

# Mạng máy tính

**TS. Phạm Tuấn Minh**

Khoa Công nghệ Thông tin, Đại học Thủy lợi

[minhpt@tlu.edu.vn](mailto:minhpt@tlu.edu.vn)

<http://netlab.tlu.edu.vn/~minhpt/>

# Chương 4: Tầng mạng

## 4.1 Giới thiệu

### 4.2 IP: Internet Protocol

- Cấu trúc datagram
- IPv4
- ICMP
- IPv6

### 4.3 Giải thuật dẫn đường

- link state
- distance vector
- hierarchical routing

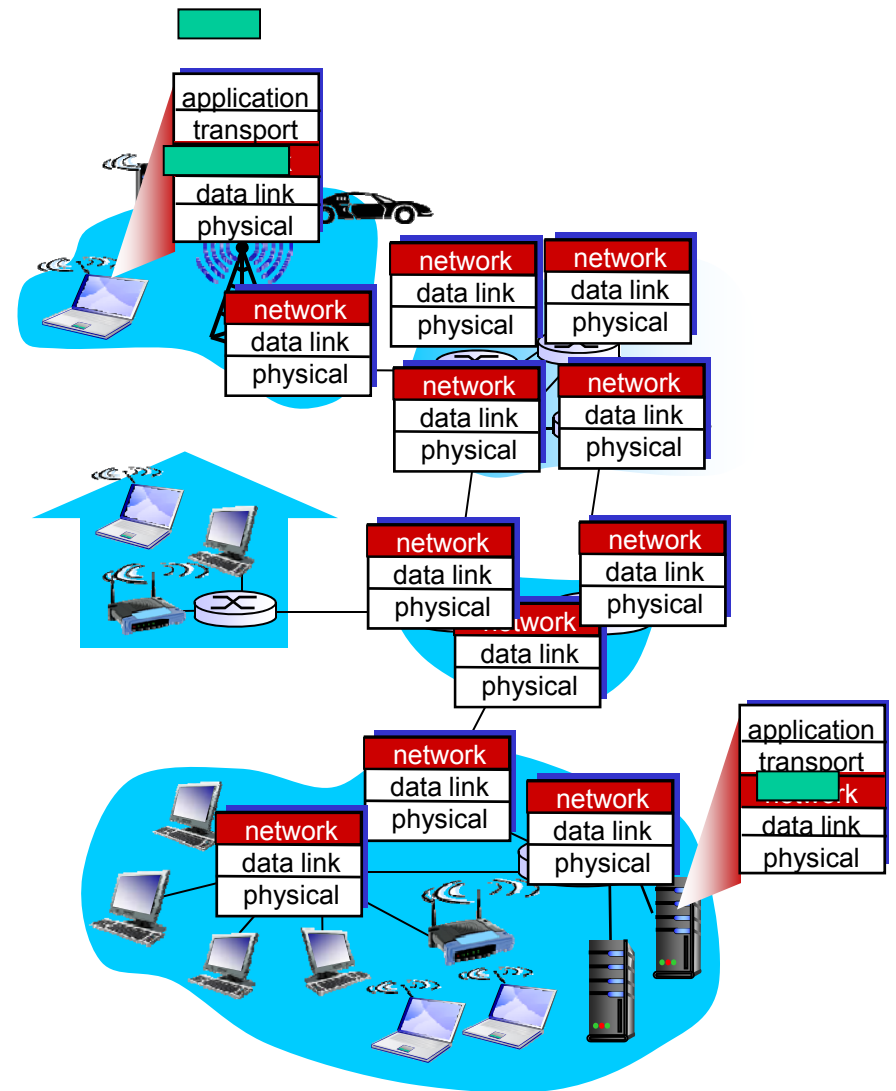
### 4.4 Dẫn đường trong Internet

- RIP
- OSPF
- BGP

### 4.5 Broadcast routing và multicast routing

# Tầng mạng

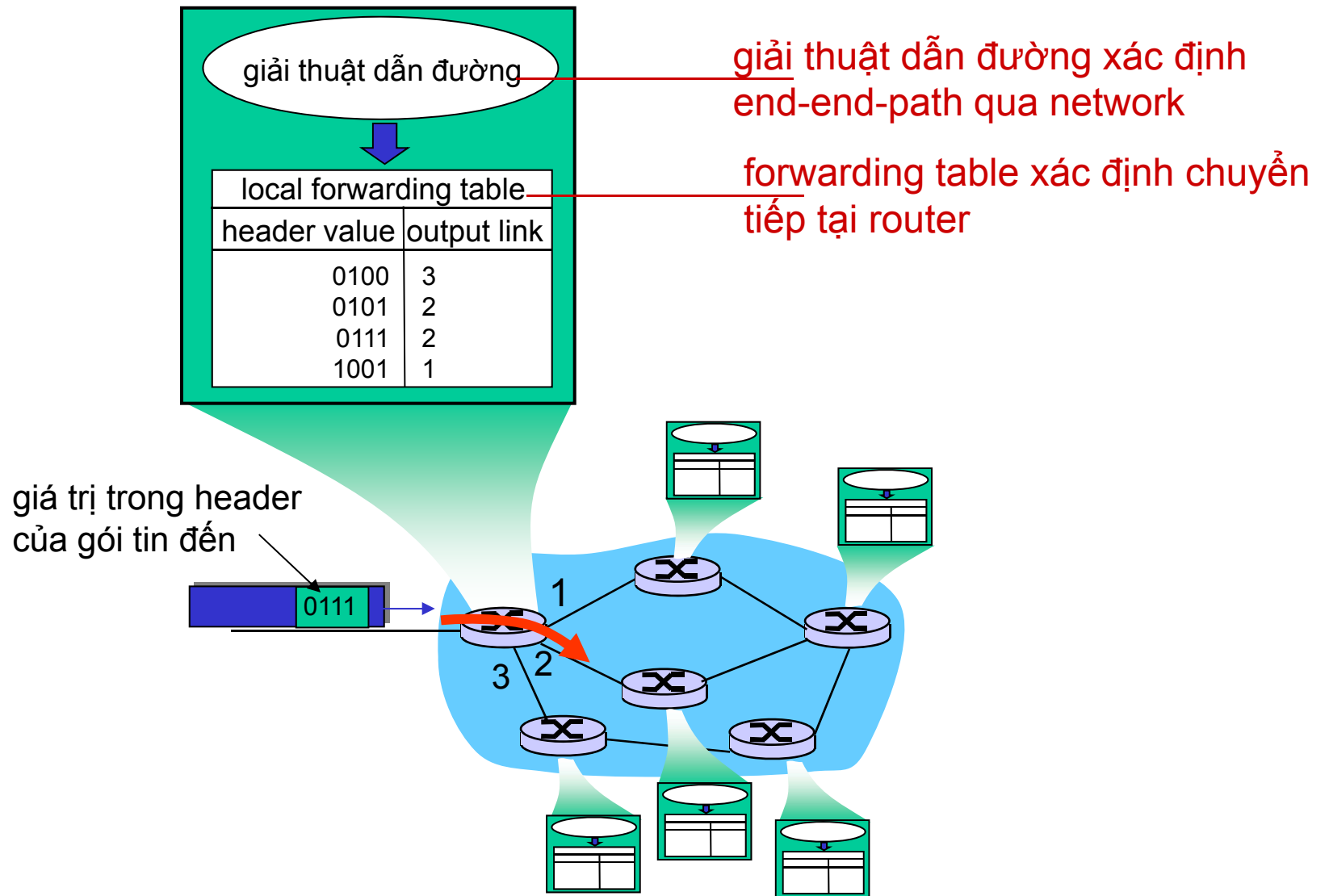
- ❑ Segment của tầng giao vận từ nút gửi tới nút nhận
- ❑ Phía gửi, đóng segment trong các datagram
- ❑ Phía nhận, chuyển segment tới tầng giao vận
- ❑ Giao thức của tầng mạng có trong mọi host và router
- ❑ Router đọc các trường header trong các IP datagram đi qua router



# Hai chức năng chính của tầng mạng

- ❑ *Chuyển tiếp (forwarding)*: chuyển gói tin từ đầu vào của router ra đầu ra thích hợp của router
- ❑ *Dẫn đường (routing)*: xác định đường đi của gói tin từ nguồn tới đích
  - *giải thuật dẫn đường*

# Dẫn đường và chuyển tiếp



# Thiết lập kết nối

- ❑ Chức năng quan trọng thứ ba trong một số kiến trúc mạng:
  - ATM, frame relay, X.25
- ❑ Trước khi gửi dữ liệu, hai host và các router trên đường đi tạo một kết nối ảo
  - router tham gia vào thiết lập kết nối
- ❑ Dịch vụ kết nối của tầng giao vận và dịch vụ kết nối của tầng mạng:
  - *tầng mạng*: giữa hai host (có thể bao gồm các router trên đường đi với mạng chuyển mạch kênh)
  - *tầng giao vận*: giữa hai tiến trình

# Mô hình dịch vụ mạng

Mô hình dịch vụ (*service model*) nào để gửi datagram từ nút gửi tới nút nhận?

*ví dụ dịch vụ cho từng datagram riêng lẻ:*

- ❑ gửi đảm bảo (guaranteed delivery)
- ❑ gửi đảm bảo với độ trễ nhỏ hơn 40 msec

*ví dụ dịch vụ cho luồng datagram:*

- ❑ gửi đảm bảo thứ tự datagram
- ❑ đảm bảo băng thông tối thiểu cho luồng datagram
- ❑ hạn chế sự thay đổi độ trễ giữa các gói tin (jitter)

# Các mô hình dịch vụ tầng mạng

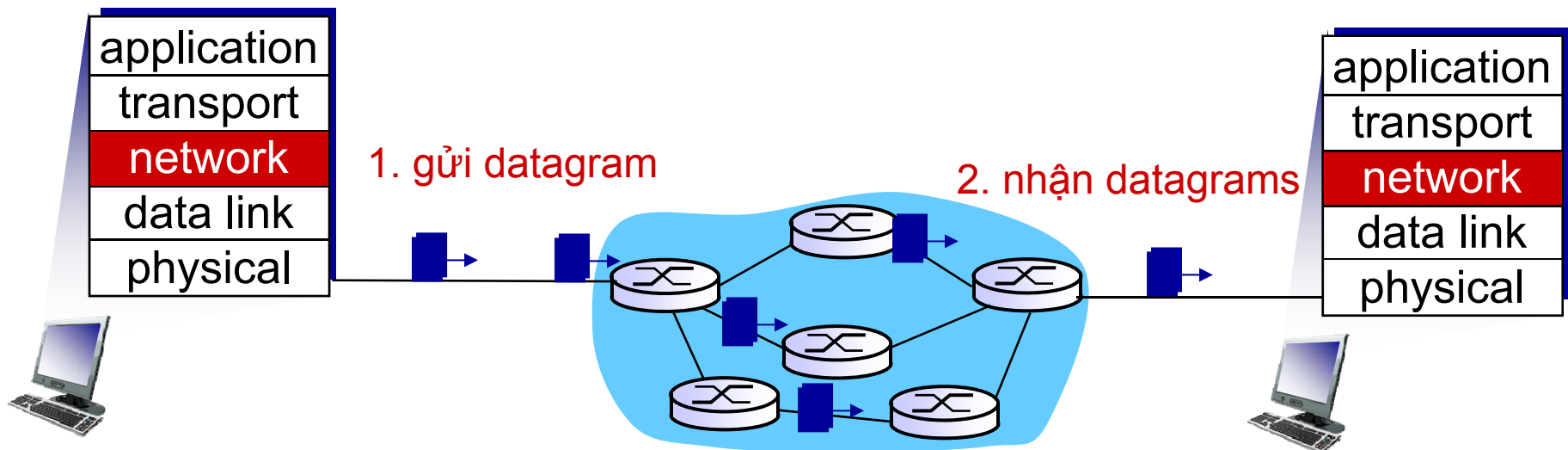
Kiến trúc mạng	Mô hình dịch vụ	Đảm bảo?				Phản hồi tắc nghẽn
		Băng thông	Mất gói	Thứ tự	Thời gian	
Internet	best effort	Không	Không	Không	Không	Không (dựa vào mất gói)
ATM	CBR	Tốc độ hằng số	Có	Có	Có	Không tắc nghẽn
ATM	VBR	đảm bảo tốc độ	Có	Có	Có	Không tắc nghẽn
ATM	ABR	đảm bảo tối thiểu	Không	Có	Không	Có
ATM	UBR	Không	Không	Có	Không	Không



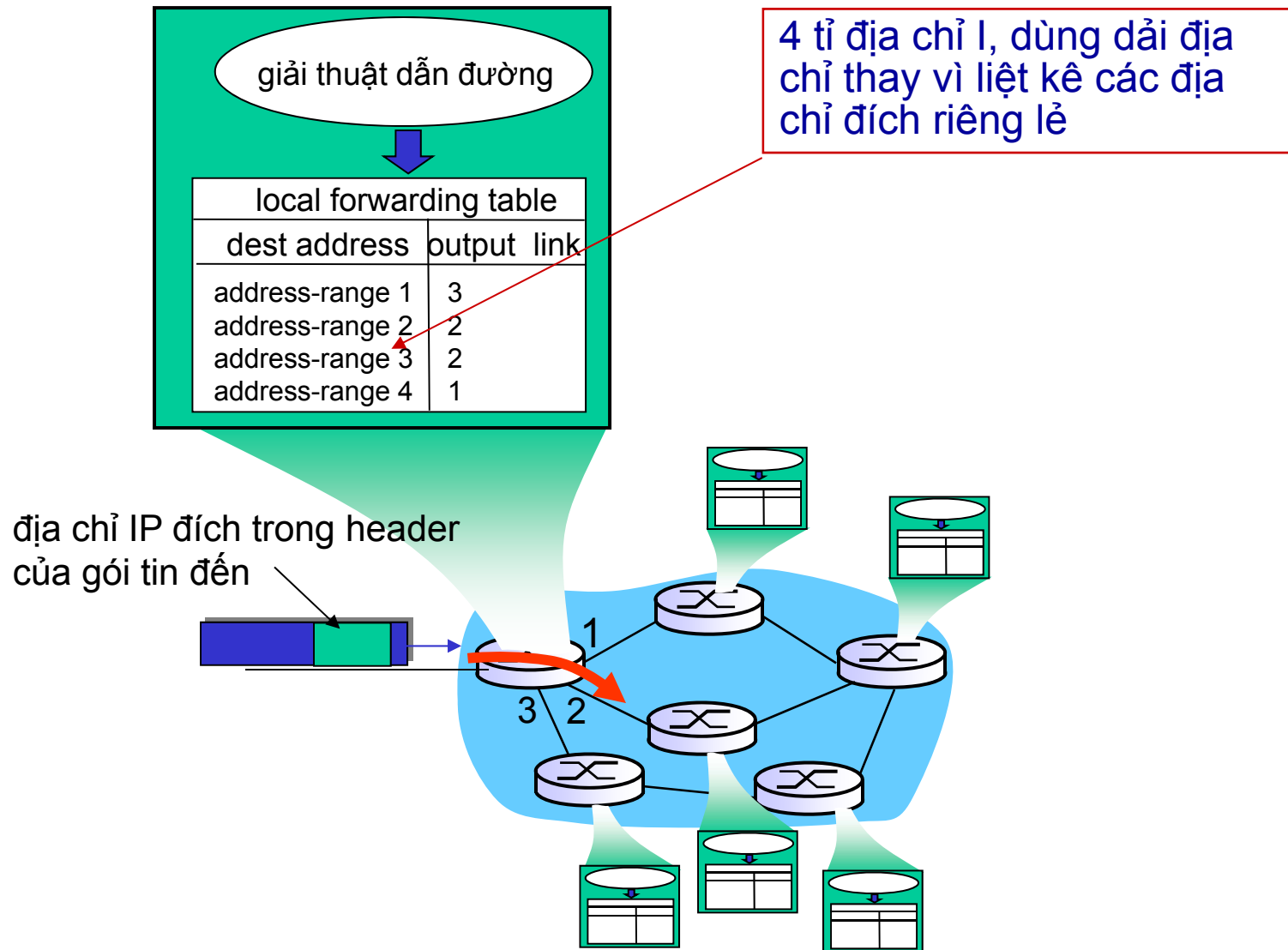
# Mạng chuyển mạch gói

Mạng chuyển mạch gói (Datagram network)

- ❑ không thiết lập kết nối tại tầng mạng
- ❑ router: không lưu trạng thái về các end-to-end connection
  - không có khái niệm kết nối (connection) tại mức mạng
- ❑ gói tin được chuyển đi dựa vào địa chỉ của destination host



# Bảng chuyển tiếp (Datagram forwarding table)



# Bảng chuyển tiếp

Dải địa chỉ đích	Liên kết ra
11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
otherwise	3

Vấn đề gì xảy ra nếu dải địa chỉ không được chia liên tục?

# Longest prefix matching

## *longest prefix matching*

khi tìm kiếm một dòng của bảng chuyển tiếp cho một địa chỉ IP đích, dùng dòng địa chỉ mà phần đầu địa chỉ dài nhất giống với địa chỉ đích

Dải địa chỉ đích	Liên kết ra
11001000 00010111 00010*** *****	0
11001000 00010111 00011000 *****	1
11001000 00010111 00011*** *****	2
khác	3

ví dụ:

DA: 11001000 00010111 00010110 10100001

liên kết ra?

DA: 11001000 00010111 00011000 10101010

liên kết ra?

# Chương 4: Tầng mạng

## 4.1 Giới thiệu

## 4.2 IP: Internet Protocol

- Cấu trúc datagram
- IPv4
- ICMP
- IPv6

## 4.3 Giải thuật dẫn đường

- link state
- distance vector
- hierarchical routing

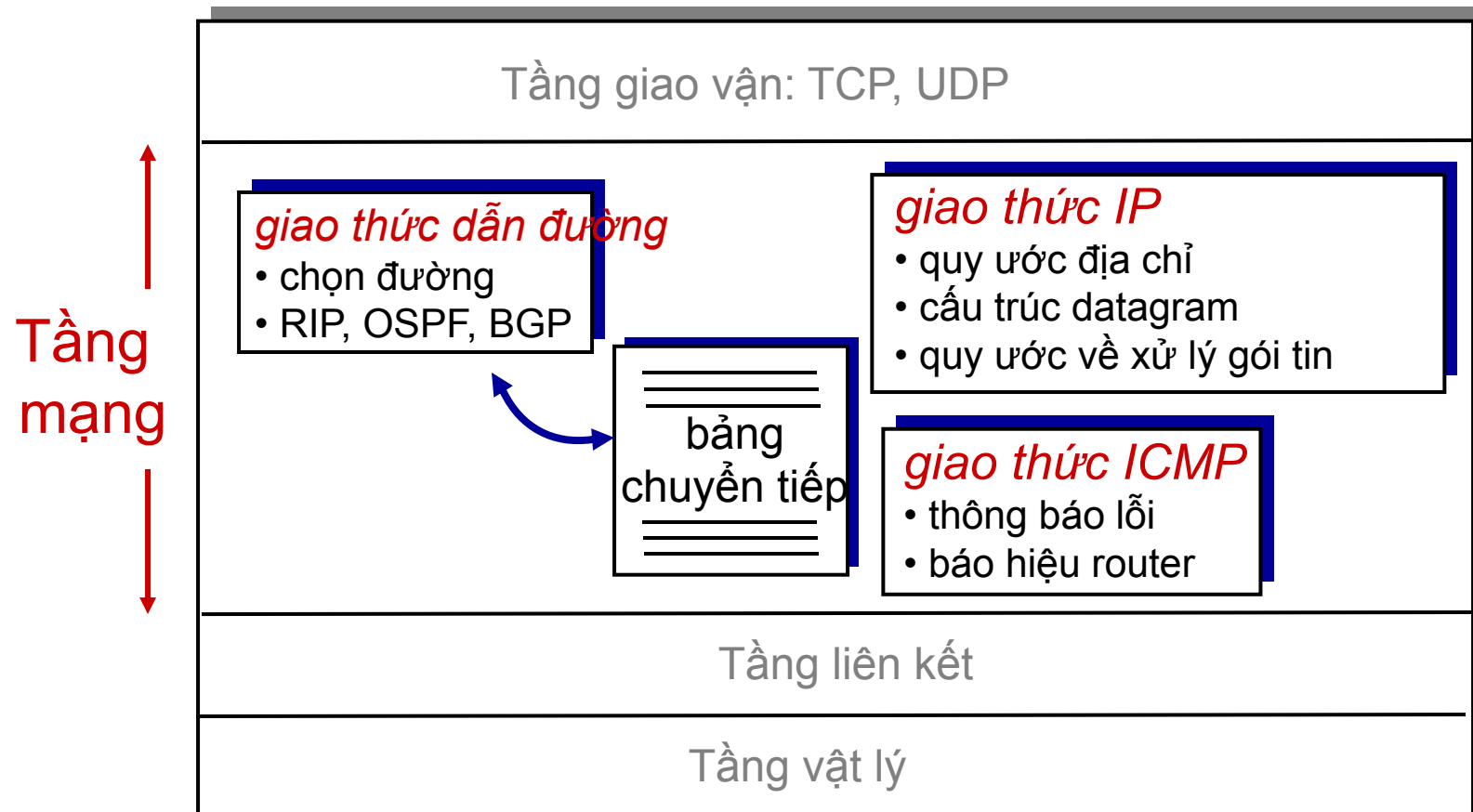
## 4.4 Dẫn đường trong Internet

- RIP
- OSPF
- BGP

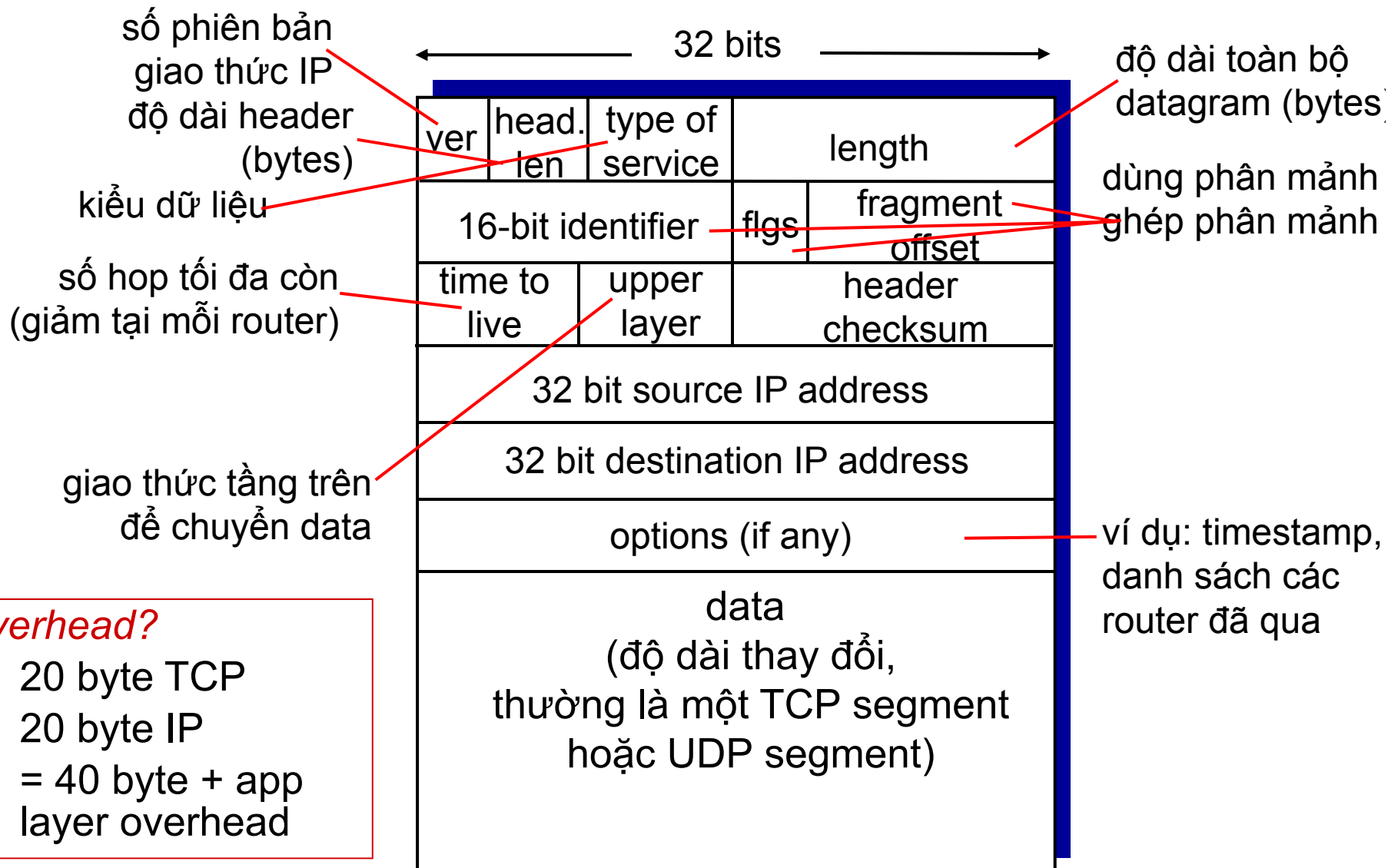
## 4.5 Broadcast routing và multicast routing

# Tầng mạng của Internet

Chức năng tầng mạng:

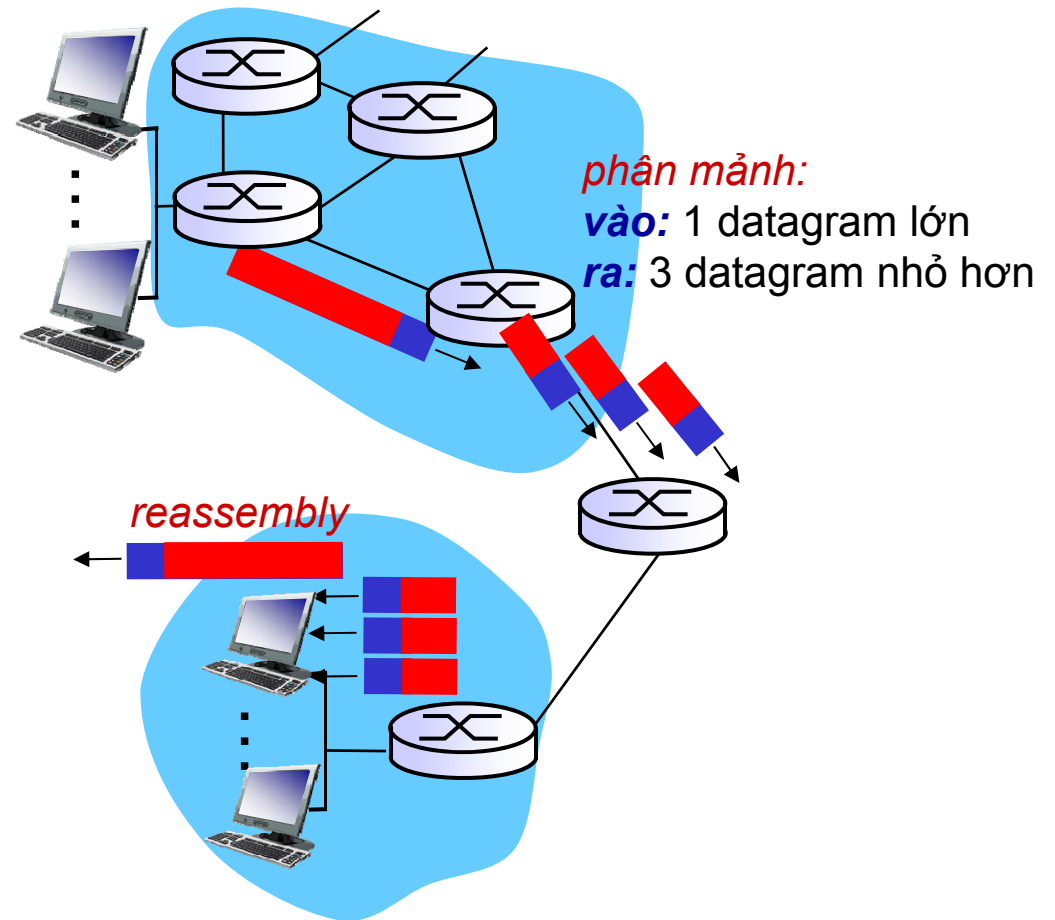


# Cấu trúc của IP datagram



# Phân mảnh IP datagram

- ❑ liên kết mạng có MTU (max.transfer size)
  - kiểu liên kết khác nhau có MTU khác nhau
- ❑ IP datagram lớn được chia nhỏ trong mạng
  - 1 datagram thành nhiều datagram
  - được ghép lại tại đích
  - Các bit trong IP header dùng để ghép các phân mảnh





# Ghép phân mảnh

*ví dụ:*

- ❖ datagram kích thước 4000 byte
- ❖ MTU = 1500 byte

	length	ID	fragflag	offset
	=4000	=x	=0	=0

*1 datagram lớn được chia thành nhiều datagram nhỏ hơn*

1480 byte phần dữ liệu

offset =  
1480/8

	length	ID	fragflag	offset
	=1500	=x	=1	=0

	length	ID	fragflag	offset
	=1500	=x	=1	=185

	length	ID	fragflag	offset
	=1040	=x	=0	=370

# Chương 4: Tầng mạng

## 4.1 Giới thiệu

## 4.2 IP: Internet Protocol

- Cấu trúc datagram
- IPv4
- ICMP
- IPv6

## 4.3 Giải thuật dẫn đường

- link state
- distance vector
- hierarchical routing

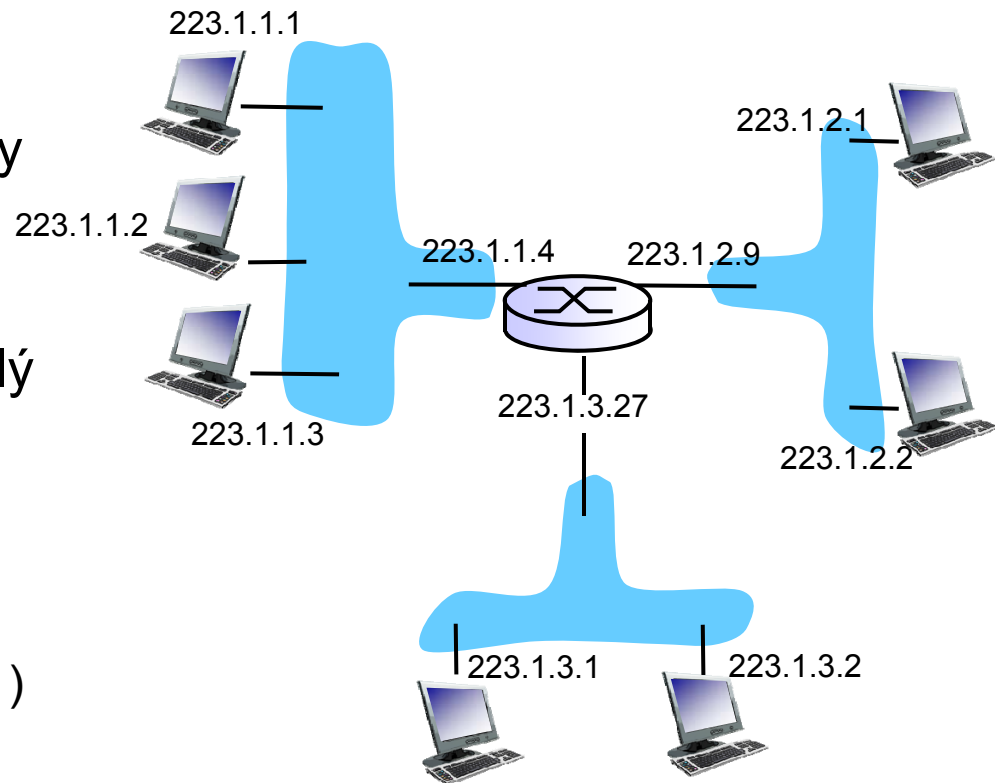
## 4.4 Dẫn đường trong Internet

- RIP
- OSPF
- BGP

## 4.5 Broadcast routing và multicast routing

# Địa chỉ IPv4

- ❑ **Địa chỉ IPv4:** 32 bit xác định interface của host hay router
- ❑ **interface:** kết nối giữa host/router và liên kết vật lý
  - router thường có nhiều interfac
  - host thường có 1 hoặc 2 interface (ví dụ: wired Ethernet, wireless 802.11)
- ❑ **Địa chỉ IP gán cho mỗi interface**



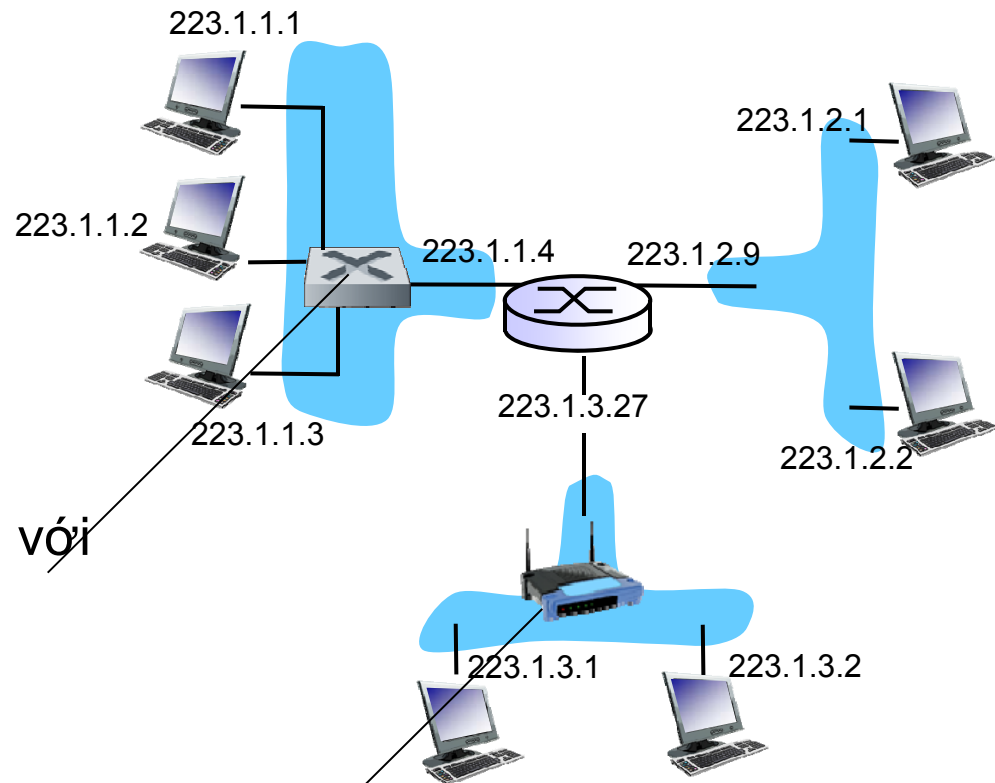
$$223.1.1.1 = \underbrace{11011111}_{223} \underbrace{00000001}_1 \underbrace{00000001}_1 \underbrace{00000001}_1$$

# Địa chỉ IPv4

*Các interface kết nối với nhau như thế nào?*

Các Ethernet interface kết nối với nhau qua Ethernet switches

Các interface cũng có thể kết nối trực tiếp với nhau



Các WiFi interface kết nối với nhau qua WiFi base station

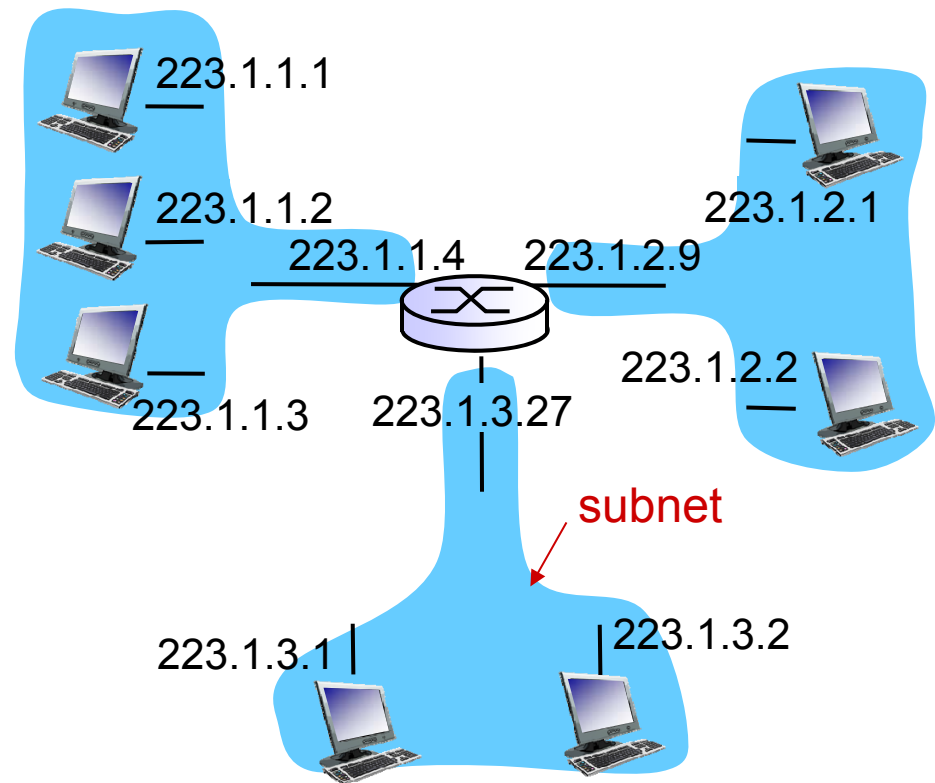
# Subnet

## □ Địa chỉ IP:

- Phần subnet: các bit cao
- Phần host: các bit thấp

## □ *subnet*

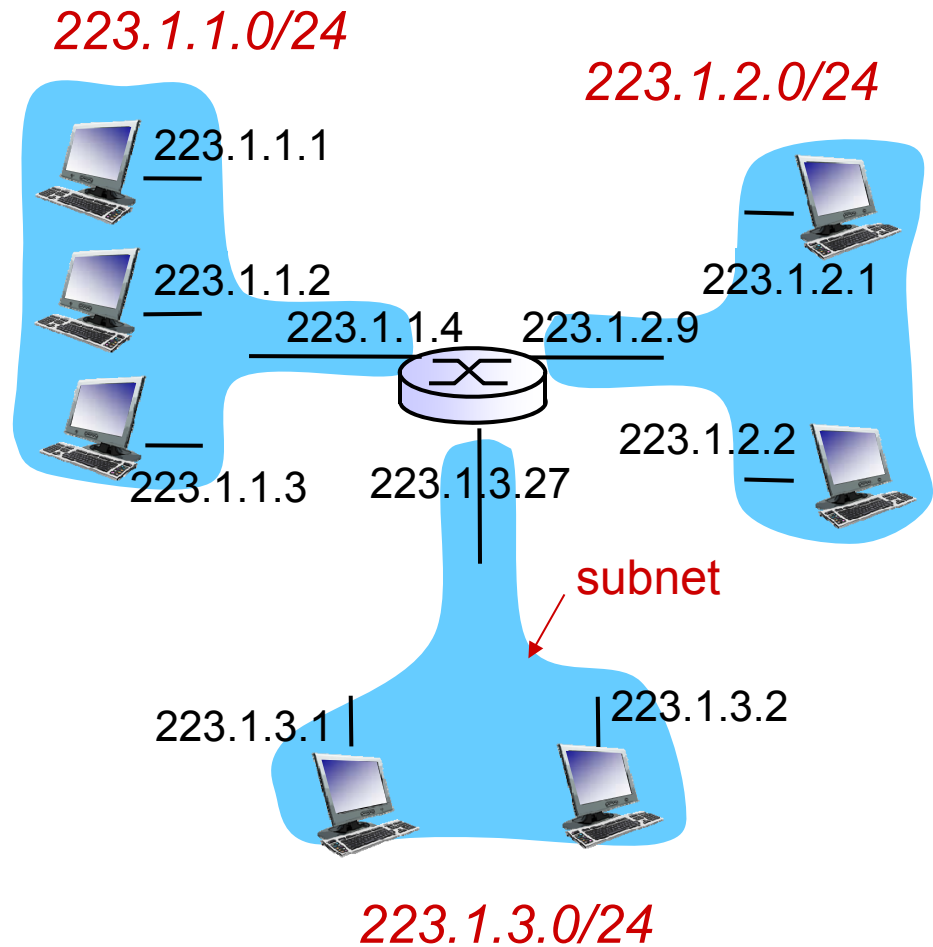
- các interface có cùng phần subnet của địa chỉ IP
- có thể giao tiếp với nhau không cần qua router



network consisting of 3 subnets

# Subnet

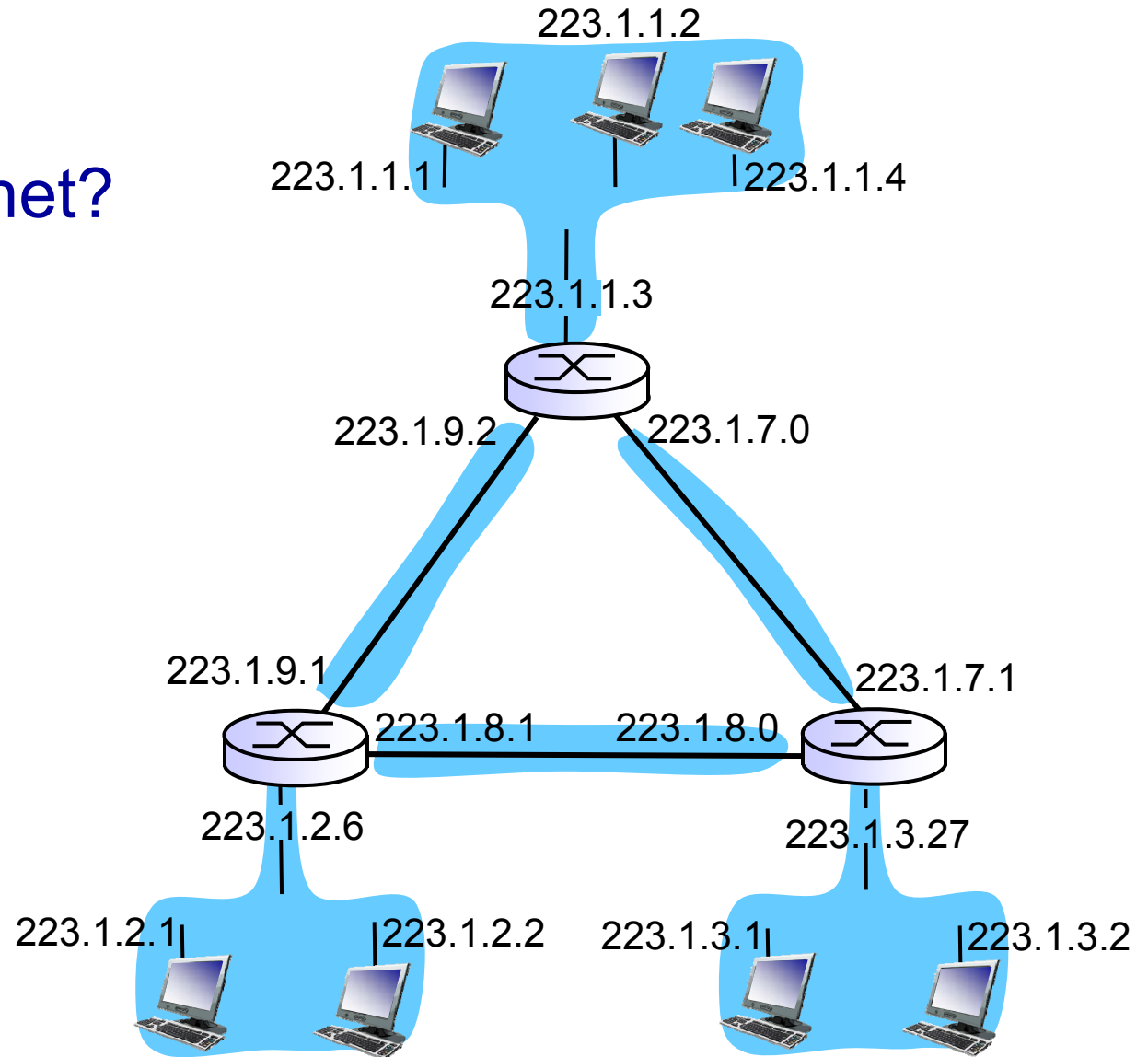
- ❑ để xác định các subnet, bỏ interface khỏi router tạo thành các mạng cô lập
- ❑ mỗi mạng cô lập này là một subnet



subnet mask: /24

# Subnet

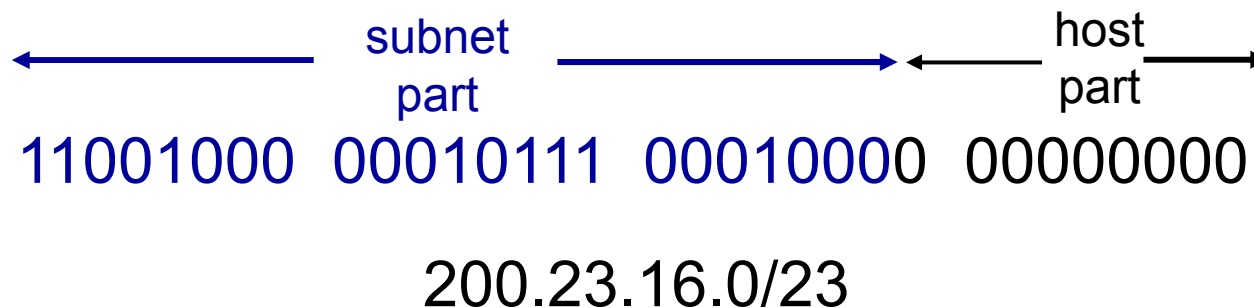
Số lượng subnet?



# Đánh địa chỉ IP: CIDR

## CIDR: Classless InterDomain Routing

- cấu trúc địa chỉ: **a.b.c.d/x**, trong đó x là số bit trong phần subnet của địa chỉ





# Làm sao một host có địa chỉ IP

- ❑ khai báo cố định bởi người quản trị hệ thống
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- ❑ **DHCP: Dynamic Host Configuration Protocol:** nhận địa chỉ động từ server
  - “plug-and-play”

# DHCP: Dynamic Host Configuration Protocol

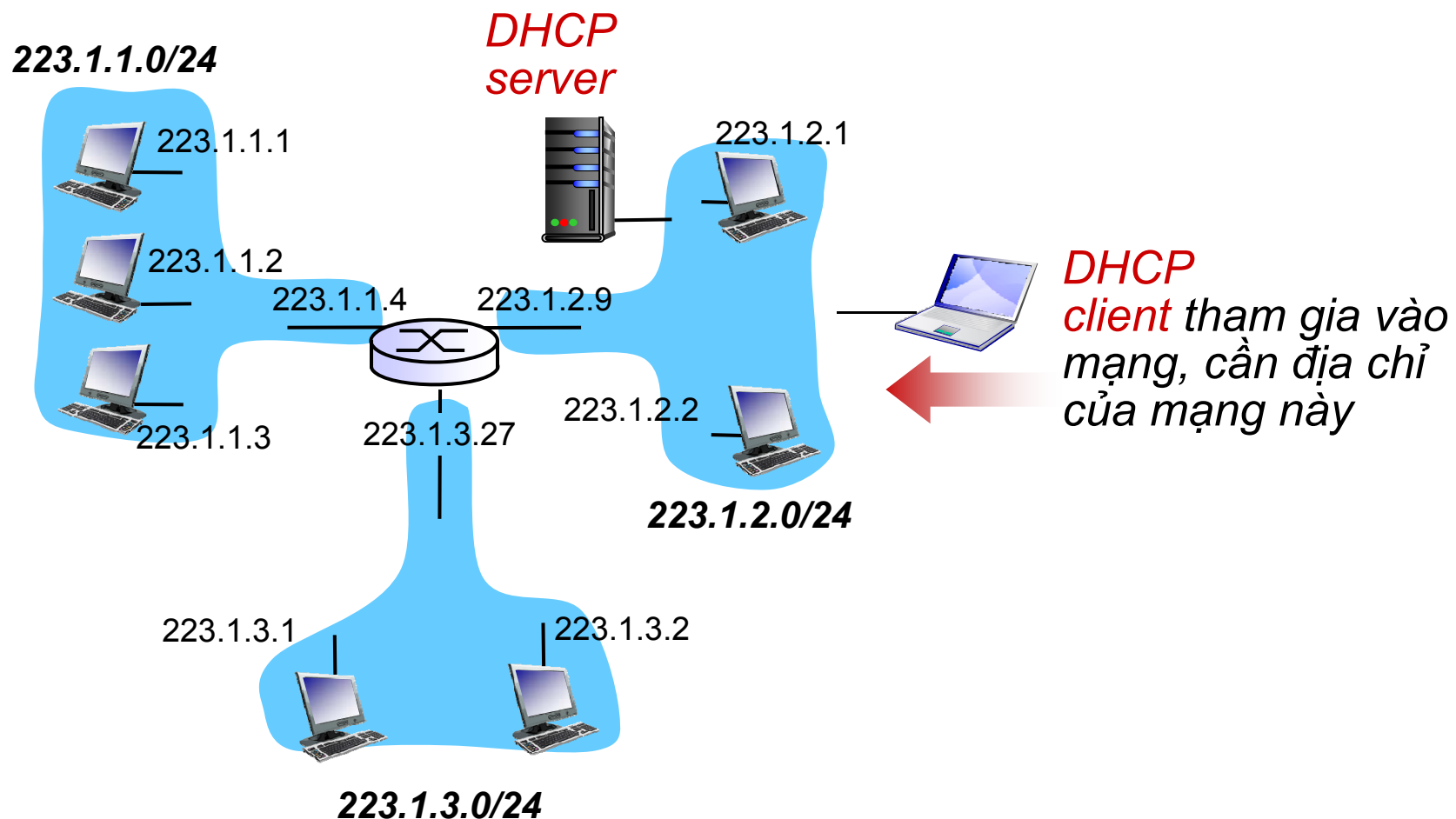
***DHCP:*** cho phép host lấy địa chỉ IP động từ network server khi tham gia vào mạng

- có thể làm mới địa chỉ đang dùng
- có thể sử dụng lại các địa chỉ (chỉ giữ địa chỉ khi kết nối)
- cho phép thiết bị di động tham gia vào mạng (thời gian kết nối ngắn)

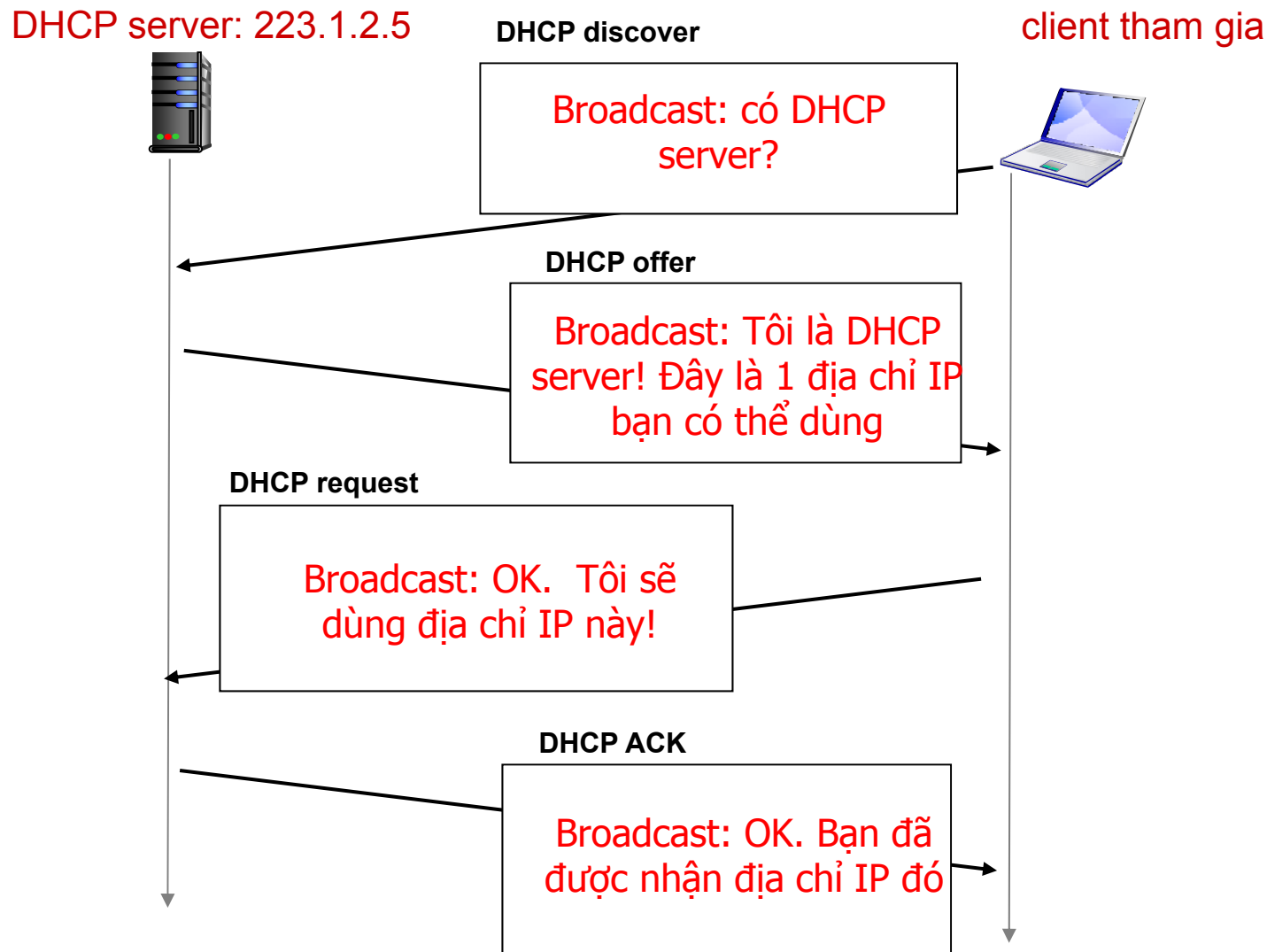
***Cơ bản về hoạt động của DHCP:***

- host quảng bá bản tin “**DHCP discover**” [tùy chọn]
- DHCP server trả lời bằng bản tin “**DHCP offer**” [tùy chọn]
- host yêu cầu địa chỉ IP bằng bản tin “**DHCP request**”
- DHCP server gửi địa chỉ: bản tin “**DHCP ack**”

# Kịch bản DHCP client-server



# Kịch bản DHCP client-server

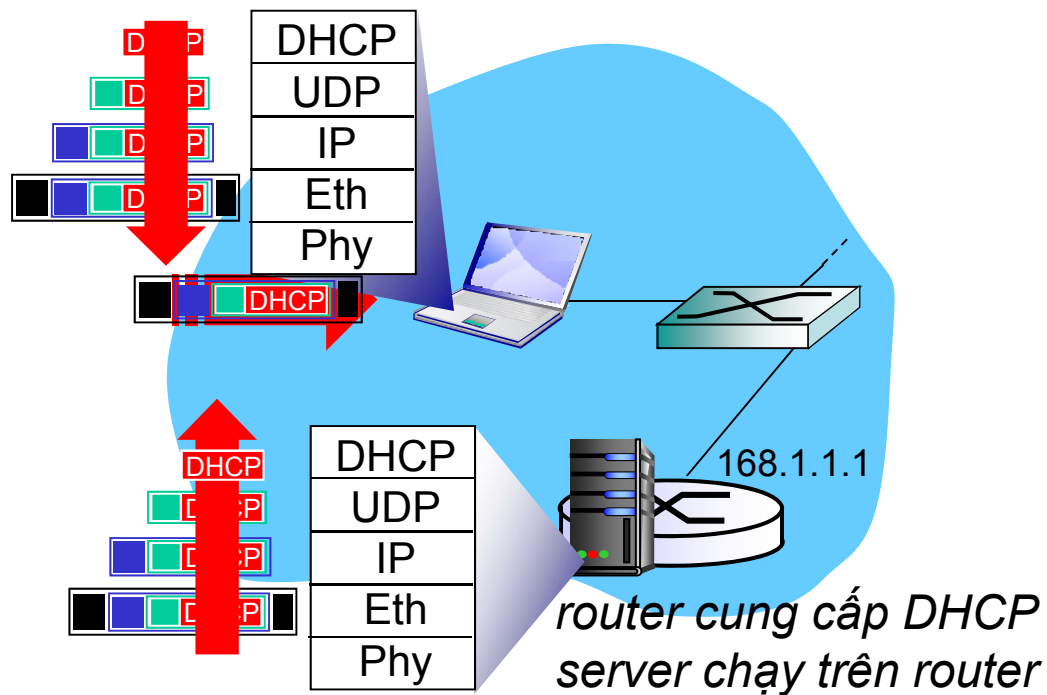


# Thông tin trả về của DHCP

DHCP có thể trả về các thông tin khác ngoài địa chỉ IP cấp phát trong subnet:

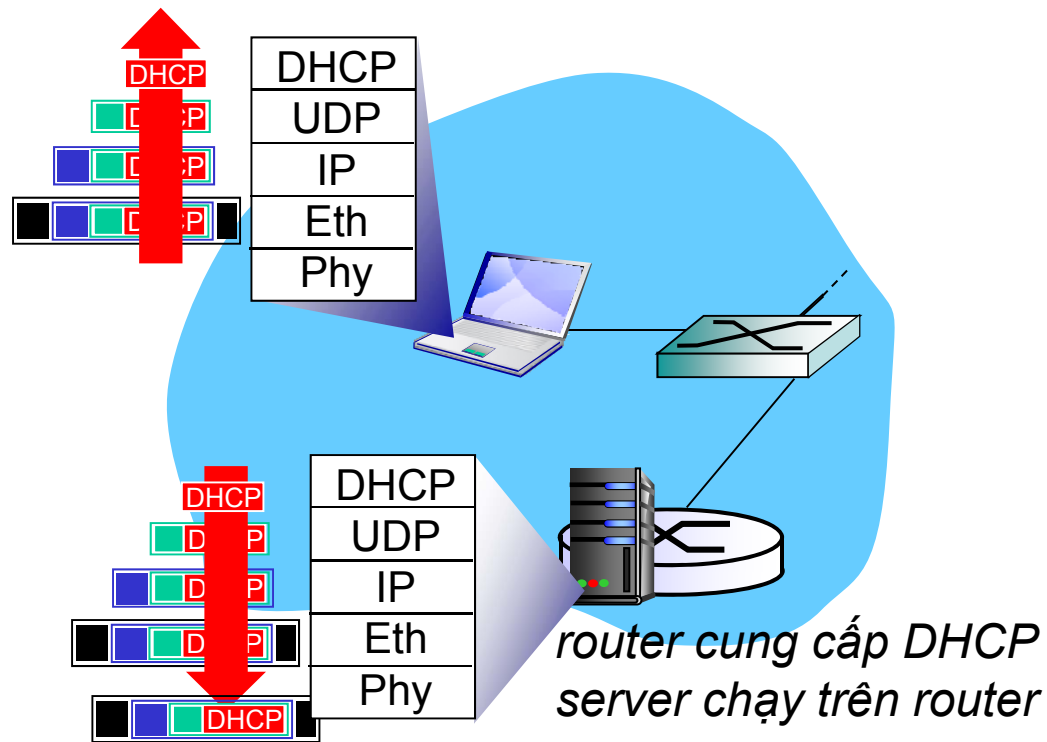
- địa chỉ của first-hop router cho client
- tên và địa chỉ IP của DNS sever
- network mask (phân biệt phần mạng và phần host của địa chỉ IP)

# Ví dụ DHCP



- ❑ kết nối laptop cần địa chỉ IP của máy, địa chỉ của first-hop router, địa chỉ DNS server: dùng DHCP
- ❖ DHCP request đóng gói trong UDP, đóng tiếp trong IP, đóng tiếp trong 802.1 Ethernet
- ❖ Ethernet frame quảng bá (đích: FFFFFFFFFFFFFFFF) trong LAN, router chạy DHCP server sẽ nhận được
- ❖ Ethernet chuyển tới IP, IP chuyển tới UDP, UDP chuyển tới DHCP

# Ví dụ DHCP



- ❑ DHCP server tạo DHCP ACK trong đó chứa địa chỉ IP của client, địa chỉ IP của first-hop router, tên và địa chỉ IP của DNS server
- ❖ đóng gói, rồi frame được chuyển tới client, rồi chuyển lên DHCP tại client
- ❖ client biết địa chỉ IP của nó, tên và địa chỉ IP của DNS server, địa chỉ IP của first-hop router của nó

# DHCP: Wireshark output (home LAN)

Message type: **Boot Request (1)**  
Hardware type: Ethernet  
Hardware address length: 6  
Hops: 0  
**Transaction ID: 0x6b3a11b7**  
Seconds elapsed: 0  
Bootp flags: 0x0000 (Unicast)  
Client IP address: 0.0.0.0 (0.0.0.0)  
Your (client) IP address: 0.0.0.0 (0.0.0.0)  
Next server IP address: 0.0.0.0 (0.0.0.0)  
Relay agent IP address: 0.0.0.0 (0.0.0.0)  
**Client MAC address: Wistron\_23:68:8a (00:16:d3:23:68:8a)**  
Server host name not given  
Boot file name not given  
Magic cookie: (OK)  
Option: (t=53,l=1) **DHCP Message Type = DHCP Request**  
Option: (61) Client identifier  
    Length: 7; Value: 010016D323688A;  
    Hardware type: Ethernet  
    Client MAC address: Wistron\_23:68:8a (00:16:d3:23:68:8a)  
Option: (t=50,l=4) Requested IP Address = 192.168.1.101  
Option: (t=12,l=5) Host Name = "nomad"  
**Option: (55) Parameter Request List**  
    Length: 11; Value: 010F03062C2E2F1F21F92B  
    **1 = Subnet Mask; 15 = Domain Name**  
    **3 = Router; 6 = Domain Name Server**  
    44 = NetBIOS over TCP/IP Name Server  
    .....

request

Message type: **Boot Reply (2)**  
Hardware type: Ethernet  
Hardware address length: 6  
Hops: 0  
**Transaction ID: 0x6b3a11b7**  
Seconds elapsed: 0  
Bootp flags: 0x0000 (Unicast)  
**Client IP address: 192.168.1.101 (192.168.1.101)**  
Your (client) IP address: 0.0.0.0 (0.0.0.0)  
**Next server IP address: 192.168.1.1 (192.168.1.1)**  
Relay agent IP address: 0.0.0.0 (0.0.0.0)  
Client MAC address: Wistron\_23:68:8a (00:16:d3:23:68:8a)  
Server host name not given  
Boot file name not given  
Magic cookie: (OK)  
**Option: (t=53,l=1) DHCP Message Type = DHCP ACK**  
**Option: (t=54,l=4) Server Identifier = 192.168.1.1**  
**Option: (t=1,l=4) Subnet Mask = 255.255.255.0**  
**Option: (t=3,l=4) Router = 192.168.1.1**  
**Option: (6) Domain Name Server**  
    Length: 12; Value: 445747E2445749F244574092;  
    IP Address: 68.87.71.226;  
    IP Address: 68.87.73.242;  
    IP Address: 68.87.64.146  
**Option: (t=15,l=20) Domain Name = "hsd1.ma.comcast.net."**

reply



# Cấp phát địa chỉ IP

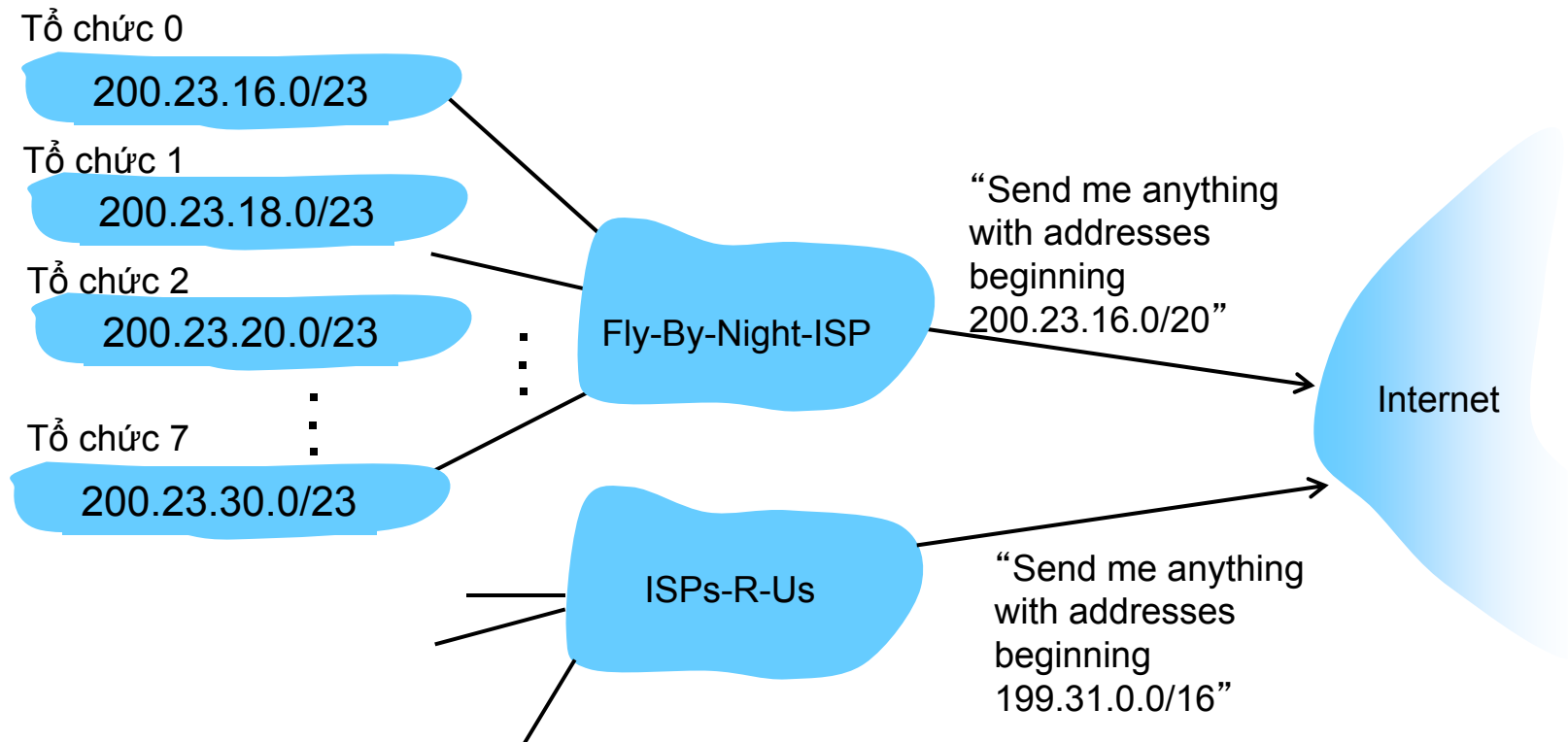
*network* lấy phần subnet của địa chỉ IP như thế nào?

Lấy phần đã cấp của không gian địa chỉ của nhà cung cấp ISP

Khối địa chỉ của ISP	<u>11001000</u>	<u>00010111</u>	<u>00010000</u>	00000000	200.23.16.0/20
Tổ chức 0	<u>11001000</u>	<u>00010111</u>	<u>00010000</u>	00000000	200.23.16.0/23
Tổ chức 1	<u>11001000</u>	<u>00010111</u>	<u>00010010</u>	00000000	200.23.18.0/23
Tổ chức 2	<u>11001000</u>	<u>00010111</u>	<u>00010100</u>	00000000	200.23.20.0/23
...	.....			....	....
Tổ chức 7	<u>11001000</u>	<u>00010111</u>	<u>00011110</u>	00000000	200.23.30.0/23

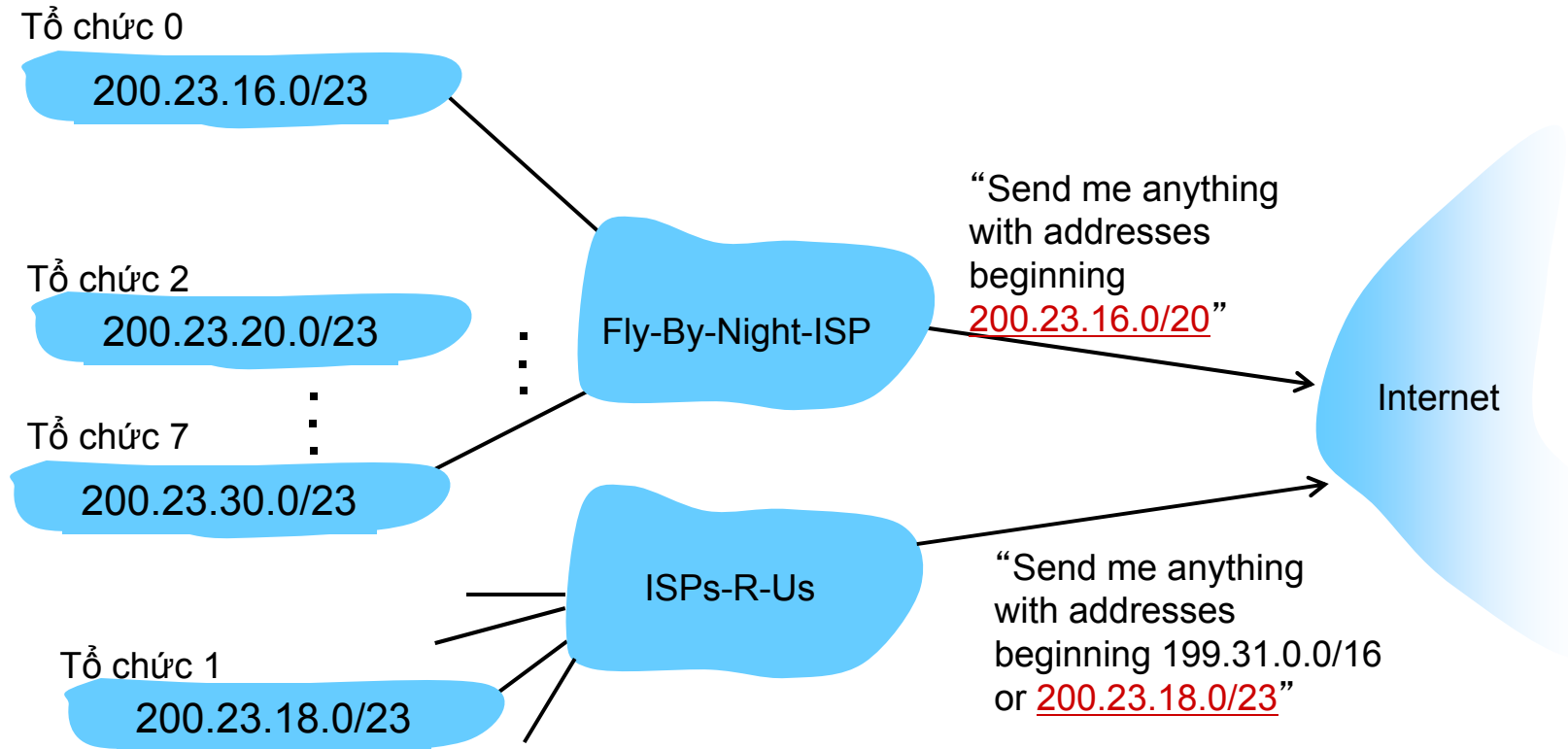
# Địa chỉ phân cấp: Kết tập đường đi

Địa chỉ phân cấp (Hierarchical addressing) cho phép thông báo hiệu quả thông tin dẫn đường



# Địa chỉ phân cấp: Đường đi cụ thể hơn

ISPs-R-Us có một đường đi cụ thể tới Tổ chức 1

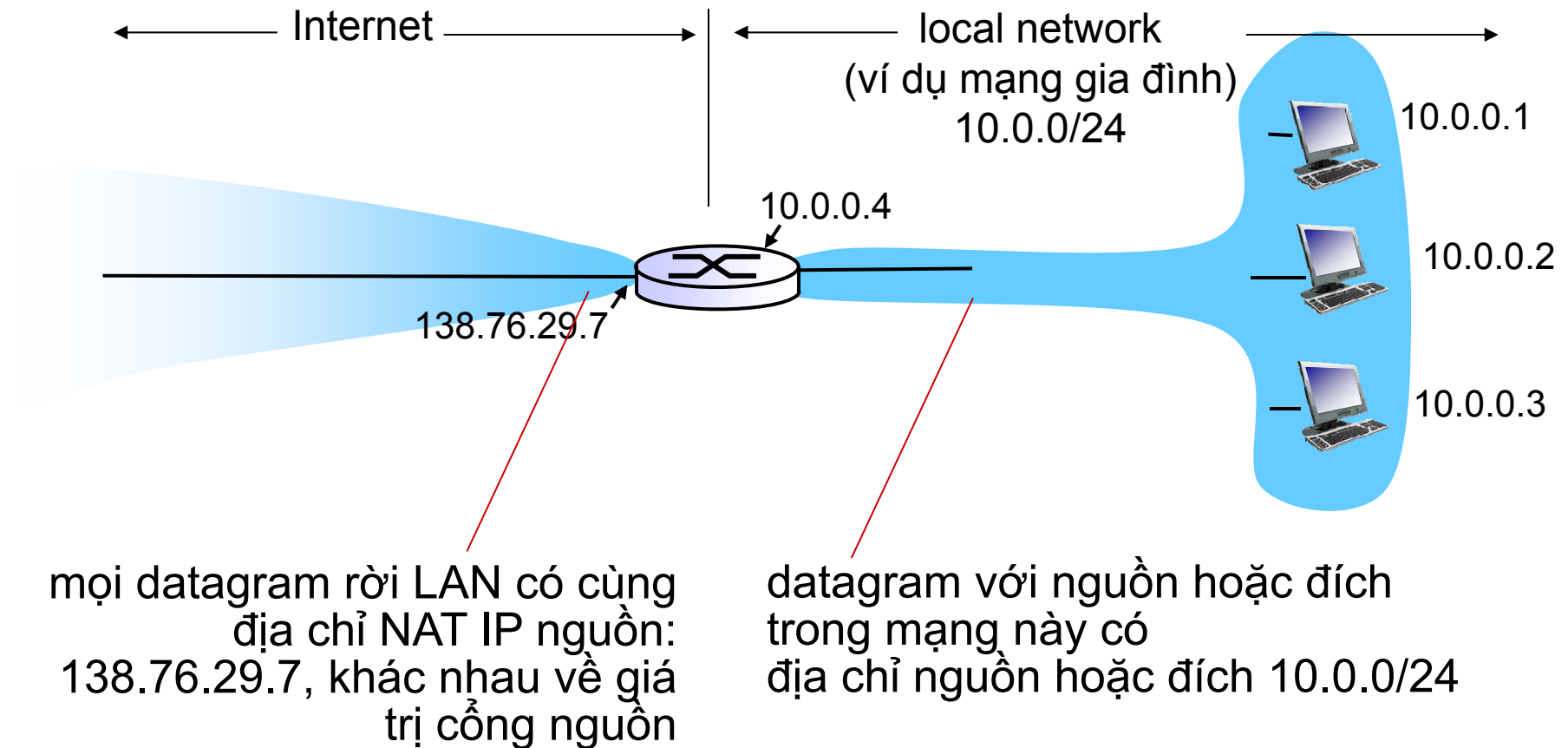


# Khối địa chỉ IP cho ISP

ICANN: Internet Corporation for Assigned Names and Numbers <http://www.icann.org/>

- cấp phát địa chỉ
- quản lý DNS
- gán tên miền, giải quyết tranh chấp

# NAT: network address translation



# NAT: network address translation

LAN sử dụng 1 địa chỉ IP để giao tiếp với bên ngoài:

- không cần một dải địa chỉ từ ISP: chỉ 1 địa chỉ IP cho mọi thiết bị
- có thể thay đổi địa chỉ của thiết bị trong LAN không cần thông báo cho bên ngoài
- có thể thay đổi ISP mà không cần thay đổi địa chỉ của các thiết bị trong LAN
- đối với bên ngoài, các thiết bị trong LAN không được gán địa chỉ riêng mạch và hiện hữu

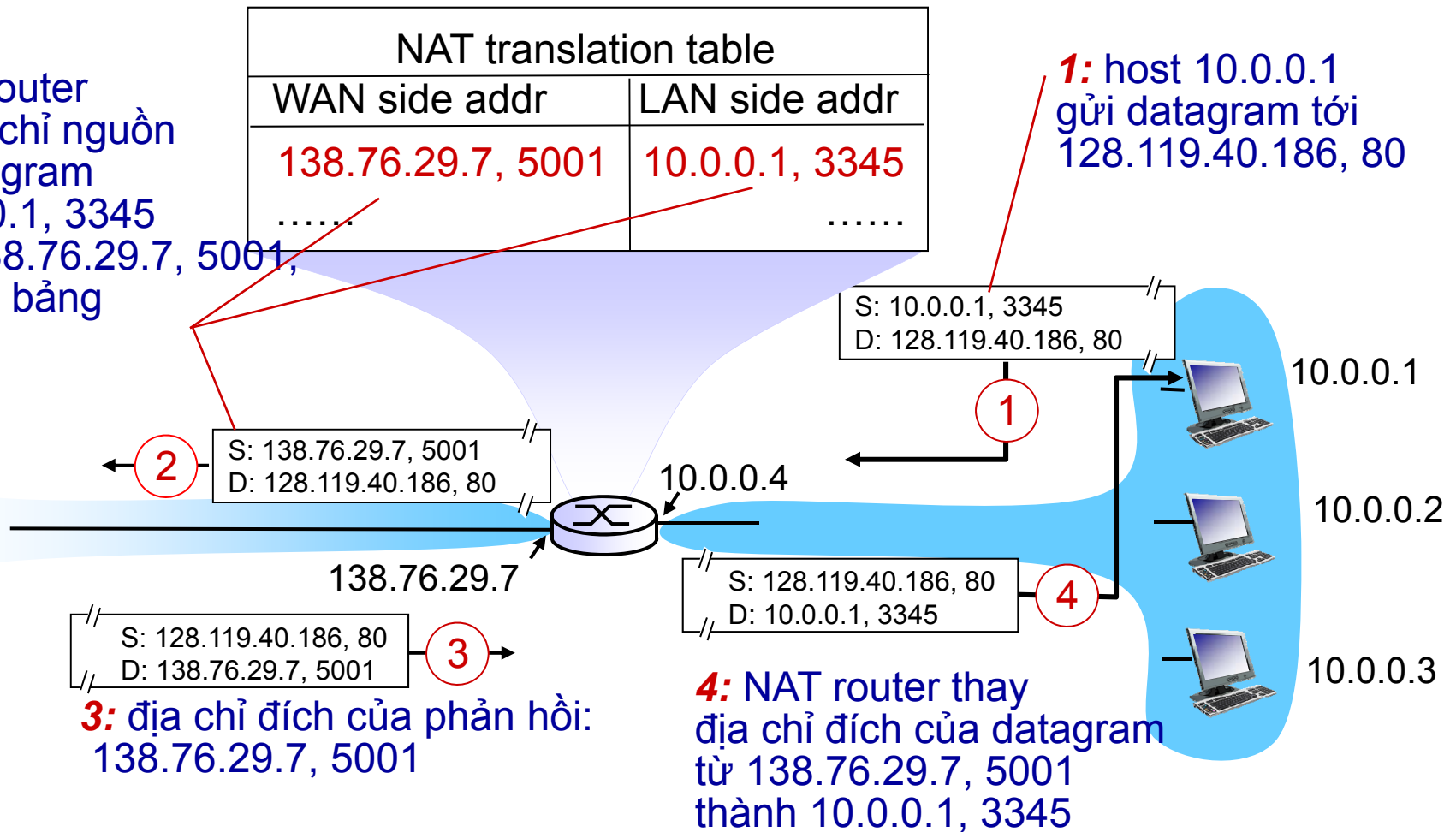
# NAT: network address translation

NAT router phải:

- *datagram đi ra: thay thế* (địa chỉ IP nguồn, số cổng) của mọi datagram đi ra thành (địa chỉ NAT IP, số hiệu cổng mới)
  - . . . clients/server từ xa sẽ phản hồi dùng địa chỉ đích là (địa chỉ NAT IP, số hiệu cổng mới)
- *ghi lại (trong NAT translation table)* cặp ánh xạ (địa chỉ IP nguồn, số hiệu cổng) và (địa chỉ NAT IP, số hiệu cổng mới)
- *datagram đi vào: thay thế* (địa chỉ NAT IP, số hiệu cổng mới) trong trường địa chỉ đích của mọi datagram đi vào thành (địa chỉ IP nguồn, số hiệu cổng) đã lưu trong bảng NAT

# NAT: network address translation

**2:** NAT router thay địa chỉ nguồn của datagram từ 10.0.0.1, 3345 thành 138.76.29.7, 5001, cập nhật bảng





# NAT: network address translation

- ❑ Trường port-number có 16 bit
  - 60,000 kết nối đồng thời trong một địa chỉ LAN
- ❑ Thảo luận:
  - router chỉ nên xử lý tới tầng 3
  - vi phạm lập luận end-to-end
    - NAT có thể xử lý bởi người thiết kế ứng dụng, ví dụ ứng dụng P2P
  - vấn đề thiết địa chỉ nên giải quyết bằng IPv6

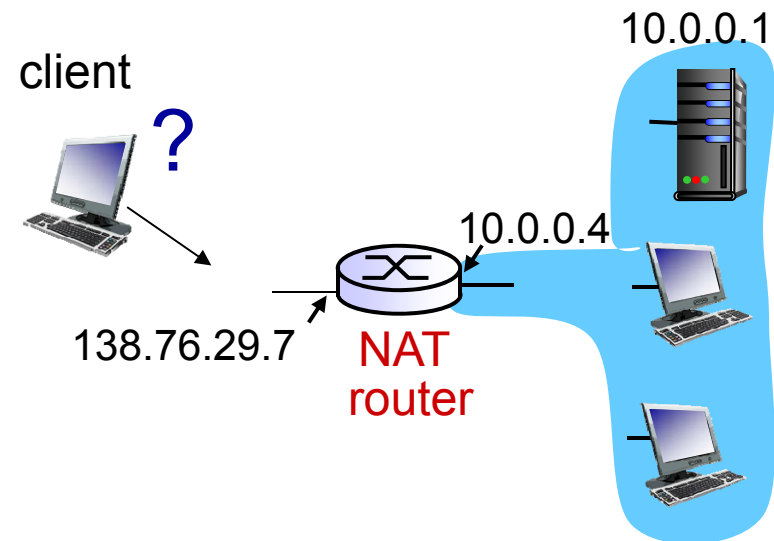
# Vấn đề vượt NAT (NAT traversal problem)

- ❑ Client muốn kết nối tới server với địa chỉ 10.0.0.1

- địa chỉ của server 10.0.0.1 là địa chỉ cục bộ trong LAN (client không thể dùng làm địa chỉ đích)
- chỉ có một địa chỉ NAT là hiện hữu đối với bên ngoài: 138.76.29.7

- ❑ **Giải pháp 1:** cấu hình NAT tĩnh để chuyển yêu cầu kết nối tới tại 1 cổng nào đó tới server

- ví dụ (123.76.29.7, port 2500) luôn được chuyển tới (10.0.0.1, port 25000)

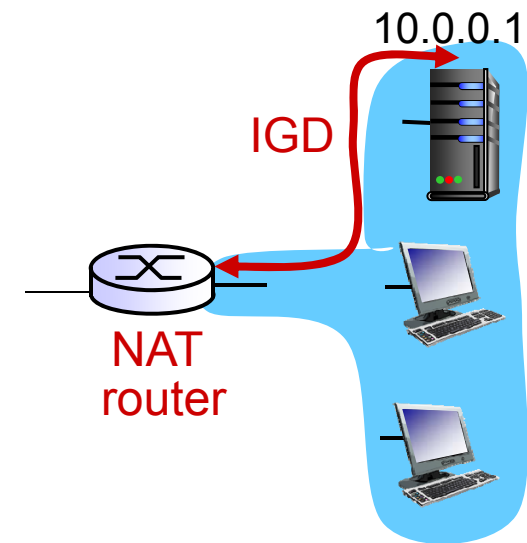


# Vấn đề vượt NAT

- ❑ *Giải pháp 2:* Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Cho phép host trong NAT:

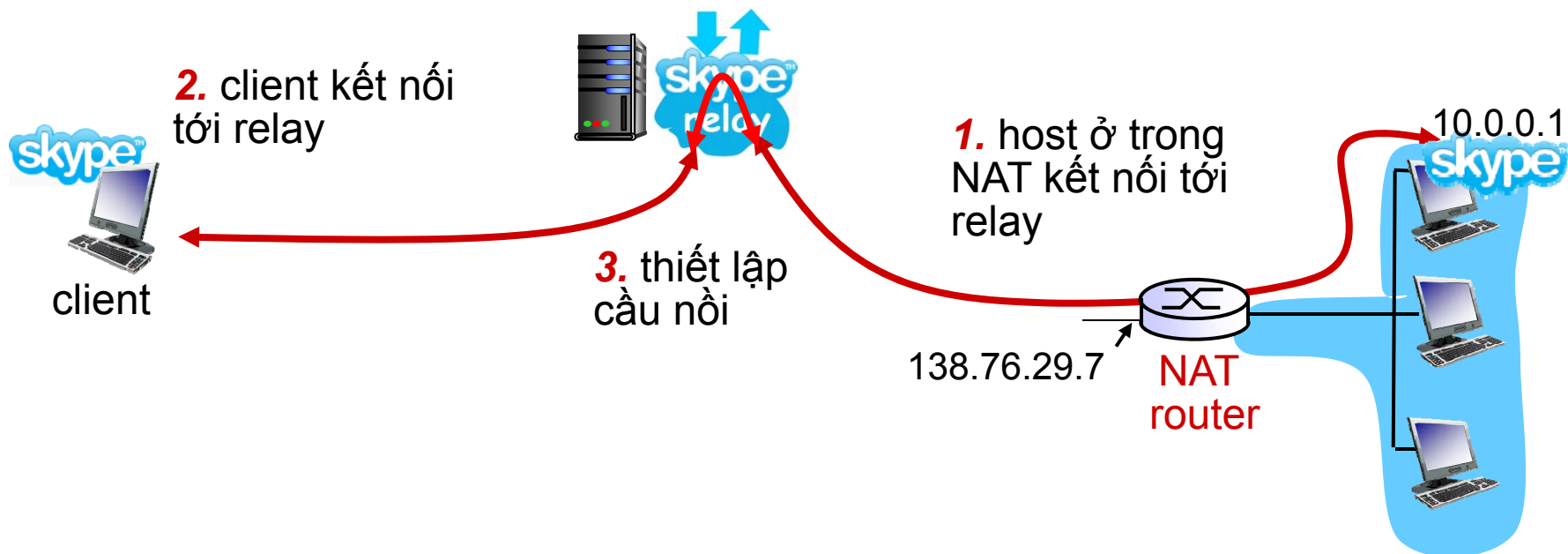
- ❖ học địa chỉ public IP (138.76.29.7)
- ❖ thêm và bỏ ánh xạ cổng (với thời gian ánh xạ)

tự động việc cấu hình ánh xạ cổng NAT



# Vấn đề vượt NAT

- ❑ **Giải pháp 3:** relaying (dùng trong Skype)
  - Client trong NAT thiết lập kết nối tới relay
  - client bên ngoài kết nối tới relay
  - relay làm cầu nối gói tin giữa các kết nối



# Chương 4: Tầng mạng

## 4.1 Giới thiệu

## 4.2 IP: Internet Protocol

- Cấu trúc datagram
- IPv4
- ICMP
- IPv6

## 4.3 Giải thuật dẫn đường

- link state
- distance vector
- hierarchical routing

## 4.4 Dẫn đường trong Internet

- RIP
- OSPF
- BGP

## 4.5 Broadcast routing và multicast routing

# ICMP: internet control message protocol

- ❑ Dùng bởi các host và router để giao tiếp thông tin ở mức mạng
  - thông báo lỗi: unreachable host, network, port, protocol
  - echo request/reply (dùng bởi ping)
- ❑ Gói tin ICMP chứa trong các IP datagram (network-layer “above” IP)
- ❑ **ICMP message:** type, code và 8 byte đầu tiên của IP datagram gây ra lỗi

<u>Type</u>	<u>Code</u>	<u>Mô tả</u>
0	0	echo reply (ping)
3	0	không tới được mạng đích
3	1	không tới được host đích
3	2	không tới được giao thức đích
3	3	không tới được cổng đích
3	6	không biết mạng đích
3	7	không biết host đích
4	0	source quench (điều khiển tắc nghẽn, không dùng)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL quá hạn
12	0	IP header không hợp lệ

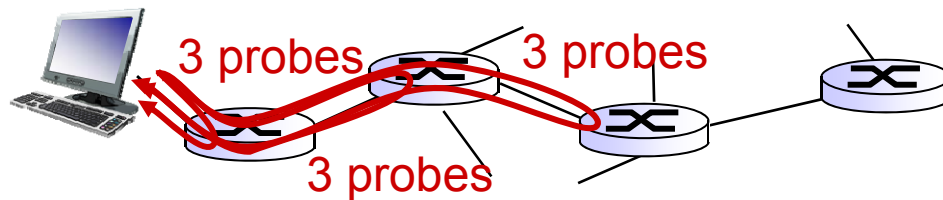
# Traceroute và ICMP

- ❑ Nút nguồn gửi tập các UDP segment tới nút đích
  - đầu tiên gán TTL = 1
  - tiếp theo gán TTL=2, ...
  - với giá trị cổng không có thực
- ❑ khi tập thứ n của datagram tới router thứ n:
  - router bỏ datagram
  - và gửi về nút nguồn ICMP message (type 11, code 0)
  - ICMP message chứa tên của router và địa chỉ IP

- ❑ khi ICMP message tới, nguồn ghi lại RTT

## *Điều kiện dừng:*

- ❖ cuối cùng UDP segment tới nút đích
- ❖ nút đích gửi lại ICMP “port unreachable” message (type 3, code 3)
- ❖ nút nguồn dừng



# IPv6

- ❑ *Lý do ban đầu:* Không gian địa chỉ 32-bit sẽ nhanh chóng dùng hết
- ❑ Lý do khác:
  - cấu trúc của header format giúp tăng tốc độ xử lý/chuyển tiếp
  - thay đổi header để hỗ trợ QoS

## *Cấu trúc của IPv6 datagram:*

- header có chiều dài cố định 40 byte
- không cho phép phân mảnh



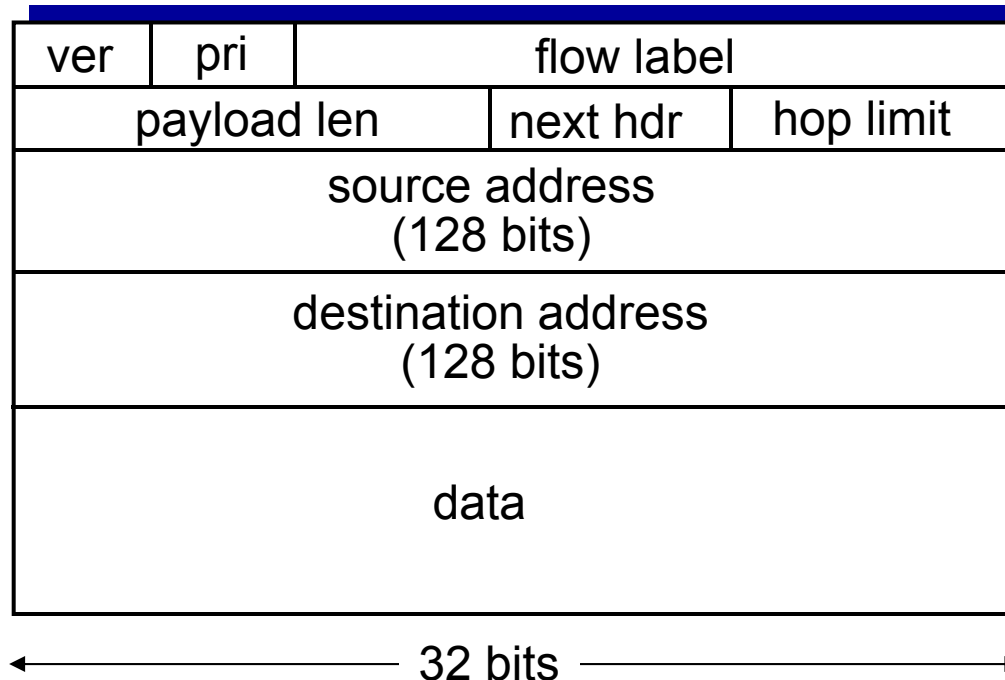
# Cấu trúc IPv6 datagram

*priority*: xác định ưu tiên giữa các datagram trong flow

*flow Label*: xác định các datagram trong cùng flow

(khái niệm “flow” không định nghĩa rõ)

*next header*: xác định giao thức của tầng cao hơn cho data

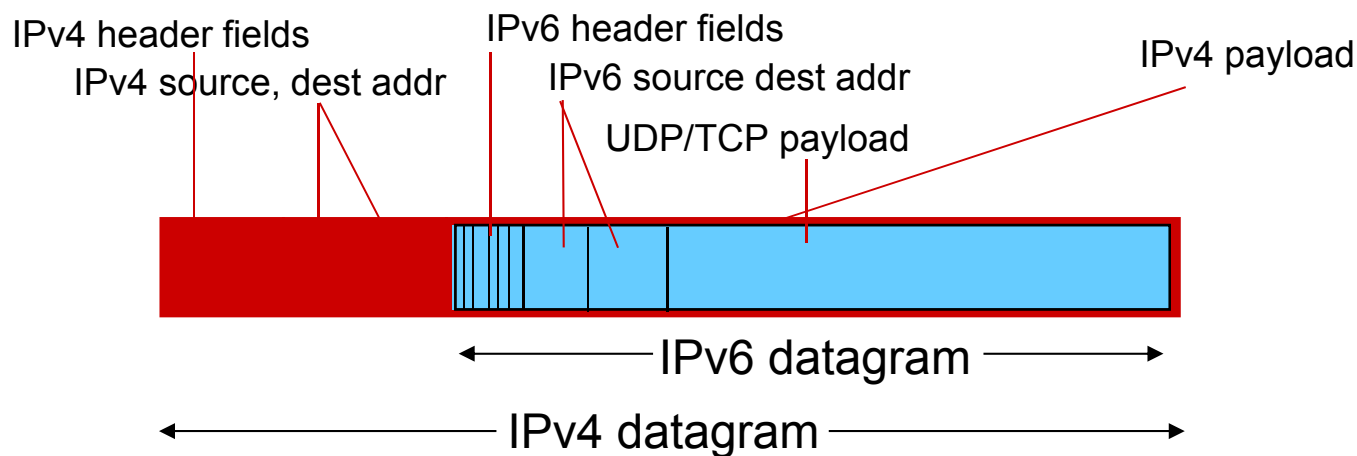


# Các thay đổi khác so với IPv4

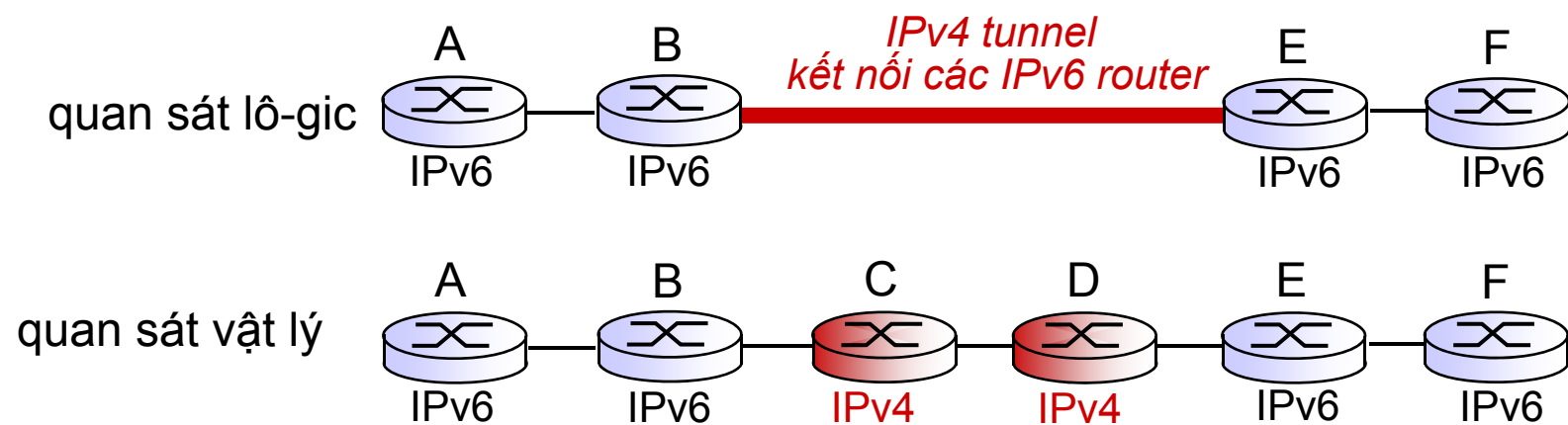
- ❑ *checksum*: loại bỏ để giảm thời gian xử lý tại mỗi hop
- ❑ *options*: cho phép, nhưng ở ngoài phần header, xác định bởi trường “Next Header”
- ❑ *ICMPv6*: phiên bản mới của ICMP
  - thêm kiểu bản tin, ví dụ “Packet Too Big”
  - chức năng quản lý nhóm multicast

# Chuyển từ IPv4 sang IPv6

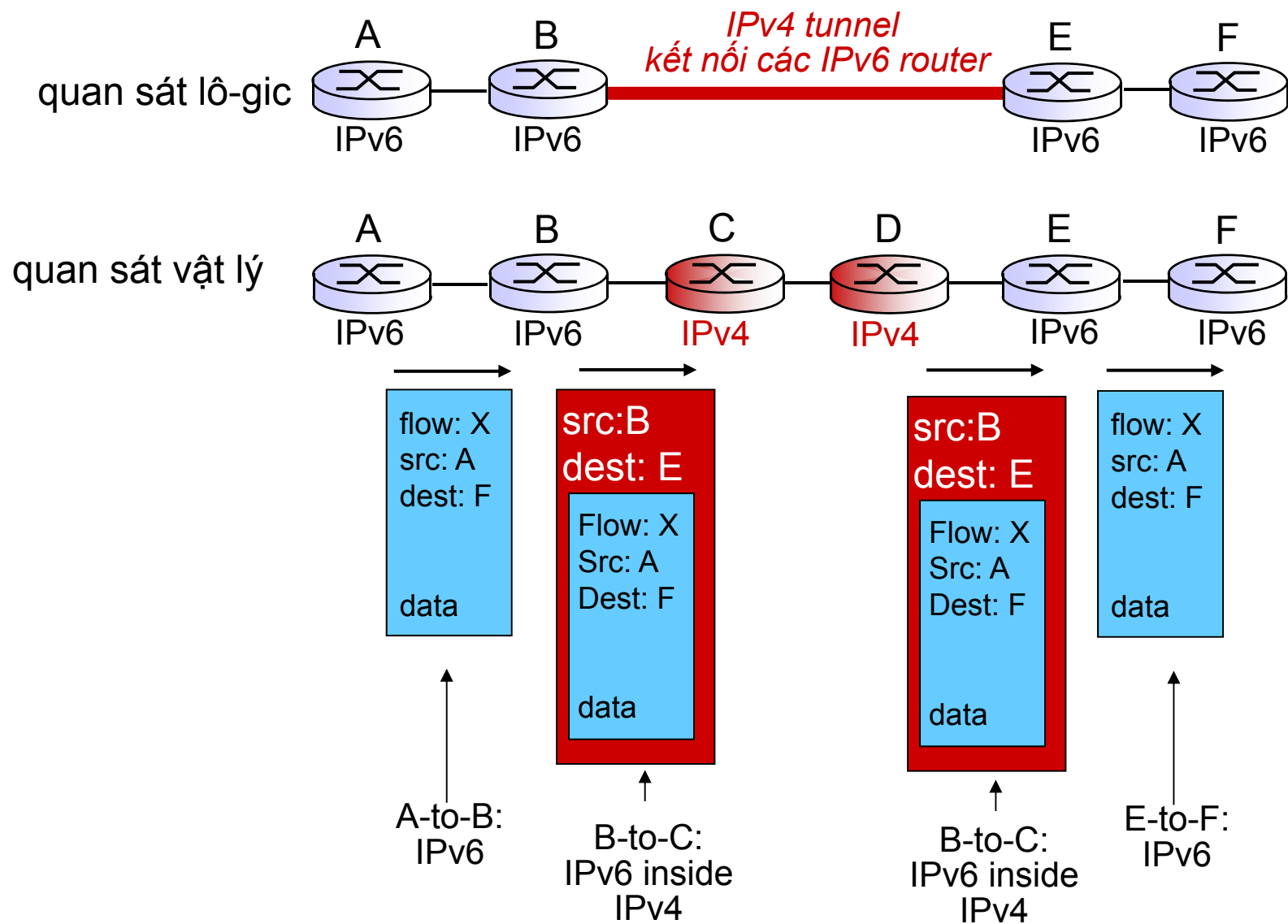
- ❑ tất cả các router không thể nâng cấp đồng thời
  - làm thế nào để mạng có cả router IPv4 và router IPv6 hoạt động?
- ❑ *tunneling*: Giữa các IPv4 router, IPv6 datagram được chứa trong payload trong IPv4 datagram



# Tunneling



# Tunneling



# Sự chấp nhận IPv6

## ❑ Chưa triển khai rộng rãi

- 20 năm và tiếp tục cần thêm thời gian
- so sánh với sự thay đổi của ứng dụng trong 20 năm:  
WWW, Facebook, ...
- Lý do?

# Chương 4: Tầng mạng

## 4.1 Giới thiệu

## 4.2 IP: Internet Protocol

- Cấu trúc datagram
- IPv4
- ICMP
- IPv6

## 4.3 Giải thuật dẫn đường

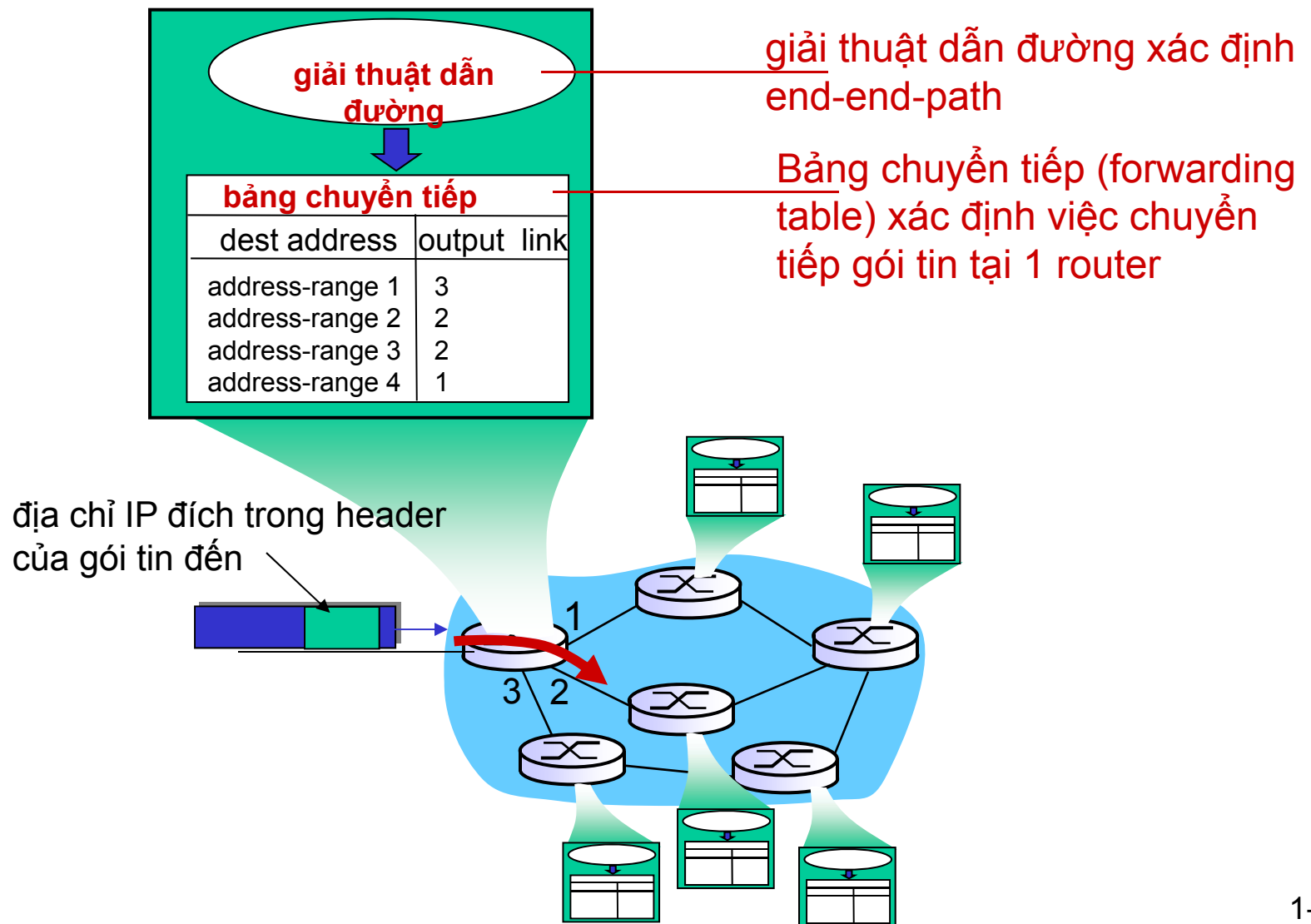
- link state
- distance vector
- hierarchical routing

## 4.4 Dẫn đường trong Internet

- RIP
- OSPF
- BGP

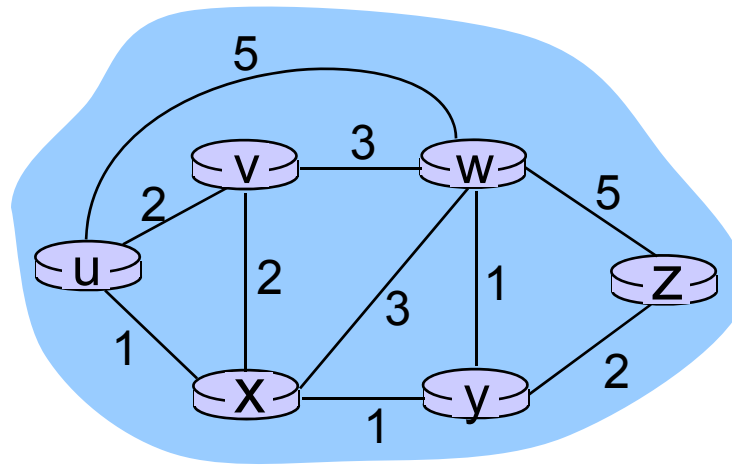
## 4.5 Broadcast routing và multicast routing

# Dẫn đường và chuyển tiếp





# Mô tả dạng đồ thị

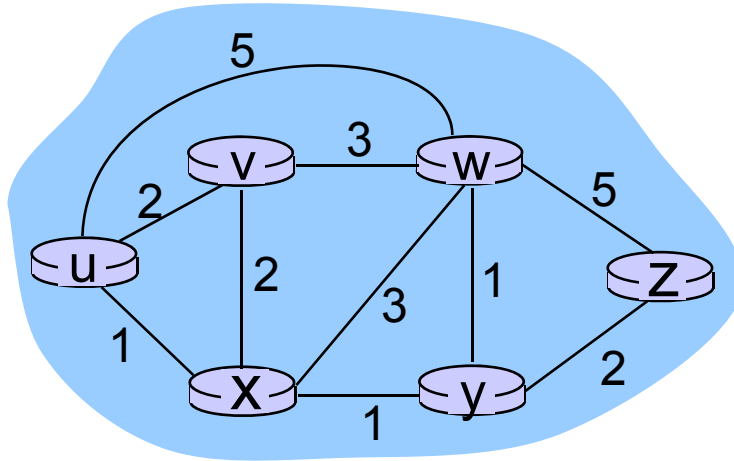


Đồ thị:  $G = (N, E)$

$N$  = tập các router =  $\{ u, v, w, x, y, z \}$

$E$  = tập các liên kết =  $\{ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) \}$

# Chi phí của liên kết



$c(x, x') =$  chi phí của liên kết  $(x, x')$   
ví dụ  $c(w, z) = 5$

Chi phí có thể cố định bằng 1,  
hoặc tỉ lệ nghịch với băng thông,  
hoặc tỉ lệ nghịch với tắc nghẽn

Chi phí của đường đi  $(x_1, x_2, x_3, \dots, x_p) = c(x_1, x_2) + c(x_2, x_3) + \dots + c(x_{p-1}, x_p)$

**Câu hỏi chính:** Tìm đường đi có chi phí nhỏ nhất giữa u và z?

**Giải thuật dẫn đường:** Giải thuật tìm đường có chi phí nhỏ nhất

# Phân loại giải thuật dẫn đường

*Thông tin tập trung hoặc phân tán?*

*Tập trung:*

- ❑ Mọi router có mô hình đầy đủ, thông tin chi phí các liên kết
- ❑ Giải thuật “link state”

*Phân tán*

- ❑ Router biết các láng giềng có liên kết vật lý tới nó và chi phí liên kết
- ❑ Quá trình lặp của việc tính toán và trao đổi thông tin với láng giềng
- ❑ Giải thuật “distance vector”

*Tĩnh hoặc động?*

*Tĩnh:*

- ❑ Route thay đổi chậm qua thời gian

*Động:*

- ❑ Route thay đổi nhanh hơn
  - Cập nhật định kì
  - Phản hồi đối với sự thay đổi chi phí liên kết

# Chương 4: Tầng mạng

## 4.1 Giới thiệu

## 4.2 IP: Internet Protocol

- Cấu trúc datagram
- IPv4
- ICMP
- IPv6

## 4.3 Giải thuật dẫn đường

- link state
- distance vector
- hierarchical routing

## 4.4 Dẫn đường trong Internet

- RIP
- OSPF
- BGP

## 4.5 Broadcast routing và multicast routing

# Giải thuật dẫn đường kiểu Link-State

## Giải thuật Dijkstra

- ❑ mọi nút mạng có đầy đủ thông tin về mô hình mạng, chi phí liên kết
  - thực hiện qua “link state broadcast”
  - mọi nút có cùng thông tin
- ❑ tính toán đường đi có chi phí thấp nhất từ một nút tới các nút khác
  - gửi *forwarding table* cho nút đó
- ❑ lặp: sau k bước lặp, tính được đường đi có chi phí thấp nhất tới k đích

## Ký hiệu:

- ❑  $c(x,y)$ : chi phí liên kết từ nút x tới nút y;  $= \infty$  nếu không là nút kề
- ❑  $D(v)$ : giá trị hiện tại của chi phí đường đi từ nguồn tới nút v
- ❑  $p(v)$ : nút trước v trên đường đi từ nguồn tới v
- ❑  $N'$ : tập các nút mà đường đi với chi phí nhỏ nhất tới nút là đã xác định

# Giải thuật Dijkstra

1 **Khởi tạo:**

2  $N' = \{u\}$

3 for all nodes  $v$

4 if  $v$  kề  $u$

5 then  $D(v) = c(u,v)$

6 else  $D(v) = \infty$

7

8 **Loop**

9 tìm  $w$  không trong  $N'$  mà  $D(w)$  nhỏ nhất

10 thêm  $w$  vào  $N'$

11 cập nhật  $D(v)$  cho mọi  $v$  kề với  $w$  và không trong  $N'$  :

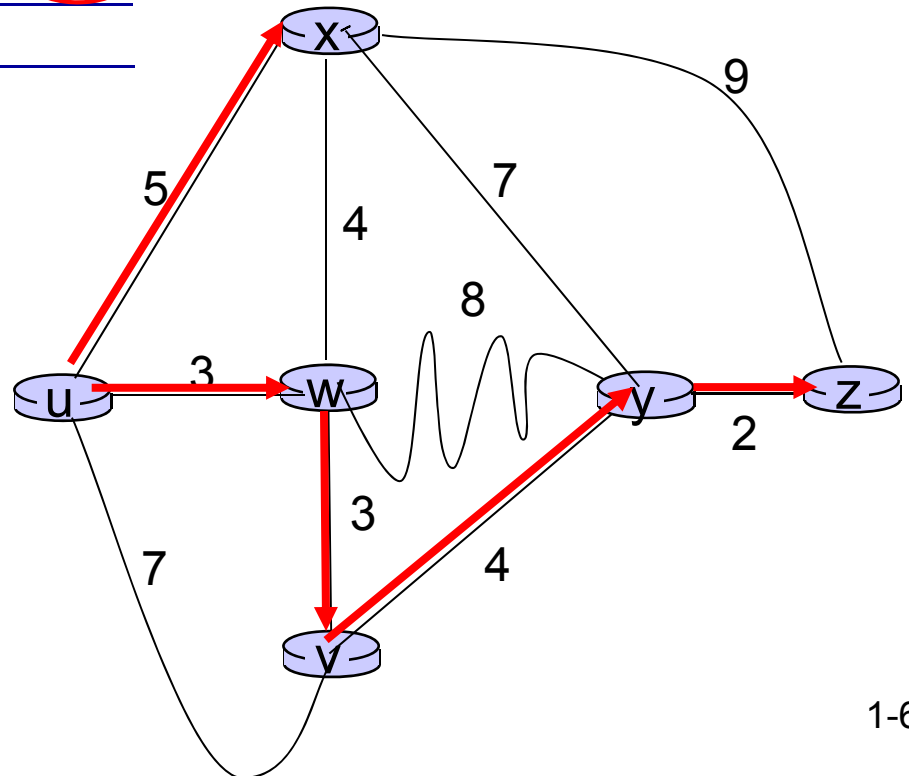
12  **$D(v) = \min( D(v), D(w) + c(w,v) )$**

13 /\* chi phí mới tới  $v$  sẽ hoặc là chi phí cũ tới  $v$  hoặc là chi phí  
đường đi ngắn nhất tới  $w$  cộng với chi phí từ  $w$  tới  $v$  \*/

15 **until mọi nút nằm trong  $N'$**

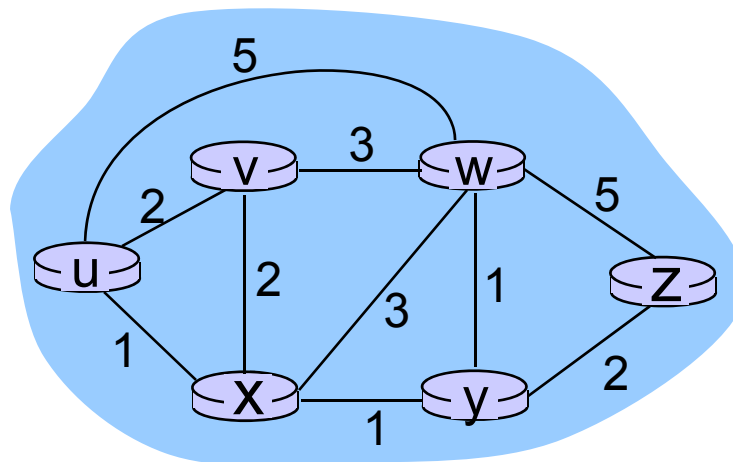
# Giải thuật Dijkstra: Ví dụ 1

Bước	N'	D( <b>v</b> ) p(v)	D( <b>w</b> ) p(w)	D( <b>x</b> ) p(x)	D( <b>y</b> ) p(y)	D( <b>z</b> ) p(z)
0	u	7,u	<b>3,u</b>	5,u	$\infty$	$\infty$
1	uw	6,w		<b>5,u</b>	11,w	$\infty$
2	uwx	<b>6,w</b>			11,w	14,x
3	uwxv				<b>10,v</b>	14,x
4	uwxvy					<b>12,y</b>
5	uwxvyz					



# Giải thuật Dijkstra: Ví dụ 2

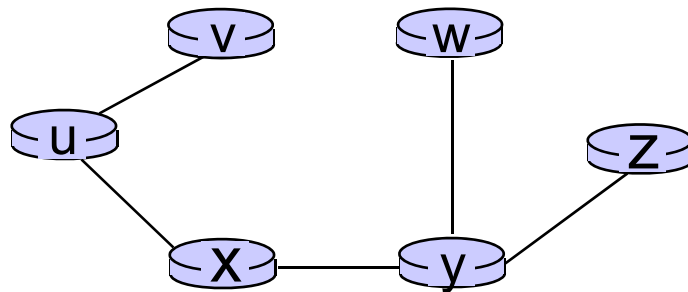
Bước	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	$\infty$	$\infty$
1	ux	2,u	4,x		2,x	$\infty$
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					





# Giải thuật Dijkstra: Ví dụ 2

Kết quả cây đường đi ngắn nhất từ u:



Kết quả forwarding table tại u:

đích	liên kết
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

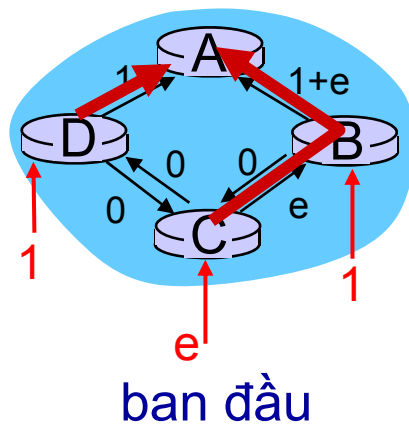
# Thảo luận về giải thuật Dijkstra

*Độ phức tạp giải thuật:*  $n$  nút

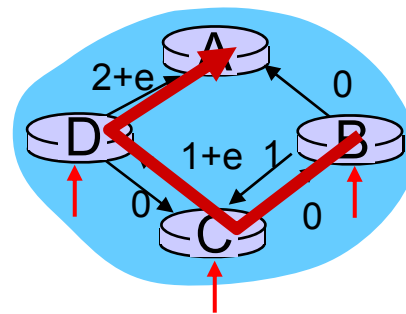
- ❑ mỗi vòng lặp: cần kiểm tra mọi nút, w, không trong N
- ❑  $n(n+1)/2$  so sánh:  $O(n^2)$
- ❑ cài đặt hiệu quả có thể đạt được:  $O(n \log n)$

**Sự dao động:**

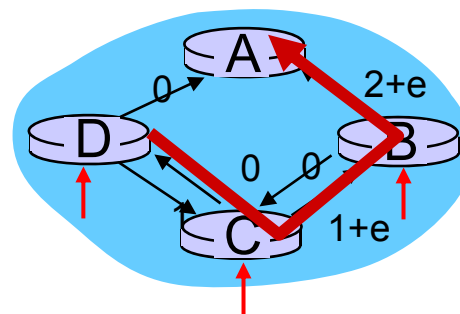
- ❑ ví dụ hỗ trợ chi phí liên kết bằng lưu lượng trên liên kết



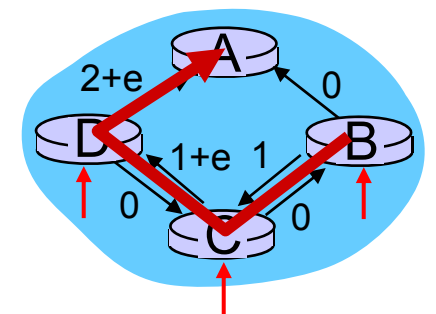
với chi phí đã có,  
tìm đường mới....  
kết quả chi phí mới



với chi phí đã có,  
tìm đường mới....  
kết quả chi phí mới



với chi phí đã có,  
tìm đường mới....  
kết quả chi phí mới



# Chương 4: Tầng mạng

## 4.1 Giới thiệu

## 4.2 IP: Internet Protocol

- Cấu trúc datagram
- IPv4
- ICMP
- IPv6

## 4.3 Giải thuật dẫn đường

- link state
- distance vector
- hierarchical routing

## 4.4 Dẫn đường trong Internet

- RIP
- OSPF
- BGP

## 4.5 Broadcast routing và multicast routing

# Giải thuật Distance vector

*Công thức Bellman-Ford (quy hoạch động)*

let

$d_x(y) :=$  chi phí của đường đi có chi phí nhỏ nhất từ  $x$  tới  $y$

then

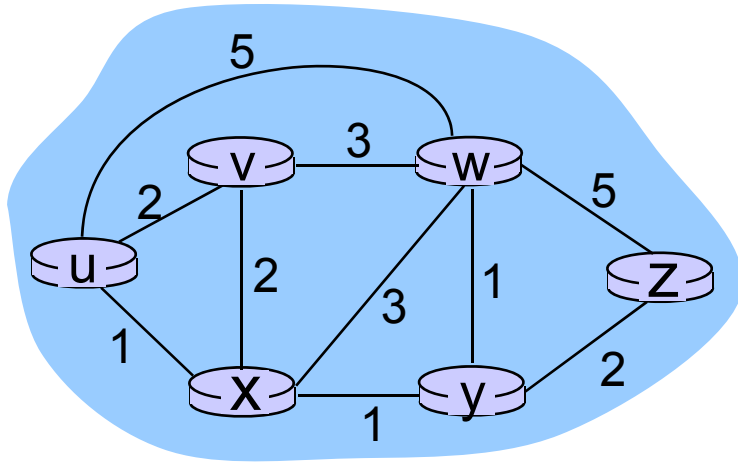
$$d_x(y) = \min_v \{ c(x,v) + d_v(y) \}$$

chi phí từ nút  $x$  tới nút kề  $v$

chi phí từ nút kề  $v$  tới đích  $y$

tính min với mọi nút kề  $v$  của nút  $x$

# Ví dụ Bellman-Ford



Khởi tạo,  $d_v(z) = 5$ ,  $d_x(z) = 3$ ,  $d_w(z) = 3$

Công thức B-F:

$$\begin{aligned} d_u(z) &= \min \{ c(u,v) + d_v(z), \\ &\quad c(u,x) + d_x(z), \\ &\quad c(u,w) + d_w(z) \} \\ &= \min \{ 2 + 5, \\ &\quad 1 + 3, \\ &\quad 5 + 3 \} = 4 \end{aligned}$$

Nút đạt được giá trị nhỏ nhất là next hop trong đường đi ngắn nhất, dùng trong forwarding table

# Giải thuật Distance vector

- $D_x(y)$  = đánh giá chi phí thấp nhất từ  $x$  tới  $y$ 
  - $x$  duy trì véc tơ khoảng cách  $\mathbf{D}_x = [D_x(y): y \in N]$
- nút  $x$ :
  - biết chi phí tới nút kề  $v$ :  $c(x,v)$
  - duy trì véc tơ khoảng cách của các nút kề.  
Với mỗi nút kề  $v$ ,  $x$  duy trì  
 $\mathbf{D}_v = [D_v(y): y \in N]$

# Giải thuật Distance vector

## *Ý tưởng:*

- định kì, mỗi nút gửi véc tơ khoảng cách của nó tới các nút kề
- khi x nhận đánh giá véc tơ khoảng cách mới từ nút kề, nó cập nhật véc tơ khoảng cách của nó (sử dụng công thức B-F) :

$$D_x(y) \leftarrow \min_v \{c(x,v) + D_v(y)\} \text{ với } y \in N$$

- ❖ bởi điều kiện thông thường, đánh giá  $D_x(y)$  hội tụ tới chi phí thấp nhất thực tế  $d_x(y)$

# Giải thuật Distance vector

*lặp, không đồng bộ:* mỗi vòng lặp cục bộ thực hiện khi:

- ❑ thay đổi chi phí liên kết cục bộ
- ❑ DV cập nhật từ nút kề

*phân tán:*

- ❑ mỗi nút thông báo cho các nút kề chỉ khi véc tơ khoảng cách thay đổi
  - các nút kề sau đó sẽ thông báo cho các nút kề của nó nếu cần

*mỗi nút:*





$$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$$

$$= \min\{2+0, 7+1\} = 2$$

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$$

$$= \min\{2+1, 7+0\} = 3$$

**bảng nút x**

	chi phí tới		
	x	y	z
x	0	2	7
từ y	$\infty$	$\infty$	$\infty$
z	$\infty$	$\infty$	$\infty$

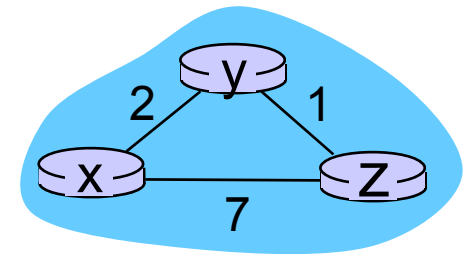
	chi phí tới		
	x	y	z
x	0	2	3
từ y	2	0	1
z	7	1	0

**bảng nút y**

	chi phí tới		
	x	y	z
x	$\infty$	$\infty$	$\infty$
từ y	2	0	1
z	$\infty$	$\infty$	$\infty$

**bảng nút z**

	chi phí tới		
	x	y	z
x	$\infty$	$\infty$	$\infty$
từ y	$\infty$	$\infty$	$\infty$
z	7	1	0



thời gian

$$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\} \\ = \min\{2+0, 7+1\} = 2$$

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\} \\ = \min\{2+1, 7+0\} = 3$$

**bảng nút x**

	chi phí tới		
	x	y	z
x	0	2	7
từ y	$\infty$	$\infty$	$\infty$
z	$\infty$	$\infty$	$\infty$

**bảng nút y**

	chi phí tới		
	x	y	z
x	$\infty$	$\infty$	$\infty$
từ y	2	0	1
z	$\infty$	$\infty$	$\infty$

**bảng nút z**

	chi phí tới		
	x	y	z
x	$\infty$	$\infty$	$\infty$
từ y	$\infty$	$\infty$	$\infty$
z	7	1	0

	chi phí tới		
	x	y	z
x	0	2	3
từ y	2	0	1
z	7	1	0

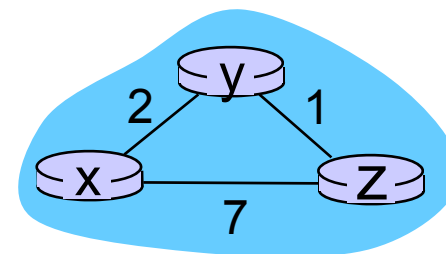
	chi phí tới		
	x	y	z
x	0	2	7
từ y	2	0	1
z	7	1	0

	chi phí tới		
	x	y	z
x	0	2	7
từ y	2	0	1
z	3	1	0

	chi phí tới		
	x	y	z
x	0	2	3
từ y	2	0	1
z	3	1	0

	chi phí tới		
	x	y	z
x	0	2	3
từ y	2	0	1
z	3	1	0

	chi phí tới		
	x	y	z
x	0	2	3
từ y	2	0	1
z	3	1	0

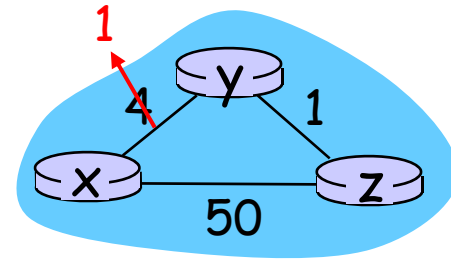


time

# Giải thuật Distance vector: Chi phí liên kết thay đổi

## *Chi phí liên kết thay đổi:*

- ❖ nút phát hiện sự thay đổi chi phí liên kết cục bộ
- ❖ cập nhật thông tin dẫn đường, tính toán lại véc tơ khoảng cách
- ❖ nếu véc tơ khoảng cách thay đổi, thông báo cho các nút kề



“thay đổi  
tốt lan  
truyền  
nhanh”

$t_0$ : y phát hiện sự thay đổi chi phí liên kết, cập nhật véc tơ trạng thái của nó, thông báo cho các nút kề của nó

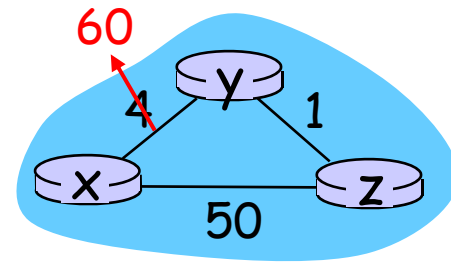
$t_1$ : z nhận cập nhật từ y, cập nhật bảng khoảng cách của nó, tính lại chi phí thấp nhất tới x, gửi tới các nút kề của nó véc tơ khoảng cách

$t_2$ : y nhận cập nhật của z, cập nhật bảng khoảng cách của nó. Chi phí thấp nhất của y không thay đổi, vì vậy y không gửi bản tin tới z

# Giải thuật Distance vector: Chi phí liên kết thay đổi

## *Chi phí liên kết thay đổi:*

- ❖ nút phát hiện sự thay đổi của chi phí liên kết cục bộ
- ❖ *thay đổi xấu lan truyền chậm* - vấn đề vô hạn



# So sánh giải thuật LS và DV

## *Độ phức tạp*

- ❑ **LS:**  $O(nE)$  bản tin được gửi
- ❑ **DV:** trao đổi giữa các nút kề
  - thời gian hội tụ thay đổi

## *Tốc độ hội tụ*

- ❑ **LS:** Giải thuật  $O(n^2)$  cần  $O(nE)$  bản tin
  - có thể có sự dao động
- ❑ **DV:** thời gian hội tụ thay đổi
  - có thể có routing loop
  - vấn đề count-to-infinity

**Độ tin cậy:** Vấn đề gì nếu router có thông tin sai?

### **LS:**

- nút có thể lan truyền chi phí **liên kết** không đúng
- mỗi nút chỉ tính bảng cầu chính nó

### **DV:**

- nút có thể lan truyền chi phí **đường đi** không đúng
- bảng của mỗi nút được sử dụng bởi các nút khác
  - lỗi lan truyền qua mạng

# Chương 4: Tầng mạng

## 4.1 Giới thiệu

## 4.2 IP: Internet Protocol

- Cấu trúc datagram
- IPv4
- ICMP
- IPv6

## 4.3 Giải thuật dẫn đường

- link state
- distance vector
- hierarchical routing

## 4.4 Dẫn đường trong Internet

- RIP
- OSPF
- BGP

## 4.5 Broadcast routing và multicast routing

# Hierarchical routing

Các nội dung dẫn đường đã tìm hiểu đã lý tưởng hóa một số yếu tố

- ❖ mọi router giống nhau
- ❖ network “flat”

... *không* đúng trong thực tế

*quy mô:* với 600 triệu

đích:

- ❑ không thể chứa mọi đích trong bảng dẫn đường
- ❑ trao đổi bảng dẫn đường làm tràn ngập liên kết

*tự quản*

- ❑ internet = mạng của các mạng
- ❑ mỗi quản trị mạng có thể muốn điều khiển việc dẫn đường trong chính mạng của nó

# Hierarchical routing

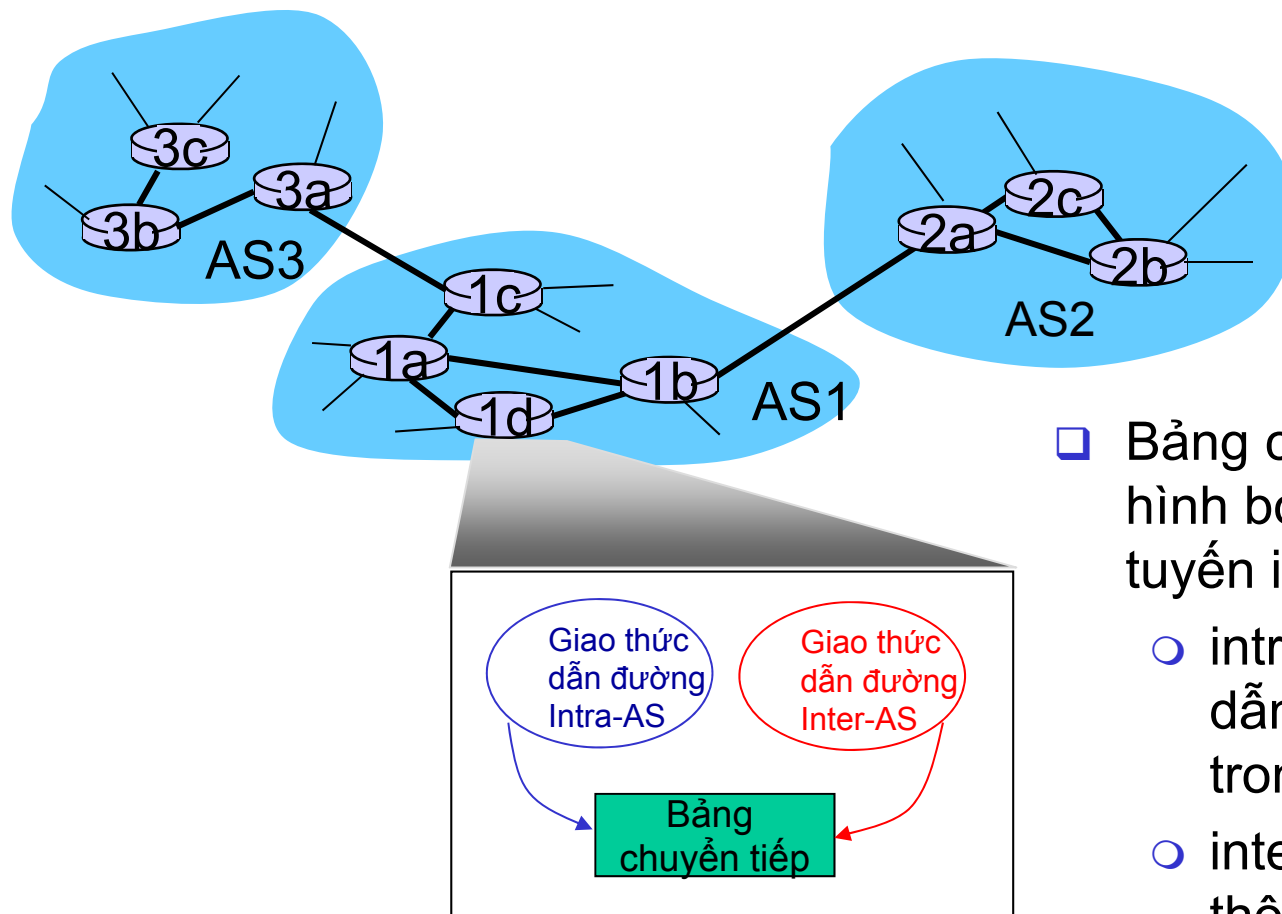
- ❑ Kết tập các router thành các vùng, “**autonomous systems**” (AS)
- ❑ Các router trong cùng AS chạy cùng giao thức định tuyến
  - giao thức dẫn đường “**intra-AS**”
  - các router trong các AS khác nhau có thể chạy các giao thức intra-AS khác nhau

## *gateway router:*

- ❑ tại “biên” của AS của nó
- ❑ có liên kết tới router khác trong AS khác



# Kết nối các AS



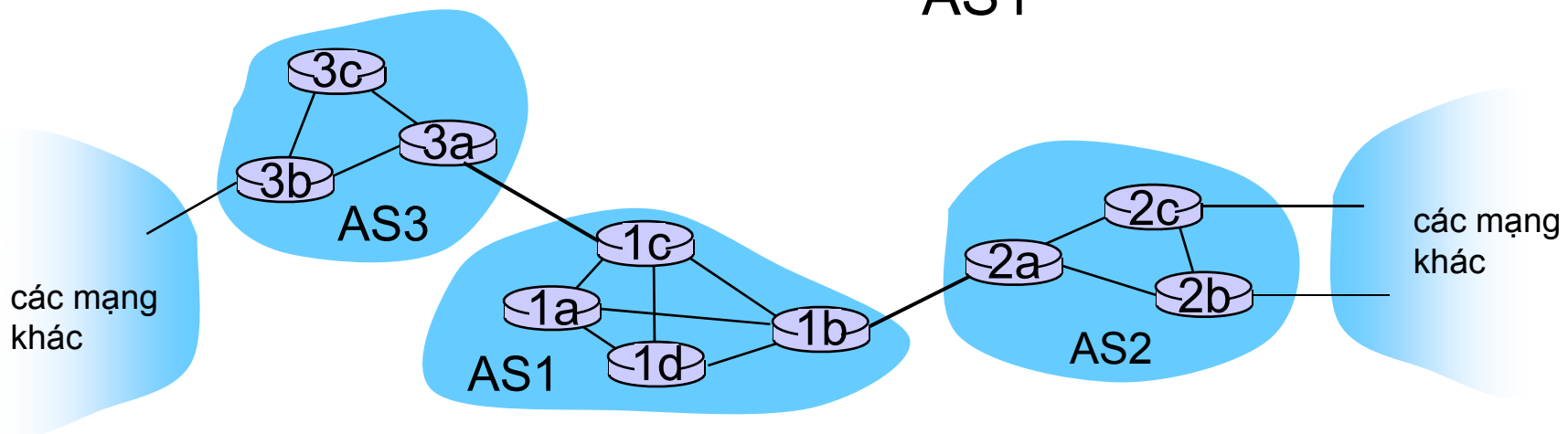
- Bảng chuyển tiếp được cấu hình bởi cả giao thức định tuyến intra-AS và inter-AS
  - intra-AS gán thông tin dẫn đường cho các đích trong AS
  - inter-AS và intra-AS gán thông tin dẫn đường cho các đích bên ngoài AS

# Công việc của Inter-AS

- ❑ Giả sử router trong AS1 nhận datagram có đích ngoài AS1:
  - Router chuyển tiếp gói tin tới gateway router. Chuyển tới gateway router nào?

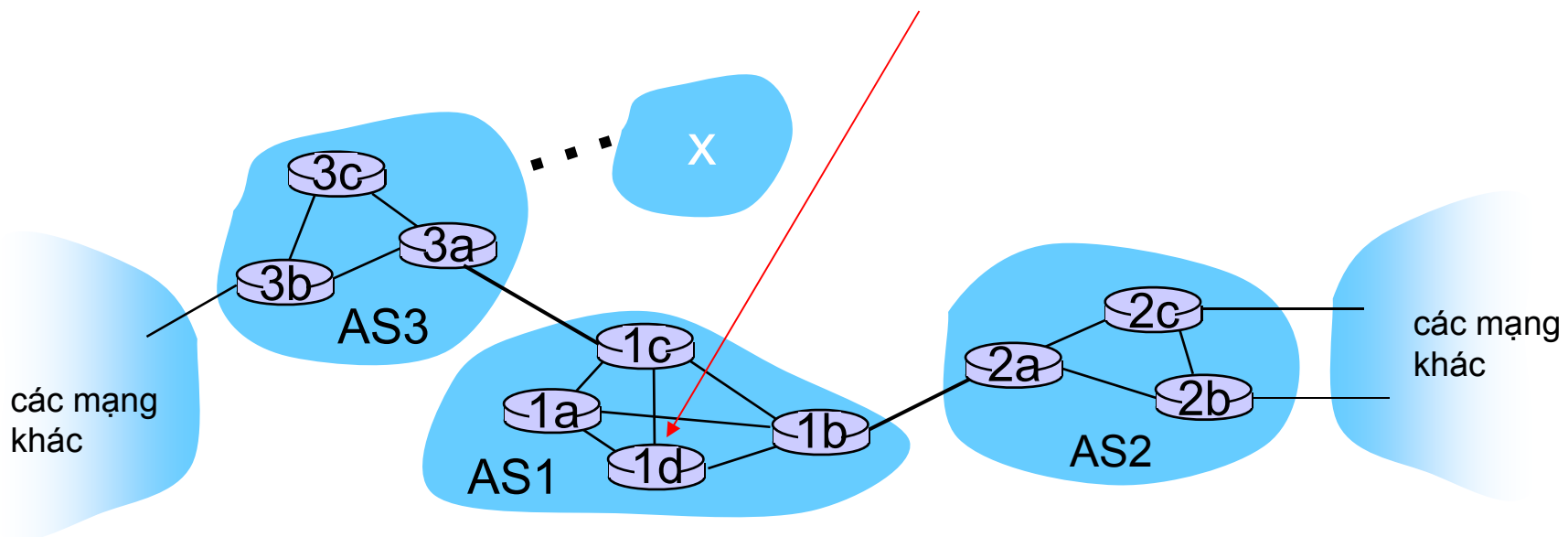
*AS1 phải:*

1. học để biết tới được các đích nào qua AS2, tới được đích nào qua AS3
2. lan truyền thông tin này tới mọi router trong AS1



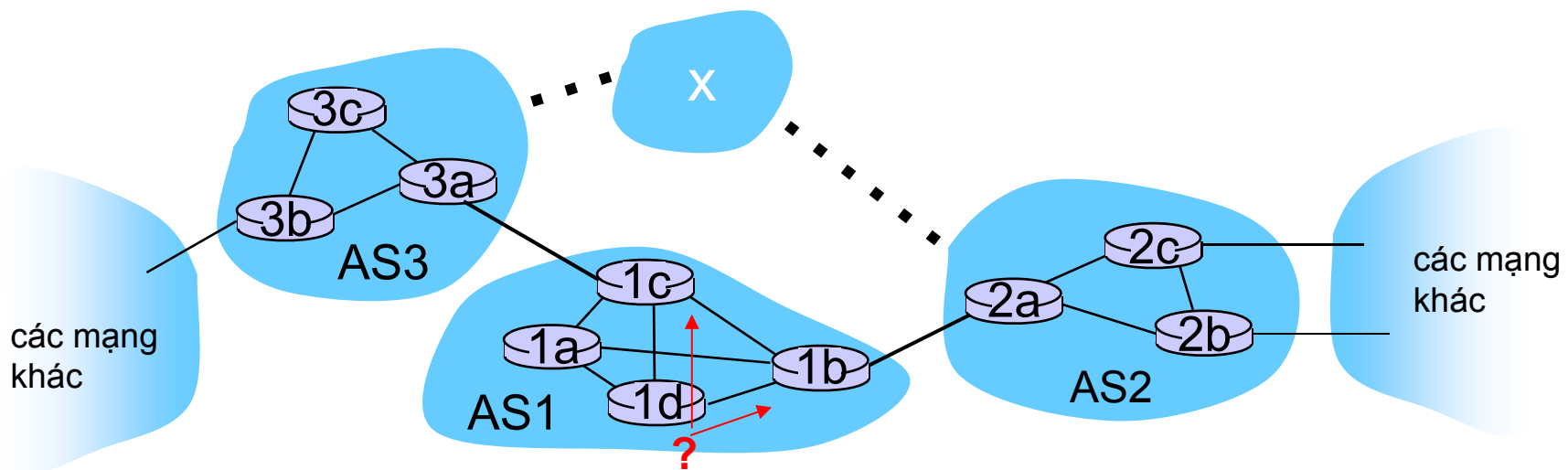
# Ví dụ: thiết lập bảng chuyển tiếp trong router 1d

- ❑ Giả sử AS1 học (qua giao thức inter-AS) là có thể tới được subnet x qua AS3 (gateway 1c), và không tới được qua AS2
  - giao thức inter-AS lan truyền thông tin này mọi internal router
- ❑ router 1d xác định từ thông tin dẫn đường intra-AS là interface / của nó là nằm trên đường chi phí thấp nhất tới 1c
  - cài đặt thông tin  $(x, l)$  trong bảng chuyển tiếp



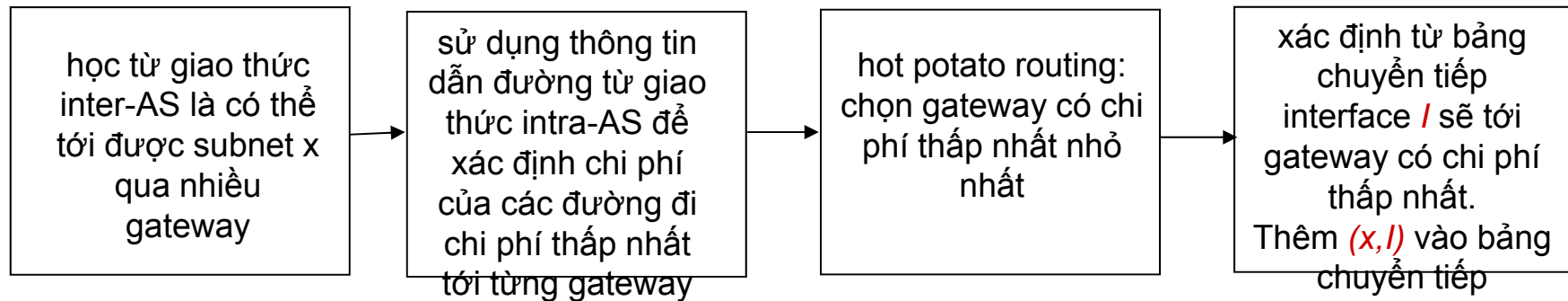
# Ví dụ: chọn giữa nhiều AS

- ❑ Giả sử AS1 học từ giao thức inter-AS là có thể tới được subnet x từ AS3 và từ AS2
- ❑ Để cấu hình bảng chuyển tiếp, router 1d phải xác định gateway nào sẽ chuyển gói tin tới đích x



# Ví dụ: chọn giữa nhiều AS

- ❑ Giả sử AS1 học từ giao thức inter-AS là có thể tới được subnet x từ AS3 và từ AS2
- ❑ Để cấu hình bảng chuyển tiếp, router 1d phải xác định gateway nào sẽ chuyển gói tin tới đích x
- ❑ *hot potato routing: gửi* gói tin tới router gần nhất.



# Chương 4: Tầng mạng

## 4.1 Giới thiệu

## 4.2 IP: Internet Protocol

- Cấu trúc datagram
- IPv4
- ICMP
- IPv6

## 4.3 Giải thuật dẫn đường

- link state
- distance vector
- hierarchical routing

## 4.4 Dẫn đường trong Internet

- RIP
- OSPF
- BGP

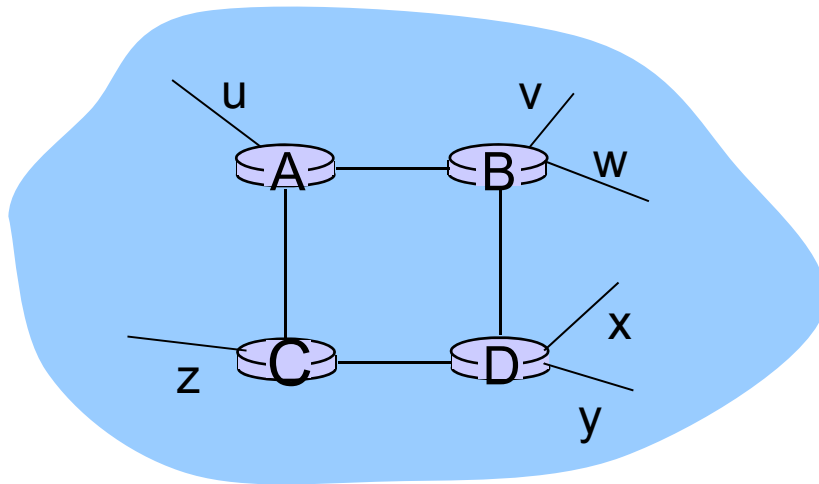
## 4.5 Broadcast routing và multicast routing

# Intra-AS Routing

- ❑ Còn gọi là *interior gateway protocols (IGP)*
- ❑ Các giao thức dẫn đường intra-AS phổ biến:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IGRP: Interior Gateway Routing Protocol  
(Cisco proprietary)

# RIP (Routing Information Protocol)

- ❑ Cài đặt trong BSD-UNIX 1982
- ❑ Giải thuật distance vector
  - đơn vị đo khoảng cách: số hop (tối đa = 15 hop), mỗi liên kết có chi phí bằng 1
  - các véc tơ khoảng cách được trao đổi với các nút kề định kì 30 giây trong bản tin response (còn gọi là **advertisement**)
  - mỗi advertisement: liệt kê tối đa 25 **subnet** đích (*dạng địa chỉ IP*)

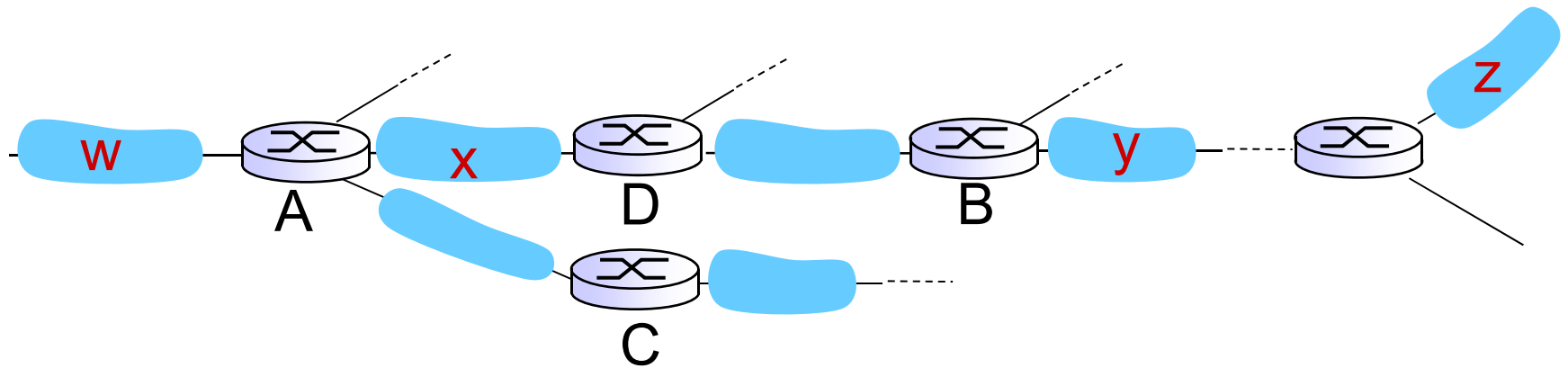


từ router A tới các **subnet** đích

<u>subnet</u>	<u>hops</u>
u	1
v	2
w	2
x	3
y	3
z	2



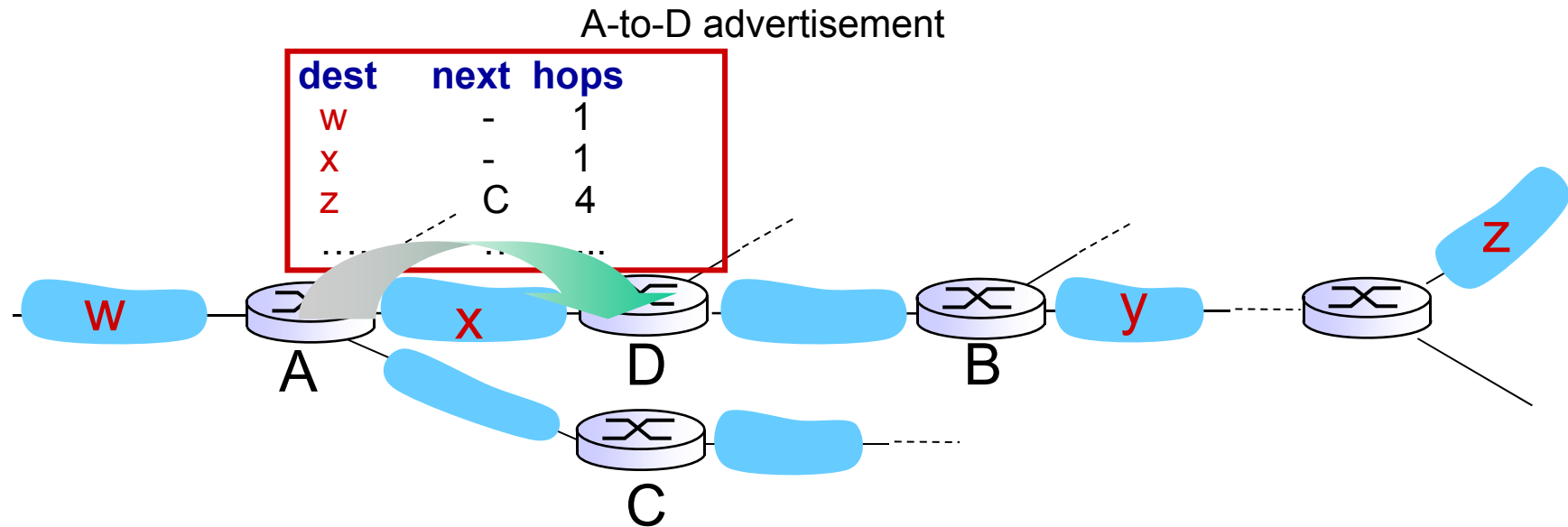
# RIP: Ví dụ



bảng chuyển tiếp trong router D

destination subnet	next router	# hops to dest
w	A	2
y	B	2
x	--	1
....	....	....

# RIP: Ví dụ



bảng chuyển tiếp trong router D

destination subnet	next router	# hops to dest
W	A	2
y	B	2
Z	<del>B</del> → A	<del>7</del> → 5
X	--	1
....	....	....

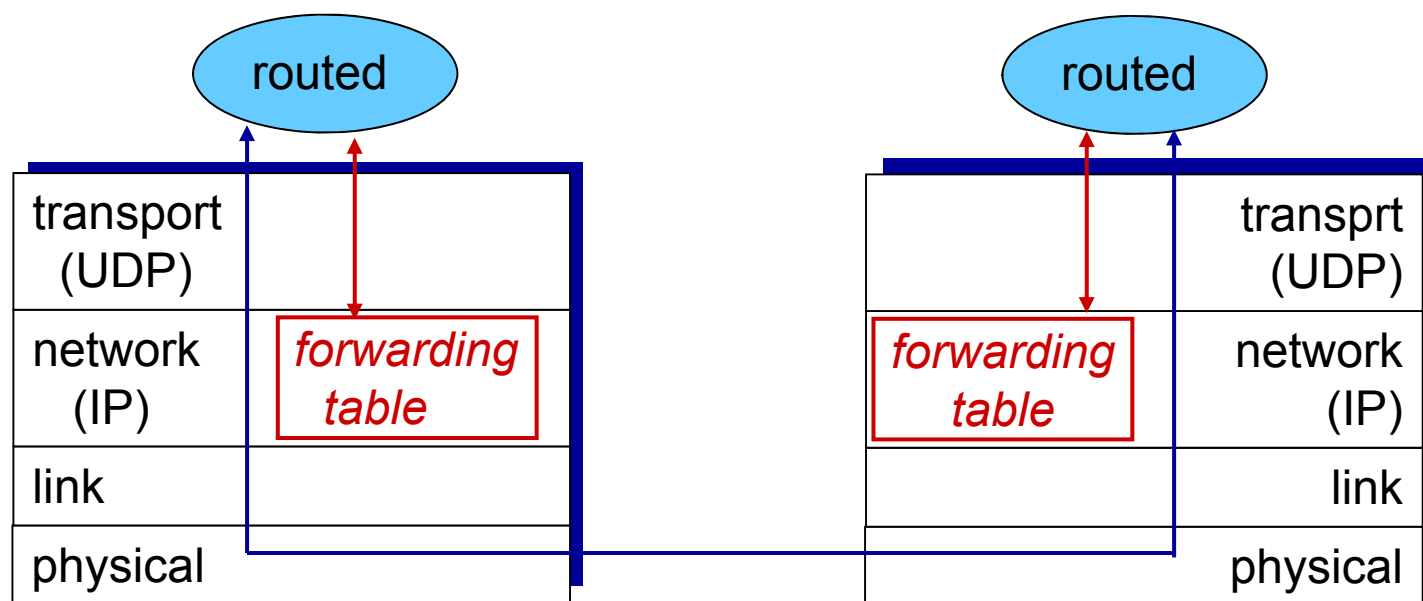
# RIP: Lỗi liên kết, phục hồi

Nếu không thấy advertisement sau 180 giây -> khai báo là nút kề/liên kết không tồn tại

- các đường đi qua nút kề không xác định
- các advertisement mới được gửi tới các nút kề
- các nút kề gửi các advertisement mới (nếu bảng thay đổi)
- thông tin lỗi liên kết lan truyền nhanh trong toàn mạng
- *poison reverse* sử dụng để ngăn chặn ping-pong loop (infinite distance = 16 hop)

# Xử lý bảng RIP

- ❑ RIP routing table do tiến trình mức ứng dụng quản lý gọi là route-d (daemon)
- ❑ Advertisement được gửi trong UDP packet, một cách đều đặn



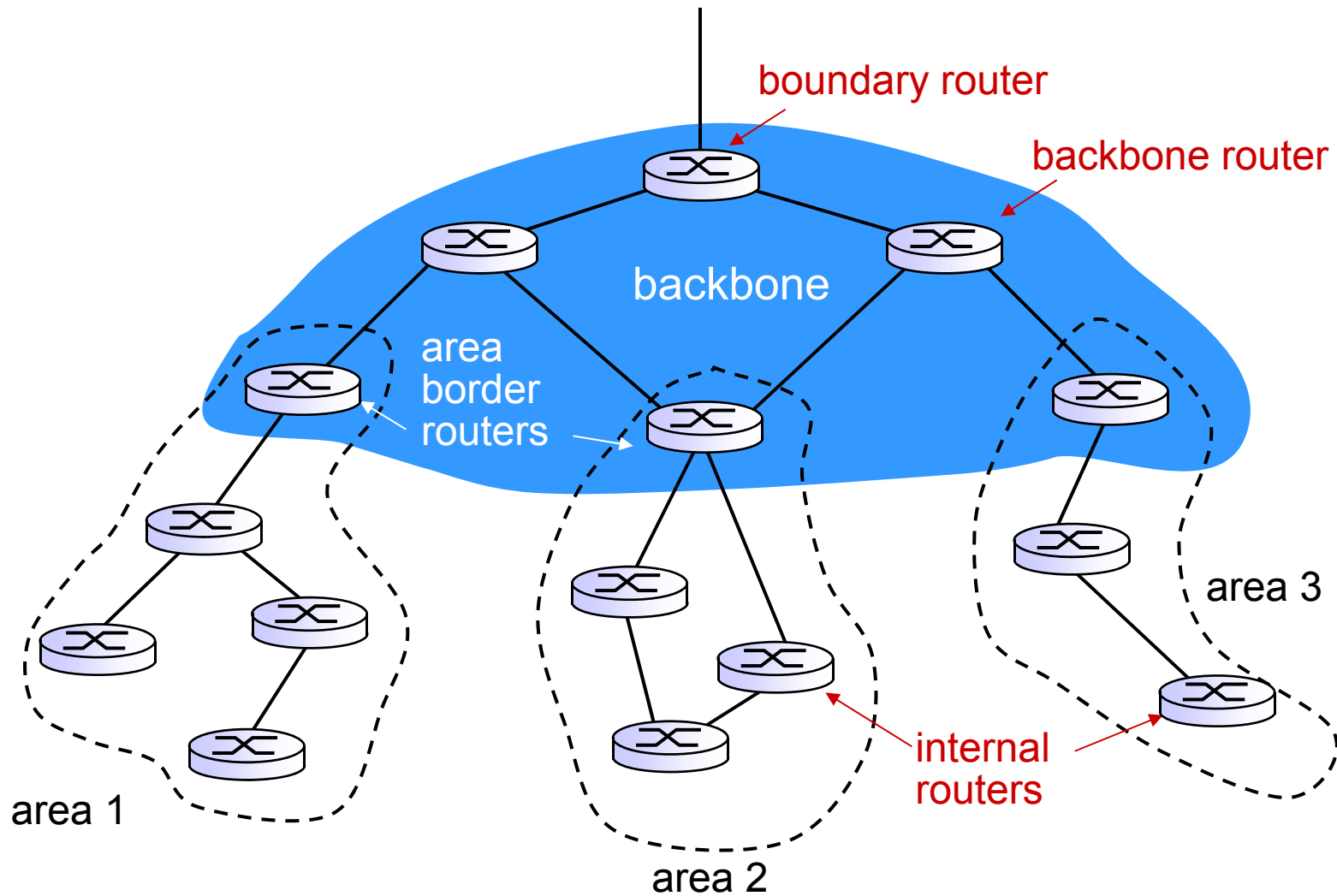
# OSPF (Open Shortest Path First)

- ❑ Dùng giải thuật link state
  - Lan truyền gói tin LS
  - Sơ đồ mạng tại mỗi nút
  - Tính toán đường đi dùng giải thuật Dijkstra
- ❑ OSPF advertisement mạng 1 dòng thông tin cho mỗi nút kề
- ❑ advertisement gửi ra toàn bộ AS
  - được mang trong OSPF message trực tiếp qua IP (thay vì TCP hay UDP)
- ❑ *IS-IS routing* protocol: gần giống OSPF

# Ưu điểm của OSPF so với RIP

- ❑ *An toàn*: Tất cả OSPF message được xác thực
- ❑ Cho phép nhiều đường đi cùng chi phí (Trong RIP chỉ 1 đường)
- ❑ Với mỗi liên kết, nhiều độ đo chi phí cho TOS khác nhau (ví dụ: chi phí liên kết satellite gán thấp đối với best effort ToS; cao đối với real time ToS)
- ❑ hỗ trợ tích hợp unicast và multicast
  - Multicast OSPF (MOSPF) sử dụng cùng cơ sở dữ liệu sơ đồ mạng như OSPF
- ❑ *hierarchical* OSPF trong domain lớn

# Hierarchical OSPF



# Hierarchical OSPF

- ❑ *two-level hierarchy*: local area, backbone
  - link-state advertisement chỉ trong area
  - mỗi nút có area topology; chỉ biết hướng (đường đi ngắn nhất) tới mạng trong area khác
- ❑ *area border router*: tổng hợp các khoảng cách tới các mạng trong area của nó, thông báo cho các Area Border router khác
- ❑ *backbone router*: chạy OSPF routing hạn chế trong backbone
- ❑ *boundary router*: kết nối tới AS khác

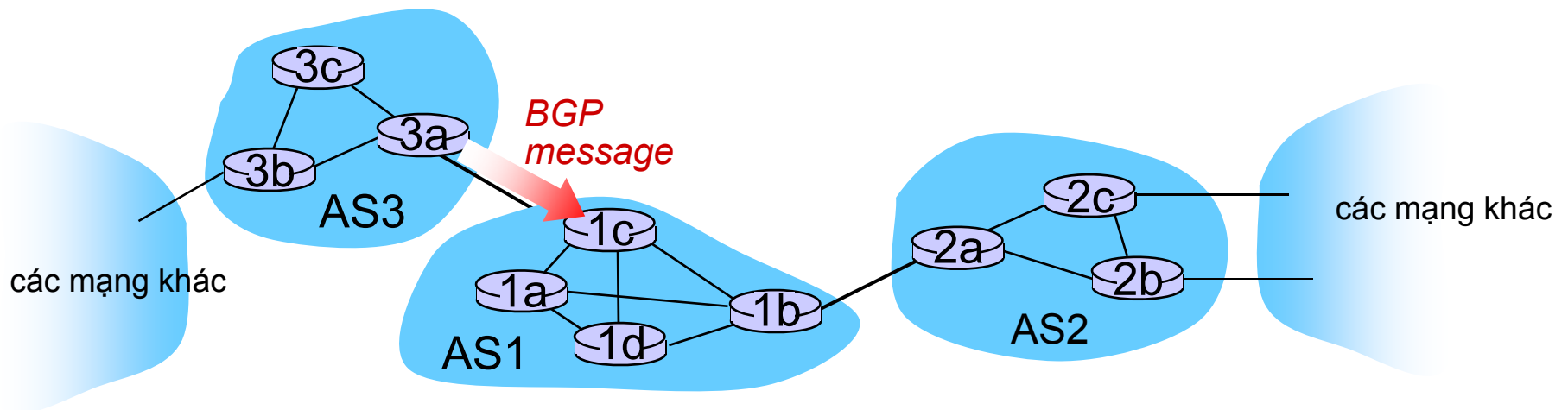


# Dẫn đường inter-AS trong Internet: BGP

- ❑ **BGP: Border Gateway Protocol**
- ❑ BGP cung cấp cho mỗi AS:
  - **eBGP**: lấy thông tin subnet tới được từ các AS kề
  - **iBGP**: lan truyền thông tin tới được tới mọi AS-internal routers
  - xác định “good” route tới các mạng khác dựa vào thông tin tới được và policy
- ❑ Cho phép subnet thông tin sự tồn tại của nó tới Internet

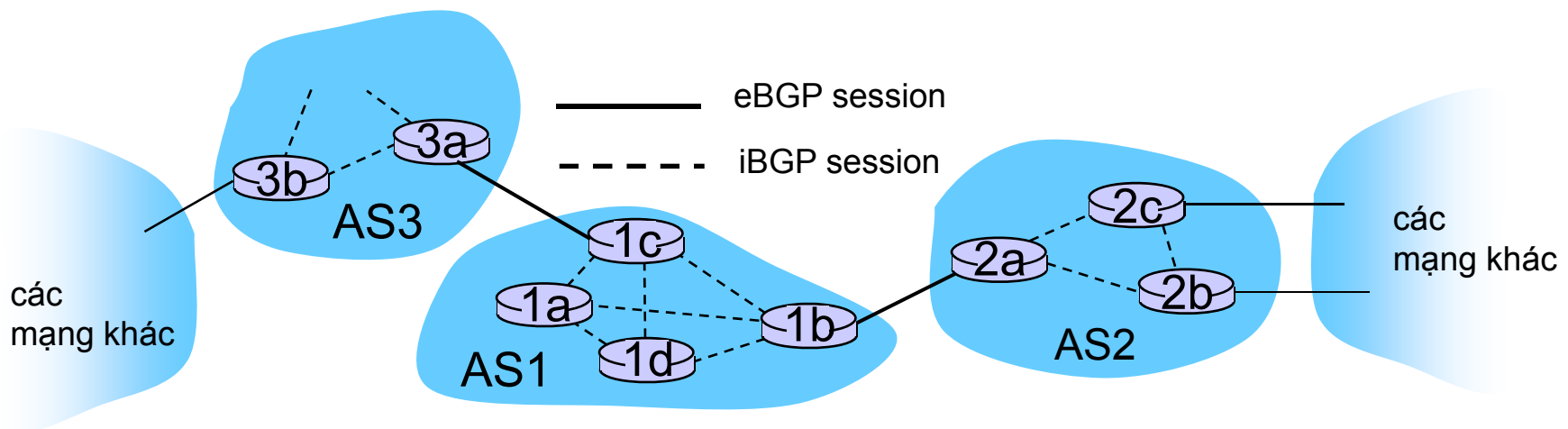
# Cơ bản về BGP

- ❖ **BGP session:** Hai BGP router (“peer”) trao đổi BGP message:
  - thông tin các đường đi tới các network prefix đích (giao thức “path vector”)
  - trao đổi qua kết nối semi-permanent TCP
- Khi AS3 thông tin một prefix tới AS1:
  - AS3 *hứa* sẽ chuyển datagram hướng tới prefix đó
  - AS3 có thể kết tập các prefixe trong advertisement của nó



# Cơ bản về BGP: Phân phát thông tin đường đi

- ❑ Dùng eBGP session giữa 3a và 1c, AS3 gửi thông tin tới được prefix tới AS1
  - 1c có thể dùng iBGP để phân phát thông tin prefix mới tới mọi router trong AS1
  - 1b sau đó có thể thông tin lại thông tin tới được mới tới AS2 qua 1b-to-2a eBGP session
- ❑ Khi router học prefix mới, nó tạo 1 dòng thông tin cho prefix trong bảng chuyển tiếp



# Path attribute và BGP routes

- ❑ Các prefix được lan truyền bao gồm BGP attribute
  - prefix + attributes = “route”
- ❑ Hai attribute quan trọng
  - **AS-PATH**: chứa các AS qua đó prefix advertisement đã qua: ví dụ AS 67, AS 17
  - **NEXT-HOP**: chỉ internal-AS router cụ thể tới next-hop AS (có thể là nhiều liên kết từ AS hiện tại tới next-hop-AS)
- ❑ Gateway router nhận route advertisement sử dụng **import policy** để chấp nhận/từ chối
  - ví dụ: không bao giờ dẫn đường qua AS x
  - *policy-based* routing

# Chọn đường đi BGP

- ❑ Router có thể học nhiều hơn 1 đường tới AS đích, chọn đường dựa vào:
  1. local preference value attribute
  2. AS-PATH ngắn nhất
  3. NEXT-HOP router gần nhất: hot potato routing
  4. các điều kiện khác

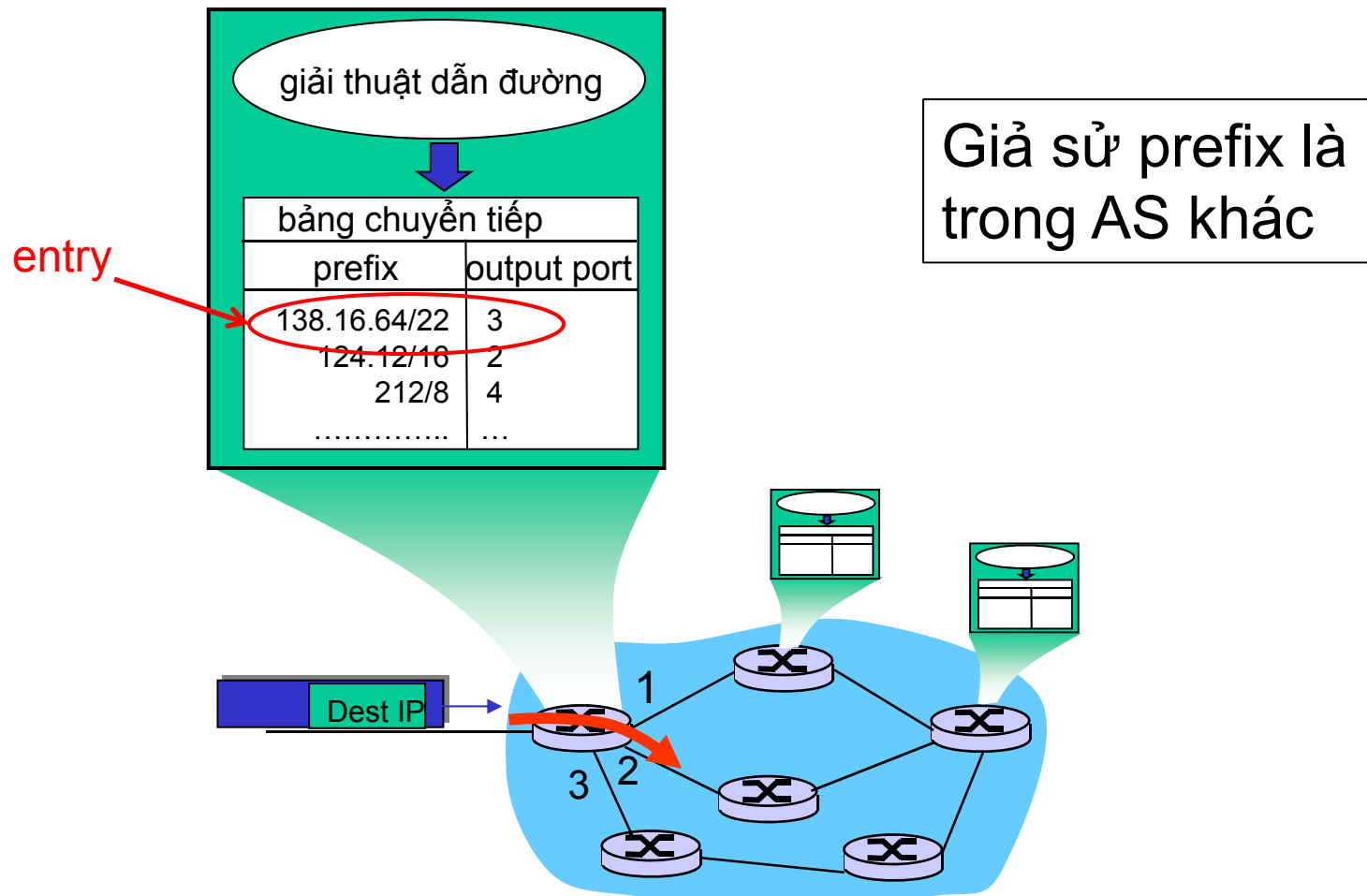
# BGP message

- ❑ BGP message được trao đổi giữa các peer qua kết nối TCP
- ❑ BGP message:
  - **OPEN**: mở kết nối TCP tới peer và xác thực nút gửi
  - **UPDATE**: thông tin đường đi mới (hoặc bỏ đường đi cũ)
  - **KEEPALIVE**: giữ kết nối khi không có UPDATE; và ACK khi có OPEN request
  - **NOTIFICATION**: thông báo lỗi trong bản tin trước, hoặc đóng kết nối

# Một dòng thông tin được đưa vào bảng chuyển tiếp như thế nào

- ❑ Hierarchical routing (Section 4.5.3) với BGP (4.6.3) và OSPF (4.6.2)

# Một dòng thông tin được đưa vào bảng chuyển tiếp như thế nào



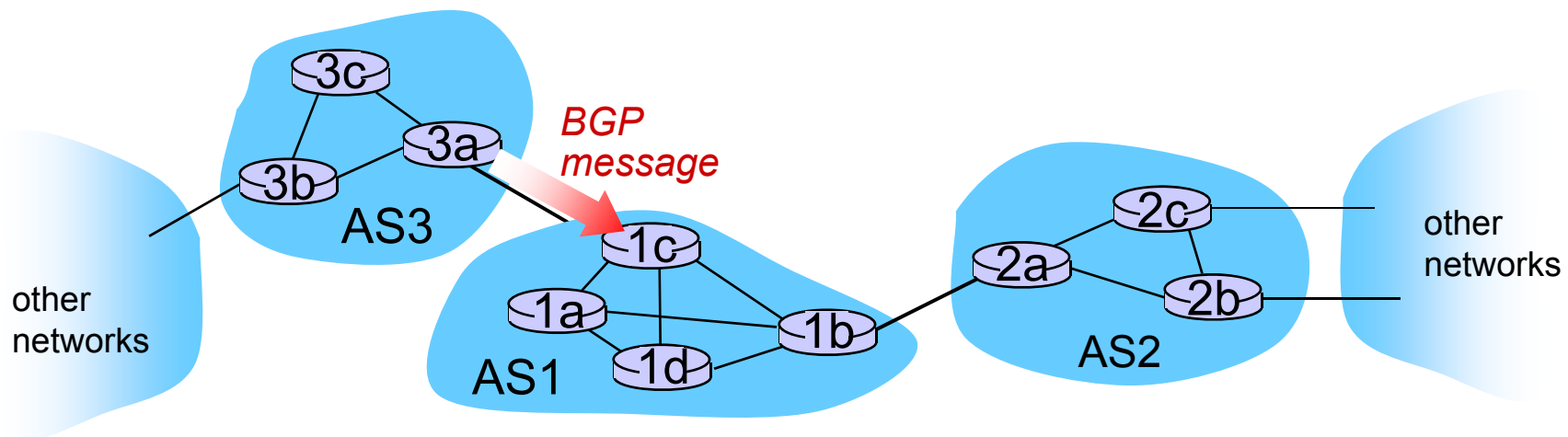


# Một dòng thông tin được đưa vào bảng chuyển tiếp như thế nào

## Tổng quan

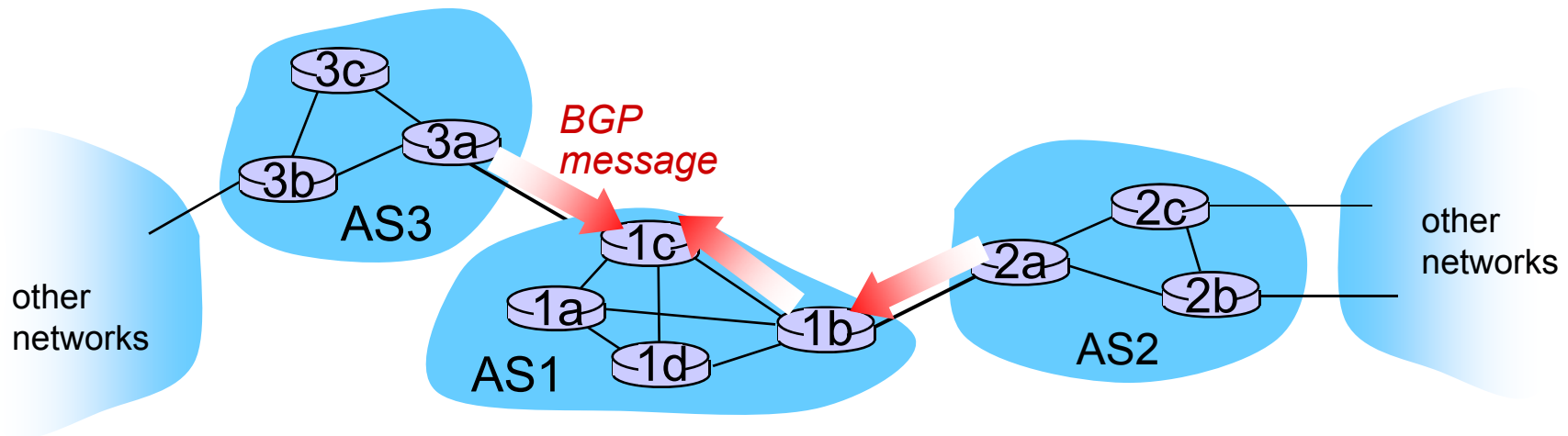
1. Router nhận biết prefix
2. Router xác định output port cho prefix
3. Router đưa vào bảng chuyển tiếp prefix-port

# Router nhận biết prefix



- ❖ BGP message chứa các “route”
- ❖ “route” là một prefix và các attribute: AS-PATH, NEXT-HOP,...
- ❖ Ví dụ route:
  - ❖ Prefix: 138.16.64/22 ; AS-PATH: AS3 AS131 ; NEXT-HOP: 201.44.13.125

# Router có thể nhận nhiều route



- ❖ Router có thể nhận nhiều route cho cùng một prefix
- ❖ Router phải chọn 1 route

# BGP chọn route cho prefix

- Router chọn route dựa trên AS-PATH ngắn nhất

❖ Ví dụ:

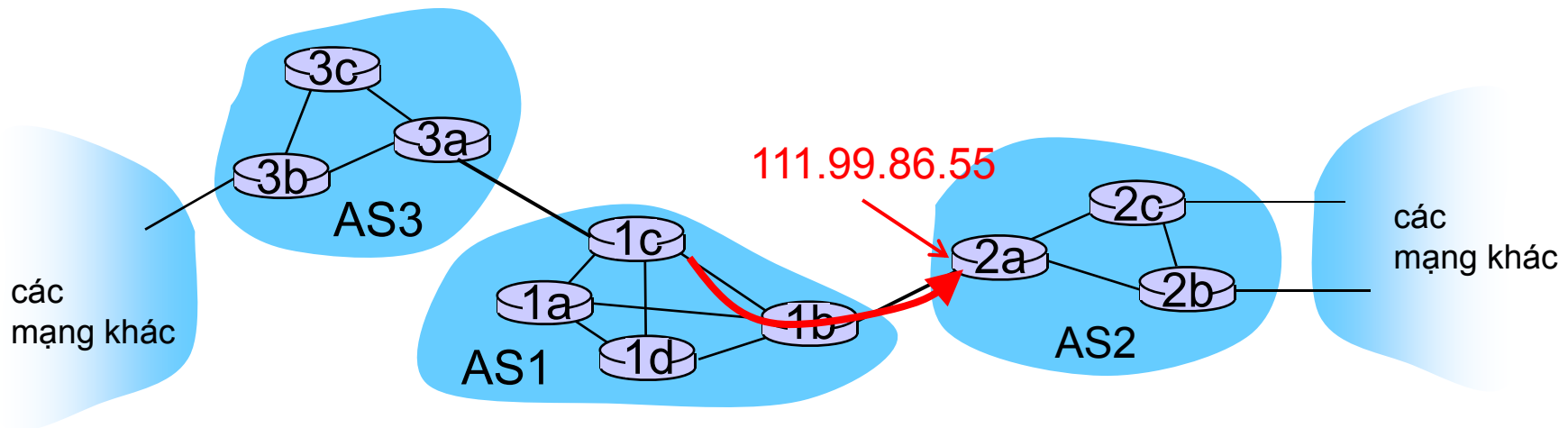
- ❖ AS2 AS17 to 138.16.64/22
- ❖ AS3 AS131 AS201 to 138.16.64/22

chọn



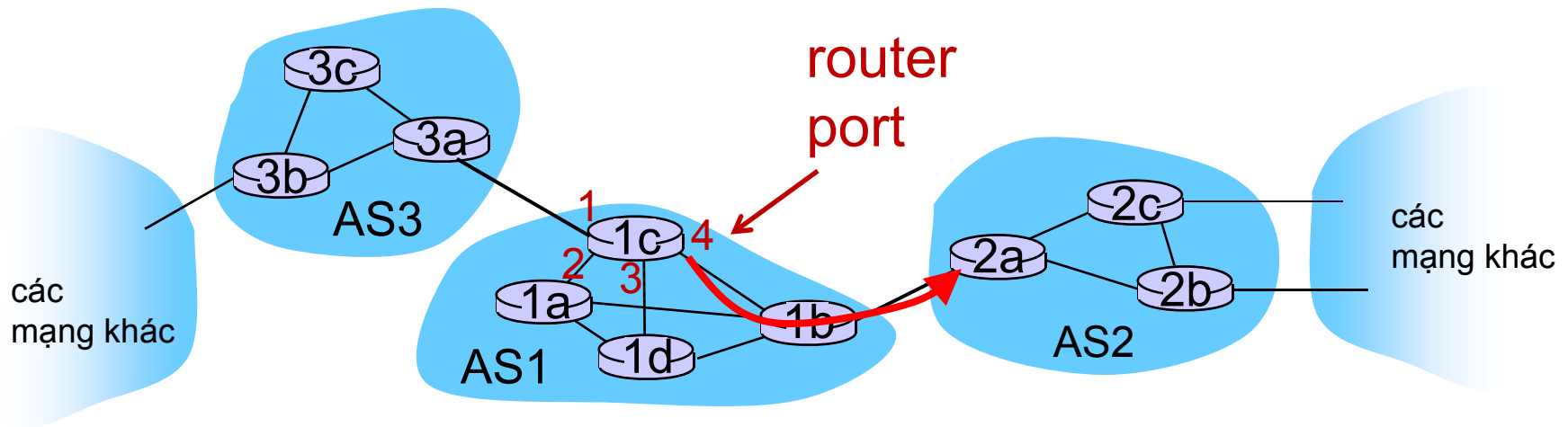
# Tìm intra-route tốt nhất cho BGP route

- ❑ Dùng NEXT-HOP attribute của route đã chọn
  - NEXT-HOP attribute của route là địa chỉ IP của router interface bắt đầu AS PATH.
- ❑ Ví dụ:
  - ❖ AS-PATH: AS2 AS17 ; NEXT-HOP: 111.99.86.55
- ❑ Router dùng OSPF để tìm đường ngắn nhất từ 1c tới 111.99.86.55



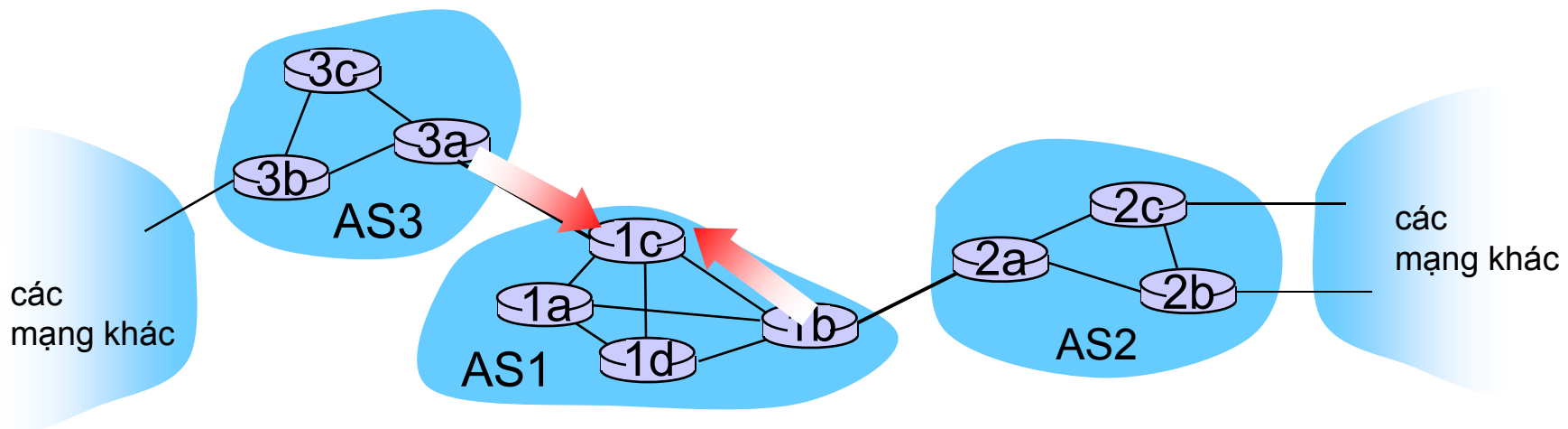
# Router xác định port cho route

- ❑ Xác định port dọc theo OSPF shortest path
- ❑ Thêm dòng thông tin prefix-port vào bảng chuyển tiếp (138.16.64/22 , port 4)



# Hot Potato Routing

- ❑ Giả sử có hai hoặc nhiều hơn inter-route tốt nhất
- ❑ Thì chọn route có NEXT-HOP gần nhất
  - Sử dụng OSPF để xác định gateway gần nhất
  - Từ 1c, chọn AS3 AS131 hay AS2 AS17?
  - Trả lời: route AS3 AS201 vì gần hơn



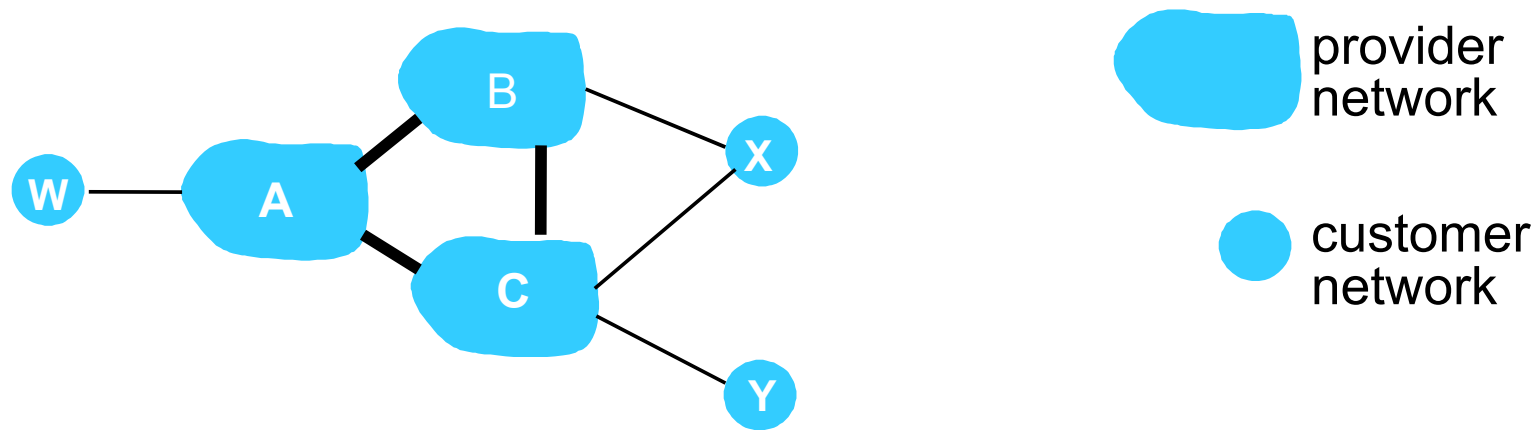
# Một dòng thông tin được đưa vào bảng chuyển tiếp như thế nào

## Tóm tắt

1. Router nhận biết prefix
  - qua các BGP route advertisement từ các router khác
2. Xác định router output port cho prefix
  - Sử dụng BGP route selection để tìm inter-AS route tốt nhất
  - Sử dụng OSPF để tìm intra-AS route tốt nhất dẫn tới inter-AS route tốt nhất
  - Router xác định router port cho best route đó
3. Đưa vào bảng chuyển tiếp dòng thông tin prefix-port

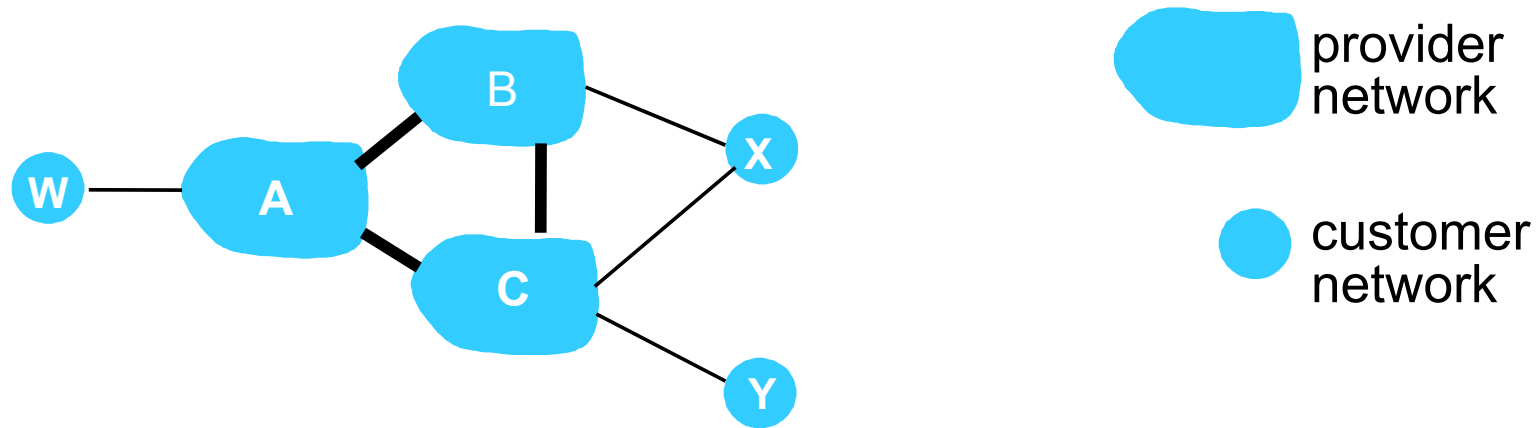


# BGP routing policy



- ❖ A,B,C là các *provider network*
- ❖ X,W,Y là customer (của provider network)
- ❖ X là *dual-homed* (nối tới 2 mạng)
  - X không muốn đi từ B qua X tới C
  - .. vì vậy X sẽ không thông tin cho B route tới C

# BGP routing policy



- ❖ A thông tin path AW to B
- ❖ B thông tin path BAW to X
- ❖ B có nên thông tin path BAW to C?
  - Không! B không nhận lợi nhuận cho việc dẫn đường CBAW vì cả W và C không phải là customer của B
  - B muốn buộc C dẫn đường tới w qua A
  - B muốn chỉ dẫn đường tới/từ customer của B

# Tại sao dẫn đường Intra-AS, Inter-AS khác nhau

## *Chính sách:*

- ❑ inter-AS: người quản trị muốn điều khiển lưu lượng được dẫn đường như thế nào, ai có thể qua mạng của họ
- ❑ intra-AS: một người quản trị, vì vậy không cần chính sách

## *Quy mô:*

- ❑ hierarchical routing giảm kích thước bảng, giảm lưu lượng cập nhật

## *Hiệu năng:*

- ❑ intra-AS: có thể tập trung vào hiệu năng
- ❑ inter-AS: chính sách có thể xem xét trước so với hiệu năng

# Chương 4: Tầng mạng

## 4.1 Giới thiệu

## 4.2 IP: Internet Protocol

- Cấu trúc datagram
- IPv4
- ICMP
- IPv6

## 4.3 Giải thuật dẫn đường

- link state
- distance vector
- hierarchical routing

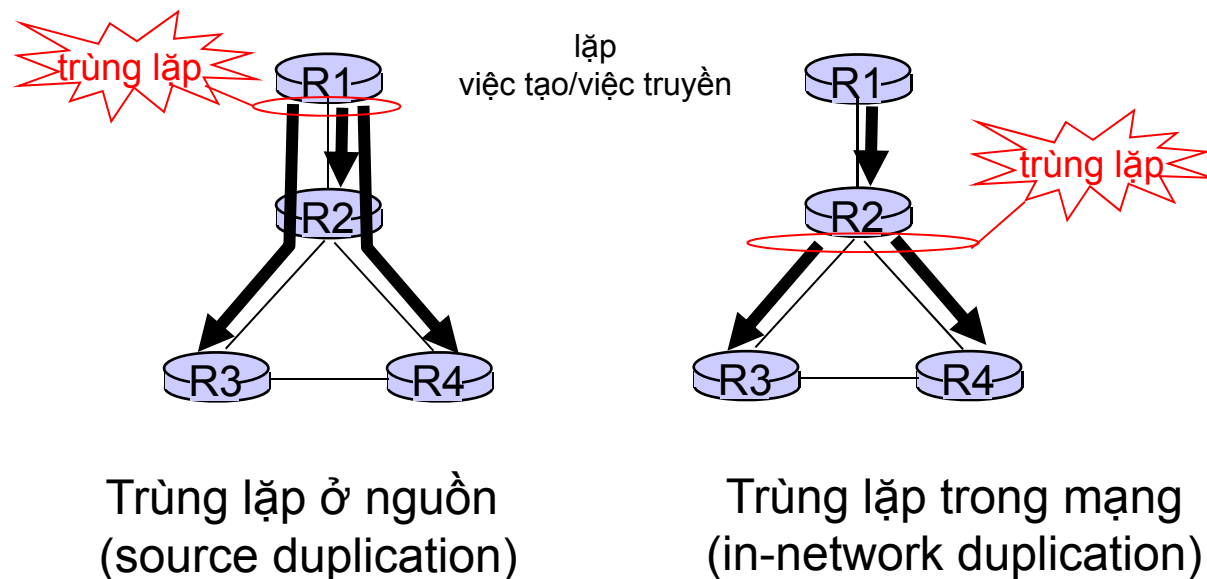
## 4.4 Dẫn đường trong Internet

- RIP
- OSPF
- BGP

## 4.5 Broadcast routing và multicast routing

# Broadcast routing

- ❑ Chuyển các gói tin từ nguồn tới mọi nút đích khác
- ❑ source duplication là không hiệu quả



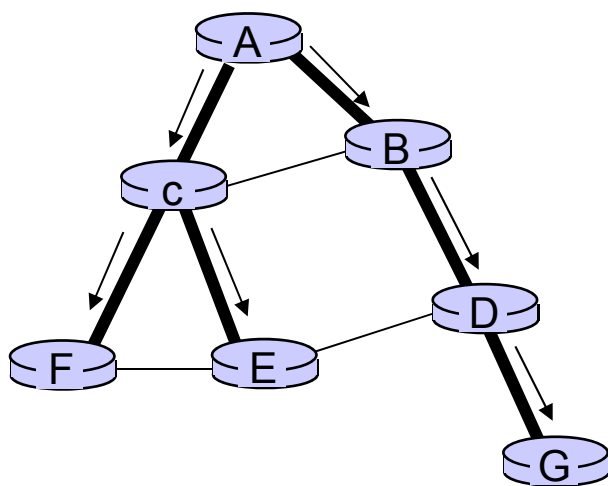
- ❖ source duplication: nút nguồn xác định địa chỉ nhận như thế nào?

# In-network duplication

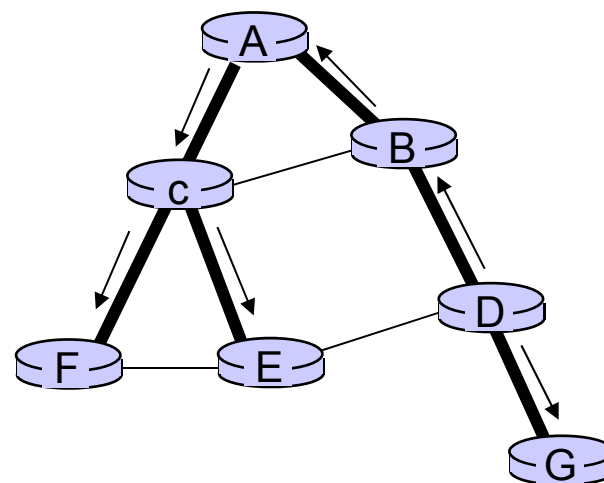
- ❑ ***Flooding:*** Khi nút nhận broadcast packet, gửi bản sao tới mọi nút kề
  - Vấn đề: chu trình, broadcast storm
- ❑ ***Controlled flooding:*** nút chỉ quảng bá gói tin khi nó chưa nhận gói tin này trước đó
  - nút lưu danh sách các packet id đã được quảng bá
  - hoặc reverse path forwarding (RPF): chỉ chuyển tiếp gói tin nếu nó đến trên đường đi ngắn nhất giữa nút và nguồn
- ❑ ***Spanning tree:***
  - không có gói tin dư thừa nhận bởi bất kì nút nào

# Spanning tree

- Đầu tiên xây dựng spanning tree
- Sau đó các nút chuyển tiếp/ tạo bản sao chỉ dọc theo spanning tree



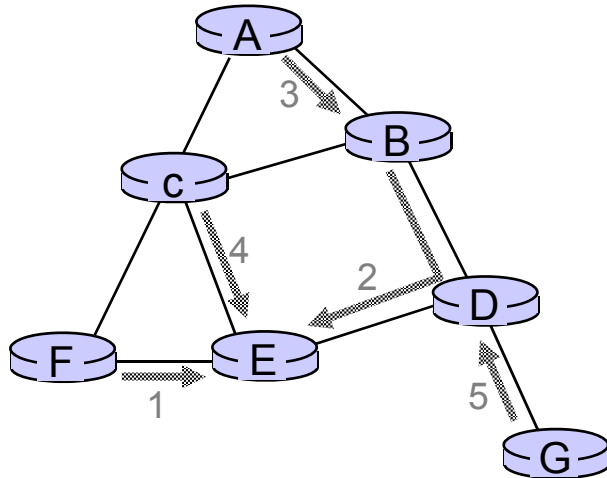
(a) broadcast khởi tạo tại A



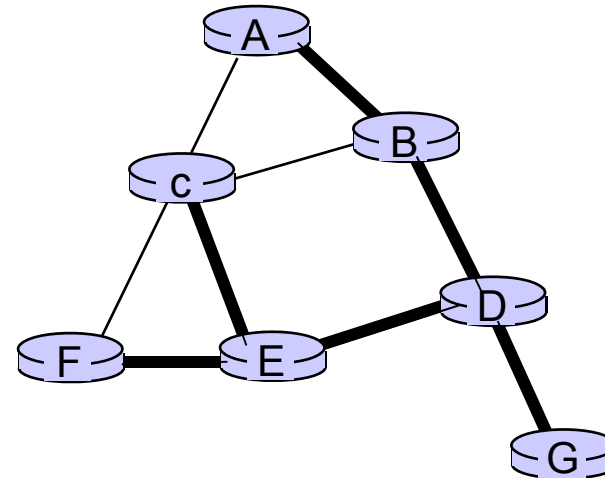
(b) broadcast khởi tạo tại D

# Spanning tree: Tạo

- ❑ Center node
- ❑ Mỗi nút gửi unicast join message tới center node
  - message được chuyển tiếp tới khi nó đến một nút đã thuộc về spanning tree



(a) xây dựng spanning tree  
(center: E)



(b) spanning tree được xây dựng



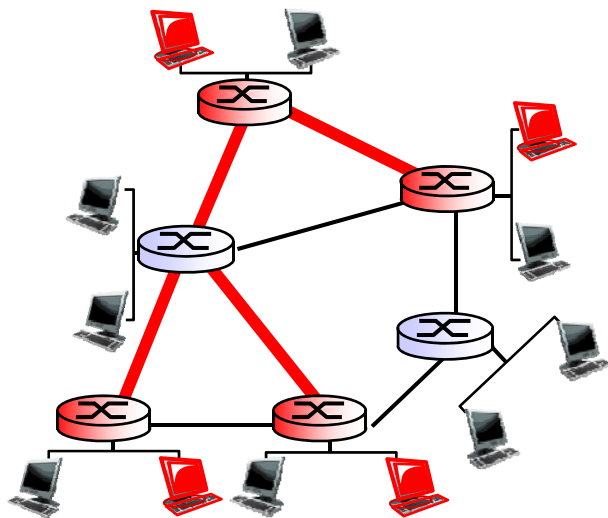
# Multicast routing: Vấn đề

**Mục đích:** Tìm một cây (hoặc các cây) nối các router có các local mcast group member

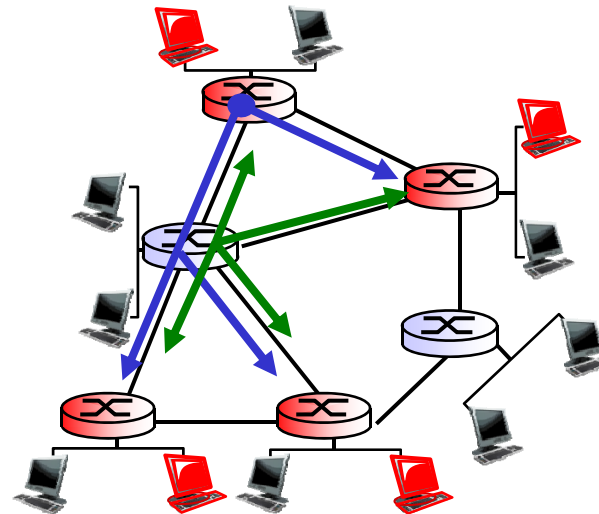
□ **tree:** không phải mọi đường đi giữa các được dùng

□ **shared-tree:** cùng cây dùng bởi tất cả group member

**source-based:** cây khác nhau từ mỗi nút gửi tới các nút nhận



shared tree



source-based trees

legend



group member



không phải group member



router với một group member



router không có group member

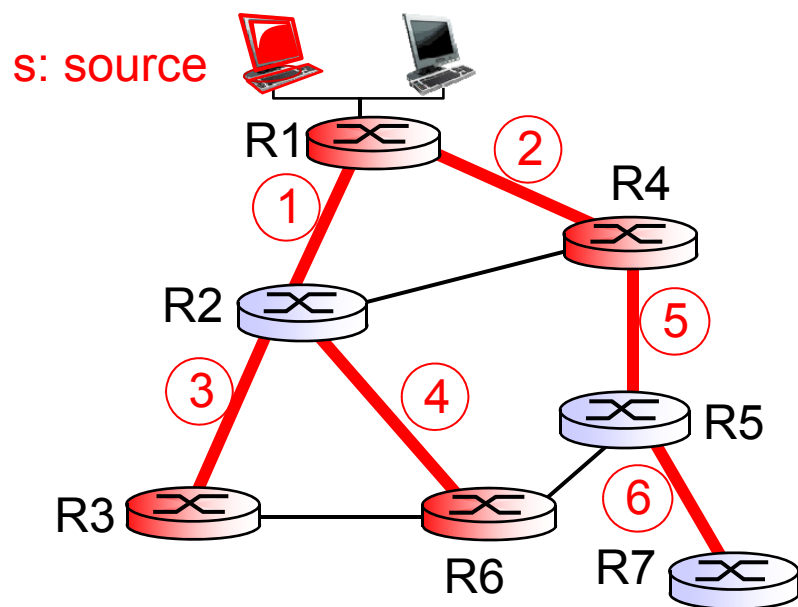
# Xây dựng mcast tree

- ❑ *source-based tree*: một cây cho mỗi nguồn
  - cây đường đi ngắn nhất
  - reverse path forwarding
- ❑ *group-shared tree*: group dùng một cây
  - minimal spanning (Steiner)
  - center-based tree


# Shortest path tree

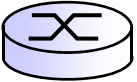
❑ mcast forwarding tree: tree of shortest path routes from source to all receivers


○ Dijkstra's algorithm



## LEGEND

 router with attached group member

 router with no attached group member

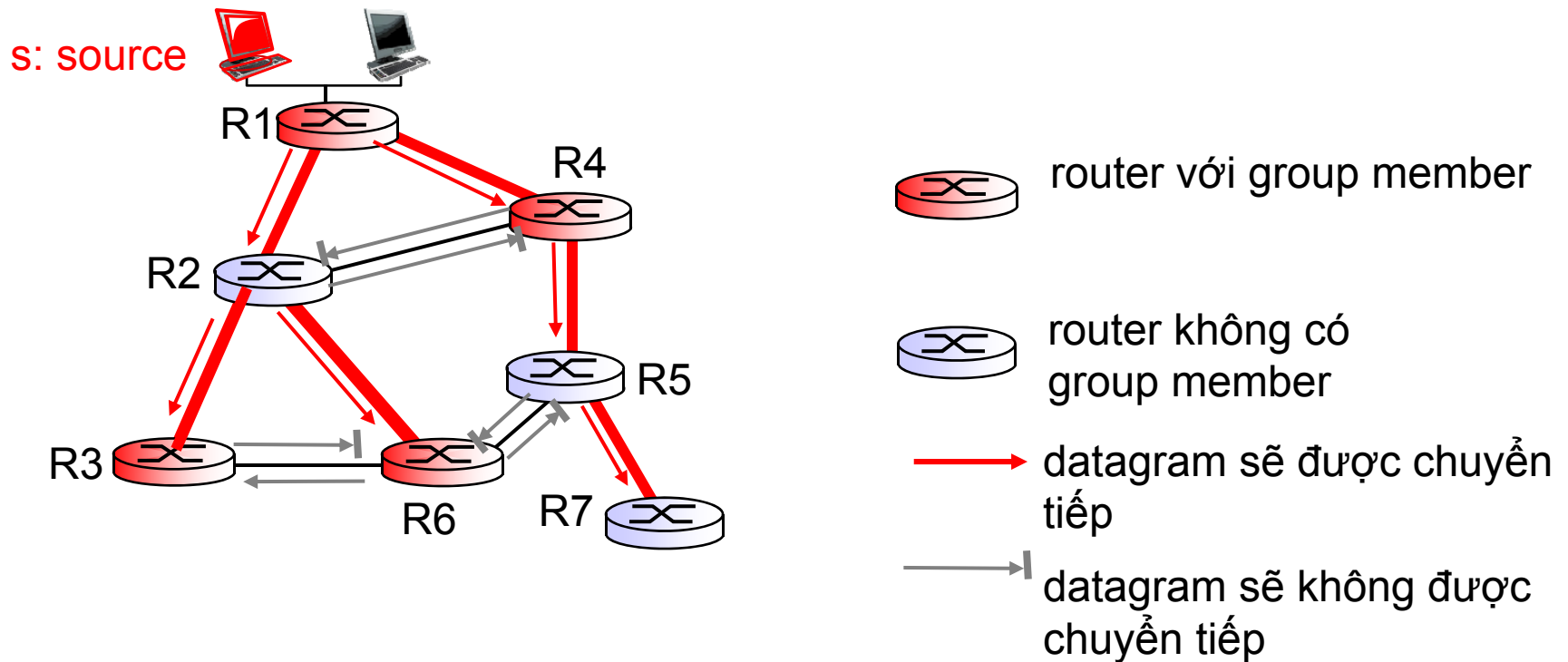
 link used for forwarding, i indicates order link added by algorithm

# Reverse path forwarding

- ❖ Dựa vào kiến thức router về unicast shortest path từ nó tới nút gửi
- ❖ Mỗi router thực hiện chuyển tiếp như sau:

***if*** (mcast datagram nhận trên incoming link trên shortest path ngược về center)  
***then*** gửi datagram trên mọi outgoing link  
***else*** bỏ qua datagram

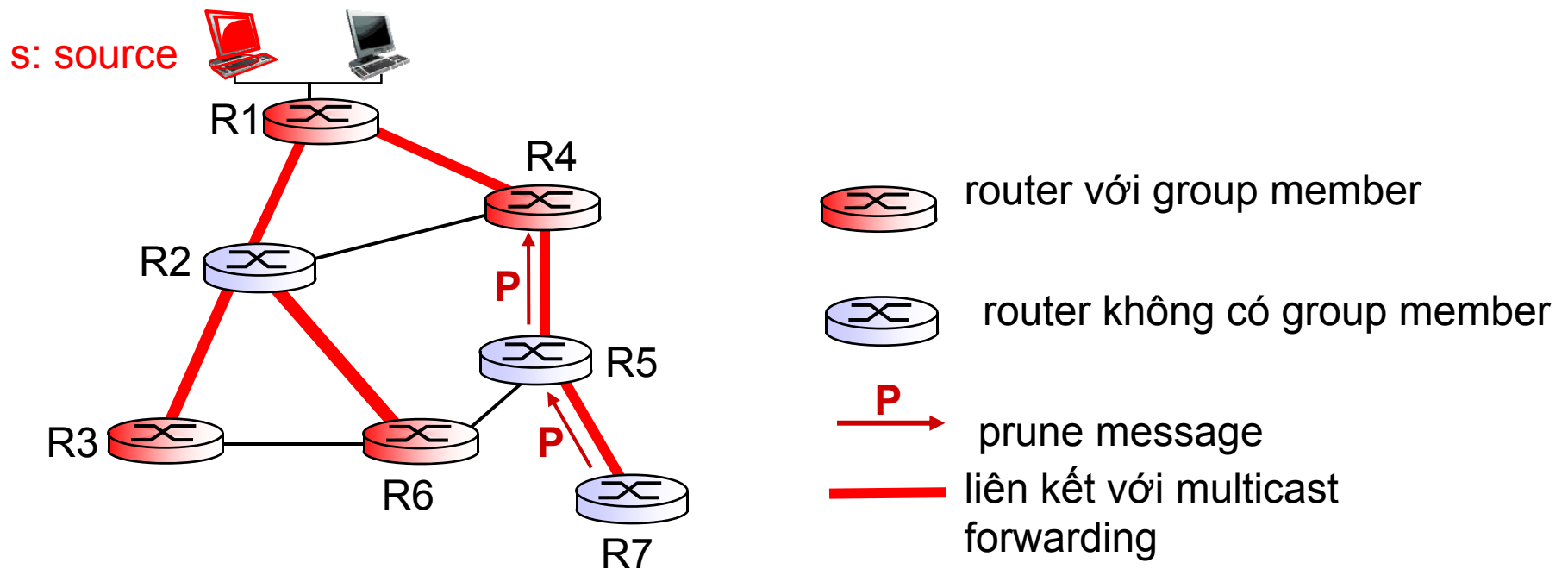
# Reverse path forwarding: Ví dụ



- ❖ Kết quả là source-specific *reverse* SPT
  - có thể không tốt đối với các asymmetric link

# Reverse path forwarding: Tỉa

- forwarding tree chứa các subtree không có mcast group member
  - không cần chuyển datagram xuống subtree
  - tỉa bản tin được gửi lên bởi router không có downstream group member



# Shared-tree: steiner tree

- ❑ *steiner tree*: cây chi phí nhỏ nhất kết nối mọi router với group member
- ❑ Bài toán NP-complete
- ❑ Tồn tại heuristics tốt
- ❑ Không dùng trong thực tế:
  - độ phức tạp tính toán
  - cần thông tin về toàn bộ mạng
  - chạy lại mỗi khi có một router tham gia vào mạng hay ra khỏi mạng

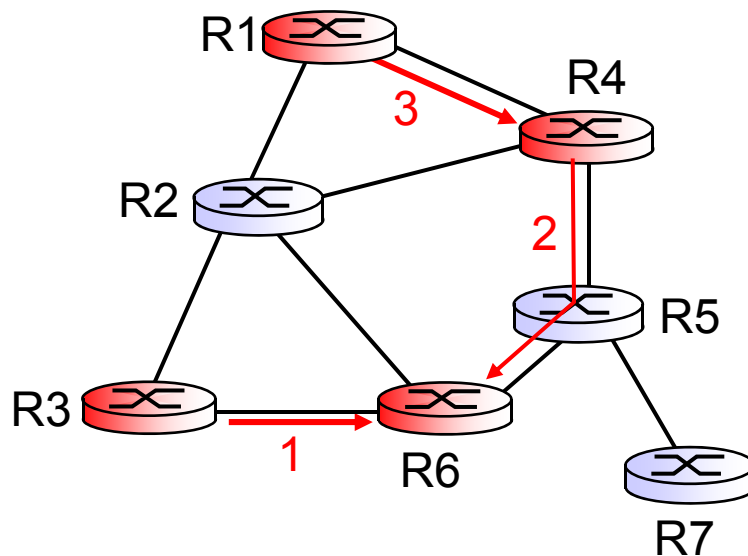
# Center-based tree




- ❑ Tất cả dùng chung một delivery tree
- ❑ Một router gọi là “*center*” của cây
- ❑ Gia nhập:
  - edge router gửi unicast *join-msg* tới center router
  - *join-msg* được xử lý bởi các intermediate router và được chuyển tiếp tới center
  - *join-msg* hoặc vào nhánh cây đã có cho center này hoặc tới center
  - đường đi của *join-msg* thành nhánh mới của cây cho router này



# Center-based tree: Ví dụ

Giả sử R6 được chọn là center:



-  router với group member
-  router không có group member
-  thứ tự path trong đó các join message được sinh ra

# Internet Multicasting Routing: DVMRP

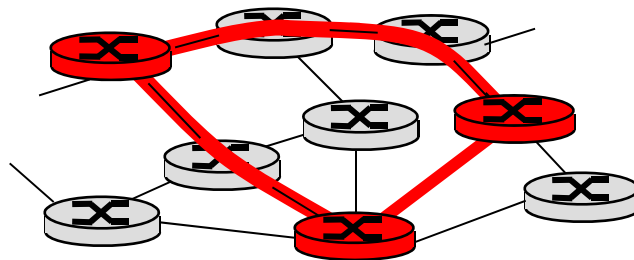
- ❑ **DVMRP**: distance vector multicast routing protocol, RFC1075
- ❑ *flood and prune*: reverse path forwarding, source-based tree
  - RPF tree dựa vào bảng dẫn đường của chính DVMRP xây dựng bằng cách trao đổi với các DVMRP router
  - không giả sử về unicast ở dưới
  - datagram ban đầu tới mcast group gửi tới tất cả qua RPF
  - router không muốn group: gửi upstream prune message

# DVMRP

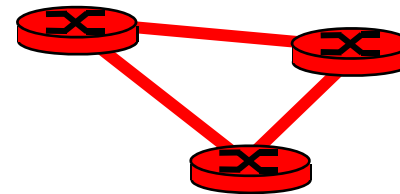
- ❑ *soft state*: DVMRP router định kì (1 min.) bỏ đi nhánh đã tỉa
  - mcast data lại có thể chuyển xuống unpruned branch
  - downstream router: tỉa lại hoặc tiếp tục nhận dữ liệu
- ❑ routers nhanh chóng regraft to tree
  - theo IGMP gia nhập tại lá
- ❑ odds and ends
  - thực hiện trong router thương mại

# Tunneling

Kết nối các island của multicast router trong rất nhiều unicast router như thế nào? > of unicast routers?



physical topology



logical topology

- ❖ mcast datagram đóng gói trong “normal” (non-multicast-addressed) datagram
- ❖ normal IP datagram được gửi qua “tunnel” qua IP unicast thông thường tới mcast router nhận (xem lại tunneling IPv6 trong IPv4)
- ❖ mcast router nhận mở gói để lấy mcast datagram

# PIM: Protocol Independent Multicast

- ❑ Không phụ thuộc vào giải thuật dẫn đường unicast bên dưới (làm việc với tất cả)
- ❑ Hai kịch bản multicast distribution:

## *dày:*

- ❖ group members dày đặc, gần.
- ❖ băng thông nhiều

## *thưa:*

- ❖ số mạng với group member nhỏ so với số mạng kết nối
- ❖ group member phân tán rộng
- ❖ băng thông không nhiều

# Chương 4

## 4.1 Giới thiệu

## 4.4 IP: Internet Protocol

- cấu trúc datagram, IPv4 addressing, ICMP, IPv6

## 4.5 Giải thuật dẫn

đường link state, distance vector, hierarchical routing

## 4.6 Dẫn đường trong Internet

- RIP, OSPF, BGP

## 4.7 Broadcast routing và multicast routing

# An interview with Vinton G. Cerf

- Vinton G. Cerf is Vice President and Chief Internet Evangelist for Google. He is widely known as the co-designer of the TCP/IP protocols and the architecture of the Internet. During his time from 1976 to 1982 at the US Department of Defense Advanced Research Projects Agency (DARPA), he played a key role leading the development of Internet and Internet-related data packet and security techniques. He received the US Presidential Medal of Freedom in 2005 and the US National Medal of Technology in 1997. He holds a BS in Mathematics from Stanford University and an MS and PhD in computer science from UCLA.
- Do you have any advice for students entering the networking/Internet field?
  - Think outside the limitations of existing systems—imagine what might be possible; but then do the hard work of figuring out how to get there from the current state of affairs. Dare to dream: A half dozen colleagues and I at the Jet Propulsion Laboratory have been working on the design of an interplanetary extension of the terrestrial Internet. It may take decades to implement this, mission by mission, but to paraphrase: “A man’s reach should exceed his grasp, or what are the heavens for?”

# Mạng máy tính

- Hình ảnh và nội dung trong bài giảng này có tham khảo từ sách và bài giảng của TS. J.F. Kurose and GS. K.W. Ross