

Chapter 5

Link Layer

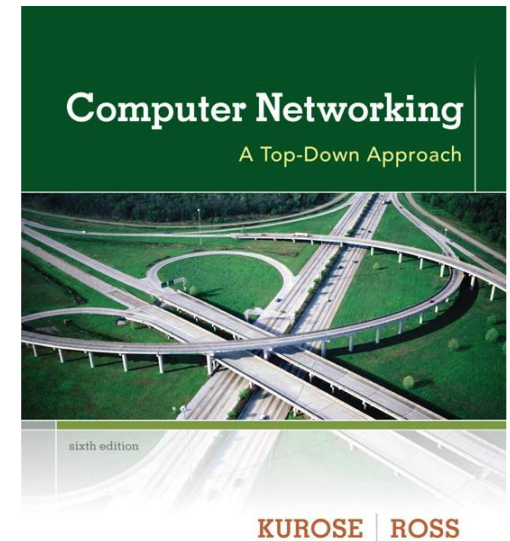
A note on the use of these ppt slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a lot of work on our part. In return for use, we only ask the following:

- ❖ If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- ❖ If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy! JFK/KWR

© All material copyright 1996-2012
J.F Kurose and K.W. Ross, All Rights Reserved



Computer Networking: A Top Down Approach

6th edition

Jim Kurose, Keith Ross

Addison-Wesley

March 2012

Chapter 5: 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

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs

- addressing, ARP
- Ethernet
- switches
- VLANs

5.5 link virtualization: MPLS

5.6 data center networking

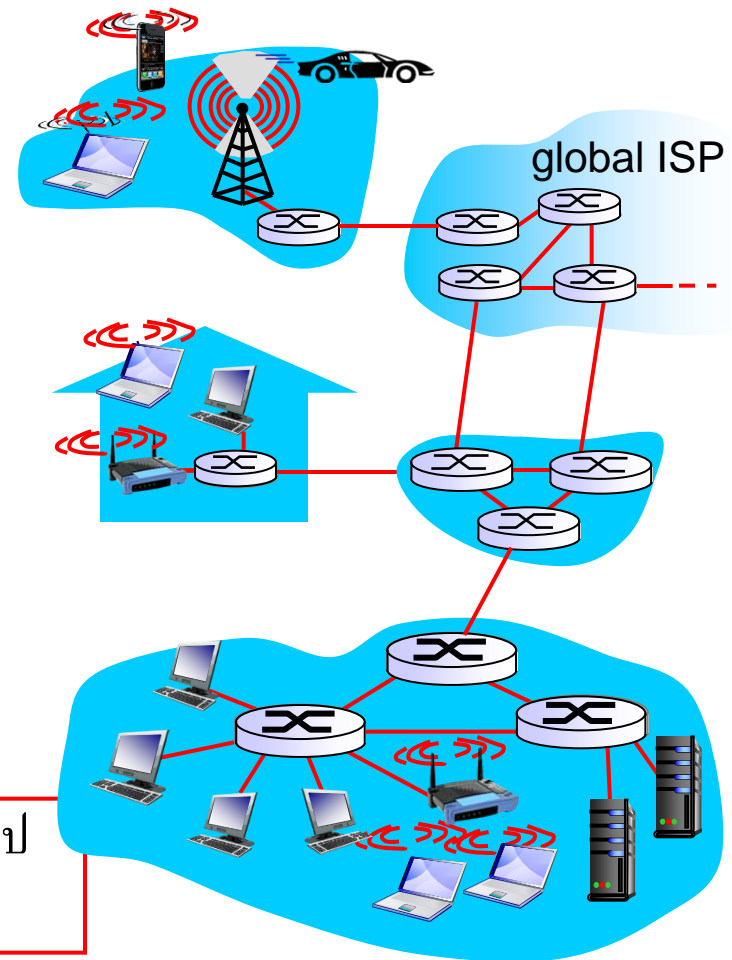
5.7 a day in the life of a web request

Link layer: introduction

terminology:

- ❖ โฮสและเราเตอร์ : **nodes**
- ❖ ช่องการเชื่อมต่อระหว่างโหนด : **links**
 - การเชื่อมต่อแบบมีสาย
 - การเชื่อมต่อแบบไร้สาย
 - LANs
- ❖ layer-2 packet: **frame**, ดาต้าแกรมที่ถูกห่อหุ้ม

data-link layer รับผิดชอบส่งดาต้าแกรมจากโหนดหนึ่งไปยังอีกโหนดผ่านทางการเชื่อมต่อ



Link layer: context

- ❖ ดาต้าแกรมถูกส่งโดยการเชื่อมต่อหลายโปรโตคอลบนการเชื่อมต่อหลายแบบ :
 - เช่นเริ่มส่งข้อมูลด้วย Ethernet แล้วต่อด้วยเฟรมรีเลย์ในระหว่างกลางสุดท้ายไปถึงผู้รับด้วย
- ❖ แต่ละโปรโตคอลการเชื่อมต่อให้บริการหลายบริการ
 - การบริการบางที่อาจไม่มี rdt

transportation analogy:

- ❖ การเดินทางจาก Princeton ไปยัง Lausanne
 - ลิμουซีน: Princeton ไป JFK
 - เครื่องบิน: JFK ไป Geneva
 - รถไฟ: Geneva ไป Lausanne
- ❖ นักท่องเที่ยว = **datagram**
- ❖ ส่วนของการเดินทาง = **communication link**
- ❖ ลักษณะการเดินทาง = **link layer protocol**
- ❖ travel agent = **routing algorithm**

Link layer services

❖ *framing, link access:*

- ห่อหุ้มดาต้าแกรมลงไปในเฟรมโดยใส่ส่วนหัวและหางลงไปด้วย
- channel access if shared medium
- ที่อยู่ “MAC” ถูกใส่ในส่วนหัวของเฟรมเพื่อบอกถึงต้นทางกับปลายทาง
 - ต่างจาก IP address!

❖ *reliable delivery between adjacent nodes*

- เราเรียนวิธีการไปแล้วในบทที่ 3 !
- ใช้น้อยครั้งในการเชื่อมต่อที่มี bit-error น้อย(fiber, some twisted pair)
- wireless links: high error rates
 - Q: why both link-level and end-end reliability?

Link layer services (more)

❖ *flow control:*

- อยู่ระหว่างโหนดผู้ส่งและผู้รับที่อยู่ติดกัน

❖ *error detection:*

- ความเสียหายเกิดจากสัญญาณอ่อนหรือน้อยซ์
- ฝ่ายผู้รับตรวจพบความเสียหาย
 - ส่งสัญญาณไปบอกให้ส่งใหม่อีกครั้งหรือทิ้งเฟรมไป

❖ *error correction:*

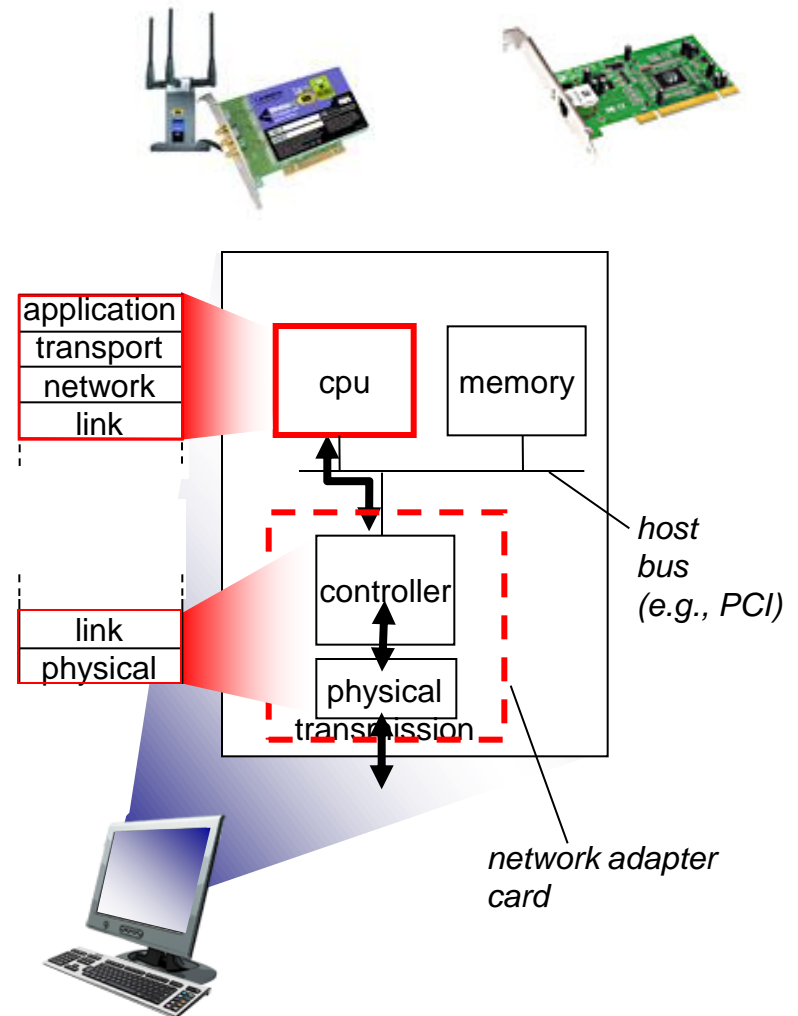
- ผู้รับตรวจและแก้ไขบิตที่เสียหายโดยไม่ต้องรอให้ส่งใหม่อีกครั้ง

❖ *half-duplex and full-duplex*

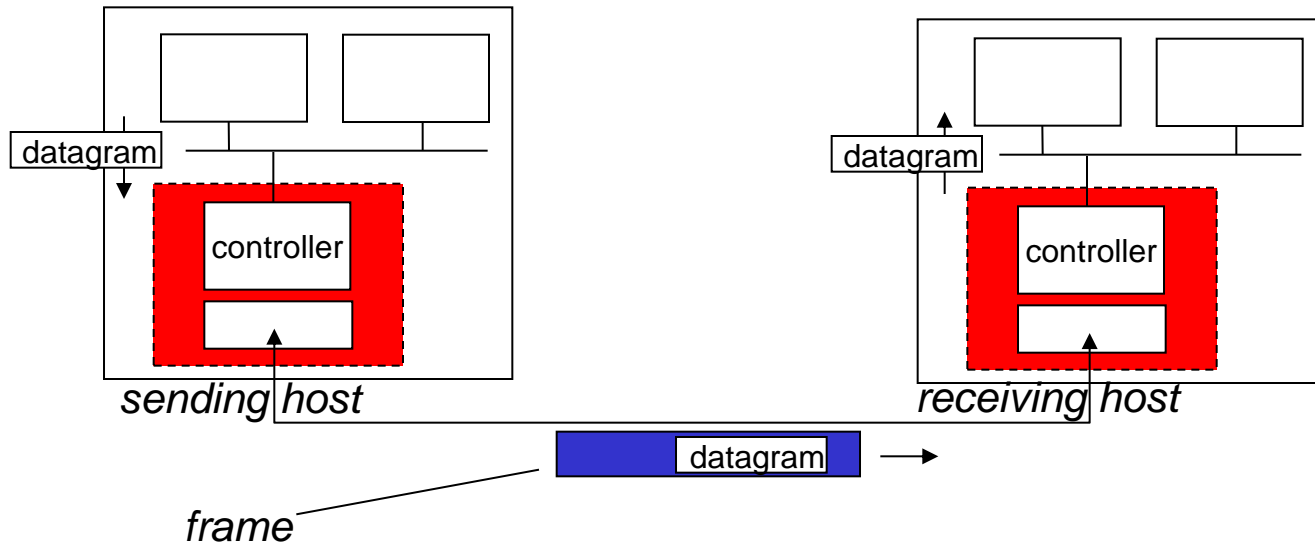
- half duplex โหนดแต่ละฝั่งจะต้องรอให้อีกฝั่งส่งเสร็จก่อนที่จะเริ่มส่งได้แต่ไม่ใช่ในเวลาเดียวกัน

Where is the link layer implemented?

- ❖ link layer ทำงานอยู่ใน “adaptor” (เช่น *network interface card* NIC) หรือบนชิป
 - Ethernet card, 802.11 card; Ethernet chipset
 - implements link, physical layer
- ❖ เชื่อมต่อกับโฮสในระบบแบบบัส
- ❖ เป็นการรวมกันของ ฮาร์ดแวร์ ซอฟต์แวร์ และเฟิร์มแวร์



Adaptors communicating



❖ ส่ง :

- ห่อหุ้มดาต้าแกรมลงในเฟรม
- เพิ่ม error checking bits, rdt, flow control, etc.

❖ ฝั่งรับ

- มองหาข้อผิดพลาด, rdt, flow control, etc
- แยกดาต้าแกรม, ส่งต่อไปยังเลเยอร์ด้านบนฝั่งผู้รับ

Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs

- addressing, ARP
- Ethernet
- switches
- VLANs

5.5 link virtualization: MPLS

5.6 data center networking

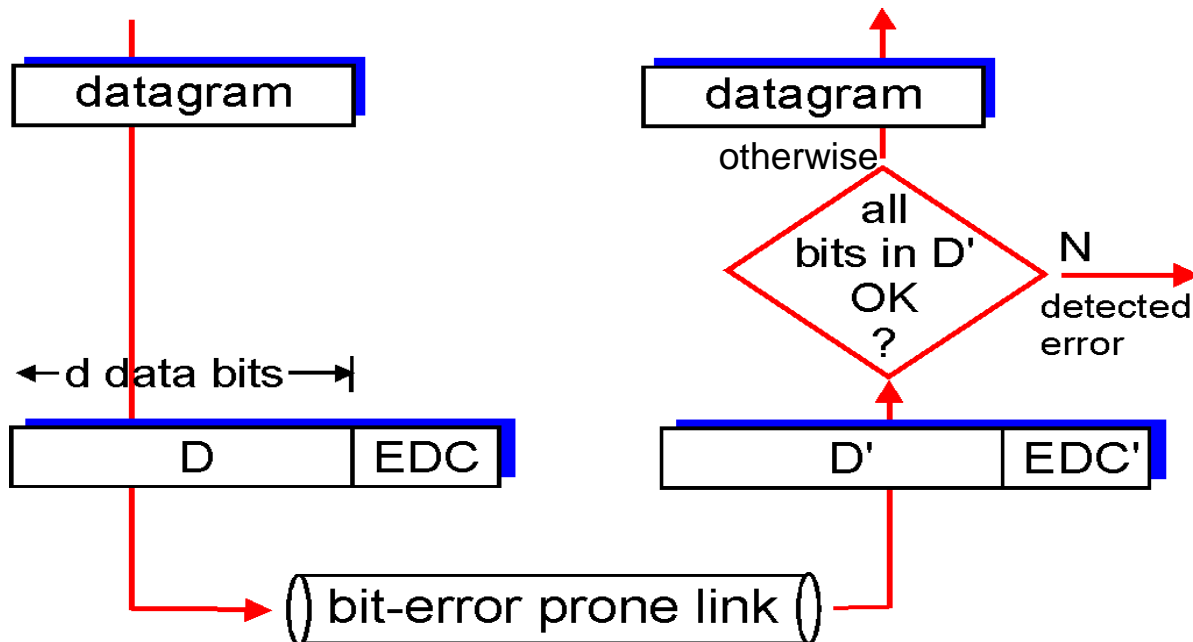
5.7 a day in the life of a web request

Error detection

EDC= Error Detection and Correction bits (redundancy)

D = ข้อมูลถูกปกป้องโดยการตรวจสอบความผิดพลาด อาจรวมถึงข้อมูลส่วนหัว

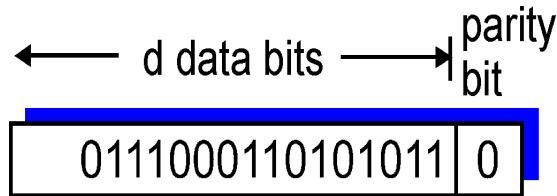
- การตรวจสอบความผิดพลาดเชื่อถือไม่ได้ 100%!
 - โพรโทคอลอาจไม่เจอในบางส่วนของข้อมูลที่เสียหายแต่เกิดขึ้นน้อยมาก
 - ยิ่ง ขนาดEDC ใหญ่เท่าไรการตรวจสอบความผิดพลาดยิ่งดีขึ้นเท่านั้น



Parity checking

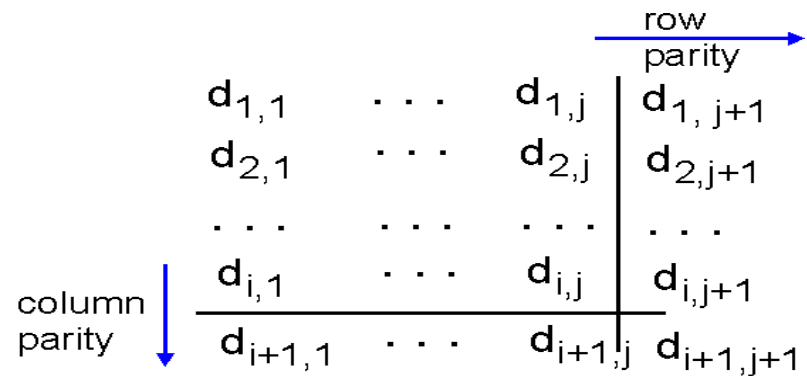
single bit parity:

❖ ตรวจสอบ bit errors



two-dimensional bit parity:

❖ ตรวจสอบและแก้ไข bit errors



1	0	1	0	1	1
1	1	1	1	0	0
0	1	1	1	0	1
0	0	1	0	1	0

no errors

1	0	1	0	1	1
1	1	1	1	0	0
0	1	1	1	0	1
0	0	1	0	1	0

parity error

*correctable
single bit error*

Internet checksum (review)

goal: ตรวจพบข้อผิดพลาด (e.g., flipped bits) ในข้อมูลที่ทำกา^รส่ง (note: used at transport layer *only*)

sender:

- ❖ ทำส่วนย่อยของข้อมูลให้อยู่ในลำดับ 16 บิต
- ❖ checksum: ถูกเพิ่มเข้าไปในส่วนย่อยของข้อมูล
- ❖ ผู้ส่งใส่ค่า checksum ลงไปใน UDP checksum field

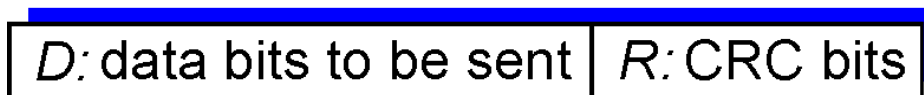
receiver:

- ❖ คำนวน checksum ของชิ้นส่วนที่ได้รับ
- ❖ ตรวจสอบว่า checksum ตรงกับที่ถูกส่งมาหรือไม่ :
 - ไม่ตรง - พบความเสียหาย
 - ตรง - ไม่พบความเสียหายแต่อาจยังมีหรือไม่ ?

Cyclic redundancy check (ซ้ำ)

- ❖ เป็นโค้ดการตรวจสอบความเสียหายที่มีประสิทธิภาพ
- ❖ ดูข้อมูลเป็นบิต, D , แทนเลขฐานสอง
- ❖ choose $r+1$ bit pattern (generator), G
- ❖ goal: choose r CRC bits, R , such that
 - $\langle D, R \rangle$ exactly divisible by G (modulo 2)
 - receiver knows G , divides $\langle D, R \rangle$ by G . If non-zero remainder: error detected!
 - can detect all burst errors less than $r+1$ bits
- ❖ ใช้ทั่วไป (Ethernet, 802.11 WiFi, ATM)

← d bits → ← r bits →



*bit
pattern*

$$D * 2^r \text{ XOR } R$$

*mathematical
formula*

CRC example (ซ้ำ)

want:

$$D \cdot 2^r \text{ XOR } R = nG$$

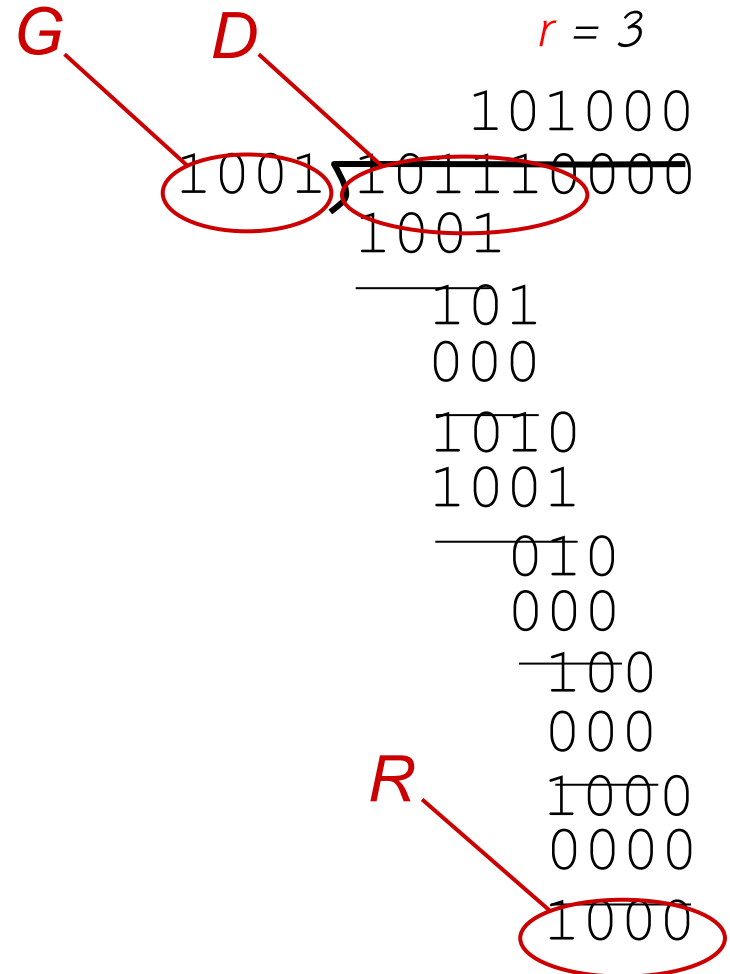
equivalently:

$$D \cdot 2^r = nG \text{ XOR } R$$

equivalently:

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

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



Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs

- addressing, ARP
- Ethernet
- switches
- VLANs

5.5 link virtualization: MPLS

5.6 data center networking

5.7 a day in the life of a web request

Multiple access links, protocols

ลักษณะสองแบบของ “links” :

❖ point-to-point

- PPP สำหรับเชื่อมต่อแบบ dial-up
- point-to-point link ที่เชื่อมต่อระหว่าง Ethernet switch, host

❖ *broadcast (shared wire or medium)*

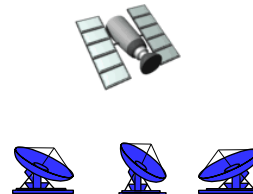
- old-fashioned Ethernet
- upstream HFC
- 802.11 wireless LAN



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

- ❖ แชร์ช่องทางการสื่อสาร single shared broadcast channel
- ❖ สองหรือมากกว่าในการส่งข้อมูลโดยโหนด: เกิดการขัดข้อง interference
 - *collision* ถ้าโหนดมากกว่าสองโหนดส่งข้อมูลพร้อมกัน

multiple access protocol

- ❖ เป็น algorithm แบบกระจายที่ช่วยหาว่าโหนดแชร์ช่องทางการสื่อสารกันอย่างไร i.e., หาว่าเมื่อไรโหนดสามารถส่ง
- ❖ การสื่อสารเพื่อที่จะบอกว่าใครสามารถส่งได้ก็ต้องใช้ช่องทางด้วย
 - ไม่มี out-of-band channel สำหรับช่องทางนี้

multiple access protocol ในอุดมคติ

given: เป็น broadcast channel ของอัตราส่ง R bps

desiderata:

1. เมื่อโหนดต้องการส่ง สามารถส่งได้ที่อัตราการส่ง $= R$
2. เมื่อมี M โหนดต้องการส่ง แต่ละโหนดสามารถส่งได้ R/M
3. fully decentralized:
 - ไม่ต้องการให้มีโหนดพิเศษในการประสานงาน
 - ไม่ต้องการตรวจสอบเวลาและช่อง
4. ง่าย

MAC protocols: มีกี่แบบ

three broad classes:

❖ แบ่งช่องทางเป็นส่วนๆ

- แบ่งช่องทางให้เป็นชิ้นเล็กลง (time slots, frequency, code)
- จัดสรรแต่ละชิ้นส่วนไปให้แต่ละโหนด

❖ ส่งแบบสุ่ม

- ช่องทางไม่มีการแบ่ง ยอมให้เกิดการชนกัน
- ต้องมีการแก้ไขในการชนกัน

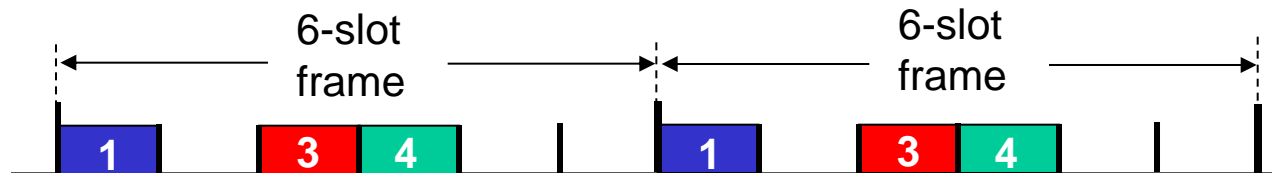
❖ ตาใครตามัน “taking turns”

- โหนดแต่ละโหนดมีตาของตัวเอง แต่บางโหนดสามารถส่งได้มากกว่า

Channel partitioning MAC protocols: TDMA

TDMA: time division multiple access

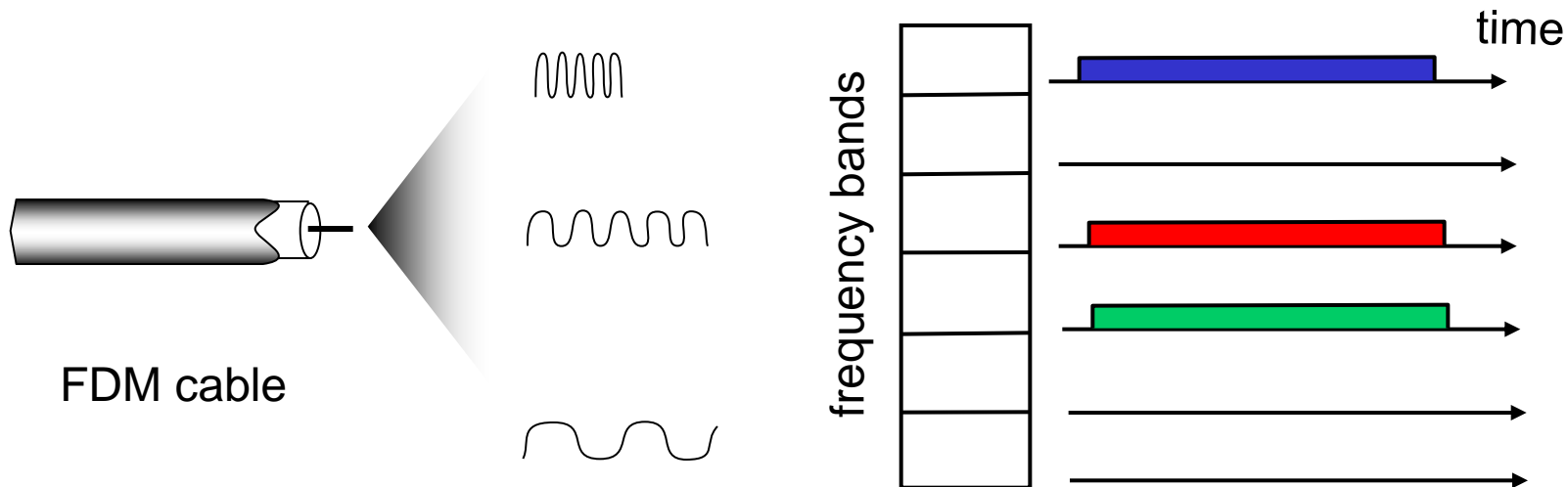
- ❖ ส่งแบบเป็นรอบๆ
- ❖ ในแต่ละรอบมีช่องการส่งจำกัด(length = pkt trans time)
- ❖ ช่องทางที่ไม่มีการส่งก็จะว่าง
- ❖ ตัวอย่าง: 6-station LAN, 1,3,4 มีแพคเกจ, slots 2,5,6 ว่าง



Channel partitioning MAC protocols: FDMA

FDMA: frequency division multiple access

- ❖ ช่องทางการส่งแบบเป็นหลายความถี่
- ❖ แต่ละสถานีถูกกำหนดความถี่ในการส่ง
- ❖ ช่องความถี่ไหนไม่มีการส่งก็ว่าง
- ❖ example: 6-station LAN, 1,3,4 have pkt, frequency bands 2,5,6 idle



Random access protocols

- ❖ เมื่อโหนดมีข้อมูลที่จะส่ง
 - ส่งข้อมูลเต็มอัตราการส่ง
 - no *a priori* coordination among nodes
- ❖ สองหรือมากกว่าอาจมีการชนกันได้
- ❖ random access MAC protocol ข้อกำหนด:
 - มีการตรวจสอบการชนกันอย่างไร
 - เกิดการชนกันแล้วกู้ข้อมูลอย่างไร (e.g., via delayed retransmissions)
- ❖ ตัวอย่างการส่งแบบ random access MAC protocols:
 - slotted ALOHA
 - ALOHA
 - CSMA, CSMA/CD, CSMA/CA

Slotted ALOHA

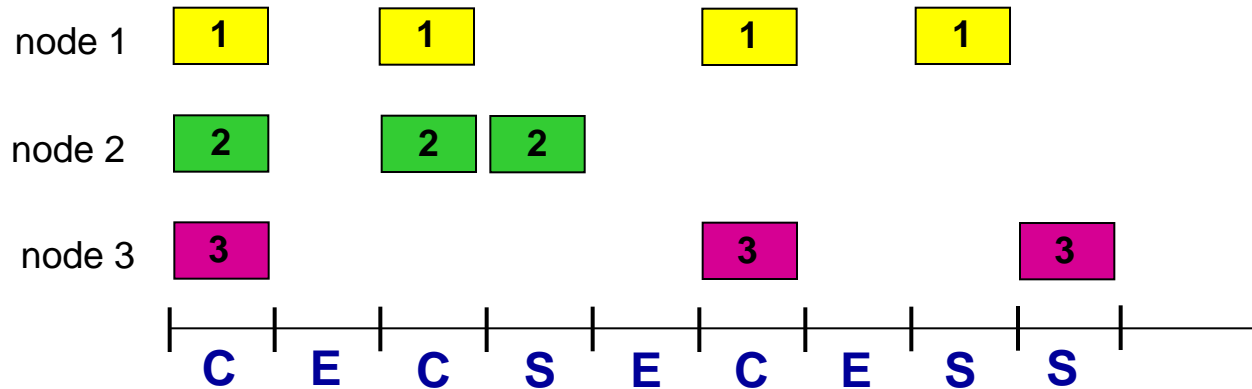
สมมติฐาน:

- ❖ แต่ละเฟรมมีขนาดเท่ากัน
- ❖ เวลาในการส่งถูกแบ่งเป็น slot เท่า ๆ กัน
- ❖ โหนดเริ่มส่งที่ต้น slot เท่านั้น (ไม่เริ่มส่งข้อมูลที่กึ่งกลาง slot ทำให้ slot มีข้อมูลไม่เต็ม)
- ❖ แต่ละโหนดมีการชิงโครโนซ์เวลา (ให้จังหวะเวลา)
- ❖ ถ้ามีสองโหนดหรือมากกว่าส่งข้อมูลใน slot เดียวกัน ทุก ๆ โหนดจะสามารถตรวจพบการชนกันได้

operationการกระทำ:

- ❖ เมื่อโหนดจะต้องส่งเฟรมใหม่ การส่งจะถูกส่งใน slot ถัดไป
 - ถ้าไม่มีการชน: โหนดส่ง frame ใหม่ใน slot นั้นได้สำเร็จ
 - ถ้ามีการชน: โหนดต้องส่ง frame ใหม่ในอีกครั้งใน แต่ละ slot ถัดไปด้วยความน่าจะเป็น p จนกว่าจะส่งสำเร็จ

Slotted ALOHA



ข้อดี:

- ❖ แต่ละโหนดที่ส่งอยู่สามารถส่งได้ในอัตราที่การส่งเต็มที่
- ❖ highly decentralized: เฉพาะช่องที่ส่งในโหนดเท่านั้นที่ต้องการซิงค์
- ❖ ง่าย

ข้อเสีย:

- ❖ เกิดการชนกัน, เกิดการรอในการส่ง
- ❖ มีช่องว่าง
- ❖ โหนดอาจจะสามารถตรวจสอบการชนกันได้ช้ากว่าเวลาที่ส่งแพ็คเกจ
- ❖ มีการตรวจสอบเวลา

Slotted ALOHA: ประสิทธิภาพ

efficiency ประสิทธิภาพ: ในระยะยาวอัตราการส่งที่สำเร็จ (many nodes, all with many frames to send)

- ❖ สมมติว่า N Node มีหลายๆเฟรมความเป็นไปได้ของแต่ละการส่งเท่ากับ 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:

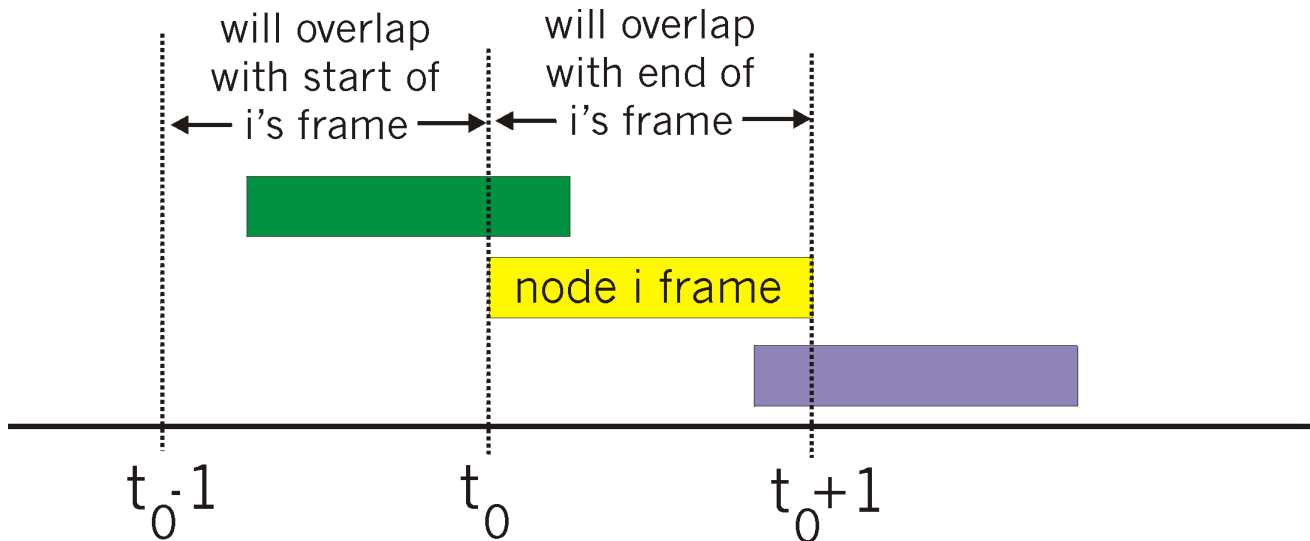
$$\text{max efficiency} = 1/e = .37$$

at best: ช่องทางการส่งดีที่สุดมีแค่ 37 %



Pure (unslotted) ALOHA

- ❖ unslotted Aloha: ง่ายกว่า ไม่มีการซิงโครไนซ์ simpler, no synchronization
- ❖ เมื่อเฟรมแรกมาถึง
 - ส่งทันที
- ❖ ความน่าจะเป็นในการชนการมีมากขึ้น :
 - ส่งข้อมูลที่ t_0 ชนกับข้อมูลอื่นที่ $[t_0-1, t_0+1]$



Pure ALOHA efficiency

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

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

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

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

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

... choosing optimum p and then letting n

$\rightarrow \infty$

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

even worse than slotted Aloha!

CSMA (carrier sense multiple access)

CSMA: ตรวจสอบก่อนการส่ง:

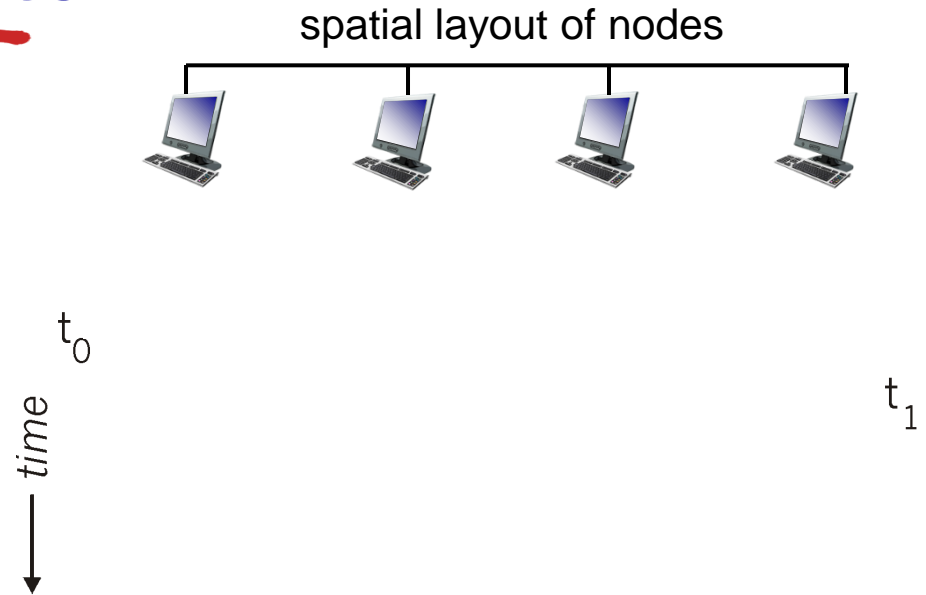
ถ้าตรวจสอบช่องทางการส่งว่าง ก็จะส่งเฟรมไปทั้งเฟรม

❖ ถ้าไม่ว่าง ก็จะรอก่อนค่อยส่ง

❖ อย่าเข้าไปขัดคนอื่น !

การชนกันของ CSMA

- ❖ การชนกันยังอาจเกิดขึ้นอยู่: ดีเลย์ในการส่งข้อมูลหมายถึงสองโหนดไม่ได้รับรู้การส่งของแต่ละฝั่ง
- ❖ การชนกัน: สูญเสียเวลาในการส่งแพคเกจทั้งหมด
 - ระยะทาง และดีเลย์มีผลต่อการชนกันของข้อมูล



CSMA/CD (collision detection)

CSMA/CD: มีการตรวจสอบเหมือน CSMA

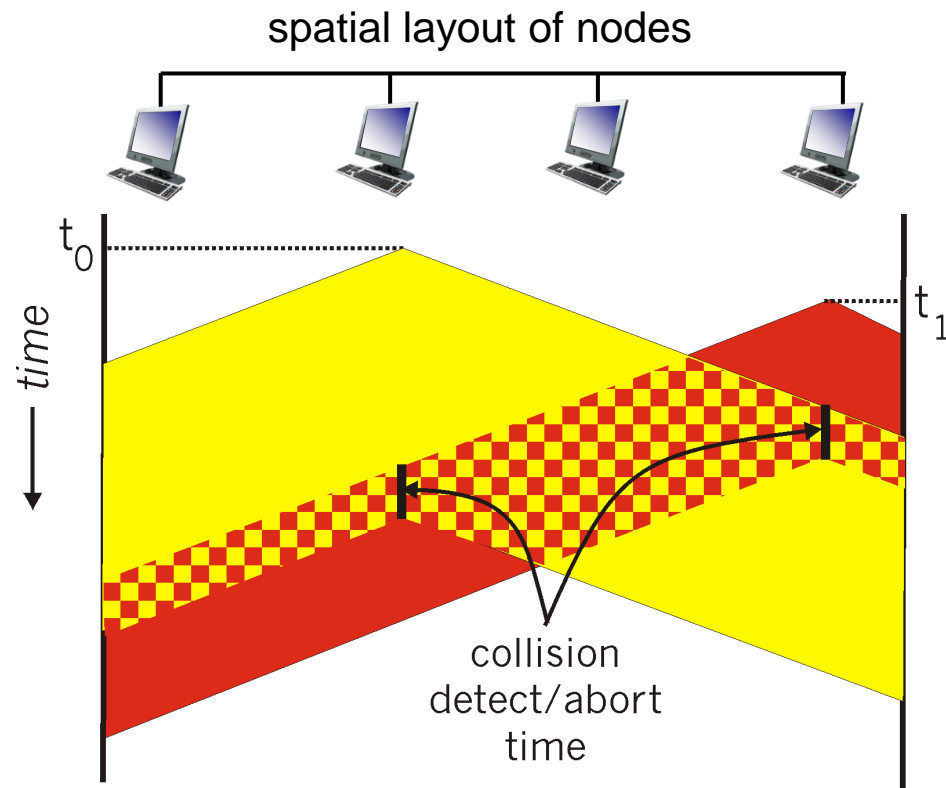
- พบการชนกันในเวลาอันสั้น
- ยกเลิกการส่ง ลดเวลาที่สูญเสียไป

❖ collision detection:

- ง่ายในการส่งแบบสาย: ตรวจสอบความเข้มสัญญาณ เปรียบเทียบการส่ง สัญญาณที่ได้รับ
- ยากในการส่งไร้สาย: การได้รับความแรงของสัญญาณขึ้นอยู่กับความแรงของการส่ง

❖ ในภาษาพูด: เป็นการสนทนาแบบสุภาพ

CSMA/CD (collision detection)



Ethernet CSMA/CD algorithm

1. NIC ได้รับดาต้าแกรมจาก network layer, สร้างเฟรม
2. ถ้า NIC ตรวจพบช่องทางการส่งที่ว่าง ก็จะเริ่มการส่ง ถ้า NIC พบว่าไม่ว่างก็จะรอจนกว่าจะว่างค่อยทำการส่ง
3. ถ้า NIC ส่งทั้งเฟรมไปโดยไม่พบว่าเจอการส่งอีกอัน NIC ถือว่าส่งได้สำเร็จ
4. ถ้า NIC พบการส่งอื่นในขณะที่ทำการส่งก็จะยกเลิกการส่งนั้นแล้วส่งสัญญาณไปแจ้งว่าชนกัน
5. หลังจากยกเลิก NIC เข้าสู่ *binary (exponential) backoff*:
 - after m th collision, NIC chooses K at random from $\{0, 1, 2, \dots, 2^m - 1\}$. NIC waits $K \cdot 512$ bit times, returns to Step 2
 - longer backoff interval with more collisions

CSMA/CD efficiency (ซ้ำ)

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

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

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

“Taking turns” MAC protocols

แบ่งช่องทางการส่ง MAC protocols:

- แบ่งได้อย่างมีประสิทธิภาพและยุติธรรมเมื่อมีอัตราการใช้สูง
- ไม่มีประสิทธิภาพเมื่ออัตราการใช้ต่ำ: เกิดความล่าช้าในการส่ง, ใช้อัตราการส่งทั้งหมด แม้ว่าจะมีโหนดที่ใช้อยู่เพียงโหนดเดียว

แบบสุ่ม MAC protocols

- ทำงานได้ดีเมื่อมีการใช้งานต่ำ : โหนดสามารถใช้ได้เต็มช่องทาง
- เมื่อมีการใช้งานสูง : เกิดการชนการ

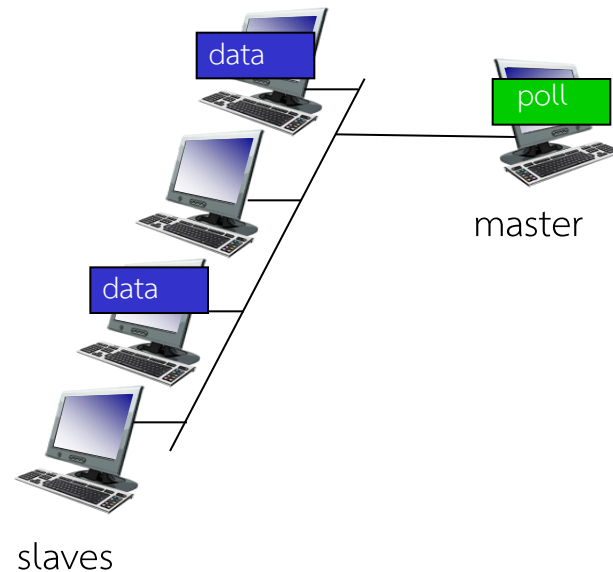
“taking turns” protocols

หาวิธีที่ดีที่สุดจากสองแบบ !

“Taking turns” MAC protocols

polling:

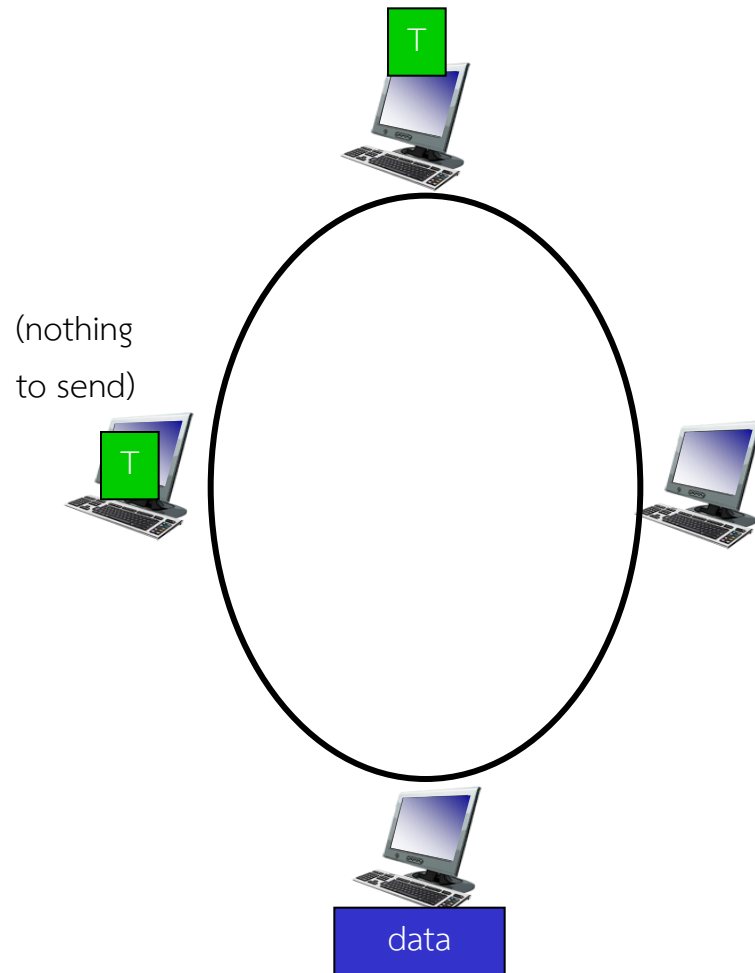
- ❖ โหนดตรงกลางจะเรียกโหนดลูกว่าถึงเวลาส่ง
- ❖ ส่วนมากใช้กับอุปกรณ์ลูกที่ความสามารถต่ำ
- ❖ ข้อเสีย:
 - ต้องมีการ polling
 - เกิดความล่าช้า
 - ถ้ามาสเตอร์เสียก็จะทำให้ส่งไม่ได้เลย



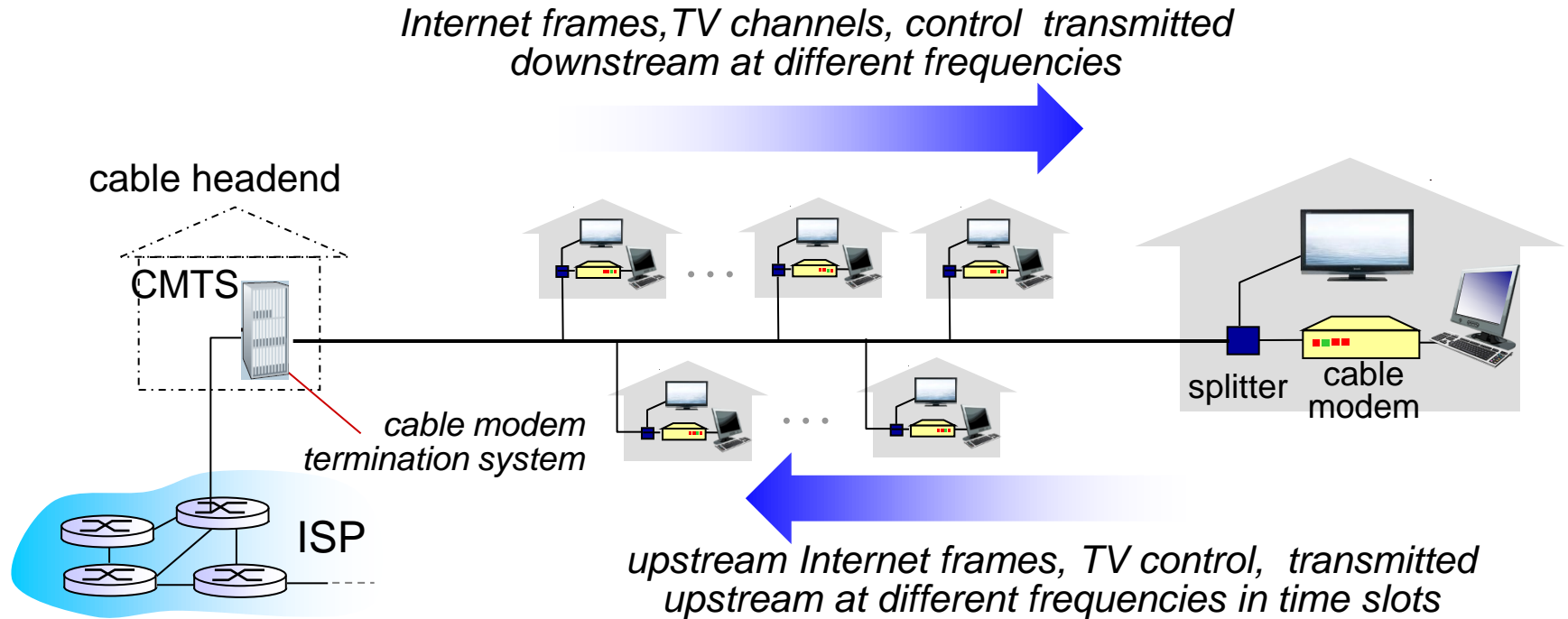
“Taking turns” MAC protocols

token passing:

- ❖ ควบคุมการส่งผ่าน token จากโหนดหนึ่งไปยังอีกโหนดหนึ่ง
- ❖ ข้อความ Token
- ❖ ข้อเสีย:
 - ต้องส่ง token
 - เกิดความล่าช้า
 - ถ้าเกิดความเสียบตรงโหนดที่มี token

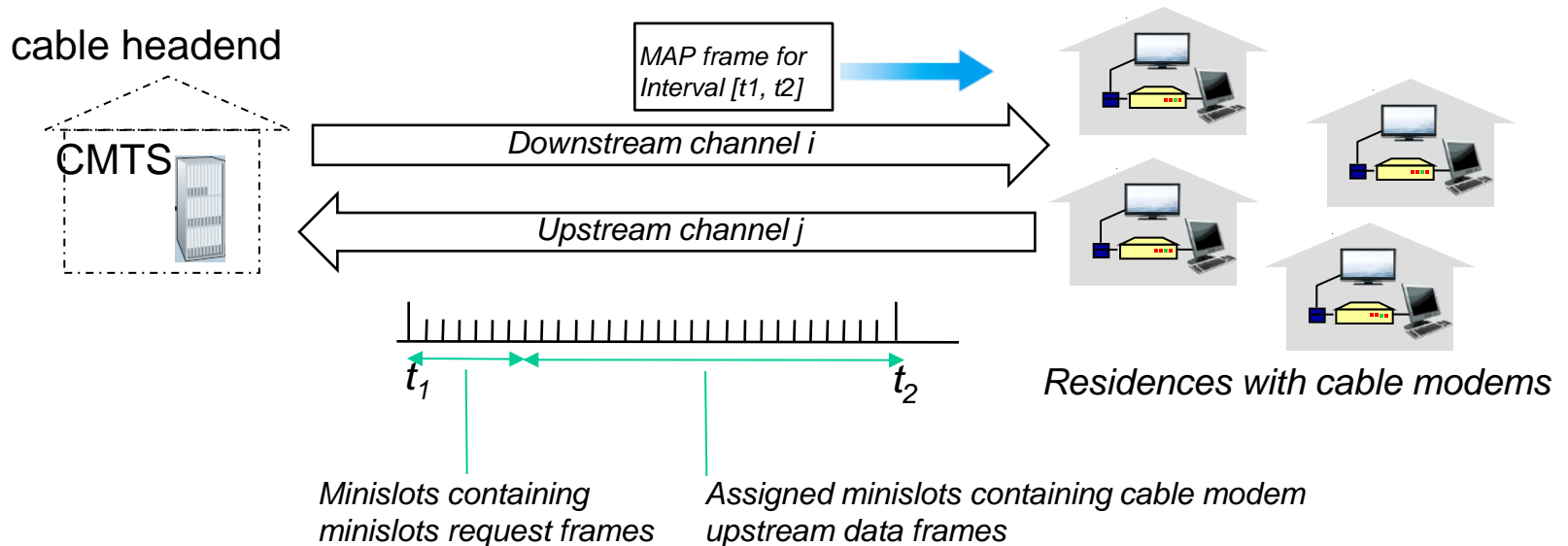


Cable access network (ချိာမ်)



- ❖ *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 (ข้อ ๗)



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

Summary of MAC protocols

- ❖ *channel partitioning*, แบ่งโดยเวลา แบ่งโดยความถี่ แบ่งโดยโค้ด
 - Time Division, Frequency Division
- ❖ *random access* (เป็นการสุ่ม),
 - ALOHA, S-ALOHA, CSMA, CSMA/CD
 - carrier sensing: ง่ายในบางเทคโนโลยี(แบบมีสาย), ยากในแบบอื่น(ไร้สาย)
 - CSMA/CD ถูกใช้ใน Ethernet
 - CSMA/CA ถูกใช้ใน 802.11
- ❖ *taking turns*
 - polling จากส่วนกลาง, มีการส่ง token
 - bluetooth, FDDI, token ring

Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs

- addressing, ARP
- Ethernet
- switches
- VLANs

5.5 link virtualization: MPLS

5.6 data center networking

5.7 a day in the life of a web
request

MAC addresses and ARP

32-bit IP address:

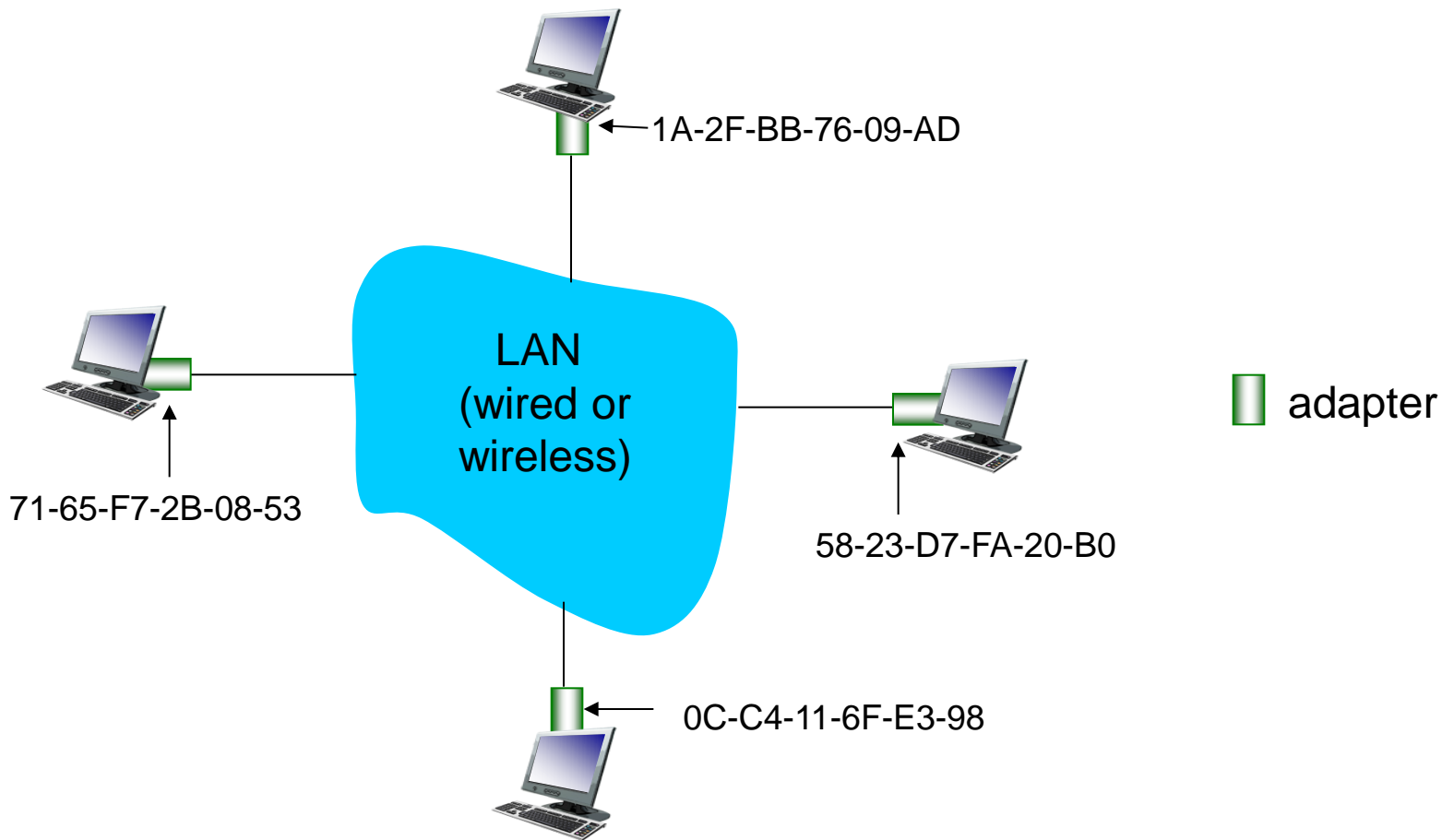
- เป็นที่อยู่ (ของ *network-layer*) สำหรับแต่ละ interface (หรือ แต่ละ LAN card)
- ใช้สำหรับส่งต่อข้อมูลใน layer 3 (network layer)

MAC (or LAN or physical or Ethernet) address:

- การใช้งาน: ใช้ทั่วไปเพื่อที่จะรับเฟรมจากอินทอเฟสหนึ่งไปยังอีกอินทอเฟสที่เชื่อมต่อกันทางกายภาพ
- 48 bit MAC address (สำหรับแลนส่วนมาก) ถูกเขียนลงในรอมของ NIC , บางครั้งสามารถเซตได้จากตัวซอฟต์แวร์
- e.g.: 1A-2F-BB-76-09-AD
 /
 hexadecimal (base 16) ข้อสังเกต
 (แต่ละหมายเลขแสดงถึง 4 bits)

LAN addresses and ARP

แต่ละแลนคอมพิวเตอร์มีที่อยู่ของตัวเอง



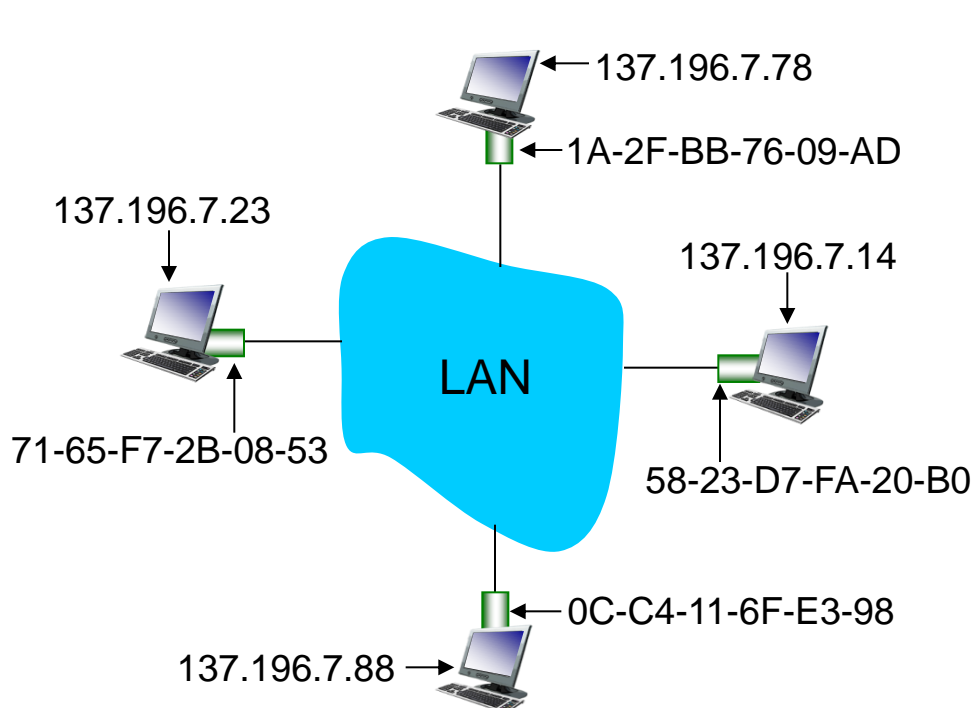
LAN addresses (more)

- ❖ MAC address ถูกกำหนดโดย by IEEE
- ❖ บริษัทผู้ผลิตซื้อชุดของ MAC address เพื่อรับประกันว่าจะไม่ซ้ำกัน
- ❖ analogy:
 - MAC address: เหมือนกับเลขบัตรประชาชน
 - IP address: ที่อยู่ตามบ้าน
- ❖ MAC flat address → ไปที่ไหนก็ได้
 - สามารถย้ายที่ได้
- ❖ IP hierarchical address ไม่สามารถเคลื่อนย้ายได้
 - เพราะว่าที่อยู่ไอพีแอดเดรสจะต้องขึ้นอยู่กับชั้นเน็ตนั้นๆ

ARP: address resolution protocol

คำถาม: เราจะรู้ MAC address ได้อย่างไรด้วย IP Address ?

ARP table: แต่ละโหนดไอพี(host, router) มีตาราง



- IP/MAC address จับคู่กันสำหรับบางโหนด:

< IP address; MAC address; TTL >

- TTL (Time To Live): เวลาก่อนที่จะหมดอายุในการจำเลข mac address กับ ip (โดยทั่วไป 20 min)

ARP protocol: same LAN

❖ A ต้องการส่งดาต้าแกรมไปหา B

- B's MAC address ไม่อยู่ใน A's ARP table.

❖ A broadcasts ARP query packet, containing B's IP address

- dest MAC address = FF-FF-FF-FF-FF-FF
- ทุกๆ โหนดในวงแลนจะได้รับ

❖ B รับแพคเกจ ARP , ตอบกลับไปยัง A ด้วย (B's) MAC address

- ข้อมูลส่งไปยัง A's MAC address (unicast)

❖ แคชจะเก็บตารางไว้จนกว่าจะหมดเวลา

- soft state: ข้อมูลจะค่อยๆหมดเวลาลงไปเรื่อยๆจนจะได้รับใหม่อีกรอบ
information that times out (goes away) unless refreshed

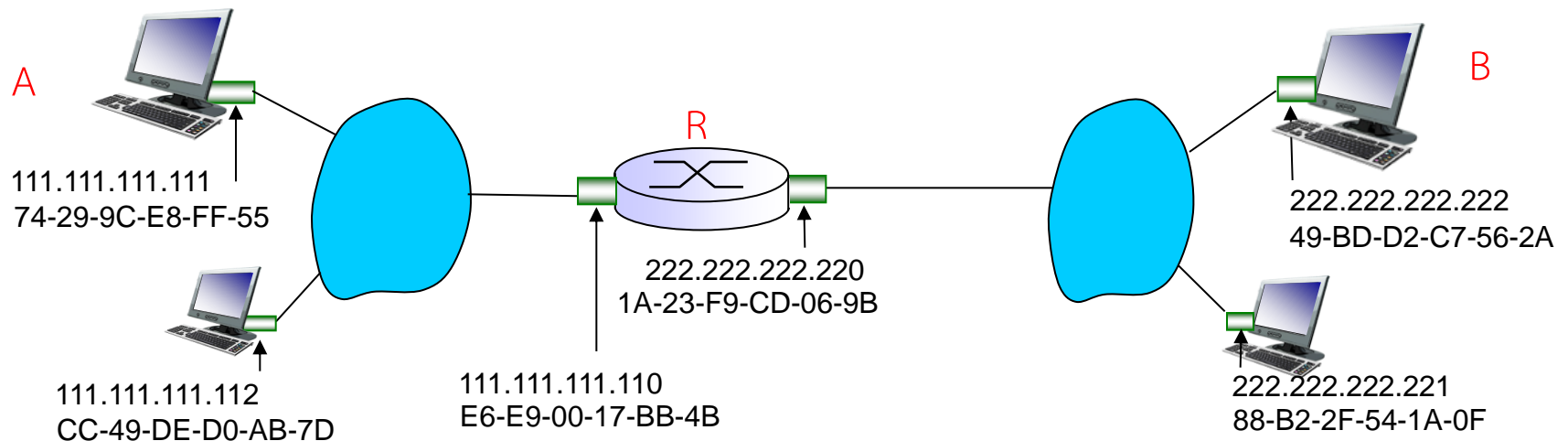
❖ ARP เป็นแบบ “plug-and-play” :

- โหนดจะสร้างตาราง arp ของตัวเองโดนแอดมินไม่ต้องจัดการ

Addressing: routing to another LAN

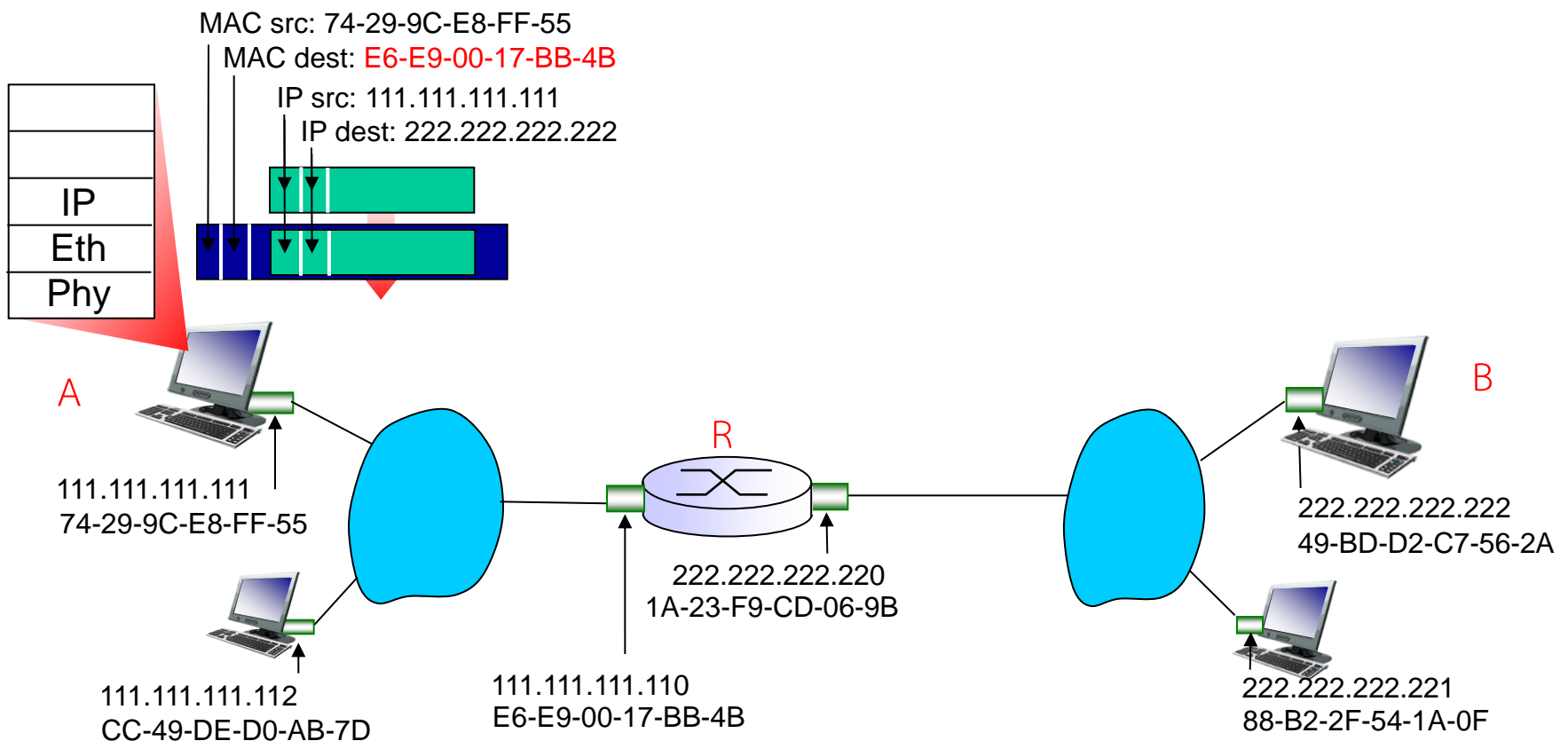
walkthrough: ส่งข้อมูลจาก A ไปยัง B ผ่านทาง R

- focus on addressing – at IP (datagram) and MAC layer (frame)
- สมมติว่า A รู้ B' s IP address
- สมมติว่า A รู้ IP address ของ router แรก, R (ได้อย่างไร?)
- สมมติว่า A รู้ R' s MAC address (ได้อย่างไร?)



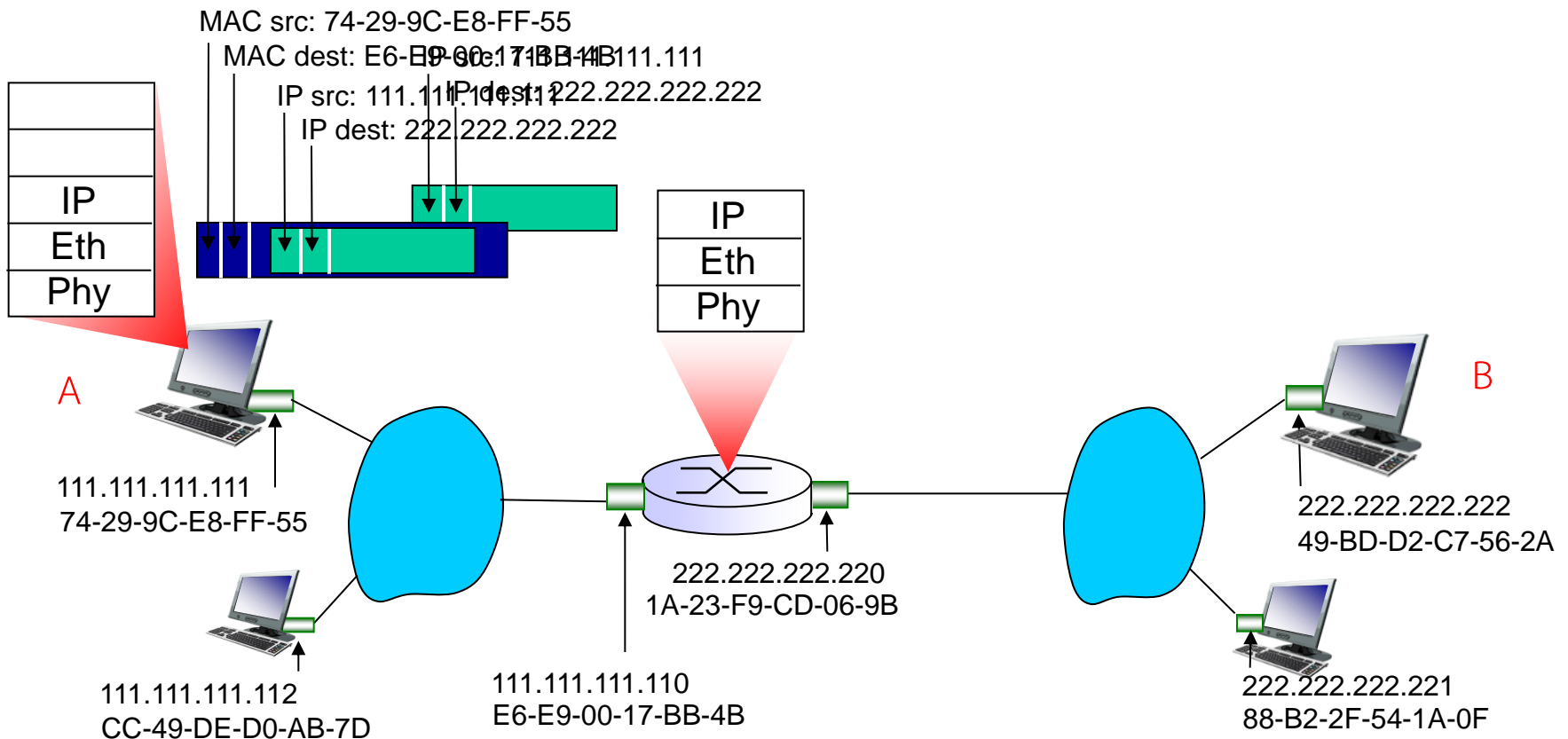
Addressing: routing to another LAN

- ❖ A สร้างข้อมูลด้วย source A, destination B
- ❖ A สร้างเฟรมการเชื่อมต่อด้วย R's MAC address เป็นปลายทาง, เฟรมจะมีข้อมูล A-to-B IP datagram



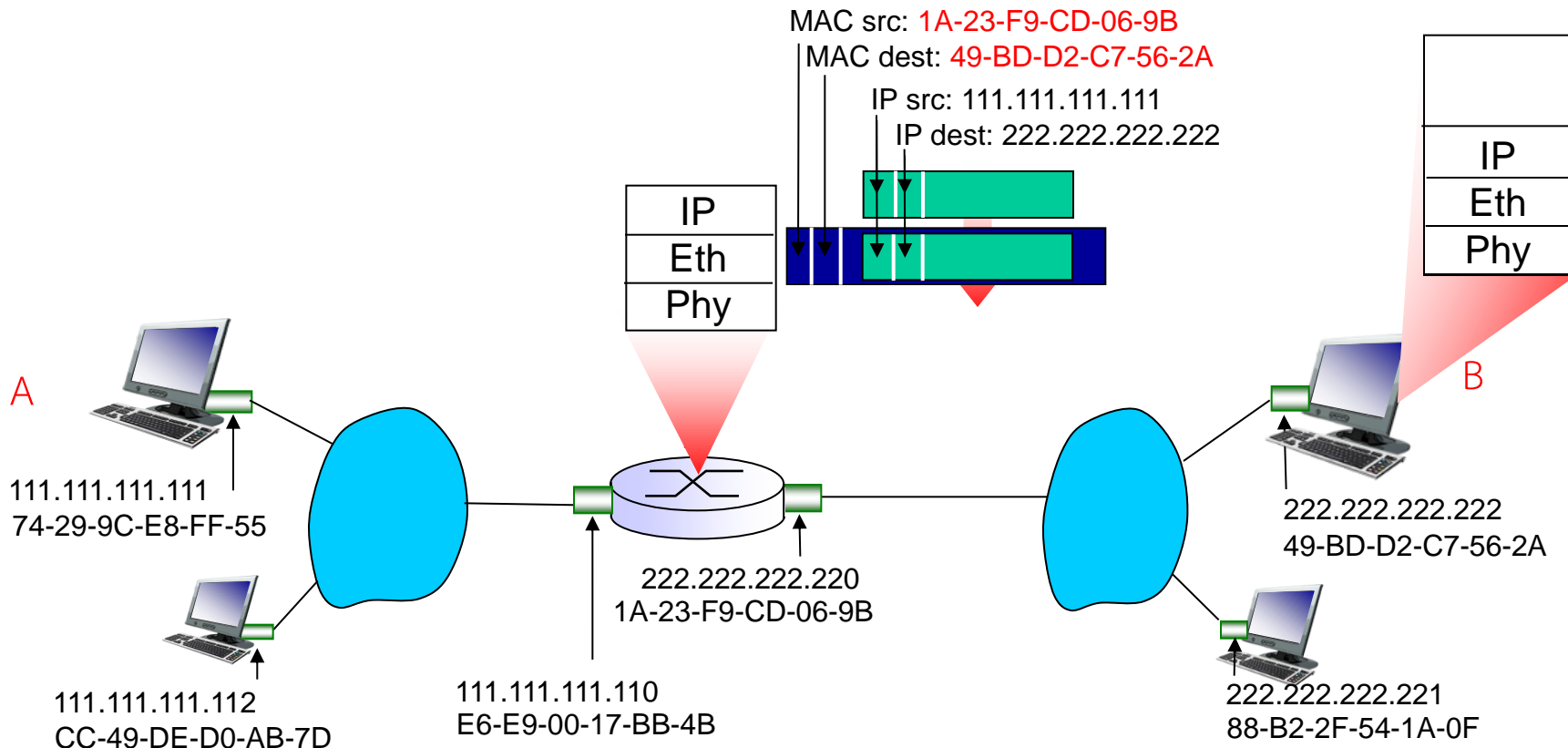
Addressing: routing to another LAN

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



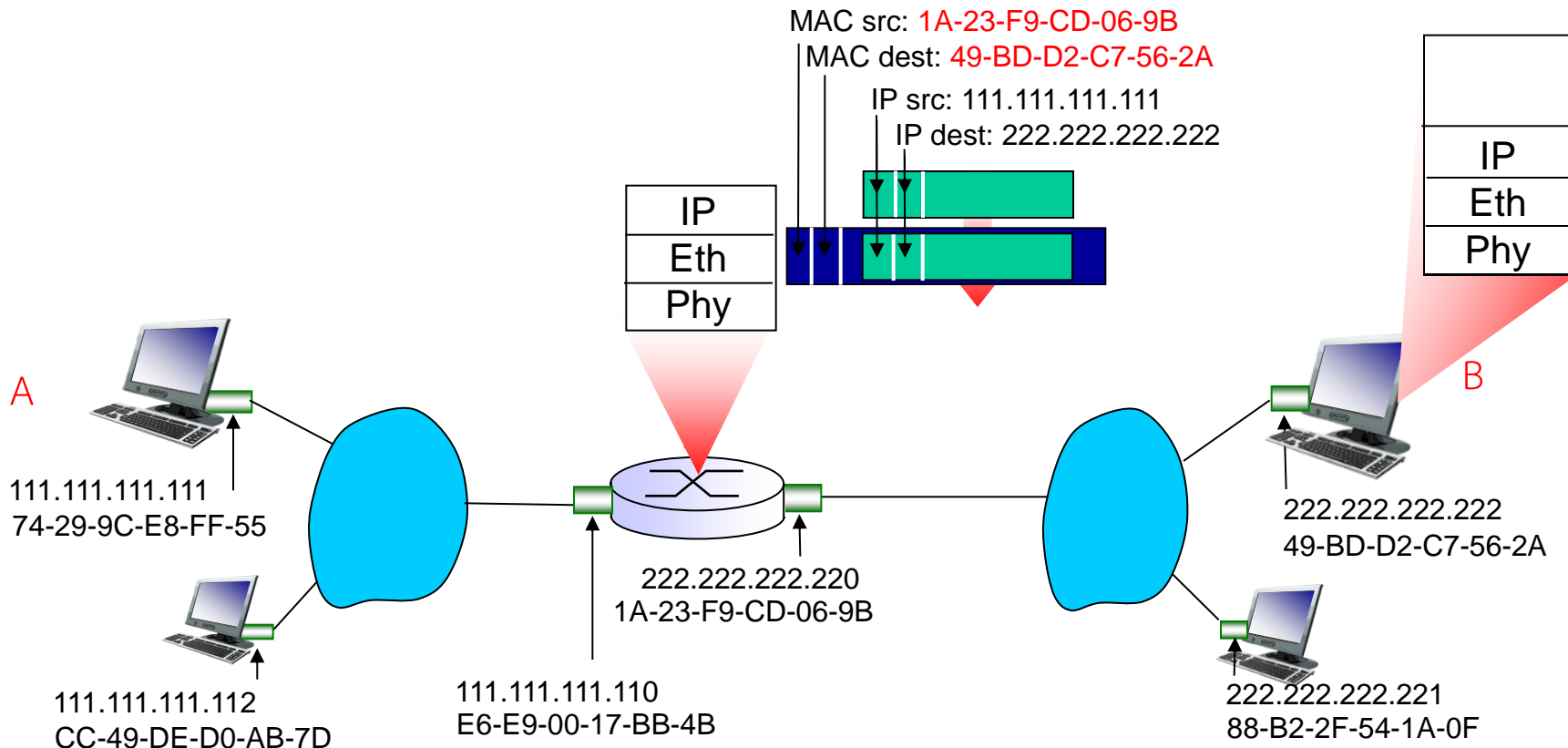
Addressing: routing to another LAN

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



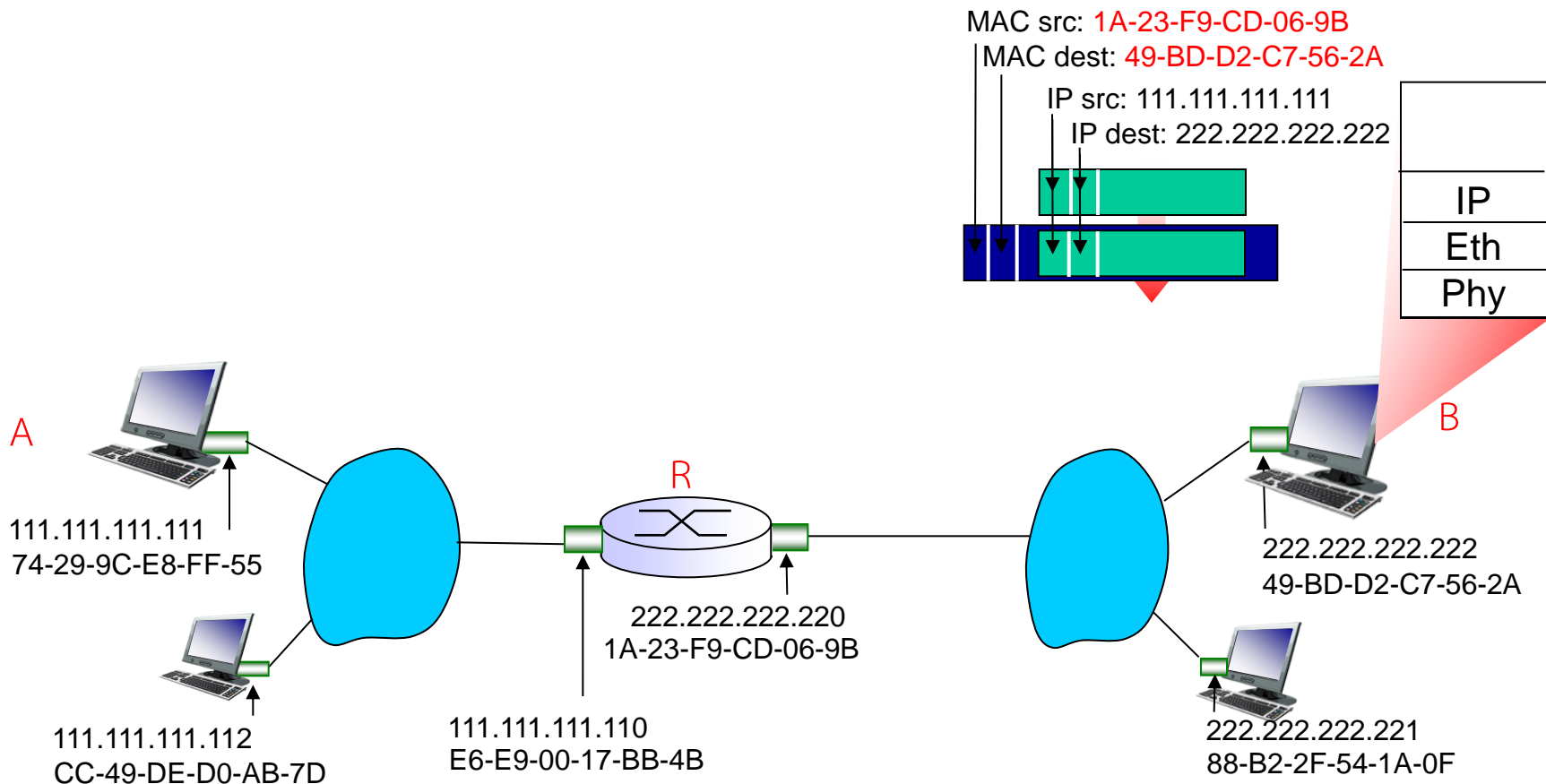
Addressing: routing to another LAN

- ❖ 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



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



Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs

- addressing, ARP
- Ethernet
- switches
- VLANs

5.5 link virtualization: MPLS

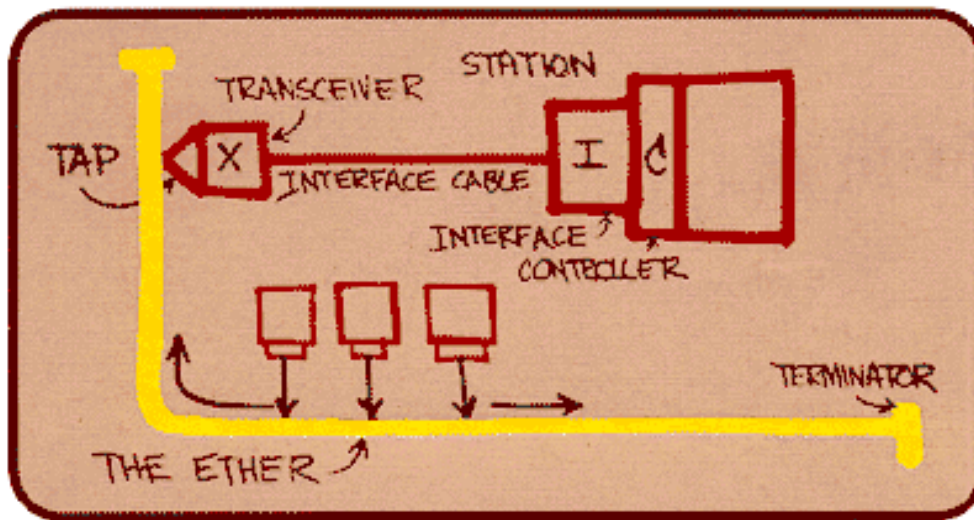
5.6 data center networking

5.7 a day in the life of a web
request

Ethernet

“เป็นที่นิยม” ในเทคโนโลยีระบบสาย:

- ❖ ราคาถูก
- ❖ ใช้เป็นวงกว้างในตอนเริ่มต้น
- ❖ ง่ายและถูกกว่า token LANs and ATM
- ❖ มีความเร็วสูง : 10 Mbps – 10 Gbps



Metcalfe's Ethernet sketch

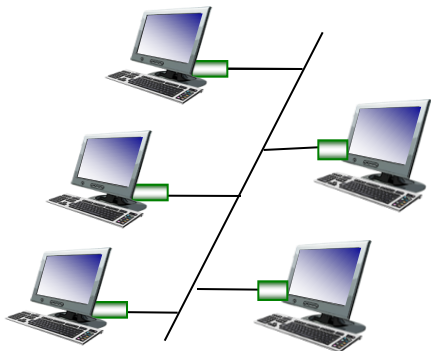
Ethernet: physical topology

❖ *bus*: แพร่หลายในช่วงกลางทศวรรษ 90

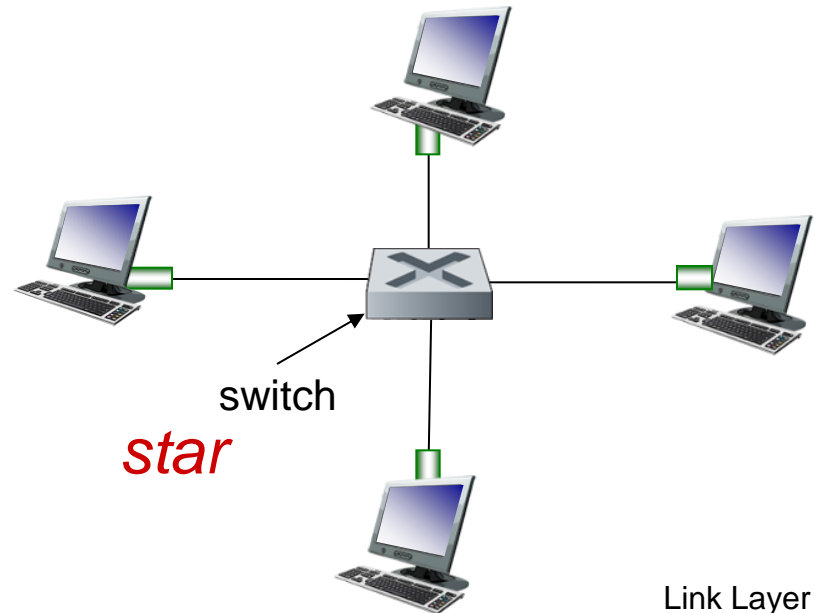
- โหนดทั้งหมดจะมีโดเมนเดียวกันซึ่งชนกันได้

❖ *star*: ปัจจุบัน

- มี *switch* อยู่ตรงกลาง
- แต่ละโหนดมีโปรโตคอลเป็นของตัวเองซึ่งลดการชนกันได้พอสมควร



bus: coaxial cable



Ethernet frame structure (ข้าม)

ลักษณะของแพคเกจที่ถูกห่อหุ้มลงไปใน 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) (ข้าม)

❖ *addresses*: มี 6 bytes, destination MAC addresses

- ถ้าอแดปเตอร์ได้รับเฟรมที่มีปลายทางตรงกันเช่น arp packet โหนดทุกโหนดก็จะได้รับข้อความมา
- ถ้าไม่เกี่ยวข้องก็ทิ้งไป

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

❖ *CRC*: cyclic redundancy ตรวจสอบที่ปลายทาง

- พบความเสียหาย : แพคเกจจะถูกทิ้งไป



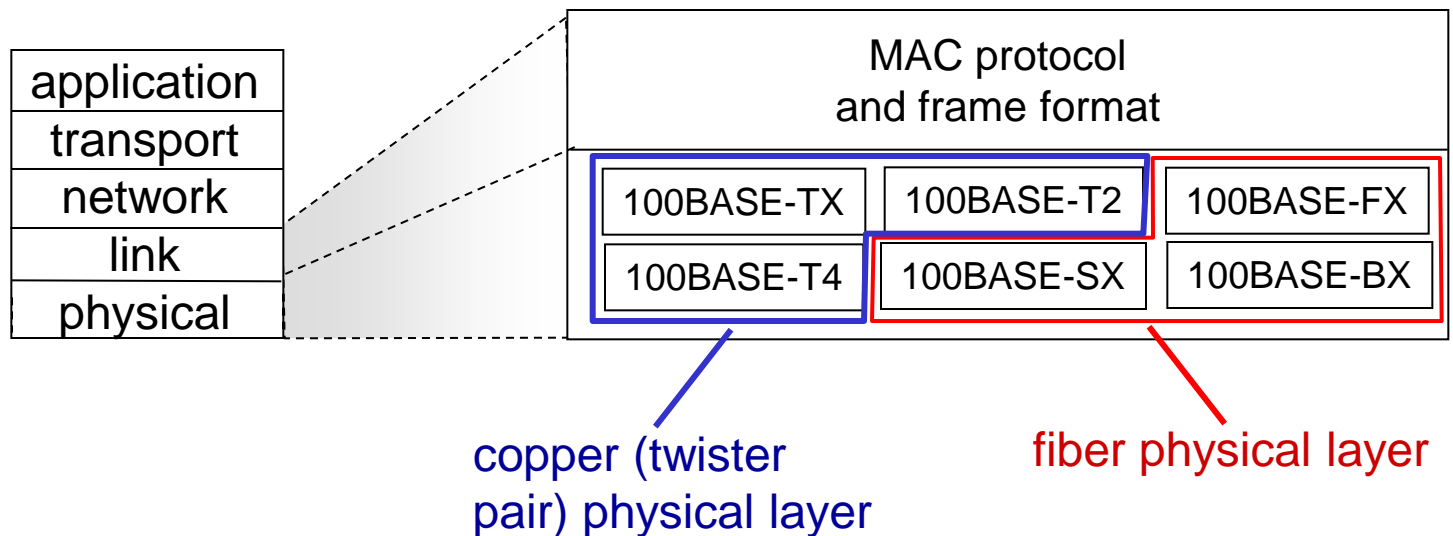
Ethernet: unreliable, connectionless

- ❖ *connectionless*: n ไม่มีการเชื่อมต่อ handshaking ระหว่างผู้ส่งและผู้รับ
sending and receiving NICs
- ❖ *unreliable*: ผู้รับ ผู้ส่งไม่มีการส่ง ack nak
 - 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

❖ มีหลายความเร็ว

- มีโปรโตคอลที่เหมือนกัน
- ความเร็วแตกต่างกันไป: 2 Mbps, 10 Mbps, 100 Mbps, 1Gbps, 10G bps
- มีลักษณะแตกต่างกันไป : fiber, cable



Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs

- addressing, ARP
- Ethernet
- switches
- VLANs

5.5 link virtualization: MPLS

5.6 data center networking

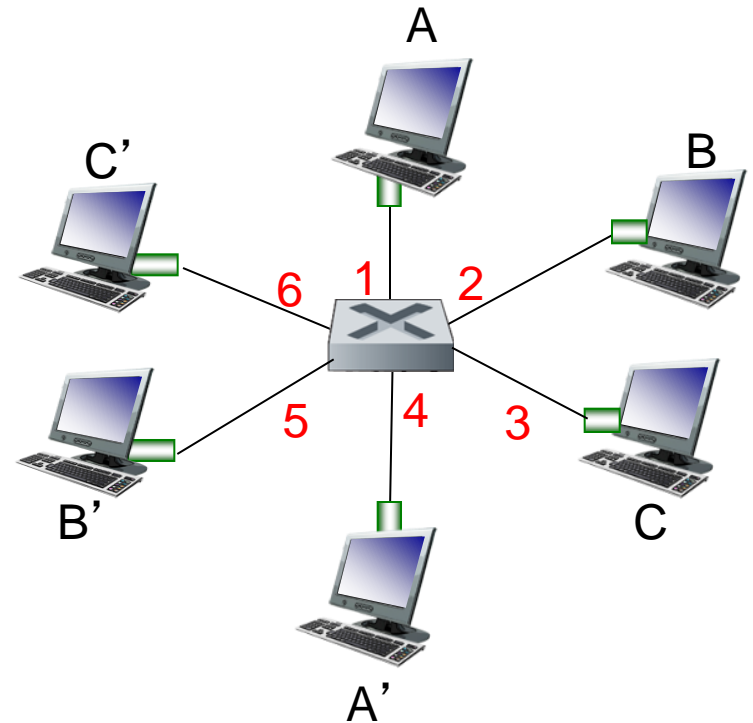
5.7 a day in the life of a web request

Ethernet switch

- ❖ link-layer device: บทบาทและหน้าที่หลัก
 - รับและส่งต่อข้อมูล
 - ตรวจสอบ frame 's MAC address ที่ได้รับ , เลือกส่งข้อมูลไป หนึ่งที่หรือมากกว่าที่ลิงค์ขาออก เมื่อข้อมูลนั้นมีการส่งต่อ
- ❖ *transparent*
 - เครื่องปลายทางไม่รู้ว่ามี switch อยู่ระหว่างทาง
- ❖ *plug-and-play, self-learning*
 - switch ไม่ต้องการการตั้งค่า

Switch: *multiple* simultaneous transmissions

- ❖ แต่ละเครื่องมีการเชื่อมต่อของตัวเอง
hosts have dedicated, direct connection to switch
- ❖ switch จะสำรองข้อมูลไว้
- ❖ ทุกเครื่องจะมีโปรโตคอลของตัวเองในการเชื่อมต่อ ข้อมูลจะไม่ชนกัน; full duplex
- ❖ *switching*: A-to-A' และ B-to-B' สามารถส่งได้พร้อมกันโดยไม่ชนกัน



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

Switch forwarding table

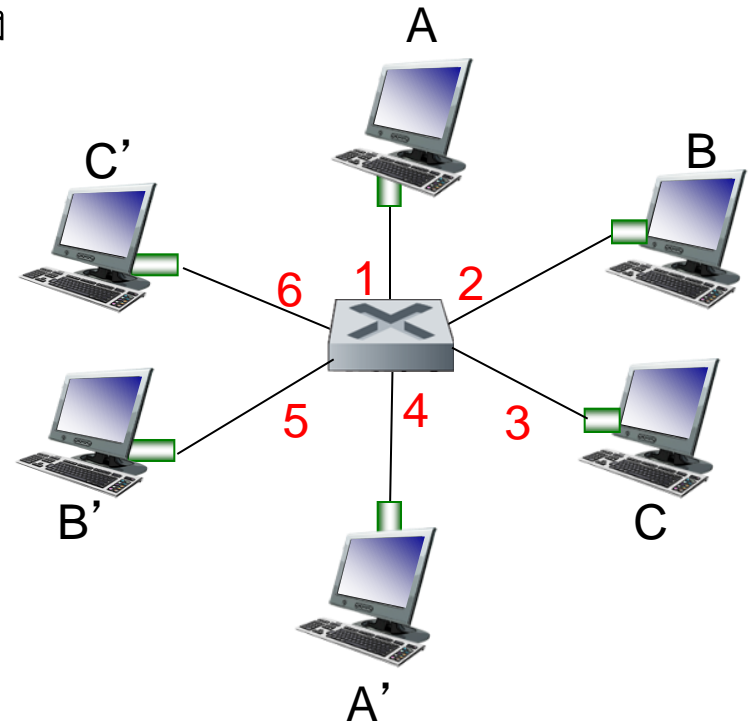
Q: switch จะรู้ได้อย่างไรว่า A จะส่งไป B ผ่านทางอินเทอร์เฟซ 4 และไปถึง B ผ่านทางอินเทอร์เฟซ 5?

❖ A: แต่ละ switch จะมี *switch table*:

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

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

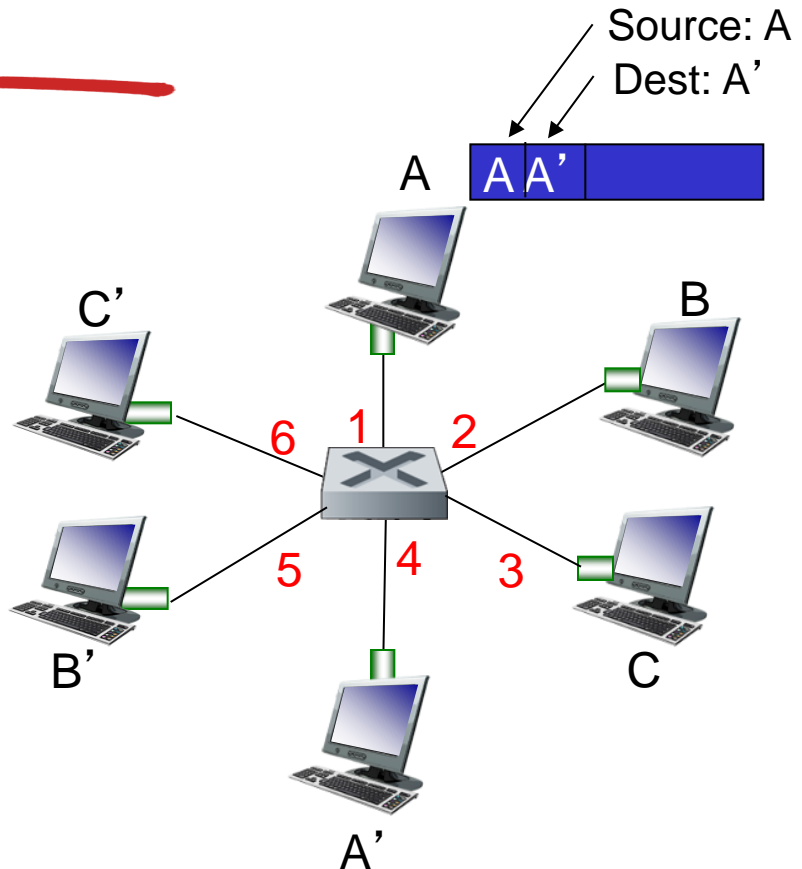
- something like a routing protocol?



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

Switch: self-learning

- ❖ switch *learns* which hosts can be reached through which interfaces
 - when frame received, switch “learns” location of sender: incoming LAN segment
 - records sender/location pair in switch table



MAC addr	interface	TTL
A	1	60

*Switch table
(initially empty)*

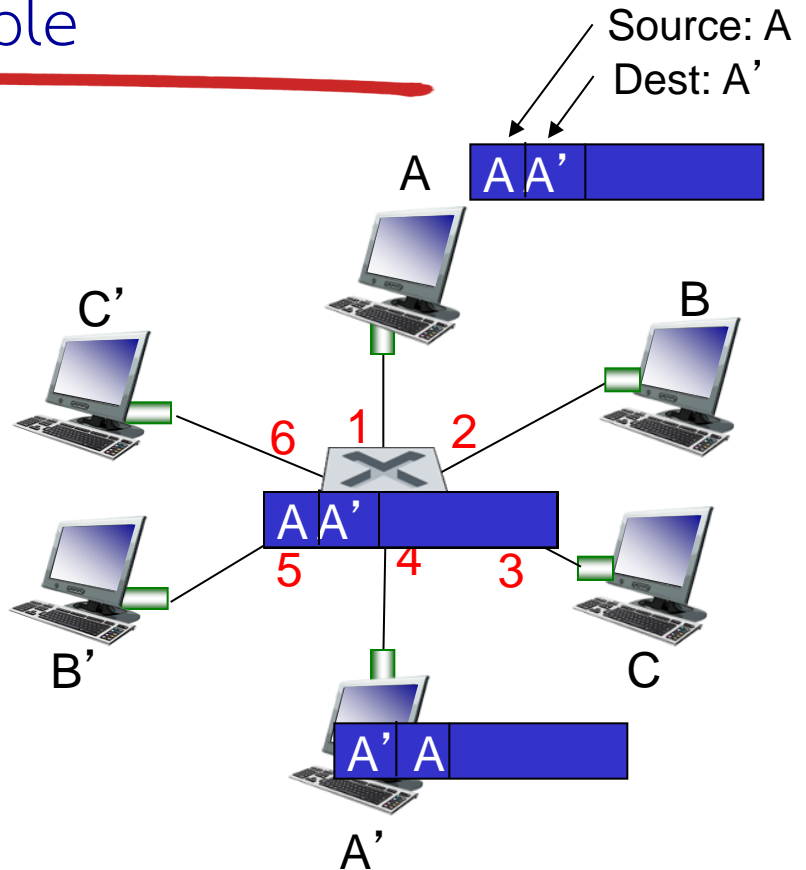
Switch: frame filtering/forwarding

เมื่อสวิตช์ได้รับเฟรม:

1. จำแนกแอดเดรสของผู้ส่งไว้
2. จัดเรียงแอดเดรสลงในตาราง
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

- ❖ ไม่รู้ที่อยู่ A ก็จะบรอดคาส
- ❖ เมื่อรู้ที่อยู่แล้วก็จะเลือกส่งไปยังลิงค์เดียว

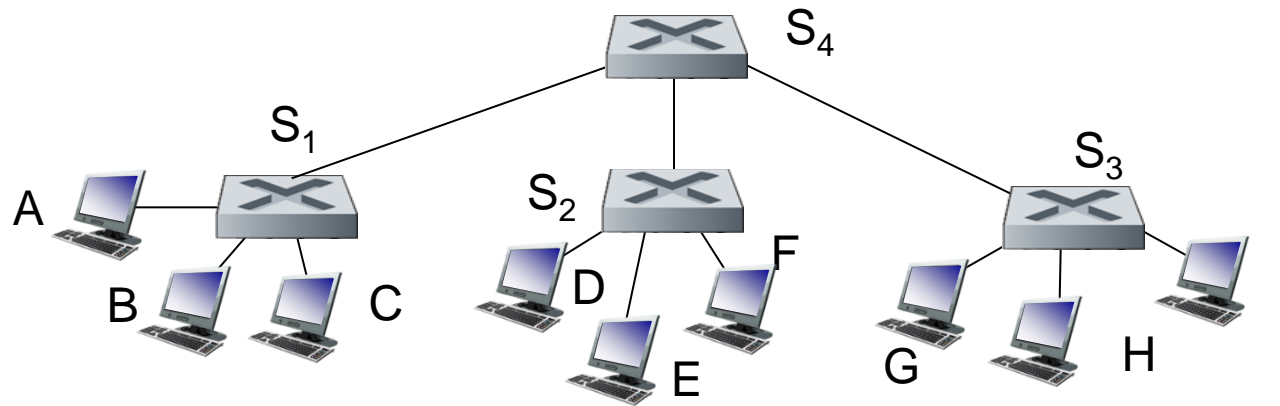


MAC addr	interface	TTL
A	1	60
A'	4	60

*switch table
(initially empty)*

Interconnecting switches

❖ สวิตช์สามารถเชื่อมต่อกันได้

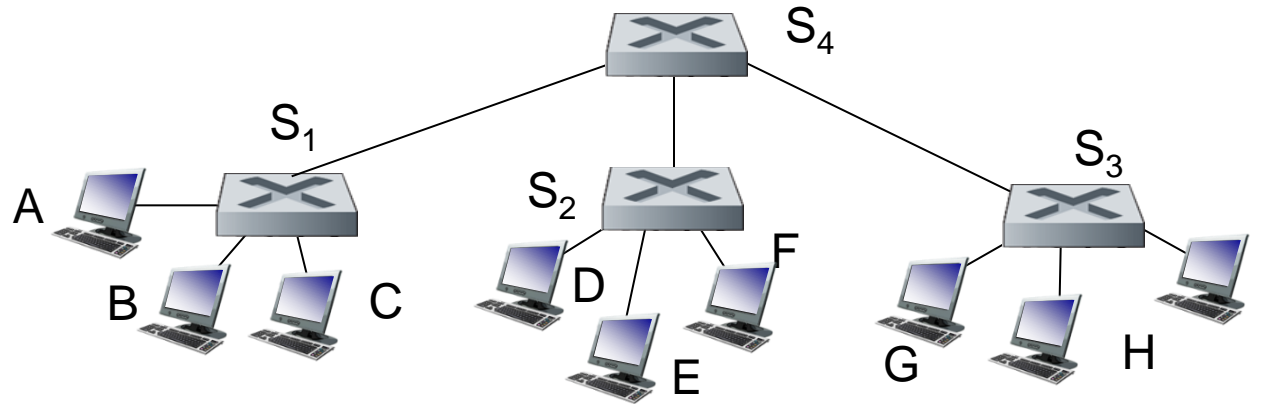


Q: ส่งข้อมูลจาก A ไปยัง G S_1 จะรู้ได้อย่างไรว่าจะต้องส่งข้อมูลไปยังปลายทางที่ G
ว่าจะผ่าน S_4 หรือ S_3 ?

❖ A: เรียนรู้ด้วยตัวเองเหมือนมีสวิตช์เดียว

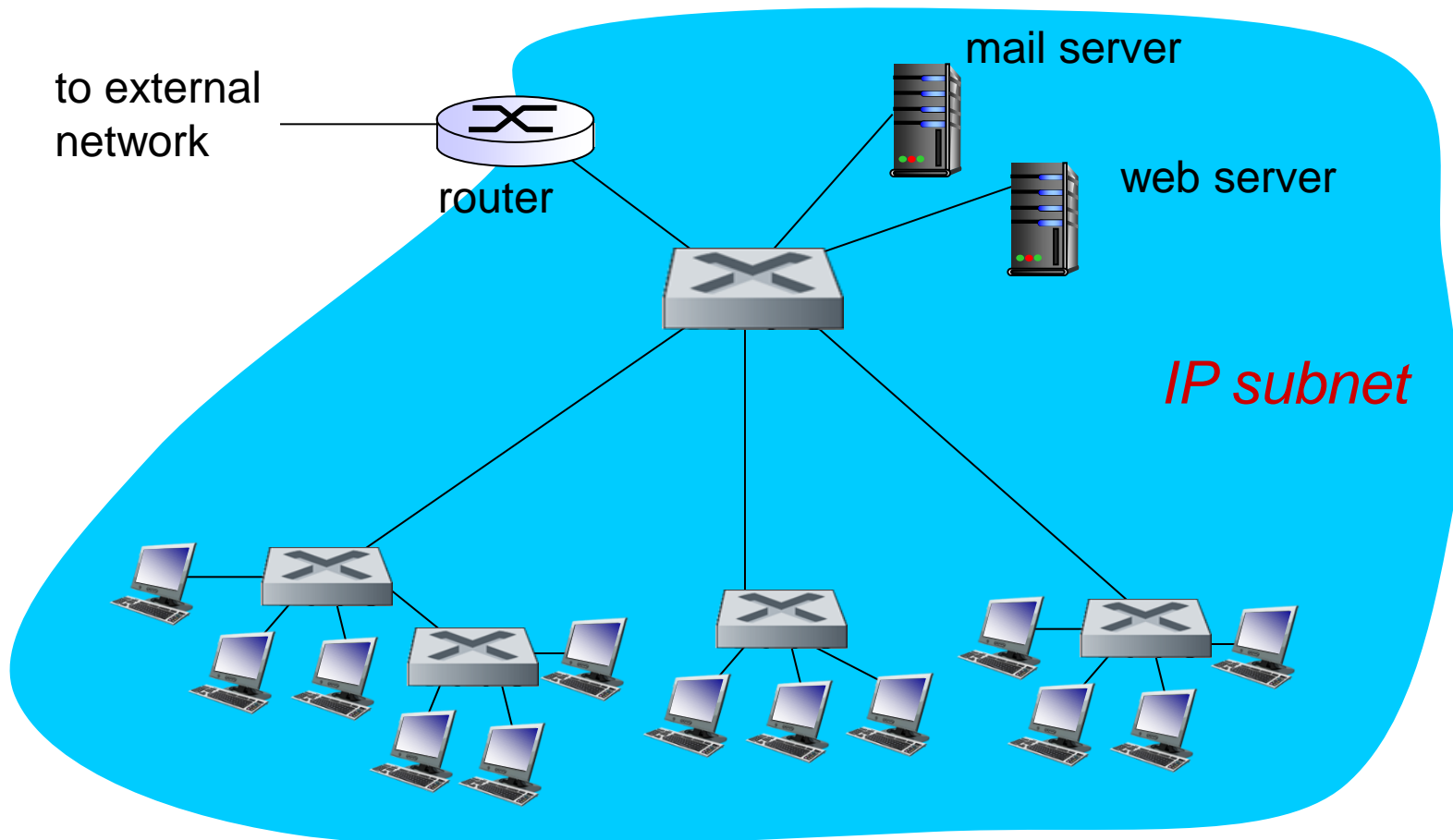
Self-learning multi-switch example

Suppose C sends frame to I, I responds to C



❖ Q: show switch tables and packet forwarding in S₁, S₂, S₃, S₄

Institutional network



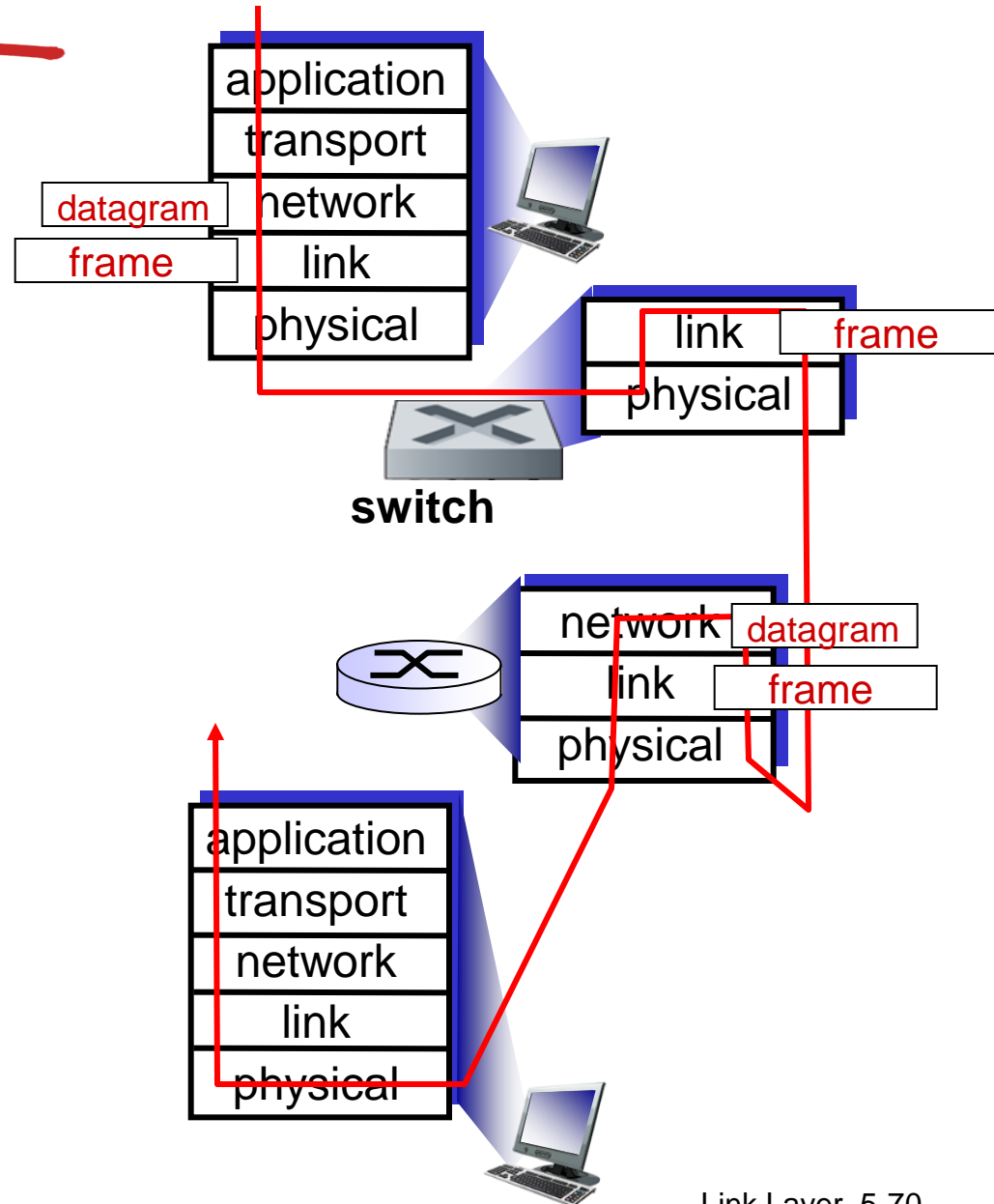
Switches vs. routers

ทั้งสองแบบเป็น store-and-forward:

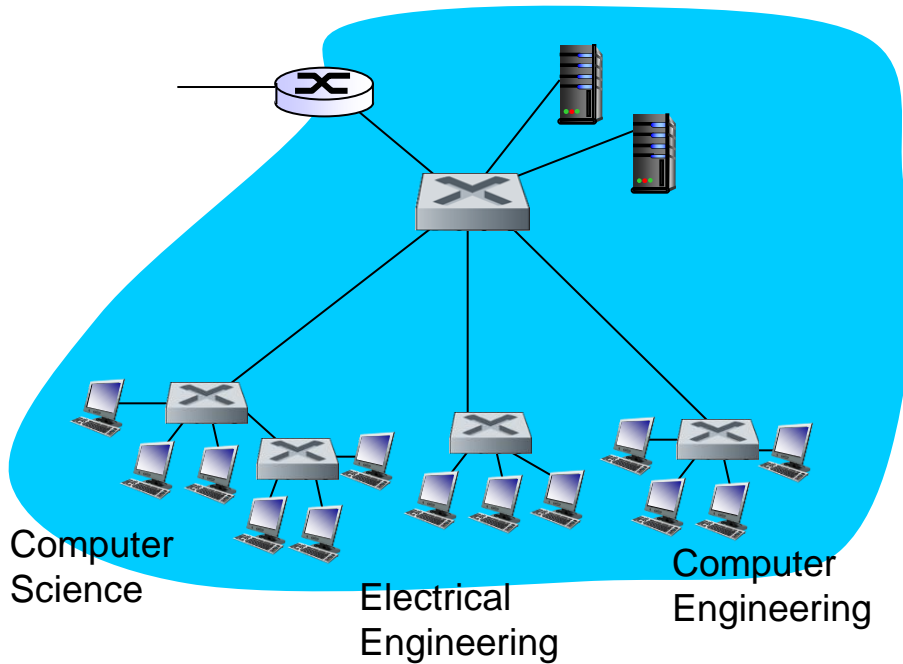
- *routers*: เป็นอุปกรณ์เลเยอร์ 3
- *switches*: เป็นอุปกรณ์เลเยอร์ 2

ทั้งคู่มี forwarding tables:

- *routers*: หาเส้นทางส่งด้วย ip address
- *switches*: หาเส้นทางผ่านทาง broadcast ใช้ MAC addresses



VLANs: แรงจูงใจ



consider:

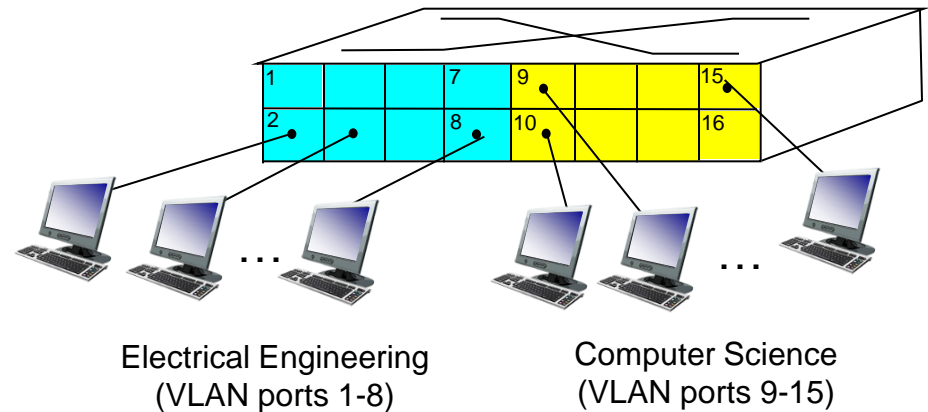
- ❖ ผู้ใช้ CS ย้ายออฟฟิศไปยัง EE แต่ยังคงต้องการเชื่อมต่อกับ CS switch
- ❖ single broadcast domain:
 - บรอดคาสบนเลเยอร์ 2 ทั้งหมดผ่านทางแลนเดียวกัน (ARP, DHCP, unknown location of destination MAC address)
 - มีความต้องการทางด้านความปลอดภัยและประสิทธิภาพ

VLANs

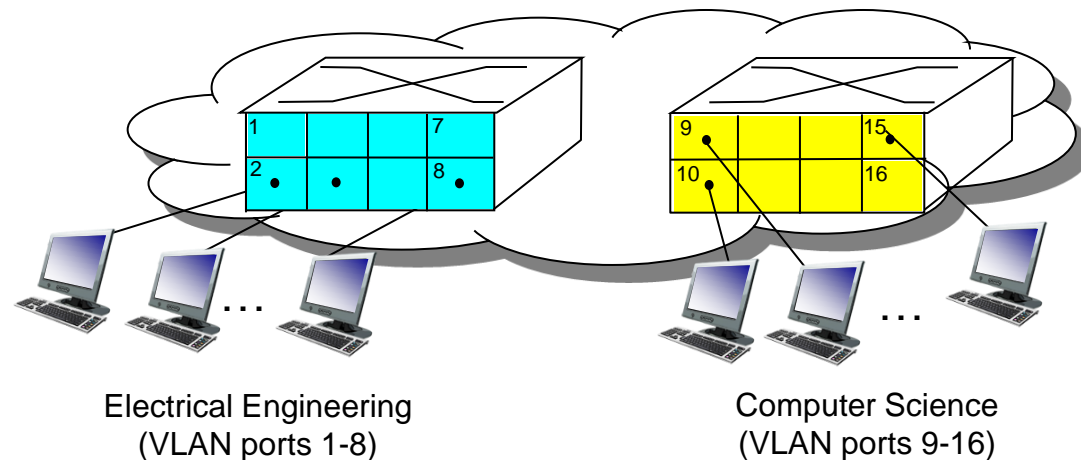
Virtual Local Area Network

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

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

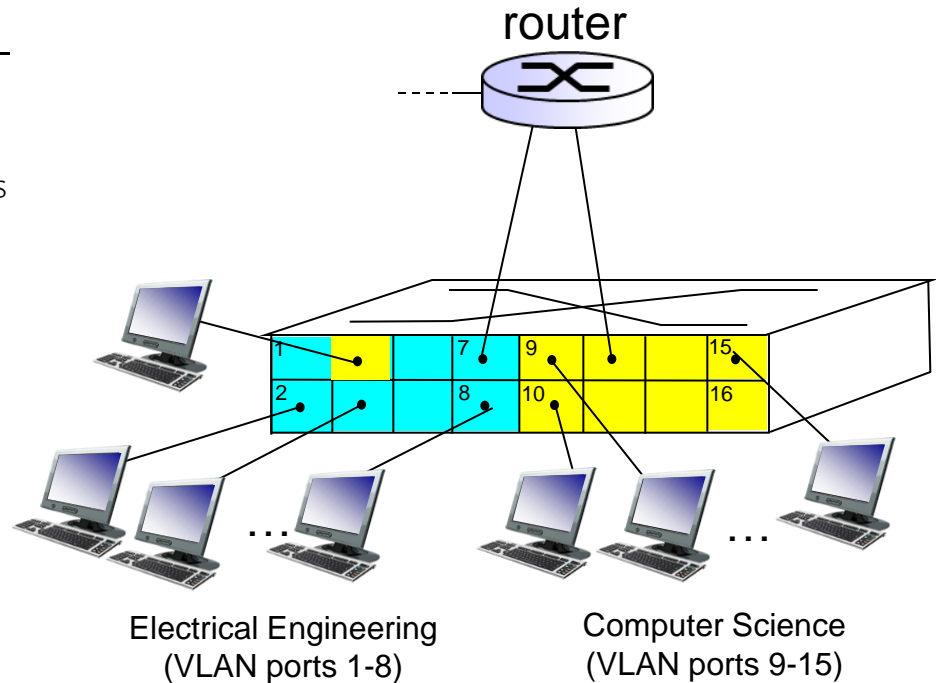


... operates as **multiple** virtual switches

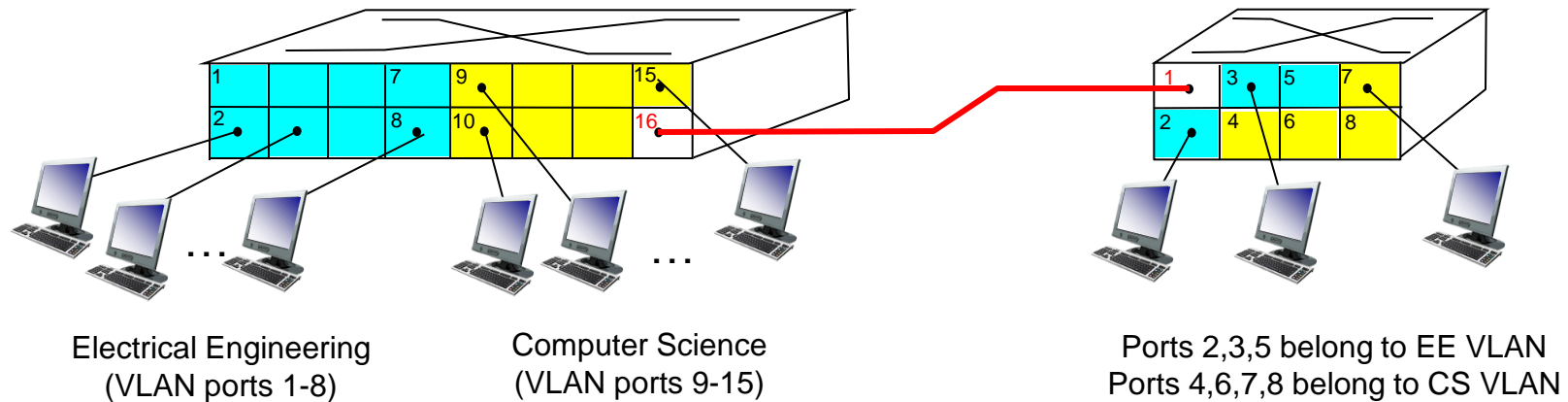


Port-based VLAN

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

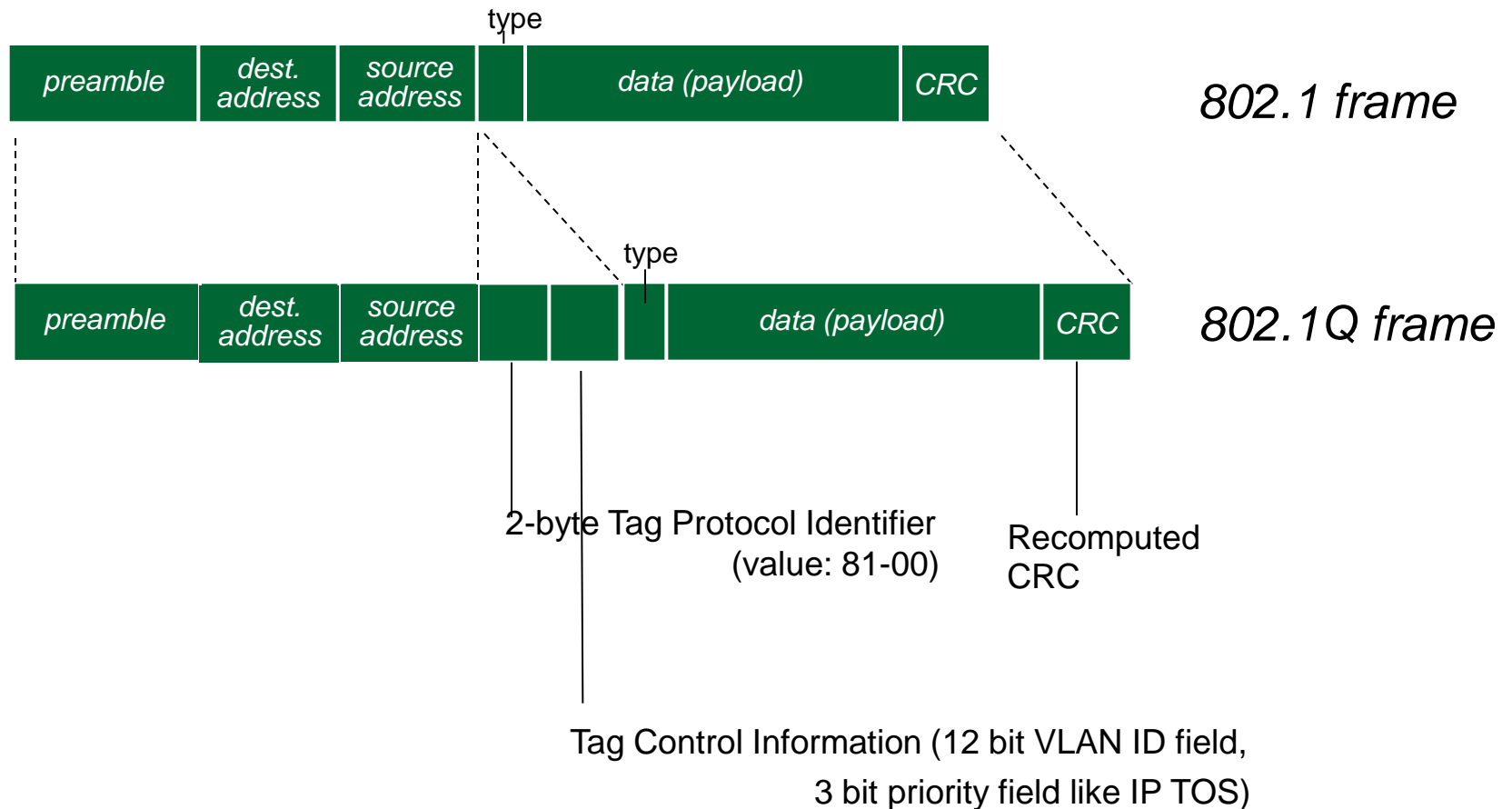


VLANs spanning multiple switches



- ❖ *trunk port*: carries frames between VLANs defined over multiple physical switches
 - frames forwarded within VLAN between switches can't be vanilla 802.1 frames (must carry VLAN ID info)
 - 802.1q protocol adds/removed additional header fields for frames forwarded between trunk ports

802.1Q VLAN frame format



Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs

- addressing, ARP
- Ethernet
- switches
- VLANs

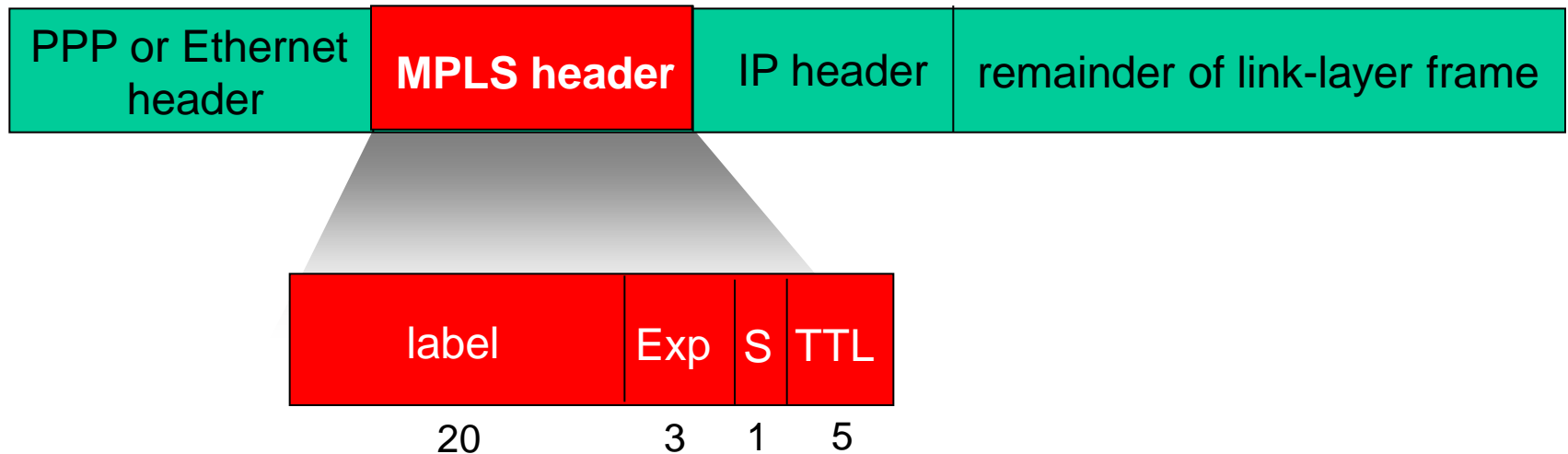
5.5 link virtualization: MPLS

5.6 data center networking

5.7 a day in the life of a web
request

Multiprotocol label switching (MPLS)

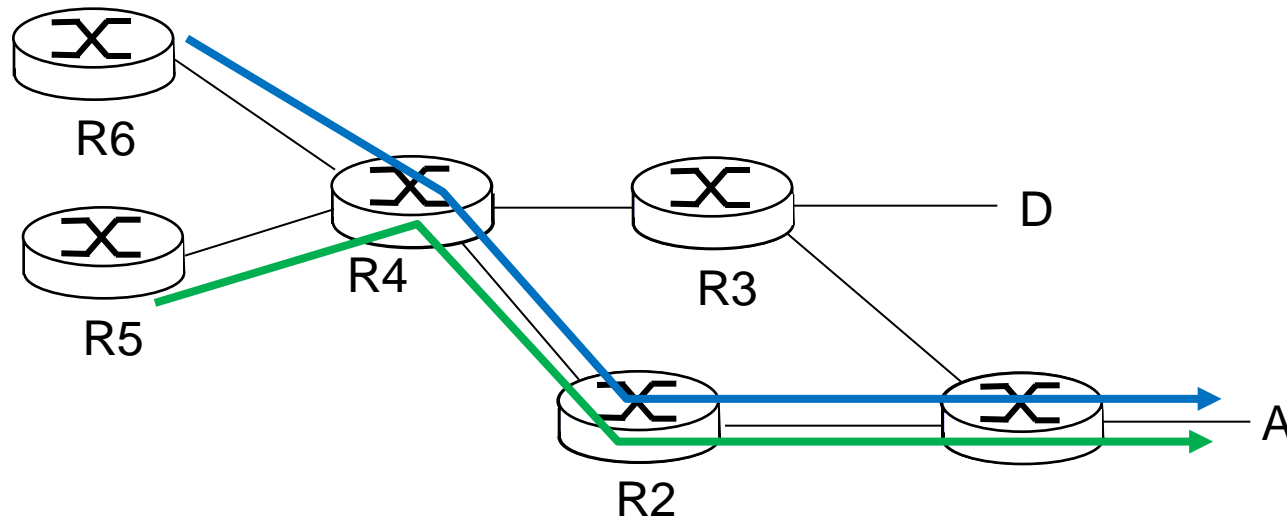
- ❖ initial goal: high-speed IP forwarding using fixed length label (instead of IP address)
 - fast lookup using fixed length identifier (rather than shortest prefix matching)
 - borrowing ideas from Virtual Circuit (VC) approach
 - but IP datagram still keeps IP address!



MPLS capable routers

- ❖ a.k.a. label-switched router
- ❖ forward packets to outgoing interface based only on label value (*don't inspect IP address*)
 - MPLS forwarding table distinct from IP forwarding tables
- ❖ *flexibility*: MPLS forwarding decisions can *differ* from those of IP
 - use destination *and* source addresses to route flows to same destination differently (traffic engineering)
 - re-route flows quickly if link fails: pre-computed backup paths (useful for VoIP)

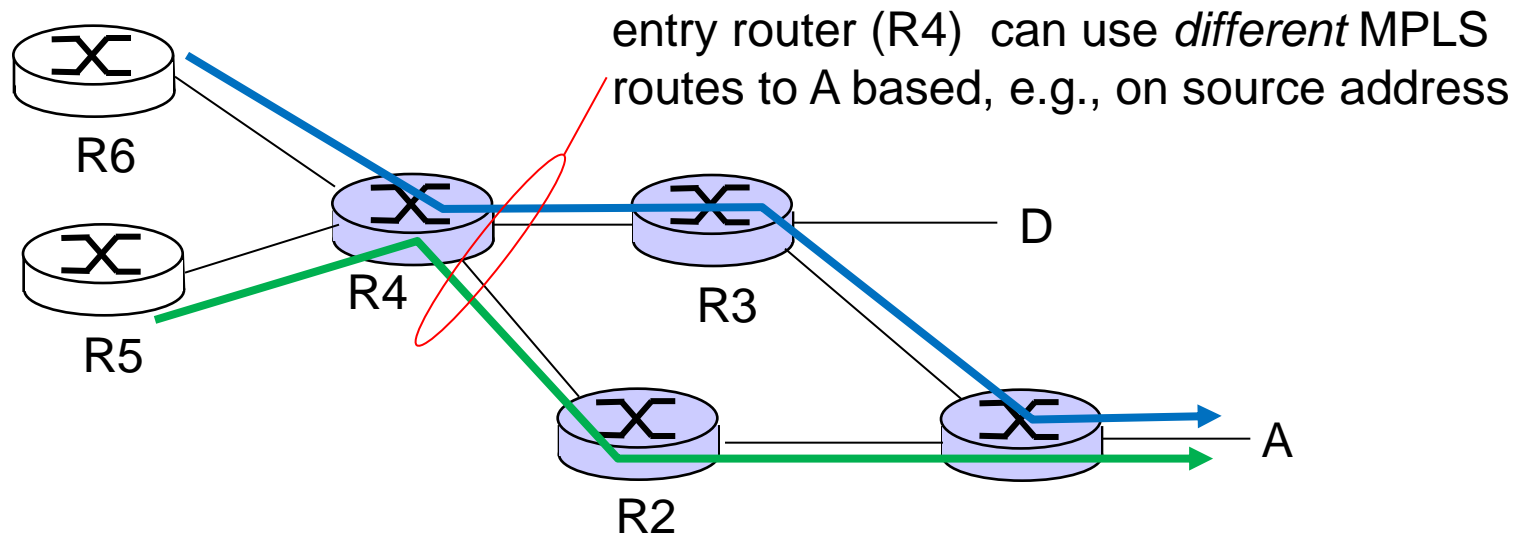
MPLS versus IP paths



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



MPLS versus IP paths



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



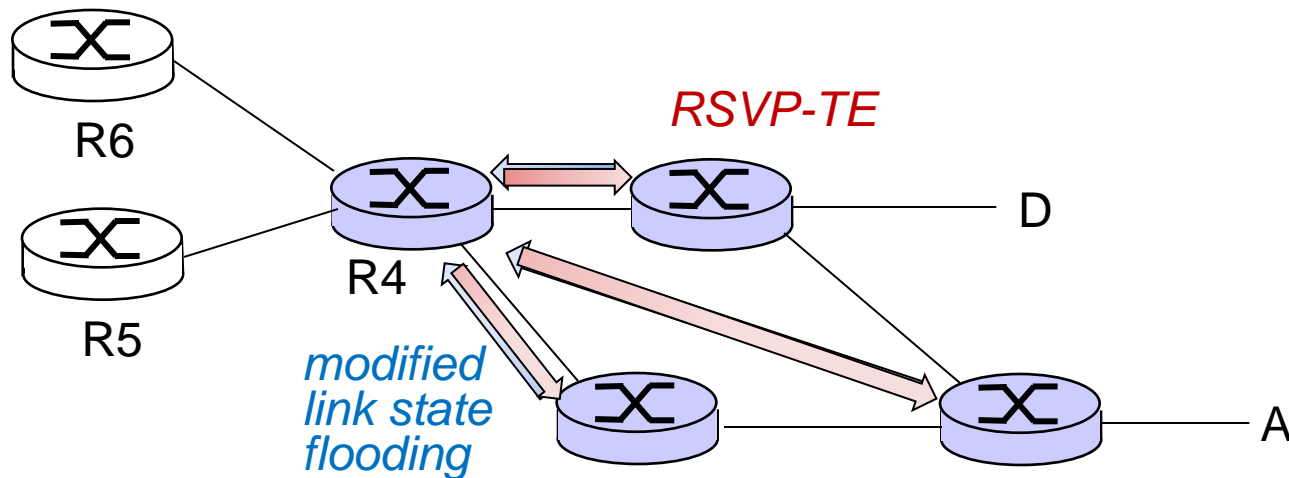
❖ *MPLS routing*: path to destination can be based on source *and* dest. address



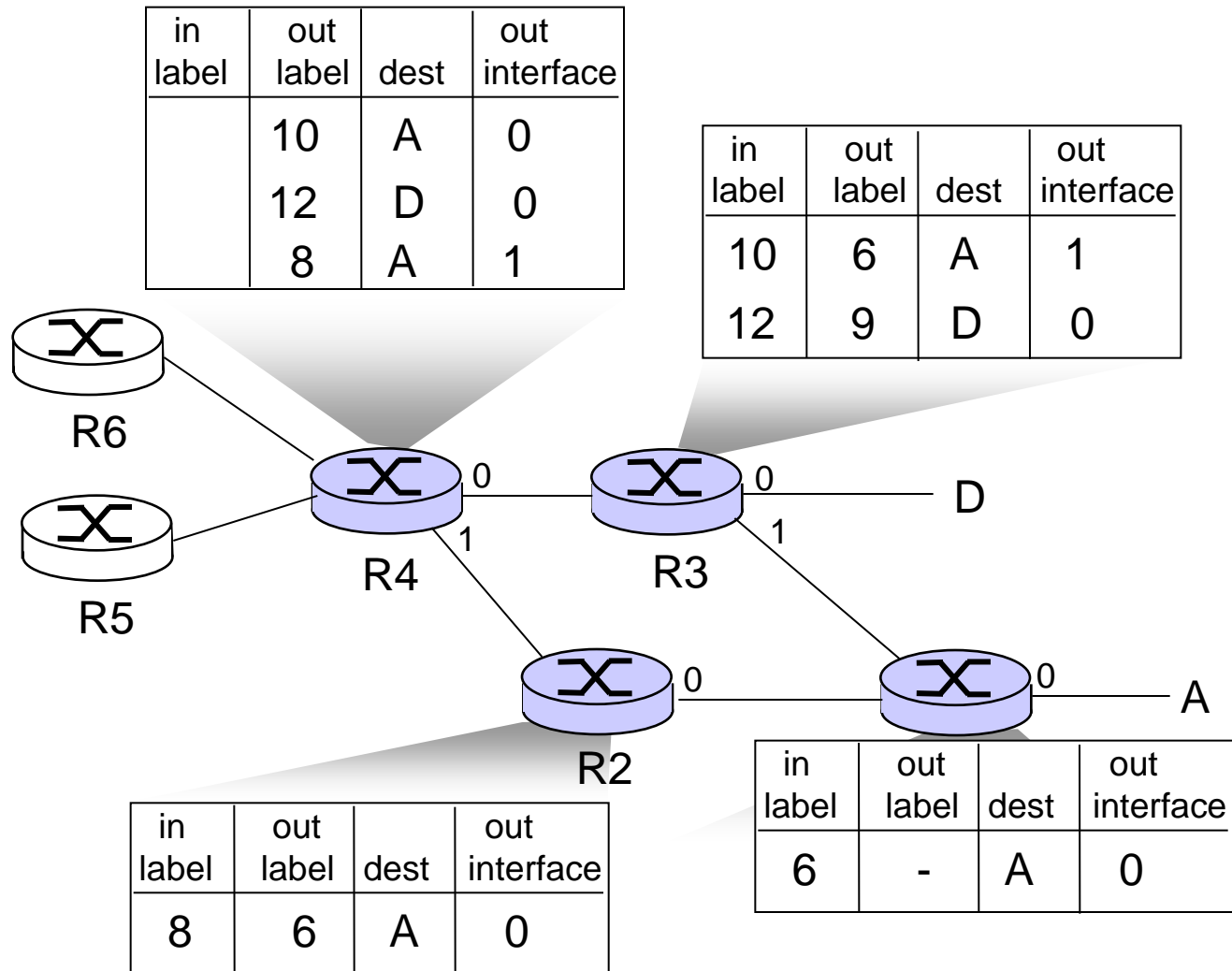
- *fast reroute*: precompute backup routes in case of link failure

MPLS signaling

- ❖ modify OSPF, IS-IS link-state flooding protocols to carry info used by MPLS routing,
 - e.g., link bandwidth, amount of “reserved” link bandwidth
- ❖ *entry MPLS router uses RSVP-TE signaling protocol to set up MPLS forwarding at downstream routers*



MPLS forwarding tables



Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs

- addressing, ARP
- Ethernet
- switches
- VLANs

5.5 link virtualization: MPLS

5.6 data center networking

5.7 a day in the life of a web request

Data center networks

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

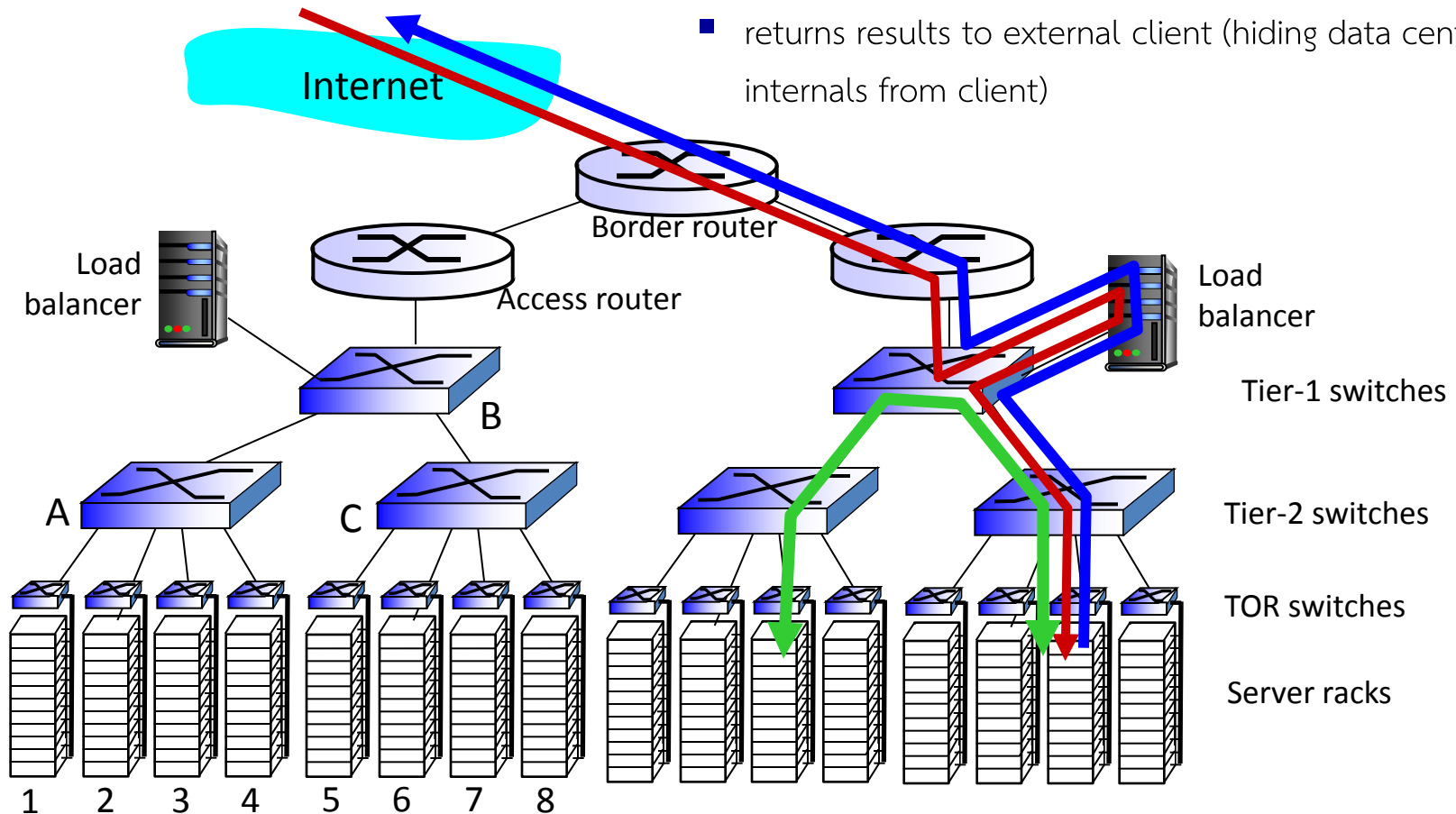


Inside a 40-ft Microsoft container,
Chicago data center

Data center networks

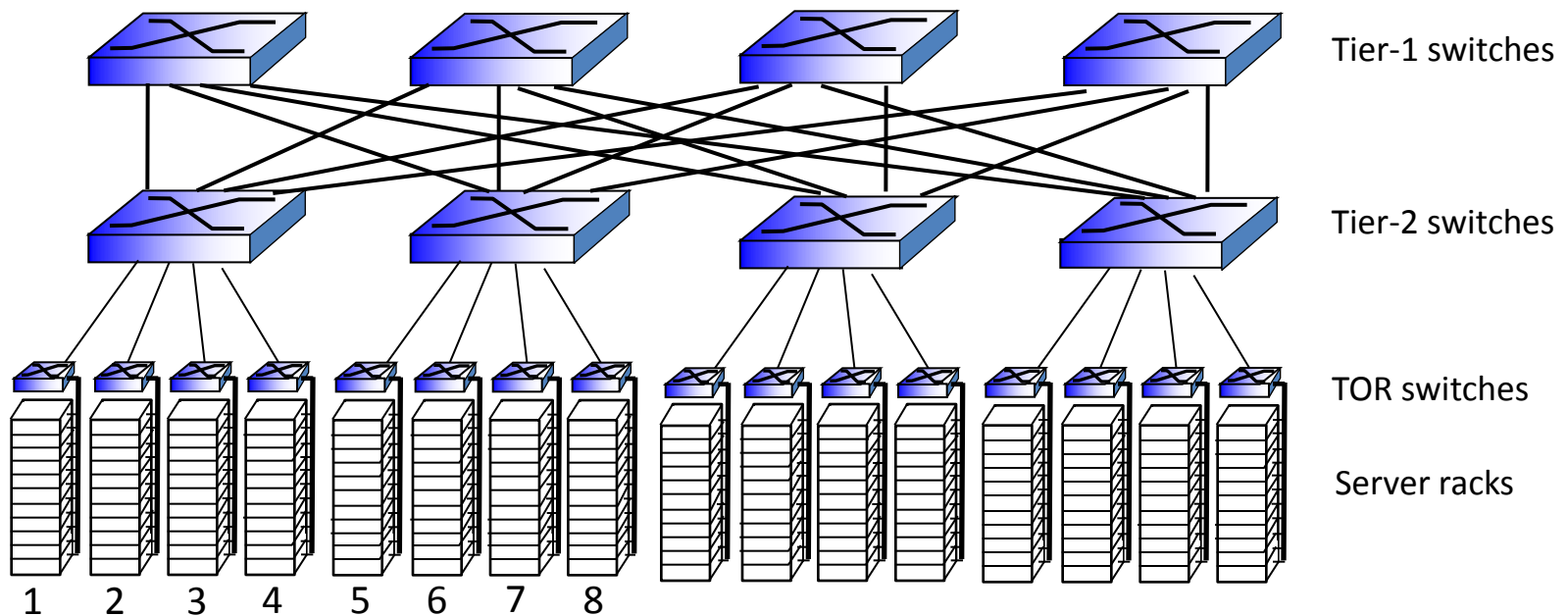
load balancer: application-layer routing

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



Data center networks

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



Link layer, LANs: outline

5.1 introduction, services

5.2 error detection, correction

5.3 multiple access protocols

5.4 LANs

- addressing, ARP
- Ethernet
- switches
- VLANs

5.5 link virtualization: MPLS

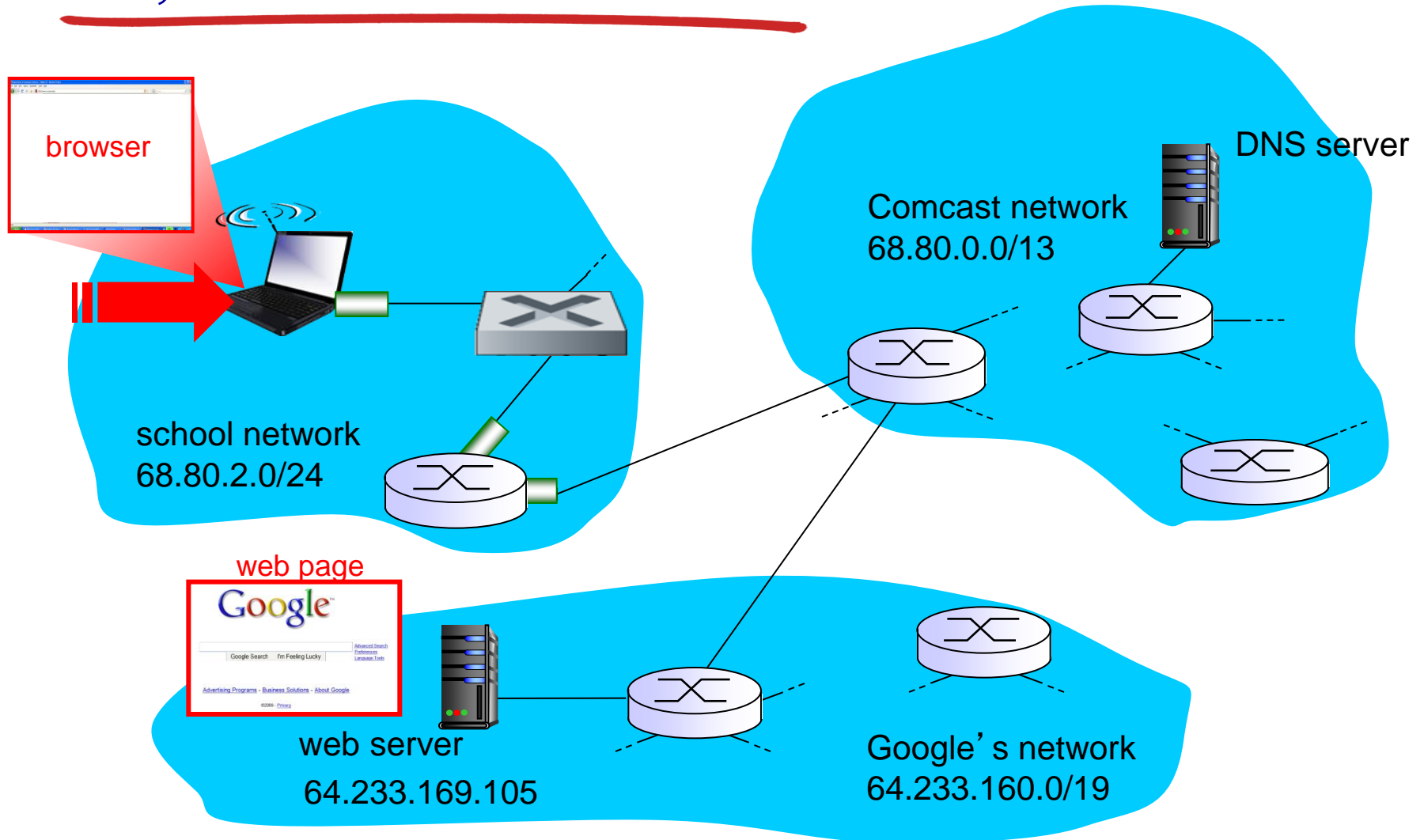
5.6 data center networking

5.7 a day in the life of a web
request

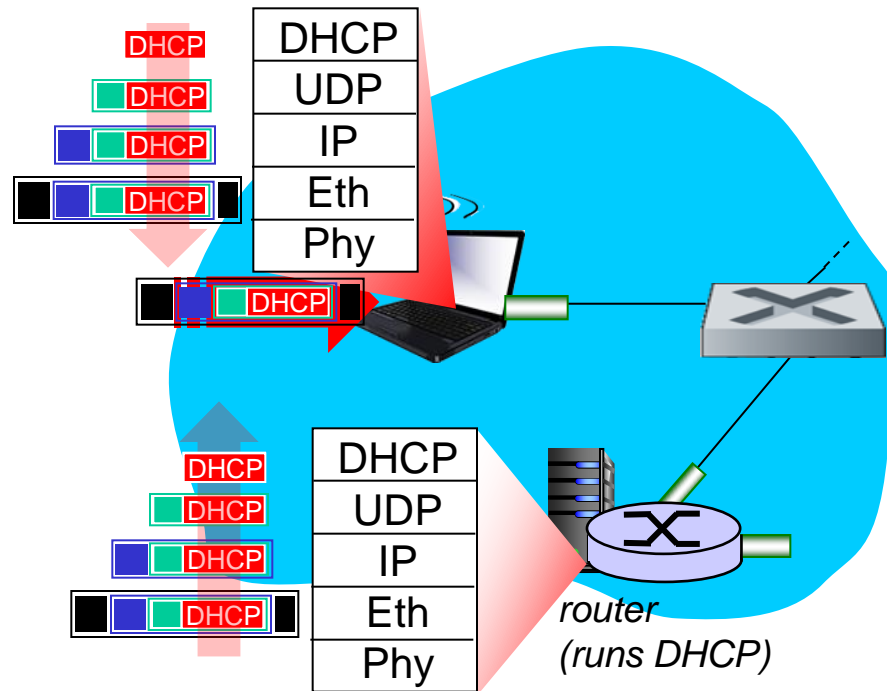
Synthesis: a day in the life of a web request

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

A day in the life: scenario

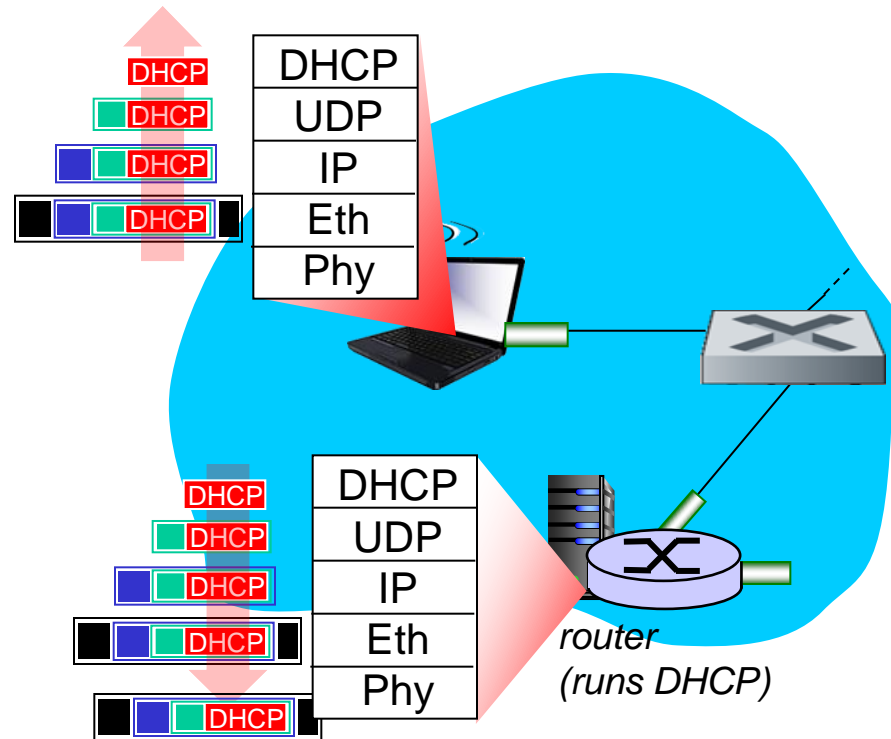


A day in the life... connecting to the Internet



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

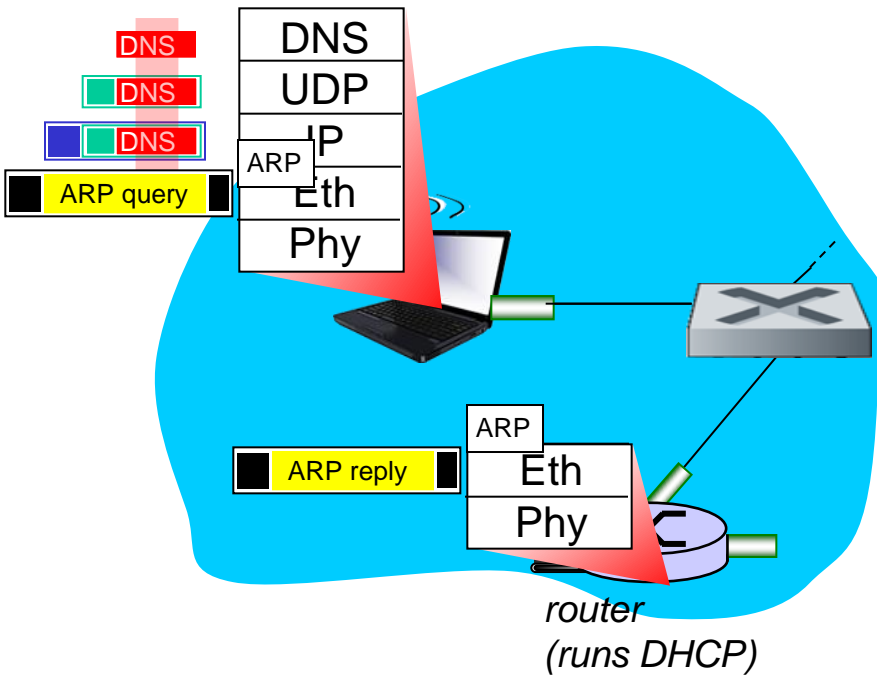
A day in the life... connecting to the Internet



- ❖ 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

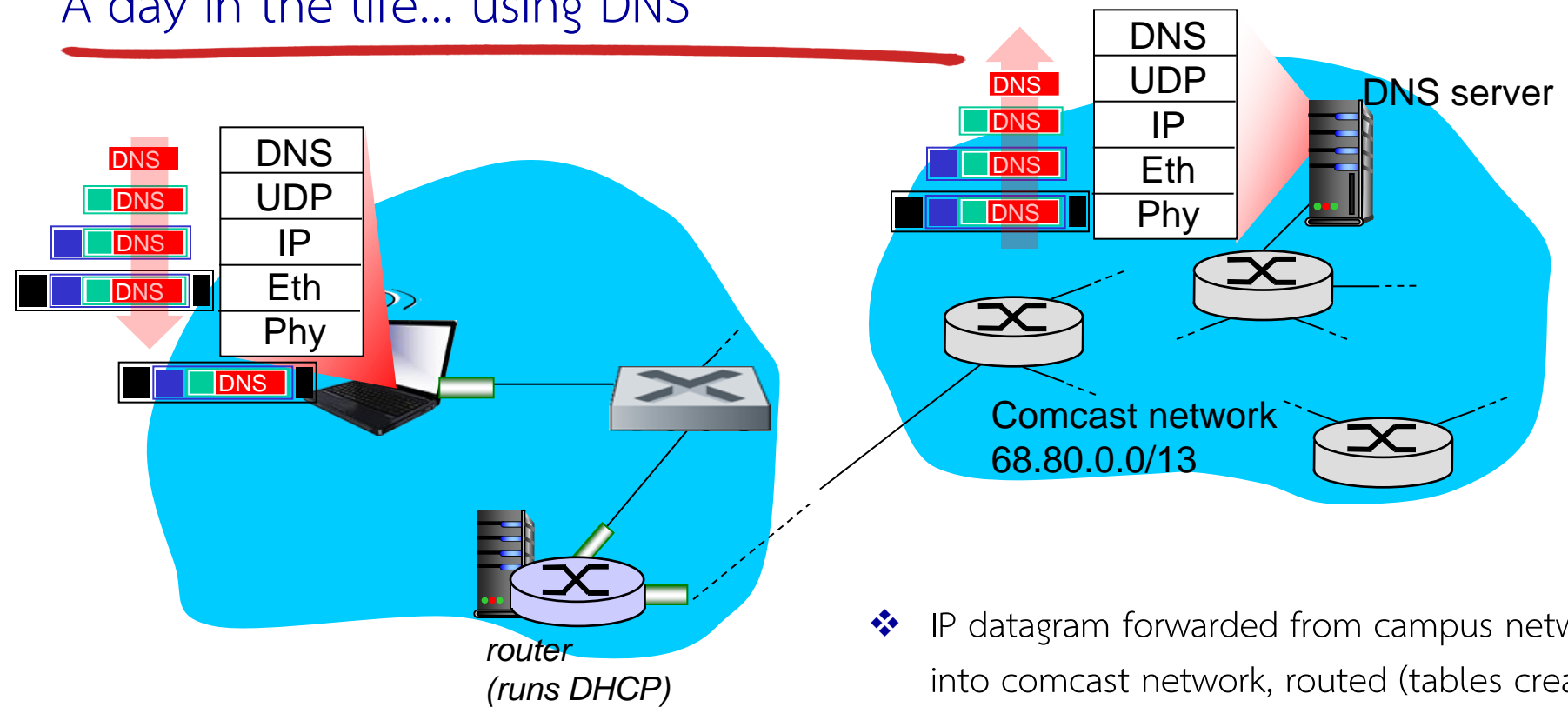
Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

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

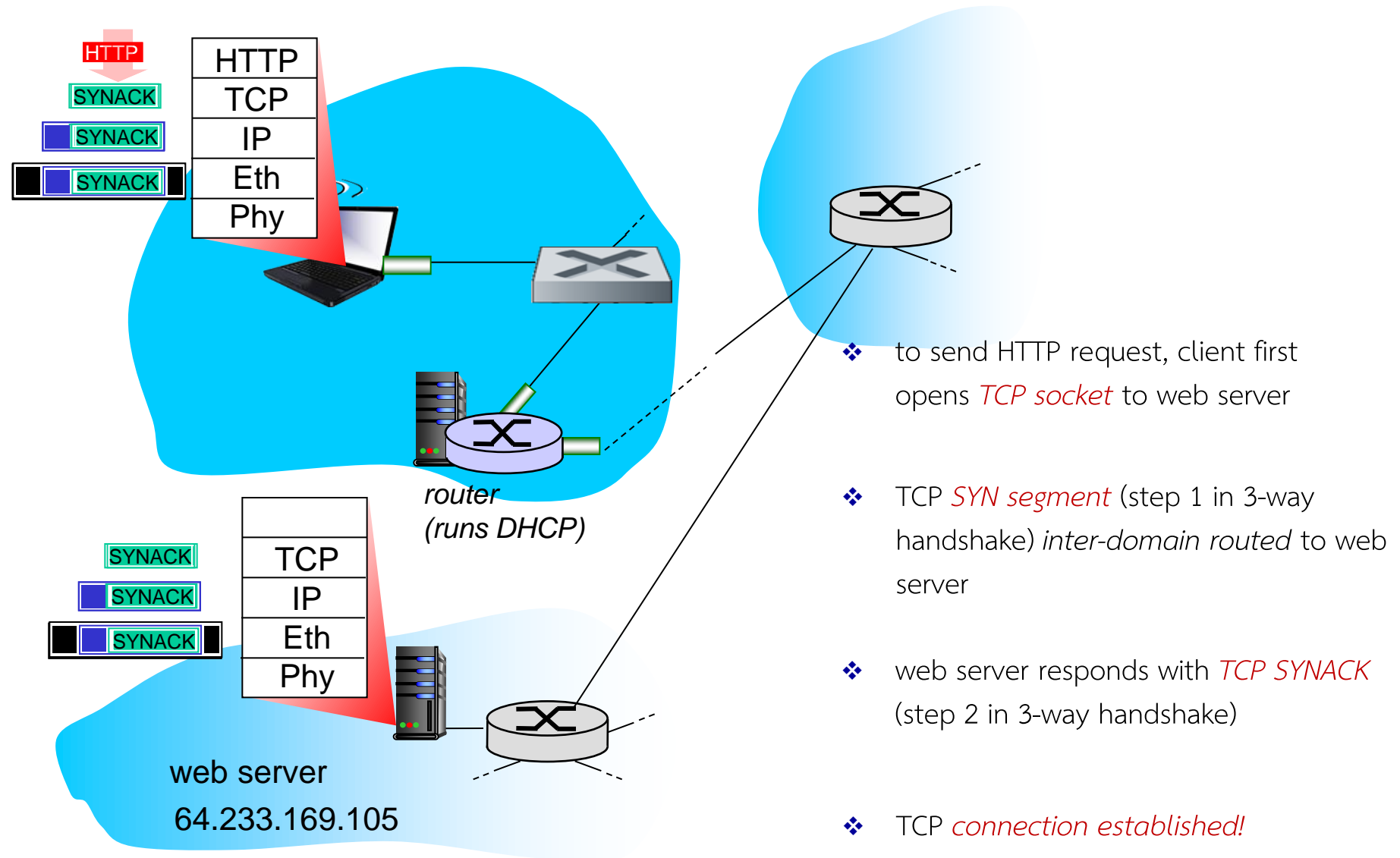
A day in the life... using DNS



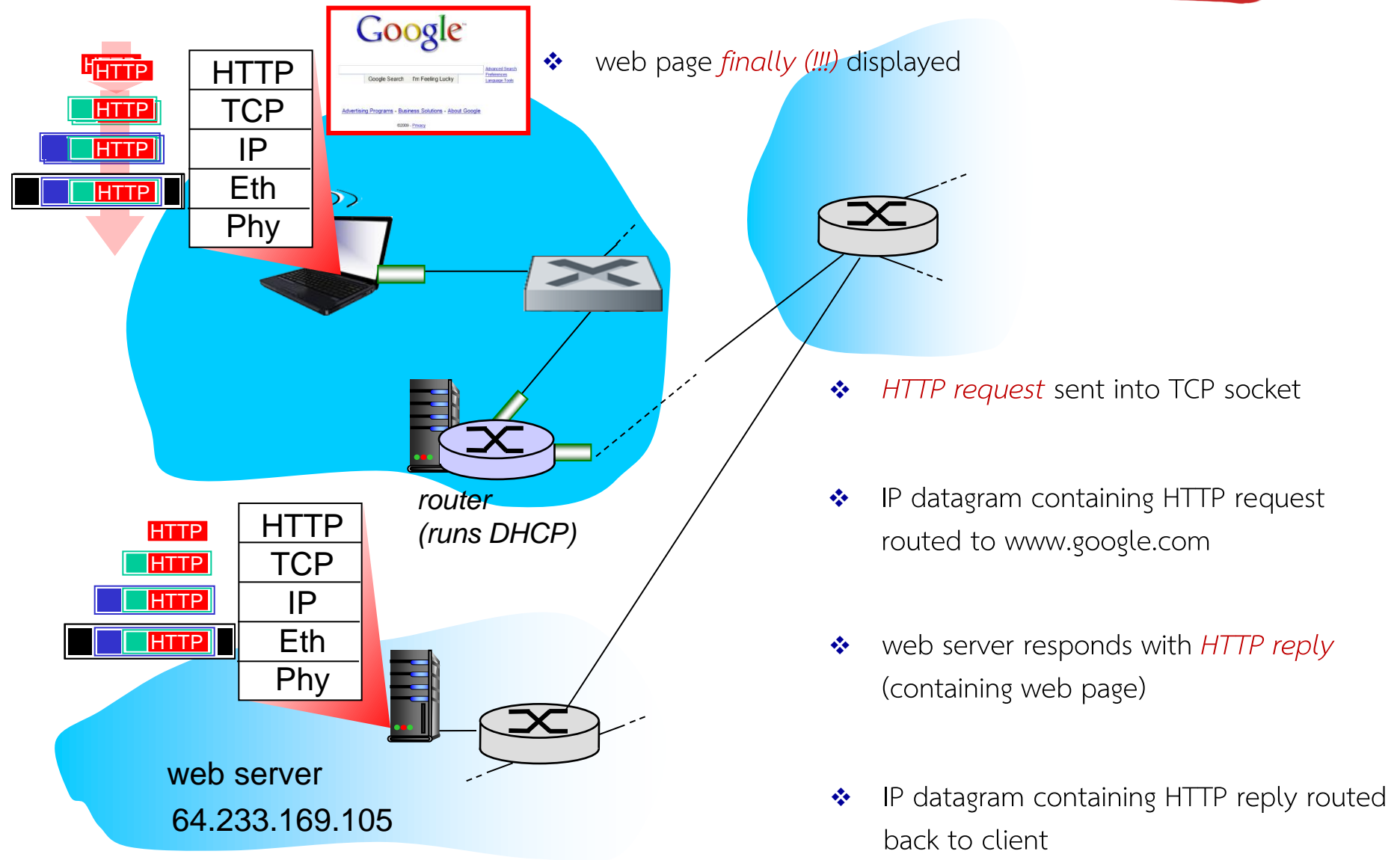
- ❖ IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router

- ❖ 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
- ❖ 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



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



Chapter 5: Summary

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

Chapter 5: let's take a breath

- ❖ journey down protocol stack *complete* (except PHY)
- ❖ solid understanding of networking principles, practice
- ❖ could stop here but *lots* of interesting topics!
 - wireless
 - multimedia
 - security
 - network management