

Enterprise Infrastructure & Networks – IT 520-A

Chapter 6: The Link Layer and LANs

Marymount University

Instructor: Dr. Ibrahim Waziri Jr.

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

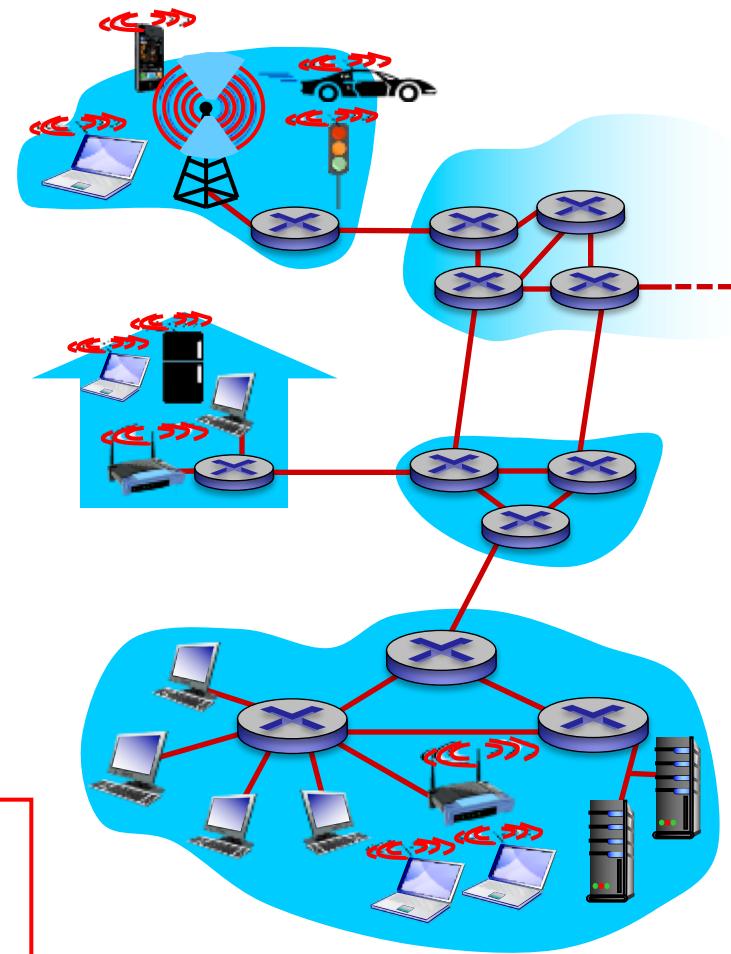
6.7 a day in the life of a
web request

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

data-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 rdt 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

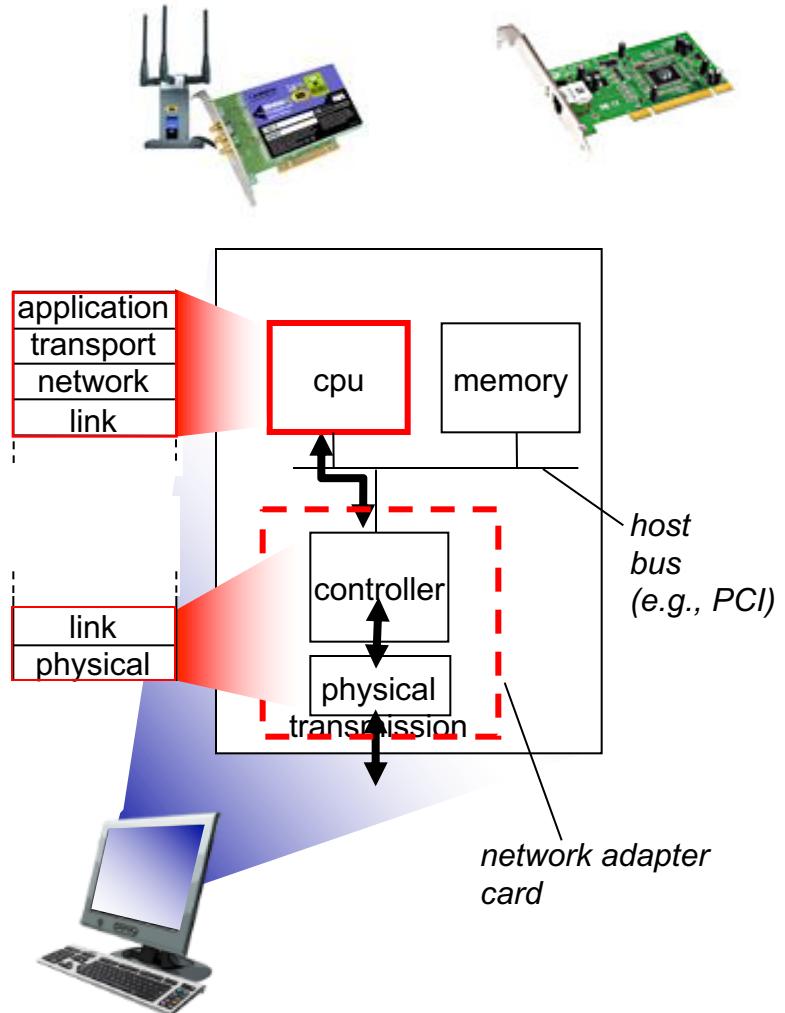
- *framing, link access:*
 - encapsulate datagram into frame, adding header, trailer
 - channel access if shared medium
 - “MAC” addresses used in frame headers to identify source, destination
 - different from IP address!
- *reliable delivery between adjacent nodes*
 - we learned how to do this already (chapter 3)!
 - seldom used on low bit-error link (fiber, some twisted pair)
 - wireless links: high error rates

Link layer services (more)

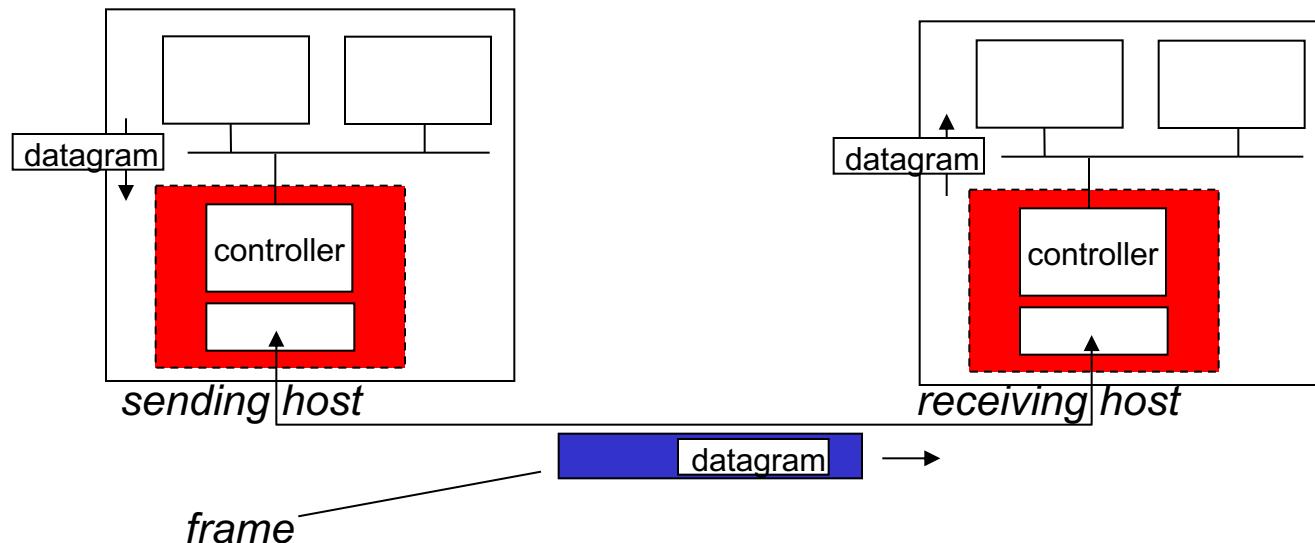
- ***flow control:***
 - pacing between adjacent sending and receiving nodes
- ***error detection:***
 - errors caused by signal attenuation, noise.
 - receiver detects presence of errors:
 - signals sender for retransmission or drops frame
- ***error correction:***
 - receiver identifies *and corrects* 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, physical layer
- attaches into host’s system buses
- combination of hardware, software, firmware



Adaptors communicating



- **sending side:**
 - encapsulates datagram in frame
 - adds error checking bits, rdt, flow control, etc.
- **receiving side**
 - looks for errors, rdt, flow control, etc.
 - extracts datagram, passes to upper layer at receiving side

Internet checksum (review)

goal: detect “errors” (e.g., flipped bits) in transmitted packet
(note: used at transport layer only)

sender:

- treat segment contents as sequence of 16-bit integers
- checksum: addition (1's complement sum) of segment contents
- sender puts checksum value into UDP checksum field

receiver:

- compute checksum of received segment
- check if computed checksum equals checksum field value:
 - NO - error detected
 - YES - no error detected.
But maybe errors nonetheless?

Multiple access links, protocols

two types of “links”:

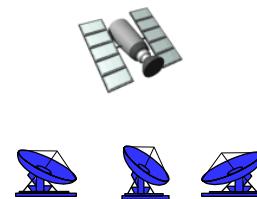
- **point-to-point**
 - PPP for dial-up access
 - point-to-point link between Ethernet switch, host
- **broadcast (shared wire or medium)**
 - old-fashioned Ethernet
 - upstream HFC
 - 802.11 wireless LAN



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 node receives two or more signals at the same time

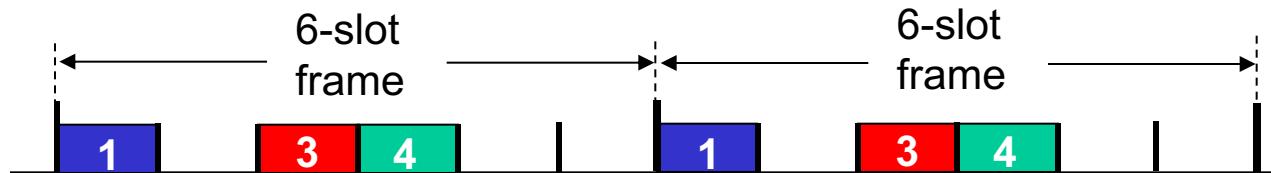
multiple access protocol

- 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

Channel partitioning MAC protocols: TDMA

TDMA: time division multiple access

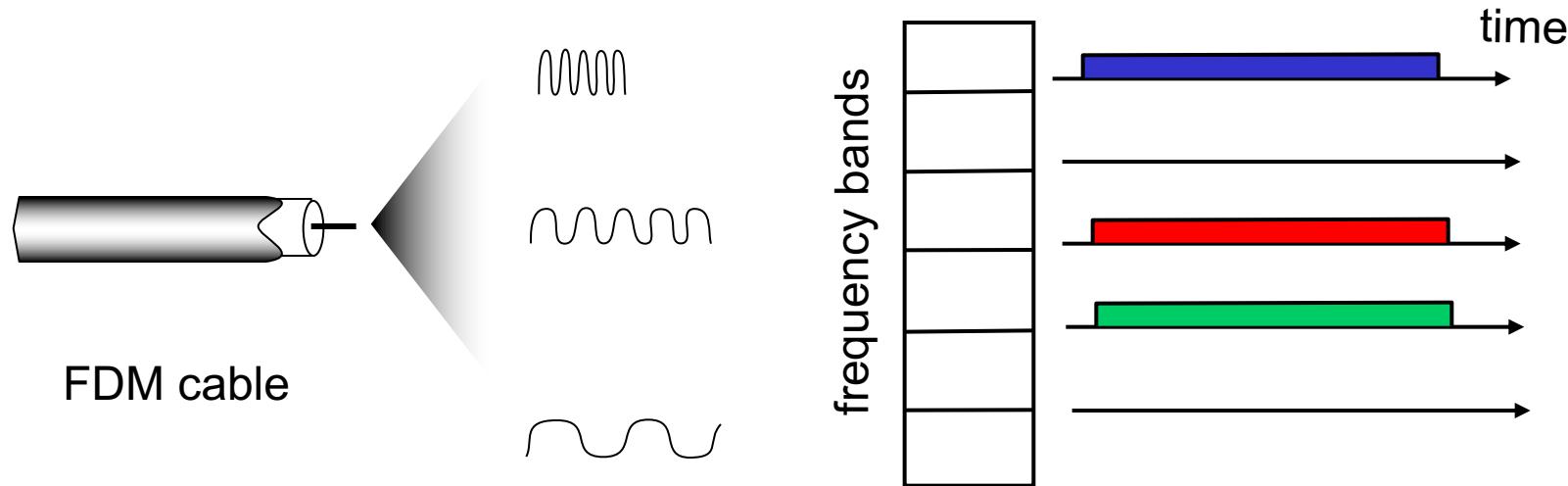
- access to channel in "rounds"
- each station gets fixed length slot (length = packet transmission time) in each round
- unused slots go idle
- example: 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle



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

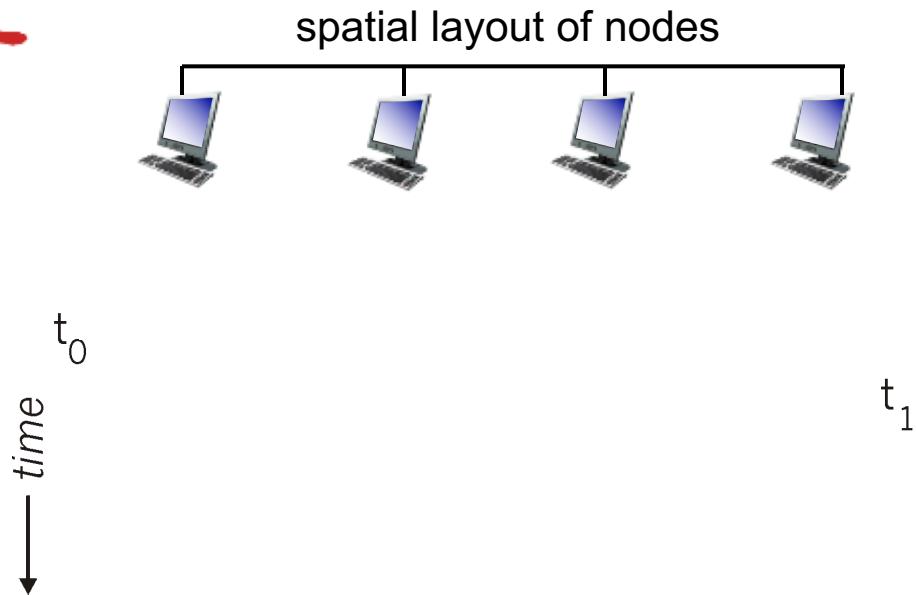
- when node has packet to send
 - transmit at full channel data rate R.
 - no *a priori* coordination among nodes
- two or more transmitting nodes → “collision”,
- **random access MAC protocol** specifies:
 - how to detect collisions
 - how to recover from collisions (e.g., via delayed retransmissions)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

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/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, received signals
 - difficult in wireless LANs: received signal strength overwhelmed by local transmission strength

Ethernet CSMA/CD algorithm

1. 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

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

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

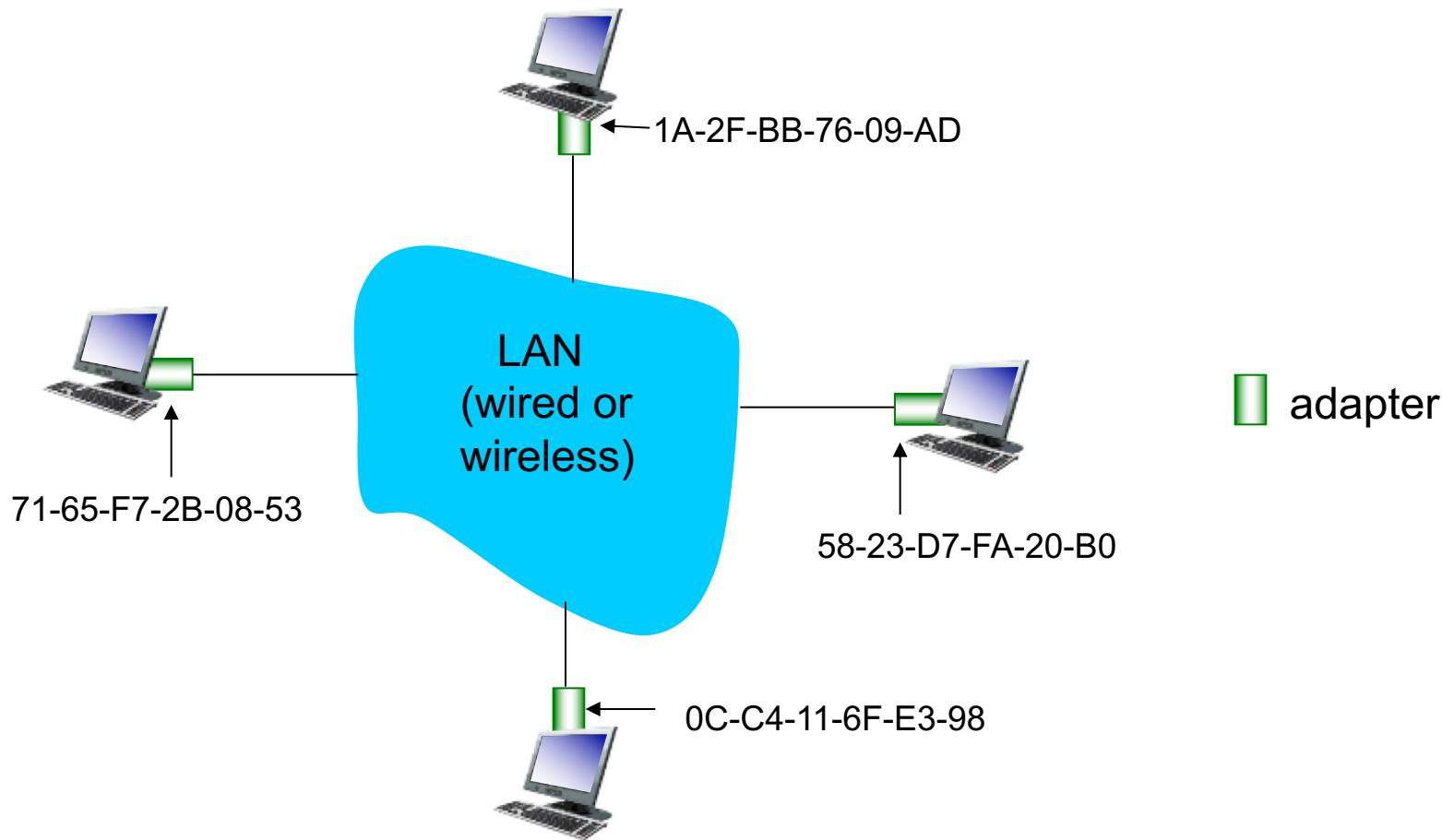
6.7 a day in the life of a
web request

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

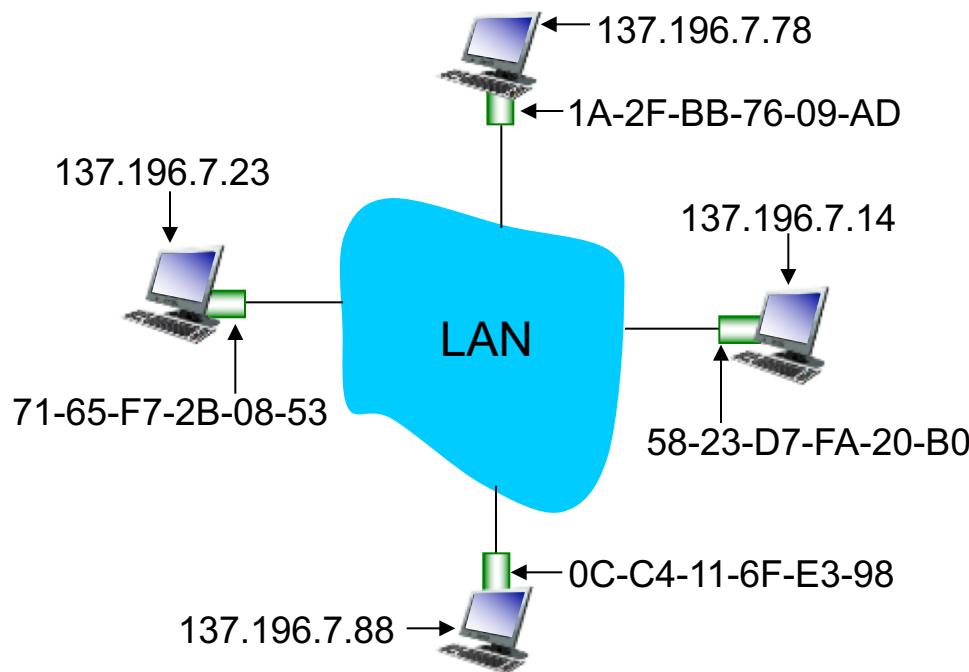


LAN addresses (more)

- 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)

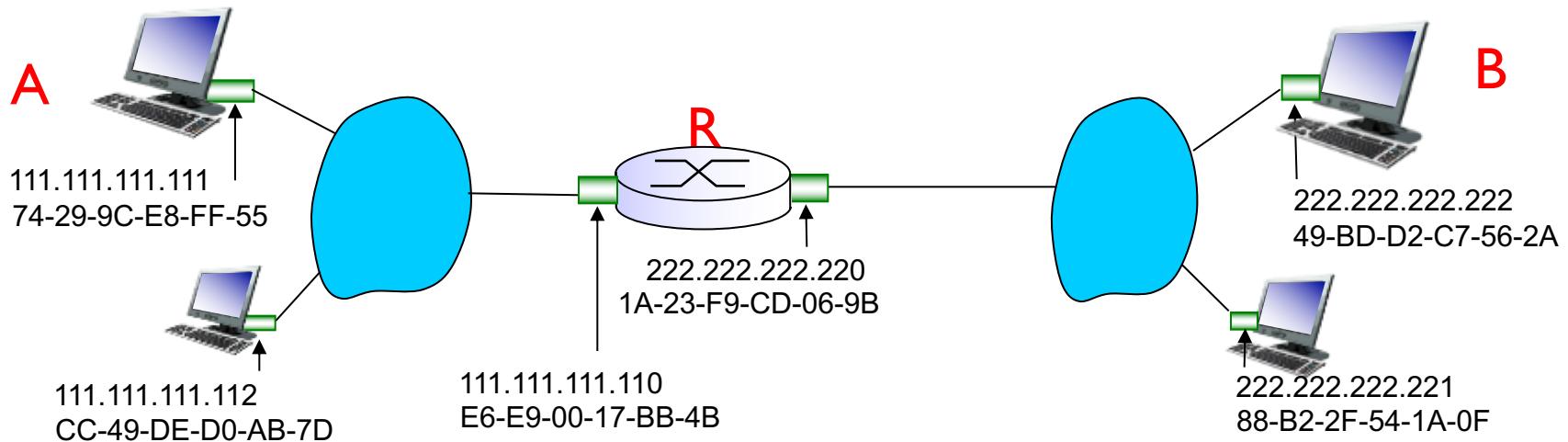
ARP protocol: same LAN

- 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

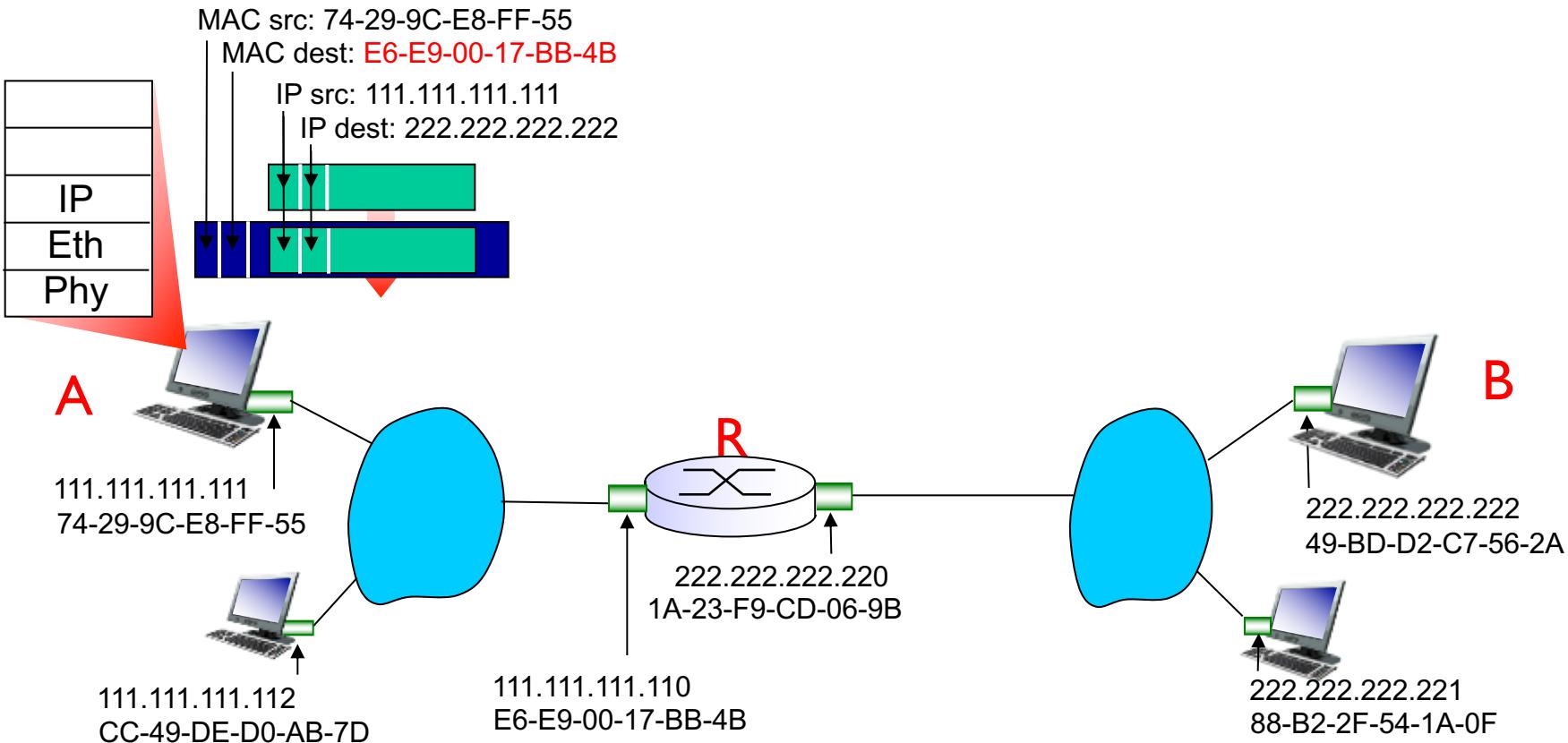
walkthrough: send datagram from A to B via R

- 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?)



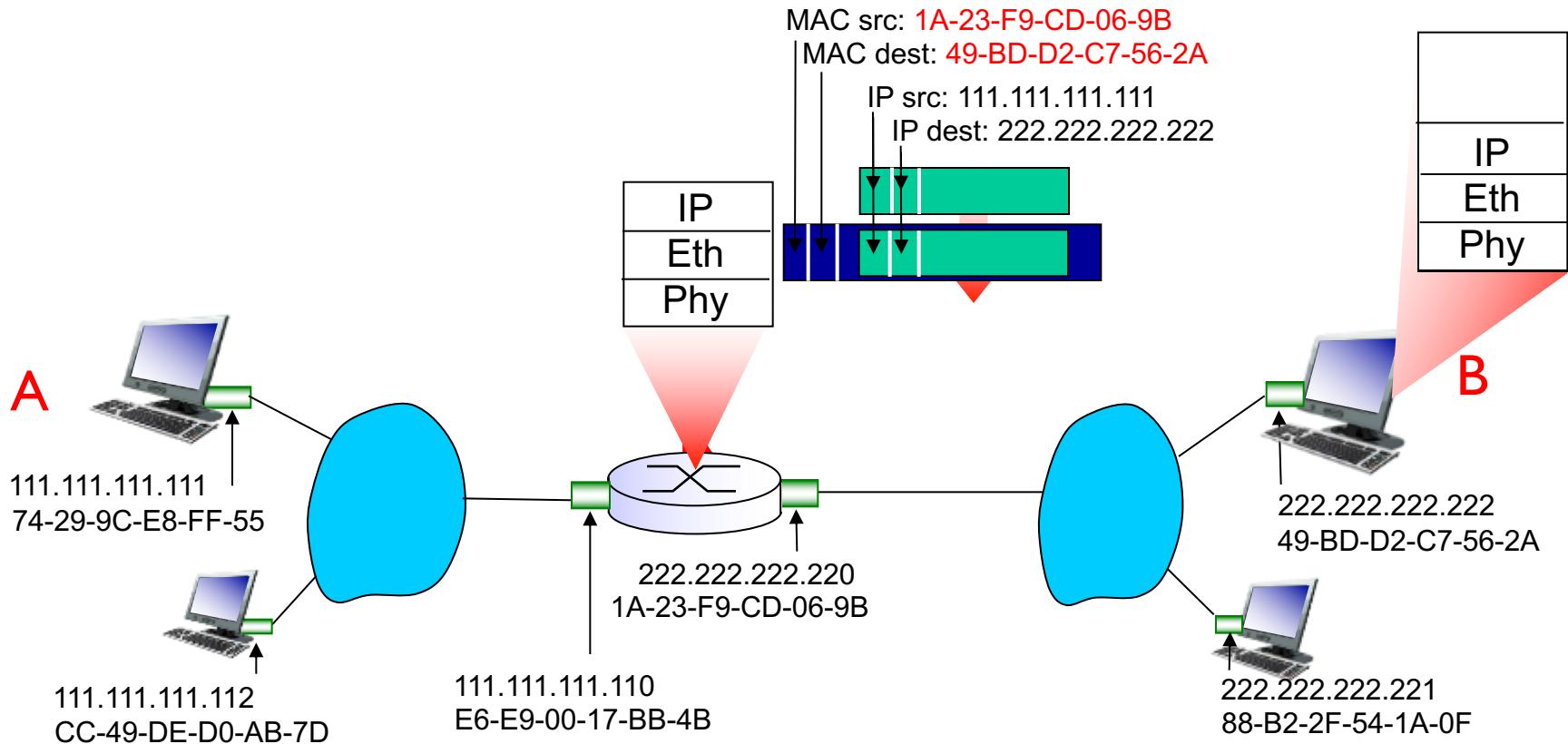
Addressing: routing to another LAN

- 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



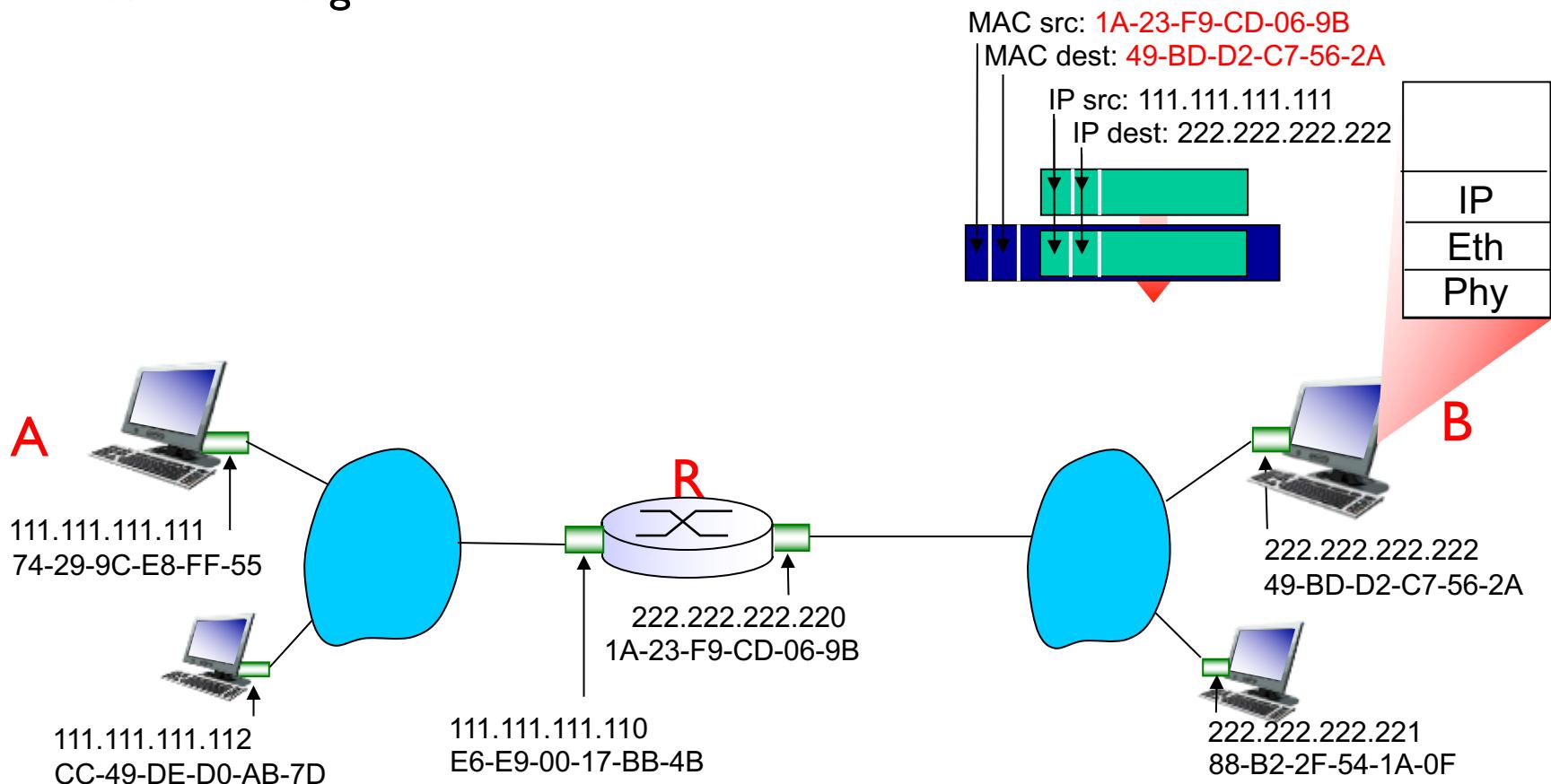
Addressing: routing to another LAN

- 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

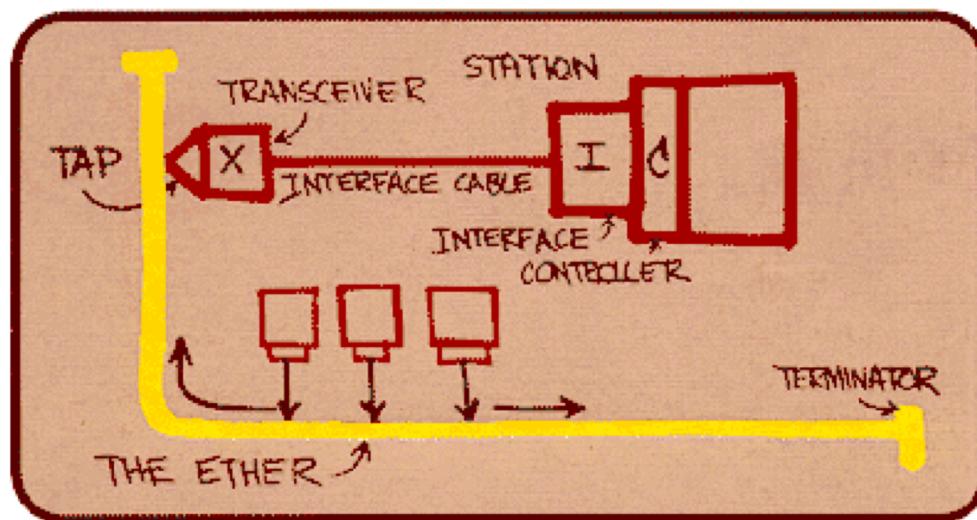


* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive/

Ethernet

“dominant” wired LAN technology:

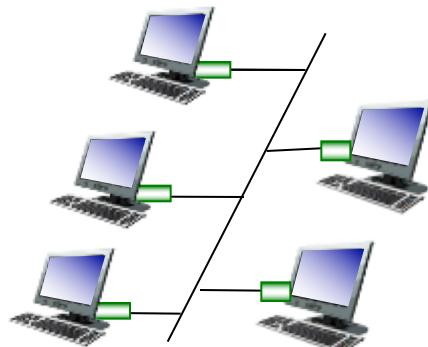
- single chip, multiple speeds (e.g., Broadcom BCM5761)
- first widely used LAN technology
- simpler, cheap
- kept up with speed race: 10 Mbps – 10 Gbps



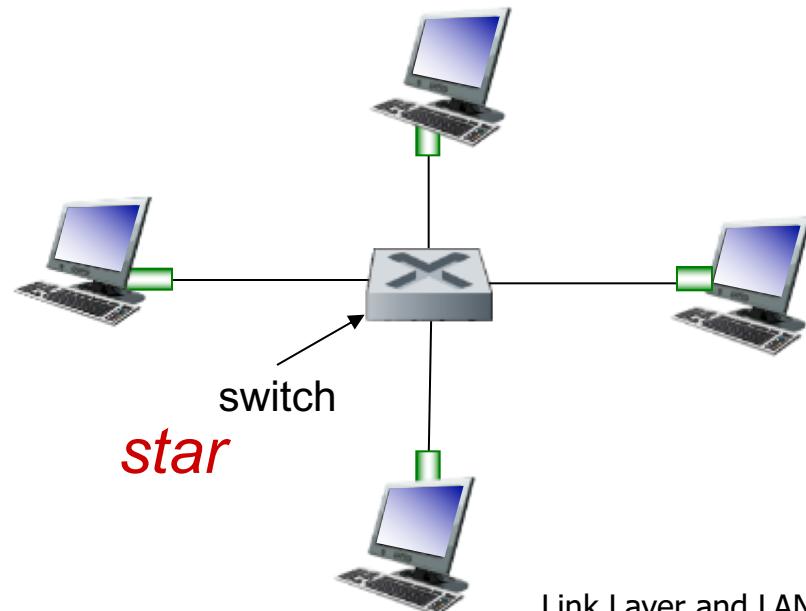
Metcalfe's Ethernet sketch

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)

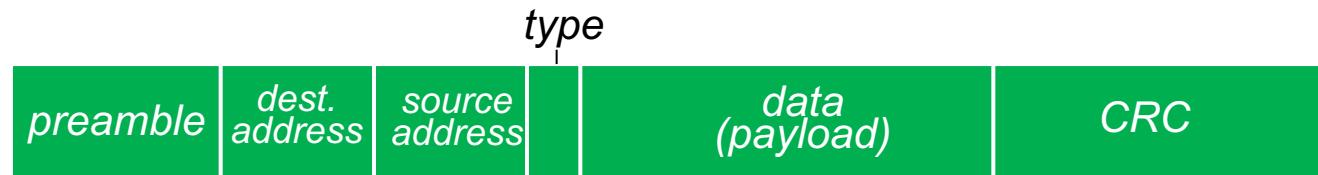


bus: coaxial cable



Ethernet frame structure

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

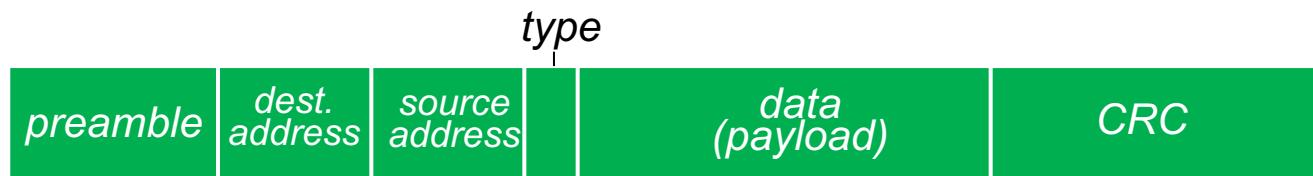


preamble:

- 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

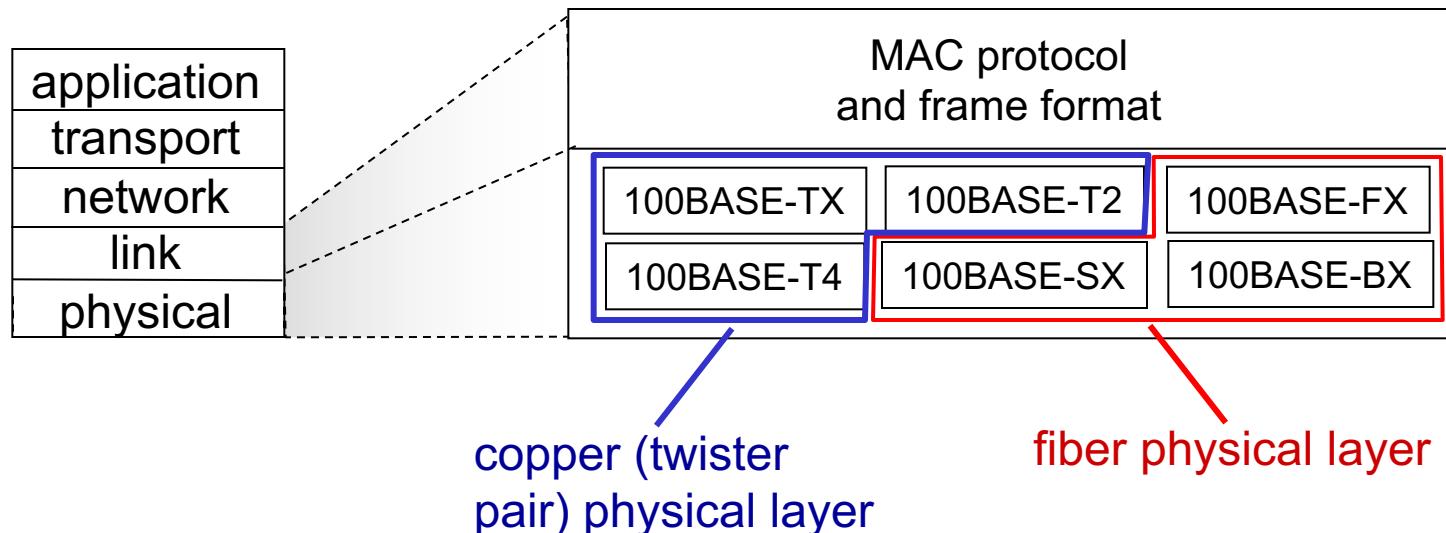


Ethernet: unreliable, connectionless

- ***connectionless***: no handshaking between sending and receiving NICs
- ***unreliable***: receiving NIC doesn't send 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, 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

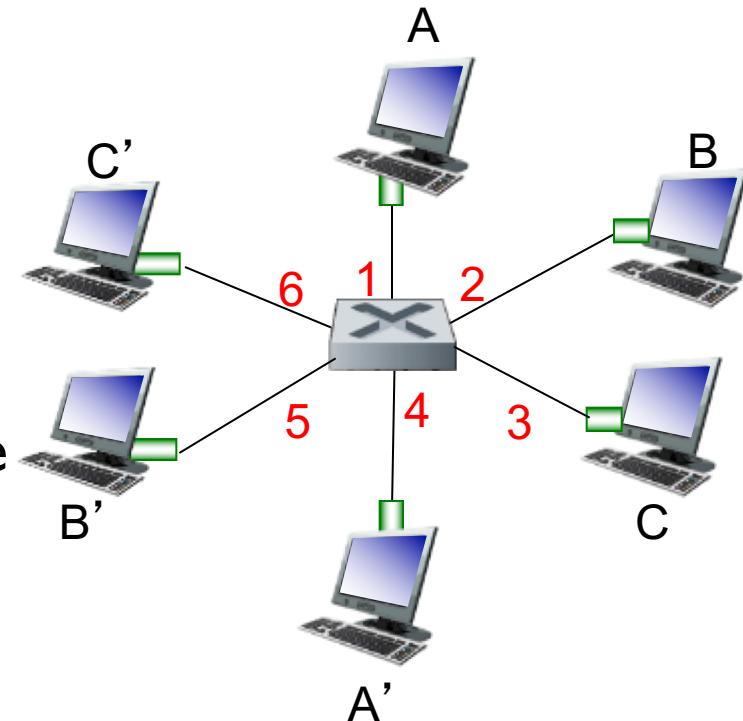
- link-layer device: takes an *active role*
 - store, forward Ethernet frames
 - examine incoming frame's MAC address,
selectively forward frame to one-or-more outgoing links when frame is to be forwarded on segment, uses CSMA/CD to access segment
- *transparent*
 - hosts are unaware of presence of switches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

Switch 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)*

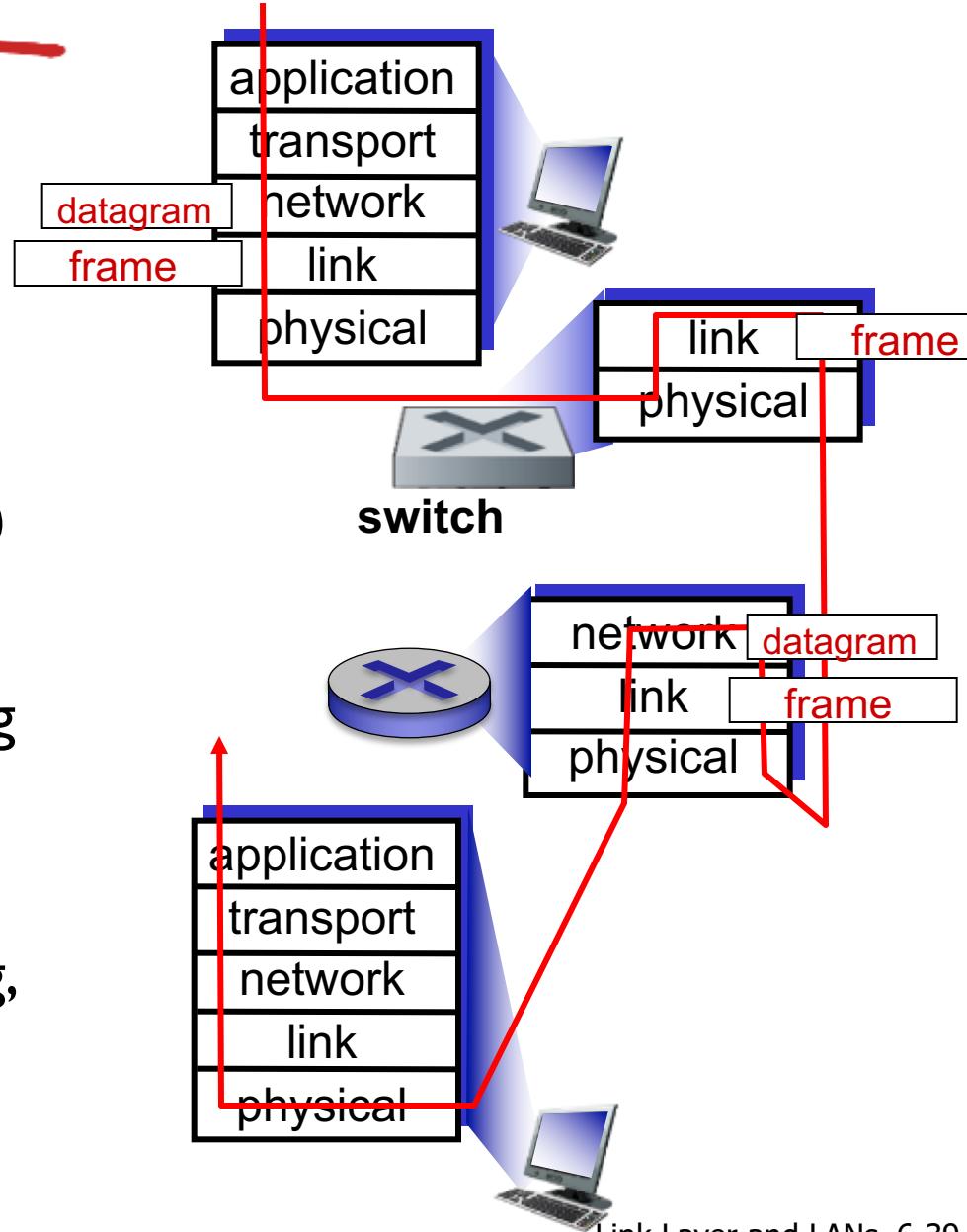
Switches vs. routers

both are store-and-forward:

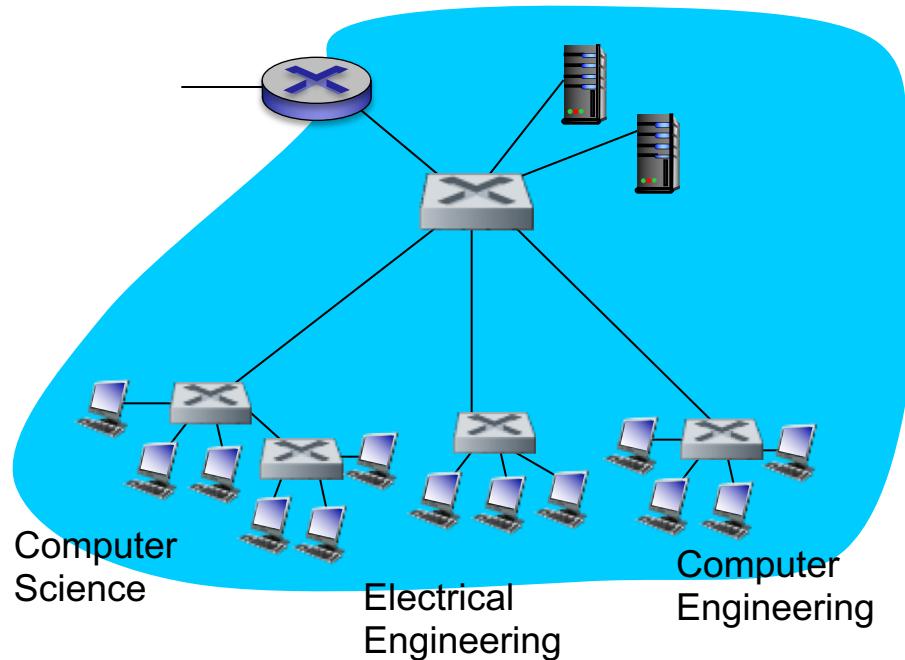
- **routers:** network-layer devices (examine network-layer headers)
- **switches:** link-layer devices (examine link-layer headers)

both have forwarding tables:

- **routers:** compute tables using routing algorithms, IP addresses
- **switches:** learn forwarding table using flooding, learning, MAC addresses



VLANs: motivation



consider:

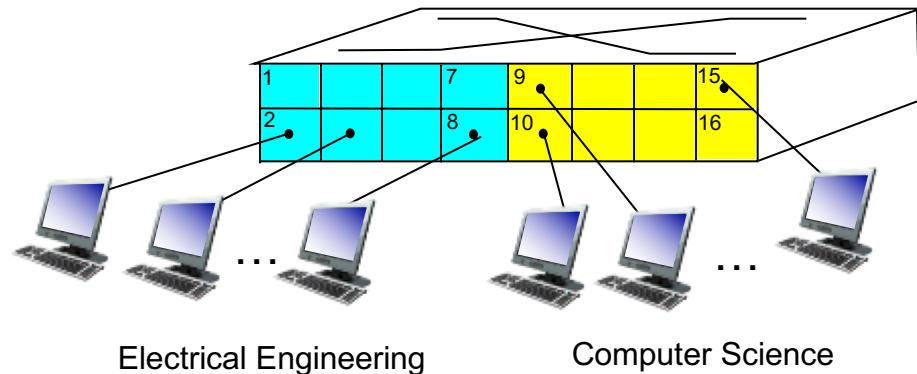
- CS user moves office to EE, but wants connect to CS switch?
- single broadcast domain:
 - all layer-2 broadcast traffic (ARP, DHCP, unknown location of destination MAC address) must cross entire LAN
 - security/privacy, efficiency issues

VLANs

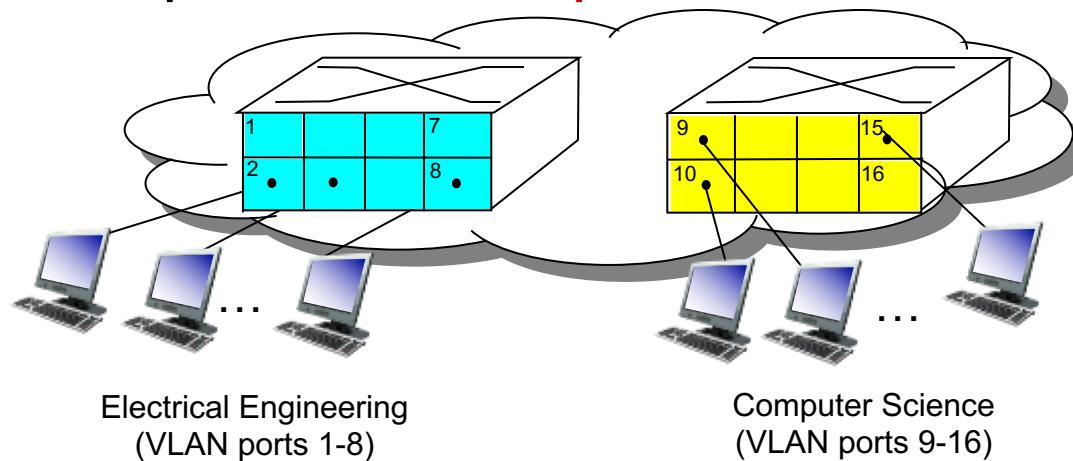
Virtual Local Area Network

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

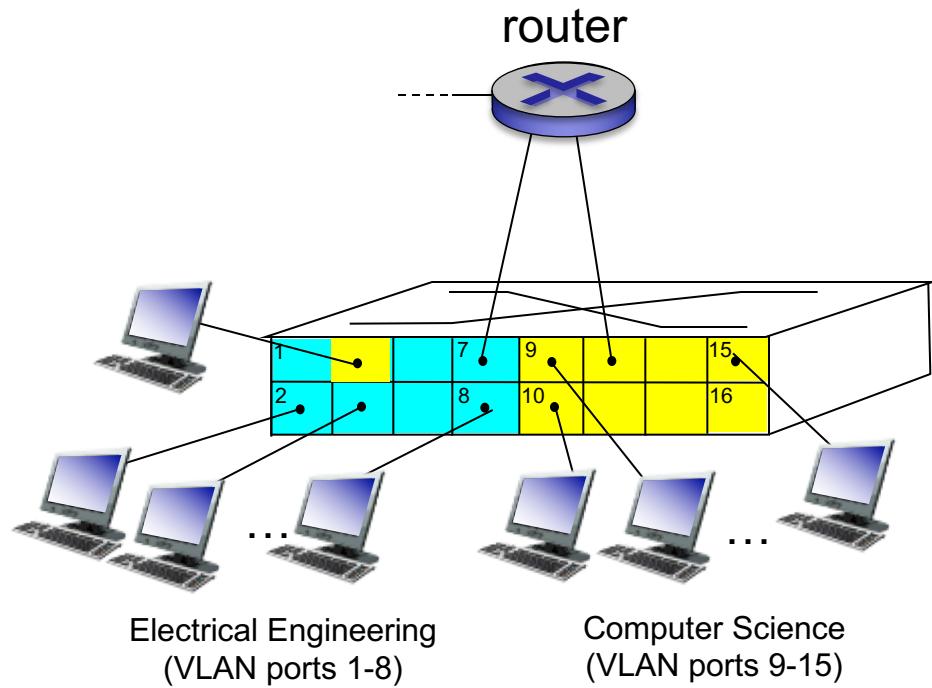


... operates as *multiple* virtual switches

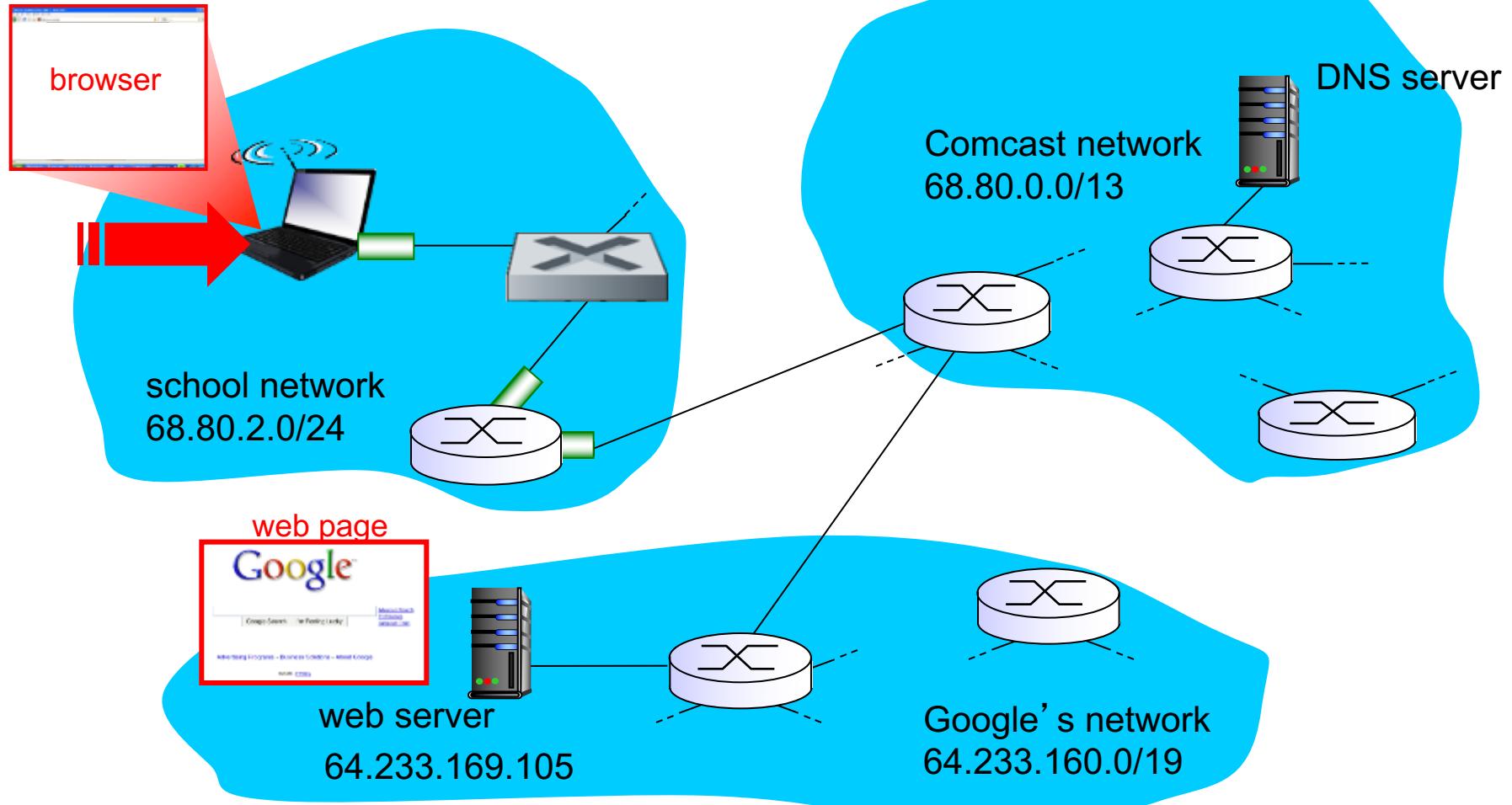


Port-based VLAN

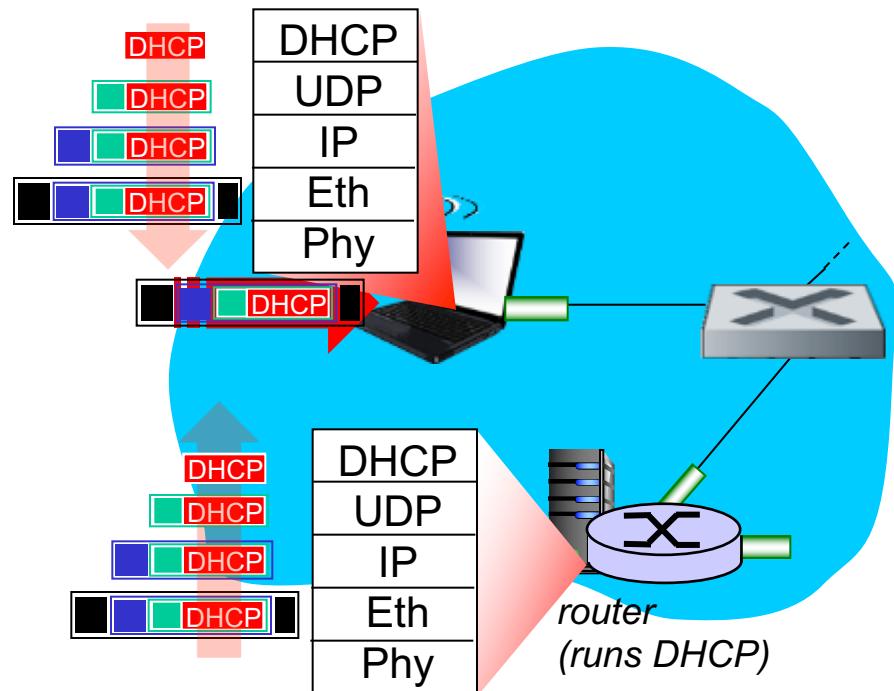
- **traffic isolation:** frames to/from ports 1-8 can *only* reach ports 1-8
 - can also define VLAN based on MAC addresses of endpoints, rather than switch port
- **dynamic membership:** ports can be dynamically assigned among VLANs
- **forwarding between VLANs:** done via routing (just as with separate switches)
 - in practice vendors sell combined switches plus routers



A day in the life: scenario

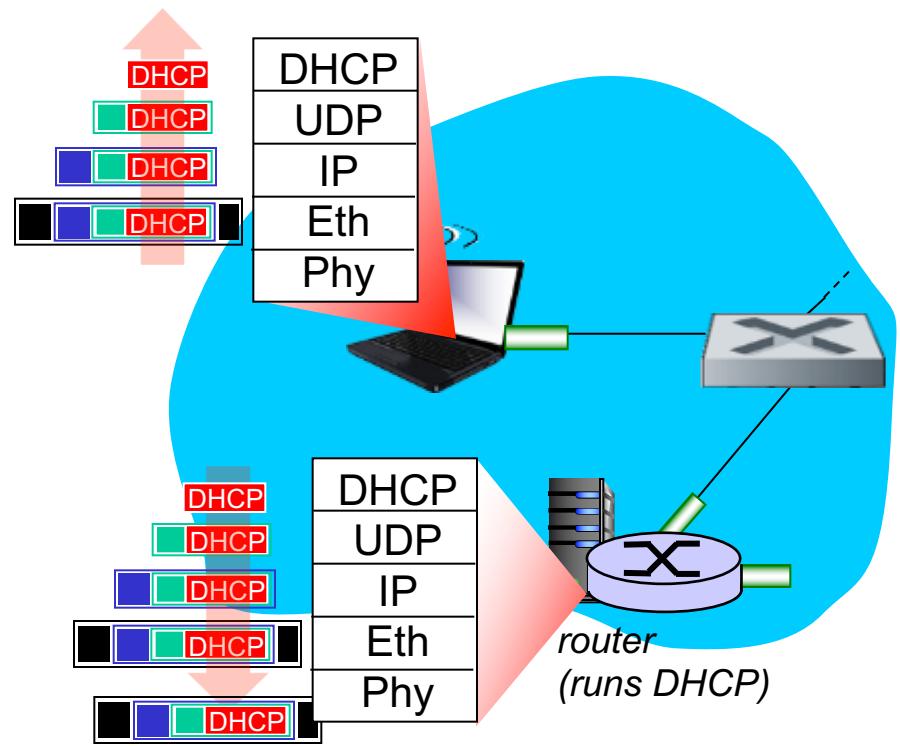


A day in the life... connecting to the Internet



- 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

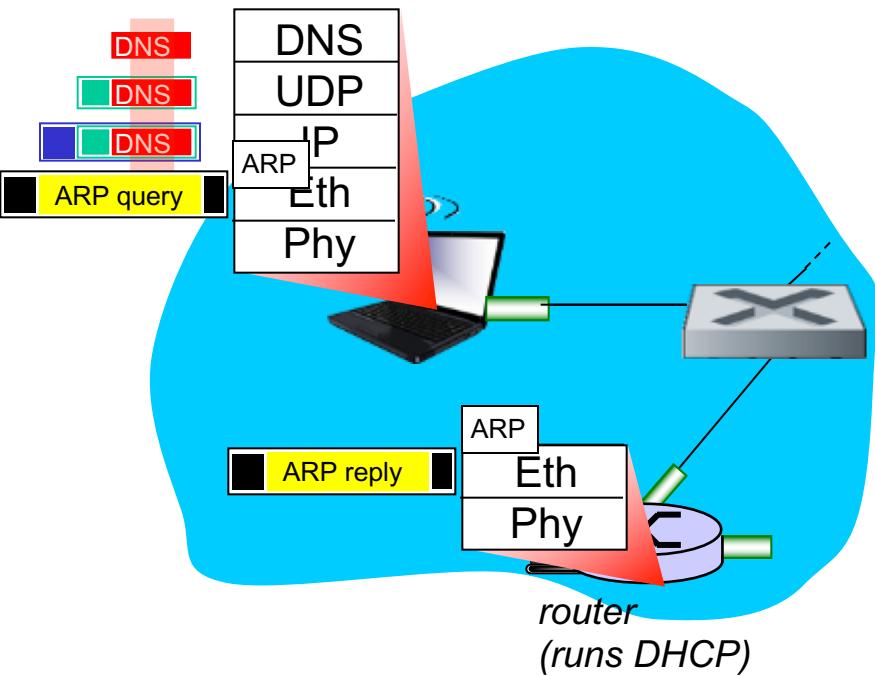
A day in the life... connecting to the Internet



- 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

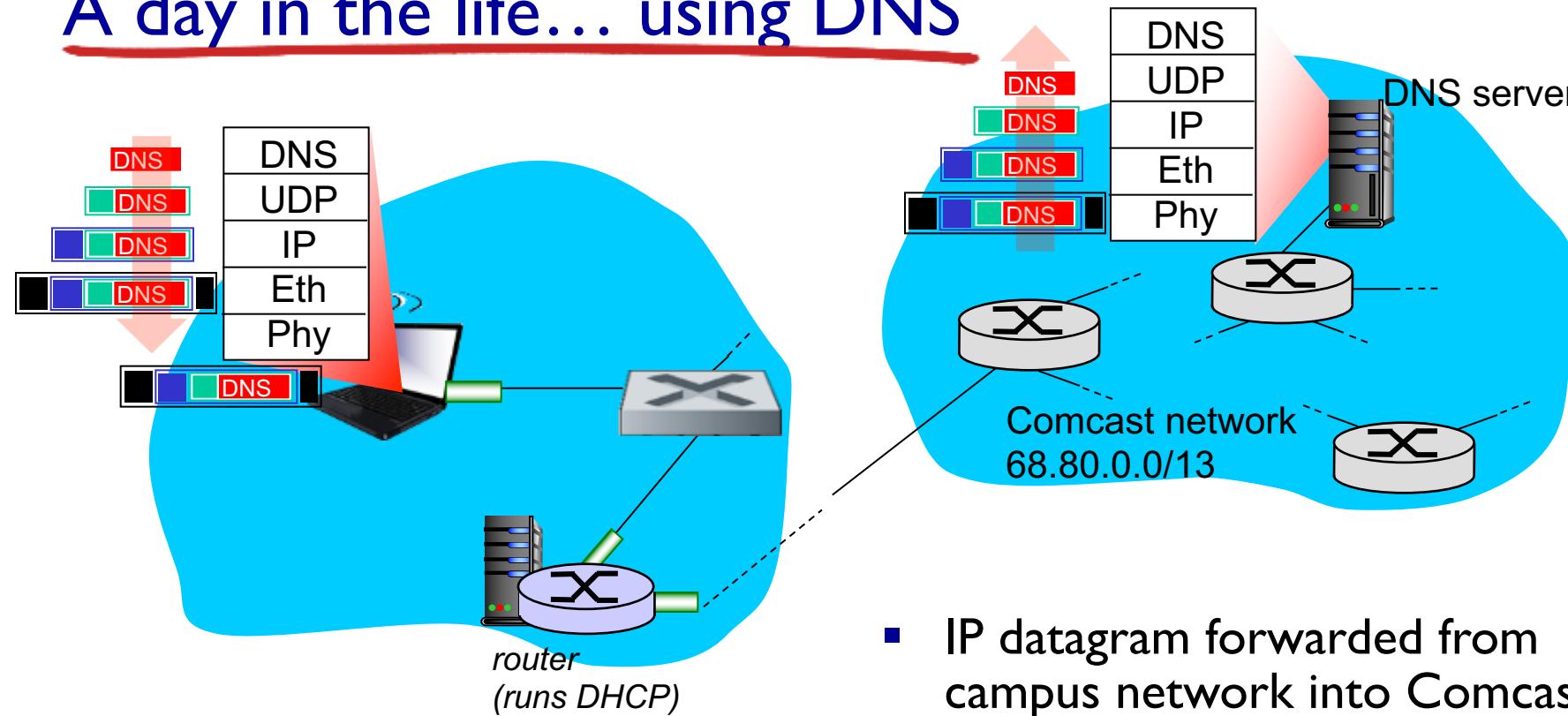
Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

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

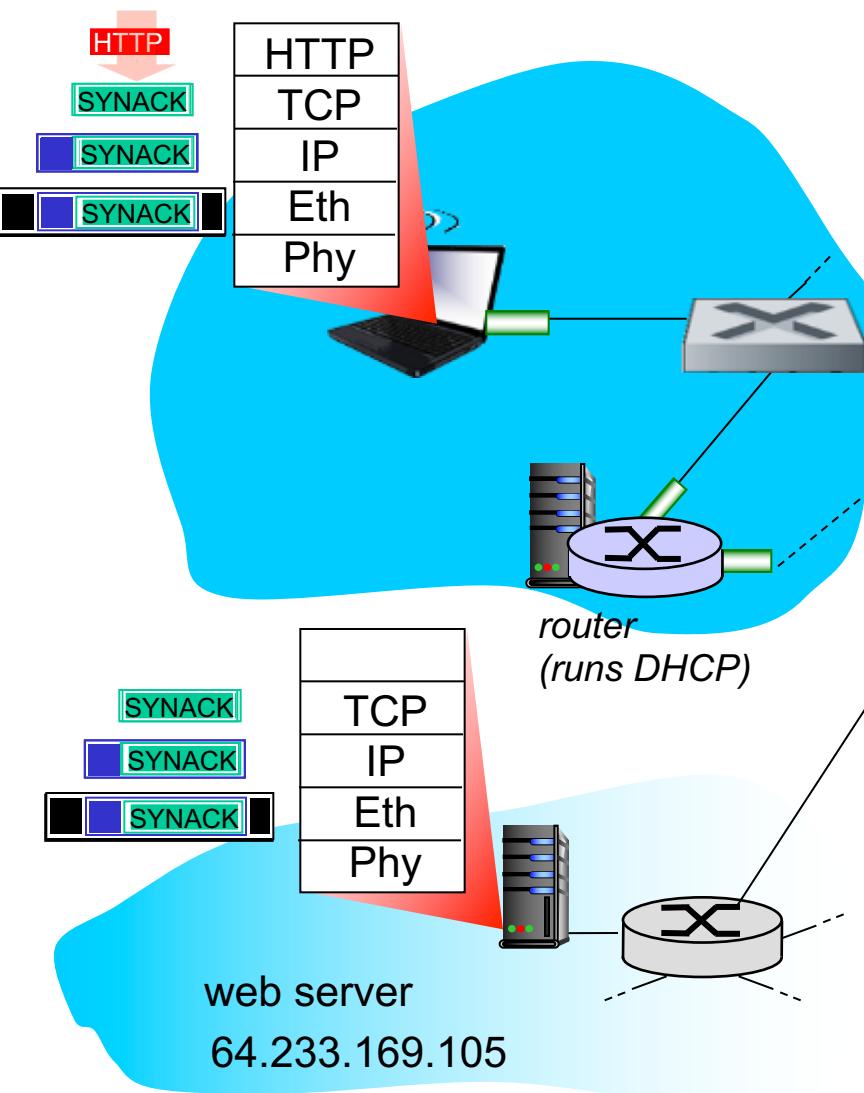
A day in the life... using DNS



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

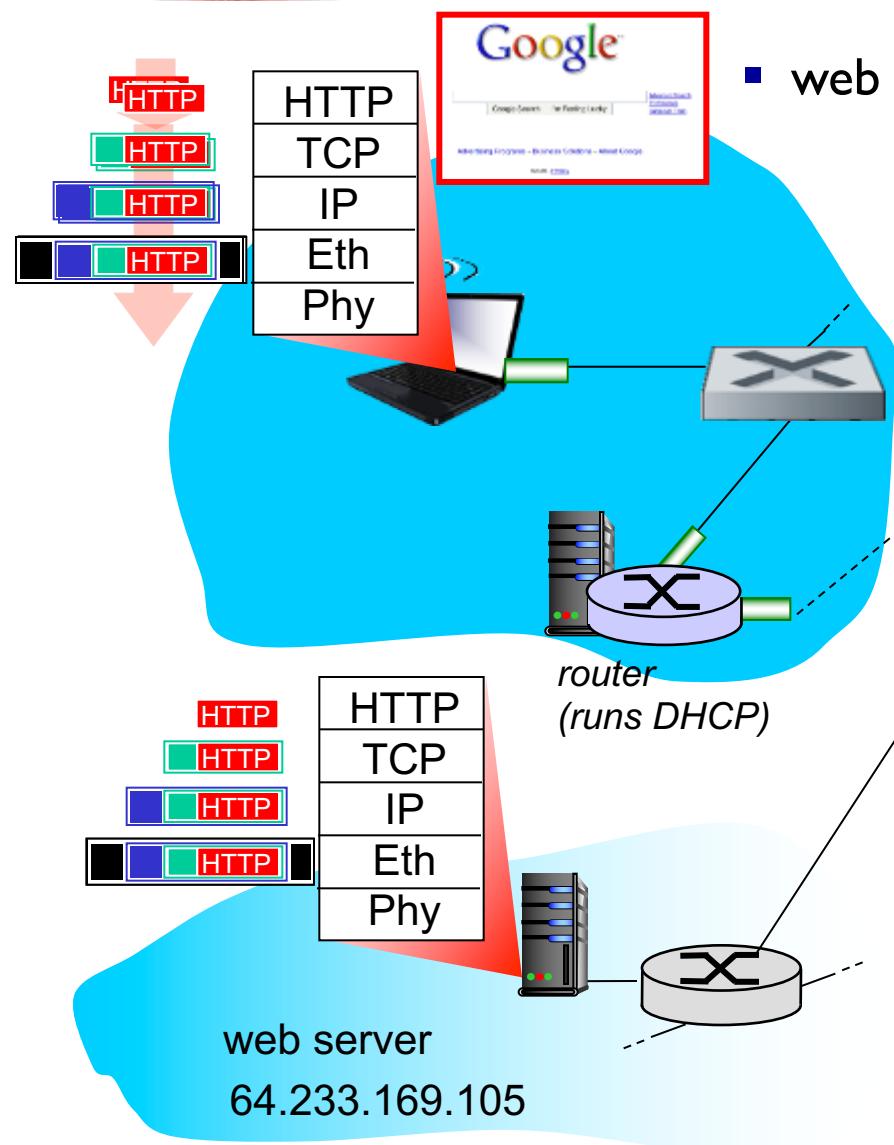
- IP datagram forwarded from campus network into Comcast network, routed (tables created by **RIP, OSPF, IS-IS** and/or **BGP** routing protocols) to DNS server
- demuxed to DNS server
- DNS server replies to client with IP address of www.google.com

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



- 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 6: Summary

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

Chapter 6: let's take a breath

- journey down protocol stack *complete*
- solid understanding of networking principles, practice
- could stop here but *lots* of interesting topics!
 - wireless
 - multimedia
 - security