

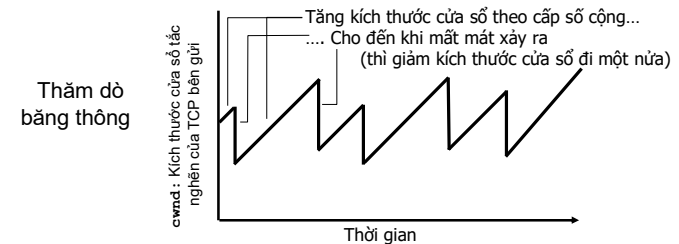
Chương 3: Nội dung

- 3.1 Các dịch vụ tầng giao vận
- 3.2 Ghép kênh và phân kênh
- 3.3 Vận chuyển không kết nối: UDP
- 3.4 Các nguyên lý truyền dữ liệu tin cậy
- 3.5 Vận chuyển hướng kết nối: TCP
 - 3.5.1 Cấu trúc đoạn dữ liệu (segment)
 - 3.5.2 Truyền dữ liệu tin cậy
 - 3.5.3 Điều khiển luồng
 - 3.5.4 Quản lý kết nối
- 3.6 Các nguyên lý điều khiển tắc nghẽn
- 3.7 Điều khiển tắc nghẽn TCP

Tầng giao vận 3-96

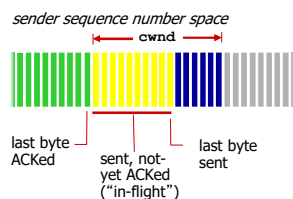
Điều khiển tắc nghẽn trong TCP: Tăng theo cấp số cộng Giảm theo cấp số nhân

- ❖ **Cách tiếp cận:** Bên gửi tăng tốc độ truyền (kích thước cửa sổ), thăm dò bằng thông sử dụng, cho đến khi có mất mát xảy ra
 - **Tăng theo cấp số cộng:** tăng **cwnd** theo 1 MSS mỗi RTT cho đến khi phát hiện mất mát
 - **Giảm theo cấp số nhân:** giảm **cwnd** đi một nửa sau khi phát hiện có mất mát



Tầng giao vận 3-97

Chi tiết điều khiển tắc nghẽn trong TCP



- ❖ Bên gửi giới hạn việc truyền:

$$\text{LastByteSent} - \text{LastByteAcked} \leq \text{cwnd}$$

- ❖ **cwnd** thay đổi, có chức năng nhận biết tắc nghẽn trên mạng

Tốc độ gửi của TCP:

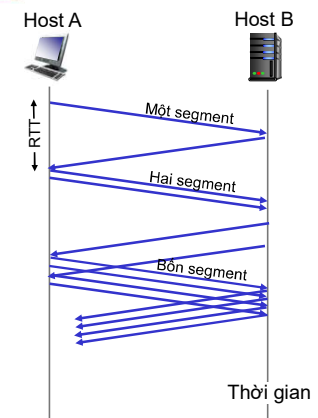
- ❖ **Được hiểu là:** gửi cwnd byte, chờ một RTT cho ACK, sau đó gửi nhiều byte hơn

$$\text{Tốc độ} \approx \frac{\text{cwnd}}{\text{RTT}} \text{ bytes/sec}$$

Tầng giao vận 3-98

TCP khởi động chậm

- ❖ Khi kết nối bắt đầu, tăng tốc độ lên theo cấp số nhân cho đến khi có sự kiện mất mát đầu tiên xảy ra:
 - Khởi tạo **cwnd** = 1 MSS
 - Tăng gấp đôi **cwnd** cho mỗi RTT
 - Thực hiện tăng **cwnd** cho mỗi ACK nhận được
- ❖ **Tổng kết:** tốc độ khởi đầu là chậm nhưng sau đó tăng lên theo cấp số nhân



Tầng giao vận 3-99

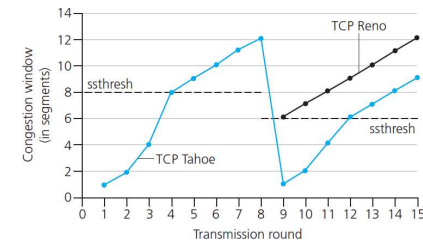
Phát hiện và phản ứng lại khi có mất mát

- Mất mát được xác định khi bị timeout:
 - cwnd** được thiết lập lại là 1 MSS;
 - Cửa sổ sau đó sẽ tăng theo cấp số nhân (như trong khởi động chậm) tới ngưỡng, thì sẽ tăng tuyến tính
- Mất mát được xác định khi thấy 3 ACK trùng lặp: TCP RENO
 - Các ACK trùng lặp xác định khả năng truyền các segment của mạng
 - cwnd** giảm đi một nửa kích thước cửa sổ, sau đó tăng tuyến tính
- TCP Tahoe luôn đặt **cwnd** là 1 (khi có timeout hoặc 3 ACK trùng lặp)

Hiện thực trong TCP

Hỏi: Khi nào nên chuyển từ tăng theo cấp số nhân sang tăng tuyến tính?

Trả lời: khi **cwnd** đạt đến 1/2 giá trị của nó trước khi timeout.



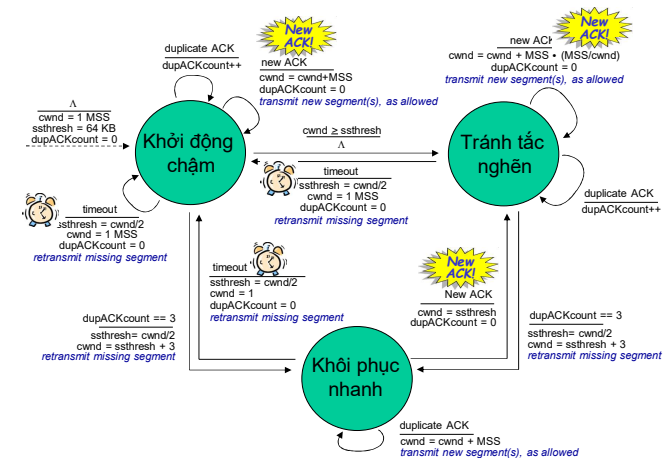
Cài đặt:

- Biến **ssthresh**
- Với mỗi sự kiện mất mát, **ssthresh** sẽ được đặt bằng 1/2 **cwnd** ngay trước khi có mất mát xảy ra

Tổng kết điều khiển tắc nghẽn trong TCP

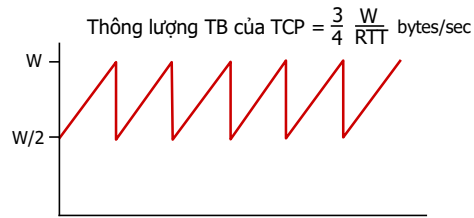
- Khi **cwnd** dưới **ssthresh**, bên gửi đang trong giai đoạn **khởi động chậm**, kích thước cửa sổ tăng nhanh theo cấp số nhân.
- Khi **cwnd** trên **ssthresh**, bên gửi đang trong giai đoạn **tránh tắc nghẽn**, kích thước cửa sổ tăng tuyến tính.
- Khi có 3 ACK trùng lặp xảy ra, **ssthresh = cwnd/2** và **cwnd = ssthresh**.
- Khi **timeout** xảy ra, **ssthresh = cwnd/2** và **cwnd=1 MSS**.

Tổng kết điều khiển tắc nghẽn trong TCP



Thông lượng của TCP

- ❖ Thông lượng trung bình của TCP được xác định qua kích thước cửa sổ và RTT như thế nào?
 - Bỏ qua khởi động chậm, giả sử dữ liệu luôn luôn được gửi
- ❖ W: kích thước cửa sổ (được tính bằng byte) khi có mất mát xảy ra
 - Kích thước cửa sổ trung bình (số byte trong lưu lượng) là $\frac{3}{4}W$
 - Thông lượng trung bình là $\frac{3}{4}W$ trên RTT



Tăng giao vận 3-104

TCP trong tương lai: TCP qua “đường truyền rộng và dài”

- ❖ Ví dụ: Các segment dài 1500 byte, RTT là 100ms, muốn đạt được thông lượng là 10 Gbps
- ❖ Yêu cầu lưu lượng với kích thước cửa sổ là $W = 83,333$ segment
- ❖ Thông lượng của xác suất mất đoạn là L [Mathis 1997]:

$$\text{Thông lượng TCP} = \frac{1.22 \cdot MSS}{RTT \sqrt{L}}$$

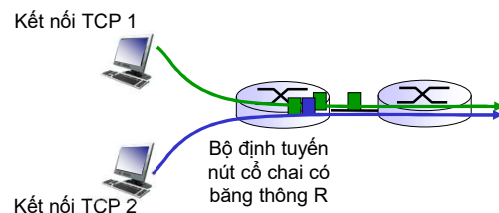
→ Để có được thông lượng là 10 Gbps, cần tỷ lệ mất mát là $L = 2 \cdot 10^{-10}$ – *một tỷ lệ mất mát rất nhỏ!*

- ❖ Các phiên bản mới của TCP dành cho tốc độ cao

Tăng giao vận 3-105

Tính công bằng trong TCP

Mục tiêu: Nếu K phiên làm việc trong TCP chia sẻ cùng liên kết nút cổ chai có băng thông là R, thì mỗi phiên nên có tốc độ trung bình là R/K

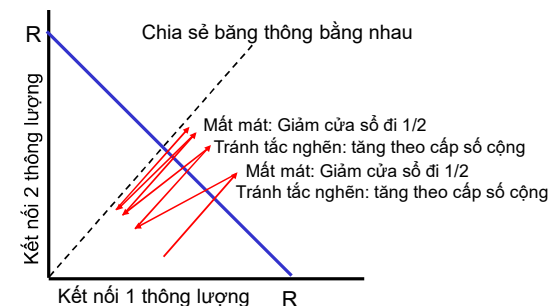


Tăng giao vận 3-106

Tại sao TCP là công bằng?

Hai phiên làm việc cạnh tranh nhau:

- ❖ Tăng theo cấp số cộng làm tăng lưu lượng liên tục
- ❖ Giảm theo cấp số nhân làm giảm lưu lượng tương ứng



Tăng giao vận 3-107

Tính công bằng (tiếp)

Tính công bằng và UDP

- ❖ Các ứng dụng đa phương tiện thường không dùng TCP
 - Không muốn tốc độ bị chặn do điều khiển tắc nghẽn
- ❖ Thay bằng dùng UDP:
 - Gửi audio/video với tốc độ ổn định, chịu mất mát gói tin

Tính công bằng và kết nối song song trong TCP

- ❖ Ứng dụng có thể mở nhiều kết nối song song giữa hai host
- ❖ Các trình duyệt web làm theo cách này
- ❖ Ví dụ: liên kết có tốc độ R hỗ trợ 9 kết nối:
 - Ứng dụng mới yêu cầu 1 TCP, có tốc độ R/10
 - Ứng dụng mới yêu cầu 11 TCP, có tốc độ R/2

Chương 3: Tổng kết

- ❖ Các nguyên lý của các dịch vụ tầng giao vận:
 - Ghép kênh, phân kênh
 - Truyền dữ liệu tin cậy
 - Điều khiển luồng
 - Điều khiển tắc nghẽn
- ❖ Hiện thực trên mạng Internet:
 - UDP
 - TCP

Tiếp theo:

- ❖ Kết thúc các vấn đề liên quan đến “phần cạnh” của mạng (tầng ứng dụng và tầng giao vận)
- ❖ Chuẩn bị đi vào “phần lõi” của mạng

Tham khảo

- Jim Kurose, Keith Ross, “*Computer Networking: A Top-Down Approach*” 8th edition, Pearson, 2020.