

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**  
**Posts and Telecommunications Institute of Technology**

# **MẠNG MÁY TÍNH**

**Chương 5: Tầng liên kết và mạng LAN**

# Mục tiêu

- Hiểu được các nguyên lý của các dịch vụ tầng liên kết
  - Phát hiện và sửa lỗi
  - Chia sẻ kênh truyền chung (broadcast channel): đa truy nhập
  - Định địa chỉ tầng liên kết
  - Các mạng cục bộ: Ethernet, VLANs
- Cài đặt và hiện thực các công nghệ tầng mạng khác nhau

# Nội dung

- Giới thiệu, các dịch vụ
- Phát hiện và sửa lỗi
- Các giao thức đa truy nhập
- Các mạng LAN
  - Định địa chỉ, ARP
  - Ethernet
  - Các switch
  - Các VLAN
- Chuyển mạch nhãn đa giao thức (MPLS)
- Mạng trung tâm dữ liệu
- Vòng đời của một yêu cầu web

# Nội dung

- Giới thiệu, các dịch vụ
- Phát hiện và sửa lỗi
- Các giao thức đa truy nhập
- Các mạng LAN
  - Định địa chỉ, ARP
  - Ethernet
  - Các switch
  - Các VLAN
- Chuyển mạch nhãn đa giao thức (MPLS)
- Mạng trung tâm dữ liệu
- Vòng đời của một yêu cầu web

# Tầng liên kết: giới thiệu

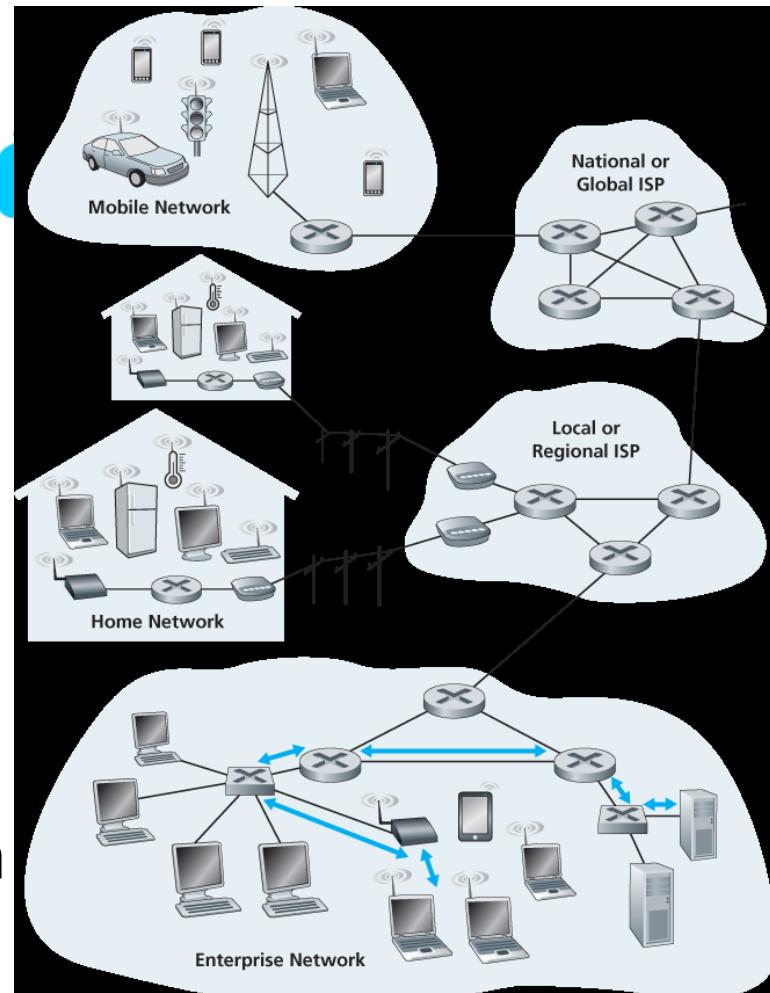
- Node?
- Links?

# Tầng liên kết: giới thiệu

- *Thuật ngữ:*

- Các host và các router: **các nút mạng (node)**
- Các kênh truyền thông kết nối giữa các nút lân cận theo đường truyền thông: gọi là các liên kết (hay các kết nối, **link**)
  - Các liên kết có dây
  - Các liên kết không dây
  - Các LAN
- Gói tin tầng 2: **khung (frame)**, đóng gói datagram

*Tầng liên kết dữ liệu* có trách nhiệm truyền datagram từ một nút đến nút **vật lý lân cận** qua một liên kết



# Tầng liên kết: ngũ cảnh

- Datagram được truyền bởi các giao thức liên kết khác nhau qua các liên kết khác nhau:
  - Ví dụ: Ethernet trên liên kết thứ nhất, frame relay trên các liên kết trung gian, 802.11 trên liên kết cuối cùng.
- Mỗi giao thức liên kết cung cấp các dịch vụ khác nhau.
  - Ví dụ: có thể hoặc không cung cấp truyền tin cây (rdt) qua liên kết

# Các dịch vụ tầng liên kết

- Tạo khung dữ liệu - Faming
  - Đóng gói datagram vào trong **frame**, thêm phần tiêu đề (header), phần đuôi (trailer)
  - Cấu trúc của Frame được quy định bởi giao thức tầng liên kết. Các giao thức khác nhau có định dạng frame khác nhau.

# Các dịch vụ tầng liên kết

- Truy nhập liên kết – Link access
  - Giao thức MAC (medium access control) chỉ ra các nguyên tắc truyền frame lên liên kết.
  - Với các liên kết point-to-point, giao thức MAC đơn giản (hoặc không tồn tại)
  - Khi nhiều node cùng chia sẻ một liên kết quảng bá (broadcast link), giao thức MAC điều phối việc truyền frame của nhiều node.

# Các dịch vụ tầng liên kết

- Truyền tin cậy - Reliable delivery
  - Dịch vụ truyền tin tin cậy ở link-layer gồm
    - Acknowledgments
    - Retransmissions
  - Thường được dùng trên liên kết có tỷ lệ lỗi truyền cao. Ít khi được dùng trên liên kết có tỷ lệ lỗi thấp (cáp quang, một số loại cáp xoắn)

*Hỏi:* Tại sao cần truyền tin cậy ở cả mức liên kết và mức đầu cuối-đến-đầu cuối?

# Các dịch vụ tầng liên kết

- Phát hiện và sửa lỗi - Error detection and correction
  - Lỗi là do suy giảm tín hiệu, nhiễu
  - Bên nhận phát hiện ra sự xuất hiện của các lỗi:
    - Thông báo cho bên gửi truyền lại hoặc loại bỏ frame đó
  - Phát hiện lỗi ở tầng liên kết thường phức tạp hơn ở tầng mạng và tầng giao vận và thường được thực thi bởi phần cứng.

# Các dịch vụ tầng liên kết

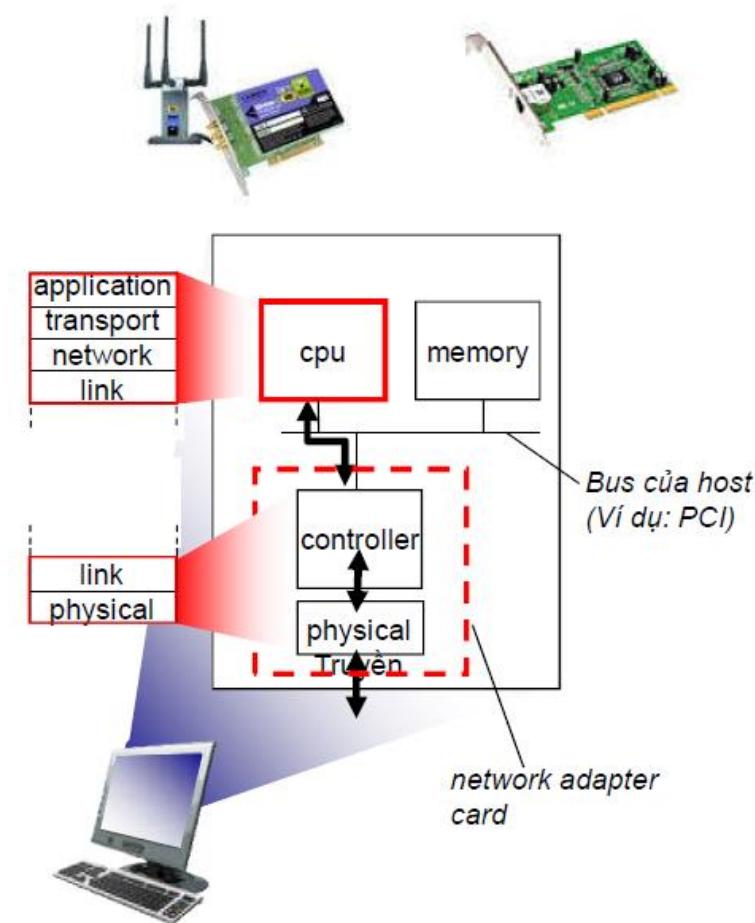
- Điều khiển luồng
  - Điều khiển tốc độ giữa các nút gửi và nhận kề nhau
- Phát hiện lỗi
  - Lỗi là do suy giảm tín hiệu, nhiễu
  - Bên nhận phát hiện ra sự xuất hiện của các lỗi:
    - Thông báo cho bên gửi truyền lại hoặc loại bỏ frame đó

# Các dịch vụ tầng liên kết

- Sửa lỗi
  - Bên nhận xác định **và sửa** các lỗi bit mà không cần phải yêu cầu truyền lại
- Bán song công (half-duplex) và song công (full-duplex)
  - Với bán song công, cả hai đầu cuối của liên kết đều có thể truyền, nhưng không được truyền tại cùng một thời điểm.

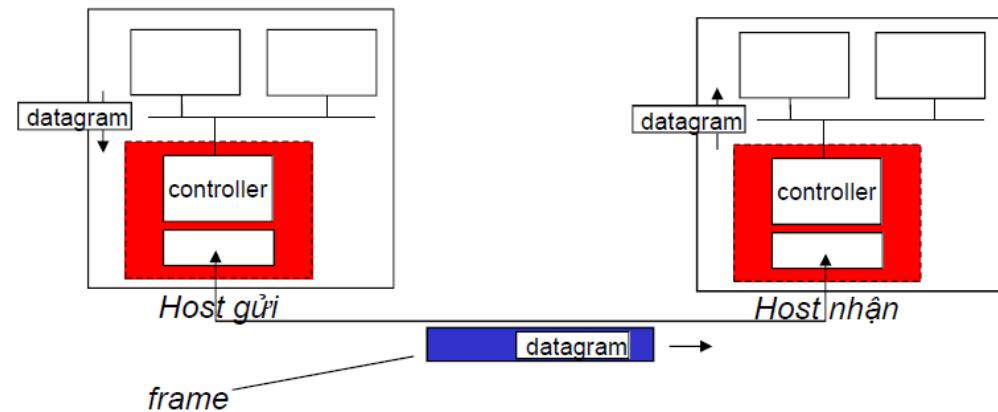
# Tầng liên kết được cài đặt ở đâu?

- Tại tất cả các host
- Tầng liên kết được cài đặt tại “adaptor” (còn được gọi là *thẻ giao diện mạng (network interface card - NIC)*) hoặc trên chip
  - Ethernet card, 802.11 card; Ethernet chipset
  - Cài đặt tầng liên kết và tầng vật lý
- Gắn vào bên trong các bus hệ thống của host
- Kết hợp phần cứng, phần mềm, phần sụn (firmware)



# Các adaptor truyền thông

- Phía gửi:
  - Đóng gói datagram trong frame
  - Bổ sung kiểm tra lỗi bit, rdt, điều khiển luồng,...
- Phía nhận:
  - Kiểm tra lỗi, rdt, điều khiển luồng,...
  - Trích xuất datagram, chuyển lên tầng cao hơn tại phía nhận

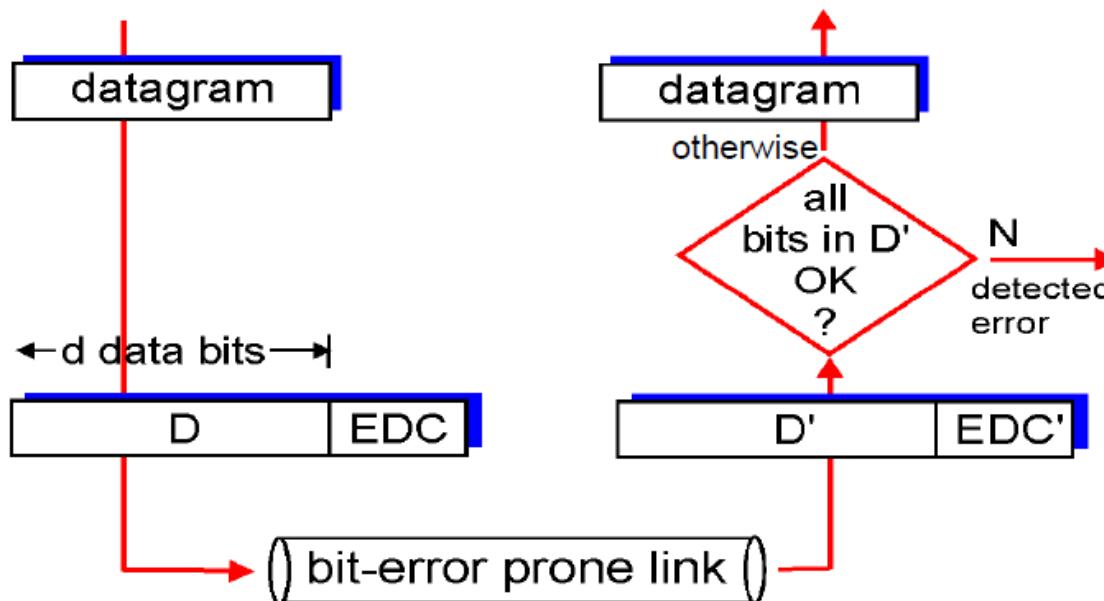


# Nội dung

- Giới thiệu, các dịch vụ
- Phát hiện và sửa lỗi
- Các giao thức đa truy nhập
- Các mạng LAN
  - Định địa chỉ, ARP
  - Ethernet
  - Các switch
  - Các VLAN
- Chuyển mạch nhãn đa giao thức (MPLS)
- Mạng trung tâm dữ liệu
- Vòng đời của một yêu cầu web

# Phát hiện và sửa lỗi

- EDC= Các bit dùng để phát hiện và sửa lỗi (Error Detection and Correction bits) (dư thừa)
- D = Dữ liệu được bảo vệ bằng cách kiểm tra lỗi, có thể bao gồm cả các trường trong phần tiêu đề.
- Phát hiện lỗi không thể đảm bảo tin cậy 100%!
  - Giao thức có thể bỏ lỡ một vài lỗi, nhưng rất hiếm khi
  - Trường EDC càng lớn thì càng tốt hơn cho việc phát hiện và sửa lỗi.

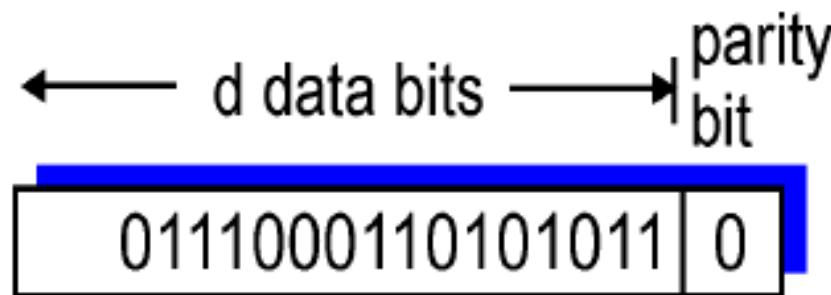


# Phát hiện và sửa lỗi

- Các phương pháp
  - Parity check (bit chẵn lẻ)
    - Parity 1 chiều
    - Parity 2 chiều
    - Hamming
  - Checksum
  - Cyclic Redundancy Check (CRC)

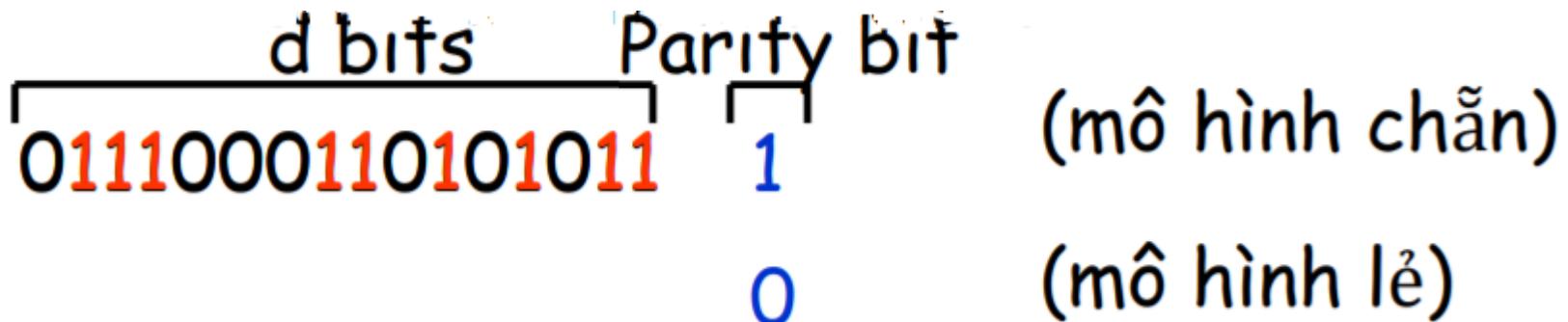
# Kiểm tra Parity

- Parity 1 chiều
  - Số bit parity: 1 bit
  - Chiều dài của dữ liệu cần gửi đi: d bit  
→ DL gửi đi sẽ có  $(d+1)$  bit



# Kiểm tra Parity

- Parity 1 chiều
  - Bên gửi
    - Thêm 1 bit parity vào dữ liệu cần gửi đi
      - Mô hình chẵn (Even parity)
        - » Số bit 1 trong  $(d+1)$  bit là một số chẵn
      - Mô hình lẻ (Odd Parity)
        - » Số bit 1 trong  $(d+1)$  bit là một số lẻ



# Kiểm tra Parity

- Parity 1 chiều
  - Bên nhận
    - Nhận D' có  $(d+1)$  bits
    - Đếm số bit 1 trong  $(d+1)$  bits = x
    - Mô hình chẵn: nếu x lẻ  $\rightarrow$  lỗi
    - Mô hình lẻ: nếu x chẵn  $\rightarrow$  Lỗi
  - Ví dụ: nhận 0111000110101011
    - Parity chẵn: sai
    - Parity lẻ: đúng
      - Dữ liệu thật: 011100011010101

# Kiểm tra Parity

- Parity 2 chiều
  - Dữ liệu gửi đi được biểu diễn thành ma trận NxM
  - Số bit parity:  $(N + M + 1)$  bit
  - Đặc điểm:
    - Phát hiện và sửa được 1 bit lỗi
  - Bên gửi
    - Biểu diễn dữ liệu cần gửi đi thành ma trận NxM
    - Tính giá trị bit parity của từng dòng, từng cột

# Kiểm tra Parity

- Parity 2 chiều

			row parity	
			$d_{1,j+1}$	$d_{2,j+1}$
			$\dots$	$\dots$
$d_{1,1}$			$d_{1,j}$	$d_{1, j+1}$
$d_{2,1}$			$d_{2,j}$	$d_{2, j+1}$
$\dots$			$\dots$	$\dots$
$d_{i,1}$			$d_{i,j}$	$d_{i, j+1}$
$d_{i+1,1}$			$d_{i+1,j}$	$d_{i+1, j+1}$

column parity

↓

# Kiểm tra Parity

- Parity 2 chiều
  - Ví dụ:
    - Dùng parity chẵn
    - $N = 3, M = 5$
    - Dữ liệu cần gửi đi: 10101 11110 01110

1	0	1	0	1	1	
1	1	1	1	0	0	
0	1	1	1	0	1	
<hr/>						
0	0	1	0	1	0	

# Kiểm tra Parity

- Parity 2 chiều
  - Bên nhận
    - Biểu diễn dữ liệu nhận thành ma trận  $(N+1) \times (M+1)$
    - Kiểm tra tính đúng đắn của từng dòng/cột
    - Đánh dấu các dòng/cột dữ liệu bị lỗi
    - Bit lỗi: bit tại vị trí giao giữa dòng và cột bị lỗi
  - Ví dụ: Dùng parity chẵn;  $N = 3, M = 5$

# Kiểm tra Parity

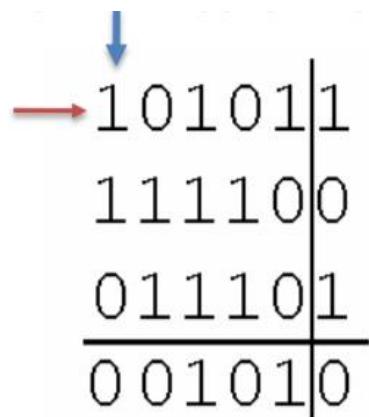
- Parity 2 chiều

- Ví dụ:

- Dùng parity chẵn; N = 3, M = 5

Dữ liệu nhận:

101011 111100 011101 001010

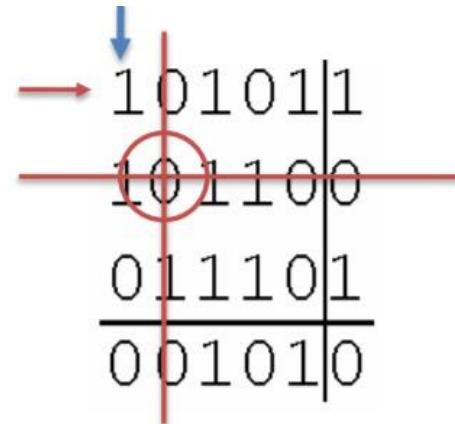


Không có lỗi

Dữ liệu thật: 10101 11110 01110

Dữ liệu nhận:

101011 101100 011101 001010



Có lỗi

Dữ liệu thật: 10101 11110 01110

# Kiểm tra Parity

- Mã Hamming
  - Đặc điểm:
    - Sửa lỗi 1 bit
    - Nhận dạng được 2 bit lỗi
    - Sửa lỗi nhanh hơn Parity code 2 chiều

# Kiểm tra Parity

- Mã Hamming
  - Redundant bits or parity bits
    - Các bit được bổ sung thêm vào với các bit dữ liệu được truyền để phát hiện và sửa lỗi.
    - Thêm bao nhiêu?

$$2^r \geq m + r + 1 \quad (r = \text{redundant bit}, m = \text{data bit})$$

Ví dụ:

$$\begin{aligned} &+ \text{data bits} = 7 \Rightarrow \text{redundant bits: } 2^4 \geq 7 + 4 + 1 \\ &\Rightarrow \text{redundant bits} = 4 \end{aligned}$$

# Kiểm tra Parity

- Mã Hamming
  - Redundant bits or parity bits
    - There are two types of parity bits
      - Even parity bit
        - » In the case of even parity, for a given set of bits, the **number of 1's** are counted. If that count is **odd**, the **parity bit** value is set to **1**, making the total count of occurrences of 1's an even number. If the total number of 1's in a given set of bits is already **even**, the parity bit's value is **0**.
        - Odd Parity bit
          - » In the case of odd parity, for a given set of bits, the number of 1's are counted. If that count is **even**, the parity bit value is set to **1**, making the total count of occurrences of 1's an odd number. If the total number of 1's in a given set of bits is already **odd**, the parity bit's value is **0**.

# Kiểm tra Parity

- Mã Hamming
  - General Algorithm of Hamming code
    - Có M bit, đánh số từ 1 đến M và biểu diễn M dưới dạng nhị phân
    - Bit parity ( $\log_2 M$  bits) được đặt tại các vị trí lũy thừa của 2
    - Bit dữ liệu được đặt tại các vị trí không là lũy thừa của 2
    - Tính Parity bit

# Kiểm tra Parity

- Mã Hamming
  - General Algorithm of Hamming code
    - Tính Parity bit
      - **Parity bit 1** covers all the bits positions whose binary representation includes a 1 in the least significant
      - **Parity bit 2** covers all the bits positions whose binary representation includes a 1 in the second position from the least significant bit.
      - **Parity bit 4** covers all the bits positions whose binary representation includes a 1 in the third position from the least significant bit..

# Kiểm tra Parity

- Mã Hamming
  - General Algorithm of Hamming code
    - Ví dụ truyền dữ liệu 1011001, dùng parity chẵn
      - Đặt các Parity bit tại các vị trí lũy thừa của 2 (1, 2, 4, 8)

11	10	9	8	7	6	5	4	3	2	1
1	0	1	R8	1	0	0	R4	1	R2	R1

# Kiểm tra Parity

- Mã Hamming
  - General Algorithm of Hamming code
    - Ví dụ truyền dữ liệu 1011001, dùng party chẵn
      - Tính Parity bit:
        - » R1: **3, 5, 7, 9, 11 => R1 = 0**

11	10	9	8	7	6	5	4	3	2	1
1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
1	0	1	R8	1	0	0	R4	1	R2	R1

» R2: **3, 6, 7, 10, 11 => R2 = 1**

11	10	9	8	7	6	5	4	3	2	1
1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
1	0	1	R8	1	0	0	R4	1	R2	R1

# Kiểm tra Parity

- Mã Hamming
  - General Algorithm of Hamming code
    - Ví dụ truyền dữ liệu 1011001, dùng party chẵn
      - Tính Parity bit:
        - » R4: **5, 6, 7 => R4 = 1**

11	10	9	8	7	6	5	4	3	2	1
1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
1	0	1	R8	1	0	0	R4	1	R2	R1

» R8: **9, 10, 11 => R8 = 0**

11	10	9	8	7	6	5	4	3	2	1
1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
1	0	1	R8	1	0	0	R4	1	R2	R1

# Kiểm tra Parity

- Mã Hamming
  - General Algorithm of Hamming code
    - Ví dụ truyền dữ liệu 1011001, dùng parity chẵn
      - Đặt các Parity bit tại các vị trí lũy thừa của 2 (1, 2, 4, 8)

11	10	9	8	7	6	5	4	3	2	1
1	0	1	R8	1	0	0	R4	1	R2	R1

- Dữ liệu được gửi đi là

11	10	9	8	7	6	5	4	3	2	1
1	0	1	0	1	0	0	1	1	1	0

# Kiểm tra Parity

- Mã Hamming
  - Phát hiện và sửa lỗi
    - Dữ liệu truyền đi

11	10	9	8	7	6	5	4	3	2	1
1	0	1	0	1	0	0	1	1	1	0

- Dữ liệu nhận được bị lỗi bít số 6

11	10	9	8	7	6	5	4	3	2	1
1	0	1	0	1	1	0	1	1	1	0

# Kiểm tra Parity

- Mã Hamming
  - Phát hiện và sửa lỗi
    - Dữ liệu nhận được bị lỗi bít số 6

11	10	9	8	7	6	5	4	3	2	1
1	0	1	0	1	1	0	1	1	1	0

- Tính các Parity bits từ dữ liệu nhận được
  - R1: 3, 5, 7, 9, 11 => R1 = 0 R1 = 0
  - R2: 3, 6, 7, 10, 11 => R2 = **0** R2 = 1
  - R4: 5, 6, 7 => R4 = **0** R4 = 1
  - R8: 9, 10, 11 => R8 = 0 R8 = 0

# Check sum

- Bên gửi
  - d bits trong dữ liệu gửi đi được xem như gồm N số k bits:  $x_1, x_2, \dots, x_N$
  - Tính tổng  $X = x_1 + x_2 + \dots + x_N$
  - Tính bù 1 của X → giá trị checksum

Ví dụ:

Dữ liệu cần gửi đi: 1110 0110 0110 0110

1110

$\rightarrow k=4$

0110

$$\text{Tổng} = 1110 + 0110 + 0110 + 0110 = 0010$$

0100

$$\text{Checksum} = 1101$$

1

0101

# Check sum

- Bên nhận
  - Tính tổng cho tất cả giá trị nhận được (kể cả giá trị checksum)
  - Nếu tất cả các bít là 1, thì dữ liệu nhận được là đúng; ngược lại: có lỗi xảy ra

## Ví dụ

- Dữ liệu nhận được: 1110 0110 0110 0110 1101

Tổng = 1111 => Đúng

- Dữ liệu nhận được: 1010 0110 0110 0110 1101

Tổng = 1011 => Sai

## Internet checksum

# Kiểm tra dư thừa theo chu kỳ

- (Cyclic redundancy check - CRC)
  - Phương pháp mã đã thúc hay mã vòng
  - Thông tin kiểm tra lỗi được gọi là checksum, được tính bằng cách dùng đa thức sinh  $G$ 
    - $G$  được quy ước dưới dạng nhị phân (hệ số của nó chỉ có giá trị 1 hoặc 0 tương ứng với các chữ số trong dãy bít)
    - $G = x^7 + x^6 + x^5 + x^2 + 1 \Rightarrow$  dạng nhị phân của  $G$  là 11100101

# Kiểm tra dư thừa theo chu kỳ

- (Cyclic redundancy check - CRC)
  - Phương pháp
    - Đầu gửi
      - Coi tin gửi đi như một đa thức
      - Chia đa thức gửi đi cho đa thức sinh, phần dư được gắn vào cuối tin gửi đi rồi gửi sang đầu nhận.
    - Đầu nhận
      - Lấy dữ liệu nhận được chia cho đa thức sinh. Nếu số dư = 0 thì không có lỗi, nếu ≠ 0 thì có lỗi xảy ra.

# Kiểm tra dư thừa theo chu kỳ

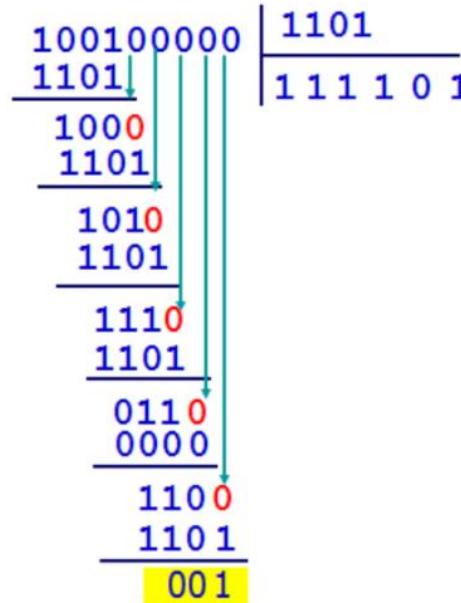
- (Cyclic redundancy check - CRC)
  - Cách tạo phần dư CRC
    - G là đa thức sinh bậc n, I là tin gửi
    - Thêm n bit 0 vào cuối chuỗi bit I được đa thức nhị phân P.
    - Chia đa thức P cho G theo quy tắc của phép trừ không nhớ. Phần dư R của phép chia được thay thế cho n bit 0 bổ sung trong P để được đa thức D là dãy bit được gửi đi thay cho I ( $D=P+R$ )
      - Theo quy tắc của phép chia đa thức nhị phân: Nếu  $(P-R)$  chia hết  $G$  thì  $(P+R)$  cũng chia hết cho  $G$

# Kiểm tra dư thừa theo chu kỳ

- (Cyclic redundancy check - CRC)
  - Cách tạo phần dư CRC
    - Dãy bit nhận được  $D'$  sẽ đem chia cho  $G$ . Nếu phép chia không có dư bên nhận sẽ xóa phần CRC đi và dữ liệu được chấp nhận. Nếu phép chia có dư thì dữ liệu có lỗi và bị yêu cầu truyền lại.

# Kiểm tra dư thừa theo chu kỳ

- (Cyclic redundancy check - CRC)
  - Cách tạo phần dư CRC
    - Ví dụ
      - Dãy bit truyền đi  $I = 100100$
      - Đa thức sinh  $G = x^3 + x^2 + 1$  (tương ứng chuỗi bit 1101)
      - Đa thức nhị phân:  $P = 100100\textcolor{red}{000}$
      - Phần dư CRC



# Nội dung

- Giới thiệu, các dịch vụ
- Phát hiện và sửa lỗi
- **Các giao thức đa truy nhập**
- Các mạng LAN
  - Định địa chỉ, ARP
  - Ethernet
  - Các switch
  - Các VLAN
- Chuyển mạch nhãn đa giao thức (MPLS)
- Mạng trung tâm dữ liệu
- Vòng đời của một yêu cầu web

# Các giao thức và các liên kết đa truy nhập

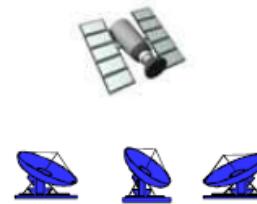
- Có hai loại “liên kết”:
  - Điểm-nối-điểm (Point-to-point)
    - PPP cho truy nhập dial-up
    - Liên kết point-to-point giữa các host, switch Ethernet
  - Quảng bá (broadcast) (chia sẻ đường truyền chung)
    - Ethernet mô hình cũ
    - upstream HFC
    - 802.11 wireless LAN (LAN không dây)



Chia sẻ đường truyền  
(Ví dụ: cabled Ethernet)



Chia sẻ RF  
(Ví dụ: 802.11 WiFi)



Chia sẻ RF  
(vệ tinh)



Con người tại bữa tiệc  
cocktail (chia sẻ không  
khí, âm thanh)

# Các giao thức đa truy nhập

- Kênh quảng bá (broadcast) được chia sẻ
- Hai hoặc nhiều nút muốn truyền: giao thoa
  - Tranh chấp (đụng độ, collision) xảy ra khi nút nhận được hai hay nhiều tín hiệu tại cùng một thời điểm.

## Giao thức đa truy nhập

- Giải thuật phân quyền xác định cách các nút chia sẻ kênh truyền, ví dụ: xác định khi nào nút có thể được truyền.
- Truyền thông về chia sẻ kênh phải dùng chính kênh đó!
  - Không có kênh riêng để điều phối

# Một giao thức đa truy nhập lý tưởng

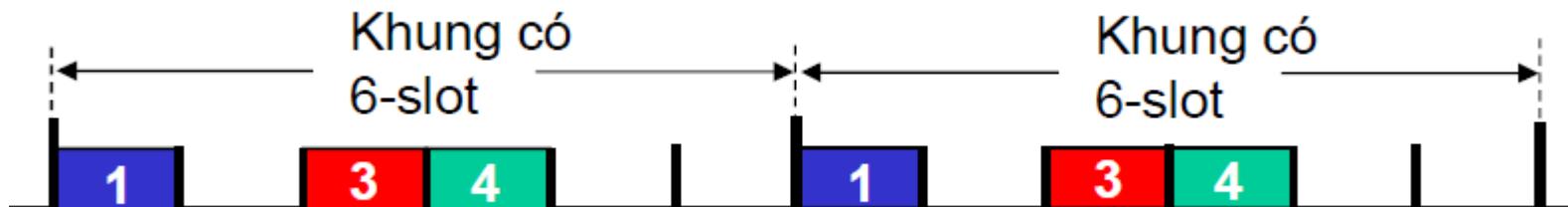
- **Cho:** Kênh quảng bá có tốc độ  $R$  bps
- **Mong muốn:**
  1. Khi một nút muốn truyền, nó có thể gửi đi với tốc độ  $R$ .
  2. Khi  $M$  nút muốn truyền, mỗi nút có thể gửi đi với tốc độ trung bình là  $R/M$ .
  3. Phân quyền hoàn toàn:
    - Không có nút đặc biệt cho việc điều phối truyền
    - Không có các khe (slot) hay đồng hồ đồng bộ
  4. Đơn giản

# Các giao thức MAC: Phân loại

- Gồm 3 loại chính:
  - **Phân chia kênh** (Channel Partitioning Protocols)
    - Chia kênh thành các “phần” nhỏ hơn (khe thời gian, tần số, mã)
    - Cấp phát các phần cho các nút để dùng riêng
  - **Truy nhập ngẫu nhiên**
    - Kênh không được phân chia, cho phép tranh chấp
    - “Giải quyết” các tranh chấp
  - **Xoay vòng**
    - Các nút lần lượt xoay vòng, nhưng nút gửi nhiều hơn được nắm quyền truyền lâu hơn.

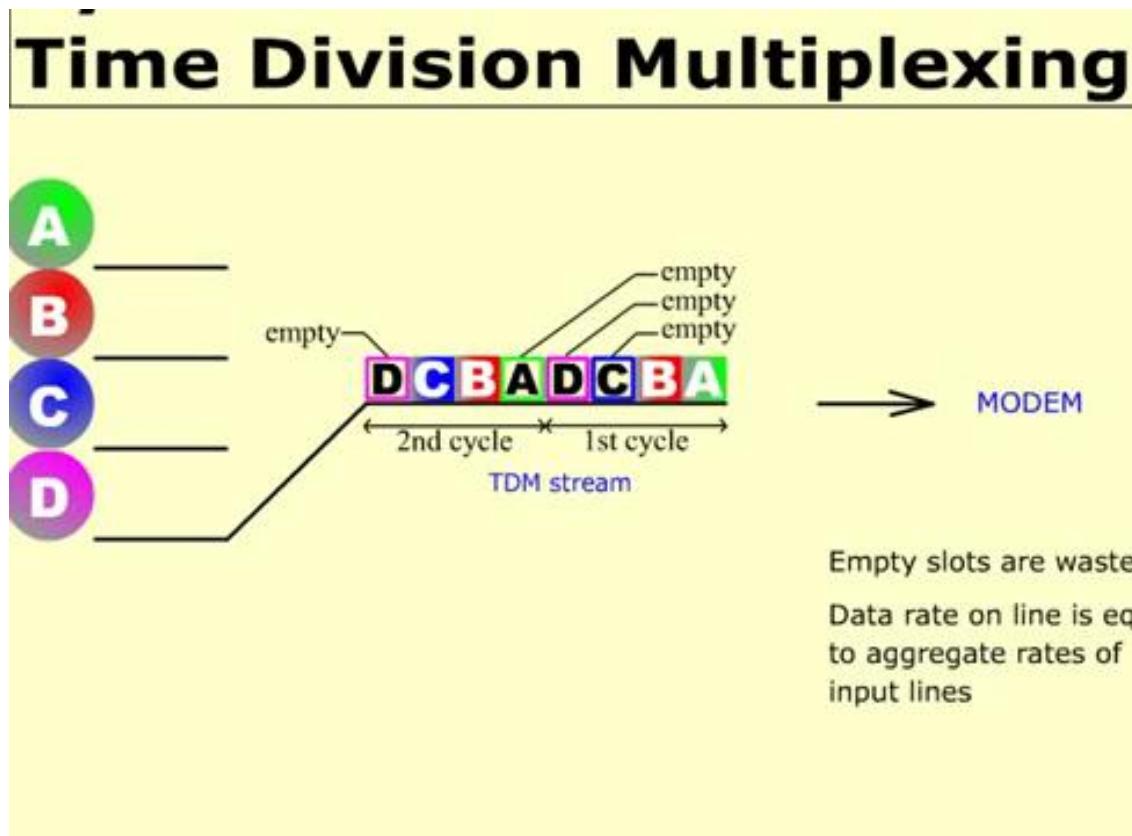
# Giao thức MAC phân chia kênh: TDMA

- TDMA: Đa truy nhập phân chia theo thời gian (time division multiple access)
  - Truy nhập tới kênh theo “các vòng”
  - Mỗi trạm có một khe thời gian (slot) có độ dài cố định (độ dài = thời gian truyền gói) trong mỗi vòng
  - Không được dùng các khe đang “rảnh” (không hoạt động - idle)
  - Ví dụ: LAN có 6 trạm, các trạm 1,3,4 có các gói tin, các khe 2,5,6 đang rảnh.



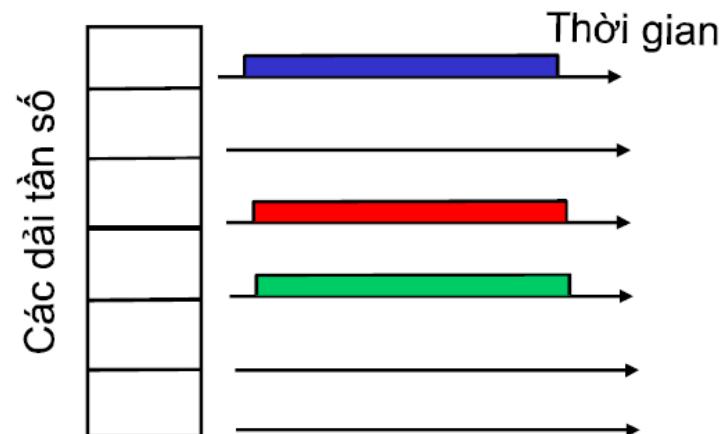
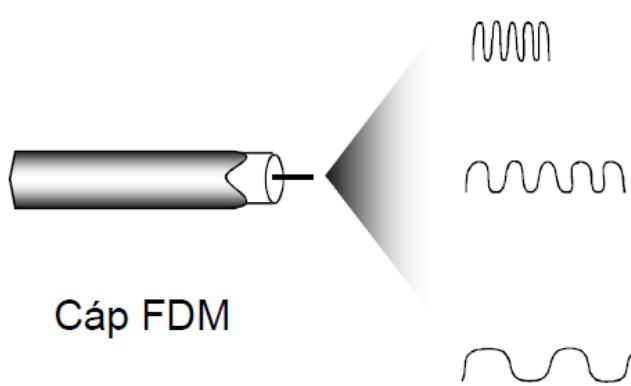
# Giao thức MAC phân chia kênh: TDMA

- TDMA: Đa truy nhập phân chia theo thời gian (time division multiple access)



# Giao thức MAC phân chia kênh: FDMA

- FDMA: Đa truy nhập phân chia theo tần số (frequency division multiple access)
  - Phổ kênh truyền được chia theo các dải tần số
  - Mỗi trạm được gán một dải tần số cố định
  - Trong lúc truyền không được dùng các dải tần số “rảnh” khác.
  - Ví dụ: LAN có 6 trạm, các trạm 1,3,4 có gói tin, các dải tần 2,5,6 đang rảnh



# Giao thức MAC phân chia kênh: CDMA

- Code division multiple access (CDMA)
  - TDM and FDM assign time slots and frequencies to the nodes
  - CDMA assigns a different code to each node
    - Each node then uses its unique code to encode the data bits it sends
      - If the codes are chosen carefully, CDMA networks have the wonderful property that different nodes can transmit simultaneously and yet have their respective receivers correctly receive a sender's encoded data bits (assuming the receiver knows the sender's code) in spite of interfering transmissions by other nodes

# Giao thức MAC phân chia kênh: CDMA

- Code division multiple access (CDMA)
  - CDMA has been used in military systems for some time (due to its anti-jamming properties) and now has widespread civilian use, particularly in cellular telephony.

# Giao thức truy nhập ngẫu nhiên

- Đặc điểm chung
  - Node luôn truyền với tốc độ của kênh truyền, R bps.
  - When there is a collision, each node involved in the collision repeatedly retransmits its frame (that is, packet) until its frame gets through without a collision.
    - But when a node experiences a collision, it doesn't necessarily retransmit the frame right away. Instead it waits a random delay before retransmitting the frame.

# Giao thức truy nhập ngẫu nhiên

- Đặc điểm chung
  - Each node involved in a collision chooses independent random delays.
    - Because the random delays are independently chosen, it is possible that one of the nodes will pick a delay that is sufficiently less than the delays of the other colliding nodes and will therefore be able to sneak its frame into the channel without a collision.

# Giao thức truy nhập ngẫu nhiên

- Các ví dụ của giao thức MAC truy nhập ngẫu nhiên:
  - Chia slot ALOHA
  - ALOHA thuần nhất
  - CSMA, CSMA/CD, CSMA/CA

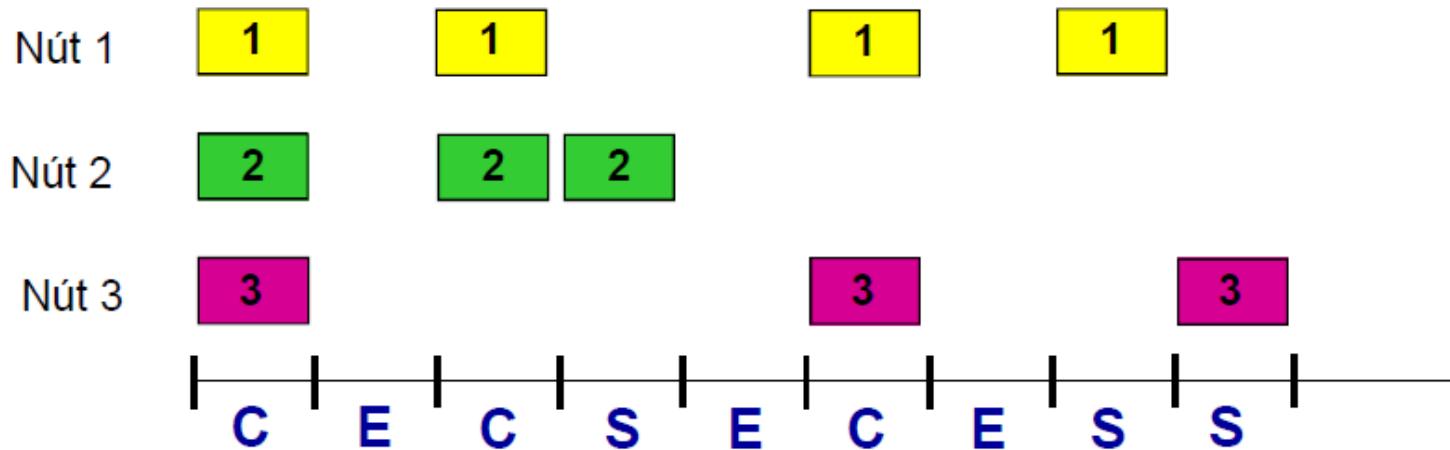
# Chia Slot ALOHA

- Giả thiết:
  - Tất cả các frame có cùng kích thước
  - Thời gian được chia thành các slot có kích thước bằng nhau (thời gian đủ để truyền 1 frame)
  - Các nút bắt đầu truyền chỉ khi slot bắt đầu
  - Các nút được đồng bộ hóa
  - Nếu có 2 hoặc nhiều nút truyền trong một slot, thì tất cả các nút đều phát hiện có tranh chấp.

# Chia Slot ALOHA

- **Hoạt động:**
  - Khi nút có một khung mới, nó được phép truyền trong slot tiếp theo.
  - Nếu không có tranh chấp: nút có thể gửi khung mới trong slot kế tiếp
  - Nếu có tranh chấp: nút truyền lại frame trong mỗi slot kế tiếp với xác xuất bằng  $p$  cho đến khi thành công

# Chia Slot ALOHA



## Ưu điểm:

- Nút kích hoạt có thể truyền liên tục với tốc độ tối đa của kênh.
- Phân quyền cao: chỉ các slot trong các nút cần được đồng bộ
- Đơn giản

## Nhược điểm:

- Có tranh chấp,
- Lãng phí các slot không hoạt động
- Các nút có thể phát hiện tranh chấp với thời gian ít hơn truyền gói
- Cần đồng hồ đồng bộ hóa

# Chia slot ALOHA: Hiệu suất

**Hiệu suất:** là phần slot truyền thành công trong số nhiều frame cần truyền của nhiều nút.

- Giả sử: N nút với nhiều frame cần truyền, mỗi cuộc truyền trong slot có xác suất là  $p$ .
- Xác suất để một nút truyền trong một slot thành công là  $p(1-p)^{N-1}$
- Xác suất để một nút bất kỳ truyền thành công là  $Np(1-p)^{N-1}$

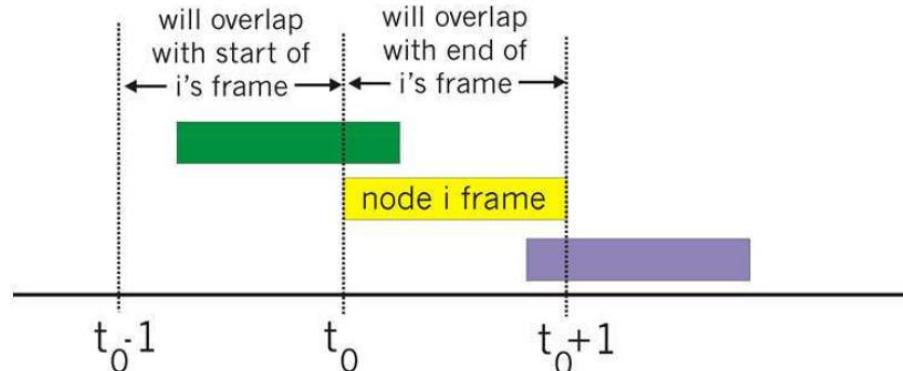
- Để tìm hiệu suất tối đa: tìm  $p^*$  để  $Np(1-p)^{N-1}$  đạt giá trị lớn nhất
- Với nhiều nút, tìm giới hạn của  $Np^*(1-p^*)^{N-1}$  khi N tiến đến vô cùng:

$$\text{Hiệu suất tối đa} = 1/e = .37$$

**Tốt nhất:** kênh hữu dụng truyền trong khoảng 37% thời gian!

# ALOHA thuần nhất (không chia slot)

- Aloha không chia slot: đơn giản hơn, không đồng bộ
- Khi frame đầu tiên đi đến
  - Truyền đi ngay lập tức
- Khả năng tranh chấp tăng lên:
  - Frame gửi tại thời điểm  $t_0$  xung đột với các frame khác được gửi trong thời điểm  $[t_0-1, t_0+1]$



# Hiệu suất của ALOHA thuần nhất

$P(\text{thành công của một nút}) = P(\text{nút truyền}) \cdot$

$P(\text{không có nút nào khác truyền trong } [t_0-1, t_0])$

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

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

... chọn  $p$  tối ưu sau đó cho  $n \rightarrow \infty$

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

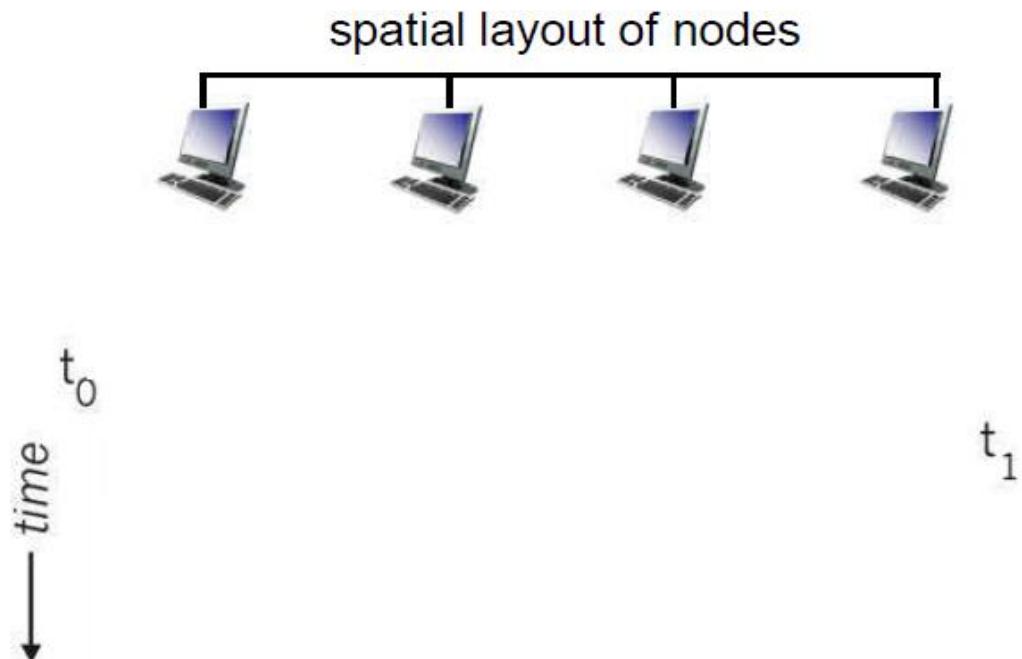
Thậm chí hiệu suất còn kém hơn chia slot Aloha!

# Đa truy nhập sóng mang CSMA

- **CSMA:** nghe trước khi truyền:
  - Nếu kênh truyền đang rảnh: truyền toàn bộ frame
  - Nếu kênh truyền đang bận, trì hoãn việc truyền
  - Tương tự với giao tiếp của con người: không ngắt lời khi người khác đang nói!

# Các tranh chấp trong CSMA

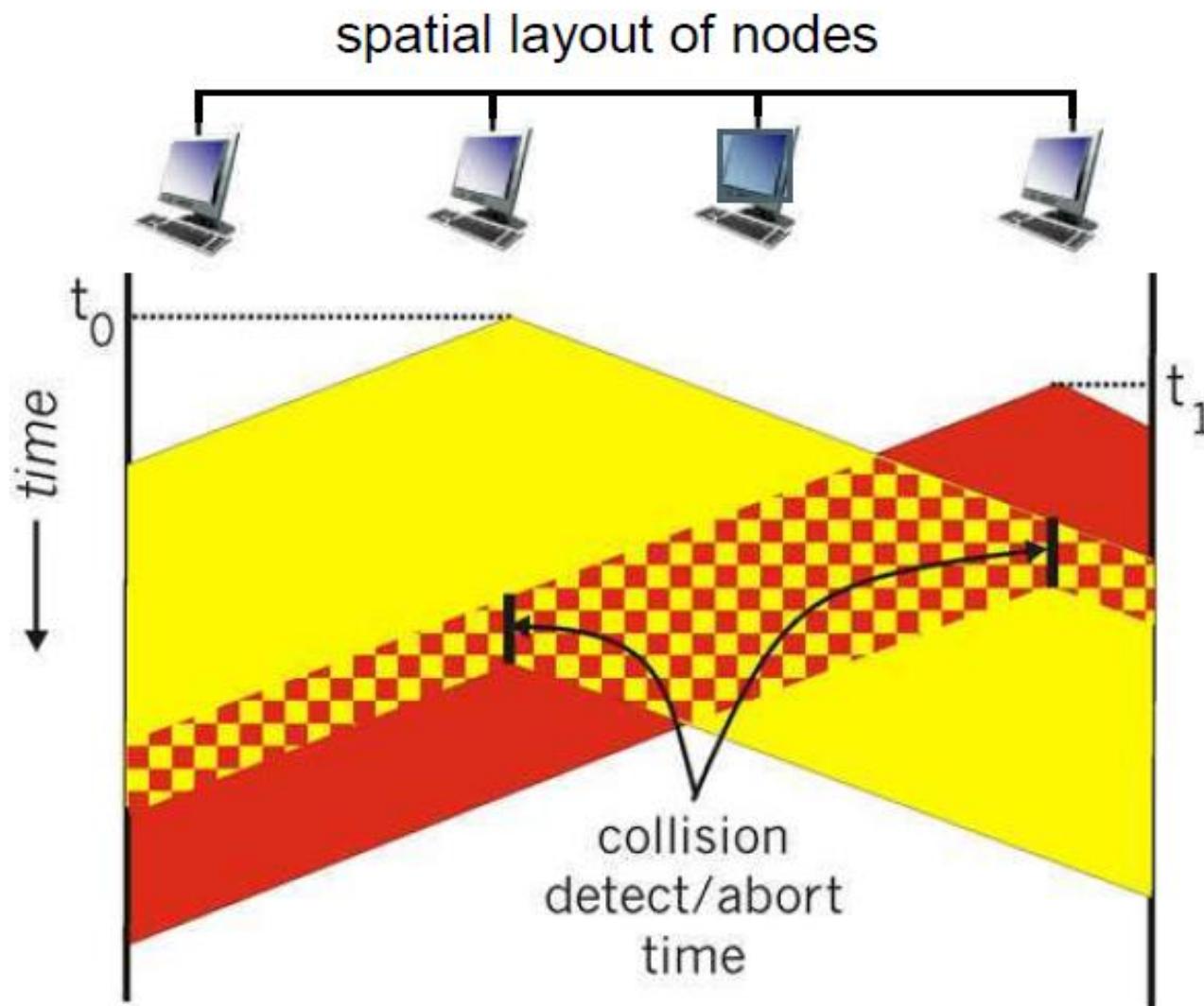
- **Tranh chấp vẫn có thể xảy ra:** do trễ lan truyền, nghĩa là hai nút không “nghe” thấy việc truyền của nhau.
- **Tranh chấp:** toàn bộ thời gian truyền gói tin bị lãng phí
  - Chú ý vai trò của khoảng cách và trễ lan truyền trong việc xác định khả năng có tranh chấp.



# Đa truy nhập sóng mang có phát hiện tranh chấp CSMA/CD (collision detection)

- CSMA/CD: trì hoãn như trong CSMA
  - Tranh chấp được phát hiện trong thời gian ngắn
  - Tranh chấp đường truyền được bỏ qua, giảm sự lãng phí kênh truyền
- Phát hiện tranh chấp:
  - Dễ dàng trong các mạng LAN không dây: đo cường độ tín hiệu, so sánh với các tín hiệu đã truyền, đã nhận.
  - Khó khăn trong các mạng LAN có dây: cường độ tín hiệu nhận được bị áp đảo bởi cường độ truyền cục bộ
  - Tương tự với con người: đàm thoại lịch sự

# Đa truy nhập sóng mang có phát hiện tranh chấp CSMA/CD (collision detection)



# Giải thuật CSMA/CD trong Ethernet

1. NIC nhận datagram từ tầng mạng, tạo ra frame
2. Nếu NIC nhận thấy kênh truyền đang rảnh, sẽ bắt đầu truyền frame. Nếu NIC nhận thấy kênh truyền đang bận, sẽ đợi cho đến khi kênh truyền rảnh thì mới truyền.
3. Nếu NIC truyền toàn bộ frame mà không phát hiện thấy bất kỳ cuộc truyền nào khác, thì NIC sẽ hoàn thành việc truyền frame!

# Giải thuật CSMA/CD trong Ethernet

4. Nếu NIC phát hiện thấy có cuộc truyền khác trong khi đang truyền thì sẽ hủy bỏ và không gửi tín hiệu nữa
5. Sau khi hủy bỏ, NIC thực hiện **quay trở lại theo cơ chế (mũ) nhị phân**:
  - Sau tranh chấp thứ m, NIC chọn K ngẫu nhiên trong  $\{0, 1, 2, \dots, 2^{m-1}\}$ . NIC chờ trong khoảng thời gian  $K \cdot 512$  bit, quay trở lại bước 2.
  - Khoảng thời gian chờ quay trở lại sẽ lâu hơn nếu có nhiều tranh chấp hơn.

# Hiệu suất CSMA/CD

- $T_{prop}$  = trễ lan truyền lớn nhất giữa 2 nút trong LAN
- $t_{trans}$  = thời gian truyền frame có kích thước lớn nhất

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

- Hiệu suất sẽ tiến đến 1
  - Khi  $t_{prop}$  tiến đến 0
  - Khi  $t_{trans}$  tiến đến vô cùng
- Hiệu suất tốt hơn ALOHA: và đơn giản, chi phí thấp và phân quyền!

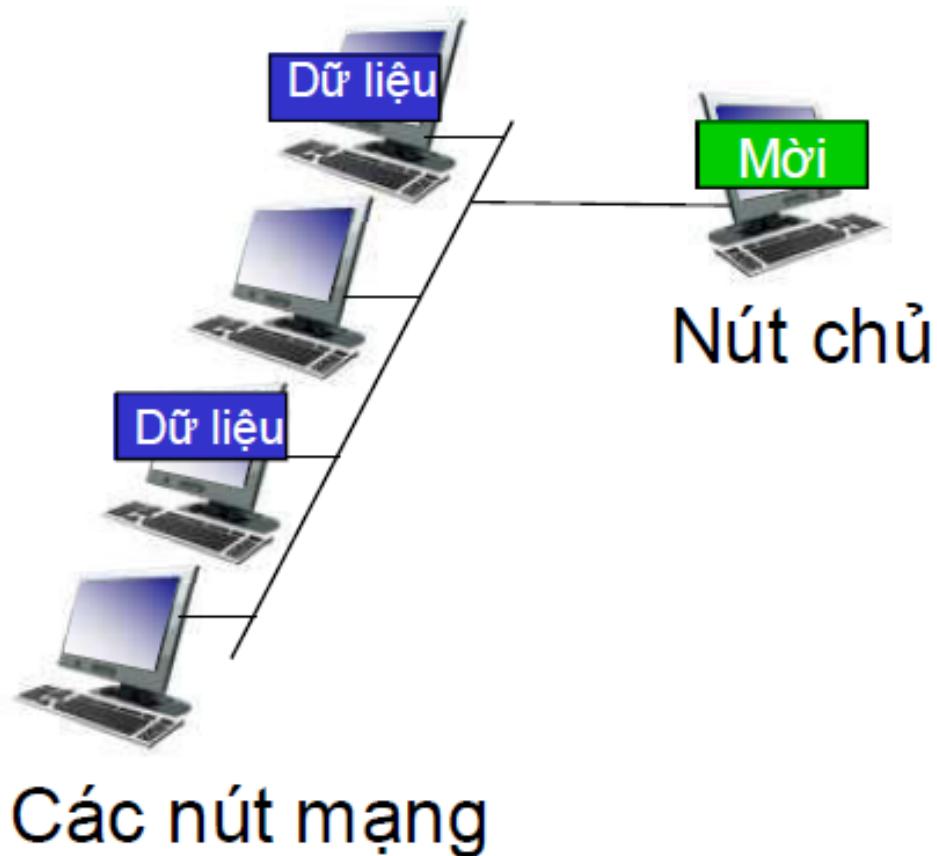
# Giao thức MAC “xoay vòng”

- Các giao thức MAC phân chia kênh:
  - Chia sẻ kênh hiệu quả và công bằng với tải cao
  - Không hiệu quả với tải thấp: trễ trong việc tiếp cận kênh, băng thông được cấp phát bằng  $1/N$  ngay cả khi chỉ có 1 nút cần truyền!
- Các giao thức MAC truy nhập ngẫu nhiên:
  - Hiệu quả với tải thấp: một nút có thể dùng hết khả năng của kênh
  - Với tải cao: có tranh chấp.
- Các giao thức MAC “xoay vòng”
  - Tìm kiếm giải pháp tốt nhất!

# Giao thức MAC “xoay vòng”

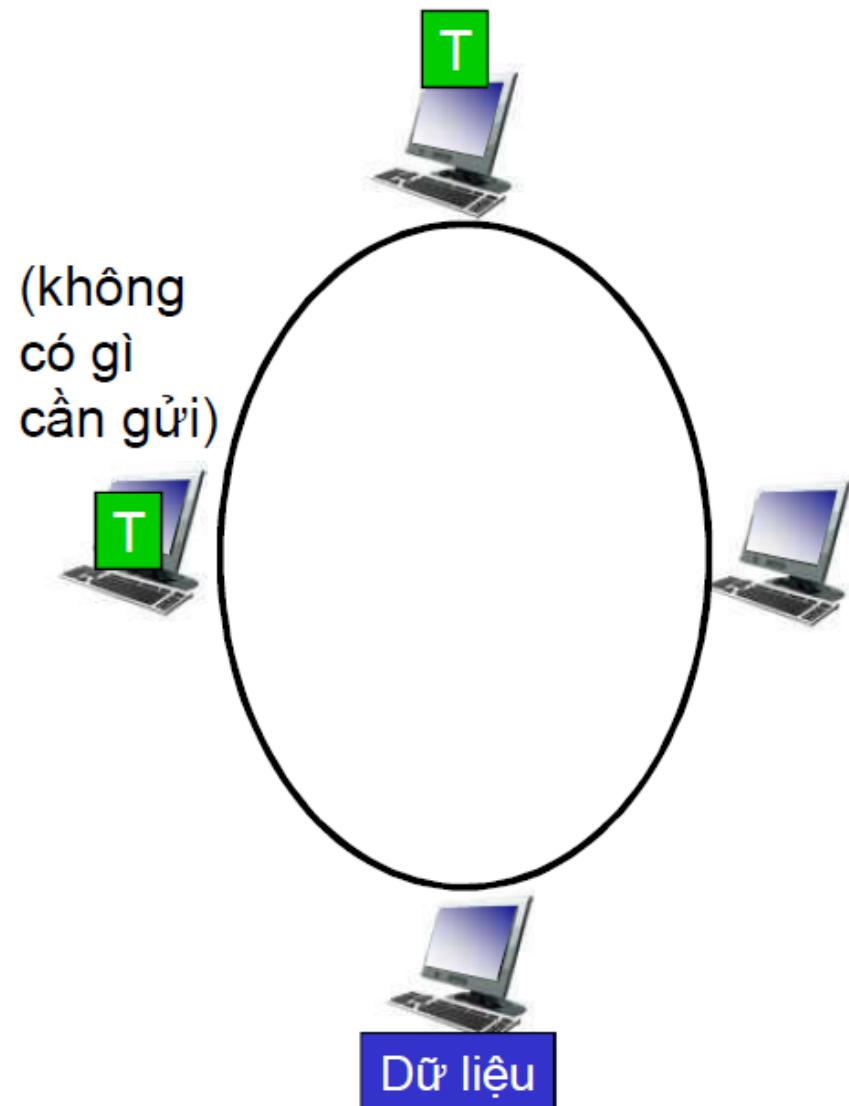
- **Mời tuần tự:**

- Nút chủ “mời” các nút truyền theo lượt tuần tự.
- Liên quan:
  - Việc mời truyền
  - Độ trễ
  - Một điểm chịu lỗi (Nút chủ)



# Giao thức MAC “xoay vòng”

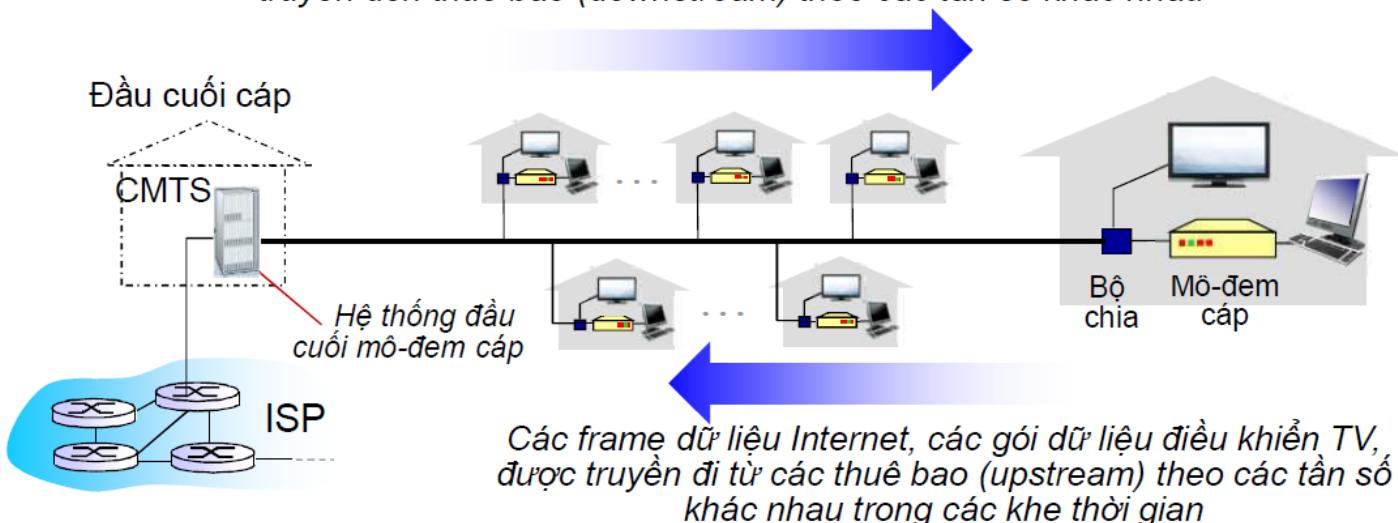
- Chuyển thẻ bài (token):
  - Điều khiển thẻ bài chuyển tuần tự từ một nút đến nút kế tiếp.
  - Thông điệp thẻ bài
  - Liên quan:
    - Thẻ bài
    - Độ trễ
    - Một điểm chịu lỗi (thẻ bài)



# Mạng truy nhập cáp

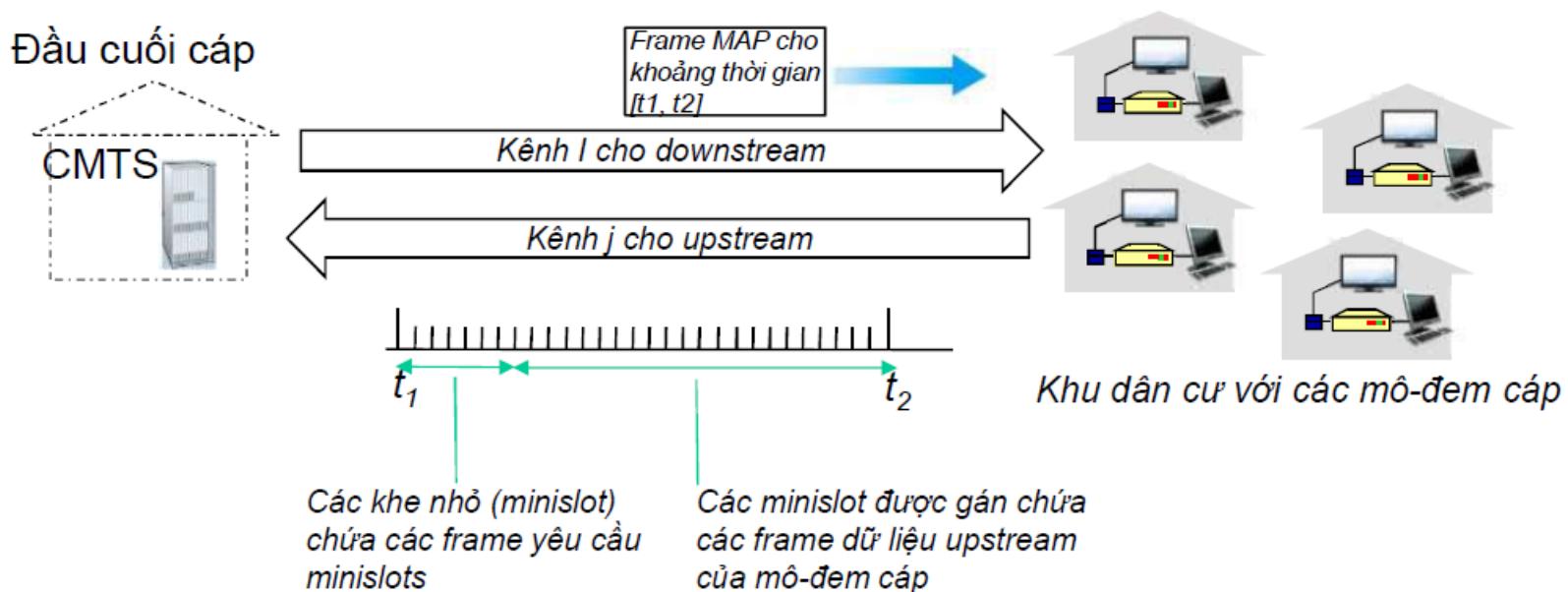
- **Nhiều** kênh downstream (broadcast) 40 Mbps
  - Từ CMTS đơn truyền vào trong các kênh
- **Nhiều** kênh upstream 30 Mbps
  - **Đa truy nhập:** Tất cả người dùng đều có thể tranh kênh upstream nào đó trong các khe thời gian (mà những người khác đã được gán).

Các frame Internet, các kênh TV hay gói dữ liệu điều khiển được truyền đến thuê bao (downstream) theo các tần số khác nhau



# Mạng truy nhập cáp

- **DOCSIS:** chuẩn dữ liệu tốc độ cao cho hệ thống cáp (data over cable service interface specifications).
  - FDM trên các kênh tần số upstream, downstream
  - TDM upstream: một số slot được gán, một số có tranh chấp
    - Các khung MAP downstream: gán các slot upstream
    - Yêu cầu cho các slot upstream (và dữ liệu) được truyền truy cập ngẫu nhiên trong các khe thời gian đã được chọn



# Tổng kết về các giao thức MAC

- **Phân chia kênh**, theo thời gian, tần số hoặc mã
  - TDMA, FDMA
- **Truy nhập ngẫu nhiên (động)**,
  - ALOHA, S-ALOHA, CSMA, CSMA/CD
  - Sóng mang: dễ dàng trong một số công nghệ (có dây), khó khăn trong một số khác (không dây)
  - CSMA/CD được dùng trong Ethernet
  - CSMA/CA được dùng trong 802.11
- **Xoay vòng**
  - Trạm trung tâm mời các trạm truyền, chuyển thẻ bài
  - Bluetooth, FDDI, token ring

# Nội dung

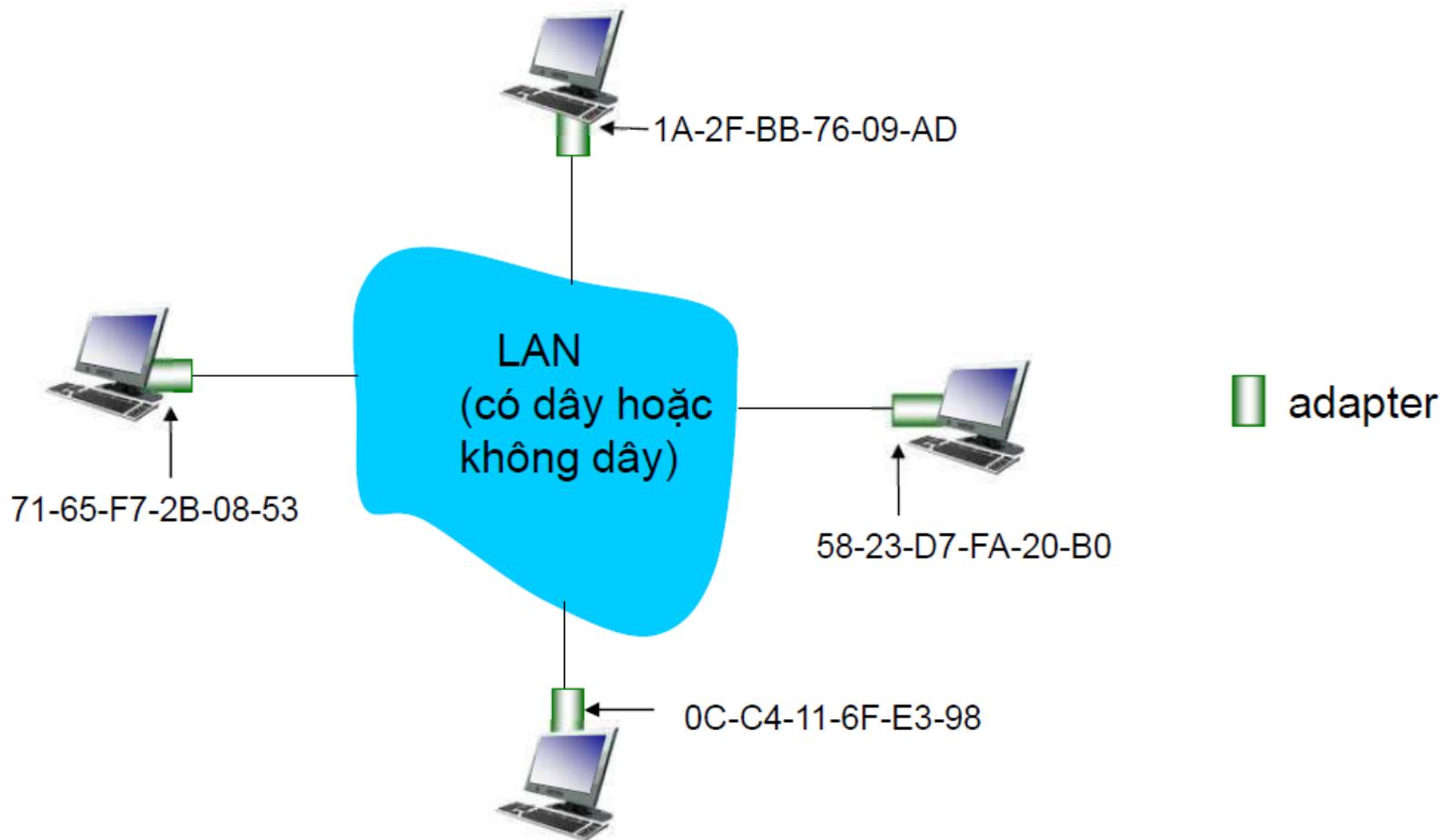
- Giới thiệu, các dịch vụ
- Phát hiện và sửa lỗi
- Các giao thức đa truy nhập
- Các mạng LAN
  - Định địa chỉ, ARP
  - Ethernet
  - Các switch
  - Các VLAN
- Chuyển mạch nhãn đa giao thức (MPLS)
- Mạng trung tâm dữ liệu
- Vòng đời của một yêu cầu web

# Địa chỉ MAC và ARP

- Địa chỉ IP 32-bit:
  - Địa chỉ tầng mạng cho giao diện
  - Được dùng cho việc chuyển tiếp gói tin tại tầng 3 (tầng mạng)
- Địa chỉ MAC (hoặc LAN/vật lý/Ethernet):
  - Chức năng: **được dùng “cục bộ” để lấy frame từ một giao diện với một giao diện được kết nối vật lý khác (cùng mạng)**
  - Địa chỉ MAC có 48 bit (cho hầu hết các LAN) được ghi sẵn trong bộ nhớ ROM của NIC, (đôi khi cũng được thiết lập bởi phần mềm)
  - Ví dụ: 1A-2F-BB-76-09-AD

# Địa chỉ LAN và ARP

- Mỗi adapter trên LAN có duy nhất một địa chỉ **LAN**



# Địa chỉ LAN

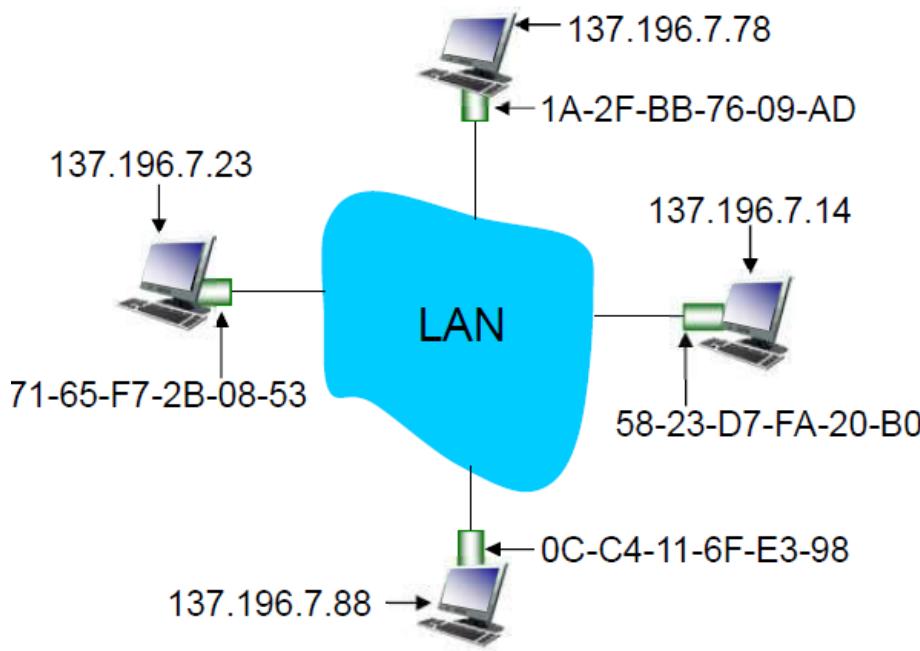
- Việc cấp phát địa chỉ MAC được quản lý bởi IEEE
- Nhà sản xuất mua phần không gian địa chỉ MAC (để đảm bảo là duy nhất)
- So sánh:
  - Địa chỉ MAC: như số chứng minh nhân dân
  - Địa chỉ IP: như số điện thoại

# Địa chỉ LAN

- Địa chỉ MAC phẳng → có thể di chuyển
  - Có thể chuyển card từ LAN này sang LAN khác
- Địa chỉ phân cấp IP không thể di chuyển
  - Địa chỉ IP phụ thuộc vào IP subnet mà nút được gắn vào

# ARP: address resolution protocol

- **Hỏi:** Làm thế nào để xác định địa chỉ MAC của một giao diện khi biết địa chỉ IP?



**Bảng ARP:** mỗi nút IP (host, router) trên LAN có một bảng ARP.

- Ánh xạ địa chỉ IP/MAC cho một số nút LAN:

**< địa chỉ IP; địa chỉ MAC; TTL>**

- TTL (Time To Live): thời gian sau đó ánh xạ địa chỉ sẽ bị hủy (thường là 20 phút)

# Giao thức ARP: cùng LAN

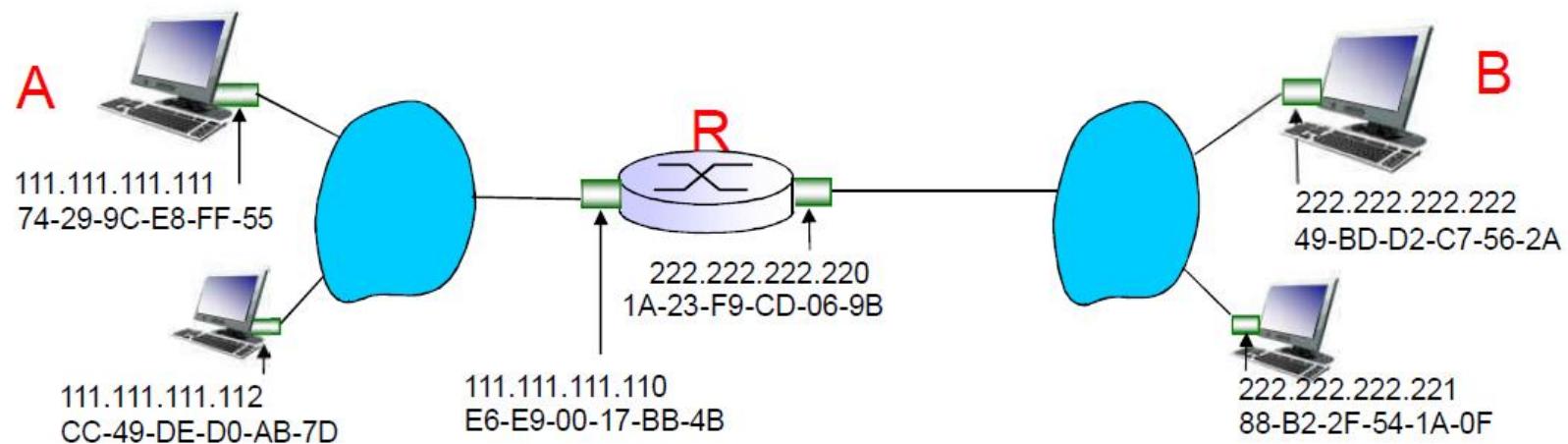
- A muốn gửi datagram tới B
  - Địa chỉ MAC của B không có trong bảng ARP của A.
- A **quảng bá (broadcasts)** gói tin truy vấn ARP, chứa địa chỉ IP của B
  - Địa chỉ MAC đích = FF-FFFF-FF-FF-FF
  - Tất cả các nút trên LAN đều nhận truy vấn ARP
- B nhận được gói tin ARP, sẽ trả lời A với địa chỉ MAC của mình.
  - Frame được gửi tới địa chỉ MAC của A (unicast)

# Giao thức ARP: cùng LAN

- A ghi lại cặp địa chỉ IP to-MAC trong bảng ARP của nó cho đến khi thông tin bị timeout.
  - Trạng thái mềm: thông tin này sẽ bị timeout trừ khi được làm mới lại.
- ARP là “plug-and-play”:
  - Các nút tạo ra bảng ARP của nó mà không cần bất kỳ sự can thiệp nào từ nhà quản trị mạng.

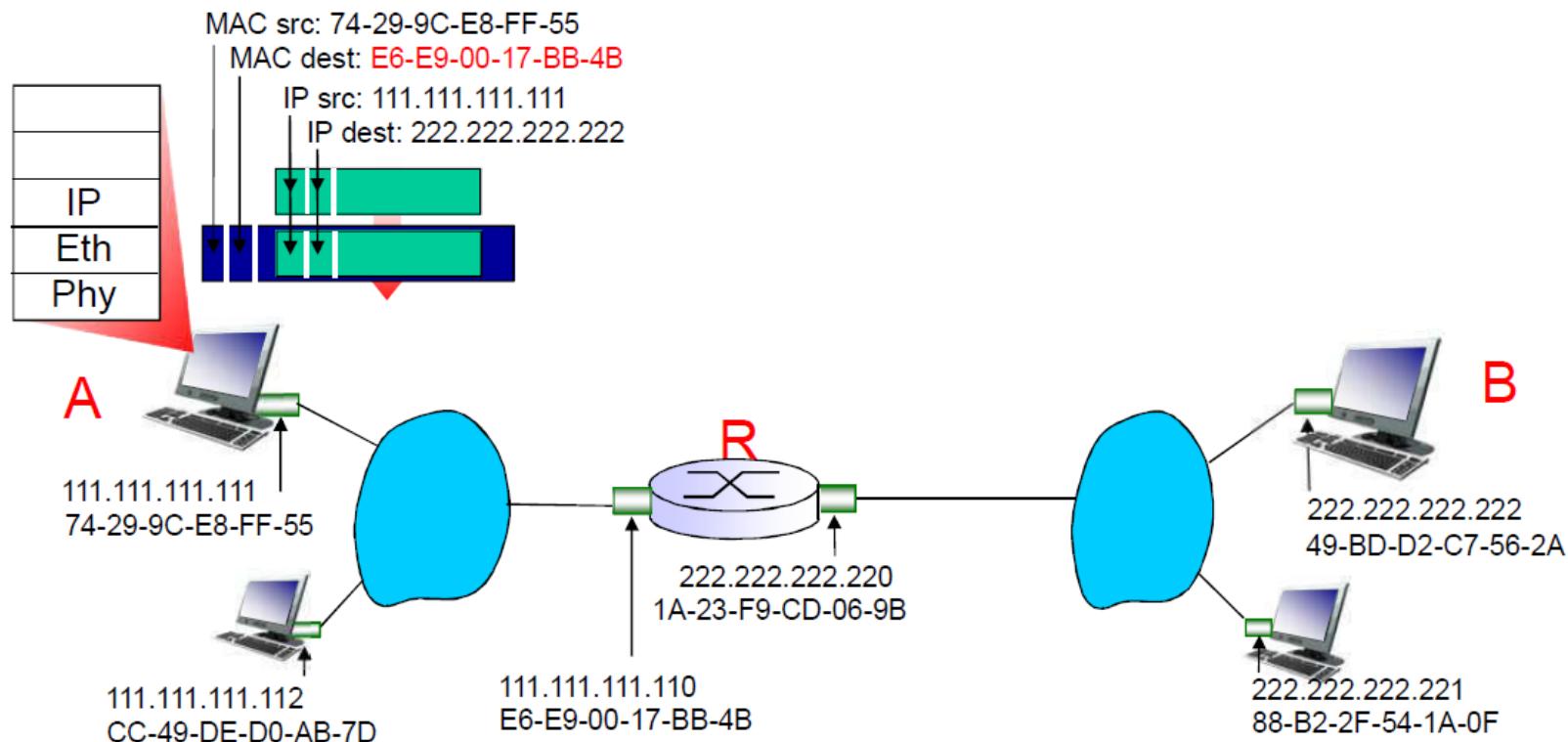
# Định địa chỉ: định tuyến tới LAN khác

- Tình huống: **gửi datagram từ A tới B qua R**
  - Tập trung vào định địa chỉ – tại IP (datagram) và tầng MAC (frame)
  - Giả thiết A biết địa chỉ IP của B
  - Giả thiết A biết địa chỉ IP của router hop đầu tiên, là R (thì như thế nào?)
  - Giả thiết A biết địa chỉ MAC của R (thì như thế nào?)



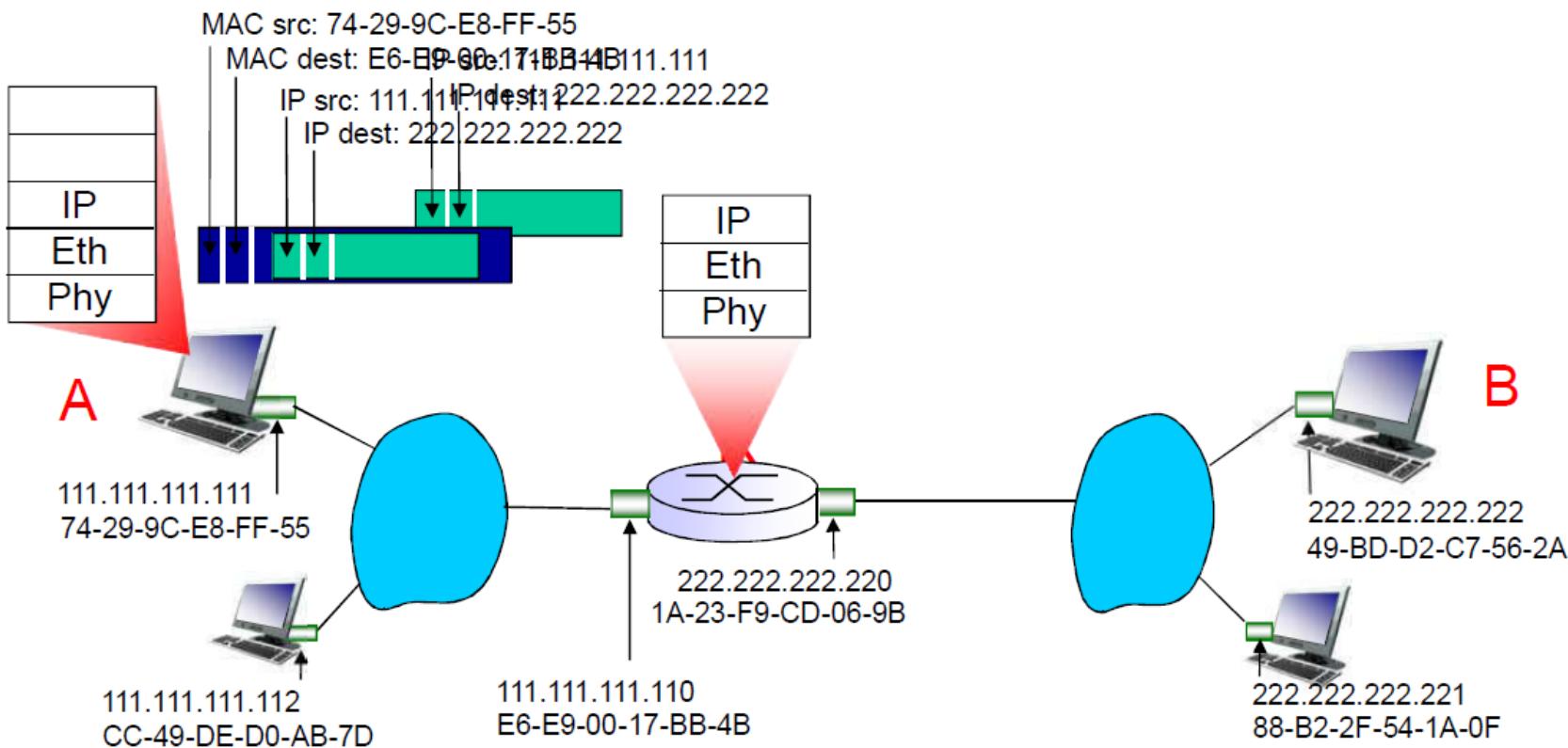
# Định địa chỉ: định tuyến tới LAN khác

- A tạo IP datagram với IP nguồn A, đích B
- A tạo frame tầng liên kết với địa chỉ MAC của R là đích, frame chứa IP datagram từ A-tới-B



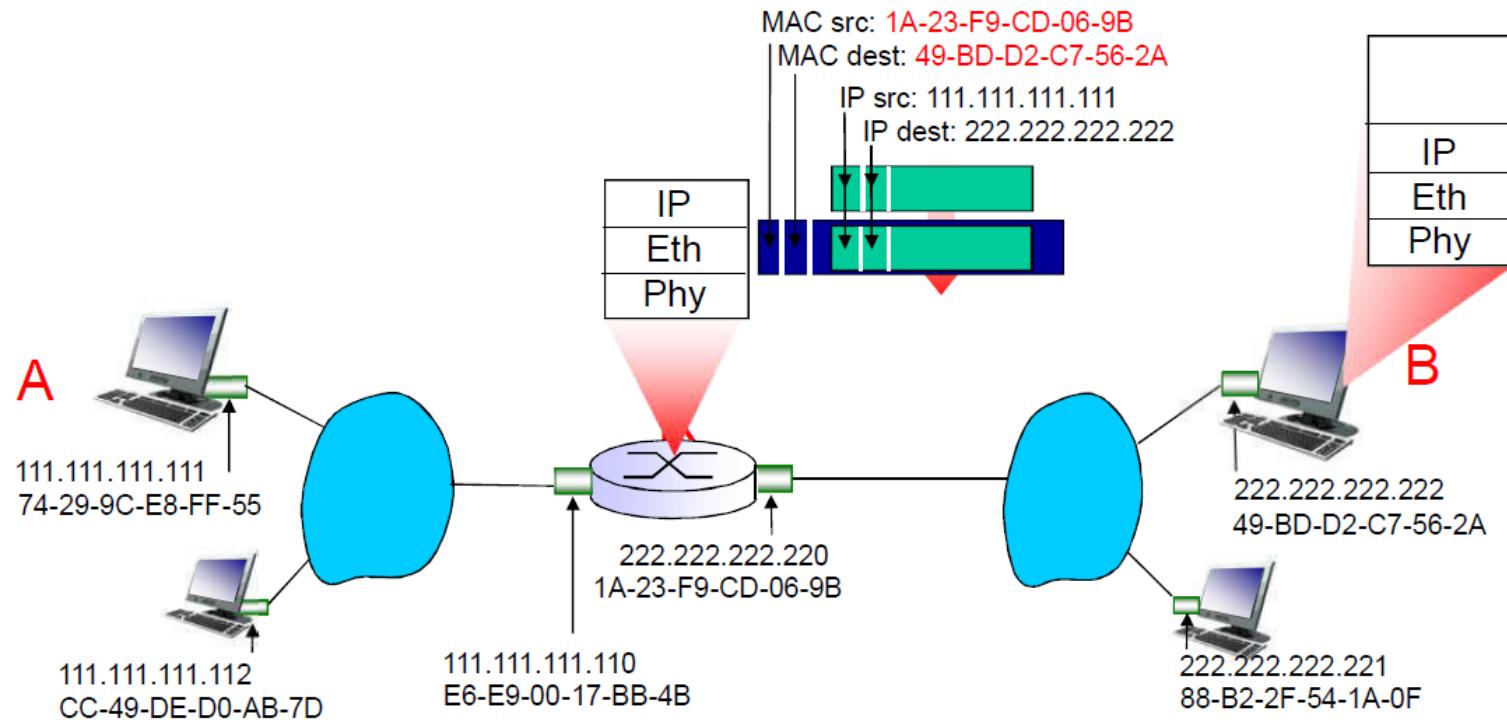
# Định địa chỉ: định tuyến tới LAN khác

- Frame được gửi từ A tới R
- Frame được nhận tại R, datagram được chuyển lên tầng IP



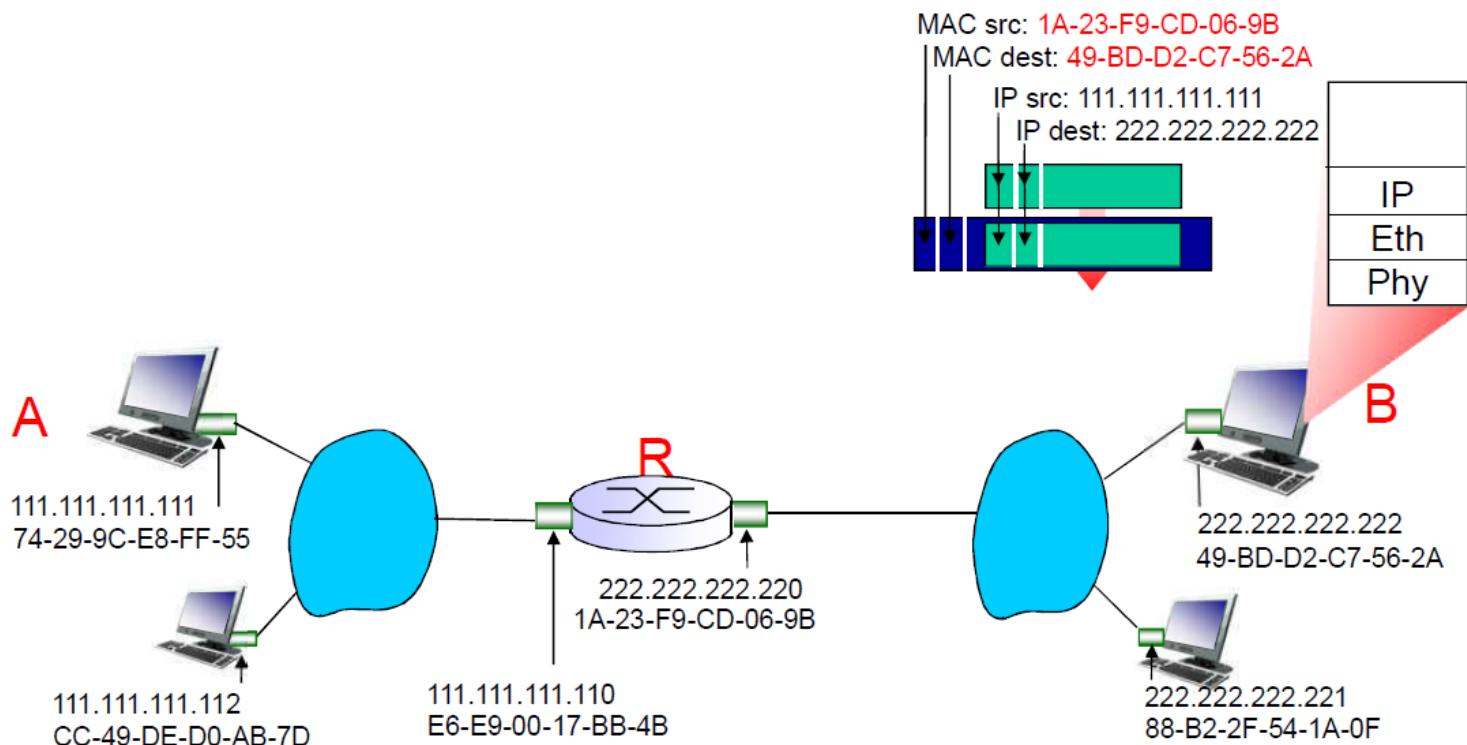
# Định địa chỉ: định tuyến tới LAN khác

- R chuyển tiếp datagram với địa chỉ IP nguồn A, đích B
- R tạo frame tầng liên kết với địa chỉ MAC của B là đích, frame chứa IP datagram từ A-tới-B



# Định địa chỉ: định tuyến tới LAN khác

- Router chuyển tiếp datagram với địa chỉ IP nguồn A, đích B
- Router tạo frame tầng liên kết với địa chỉ MAC của B là đích, frame chứa IP datagram từ A-tới-B

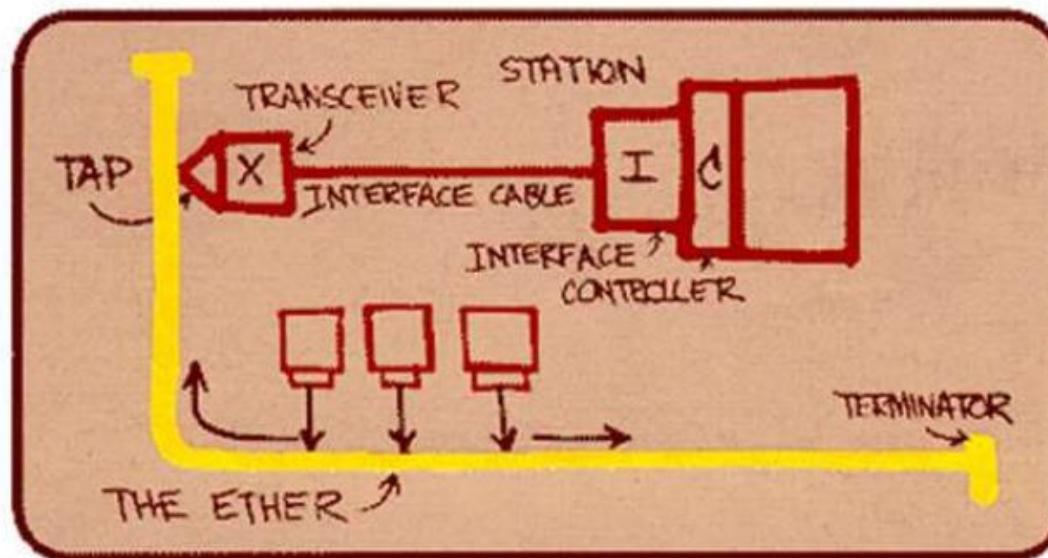


# Nội dung

- Giới thiệu, các dịch vụ
- Phát hiện và sửa lỗi
- Các giao thức đa truy nhập
- Các mạng LAN
  - Định địa chỉ, ARP
  - Ethernet
  - Các switch
  - Các VLAN
- Chuyển mạch nhãn đa giao thức (MPLS)
- Mạng trung tâm dữ liệu
- Vòng đời của một yêu cầu web

# Ethernet

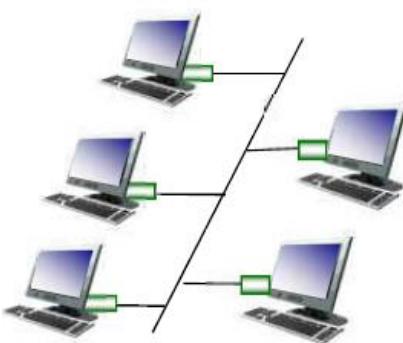
- “Thống trị” công nghệ mạng LAN có dây:
  - Rẻ hơn \$20 cho NIC
  - Công nghệ LAN được sử dụng phổ biến đầu tiên
  - Đơn giản, rẻ hơn so với token LAN và ATM
  - Giữ tốc độ trung bình từ: 10 Mbps – 10 Gbps



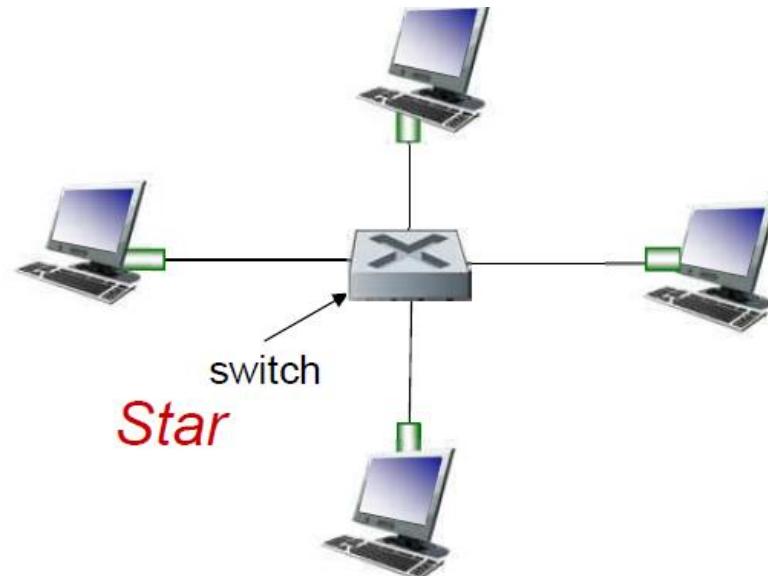
Phác họa Ethernet của  
Metcalfe

# Ethernet: cấu trúc vật lý

- **Bus:** phổi biến cho đến giữa thập niên 90
  - Tất cả các nút đều nằm trong vùng tranh chấp (có thể tranh chấp với các nút khác)
- **Star (hình sao):** chiếm ưu thế hiện nay
  - **Switch** hoạt động ở trung tâm
  - Mỗi “chi nhánh” (văn phòng, spoke) chạy một giao thức Ethernet (riêng) (các nút không tranh chấp với nút khác)



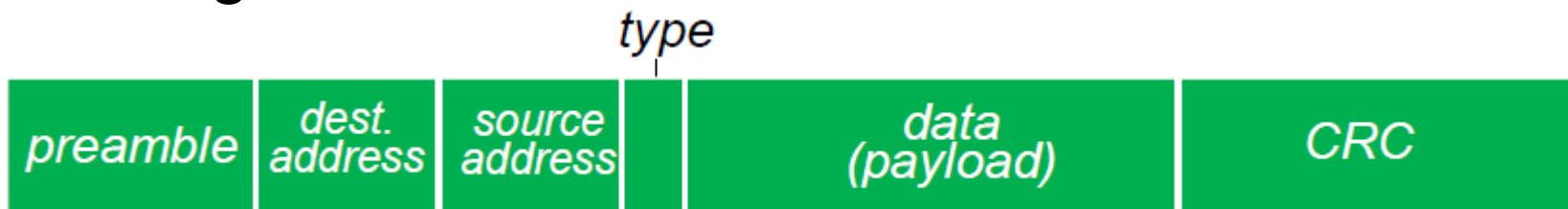
**Bus:** cáp đồng trục



**Star**

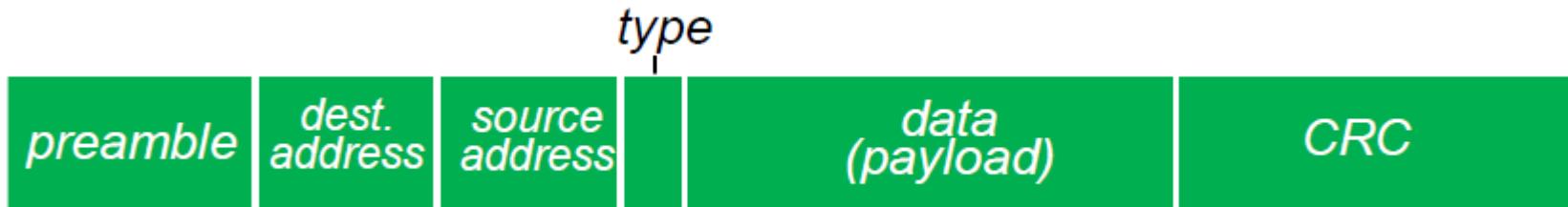
# Cấu trúc Frame của Ethernet

- Gửi IP datagram (hoặc gói giao thức tầng mạng khác) đã được đóng gói trong **frame** của **Ethernet**
- Trường **preamble**:
  - 7 byte với mẫu 10101010 được theo sau bởi một byte với mẫu 10101011.
  - Được dùng để đồng bộ tốc độ của bên nhận, bên gửi.



# Cấu trúc Frame của Ethernet

- Các trường **địa chỉ (nguồn và đích)**: 6 byte địa chỉ MAC nguồn và đích
  - Nếu adapter nhận frame với địa chỉ đích phù hợp, hoặc địa chỉ quảng bá (ví dụ: gói ARP), thì nó sẽ chuyển dữ liệu trong frame tới giao thức tầng mạng.
  - Ngược lại, adapter sẽ bỏ qua frame
- Trường **type**: chỉ ra giao thức tầng cao hơn (thường là IP nhưng cũng có thể là giao thức khác, ví dụ như Novell IPX, AppleTalk)
- Trường **CRC**: kiểm tra mã vòng dư thừa tại phía nhận
  - Phát hiện có lỗi: hủy bỏ frame

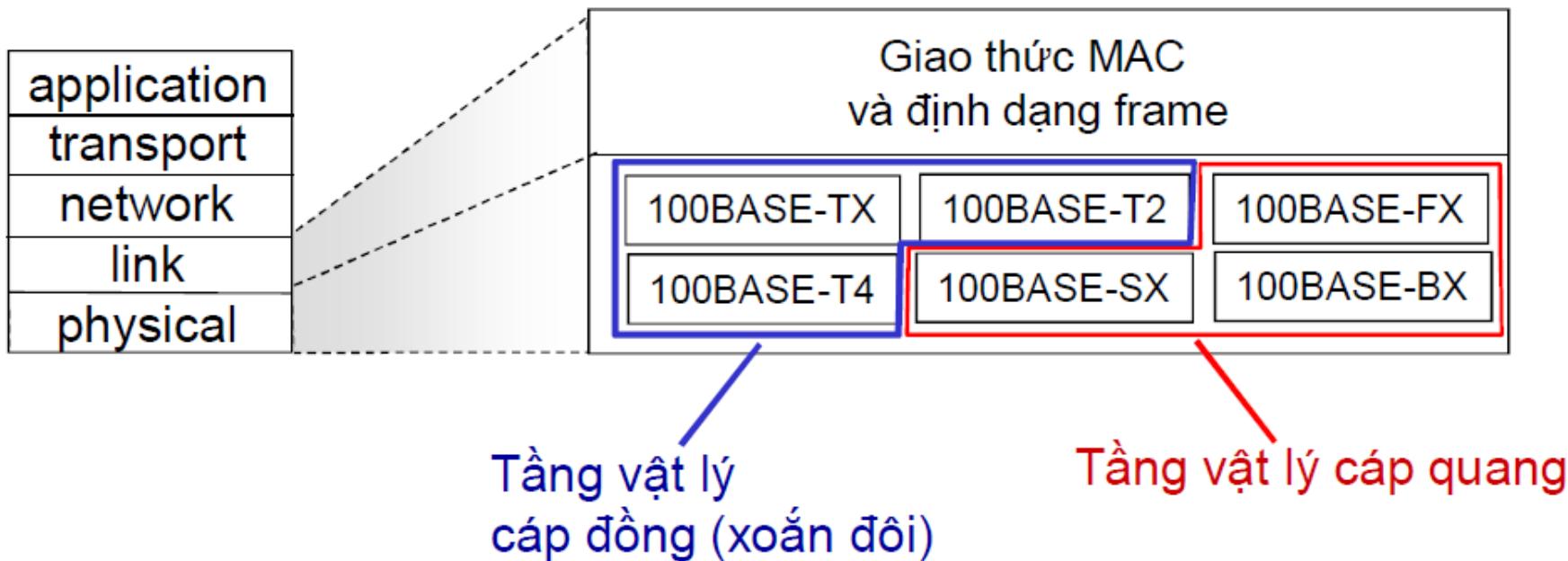


# Ethernet: truyền không tin cậy, không hướng kết nối

- Không hướng kết nối: không có bắt tay giữa bên NIC gửi và bên NIC nhận
- Không tin cậy: NIC nhận không gửi báo nhận (ACK hoặc NACK) cho NIC gửi
  - Dữ liệu trong các frame đã bị hủy chỉ được khôi phục lại khi bên gửi khởi tạo việc dùng giao thức truyền tin cậy (rdt) ở tầng trên (ví dụ: TCP), còn không thì dữ liệu đó sẽ bị mất.
- Giao thức MAC của Ethernet: **CSMA/CD**

# Chuẩn Ethernet 802.3: tầng liên kết và tầng vật lý

- Có nhiều chuẩn Ethernet khác nhau
  - Giao thức MAC và định dạng frame chung
  - Tốc độ khác nhau: 2 Mbps, 10 Mbps, 100 Mbps, 1Gbps, 10G bps
  - Phương tiện tầng vật lý khác nhau: cáp quang, cáp



# Nội dung

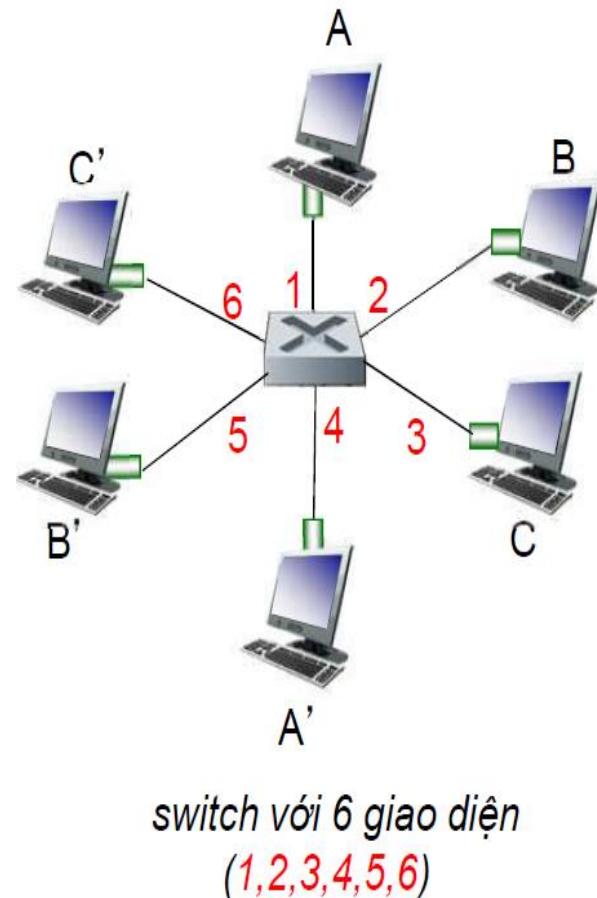
- Giới thiệu, các dịch vụ
- Phát hiện và sửa lỗi
- Các giao thức đa truy nhập
- **Các mạng LAN**
  - Định địa chỉ, ARP
  - Ethernet
  - Các switch
  - Các VLAN
- Chuyển mạch nhãn đa giao thức (MPLS)
- Mạng trung tâm dữ liệu
- Vòng đời của một yêu cầu web

# Switch Ethernet

- Thiết bị tầng liên kết: có nhiệm vụ
  - Lưu và chuyển tiếp các frame Ethernet
  - Kiểm tra địa chỉ MAC của frame đến, **chọn** và chuyển tiếp frame tới một hoặc nhiều liên kết ra
  - Khi frame được chuyển tiếp trên segment, dùng CSMA/CD để truy nhập segment
- Trong suốt
  - Các host không cần biết đến sự có mặt của các switch
- Plug-and-play, tự học
  - Các switch không cần được cấu hình

# Switch: Đa truyền đồng thời

- Các host trực tiếp kết nối với switch
- Các gói tin đệm trong switch
- Giao thức Ethernet được dùng trên mỗi liên kết đến, nhưng không tranh chấp; truyền song công.
  - Mỗi liên kết là vùng tranh chấp của riêng nó.
- **Chuyển mạch:** A-tới-A' và B tới-B' có thể truyền đồng thời, mà không bị tranh chấp.



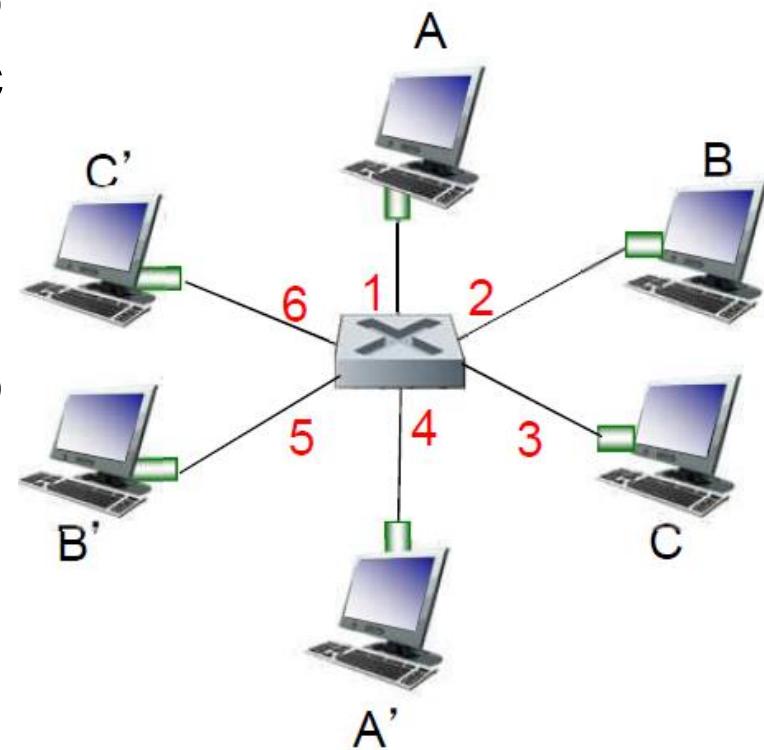
# Bảng chuyển tiếp trong Switch

**Hỏi:** Làm thế nào switch biết được A' có thể truy cập thông qua giao diện 4, B' có thể truy cập được thông qua giao diện 5?

- Trả lời:** Mỗi switch có một **bảng chuyển mạch**, mỗi mục:
  - (Địa chỉ MAC của host, giao diện tới host, nhãn thời gian)
  - Nhìn giống như bảng định tuyến!

**Hỏi:** Cách tạo các mục và duy trì trong bảng chuyển mạch như thế nào?

- Một số giống như giao thức định tuyến?

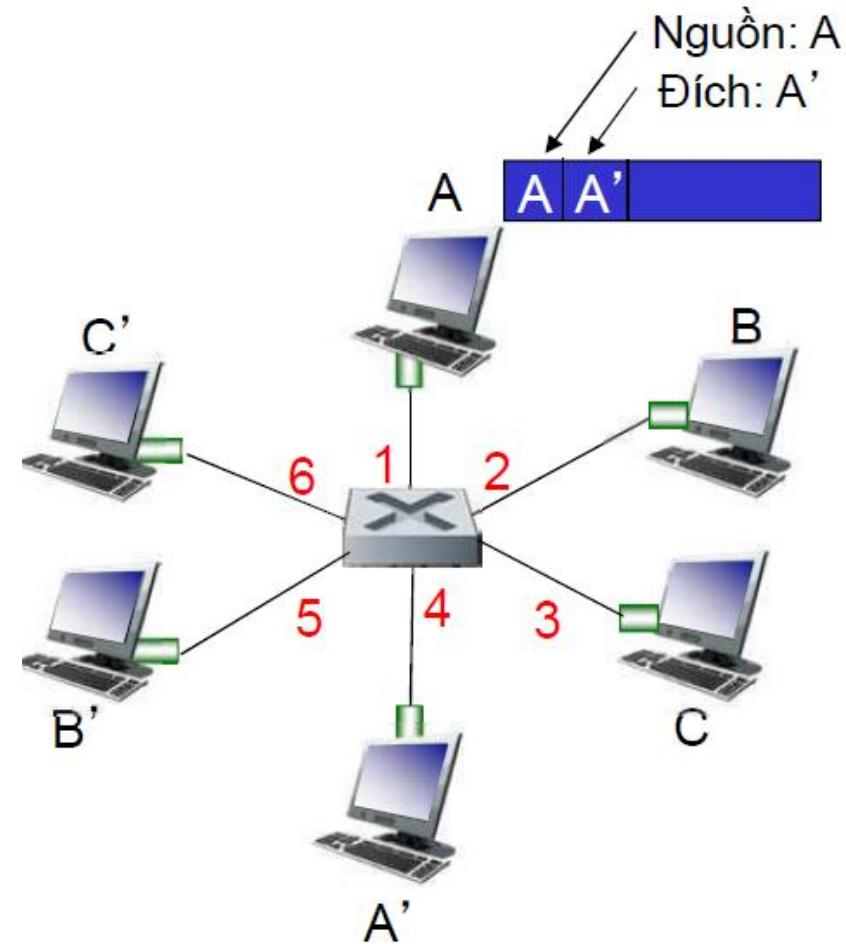


switch với 6 giao diện  
(1,2,3,4,5,6)

# Switch: tự học

- Switch **học** để biết những host nào có thể được truy nhập đến thông qua những giao diện nào
  - Khi nhận được frame, switch “học” vị trí của bên gửi: segment LAN đi đến
  - Ghi cặp bên gửi/vị trí vào trong bảng chuyển mạch

Địa chỉ MAC	Giao diện	TTL
A	1	60



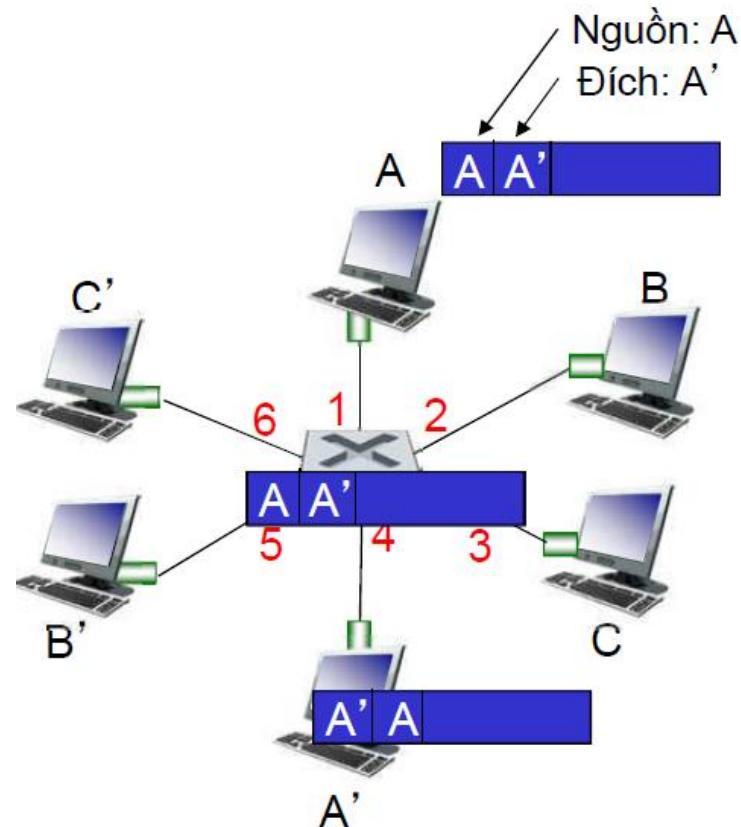
Bảng chuyển mạch  
(khởi tạo rỗng)

# Switch: lọc/chuyển tiếp frame

- Khi switch nhận được frame:
  1. Ghi lại liên kết đến, địa chỉ MAC của host gửi
  2. Đánh chỉ mục bảng chuyển mạch dùng địa chỉ MAC đích
  3. **if** mục được tìm thấy cho đích  
**then** {  
    **if** đích trên segment mà từ đó frame đến  
        **then** bỏ qua frame  
        **else** chuyển tiếp frame trên giao diện được xác định bởi mục  
    }  
    **else** ngập tràn /\* chuyển tiếp trên tất cả các giao diện ngoại trừ giao diện đến\*/}

# Tự học, chuyển tiếp: Ví dụ

- Đích frame, A', vị trí không được biết: **tràn ngập**
- Vị trí đích A được biết:
  - Lựa chọn gửi chỉ trên một liên kết

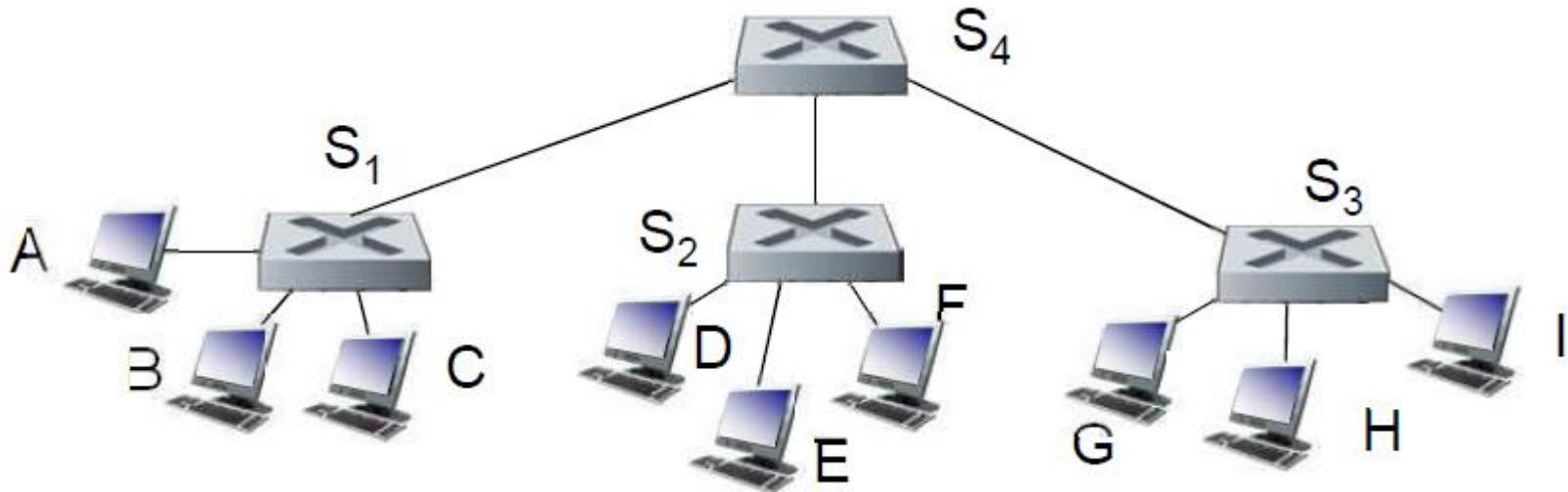


Địa chỉ MAC	Giao diện	TTL
A	1	60
A'	4	60

*Bảng chuyển mạch  
(khởi tạo rỗng)*

# Kết nối các switch

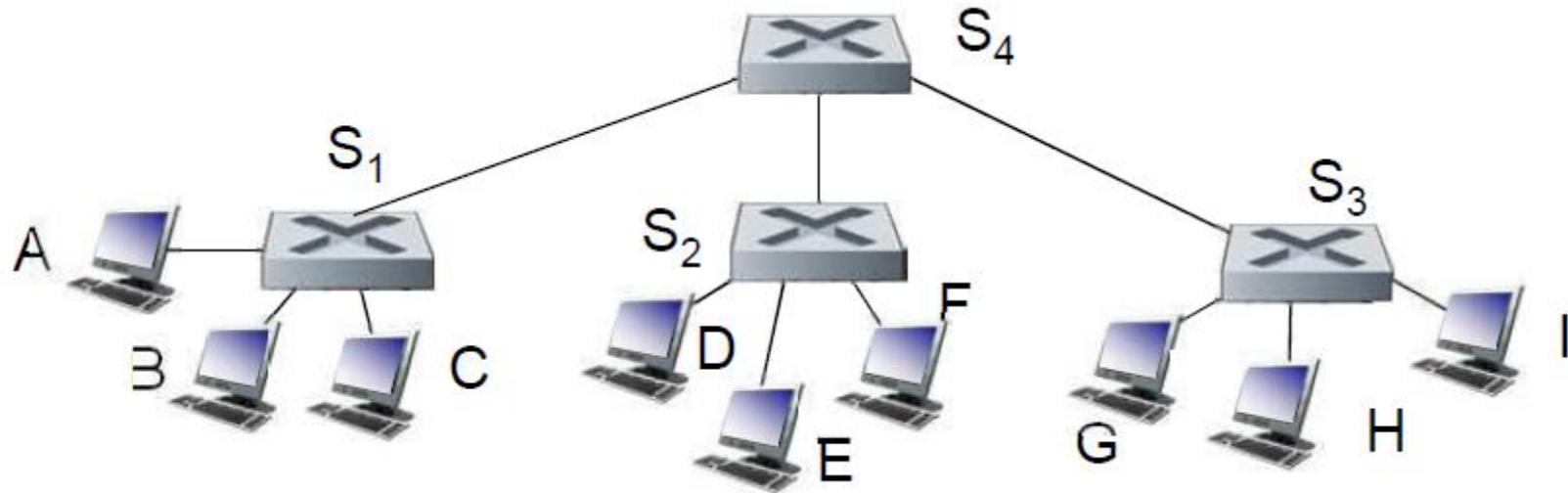
- Các switch có thể được kết nối với nhau



- Hỏi:** gửi từ A đến G – Làm thế nào S1 biết cách chuyển tiếp frame hướng đích G qua S4 và S3?
  - Trả lời:** tự học! (làm theo đúng cách trong trường hợp switch đơn!)

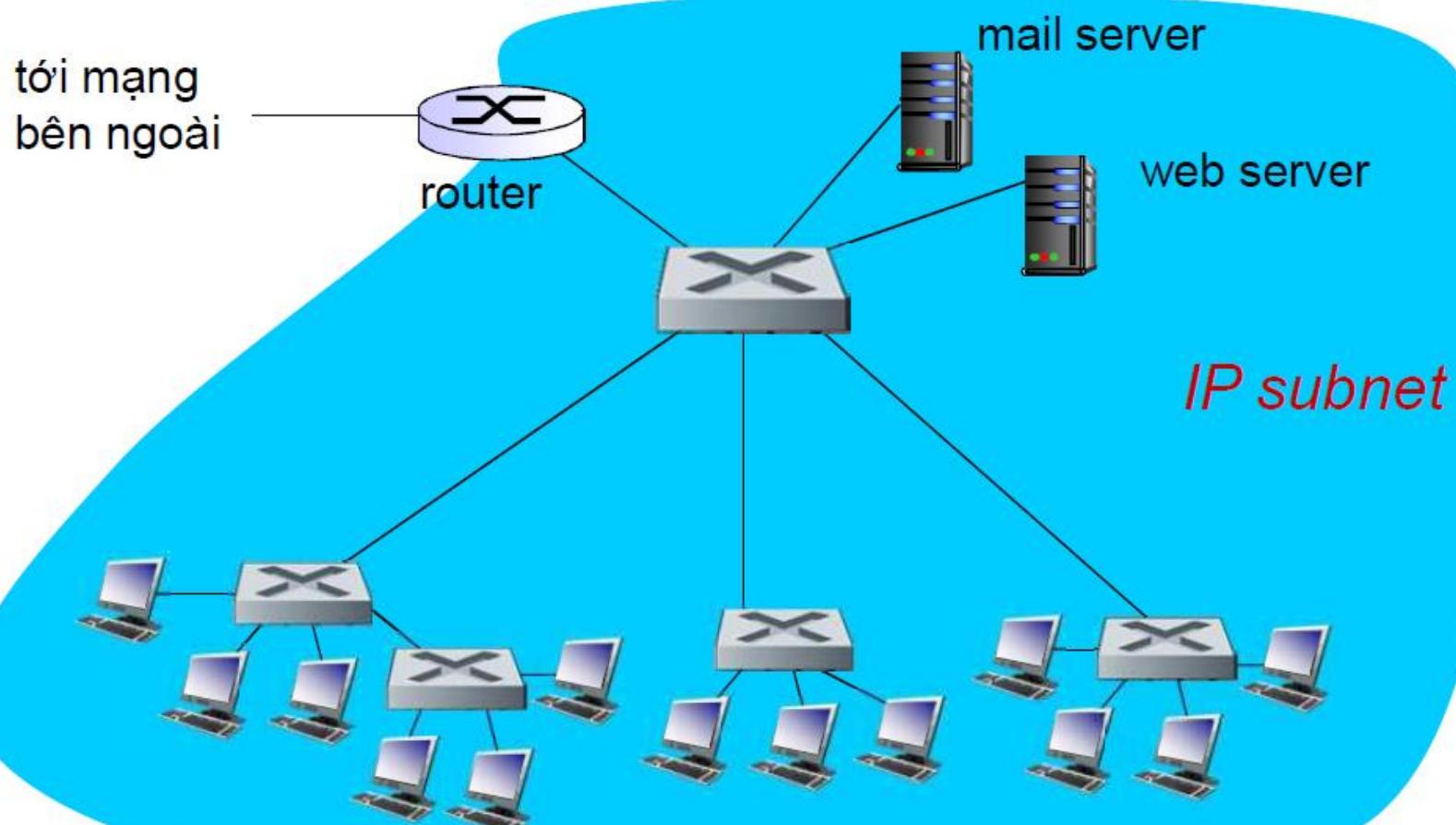
# Ví dụ tự học nhiều switch

- Giả sử C gửi frame tới I, I trả lời lại C



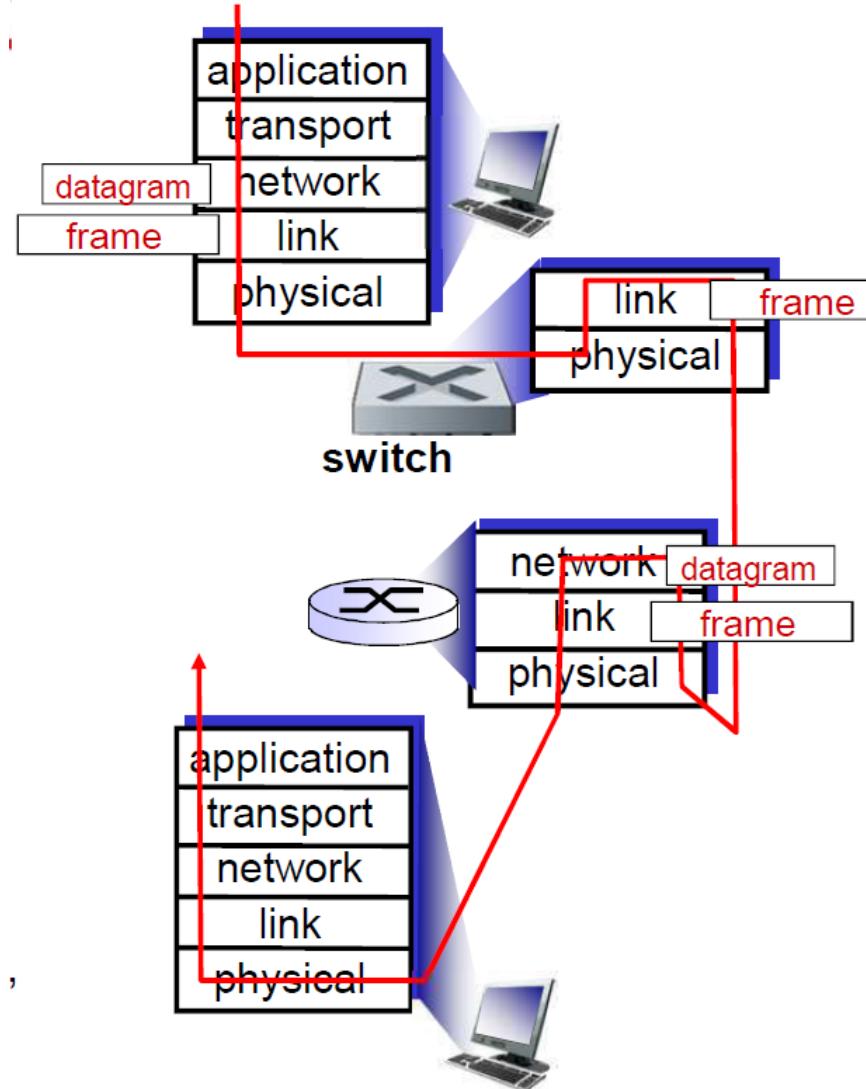
- Hỏi:** Đưa ra các bảng chuyển mạch và chuyển tiếp gói tin trong S<sub>1</sub>, S<sub>2</sub>, S<sub>3</sub>, S<sub>4</sub>

# Mạng nội bộ trong một tổ chức



# Các switch và router

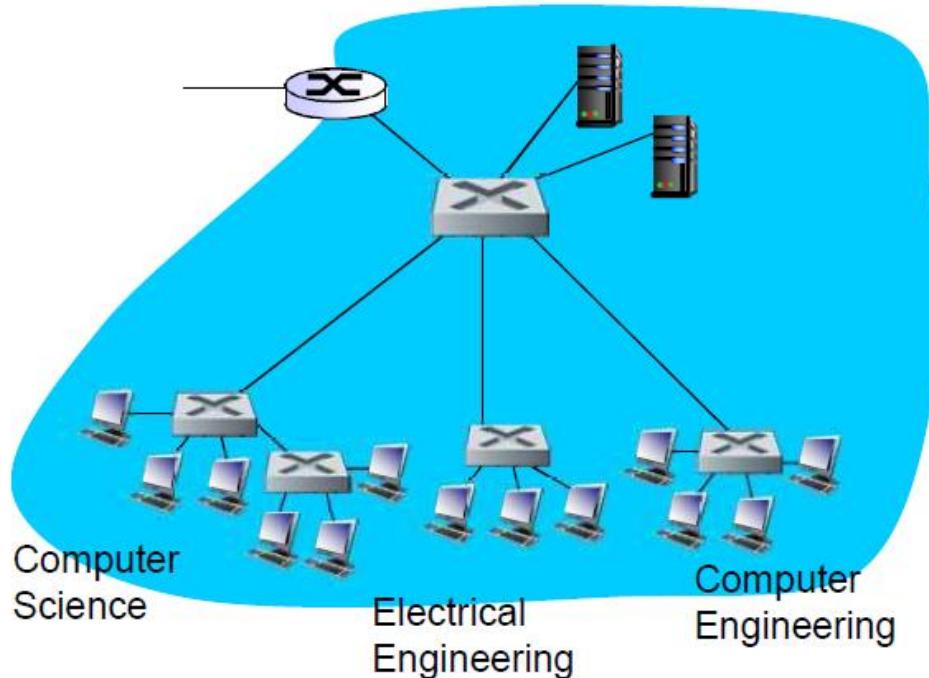
- Cả hai đều có chức năng lưu và chuyển tiếp (store-and-forward):
  - **router**: thiết bị tầng mạng (kiểm tra phần tiêu đề tầng mạng)
  - **switch**: thiết bị tầng liên kết (kiểm tra phần tiêu đề tầng liên kết)
- Cả hai đều có bảng chuyển tiếp:
  - **router**: tính toán bảng chuyển tiếp dùng các giải thuật định tuyến và địa chỉ IP
  - **switch**: học bảng chuyển tiếp dùng kỹ thuật ngập lụt, tự học, và địa chỉ MAC



# VLAN: Động lực

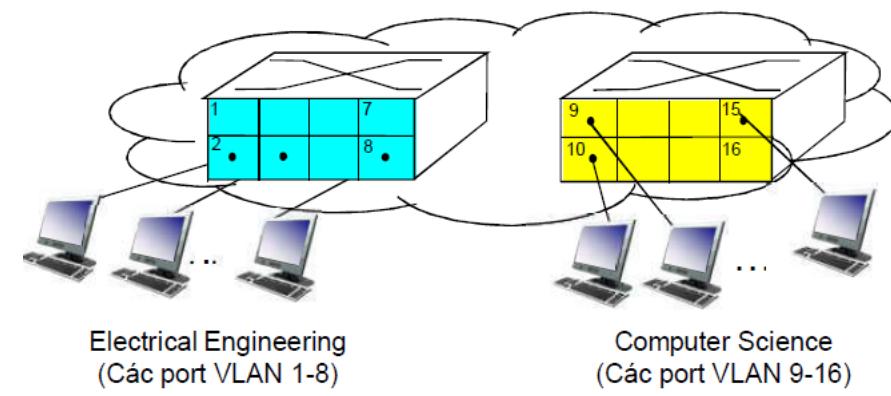
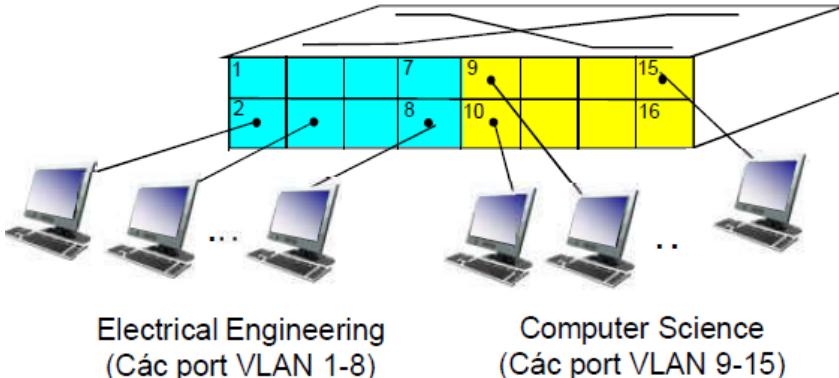
## Xem xét:

- Người dùng CS chuyển văn phòng tới EE, nhưng muốn kết nối với switch của CS?
- Miền quảng bá đơn:
  - Tất cả lưu lượng quảng bá tầng 2 (ARP, DHCP, không biết vị trí của địa chỉ MAC đích) đều phải đi qua toàn bộ LAN.
  - Các vấn đề an toàn/sự riêng tư và hiệu suất.



# VLAN

- Virtual Local Area Network
  - Các switch hỗ trợ khả năng VLAN có thể được cấu hình để xác định nhiều mạng LAN **ảo** trên cơ sở hạ tầng mạng LAN vật lý duy nhất.
- **VLAN dựa trên cổng:**
  - Các port switch được nhóm lại (bởi phần mềm quản lý switch) để thành một switch vật lý **duy nhất**



hoạt động giống như là **nhiều** switch ảo

# VLAN dựa trên cổng

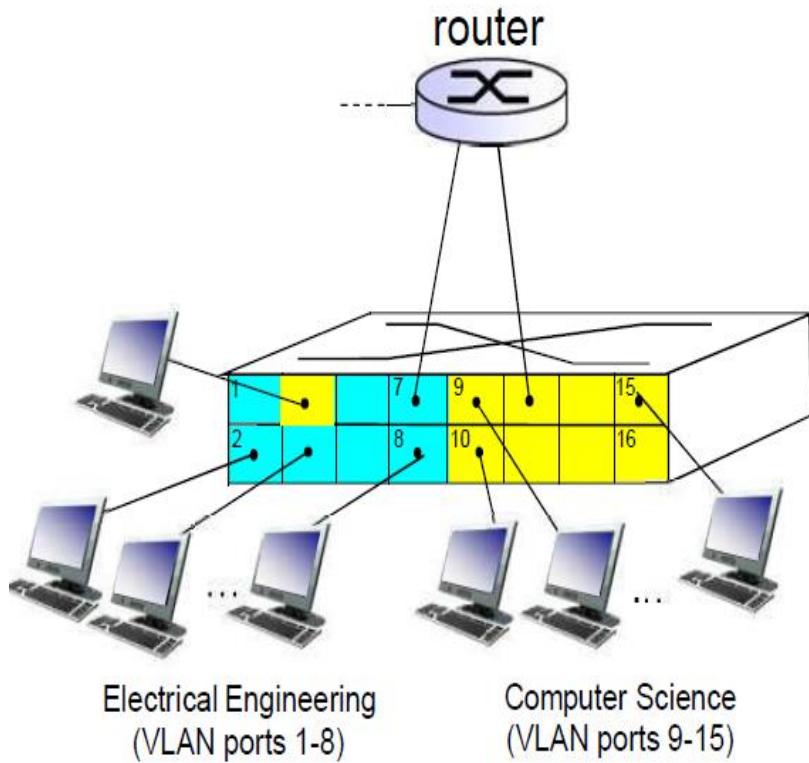
**Cô lập lưu lượng:** các frame tới/từ các port 1-8 chỉ có thể tới các port 1-8

- Cũng có thể xác định VLAN dựa trên địa chỉ MAC của các điểm cuối (endpoint), thay vì các port của switch

**Thành viên động:** các port có thể được gán động giữa các VLAN

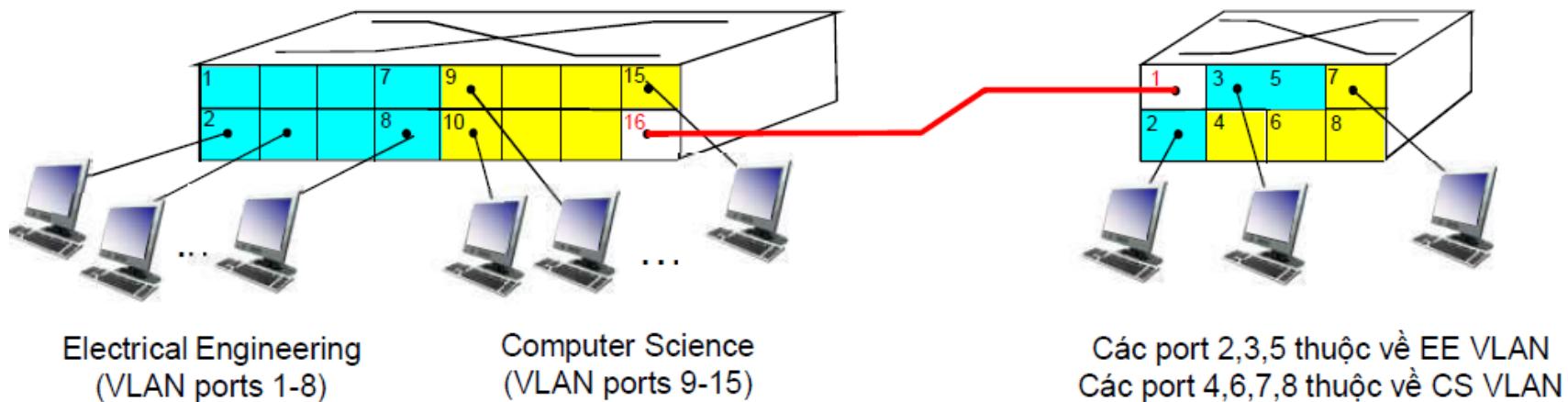
**Chuyển tiếp giữa các VLAN:** được thực hiện thông qua định tuyến (chỉ như là các switch riêng)

- Thực tế các nhà cung cấp bán các switch được kết hợp với các router

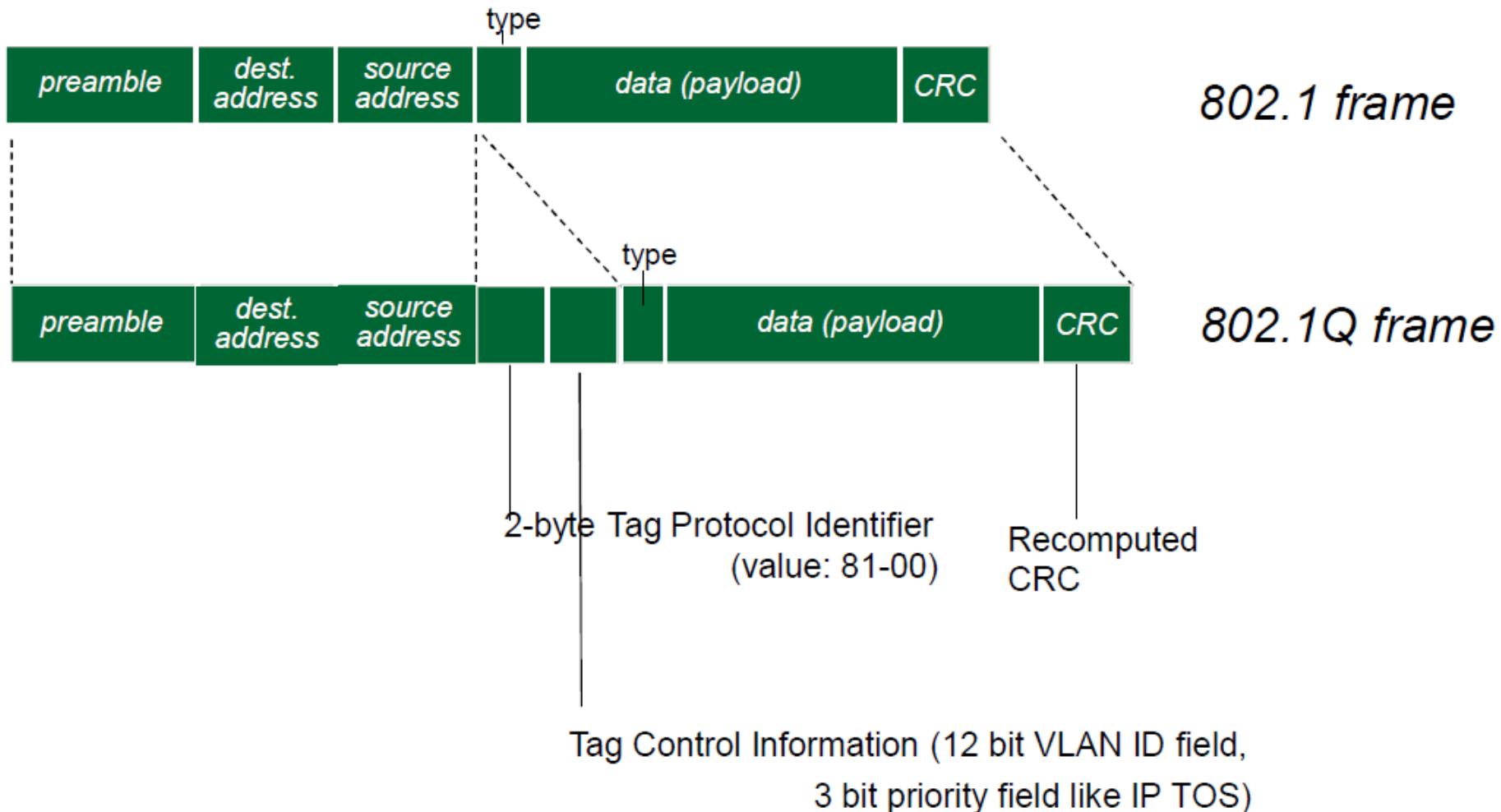


# VLAN mở rộng qua nhiều switch

- **Trunk port:** mang các frame giữa các VLAN được xác định qua nhiều switch vật lý.
  - Các frame được chuyển tiếp bên trong VLAN giữa các switch cần mang thông tin ID của VLAN.
  - Giao thức 802.1q thêm/xóa các trường tiêu đề bổ sung của frame được chuyển tiếp giữa các trunk port.



# Định dạng frame 802.1Q VLAN



# Nội dung

- Giới thiệu, các dịch vụ
- Phát hiện và sửa lỗi
- Các giao thức đa truy nhập
- Các mạng LAN
  - Định địa chỉ, ARP
  - Ethernet
  - Các switch
  - Các VLAN
- Chuyển mạch nhãn đa giao thức (MPLS)
- Mạng trung tâm dữ liệu
- Vòng đời của một yêu cầu web

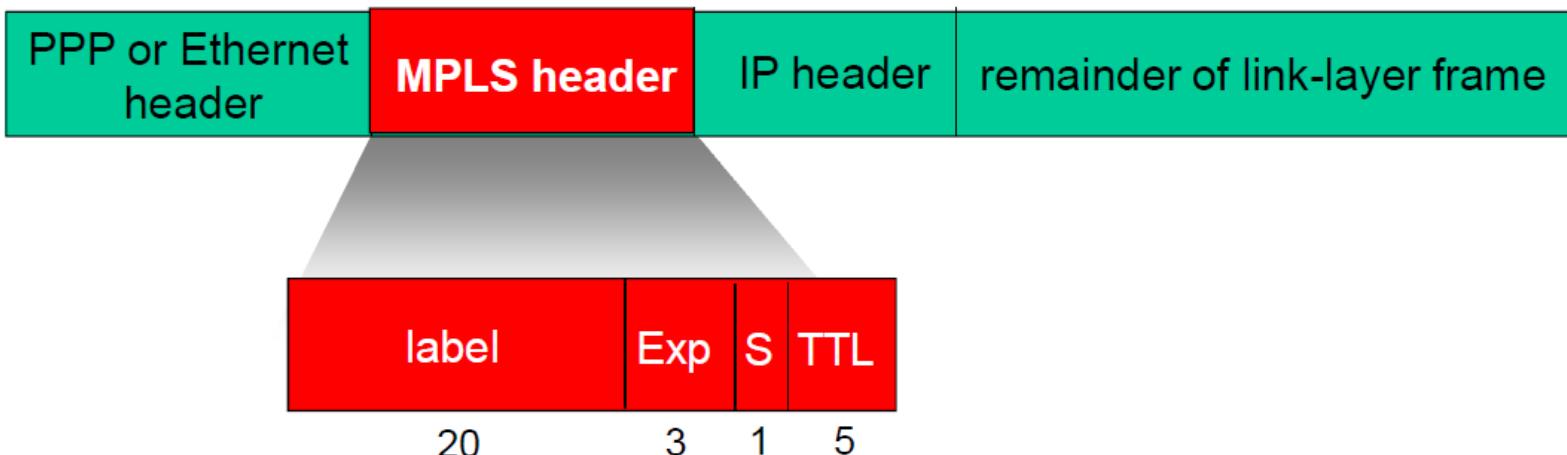
# Chuyển mạch nhãn đa giao thức

- Mục tiêu ban đầu: chuyển mạch IP tốc độ cao sử dụng nhãn chiều dài cố định (thay vì dùng địa chỉ IP)
  - Tìm kiếm nhanh sử dụng định danh chiều dài cố định (thay vì so khớp prefix ngắn nhất)
  - Dựa theo cách tiếp cận mạch ảo (VC)
  - Nhưng IP datagram vẫn giữ địa chỉ IP!

# Chuyển mạch nhãn đa giao thức

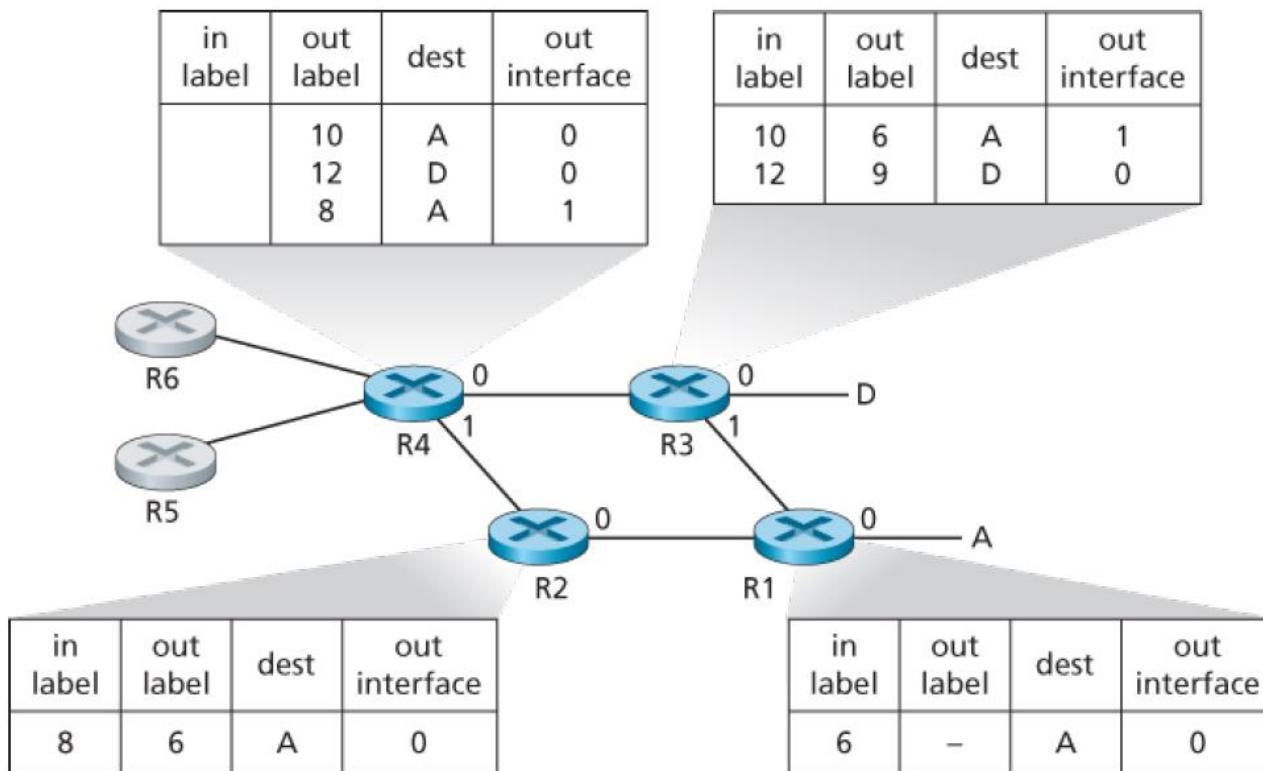
- Định dạng gói tin tầng liên kết (link-layer frame) được xử lý bởi router MPLS
  - a link-layer frame transmitted between MPLS-capable devices has a small MPLS header added between the layer-2 (e.g., Ethernet) header and layer-3 (i.e.,IP) header.
  - RFC 3032 defines the format of the MPLS header for such links

MPLS router (label-switched router)



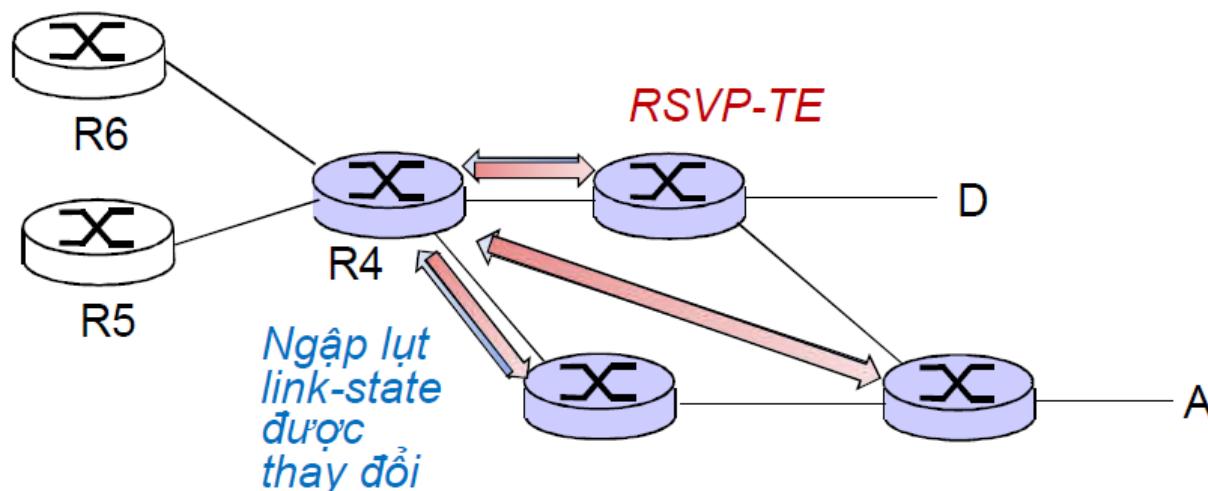
# Các router có khả năng MPLS

- Còn được gọi là các router chuyển mạch nhãn
- Chuyển tiếp các gói tin tới giao diện ra chỉ dựa trên giá trị nhãn (không kiểm tra địa chỉ IP)
  - Bảng chuyển tiếp MPLS khác với bảng chuyển tiếp IP



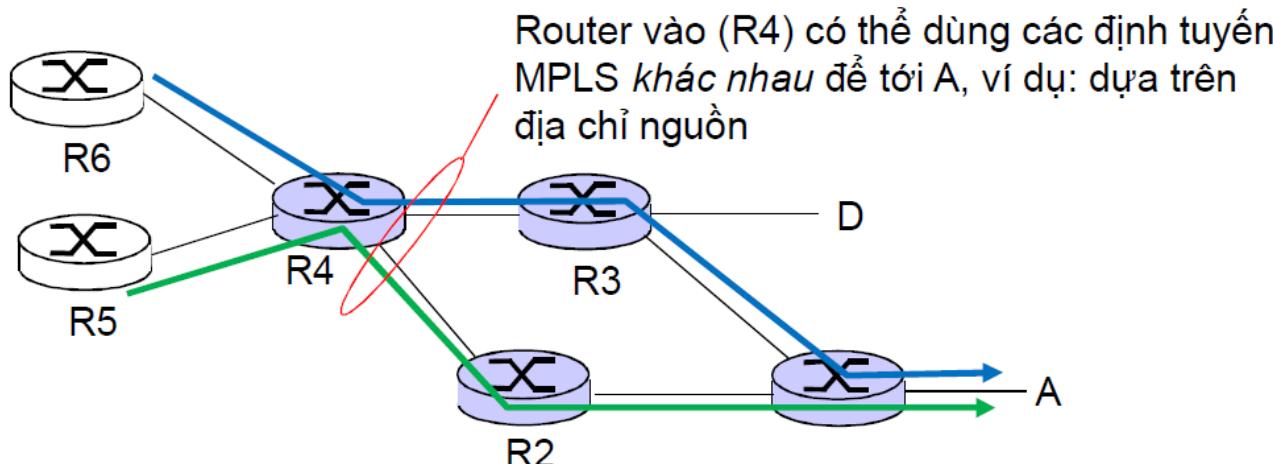
# Báo hiệu trong MPLS

- Thay đổi OSPF, các giao thức ngập lụt link-state IS-IS để mang thông tin cho định tuyến MPLS
  - Ví dụ: băng thông liên kết, tổng băng thông của liên kết “dự phòng”
- Mục router MPLS sử dụng giao thức báo hiệu RSVP-TE để thiết lập chuyển tiếp MPLS tại dòng downstream của các router.



# Các router có khả năng MPLS

- **Tính mềm dẻo:** các quyết định chuyển tiếp MPLS có thể khác với các quyết định chuyển tiếp của IP
  - Dùng địa chỉ đích và nguồn để định hướng luồng tới cùng đích theo các cách khác nhau (kỹ thuật luồng)
  - Định tuyến lại luồng nhanh chóng nếu liên kết bị lỗi: có các đường đi dự phòng được tính toán từ trước (hữu dụng cho VoIP).



- **Ưu điểm của MPLS**
  - Tăng tốc độ chuyển mạch
  - **Cung cấp các khả năng quản lý lưu lượng mới**
    - Như ở ví dụ trên R4 có hai đường MPLS tới A. Nếu như chuyển tiếp được thực hiện ở tầng IP trên cơ sở địa chỉ IP thì với các giao thức định tuyến IP chỉ có một đường duy nhất, chi phí thấp nhất tới A.

- Ưu điểm của MPLS
  - Cung cấp các khả năng quản lý lưu lượng mới
  - MPLS cung cấp khả năng chuyển tiếp gói tin giữa các router không thể sử dụng các giao thức định tuyến IP tiêu chuẩn. Đây là một dạng đơn giản của kỹ thuật lưu lượng sử dụng MPLS [RFC 3346; RFC 3272; RFC 2702; Xiao 2000], ở đó một network operator có thể ghi đè định tuyến IP thông thường và buộc một số lưu lượng mạng hướng về một đích đến nhất định dọc theo một đường, và các lưu lượng khác được hướng đến cùng một đích đến theo đường khác.

- Sử dụng MPLS cho các mục đích khác
  - Fast restoration of MPLS forwarding paths, e.g., to reroute traffic over a precomputed failover path in response to link failure
  - Used to implement VPNs
    - In implementing a VPN for a customer, an ISP uses its MPLS-enabled network to connect together the customer's various networks.
    - Used to isolate both the resources and addressing used by the customer's VPN from that of other users crossing the ISP's network;

# Nội dung

- Giới thiệu, các dịch vụ
- Phát hiện và sửa lỗi
- Các giao thức đa truy nhập
- **Các mạng LAN**
  - Định địa chỉ, ARP
  - Ethernet
  - Các switch
  - Các VLAN
- Chuyển mạch nhãn đa giao thức (MPLS)
- **Mạng trung tâm dữ liệu**
- Vòng đời của một yêu cầu web

# Mạng trung tâm dữ liệu (Data center networks)



Bên trong một container 40-ft của Microsoft, tại trung tâm dữ liệu ở Chicago

# Mạng trung tâm dữ liệu (Data center networks)

- Cost of a large data center is huge, exceeding \$12 million per month for a 100,000 host data center
  - 45% attributed to the hosts themselves (which need to be replaced every 3–4 years);
  - 25% to infrastructure (including transformers, uninterruptable power supplies (UPS) systems, generators for long-term outages, and cooling systems)
  - 15% for electric utility costs for the power draw;
  - 15% for networking (including network gear-switches, routers and load balancers, external links, and transit traffic costs).

# Mạng trung tâm dữ liệu (Data center networks)

- The worker bees in a data center are the hosts
  - They serve content (e.g., Web pages and videos), store e-mails and documents, and collectively perform massively distributed computations (e.g., distributed index computations for search engines).
  - The hosts in data centers, called **blades** and resembling pizza boxes, are generally commodity hosts that include CPU, memory, and disk storage

# Mạng trung tâm dữ liệu (Data center networks)

- The worker bees in a data center are the hosts
  - The hosts are stacked in racks, with each rack typically having 20 to 40 blades.
  - At the top of each rack there is a switch, named the Top of Rack (TOR) switch, that interconnects the hosts in the rack with each other and with other switches in the data center.

# Mạng trung tâm dữ liệu (Data center networks)

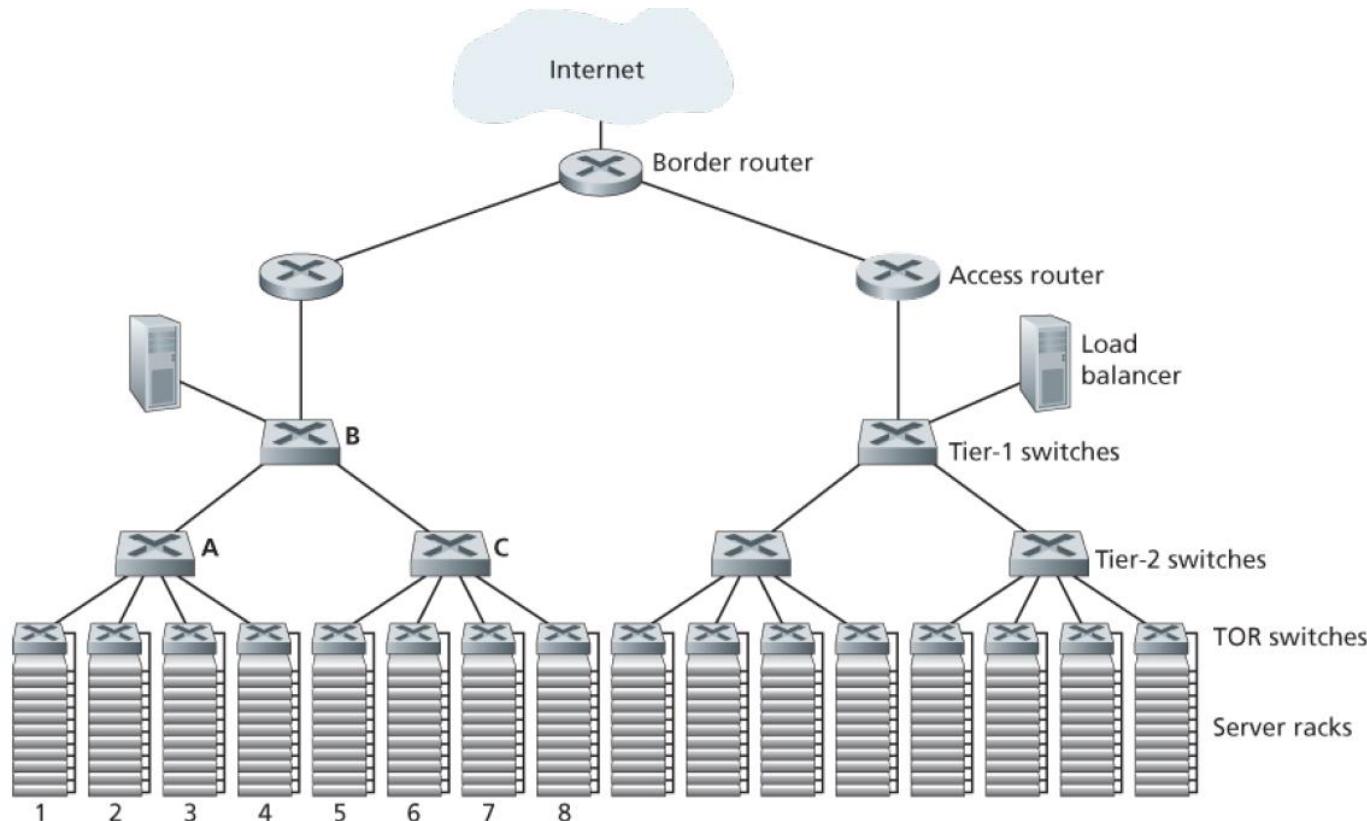
- The worker bees in a data center are the hosts
  - Each host in the rack has a network interface card that connects to its TOR switch, and each TOR switch has additional ports that can be connected to other switches.
  - Today hosts typically have 40 Gbps Ethernet connections to their TOR switches.
  - Each host is also assigned its own data-center-internal IP address.

# Mạng trung tâm dữ liệu (Data center networks)

- The data center network supports two types of traffic
  - Traffic flowing between external clients and internal hosts
    - To handle flows between external clients and internal hosts, the data center network includes one or more border routers, connecting the data center network to the public Internet.
  - Traffic flowing between internal hosts.

# Mạng trung tâm dữ liệu (Data center networks)

- The data center network
  - Interconnects the racks with each other
  - Connects the racks to the border routers



# Mạng trung tâm dữ liệu (Data center networks)

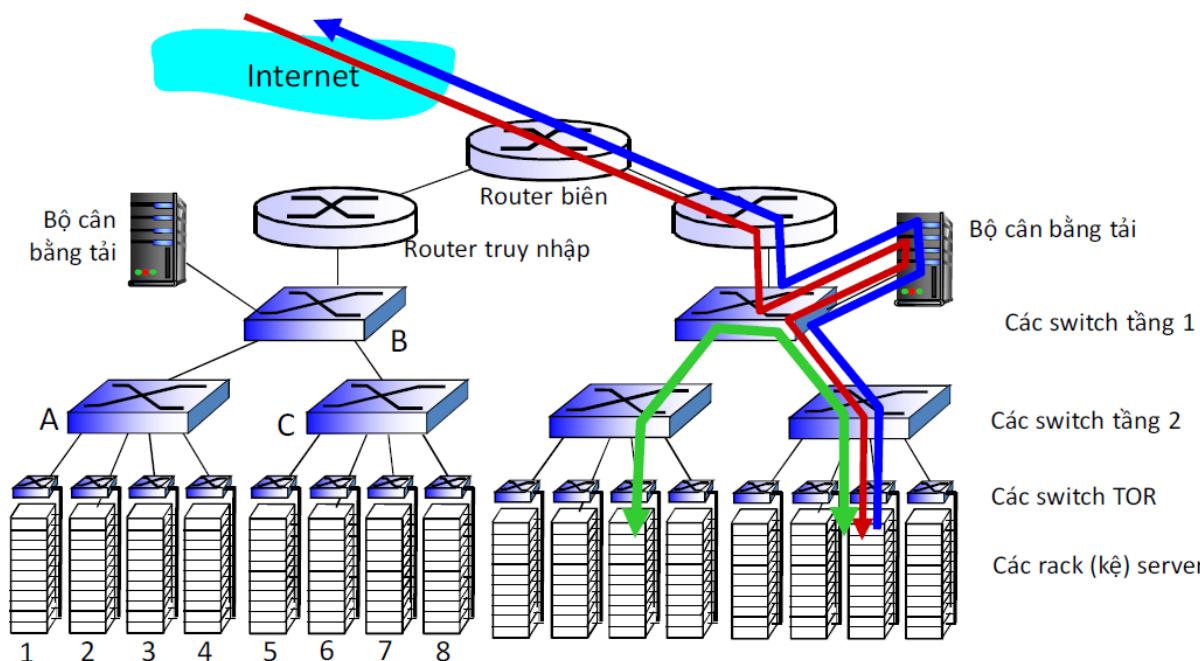
- The data center network
  - Data center network design, the art of designing the interconnection network and protocols that connect the racks with each other and with the border routers, has become an important branch of computer networking research in recent years.

# Mạng trung tâm dữ liệu

- Cân bằng tải (Load Balancing)
  - A cloud data center, such as a Google or Microsoft data center, provides many applications concurrently, such as search, e-mail, and video applications.
    - To support requests from external clients, **each application** is associated with a **publicly visible IP address** to which clients send their requests and from which they receive responses.
    - Inside the data center, the external requests are first directed to a load balancer whose job it is to distribute requests to the hosts, balancing the load across the hosts as a function of their current load.

# Mạng trung tâm dữ liệu

- Bộ cân bằng tải: định tuyến tầng ứng dụng
  - Nhận các yêu cầu từ client ở phía ngoài
  - Chỉ đạo công việc bên trong trung tâm dữ liệu
  - Trả lại kết quả cho các client (ẩn trung tâm dữ liệu bên trong với client)



# Mạng trung tâm dữ liệu

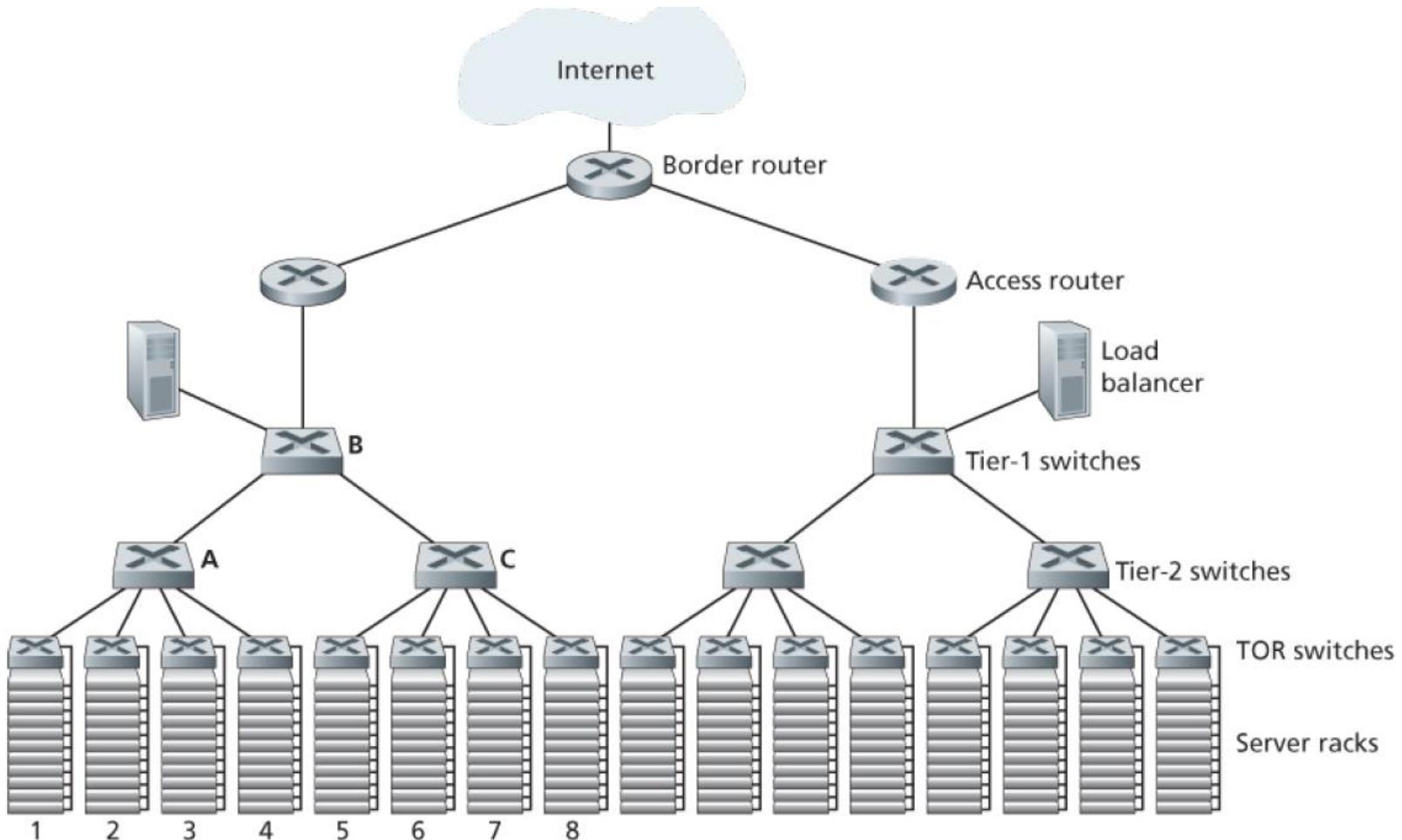
- Cân bằng tải (Load Balancing)
  - A large data center will often have several load balancers, each one devoted to a set of specific cloud applications. Such a load balancer is sometimes referred to as a “layer-4 switch” since it makes decisions based on the destination port number (layer 4) as well as destination IP address in the packet.
  - Upon receiving a request for a particular application, the load balancer forwards it to one of the hosts that handles the application. (A host may then invoke the services of other hosts to help process the request.).

# Mạng trung tâm dữ liệu

- Cân bằng tải (Load Balancing)
  - When the host finishes processing the request, it sends its response back to the load balancer, which in turn relays the response back to the external client.
  - The load balancer not only balances the work load across hosts, but also provides a NAT-like function

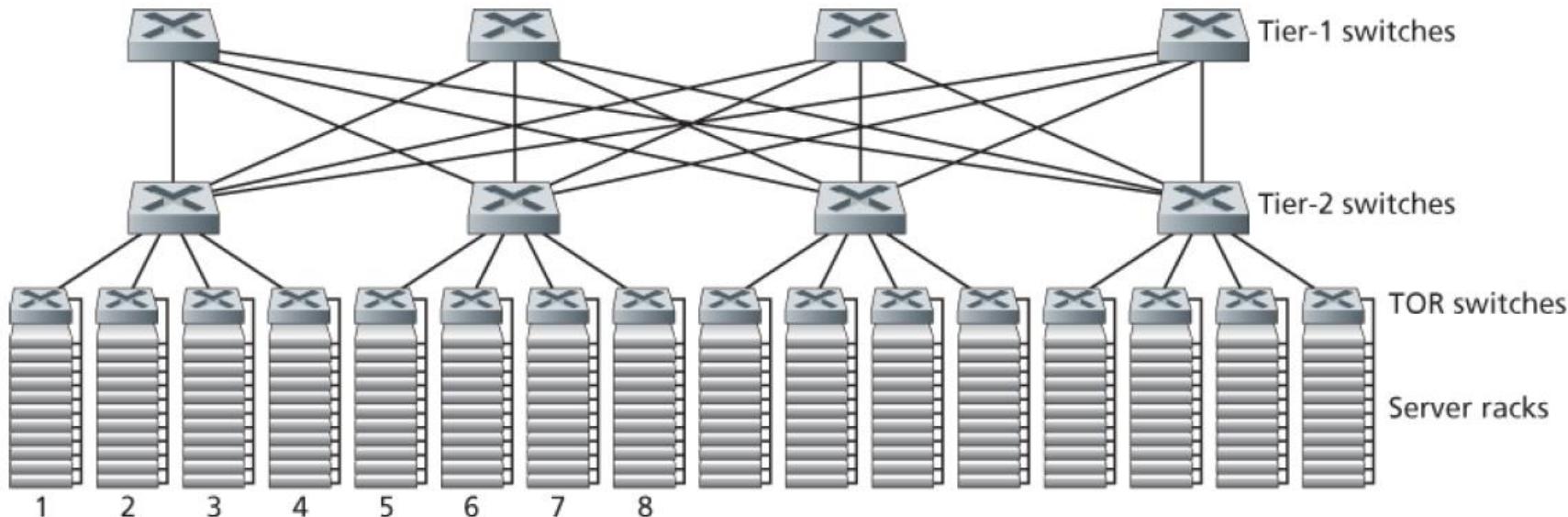
# Mạng trung tâm dữ liệu

- Hierarchical Architecture: Kết nối rất nhiều các switch và các kệ (rack)



# Mạng trung tâm dữ liệu

- Trends in Data Center Networking
  - Deploy new interconnection architectures and network protocols that overcome the drawbacks of the traditional hierarchical designs.
    - Replace the hierarchy of switches and routers with a **fully connected topology**



# Mạng trung tâm dữ liệu

- Trends in Data Center Networking
  - Fully connected topology
    - Each tier-1 switch connects to all of the tier-2 switches so that
      - (1) host-to-host traffic never has to rise above the switch tiers
      - (2) with  $n$  tier-1 switches, between any two tier-2 switches there are  $n$  disjoint paths.

# Mạng trung tâm dữ liệu

- Trends in Data Center Networking
  - Fully connected topology
    - Significantly improve the host-to-host capacity
      - An example of 40 flows. The topology can handle such a flow pattern since there are four distinct paths between the first tier-2 switch and the second tier-2 switch, together providing an aggregate capacity of 40 Gbps between the first two tier-2 switches.
      - Such a design not only alleviates the host-to-host capacity limitation, but also creates a more flexible computation and service environment in which communication between any two racks not connected to the same switch is logically equivalent, irrespective of their locations in the data center.

# Mạng trung tâm dữ liệu

- Trends in Data Center Networking
  - Employ shipping container-based modular data centers (MDCs)
    - A factory builds within a standard **12-meter shipping container**
      - a “mini data center” and ships the container to the data center location.
      - has up to a few thousand hosts, stacked in tens of racks, which are packed closely together.
    - At the data center location, multiple containers are interconnected with each other and also with the Internet.

# Mạng trung tâm dữ liệu

- Trends in Data Center Networking
  - Employ shipping container-based modular data centers (MDCs)
    - Khi một container đúc sẵn được triển khai tại trung tâm dữ liệu, nó thường khó bảo trì. Do đó, mỗi container được thiết kế để dễ dàng giảm hiệu năng
      - Các thành phần (máy chủ và thiết bị chuyển mạch) bị lỗi theo thời gian, container tiếp tục hoạt động nhưng với hiệu suất bị giảm sút.
      - Khi nhiều thành phần bị lỗi và hiệu suất giảm xuống dưới ngưỡng, toàn bộ container được tháo ra và thay thế bằng container mới.

# Mạng trung tâm dữ liệu

- Trends in Data Center Networking
  - Employ shipping container-based modular data centers (MDCs)
    - Việc xây dựng một trung tâm dữ liệu ngoài container sẽ tạo ra những thách thức mạng mới.
    - With an MDC, there are two types of networks
      - The container-internal networks within each of the containers.
        - » Within each container, at the scale of up to a few thousand hosts, it is possible to build a fully connected network using inexpensive commodity Gigabit Ethernet switches.

# Mạng trung tâm dữ liệu

- Trends in Data Center Networking
  - With an MDC, there are two types of networks
    - The core network connecting interconnecting hundreds to thousands of containers providing high host-to-host bandwidth across containers for typical workloads, remains a challenging problem.
      - » One of the major issues is designing routing algorithms among the switches.
      - » One possibility is to use a form of random routing
      - » Another possibility is to deploy multiple network interface cards in each host, connect each host to multiple low-cost commodity switches, and allow the hosts themselves to intelligently route traffic among the switches. Variations and extensions of these approaches are currently being deployed in contemporary data centers.

# Mạng trung tâm dữ liệu

- Trends in Data Center Networking
  - Large cloud providers are increasingly building or customizing just about everything that is in their data centers, including network adapters, switches routers, TORs, software, and networking protocols.
  - Another trend, pioneered by Amazon, is to improve reliability with “availability zones,” which essentially replicate distinct data centers in different nearby buildings. By having the buildings nearby (a few kilometers apart), transactional data can be synchronized across the data centers in the same availability zone while providing fault tolerance.

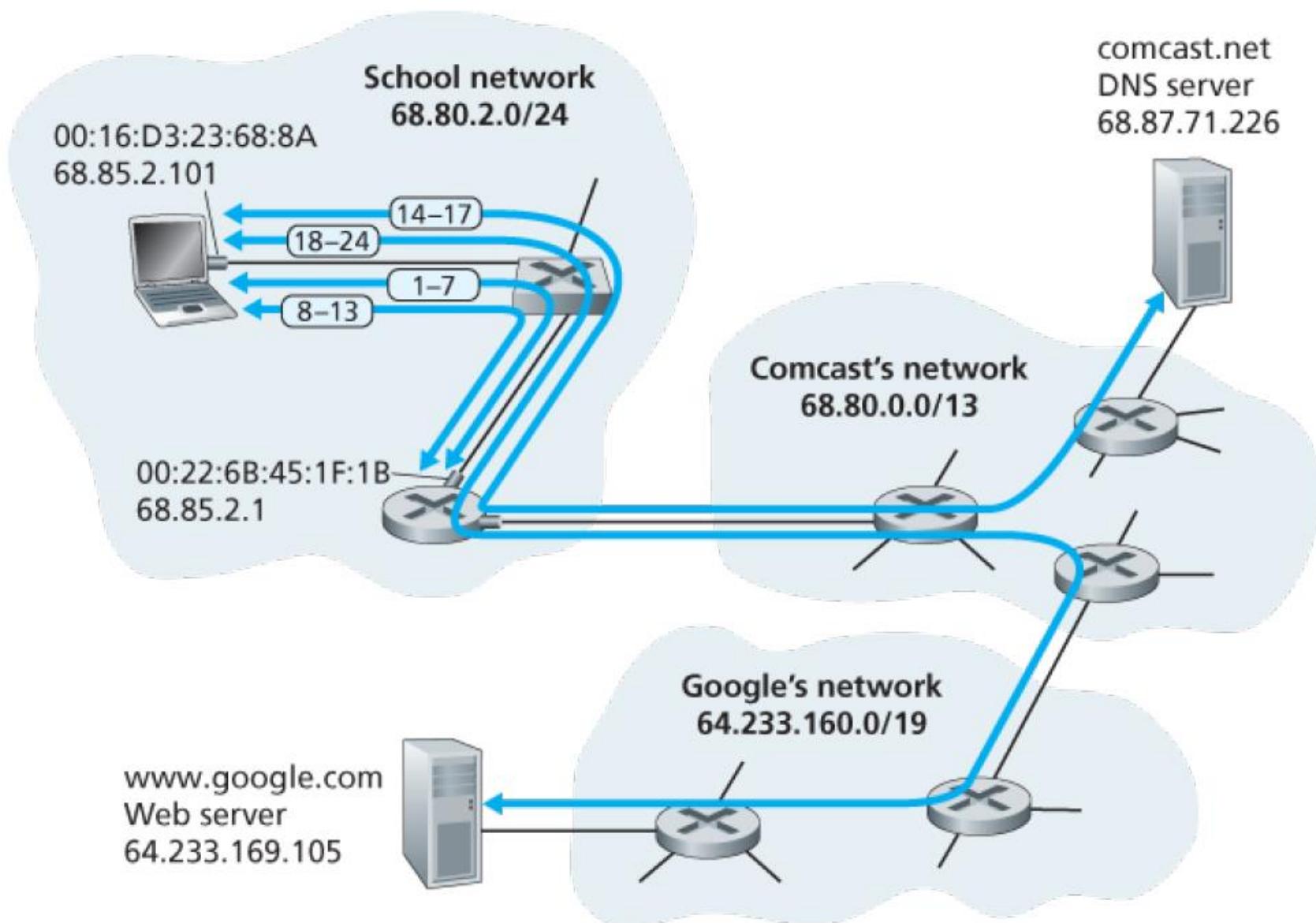
# Nội dung

- Giới thiệu, các dịch vụ
- Phát hiện và sửa lỗi
- Các giao thức đa truy nhập
- Các mạng LAN
  - Định địa chỉ, ARP
  - Ethernet
  - Các switch
  - Các VLAN
- Chuyển mạch nhãn đa giao thức (MPLS)
- Mạng trung tâm dữ liệu
- Vòng đời của một yêu cầu web

# Tổng hợp: Vòng đời của một yêu cầu web

- Hành trình đi xuống chồng giao thức đã hoàn thành!
  - Tầng ứng dụng, tầng giao vận, tầng mạng, tầng liên kết.
- Đặt tất cả mọi thứ lại cùng nhau: tổng hợp!
  - **Mục đích:** xác định, xem xét, hiểu các giao thức (tại tất cả các tầng) liên quan trong một kịch bản khá đơn giản: yêu cầu một trang web.
  - **Kịch bản:** Sinh viên thực hiện yêu cầu/đáp ứng từ hệ thống mạng trong trường: [www.google.com](http://www.google.com)

# Kịch bản: Vòng đời của một yêu cầu web



# Kịch bản: Vòng đời của một yêu cầu web

- DHCP, UDP, IP, và Ethernet
  - The operating system on Bob's laptop creates a **DHCP request** message and puts this message within a **UDP segment** with destination port 67 (DHCP server) and source port 68 (DHCP client). The UDP segment is then placed within an **IP datagram** with a broadcast IP destination address (255.255.255.255) and a source IP address of 0.0.0.0
    - since Bob's laptop doesn't yet have an IP address.

# Kịch bản: Vòng đời của một yêu cầu web

- DHCP, UDP, IP, và Ethernet
  - The IP datagram containing the DHCP request message is then placed within an **Ethernet frame**. The Ethernet frame has a destination MAC addresses of FF:FF:FF:FF:FF:FF so that the frame will be broadcast to all devices connected to the switch (hopefully including a DHCP server); the frame's source MAC address is that of Bob's laptop, 00:16:D3:23:68:8A.

# Kịch bản: Vòng đời của một yêu cầu web

- DHCP, UDP, IP, và Ethernet
  - The broadcast Ethernet frame containing the DHCP request is the first frame sent by Bob's laptop to the Ethernet switch. The switch broadcasts the incoming frame on all outgoing ports, including the port connected to the router.

# Kịch bản: Vòng đời của một yêu cầu web

- DHCP, UDP, IP, và Ethernet
  - The router receives the broadcast Ethernet frame containing the DHCP request on its interface with MAC address 00:22:6B:45:1F:1B and the IP datagram is extracted from the Ethernet frame. The datagram's broadcast IP destination address indicates that this IP datagram should be processed by upper layer protocols at this node, so the datagram's payload (a UDP segment) is thus demultiplexed up to UDP, and the DHCP request message is extracted from the UDP segment. The DHCP server now has the DHCP request message.

# Kịch bản: Vòng đời của một yêu cầu web

- DHCP, UDP, IP, và Ethernet
  - Let's suppose that the DHCP server running within the router can allocate IP addresses in the CIDR block 68.85.2.0/24.
    - In this example, all IP addresses used within the school are thus within Comcast's address block.
  - Let's suppose the **DHCP server allocates address 68.85.2.101 to Bob's laptop**. The DHCP server creates a **DHCP ACK message** containing this IP address, as well as the IP address of the DNS server (68.87.71.226), the IP address for the default gateway router (68.85.2.1), and the subnet block (68.85.2.0/24) (equivalently, the “network mask”).

# Kịch bản: Vòng đời của một yêu cầu web

- DHCP, UDP, IP, và Ethernet
  - The **DHCP message** is put inside a **UDP segment**, which is put inside an **IP datagram**, which is put inside an **Ethernet frame**. The Ethernet frame has a source MAC address of the router's interface to the home network (00:22:6B:45:1F:1B) and a destination MAC address of Bob's laptop (00:16:D3:23:68:8A).
  - The **Ethernet frame** containing the **DHCP ACK** is sent (unicast) by the router to the switch. Because the switch is self-learning and previously received an Ethernet frame (containing the DHCP request) from Bob's laptop, the switch knows to forward a frame addressed to 00:16:D3:23:68:8A only to the output port leading to Bob's laptop.

# Kịch bản: Vòng đời của một yêu cầu web

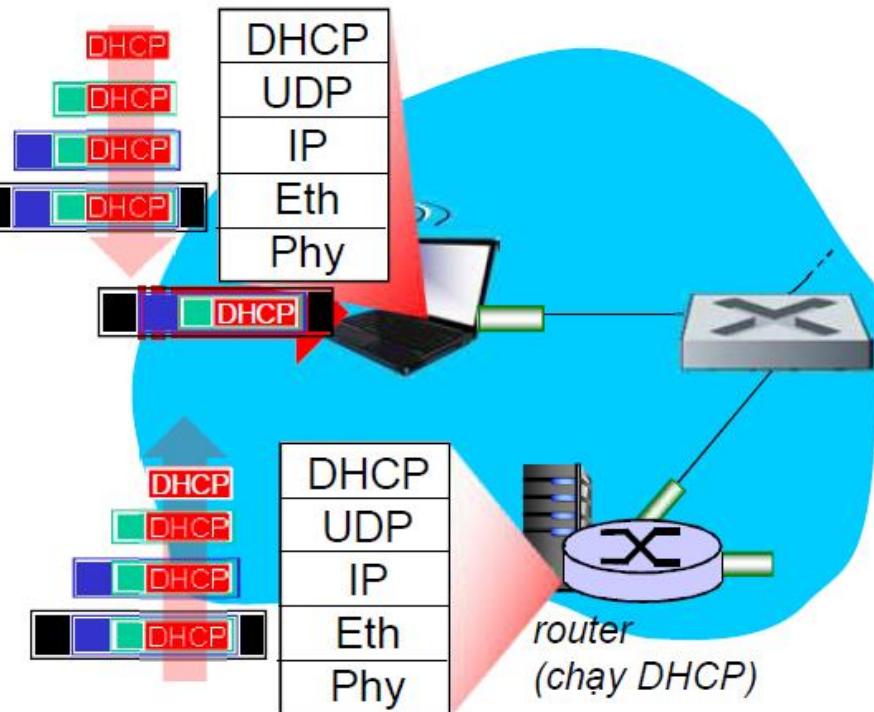
- DHCP, UDP, IP, và Ethernet
  - Bob's laptop receives the **Ethernet frame** containing the DHCP ACK, extracts the **IP datagram** from the Ethernet frame, extracts the **UDP segment** from the IP datagram, and extracts the **DHCP ACK message** from the UDP segment.
  - Bob's **DHCP client** then records its **IP address** and the **IP address of its DNS server**. It also installs the address of the default gateway into its **IP forwarding table**.

# Kịch bản: Vòng đời của một yêu cầu web

- DHCP, UDP, IP, và Ethernet
  - Bob's laptop will send all datagrams with destination address outside of its subnet 68.85.2.0/24 to the default gateway.
  - At this point, Bob's laptop has initialized its networking components and is ready to begin processing the Web page fetch.

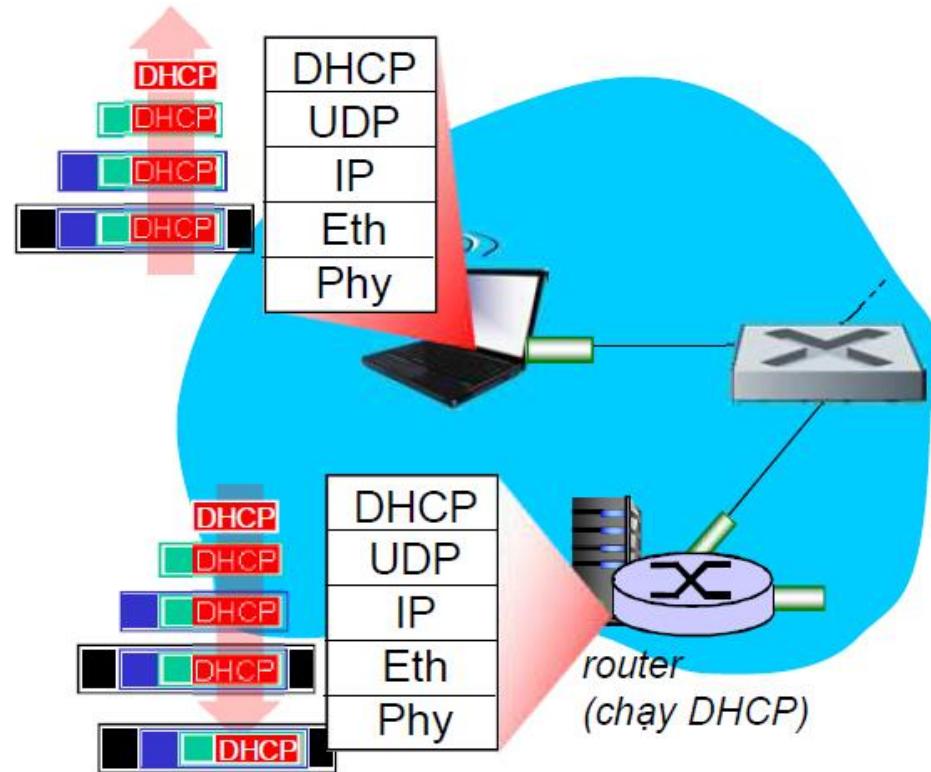
# Kịch bản: Vòng đời của một yêu cầu web

- Việc kết nối đến máy tính cần có địa chỉ IP của máy, địa chỉ của router hop đầu tiên, địa chỉ của DNS server: dùng **DHCP**
- DHCP request **được đóng gói** trong **UDP**, được đóng gói trong **IP**, được đóng gói trong **802.3** Ethernet Ethernet frame quảng bá (dest: FFFFFFFFFFFF) trên LAN, được nhận tại router đang chạy **DHCP** server
- Ethernet **mở gói** thành IP, IP **mở gói** thành UDP, UDP **mở gói** thành DHCP



# Kịch bản: Vòng đời của một yêu cầu web

- DHCP server định dạng **DHCP ACK** chứa địa chỉ IP của client, địa chỉ IP của router hop đầu tiên cho client, tên và địa chỉ IP của DNS server
- Đóng gói tại DHCP server, frame được chuyển tiếp (**học chuyển mạch**) qua LAN, việc mở gói tại client
- DHCP client nhận trả lời DHCP ACK



Bây giờ, client có địa chỉ IP, biết được tên và địa chỉ của DNS server, địa chỉ IP của router của hop đầu tiên của nó.

# Kịch bản: Vòng đời của một yêu cầu web

- DNS và ARP
  - When Bob types the URL for [www.google.com](http://www.google.com) into his Web browser, he begins the long chain of events that will eventually result in Google's home page being displayed by his Web browser.

# Kịch bản: Vòng đời của một yêu cầu web

- DNS và ARP
  - Bob's Web browser begins the process by creating a TCP socket that will be used to send the HTTP request to [www.google.com](http://www.google.com).
    - In order to create the socket, Bob's laptop will need to know the IP address of www.google.com. DNS protocol is used to provide this name-to-IP-address translation service.

# Kịch bản: Vòng đời của một yêu cầu web

- DNS và ARP
  - The operating system on Bob's laptop thus creates a DNS query message, putting the string "www.google.com" in the question section of the DNS message.
  - This **DNS message** is then placed within a **UDP segment** with a destination port of 53 (DNS server). The UDP segment is then placed within an **IP datagram** with an IP destination address of 68.87.71.226 (the address of the DNS server returned in the DHCP ACK) and a source IP address of 68.85.2.101.

# Kịch bản: Vòng đời của một yêu cầu web

- DNS và ARP
  - Bob's laptop then places the datagram containing the DNS query message in an **Ethernet frame**. This frame will be sent (addressed, at the link layer) to the gateway router in Bob's school's network.
  - However, even though Bob's laptop knows the IP address of the school's gateway router (68.85.2.1) via the DHCP ACK message, it doesn't know the gateway router's MAC address. In order to obtain the MAC address of the gateway router, Bob's laptop will need to use the **ARP protocol**

# Kịch bản: Vòng đời của một yêu cầu web

- DNS và ARP
  - Bob's laptop creates an **ARP query message** with a target IP address of 68.85.2.1 (the default gateway), places the ARP message within an **Ethernet frame** with a broadcast destination address (FF:FF:FF:FF:FF) and sends the Ethernet frame to the switch, which delivers the frame to all connected devices, including the gateway router.

# Kịch bản: Vòng đời của một yêu cầu web

- DNS và ARP
  - The gateway router receives the **frame** containing the ARP query message on the interface to the school network, and finds that the target IP address of 68.85.2.1 in the ARP message matches the IP address of its interface.
  - The gateway router thus prepares an ARP reply, indicating that its MAC address of 00:22:6B:45:1F:1B corresponds to IP address 68.85.2.1. It places the ARP reply message in an Ethernet frame, with a destination address of 00:16:D3:23:68:8A (Bob's laptop) and sends the frame to the switch, which delivers the frame to Bob's laptop.

# Kịch bản: Vòng đời của một yêu cầu web

- DNS và ARP
  - Bob's laptop receives the **frame containing the ARP reply message** and extracts the **MAC address** of the gateway router (00:22:6B:45:1F:1B) from the ARP reply message.
  - Bob's laptop can now (finally!) address the Ethernet frame containing the DNS query to the gateway router's MAC address. Note that the IP datagram in this frame has an IP destination address of 68.87.71.226 (the DNS server), while the frame has a destination address of 00:22:6B:45:1F:1B (the gateway router).
  - Bob's laptop sends this frame to the switch, which delivers the frame to the gateway router.

# Kịch bản: Vòng đời của một yêu cầu web

- Web Client-Server Interaction: TCP and HTTP
  - Now that Bob's laptop has the IP address of [www.google.com](http://www.google.com), it can create the TCP socket that will be used to send the HTTP GET message to [www.google.com](http://www.google.com).
  - When Bob creates the TCP socket, the TCP in Bob's laptop must first perform a three-way handshake with the TCP in [www.google.com](http://www.google.com).
  - Bob's laptop thus first creates a **TCP SYN segment** with destination port 80 (for HTTP), **places** the TCP segment **inside** an **IP datagram** with a destination IP address of 64.233.169.105 ([www.google.com](http://www.google.com)), **places the datagram inside a frame** with a destination MAC address of 00:22:6B:45:1F:1B (the gateway router) and sends the frame to the switch.

# Kịch bản: Vòng đời của một yêu cầu web

- Web Client-Server Interaction: TCP and HTTP
  - The routers in the school network, Comcast's network, and Google's network forward the datagram containing the TCP SYN toward www.google.com, using the forwarding table in each router, as in steps above.
    - Recall that the router forwarding table entries governing forwarding of packets over the inter-domain link between the Comcast and Google networks are determined by the BGP protocol

# Kịch bản: Vòng đời của một yêu cầu web

- Web Client-Server Interaction: TCP and HTTP
  - Eventually, the **datagram** containing the TCP SYN arrives at www.google.com. The **TCP SYN message** is extracted from the datagram and demultiplexed to the welcome socket associated with port 80.
  - A **connection socket** is created for the TCP connection between the Google HTTP server and Bob's laptop. A **TCP SYNACK segment** is generated, placed inside a datagram addressed to Bob's laptop, and finally placed inside a link-layer frame appropriate for the link connecting www.google.com to its first-hop router.

# Kịch bản: Vòng đời của một yêu cầu web

- Web Client-Server Interaction: TCP and HTTP
  - The **datagram** containing the **TCP SYNACK** segment is forwarded through the Google, Comcast, and school networks, eventually arriving at the Ethernet card in Bob's laptop.
  - The **datagram** is **demultiplexed** within the operating system to the TCP socket, which enters the connected state.

# Kịch bản: Vòng đời của một yêu cầu web

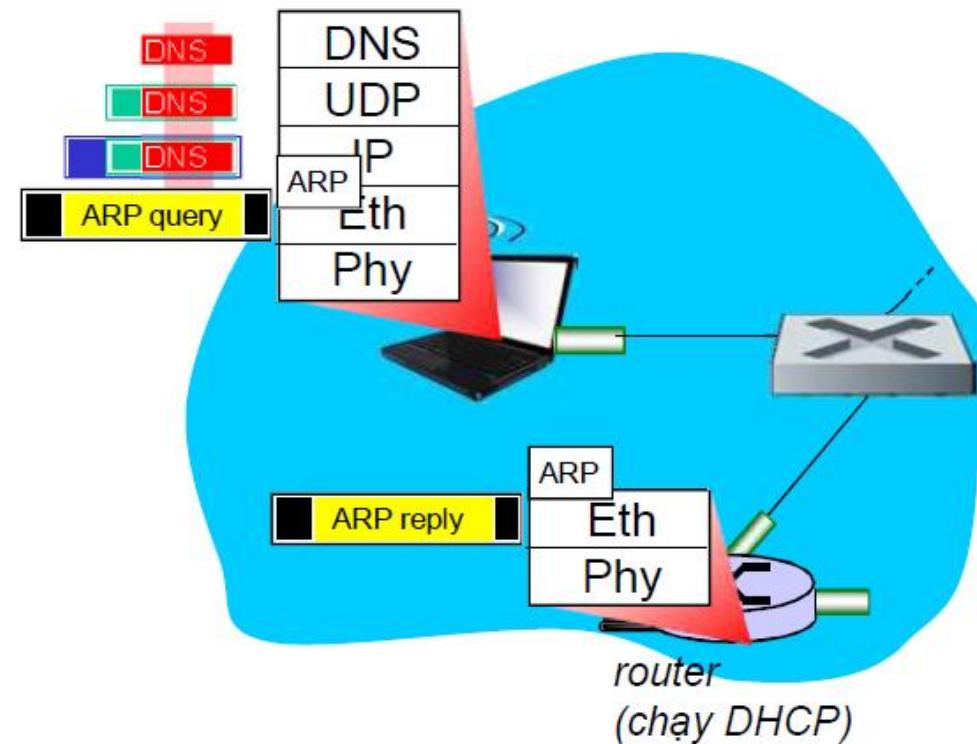
- Web Client-Server Interaction: TCP and HTTP
  - With the socket on Bob's laptop now (finally!) ready to send bytes to www.google.com, Bob's browser creates the **HTTP GET message** containing the URL to be fetched.
  - The HTTP GET message is then written into the socket, with the **GET message** becoming the **payload** of a **TCP segment**. The TCP segment is placed in a **datagram** and sent and delivered to www.google.com as in steps above.

# Kịch bản: Vòng đời của một yêu cầu web

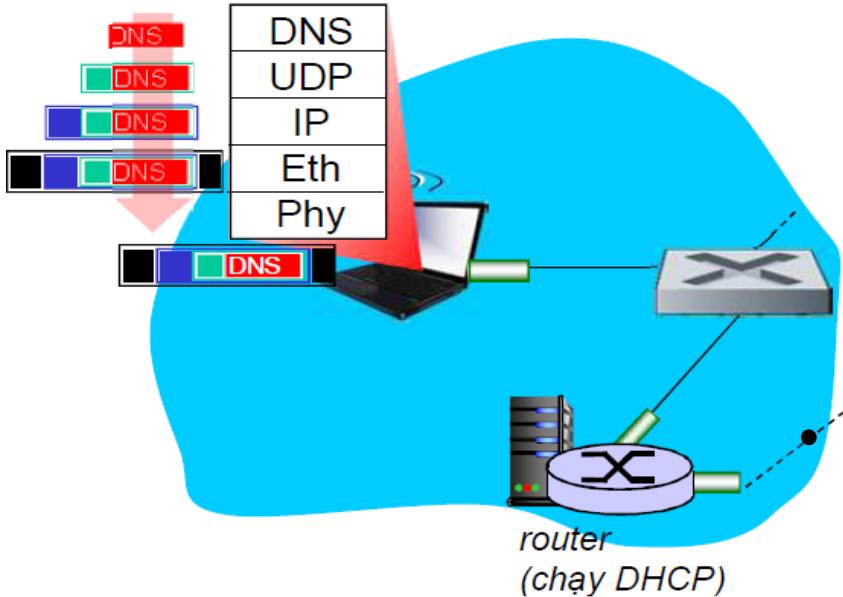
- Web Client-Server Interaction: TCP and HTTP
  - The HTTP server at [www.google.com](http://www.google.com) reads the HTTP GET message from the TCP socket, creates an HTTP response message, places the requested Web page content in the body of the HTTP response message, and sends the message into the TCP socket.
  - The **datagram containing the HTTP reply message** is forwarded through the Google, Comcast, and school networks, and arrives at Bob's laptop.
  - **Bob's Web browser** program reads the HTTP response from the socket, extracts the html for the Web page from the body of the HTTP response, and finally displays the Web page!

# ARP (trước DNS, trước HTTP)

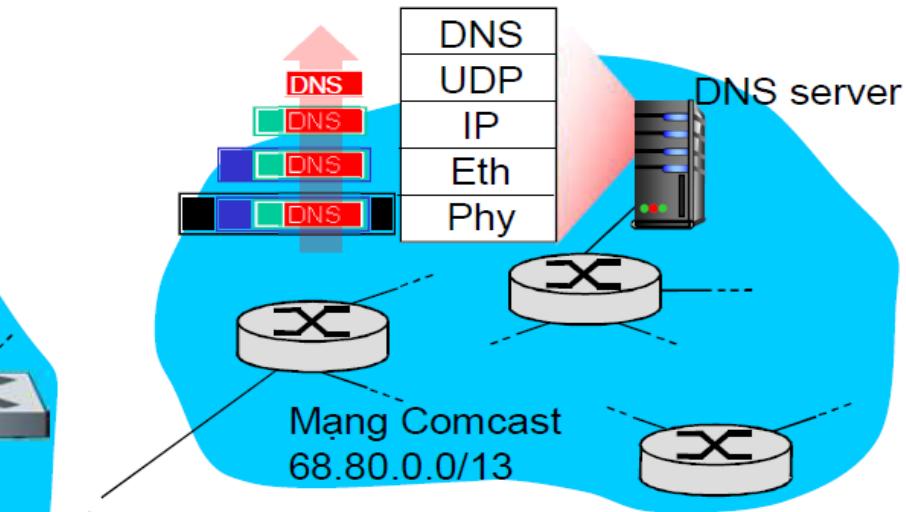
- Trước khi gửi yêu cầu **HTTP**, cần địa chỉ IP của www.google.com: **DNS**
- Truy vấn DNS được tạo ra, được đóng gói trong UDP, được đóng gói trong IP, được đóng gói trong Ethernet. Để gửi frame tới router, cần địa chỉ MAC của giao diện router: **ARP**
- ARP query** quảng bá, được nhận bởi router mà sẽ trả lời với **ARP reply**, cho biết địa chỉ MAC của giao diện router.
- Lúc này, client biết địa chỉ MAC của router hop đầu tiên, do đó có thể gửi frame chứa truy vấn DNS



# Sử dụng DNS

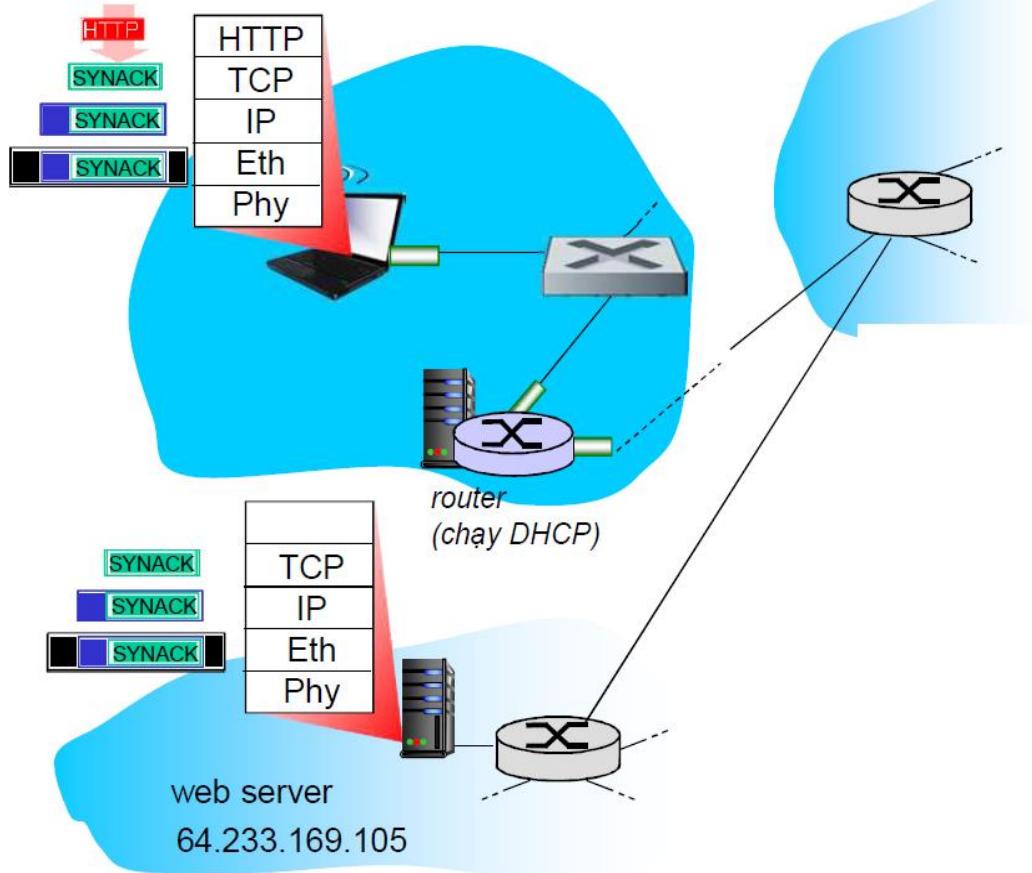


IP datagram chứa truy vấn DNS được chuyển tiếp qua switch LAN từ client tới router hop đầu tiên



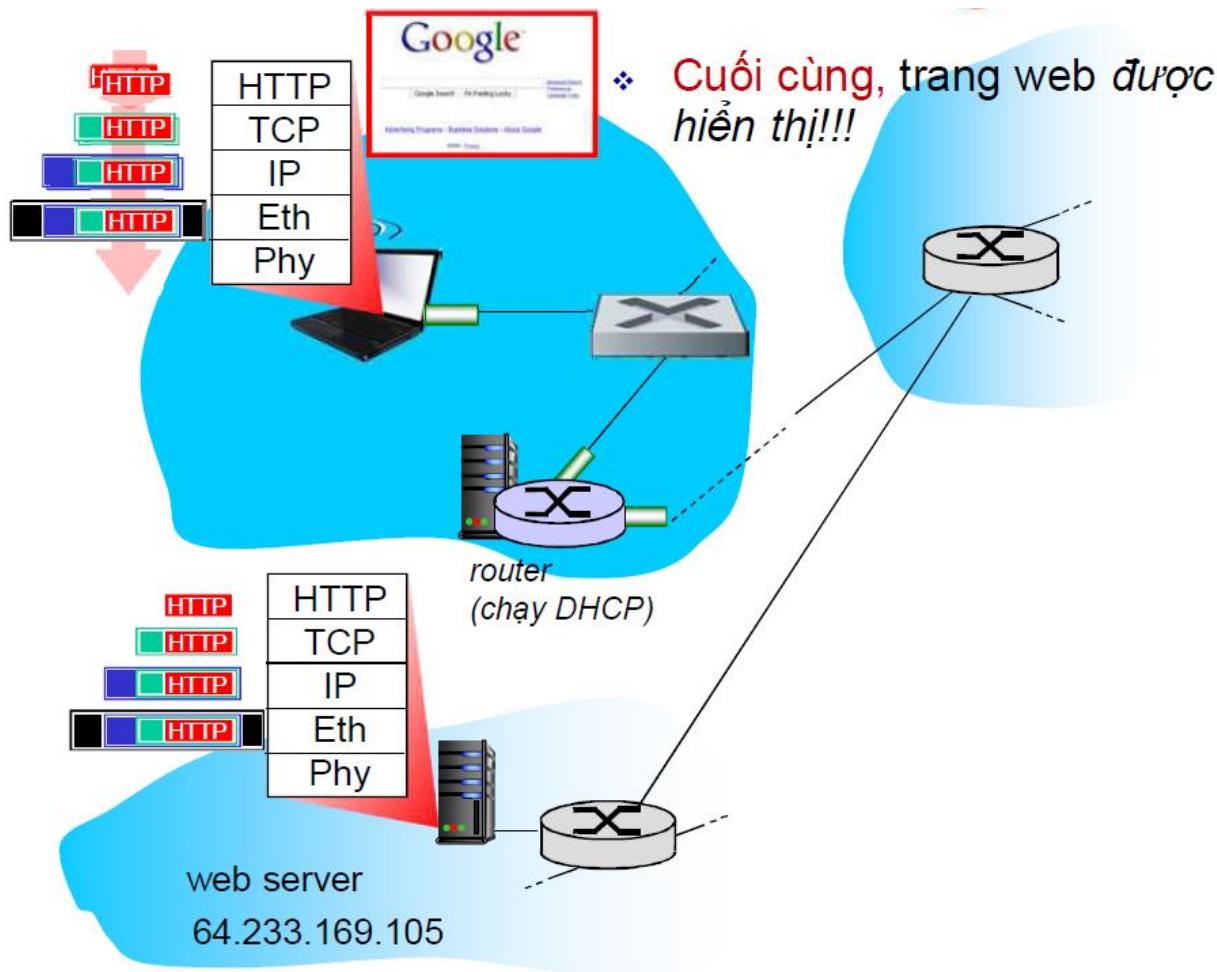
- IP datagram được chuyển tiếp từ mạng trong trường vào trong mạng comcast, được định tuyến (các bảng được tạo ra bởi các giao thức định tuyến **RIP**, **OSPF**, **IS-IS** và/hoặc **BGP**) tới DNS server
- DNS server trả lời lại client với địa chỉ IP của www.google.com

# Kết nối TCP mang thông điệp HTTP



- Để gửi yêu cầu HTTP, đầu tiên client mở **TCP socket** tới web server
- TCP **SYN segment** (bước 1 trong bắt tay 3 bước) định tuyến ngoại miền (interdomain) tới web server
- Web server trả lời với **TCP SYNACK** (bước 2 trong bắt tay 3 bước)
- Kết nối TCP **được thiết lập!**

# HTTP yêu cầu/đáp ứng



- **HTTP yêu cầu** được gửi vào trong TCP socket
- IP datagram chứa HTTP request được gửi tới www.google.com
- IP datagram chứa đáp ứng HTTP được gửi quay lại client

# Tổng kết

- Nguyên lý các dịch vụ bên trong tầng liên kết dữ liệu:
  - Phát hiện và sửa lỗi
  - Chia sẻ các kênh truyền chung: đa truy nhập
  - Định địa chỉ tầng liên kết
- Hiện thực và cài đặt một số công nghệ tầng liên kết
  - Ethernet
  - switched LANS, VLANs
  - MPLS
- Tổng hợp: Vòng đời của một yêu cầu web