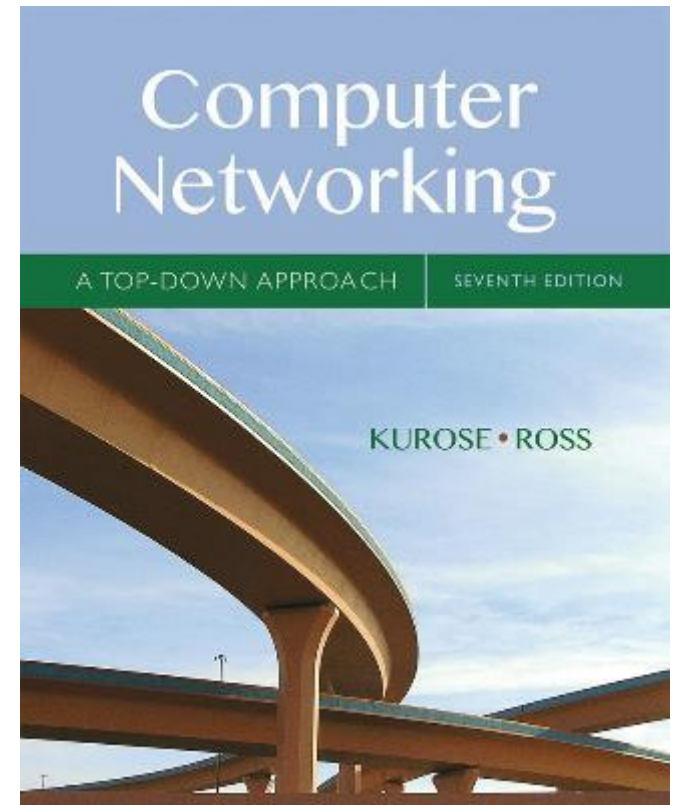# Chapter 5
Network Layer:
The Control Plane

Slides adopted from original ones provided by the textbook authors.

Computer Networking: A Top Down Approach

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

Chapter 5: outline

Network-layer routing function

Two approaches for network control plane:

- per-router control (traditional)
- logically centralized control (software defined networking)

Chapter 5: outline

Graph abstraction of the network
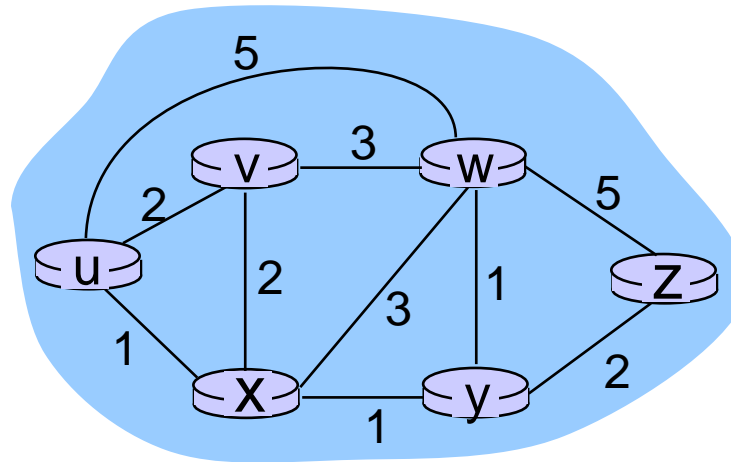
graph: G = (N,E)

N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

c(u, v): cost of link (u, v)

Goal: find minimum cost path

Routing algorithms

- ## Dijkstra's algorithm
  - Each router collects entire network topology via link advertisements,
  - Each router makes its own decisions.

- ## Distance vector algorithm
  - Routers exchange distance vectors.
  - Routers make distributed decisions.

Dijsktra's algorithm

```
1  Initialization:
2    N' = {u}
3    for all nodes v
4      if v adjacent to u
5         then D(v) = c(u,v)
6      else D(v) = ∞
7
8  Loop
9    find w not in N' such that D(w) is a minimum
10    add w to N'
11    update D(v) for all v adjacent to w and not in N' :
12       D(v) = min( D(v), D(w) + c(w,v) )
13    /* new cost to v is either old cost to v or known
14      shortest path cost to w plus cost from w to v */
15  until all nodes in N'
```

Distance vector algorithm

Bellman-Ford equation

$$d_x(y) = \min_v \{c(x,v) + d_v(y) \}$$

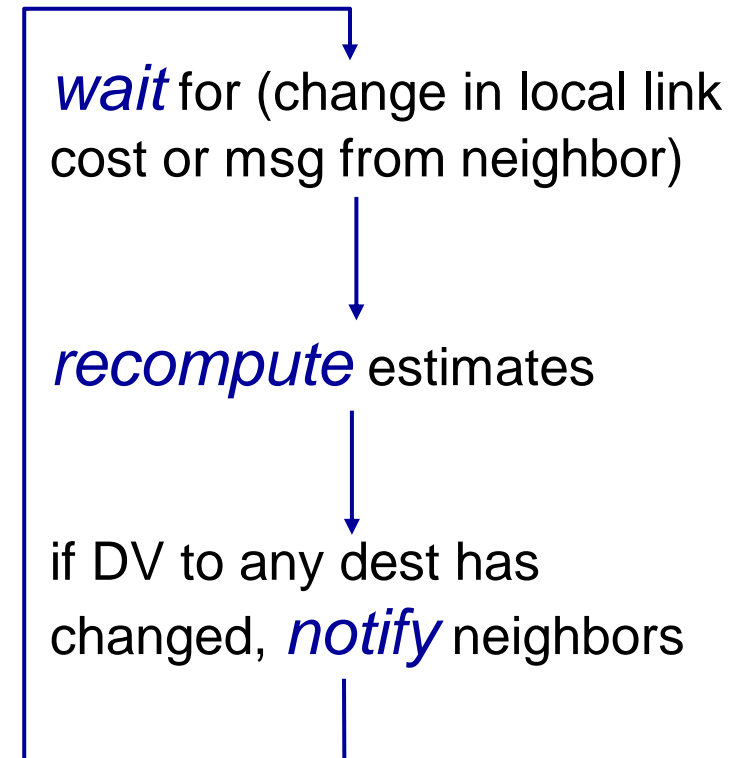*iterative, asynchronous:* each local iteration caused by:

- local link cost change
- DV update message from neighbor

distributed:

- each node notifies neighbors *only* when its DV changes
  - neighbors then notify their neighbors if necessary

each node:

wait for (change in local link cost or msg from neighbor)

↓

recompute estimates

↓

if DV to any dest has changed, *notify* neighbors

Chapter 5: outline

5.1 introduction

5.2 routing protocols
- link state
- distance vector

5.3 intra-AS routing in the Internet: OSPF

5.4 routing among the ISPs: BGP

5.5 The SDN control plane

5.6 ICMP: The Internet Control Message Protocol

5.7 Network management and SNMP

Hierarchical routing

aggregate routers into regions known as "autonomous systems" (AS) (a.k.a. "domains")

intra-AS routing

- routing among hosts, routers in same AS ("network")
- all routers in AS must run *same* intra-domain protocol
- routers in *different* AS can run *different* intra-domain routing protocol
- gateway router: at "edge" of its own AS, has link(s) to router(s) in other AS'es

inter-AS routing

- routing among AS'es
- gateways perform inter-domain routing (as well as intra-domain routing)

OSPF (Open Shortest Path First)

- open protocol, based on Link State algorithm
- router floods OSPF link-state advertisements to all other routers in *entire* AS
- *security:* all OSPF messages authenticated
- ECMP: equal-cost multiple paths
- *two-level hierarchy:* local area, backbone

Chapter 5: outline

BGP (Border Gateway Protocol)

- two types of BGP (TCP) sessions
  - eBGP: between neighboring ASs
  - iBGP: inside same AS

- router may learn about more than 1 route to destination AS, selects route based on:
  1. local preference value attribute: policy decision
  2. shortest AS-PATH
  3. closest NEXT-HOP router: hot potato routing
  4. additional criteria
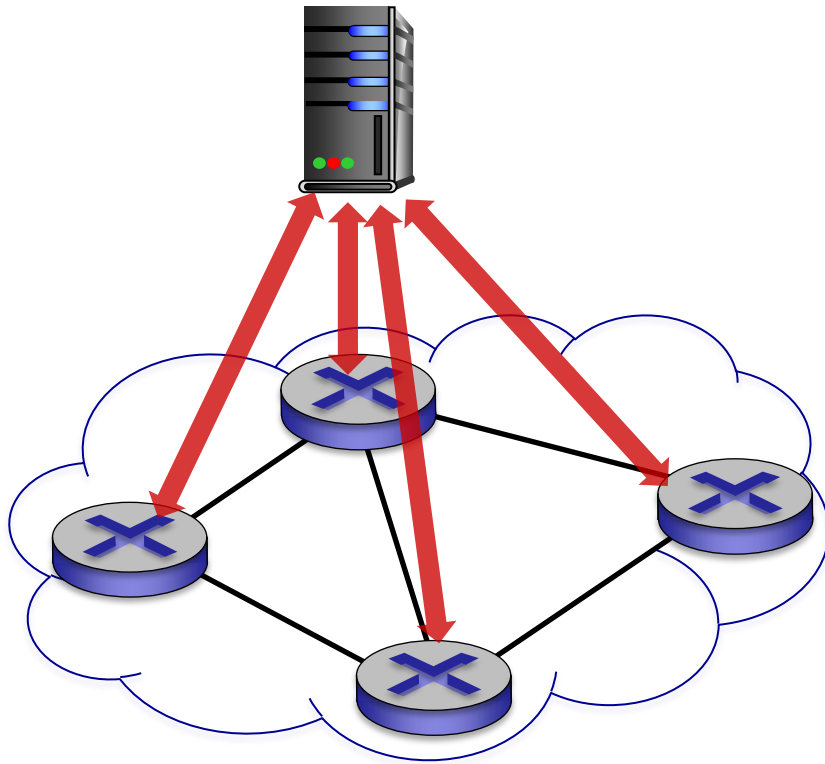
Chapter 5: outline

Software defined networking (SDN)

SDN: logically centralized control plane

- easier network management
- ability to program routers
- open implementation of control plane

OpenFlow protocol

- operates between controller, switch
- TCP used to exchange messages

OpenFlow Controller

Chapter 5: outline

ICMP: internet control message protocol

- **used by hosts & routers to communicate network-level information**
  - error reporting: unreachable host, network, port, protocol
  - echo request/reply (used by ping)
- **network-layer "above" IP:**
  - ICMP msgs carried in IP datagrams
- **ICMP message:** type, code plus first 8 bytes of IP datagram causing error

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Traceroute and ICMP

- source sends series of UDP segments to destination
  - first set has TTL =1
  - second set has TTL=2, etc.
  - unlikely port number
- when datagram in *n*th set arrives to nth router:
  - router discards datagram and sends source ICMP message (type 11, code 0)
  - ICMP message include name of router & IP address

- when ICMP message arrives, source records RTTs

stopping criteria:
- UDP segment eventually arrives at destination host
- destination returns ICMP "port unreachable" message (type 3, code 3)
- source stops

3 probes   3 probes

3 probes

Chapter 5: outline

What is network management?

- manage hardware, software, and human elements
- to meet performance requirements

SNMP protocol

Two ways to convey MIB info, commands:

request/response mode

trap mode

Chapter 6
The Link Layer and LANs

Computer Networking: A Top Down Approach

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

Slides adopted from original ones provided by the textbook authors.

Chapter 6: Link layer

our goals:

❖ understand principles behind link layer services:

- error detection, correction
- sharing a broadcast channel: multiple access
- link layer addressing
- local area networks: Ethernet, VLANs

❖ instantiation, implementation of various link layer technologies

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

global ISP
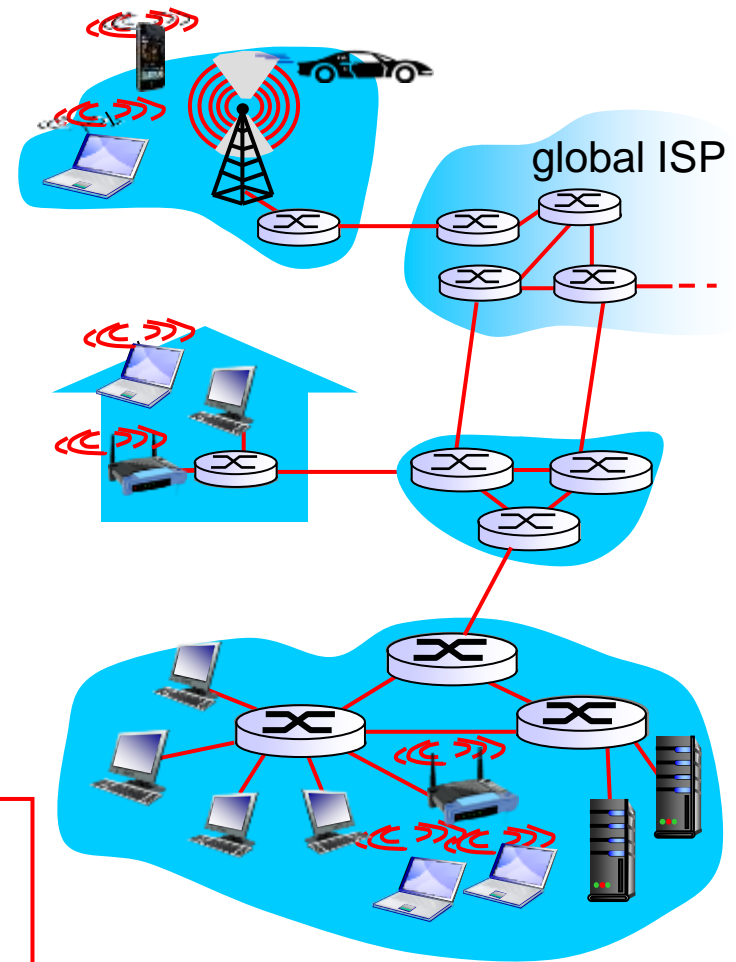
Link layer: context

❖ 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:*
  ▪ encapsulate datagram into frame, adding header, trailer

❖ *link access:*
  ▪ channel access if shared medium

❖ *error detection/correction:*
  ▪ errors caused by signal attenuation, noise
  ▪ receiver identifies *and corrects* bit error(s) without resorting to retransmission

Where is the link layer implemented?

❖ 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

| application |
| transport |
| network |
| link |

cpu

memory

host bus (e.g., PCI)

controller

| link |
| physical |

physical

transmission

network adapter card
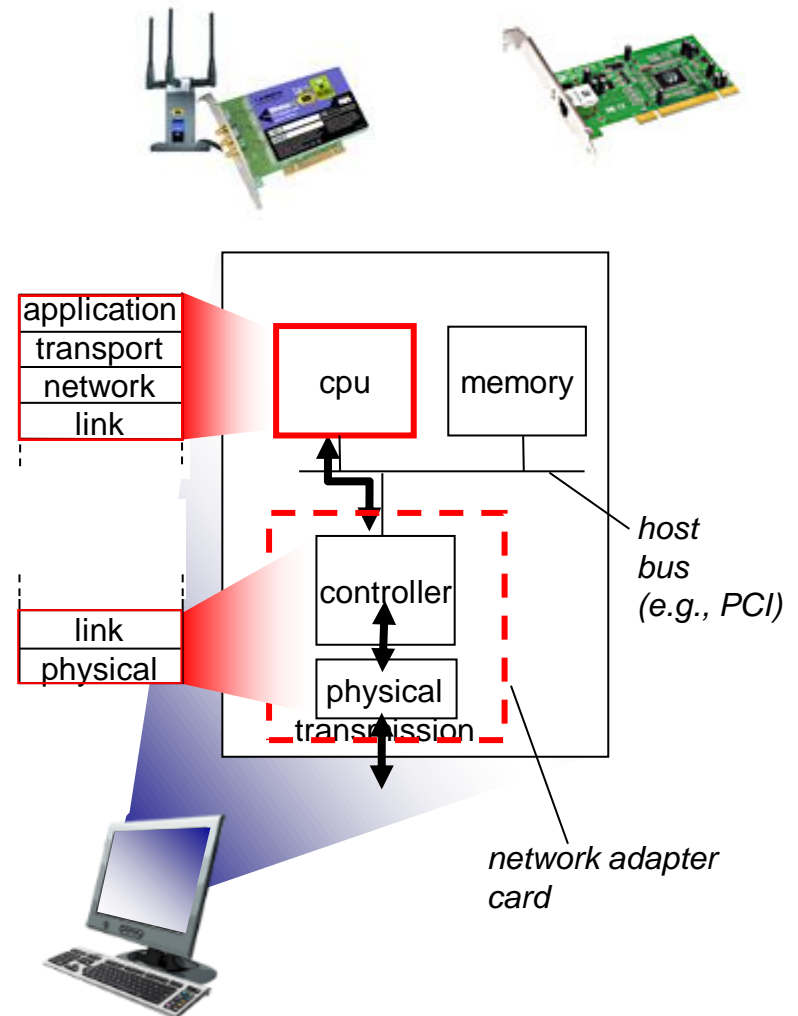
Adaptors communicating

sending host   *receiving host*
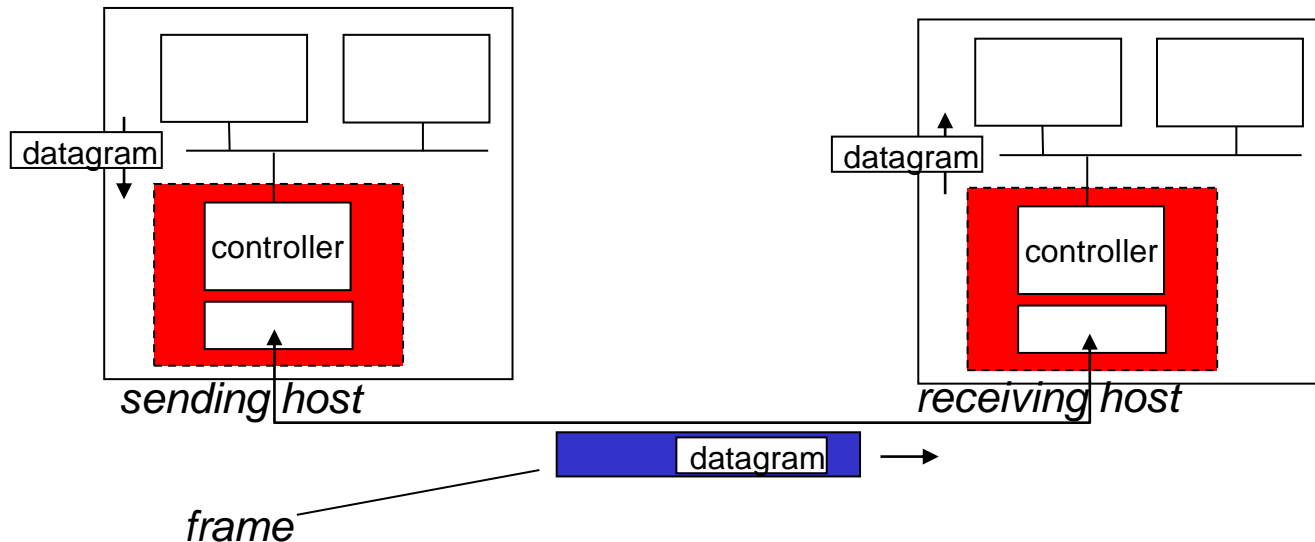
frame

❖ 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

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

Error detection

EDC= Error Detection and Correction bits (redundancy)
D   = Data protected by error checking, may include header fields

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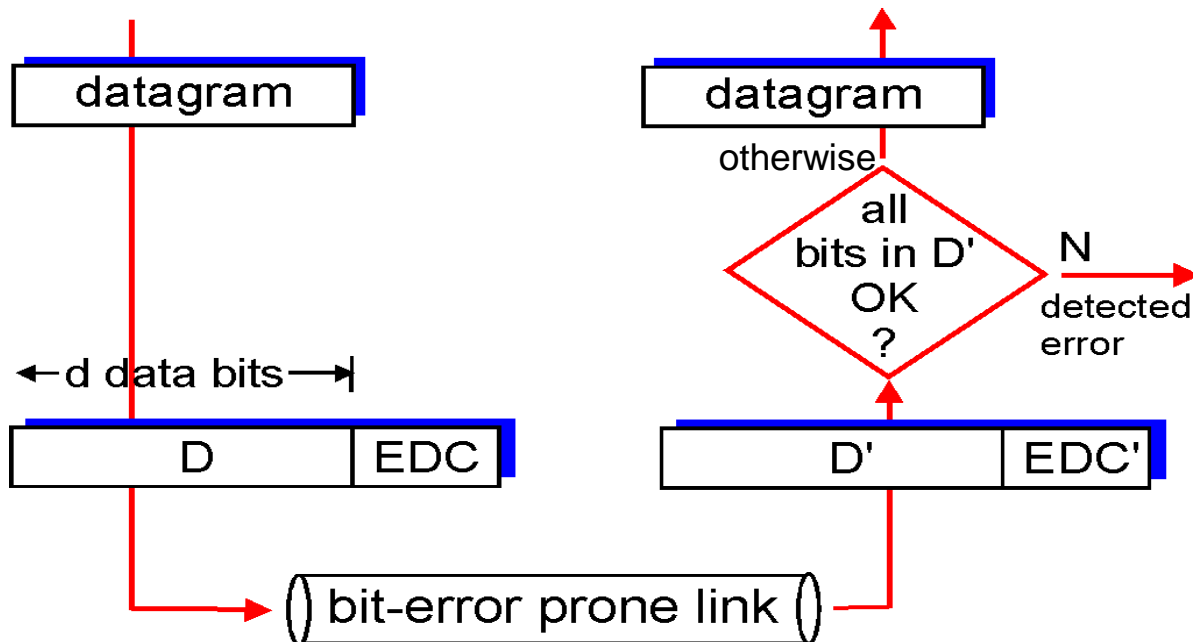
Parity checking

single bit parity:
- ❖ *d*etect single bit errors

$$\longleftarrow d \text{ data bits} \longrightarrow | \text{parity bit}$$

| 011100011010101011 | 0 |

Even parity: parity bit chosen for even # of 1s
Odd parity: parity bit chose for odd # of 1s
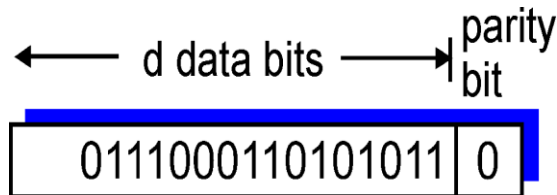
two-dimensional bit parity:
- ❖ detect and correct single bit errors

row parity →

$$
\begin{array}{cccc}
d_{1,1} & \cdots & d_{1,j} & d_{1,j+1} \\
d_{2,1} & \cdots & d_{2,j} & d_{2,j+1} \\
\cdots & \cdots & \cdots & \cdots \\
d_{i,1} & \cdots & d_{i,j} & d_{i,j+1} \\
\hline
d_{i+1,1} & \cdots & d_{i+1,j} & d_{i+1,j+1}
\end{array}
$$

column parity ↓

```
1 0 1 0 1|1        1 0 1 0 1|1
1 1 1 1 0|0        1 0 1 1 0 0 → parity error
0 1 1 1 0|1        0 1 1 1 0|1
0 0 1 0 1|0        0 0 1 0 1|0
```
no errors   parity error

correctable single bit error

Example

Suppose a packet has the bit pattern 1010 1100 1000 1011, and an even parity scheme is used.

a. What would the parity bits for a 4x4 two-dimensional parity scheme?

b. Show a double-bit error that can be detected but not corrected.

Cyclic redundancy check

❖ more powerful error-detection coding

❖ view data bits, D, as a binary number

❖ choose r+1 bit pattern (generator), G

❖ goal: choose r CRC bits, R, such that
  - ▪ <D,R> exactly divisible by G (modulo 2)
  - ▪ receiver knows G, divides <D,R> by G. If non-zero remainder: error detected!
  - ▪ can detect all burst errors less than r+1 bits

❖ widely used in practice (Ethernet, 802.11 WiFi, Bluetooth)

```
←——————— d bits ———————→ ←— r bits —→
┌─────────────────────────┬─────────────┐
│ D: data bits to be sent │ R: CRC bits │   bit
└─────────────────────────┴─────────────┘   pattern
```
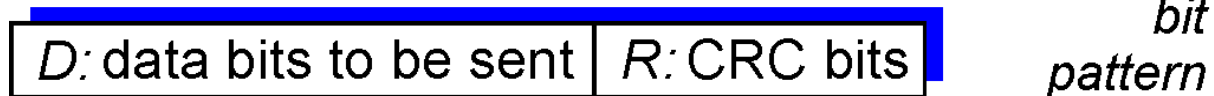
$$D * 2^r \quad XOR \quad R$$ *mathematical formula*

CRC example

want:

   $D \cdot 2^r$ XOR R = nG

equivalently:

   $D \cdot 2^r$ = nG XOR R

equivalently:

   if we divide $D \cdot 2^r$ by G, want remainder R to satisfy:

$$R = remainder[\frac{D \cdot 2^r}{G}]$$

```
                        101011
              ┌─────────────────
         1001 ) 101110000
         ─────  1001
      G ←──┘    ────
               101
               000
               ────
               1010
               1001
               ────
                110
                000
                ────
                1100
                1001
                ────
                 1010
                 1001
                 ────
                  011
             R ←──┘
```
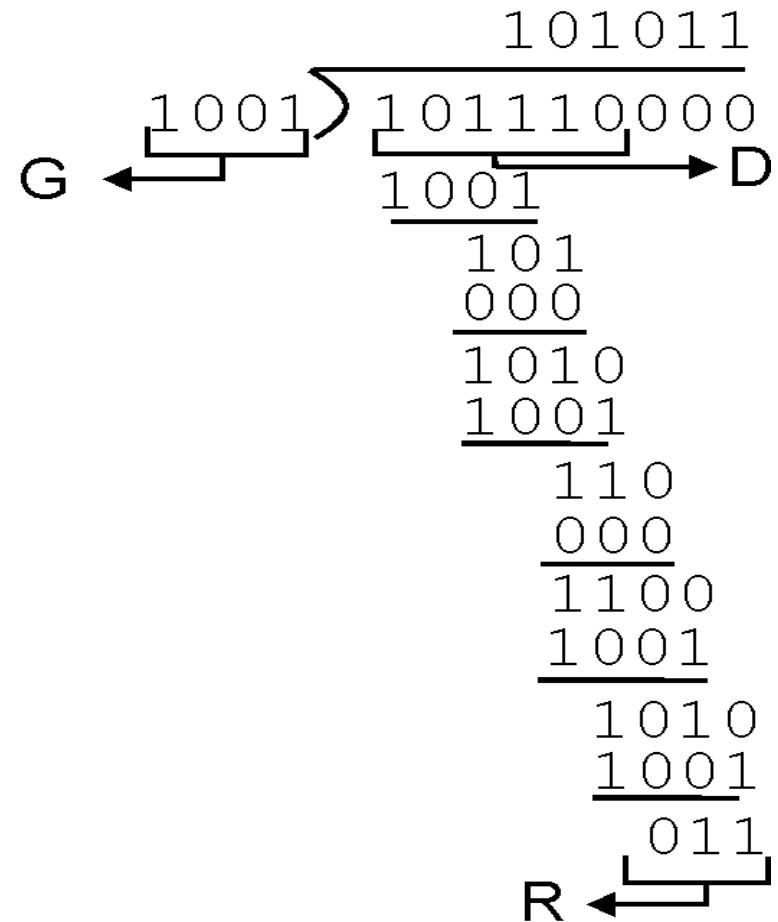
Example

Consider the 4-bit generator, G=1001, and suppose that D has the value 101010. What is the value of R?

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
  ▪ upstream HFC
  ▪ 802.11 wireless LAN

shared wire (e.g., cabled Ethernet)

shared RF (e.g., 802.11 WiFi)
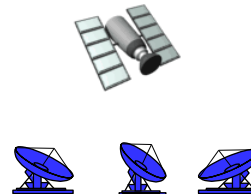
shared RF (satellite)

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

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 = pkt trans time) in each round
- ❖ unused slots go idle
- ❖ example: 6-station LAN, 1,3,4 have pkt, slots 2,5,6 idle
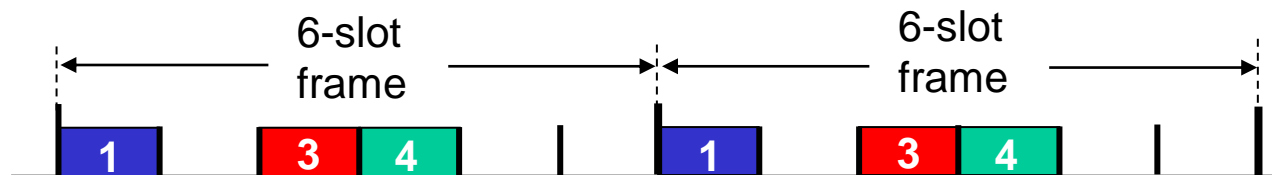
Channel partitioning MAC protocols: FDMA

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 pkt, frequency bands 2,5,6 idle

FDM cable

time

frequency bands
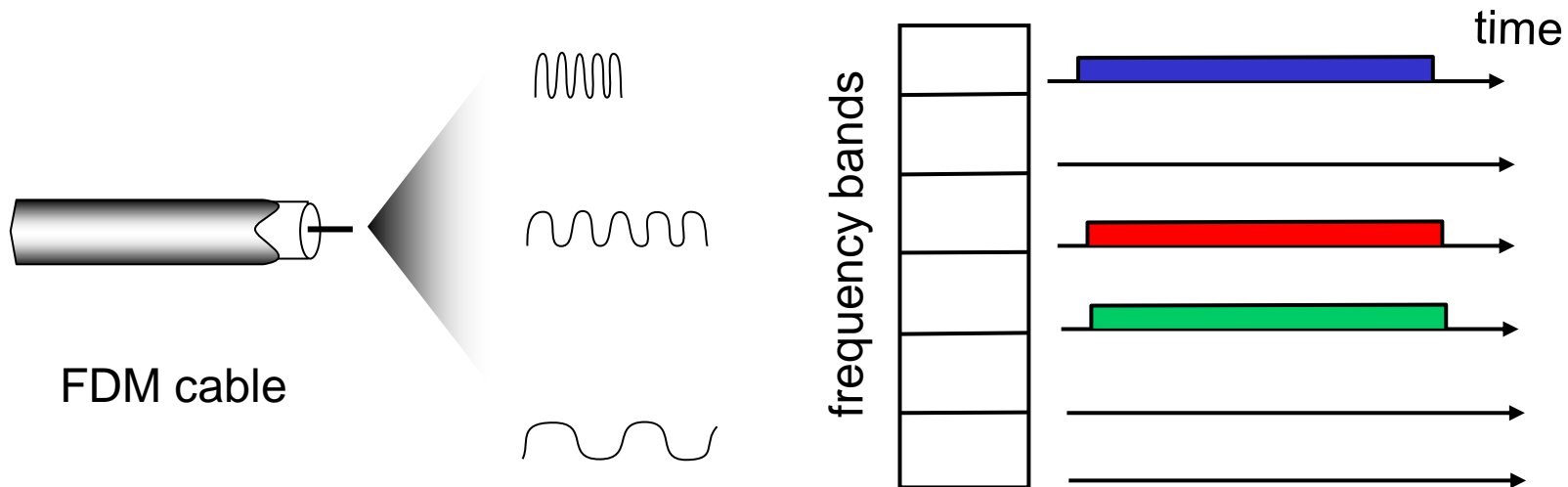
Random access protocols

❖ **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

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

node 1	**1**		**1**			**1**		**1**		
node 2	**2**		**2**	**2**						
node 3	**3**					**3**			**3**	

C   E   C   S   E   C   E   S   S
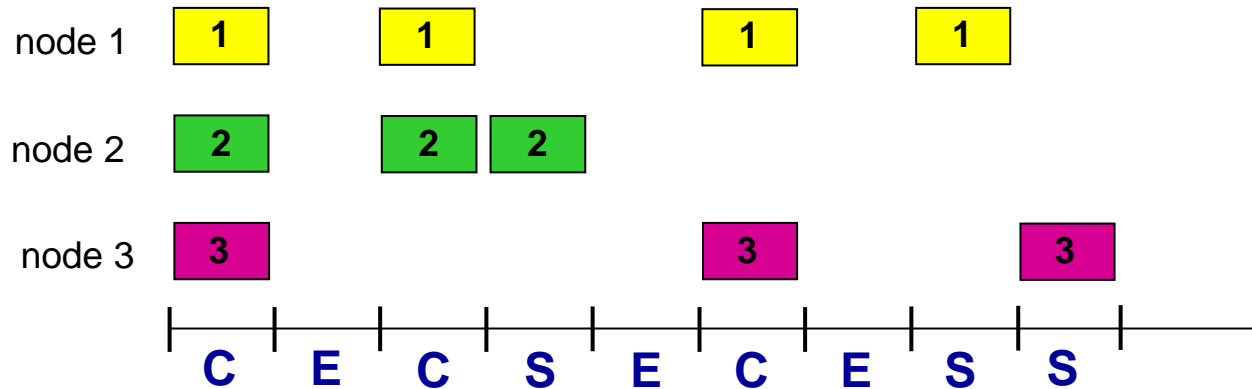
Pros:

❖ 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 $[t_0-1, t_0+1]$

will overlap with start of
← i's frame →

will overlap with end of
← i's frame →

node i frame

t_0-1 t_0 t_0+1
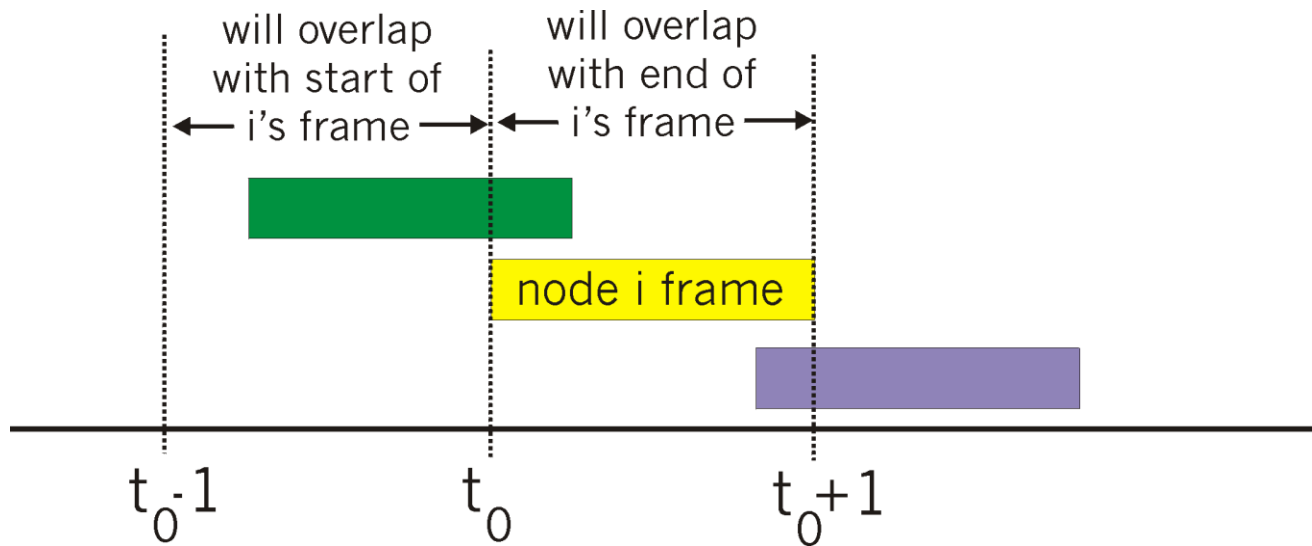
Pure ALOHA efficiency

P(success by given node) = P(node transmits) ·

P(no other node transmits in $[t_0-1, t_0]$ ·

P(no other node transmits in $[t_0-1, t_0]$

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

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

… choosing optimum p and then letting n → ∞

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

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

spatial layout of nodes

❖ collisions *can* still occur: propagation delay means two nodes may not hear each other's transmission

❖ collision: entire packet transmission time wasted
  ▪ distance & propagation delay play role in in determining collision probability

t_0

time

t_1
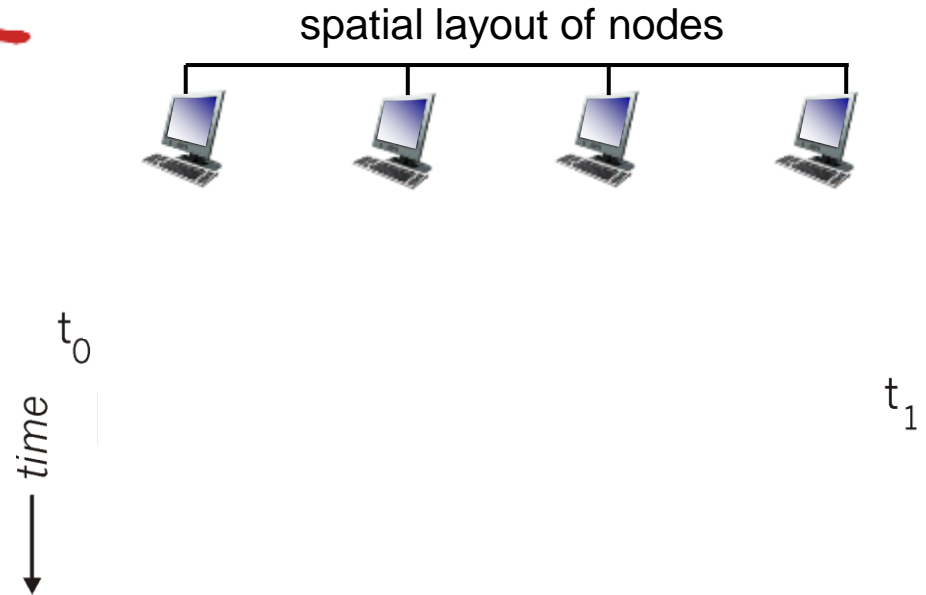
CSMA/CD (collision detection)

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

❖ human analogy: the polite conversationalist

CSMA/CD (collision detection)

spatial layout of nodes

time

t_0

t_1

collision detect/abort time
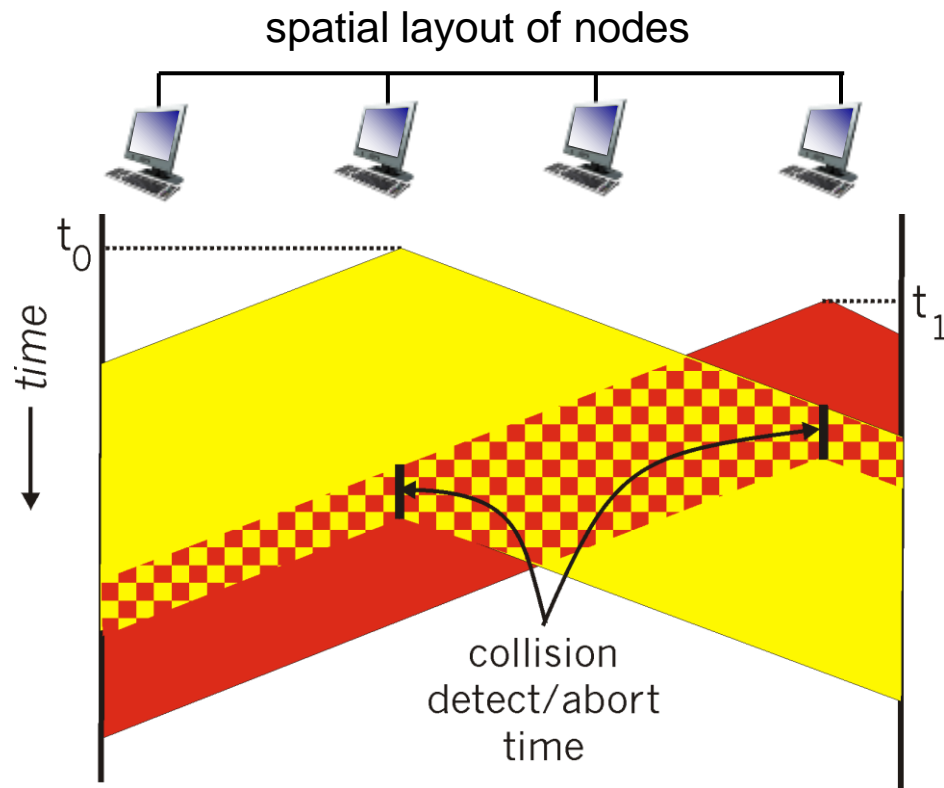
"Taking turns" MAC protocols

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!

"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)

data

data

poll

master

slaves
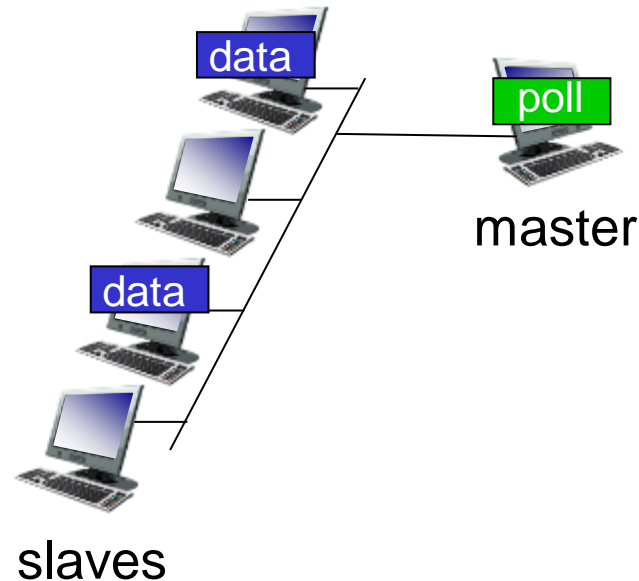
"Taking turns" MAC protocols

token passing:

- ❖ control *token* passed from one node to next sequentially.
- ❖ token message
- ❖ concerns:
  - ■ token overhead
  - ■ latency
  - ■ single point of failure (token)

T

(nothing to send)

T

data

Cable access network

Internet frames,TV channels, control transmitted downstream at different frequencies

cable headend

CMTS

cable modem termination system

ISP

splitter

cable modem

upstream Internet frames, TV control, transmitted upstream at different frequencies in time slots
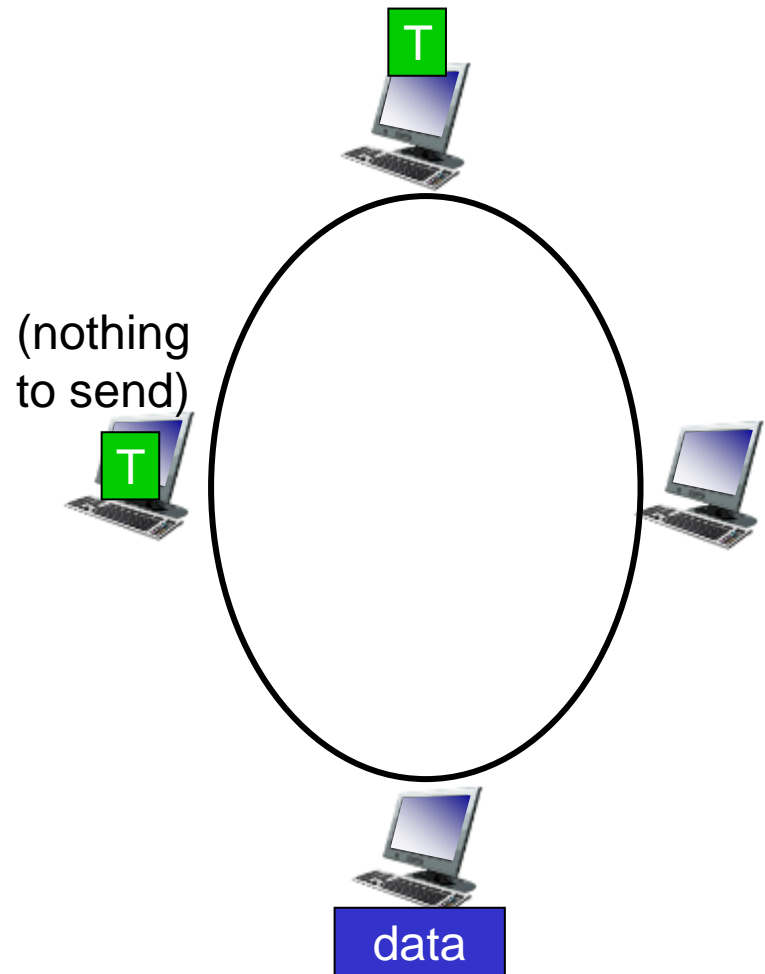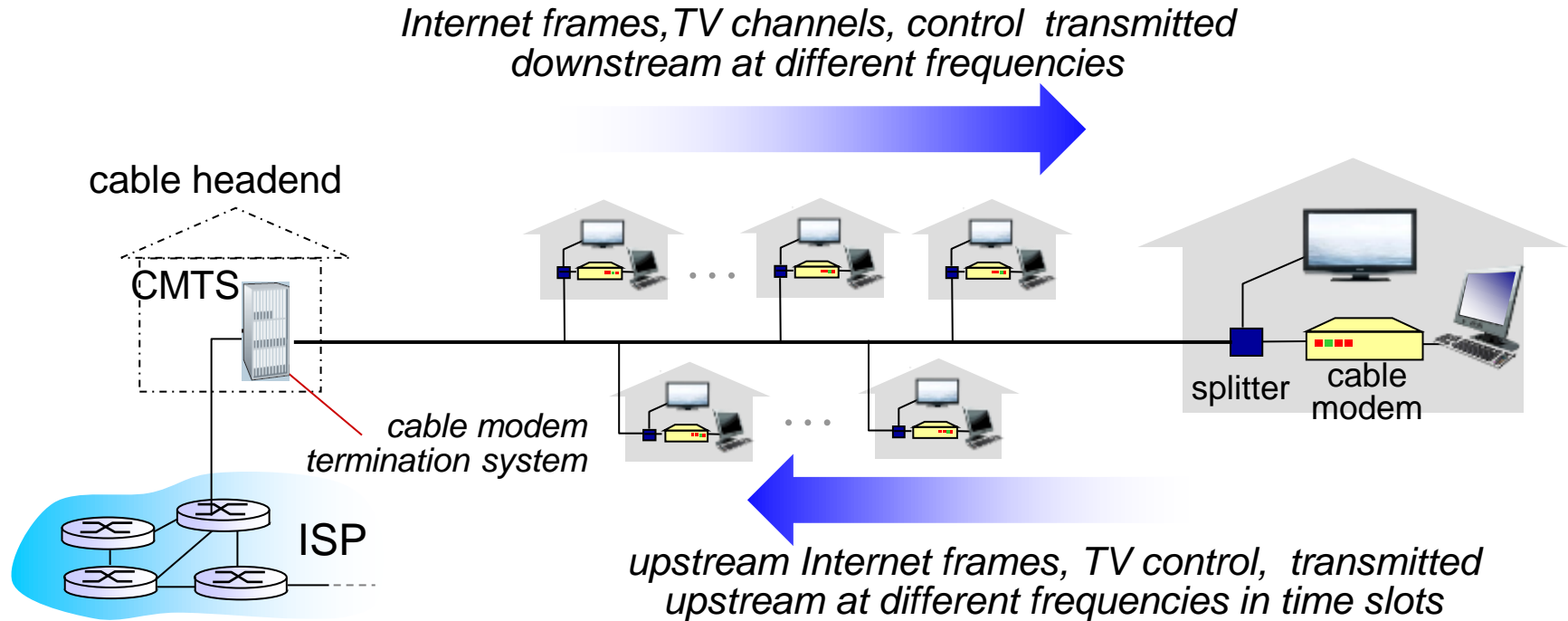
❖ *multiple* 40Mbps downstream (broadcast) channels
  ▪ single CMTS transmits into channels
❖ *multiple* 30 Mbps upstream channels
  ▪ *multiple access:* *all* users contend for certain upstream channel time slots (others assigned)

Cable access network

cable headend

CMTS

MAP frame for Interval [t1, t2]

Downstream channel i

Upstream channel j

t_1 t_2

Residences with cable modems

Minislots containing minislots request frames

Assigned minislots containing cable modem upstream data frames
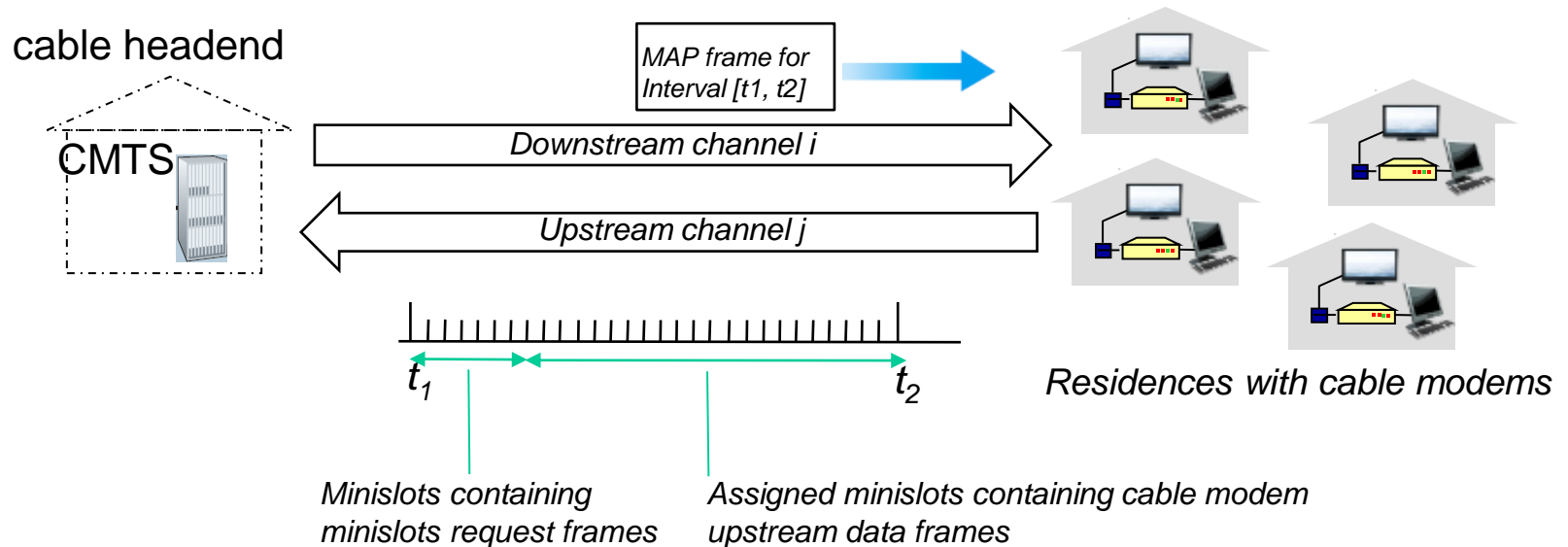
DOCSIS: data over cable service interface spec

- ❖ 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

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

ARP: mapping IP to MAC

❖ 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
  ▪ dest 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":
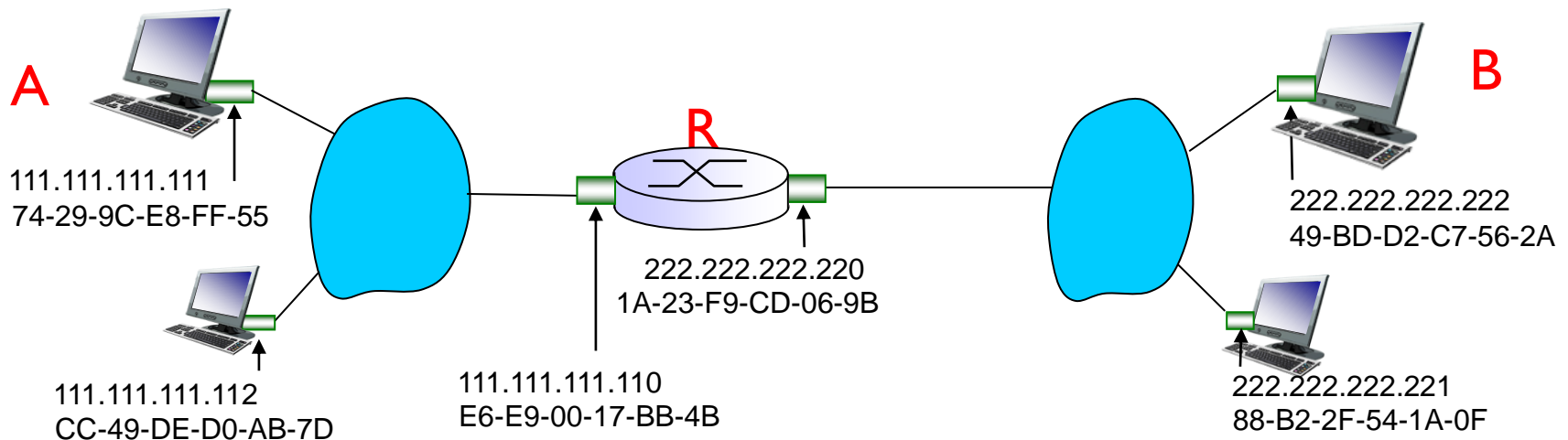 ▪ nodes create their ARP tables *without intervention from net administrator*

Link v.s. App Layer addressing

❖ Address translation
  ▪ Domain name => IP => MAC

❖ Recall Application Layer addressing
  ▪ DNS: hierarchical & distributed databases

❖ Can the Link Layer addressing approach be used in the Application Layer?

Addressing: routing to another LAN

walkthrough: send datagram from A to B via R

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

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

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

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

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

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

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

Addressing: routing to another LAN

❖ A creates IP datagram with IP source A, destination B

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

MAC src: 74-29-9C-E8-FF-55
MAC dest: E6-E9-00-17-BB-4B
IP src: 111.111.111.111
IP dest: 222.222.222.222

IP
Eth
Phy

A

B

R

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

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

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

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

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

222.222.222.221
88-B2-2F-54-1A-0F
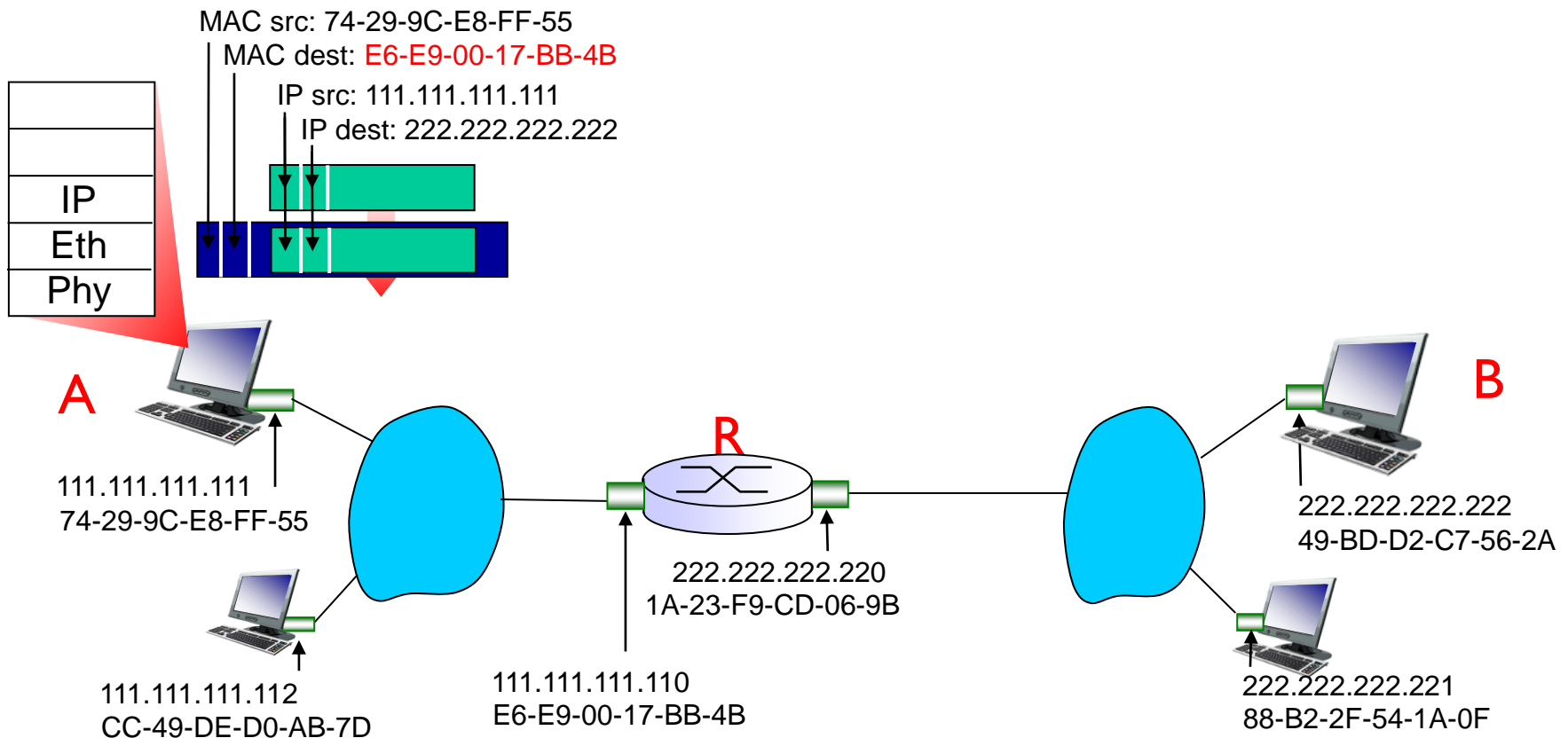
Addressing: routing to another LAN

❖ frame sent from A to R

❖ frame received at R, datagram removed, passed up to IP

MAC src: 74-29-9C-E8-FF-55

MAC dest: E6-E9-00-17-BB-4B

IP src: 111.111.111.111

IP dest: 222.222.222.222

IP
Eth
Phy

IP
Eth
Phy

A

B

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

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

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

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

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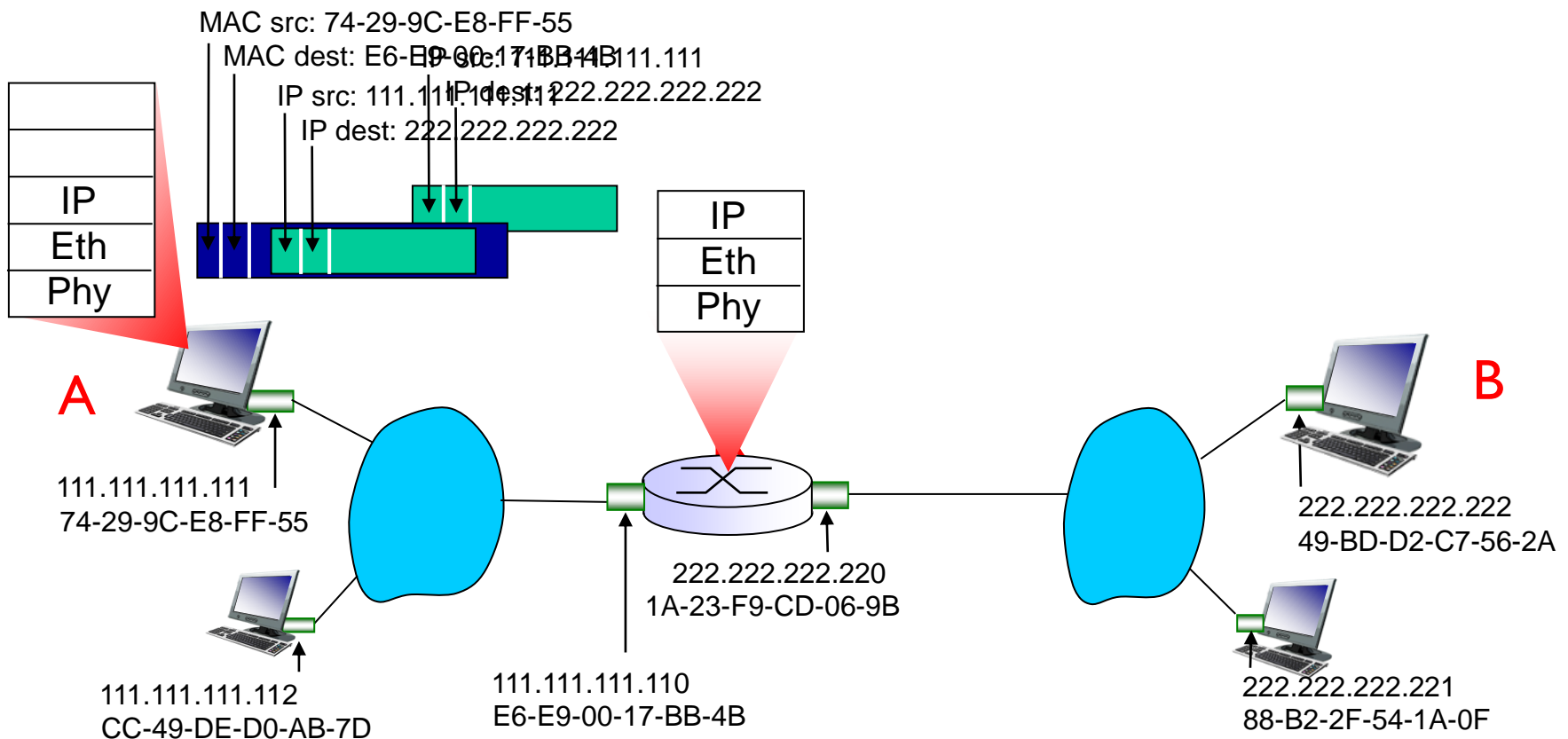E6-E9-00-17-BB-4B

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

Addressing: routing to another LAN

❖ R forwards datagram with IP source A, destination B

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

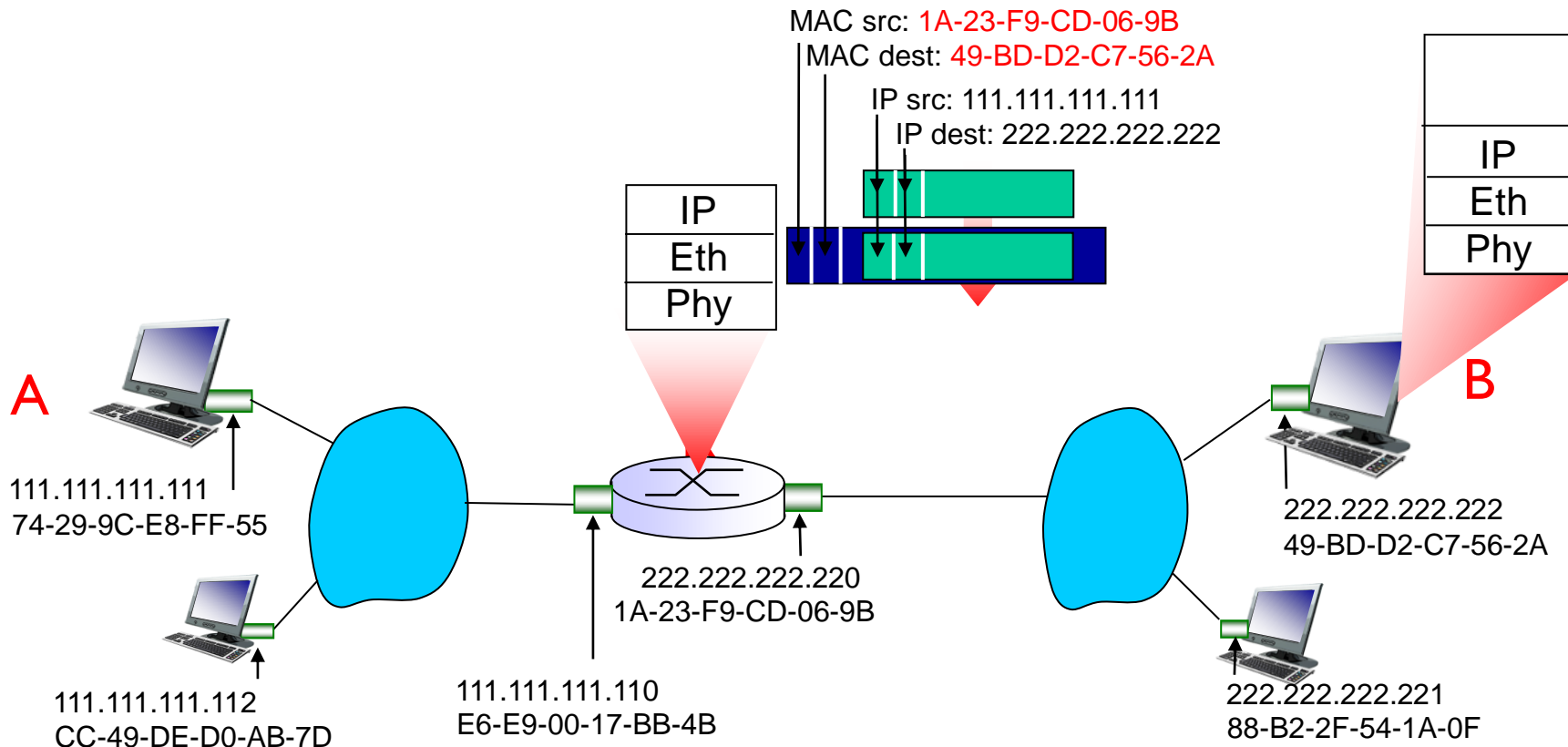MAC src: 1A-23-F9-CD-06-9B
MAC dest: 49-BD-D2-C7-56-2A

IP src: 111.111.111.111
IP dest: 222.222.222.222

IP
Eth
Phy

IP
Eth
Phy

A

B

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

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

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

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

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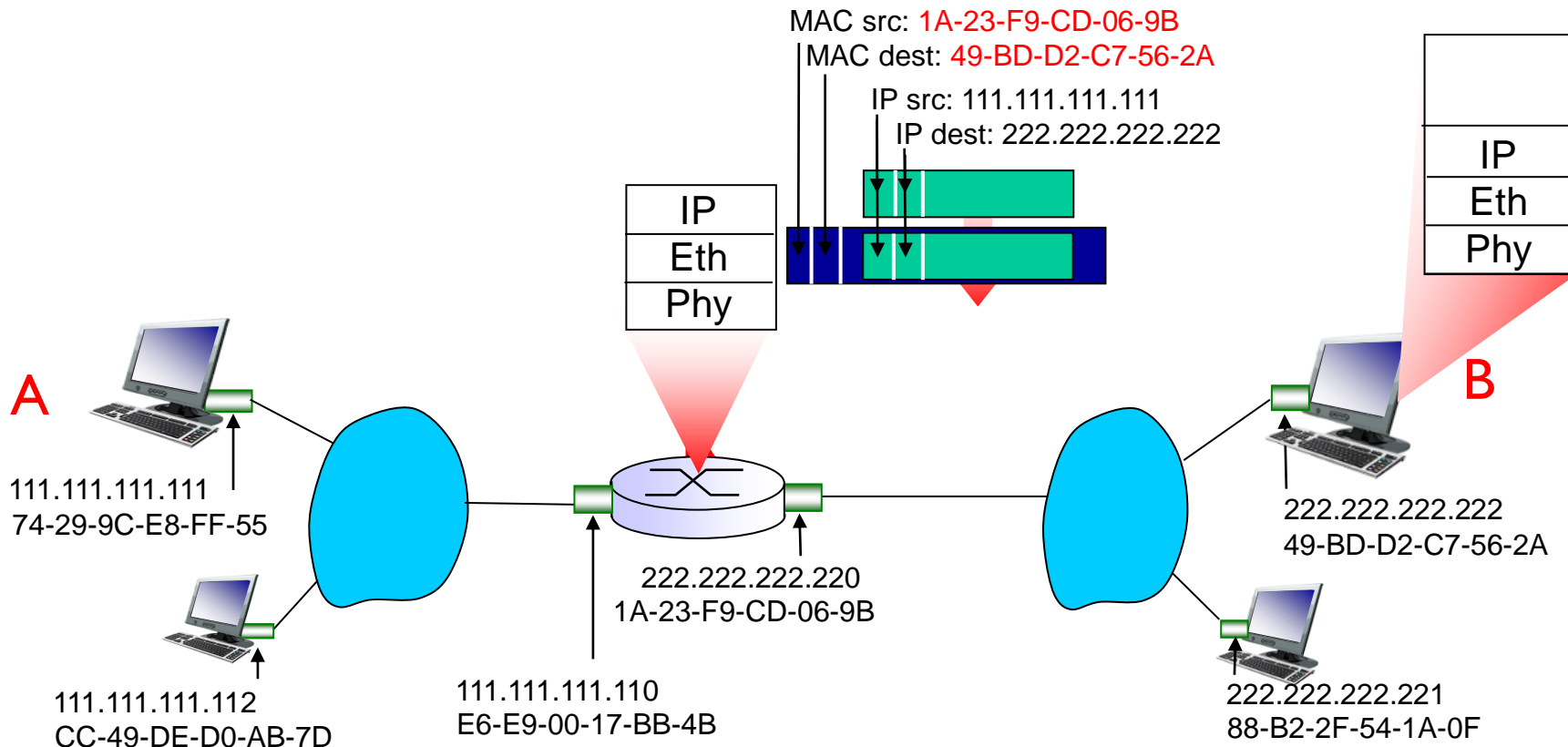E6-E9-00-17-BB-4B

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

Addressing: routing to another LAN

❖ R forwards datagram with IP source A, destination B

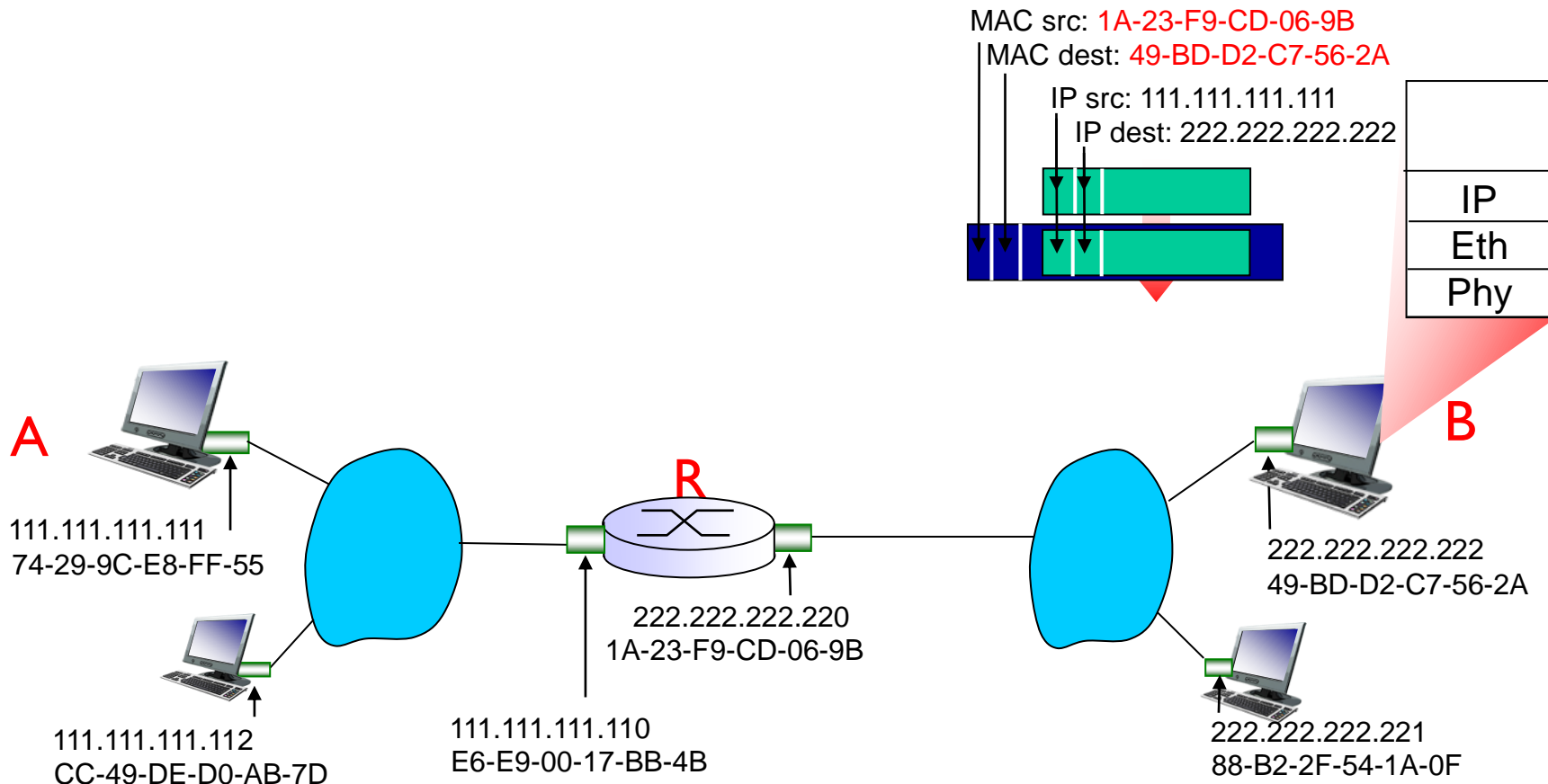❖ R creates link-layer frame with B's MAC address as dest, frame contains A-to-B IP datagram

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

IP src: 111.111.111.111
IP dest: 222.222.222.222

IP
Eth
Phy

IP
Eth
Phy

A

B

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

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

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

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

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

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

Addressing: routing to another LAN

❖ R forwards datagram with IP source A, destination B

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

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

IP src: 111.111.111.111
IP dest: 222.222.222.222

IP
Eth
Phy

A

R

B

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

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

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

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

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

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

Link layer, LANs: outline

6.1 introduction, services

6.2 error detection, correction

6.3 multiple access protocols

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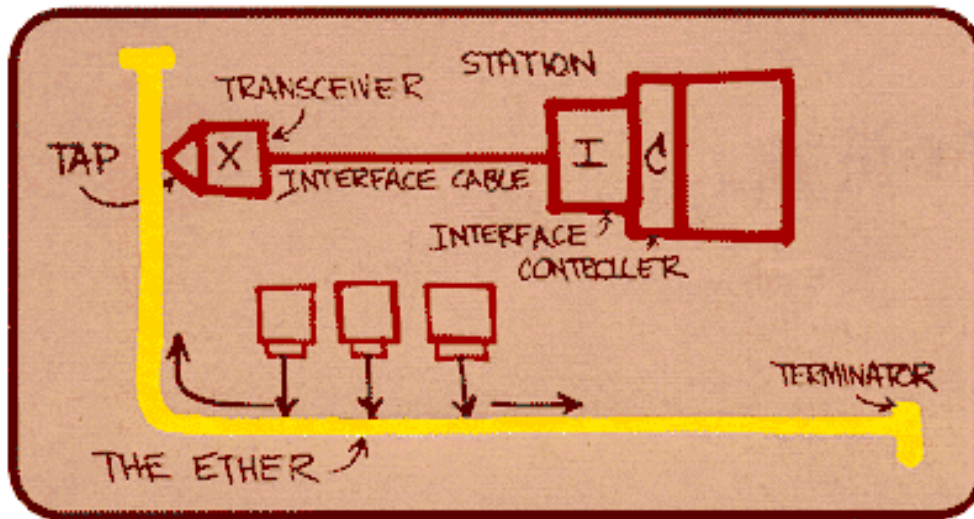- switches
- VLANS

6.5 link virtualization: MPLS

6.6 data center networking

6.7 a day in the life of a web request
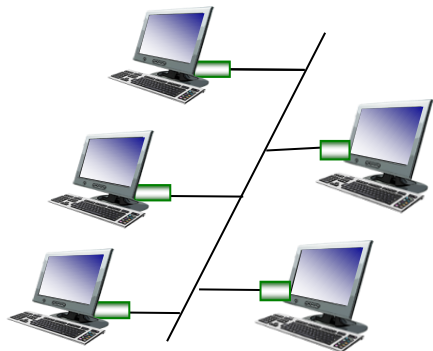
Ethernet

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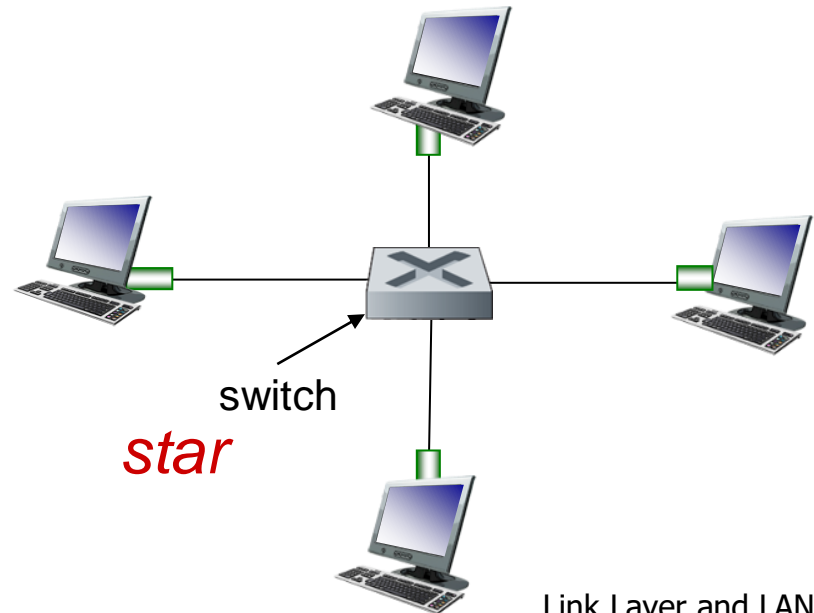- ❖ kept up with speed race: 10 Mbps – 100 Gbps

Metcalfe's Ethernet sketch

Ethernet: physical topology

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

switch

*star*

bus: coaxial cable

Ethernet frame structure

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

type

preamble	dest. address	source address		data (payload)	CRC

preamble:

❖ 7 bytes with pattern 10101010 followed by one byte with pattern 10101011

❖ used to synchronize receiver, sender clock rates

SOF
↓

`10101010101010101010101010101010101010101010101010101010101010` `11`
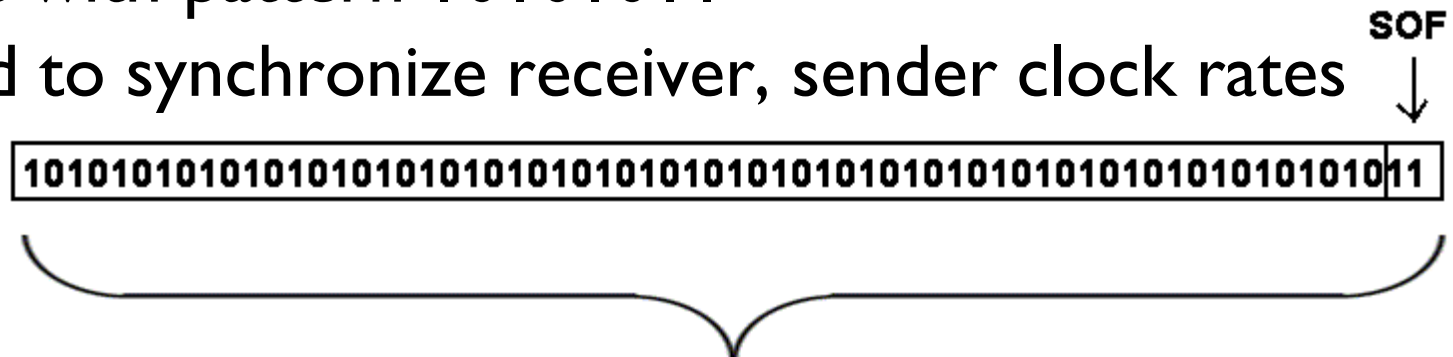
Preamble - 64 bits (8 bytes)

Ethernet frame structure (more)

❖ *addresses:* 6 byte source, destination MAC addresses
  - if adapter receives frame with matching destination address, or with broadcast address (e.g. ARP packet), it passes data in frame to network layer protocol
  - otherwise, adapter discards frame

❖ *type:* indicates higher layer protocol (mostly IP but others possible, e.g., Novell IPX, AppleTalk)

❖ *CRC:* cyclic redundancy check at receiver
  - error detected: frame is dropped

type

preamble	dest. address	source address		data (payload)	CRC

Ethernet: unreliable, connectionless

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

❖ *unreliable:* receiving NIC doesnt 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 wth binary backoff*
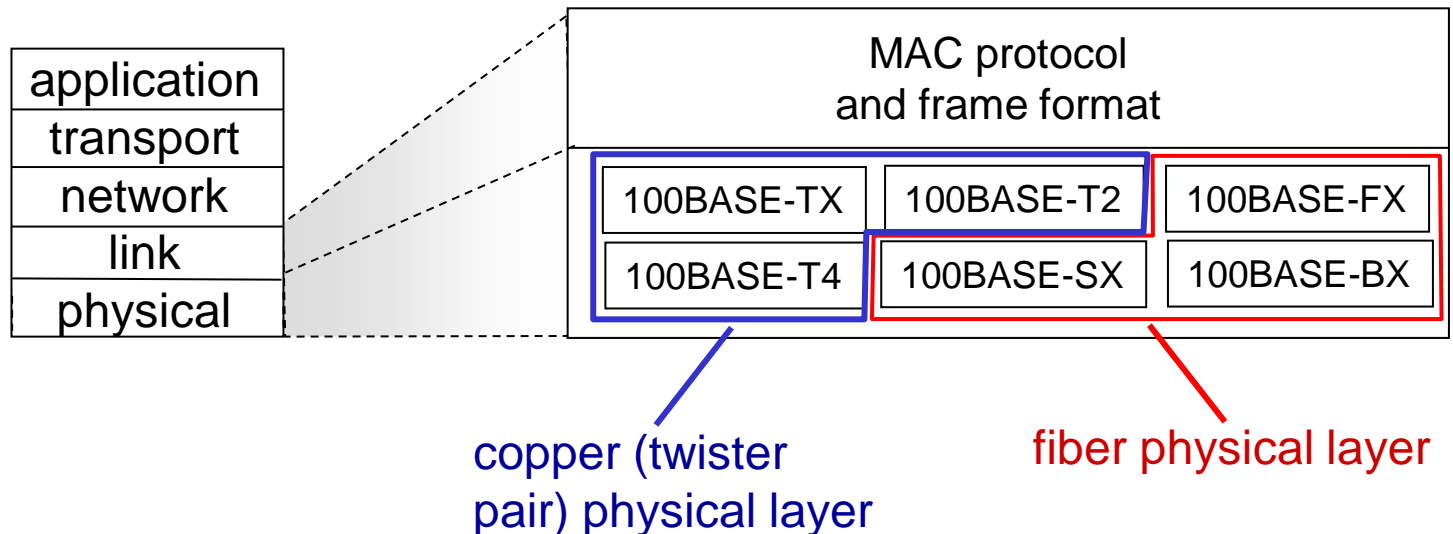
Ethernet CSMA/CD algorithm

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 mth collision, NIC chooses K at random from $\{0,1,2, \ldots, 2^m-1\}$. NIC waits $K \cdot 512$ bit times, returns to Step 2
   - longer backoff interval with more collisions

Example

❖ In CSMA/CD Ethernet, after the fifth collision, what is the probability that a node chooses K=4? The result K=4 corresponds to a delay of how many seconds on a 10 Mbps Ethernet?

802.3 Ethernet standards: link & physical layers

❖ *many* different Ethernet standards
  - common MAC protocol and frame format
  - different speeds: 10 Mbps - 100G bps
  - different physical layer media: fiber, cable

application
transport
network
link
physical

MAC protocol
and frame format

100BASE-TX	100BASE-T2	100BASE-FX
100BASE-T4	100BASE-SX	100BASE-BX

copper (twister pair) physical layer

fiber physical layer

Link layer, LANs: outline

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

❖ switch buffers packets

❖ Ethernet protocol used on *each* incoming link, but no collisions; full duplex

  ▪ each link is its own collision domain

❖ *switching:* A-to-A' and B-to-B' can transmit simultaneously, without collisions
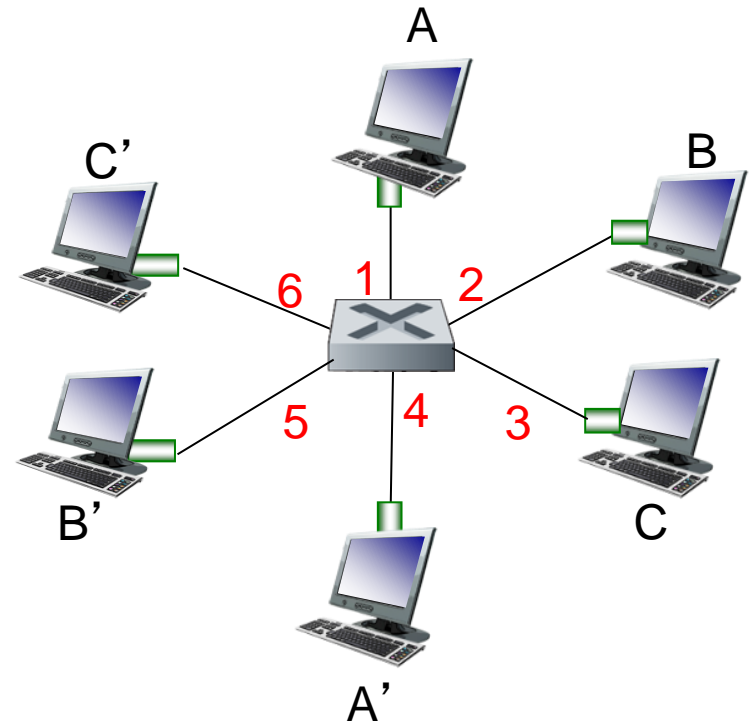
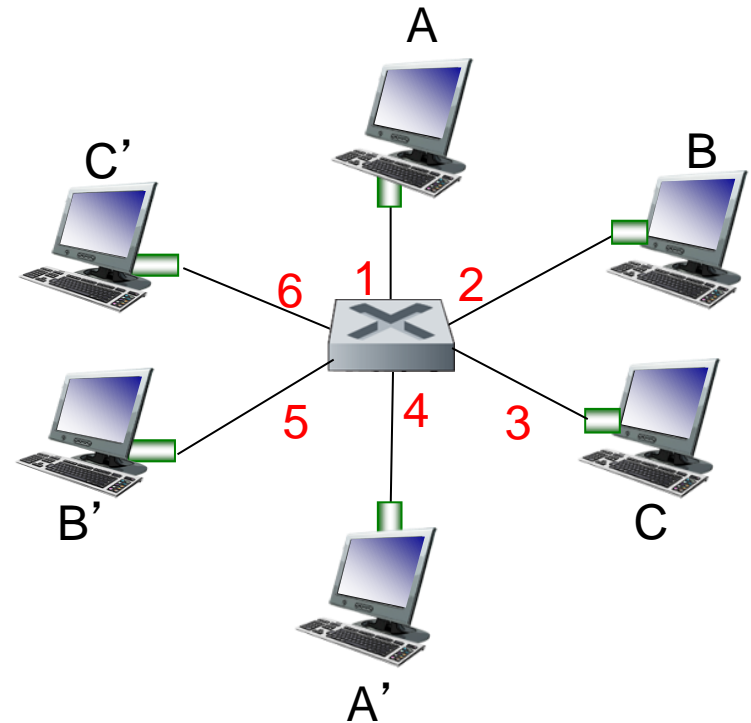*switch with six interfaces*
(1,2,3,4,5,6)

Switch forwarding table

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

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

   ▪ *(MAC address of host, interface to reach host, time stamp)*

   ▪ *looks like a routing table!*

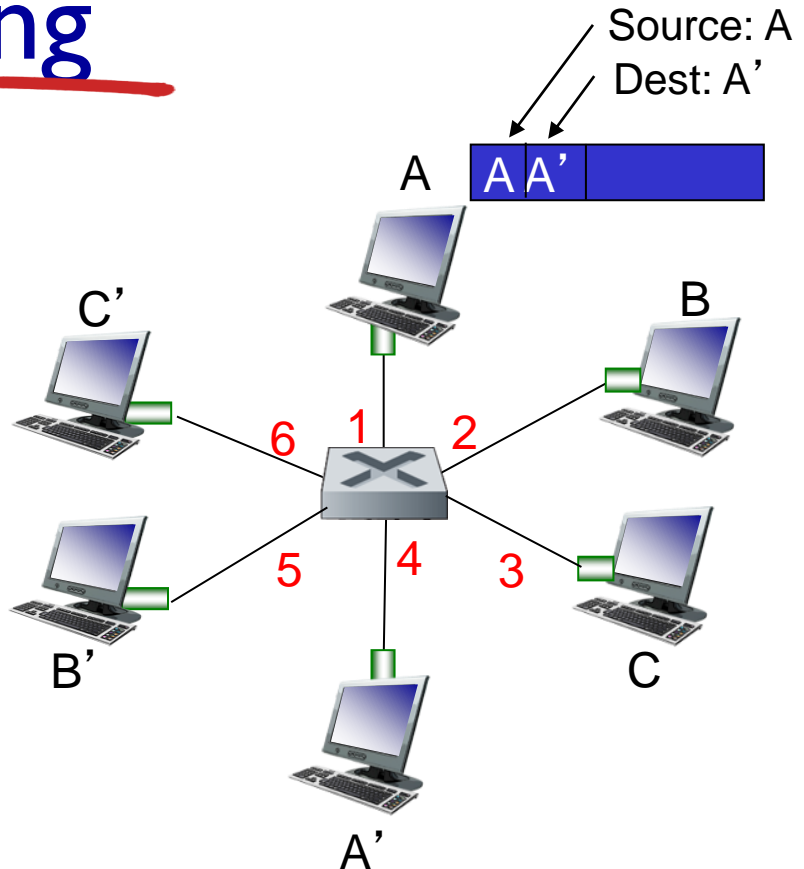*Q: how are entries created, maintained in switch table?*

   ▪ *something like a routing protocol?*

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

Switch: self-learning

Source: A
Dest: A'

A   [A|A'|]

❖ switch *learns* which hosts can be reached through which interfaces

  ▪ when frame received, switch "learns" location of sender: incoming LAN segment

  ▪ records sender/location pair in switch table

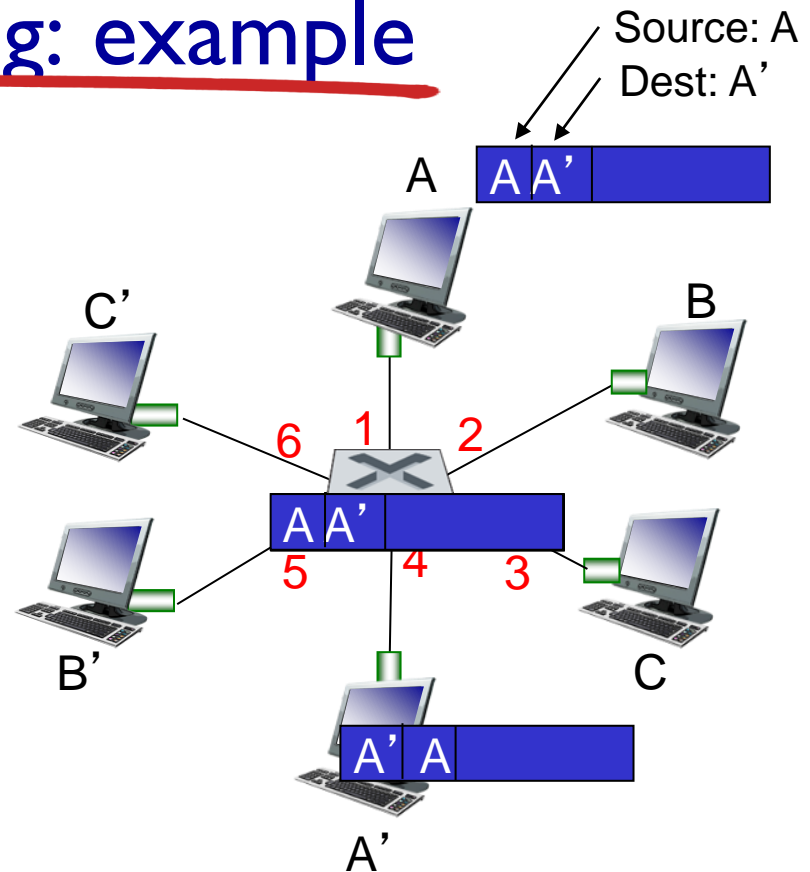| MAC addr	interface	TTL
A	1	60

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*

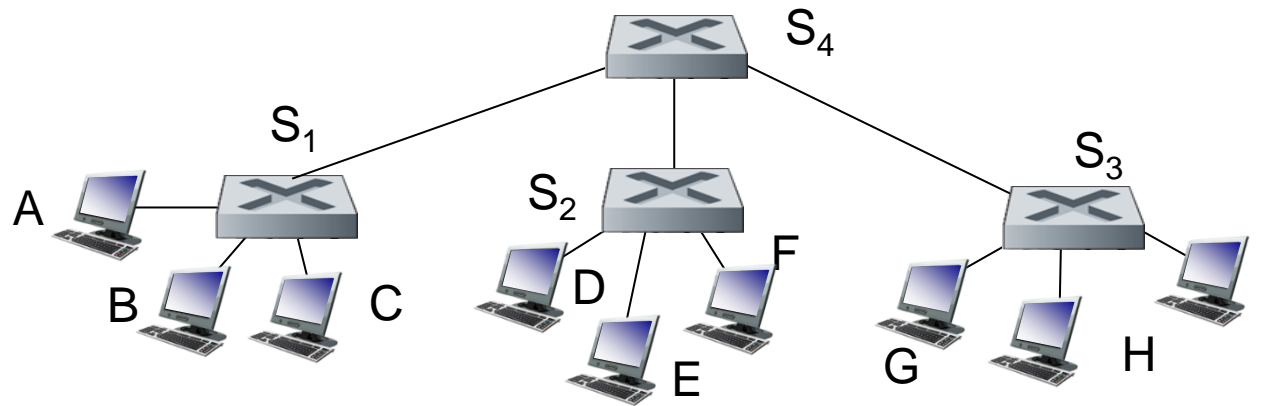❖ destination A location known: *selectively send on just one link*

Source: A
Dest: A'

MAC addr	interface	TTL
A	1	60
A'	4	60

switch table (initially empty)

Interconnecting switches
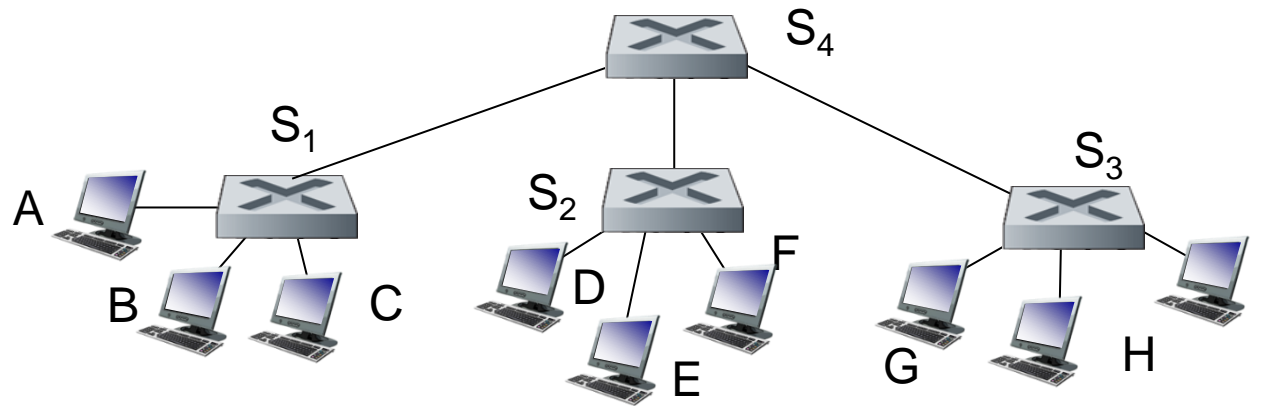
❖ switches can be connected together

$Q:$ sending from A to G - how does S_1 know to forward frame destined to 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

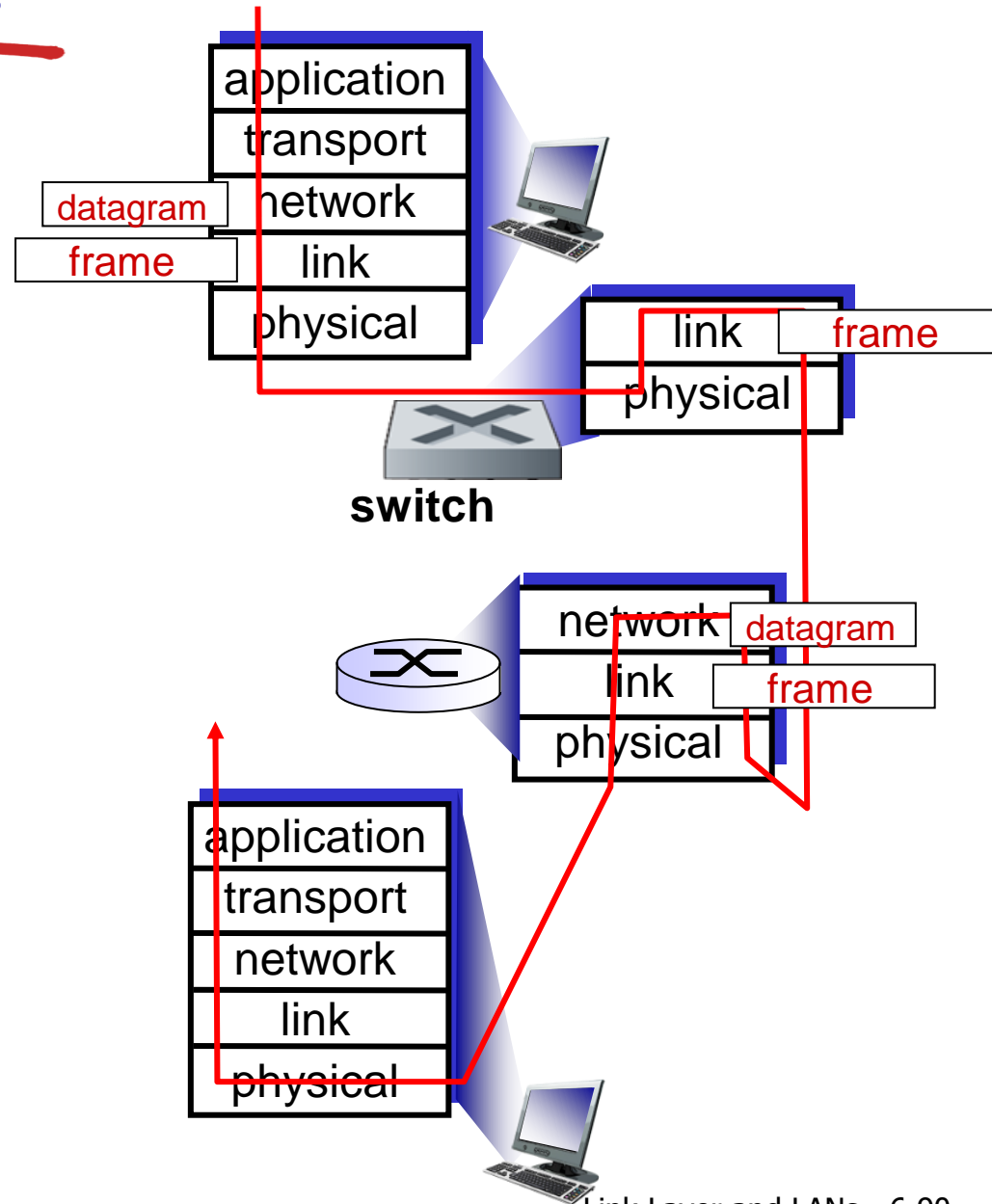* *Q:* show switch tables and packet forwarding in S_1, S_2, S_3, S_4

Switches vs. routers

both are store-and-forward:

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

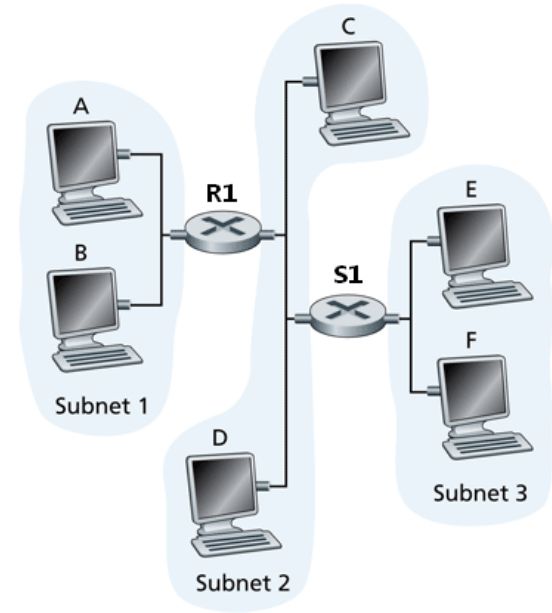▪*switches*: link-layer devices (examine link-layer headers)

both have forwarding tables:

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

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

application
transport
network
link
physical

datagram

frame

link
physical

frame

switch

network
link
physical

datagram

frame

application
transport
network
link
physical

Example

❖ Consider the following network. The router R1 and hosts C, D, E and F are all star-connected into a switch S1. Suppose host A sends an IP datagram to host F. Give the source and destination IP and MAC addresses in the frame encapsulating this IP datagram as the frame is transmitted (i) from A to the R1, (ii) from R1 to the S1, (iii) from S1 to F.

A
C
R1
E
B
S1
F
Subnet 1
D
Subnet 3
Subnet 2

	Src IP	Dst IP	Src MAC	Dst MAC
A => R1				
R1 => S1				
S1 => F				

Link layer, LANs: outline

VLANs: motivation

Computer Science

Electrical Engineering

Computer Engineering
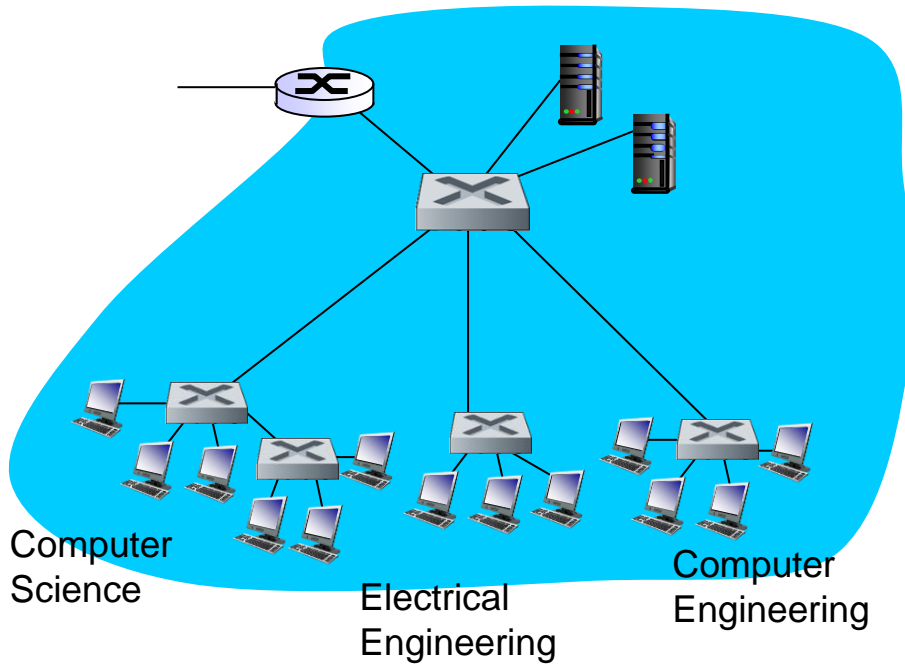
consider:

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

❖ single broadcast domain:

▪ all layer-2 broadcast traffic (ARP, DHCP, unknown location of destination MAC address) must cross entire LAN

▪ security/privacy, efficiency issues

VLANs

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

Electrical Engineering
(VLAN ports 1-8)

Computer Science
(VLAN ports 9-15)

Virtual Local Area Network

switch(es) supporting VLAN capabilities can be configured to define multiple **virtual** LANS over single physical LAN infrastructure.
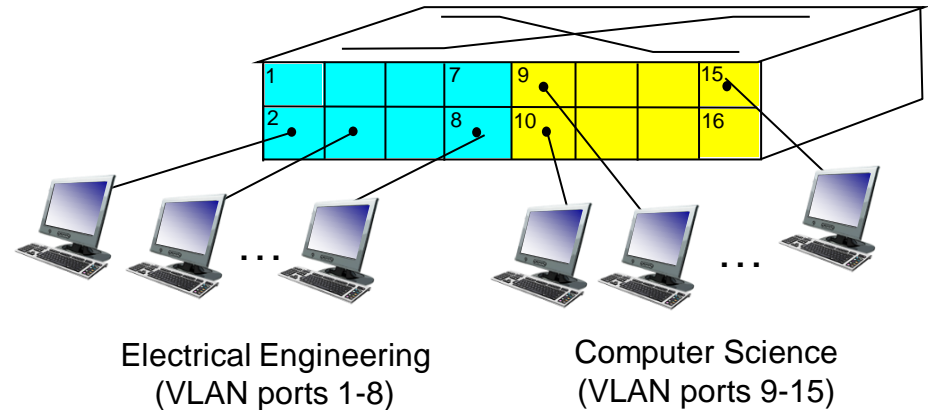
… operates as *multiple* virtual switches

Electrical Engineering
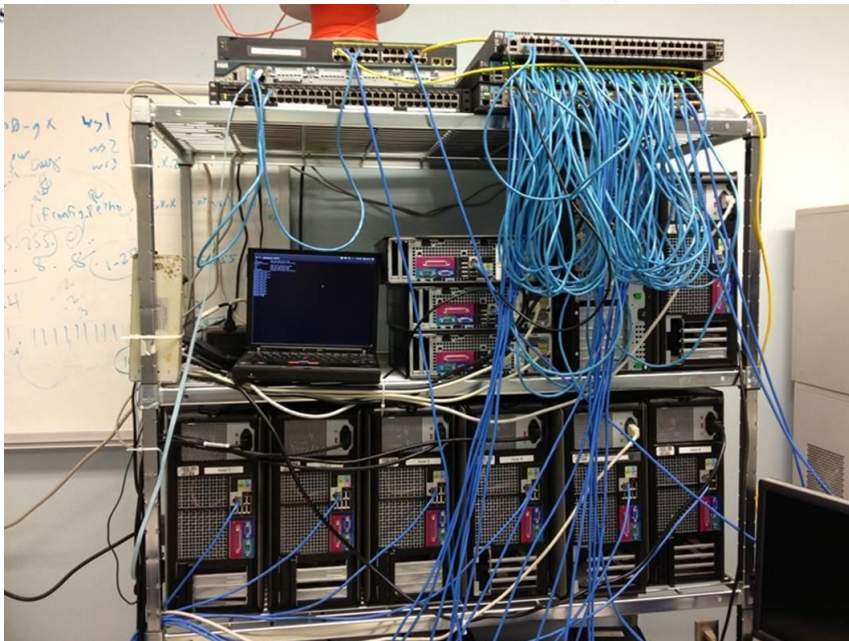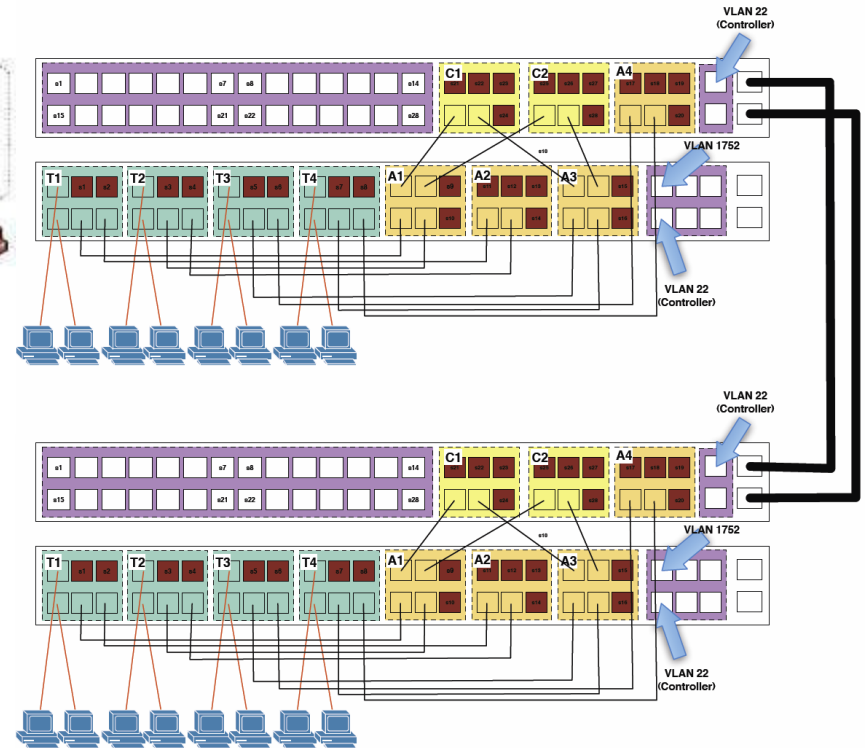(VLAN ports 1-8)

Computer Science
(VLAN ports 9-16)
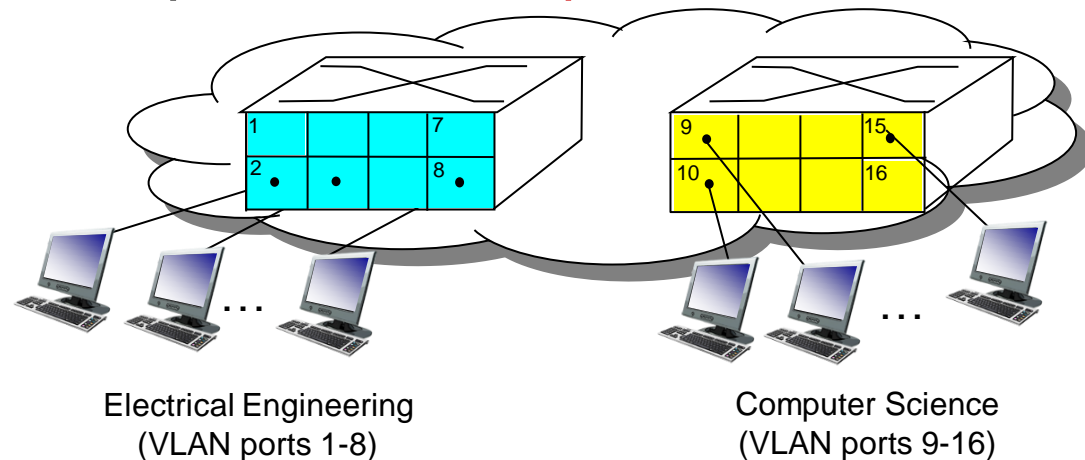
Port-based VLAN

❖ *traffic isolation:* frames to/from ports 1-8 can *only* reach ports 1-8

  ▪ can also define VLAN based on MAC addresses of endpoints, rather than switch port

❖ *dynamic membership:* ports can be dynamically assigned among VLANs

❖ *forwarding between VLANS:* done via routing (just as with separate switches)

  ▪ in practice vendors sell combined switches plus routers

router

Electrical Engineering
(VLAN ports 1-8)

Computer Science
(VLAN ports 9-15)
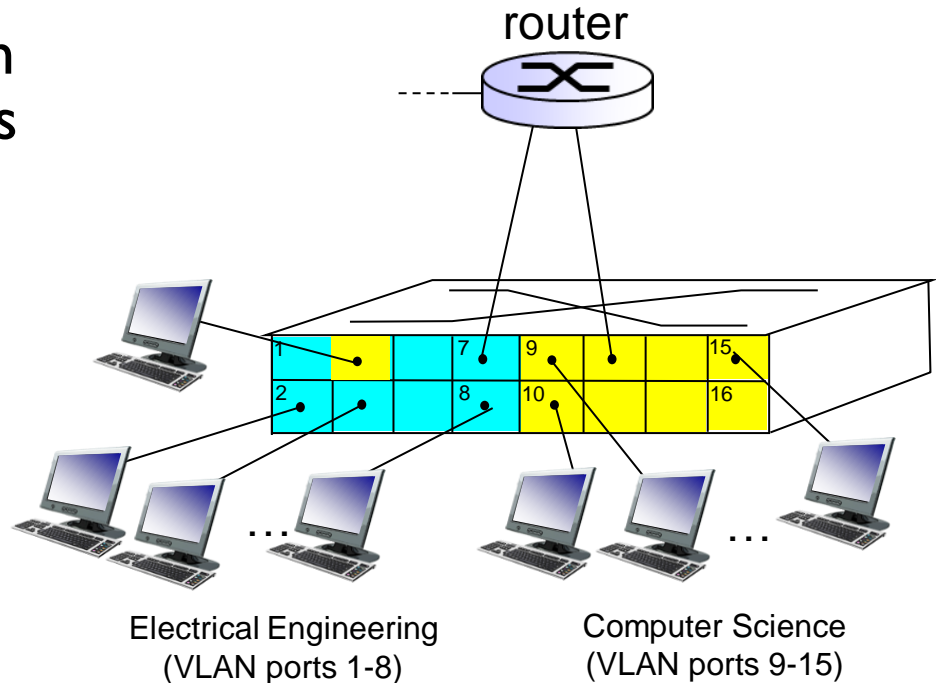
VLAN testbed
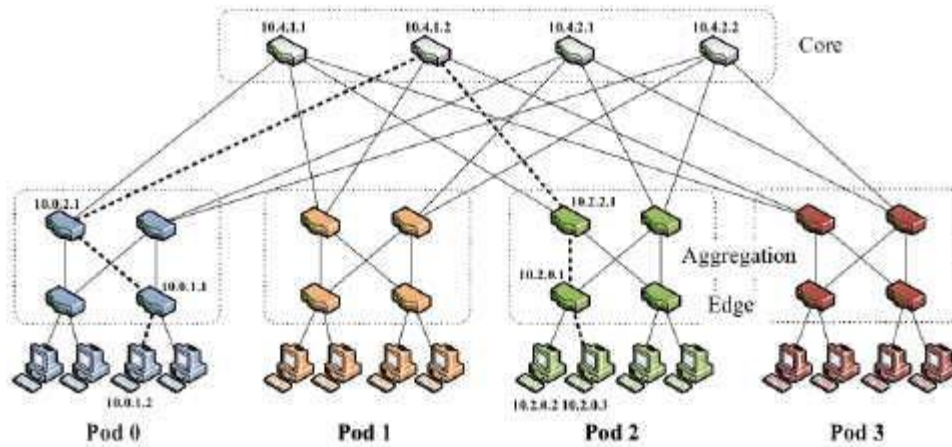
VLANS spanning multiple switches

Electrical Engineering
(VLAN ports 1-8)

Computer Science
(VLAN ports 9-15)

Ports 2,3,5 belong to EE VLAN
Ports 4,6,7,8 belong to CS VLAN
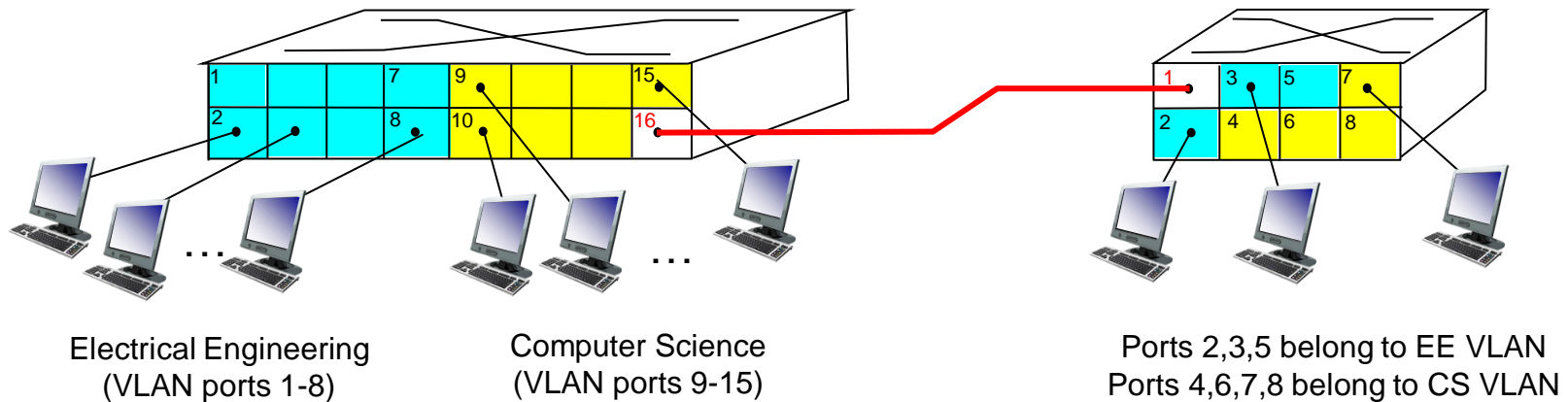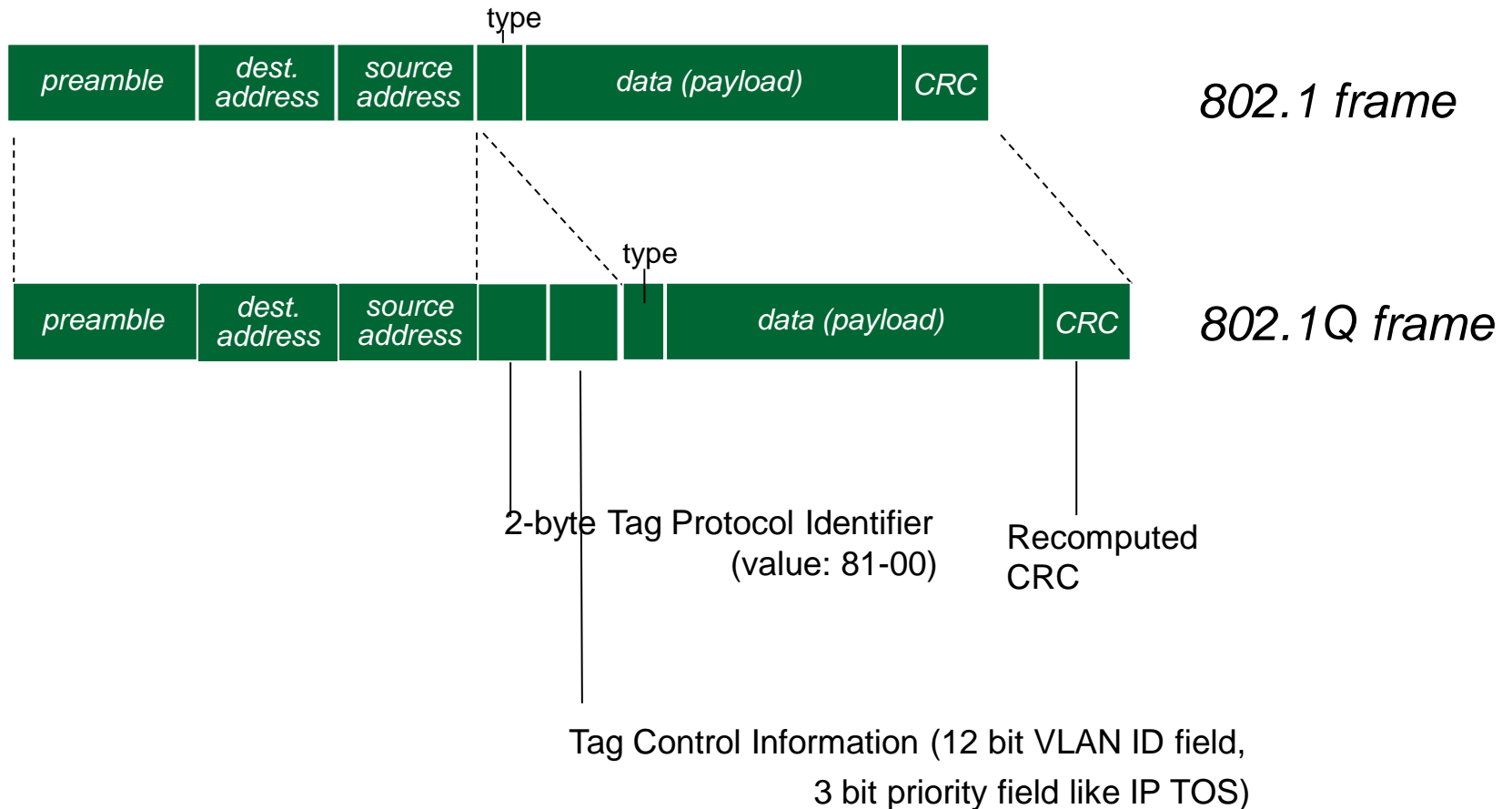
❖ *trunk port:* carries frames between VLANS defined over multiple physical switches

 ▪ frames forwarded within VLAN between switches can't be vanilla 802.1 frames (must carry VLAN ID info)

 ▪ 802.1q protocol adds/removed additional header fields for frames forwarded between trunk ports

802.1Q VLAN frame format

type

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

802.1 frame

type

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

802.1Q frame

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

Recomputed CRC

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

Link layer, LANs: outline

6.1 introduction, services

6.2 error detection, correction

6.3 multiple access protocols

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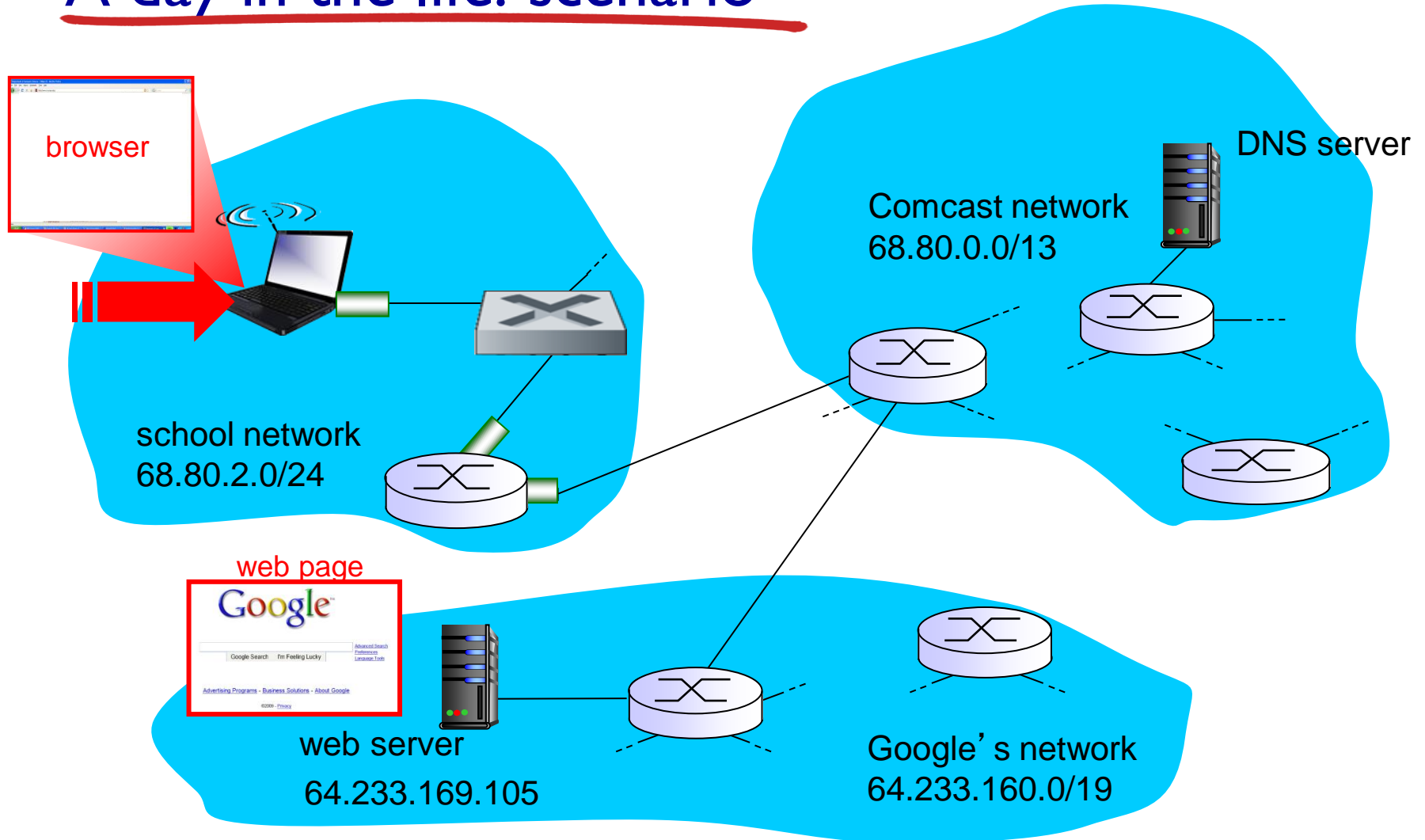- Ethernet
- switches
- VLANS

6.5 link virtualization: MPLS

6.6 data center networking

6.7 a day in the life of a web request

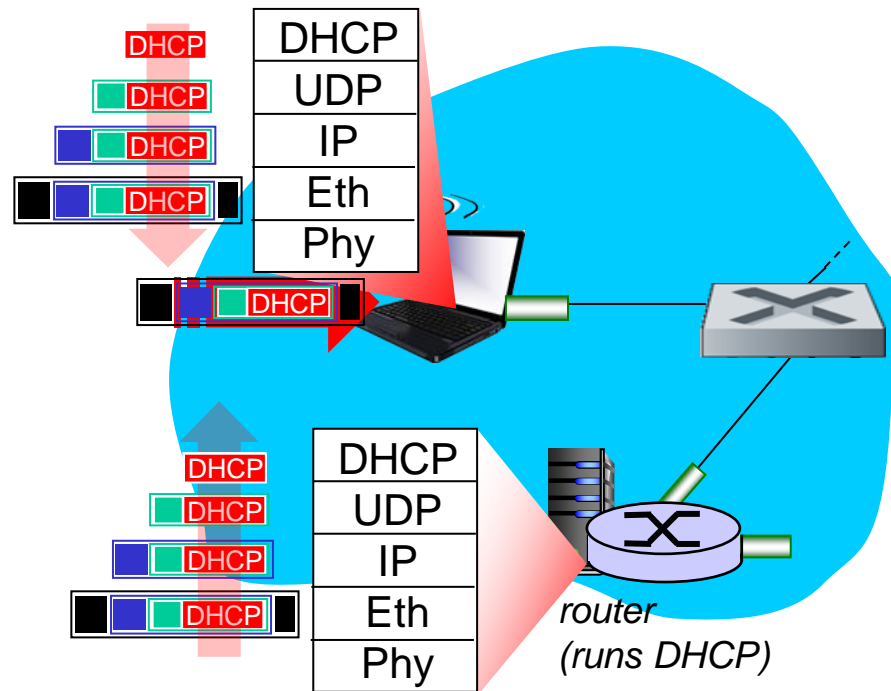# *Synthesis:* a day in the life of a web request

❖ journey down protocol stack complete!

  ▪ application, transport, network, link

❖ putting-it-all-together: synthesis!

  ▪ *goal:* identify, review, understand protocols (at all layers) involved in seemingly simple scenario: requesting www page

  ▪ *scenario:* student attaches laptop to campus network, requests/receives www.google.com

A day in the life: scenario

browser

DNS server

Comcast network
68.80.0.0/13

school network
68.80.2.0/24

web page

Google

Google Search   I'm Feeling Lucky

web server
64.233.169.105

Google's network
64.233.160.0/19

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

router
(runs DHCP)

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

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

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

- ❖ Ethernet *demuxed* to IP demuxed, UDP demuxed to DHCP

A day in the life… connecting to the Internet

DHCP
UDP
IP
Eth
Phy

DHCP
UDP
IP
Eth
Phy
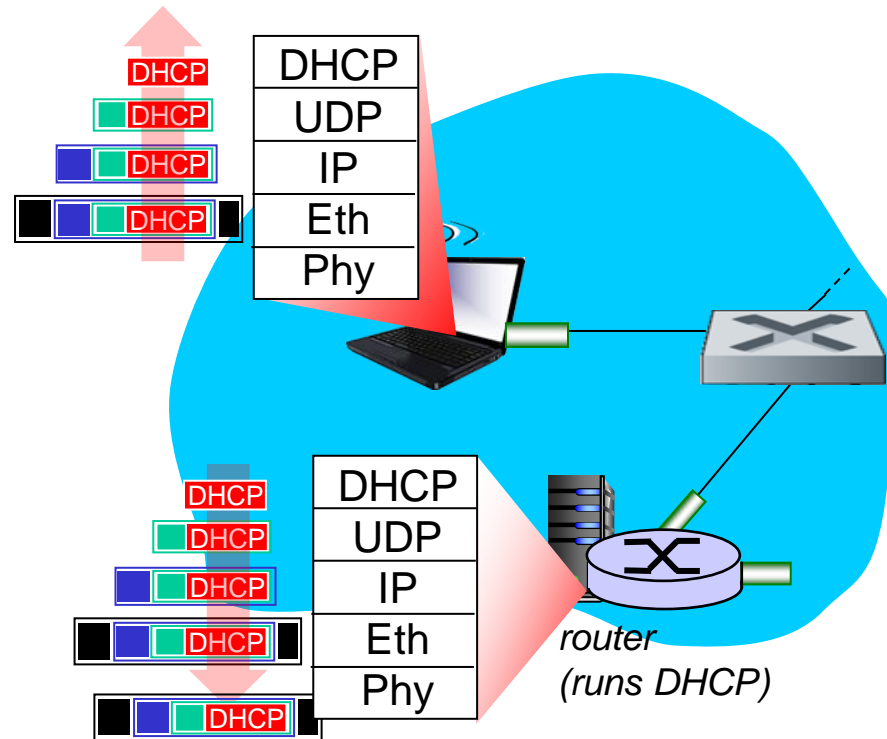
*router
(runs DHCP)*

- ❖ DHCP server formulates *DHCP ACK* containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- ❖ encapsulation at DHCP server, frame forwarded (*switch learning*) through LAN, demultiplexing at client
- ❖ DHCP client receives DHCP ACK reply

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

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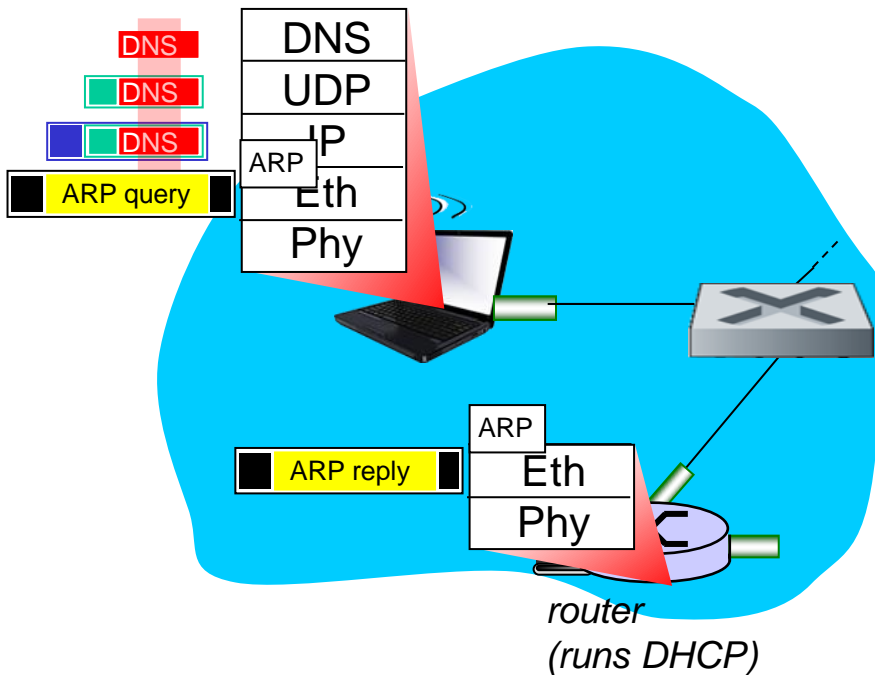
router
(runs DHCP)

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

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

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

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

A day in the life… using DNS

DNS server

Comcast network
68.80.0.0/13
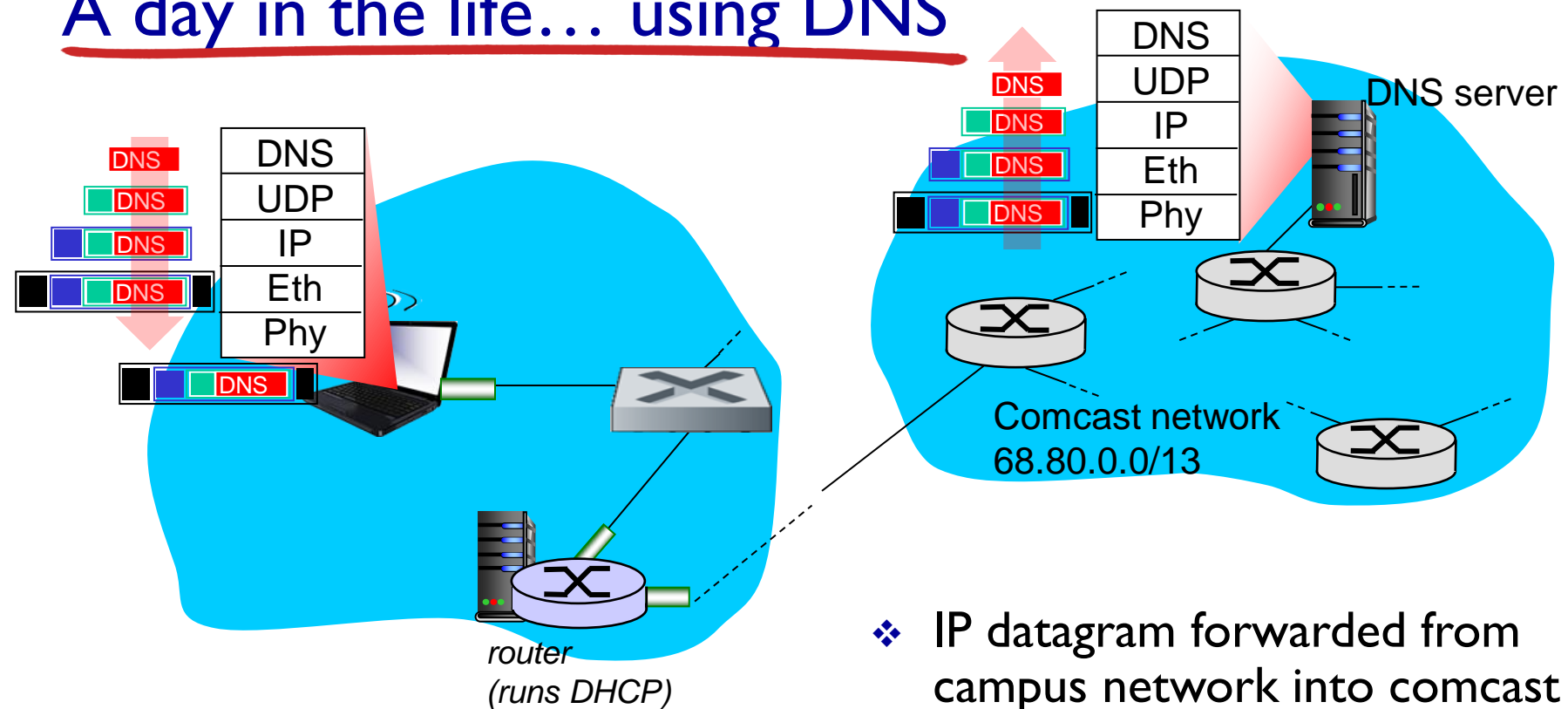
router
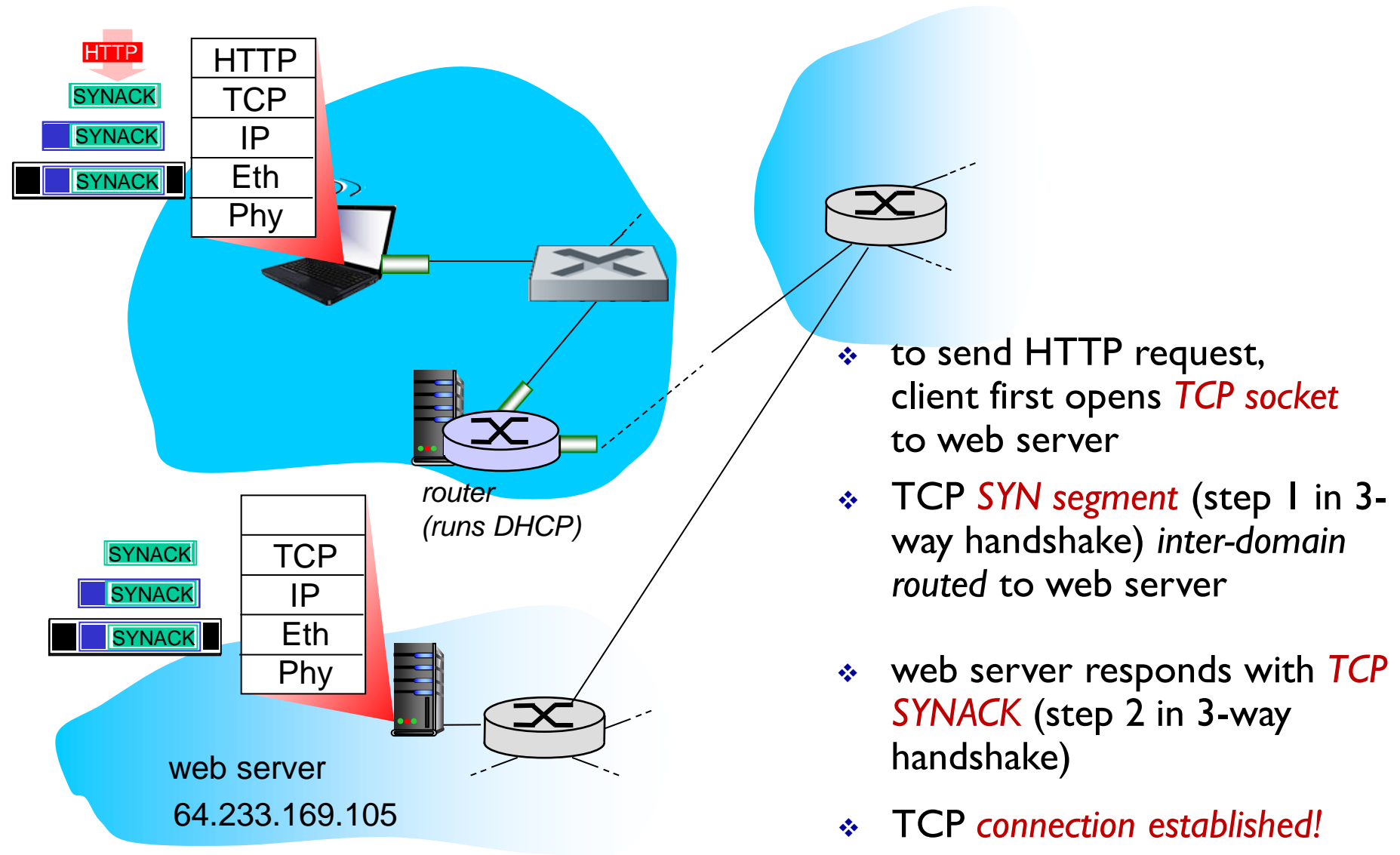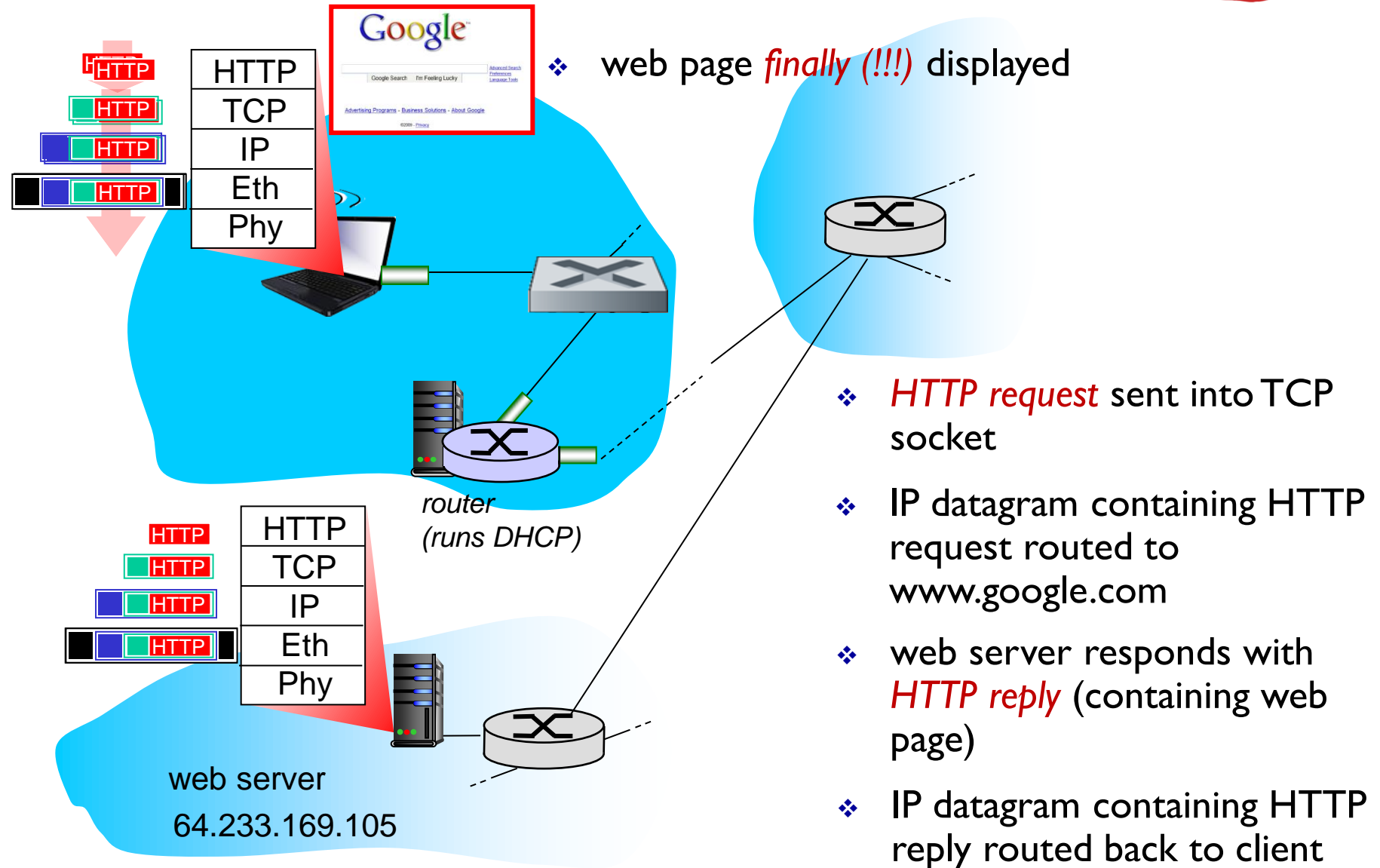*(runs DHCP)*

❖ 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

❖ demux'ed 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!*

router
(runs DHCP)

web server
64.233.169.105

A day in the life… HTTP request/reply

HTTP
TCP
IP
Eth
Phy

❖ web page *finally (!!!)* displayed

HTTP
TCP
IP
Eth
Phy

*router
(runs DHCP)*

web server
64.233.169.105

❖ *HTTP request* sent into TCP socket

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

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

❖ IP datagram containing HTTP reply routed back to client

Chapter 6: Summary

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