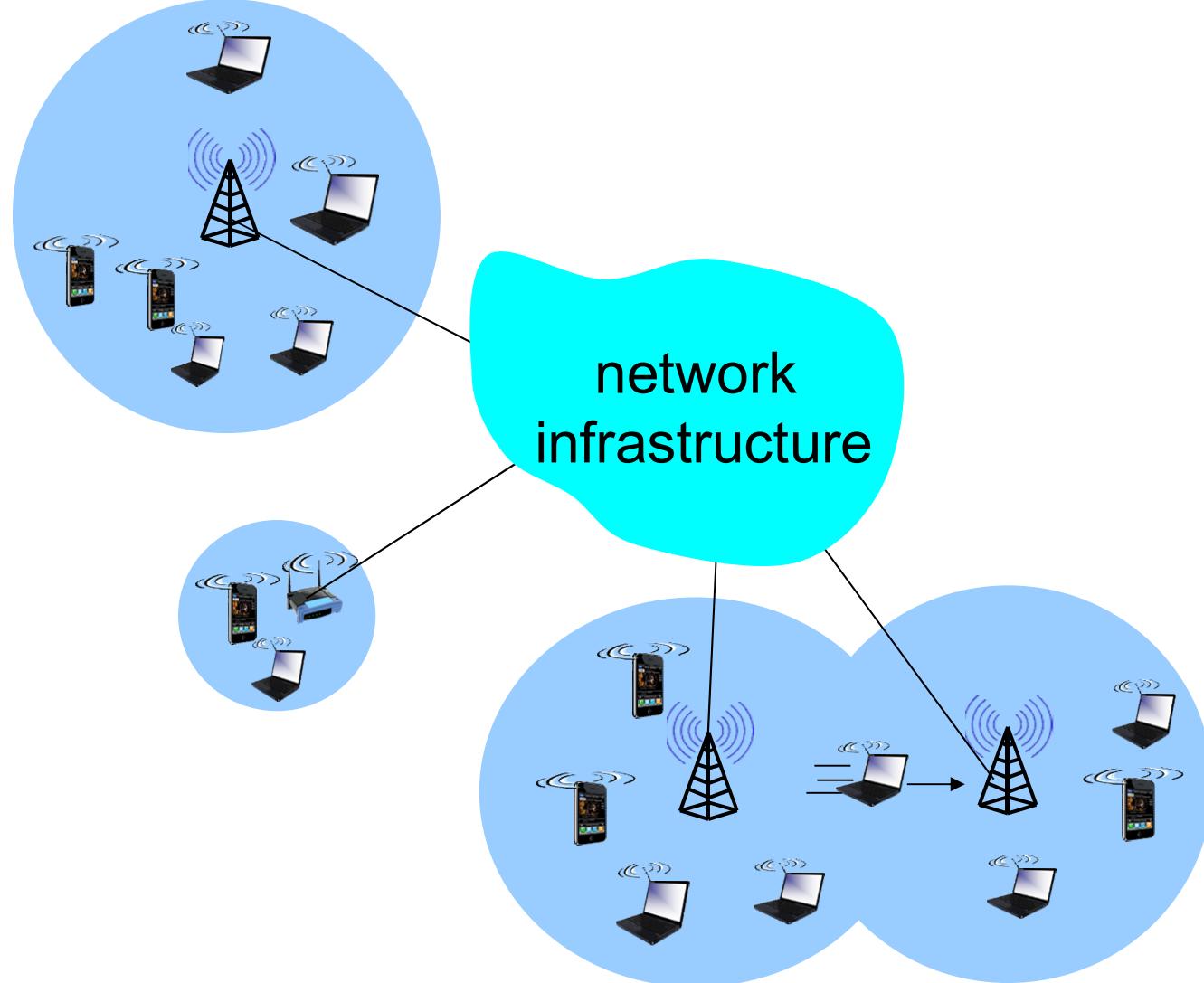
- addressing, ARP
- Ethernet
- switches
- VLANS

6.5 link virtualization:
MPLS

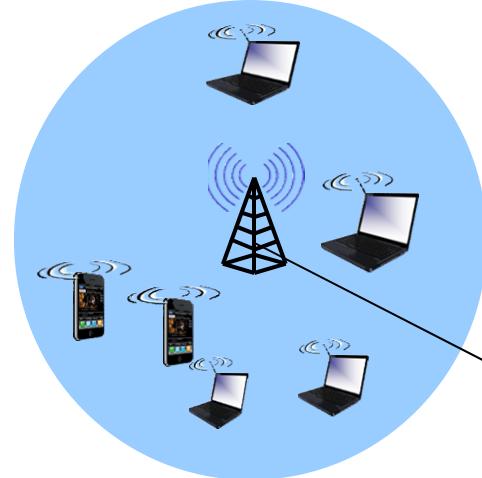
6.6 data center
networking

6.7 a day in the life of a
web request

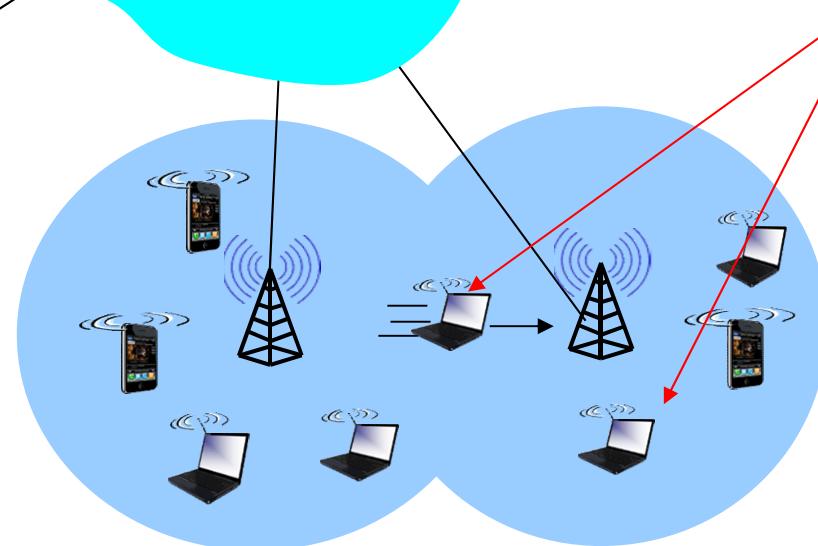
Elements of a wireless network



Elements of a wireless network



network
infrastructure

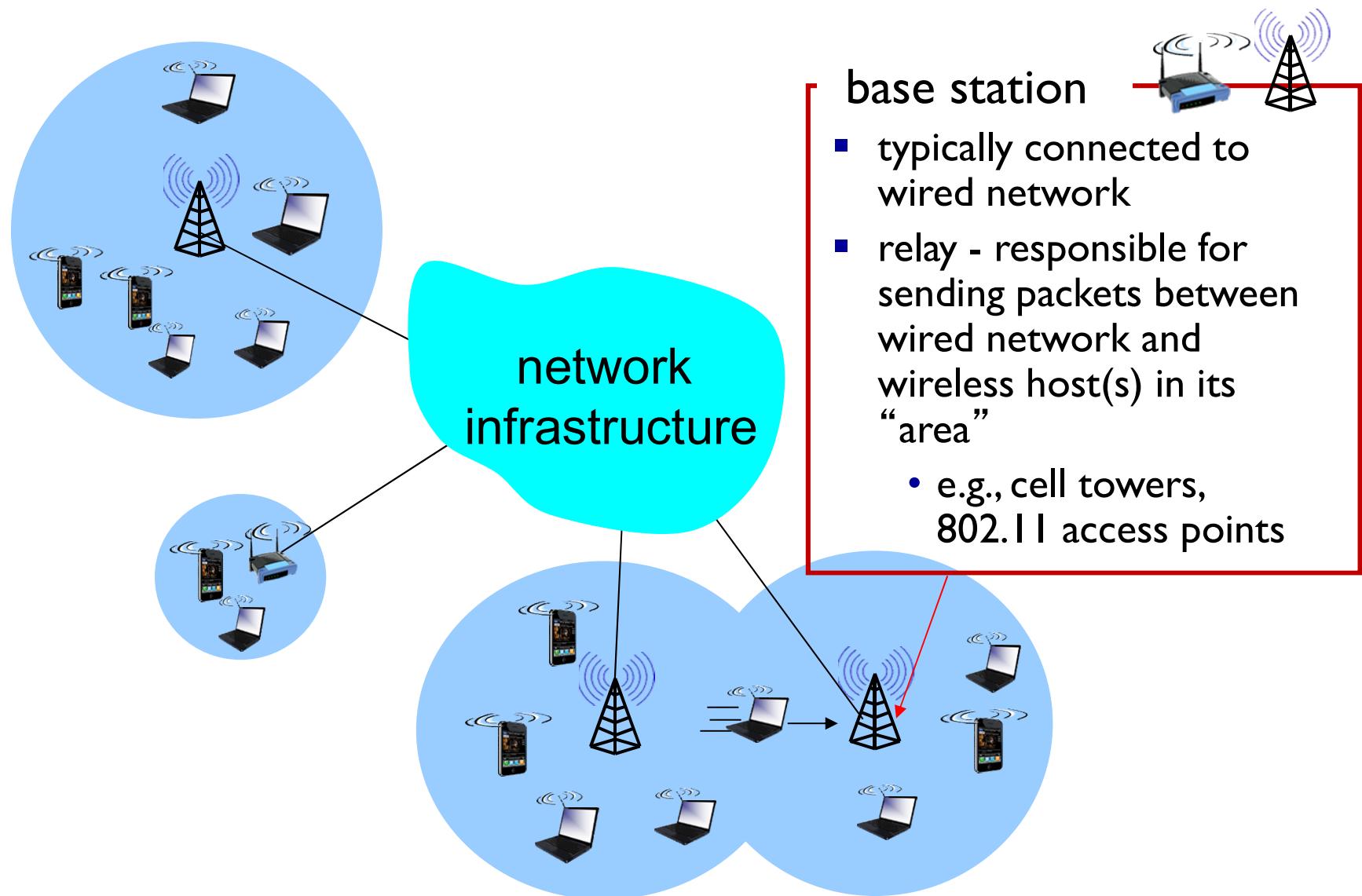


wireless hosts

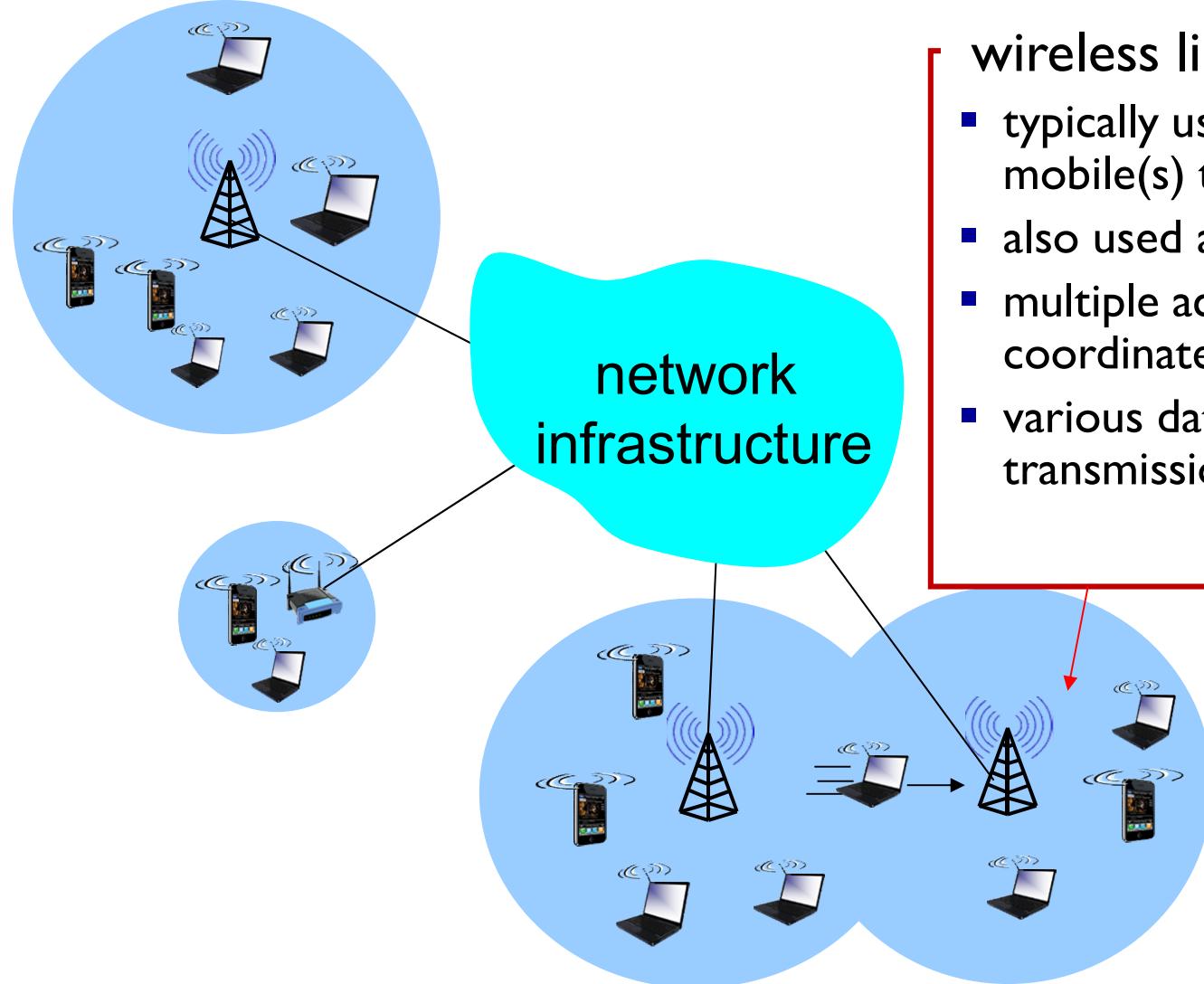
- laptop, smartphone
- run applications
- may be stationary (non-mobile) or mobile
 - wireless does *not* always mean mobility



Elements of a wireless network



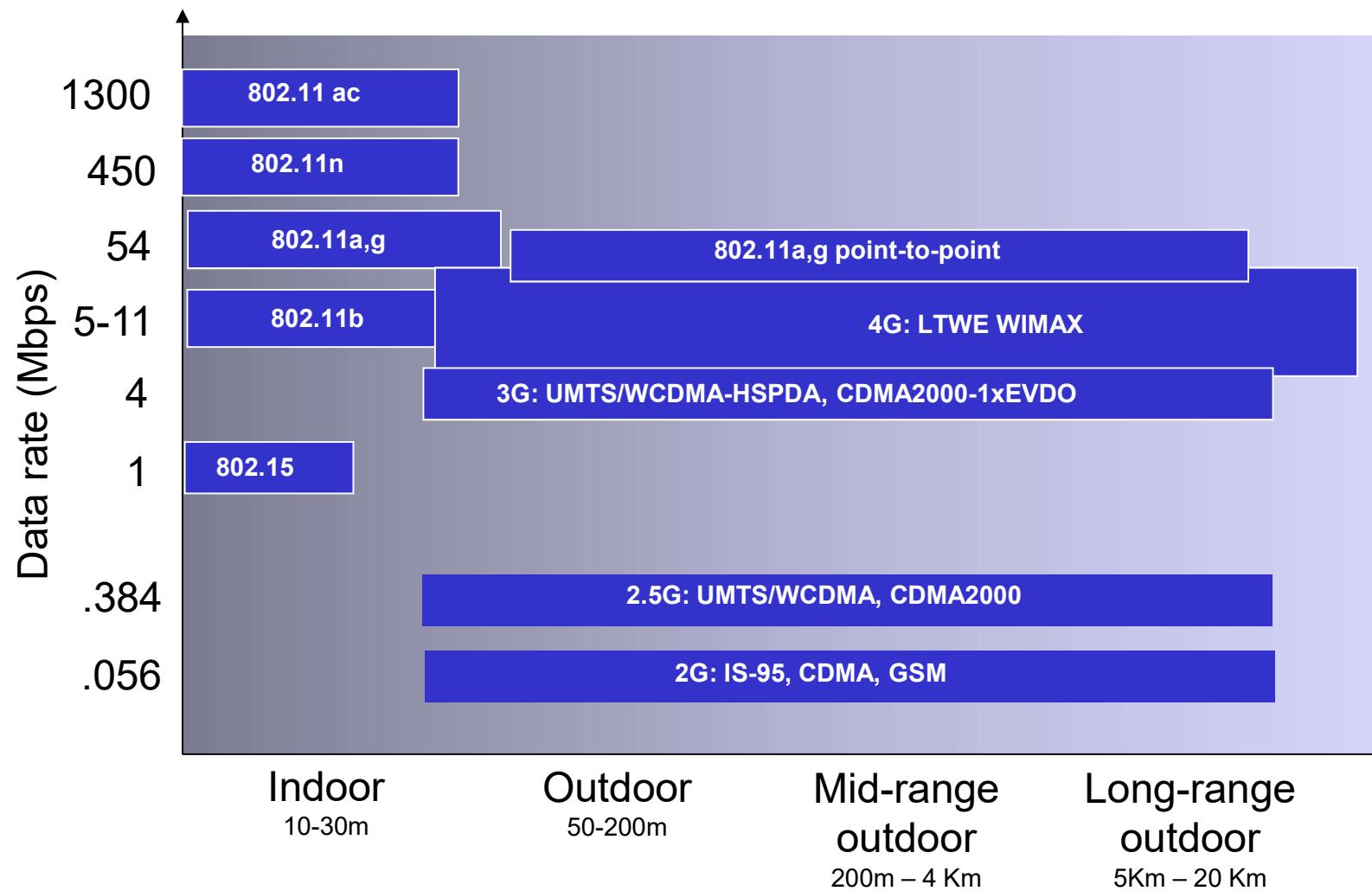
Elements of a wireless network



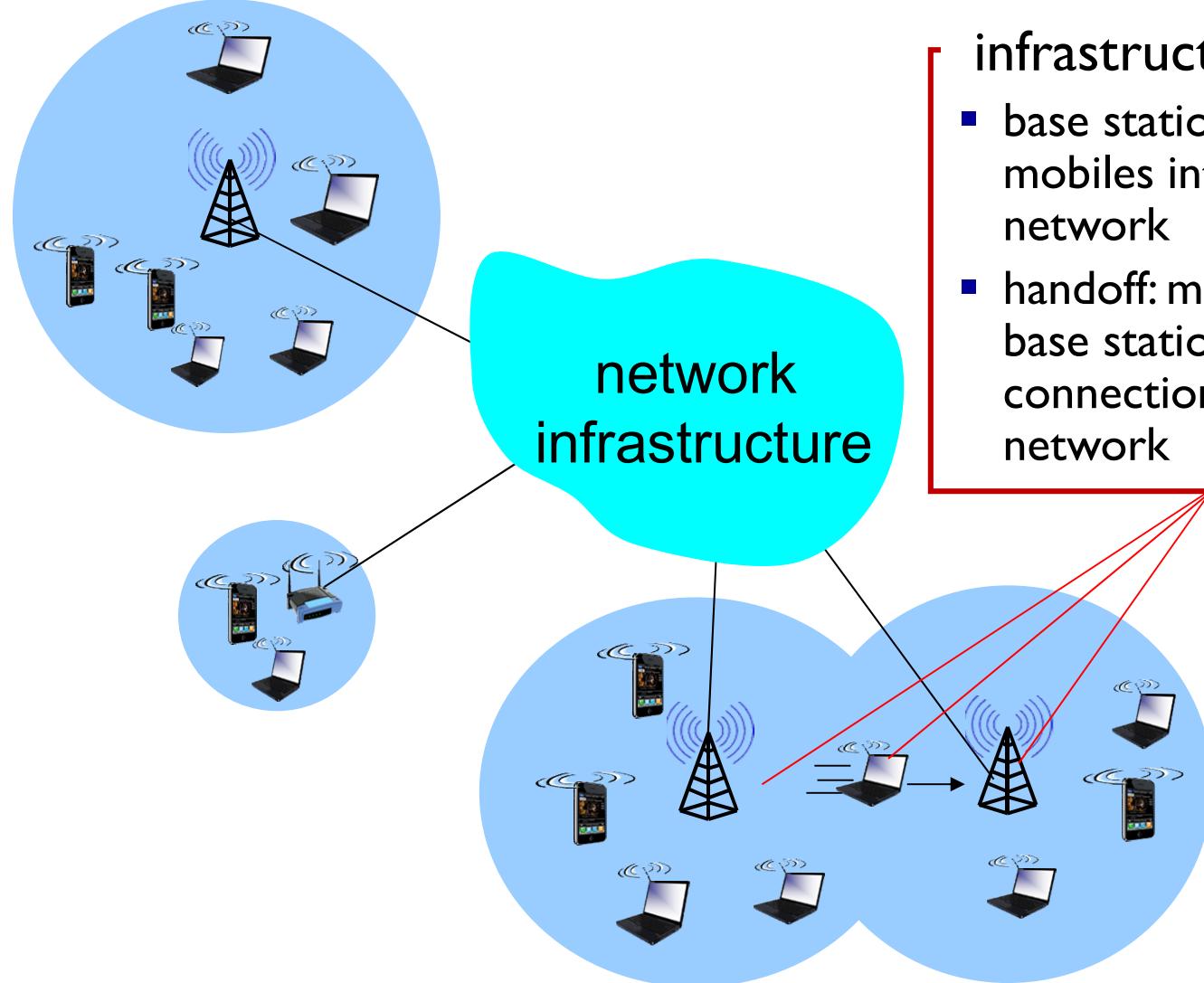
wireless link

- typically used to connect mobile(s) to base station
- also used as backbone link
- multiple access protocol coordinates link access
- various data rates, transmission distance

Characteristics of selected wireless links



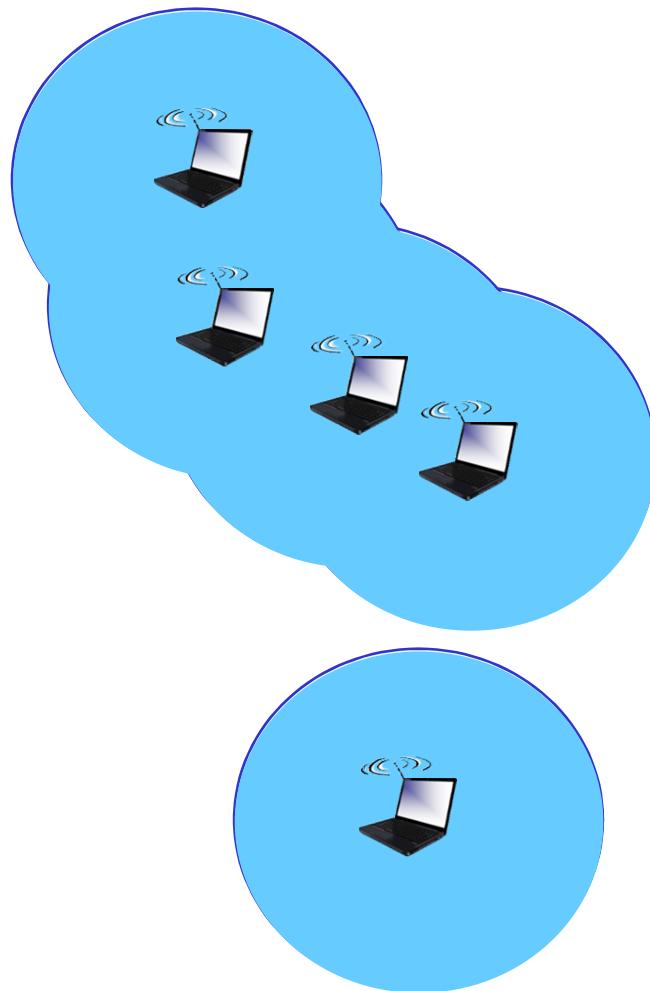
Elements of a wireless network



infrastructure mode

- base station connects mobiles into wired network
- handoff: mobile changes base station providing connection into wired network

Elements of a wireless network



ad hoc mode

- no base stations
- nodes can only transmit to other nodes within link coverage
- nodes organize themselves into a network: route among themselves

Wireless network taxonomy

	single hop	multiple hops
infrastructure (e.g., APs)	host connects to base station (WiFi, WiMAX, cellular) which connects to larger Internet	host may have to relay through several wireless nodes to connect to larger Internet: <i>mesh net</i>
no infrastructure	no base station, no connection to larger Internet (Bluetooth, ad hoc nets)	no base station, no connection to larger Internet. May have to relay to reach other a given wireless node MANET, VANET

IEEE 802.11 Wireless LAN

802.11b

- 2.4-5 GHz unlicensed spectrum
- up to 11 Mbps
- direct sequence spread spectrum (DSSS) in physical layer
 - all hosts use same chipping code

802.11a

- 5-6 GHz range
- up to 54 Mbps

802.11g

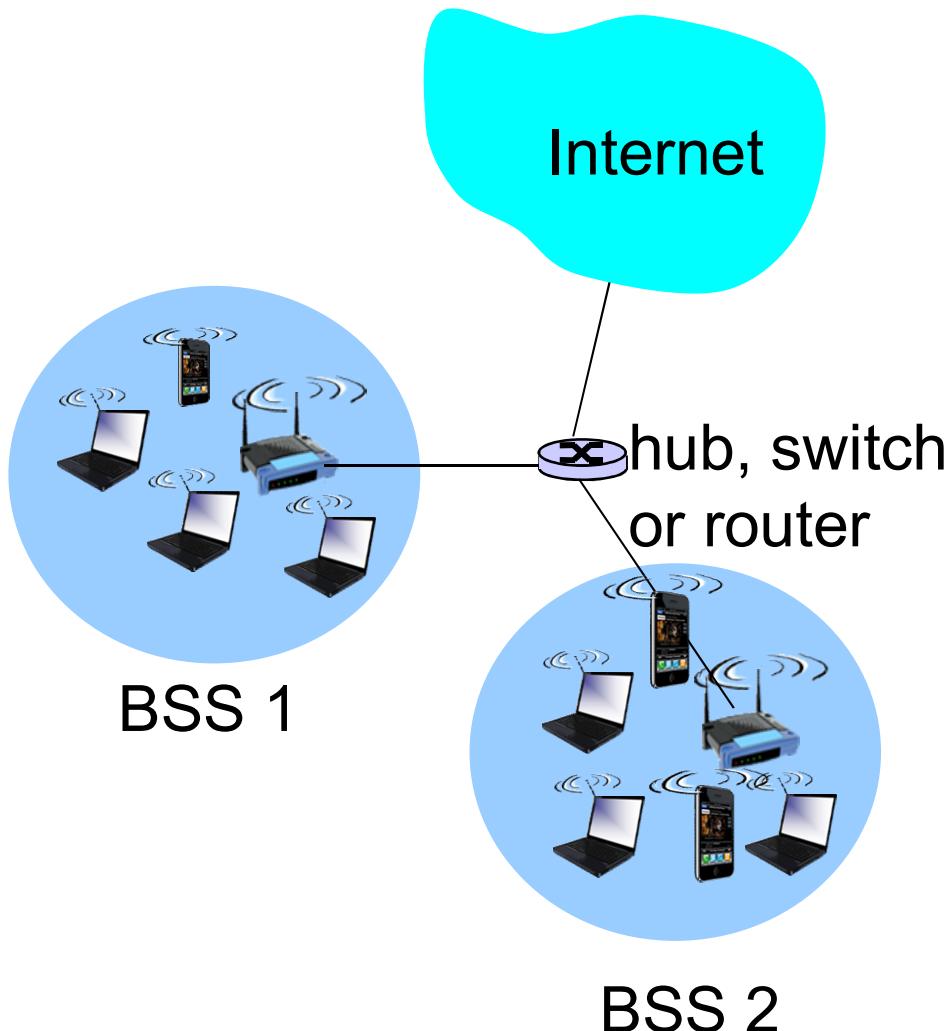
- 2.4-5 GHz range
- up to 54 Mbps

802.11n: multiple antennae

- 2.4-5 GHz range
- up to 200 Mbps

-
- all use CSMA/CA for multiple access
 - all have base-station and ad-hoc network versions

802.11 LAN architecture

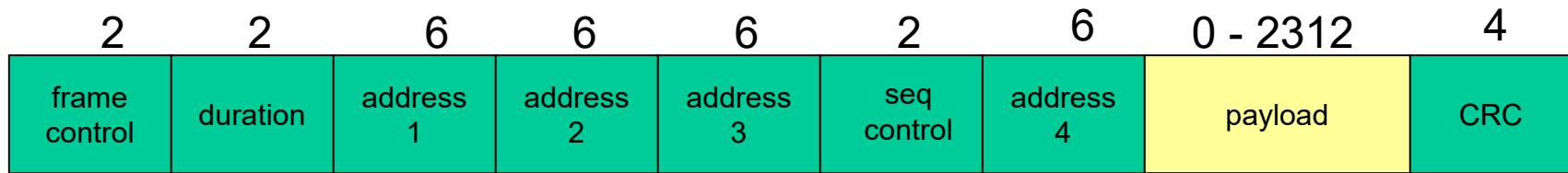


- wireless host communicates with base station
 - **base station = access point (AP)**
- **Basic Service Set (BSS)** (aka “cell”) in infrastructure mode contains:
 - wireless hosts
 - access point (AP): base station
 - ad hoc mode: hosts only

802.11: Channels, association

- 802.11b: 2.4GHz-2.485GHz spectrum divided into 11 channels at different frequencies
 - AP admin chooses frequency for AP
 - interference possible: channel can be same as that chosen by neighboring AP!
- host: must **associate** with an AP
 - scans channels, listening for *beacon frames* containing AP's name (SSID) and MAC address
 - selects AP to associate with
 - may perform authentication [Chapter 8]
 - will typically run DHCP to get IP address in AP's subnet

802.11 frame: addressing



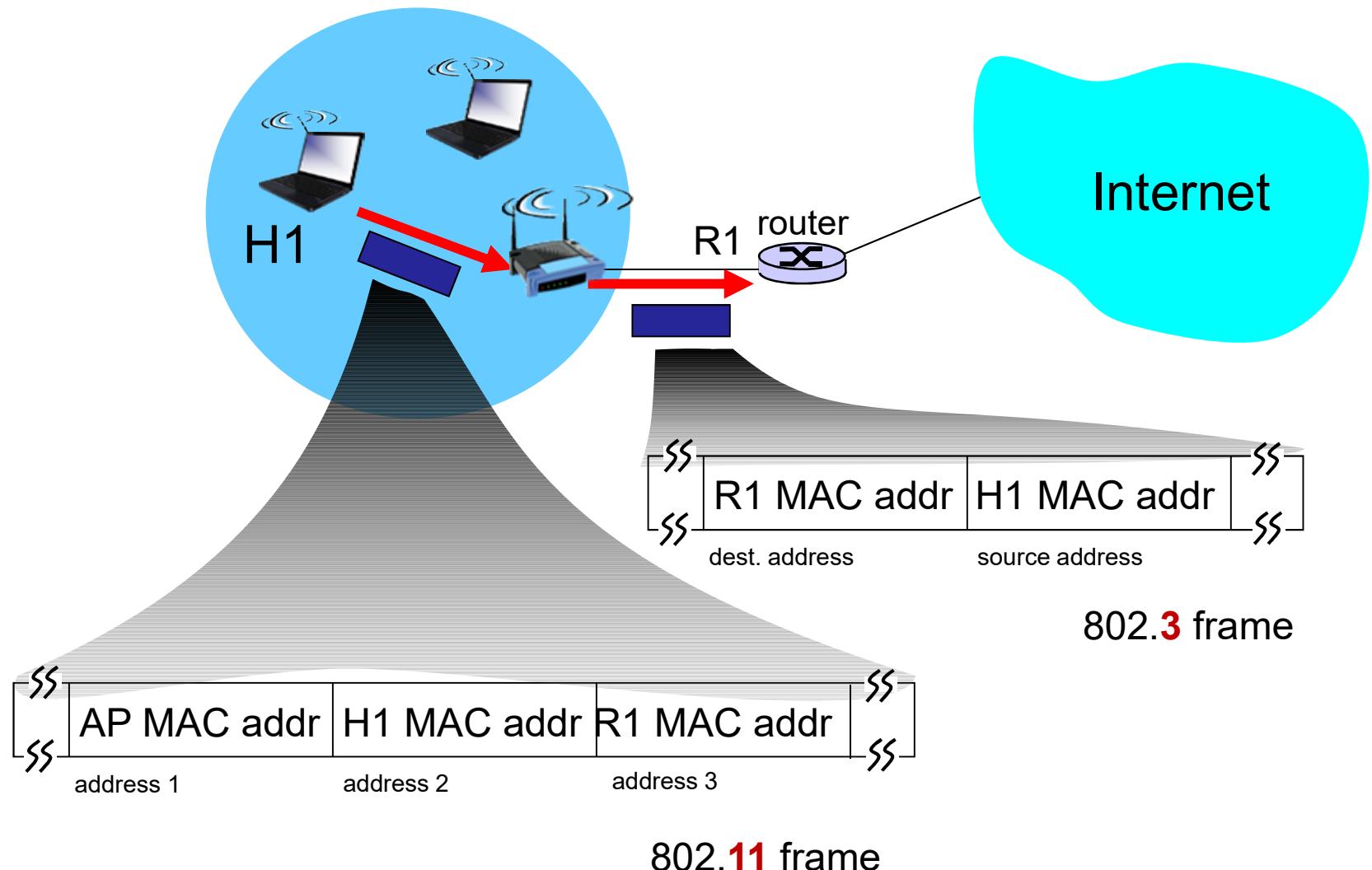
Address 1: MAC address
of wireless host or AP
to receive this frame

Address 2: MAC address
of wireless host or AP
transmitting this frame

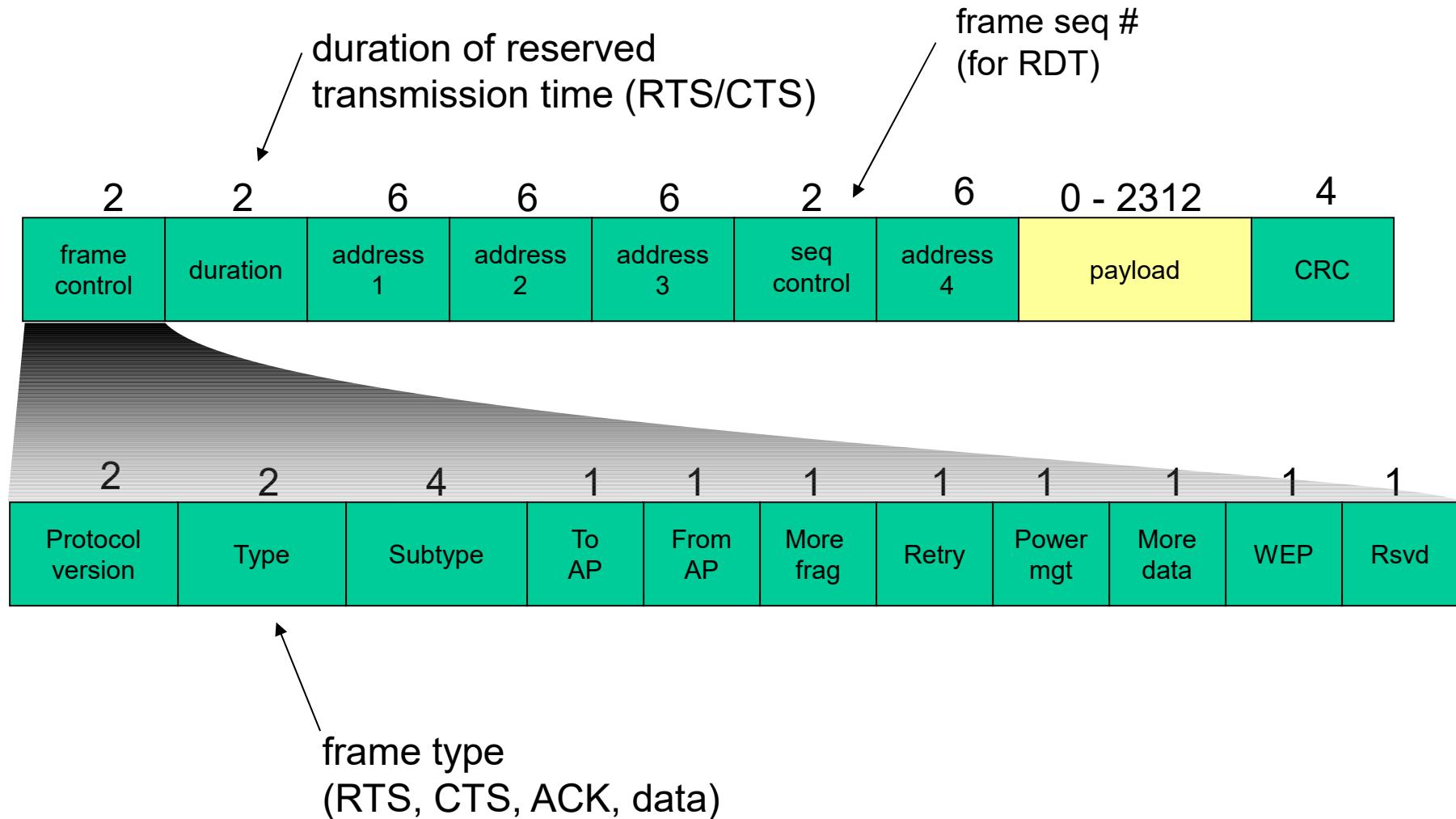
Address 3: MAC address
of router interface to
which AP is attached

Address 4: used only in
ad hoc mode

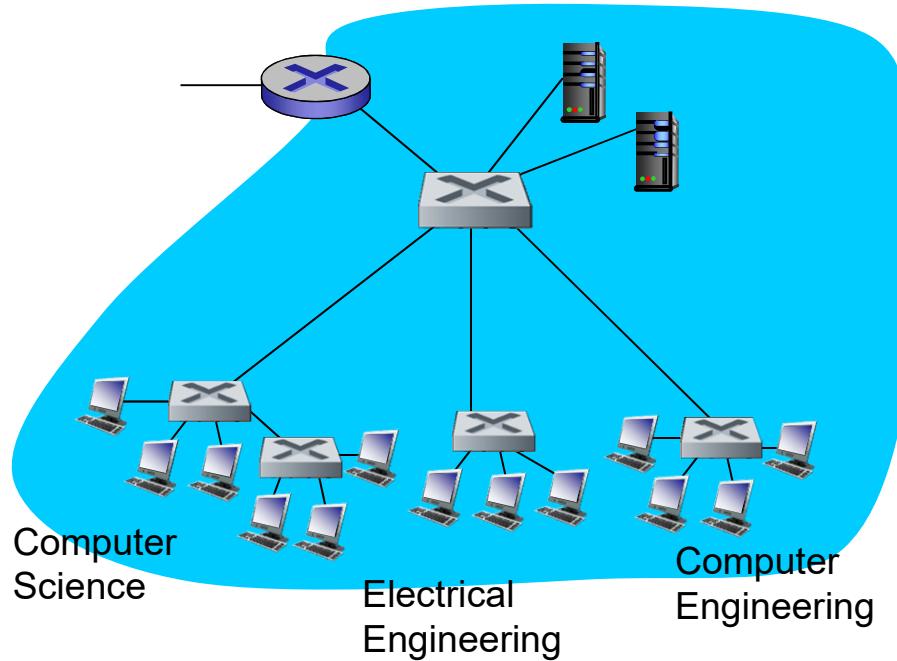
802.11 frame: addressing



802.11 frame: more



VLANs: motivation



consider:

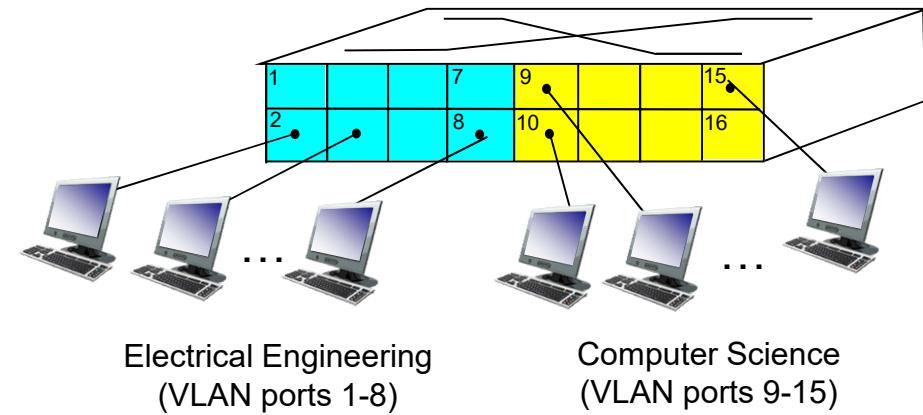
- CS user moves office to EE, but wants connect to CS switch?
- single broadcast domain:
 - all layer-2 broadcast traffic (ARP, DHCP, unknown location of destination MAC address) must cross entire LAN
 - security/privacy, efficiency issues

VLANs

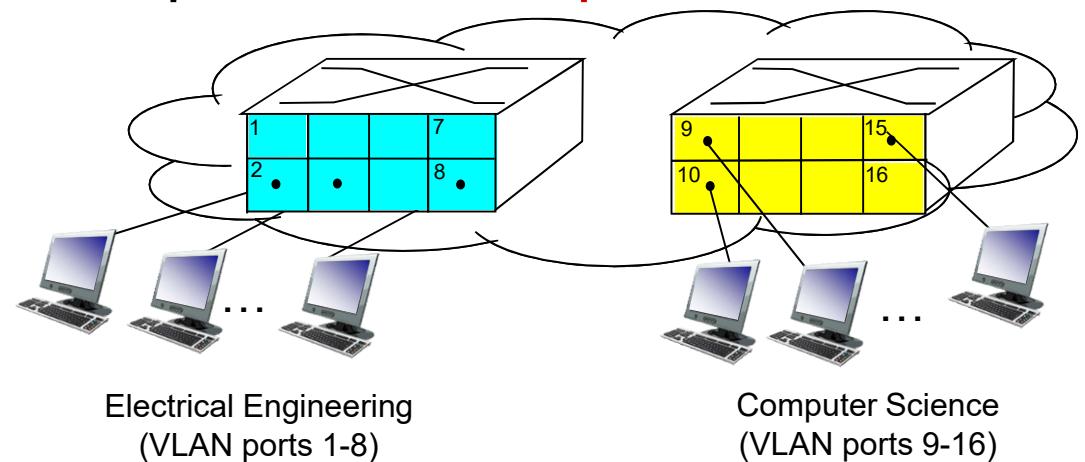
Virtual Local Area Network

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

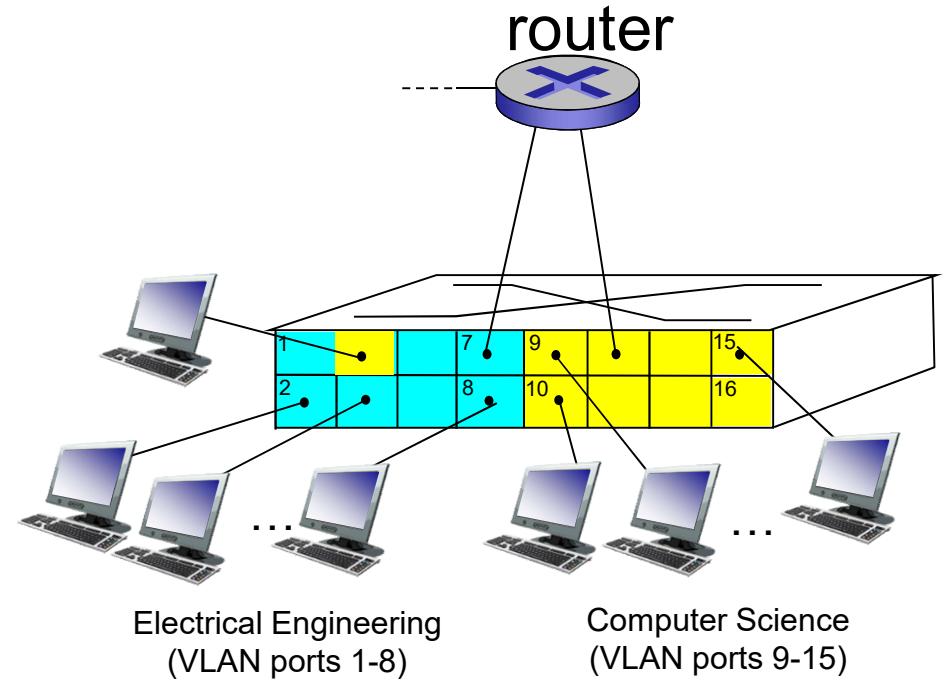


... operates as *multiple virtual switches*

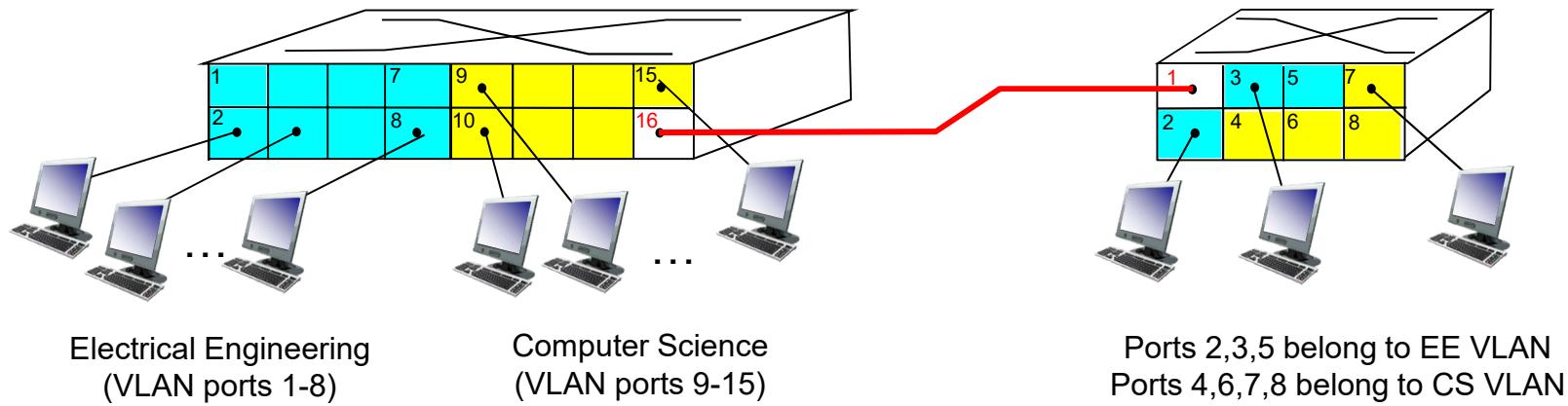


Port-based VLAN

- ***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

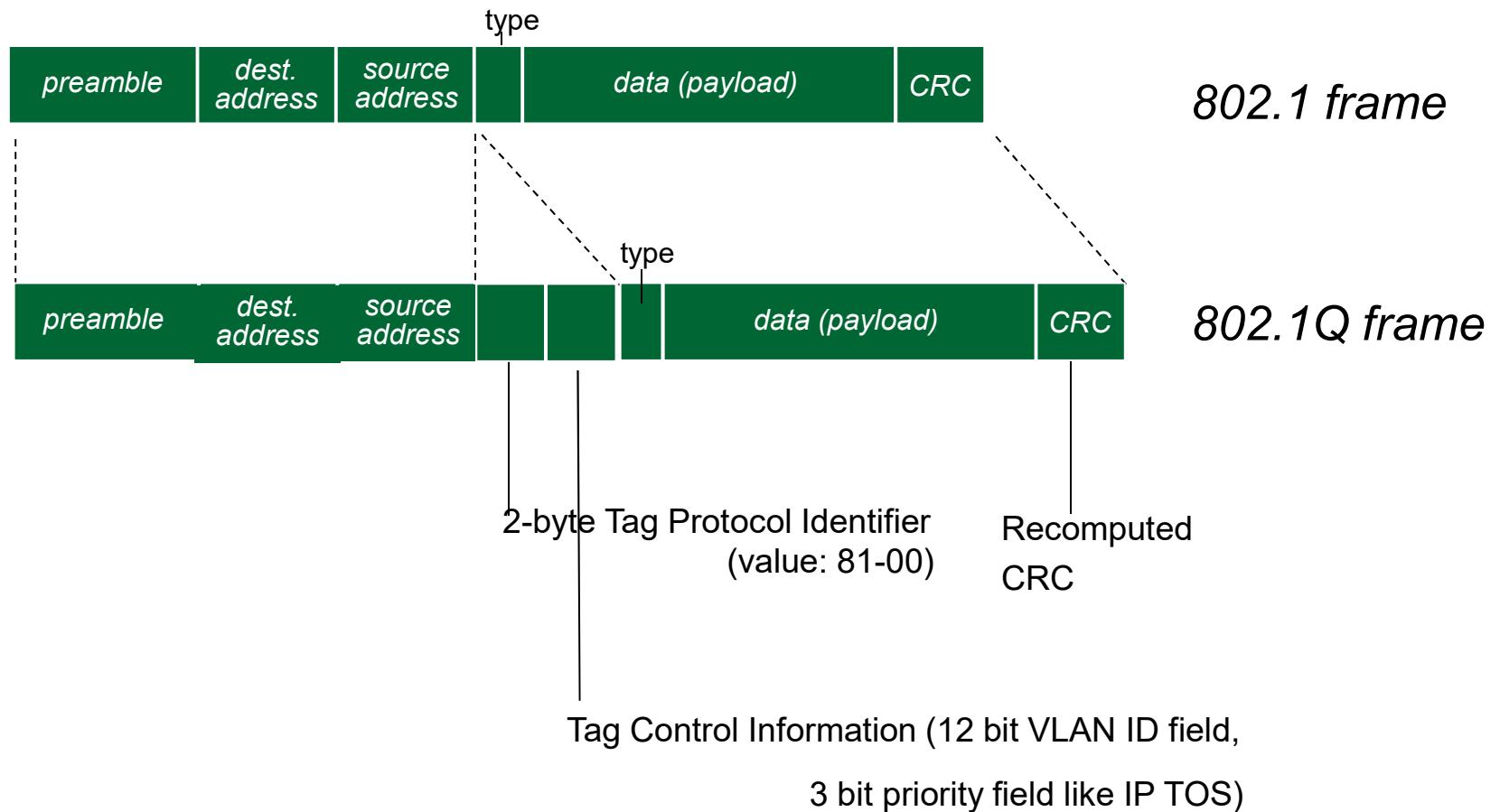


VLANS spanning multiple switches



- **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



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

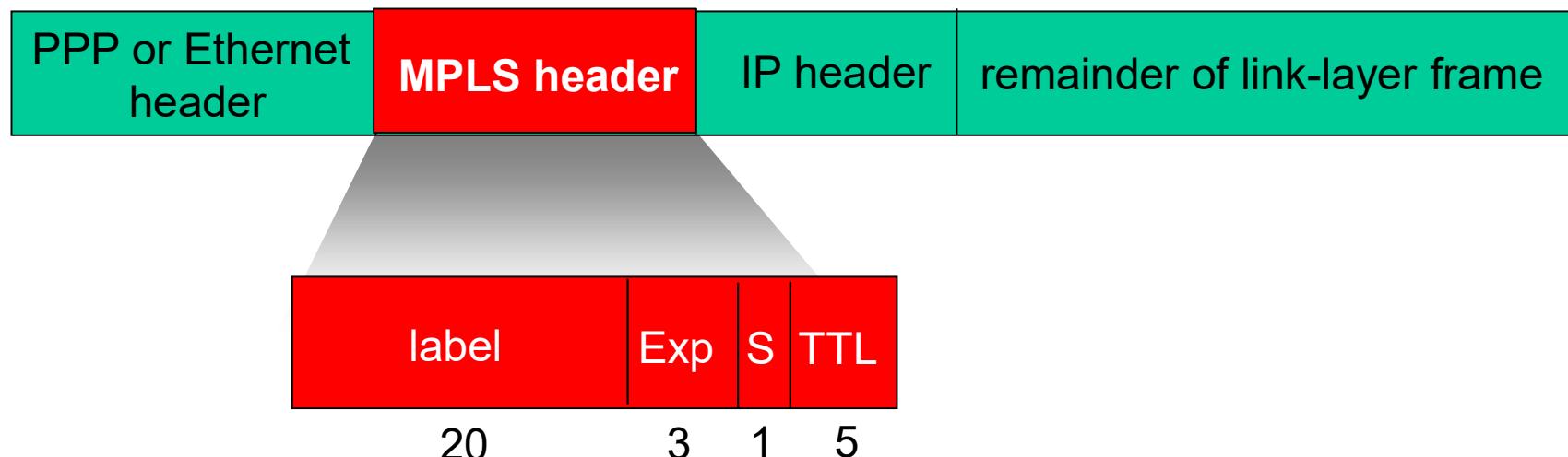
6.5 link virtualization:
MPLS

6.6 data center
networking

6.7 a day in the life of a
web request

Multiprotocol label switching (MPLS)

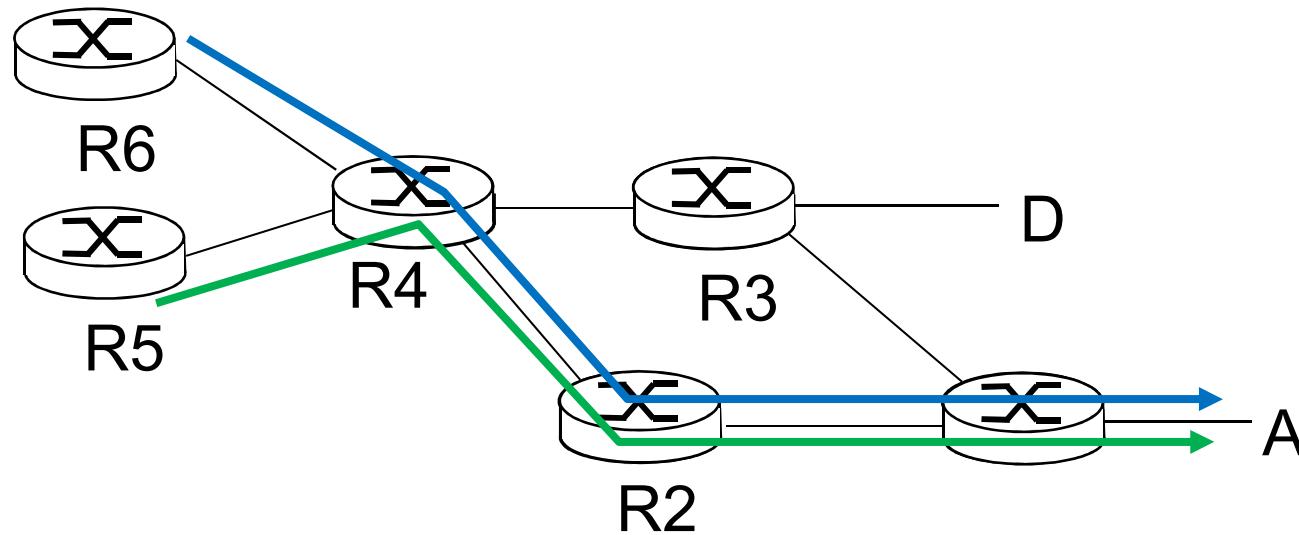
- initial goal: high-speed IP forwarding using fixed length label (instead of IP address)
 - fast lookup using fixed length identifier (rather than longest prefix matching)
 - borrowing ideas from Virtual Circuit (VC) approach
 - but IP datagram still keeps IP address!



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)

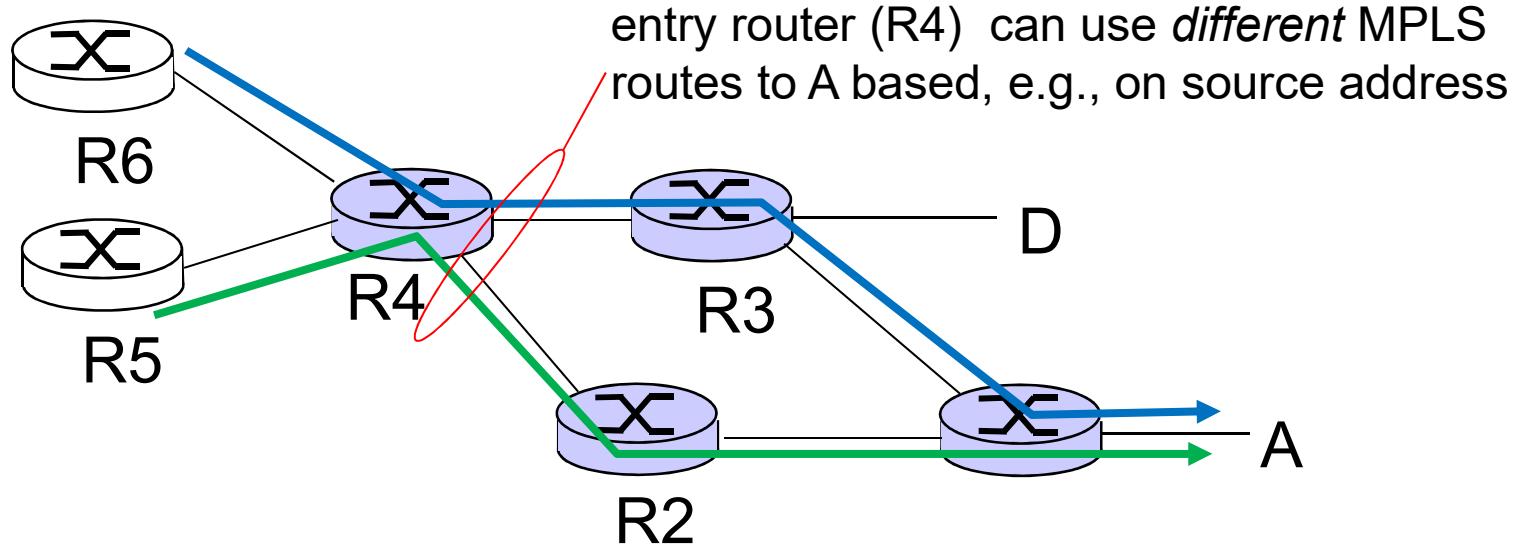
MPLS versus IP paths



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



MPLS versus IP paths



- **IP routing:** path to destination determined by destination address alone
- **MPLS routing:** path to destination can be based on source and destination address
 - **fast reroute:** precompute backup routes in case of link failure



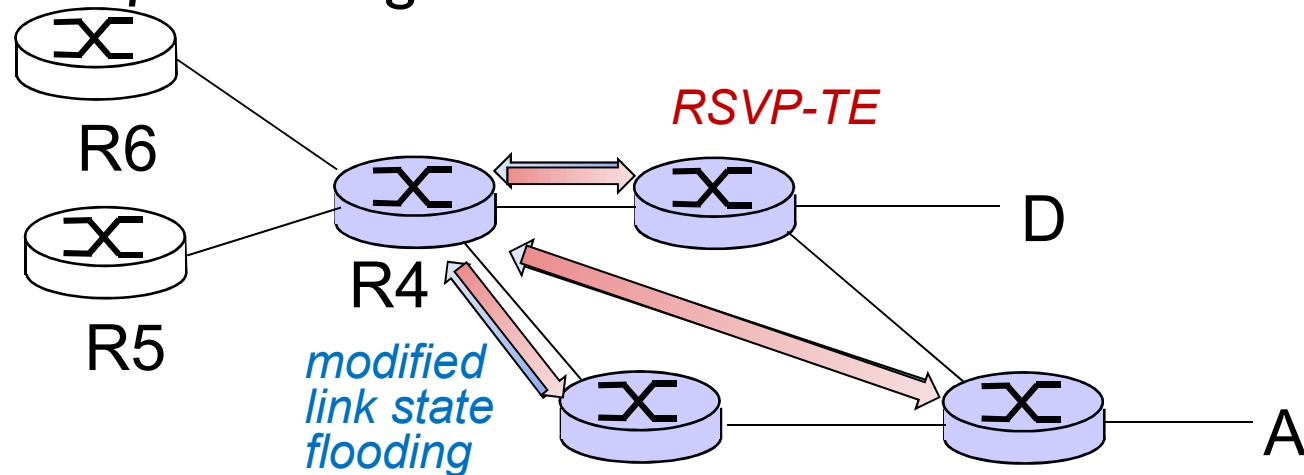
IP-only
router



MPLS and
IP router

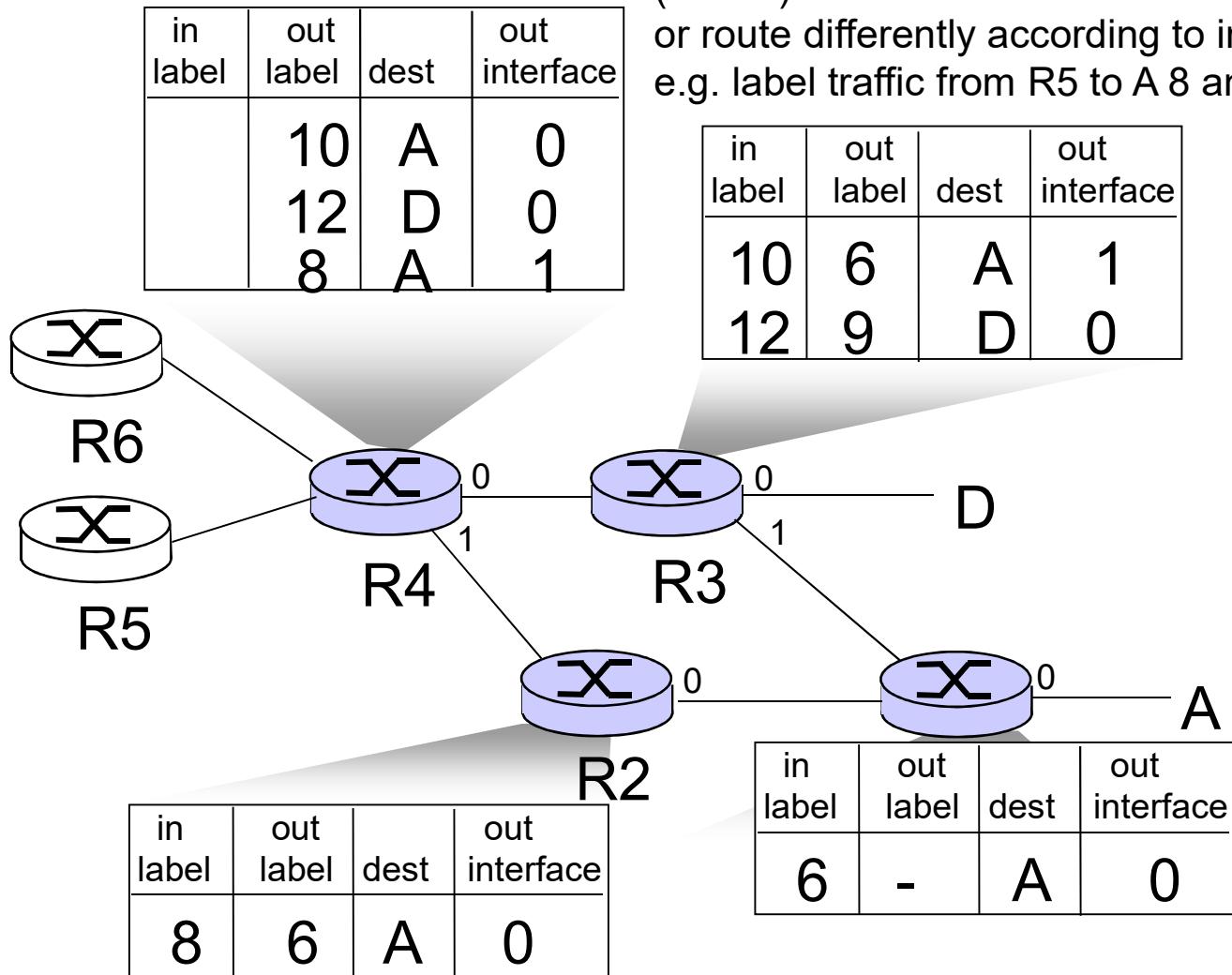
MPLS signaling

- 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 *RSPV-TE* signaling protocol to set up MPLS forwarding at downstream routers



MPLS forwarding tables

Note in this example R4 has two possible labels (routes) to A – it could use one for redundancy or route differently according to inbound router e.g. label traffic from R5 to A 8 and from R6 10



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

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

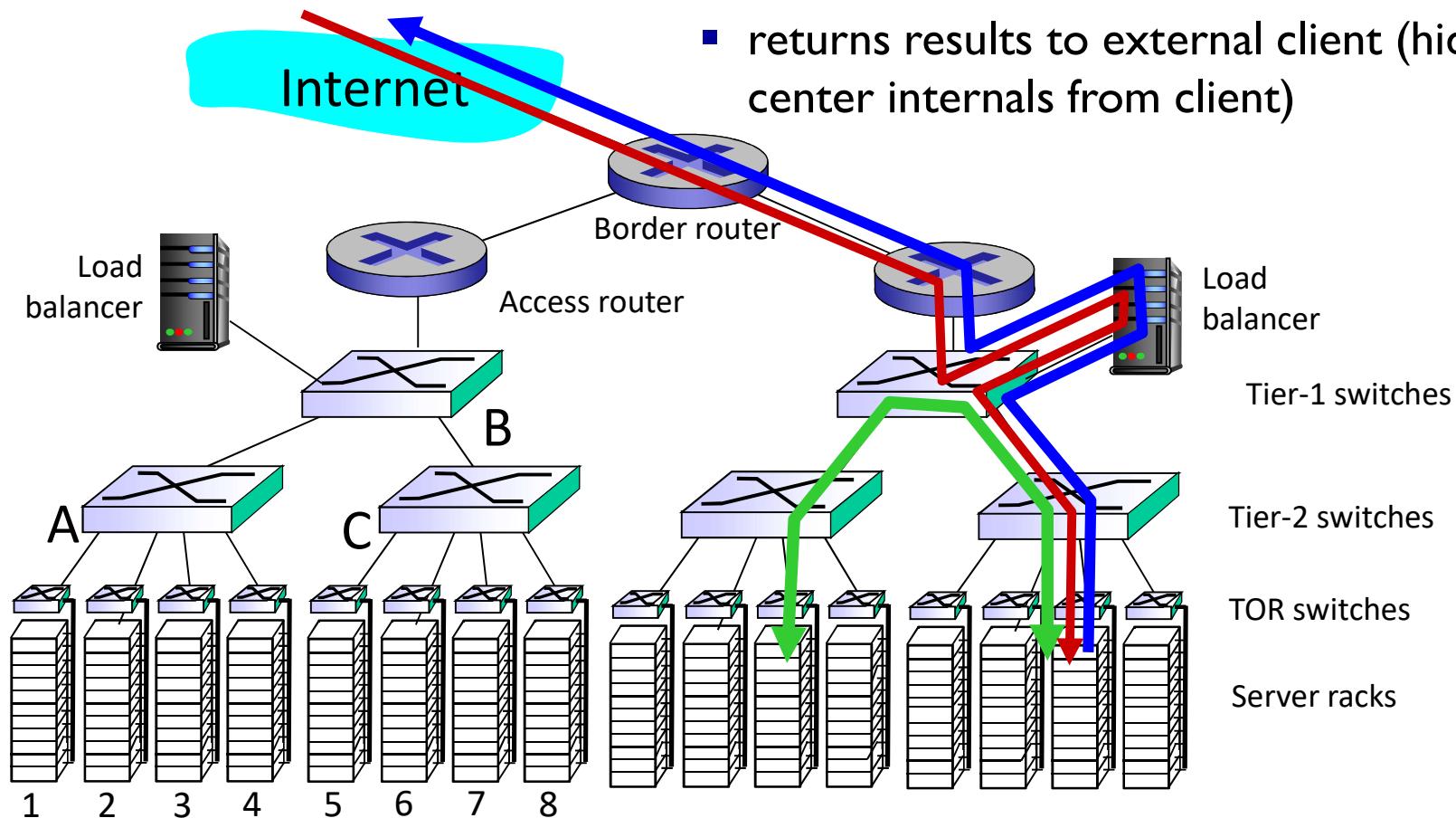


Inside a 40-ft Microsoft container,
Chicago data center

Data center networks

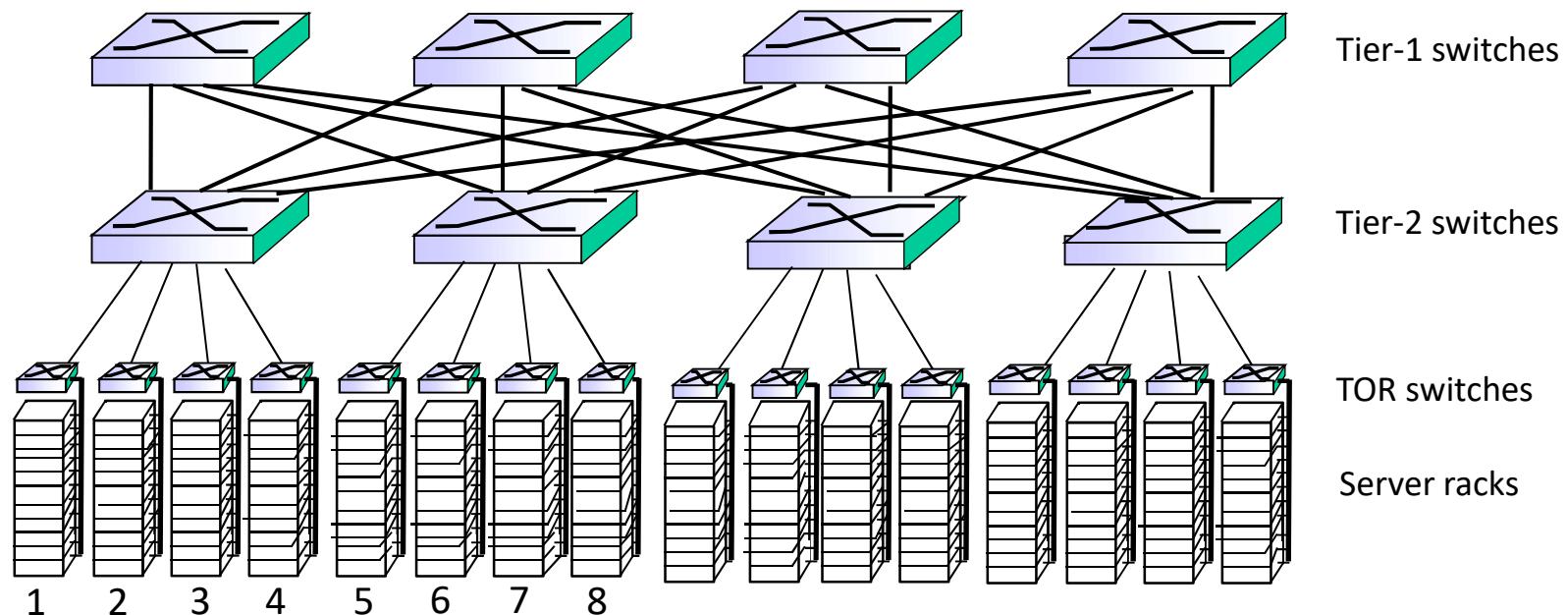
load balancer: application-layer routing

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



Data center networks

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



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

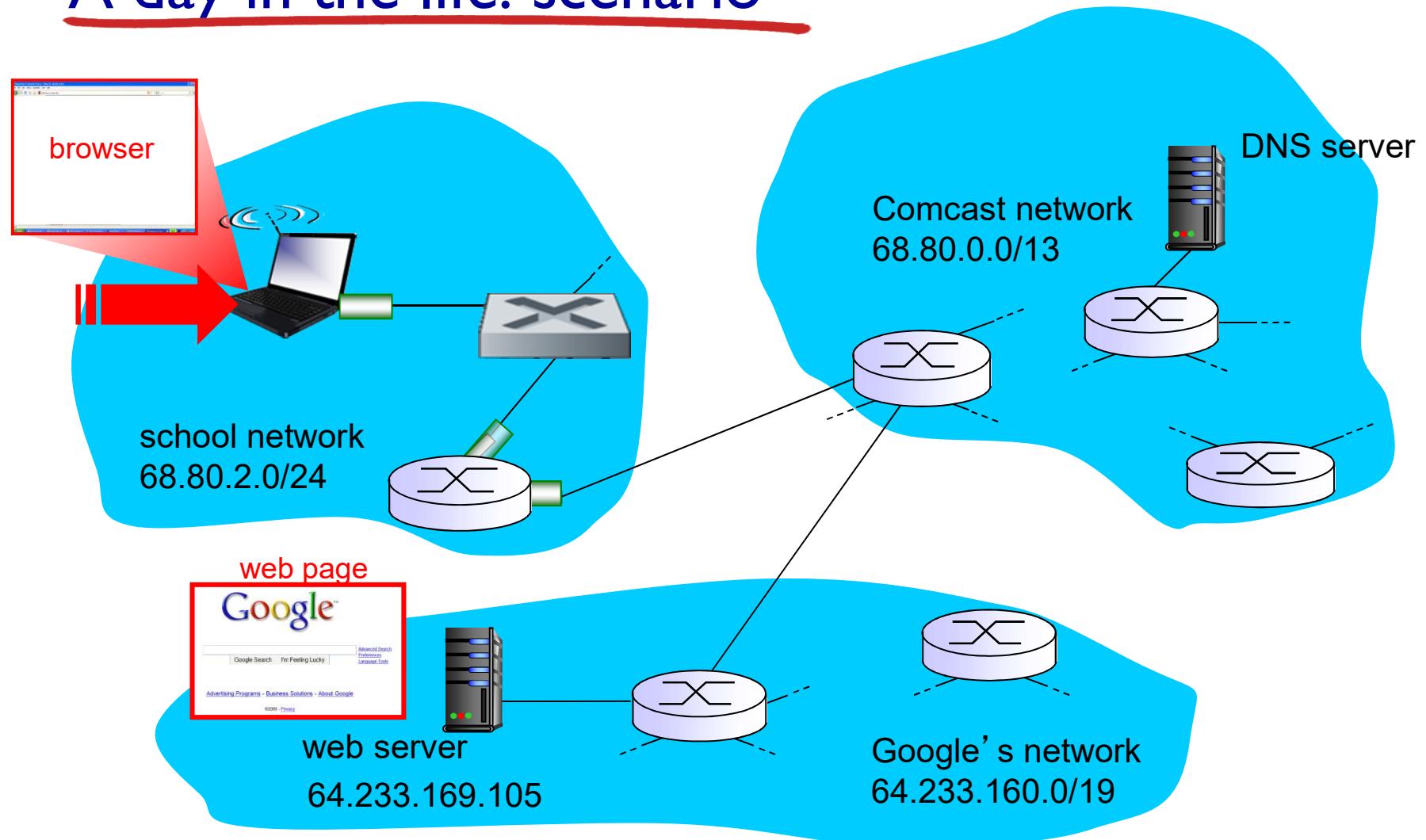
6.6 data center
networking

6.7 a day in the life of a
web request

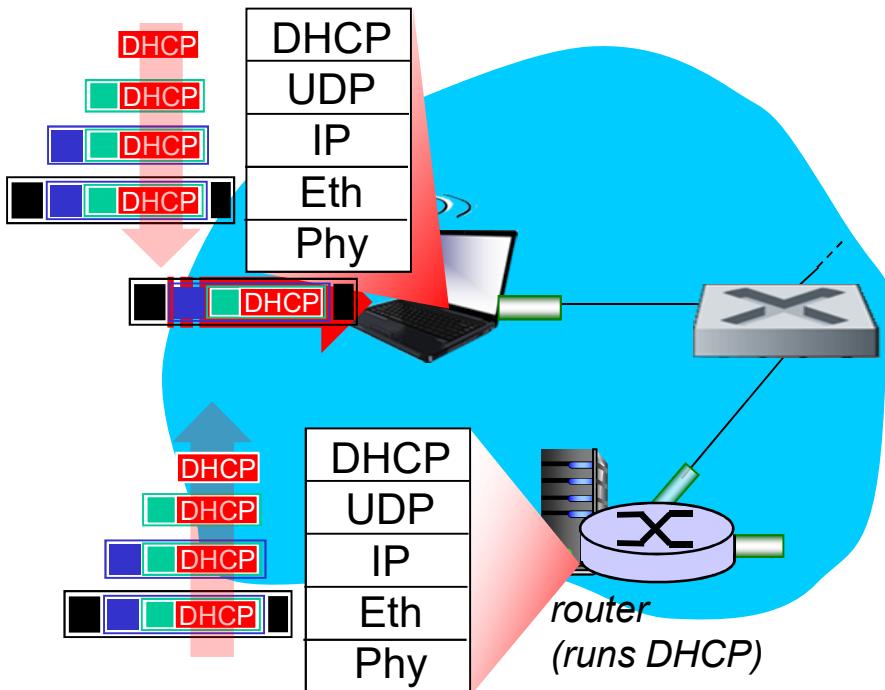
Synthesis: a day in the life of a web request

- 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

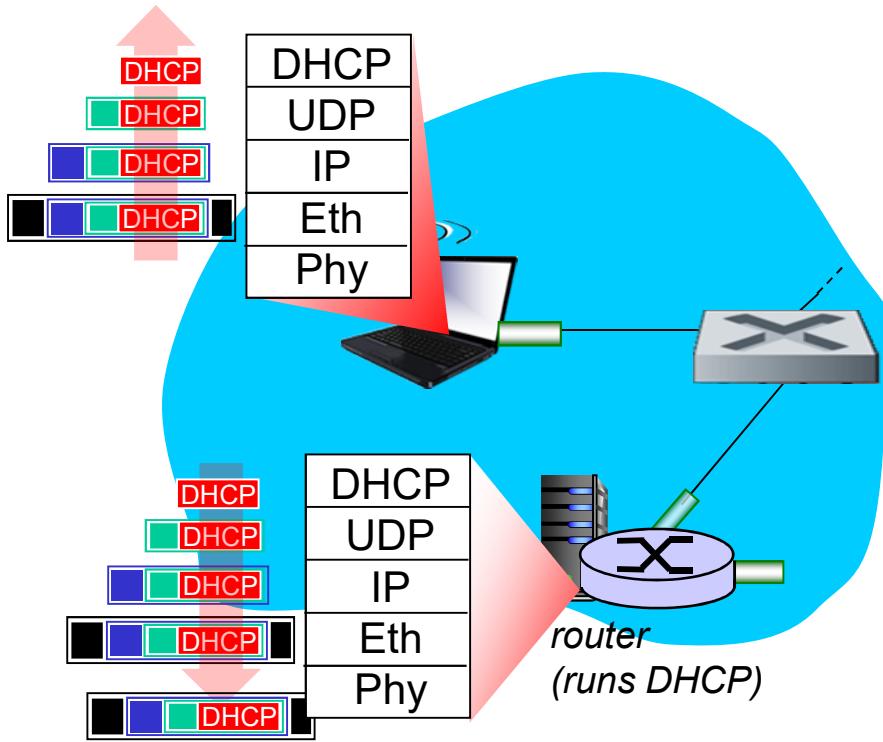


A day in the life... connecting to the Internet



- 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**

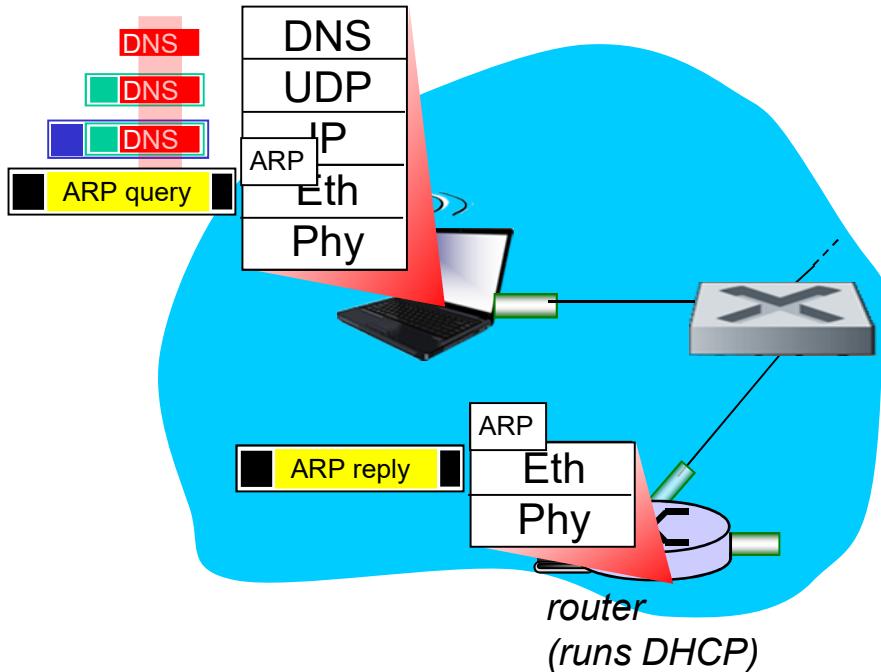
A day in the life... connecting to the Internet



- 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

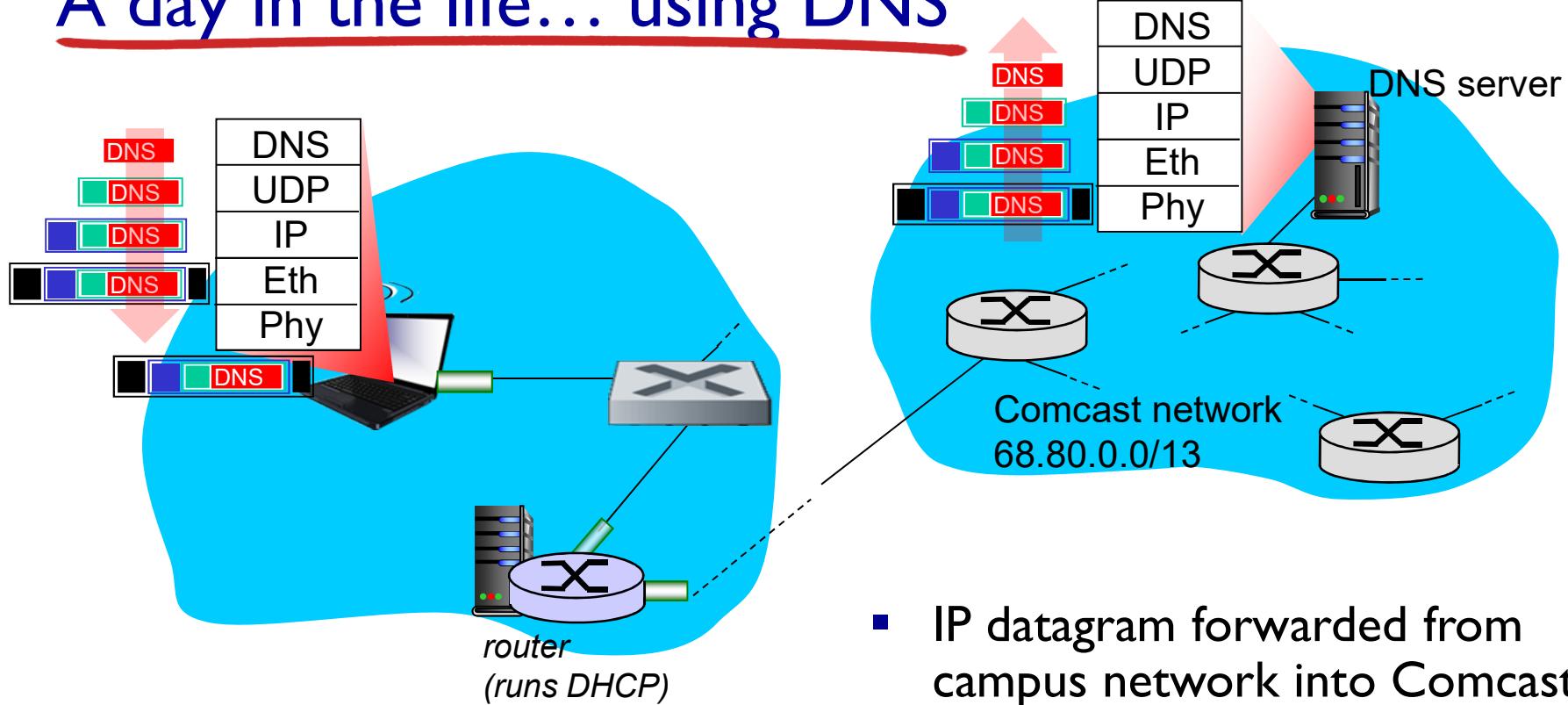
Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

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



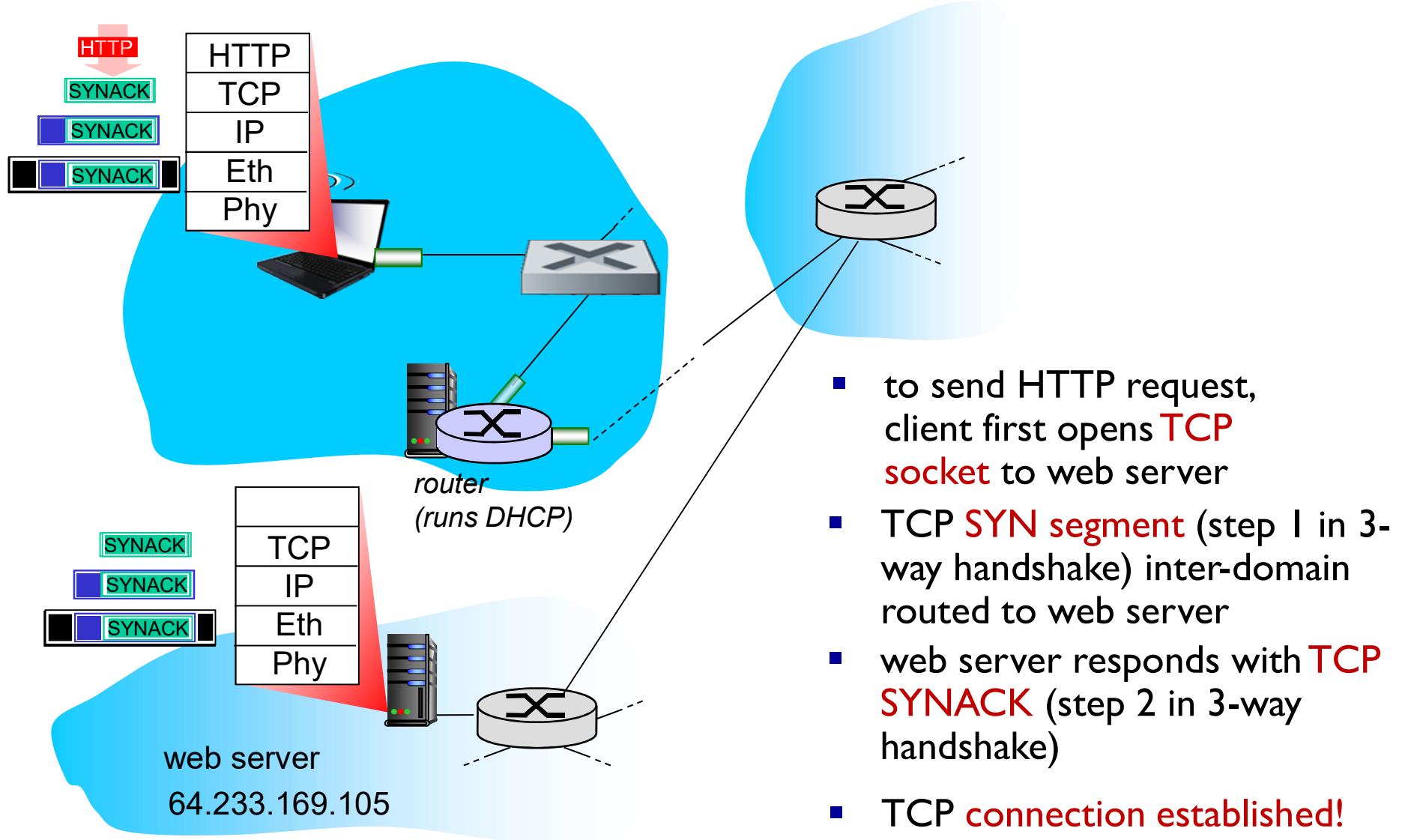
- before sending **HTTP** request, need IP address of www.google.com: **DNS**
- DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: **ARP**
- **ARP query** broadcast, received by router, which replies with **ARP 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

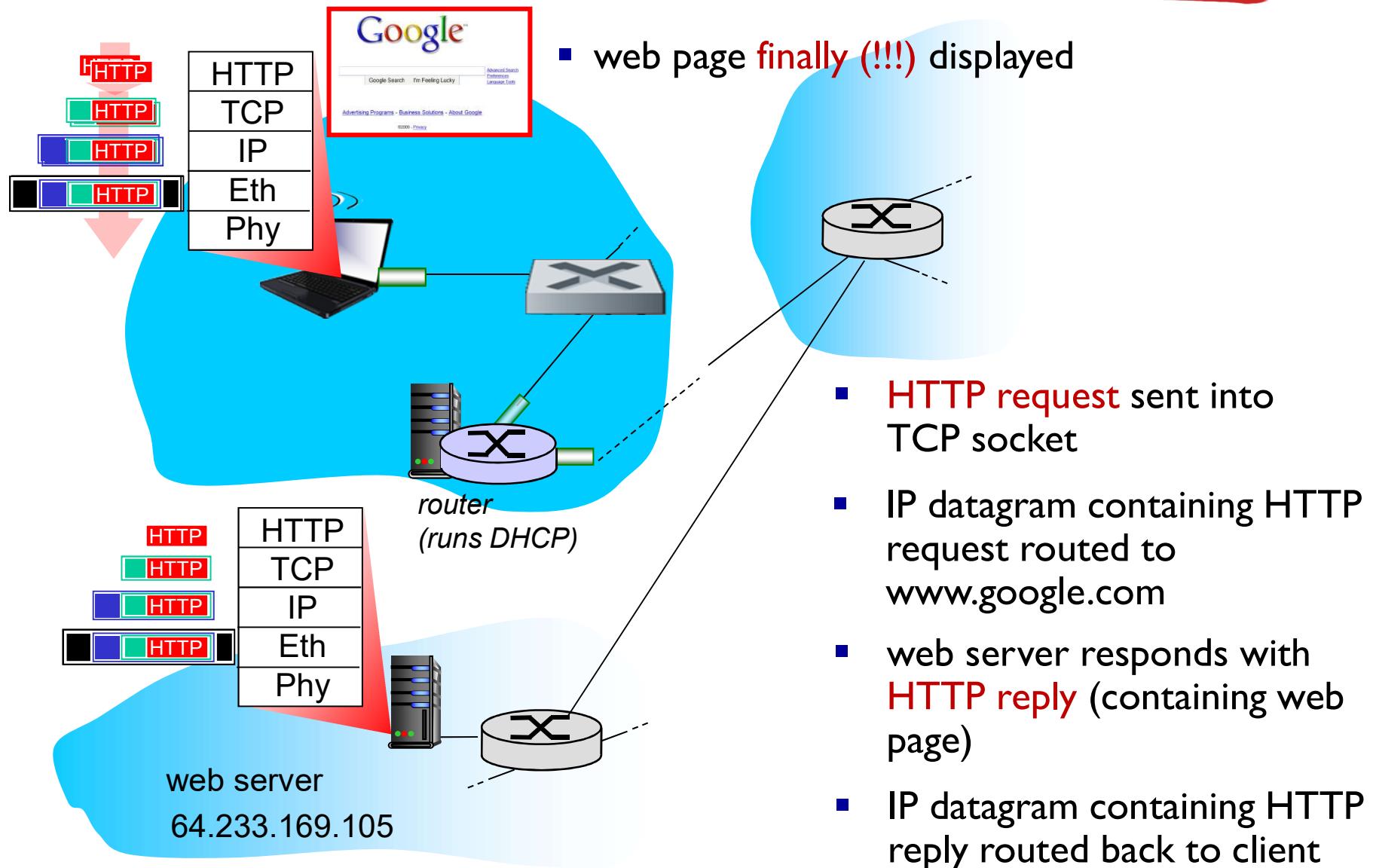


- IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router
- IP datagram forwarded from campus network into Comcast network, routed (tables created by **RIP, OSPF, IS-IS** and/or **BGP** routing protocols) to DNS server
- demuxed to DNS server
- DNS server replies to client with IP address of www.google.com

A day in the life...TCP connection carrying HTTP



A day in the life... HTTP request/reply



What have we learned?

- **Wireless Networks**
 - Infrastructure and ad hoc networks
 - IEEE 802.11x: Architecture and Frame structure
- **VLAN:** switch(es) supporting VLAN capabilities can be configured to define multiple virtual LANs over single physical LAN infrastructure.
 - Topology
 - Frame format
- **MPLS**, high-speed IP forwarding using fixed length label (instead of IP address)
 - Signaling
 - Forwarding decision
- **Data center networking:** Real examples of data center networks
- **A day in the life of a web request:** example of journey down protocol stack complete

Link layer: 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

Data link layer: let's take a breath

- journey down protocol stack *complete* (except PHY or physical)
- solid understanding of networking principles, practice
- We are finished here but *lots* of interesting topics in the text book!
 - mobile
 - multimedia
 - security
- For this module we will use the final lecture to look at network management and security