

CS 305: Computer Networks

Fall 2024

Link Layer

Ming Tang

Department of Computer Science and Engineering
Southern University of Science and Technology (SUSTech)

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

Multiple access links, protocols

two types of “links”:

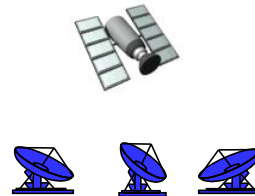
- point-to-point
 - PPP for dial-up access
 - point-to-point link between Ethernet switch, host
- *broadcast (shared wire or medium)*
 - old-fashioned Ethernet
 - 802.11 wireless LAN



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 which and 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

Desired properties:

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

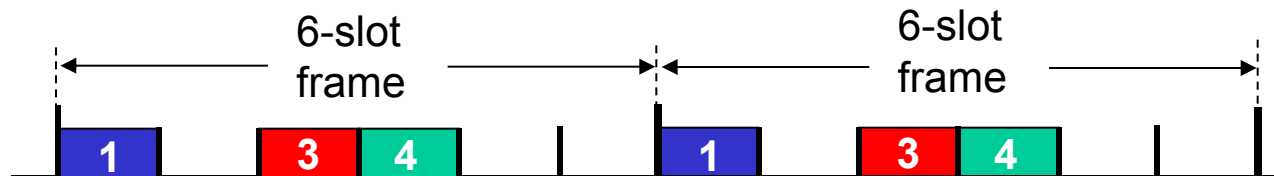
three broad classes:

- *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

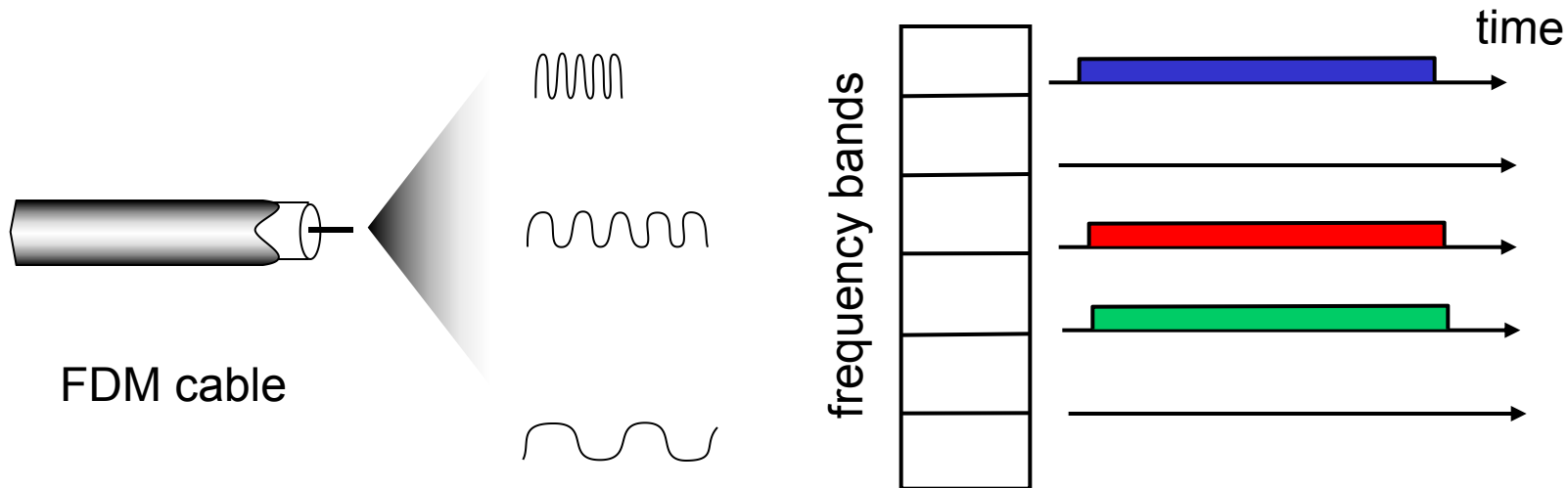
- access to channel in "rounds"
- each station gets fixed length slot (length = packet transmission time) in each round
- unused slots go idle
- example: 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle



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



Channel partitioning: limitations

Desired properties:

1. **【x】** 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

Random access protocols

- Random access; if fails, wait for a random time
- 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)
- **Advantage:** when node has packet to send
 - transmit at full channel data rate R .
 - no *a priori* coordination among nodes
- examples of random access MAC protocols:
 - slotted ALOHA
 - ALOHA
 - CSMA, CSMA/CD (Ethernet), CSMA/CA (802.11)

Slotted ALOHA

assumptions:

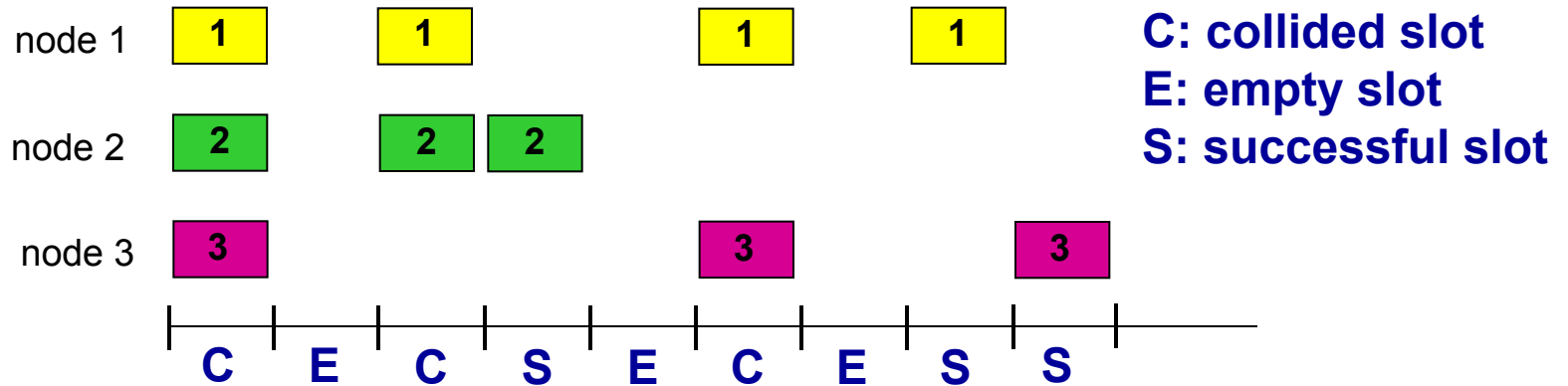
- all frames same size
- time divided into equal size slots (time to transmit 1 frame)
- nodes start to transmit only slot beginning
- nodes are synchronized
- if 2 or more nodes transmit in slot, all nodes detect collision

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 **probability p** until success

Slotted ALOHA



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
- 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:

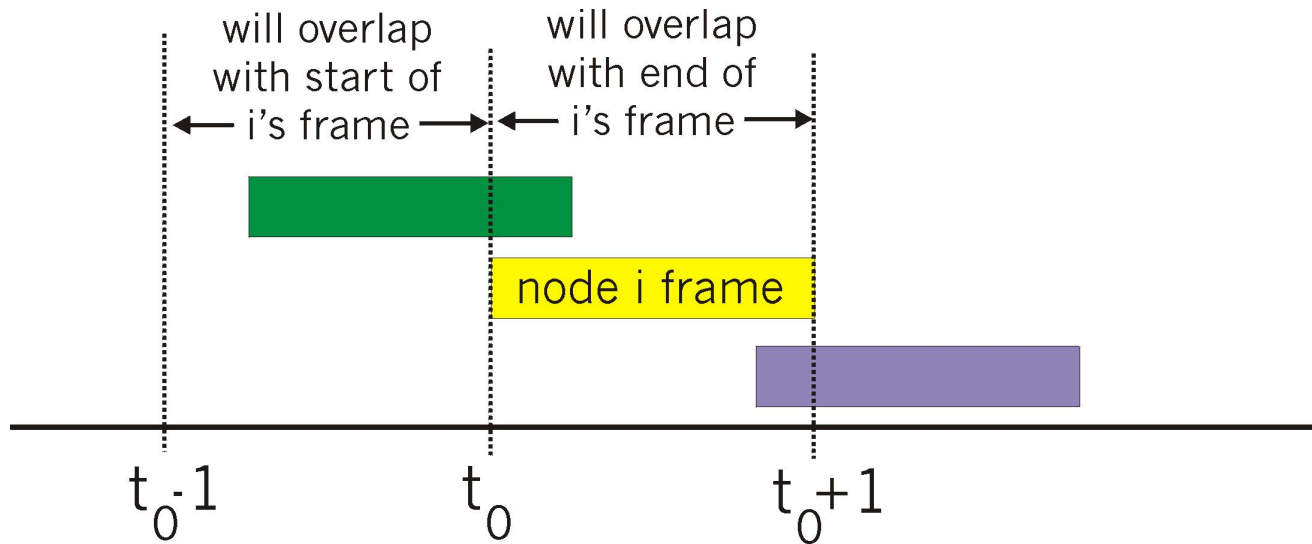
$$\text{max efficiency} = 1/e = 0.37$$

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 $[t_0-1, t_0+1]$



Pure ALOHA efficiency

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

$P(\text{no other node transmits in } [t_0-1, t_0]) \cdot$

$P(\text{no other node transmits in } [t_0, t_0+1])$

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

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

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

$$= 1/(2e) = 0.18$$

even worse than slotted Aloha!

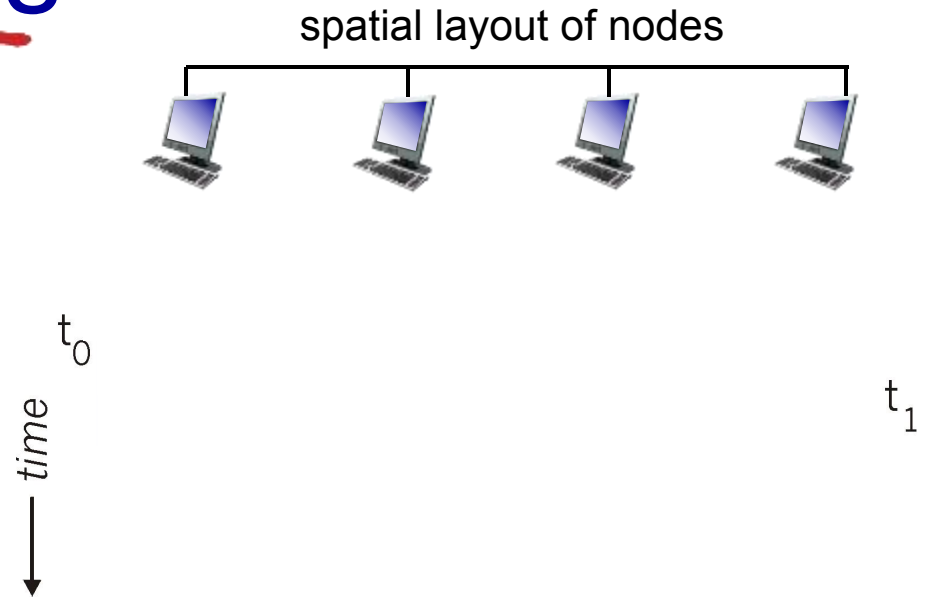
CSMA (carrier sense multiple access)

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

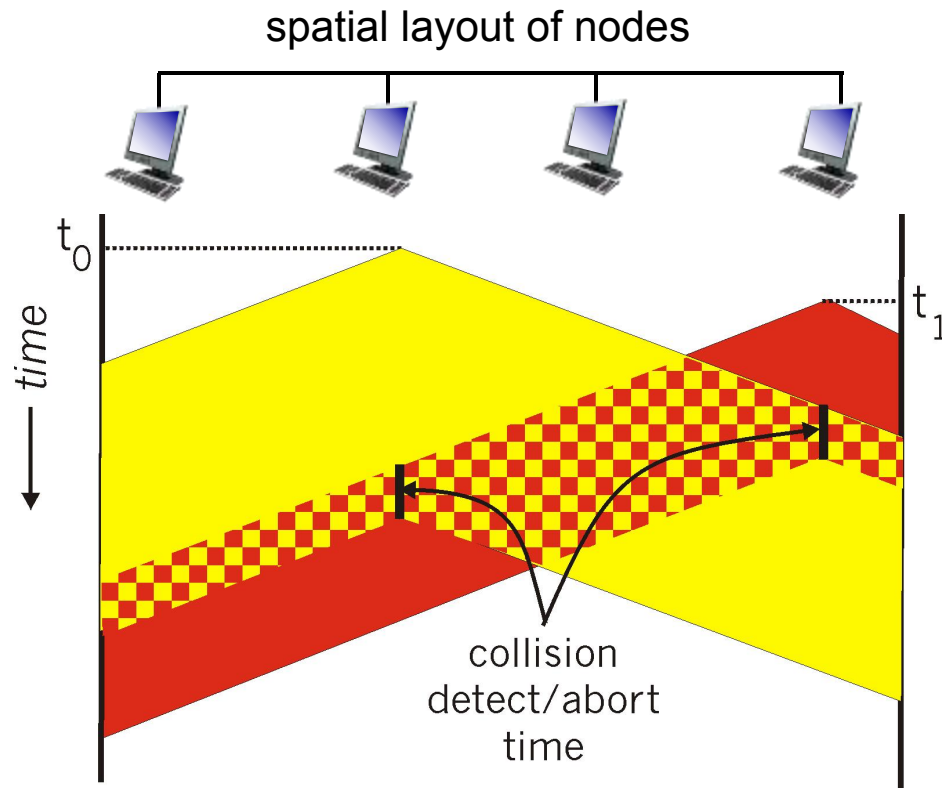


CSMA/CD (collision detection)

CSMA/CD: CSMA+CD (collision detection)

- collision detection → stop talking:
 - collisions *detected* within short time
 - colliding transmissions are aborted, reducing channel wastage
- Suitable scenarios:
 - 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: the polite conversationalist

CSMA/CD (collision detection)



Ethernet CSMA/CD algorithm

1. network adapter (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

“Taking turns” MAC protocols

channel partitioning MAC protocols:

- share channel *efficiently* and *fairly* at high load
- inefficient at low load: delay in channel access, $1/N$ 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

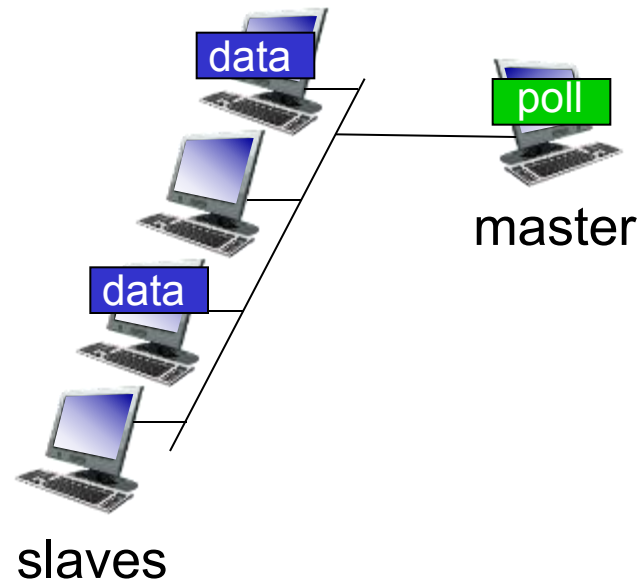
look for best of both worlds!

- when one node wants to transmit, it can send at rate R .
- when M nodes want to transmit, each can send at average rate R/M

“Taking turns” MAC protocols

polling:

- master node
“invites” slave nodes to transmit **in turn**
- maximum number of frames
- concerns:
 - polling overhead
 - single point of failure (master)

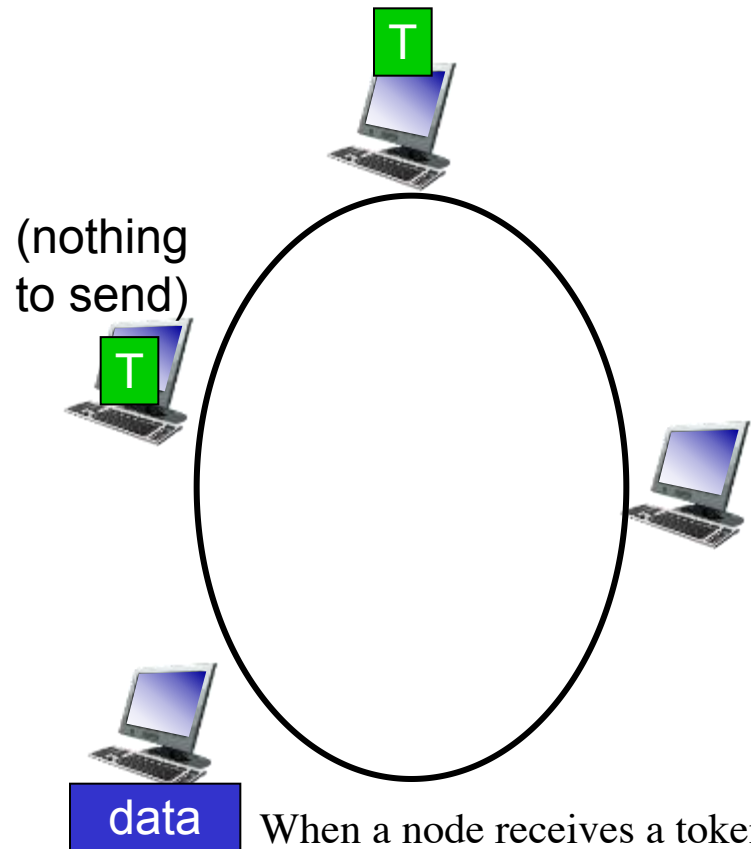


The master node must poll **each** of the nodes in turn no matter they have frames or not

“Taking turns” MAC protocols

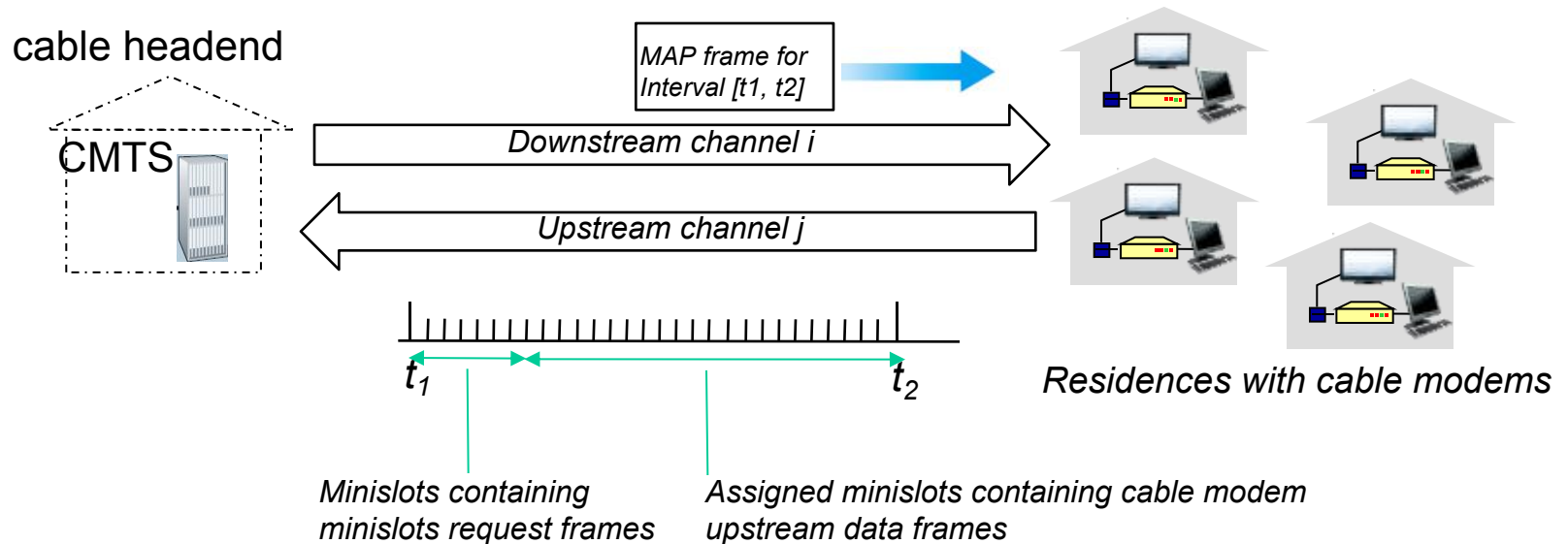
token passing:

- control *token* passed from one node to next sequentially (fixed order).
- distributed
- concerns:
 - token overhead
 - single point of failure (token)



- When a node receives a token,
- if it has some frames to transmit. it holds onto the token;
 - otherwise, it immediately forwards the token to the next node.

Case study: Cable access network



DOCSIS: data over cable service interface specification

- FDM over upstream, downstream frequency channels
 - Downstream (no multiple access); upstream (collision)
- 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

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

Link layer, LANs: outline

6.1 introduction, services

6.2 error detection, correction

6.3 multiple access protocols

6.4 LANs

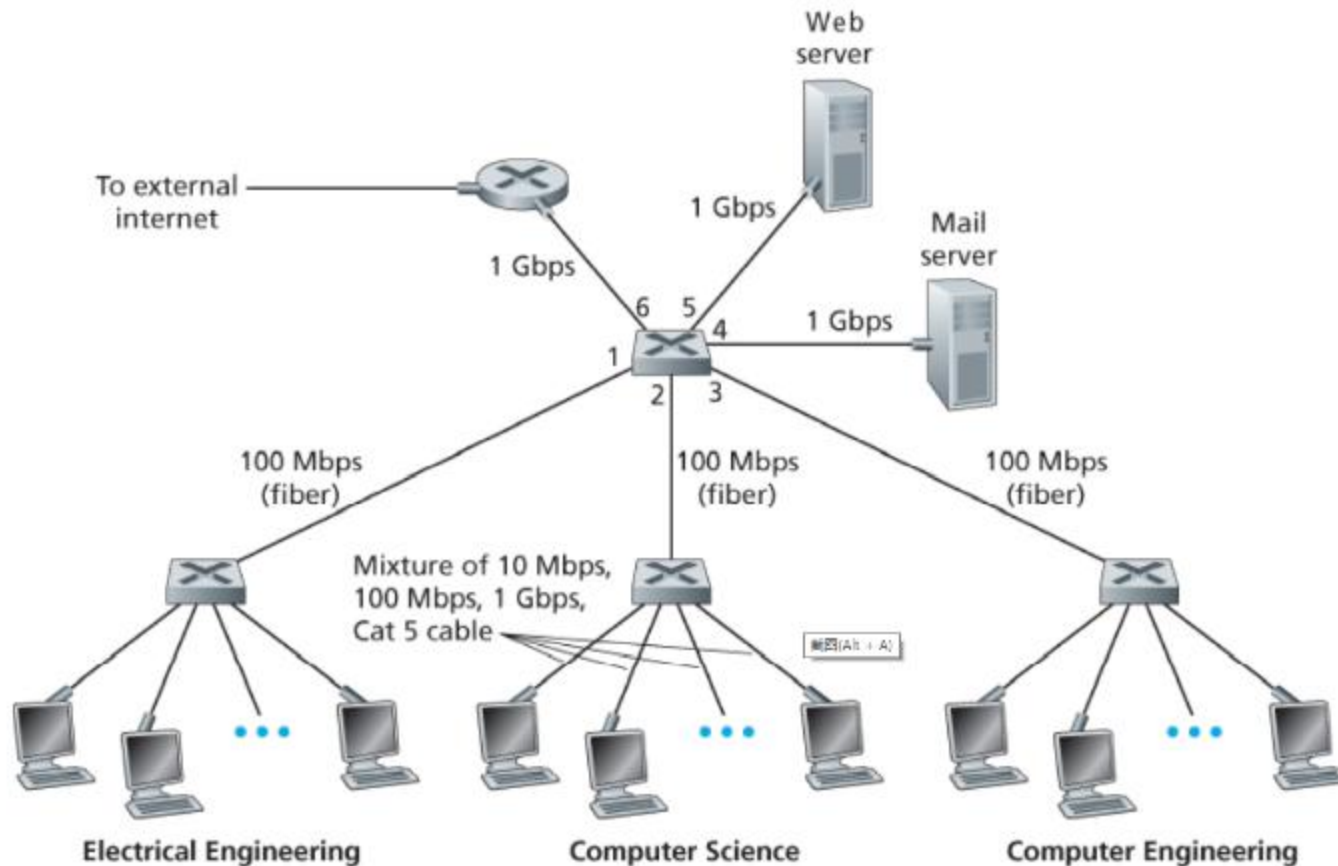
- addressing, ARP
- Ethernet
- switches
- VLANs

6.5 link virtualization: MPLS

6.6 data center networking

6.7 a day in the life of a web request

LANs



Because these switches operate at the link layer,

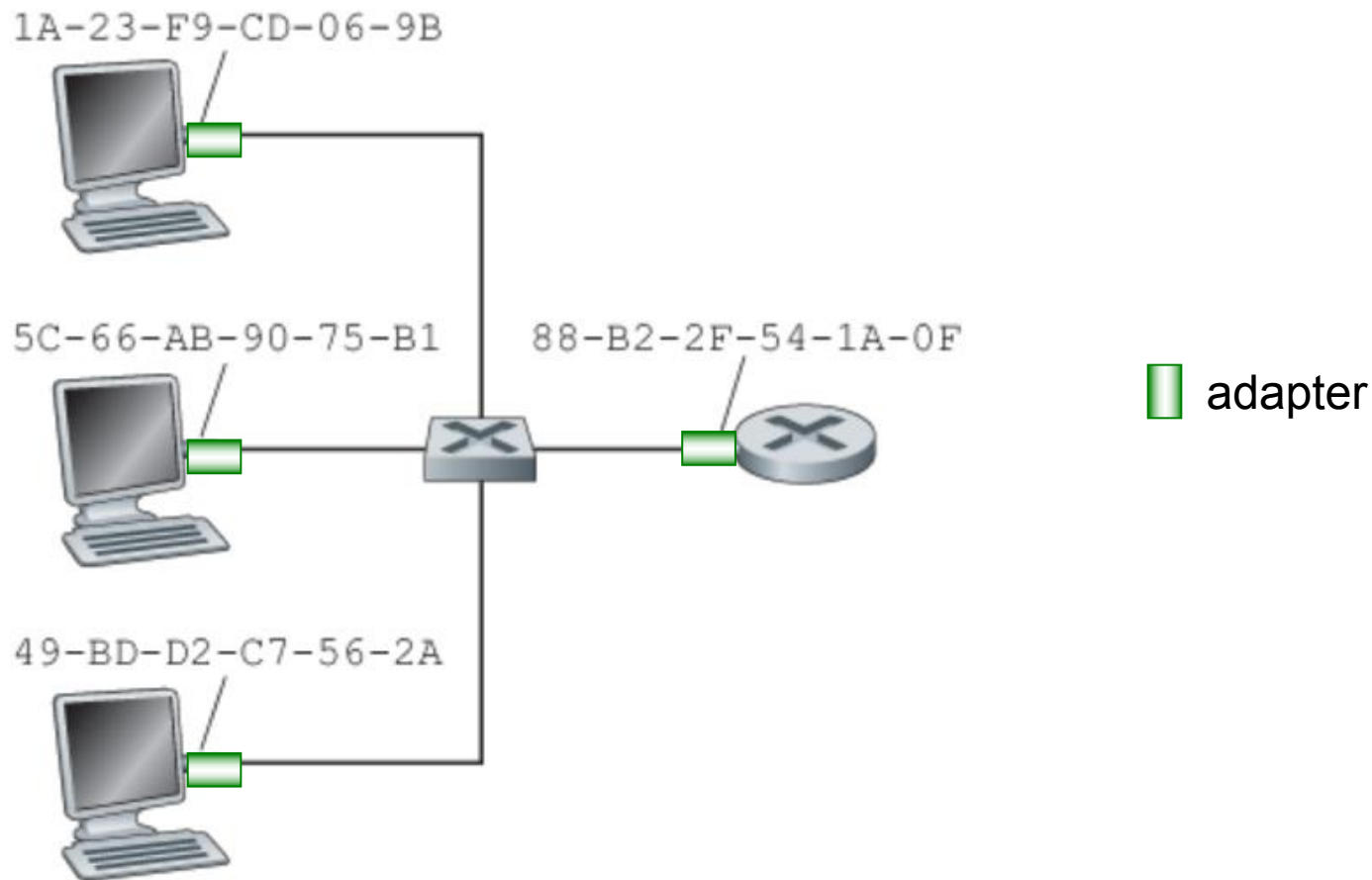
- don't recognize network-layer addresses
- don't use routing algorithms like RIP or OSPF to determine paths through switches

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:
 - Adapter (network interface) rather than host or routers
 - Link-layer switches do NOT have MAC addresses
 - 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; no two adapters have the same address
 - e.g.: 1A-2F-BB-76-09-AD
 - hexadecimal (base 16) notation
 - (each “numeral” represents 4 bits)

LAN addresses and ARP

each adapter on LAN has unique *LAN* address



LAN addresses (more)

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness)
- analogy:
 - MAC address: like ID Number
 - IP address: like postal address
- MAC flat address
 - can move LAN card from one LAN to another
- IP hierarchical address
 - address depends on IP subnet to which node is attached

LAN addresses (more)

- an adapter sends a frame to some destination adapter,
 - inserts the destination adapter's MAC address into the frame and then sends the frame into the LAN
- an adapter receive a frame
 - If there is a **match**, extracts the enclosed datagram and passes the datagram up the protocol stack;
 - If there **isn't a match**, discards
- MAC broadcast address FF-FF-FF-FF-FF-FF

Questions

- How to determine interface's MAC address, knowing its IP address?
- How to send a datagram from one host to another?

ARP: address resolution protocol

Question: how to determine interface's MAC address, knowing its IP address?

ARP: IP → MAC

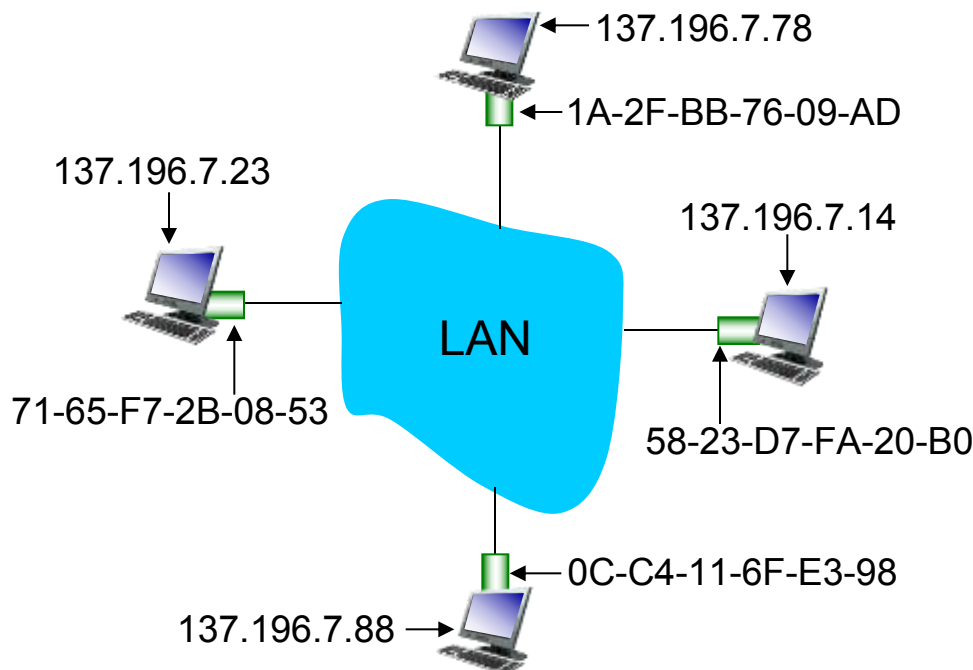
- Resolve addresses only for interfaces on the same subnet

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

Host A wants to send datagram to host B (same subnet)

- B's MAC address not in A's ARP table.

Step 1: A **broadcasts** ARP query packet, containing B's IP address

- ARP packet: sending IP and MAC, receiving IP and MAC
- destination MAC address = FF-FF-FF-FF-FF-FF
- all nodes on LAN receive ARP query

Step 2: 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)

ARP is “plug-and-play”: nodes create their ARP tables *without intervention from net administrator*

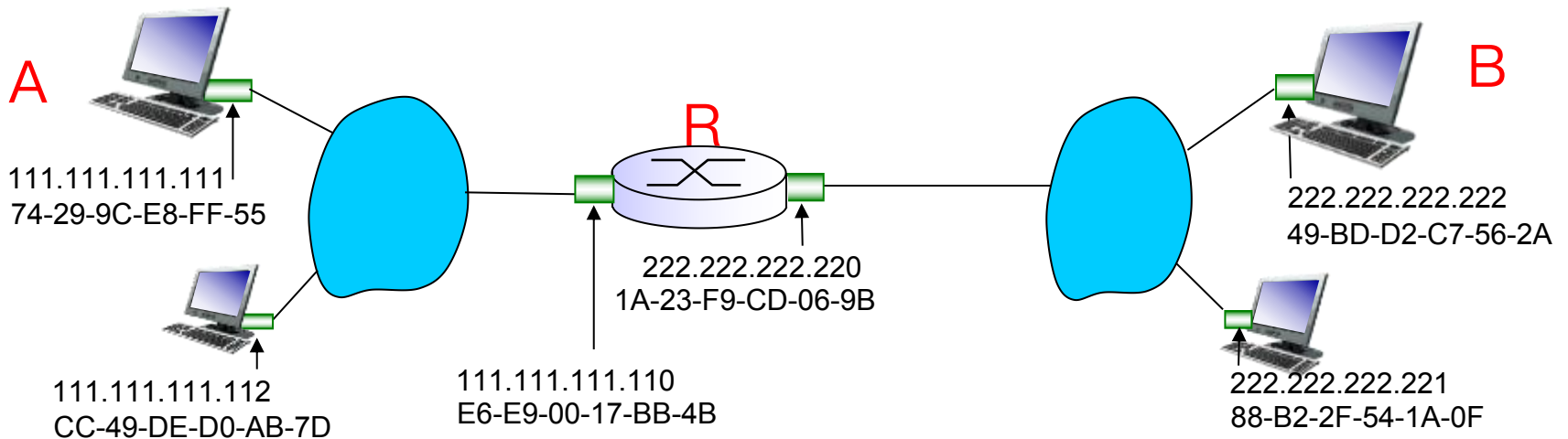
Questions

- How to determine interface's MAC address, knowing its IP address?
- How to send a datagram from one host to another?
 - Same subnet: framing with destination MAC; send it
 - Different subnets

Addressing: routing to another LAN

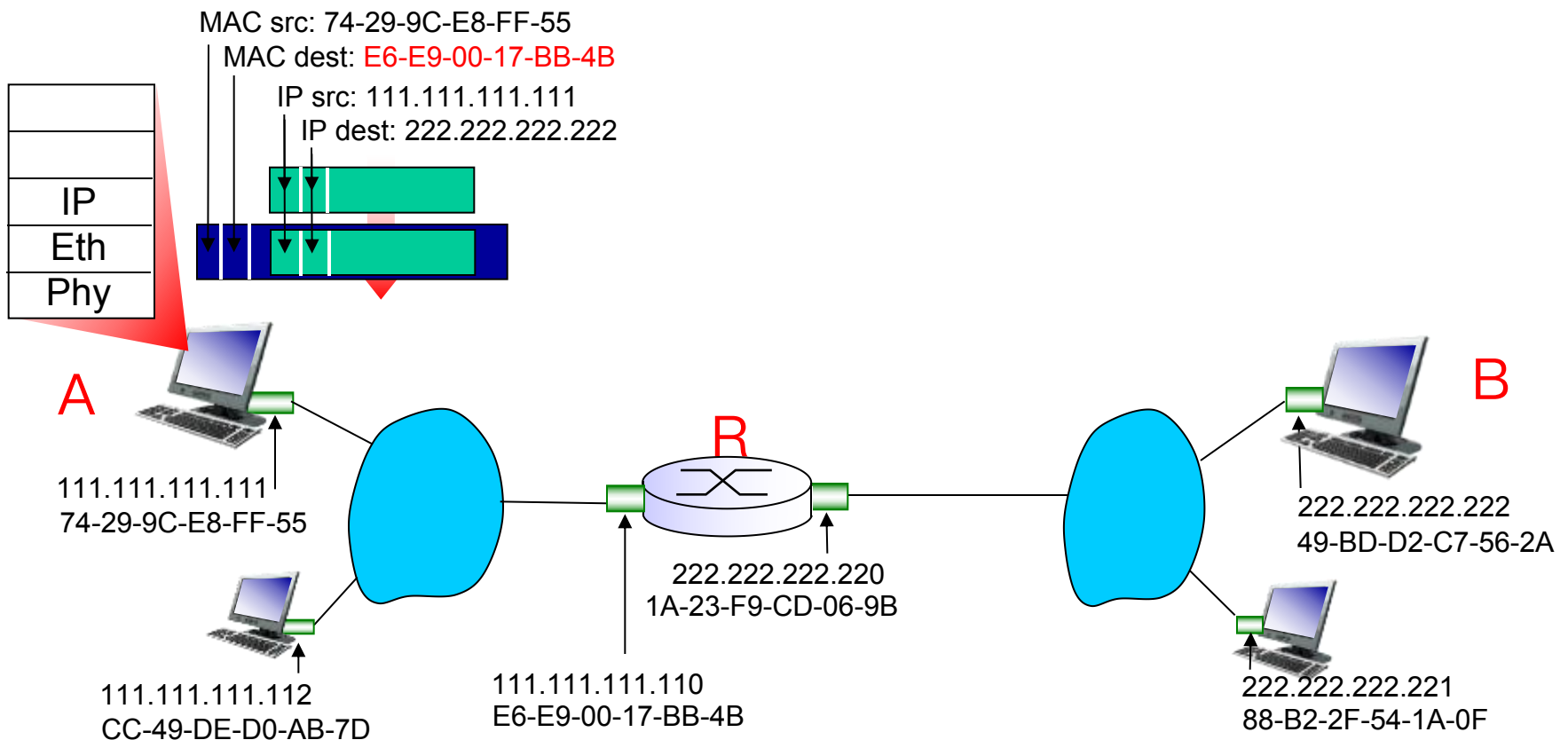
walkthrough: send datagram from A to B via R

- 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?)



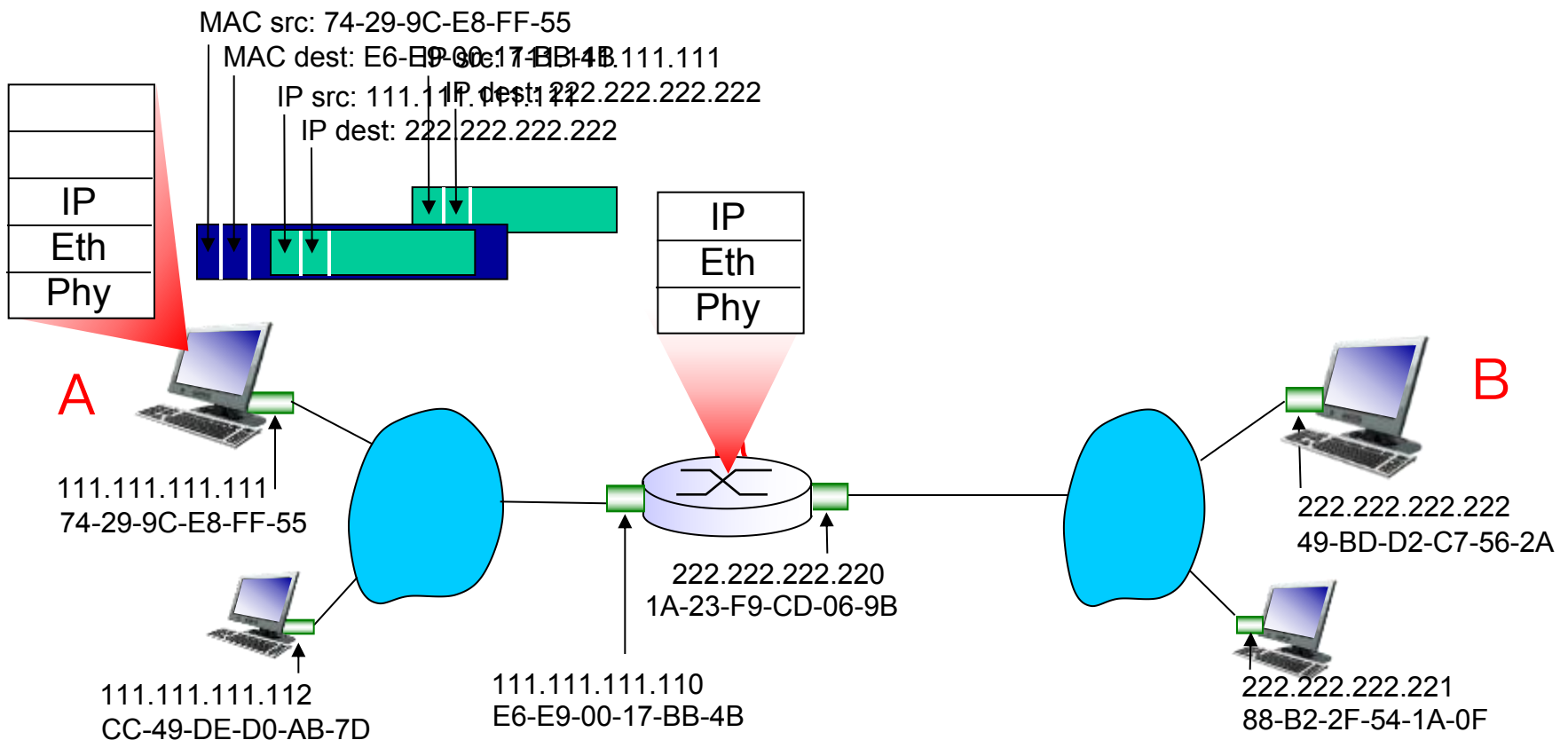
Addressing: routing to another LAN

- 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



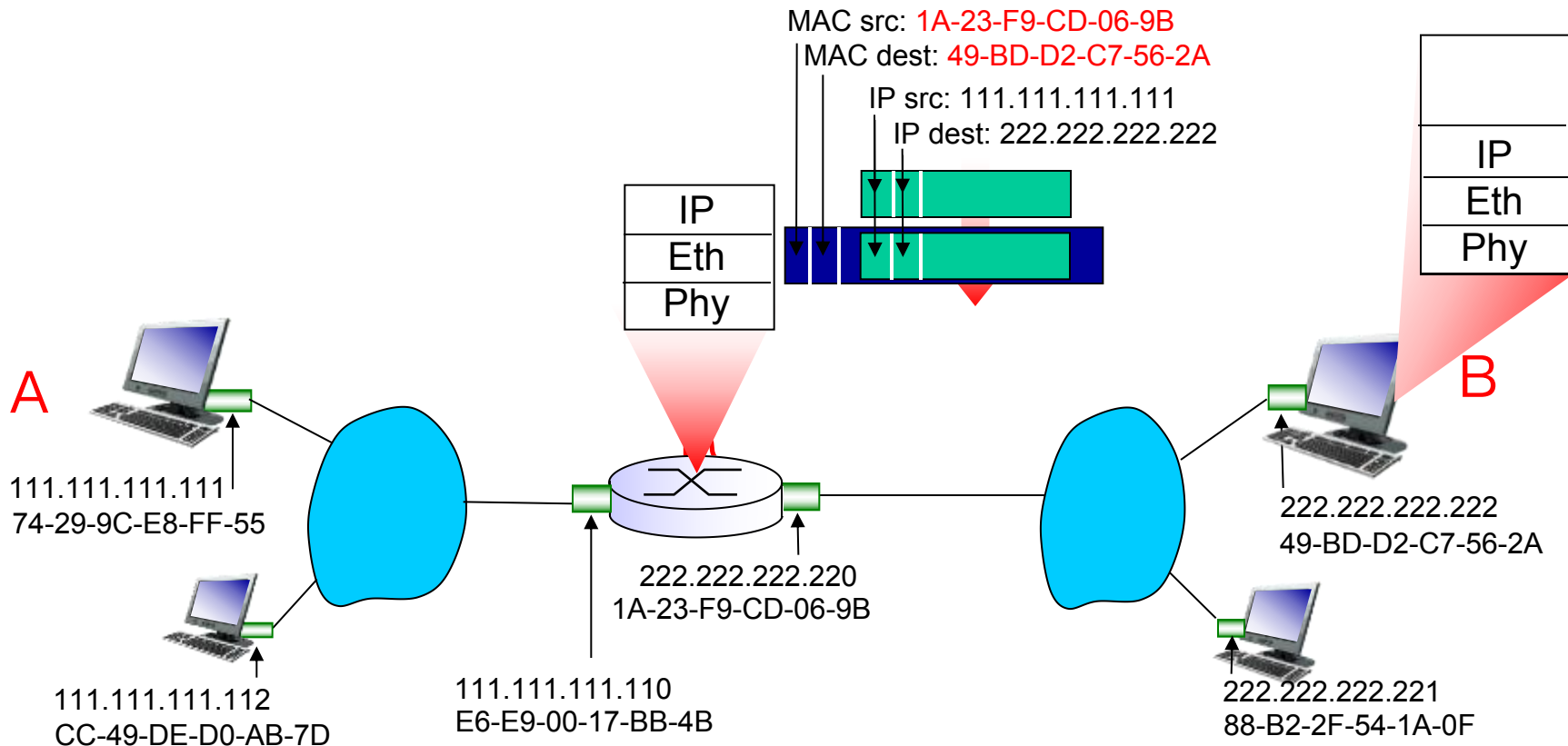
Addressing: routing to another LAN

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



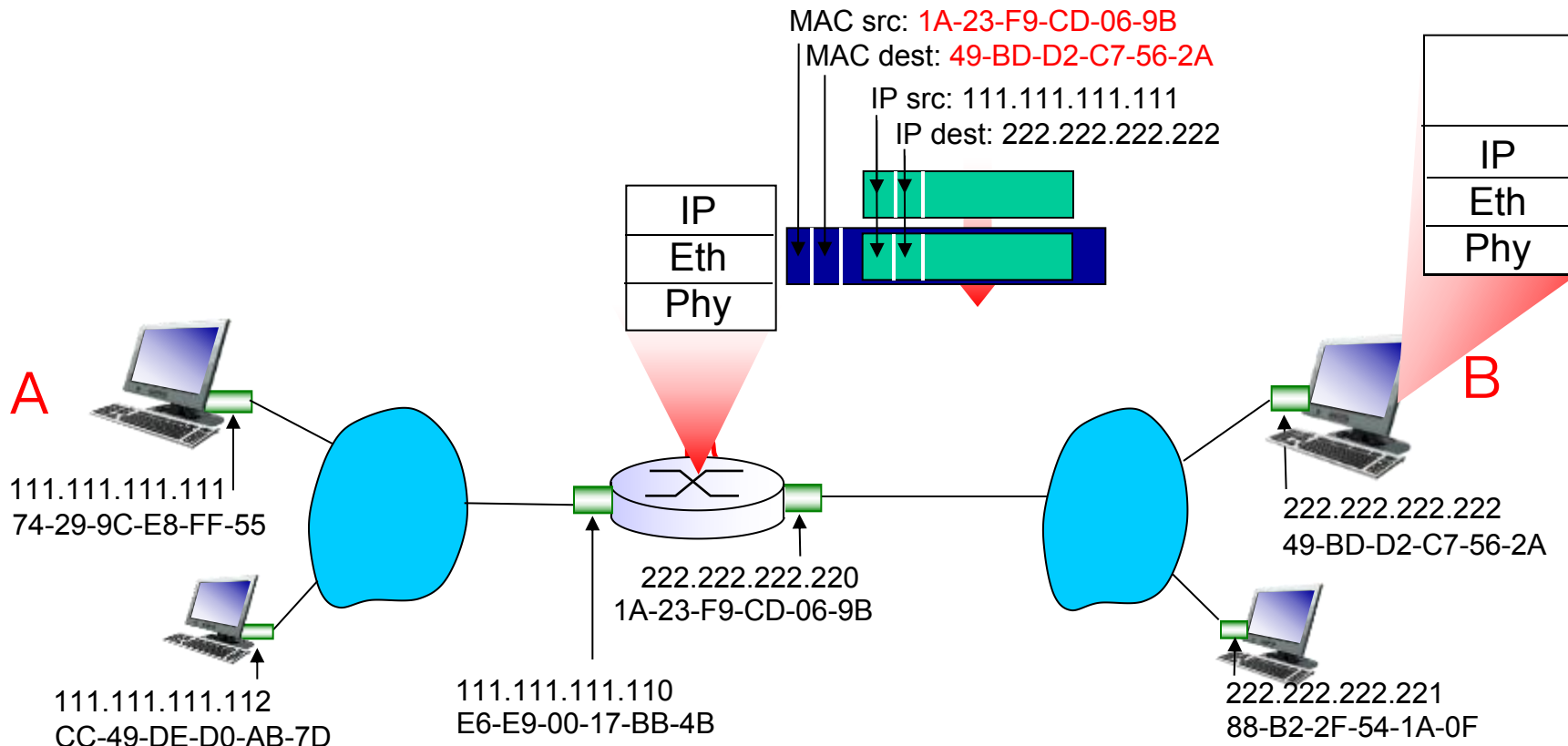
Addressing: routing to another LAN

- 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



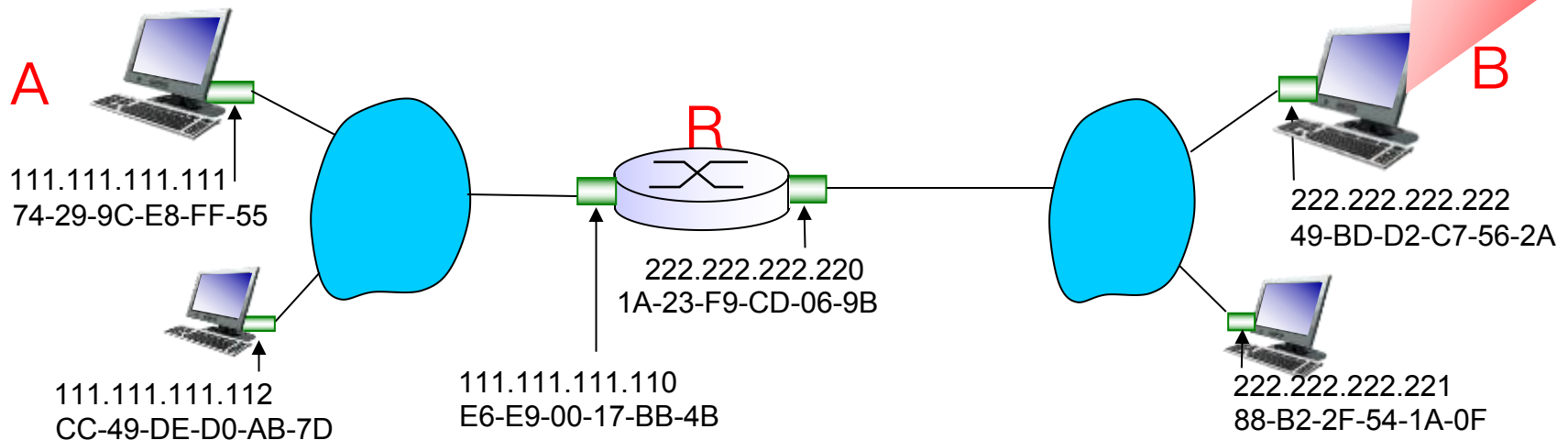
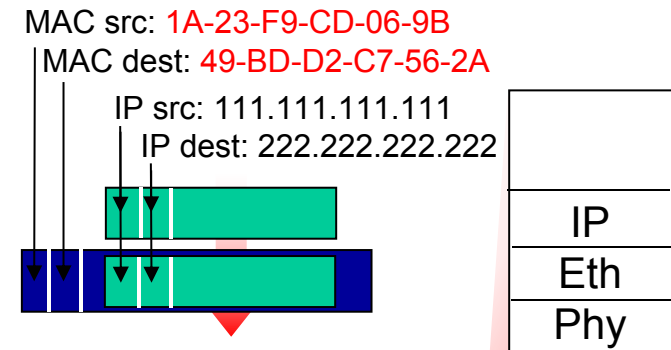
Addressing: routing to another LAN

- 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

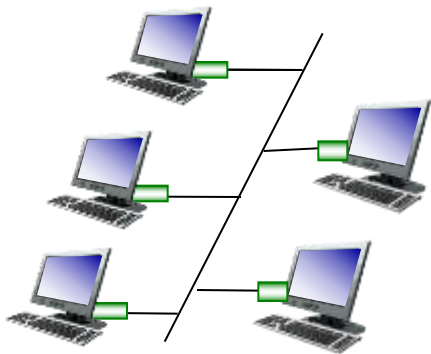
Ethernet

“dominant” wired LAN technology:

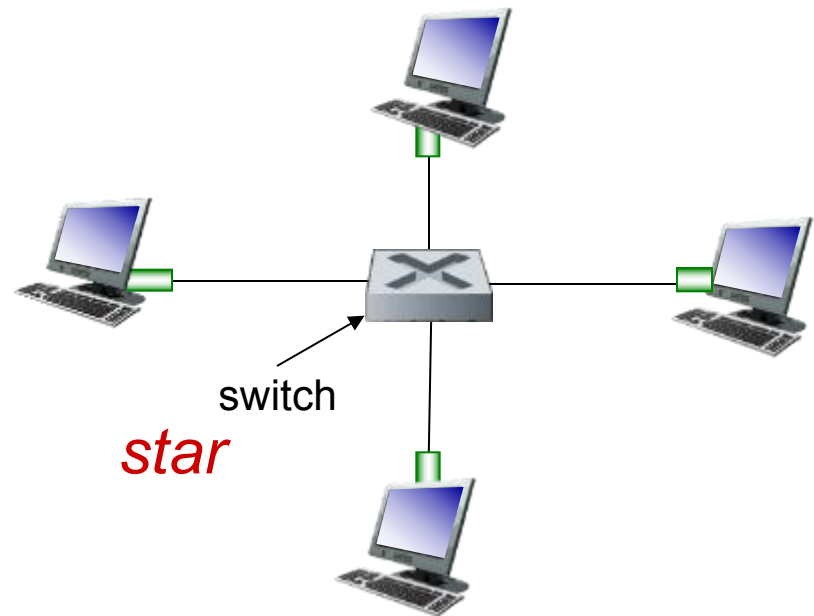
- first widely used LAN technology
- simpler, cheap
- kept up with speed race: 10 Mbps – 10 Gbps

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)



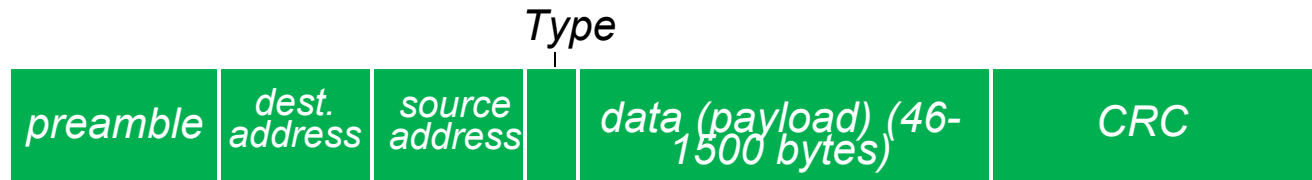
bus: coaxial cable



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
- Transmit frame at 10Mbps, 100Mbps, 1Gbps
- Drift from target rate
- 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'

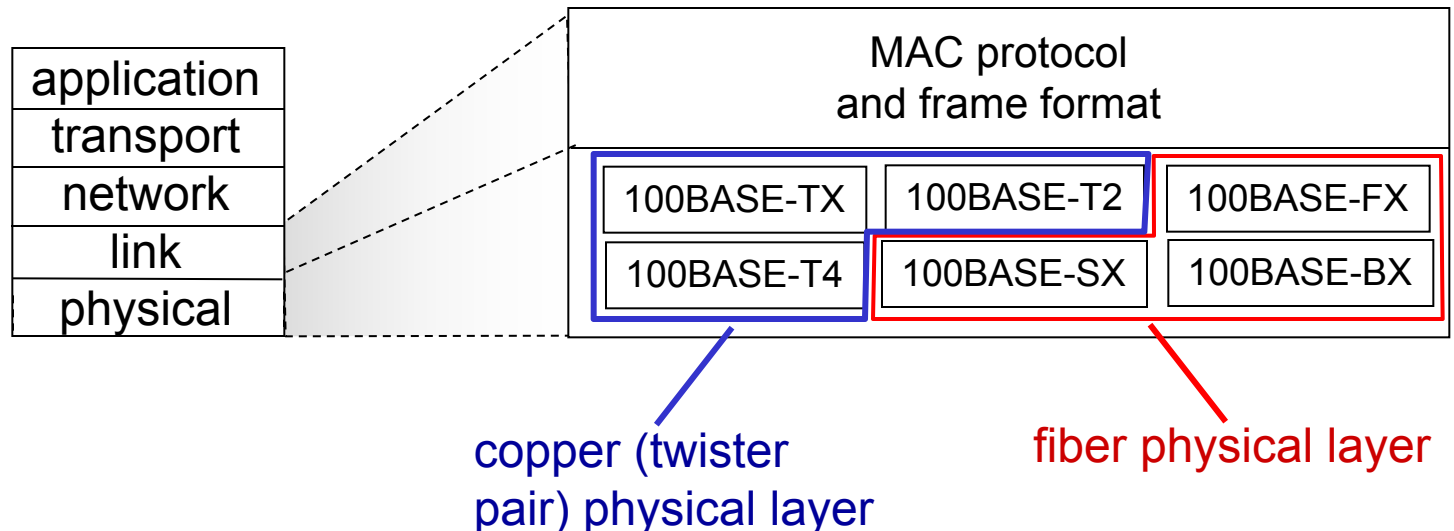


Ethernet: unreliable, ~~connectionless~~

- *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
 - Unaware of whether it is transmitting a brand-new datagram with brand-new data
- Ethernet's MAC protocol: unslotted *CSMA/CD with binary backoff*

802.3 Ethernet standards: link & physical layers

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



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

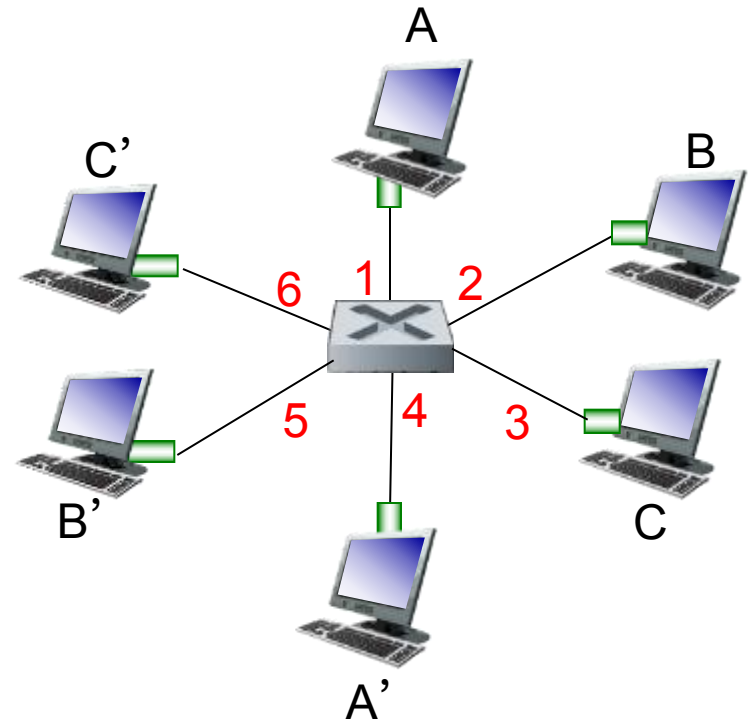
6.7 a day in the life of a web request

Ethernet switch

- **link-layer device**
 - 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
- ***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
- *switching*: A-to-A' and B-to-B' can transmit simultaneously, without collisions



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

Switch forwarding table

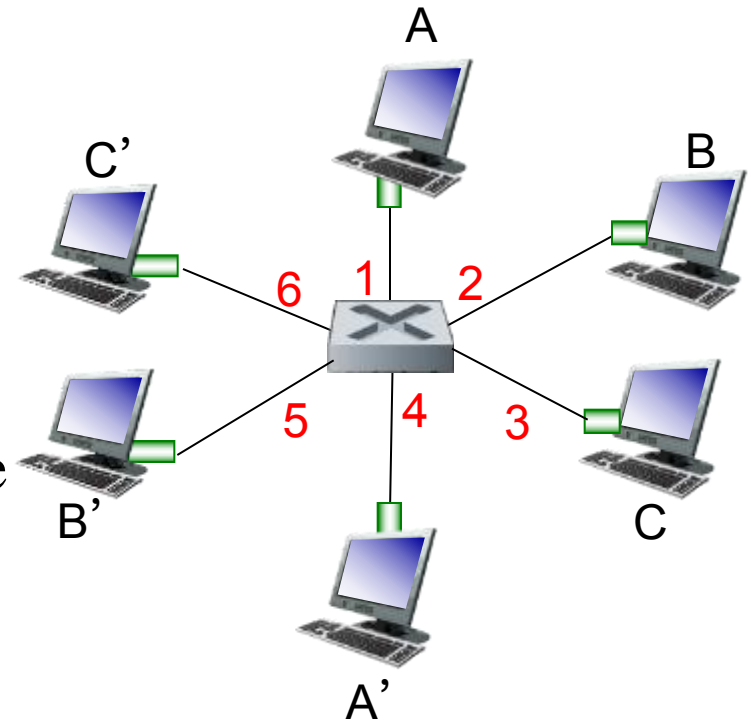
Q: how does switch know A' reachable via interface 4, B' reachable via interface 5?

A: each switch has a **switch table**, each entry:

- (MAC address of host, interface to reach host, time stamp)
- looks like a routing table!

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

- something like a routing protocol?

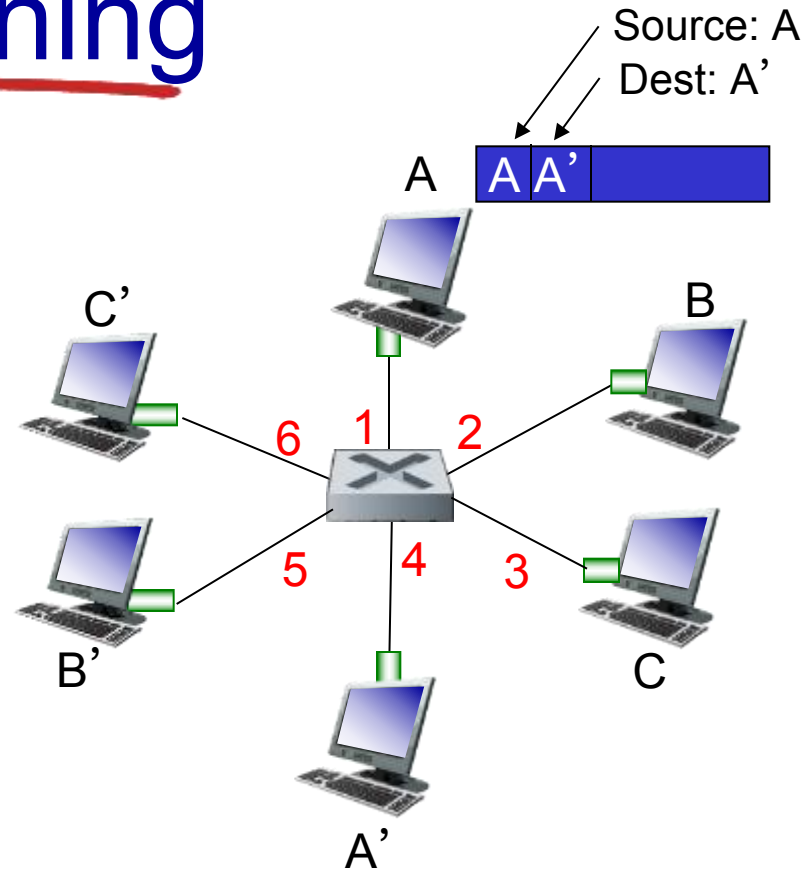


*switch with six interfaces
(1,2,3,4,5,6)*

Switch: self-learning

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

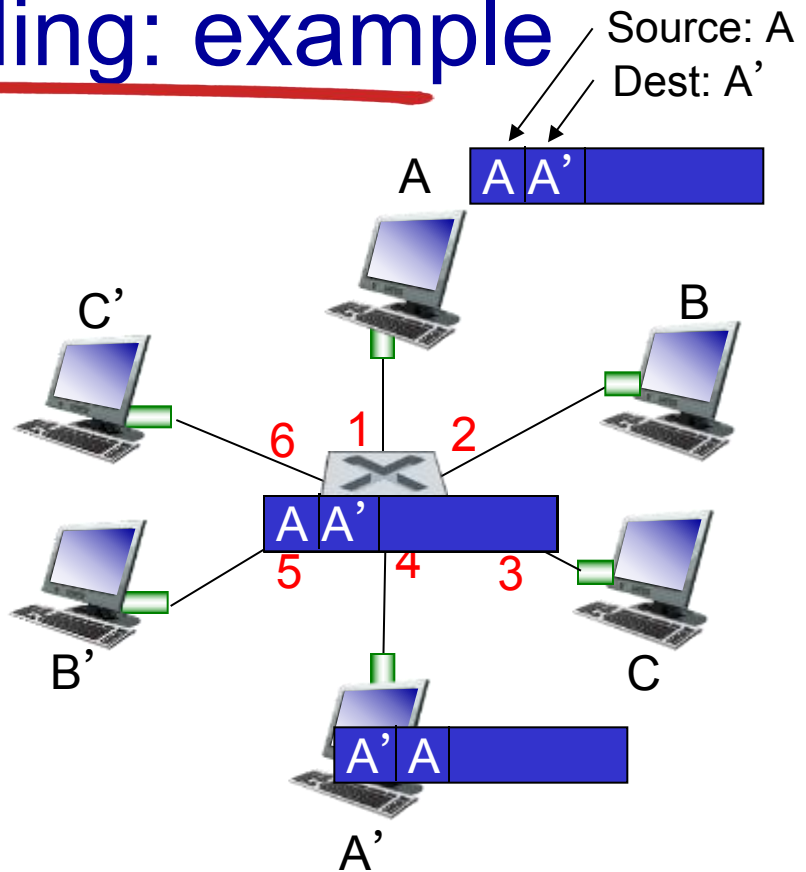


MAC addr	interface	TTL
A	1	60

*Switch table
(initially empty)*

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)*

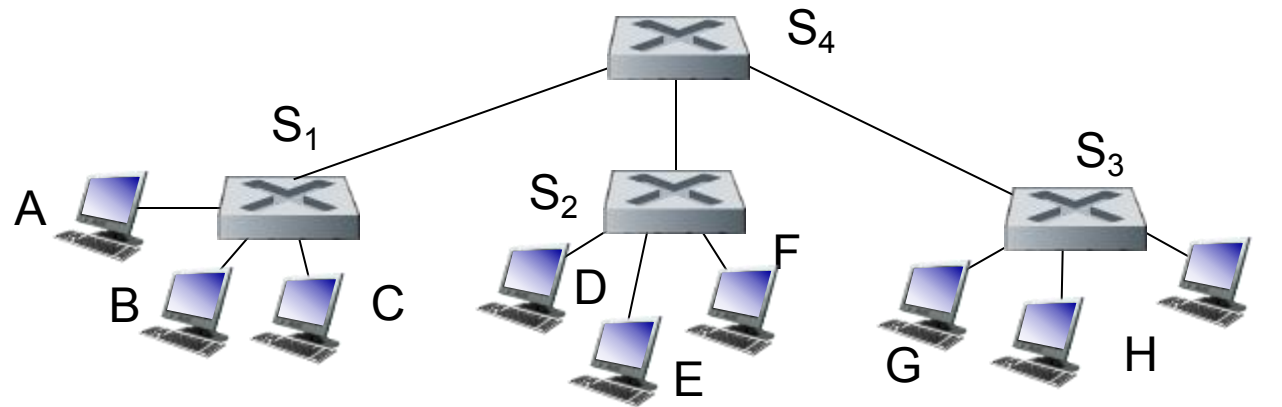
Self-learning, forwarding: example

Suppose a frame with destination address DDDD-DD-DD-DD-DD arrives at the switch **on interface x**.

- **no entry** in the table for DD-DD-DD-DD-DD-DD:
 - forwards copies of the frame to the output buffers preceding all interfaces except for interface x.
- an entry in the table associating DD-DD-DD-DD-DD-DD with **interface x**:
 - the switch performs the filtering function by discarding the frame.
- an entry in the table associating DD-DD-DD-DD-DD-DD with **interface y**:
 - frame needs to be forwarded to the LAN segment attached to interface y.

Interconnecting switches

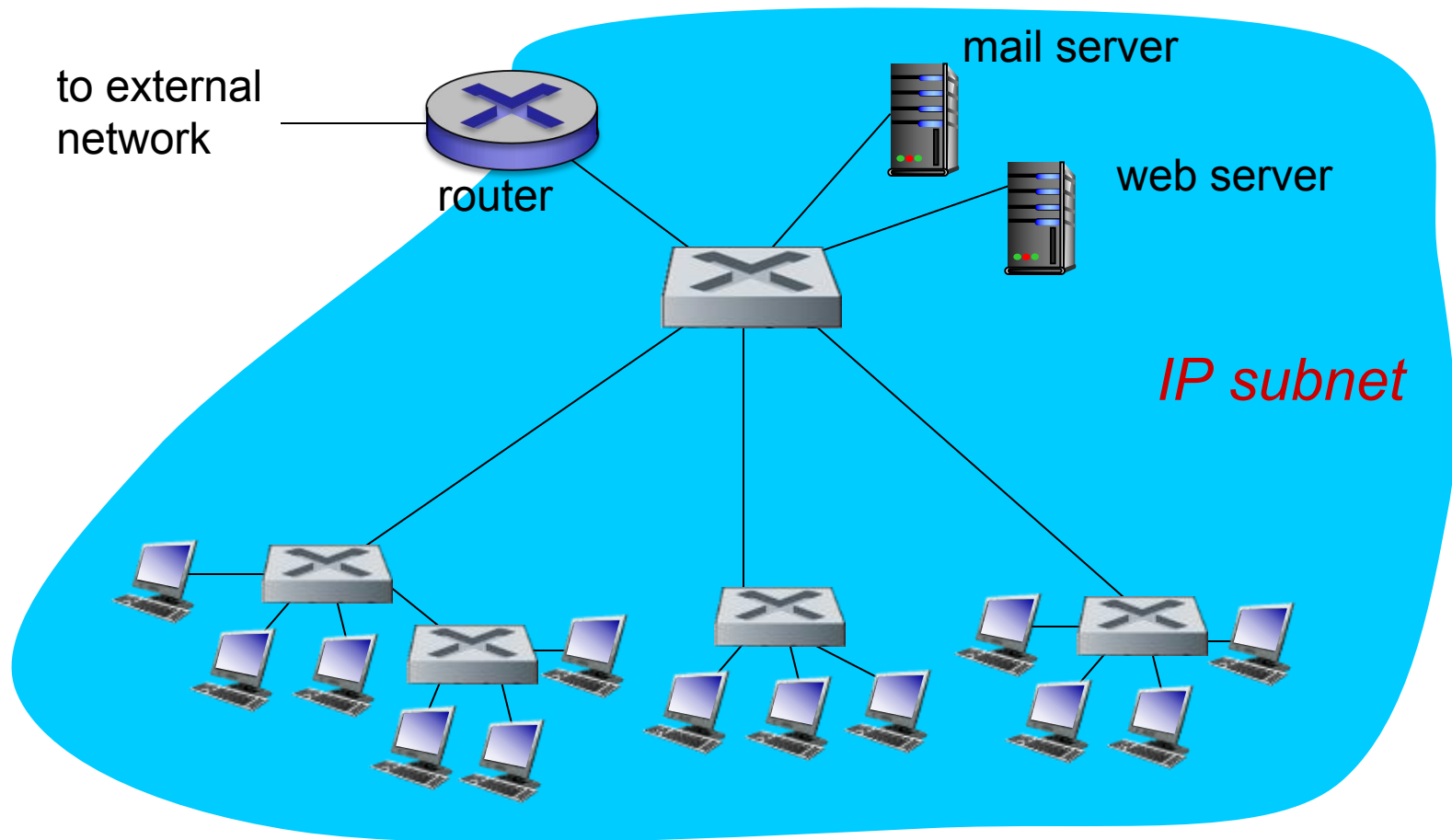
self-learning switches can be connected together:



Q: sending from A to G - how does S₁ know to forward frame destined to G via S₄ and S₃?

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

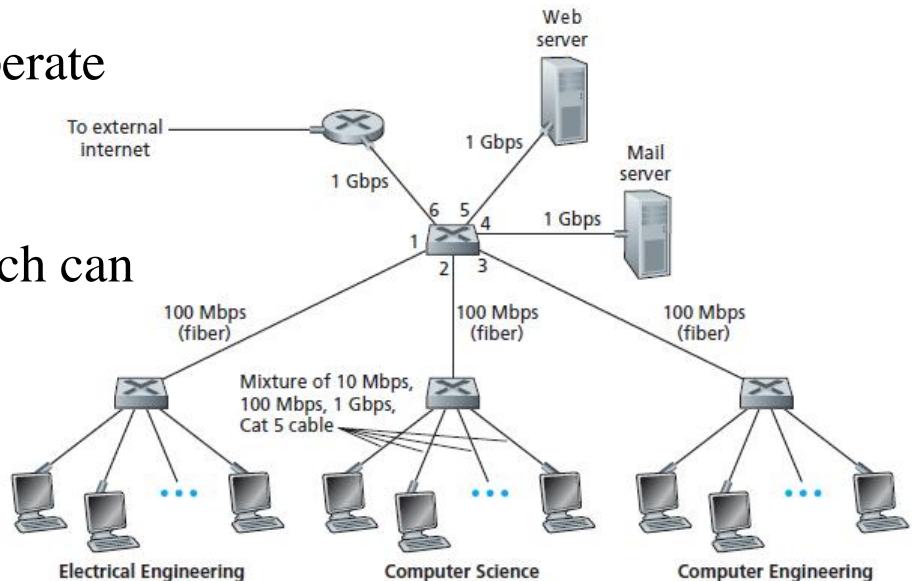
Institutional network



Properties of link-layer Switching

When compared with bus or hub:

- **Elimination of collisions**
 - Buffer frames; never transmit more than one frame on a segment at a time
- **Heterogeneous links**
 - Switch can isolate one link from another
 - Different links in the LAN can operate at different speeds and media
- **Management**
 - If one NIC malfunctions, the switch can detect it and disconnect it.



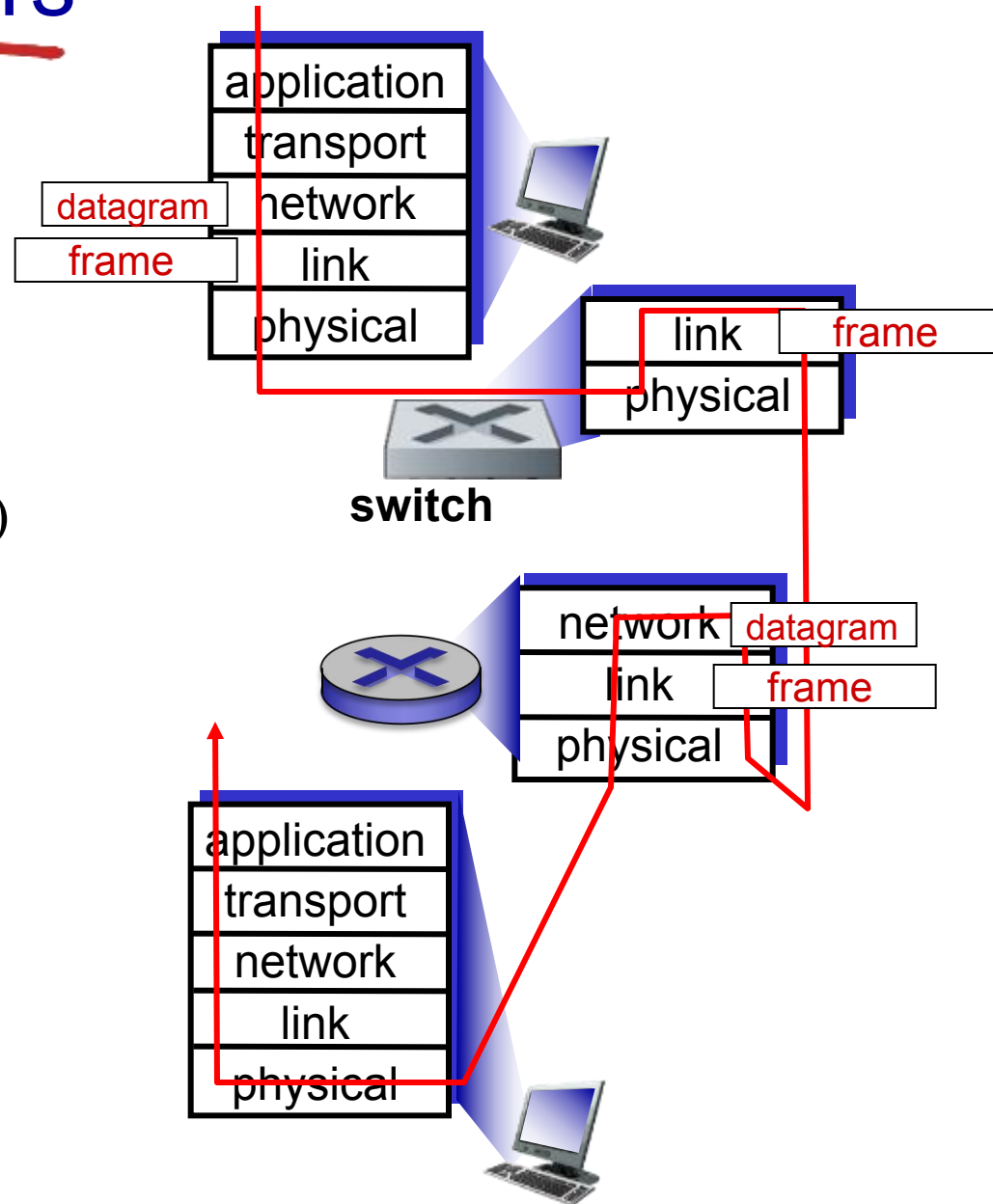
Switches vs. routers

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



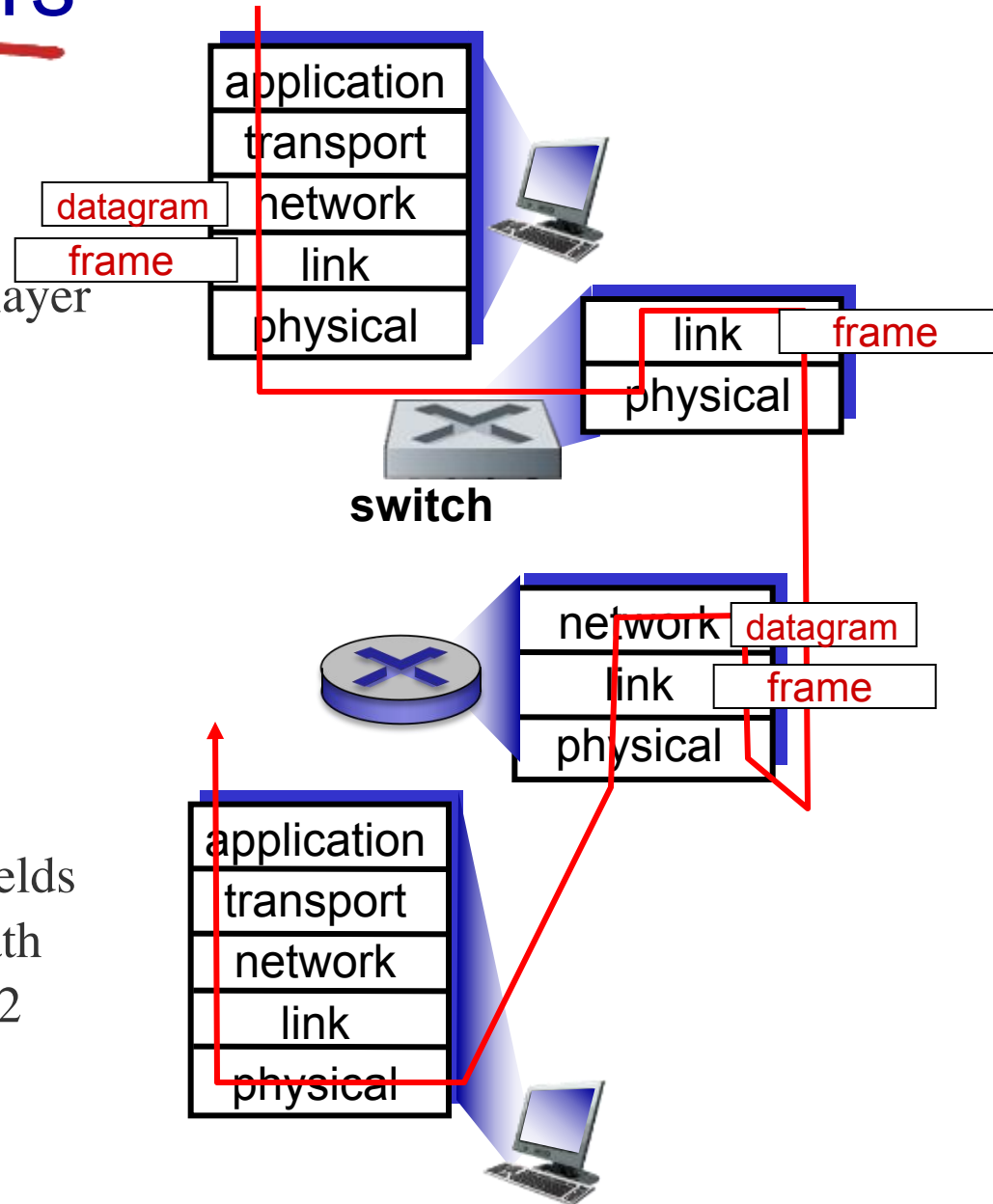
Switches vs. routers

Switches:

- plug-and-play
- Process frames only up through layer 2, relatively high filtering and forwarding rates
- a spanning tree
- a large switched network would require large ARP
- broadcast storms

Routers

- not plug-and-play
- process up through the layer-3 fields
- rich topology; choose the best path
- firewall protection against layer-2 broadcast storms.



Link layer, LANs: outline

6.1 introduction, services

6.2 error detection, correction

6.3 multiple access protocols

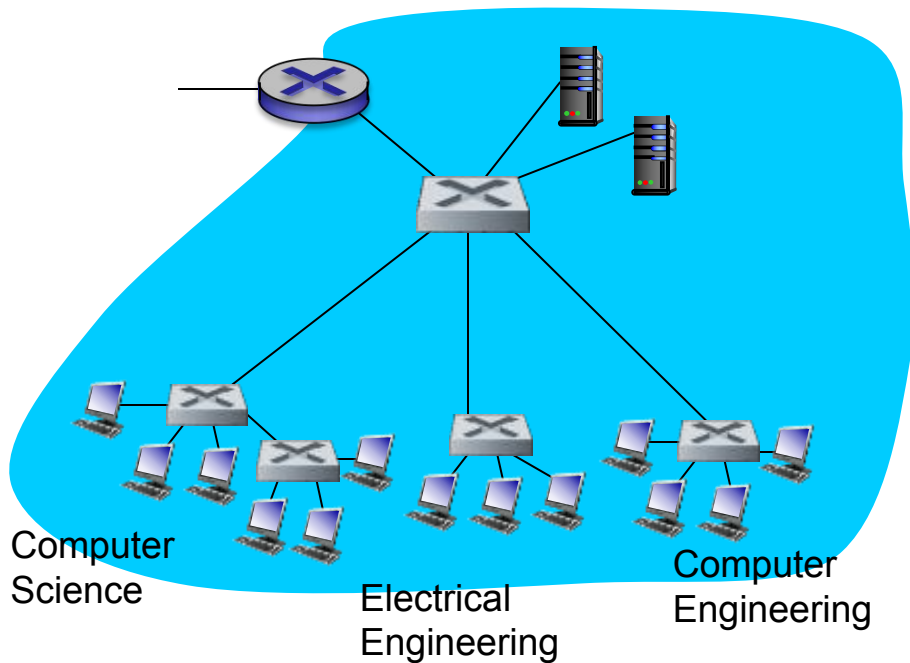
6.4 LANs

- addressing, ARP
- Ethernet
- switches
- VLANs

6.6 data center networking

6.7 a day in the life of a web request

VLANs: motivation



consider:

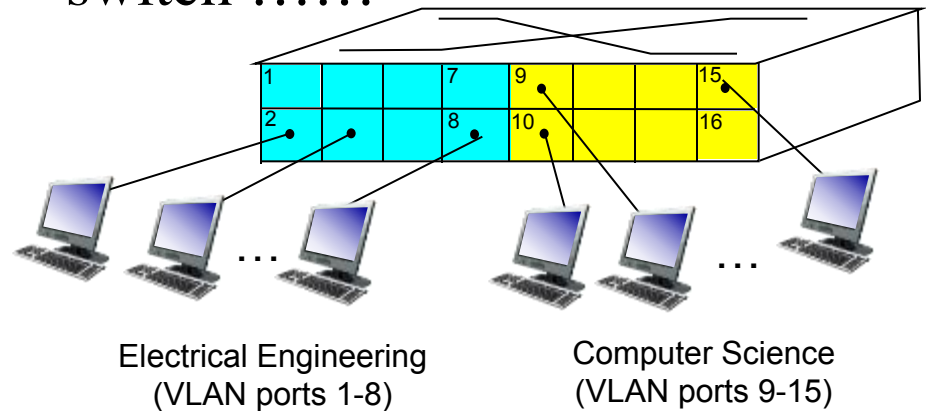
- CS user moves office to EE, but wants to connect to CS switch
- Inefficient use of switches
- single broadcast domain:
 - all layer-2 broadcast traffic (ARP, DHCP, unknown location of destination MAC address) must cross entire LAN
 - security/privacy, efficiency issues

VLANs

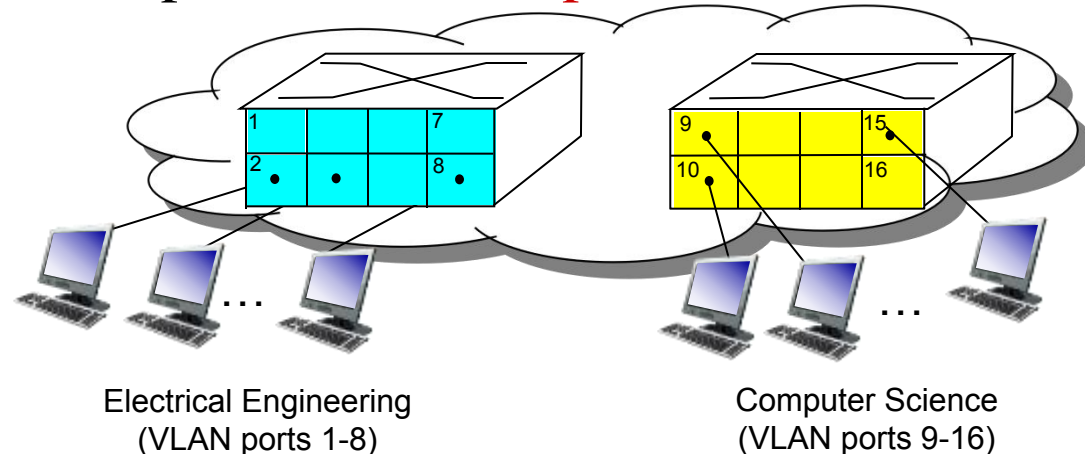
Virtual Local Area Network

define multiple *virtual* LANS over single physical LAN infrastructure.

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

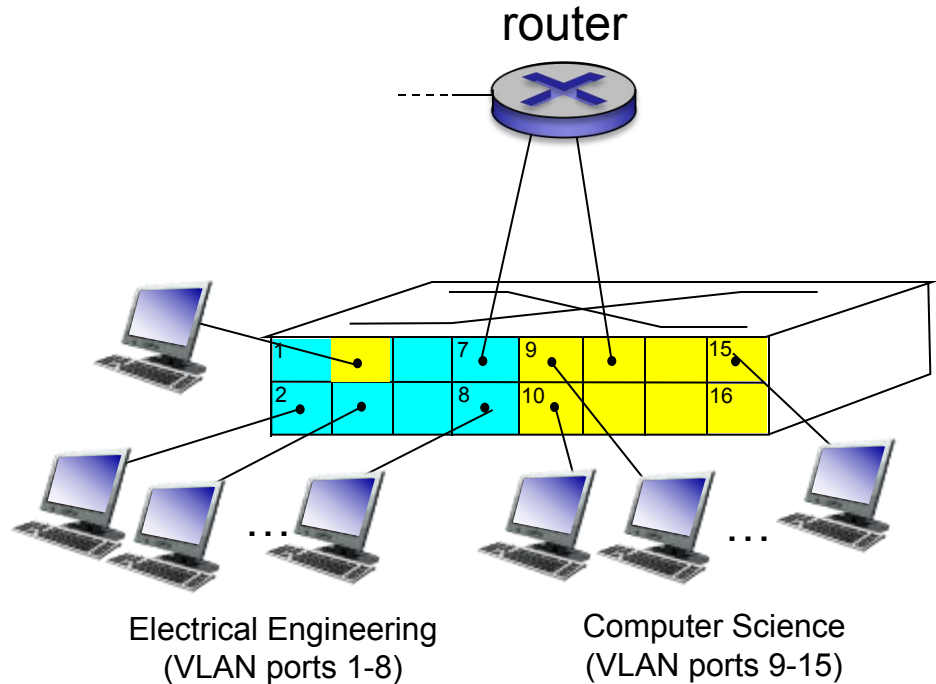


... operates as **multiple** virtual switches

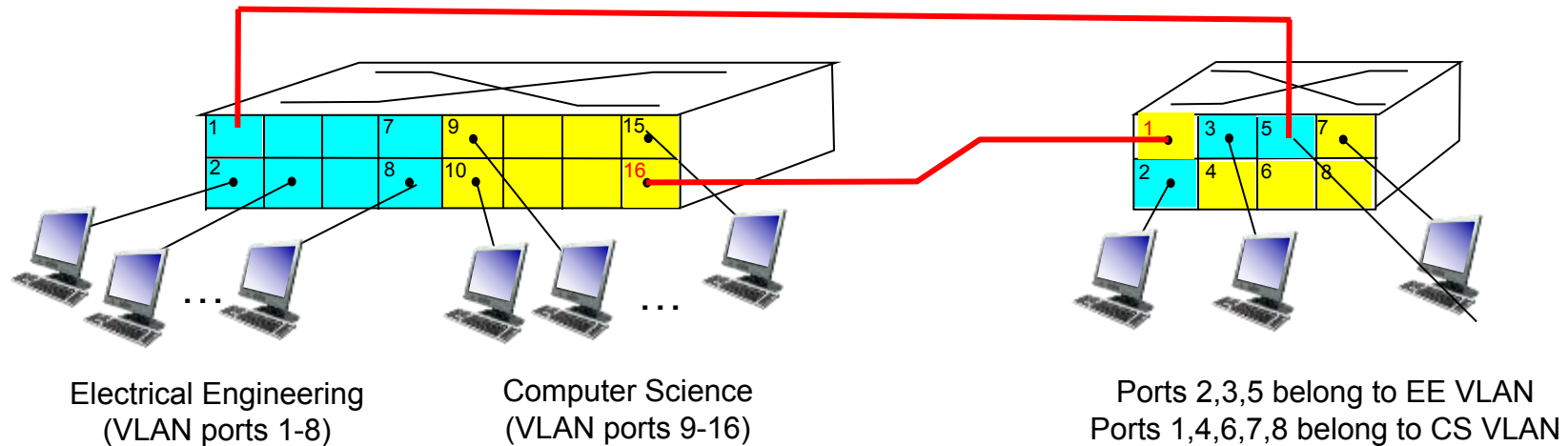


Port-based VLAN

- **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

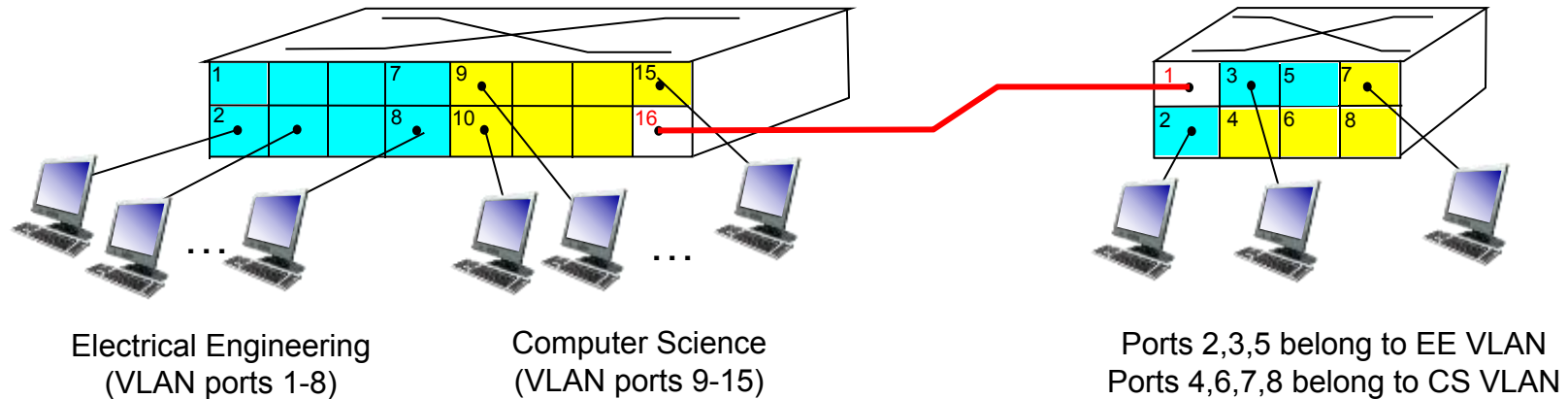


VLANs spanning multiple switches



- *If some of the CS and EE faculties are in another building, how to connect two switches together as two VLANs?*
 - *Two links connect both CS VLAN and EE VLAN.*

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)
 - Extended Ethernet frame format: 802.1q protocol adds/removed additional header fields for frames forwarded between trunk ports