BAN HỌC TẬP
ĐOÀN KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

Ôn tập cuối kỳ I
Nhập môn mạng máy tính

- **Nguyễn Thành Nhân**
  MMTT2022.2
- **Nguyễn Phi Học**
  KTPM2022.1
- **Lê Hoàng Vũ**
  MMTT2022.3

Nội dung ôn tập

01. Giới thiệu

02. Tầng ứng dụng

03. Tầng vận chuyển

04. Tầng mạng

05. Tầng liên kết

06. Giải bài tập

III.TẦNG VẬN CHUYỂN

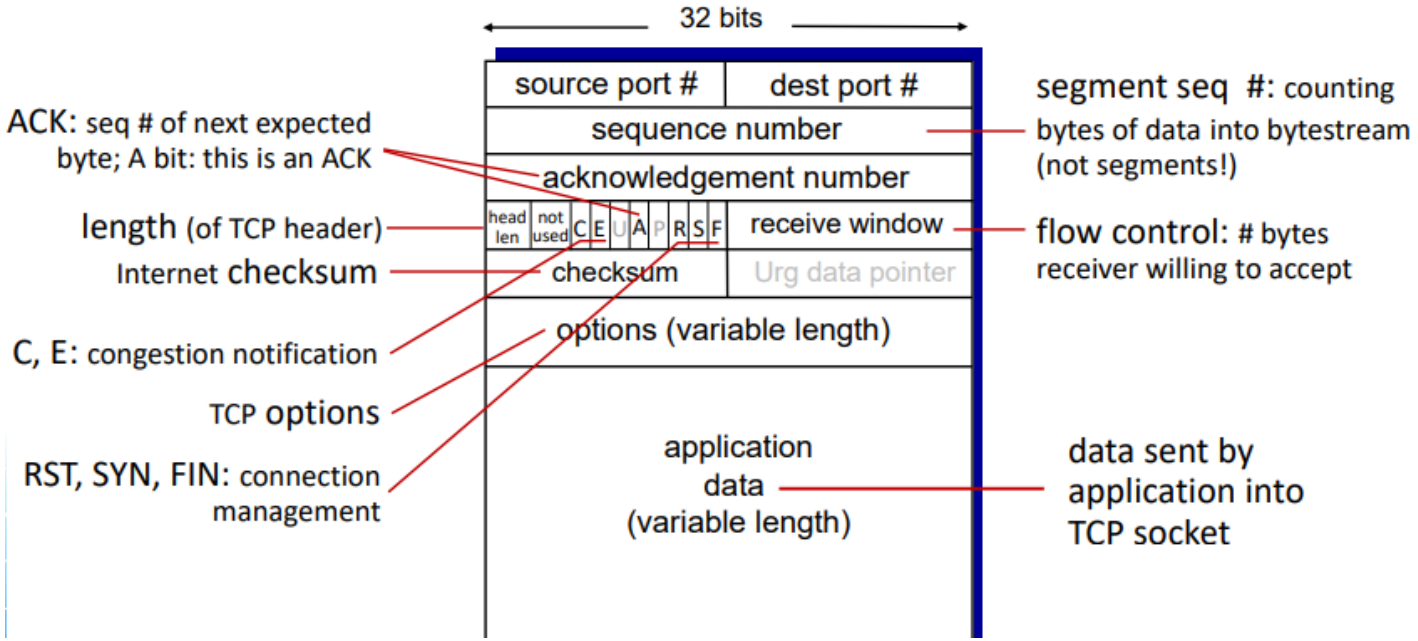
1.1.TCP là gì?

- TCP(Transmission Control Protocol) là một giao thức mạng quan trọng được sử dụng trong việc truyền dữ liệu qua mạng.

- TCP đảm bảo rằng các byte được truyền theo thứ tự mà chúng được gửi mà không có lỗi hoặc thiếu sót nào.

III.TẦNG VẬN CHUYỂN

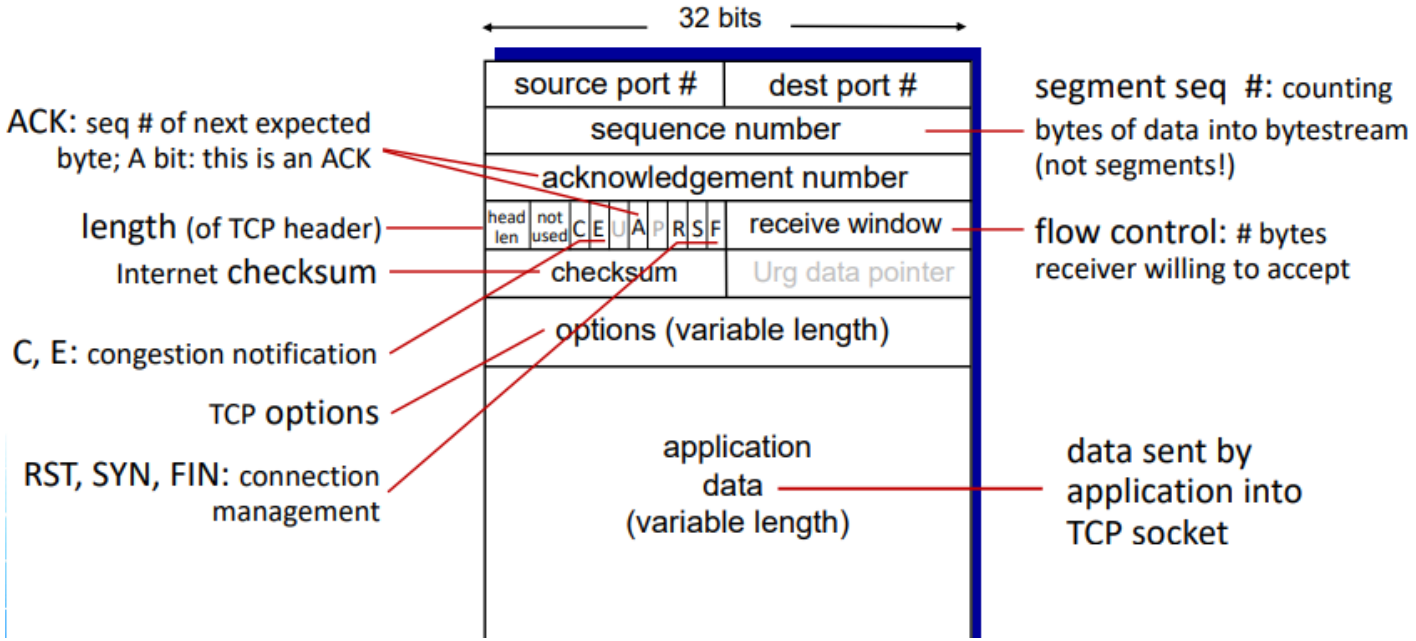
1.2.Cấu trúc của TCP



- Source port (16 bit): Số cổng của thiết bị gửi.
- Destination port (16 bit): Số cổng của thiết bị nhận.
- Sequence number (32 bit): Dùng để đánh số thứ tự gói tin.
- Acknowledgment number (32 bit): Dùng để báo đã nhận được gói tin nào và mong nhận được byte mang số thứ tự nào tiếp theo.
- Head len(4 bit): Cho biết toàn bộ header dài bao nhiêu tính theo đơn vị word (1 Word = 4 byte).

III.TẦNG VẬN CHUYỂN

1.2.Cấu trúc của TCP



- Urgent pointer (16 bit): Sử dụng trong trường hợp cần ưu tiên dữ liệu.
- Options (tối đa 32 bit): Cho phép thêm vào TCP các tính năng khác.

- Flags (9 bit): Được sử dụng để thiết lập kết nối, gửi dữ liệu và chấm dứt kết nối.

+ URG: Ưu tiên dữ liệu này hơn các dữ liệu khác.

+ ACK: Được sử dụng để xác nhận.

+ PSH: Segment yêu cầu chức năng push.

+ RST: Thiết lập lại kết nối.

+ SYN: Được sử dụng để đặt số thứ tự ban đầu.

+ FIN: Kết thúc kết nối TCP.

- Windows (16 bit): Số lượng byte được thiết bị sẵn sàng tiếp nhận.
- Checksum (16 bit): Kiểm tra lỗi của toàn bộ TCP segment.

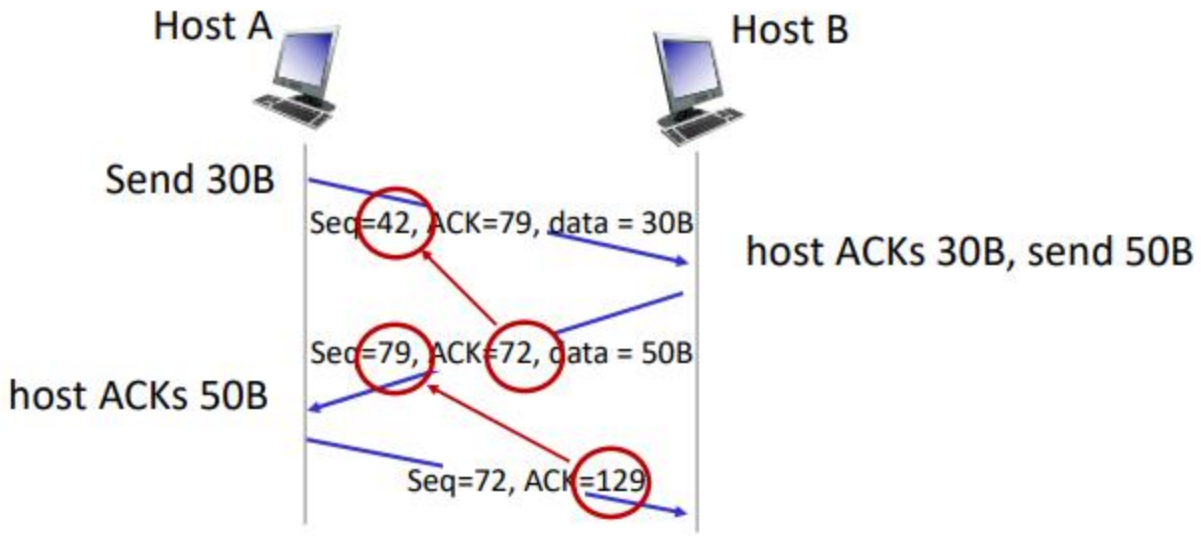
III.TẦNG VẬN CHUYỂN

1.2.Cấu trúc của TCP

```
Transmission Control Protocol, Src Port: 1029, Dst Port: 443, Seq: 1, Ack: 1, Len: 1
    Source Port: 1029
    Destination Port: 443
    [Stream index: 1]
    [TCP Segment Len: 1]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 2185425389
    [Next Sequence Number: 2    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 3497536912
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x010 (ACK)
        000. .... .... = Reserved: Not set
        ...0 .... .... = Nonce: Not set
        .... 0... .... = Congestion Window Reduced (CWR): Not set
        .... .0.. .... = ECN-Echo: Not set
        .... ..0. .... = Urgent: Not set
        .... ...1 .... = Acknowledgment: Set
        .... .... 0... = Push: Not set
        .... .... .0.. = Reset: Not set
        .... .... ..0. = Syn: Not set
        .... .... ...0 = Fin: Not set
        [TCP Flags: ·····A····]
    Window: 508
    [Calculated window size: 508]
    [Window size scaling factor: -1 (unknown)]
    Checksum: 0x641f [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > [SEQ/ACK analysis]
  > [Timestamps]
```

III.TẦNG VẬN CHUYỂN

1.3.TCP truyền dữ liệu đáng tin cậy



-Truyền theo thứ tự: Sequence number, ACK

-Truyền lại:

+Segment lỗi: ACK

+Segment mất: timeout

III.TẦNG VẬN CHUYỂN

1.3.TCP truyền dữ liệu đáng tin cậy

$$EstimatedRTT = (1- \alpha)*EstimatedRTT + \alpha*SampleRTT$$

$$TimeoutInterval = EstimatedRTT + 4*DevRTT$$

estimated RTT

"safety margin"

$$DevRTT = (1-\beta)*DevRTT + \beta*|SampleRTT-EstimatedRTT|$$

-Trong đó:

+ EstimatedRTT: RTT ước tính

+ SampleRTT: RTT mẫu

+DevRTT: biên độ an toàn

+TimeoutInterval: khoảng thời gian chờ

+Giá trị tiêu biểu : α=0.125

+Thông thường β=0.25

III.TẦNG VẬN CHUYỂN

1.3.TCP truyền dữ liệu đáng tin cậy

- Sự kiện tại bên gửi: khi nhận dữ liệu từ tầng application

+ Tạo segment với STT seq #

+seq #: STT của byte đầu tiên trong segment

+Bắt đầu tính thời gian với segment cũ nhất

- Sự kiện tại bên gửi: khi timeout

+ Gửi lại segment bị timeout

+Bắt đầu tính lại thời gian

- Sự kiện tại bên gửi: khi nhận được ACK

+ Nếu ACK xác nhận cho các segment chưa được ACK.

+Cập nhật trạng thái .

+tính thời gian với các segment chưa được ACK.

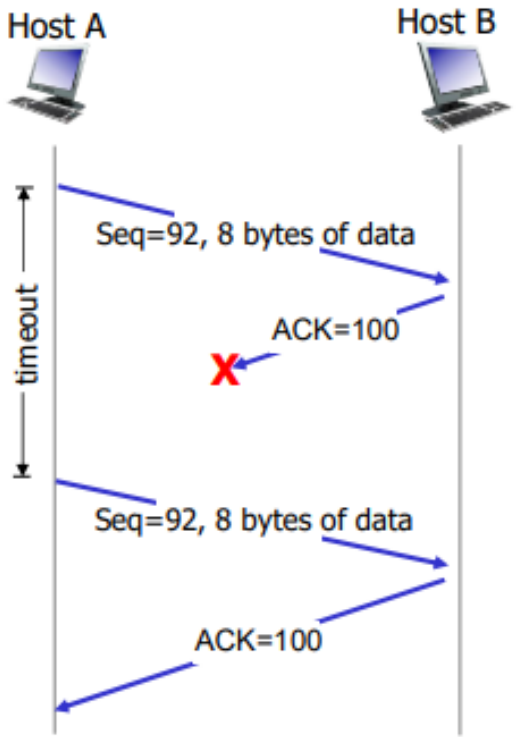
III.TẦNG VẬN CHUYỂN

1.3.TCP truyền dữ liệu đáng tin cậy

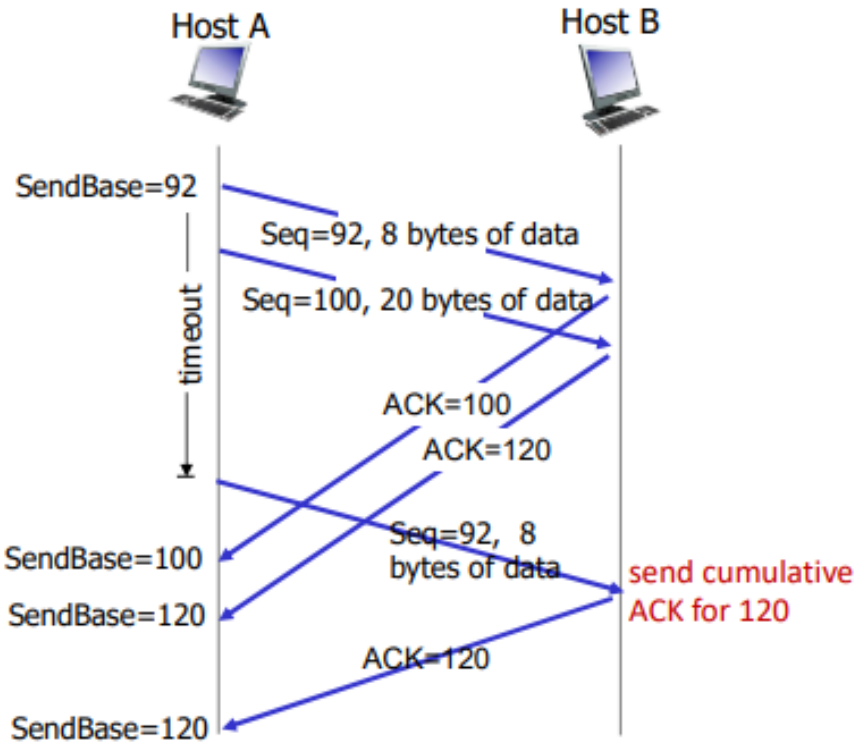
Sự kiện tại bên nhận	*Hành động*
Nhận được segment đúng với STT đang chờ. Tất cả segment trước đó đã được ACK.	Chờ segment kế tiếp trong 500ms, nếu không nhận được thì sẽ ACK.
Nhận được segment đúng với STT đang chờ, một segment chưa được ACK.	Gửi ACK tích lũy để ACK cho 2 segment.
Nhận được segment không đúng thứ tự (STT cao hơn).	Gửi ACK trùng ngay lập tức, để chỉ cho bên gửi biết đang chờ segment nào.
Nhận được segment trong khoảng bị trống (giữa STT đang chờ và STT nhận được trước đó).	ACK ngay lập tức cho segment có thứ tự thấp nhất trong khoảng bị trống

III.TẦNG VẬN CHUYỂN

1.3.TCP truyền dữ liệu đáng tin cậy



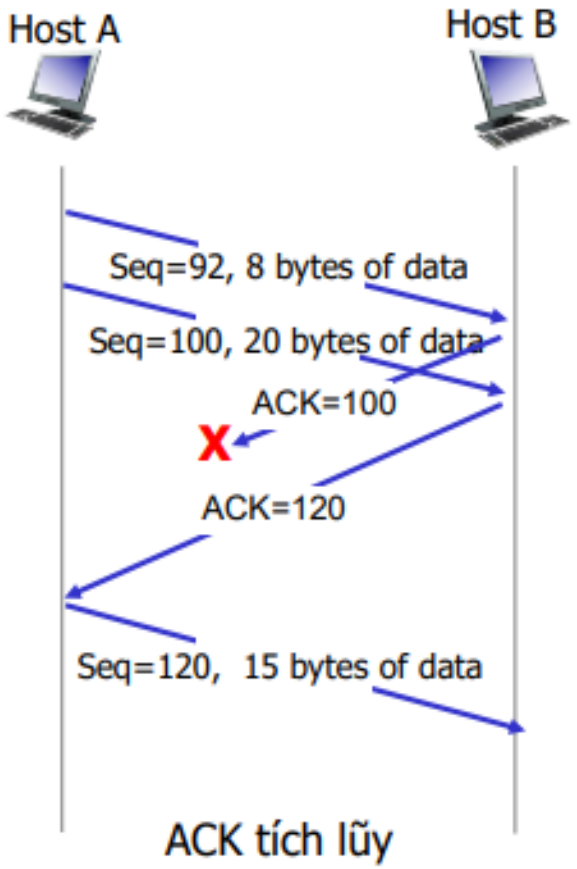
Mất ACK

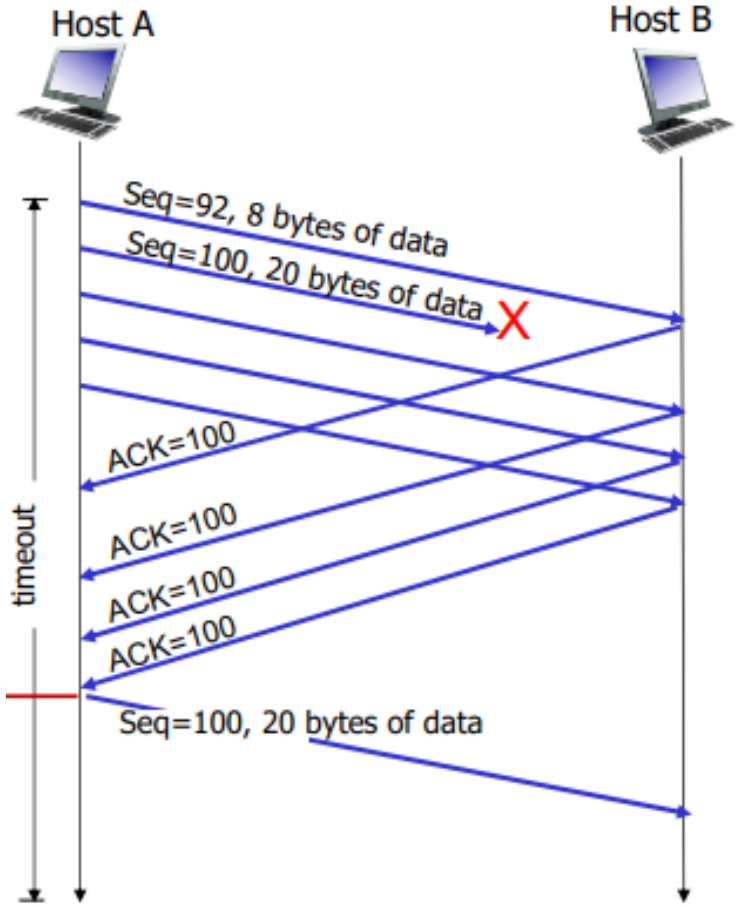


timeout

III.TẦNG VẬN CHUYỂN

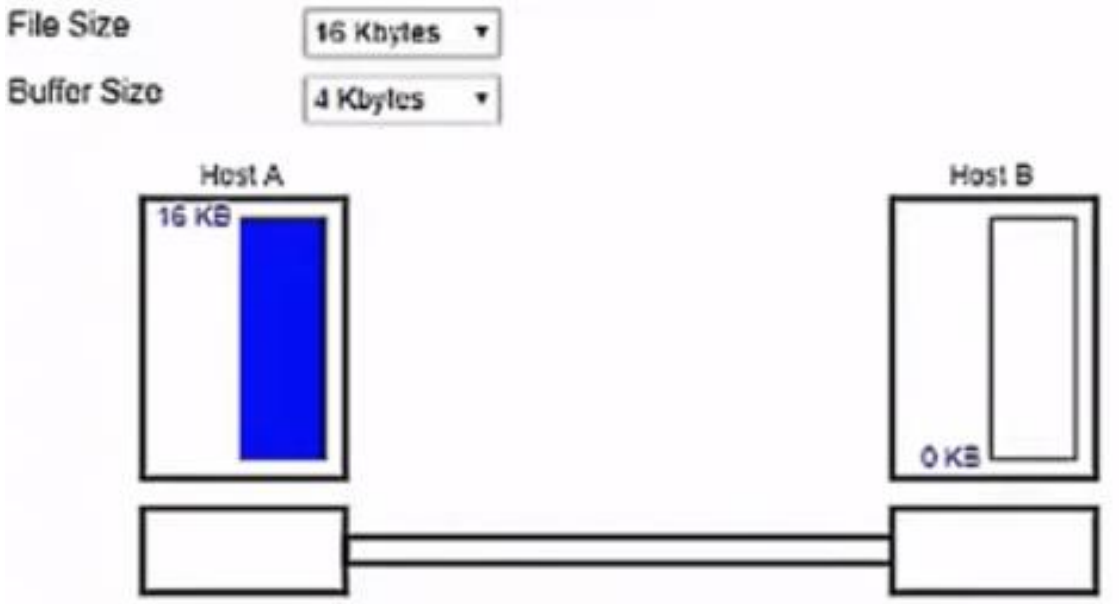
1.3.TCP truyền dữ liệu đáng tin cậy

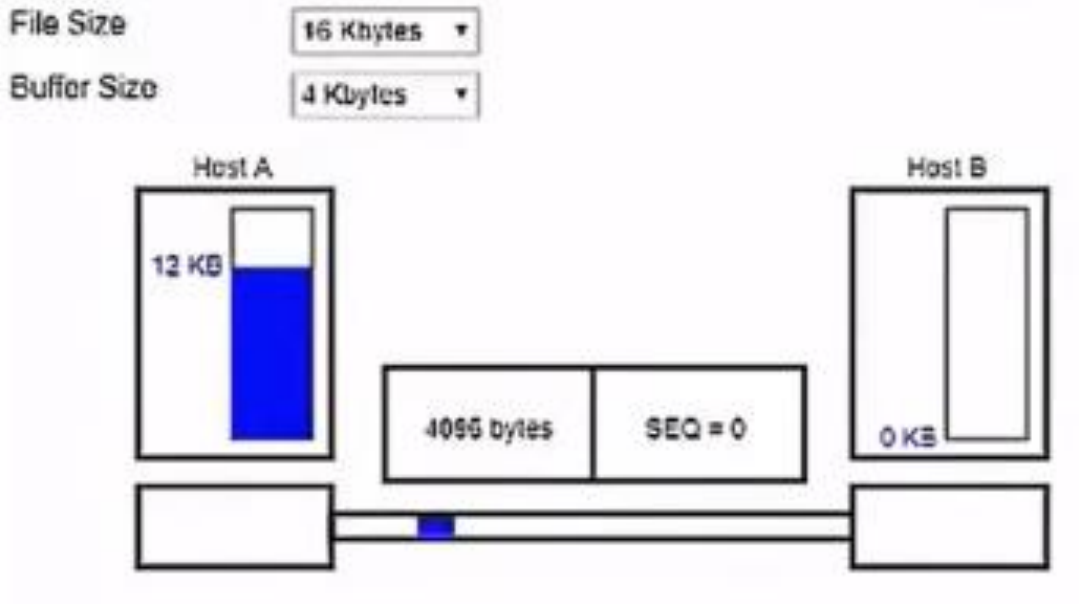




III.TẦNG VẬN CHUYỂN

1.4.TCP điều khiển luồng

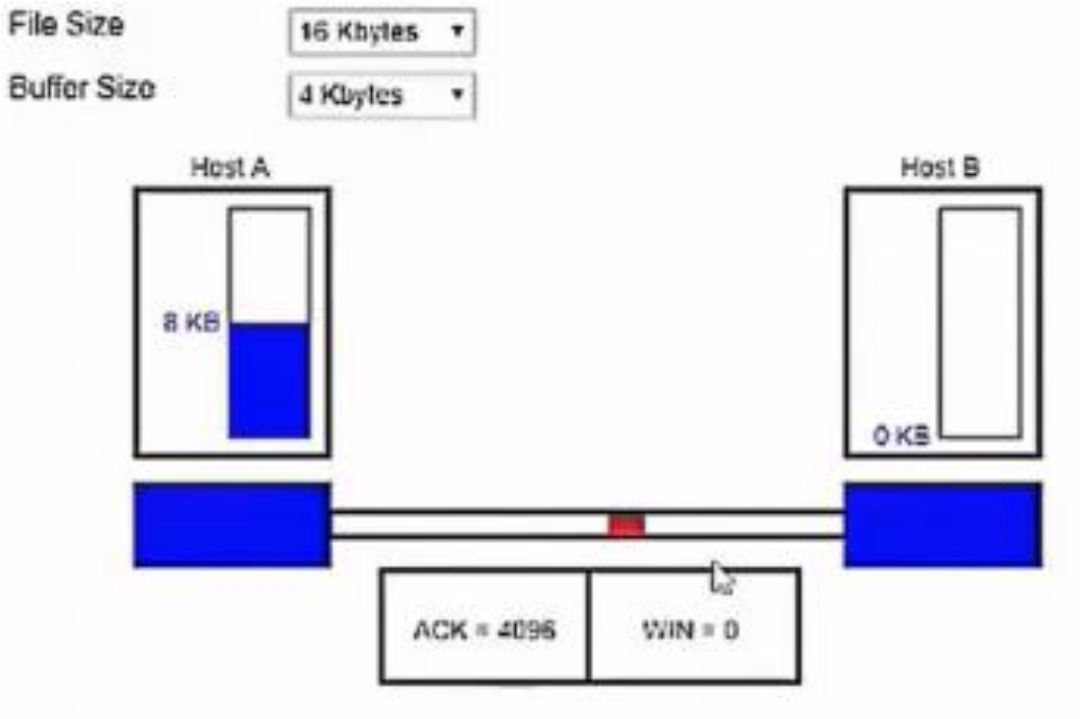


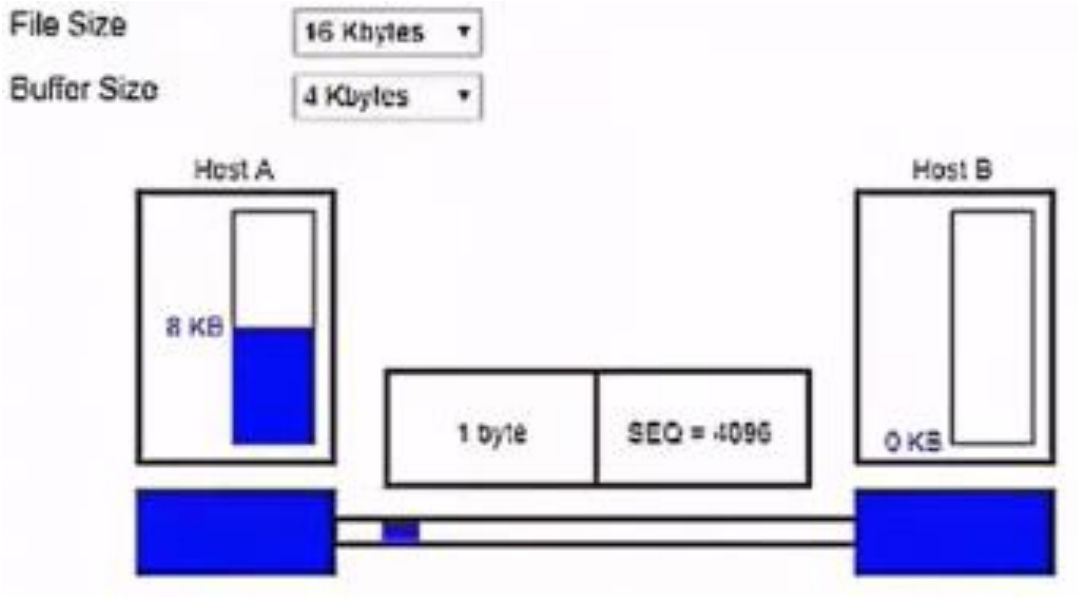


III.TẦNG VẬN CHUYỂN

1.4.TCP điều khiển luồng

- Bên gửi thông báo bộ đệm trống bằng thông tin rwnd trong TCP header.
- Bên gửi giới hạn lượng dữ liệu được gửi khi nhận được rwnd để đảm bảo bên nhận không bị quá tải.

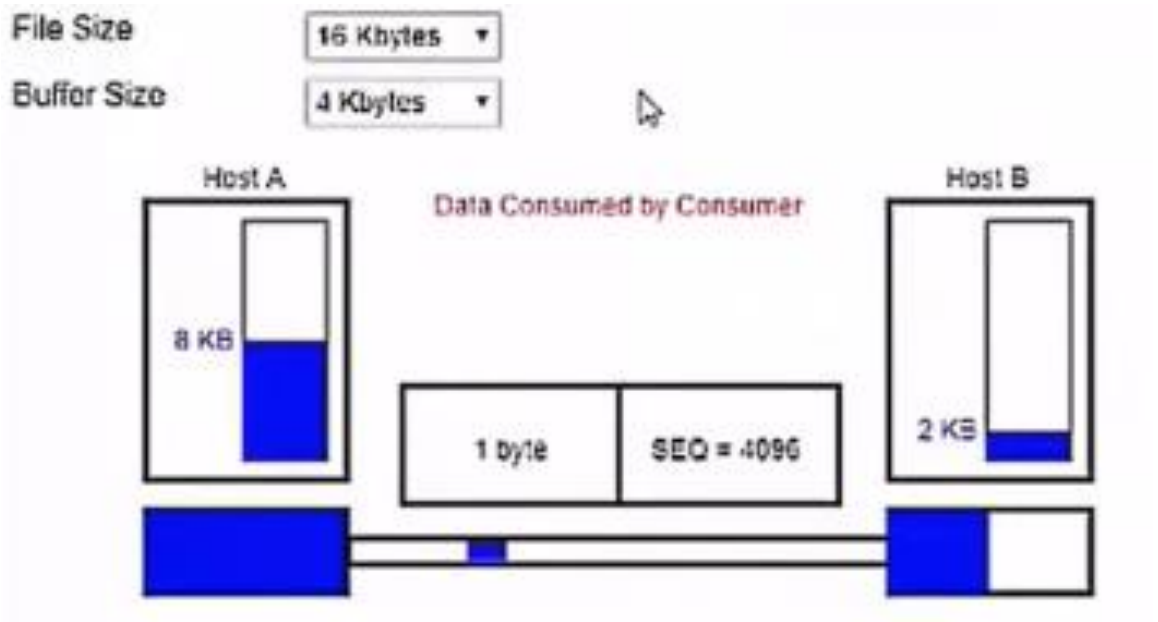


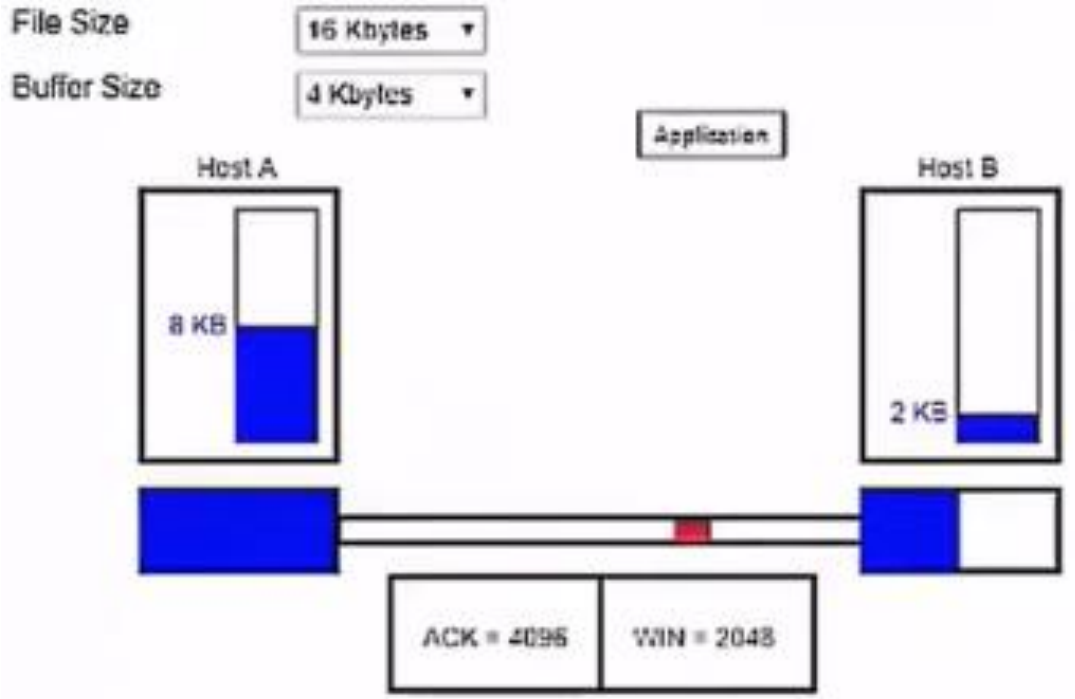


III.TẦNG VẬN CHUYỂN

1.4.TCP điều khiển luồng

- Bên gửi thông báo bộ đệm trống bằng thông tin rwnd trong TCP header.
- Bên gửi giới hạn lượng dữ liệu được gửi khi nhận được rwnd để đảm bảo bên nhận không bị quá tải.





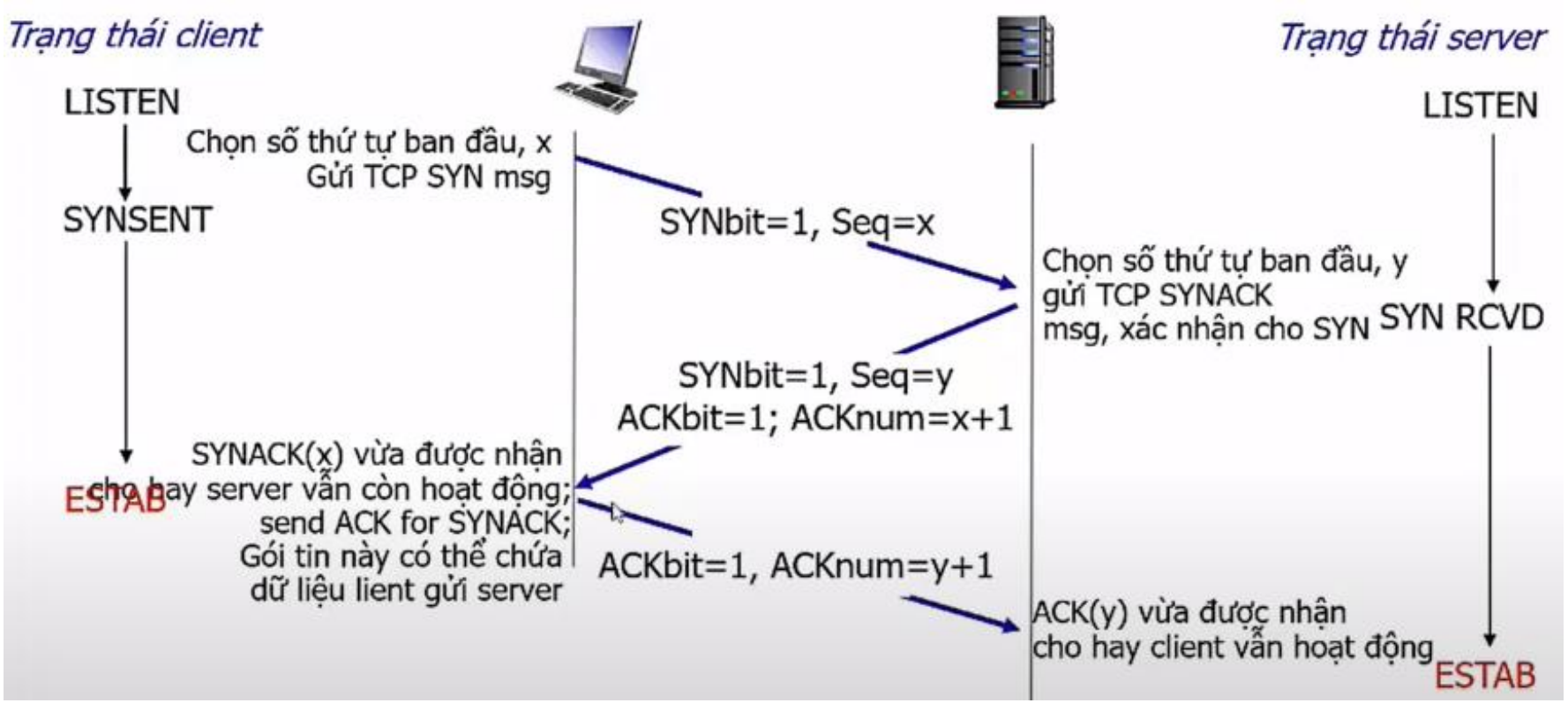
III.TẦNG VẬN CHUYỂN

1.5.TCP bắt đầu kết nối bằng bắt tay 3 bước

- Trước khi trao đổi dữ liệu, bên gửi/bên nhận “handshake – bắt tay”:

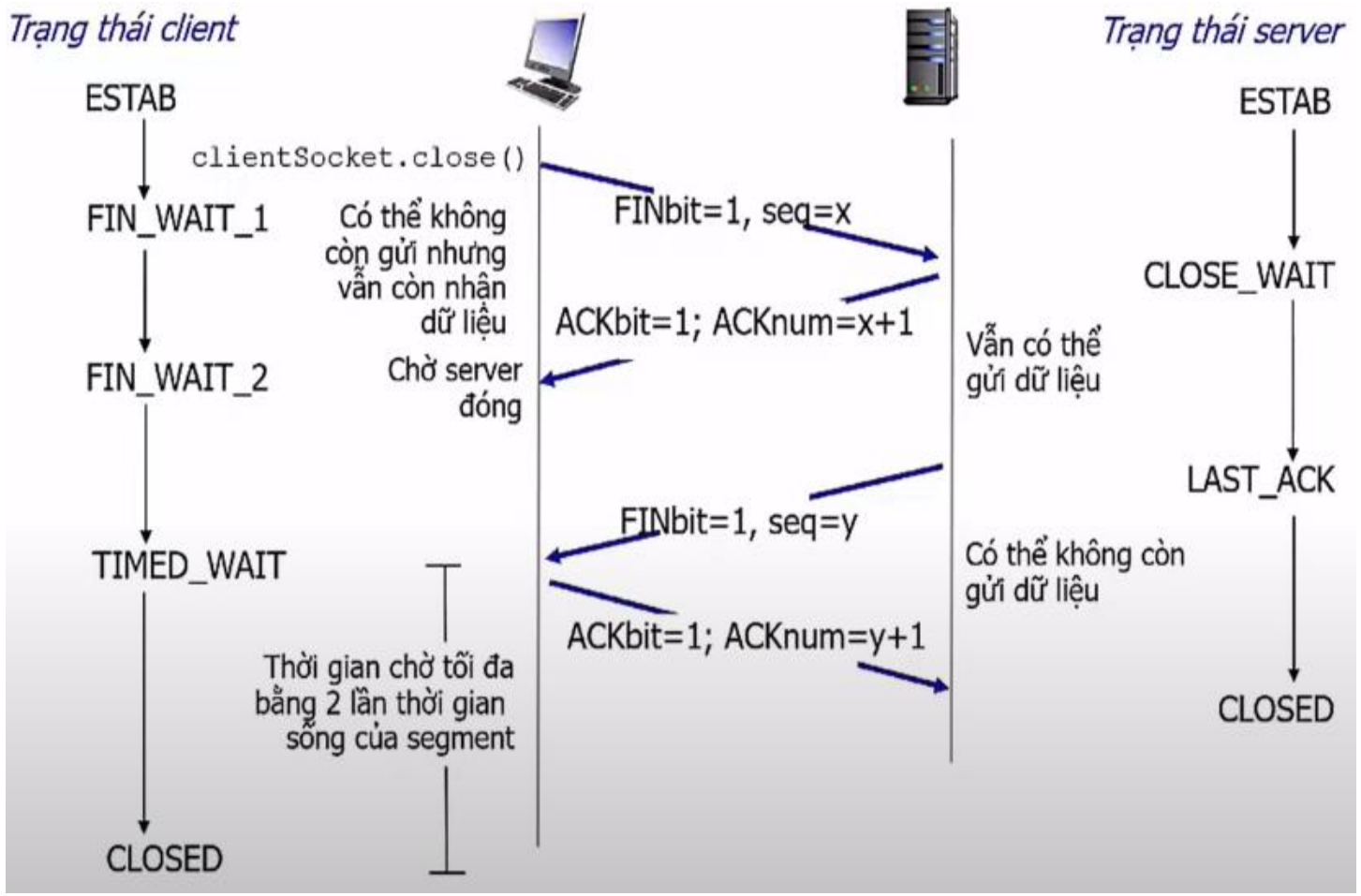
  + Đồng ý thiết lập kết nối (mỗi bên đều biết bên kia sẵn sàng thiết lập kết nối)

  + Đồng ý về các thông số kết nối (ví dụ: bắt đầu từ seq #s)



III.TẦNG VẬN CHUYỂN

1.6.TCP đóng kết nối



-Client, server đóng kết nối cho mỗi bên

+Gửi TCP segment có FIN bit = 1

-Phản hồi khi nhận segment có FIN với ACK

+ Khi nhận FIN, ACK có thể kết hợp với FIN của nó

+ có thể xử lý các trao đổi FIN đồng thời

III.TẦNG VẬN CHUYỂN

1.7.Ôn tập

1. Giả sử rằng giá trị ước tính hiện tại estimatedRTT = 240 ms và độ lệch RTT DevRTT = 10ms, ba giá trị đo tiếp theo của RTT lần lượt là 360, 320 và 390.

Tính giá trị mới của estimatedRTT, DevRTT và TCP timeout sau mỗi ba giá trị RTT đo được này. Sử dụng các giá trị của α = 0,125 và β = 0,25.

III.TẦNG VẬN CHUYỂN

1.7.Ôn tập

Giả sử rằng giá trị ước tính hiện tại estimatedRTT = 240 ms và độ lệch RTT DevRTT = 10ms, ba giá trị đo tiếp theo của RTT lần lượt là 360, 320 và 390.

Tính giá trị mới của estimatedRTT, DevRTT và TCP timeout sau mỗi ba giá trị RTT đo được này. Sử dụng các giá trị của α = 0,125 và = 0,25.

RTT thứ nhất: estimatedRTT= 255 ms, DevRTT= 37.5 ms, TCP timeout= 280 ms

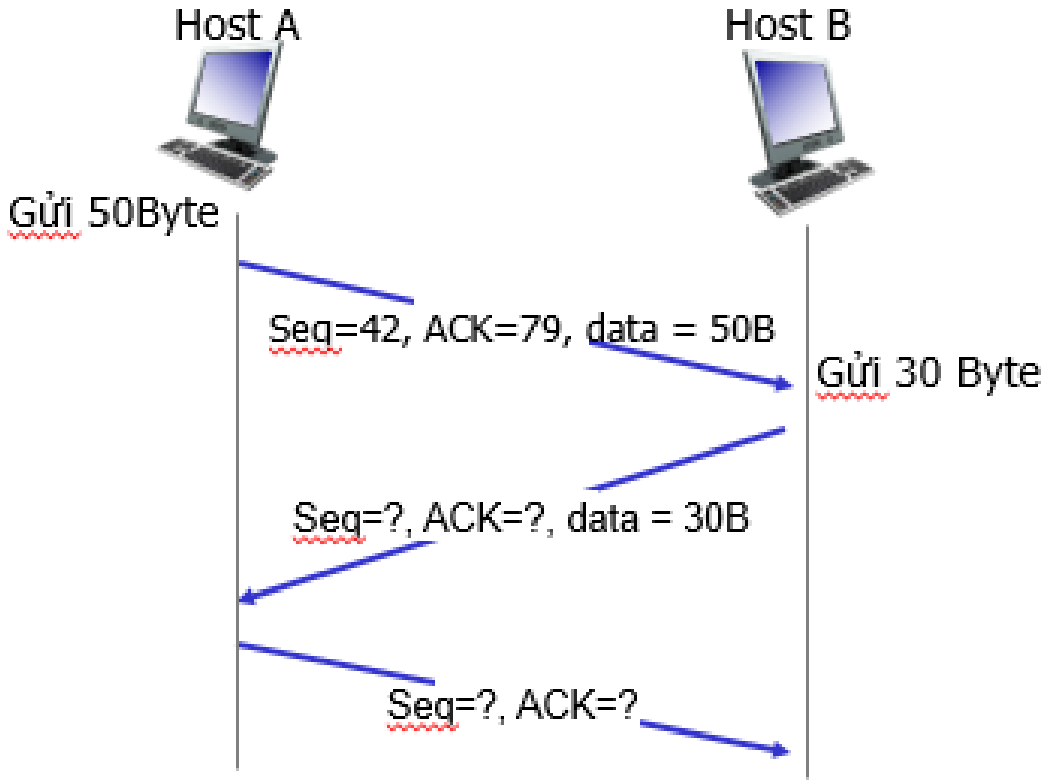
RTT thứ hai: estimatedRTT= 250 ms, DevRTT= 27.5 ms, TCP timeout= 280 ms

RTT thứ ba: estimatedRTT= 258.75 ms, DevRTT= 45 ms, TCP timeout= 280 ms

III.TẦNG VẬN CHUYỂN

1.7.Ôn tập

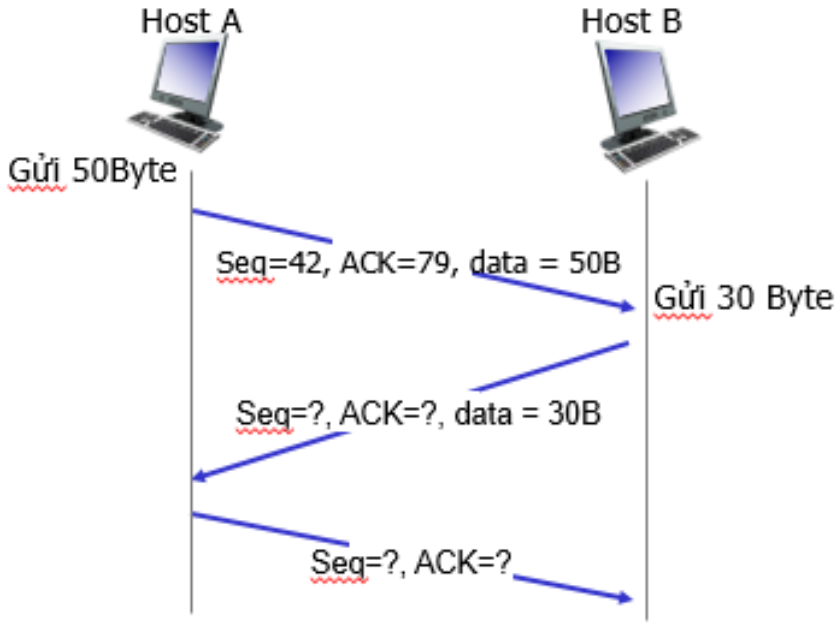
2. Tính SEQ và ACK



III.TẦNG VẬN CHUYỂN

1.7.Ôn tập

2. Tính SEQ và ACK ở gói tin cuối



Đáp án:

Seq= 92, ACK=109

III.TẦNG VẬN CHUYỂN

2.1.TCP điều khiển tắc nghẽn

-**Ký hiệu**

+ Cwnd(n) - congestion windows: số segment được gửi trong cùng một window.

+ Ssthresh (slowstart threshold): ngưỡng kết thúc giai đoạn slowstart và chuyển sang congestion avoidance.

2.2. Slow start

-khi bắt đầu, tăng tốc độ theo cấp số nhân cho đến khi xảy ra sự kiện mất đầu tiên :

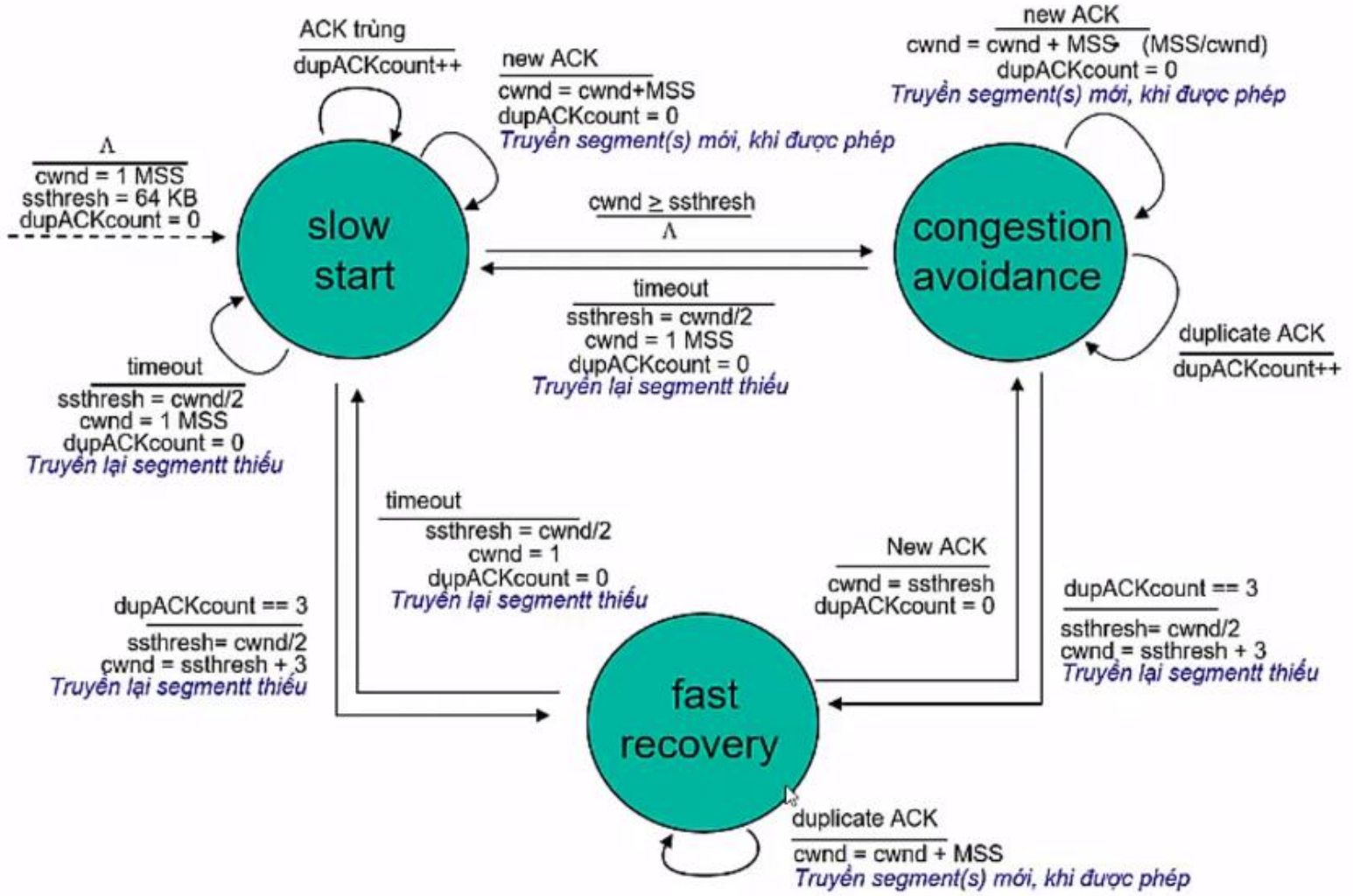
+ Ban đầu cwnd = 1 MSS .

+ Nhân đôi cwnd sau mỗi RTT .

+ được thực hiện bằng cách tăng cwnd cho mỗi ACK nhận được.

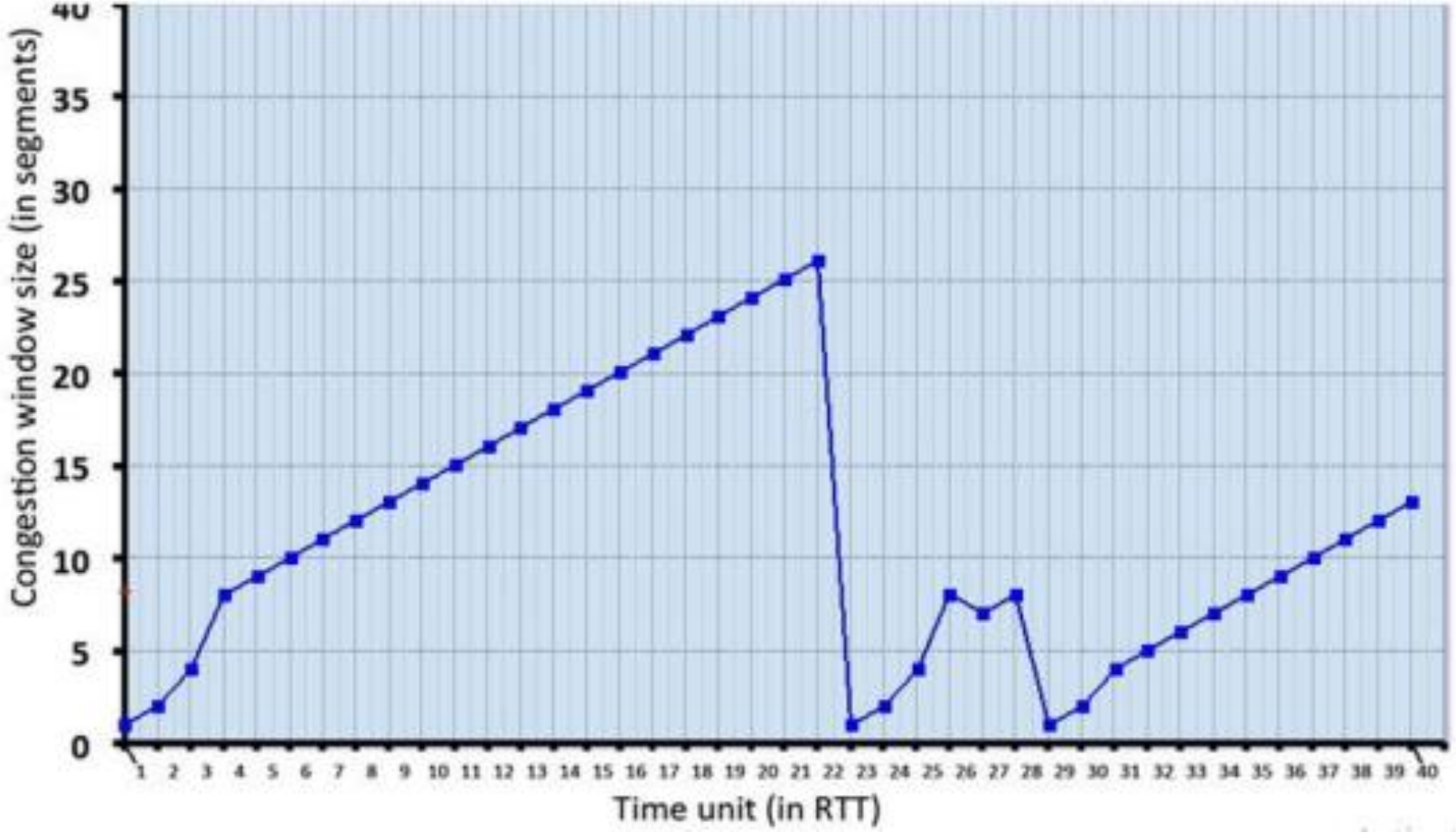
III.TẦNG VẬN CHUYỂN

2.1.TCP điều khiển tắc nghẽn



III.TẦNG VẬN CHUYỂN

2.3.Ôn tập tắc nghẽn gói tin:



Câu 1: Thời điểm nào bên gửi nhận ra có sự tắc nghẽn do nhận được 3 ACKs trùng?

A.t=26RTT

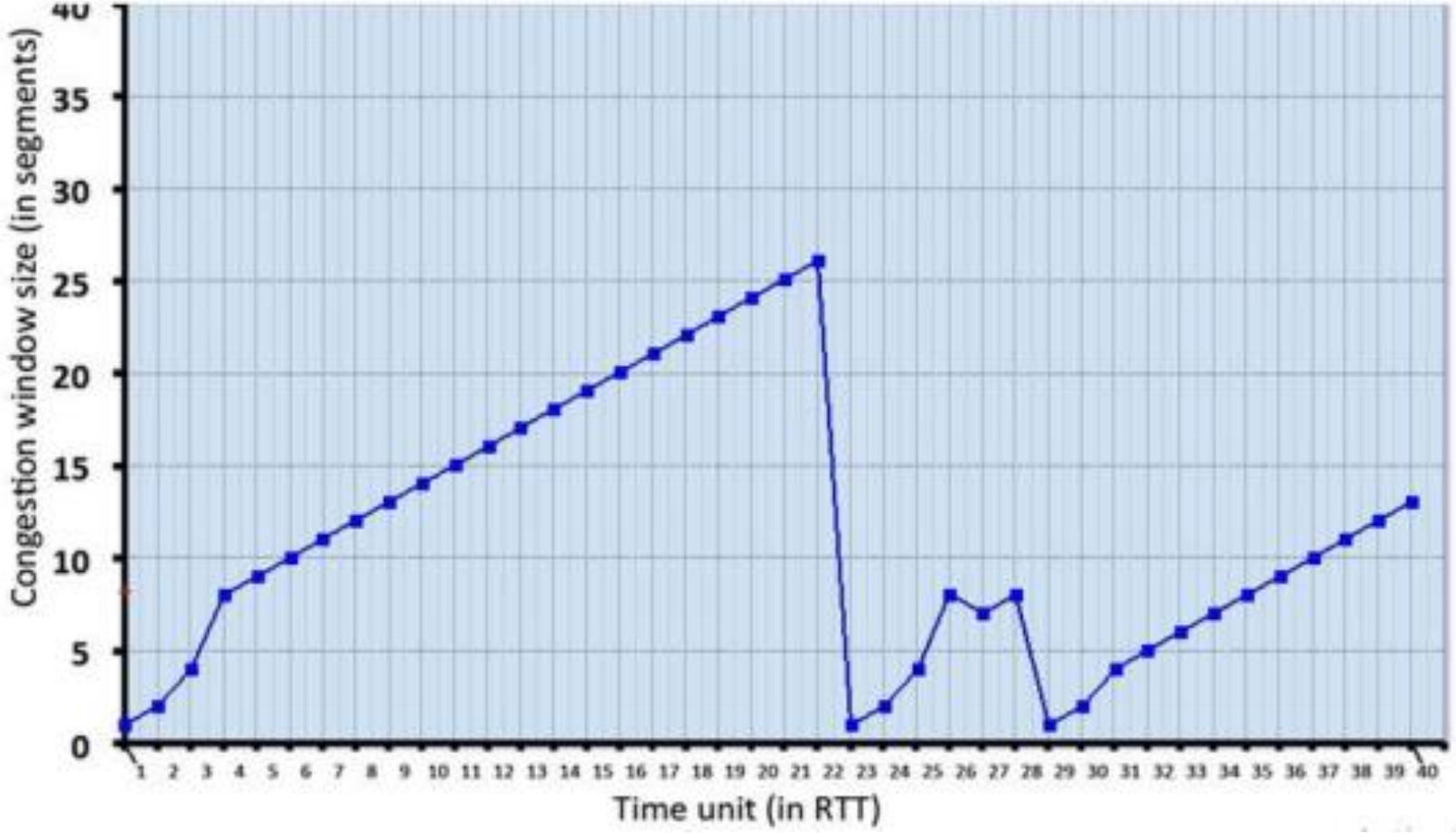
B. t=4RTT

C. t=22RTT

D. t=28RTT

III.TẦNG VẬN CHUYỂN

2.3.Ôn tập tắc nghẽn gói tin:



Câu 1: Thời điểm nào bên gửi nhận ra có sự tắc nghẽn do nhận được 3 ACKs trùng?

A.t=26RTT

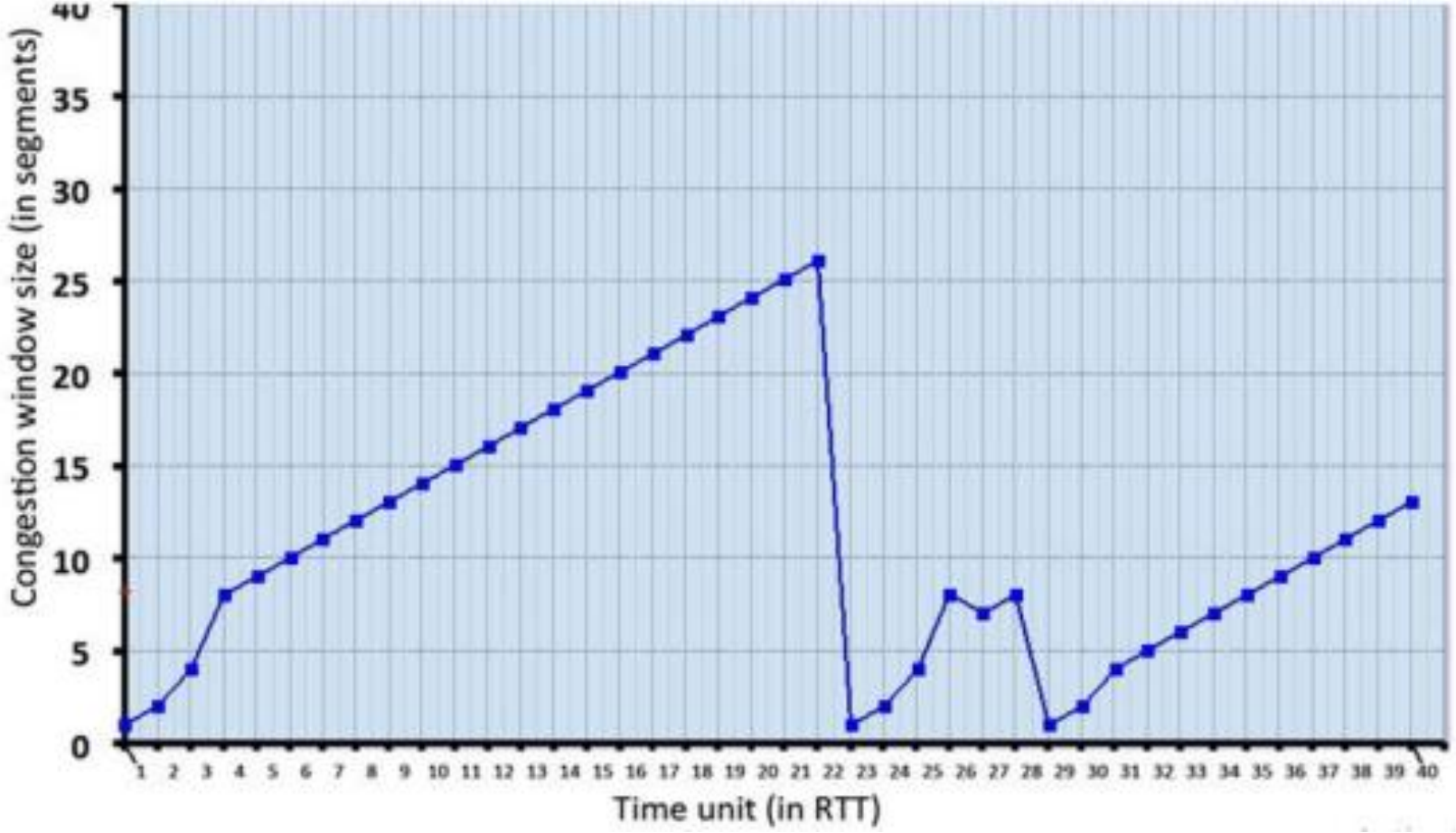
B. t=4RTT

C. t=22RTT

D. t=28RTT

III.TẦNG VẬN CHUYỂN

2.3.Ôn tập tắc nghẽn gói tin:



Câu 2: Giá trị ssthresh tại thời điểm t=24 là bao nhiêu?

A. 8

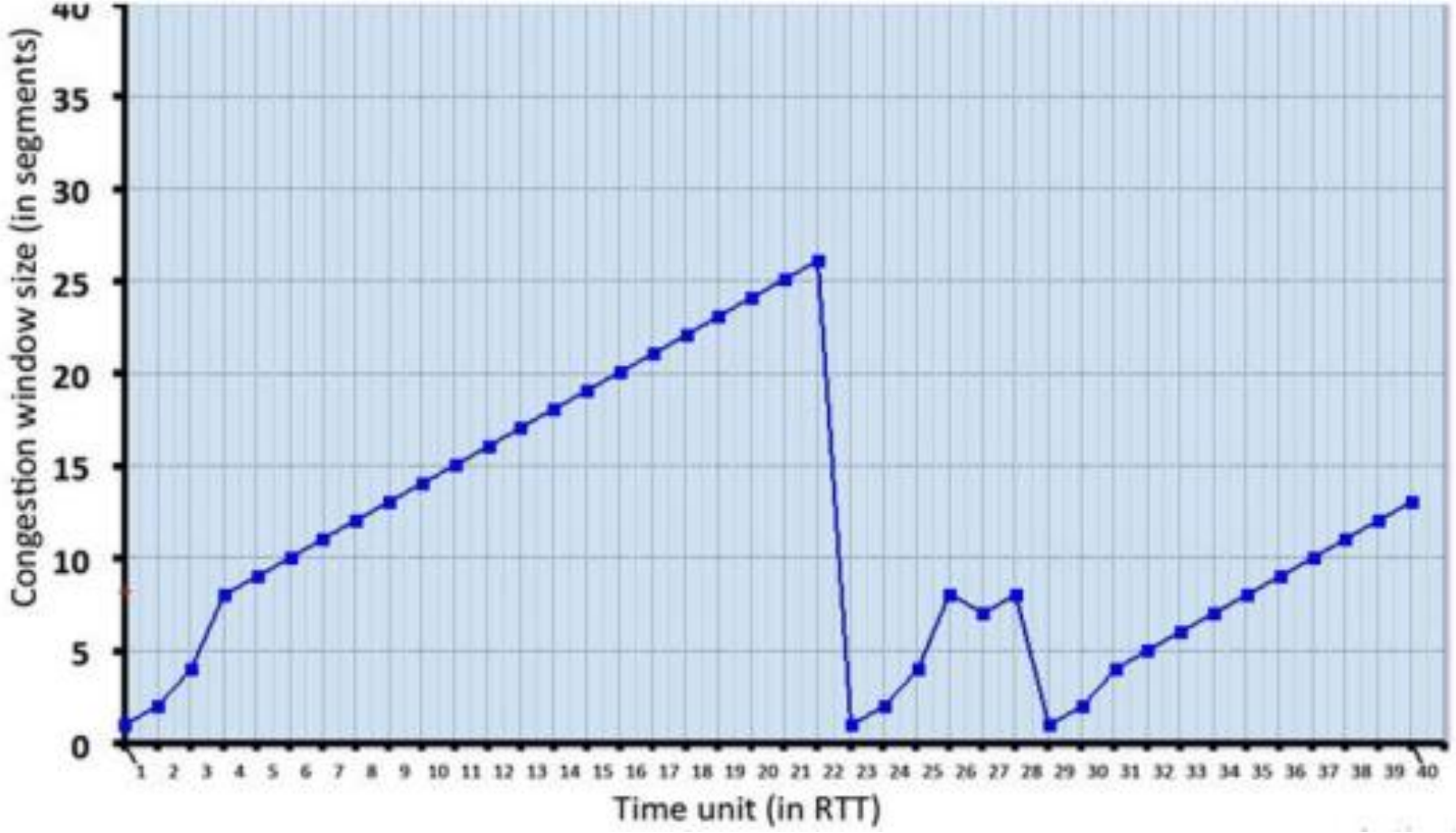
B. 13

C. 4

D. Đáp án khác

III.TẦNG VẬN CHUYỂN

2.3.Ôn tập tắc nghẽn gói tin:



Câu 2: Giá trị ssthresh tại thời điểm t=24 là bao nhiêu?

A. 8

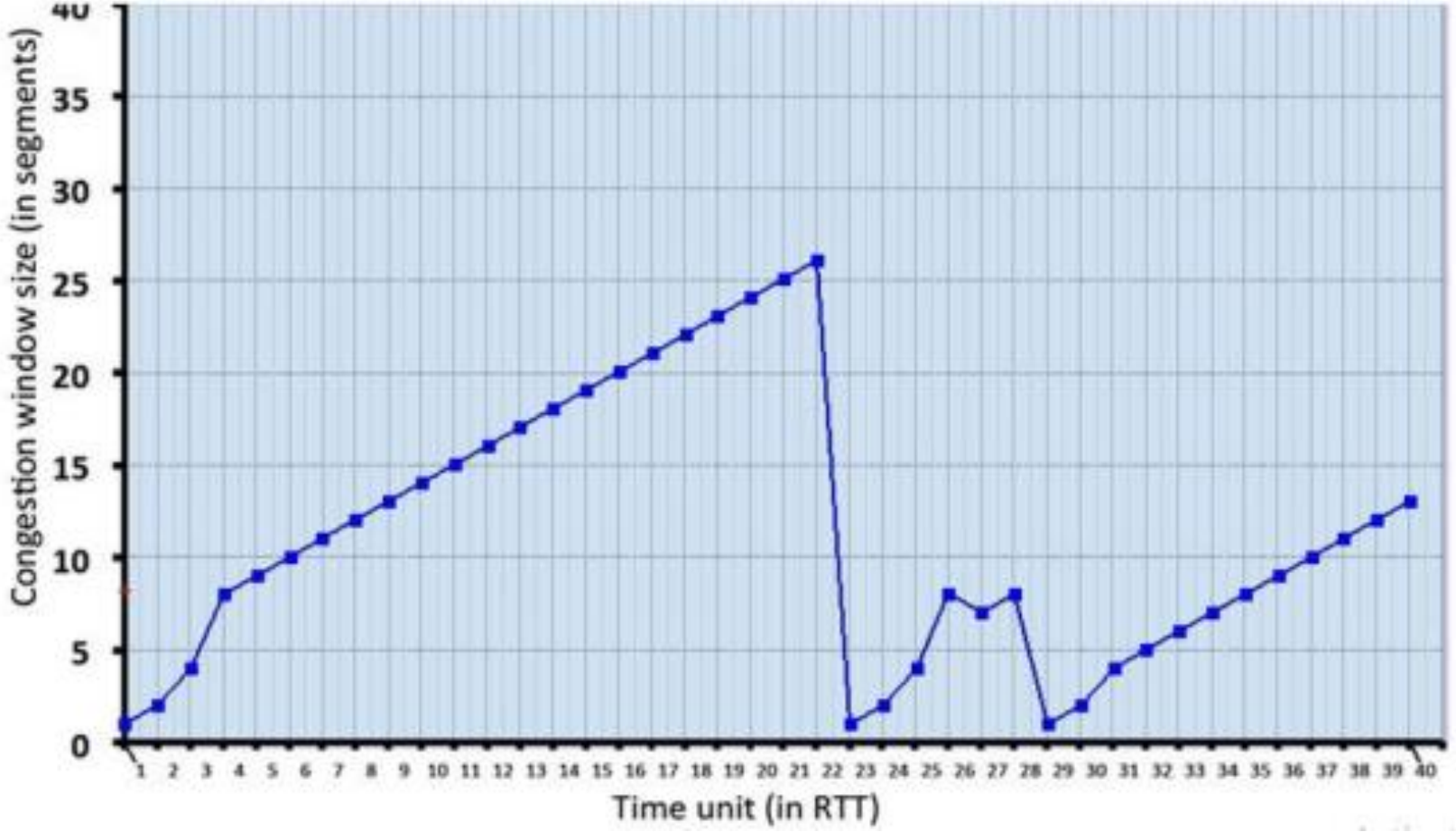
B. 13

C. 4

D. Đáp án khác

III.TẦNG VẬN CHUYỂN

2.3.Ôn tập tắc nghẽn gói tin:



Câu 3: Xác định giai đoạn Slow Start.

A1-4

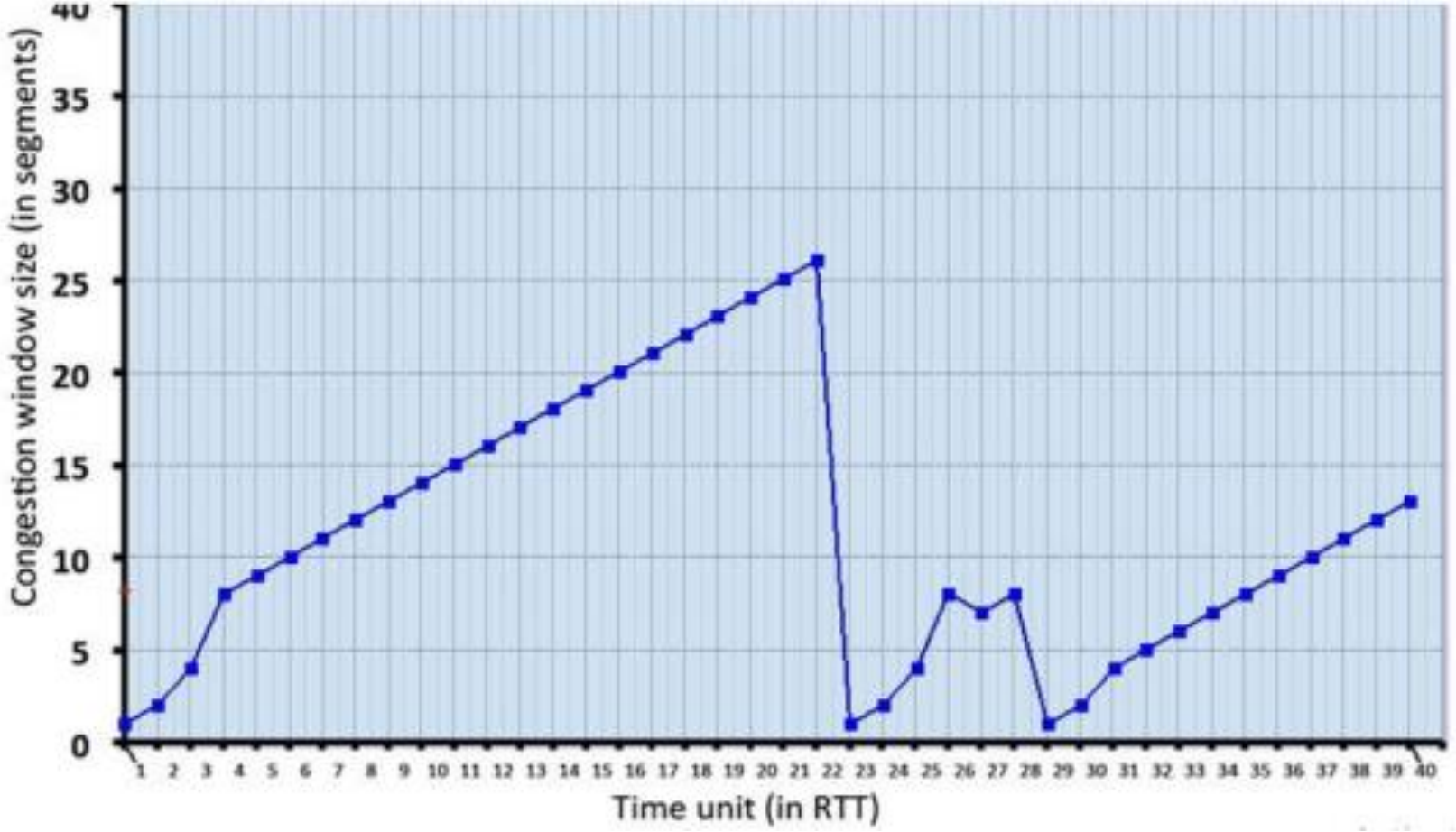
B. 23-26

C. 29-31

D. Tất cả đều đúng

III.TẦNG VẬN CHUYỂN

2.3.Ôn tập tắc nghẽn gói tin:



Câu 3: Xác định giai đoạn Slow Start.

A1-4

B. 23-26

C. 29-31

D. Tất cả đều đúng

Nội dung ôn tập

01. Giới thiệu

02. Tầng ứng dụng

03. Tầng vận chuyển

04. Tầng mạng

05. Tầng liên kết

06. Giải bài tập

Giới thiệu

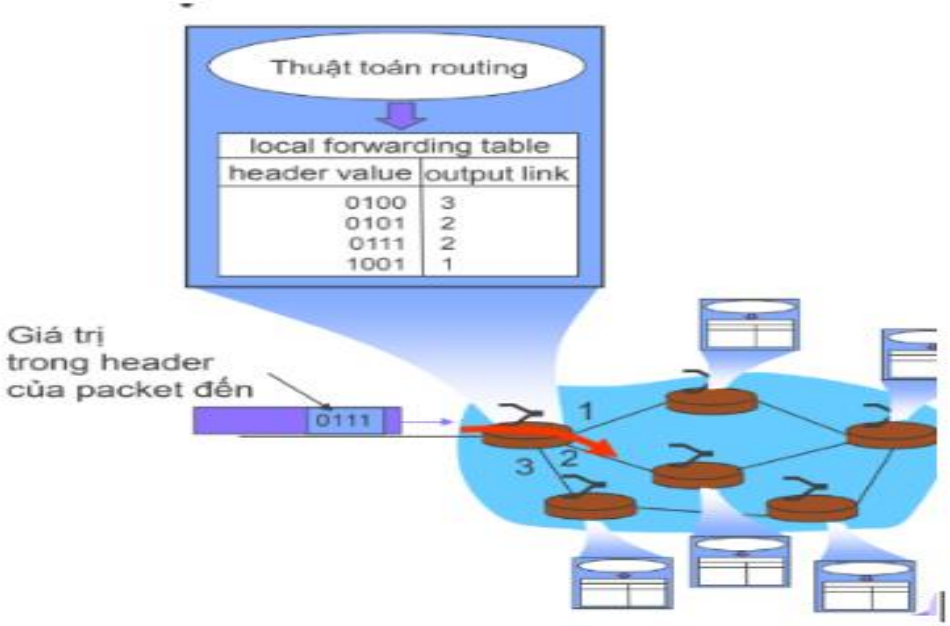
Forwarding và Routing

Tầng mạng có 2 chức năng chính

+ Forwarding (chuyển tiếp): di chuyển các gói tin từ đầu vào đến đầu ra thích hợp của router

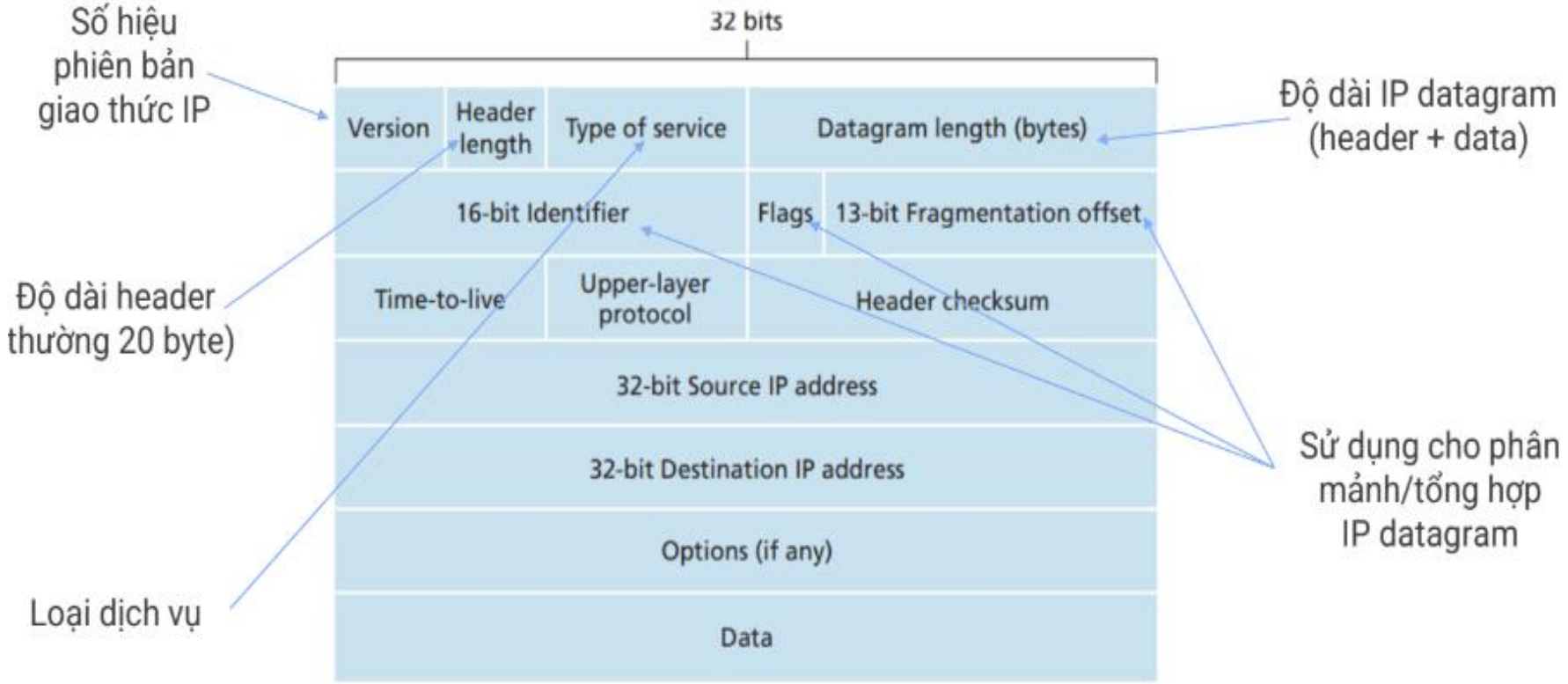
+ Routing (định tuyến): xác định đường đi cho gói tin từ nguồn tới đích

=> Thuật toán định tuyến lựa chọn đường đi tốt nghiệp giữa 2 đầu cuối dựa vào bảng chuyển tiếp trong mỗi router trong network



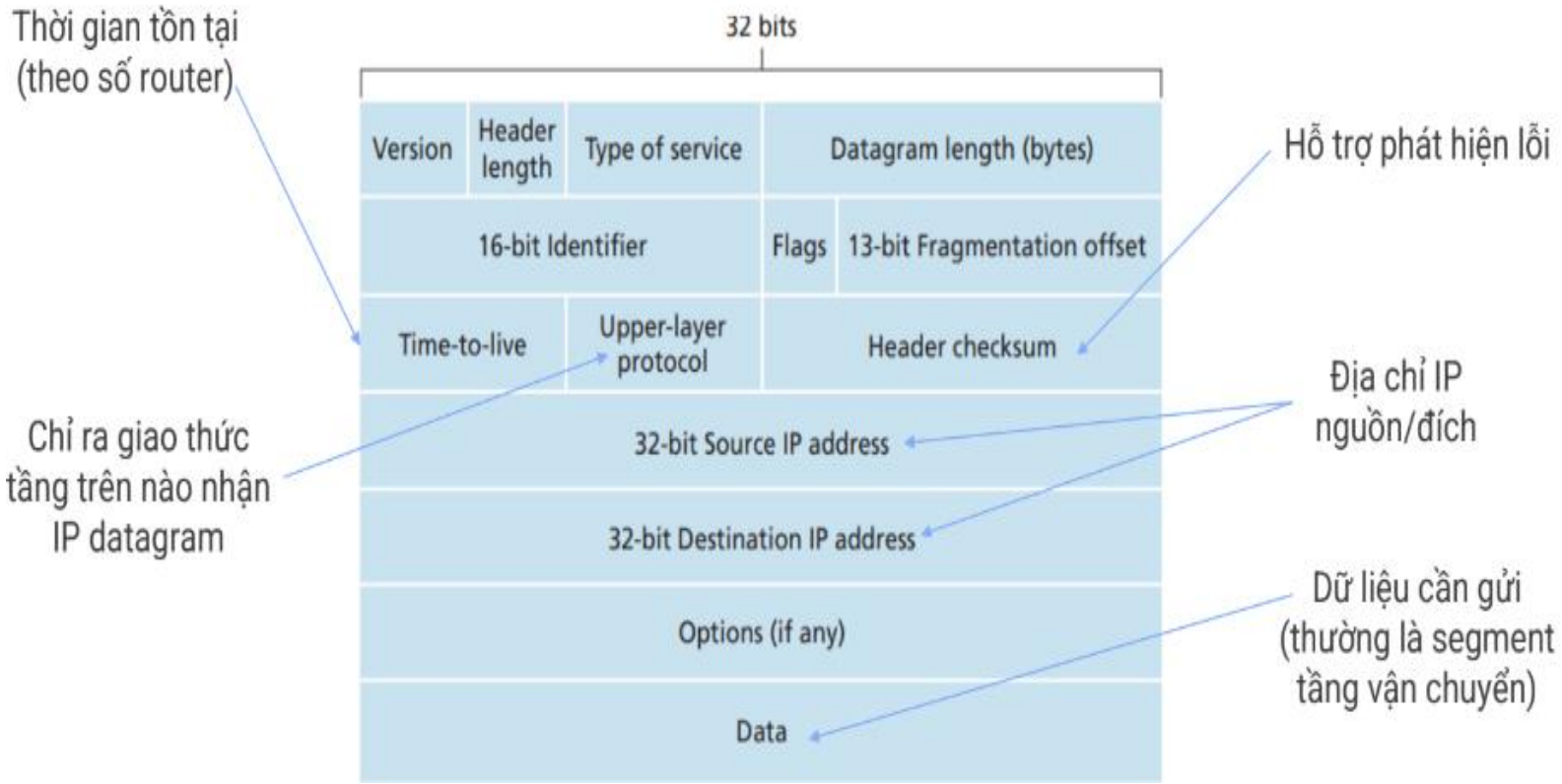
Internet Protocol (IP)

Cấu trúc gói tin giao thức IP



Internet Protocol (IP)

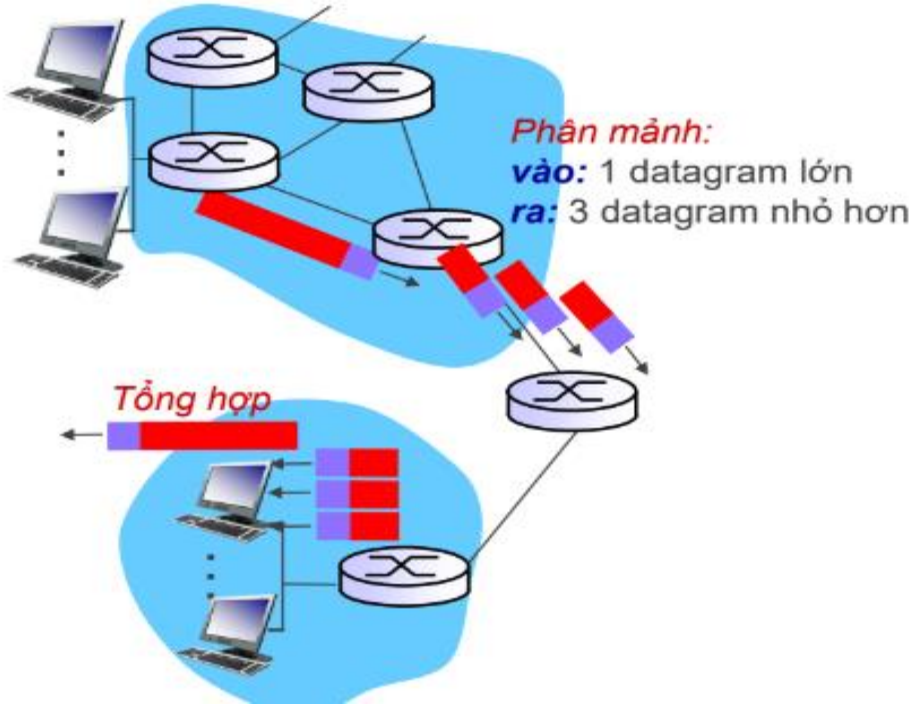
Cấu trúc gói tin giao thức IP



Phân mảnh và tổng hợp IP

- Các đường link mạng khác nhau có kích thước frame lớn nhất (MTU) có thể gửi khác nhau

- Các IP datagram lớn cần được phân mảnh tại nơi gửi và tổng hợp lại trước khi gửi cho bên nhận



Phân mảnh và tổng hợp IP

Các trường Identification, Flag và Fragmentation offset hỗ trợ trong việc phân rã và tổng hợp IP datagram

+ Identification: Khi một gói tin IP được tạo, bên gửi sẽ đánh dấu bằng số ‘identification’ và địa chỉ gửi/nhận. Với mỗi lần gói tin được phân mảnh, trường Identification của gói tin phân mảnh sẽ giống với trường Identification của gói tin gốc

+ Flag: Đối với gói tin IP được phân mảnh cuối cùng, flag bit được đặt bằng 0. Các gói tin phân mảnh còn lại có flag bằng 1

+ Fragmentation offset: Được sử dụng để xác định vị trí gói tin phân mảnh trong gói tin gốc và kiểm tra mất mát gói tin (offset * 8 = vị trí cần ghép vào)

Phân mảnh và tổng hợp IP

Ví dụ:
Cần gửi một datagram có kích thước 2400 byte qua một đường link có MTU bằng 700 byte. Biết trường 'Identification' của datagram gốc có giá trị 422. Hỏi có bao nhiêu phân mảnh được tạo ra? Giá trị của các trường liên quan tới việc phân mảnh của mỗi gói tin?

Datagram gốc:

...	Length = 2400	Identification = 422	Flag = 0	Offset = 0	...

Datagram gốc có kích thước 2400 byte => 20 byte header và 2380 byte data
Đường link có MTU bằng 700 byte => 20 byte header và 680 byte data
Số phân mảnh được tạo ra = [2380/680] = 4

...	Length = 700	Identification = 422	Flag = 1	Offset = 0	...
...	Length = 700	Identification = 422	Flag = 1	Offset = 85	...
...	Length = 700	Identification = 422	Flag = 1	Offset = 170	...
...	Length = 360	Identification = 422	Flag = 0	Offset = 255	...

Phân mảnh và tổng hợp IP

Ví dụ:
Giả sử đường link giữa host A và B giới hạn datagram có kích thước tối đa 1500 byte (header + data). Biết header IP datagram có kích thước 20 byte, cần bao nhiêu datagram để gửi gói tin MP3 có kích thước 5 * 106 byte

Cần gửi gói tin MP3 => Sử dụng giao thức TCP
=> Datagram gồm 20 byte header IP và 20 byte header TCP, còn lại 1460 byte dữ liệu MP3
=> Số datagram cần tạo: $\lceil 5 * 10^6 / 1460 \rceil = 3425$ datagram
Các datagram có kích thước 1500 byte
Datagram cuối có kích thước 1000 byte
Lưu ý: không xảy ra phân mảnh IP datagram do các datagram đều nhỏ hơn hoặc bằng MTU của đường link => Các datagram có trường identification khác nhau

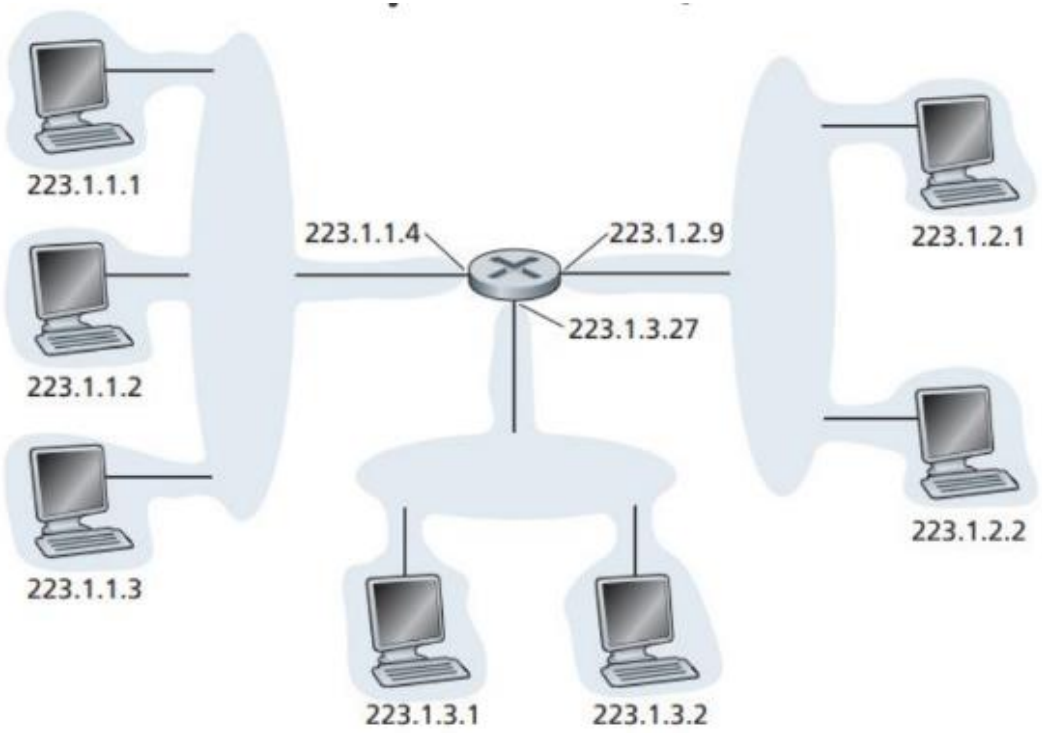
Địa chỉ IPv4

Mỗi địa chỉ IP có độ dài 32 bit (4 byte) chia thành 4 cụm 8 bit (gọi là các octet) dùng để phân biệt các host, router interface

- Interface: Kết nối giữa host/router và đường kết nối vật lý
+ Router thường có nhiều interface
+ Host thường có 1 hoặc 2 interface
=> Mỗi địa chỉ được kết nối với mỗi interface
- Được biểu diễn dưới dạng dotted – decimal – notation, mỗi byte của địa chỉ IP được viết dưới dạng số thập phân và chia cách bằng dấu “.”

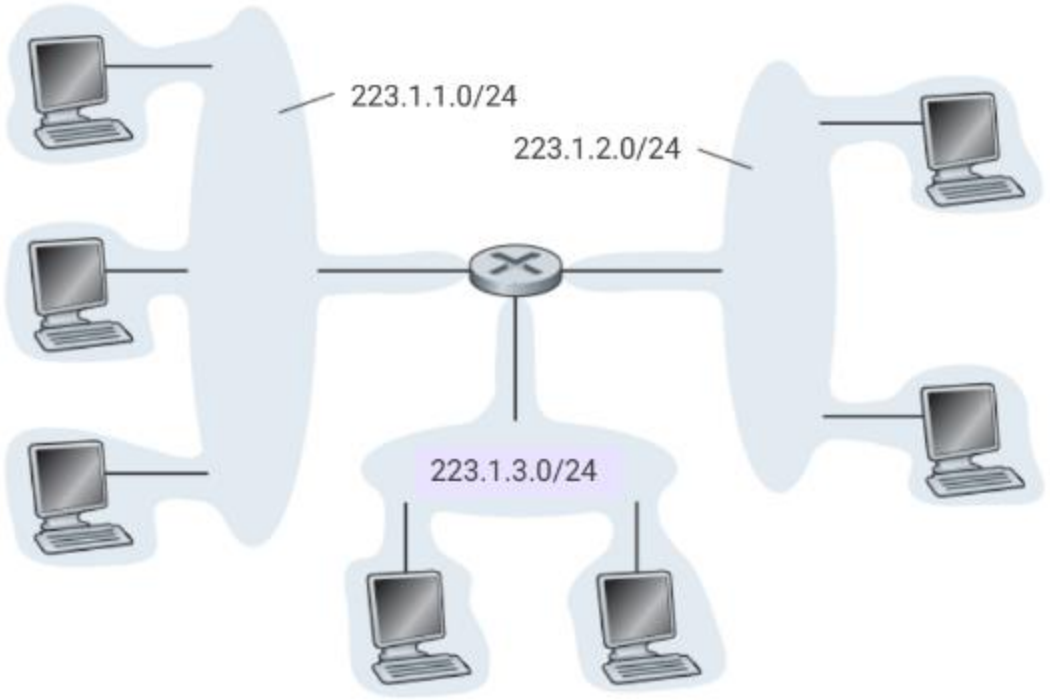
VD: Địa chỉ IPv4 có dạng nhị phân “11000001 00100000 11011000 00001001” có thể được viết lại thành “193.32.216.9”

Địa chỉ IPv4



Một router với 3 interface kết nối với 7 host

Subnet



3 subnet kết nối các interface của router với các host

Subnet

Địa chỉ IP được chia làm 2 phần:

- Subnet: các bit bên trái

- Host: các bit bên phải

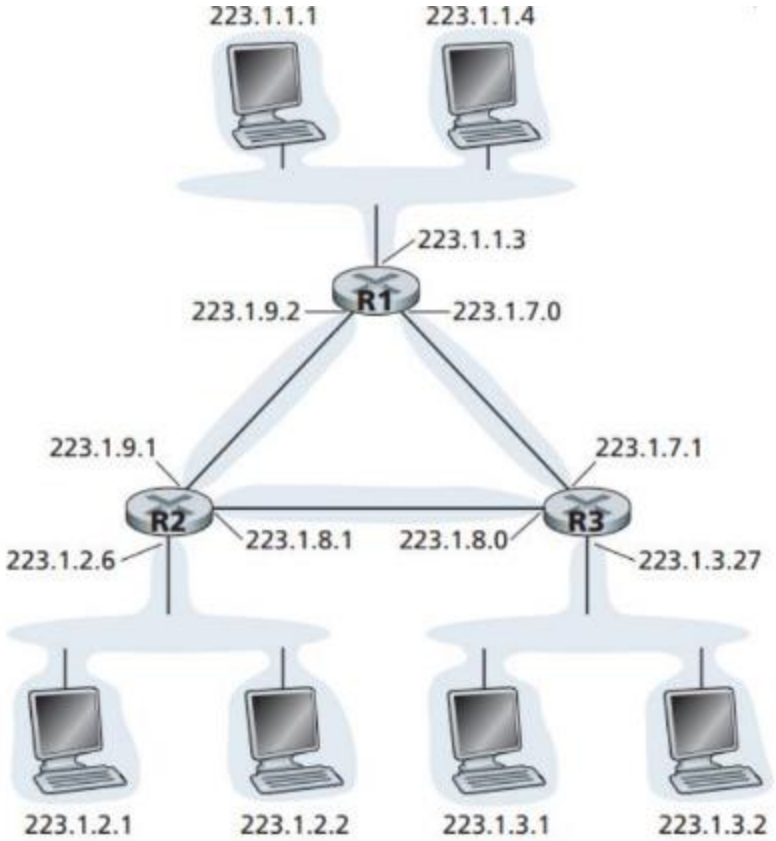
Subnet là các interface của thiết bị có phần subnet của địa chỉ IP giống nhau

Các thiết bị trong cùng một subnet có thể giao tiếp vật lý với nhau mà không cần router trung gian can thiệp

Phương pháp xác định subnet:

- Tách mỗi interface từ host hoặc router của nó, tạo thành vùng các mạng độc lập
- Mỗi mạng độc lập được gọi là một subnet

Subnet

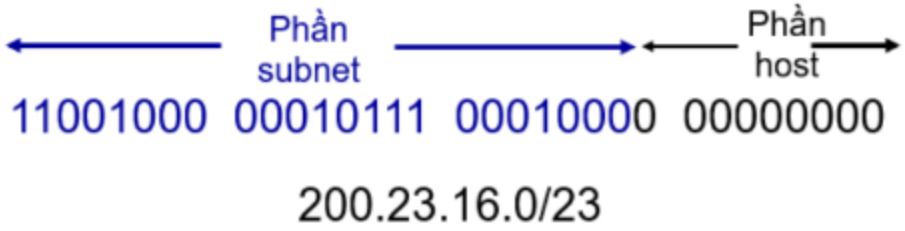

223.1.1.1
223.1.1.4
223.1.1.3
R1
223.1.9.2
223.1.7.0
223.1.9.1
223.1.7.1
R2
R3
223.1.8.1
223.1.8.0
223.1.2.6
223.1.3.27
223.1.2.1
223.1.2.2
223.1.3.1
223.1.3.2

Phân lớp địa chỉ IPv4

Lớp	Octet thứ 1 hệ thập phân	MSB của octet thứ 1	Network/Host	Subnet mask mặc định	Số mạng	Số host mỗi mạng
A	1 – 126	0	N.H.H.H	255.0.0.0	126	16777214
B	128 – 191	10	N.N.H.H	255.255.0.0	16382	65534
C	192 – 223	110	N.N.N.H	255.255.255.0	2097150	254
D	224 – 239	1110	Dành riêng cho các nhóm Multicast			
E	240 - 255	1111	Dành riêng cho nghiên cứu			

Classless InterDomain Routing (CIDR)

Sử dụng thay thế cho phương pháp phân lớp địa chỉ IP Đặc điểm:
- Phần subnet có độ dài bất kỳ
- Định dạng địa chỉ: a.b.c.d/x, trong đó x là số các bit trong phần subnet của địa chỉ



Quy tắc đặt địa chỉ IPv4:
- Các bit phần mạng không cùng bằng 0 (0.0.0.1 là địa chỉ không hợp lệ)
- Các bit phần host đồng thời bằng 0 => Địa chỉ mạng (192.168.2.0/24 là địa chỉ mạng)
- Các bit phần host đồng thời bằng 1 => Địa chỉ broadcast (192.168.2.255/24 là địa chỉ broadcast cho mạng 192.168.2.0/24)

Chia mạng con trong IPv4

Quy tắc
- Mượn thêm một số bit bên phần host để làm phần mạng
- Công thức mượn i bit (2^i >= n), với n là số mạng con cần chia
- Các bit mượn được gọi là các bit subnet
- Số bit tối đa có thể mượn phải tuân theo công thức: Subnet <= Host - 2

Network	Subnet	Host

Các bước thực hiện chia mạng con
- Xác định lớp (class) và subnet mask mặc định của địa chỉ
- Xác định số bit cần mượn và subnet mask mới, tính số lượng mạng con, số host mỗi mạng con thực sự có được
- Xác định các vùng địa chỉ host và chọn mạng con muốn dùng

Chia mạng con trong IPv4

Ví dụ:

Cho địa chỉ IP sau "172.16.0.0/16". Hãy chia thành 8 mạng con và có tối thiểu 1000 host trên mỗi mạng con đó

Địa chỉ IP 172.16.0.0 thuộc lớp B
=> Subnet mask mặc định: 255.255.0.0
Cần chia thành 8 mạng con
=> Mượn 3 bit
Phần host còn lại 16 – 3 = 13 bit
=> Mỗi mạng con có 2^{13} – 2 host (>1000)
Subnet mask mới: 11111111 11111111 11100000 00000000 hay 255.255.224.0

STT	Subnet	Vùng Host	Broadcast
1	172.16.0.0	172.16.0.1 – 172.16.31.254	172.16.31.255
2	172.16.32.0	172.16.32.1 - 172.16.63.254	172.16.63.255
...	...	...	...
7	172.16.192.0	172.16.192.1 – 172.16.223.254	172.16.223.255
8	172.16.224.0	172.16.224.1 – 172.16.255.254	172.16.255.255

Chia mạng con trong IPv4

Ví dụ:
Cho 2 địa chỉ IP sau “192.168.5.9/28” và “192.168.5.39/28”
- Hãy cho biết địa chỉ network, host của từng IP trên
- Các địa chỉ IP trên có cùng 1 mạng hay không
- Liệt kê tất cả các địa chỉ IP thuộc các mạng vừa tìm được

Cả 2 địa chỉ IP trên đều thuộc lớp C
28 là số bit dành cho phần network
=> Subnet mask mặc định của địa chỉ IP có dạng:
11111111 11111111 11111111 1111000 hay 255.255.255.240
Để xác định địa chỉ network và host, thực hiện phép **AND** giữa địa chỉ IP và Subnet mask mặc đị

Chia mạng con trong IPv4

Địa chỉ IP (thập phân)	192	168	5	9
Địa chỉ IP (nhị phân)	11000000	10101000	00000101	00001001
Subnet mask mặc định	11111111	11111111	11111111	11110000
Kết quả phép AND	11000000	10101000	00000101	00000000
Địa chỉ Network	192	168	5	0
Host				9

Kết luận: IP 192.168.5.9/28
- Thuộc mạng có địa chỉ IP: 192.168.5.0/28
- Là host thứ 9 trong mạng

Chia mạng con trong IPv4

Địa chỉ IP (thập phân)	192	168	5	39
Địa chỉ IP (nhị phân)	11000000	10101000	00000101	00100111
Subnet mask mặc định	11111111	11111111	11111111	11110000
Kết quả phép AND	11000000	10101000	00000101	00100000
Địa chỉ Network	192	168	5	32
Host				7

Kết luận: IP 192.168.5.39/28
- Thuộc mạng có địa chỉ IP: 192.168.5.32/28
- Là host thứ 7 trong mạng

Vậy 2 địa chỉ IP “192.168.5.9/28” và “192.168.5.39/28” không cùng thuộc 1 mạng

Chia mạng con trong IPv4

Liệt kê tất cả các địa chỉ host có trong mạng

Mạng tương ứng với IP	Vùng địa chỉ Host dưới dạng nhị phân	Vùng địa chỉ Host dưới dạng thập phân
1	11000000 10101000 00000101 00000001 đến 11000000 10101000 00000101 00001110	192.168.5.1/28 đến 192.168.5.14/28
2	11000000 10101000 00000101 00100001 đến 11000000 10101000 00000101 00101110	192.168.5.33/28 đến 192.168.5.46/28

Tính nhanh vùng địa chỉ IP

Gọi số bit làm subnet là n

Số mạng con S = 2n

Số gia địa chỉ mạng con M = 28 - n (n <= 8)

➢ Byte cuối của IP địa chỉ mạng, ví dụ lớp C: (k – 1) * M (với k = 1, 2, ...)

➢ Byte cuối của IP host đầu tiên, ví dụ lớp C: (k – 1) * M + 1 (với k = 1, 2, ...)

➢ Byte cuối của IP host cuối cùng, ví dụ lớp C: k * M – 2 (với k = 1, 2, ...)

➢ Byte cuối của IP broadcast, ví dụ lớp C: k * M – 1 (với k = 1, 2, ...)

Tính nhanh vùng địa chỉ IP

Cho địa chỉ 192.168.0.0/24 (thuộc lớp C)
Giả sử mượn 2 bit để chia mạng con (n = 2)
Số mạng con S = $2^2 = 4$
Số gia địa chỉ mạng con M = $2^{8-2} = 2^6 = 64$

- Mạng con 1:
Địa chỉ mạng: 192.168.0.0 (Byte cuối của IP địa chỉ mạng = (1 – 1) * 64)
Vùng host: từ 192.168.0.1 (Byte cuối của IP host đầu tiên = (1 – 1) *64 + 1) đến 192.168.0.62 (Byte cuối của IP host cuối cùng = 1 * 64 – 2)
Địa chỉ broadcast: 192.168.0.63 (Byte cuối của IP broadcast = 1 * 64 – 1)

- Mạng con 2:
Địa chỉ mạng: 192.168.0.64 (Byte cuối của IP địa chỉ mạng = (2 – 1) * 64)
Vùng host: từ 192.168.0.65 (Byte cuối của IP host đầu tiên = (2 – 1) *64 + 1) đến 192.168.0.126 (Byte cuối của IP host cuối cùng = 2 * 64 – 2)
Địa chỉ broadcast: 192.168.0.127 (Byte cuối của IP broadcast = 2 * 64 – 1)

Tính nhanh vùng địa chỉ IP

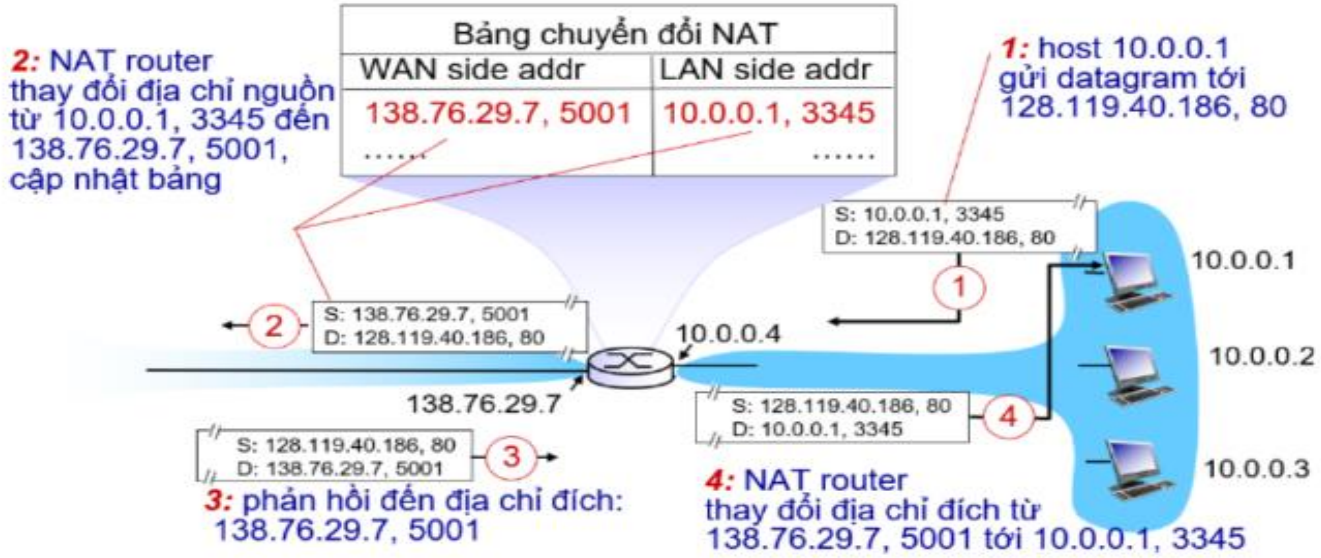
Cho địa chỉ 192.168.0.0/24 (thuộc lớp C)
Giả sử mượn 4 bit để chia mạng con (n = 4)
Số mạng con S = 2^4 = 16
Số gia địa chỉ mạng con M = 2^{8-4} = 16

- Mạng con 3:
Địa chỉ mạng: 192.168.0.128 (Byte cuối của IP địa chỉ mạng = (3 – 1) * 64)
Vùng host: từ 192.168.0.129 (Byte cuối của IP host đầu tiên = (3 – 1) *64 + 1) đến 192.168.0.190 (Byte cuối của IP host cuối cùng = 3 * 64 – 2)
Địa chỉ broadcast: 192.168.0.191 (Byte cuối của IP broadcast = 3 * 64 – 1)

- Mạng con 4:
Địa chỉ mạng: 192.168.0.192 (Byte cuối của IP địa chỉ mạng = (4 – 1) * 64)
Vùng host: từ 192.168.0.193 (Byte cuối của IP host đầu tiên = (4 – 1) *64 + 1) đến 192.168.0.254 (Byte cuối của IP host cuối cùng = 4 * 64 – 2)
Địa chỉ broadcast: 192.168.0.255 (Byte cuối của IP broadcast = 4 * 64 – 1)

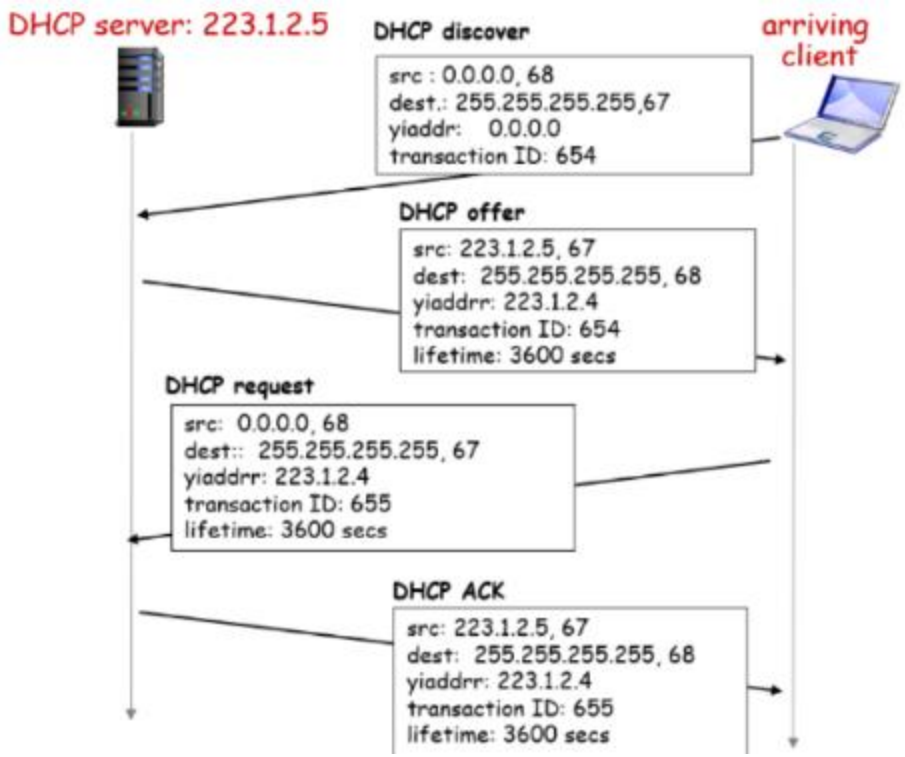
Network Address Translation

Chức năng: chuyển đổi IP private sang IP public và ngược lại
IP private (địa chỉ IP tĩnh): là địa chỉ IP chỉ có ý nghĩa đối với các thiết bị trong mạng nội bộ
- Lớp A: 10.x.x.x
- Lớp B: 172.16.0.0 – 172.31.255.255
- Lớp C: 192.168.x.x



Dynamic Host Configuration Protocol

Chức năng:
- Cho phép host (máy) tự động lấy địa chỉ IP của nó từ server
trong mạng khi host đó tham gia vào mạng
- Có thể gia 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)
- Hỗ trợ người dùng di động muốn tham gia vào mạng (trong thời gian ngắn)
- Cung cấp thông tin: địa chỉ IP, subnet mask, default gateway, DNS server



Internet Control Message Protocol

Được sử dụng bởi các host và router để truyền thông tin tầng network.
Các thông điệp ICMP được gửi trong các IP datagram
- Thông báo: host, network, port, giao thức không có thực
- Phản hồi request/reply (được dùng bởi ping)

Thông điệp ICMP: loại, mã cộng thêm 8 byte đầu tiên của IP datagram gây ra lỗi

Loại	mã	Mô tả
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

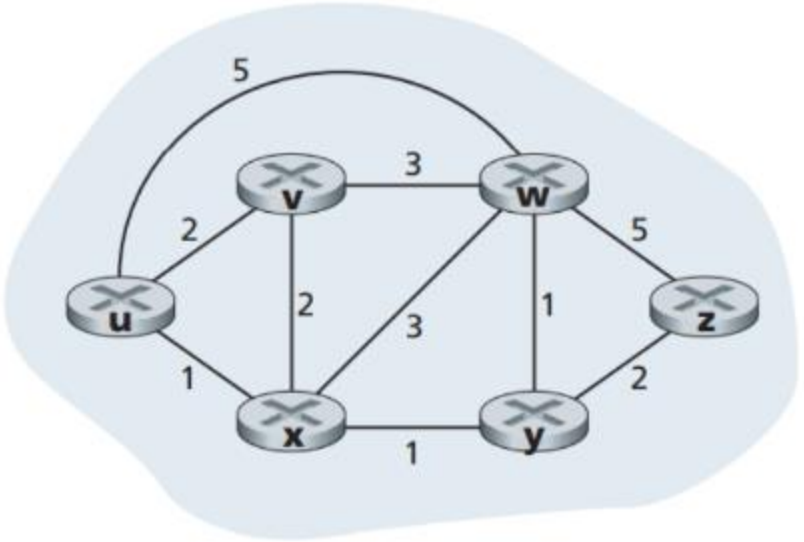
Thuật toán Routing

- Nhiệm vụ: với một mô hình mạng lưới các router, tìm đường đi có chi phí thấp nhất giữa 2 đỉnh bất kì

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), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z)}



Phân loại thuật toán Routing

Thuật toán Link State:
- Dựa trên thuật toán Dijkstra
- Được sử dụng khi tất cả các router có toàn bộ thông tin về chi phí kết nối, cấu trúc mạng
- Thuộc loại thuật toán định tuyến toàn cục

Thuật toán Distance Vector:
- Dựa trên thuật toán Bellman Ford
- Được sử dụng khi router biết các router, node lân cận được kết nối vật lý với nó và chi phí kết nối tới các node lân cận đó
- Lặp lại quá trình tính toán, trao đổi thông tin với các node lân cận
- Thuộc loại thuật toán định tuyến phân cấp

Thuật toán Link State

Giải thuật:

1. Khởi tạo tập đỉnh N rỗng
2. Bắt đầu từ K bất kì. Lần lượt xác định chi phí từ K đến các node lân cận
3. Sau mỗi bước, chọn ra node có chi phí thấp nhất. Thêm node đó vào tập N
4. Lặp lại bước 2. Cộng chi phí mới vào chi phí cũ của K

Lưu ý:

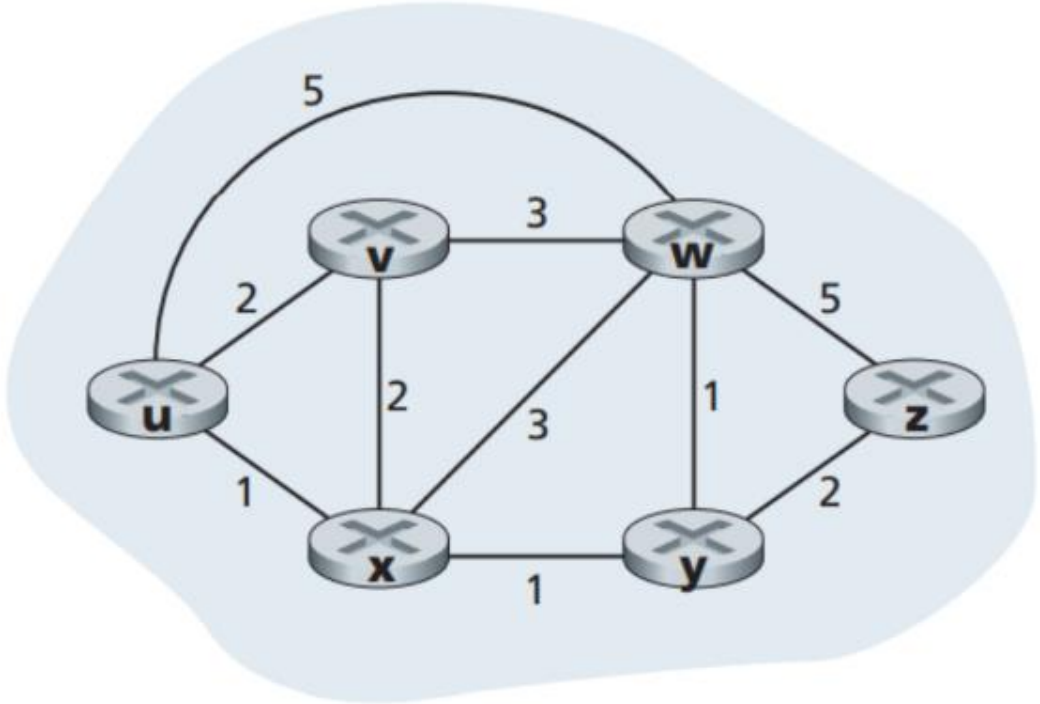
- Các node không có đường đi từ K (không lân cận với K) thì để ∞ (hoặc x)
- Nếu chi phí mới bằng chi phí cũ thì chọn đường đi nào cũng được
- Có thể có nhiều cây đường đi ngắn nhất
- Khi vẽ cây đường đi ngắn nhất, để ý đỉnh mà nó xuất phát

Ký hiệu:

- c(x, y): chi phí kết nối từ node x tới node y, bằng ∞ nếu không kết nối trực tiếp đến node lân cận
- 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

Thuật toán Link State

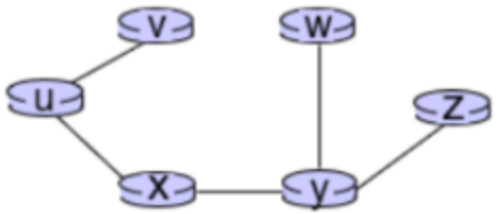
Ví dụ:
Tìm đường đi ngắn nhất từ đỉnh u tới các đỉnh còn lại, vẽ bảng forwarding



Thuật toán Link State

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	∞	∞
1	ux	2, u	4, x		2, x	∞
2	uxy	2, u	3, y			4, y
3	uxyv		3, y			4, y
4	uxyvw					4, y
5	uxyvwz					

Cây đường đi ngắn nhất



Bảng forwarding

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

Thuật toán Distance Vector

Giải thuật:
1. Khởi tạo tập đỉnh N rỗng
2. Bắt đầu từ K bất kì. Lần lượt xác định chi phí từ K đến các node lân cận
3. Sau mỗi bước, chọn ra node có chi phí thấp nhất. Thêm node đó vào tập N
4. Lặp lại bước 2. Cộng chi phí mới vào chi phí cũ của K

Lưu ý:
- Các node không có đường đi từ K (không lân cận với K) thì để ∞ (hoặc x)
- Nếu chi phí mới bằng chi phí cũ thì chọn đường đi nào cũng được
- Có thể có nhiều cây đường đi ngắn nhất
- Khi vẽ cây đường đi ngắn nhất, để ý đỉnh mà nó xuất phát

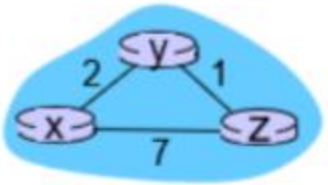
Ký hiệu:
- c(x, y): chi phí kết nối từ node x tới node y, bằng ∞ nếu không kết nối trực tiếp đến
node lân cận
- D(v): giá trị chi phí hiện tại của đường đi từ nguồn tới đích v
- p(v): node trước nằm trên đường đi từ nguồn tới v

Thuật toán Distance Vector

Công thức: dx(y) = min {c(x, y) + dv(y)}

- dx(y) là chi phí đường đi từ x tới y
- c(x, v) là chi phí từ x tới v (lân cận của x)
- dv(y) là chi phí đường đi từ v tới y (lân cận của v)

Giải thuật:
1. Khởi tạo bảng distance vector cho mỗi node
2. Cập nhật lại bảng distance vector từ các node lân cận
3. Hội tụ, tổng hợp các bảng distance vector



1. Khởi tạo:
- x: (x, y, z) = (0, 2, 7)
- y: (x, y, z) = (2, 0, 1)
- z: (x, y, z) = (7, 1, 0)

2. Cập nhật lại distance vector từ các node lân cận:
- x: (x, y, z) = (0, 2, 3)
- y: (x, y, z) = (2, 0, 1)
- z: (x, y, z) = (3, 1, 0)

3. Hội tụ:
- x: (x, y, z) = (0, 2, 3)
- y: (x, y, z) = (2, 0, 1)
- z: (x, y, z) = (3, 1, 0)

Routing trong Internet

Định tuyến Intra – AS: còn gọi là IGP (Interior Gateway Protocol)

Các giao thức định tuyến Intra – AS phổ biến:
- RIP: Routing Information Protocol (công bố năm 1982 trong BSD – UNIX)
• Sử dụng thuật toán Distance Vector
1. Đơn vị đo khoảng cách: số lượng hop (max = 15), mỗi link có giá trị là 1
2. Các DV được trao đổi giữa các node lân cận mỗi 30 giây trong thông điệp phản hồi (còn gọi là advertisement)
3. Mỗi advertisement liệt kê lên đến 25 subnet đích

- OSPF: Open Shortest Path First
• Sử dụng thuật toán Link State
1. Quảng bá gói tin LS
2. Bản đồ cấu trúc mạng tại mỗi node
3. Tính toán đường đi dùng thuật toán Dijkstra
• Thông điệp quảng bá OSPF chứa 1 mục thông tin cho mỗi router lân cận
• Các thông tin này được phát tán tới toàn bộ AS

Bài tập

Một Host có địa chỉ IP: 113.160.111.143/19
a. Host trên thuộc mạng có chia mạng con không? Nếu có thì bao nhiêu mạng con và bao nhiêu host trên mỗi mạng?
b. Hãy cho biết địa chỉ đường mạng chứa host.
c. Hãy cho biết địa chỉ Broadcast của mạng và liệt kê danh sách các host.

Đáp án:
a. Địa chỉ mạng lớp A, subnet mask = 255.255.224.0 => Số bit chia mạng con: 11 bit.
Số mạng con = 2^{11} = 2048 mạng con.
Số host mỗi mạng = 2^{13} –2 = 8190 host.

b. Địa chỉ đường mạng:
113.160.111.143 = 01110001.10100000.011|01111.10001111
=> Network ID: 113.160.96.0
c. Broadcast: 113.160.127.255
Host đầu: 192.160.96.1
Host cuối: 192.160.127.254

Nội dung

Tầng liên kết và LAN

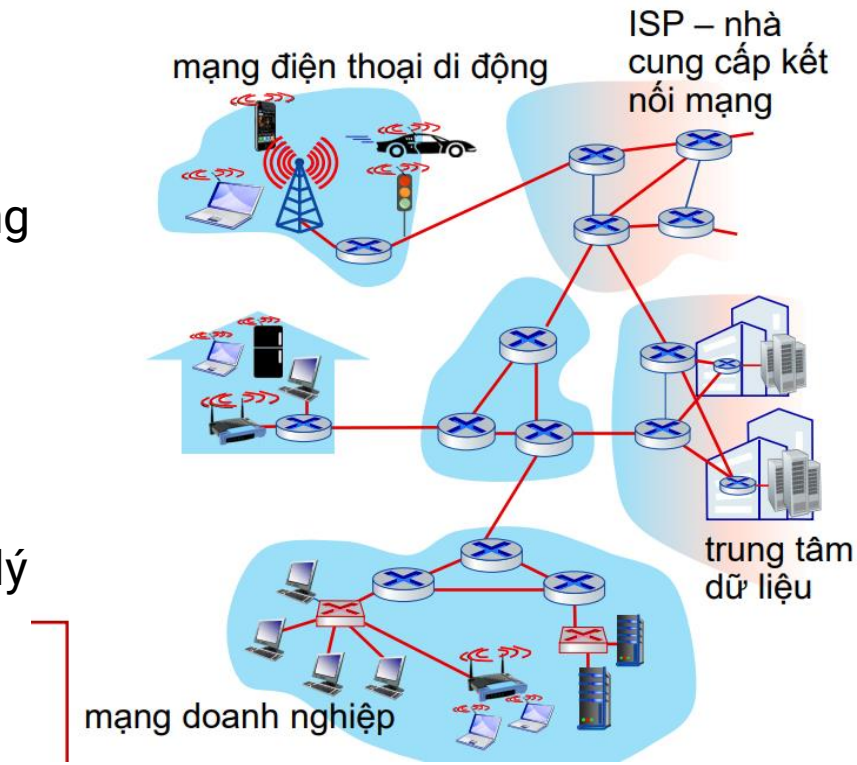
1. Mục tiêu:

- Hiểu về các nguyên lý của các dịch vụ tầng Liên kết dữ liệu:
  - Phát hiện và sửa lỗi
  - Chia sẻ kênh broadcast: đa truy cập
  - Địa chỉ lớp liên kết
  - Mạng cục bộ: Ethernet, Vlan
- Khởi tạo và hiện thực một số công nghệ tầng Liên kết dữ liệu.

Tầng liên kết và LAN

2. Các thuật ngữ:

- Host và router: các nút (node)
- Liên kết (links): Các kênh liên lạc kết nối các nút lân cận với nhau:
  - Kết nối có dây (wired links): cáp Ethernet, cáp quang, cáp đồng trục
  - Kết nối không dây (wireless links): Wi-Fi, Bluetooth, LTE
  - LANs: Ethernet, Wi-Fi, Bluetooth
- Gói tin thuộc lớp 2: frame, đóng gói datagram
- Tầng liên kết có nhiệm vụ truyền 1 node đến node lân cận vật lý trên một đường kết nối



Tầng liên kết và LAN

3. Các dịch vụ tầng liên kết:

- Truy cập liên kết, đóng gói tin (framing):

- Đóng gói datagram vào trong frame, thêm header và trailer

- Truyền tin cậy giữa các node lân cận (adjacent nodes)

- Điều khiển luồng (flow control):

- Điều khiển tốc độ truyền giữa các node gửi và nhận liền kề nhau

- Phát hiện lỗi (error detection):

- Lỗi gây ra bởi suy giảm tín hiệu, nhiễu
- Bên nhận phát hiện sự xuất hiện lỗi (thông báo bên gửi truyền lại hoặc bỏ frame đó)

- Sửa lỗi (error correction):

- Bên nhận xác định và sửa các bit lỗi mà không cần phải truyền lại

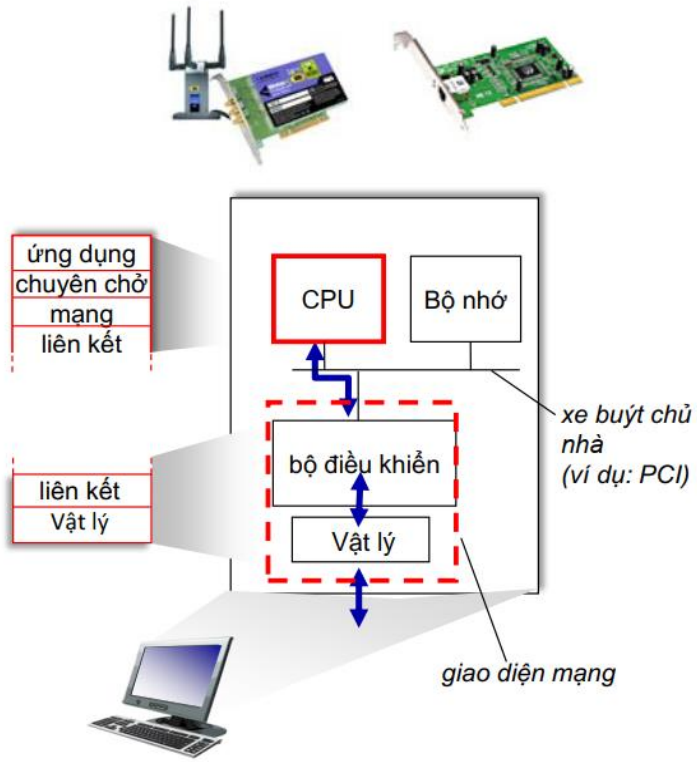
- half - duplex và full - duplex:

- Các node tại các đầu cuối của mỗi kết nối đều có thể truyền và nhận, nhưng với half - duplex thì thao tác này không được thực hiện đồng thời

Tầng liên kết và LAN

4. Tầng liên kết được triển khai ở đâu?

- Trong mỗi host
- Tầng liên kết được triển khai trong *card mạng* (NIC) hoặc trên các chip
  - Ethernet, card Wifi hoặc chip
  - Triển khai trong tầng physical và tầng link
- Gắn vào các bus hệ thống của host
- Hoặc tổ hợp của hardware, soft ware, firmware



Phát hiện lỗi và sửa lỗi

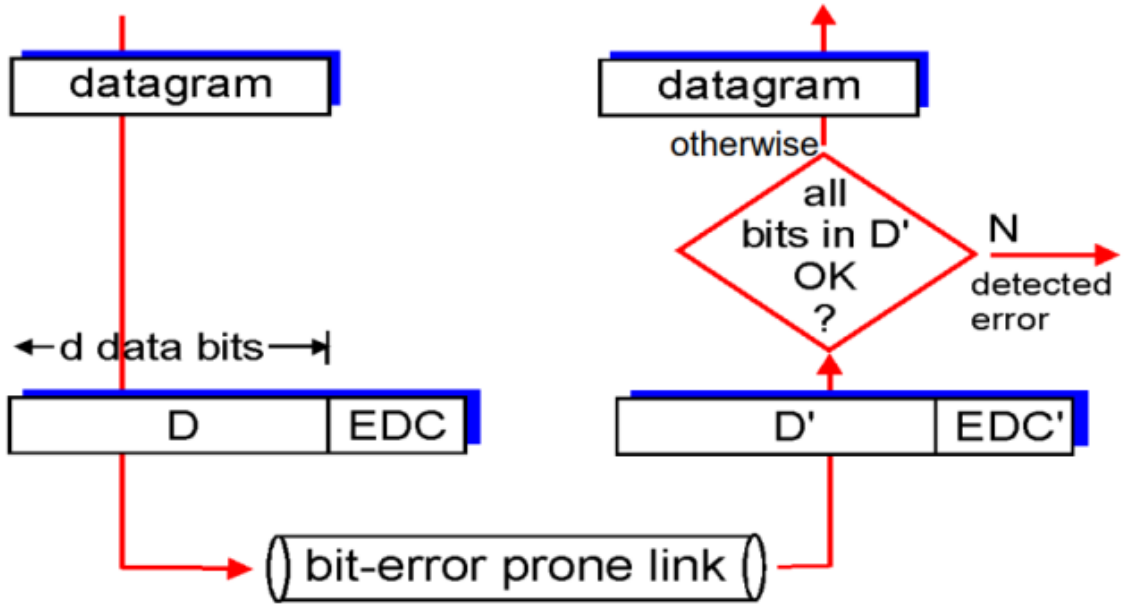
1. Phát hiện lỗi:

EDC (error detection and correction): các bit phát hiện và sửa lỗi

D (data): dữ liệu được bảo vệ bằng cách kiểm tra lỗi, có thể bao gồm các trường header

Chức năng phát hiện lỗi không hoàn toàn đáng tin cậy 100%!

- Giao thức có thể bỏ qua một số lỗi
- Trường EDC càng lớn thì việc phát hiện và sửa lỗi càng tốt hơn

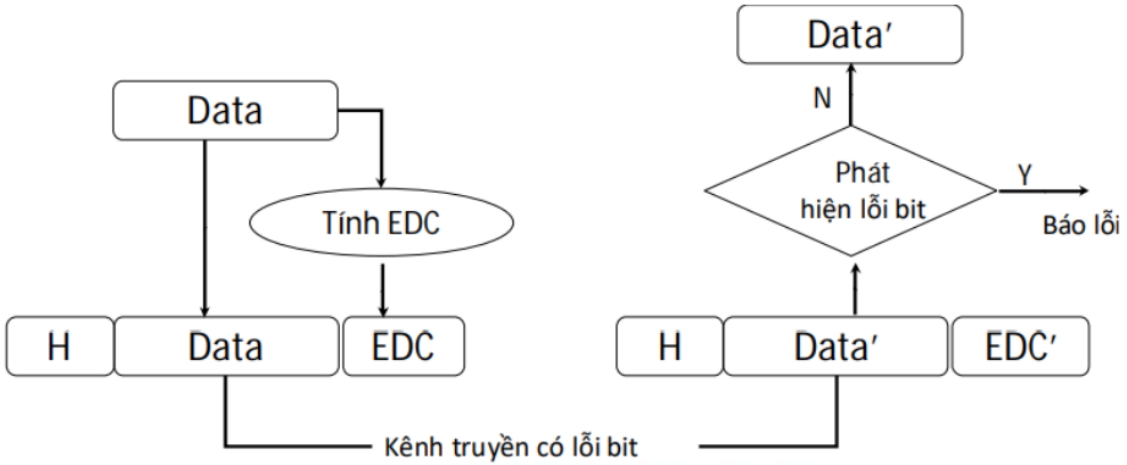


Phát hiện lỗi và sửa lỗi

2. Nguyên lý phát hiện lỗi:

Có rất nhiều loại EDC, một số loại khá phổ biến như sau:

- Mã parity
- Mã checksum
- Mã vòng CRC (Cyclic Redundancy Check)



Phát hiện lỗi và sửa lỗi

3. Kiểm tra chẵn lẻ (Parity checking):

- Parity 1 chiều: thể hiện dưới dạng 1 ma trận có kích thước 1xM
  - Số bit parity: 1 bit
  - Có 2 mô hình cho parity bit:
    - Mô hình chẵn: số bit 1 trong dữ liệu gửi đi là một số chẵn
    - Mô hình lẻ: số bit 1 trong dữ liệu gửi đi là một số lẻ

VD: **Bên gửi:** gửi dữ liệu d = 01010100 theo mô hình parity chẵn

Bên nhận:

- Nhận dữ liệu D = 010101001 - Lý tưởng
- Sai 1 bit D = 000101001 - Phát hiện sai
- Sai 2 bit D = 001100101 - Không phát hiện lỗi

➢ Nếu bên nhận kiểm trong mô hình parity bit bị sai, thì kết luận dữ liệu lỗi

➢ Ngược lại, không thể kết luận được gì. Nên có thể kết luận dạng: "Dữ liệu có thể bị lỗi hoặc là không".

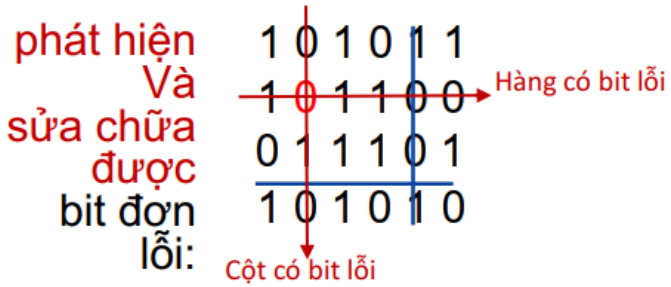
Phát hiện lỗi và sửa lỗi

3. Kiểm tra chẵn lẻ (Parity checking):

- Parity 2 chiều: thể hiện dưới dạng 1 ma trận có kích thước MxN
  - Số bit parity: N + M + 1 bit
  - Có 2 mô hình cho parity bit:
    - Mô hình chẵn: số bit 1 trong dữ liệu gửi đi là một số chẵn
    - Mô hình lẻ: số bit 1 trong dữ liệu gửi đi là một số lẻ

VD:

không có lỗi:1 0 1 0 1 | 1
1 1 1 1 0 | 0
0 1 1 1 0 | 1
1 0 1 0 1 | 0

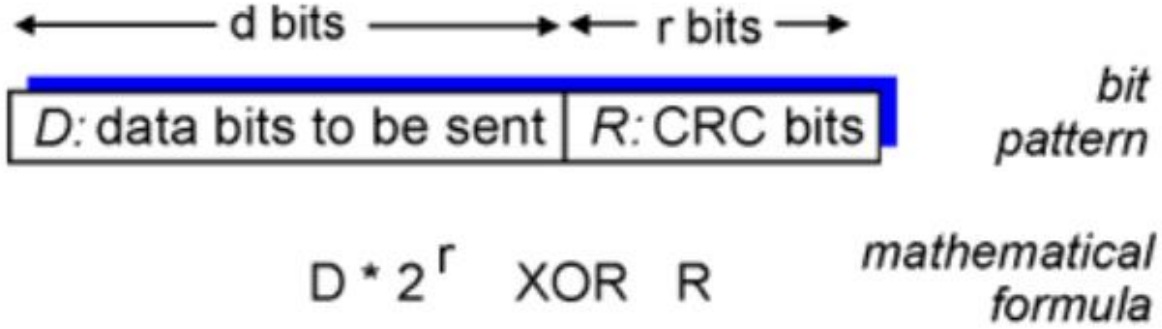


Lưu ý: Khả năng phát hiện lỗi cao hơn Parity bit 1 chiều
Chỉ sửa được lỗi khi và chỉ khi có đúng 1 bit lỗi

Phát hiện lỗi và sửa lỗi

4. **Cyclic Redundancy Check (CRC):**

- **Khái niệm:** là một phương pháp để phát hiện lỗi bit bằng cách gắn thêm 1 khối bit phía sau khối dữ liệu
- **Khối bit CRC:** Thể hiện các bit bổ sung vào sau khối dữ liệu
- **Mã CRC:** thể hiện phần dư của phép chia nhị phân không nhớ
- **Mô hình:**



- D: bit dữ liệu
- r: số bit lỗi có thể phát hiện
- G: mẫu bit, có độ dài r + 1 bit

✓ Mục tiêu: chọn r bit CRC, R, sao cho <D,R> chia hết cho G

Phát hiện lỗi và sửa lỗi

4. **Cyclic Redundancy Check (CRC):**

VD: Cho 1 dữ liệu X : 100100, được mã hóa rời dạng CRC với số chia có dạng 1101. Tìm CRC?

$$
\begin{array}{l}
1101\overline{)100100000} \\
\quad 1101 \\
\quad \overline{\quad 1000} \\
\quad\ \ 1101 \\
\quad\ \ \overline{\quad 1010} \\
\quad\ \ \ 1101 \\
\quad\ \ \ \overline{\quad 1110} \\
\quad\ \ \ \ 1101 \\
\quad\ \ \ \ \overline{\quad 0110} \\
\quad\ \ \ \ \ 0000 \\
\quad\ \ \ \ \ \overline{\quad 1100} \\
\quad\ \ \ \ \ \ 1101 \\
\quad\ \ \ \ \ \ \overline{\quad 001} \longrightarrow \text{CRC}
\end{array}
$$

Các giao thức truy cập

Ta có 2 kiểu kết nối:

- Point to point: liên kết điểm - điểm giữa bộ chuyển mạch ethernet, thiết bị đầu cuối, cho truy cập quay số

- Broadcast: chia sẻ đường truyền chung

  - Áp dụng: Ethernet mô hình cũ, Upstream HFC, 802.11 wireless LAN
  - Kênh Broadcast được chia sẻ
  - ➢ Nhiều node có thể truyền đồng thời
  - ➢ Một node có thể nhận được nhiều tín hiệu cùng lúc
  - ➢ *Sự xung đột* là khi một nút nhận được hai hoặc nhiều tín hiệu cùng một lúc

Các giao thức truy cập

Giao thức đa truy cập lý tưởng:

- Cho trước: kênh đa truy cập (MAC) với tốc độ R bps
  - Khi 1 node muốn truyền, nó có thể gửi dữ liệu với tốc độ R
  - Khi M node muốn truyền, mỗi node có thể gửi với tốc độ trung bình R/M

Được chia làm 3 lớp:

- *Phân hoạch kênh (channel partitioning)*
- *Truy cập ngẫu nhiên (random access)*
- *Truyền luân phiên*

Các giao thức truy cập

1. Phân hoạch kênh (channel partitioning)

- Khái niệm:

- Là chia kênh thành các “kênh nhỏ” (khe thời gian, tần số, mã)
- Phân bổ các kênh nhỏ đã chia cho các nút để sử dụng độc quyền
- Có 2 loại phổ biến: *TDMA* và *FDMA*

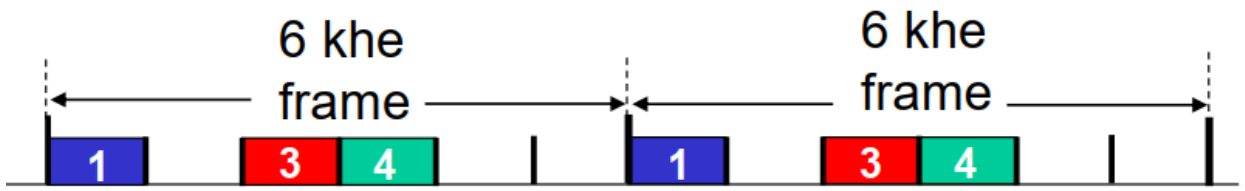
Các giao thức truy cập

1. Phân hoạch kênh (channel partitioning)

- *TDMA: Time division multiple access:* đa truy cập phân chia theo thời gian

- Truy cập kênh truyền theo hình thức "xoay vòng"
- Mỗi trạm có slot với thời gian để gửi nhất định
- Các trạm không truyền trong slot của mình thì các trạm khác cũng không được sử dụng slot đó

VD: LAN 6 trạm, 1,3,4 có gói để gửi, các khe 2,5,6 nhàn rỗi



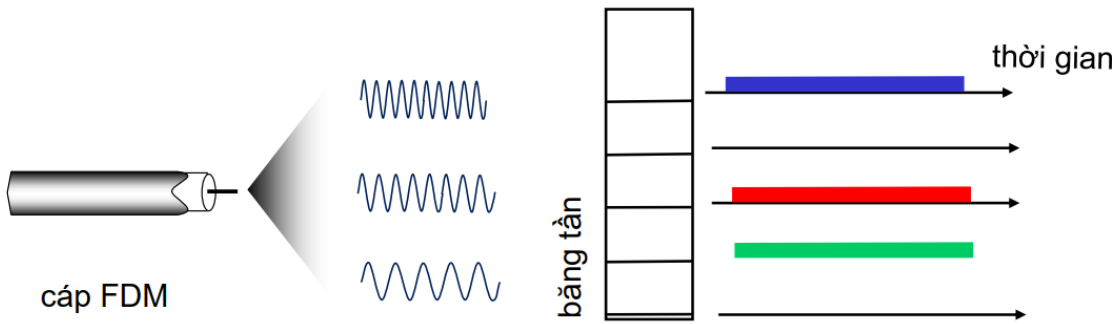
Các giao thức truy cập

1. Phân hoạch kênh (channel partitioning)

- ***FDMA: Frequency division multiple access:*** đa truy cập phân chia theo tần số

- Phổ kênh truyền được chia thành các dải tần số
- Mỗi trạm được gán một dải tần số cố định
- Trong thời gian không truyền, các dải tần số trống

VD: LAN 6 trạm, 1,3,4 có gói để gửi, các khe 2,5,6 nhàn rỗi



Các giao thức truy cập

2. Truy cập ngẫu nhiên (random access)

Khái niệm:

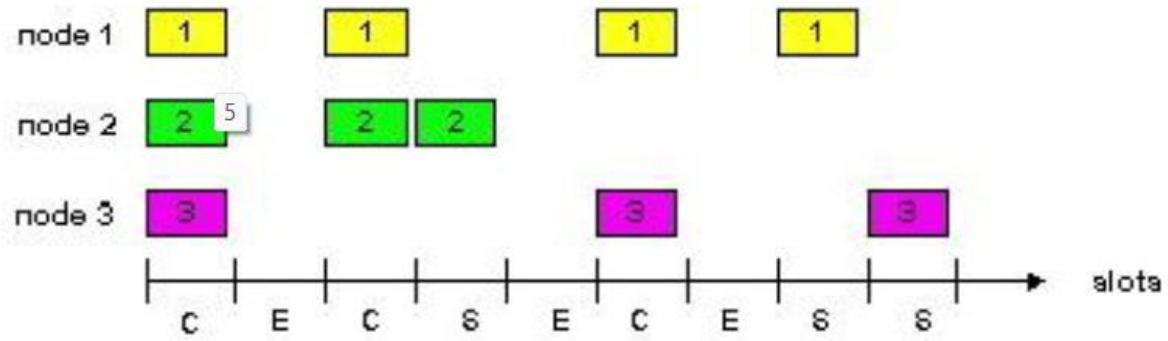
- Kênh truyền không được chia nhỏ, cho phép đụng độ và xử lý
- Khi 1 node có packet cần gửi:
  - Truyền dữ liệu với trọn tốc độ của kênh truyền dữ liệu R
  - Không có sự ưu tiên giữa các node
- Giao thức: Slotted ALOHA, Pure ALOHA, CSMS, CSMA/CD, CSMA/CA

Các giao thức truy cập

2. Truy cập ngẫu nhiên (random access)

a. Slotted ALOHA

- Chia thời gian thành các slot bằng nhau
- Các frame có kích thước bằng nhau
- Bắt đầu gửi lúc bắt đầu slot
- Khi node có được frame mới, nó sẽ truyền trong slot kế tiếp:
  - Nếu không có đụng độ: node có thể gửi frame mới trong slot kế tiếp
  - Nếu có đụng độ: node truyền lại frame trong mỗi slot tiếp theo với xác xuất *p* cho đến khi thành công

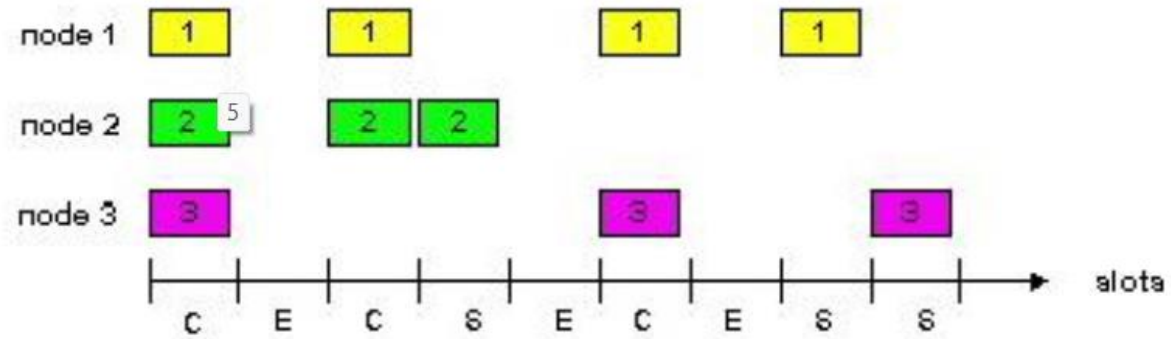


Các giao thức truy cập

2. Truy cập ngẫu nhiên (random access)

a. Slotted ALOHA

- Ưu điểm:
  - Đơn giản
  - Node hoạt động có thể truyền liên tục với tốc độ tối đa của kênh
  - Phân tán cao: chỉ có các slot trong các node cần được đồng bộ
- Nhược điểm:
  - Đụng độ, lãng phí slot
  - Các node có thể phát hiện tranh chấp nhanh hơn thời gian để truyền gói
  - Đồng bộ hóa



Các giao thức truy cập

2. Truy cập ngẫu nhiên (random access)

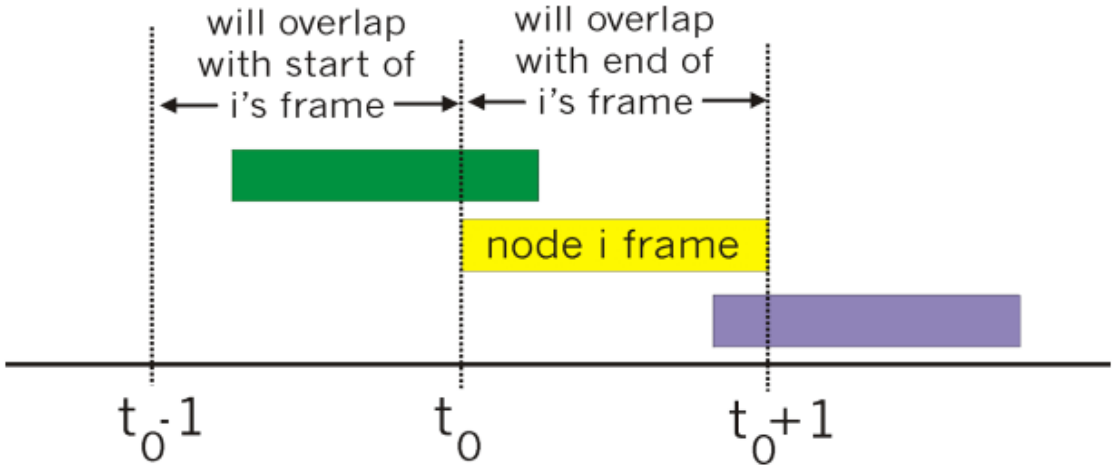
b. Pure ALOHA

- Khi frame đến: truyền lập tức

- Khả năng đụng độ tăng: frame được truyền tại thời điểm t0 đụng độ với các frame khác được truyền trong thời điểm $[t_0 - 1, t_0 + 1]$

- Ưu điểm:
  - Đơn giản, không động bộ
  - Truyền đi ngay lần đầu các frame đến

- Nhược điểm:
  - Khả năng va chạm tăng hơn so với Slotted ALOHA
  - Hiệu suất thấp hơn ½ Slotted ALOHA

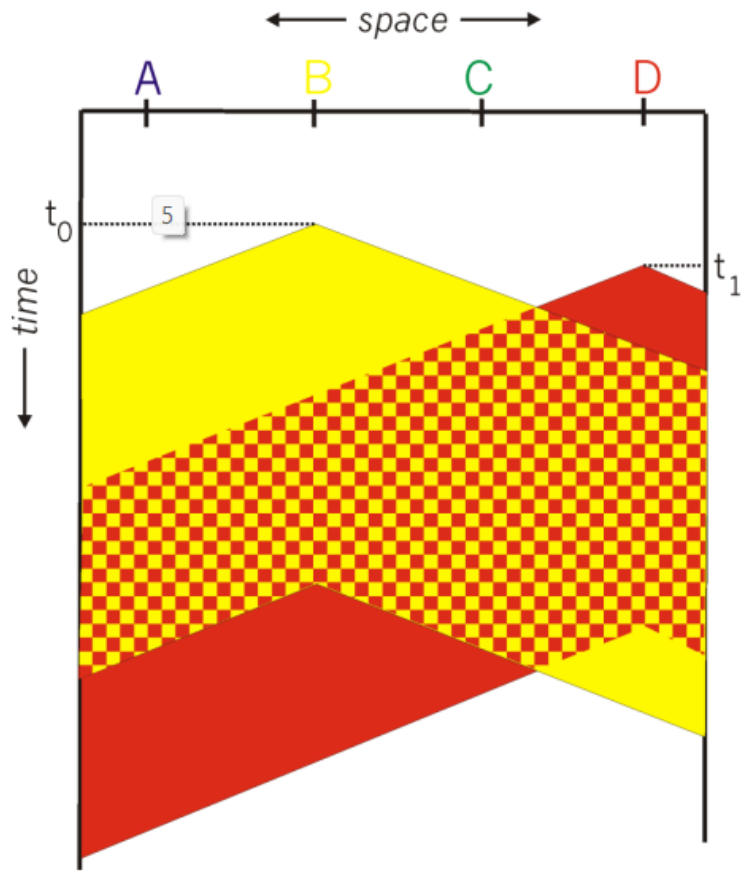


Các giao thức truy cập

2. Truy cập ngẫu nhiên (random access)

c. CSMA

- Lắng nghe kênh truyền
- Nếu kênh truyền nhàn rỗi: truyền toàn bộ frame
- Nếu kênh truyền bận: trì hoãn truyền
- Đụng độ vẫn có thể xảy ra do trễ lan truyền, nghĩa là 2 node không thể nghe thấy quá trình truyền lẫn nhau
- Đụng độ: toàn bộ thời gian truyền packet bị lãng phí

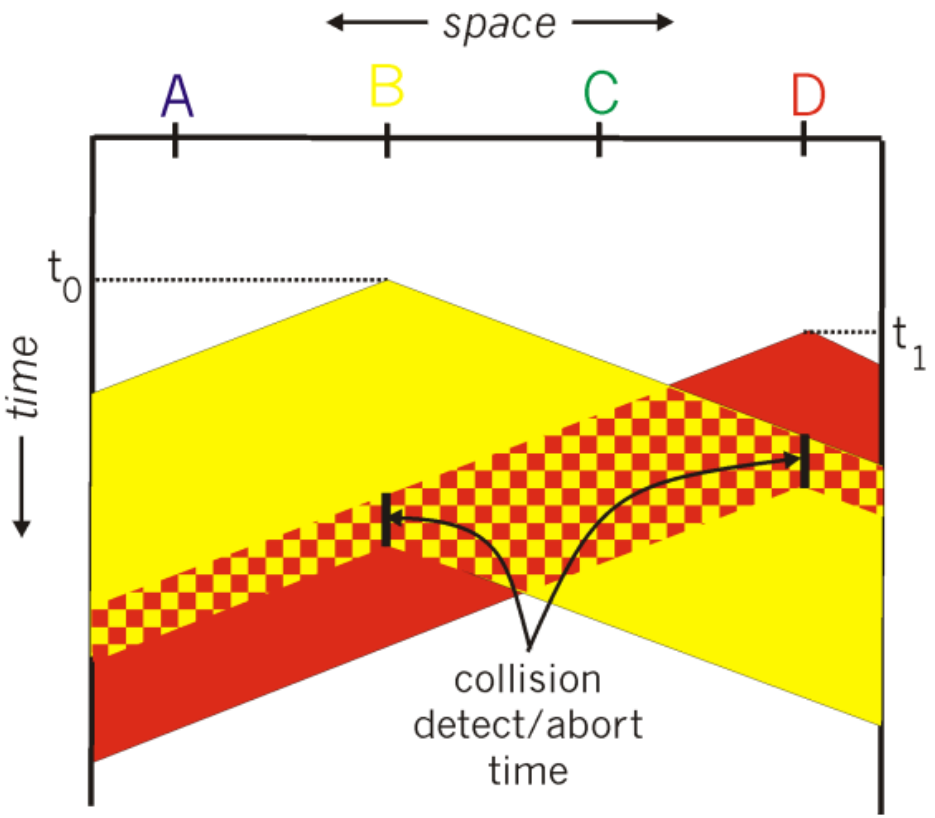


Các giao thức truy cập

2. Truy cập ngẫu nhiên (random access)

d. CSMA/CD (CSMA với kỹ thuật phát hiện xung đột)

- Đụng độ được phát hiện trong thời gian ngắn
- Thông tin đang truyền bị hủy bỏ, giảm sự lãng phí kênh
- Khi phát hiện đụng độ:
  - Phát hiện dễ trong mạng LAN hữu tuyến: đo chiều dài tín hiệu, so sánh tín hiệu đã truyền và tín hiệu nhận được
  - Phát hiện khó trong mạng LAN vô tuyến: chiều dài tín hiệu nhận được bị chiều dài cuộc truyền cụ bộ lấn át



Các giao thức truy cập

2. Truy cập ngẫu nhiên (random access)

e. Thuật toán CSMA/CD Ethernet

- NIC (network interface card) nhận datagram từ tầng network, tạo frame

- Nếu NIC dò được kênh rỗi, nó sẽ bắt đầu việc truyền frame. Nếu NIC dò được kênh bận, đợi cho đến khi kênh rảnh, sau đó mới truyền

- Nếu NIC truyền toàn bộ frame mà không phát hiện việc truyền khác, NIC được truyền toàn bộ frame đó!

- Nếu NIC phát hiện có phiên truyền khác trong khi đang truyền, thì nó sẽ hủy bỏ việc truyền và phát tín hiệu tắc nghẽn

- Sau khi hủy bỏ truyền, NIC thực hiện binary (exponential) backoff:

  - Sau lần đụng độ thứ m, NIC chọn ngẫu nhiên số K trong khoảng $\{0,1,2,\dots,2m-1\}$. NIC sẽ đợi $K * 512 * (thời\ gian\ truyền\ 1\ bit)$ và quay lại bước 2
  - Càng nhiều đụng độ dẫn đến sẽ có khoảng thời gian chờ lâu hơn

Các giao thức truy cập

2. Truy cập ngẫu nhiên (random access)

e. Thuật toán CSMA/CD Ethernet

- Độ hiệu quả = $\frac{1}{1+5\frac{T_{proc}}{T_{trans}}}$

Với: T_{proc} là độ trễ truyền lớn nhất giữa 2 node trong mạng LAN

T_{trans} là thời gian để truyền frame có kích thước lớn nhất

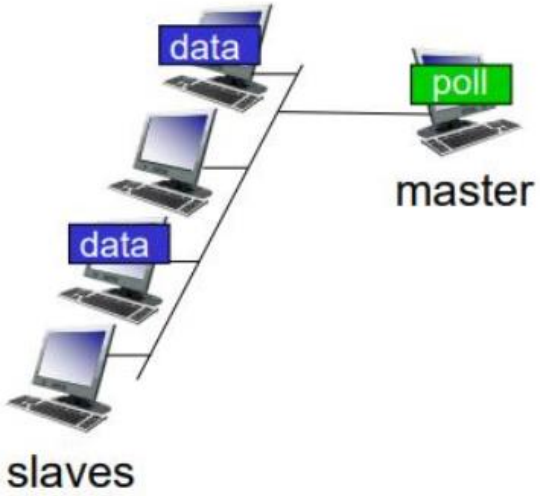
Các giao thức truy cập

3. Truyền luân phiên

- Các node thay phiên nhau, node có nhiều dữ liệu được giữ thời gian truyền lâu hơn

- Polling:

- Node chủ (master) mời các node con (slave) truyền lần lượt.
- Lượt truyền được cấp phát cho trạm tớ có thể bằng cách: trạm chủ dành phần cho trạm tớ hoặc trạm tớ yêu cầu và được trạm chủ đáp ứng.
- Thường sử dụng với các máy con “không thông minh”.



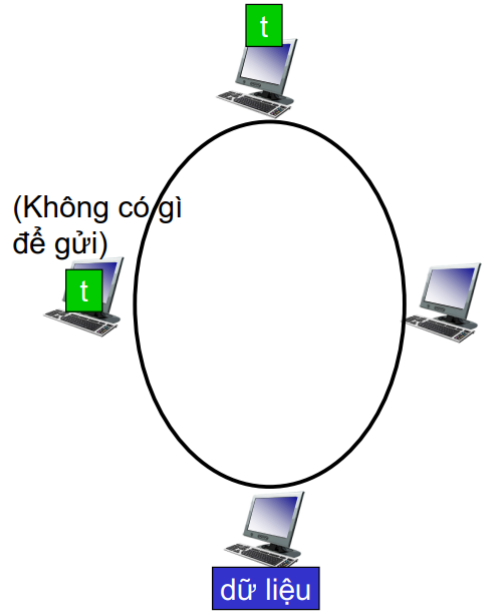
Các giao thức truy cập

3. Truyền luân phiên

- Các node thay phiên nhau, node có nhiều dữ liệu được giữ thời gian truyền lâu hơn

- Chuyển token:

- Điều hành việc chuyển token tuần tự từ 1 node đến node kế tiếp.
- Trạm nào có trong tay token sẽ được quyền truyền, truyền xong phải chuyền token qua trạm kế tiếp



Mạng LAN

1. Địa chỉ MAC

a. Giới thiệu:

- Chức năng: được sử dụng “cục bộ” để chuyển frame từ 1 interface đến 1 interface khác được kết nối vật lý trực tiếp với nhau
- Độ dài: 48 bits (biểu dễn bằng hệ hexa)
- Vị trí: thường được ghi vào trong NIC ROM hoặc thiết lập trong phần mềm và được IEEE quản lý

VD: Windows: 1A - 2F - BB - 76 - 09 - AD *(ký hiệu thập lục phân, mỗi ký tự đại diện cho 4 bit)*

MacOS, Linux: 00:BD:23:F5:D5:G8

Cisco: 0004.23F5.D5G8

Câu hỏi: Phân biệt địa chỉ MAC và địa chỉ IP?

Trả lời: Địa chỉ IP được dùng để xác định vị trí thiết bị còn địa chỉ MAC dùng để xác định thiết bị

Mạng LAN

1. Địa chỉ MAC

b. ARP (Address Resolution Protocol)

- Chức năng: Mỗi nút IP (host, router) trên mạng LAN đều có bảng ARP của nó (Bảng ARP: Lưu ánh xạ IP/MAC của 1 số nút trên LAN)

- Cách thức hoạt động:

  - A muốn gửi 1 datagram cho B với địa chỉ MAC của B không có trong ARP cho A
  - A sẽ gửi broadcast gói tin ARP query có chứa địa chỉ IP của mình
    - Địa chỉ MAC đích là: FF-FF-FF-FF-FF-FF
    - Tất cả các node trên mạng LAN sẽ nhận ARP query này
  - B nhận gói tin ARP, trả lời đến A với địa chỉ MAC của B (unicast)
  - A sẽ lưu cặp địa chỉ IP - MAC trong bảng ARP của nó cho tới khi thông tin này hết hạn sử dụng

- ARP ⇒ Plug-and-play (cắm và chạy): chỉ cần kết nối là hoạt động, không cần cài đặt phức tạp.

Mạng LAN

1. Địa chỉ MAC

b. ARP (Address Resolution Protocol)

- Nếu định tuyến tới mạng LAN khác:

- A tạo IP datagram với IP nguồn A, đích B
- A tạo frame tầng liên kết dữ liệu với địa chỉ MAC của R là địa chỉ đích, frame này chứa IP datagram từ A đến B
- Frame được gửi từ A tới R
- Frame được R nhận, datagram được gỡ bỏ, được chuyển tới IP
- R sẽ chuyển tiếp datagram với IP nguồn A, đích B
- R tạo frame này chứa IP datagram từ A tới B

Mạng LAN

2. Ethernet

a. Giới thiệu:

- Ethernet là công nghệ mạng LAN “chiếm ưu thế”
- Công nghệ mạng LAN được sử dụng rộng rãi
- Đơn giản, rẻ hơn mạng LAN token và ATM
- Tăng tốc độ trung bình từ 10 Mbps - 10 Gbps

Mạng LAN

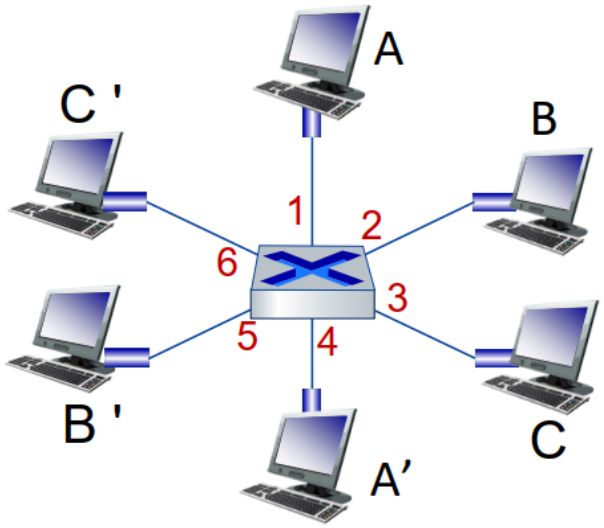
2. Ethernet

b. Các cấu trúc liên kết mạng

Dạng bus: phổ biến trong những thập niên 90. Tất cả các node trong cùng vùng xung đột (collision domain) (có thể đụng độ lẫn nhau).

Dạng star: chiếm ưu thế hiện nay

- Switch hoạt động ở trung tâm
- Mỗi chặng kết nối Ethernet hoạt động riêng biệt



Mạng LAN

2. Ethernet

c. Cấu trúc Ethernet frame:

- Preamble (8 byte): 7 byte đầu, mỗi byte có giá trị 10101010; byte cuối có giá trị 10101011. Sử dụng để đồng bộ hóa tốc độ đồng hồ của người nhận, người gửi
- Address (6 byte): gồm 6 byte địa chỉ MAC nguồn, đích
- Type: chỉ ra giao thức tầng cao hơn
- CRC: kiểm tra lỗi bên nhận - loại bỏ frame nếu phát hiện lỗi

Mạng LAN

3. Switch

- Switch là một thiết bị thực hiện chức năng chuyển mạch trong mạng
- Các host kết nối trực tiếp với switch
- Mỗi kết nối là 1 vùng đụng độ riêng
- Switch Fowarding

- Plug and play: switch không cần được cấu hình sẵn
- Switch tự học các vị trí mà các host có thể chuyển tới được thông qua các gói tin được gửi đến switch
- Khi switch nhận được một frame bất kì, nó lưu lại interface và địa chỉ MAC của host gửi và điền vào bảng switch forwarding
- Khi chuyển tiếp frame đi đến một vị trí chưa biết thì nó gửi frame ra tất cả các interface