

# Chương 4 Tầng Network

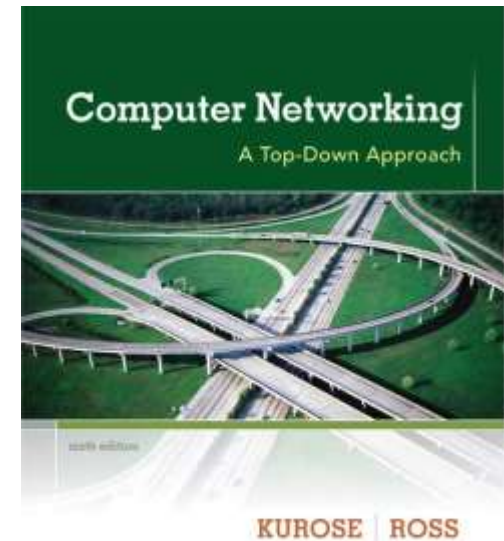
## A note on the use of these ppt slides:

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

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

Thanks and enjoy! JFK/KWR

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



**Computer  
Networking: A Top  
Down Approach**  
6<sup>th</sup> edition  
Jim Kurose, Keith Ross  
Addison-Wesley  
March 2012

# Chương 4: tầng network

## *Mục tiêu:*

- ❖ Hiểu các nguyên lý nền tảng của các định vụ tầng network:
  - Các mô hình dịch vụ tầng network
  - forwarding so với routing
  - Cách mà router hoạt động
  - routing (chọn đường)
  - broadcast, multicast
- ❖ Hiện thực trong Internet

# Chương 4: Nội dung

## 4.1 Giới thiệu

4.2 virtual circuit network (Mạng mạch ảo) và datagram network

4.3 Cấu trúc bên trong router

4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

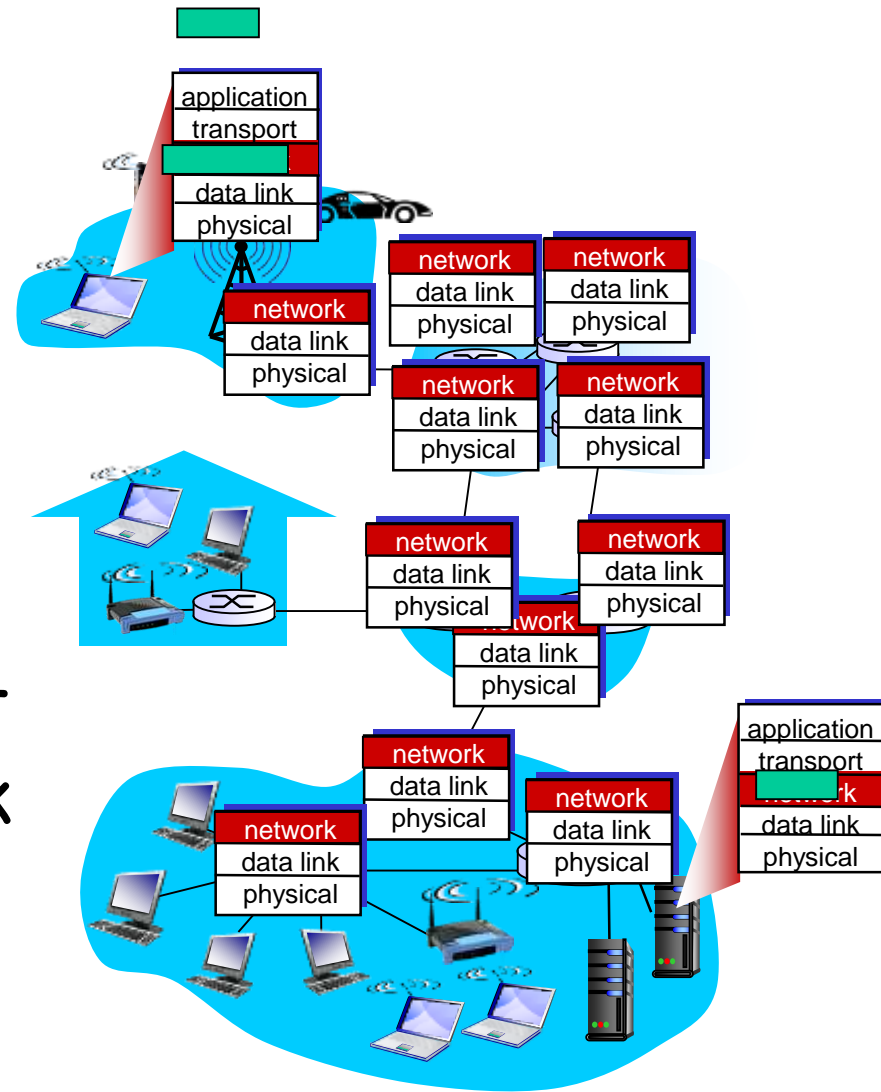
4.6 routing trong Internet

- RIP
- OSPF
- BGP

4.7 broadcast và multicast routing

# Tầng Network

- ❖ Segment của tầng transport từ host gửi đến host nhận
- ❖ Bên phía gửi sẽ đóng gói (encapsulate) các segment vào trong các datagram
- ❖ Bên phía nhận, chuyển các segment lên tầng transport
- ❖ Các giao thức tầng network trong *mọi* host, mọi router
- ❖ router sẽ xem xét các trường của header trong tất cả các IP datagram đi qua nó



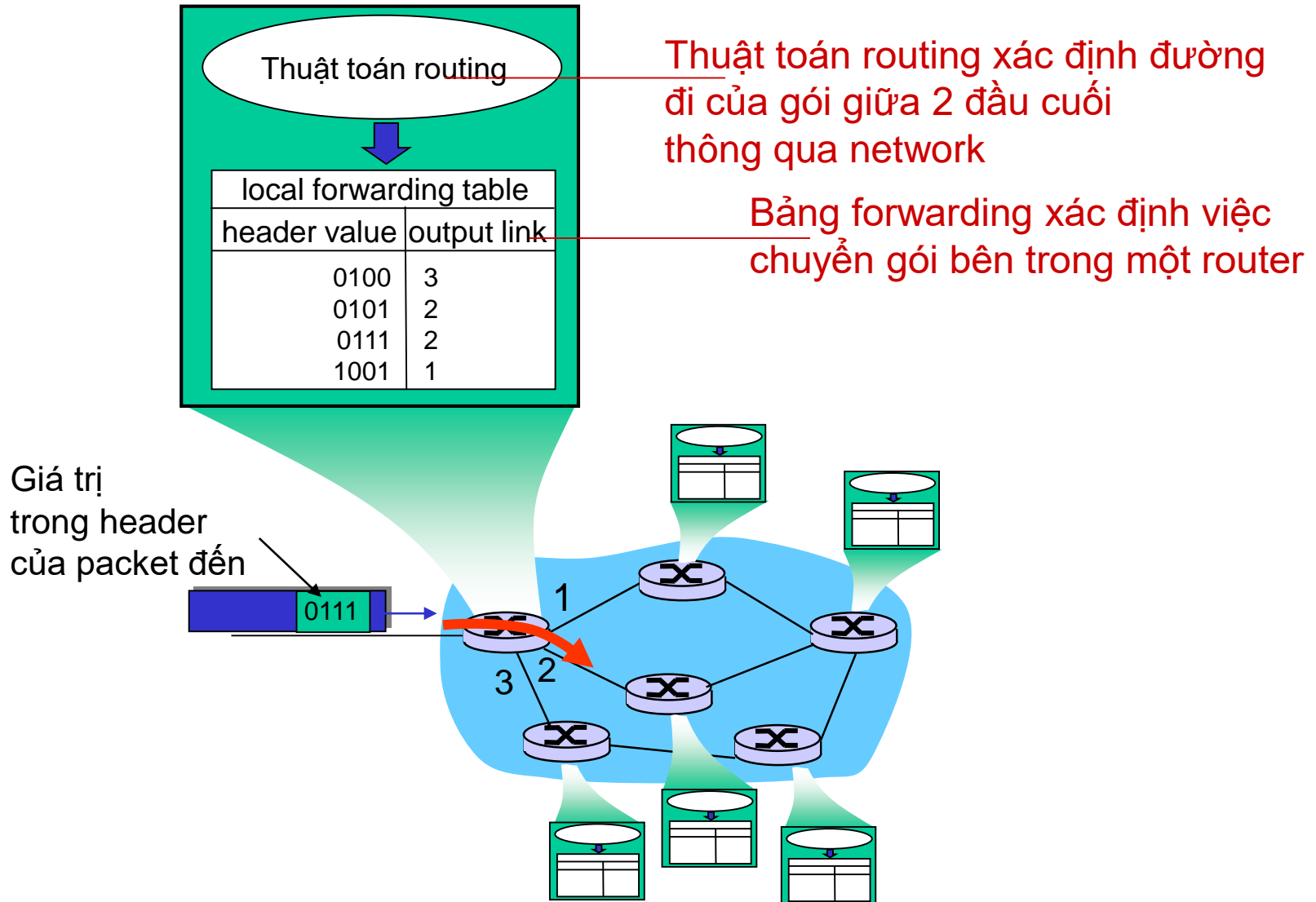
## 2 chức năng chính của tầng network

- ❖ *forwarding*: di chuyển các packet từ đầu vào đến đầu ra thích hợp của router
- ❖ *routing*: xác định đường đi cho các gói từ nguồn đến đích.
  - Các thuật toán định tuyến

### *Tương tự:*

- ❖ *routing*: tiến trình lập kế hoạch cho chuyển đi của packet từ nguồn tới đích
- ❖ *forwarding*: tiến trình vận chuyển qua 1 giao điểm

# Tác động qua lại giữa routing và forwarding



# Thiết lập kết nối

- ❖ Chức năng qua trọng thứ 3 trong một số kiến trúc mạng:
  - ATM, frame relay, X.25
- ❖ Trước khi các datagram di chuyển, 2 host đầu cuối và router trung gian (intervening routers) thiết lập kết nối ảo
  - Các router cũng tham gia
- ❖ Dịch vụ kết nối tầng transport so với tầng network :
  - **network**: giữa 2 hosts (cũng có thể bao gồm các router trung gian trong trường hợp kết nối ảo)
  - **transport**: giữa 2 tiến trình

# Mô hình dịch vụ Network

*Hỏi:* mô hình dịch vụ nào cho “kênh” truyền các datagram từ bên gửi đến bên nhận?

*Ví dụ các dịch vụ cho các datagram riêng biệt:*

- ❖ Giao nhận bảo đảm
- ❖ Giao nhận bảo đảm với độ trễ < 40ms

*Ví dụ các dịch vụ cho 1 luồng các datagram:*

- ❖ Giao nhận datagram theo thứ tự
- ❖ Băng thông được bảo đảm tối thiểu cho luồng
- ❖ Hạn chế các thay đổi trong khoảng trống giữa các packet



# Các mô hình dịch vụ tầng Network:

Kiến trúc Network	Mô hình dịch vụ	Bảo đảm?			Phản hồi tắt nghẽn	
		Băng thông	Mất	Thứ tự		Định thì
Internet	best effort	không	không	không	không	Không (inferred via loss)
ATM	CBR	Tốc độ không đổi	có	có	có	Không tắt nghẽn
ATM	VBR	Tốc độ bảo đảm	có	có	có	Không tắt nghẽn
ATM	ABR	Tốc độ bảo đảm	không	có	không	có
ATM	UBR	không	không	có	không	không

# Chương 4: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network (Mạng mạch ảo) và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

## 4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

## 4.6 routing trong Internet

- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing

# Dịch vụ connection (hướng kết nối) và connection-less (phi kết nối)

- ❖ Mạng *datagram* cung cấp dịch vụ *connectionless* tại tầng network
- ❖ Mạng *mạch ảo (virtual-circuit network)* cung cấp dịch vụ *connection* tại tầng network
- ❖ Tương tự như các dịch vụ kết nối định hướng và không định hướng của tầng transport, nhưng:
  - *Dịch vụ*: từ host này đến host kia (host-to-host)
  - *Không lựa chọn*: network chỉ cung cấp 1 dịch vụ
  - *Thực hiện*: trong mạng lõi

# Các mạch ảo (Virtual circuits)

“đường đi từ nguồn tới đích tương tự như mạng điện thoại (telephone circuit)”

- Hiệu quả
- Các hoạt động của mạng dọc theo đường đi từ nguồn tới đích

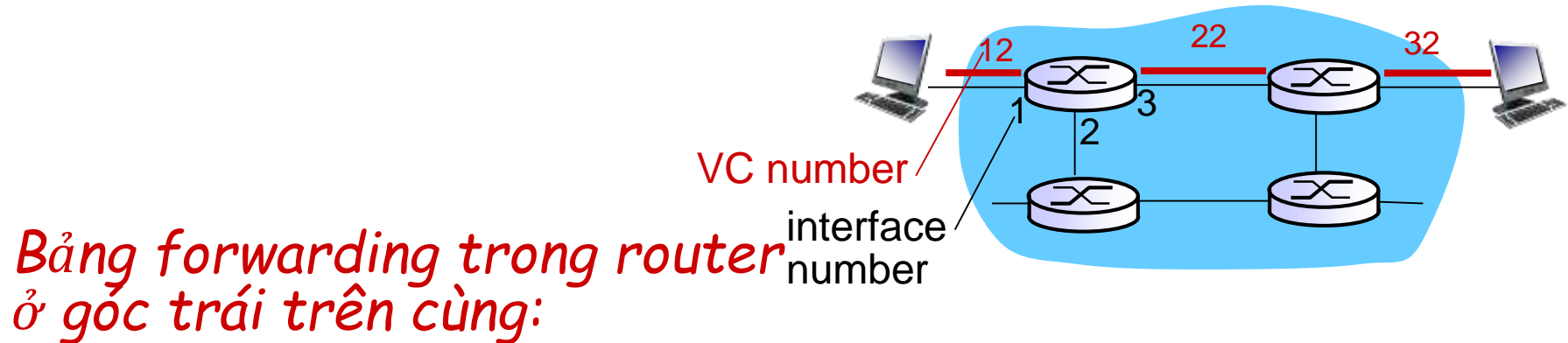
- ❖ Thiết lập cuộc gọi, chia nhỏ mỗi cuộc gọi trước khi dữ liệu có thể truyền
- ❖ Mỗi packet mang định dạng của kết nối ảo (VC identifier) (không phải là địa chỉ của host đích)
- ❖ Mỗi router trên đường đi từ nguồn tới đích duy trì trạng thái cho mỗi kết nối mà gói đi qua.
- ❖ Đường link, các tài nguyên router (bảng thông, bộ nhớ đệm) có thể được cấp phát cho kết nối ảo (các tài nguyên dành riêng= dịch vụ có thể dự đoán trước)

# Sự thực hiện kết nối ảo (VC)

*Một kết nối ảo bao gồm:*

1. *Đường đi (path)* từ nguồn tới đích
  2. *Số hiệu kết nối ảo (VC numbers)*, một số cho một kết nối dọc theo đường đi
  3. *Các mục trong các bảng forwarding* ở trong các router dọc theo đường đi
- ❖ packet thuộc về kết nối ảo mang số hiệu (chứ không phải là điểm đến)
  - ❖ Số hiệu của kết nối ảo có thể được thay đổi trên mỗi kết nối.
    - Số hiệu mới của kết nối ảo được cấp phát từ bảng forwarding

# Bảng forwarding của kết nối ảo

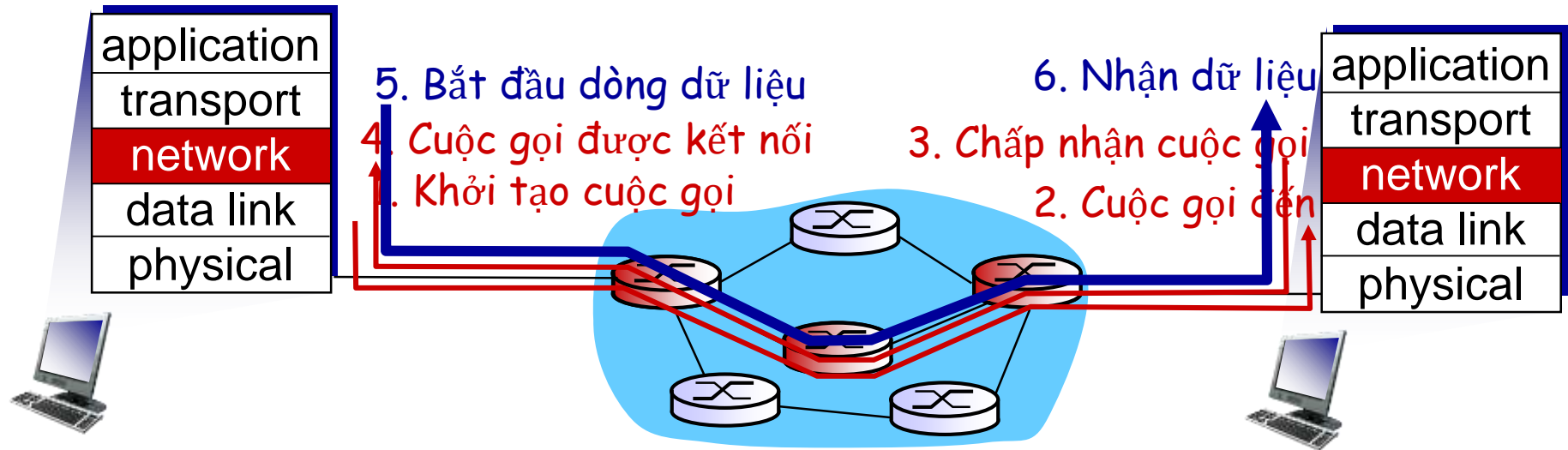


Cổng vào	số hiệu của kết nối ảo vào	Cổng ra	số hiệu của kết nối ảo ra
1	12	3	22
2	63	1	18
3	7	2	17
1	97	3	87
...	...	...	...

*Các router kết nối ảo duy trì thông tin trạng thái kết nối!*

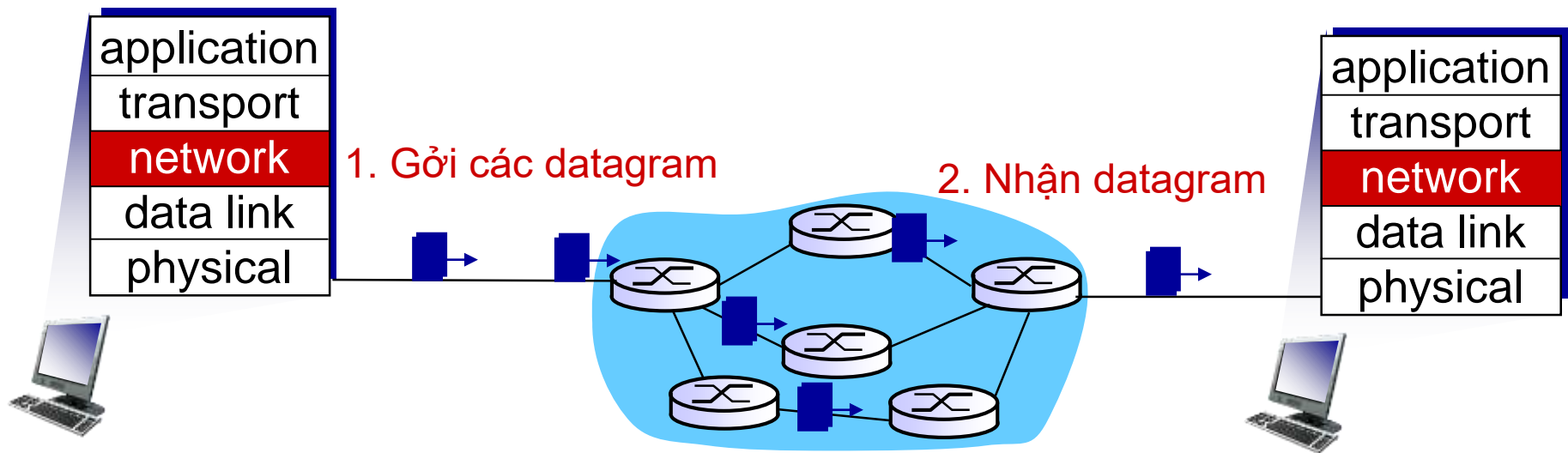
# Các mạch ảo: các giao thức gửi tín hiệu

- ❖ Được dùng để thiết lập, duy trì kết nối ảo
- ❖ Được dùng trong ATM, frame-relay, X.25
- ❖ Không được sử dụng trong Internet ngày nay



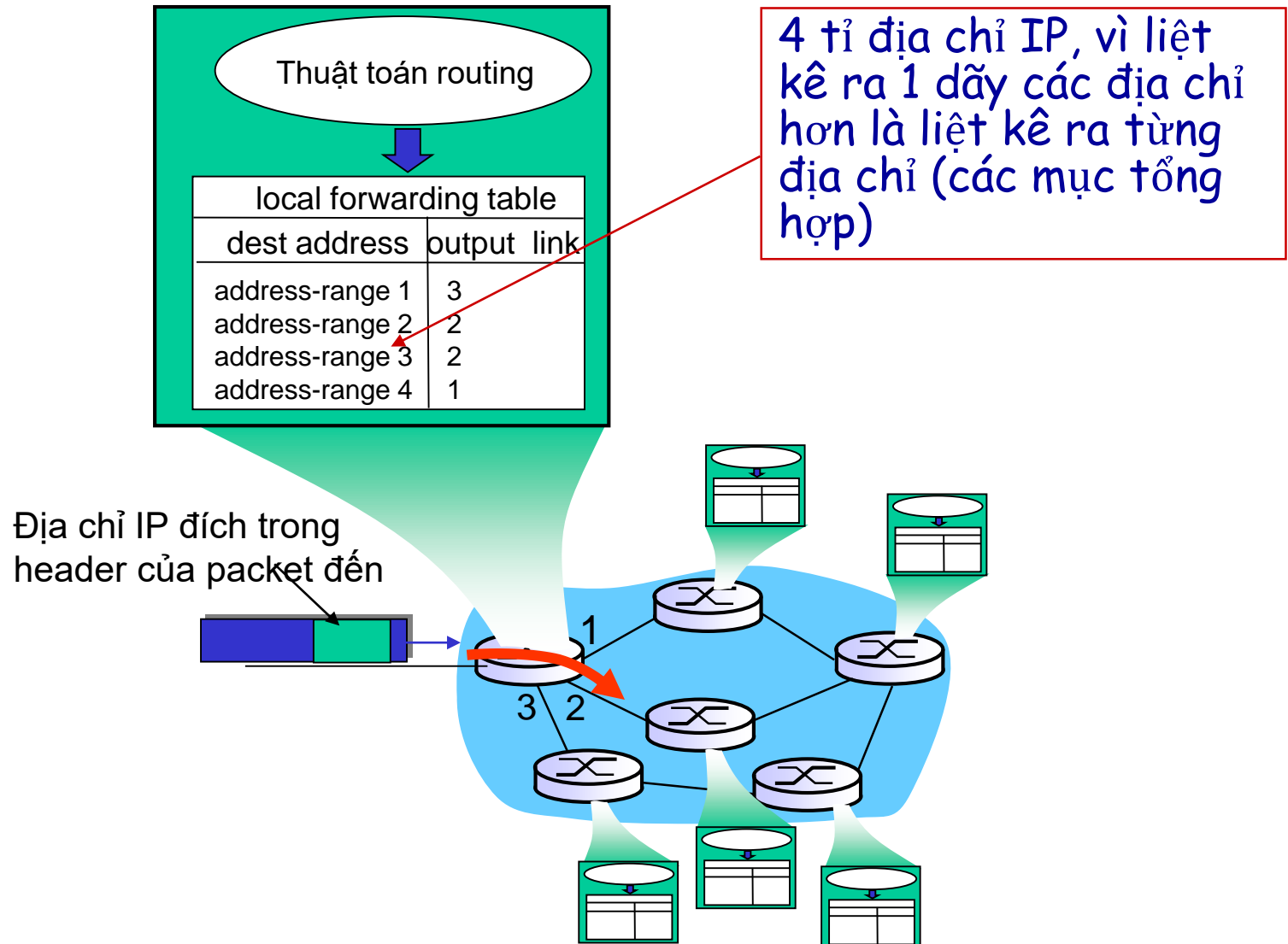
# Mạng Datagram

- ❖ Không thiết lập cuộc gọi tại tầng network
- ❖ Các router: không có trạng thái về các kết nối giữa 2 điểm cuối
  - Không có khái niệm về mức network của “kết nối”
- ❖ Các packet được chuyển dùng địa chỉ của host đích





# Bảng Datagram forwarding



# Bảng Datagram forwarding

Dãy địa chỉ đích	Link Interface
11001000 00010111 00010000 00000000 đến 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 đến 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 đến 11001000 00010111 00011111 11111111	2
khác	3

**Q:** nhưng cái gì sẽ xảy ra nếu các dãy địa chỉ này không được chia hợp lý?

# So trùng prefix dài nhất

## *So trùng prefix dài nhất*

Khi tìm kiếm 1 mục trong bảng forwarding table cho địa chỉ đích, dùng prefix dài nhất của địa chỉ trùng với địa chỉ đích.

Dãy địa chỉ đích	Link interface
11001000 00010111 00010*** *****	0
11001000 00010111 00011000 *****	1
11001000 00010111 00011*** *****	2
otherwise	3

Ví dụ:

DA: 11001000 00010111 00010110 10100001

Interface nào?

DA: 11001000 00010111 00011000 10101010

Interface nào?

# Mạng datagram hoặc mạch ảo: tại sao?

---

## *Internet (datagram)*

- ❖ Dữ liệu trao đổi giữa các máy tính
  - Dịch vụ “mềm dẻo, không định thì chặt chẽ.
- ❖ Nhiều kiểu link
  - Các đặc tính khác nhau
  - Khó đồng nhất dịch vụ
- ❖ Các hệ thống đầu cuối “thông minh” (các máy tính)
  - Có thể thích ứng, điều khiển và sửa lỗi
  - *Mạng bên trong đơn giản, phức tạp tại “mạng bên ngoài”*

## *ATM (mạch ảo)*

- ❖ Được phát triển từ hệ thống điện thoại
- ❖ Đàm thoại của con người:
  - Định thì chặt chẽ, yêu cầu về độ tin cậy
  - Cần cho các dịch vụ bảo đảm
- ❖ Các hệ thống đầu cuối “ít thông minh”
  - Điện thoại
  - *Bên trong mạng phức tạp*

# Chương 4: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network (mạng mạch ảo) và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

## 4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

## 4.6 routing trong Internet

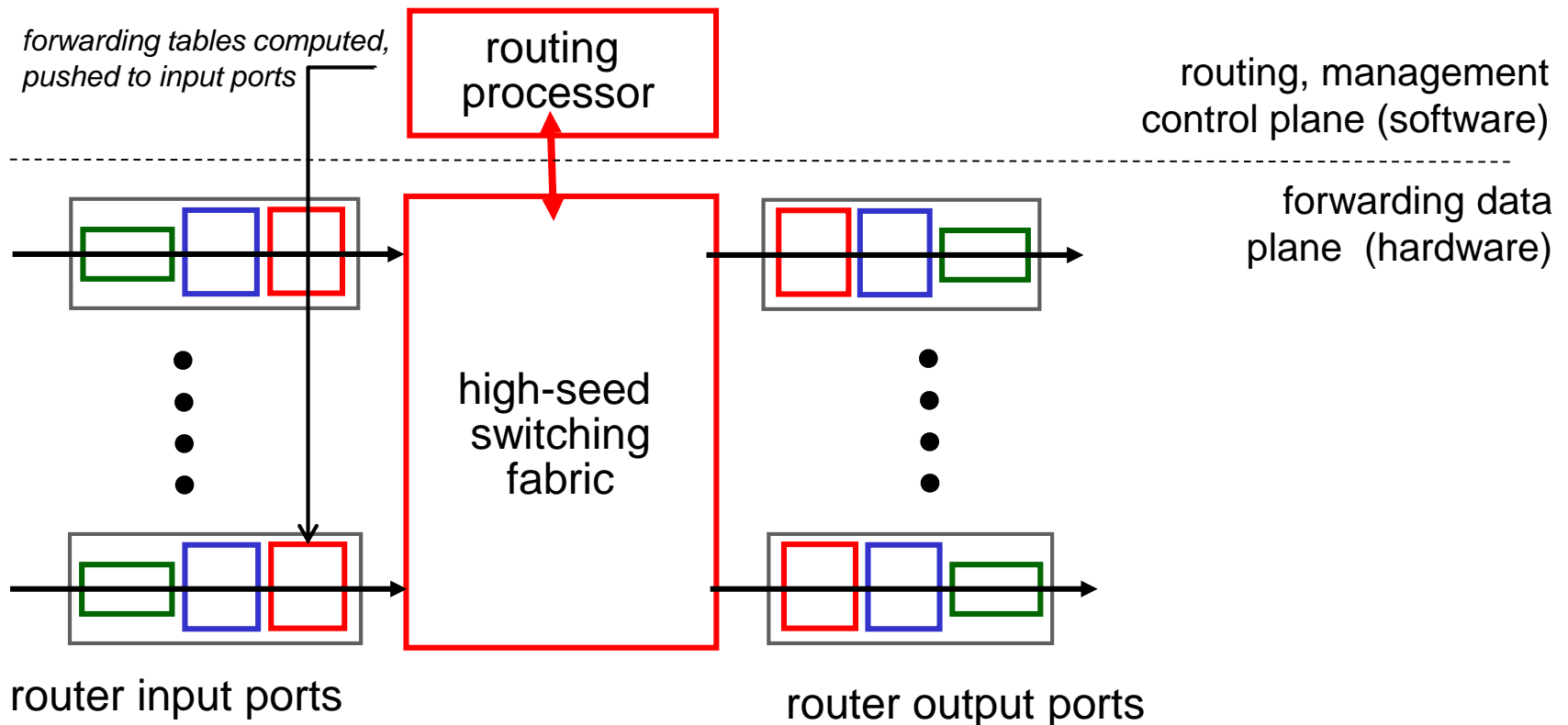
- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing

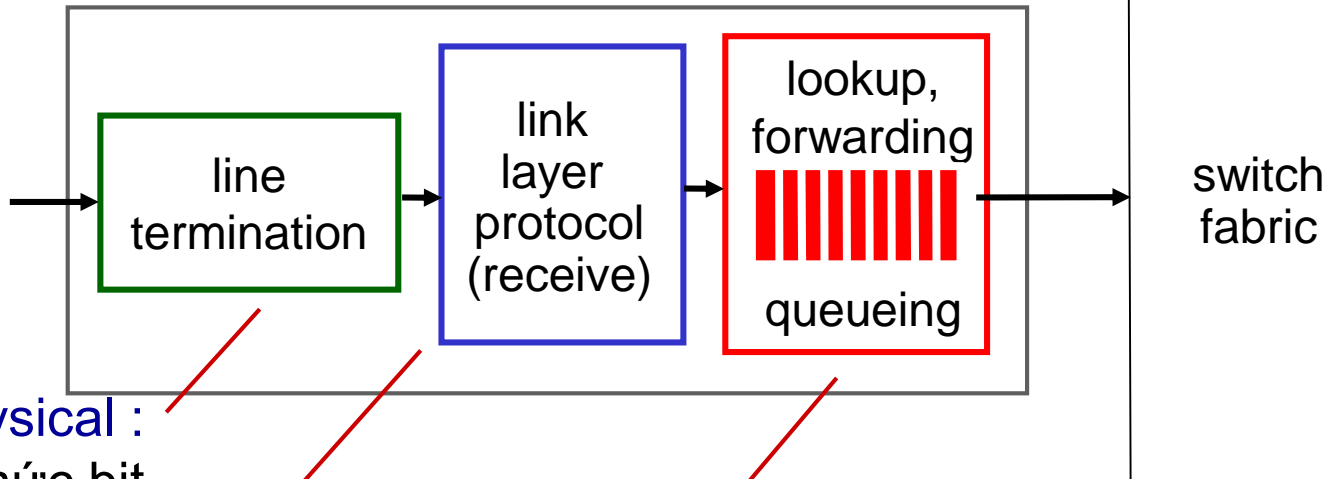
# Tổng quan kiến trúc Router

2 chức năng chính của router:

- ❖ Chạy các giao thức/thuật toán routing (RIP, OSPF, BGP)
- ❖ *Chuyển tiếp* các datagram từ đường link vào tới đường link ra



# Các chức năng của cổng Input



Tầng physical :  
Tiếp nhận mức bit

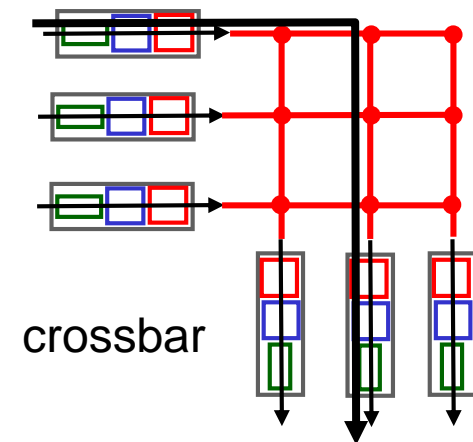
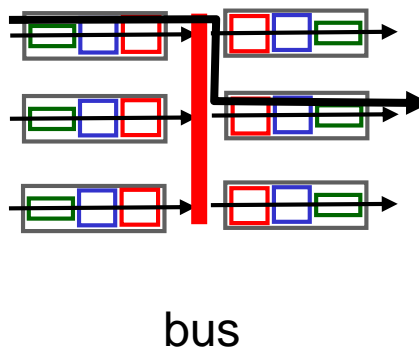
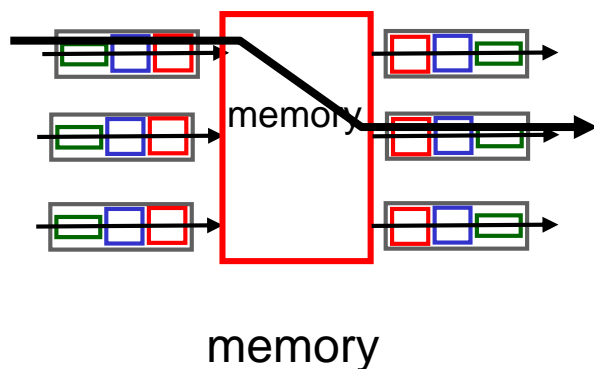
Tầng data link:  
Như là Ethernet  
Xem chương 5

## decentralized switching:

- ❖ Với đích của datagram biết trước., tìm output port (cổng ra) bằng cách dùng bảng forwarding trong bộ nhớ của input port (cổng vào) ("*match plus action*")
- ❖ Mục tiêu: hoàn tất xử lý input port tại "tốc độ dòng" ('line speed')
- ❖ Xếp hàng: nếu datagram đến nhanh hơn tốc độ chuyển tiếp bên trong switch

# Switching fabrics

- ❖ Truyền packet từ bộ nhớ đệm đầu vào đến bộ nhớ đệm đầu ra thích hợp
- ❖ Tốc độ switching: tốc độ mà các packet có thể được truyền từ đầu vào (inputs) đến đầu ra (outputs)
  - Thường được đo như nhiều tốc độ dòng của đầu vào/đầu ra
- ❖ 3 kiểu switching fabrics

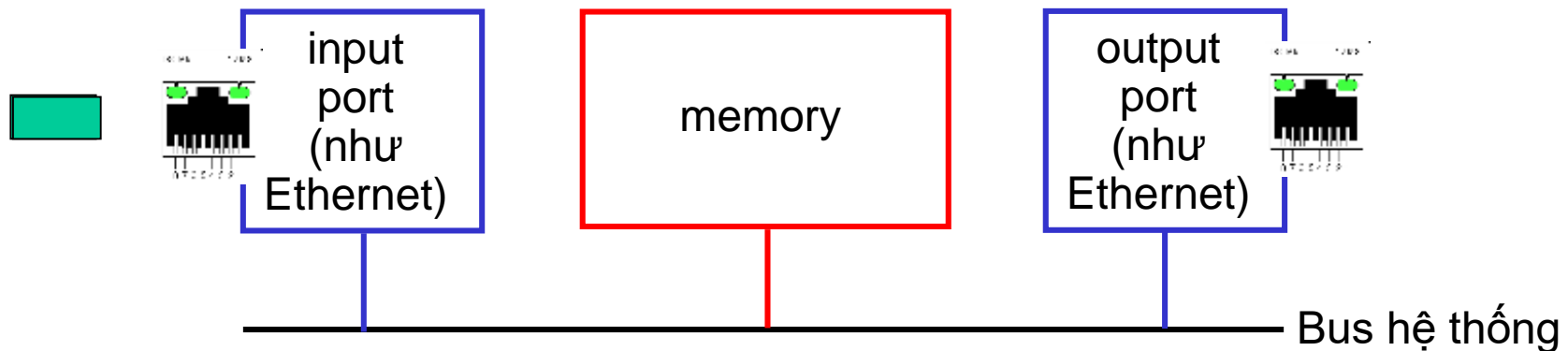




# Switching thông qua bộ nhớ (memory)

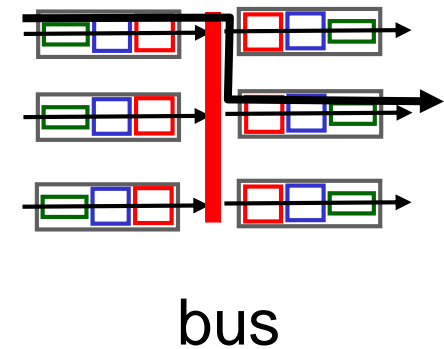
## *Các router thế hệ đầu tiên:*

- ❖ Các máy tính cổ điển với switching dưới sự điều khiển của CPU
- ❖ packet được sao chép đến bộ nhớ của hệ thống
- ❖ Tốc độ bị giới hạn bởi băng thông của bộ nhớ (2 bus qua mỗi datagram)



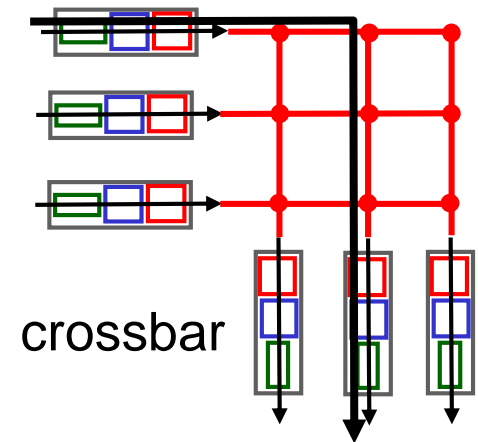
# Switching thông qua 1 bus

- ❖ datagram từ bộ nhớ của port vào đến bộ nhớ của port ra thông qua một bus được chia sẻ
- ❖ *Sự tranh chấp bus*: tốc độ switching bị giới hạn bởi băng thông của bus
- ❖ 32 Gbps bus, Cisco 5600: tốc độ đủ cho truy cập và các enterprise router

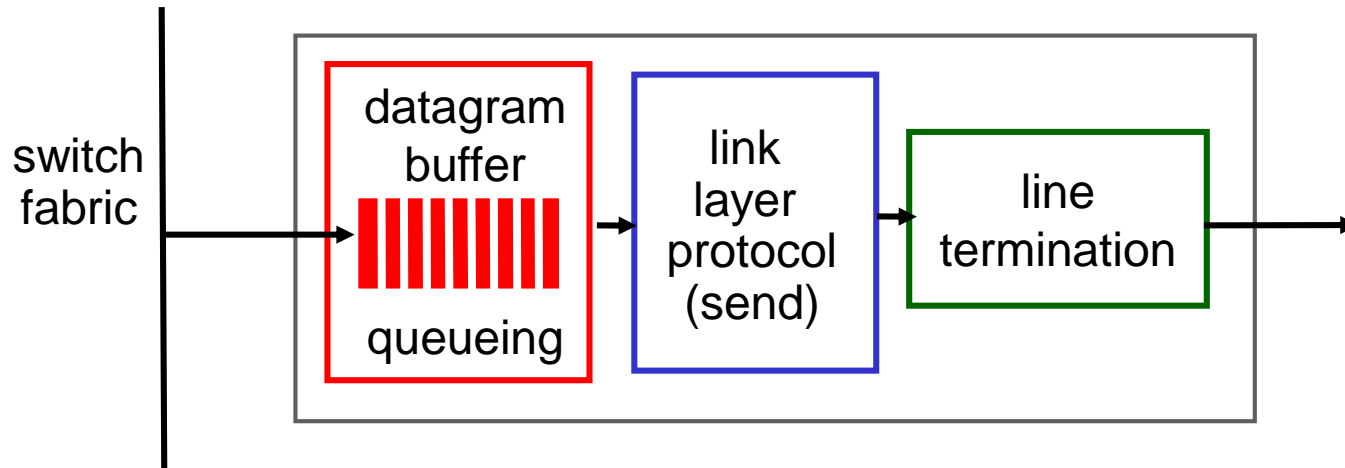


# Switching thông qua interconnection network

- ❖ Vượt qua các giới hạn của băng thông bus
- ❖ Đa mạng, crossbar, các mạng kết nối nội bộ khác lúc đầu được phát triển để kết nối các bộ vi xử lý trong bộ đa xử lý
- ❖ Thiết kế được nâng cao: phân mảnh datagram vào các ô có độ dài cố định, chuyển các ô thông qua fabric.
- ❖ Cisco 12000: chuyển 60 Gbps thông qua interconnection network

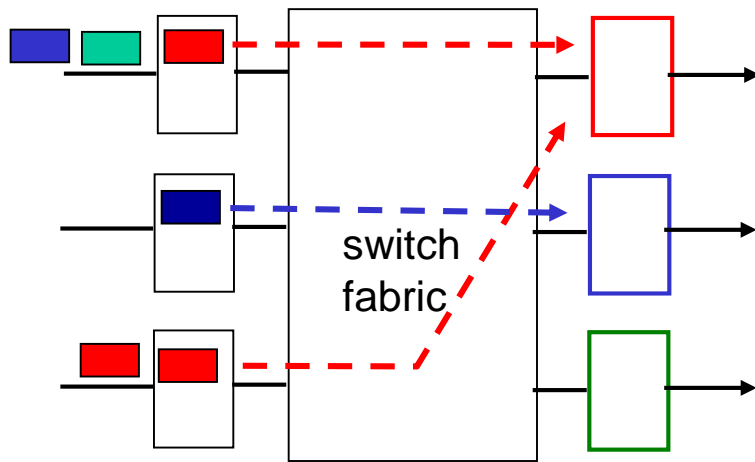


# Các cổng ra (Output)

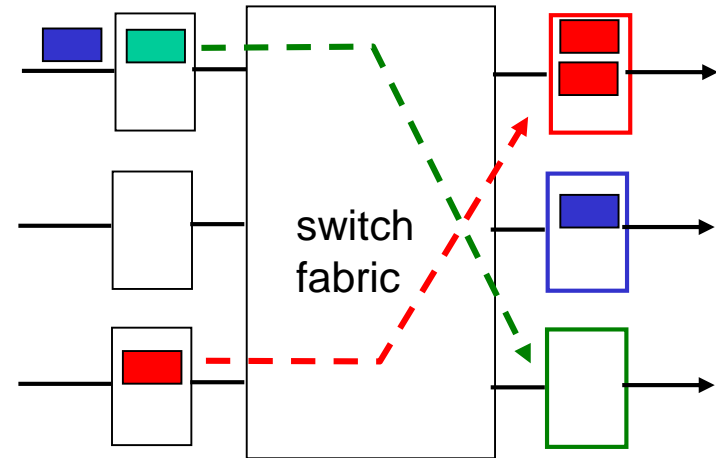


- ❖ *Đệm (buffering)* được yêu cầu khi các datagram đến từ fabric nhanh hơn tốc độ truyền
- ❖ *scheduling discipline* chọn trong số các datagram xếp hàng để truyền

# Sắp hàng tại cổng ra



at  $t$ , các packet nhiều hơn  
từ đầu vào đến đầu ra



one packet time later

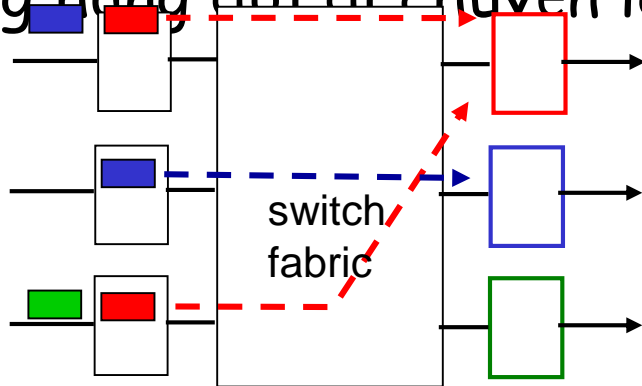
- ❖ Đệm khi tốc độ đến qua switch vượt quá tốc độ dòng ra (output line)
- ❖ *Sắp hàng (trễ) và mất gói vì bộ nhớ đệm tại cổng ra bị tràn (overflow)!*

# Bao nhiêu đệm?

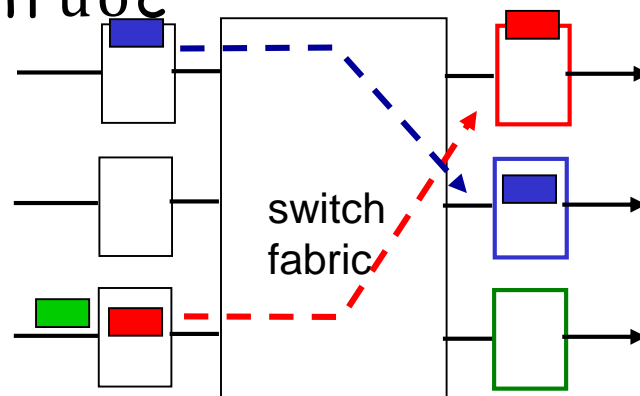
- ❖ RFC 3439 quy tắc ngón tay cái: đệm trung bình bằng với thời gian RTT “điển hình” (250 msec) nhân với dung lượng đường link  $C$ 
  - Ví dụ  $C = 10$  Gpbs link: 2.5 Gbit buffer
- ❖ Khuyến nghị gần đây: với  $N$  luồng, đệm bằng với
$$\frac{RTT \cdot C}{\sqrt{N}}$$

# Sắp hàng tại cổng vào

- ❖ fabric chậm hơn sự phối hợp của các cổng vào -> sắp hàng có thể xảy ra tại các hàng vào
  - *Sắp hàng chậm trễ vào mất gói do tràn bộ đệm đầu vào!*
- ❖ *Head-of-the-Line (HOL) blocking*: datagram được sắp hàng tại phía trước hàng đợi ngăn cản các datagram khác trong hàng đợi di chuyển lên trước



Sự cạnh tranh tại cổng ra:  
Chỉ có một datagram màu đỏ  
có thể được truyền.  
*packet màu đỏ thấp hơn bị  
chặn lại*



one packet time  
later: packet màu  
xanh lá trải qua  
HOL blocking

# Chương 4: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

## 4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

## 4.6 routing trong Internet

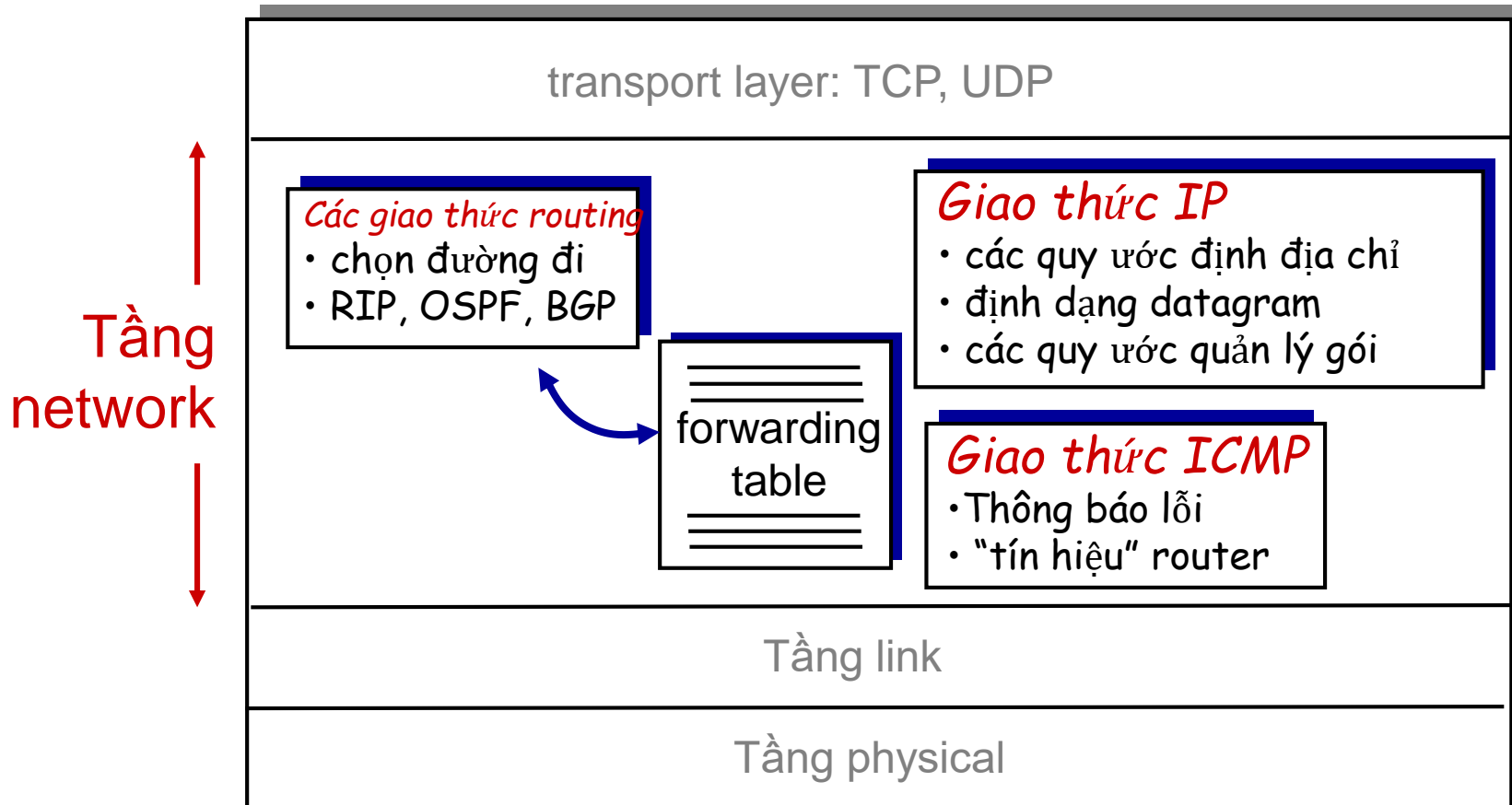
- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing



# Tầng Internet network

Các chức năng tầng network của host và router:



# Định dạng IP datagram

Số hiệu phiên bản  
giao thức IP

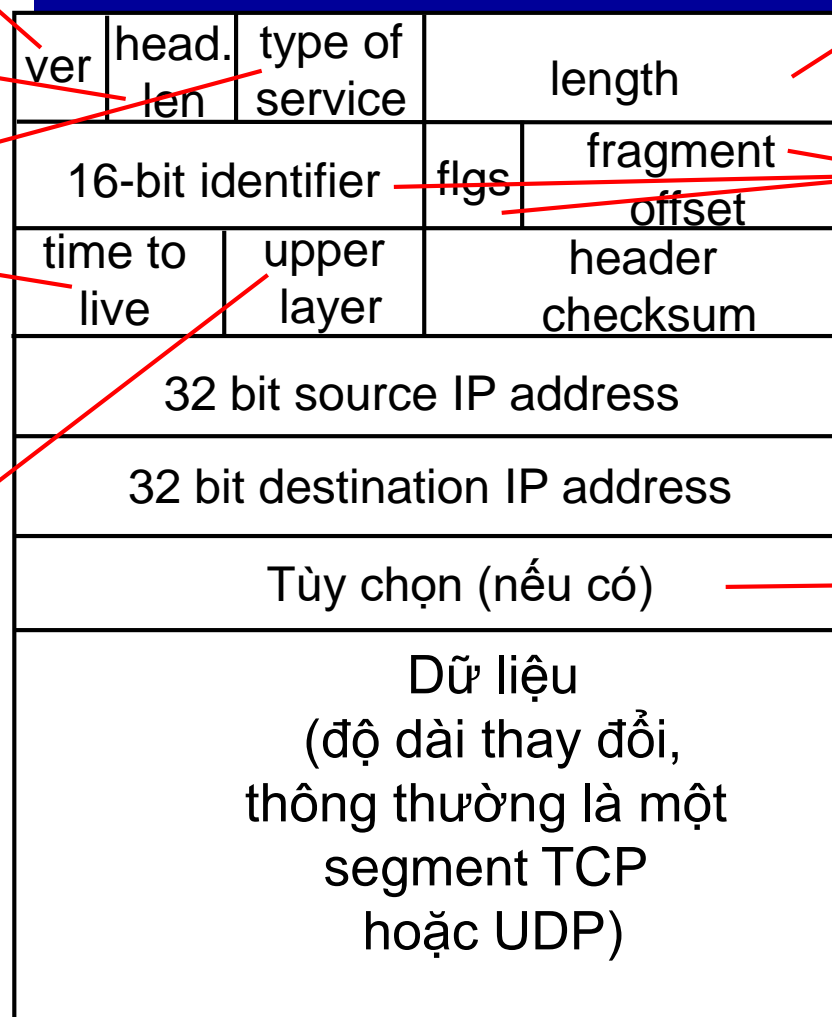
Độ dài header  
(bytes)

“kiểu” dữ liệu

Số hop còn lại  
tối đa (giảm xuống  
tại mỗi router)

Giao thức lớp trên  
để đưa payload đến

32 bits



Tổng độ dài  
datagram(byte)

Dành cho  
phân mảnh/  
tổng hợp

Ví dụ: trường  
timestamp ghi lại  
đường đi, danh  
các router đi đến

## *how much overhead?*

- ❖ 20 bytes of TCP
- ❖ 20 bytes of IP
- ❖ = 40 bytes + app layer overhead

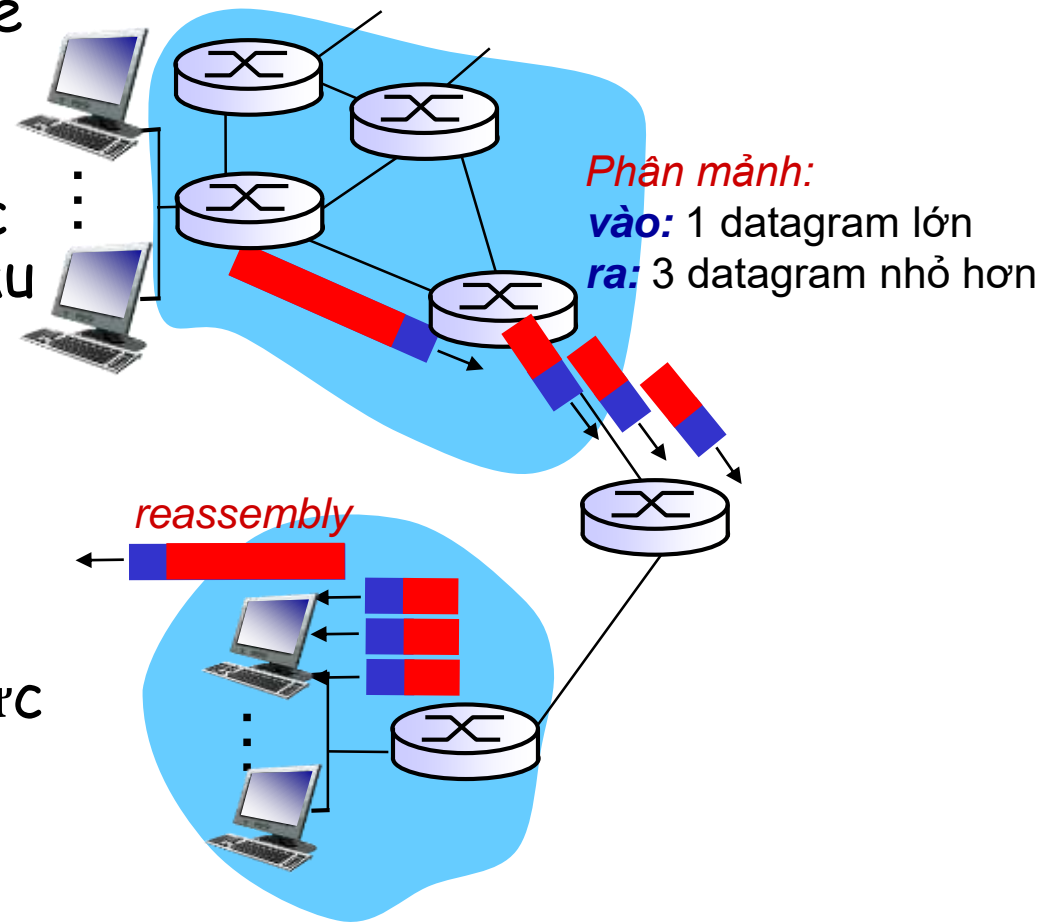
# Phân mảnh và tổng hợp IP

- ❖ Các đường link mạng có MTU (max.transfer size) - frame lớn nhất có thể ở mức kết nối

- Các kiểu đường link khác nhau, các MTU khác nhau

- ❖ IP datagram lớn được chia (“fragmented”) bên trong mạng

- 1 datagram thành 1 vài datagram
- “tổng hợp” chỉ được thực hiện ở đích cuối cùng
- Các bit của IP header được sử dụng để xác định, sắp đặt các fragment liên quan



# Phân mảnh và tổng hợp IP

*Ví dụ:*

- ❖ 4000 byte datagram
- ❖ MTU = 1500 bytes

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

*1 datagram lớn thành vài datagram nhỏ hơn*

1480 bytes  
trong trường 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: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

## 4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

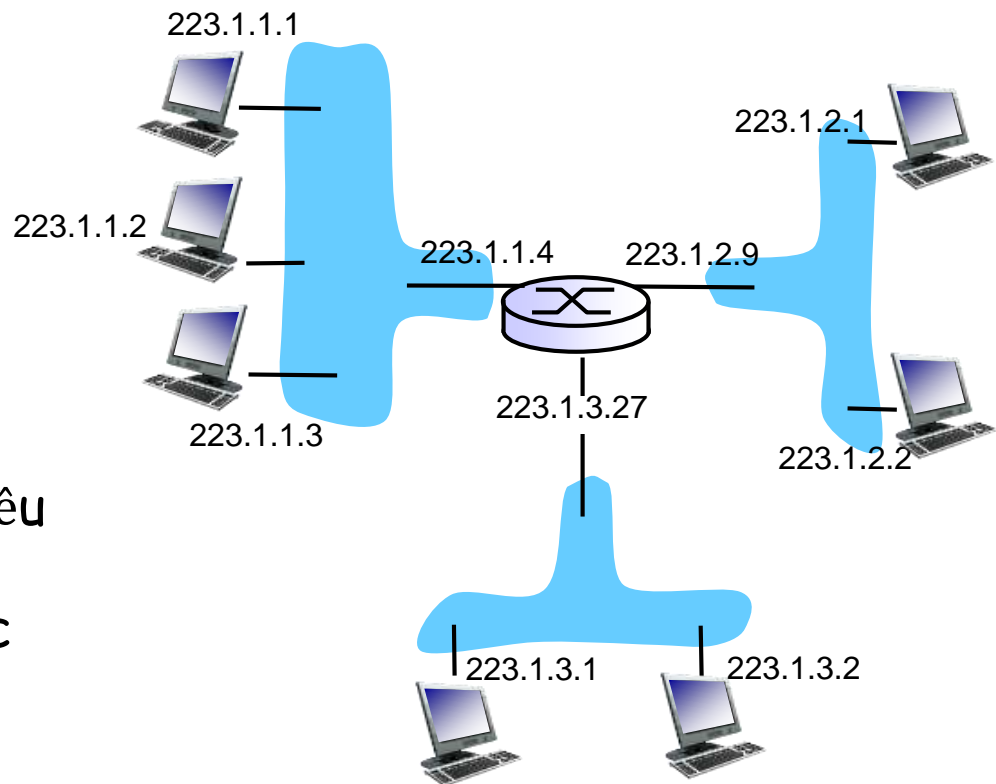
## 4.6 routing trong Internet

- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing

# Định địa chỉ IP: giới thiệu

- ❖ **Địa chỉ IP:** 32-bit nhận dạng cho host, router *interface*
- ❖ **interface:** kết nối giữa host/router và đường link vật lý
  - Router thường có nhiều interface
  - host thường có 1 hoặc 2 interface (ví dụ wired Ethernet, wireless 802.11)
- ❖ **Mỗi địa chỉ IP được liên kết với mỗi interface**



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

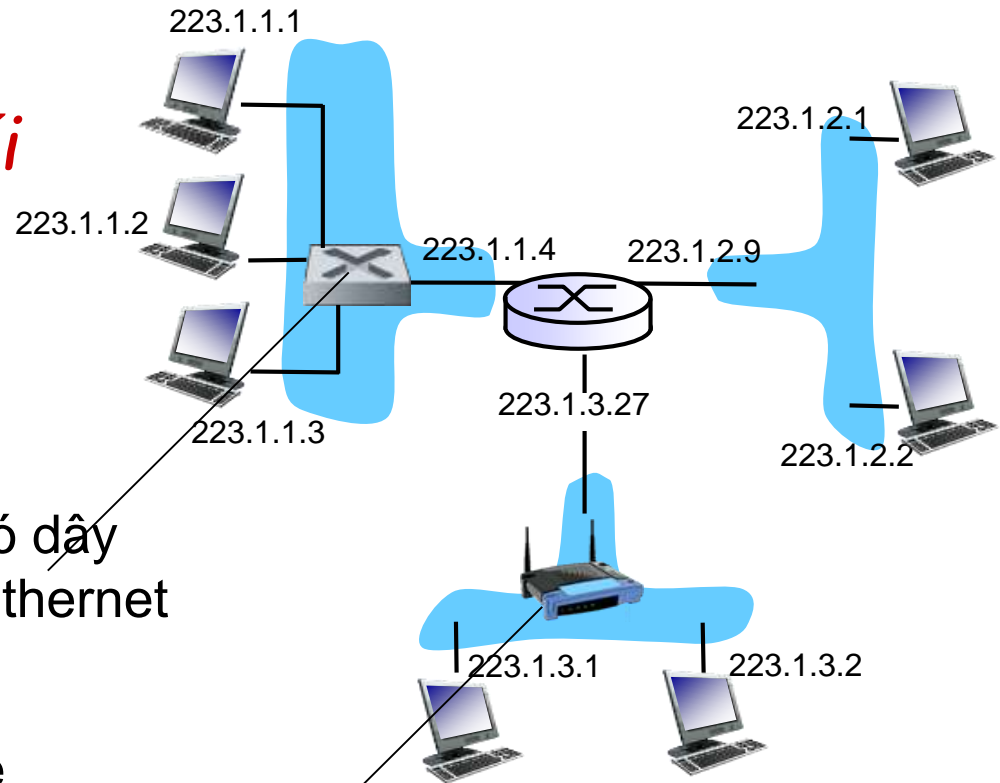
# Định địa chỉ IP: giới thiệu

*Hỏi: các interface  
thật sự được kết nối  
như thế nào?*

*đáp: sẽ tìm hiểu  
trong chương 5, 6.*

*Đáp:* các interface Ethernet có dây  
được kết nối bởi các switch Ethernet

*Bây giờ:* không cần lo lắng về  
cách mà 1 interface được kết  
nối với một interface khác  
(không có router trung gian))



*Đáp:* interface WiFi không dây được  
kết nối thông qua WiFi base station

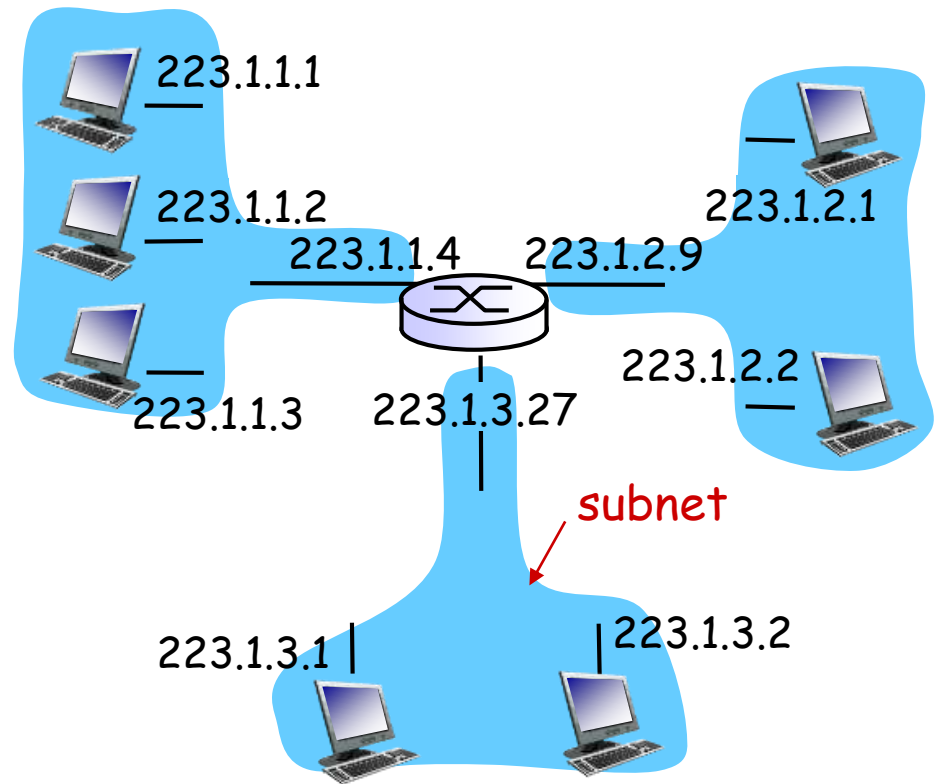
# Các Subnet (mạng con)

## ❖ Đại chỉ IP:

- Phần subnet - các bit có trọng số cao
- Phần host - các bit có trọng số thấp

## ❖ subnet là gì?

- Các interface của thiết bị có phần subnet của địa chỉ IP giống nhau
- Có thể giao tiếp vật lý với nhau mà không cần *router trung gian* can thiệp



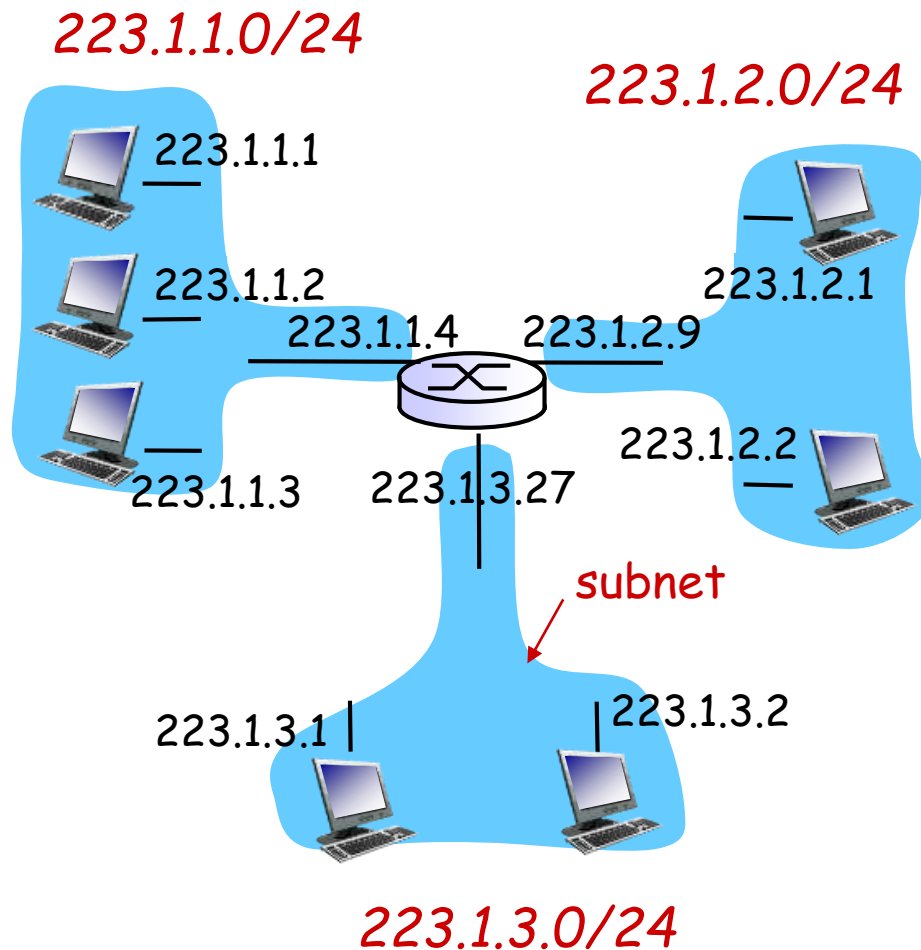
Mạng gồm 3 subnet



# Các mạng con (subnet)

## Phương pháp

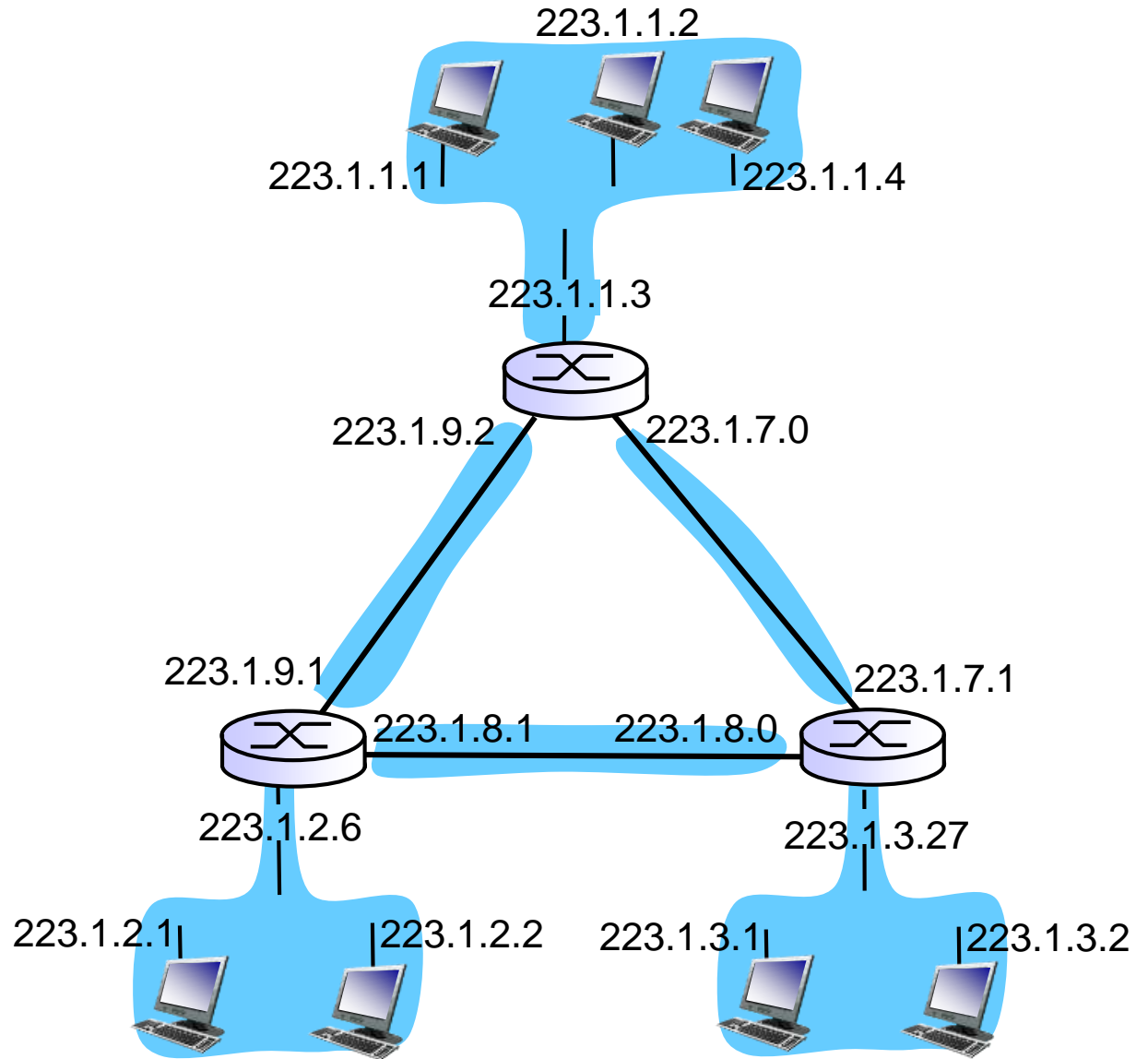
- ❖ Để xác định các subnet, tách mỗi interface từ host hoặc router của nó, tạo vùng các mạng độc lập
- ❖ Mỗi mạng độc lập được gọi là một *subnet*



subnet mask: /24

# Subnets

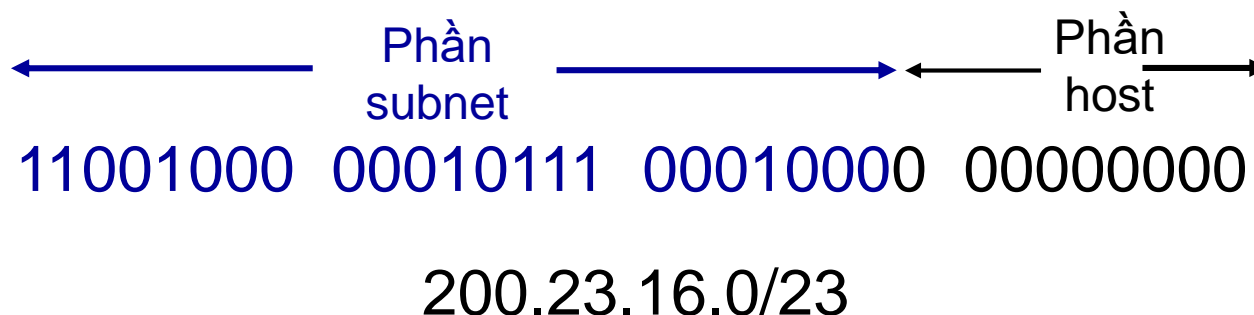
Có bao nhiêu?



# Định địa chỉ IP: CIDR

## CIDR: Classless InterDomain Routing

- Phần subnet của địa chỉ có độ dài bất kỳ
- Định dạng địa chỉ: **a.b.c.d/x**, trong đó x là số các bits trong phần subnet của địa chỉ



# Địa chỉ IP: làm sao để lấy một địa chỉ?

---

**Hỏi:** làm thế nào một host lấy được địa chỉ IP?

- ❖ Mã hóa cứng bởi người quản trị hệ thống trong 1 file
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- ❖ **DHCP: Dynamic Host Configuration Protocol:** tự động lấy địa chỉ IP từ server
  - “plug-and-play”

# DHCP: Dynamic Host Configuration Protocol

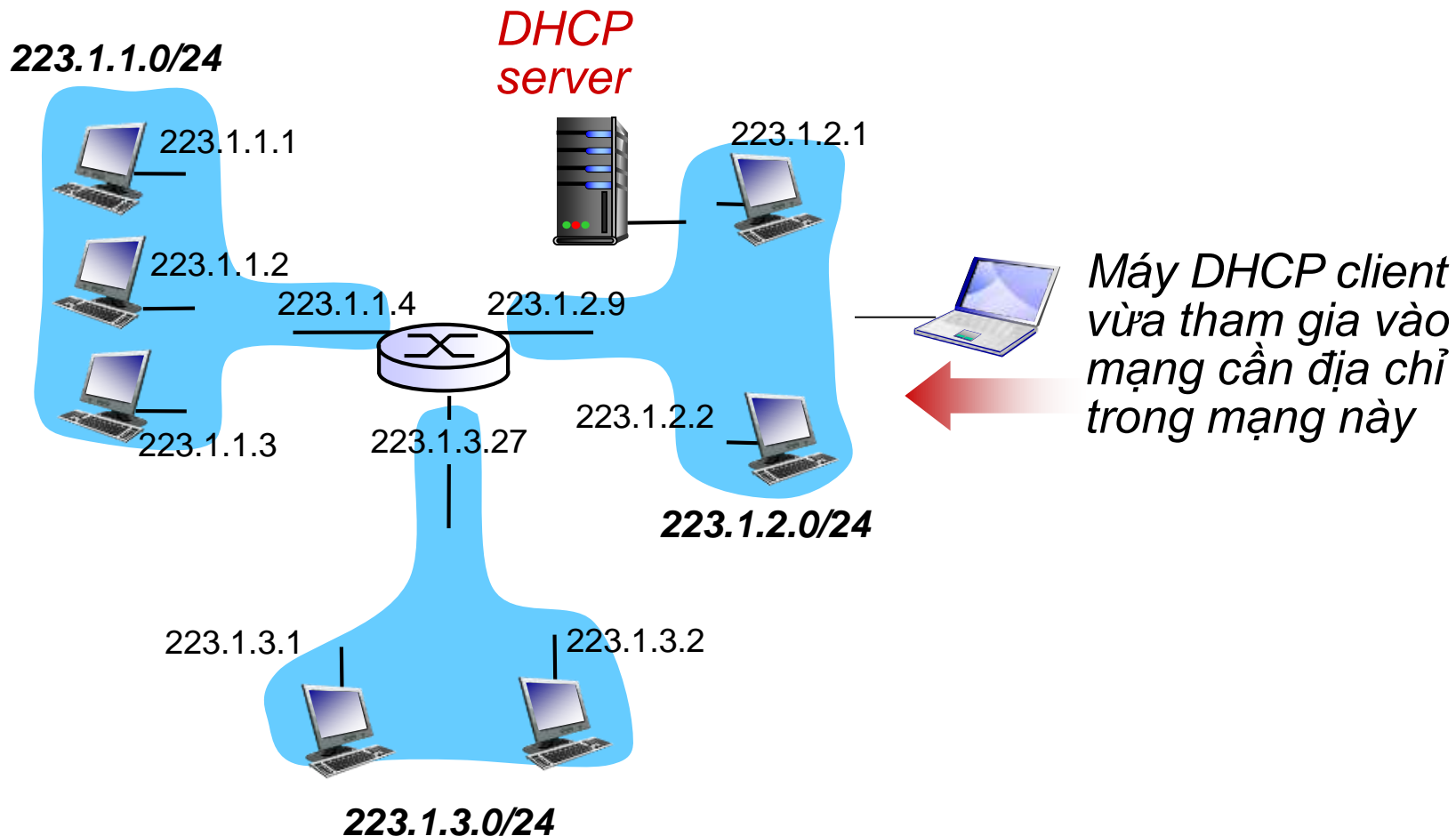
**Mục tiêu:** cho phép host tự động lấy địa chỉ IP của nó từ server trong mạng khi host đó tham gia vào mạng

- Có thể gán hạn địa chỉ IP mà host đó vừa được cấp
- Cho phép tái sử dụng các địa chỉ IP (chỉ giữ địa chỉ trong khi được kết nối/"on")
- Hỗ trợ cho người dùng di động muốn tham gia vào mạng (trong thời gian ngắn)

## **Tổng quan DHCP :**

- host quảng bá (broadcasts) thông điệp “**DHCP discover**” [tùy chọn]
- DHCP server đáp ứng bằng thông điệp “**DHCP offer**” [tùy chọn]
- host yêu cầu địa chỉ IP: “**DHCP request**” msg
- DHCP server gởi địa chỉ: “**DHCP ack**” msg

# Ngữ cảnh DHCP client-server



# Ngũ cảnh DHCP client-server

DHCP server: 223.1.2.5

DHCP discover

src : 0.0.0.0, 68  
dest.: 255.255.255.255, 67  
yiaddr: 0.0.0.0  
transaction ID: 654

arriving  
client



DHCP offer

src: 223.1.2.5, 67  
dest: 255.255.255.255, 68  
yiaddr: 223.1.2.4  
transaction ID: 654  
lifetime: 3600 secs

DHCP request

src: 0.0.0.0, 68  
dest:: 255.255.255.255, 67  
yiaddr: 223.1.2.4  
transaction ID: 655  
lifetime: 3600 secs

DHCP ACK

src: 223.1.2.5, 67  
dest: 255.255.255.255, 68  
yiaddr: 223.1.2.4  
transaction ID: 655  
lifetime: 3600 secs

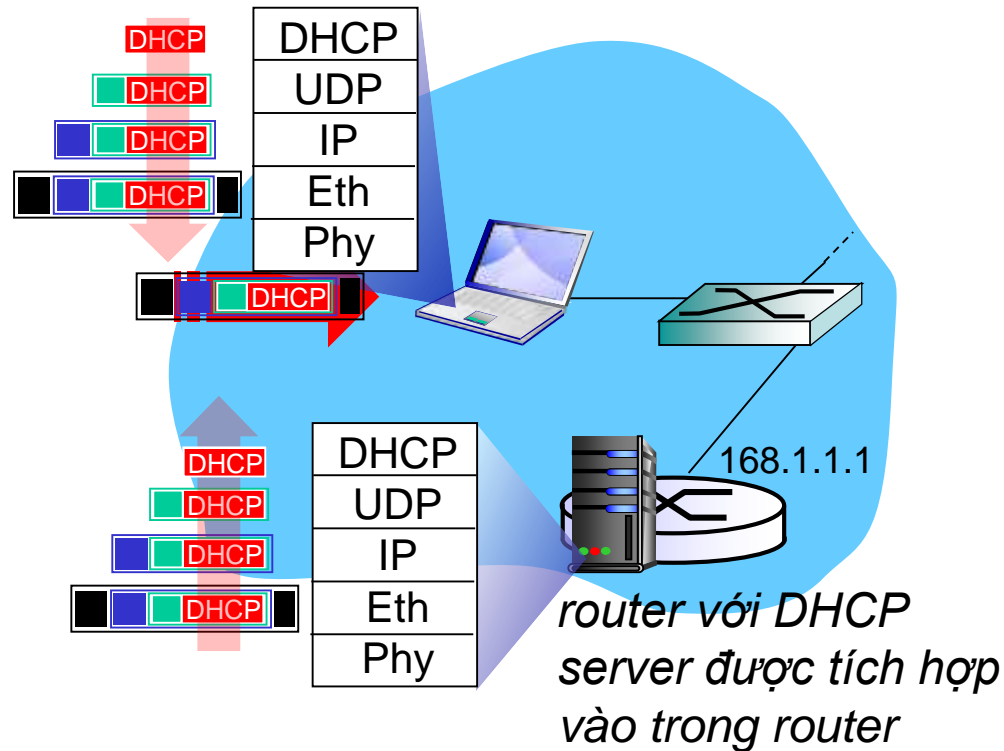
# DHCP: nhiều thông tin

DHCP không chỉ trả về địa chỉ IP được chỉ định trên subnet, mà nó còn có thể trả về nhiều thông tin như sau:

- Địa chỉ của router first-hop của client
- Tên và địa chỉ IP của DNS sever
- network mask (cho biết phần của mạng so với phần phần host của địa chỉ IP)

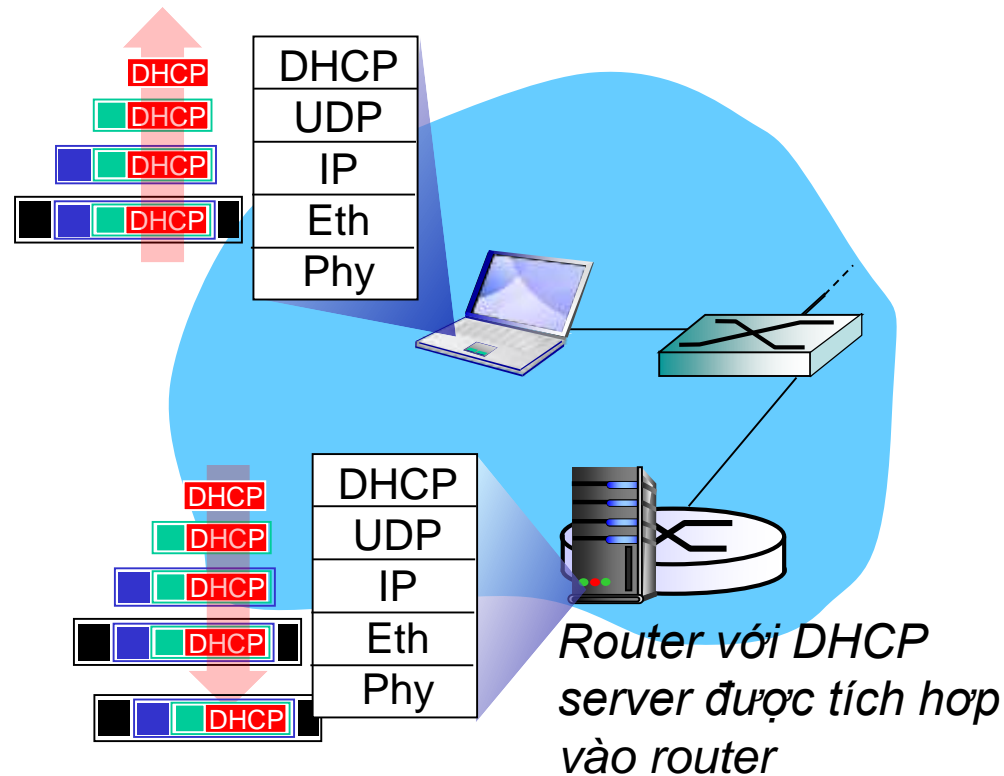


# DHCP: ví dụ



- ❖ laptop tham gia vào mạng cần địa chỉ IP của nó, địa chỉ của router first-hop, địa chỉ của : dùng DHCP
- ❖ DHCP request được đóng gói trong UDP, được đóng gói trong IP, được đóng gói trong 802.1 Ethernet
- ❖ Ethernet frame broadcast (dest: FFFFFFFFFFFFFFFF) trên LAN, được nhận tại router đang chạy DHCP server
- ❖ Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

# DHCP: ví dụ



- ❖ DHCP server lập địa chỉ IP của client chứa DHCP ACK, địa chỉ IP của router first-hop cho client, tên và địa chỉ IP của DNS server
- ❖ Đóng gói của DHCP server, frame được chuyển cho client, tách DHCP tại client
- ❖ Bây giờ, client biết địa chỉ IP của nó, tên và địa chỉ IP của DNS server, địa chỉ IP của router first-hop 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

.....

Yêu cầu

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."**

Trả lời

# Địa chỉ IP: làm sao để lấy được 1 địa chỉ IP?

**Hỏi:** làm sao mạng lấy được phần subnet của địa chỉ IP?

**Đáp:** lấy phần đã được cấp phát của không gian địa chỉ IP do ISP cung cấp

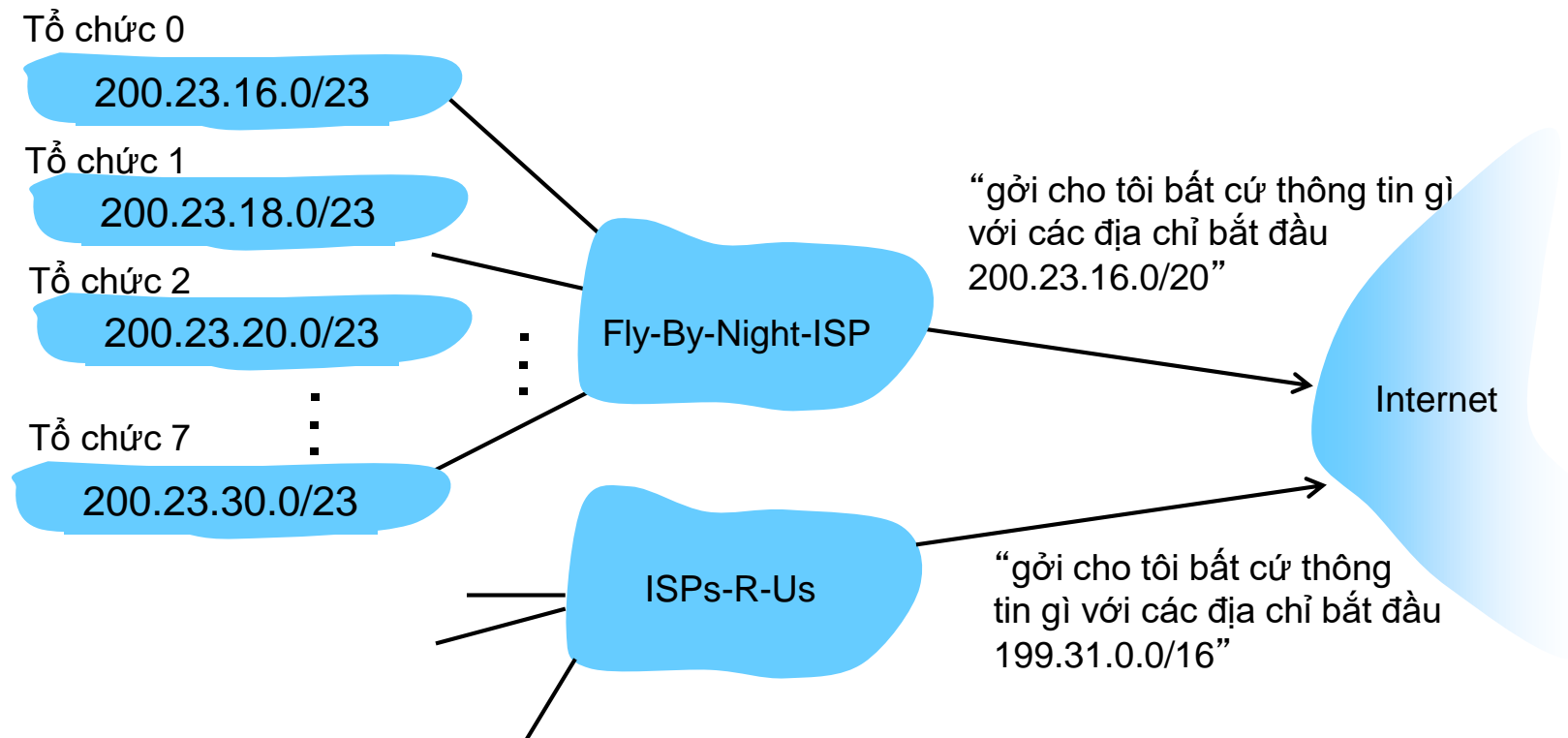
Dãy địa chỉ của ISP  
200.23.16.0/20

11001000 00010111 00010000 00000000

Tổ chức 0	<u>11001000 00010111 00010000</u>	00000000	200.23.16.0/23
Tổ chức 1	<u>11001000 00010111 00010010</u>	00000000	200.23.18.0/23
Tổ chức 2	<u>11001000 00010111 00010100</u>	00000000	200.23.20.0/23
...	....	....	....
Tổ chức 7	<u>11001000 00010111 00011110</u>	00000000	200.23.30.0/23

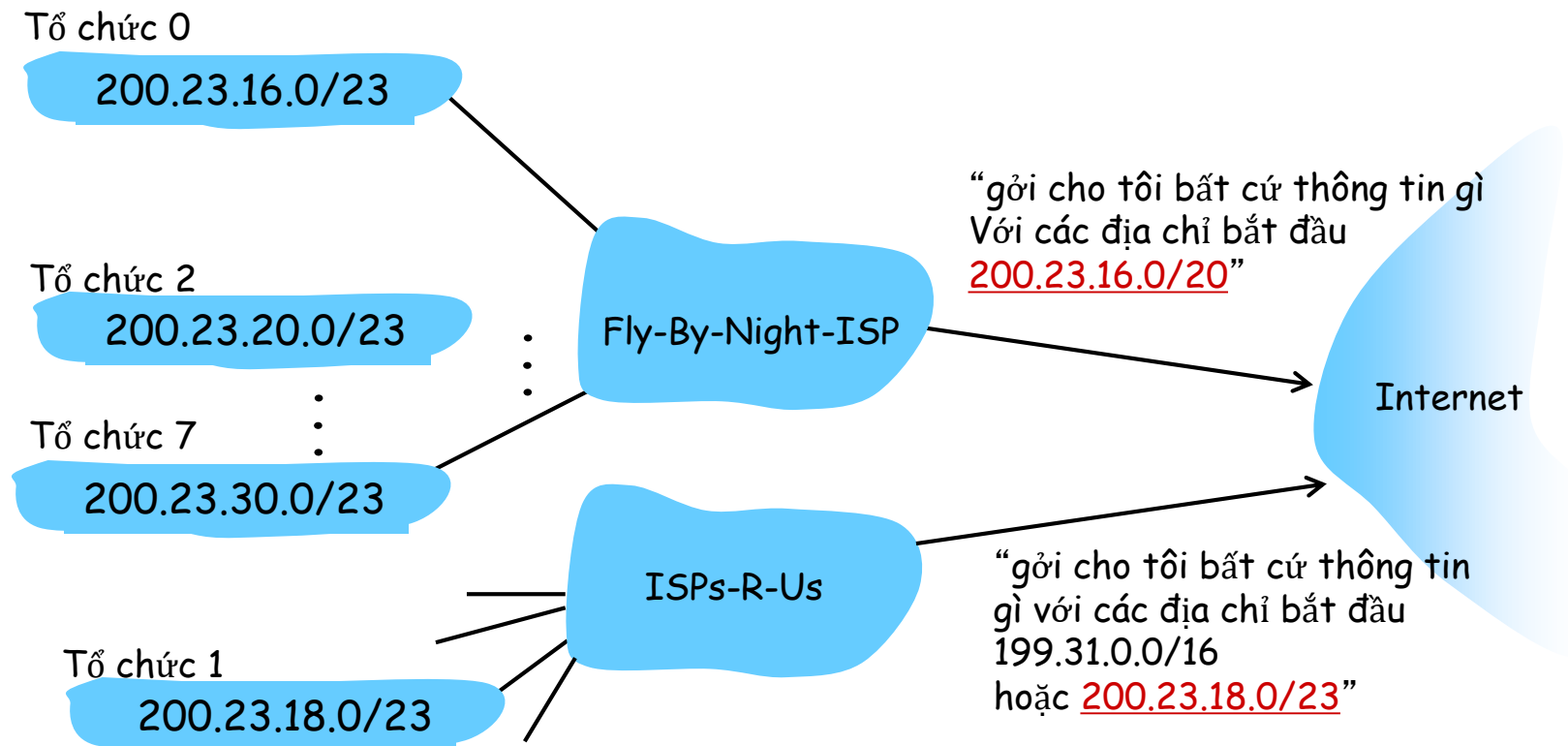
# Định địa chỉ phân cấp: route tích hợp

Định địa chỉ phân cấp cho phép quảng cáo hiệu quả thông tin định tuyến



# Định địa chỉ phân cấp: các route cụ thể hơn

ISPs-R-Us có 1 route cụ thể hơn tới tổ chức 1



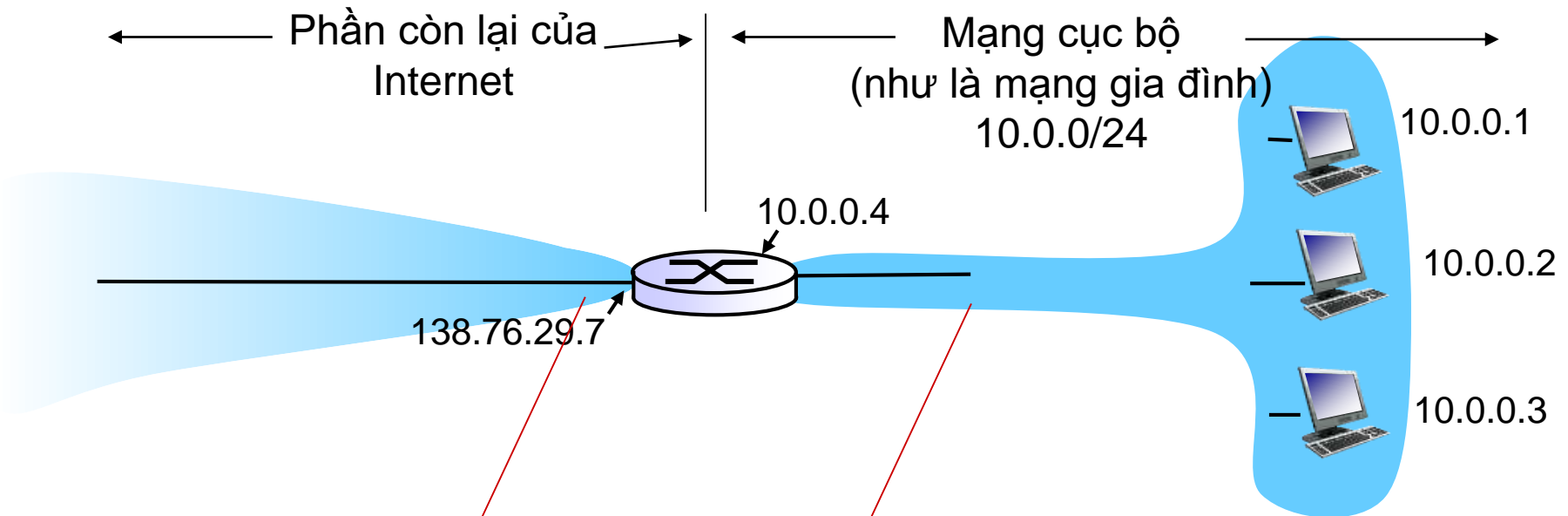
# Định địa chỉ IP: the last word...

**Q:** làm cách nào mà một ISP lấy được khối địa chỉ?

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

- Cấp phát địa chỉ
- Quản lý DNS
- Gán các tên miền, giải quyết tranh chấp

# NAT: network address translation



**Tất cả** datagram đi ra khỏi mạng cục bộ có cùng một địa chỉ IP NAT là: 138.76.29.7, với các số hiệu cổng nguồn khác nhau

Các datagram với nguồn hoặc đích trong mạng này có địa chỉ 10.0.0/24 cho nguồn, đích (như thông thường)



# NAT: network address translation

*Trình bày:* mạng cục bộ chỉ dùng 1 địa chỉ IP đối với thế giới bên ngoài:

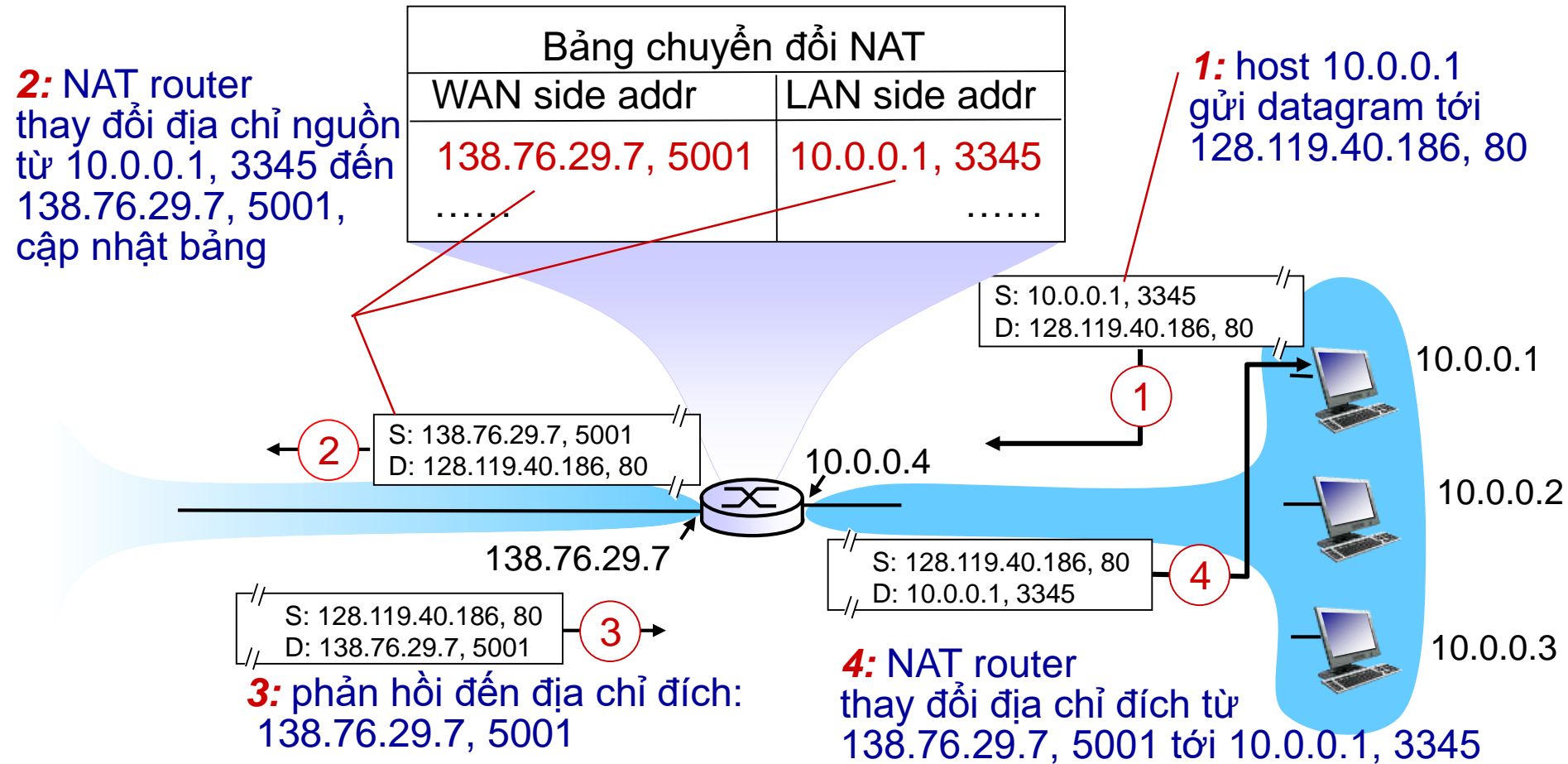
- Không cần thiết dùng 1 vùng địa chỉ từ ISP: chỉ cần 1 địa chỉ IP cho tất cả các thiết bị
- Có thể thay đổi các địa chỉ IP của các thiết bị trong mạng cục bộ mà không cần thông báo cho thế giới bên ngoài
- Có thể thay đổi ISP mà không cần thay đổi địa chỉ IP của các thiết bị trong mạng nội bộ
- Các thiết bị bên trong mạng cục bộ không nhìn thấy và không định địa chỉ rõ ràng từ bên ngoài (tăng cường bảo mật)

# NAT: network address translation

*thực hiện:* NAT router phải:

- *Các datagram đi ra: thay thế* (địa chỉ IP nguồn, số hiệu cổng nguồn) mọi datagram đi ra bên ngoài bằng (địa chỉ IP NAT, số hiệu port mới)  
... Các client/server ở xa sẽ dùng địa chỉ đó ( địa chỉ IP NAT, số hiệu port mới) như là địa chỉ đích
- *Ghi nhớ (trong bảng chuyển đổi NAT)* mọi cặp chuyển đổi (địa chỉ IP nguồn, số hiệu port) sang (địa chỉ IP NAT, số hiệu port mới)
- *Datagram đi đến: thay thế* (địa chỉ IP NAT, số hiệu port mới) trong các trường đích của mọi datagram đến với giá trị tương ứng (địa chỉ IP và số hiệu cổng nguồn) trong bảng NAT

# NAT: network address translation

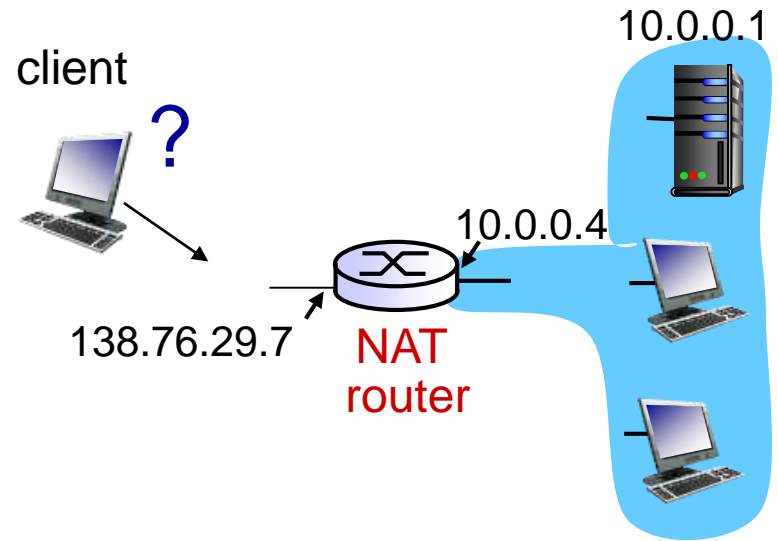


# NAT: network address translation

- ❖ Trường số hiệu port 16-bit:
  - 60,000 kết nối đồng thời với một địa chỉ phía LAN!
- ❖ NAT gây ra tranh luận:
  - Các router chỉ nên xử lý đến tầng 3
  - Vi phạm thỏa thuận end-to-end
    - Những nhà thiết kế ứng dụng phải tính đến khả năng của NAT, ví dụ ứng dụng P2P
  - Việc thiếu địa chỉ IP sẽ được giải quyết khi dùng IPv6

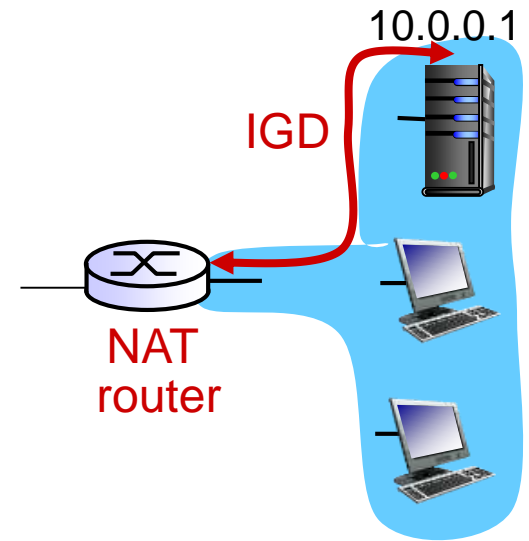
# Vấn đề NAT traversal

- ❖ Các client muốn kết nối tới server có địa chỉ 10.0.0.1
  - Địa chỉ 10.0.0.1 của server là ở trong mạng LAN (client không thể dùng nó như là địa chỉ IP đích)
  - Chỉ có 1 địa chỉ có thể được nhìn thấy từ bên ngoài địa chỉ được NAT: 138.76.29.7
- ❖ **Giải pháp 1:** cấu hình NAT tĩnh để chuyển các yêu cầu kết nối đến tại port được cho trước tới server
  - Ví dụ (138.76.29.7, port 2500) luôn luôn được chuyển tới 10.0.0.1 port 25000



# Vấn đề NAT traversal

- ❖ *Giải pháp 2:* Giao thức Universal Plug and Play (UPnP) Internet Gateway Device (IGD). Cho phép host được NAT:
  - ❖ Học địa chỉ IP public (138.76.29.7)
  - ❖ thêm/gỡ bỏ các port mapping (thời gian thuê)



# Các vấn đề NAT traversal

- ❖ **Giải pháp 3:** chuyển tiếp (relaying) (được sử dụng trong Skype)
  - Client được NAT thiết lập kết nối để chuyển tiếp
  - client bên ngoài kết nối để chuyển tiếp
  - Sự chuyển tiếp liên tục cầu các packet giữa các

2. Kết nối chuyển tiếp được khởi tạo bởi client

3. Chuyển tiếp được thiết lập

1. Kết nối chuyển tiếp được khởi tạo tại host được NAT

138.76.29.7

NAT router

10.0.0.1



# Chương 4: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- **ICMP**
- **IPv6**

## 4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

## 4.6 routing trong Internet

- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing



# ICMP: internet control message protocol

- ❖ Được sử dụng bởi các host và router để truyền thông tin tầng network
  - Thông báo: host, network, port, giao thức không có thực
  - Phản hồi request/reply (được dùng bởi ping)
- ❖ Tầng network “trên” IP:
  - Các thông điệp ICMP được mang trong các IP datagram
- ❖ **Thông điệp ICMP:** loại, mã cộng thêm 8 byte đầu tiên của IP datagram gây ra lỗi

<u>Loại</u>	<u>mã</u>	<u>Mô tả</u>
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

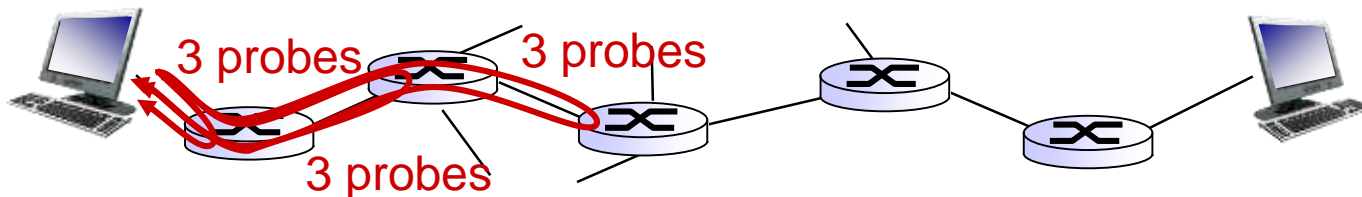
# Traceroute và ICMP

- ❖ Nguồn gửi một chuỗi các segment UDP đến đích
  - Cái đầu tiên có TTL = 1
  - Cái thứ 2 có TTL=2, tương tự.
  - Không giống số port
- ❖ Khi datagram thứ n đến router thứ n:
  - router hủy datagram
  - Và gửi đến nguồn một thông điệp ICMP (loại 11, mã 0)
  - Thông điệp ICMP bao gồm tên và địa chỉ IP của router

- ❖ Khi thông điệp ICMP đến, nguồn ghi lại RTTs

## *Tiêu chuẩn dừng:*

- ❖ Segment UDP lần lượt đến tới host đích
- ❖ Đích trả về thông điệp ICMP “port không có thực” (loại 3, mã 3)
- ❖ Nguồn dừng



# IPv6: Động lực

- ❖ *Động lực thúc đẩy ban đầu:* không gian địa chỉ 32-bit sớm được cấp phát cạn kiệt.
- ❖ Động lực khác:
  - Định dạng của header giúp tăng tốc xử lý/chuyển gói
  - header thay đổi để tạo điều kiện thuận lợi cho QoS

## *Định dạng datagram IPv6 :*

- Header có độ dài cố định 40 byte
- Không cho phép phân mảnh

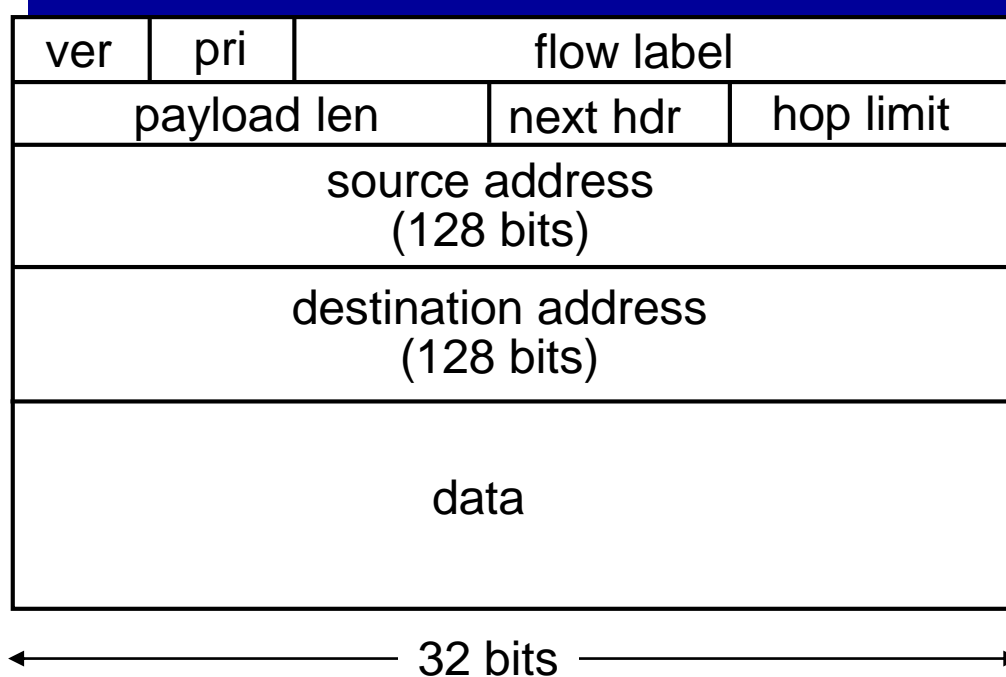
# Định dạng datagram IPv6

**Độ ưu tiên (priority):** xác định độ ưu tiên của các datagram trong luồng

**Nhãn luồng (flow Label):** xác định các datagram trong cùng "luồng".

(khái niệm "luồng" không được định nghĩa rõ ràng).

**Header kế tiếp (next header):** xác định giao thức Tầng trên cho dữ liệu.

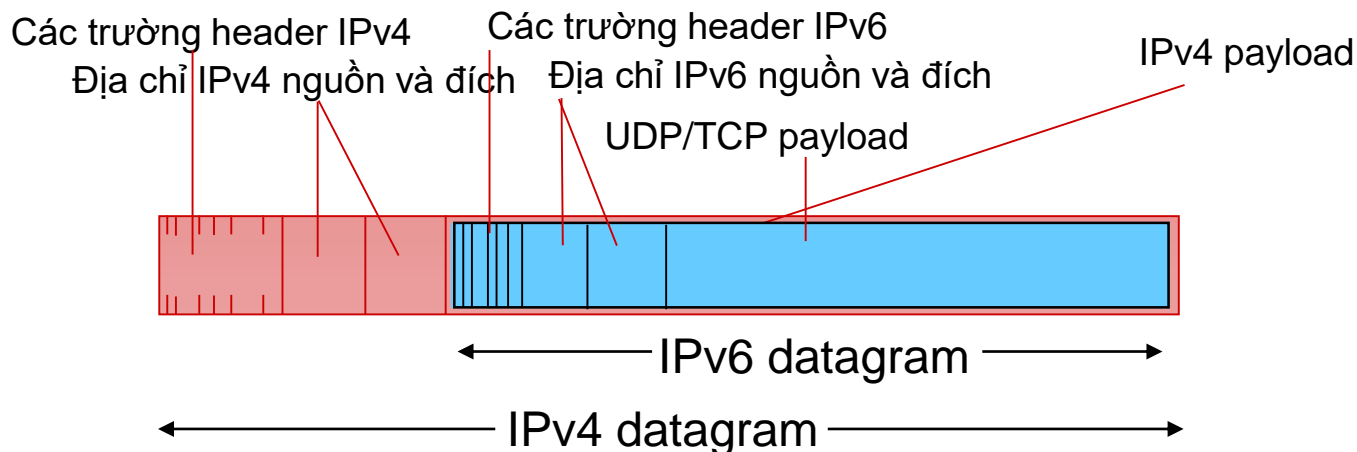


# Những thay đổi khác so với IPv4

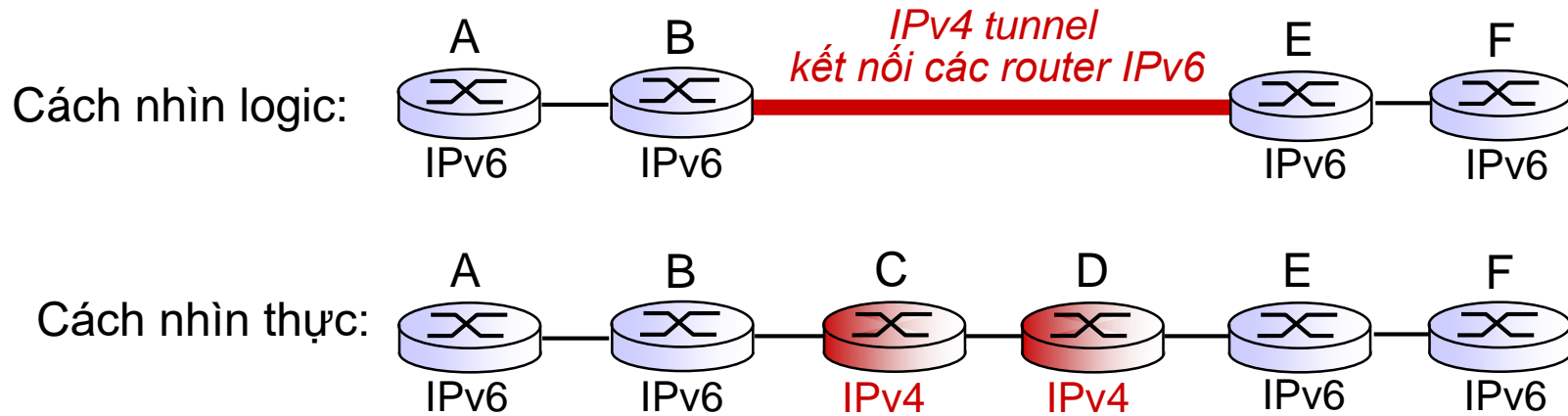
- ❖ **checksum**: được bỏ toàn bộ để giảm thời gian xử lý tại mỗi hop
- ❖ **options**: được cho phép, nhưng nằm ở ngoài header, được chỉ ra bởi trường “Next Header”
- ❖ **ICMPv6**: phiên bản mới của ICMP
  - Các kiểu thông điệp bổ sung. Ví dụ “Packet Too Big”
  - Các chức năng quản lý nhóm multicast

# Chuyển từ IPv4 sang IPv6

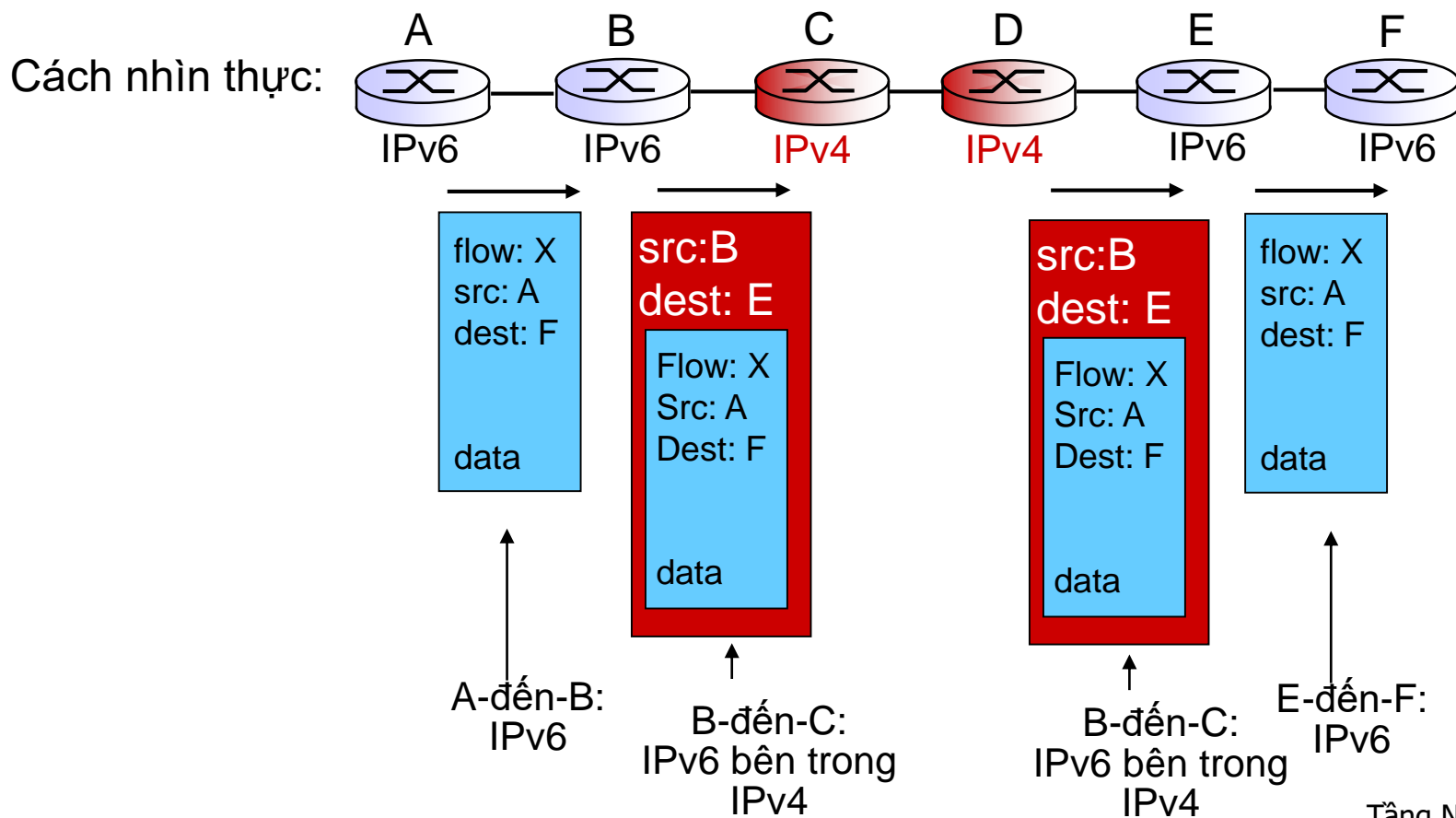
- ❖ Không phải tất cả router đều có thể được nâng cấp đồng thời
  - Không có “flag days”
  - Mạng sẽ hoạt động như thế nào với các router dùng cả IPv4 và IPv6?
- ❖ *tunneling*: datagram IPv6 được mang như là *payload* trong datagram của IPv4 giữa các router IPv4



# Tunneling



# Tunneling





# Chương 4: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

## 4.5 các thuật toán routing

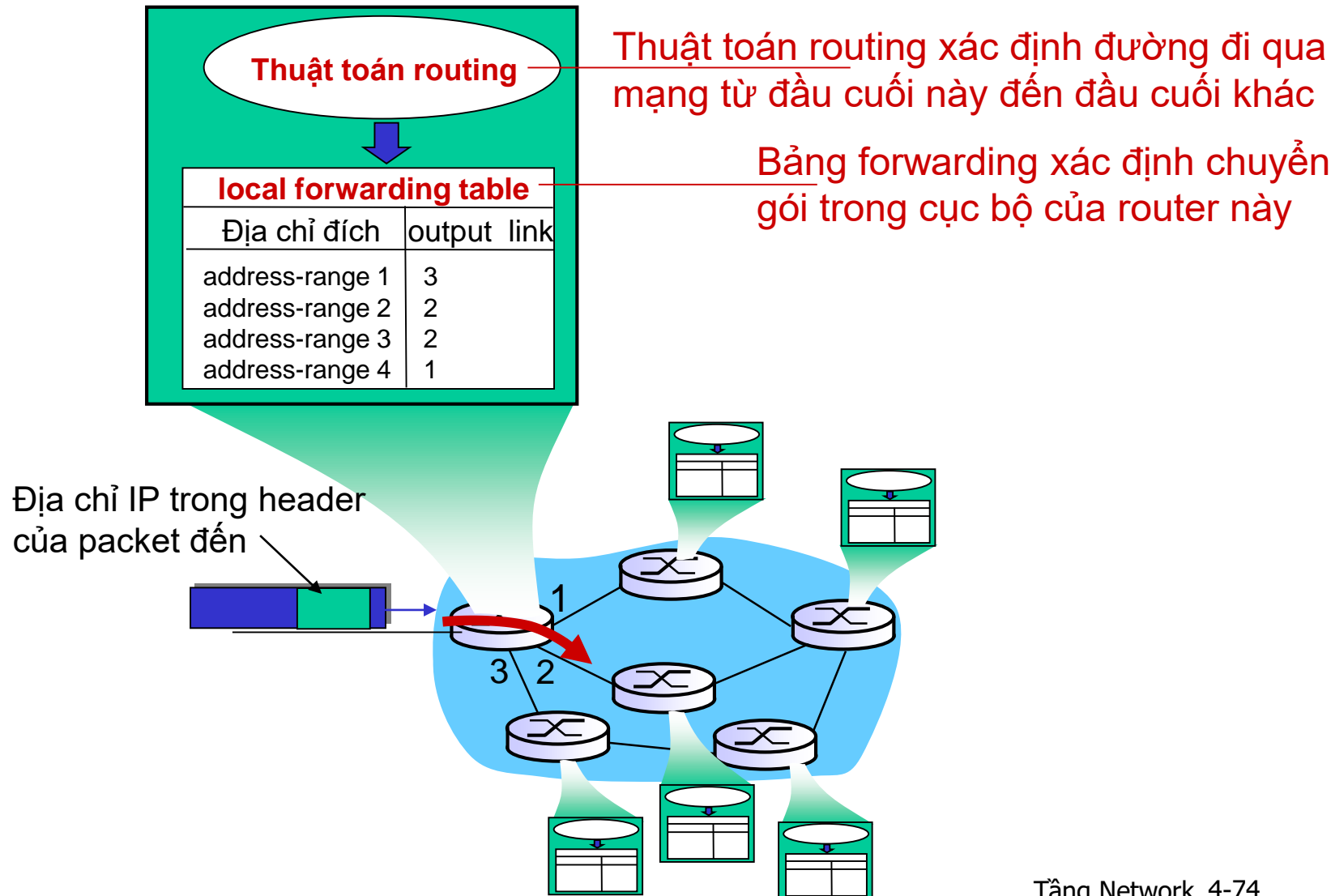
- link state
- distance vector
- hierarchical routing

## 4.6 routing trong Internet

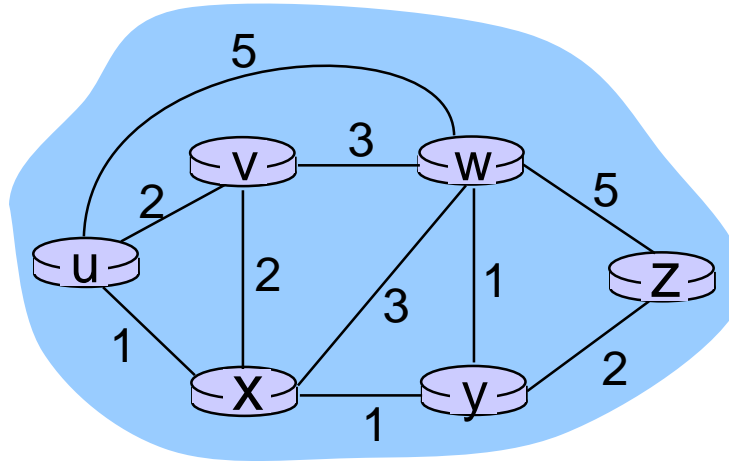
- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing

# Tác động lẫn nhau giữa routing và forwarding



# Mô hình đồ thị



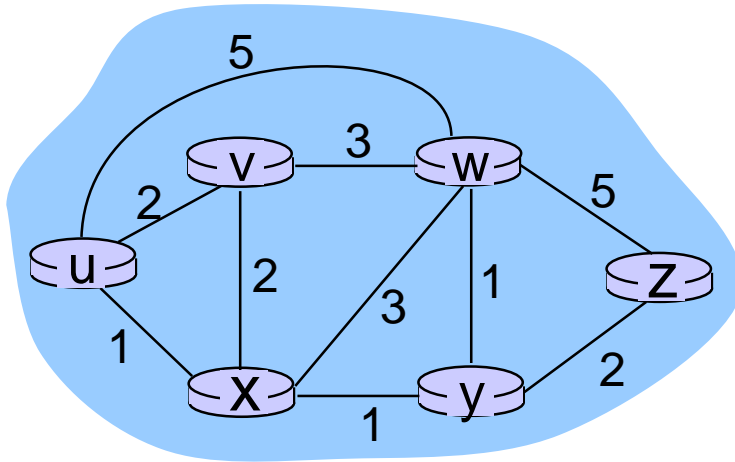
Đồ thị:  $G = (N, E)$

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

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

*Ghi chú:* mô hình đồ thị cũng hữu ích trong các ngữ cảnh khác.  
Ví dụ, P2P, trong đó  $N$  là tập các peer và  $E$  là tập các kết nối TCP

# Mô hình đồ thị: Chi phí



$c(x, x') =$  chi phí kết nối  $(x, x')$   
e.g.,  $c(w, z) = 5$

Chi phí có thể luôn luôn là 1, hoặc ngược lại liên quan đến băng thông, hoặc liên quan đến tắc nghẽn

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

**Hỏi:** Chi phí đường đi thấp nhất từ u tới z?

**Thuật toán routing:** thuật toán tìm đường có chi phí thấp nhất

# Phân loại thuật toán Routing

*Q: thông tin toàn cục hay phân cấp (global or decentralized)?*

*Toàn cục:*

- ❖ Tất cả các router có toàn bộ thông tin về chi phí kết nối, cấu trúc mạng
- ❖ Thuật toán “link state”

*Phân cấp:*

- ❖ router biết các neighbor được kết nối vật lý với nó, và chi phí kết nối đến neighbor đó
- ❖ Lặp lại qua trình tính toán, trao đổi thông tin với các neighbor
- ❖ Thuật toán “distance vector”

*Q: tĩnh hay động?  
tĩnh:*

- ❖ Các route thay đổi chậm theo thời gian

*Động:*

- ❖ Các route thay đổi nhanh
  - Cập nhật theo chu kỳ
  - Phản ứng với những thay đổi về chi phí kết nối

# Chương 4: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

## 4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

## 4.6 routing trong Internet

- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing

# Thuật toán routing Link-State

## Thuật toán Dijkstra

- ❖ Biết chi phí kết nối, cấu trúc mạng của tất cả các node
  - Được thực hiện thông qua “link state broadcast”
  - Tất cả các nodes có cùng thông tin với nhau
- ❖ Tính toán đường đi có chi phí thấp nhất từ 1 node ('nguồn') đến tất cả các node khác
  - Cho trước bảng *forwarding* của node đó
- ❖ Lặp lại: sau k lần lặp lại, biết được đường đi có chi phí thấp nhất của k đích

## Ký hiệu:

- ❖  $c(x,y)$ : chi phí kết nối từ node x đến y;  $= \infty$  nếu không kết nối trực tiếp đến neighbor
- ❖  $D(v)$ : giá trị chi phí hiện tại của đường đi từ nguồn tới đích v
- ❖  $p(v)$ : node trước nằm trên đường đi từ nguồn tới v
- ❖  $N'$ : tập các node mà chi phí đường đi thấp nhất đã được xác định

# Thuật toán Dijkstra

1 **Khởi tạo:**

2  $N' = \{u\}$

3 for all nodes  $v$

4 if  $v$  adjacent to  $u$

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

6 else  $D(v) = \infty$

7

8 **Lặp**

9 find  $w$  not in  $N'$  such that  $D(w)$  is a minimum

10 add  $w$  to  $N'$

11 update  $D(v)$  for all  $v$  adjacent to  $w$  and not in  $N'$  :

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

13 /\* chi phí mới đến  $v$  là chính nó hoặc chi phí đường đi ngắn nhất

14 cộng với chi phí từ  $w$  đến  $v$ \*/

15 **until all nodes in  $N'$**

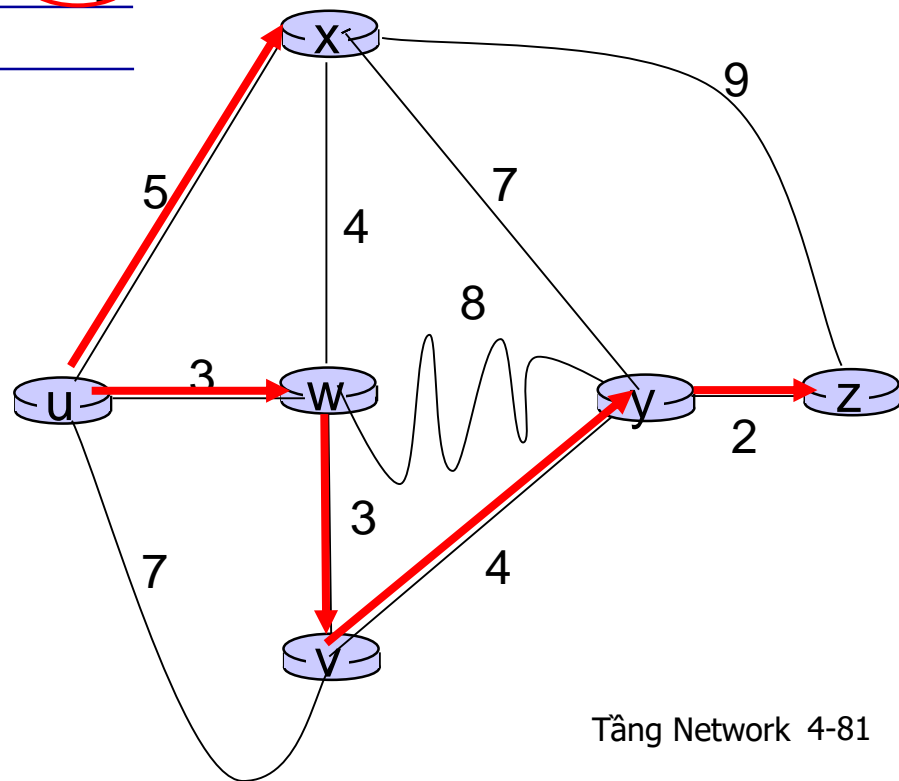


# Thuật toán Dijkstra : ví dụ

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

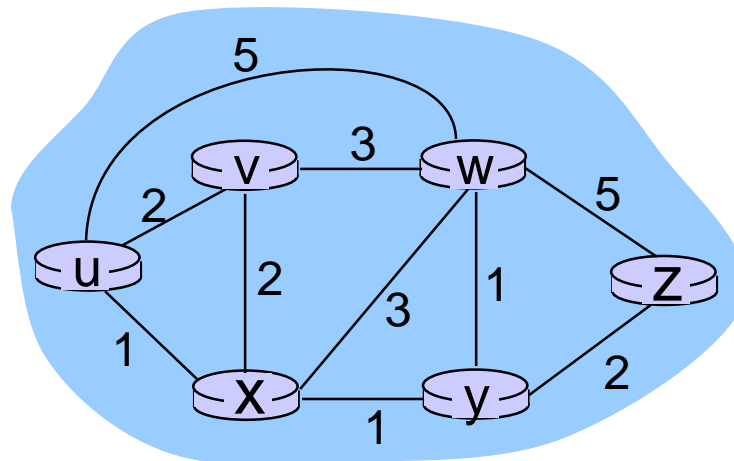
## Ghi chú:

- ❖ Xây dựng cây đường đi ngắn nhất bằng cách lần theo các predecessor node
- ❖ Các đường có chi phí bằng nhau có thể tồn tại (có thể bị chia tùy tiện)



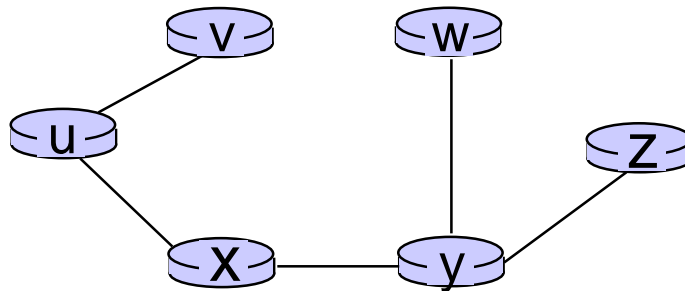
# Thuật toán Dijkstra: ví dụ

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					



# Thuật toán Dijkstra: ví dụ (2)

Kết quả cây đường đi ngắn nhất từ u:



Kết quả bảng forwarding trong u:

Đích đến	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

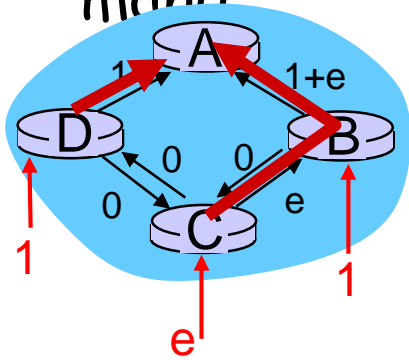
# Thuật toán Dijkstra, thảo luận

**Độ phức tạp của thuật toán:** n nodes

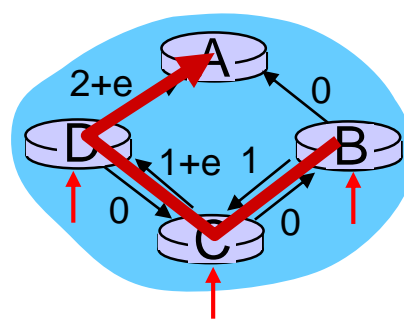
- ❖ Mỗi lần duyệt: cần kiểm tra tất cả các node, w, không có trong N
- ❖  $n(n+1)/2$  phép so sánh:  $O(n^2)$
- ❖ Có nhiều cách thực hiện hiệu quả hơn:  $O(n \log n)$

**Có thể dao động:**

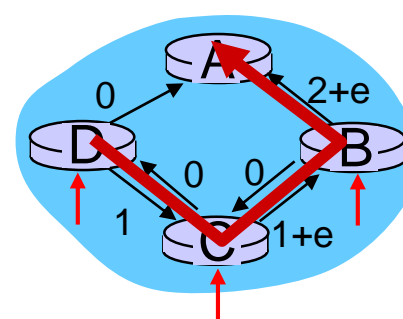
- ❖ Ví dụ: chi phí kết nối bằng với số lượng lưu thông được mang



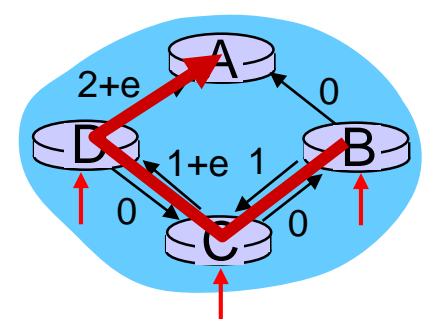
Khởi tạo



Cho các chi phí,  
Tìm định tuyến mới....  
Kết quả trong chi phí mới



Cho các chi phí,  
Tìm định tuyến mới....  
Kết quả trong chi phí mới



Cho các chi phí,  
Tìm định tuyến mới....  
Kết quả trong chi phí mới

# Chương 4: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

## 4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

## 4.6 routing trong Internet

- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing

# Thuật toán Distance vector

*Công thức Bellman-Ford (dynamic programming)*

cho

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

thì

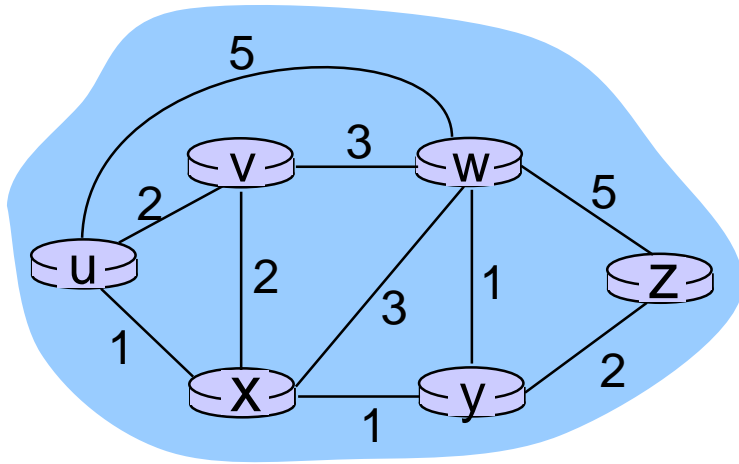
$$d_x(y) = \min \{ c(x,v) + d_v(y) \}$$

Chi phí từ neighbor  $v$  tới đích  $y$

Chi phí tới neighbor  $v$

$\min$  được thực hiện trên tất cả các neighbor  $v$  của  $x$

# Bellman-Ford ví dụ



Rõ ràng,  $d_v(z) = 5$ ,  $d_x(z) = 3$ ,  $d_w(z) = 3$

Cộng B-F cho:

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

node đạt được tối thiểu là hop kế tiếp trong đường đi ngắn nhất, được sử dụng trong bảng forwarding

# Thuật toán Distance vector

- ❖  $D_x(y)$  = ước lượng chi phí thấp nhất từ  $x$  đến  $y$ 
  - $x$  duy trì distance vector  $D_x = [D_x(y): y \in N]$
- ❖ node  $x$ :
  - Biết chi phí đến mỗi neighbor  $v$ :  $c(x,v)$
  - Duy trì distance vectors của các neighbor của nó. Cho mỗi duy trì neighbor  $v$ ,  $x$   
 $D_v = [D_v(y): y \in N]$



# Thuật toán Distance vector

## Ý tưởng chính:

- ❖ Mỗi node định kỳ gửi ước lượng distance vector của nó cho các neighbor
- ❖ Khi x nhận ước lượng DV mới từ neighbor, nó cập nhật DV cũ của nó dùng công thức B-F:

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

- ❖ Dưới những điều kiện tự nhiên, ước lượng  $D_x(y)$  hội tụ tới chi phí dx thực sự nhỏ nhất  $d_x(y)$

# Thuật toán Distance vector

## *Lặp, không đồng bộ:*

mỗi lần cục bộ được  
gây ra bởi:

- ❖ Chi phí kết nối cục bộ thay đổi
- ❖ Thông điệp cập nhật DV từ neighbor

## *Phân bố:*

- ❖ Mỗi node thông báo đến các neighbor *chỉ* khi DV của nó thay đổi
  - Các neighbor sau đó thông báo đến các neighbor của nó nếu cần thiết

## *Mỗi node:*

*Chờ cho* (thay đổi trong chi phí link cục bộ hoặc thông điệp từ neighbor)

*Tính toán lại* các ước lượng

Nếu DV đến đích bất kỳ vừa thay đổi, thì thông báo neighbors

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

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

**Bảng  
node x**

		Chi phí đến		
		x	y	z
Từ	x	0	2	7
	y	$\infty$	$\infty$	$\infty$
	z	$\infty$	$\infty$	$\infty$

Chi phí đến

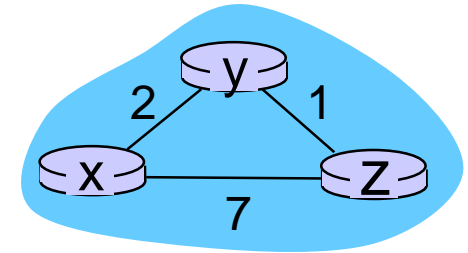
		x	y	z
Từ	x	0	2	3
	y	2	0	1
	z	7	1	0

**Bảng  
node y**

		Chi phí đến		
		x	y	z
Từ	x	$\infty$	$\infty$	$\infty$
	y	2	0	1
	z	$\infty$	$\infty$	$\infty$

**Bảng  
node z**

		Chi phí đến		
		x	y	z
Từ	x	$\infty$	$\infty$	$\infty$
	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  
node x**

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

chi phí đến

	x	y	z
từ x	0	2	3
y	2	0	1
z	7	1	0

chi phí đến

	x	y	z
từ x	0	2	3
y	2	0	1
z	3	1	0

**Bảng  
node y**

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

chi phí đến

	x	y	z
từ y	2	0	1
x	0	2	7
z	7	1	0

chi phí đến

	x	y	z
từ y	2	0	1
x	0	2	3
z	3	1	0

**Bảng  
node z**

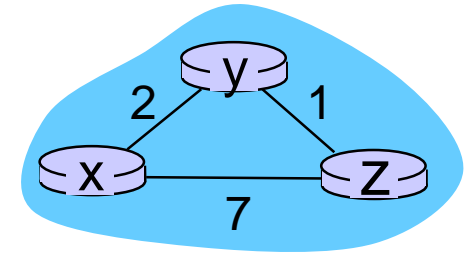
	chi phí đến		
	x	y	z
từ z	$\infty$	$\infty$	$\infty$
x	$\infty$	$\infty$	$\infty$
y	7	1	0

chi phí đến

	x	y	z
từ z	3	1	0
x	0	2	7
y	2	0	1

chi phí đến

	x	y	z
từ z	3	1	0
x	0	2	3
y	2	0	1

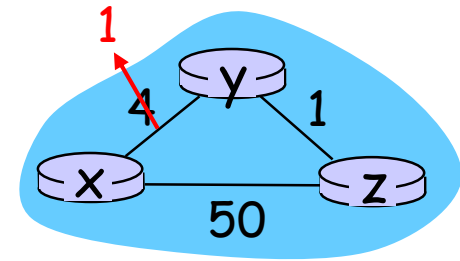


Thời gian

# Distance vector: chi phí kết nối thay đổi

## Chi phí kết nối thay đổi:

- ❖ node phát hiện sự thay đổi chi phí kết nối cục bộ
- ❖ Cập nhật thông tin định tuyến, tính toán lại distance vector
- ❖ Nếu DV thay đổi, thì thông báo cho neighbor



“tin  
tốt  
đi  
nhanh”

$t_0$ : y phát hiện sự thay đổi chi phí kết nối, và cập nhật DV của nó, thông báo đến các neighbor của nó.

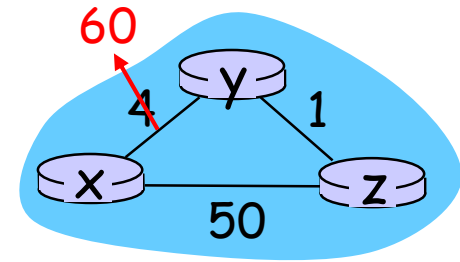
$t_1$ : z nhận được cập nhật từ y, và cập nhật bảng của nó, tính chi phí mới thấp nhất đến x, gửi DV của nó đến các neighbor của nó.

$t_2$ : y nhận được cập nhật của z, và cập nhật bảng distance của nó. Các chi phí thấp nhất của y *không* thay đổi, vì vậy y không gửi thông điệp đến z.

# Distance vector: Chi phí kết nối thay đổi

## *Chi phí kết nối thay đổi:*

- ❖ node phát hiện sự thay đổi chi phí kết nối cục bộ
- ❖ *Tin xấu đi chậm*- “đếm đến vô cùng” vấn đề!
- ❖ 44 lần duyệt trước khi thuật toán ổn định



## *Tác động xấu ngược lại:*

- ❖ Nếu Z đi qua Y để tới X :
  - Z nói với Y khoảng cách của nó đến X là không xác định (vì vậy Y sẽ không đi tới X thông qua Z)
- ❖ Điều này sẽ giải quyết được vấn đề đếm đến vô tận hay không?

# So sánh giữa thuật toán LS và DV

## *Độ phức tạp của thông điệp*

- ❖ **LS:** với  $n$  nodes,  $E$  kết nối, thì có  $O(nE)$  các thông điệp được gửi
- ❖ **DV:** chỉ trao đổi giữa các neighbor
  - Thời gian hội tụ khác nhau

## *Tốc độ hội tụ*

- ❖ **LS:** thuật toán  $O(n^2)$  yêu cầu  $O(nE)$  thông điệp
  - Có thể có dao động
- ❖ **DV:** thời gian hội tụ khác nhau
  - Có thể là routing loop
  - Vấn đề đếm đến vô hạn (count-to-infinity)

**Sự linh hoạt:** điều gì xảy ra nếu router gặp sự cố?

### **LS:**

- node có thể quảng cáo sai chi phí kết nối
- Mỗi node chỉ tính toán bảng *riêng* của nó

### **DV:**

- DV node có thể quảng cáo sai về chi phí *đường đi*
- Bảng của mỗi node được sử dụng bởi các node khác
  - Lỗi được lan truyền thông qua mạng

# Chương 4: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

## 4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

## 4.6 routing trong Internet

- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing



# Định tuyến có cấu trúc (Hierarchical routing)

Cho tới đây, tìm hiểu về routing của chúng ta thực hiện trong môi trường lý tưởng hóa

- ❖ Tất cả các router là đồng nhất
- ❖ Mạng “phẳng”

*... không đúng trong thực tế*

**Quy mô:** với 600 triệu  
đích đến:

- ❖ Không thể lưu trữ tất cả các đích đến trong các bảng định tuyến
- ❖ Việc trao đổi bảng định sẽ làm tràn ngập các liên kết!

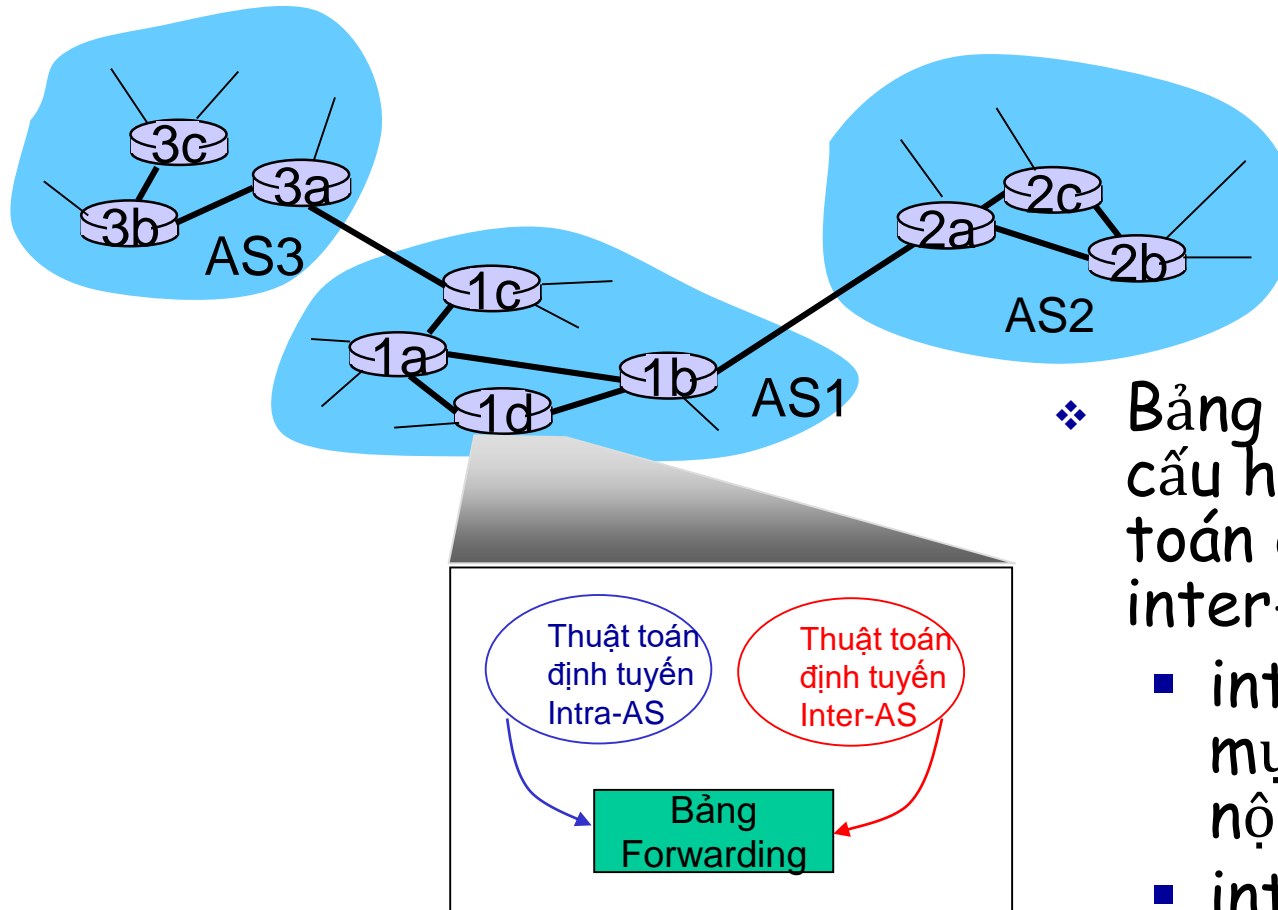
**Quản trị**

- ❖ internet = mạng của các mạng
- ❖ Mỗi quản trị mạng có thể muốn điều hành định tuyến trong mạng của riêng họ

# Định tuyến có cấu trúc

- ❖ Các router được gom vào các vùng, “autonomous systems” (AS)
- ❖ Các router trong cùng AS chạy cùng giao thức định tuyến với nhau
  - giao thức định tuyến “intra-AS”
  - Các router trong các AS khác nhau có thể chạy các giao thức định tuyến intra-AS khác nhau
- gateway router:*
  - ❖ Tại “biên” (“edge”) của AS của nó
  - ❖ Có liên kết đến router trong AS khác

# Kết nối các AS



❖ Bảng forwarding được cấu hình bởi cả thuật toán định tuyến intra- và inter-AS

- intra-AS thiết lập các mục cho các đích đến nội mạng
- inter-AS & intra-AS thiết lập các mục cho các đích đến ngoại

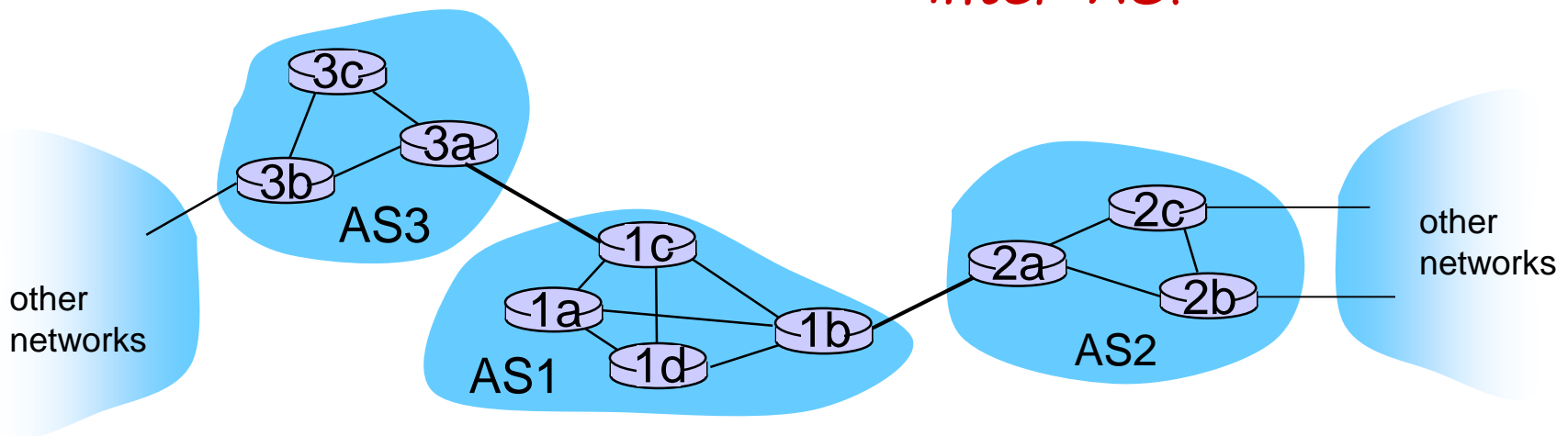
# Tác vụ Inter-AS

- ❖ Giả sử router trong AS1 nhận được datagram với đích đến nằm ngoài AS1:
  - router nên chuyển packet đến router gateway, nhưng là cái nào?

## *AS1 phải:*

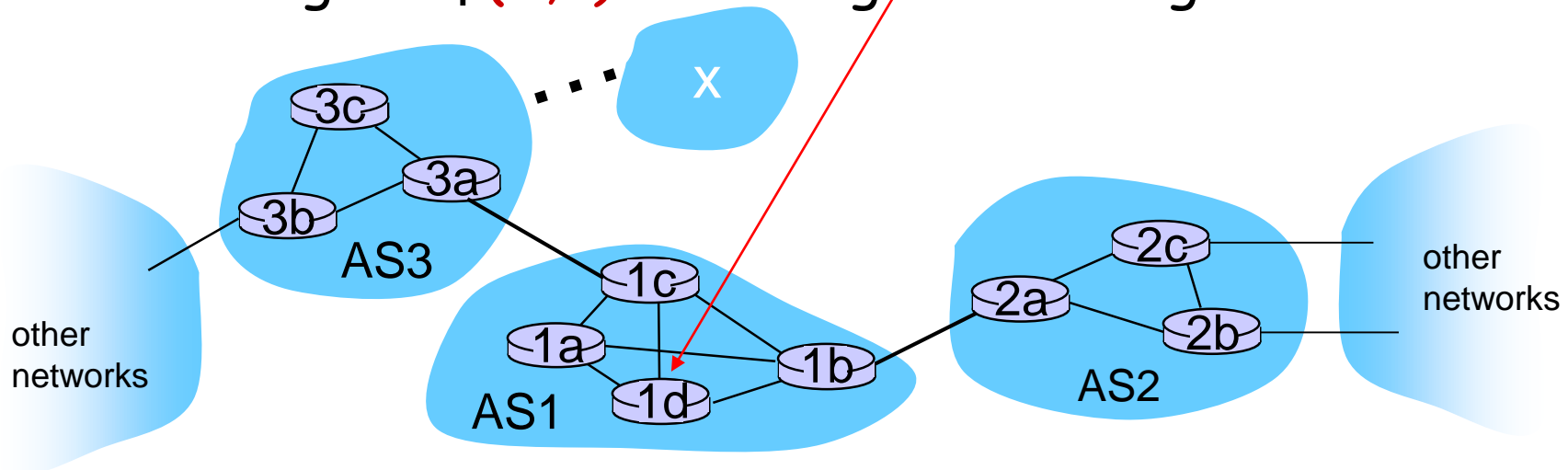
1. Học các đích đến nào có thể tới được thông qua AS2 và AS3
2. Lan truyền thông tin này đến tất cả các router trong AS1

## *Công việc của định tuyến inter-AS!*



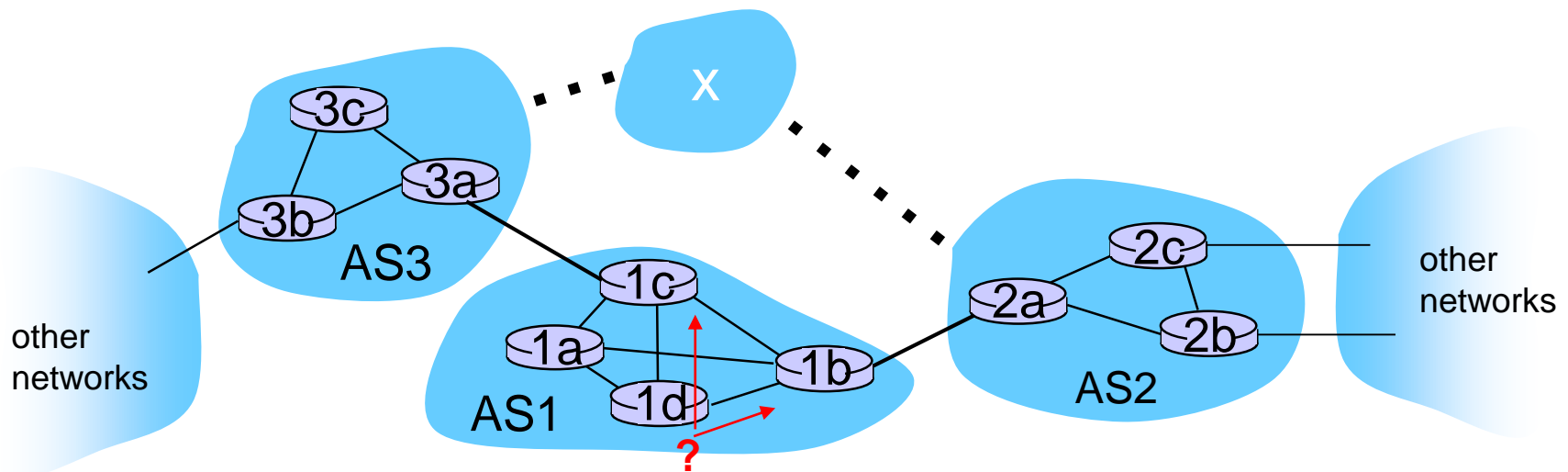
# Ví dụ: thiết lập bảng forwarding trong router 1d

- ❖ Giả sử AS1 học (thông qua giao thức inter-AS) mà subnet **x** có thể chạm tới thông qua AS3 (gateway 1c), nhưng không qua AS2
  - Giao thức inter-AS lan truyền thông tin này đến tất cả các router nội mạng
- ❖ router 1d xác định từ thông tin định tuyến intra-AS mà interface **I** của nó nằm trên đường đi có chi phí thấp nhất tới 1c
  - Đưa giá trị **(x,I)** vào bảng forwarding



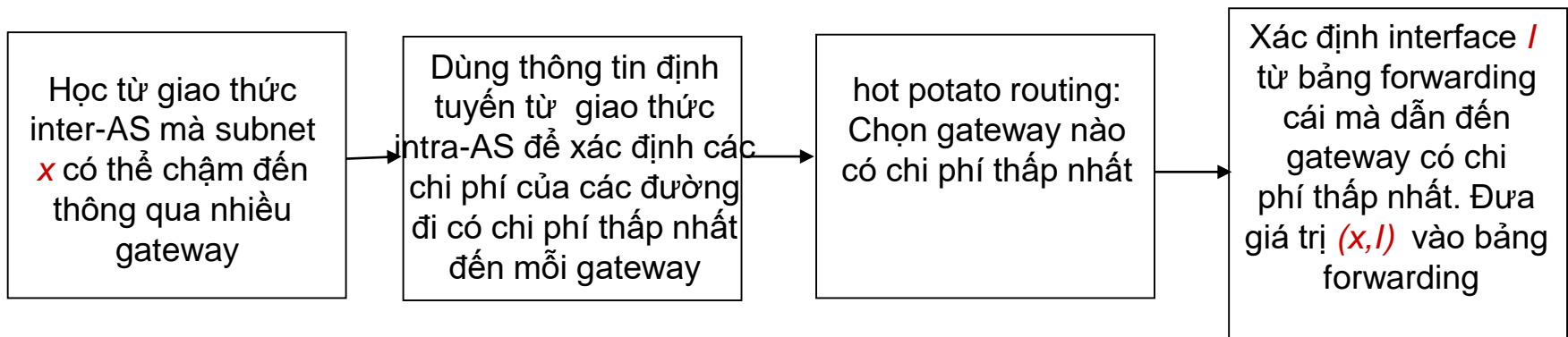
# Ví dụ: chọn giữa nhiều AS

- ❖ Bây giờ, giả sử AS1 học từ giao thức inter-AS mà subnet **x** có thể chạm tới từ AS3 và từ AS2.
- ❖ Để cấu hình bảng forwarding, router 1d phải xác định gateway nào mà nó nên dùng để chuyển các packet đến đích **x**
  - Đây cũng là công việc của giao thức định tuyến inter-AS!



# Ví dụ: chọn giữa nhiều AS

- ❖ Bây giờ, giả sử AS1 học từ giao thức inter-AS mà subnet **x** có thể chạm tới được từ AS3 và từ AS2.
- ❖ Để cấu hình bảng forwarding, router 1d phải xác định gateway nào nó nên dùng để chuyển packet tới đích x
  - Đây cũng là công việc của giao thức định tuyến inter-AS!
- ❖ *hot potato routing: gửi* packet tới 2 router gần nhất



# Chương 4: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

## 4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

## 4.6 routing trong Internet

- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing

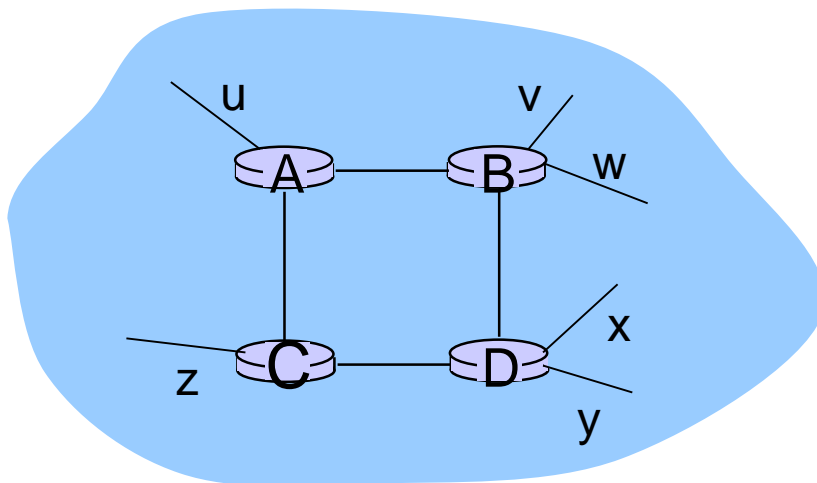


# Định tuyến Intra-AS

- ❖ Còn gọi là *interior gateway protocols (IGP)*
- ❖ Các giao thức định tuyến intra-AS phổ biến:
  - RIP: Routing Information Protocol
  - OSPF: Open Shortest Path First
  - IGRP: Interior Gateway Routing Protocol (độc quyền của Cisco)

# RIP ( Routing Information Protocol)

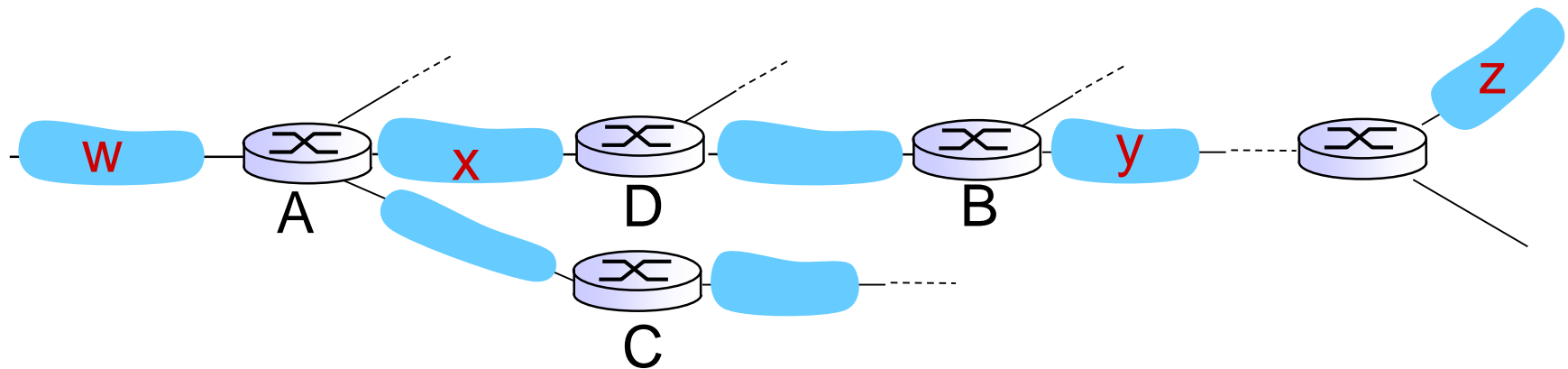
- ❖ Công bố vào năm 1982 trong BSD-UNIX
- ❖ Thuật toán distance vector
  - Metric khoảng cách: số lượng hop (max = 15 hops), mỗi link có giá trị là 1
  - Các DV được trao đổi giữa các neighbors mỗi 30 giây trong thông điệp phản hồi (còn gọi là **advertisement**)
  - Mỗi advertisement: danh sách lên đến 25 **subnet** đích



Từ router A đến các **subnet** đích:

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

# RIP: ví dụ



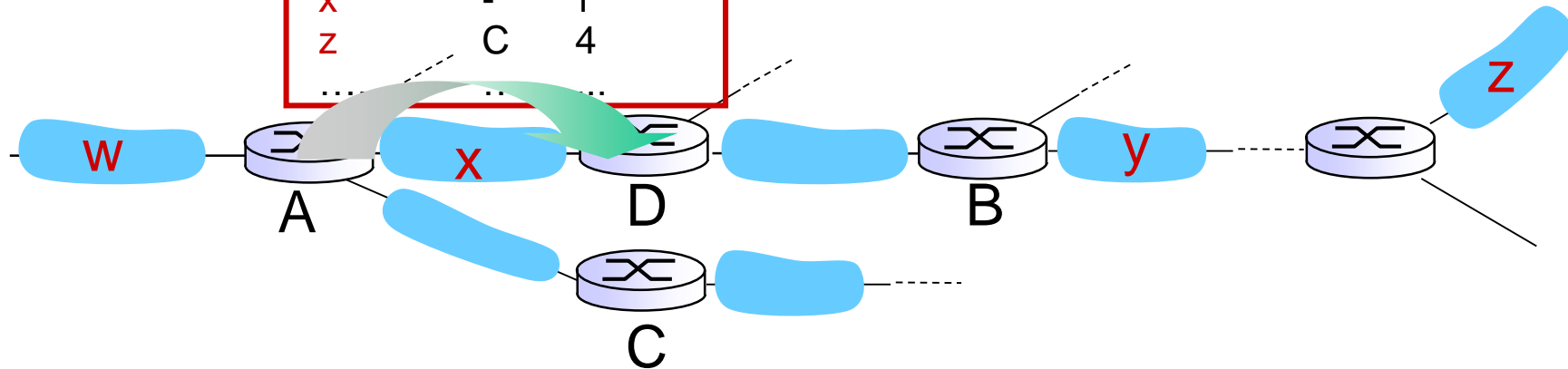
Bảng định tuyến trong router D

Subnet đích	router kế tiếp	số lượng hop đến đích
W	A	2
y	B	2
z	B	7
x	--	1
....	....	....

# RIP: ví dụ

Quảng cáo từ A-tới-D

đích	kế	hops
w	-	1
x	-	1
z	C	4
...	...	...



Bảng định tuyến trong router D

Subnet đích	router kế tiếp	số lượng hops tới đích
w	A	2
y	B	2
z	<del>B</del> → A	<del>7</del> → 5
x	--	1
....	....	....

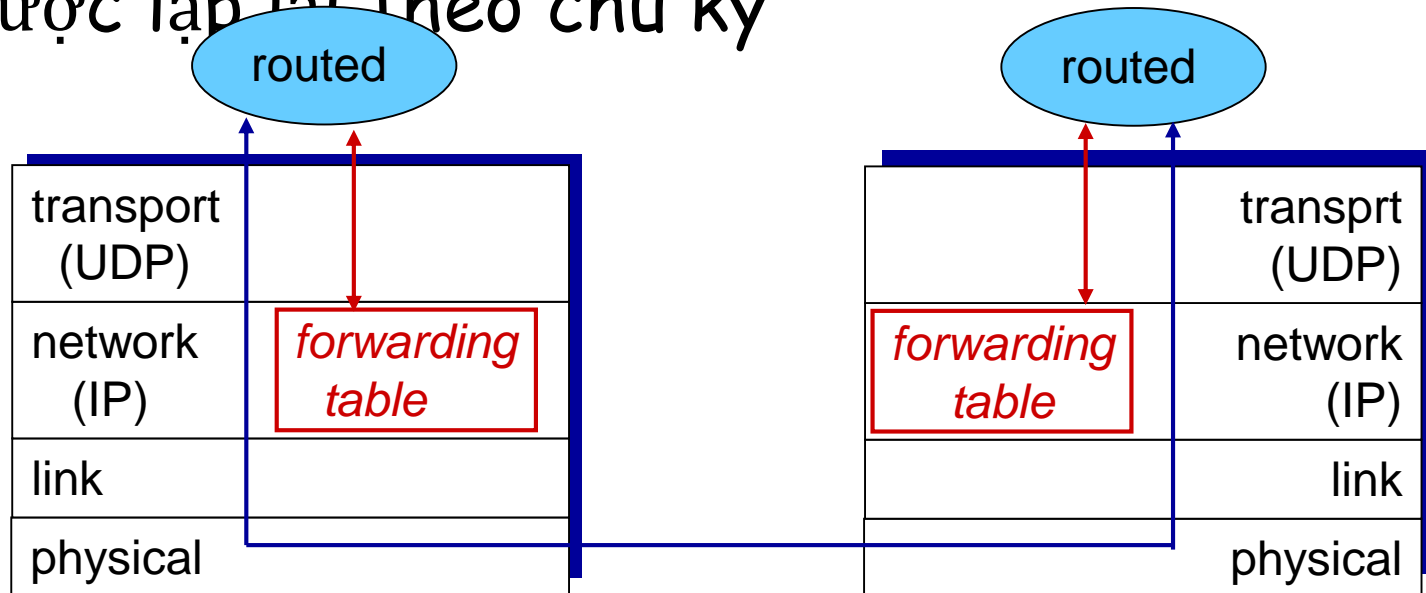
# RIP: lỗi đường kết nối và phục hồi

Nếu không có quảng cáo nào sau 180 giây --> neighbor/kết nối được xem như đã chết

- Những đường đi qua neighbor bị vô hiệu
- Các quảng cáo mới được gửi tới các neighbor
- Các neighbor đó tiếp tục gửi ra những quảng cáo mới đó (nếu các bảng bị thay đổi)
- Thông tin về lỗi đường kết nối nhanh chóng (?) lan truyền trên toàn mạng
- *poison reverse* được dùng để ngăn chặn vòng lặp ping-pong (khoảng cách vô hạn = 16 hops)

# RIP: xử lý bảng

- ❖ Các bảng định tuyến được quản lý bởi tiến trình ở mức *application* được gọi là route-d (daemon)
- ❖ Các quảng cáo được gửi trong các packet UDP, được lặp lại theo chu kỳ



# OSPF (Open Shortest Path First)

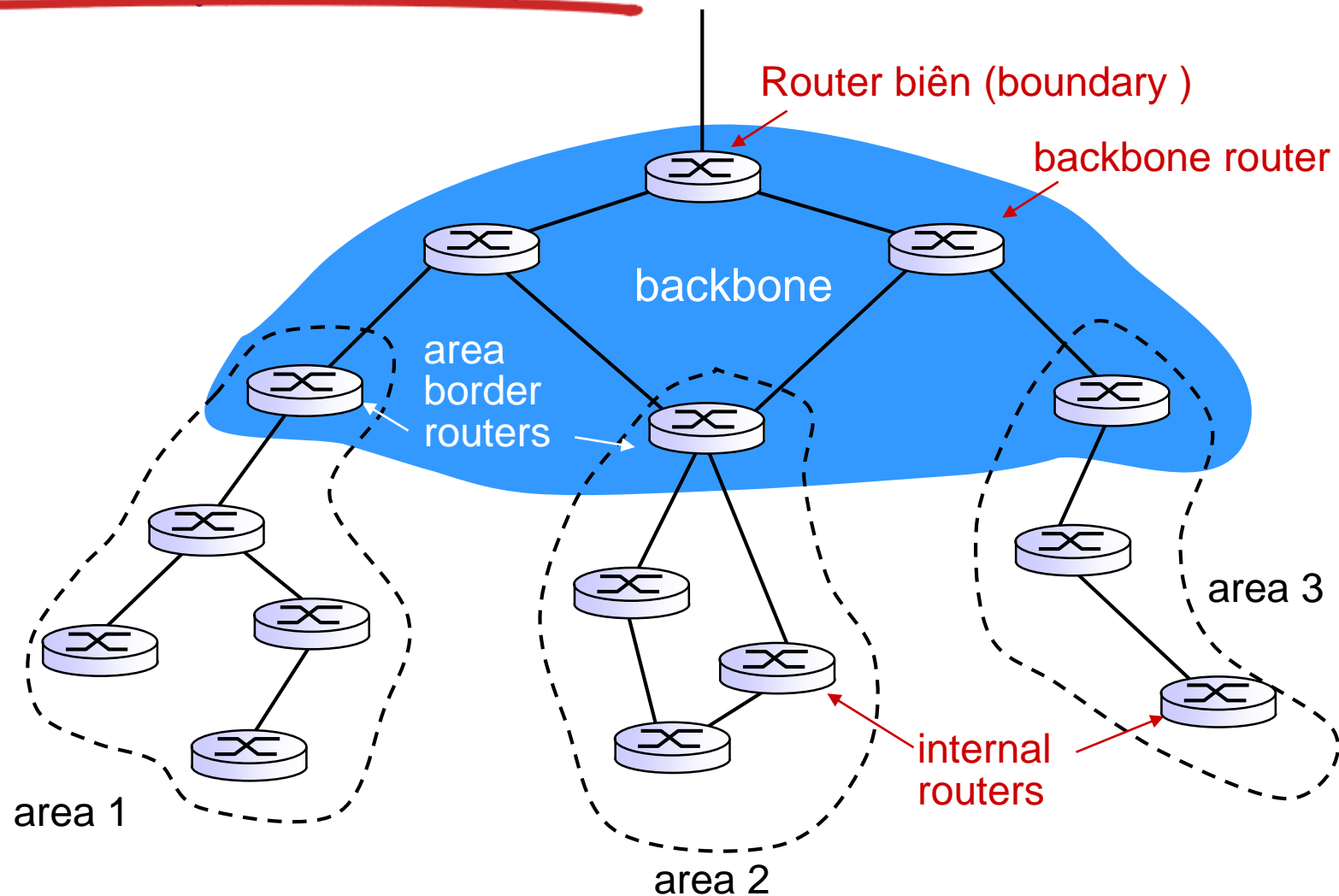
- ❖ “open”: sẵn sàng công khai
- ❖ Dùng thuật toán link state
  - Phổ biến packet LS
  - Bảng đồ cấu trúc mạng tại mỗi node
  - Tính toán đường đi dùng thuật toán Dijkstra
- ❖ Quảng cáo OSPF mang 1 entry cho mỗi neighbor
- ❖ Các quảng cáo được phát tán đến **toàn bộ** AS
  - Được mang trong thông điệp OSPF trực tiếp trên IP (chứ không phải là TCP hoặc UDP)
- ❖ Giao thức **định tuyến IS-IS** : gần giống với OSPF

# Các đặc tính “tiến bộ” của OSPF (không có trong RIP)

- ❖ **Bảo mật**: tất cả các thông điệp OSPF được chứng thực (để chống lại sự xâm nhập có hại)
- ❖ Cho phép có **nhiều đường đi** có chi phí như nhau (RIP chỉ cho 1)
- ❖ Với mỗi đường kết nối, nhiều số liệu chi phí (cost metrics) cho **TOS** khác nhau (ví dụ: chi phí đường kết nối vệ tinh được thiết lập “thấp” cho ToS nỗ lực tốt nhất; cao cho ToS thời gian thực)
- ❖ Hỗ trợ uni- và **multicast** tích hợp:
  - Multicast OSPF (MOSPF) dùng cùng cơ sở dữ liệu cấu trúc mạng như OSPF
- ❖ OSPF **phân cấp** trong các miền lớn (large domains).



# OSPF phân cấp



# OSPF phân cấp

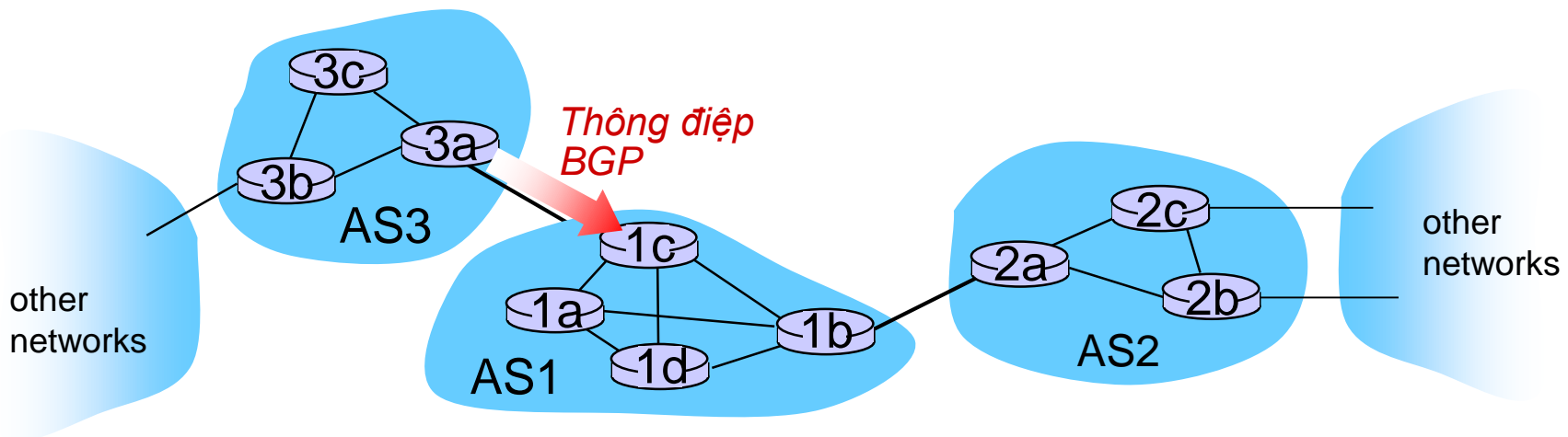
- ❖ *Phân cấp mức 2*: vùng cục bộ (local area), backbone.
  - Các quảng cáo link-state chỉ trong vùng
  - Mỗi node có chi tiết cấu trúc của vùng; chỉ biết hướng (đường ngắn nhất) đến các mạng trong các vùng khác.
- ❖ *Các router area border*: “tóm tắt” các khoảng cách đến các mạng trong vùng của nó, quảng cáo đến các router Area Border của các vùng khác.
- ❖ *Các backbone router*: chạy định tuyến OSPF được giới hạn đến backbone.
- ❖ *boundary routers*: kết nối đến các AS khác.

# Định tuyến Internet inter-AS : BGP

- ❖ **BGP (Border Gateway Protocol):** giao thức định tuyến liên miền (inter-domain ) trên thực tế
  - “gắn kết Internet lại với nhau”
- ❖ BGP cũng cấp cho mỗi AS một phương tiện để:
  - **eBGP:** lấy thông tin có thể chạm tới subnet từ các AS neighbor.
  - **iBGP:** lan truyền thông tin đó đến tất cả các router bên trong AS.
  - Xác định các đường đi “tốt” đến các mạng khác dựa trên thông tin khả năng chạm tới và chính sách.
- ❖ Cho phép subnet quảng cáo sự tồn tại của nó đến phần còn lại của Internet: ***“I am here”***

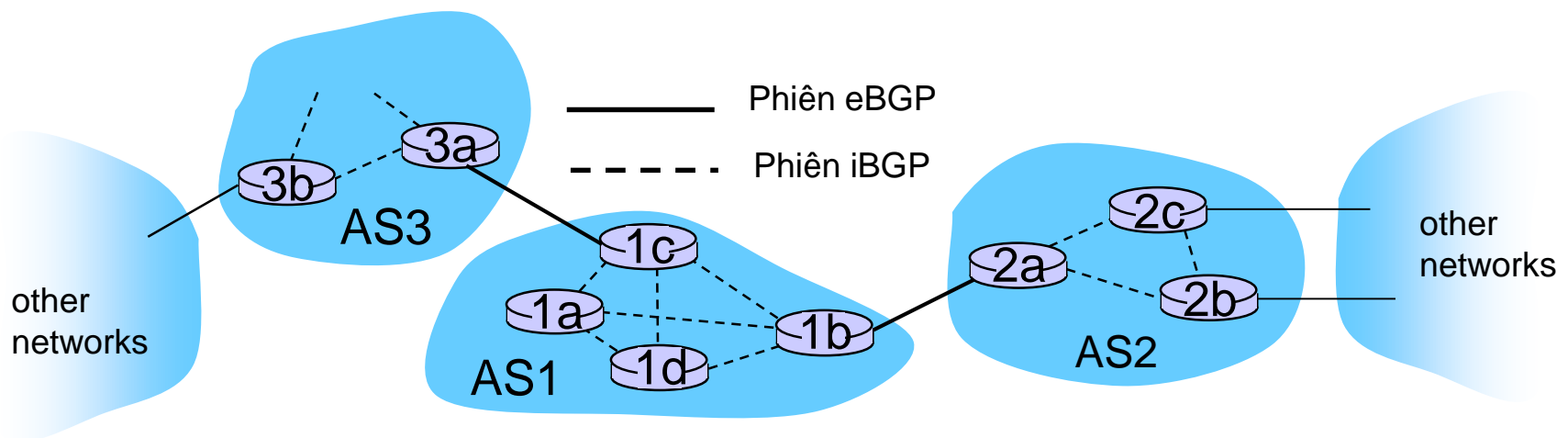
# BGP cơ bản

- ❖ **BGP session:** 2 router BGP (“cặp”) trao đổi các thông điệp BGP:
  - Quảng cáo **các đường đi** đến các tiền tố (prefixes) mạng đích khác nhau (Giao thức “path vector”)
  - Được trao đổi trên các kết nối semi-permanent TCP
- ❖ Khi AS3 quảng cáo 1 prefix tới AS1:
  - AS3 **hứa** là nó sẽ chuyển các datagram tới prefix đó
  - AS3 có thể tổng hợp các prefixe trong quảng cáo của nó



# BGP cơ bản: phân phối thông tin đường đi

- ❖ Sử dụng phiên eBGP giữa 3a và 1c, AS3 gửi prefix về thông tin khả năng chạm (prefix reachability info) tới AS1.
  - 1c sau đó có thể dùng iBGP phân phối thông tin prefix mới tới tất cả các router trong AS1
  - 1b sau đó có thể quảng cáo lại thông tin về khả năng chạm mới tới AS2 trên phiên eBGP từ 1b-tới-2a
- ❖ Khi router học prefix mới, thì nó sẽ tạo mục cho prefix đó trong bảng forwarding của nó.



# Các thuộc tính đường đi và các route BGP

---

- ❖ prefix được quảng cáo bao gồm các thuộc tính BGP
  - prefix + attributes = “route”
- ❖ 2 thuộc tính quan trọng:
  - **AS-PATH**: chứa các AS mà thông qua đó quảng cáo prefix vừa đi qua: ví dụ: AS 67, AS 17
  - **NEXT-HOP**: chỉ ra router bên trong AS cụ thể tới AS hop kế tiếp. (có thể có nhiều đường kết nối từ AS hiện tại tới AS hop kế tiếp)
- ❖ router gateway mà nhận quảng cáo route sử dụng **chính sách quan trọng (import policy)** để chấp nhận/từ chối
  - Ví dụ: không bao giờ đi qua AS x
  - Định tuyến **dựa trên chính (sách policy-based routing)**

# Sự lựa chọn BGP route

- ❖ router có thể học hơn 1 route tới AS đích, chọn route dựa trên:
  1. Thuộc tính giá trị ưu tiên cục bộ: quyết định chính sách
  2. AS-PATH ngắn nhất
  3. NEXT-HOP router gần nhất: định tuyến hot potato
  4. Tiêu chuẩn bổ sung

# Các thông điệp BGP

- ❖ Các thông điệp BGP được trao đổi giữa các peer trên kết nối TCP
- ❖ Các thông điệp BGP:
  - **OPEN**: mở kết nối TCP đến peer và chứng thực người gửi
  - **UPDATE**: quảng cáo đường đi mới(hoặc là lấy lại cái cũ)
  - **KEEPALIVE**: giữ kết nối sống khi các UPDATES không được cập nhật; còn gọi là yêu cầu OPEN các ACK
  - **NOTIFICATION**: thông báo các lỗi trong thông điệp trước đó; cũng được sử dụng để đóng kết nối

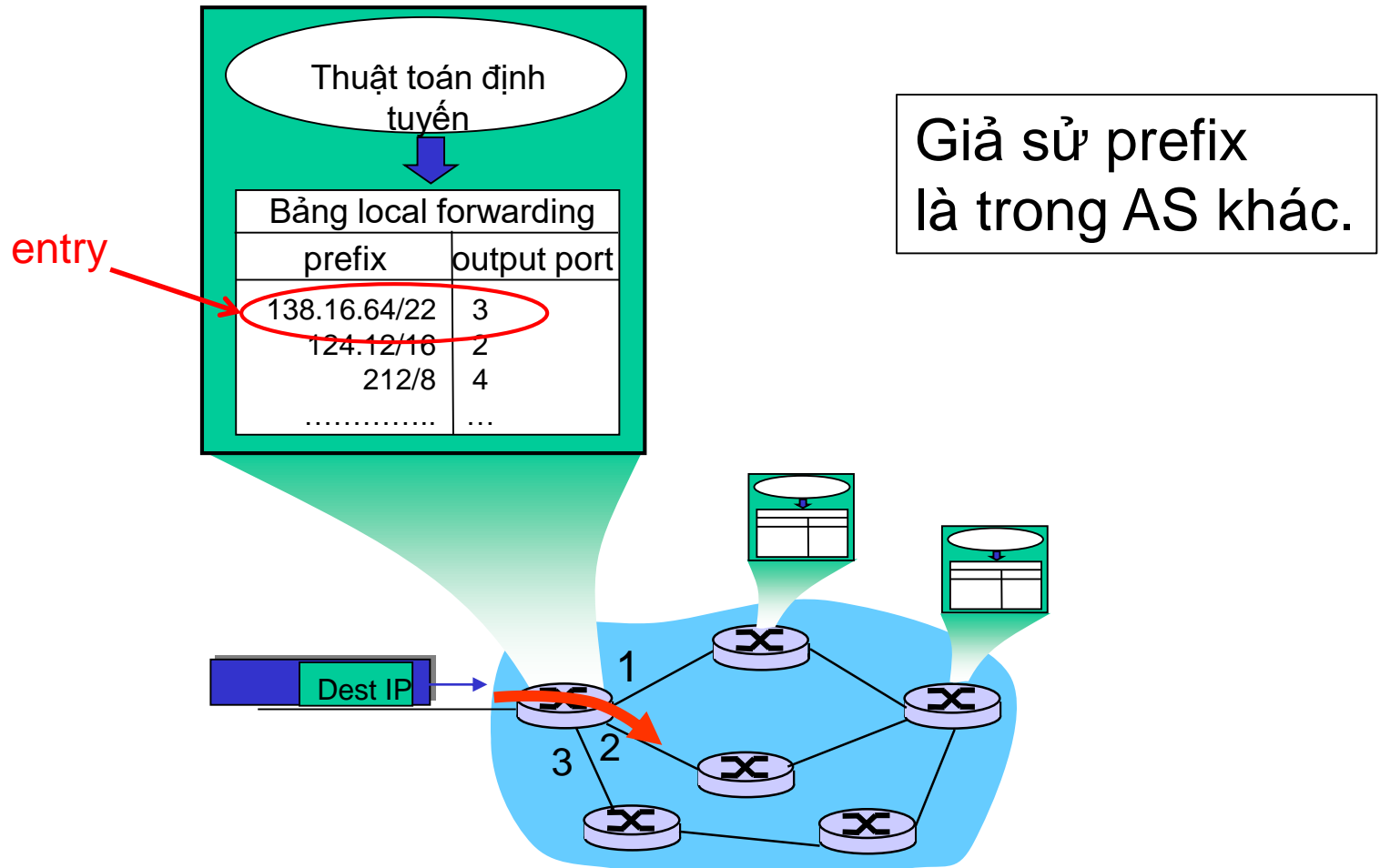


Đặt các giao thức định tuyến lại với nhau:

Làm thế nào một entry đưa vào trong bảng forwarding của 1 router?

- ❖ Câu trả lời rất phức tạp!
- ❖ Buộc các định tuyến phân cấp (mục 4.5.3) lại với nhau với BGP (4.6.3) và OSPF (4.6.2).
- ❖ Cung cấp tổng quan về BGP!

# Làm thế nào để entry được đưa vào bảng forwarding?

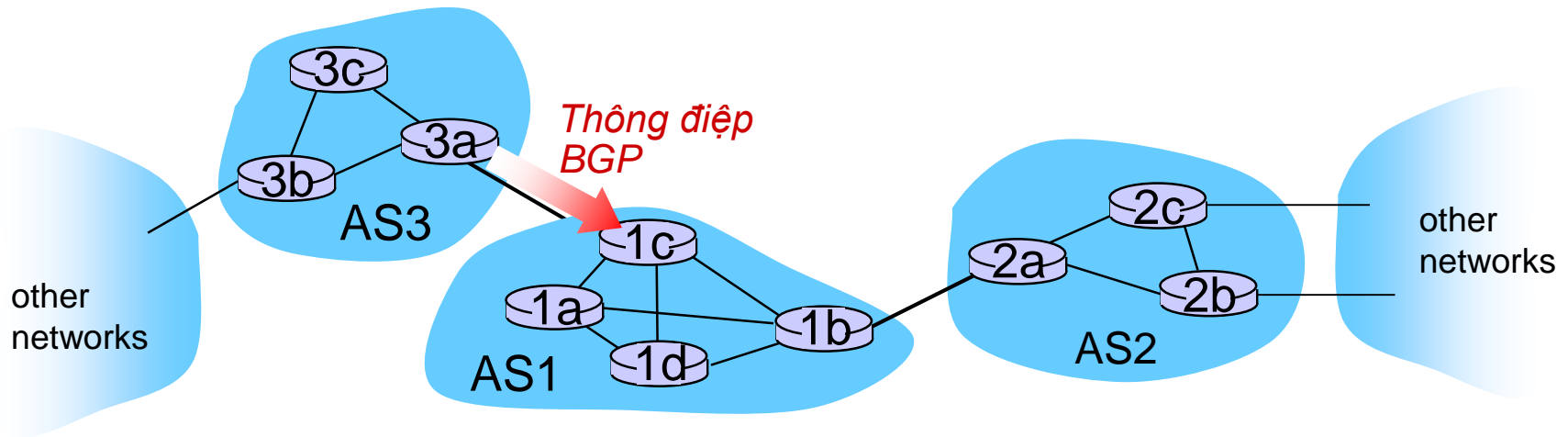


# Làm thế nào để entry được đưa vào bảng forwarding?

---

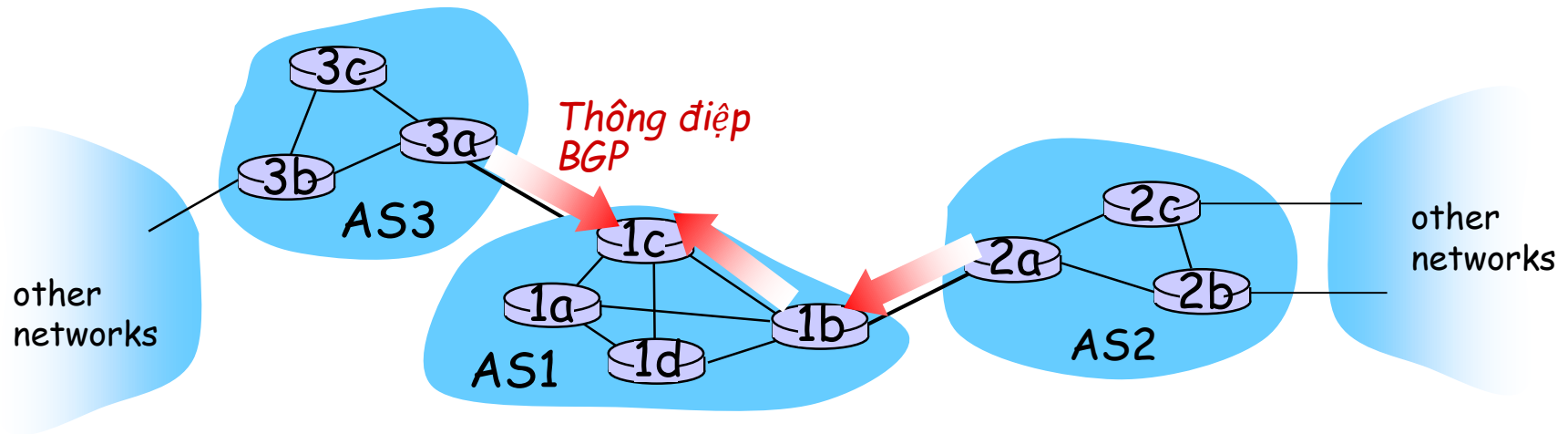
1. Router nhận thức được prefix
2. Router xác định port ra (output port) cho prefix
3. Router đưa cổng cho prefix đó vào trong bảng forwarding

# Router nhận thức được prefix



- ❖ Thông điệp BGP chứa "các route"
- ❖ "route" là 1 prefix và các thuộc tính: 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 được nhiều route



- ❖ Router có thể nhận được nhiều route cho cùng prefix
- ❖ Phải chọn 1 route

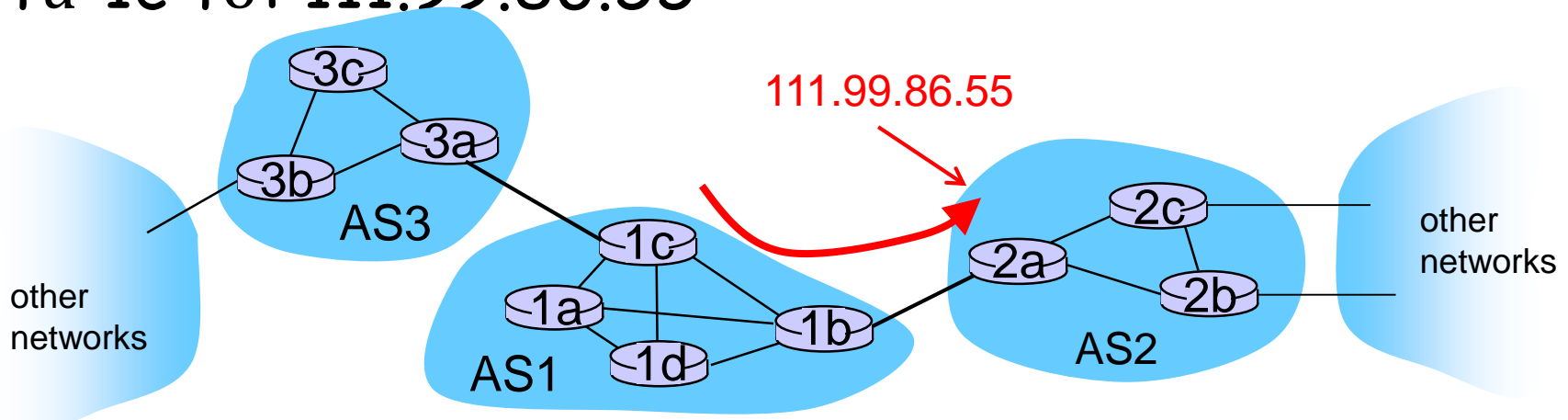
# Chọn route BGP tốt nhất tới prefix

- ❖ 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
- ❖ Điều gì sẽ xảy ra nếu có mối ràng buộc?  
Chúng ta sẽ bàn sau!

Chọn

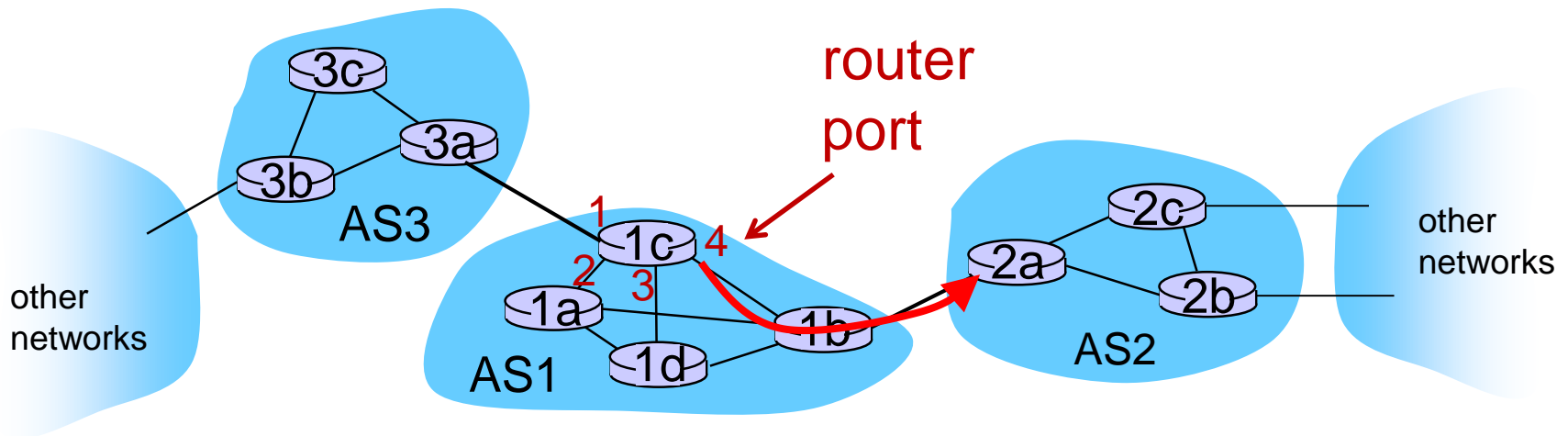
# Tìm route nội bộ tốt nhất đến route BGP

- ❖ Dùng thuộc tính NEXT-HOP của route được lựa chọn
  - Thuộc tính NEXT-HOP của Route là địa chỉ IP của interface của router đó, cái mà bắt đầu AS PATH đó.
- ❖ Ví dụ:
  - ❖ AS-PATH: AS2 AS17 ; NEXT-HOP: 111.99.86.55
- ❖ Router sử dụng OSPF để tìm ra đường đi ngắn nhất từ 1c tới 111.99.86.55



# Router xác định port cho route

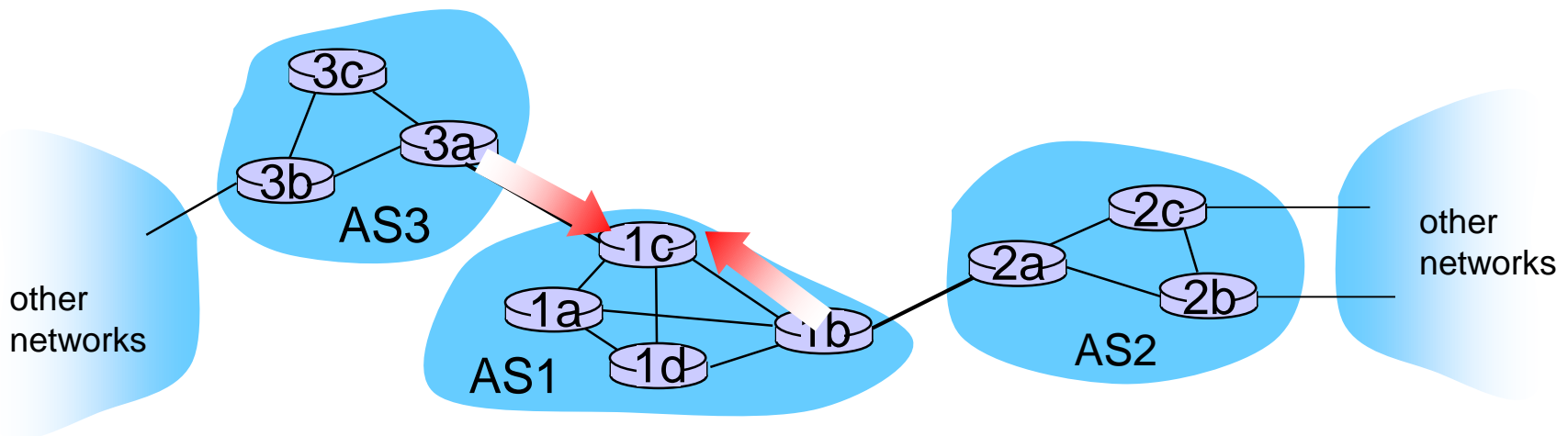
- ❖ Xác định port theo đường đi OSPF ngắn nhất
- ❖ Thêm prefix-port entry vào bảng forwarding của nó:
  - (138.16.64/22 , port 4)





# Định tuyến Hot Potato

- ❖ Giả sử có 2 hoặc nhiều route liên tuyến tốt nhất (best inter-routes).
- ❖ Sau đó chọn route với NEXT-HOP gần nhất
  - Dùng OSPF để xác định cổng nào là gần nhất
  - Hỏi: từ 1c, chọn AS3 AS131 hoặc AS2 AS17?
  - Đáp: route AS3 AS201 vì nó gần hơn



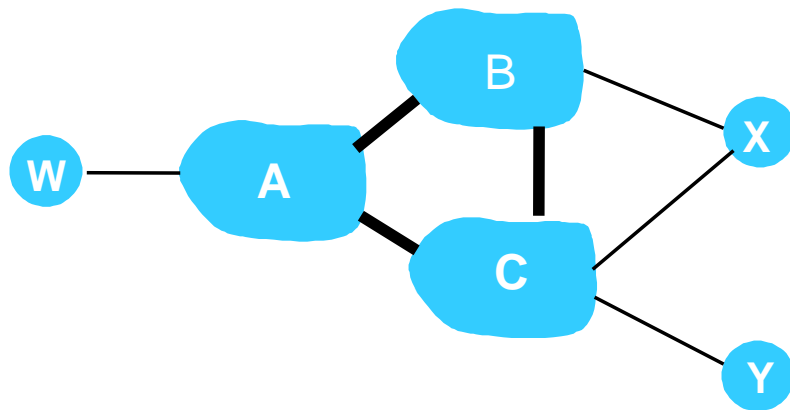
# Làm thế nào để entry được đưa vào bảng forwarding?


---


## Tóm tắt

1. Router có nhận thức về prefix
  - Thông qua các quảng cáo route BGP từ các router khác
2. Xác định port ra của router cho prefix đó
  - Dùng route BGP vừa chọn để tìm ra route liên AS (inter-AS ) tốt nhất
  - Dùng OSPF để tìm ra route tốt nhất trong nội bộ AS (intra-AS ) cái mà dẫn đến route liên AS tốt nhất (inter-AS route)
  - Router xác định port của router cho route tốt nhất đó
3. Đưa prefix-port entry vào trong bảng forwarding

# Chính sách BGP routing

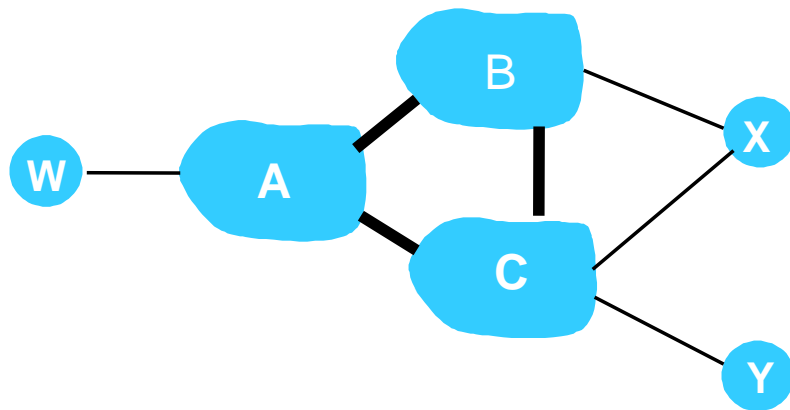


Ký hiệu:  Mạng của nhà cung cấp

 Mạng của khách hàng

- ❖ A,B,C là các nhà cung cấp mạng
- ❖ X,W,Y là khách hàng (của nhà cung cấp mạng)
- ❖ X là *dual-homed*: được kết nối vào 2 mạng
  - X không muốn có tuyến đường từ B thông qua X tới C
  - .. Vì vậy X sẽ không quảng cáo cho B tuyến đường tới C

# Chính sách BGP routing (2)



Ký hiệu:  Mạng của nhà cung cấp  
 Mạng của khách hàng

- ❖ A quảng cáo đường đi AW cho B
- ❖ B quảng cáo đường đi BAW cho X
- ❖ B có nên quảng cáo đường đi BAW cho C hay không?
  - Không nên! B không nhận được "thu nhập" cho việc định tuyến CBAW vì cả W và C không phải là khách hàng của B
  - B muốn bắt buộc C đi tới w thông A
  - B *chỉ* muốn dẫn đường đi/đến các khách hàng của nó!

# Tại sao phải định tuyến Intra-, Inter-AS khác nhau?

---

## *Chính sách:*

- ❖ inter-AS: người quản trị muốn kiểm soát cách mà traffic của nó được định tuyến, ai mà định tuyến thông qua mạng của nó.
- ❖ intra-AS: 1 người quản trị, vì vậy không cần các quyết định chính sách

## *Linh hoạt:*

- ❖ Định tuyến phân cấp làm giảm kích thước bảng định tuyến, giảm lưu lượng cập nhật

## *Hiệu suất:*

- ❖ intra-AS: có thể tập trung vào hiệu suất
- ❖ inter-AS: chính sách quan trọng hơn hiệu suất

# Chương 4: Nội dung

## 4.1 Giới thiệu

## 4.2 virtual circuit network và datagram network

## 4.3 Cấu trúc bên trong router

## 4.4 IP: Internet Protocol

- Định dạng datagram
- IPv4 addressing
- ICMP
- IPv6

## 4.5 các thuật toán routing

- link state
- distance vector
- hierarchical routing

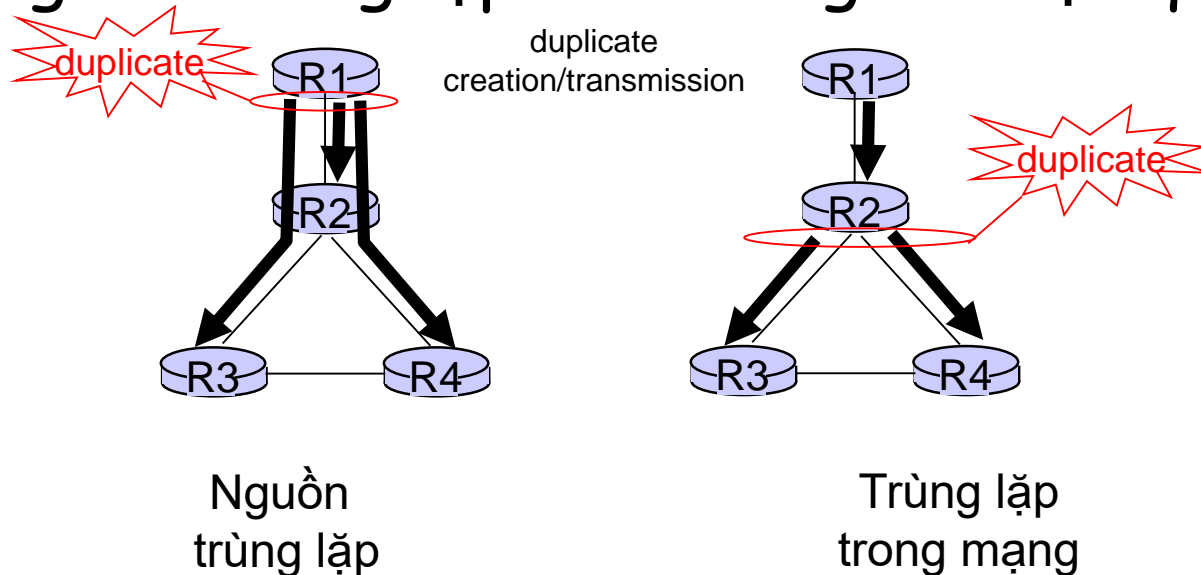
## 4.6 routing trong Internet

- RIP
- OSPF
- BGP

## 4.7 broadcast và multicast routing

# Broadcast routing

- ❖ Chuyển các packet từ nguồn tới tất cả các node khác
- ❖ Nguồn trùng lặp thì không có hiệu quả:



- ❖ Nguồn trùng lặp: làm sao xác định được địa chỉ người nhận?

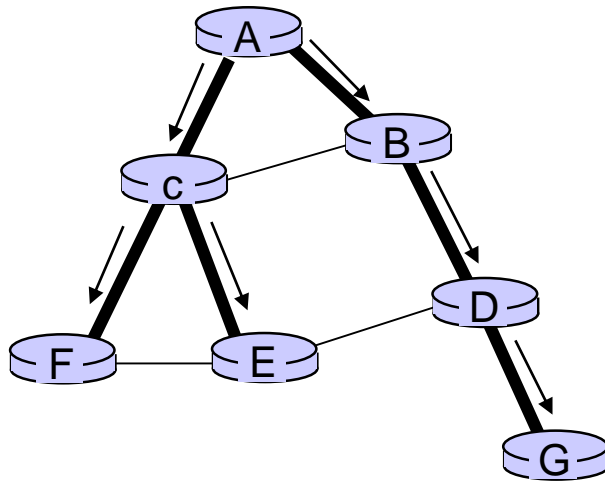
# Trùng lặp trong mạng

- ❖ *flooding*: khi node nhận được packet broadcast, nó gửi bản sao đến tất cả các neighbor
  - Vấn đề: lặp lại & bão broadcast
- ❖ *Flooding có điều khiển*: node chỉ broadcast packet nếu nó không gửi broadcast giống như vậy trước đó
  - node theo dõi các packet đã broadcast
  - Hoặc reverse path forwarding (RPF): chỉ chuyển các packet nếu nó đã đến trên đường đi ngắn nhất giữa node và nguồn
- ❖ *spanning tree*:
  - Không có các packet dư thừa được nhận tại bất cứ node nào

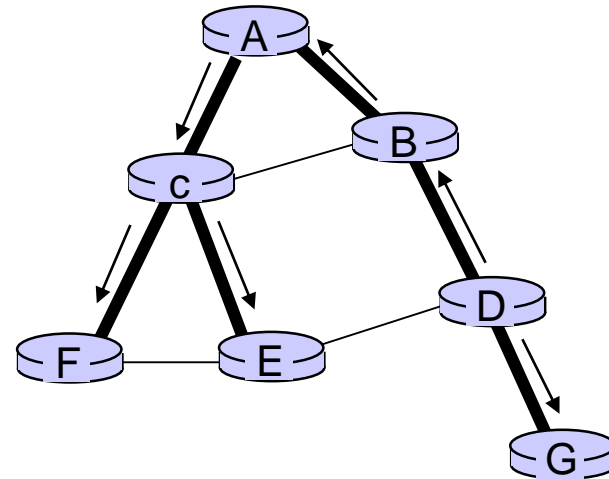


# Spanning tree

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



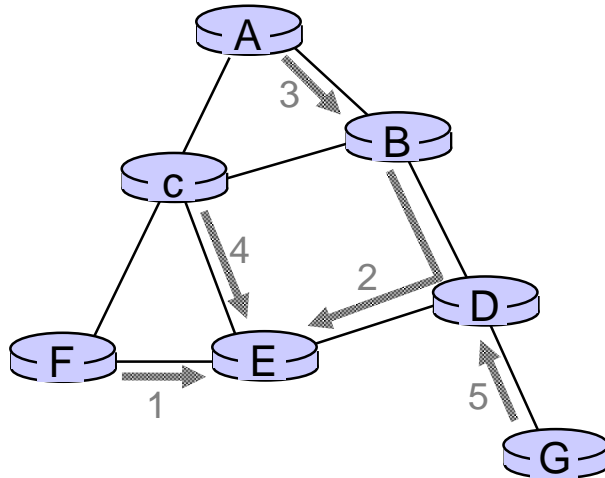
(a) broadcast được khởi tạo tại A



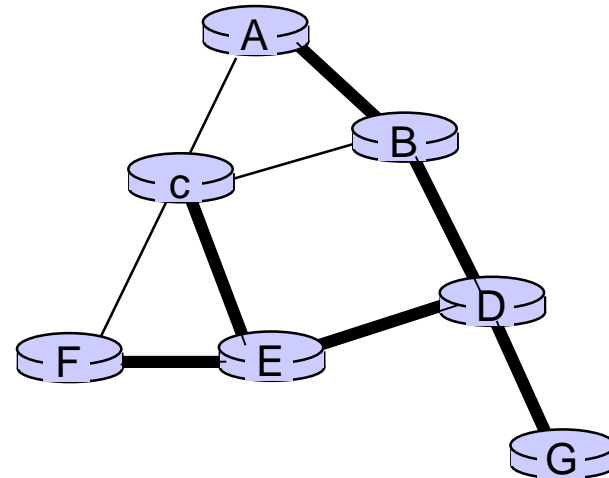
(b) broadcast được khởi tạo tại D

# Spanning tree: tạo

- ❖ Node trung tâm
- ❖ Mỗi node gửi thông điệp gia nhập unicast (unicast join message) đến node trung tâm
  - Thông điệp này được chuyển tiếp cho đến khi nó đến tại một node đã nằm trên spanning tree



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

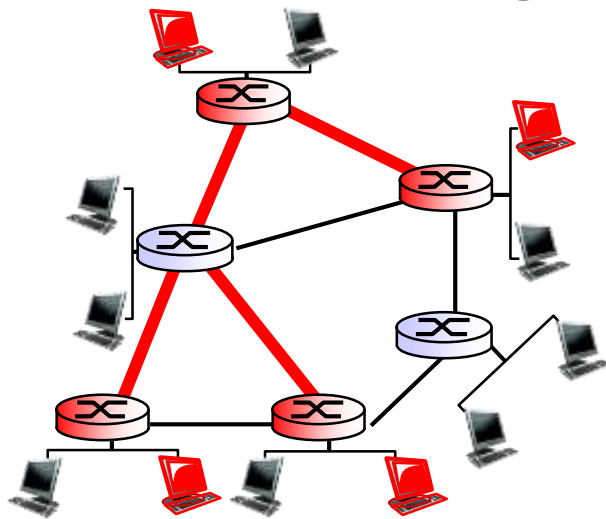


(b) spanning tree đã được xây dựng xong

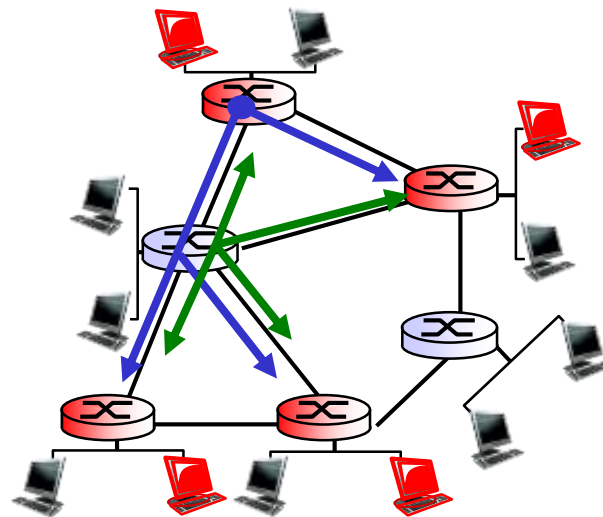
# Multicast routing: phát biểu vấn đề

**Mục tiêu:** tìm một cây (hoặc các cây) kết nối các router có các thành viên trong nhóm multicast

- ❖ **Cây (tree):** không phải tất cả các đường đi giữa các router đều được sử dụng
- ❖ **Cây chia sẻ (shared-tree):** cây giống nhau được sử dụng bởi các thành viên trong nhóm
- ❖ **Cây dựa trên nguồn (source-based):** cây khác nhau từ nơi gửi tới nơi nhận



Cây chia sẻ



Cây dựa trên nguồn

Ký hiệu



group member



not group member



router with a group member



router without group member

# Các cách tiếp cận để xây dựng các cây multicast

---

Các hướng tiếp cận:

- ❖ **Cây dựa trên nguồn (source-based tree):** một cây cho mỗi nguồn
  - Các cây đường đi ngắn nhất
  - Cây đường đi ngược
- ❖ **Cây chia sẻ nhóm:** nhóm dùng 1 cây
  - Mở rộng tối thiểu (Steiner)
  - Các cây dựa trên trung tâm (center-based trees)

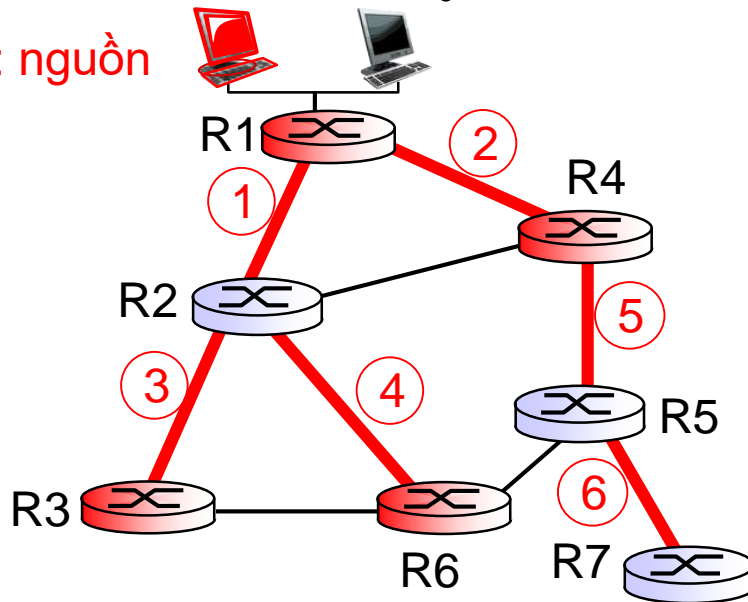
...tìm hiểu các cách tiếp cận cơ bản, sau đó các giao thức cụ thể áp dụng cho các hướng tiếp cận này

# Cây đường đi ngắn nhất

- ❖ Cây chuyển tiếp multicast (mcast forwarding tree): cây đường đi ngắn nhất dẫn đường từ nguồn tới tất cả các nơi nhận

- Thuật toán Dijkstra

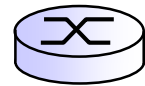
s: nguồn



Ký hiệu



router với các thành viên của nhóm đã được gắn vào



router không có các thành viên nào của nhóm được gắn vào



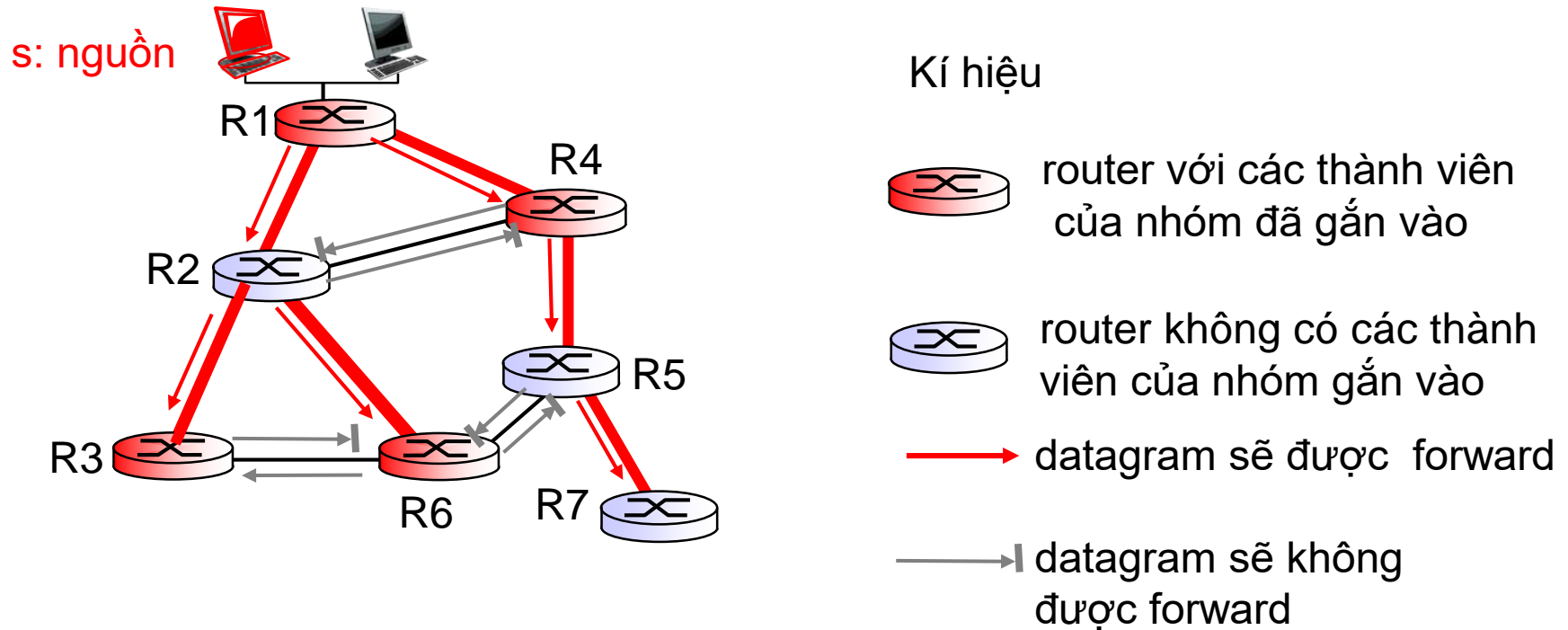
Đường kết nối được sử dụng cho forwarding, i chỉ thứ tự đường link được thêm vào bởi thuật toán

# Reverse path forwarding

- ❖ Dựa vào kiến thức của router của đường đi ngắn nhất unicast từ nó đến nơi gửi
- ❖ Mỗi router có cách xử lý forwarding đơn giản:

*if* (datagram multicast được nhận trên đường kết nối đến trên đường đi ngắn nhất kể từ trung tâm)  
    *then* flood datagram lên tất cả các kết nối ra  
    *else* bỏ qua datagram

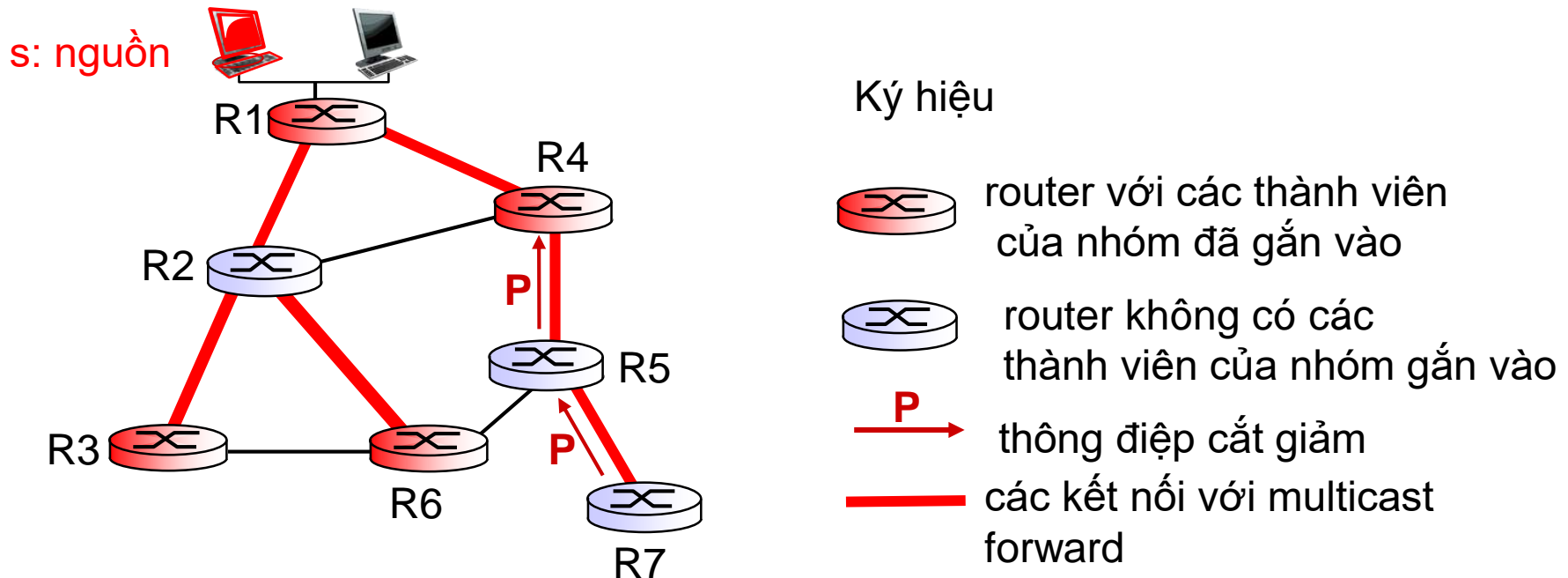
# Reverse path forwarding: ví dụ



- ❖ Kết quả là một cây SPT đảo ngược
  - Có thể là một lựa chọn không tốt với các kết nối không đồng bộ

# Reverse path forwarding: cắt giảm (pruning)

- ❖ Cây forwarding chứa các cây con không có các thành viên trong nhóm multicast
  - Không cần forward các datagram xuống cây con
  - “cắt giảm” các thông điệp được gửi lên trên bởi router không có các thành viên nhóm downstream





# Cây chia sẻ (Shared-tree): cây steiner

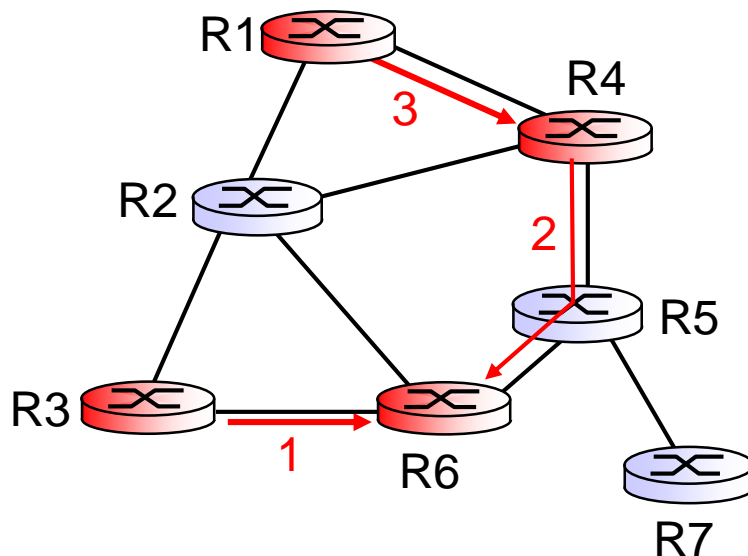
- ❖ **Cây steiner** : cây có chi phí thấp nhất kết nối đến tất cả các router với các thành viên nhóm đã được gắn vào
- ❖ Vấn đề là NP-complete
- ❖ Các heuristics rất tốt tồn tại
- ❖ Không sử dụng trong thực tế:
  - Độ phức tạp tính toán
  - Cần thông tin về về toàn bộ mạng
  - monolithic: chạy lại mỗi khi router cần gia nhập/rời khỏi

# Cây dựa vào trung tâm (Center-based trees)




- ❖ Một cây truyền nhận được chia sẽ cho tất cả
- ❖ Một router được xác định như là “*trung tâm*” của cây
- ❖ Để gia nhập:
  - router biên (edge ) gửi thông điệp gia nhập unicast đến router trung tâm
  - *Thông điệp gia nhập (join-msg)* “được xử lý” bởi các router trung gian và chuyển đến các router trung tâm
  - *Thông điệp gia nhập*, hoặc đến nhánh cây của trung tâm này, hoặc đến ngay trung tâm
  - Đường đi được thực hiện bởi *thông điệp gia nhập* trở thành nhánh cây mới của router này

# Cây dựa vào trung tâm: ví dụ

Giả sử R6 được chọn là trung tâm:



## LEGEND

-  router với các thành viên của nhóm đã gắn vào
-  router không có các thành viên của nhóm gắn vào
-  thứ tự đường đi trong ấy các thông điệp gia nhập đã sinh ra

# Internet Multicasting Routing: DVMRP

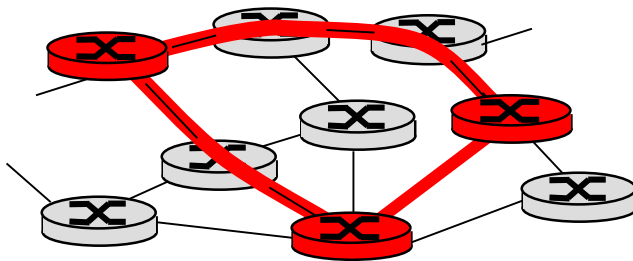
- ❖ **DVMRP**: distance vector multicast routing protocol, RFC1075
- ❖ **flood và prune**: forwarding đường đi ngược (reverse path forwarding), cây dựa vào nguồn (source-based tree)
  - Cây RPF dựa trên các bảng định tuyến của DVMRP của nó được xây dựng bởi truyền thông các router DVMRP
  - Không có các giả thuyết về unicast bên dưới
  - datagram ban đầu đến nhóm multicast được flood mọi nơi thông qua RPF
  - Các router không phải nhóm: gửi thông điệp cắt giảm lên trên

# DVMRP: tiếp tục...

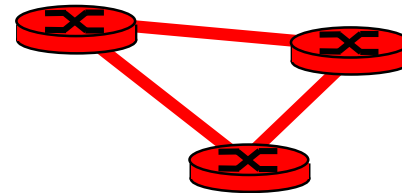
- ❖ *Trạng thái mềm:* router DVMRP chu kỳ (1 phút) “quên” các nhánh cây bị cắt giảm:
  - Dữ liệu mcast một lần nữa đổ xuống các nhánh không được cắt giảm
  - Router phía dưới: cắt giảm lần nữa hoặc tiếp tục nhận dữ liệu
- ❖ Các router có thể nhanh chóng tái cắt giảm
  - Gia nhập IGMP tại các lá
- ❖ Còn lại
  - Thường được thực hiện trong router thương mại

# Tunneling

**Q:** làm cách nào để kết nối "các đảo" của các router multicast trong một "biển" của các router unicast?



Cấu trúc vật lý



Cấu trúc logic

- ❖ datagram multicast được đóng gói trong datagram "thông thường" (không có multicast)
- ❖ datagram IP thông thường được gửi thông qua "đường hầm" ("tunnel") và qua IP unicast đến router multicast nhận (xem lại IPv6 bên trong đường hầm IPv4)
- ❖ router mcast nhận mở gói để lấy datagram multicast

# PIM: Protocol Independent Multicast

- ❖ Không phụ thuộc vào bất kỳ thuật toán định tuyến unicast bên dưới nào (underlying unicast routing algorithm) (làm việc với tất cả)

- ❖ 2 ngữ cảnh phân phối multicast khác nhau:

## *Dày đặc:*

- ❖ Các thành viên nhóm đóng gói dày đặc, trong khoảng cách "gần".
- ❖ Bảng thông dư thừa

## *Thưa thớt:*

- ❖ Số lượng các mạng với các thành viên nhóm ít
- ❖ Các thành viên nhóm "được phân bố thưa thớt"
- ❖ Bảng thông không dư thừa

# Kết quả của sự phân chia thừa thớt-dày đặc:

---

## *Dày đặc*

- ❖ Nhóm các thành viên bởi các router được giả định cho đến khi các router cắt giảm thực sự
- ❖ Kiến trúc *data-driven* trên cây mcast (ví dụ RPF)
- ❖ Bảng thông và router không thuộc nhóm xử lý phung phí

## *Thừa thớt:*

- ❖ Không có thành viên cho đến khi các router thực sự gia nhập
- ❖ Kiến trúc *receiver-driven* của cây mcast (ví dụ cây dựa vào trung tâm)
- ❖ Bảng thông và router không thuộc nhóm xử lý vừa phải



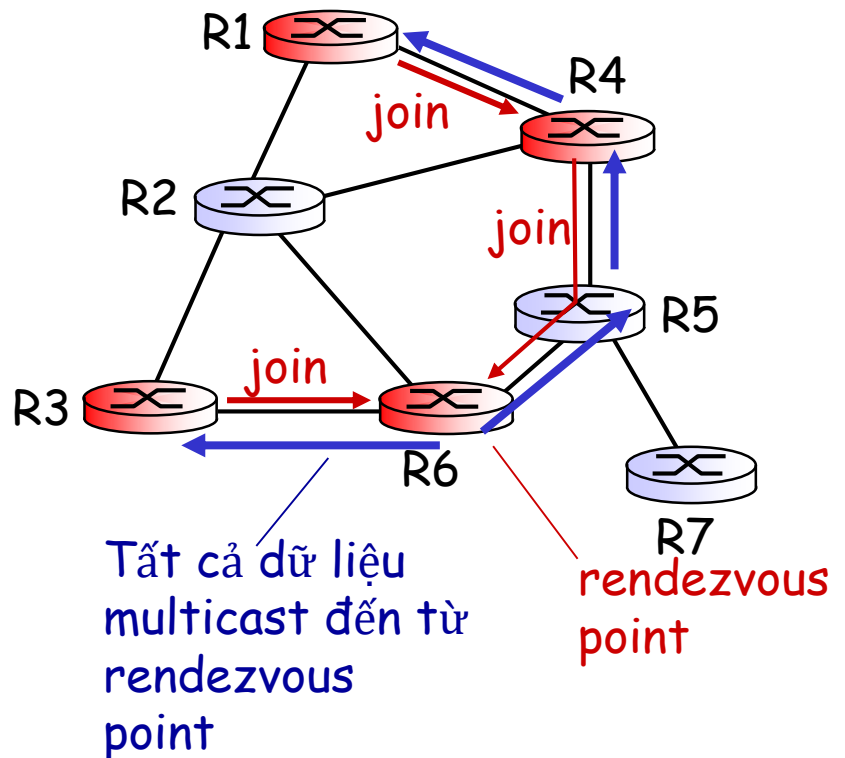
# PIM- kiểu dày đặc

**flood-and-prune RPF**: tương tự như DVMRP nhưng...

- ❖ Giao thức bên dưới cung cấp thông tin RPF cho datagram đến
- ❖ flood xuống dưới (downstream) ít phức tạp (ít hiệu quả) hơn so với DVMRP giảm sự phụ thuộc vào thuật toán định tuyến cơ bản
- ❖ Có cơ chế giao thức cho router để phát hiện có phải là router ở node lá (leaf-node)

# PIM - kiểu thưa thớt

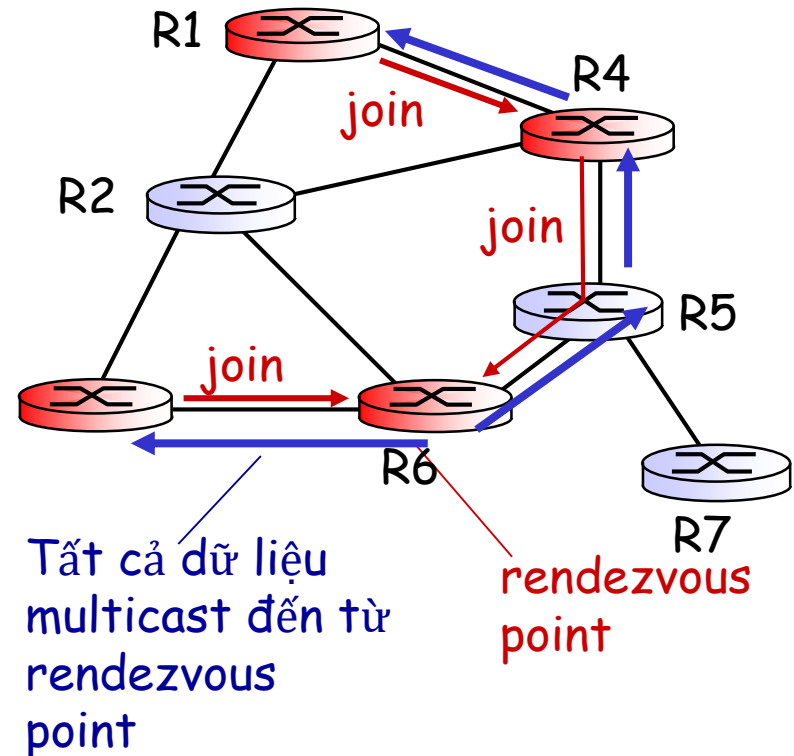
- ❖ Tiếp cận dựa vào trung tâm (center-based)
- ❖ router gửi thông điệp *gia nhập* (join msg) đến rendezvous point (RP)
  - Các router trung gian cập nhật trạng thái và forward thông điệp *gia nhập*
- ❖ Sau khi gia nhập thông qua RP, router có thể chuyển sang cây xác định nguồn (source-specific tree)
  - Hiệu suất tăng: ít tập trung, các đường đi ngắn hơn



# PIM - kiểu thưa thớt

## (Các) Bên gửi:

- ❖ Dữ liệu unicast đến RP, RP phân phối xuống cây có nút gốc là RP
- ❖ RP có thể mở rộng cây multicast dòng lên đến  $R_3$  nguồn
- ❖ RP có thể gửi thông điệp dừng (*stop msg*) nếu không có bên nhận nào được gắn vào
  - “không có ai đang lắng nghe!”



# Chapter 4: Hoàn thành!

## 4.1 Giới thiệu

## 4.2 Mạng mạch ảo và mạng datagram (virtual circuit and datagram networks)

## 4.3 bên trong router

## 4.4 IP: Internet Protocol

- datagram định dạng, định địa chỉ IPv4, ICMP, IPv6

## 4.5 các thuật toán định tuyến

- link state, distance vector, định tuyến phân cấp

## 4.6 định tuyến trong Internet

- RIP, OSPF, BGP

## 4.7 broadcast and multicast routing

## ❖ Hiểu về các nguyên tắc đằng sau các dịch vụ tầng network:

- Các mô hình dịch vụ tầng network, so sánh cách mà router forwarding và routing dữ liệu, định tuyến(chọn đường đi), broadcast, multicast

## ❖ Hiện thực trên Internet