Chapter1: Introduction-Review Question

Câu 1: Sự khác nhau giữa 1 host và 1 end system. Liệt kê các kiểu end system. Web server có phải là 1 end system không?

Không có sự khác nhau. "host" và "end system" là 2 từ có thể thay thế cho nhau.

End system gồm có: PC, máy trạm, web server, mail server, web TV, các PDA có kết nối internet...

Câu 2: V dụ v giao thức ngo i giao (diplomatic protocol)

Giả sử Alice, một đại sứ của quốc gia A muốn mời Bob, một đại sứ nước B, ăn tối. Alice không chỉ đơn giản là chỉ cần gọi Bob trên điện thoại và nói, "Đến bàn ăn của chúng tôi bây giờ". Thay vào đó, cô gọi Bob và cho thấy một ngày và thời gian. Bob có thể đáp ứng bằng cách nói rằng ông không phải có sẵn mà cụ thể ngày, nhưng anh có sẵn một ngày khác. Alice và Bob tiếp tục gửi "thông điệp" qua lại cho đến khi họ đồng ý vào một ngày và thời gian. Bob sau đó cho thấy tại Đại sứ quán vào ngày đã thoả thuận, hy vọng không quá 15 phút trước khi hoặc sau khi thời gian đã thảo thuận, ngoại giao giao thức cũn cho phép hoặc Alice hoặc Bob lịch sự hủy bỏ sự tham gia nếu học có lý do hợp lý.

Câu 3: Chương trình client là gì? Chương trình server là gì? Một chương trình server yêu cầu và nhận dịch vụ từ 1 chương trình client đúng không?

Một chương trình mạng thường có 2 chương trình, mỗi cái chạy trên 1 host khác nhau, được nối với nhau. Chương trình mà khởi đầu việc trao đổi thông tin là client. Thông thường, chương trình client yêu cầu và nhận dịch vụ từ chương trình server.

Câu 4: Liệt kê 6 công nghệ truy cập.

1. Dial-up modem over telephone line: residential;
2. DSL over telephone line: residential or small office;
3. Cable to HFC: residential;
4. 100 Mbps switched Ethernet: company;
5. Wireless LAN: mobile;
6. Cellular mobile access (for example, WAP): mobile

Câu 7: Tốc độ truyền của LAN Ethernet là gì? Với 1 tốc độ truyền mỗi user trong mạng LAN có thể truyền liên tục với cùng tốc độ được không?

Ethernet LANs có tốc độ truyền là 10 Mbps, 100 Mbps, 1 Gbps and 10 Gbps. VD với X Mbps Ethernet (X = 10, 100, 1,000 or 10,000), một user có thể truyền lien tục với tốc độ X Mbps nếu

chỉ có user đó đang gửi dữ liệu. Nếu có nhiều user hoạt động thì mỗi user không thể truyền liên tục với tốc độ X Mbps.

Câu 8: Môi trường vật lý mà Ethernet có thể truyền qua là gì?

Ethernet có thể truyền qua cáp đồng trục mảnh và dây đồng xoắn đôi. Nó cũng có thể truyền qua sợi quang và cáp đồng trục dày.

Câu 9: Cho biết khoảng tốc độ truyền của Dial up modems, ISDN, ADSL, HFC, FTTH. Chúng thuộc kiểu chia sẻ hay riêng biệt?

Dial up modems: up to 56 Kbps, bandwidth is dedicated; ISDN: up to 128 kbps, bandwidth is dedicated; ADSL: downstream channel is .5-8 Mbps, upstream channel is up to 1 Mbps, bandwidth is dedicated; HFC, downstream channel is 10-30 Mbps and upstream channel is usually less than a few Mbps, bandwidth is shared. FTTH: 2-10Mbps upload; 10-20 Mbps download; bandwidth is not shared.

Câu 10: Mô tả các công nghệ truy cập Internet không dây phổ biến hiện nay. So s nh và nêu điểm khác biệt của chúng

Hiện nay có 2 công nghệ truy cập Internet không dây phổ biến.

- Mạng LAN không dây (Wireless LAN). Trong WLAN, người dùng không dây truyền và các nhận gói tin đến 1 trạm cơ sở (điểm truy cập không dây) trong bán kính khoảng vài chục mét. Trạm cơ sở thường được kết nối Internet bằng dây.
- Mạng truy cập không dây diện rộng: Trong các hệ thống này, các gói tin được truyền qua cùng cơ sở hạ tầng không dây được dùng cho hệ thống điện thoại, trạm cơ sở này được quản lý bởi 1 nhà cung cấp dịch vụ viễn thông. Nó cung cấp sự truy cập không dây cho người dung trong vòng bán kính vài chục kilomet từ máy trạm cơ sở.

Câu 11: Thuận lợi của mạng chuyển mạch điện so với chuyển mạch gói là gì?

Mạng chuyển mạch điện có thể đảm bảo 1 lượng cố định băng thông giữa 2 điểm nối để duy trì 1 cuộc gọi. Hầu hết các mạng chuyển mạch gói hiện nay (bao gồm cả Internet) không tạo sự đảm bảo băng thông giữa 2 điểm nối.

Câu 13: Giả sử có 1 gói tin được chuyển từ host gửi đến host nhận. Tốc độ truyền giữa host gửi và switch và giữa switch và host nhận lần lượt là R1 và R2. Giả sử switch sử dụng chuyển mạch gói để lưu trữ và chuyển tiếp. Tính tổng thời gian delay giữa 2 điểm nối để gửi 1 packet có độ dài L (bỏ qua hàng đợi độ trễ khi truyền và khi xử lý)

Tại thời điểm t_0, host gửi bắt đầu truyền. Tại thời điểm $t_1 = L/R1$, host gửi hoàn thành việc truyền và toàn bộ gói tin được nhận tại router (không có trễ khi truyền). Vì router có toàn bộ gói tin tại thời điểm t_1 nên nó có thể truyền gói tin đến host nhận tại thời điểm t_1. Tại thời điểm $t_2 = t_1 + L/R2$, router hoàn thành việc truyền và toàn bộ gói tin được nhận tại host nhận. Vì vậy, tổng thời gian trễ là L/R1 + L/R2.

Câu 15: Giả sử nhiều user chia sẻ 1 link 2Mbps. Và cũng giả sử rằng mỗi user truyền liên tục với tốc độ 1Mbps khi truyền, nhưng mỗi user chỉ truyền 20% thời gian.

a. ***Nếu dùng chuyển mạch điện thì có bao nhiêu user được sử dụng?***
b. ***Giả sử dùng chuyển mạch gói. Tại sao nếu có ít hơn hoặc bằng 2 user cùng truyền thì không phải đợi? và nếu nhiều hơn 2 user cùng truyền thì phải có hàng đợi?***
c. ***Tính xác suất để mỗi user được truyền.***

Giả sử có 3 user. Tính xác suất tại bất kì thời điểm nào, cả 3 user cùng truyền 1 lúc. Tính khoảng thời gian để hình thành hàng đợi.

TL:

a. Có 2 user được truyền vì mỗi user cần 1 nửa băng thông đường truyền.

b. Theo giả thiết, mỗi user cần 1Mbps khi truyền, nếu có 2 hoặc ít hơn 2 user cùng truyền thì tối đa cần 2Mbps băng thông. Mà đường truyền đã cho có băng thông là 2Mbps nên không có hàng đợi nào cả.

   Ngược lại, nếu 3 user cùng truyền thì băng thông cần sẽ là 3Mbps, nhiều hơn băng thông hiện có. Trong trường hợp này sẽ có hàng đợi trước khi vào đường truyền.

c. Xác suất = 0.2

d. Xác suất để 3 user cùng truyền cùng lúc = $C(3,3) . p^3 . (1-p)^{3-3} = (0.2)^3 = 0.008$. Theo gt, hàng đợi tăng khi tất cả user đang truyền, nên khoảng tgian mà hàng đợi tăng (= xác suất để tất cả 3 user cùng truyền) là 0.008.

Câu 16: Giả sử đang gửi 1 gói tin từ 1 host nguồn đến 1 host đ ch qua 1 đường cố định. Liệt kê các khoảng trễ. Cái nào là cố định, cái nào thay đổi?

Các thành phần trễ gồm: trễ khi xử lý, trễ khi lan tỏa (propagation delay), trễ khi truyền (transmission delay) và trễ khi đợi. Trừ trễ khi đợi là thay đổi, tất cả những khoảng trễ còn lại đều là cố định.

Câu 18: Tính khoảng thời gian để truyền 1 gói tin độ dài L qua 1 đường truyền khoảng cách d, tốc độ truyền là s, tốc độ chuyển giao là R bps? Nó có phụ thuộc vào L và R không?

Khoảng thời gian = d/s.

Nó không phụ thuộc vào độ dài gói tin (L) và tốc độ chuyển R.

Câu 19: Giả sử host A mướn gởi 1 file lớn đến host B, có 3 đường đi từ A đếnB, tốc độ lần lượt là R1 =

500 kbps, R2 = 2 Mbps và R3 = 1 Mbps.

a. **Giả sử không có tắc nghẽn trong mạng. Tính thông lượng truyền file?**
b. **Giả sử file có dung lượng 4 triệu byte. Chia kích thước file bằng thông lượng. Tính thời gian xấp xỉ để truyền file đến host B?**
c. **Làm câu a, b với R2 giảm còn 100 kbps.**

TL:

a. 500 kbps. *(chọn nhỏ nhất để đảm bảo cho RI đáp ứng được)*

b. 4.000.000 byte = 4.000 KB = 32.000 Kbit. Tgian = 32000/500 = 64 s

c. thông lượng = 100 kbps. Tgian = 320 s.

Câu 22: Liệt kê 5 nhiệm vụ mà 1 tầng có thể thực hiện được. Có thể xảy ra trường hợp một hay nhiều nhiệm vụ được thực hiện bởi nhiều tầng được không?

5 nhiệm vụ chung là: kiểm soát lỗi, điều khiển luồng, phân đoạn và lắp ráp lại, dồn kênh và cài đặt kết nối. Có thể xảy ra trường hợp 1 hay nhiều nhiệm vụ được thực hiện bởi nhiều tầng. VD: kiểm soát lỗi thường được cung cấp ở nhiều tầng.

Câu 24: Một message ở tầng ứng dụng là gì? Một segment ở tầng giao vận? Một datagram ở tầng network? Một frame ở tầng liên kết?

- Message: dữ liệu mà 1 ứng dụng muốn gửi và truyền xuống tầng giao vận.
- Segment: được tạo ra bởi tầng giao vận bằng cách đóng gói message của tầng ứng dụng với

header của tầng giao vận.

- Datagram: đóng gói segment của tầng giao vận với header của tầng mạng.
- Frame: đóng gói datagram của tầng mạng với header của tầng liên kết.

Câu 25: Một router xử lý ở tầng nào? Một switch xử lý ở tầng nào? Và một host xử lý ở tầng nào?

Các router xử lý ở tầng 1, 2, 3 tức là tầng vật lý, tầng liên kết và tầng network. Các switch xử lý ở tầng 1 và 2 tức là tầng vật lý và tầng liên kết. Các host xử lý ở cả 5 tầng.

Chapter1: : Introduction- Problem

P5: Giả sử có 2 host A và B kết nối với nhau bởi 1 đường có tốc độ R bps. 2 host cách nhau m mét và tốc độ lan tỏa (propagation speed) là s (m/s). Host A gửi 1 packet kích thước L đến host B.

a. **Biểu diễn độ trễ lan tỏa (d_{prop}) thông qua m và s.**
b. **Xác định thời gian truyền của gói tin d_{trans} thông qua L và R.**
c. **Bỏ qua độ trễ xử lý và hàng đợi. Viết biểu thức t nh độ trễ giữa 2 điểm kết thúc.**
d. **Giả sử Host A bắt đầu truyền gói tin tại thời điểm t = 0. T i thời điểm t = d_{trans}, bit cuối cùng của gói tin ở đâu?**
e. **Giả sử $d_{prop} > d_{trans}$. Tại thời điểm t = d_{trans}, bit đầu tiên của gói tin ở đâu?**
f. **Giả sử $d_{prop} < d_{trans}$. Tại thời điểm t = d_{trans}, bit đầu tiên của gói tin ở đâu?**
g. **Giả sử s = $2{,}5.10^8$, L = 120 bits và R = 56 kbps. Tính khoảng c ch m để $d_{prop} = d_{trans}$**

TL:

a. d_{prop} = m/s (s) – là độ trễ lan tỏa (thời gian gói tin đi hết khoảng cách m)
b. d_{trans} = L/R (s) – là độ trễ truyền (thời gian từ lúc bit đầu tiên của gói tin ra khỏi host đến khi bit cuối cùng của gói tin ra khỏi host).
c. $d_{A-B} = d_{prop} + d_{trans}$ = m/s + L/R (s)
d. Tại t = d_{trans}, bit cuối cùng của gói tin vừa rời khỏi host A.
e. Tại t = d_{trans}, bit đầu tiên của gói tin vẫn đang ở trên đường truyền, chưa đến được host B.
f. Tại t = d_{trans}, bit đầu tiên của gói tin đã đến host B.
g. $d_{prop} = d_{trans}$ m/s = L/R m = Ls/R = 523158 (m)

P6: Xét việc gửi giọng nói thời gian thực từ host A đến host B qua 1 mạng chuyển mạch gói (VoIP). Host A chuyển giọng nói thành luồng bit t n hiệu số 64 kbps. Sau đó host A nhóm các bit thành từng gói tin 56 bytes. Chỉ có 1 đường duy nhất từ A đến B tốc độ truyền là 2 Mbps và đỗ trễ lan tỏa là 10 ms. Ngay khi host A tạo được 1 gói tin nó gửi gói tin đó cho B. Ngay khi host B nhận được gói toàn bộ gói tin nó chuyển các bit của gói tin thành tín hiệu tượng tự. Tính thời gian kể từ khi bit đầu tiên được tạo (từ tín hiệu gốc ở host A) đến khi bit cuối cùng được giải mã (ở host B)?

Xét bit đầu tiên trong 1 gói tin. Trước khi bit này được truyền đi, tất cả các bit trong gói tin phải được tạo ra

Vì vậy cần $(56.8)/64.10^3$ (s) = 7 (ms)

Thời gian cần để truyền gói tin là: $(56.8)/(2.10^6)$ (s) = 224 (μs)

Độ trễ lam tỏa là 10ms
Độ trễ cho đến khi giải mã xong là: 7ms + 224 μs + 10ms = 17,224ms

P12: Giả sử N gói tin cùng đến 1 đường truyền và hiện tại không có gói tin nào được truyền đi hoặc phải xếp hàng. Mỗi gói tin có độ dài L và tốc độ đ ờng truyền là R. T nh độ trễ hàng đợi trung bình của N gói tin.

Độ trễ hàng đợi là 0 khi gói tin đầu tiên đã truyền đi, là L/R khi gói tin thứ 2 đã truyền đi và tổng quát, là (n – 1) L/R khi gói tin thứ n đã truyền đi.

Vì vậy, đỗ trễ trung bình của N gói tin là:

$$(L/R + 2L/R + ... + (N-1).L/R) / N$$
$$= L/(RN) * (1 + 2 + ... + (N-1))$$
$$= L/(RN) * N(N-1)/2$$
$$= LN(N-1)/(2RN)$$
$$= (N-1)L/(2R)$$

P24: Giả sử có 2 host A và B cách nhau 20.000 km, được nối trực tiếp với nhau bằng 1 đường tốc độ R = 2 Mbps. Tốc độ lan tỏa trên đường truyền là $2,5.10^8$ m/s.

a. **Tính tích của băng thông và độ trễ ($R.d_{prop}$).**
b. **Xét việc gửi 1 file độ dài 800.000 bits từ A đến B. Giả sử file được truyền liên tục. Số bit tối đa được truyền đi trên đ ờng truyền tại bất kì thời điểm nào là bao nhiêu?**
c. **Giải thích ý nghĩa tích $R.d_{prop}$.**
d. **Độ rộng của 1 bit (tính theo mét) trên đường truyền bằng bao nhiêu?**
e. **Viết công thức tổng quát tính độ rộng của 1 bit trên đường truyền. Cho biết tốc độ lan tỏa là s, tốc độ truy n là R và độ dài đường truy n là m.**

TL:

a. $R.d_{prop} = R. m/s = 2. 20.10^6 / (2,5.10^8) = 160.000$ bits.
b. Số bit tối đa được truyền đi trên đường truyền tại bất kì thời điểm = $R.d_{prop} = 160.000$ bits.
c. Tích số giữa băng thông(R) và độ trễ(d_{prop}) của đường truyền là số bit tối đa có thể ở trên đường truyền
d. Độ rộng của 1 bit = độ dài đường truyền / ($R.d_{prop}$) = $20.10^6 / 160.000 = 125$ m
e. Độ rộng của 1 bit = m / ($R.d_{prop}$) = s/R.

Chapter2: Application Layer -Review Question

Câu 1: Liệt kê 5 ứng dụng internet không độc quyền và giao thức mà chúng sử dụng tại tầng ứng dụng

- The Web - HTTP;
- File transfer (bittorrent) - FTP;
- Remote login - Telnet;
- Network News - NNTP;
- E-mail - SMTP;

Câu 2: Sự khác biệt giữa Network architecture và Application architecture?

- *Network architecture* là một hệ thống các tiến trình giao tiếp với nhau thông qua các tầng
- *Application architecture* được hiểu theo nghĩa khác, nó được thiết kế bởi nhà phát triển ứng dụng và nó điều khiển hoạt động của các application.

Câu 4: Cho 1 ứng dụng chia sẻ file P2P. B n có đồng ý với nhận xét "không có định nghĩa bên client and server trong 1 giao tiếp giữa chúng? T i sao?

Không, vì trong mọi giao dịch đều có 1 bên là client và 1 bên là server. Trong ứng dụng chia sẻ file P2P thì bên nhận được file là client còn bên gửi file là server.

Câu 6: Giả sử bạn muốn làm một giao dịch giữa 1 remote client và 1 server với tốc độ cao, bạn nên chọn UDP hay TCP?

Nên chọn UDP, vì sử dụng UDP không cần thiết lập đường chuyền nên nhanh hơn bạn chỉ mất 1 RTT đó là client gửi yêu cầu tới UDP socket. Còn nếu sử dụng TCP bạn mất tối thiểu 2 RTT: 1 cho việc thiết lập kết nối TCP, còn lại cho client gửi yêu cầu và server trả phản hồi.

Câu 15: Vì sao nói FTP gửi thông tin điều khiển là out-of-band?

FTP sử dụng 2 kết nối TCP song song. Một kết nối để kiểm soát thông tin (chẳng hạn như 1 yêu cầu chuyển giao 1 tập tin) và một kết nối khác dùng để chuyển giao các tập tin . Bởi vì các thông tin kiểm soát không được gửi qua cùng 1 kết nối với tập tin nên có thể nói rằng FTP gửi thông tin điều khiển "out-of-band" .

Câu 18 : Nêu sự khác biệt giữa download-and-delete mode and the download-and-keep mode in POP3?

- Với chế độ **download-and-delete** sau khi người dùng lấy tin nhắn từ một máy chủ POP, các tin nhắn sẽ bị xóa. Điều này đặt ra một vấn đề cần gải quyết đó là khi nhiều người cùng truy cập các tin nhắn từ nhiều máy khác nhau (văn phòng máy tính, máy tính gia đình,vv.)
- Trong chế độ **download-and-keep**, tin nhắn sẽ không bị xóa sau khi người sử dụng lấy các tin nhắn. Điều này cũng có thể là bất tiện là mỗi lần người sử dụng lấy các tin nhắn, tất cả các tin nhắn không bị xóa sẽ được chuyển vào máy tính (bao gồm các thông báo rất cũ).

Câu 19 : Có hay không một tổ chức Mail server and Web server nào có cùng 1 bí danh(alias) cho 1 hostname không? C i gì sẽ l u trữ host name of the mail server?

Có. The MX record được sử dụng để ánh xạ tên của mail server với địa chỉ IP.

Câu 22: Overlay network (mạng bao phủ) là gì? Nó có chứa router không? The edges(rìa) của nó là gì? Nó được tạo ra và duy trì như thế nào?

- Overlay network là một hệ thống gồm các nút tham gia vào chia sẻ tập tin và các liên kết giữa chúng.
- Nó không chứa router.
- The edges của nó là logical link (liên kết logic).
- Cách tạo ra nó là: khi 1 node mới muốn tham gia vào hệ thống nó cần biết địa chỉ IP của 1 hay nhiều node của hệ thống, sau đó nó sẽ gửi thông điệp cho các node này, các nút này nhận và xác nhận, và nó sẽ trở thành 1 phần của hệ thống.

Câu 25: Skype công nghệ P2P cho 2 giao thức quan trọng nào?

User location và NAT traversal.

Câu 26: 4 ứng dụng quan trọng phù hợp với kiến trúc P2P?

a) File Distribution
b) Instant Messaging
c) Video Streaming
d) Distributed Computing

Chapter2: Application Layer -Problem

Câu 1: Đúng/ Sai

a. Một người dùng yêu cầu từ 1 trang web 1 text và 3 hình ảnh, client sẽ gửi 1 tin nhắn yêu cầu và nhận 4 tin nhắn phản hồi? **- Sai: gửi 4 nhận 4**
b. 2 trang web khác biệt có thể gửi qua cùng 1 kết nối kiên trì. - **úng**
c. Với một kết nối không kiên trì giữa trình duyệt và máy chủ có thể cho 1 gói tin TCP thực hiện 2 request HTTP khác nhau. - **Sai**
d. Thông điệp yêu cầu của giao thức HTTP không thể rỗng.

Câu 3: Giả sử một HTTP client muốn lấy 1 tài liệu của trang web mà đã biết URL, địa chỉ IP chưa biết. Thì tầng ứng dụng và tầng giao vận cần giao thức nào

- Application layer protocols: DNS and HTTP
- Transport layer protocols: UDP for DNS; TCP for HTTP

Câu 7: Giả sử trong trình duyệt web bạn click vào một link chứa 1 trang web. Bạn cần lấy 1 địa chỉ IP của trang web đó chưa có trong cache. Bạn phải đi qua n DNS sau đó mới có được nó. Mỗi lần thăm 1 DNS bạn mất RTT thời gian lần lượt là RTT_1, RTT_2… RTT_n. Trong trang web đó có 1 đối tượng text. Bạn mất RTT_0 để đi từ host tới server chứa đối tượng đó. Tính thời gian từ khi click đến khi nhận được đối tượng.

Tổng thời gian để lấy được địa chỉ IP là: RTT_1+…+RTT_n;
Sau khi đã biết địa chỉ IP bạn mất RTT_0 để kết nối TCP và RTT_0 nữa để gửi thông điệp và nhận đối tượng.
Nên tổng thời gian cần thiết là: $2.RTT_0 + RTT_1 + … + RTT_n$;

Câu 8: Với đề bài câu 7, giả sử HTML có thêm 8 đối tượng nữa trên cùng server. Mất bao lâu thời gian với:

a. **Không kiên trì và không có kết nối song song**:
   Khi trang web thêm 8 đối tượng ta cần 8 lần thiết lập và 8 lần gửi thông điệp và nhận đối tượng nên mất thời gian là: $8.2.RTT_0 = 16\ RTT_0$
   Nên tổng thời gian là: $2RTT_0 + RTT_1 + \ldots + RTT_n + 16\ RTT_0$

b. **Không kiên trì có 5 kết nối song song**:
   1 lần kết nối gửi nhận được 5 đối tượng nên 8 đối tượng cần 2 lần kết nối nên mất thời gian là: $4RTT_0$ Nên tổng thời gian là: $2.RTT_0 + RTT_1 + \ldots + RTT_n + 4\ RTT_0$

c. **Kết nối kiên trì**
   Do đã khởi tạo kết nối để lấy đối tượng text nên 8 đối tượng này không cần khởi tạo kết nối nữa. Sử dụng kết nối kiên trì nên cần 1 RTT_0 để gửi yêu cầu và nhận đối tượng
   Nên tổng thời gian là: $2.RTT_0 + RTT_1 + \ldots + RTT_n + RTT_0 = 3\ RTT_0 + RTT_1 + \ldots + RTT_n$

Câu 14: SMTP kết thúc thân mail như thế nào? HTTP thì sao? HTTP có thể sử dụng phương thức giống SMTP để kết thúc mail được không?

- SMTP kết thúc thư bằng một dòng chỉ chứa dấu chấm.
- HTTP quản lí thư bằng trường độ dài trong header.
- HTTP không thể sử dụng phương thức giống SMTP được. vì HTTP message có thể để ở dạng mã nhị phân còn SMTP thì phải để ở dạng ASCII

Câu 17: Giả sử truy cập mail của bạn bằng POP3

a. Giả sử bạn định dạng cho thư của bạn to làm việc ở chế độ down xong xóa. Hoàn thành giao dịch bên dưới:
   C: dele 1
   C: retr 2
   S:(blah
   blah … S
   .....
   blah)
   S: .
   C: dele 2
   C: quit
   S: +OK POP3 server signing of

b. Chế độ down xong
   giữ
   C: retr 2
   S:blah blah …
   S:…….. blah
   S: .
   C: quit
   S: +OK POP3 server signing off

c. Ban đầu ở chế độ down xong giữ đọc thư 1 xong tắt đi sau 5 phút đọc thư 2. Đưa ra bản ghi cho trường hợp này?

C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: blah …..
S: ….blah
S: .
C: retr 2
S: blah blah …
S:…….blah
S: .
C: quit
S: +OK POP3 server signing off

Câu 20: Suppose you can access the caches in the local DNS servers of your depart ment. Can you propose a way to roughly determine the Web servers (outside your department) that are most popular among the users in your department?

Chúng ta có thể định kì lưu nhanh các bản của DNS caches trong những máy chủ DNS địa phương đó. Web server xuất hiện thường xuyên nhất trong DNS caches chính là server phổ biến nhất. Điều này là bởi nếu nhiều người dùng quan tâm đến 1 Web server thì DNS requests đến Web server đó được gửi thường xuyên hơn. Do đó, Web server đó sẽ đc xuất hiện trong DNS caches nhiều hơn local DNS cache, vì vậy thời gian truy vấn sẽ là 0 msec. Các trường hợp khác, thời gian truy vấn lớn hơn.

Câu 22: Consider distributing a file of F = 15 Gbits to N peers. The server has an upload rate of us = 30 Mbps, and each peer has a download rate of d i = 2 Mbps and an upload rate of u. For N = 10, 100, and 1,000 and u = 300 Kbps, 700 Kbps, and 2 Mbps, prepare a chart giving the minimum distribution time for each of the combinations of Nand u for both client-server distribution and P2P distribution.

Để tính toán thời gian điều phối tối thiểu (minimum distribution time) cho điều phối client-server, ta sử dụng công thức:

Dcs = max {NF/us, F/dmin}

Tương tự, để tính toán thời gian điều phối tối thiểu cho điều phối P2P, ta sử dụng công thức:

$$D_{P2P} = max\{F/u_s,\ F/d_{min},\ NF/(u_s + \sum_{i=1}^{N} u_i)\}$$

Ở đây, F = 15 Gbits = 15 * 1024 Mbits u_s = 30 Mbps

d_{min} = di = 2 Mbps

Note, 300Kbps = 300/1024 Mbps

Client Server

		N		
		10	**100**	**1000**
	300 Kbps	7680	51200	512000
u	**700 Kbps**	7680	51200	512000
	2 Mbps	7680	51200	512000

Peer to Peer

		N		
		10	**100**	**1000**
	300 Kbps	7680	25904	47559
u	**700 Kbps**	7680	15616	21525
	2 Mbps	7680	7680	7680

Câu 23. Consider distributing a file of F bits to N peers using a client-server architecture. Assume a fluid model where the server can simultaneously transmit to multiple peers, transmitting to each peer at different rates, as long as the combined rate does not exceed us

a. **Suppose that us/N <= dmin . Specify a distribution scheme that has a distribution time of NF/us.**
b. **Suppose that us/ N ~ dmin . Specify a distribution scheme that has a distribution time of F/dmin .**
c. **Conclude that the minimum distribution lime is in general given by max{NF/us, F/ dmin).**

TL

Xét một chương trình điều phối trong đó server gửi file tới từng client, song song, v một tốc độ us/N. Lưu ý tốc độ này nhỏ hơn tốc độ download của từng client, vì theo giả thiết us/N <= dmin. Do đó, mỗi client cũng có thể nhận được file với một tốc độ là us/N. Từ lúc mỗi client nhận tốc độ us/N, thời gian để mỗi client

nhận file hoàn chỉnh là F/(us/N) = NF/us. Tất cả các client đều nhận hoàn chỉnh file trong thời gian NF/us, nên tổng thời gian điều phối cũng là NF/us.

Câu 34

a. **Giả sử bạn mở FTPclient trước khi mở FTPserver. Có vấn đề gì xãy ra không?**

Khi đó client sẽ cố gắng kết nối với server trong khi server chưa mở=> kết nối lỗi.

b. **Bạn chạy UDP client trước khi chạy UDP server?**

Không có vấn đề gì. Vì client không cần khơi tạo tạo kết nối tới server.

c. **Sử dụng port khác nhau giữa client và server.**

Khi đó client sẽ cố gắng kết nối TCP với 1 tiến trình không đúng. Nên sẽ có lỗi

Chapter3: Transport Layer -Review Question

Câu 1. Xem xét một kết nối TCP từ host A đến B.Giả sử TCP segment đi từ A -> B có port nguồn là x và port đích là y.Vậy port nguồn và port đích của segment đi từ B -> A là j?

Source Port: y, Dest Port: x

Question 2

Giải thích tại sao nhà phát triển ứng dụng lại chọn UDP hơn TCP.

Solution :

Nhà phát triển ứng dụng chọn UDP vì họ không muốn ứng dụng của họ phải sử dụng cơ chế kiểm soát tắc nghẽn(cơ chế điều tiết tốc độ truyền dữ liệu của ứng dụng lúc bị tắc nghẽn).Cơ chế này có thể làm giảm tốc độ truyền,điều này có thể ảnh hưởng tới ứng dụng,nhất là những ứng dụng chat voice hay hội thảo trực tuyến).Những ứng dụng đó k cần độ tin cậy của dữ liệu được truyền,mà quan trọng là thời gian.

Question 3. Liệu có ứng dụng nào có cả khả năng truyền dữ liệu đáng tin cậy ngay cả khi nó chạy trên giao thức UDP?

Solution :

Có. Nhà phát triển ứng dụng có thể thêm giao thức truyền dữ liệu đáng tin cậy vào giao thức của lớp ứng dụng. Tất nhiên, nó sẽ đòi hỏi một lượng công việc đáng kể cho việc phát hiện lỗi.

Question 4 True or False

a. Giả sử host A truyền một tập tin lớn tới host B trên giao thức TCP,và host B không có dữ liệu j để gửi tới host A. Host B sẽ không gửi ACK tới host A vì nó không thể đóng gói ACK vào dữ liệu đc.(unlogical)
b. Kích thươc của cửa sổ bên nhận (TCP Rcv window) không bao h thay đổi trong suốt quá trình kết nối.
c. Giả sử host A truyền một tập tin lớn tới host B trên giao thức TCP,số byte dữ liệu NAK không thể vượt quá bộ nhớ đệm của phía nhận.
d. Giả sử host A truyền một tập tin lớn tới host B trên giao thức TCP,Số thứ tự của segment cho lần gửi này là m,thì số thứ tự cho segment tiếp theo phải là m+1. (Sai vì nếu quá trình gửi gói trc mà bị lỗi thì phải gửi lại gói m).
e. TCP segment có 1 trường trong tiêu đề cho cửa sổ bên nhận.
f. Giả sử thời gian RTT gần nhất trong kết nối TCP là 1 s.Vậy phải cài đặt thời gian timeout >=1 cho lần truyền này.(Sai vì cần sd nhiều RTT trong quá khứ chứ k sd RTT gần nhất).
g. Giả sử host A gửi 1 segment tới B vs STT là 38 và 4-bytes dữ liệu.Vậy ACK cho segment này phải này 42.(sai.ACK là 38)

Solution : a. **F** b. **F** c.**T** d. **F** e.**T** f. **F** g. **F**

Question 5

Giả sử A gửi 2 gói TCP liền nhau tới B.Segment thứ nhất có STT là 90, segment thứ 2 có STT là 110.

a. Segment đầu tiên có dung l ợng là bao nhiêu.

b. Giả sử segment đầu tiên bị mất, nhưng segment thứ 2 vẫn đến được B. Vậy stt của ACK được gửi từ B về A là j?

Solution :

a. 110-90 = 20 bytes.

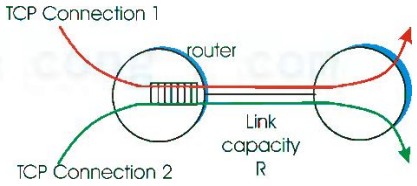
b. Vì gói 1 mất nên ACK vẫn là 90

Question 7: Hiện tại có 2 kết nối TCP với 1 nút cổ chai có tốc độ là R bps. Cả 2 kết nối đều có 1 file lớn cần gửi (đến cùng 1 h ớng qua nút cổ chai đó). Việc truyền dữ liệu bắt đầu ở cùng một thời điểm. Vậy tốc độ truyền mà TCP sẽ phân cho mỗi kết nối là bao nhiêu?

Chú ý : Qua tính công bằng của giao thức TCP ta biết được, nếu N kết nối cùng nhau chia sẻ một kênh truyền tắc nghẽn thì mỗi kết nối sẽ nhận được 1/N băng thông. (Tức là nhận được băng thông bằng nhau).

Solution **:**

Qua chú ý thì đáp án là : R/2



Question 8

Xem xét cơ chế kiểm soát tắc nghẽn trong TCP. Nếu xảy ra timeout ở bên gửi thì threshold sẽ được đặt bằng một nửa của giá trị threshold trước đó.

Sai vì nếu xảy ra mất gói tin, ngưỡng threshold sẽ được đặt bằng một nửa của Congwin (Congrestion window)

Note : Tốc độ truyền bị giới hạn bởi cửa sổ tắc nghẽn Congwin. Threshold là ngưỡng giữa 2 pha

Sự kiện	Trạng thái	Hành động bên gửi TCP	Giải thích
Nhận ACK cho dữ liệu không được ack trước đó	Slow Start (SS)	CongWin = CongWin + MSS, If (CongWin > Threshold) Đặt trạng thái thành "Congestion Avoidance"	Kết quả bởi gấp đôi CongWin mỗi RTT
Nhận ACK cho dữ liệu không được ack trước đó	Congestion Avoidance (CA)	CongWin = CongWin+MSS * (MSS/CongWin)	Tăng theo cấp số cộng, tăng CongWin lên 1 MSS mỗi RTT
Sự kiện mất gói phát hiện bởi 3 ACK lặp	SS hoặc CA	Threshold = CongWin/2, CongWin = Threshold, Đặt trạng thái thành "Congestion Avoidance"	Nhanh chóng phục hồi, thực hiện tăng cấp số nhân. CongWin sẽ không giảm dưới 1 MSS.
Timeout	SS hoặc CA	Threshold = CongWin/2, CongWin = 1 MSS, Đặt trạng thái thành "Slow Start"	Vào slow start
ACK lặp	SS hoặc CA	Tăng bộ đếm ACK lặp cho segment được ack	CongWin và Threshold không thay đổi

Question 9 (P1 trong sách)

Giả sử Network Layercung cấp dịch vụ sau. Network layer ở host nguồn chấp nhận gửi 1 segment có max size là 1200 bytes và địa chỉ đích từ transport layer.Network layer đảm bảo gửi thành công segment tới transport layer tại host đích.

Giả sử có nhiều tiến trình cùng chạy tại host đích.

a.Thiết kế một giao thức đơn giản cho transport layer để nhận được dữ liệu phù hợp cho mỗi tiến trình tại host đích.Giả sử HĐH ở host đích có thể cấp 4 bytes port number cho mỗi tiến trình đang chạy.

b.Điều chỉnh giao thức đó để nó có thể trả về địa chỉ ("return address") cho host đích. c.Trong giao thức của bạn liệu transport layer có phải làm j trong lõi của mạng máy tính?

Solution :

a. Gọi giao thức này là STP (Simple transfer protocol).Bên phía gửi,STP chấp nhận cho tiến trình gửi 1 chunk kích thươc k vượt quá 1196bytes dữ liệu ,một địa chỉ host đích và một cổng nhận.STP thêm 4 bytes vào trg header của mỗi chunk,4 bytes đó để lưu port number của tiến trình bên nhận.Và network layer sẽ chuyển gói tin tới transport layer của bên nhận.Sau đó STP sẽ giải nén segment và kiểm tra segment nhận đc có port number ứng với tiến trình nào rồi gửi segment đó tới đúng tiến trình cần nhận.
b. Bây h mỗi segment sẽ có 2 trg tiêu đề (header).Một trg cho port nguồn và 1 trg cho port đích.Bên phía gửi chấp nhận gửi 1 chunk kích thước k vượt quá 1192 bytes, một địa chỉ host đích ,một port nguồn và một port đích.
   STP tạo một segment bao gồm dữ liệu của application,port nguồn và port đích.Sau đó nó sẽ chuyển segment và địa chỉ host đích cho tầng network để gửi sang bên nhận…Sau khi bên đích nhận đc segment,STP sẽ chuyển tới ứng dụng dữ liệu cần nhận và cả port nguồn.
c. Nó k phải làm j.Nó chỉ nằm trên thiết bị đầu cuối.

Question 10.(P2)

Giả sử có 1 hành tinh có dịch vụ gửi thư.Mỗi gia đình có 1 địa chỉ(1 hòm thư riêng). Mỗi thành viên trong gia đình đều có một tên riêng. Dịch vụ gửi thư có thể gửi từ nhà này sang nhà khác. Nó yêu cầu phải cho bức thư và địa chỉ của nhà cần gửi vào trong phong bì. Mỗi gia đình đều có 1 người đại diện, người này sẽ nhận thư và phân phát thư cho các thành viên khác trong gia đình.

a. Sử dụng giao thức ở Question 9 để gửi thư. Mô tả cách ng đại diện nhận và phân phát thư. rong giao thức này,liệu mail server có phải mở phong bì để kiểm tra bức thư bên trong hay không?

Solution :

a) Người gửi sẽ đưa cho người đại diện của gia đình họ bức thư cùng với tên người nhận và địa chỉ của người đó.Sau đó người đại diện sẽ viết tên người nhận vào đầu bức thư, nhét nó vào phong bì. Bên ngoài phong bì ghi địa chỉ của ngôi nhà phía người nhận. Sau đó đưa cho bưu điện (chính là dịch vụ gửi thư của hành tinh đó) Bên kia, người đại diện nhận được

phong bì sẽ lấy bức thư ra, xem tên người nhận, và gửi cho đúng người đó. (Giả sử đó là thư tình mà ng đại diện là bố or mẹ thì e k bik "What will happen?" lol)

b) Mail server k cần mở phong bì, nó chỉ kiểm tra địa chỉ trên phong bì .

Question 11(P5) Tại sao bây giờ các ứng dụng chat voice hay video lại dùng TCP hơn UPD, trong khi TCP có cơ chế kiểm soát tắc nghẽn gây trễ đường truyền?

Sol: vì bây giờ firewall hay chặn các luồng truyền tải data của UDP.Trong khi TCP có thể vượt qua đc.

Question 12.(P7) Giả sử một tiến trình ở host C có một UDP socket với port là 6789. Giả sử cả 2 host A và B đều có thể gửi segment tới C với cùng một port đích là 6789. Liệu cả 2 segment đều cùng hướng đến cùng một socket đích ở host C hay không?Nếu có thì làm sao để host C có thể biết đc sự khác nhau giữa 2 segment đó (nó có nguồn gốc từ đâu)?

Solution: Có.Mỗi segment nhận đc dều đc thì hệ điều hành sẽ cung cấp 1 tiến trình với địa chỉ IP để xác định nguồn gốc của segment đó.

Question 13.(P8)

Giả sử Web server chạy trên host C tại port 80 và dùng kết nối kiên trì.Nó nhận được 2 yêu cầu từ host A và B.Liệu cả hai yêu cầu được gửi trên cùng một socket của C. Nếu 2 yêu cầu được trả lời qua 2 socket khác nhau thì liệu cả 2 socket đều có port là 80?

Solution : Trong kết nối kiên trì, Web server tạo ra từng kết nối socket riêng biệt . Mỗi socket kết nối được xác định với một bộ bốn : (địa chỉ IP nguồn, port nguồn , địa chỉ IP đích ,port đích) .Khi host C nhận được IP datagram/segment , nó kiểm tra 4 trg này để xác định socket mà nó phải truyền segment qua . Vì vậy , các yêu cầu từ A và B đc truyền thông qua các socket khác nhau .Cả 2 đều có port number là 80 , tuy nhiên , việc nhận diện các socket này dựa vào địa chỉ IP nguồn.Không giống như UDP, khi lớp transport gửi một segment TCP lên tiến trình ở tầng ứng dụng , nó không chỉ ra địa chỉ IP nguồn ,mà nó ngầm quy định bởi các trình định danh socket.

Question 14(P9) Trong giao thức rdt,tại sao chúng ta cần sử dụng STT(sequence number)?

Solution : STT đc sử dụng cho bên nhận,giúp phân biệt được khi nào dữ liệu nhận đc là bị trùng.

Question 15(P10) Trong rdt protocol, tại sao phải sử dụng bộ đếm thời gian(timer)?

Solution : Nếu như thời gian bên gửi chờ ACK (or NAK) vượt quá thời gian quy định của bộ đếm,thì bên gửi kết luận gói tin có thể bị mất.Do đó,nó sẽ truyền lại gói tin đó.

Question 16 (P11) Giả sử roundtrip delay (độ trễ phản hồi) giữa bên nhận và bên gửi là hằng số,và bên gửi bik độ trễ này.Vậy ta có cần timer cho rdt 3.0 nữa hay không?Giả sử gói tin có thể bị mất.

Solution : Timer vẫn cần thiết cho rdt 3.0 vì nếu bên gửi bik đc RTT(RTT là thời gian từ lúc gửi đến lúc nhận đc ACK) thì có 1 ưu điểm duy nhất là bên gửi bik đc packet hoặc ACK (NAK) có thể bị mất.Hãy đặt vào tình huống cụ thể,nếu ACK k mất mà chỉ đến trễ, lúc đó bên gửi vẫn cần timer để biết được gói tin đã mất hay chưa.

Chapter3: Transport Layer - Problem

1. **Giả sử tại cùng 1 thời điểm, client A và client B đều muốn khởi tạo 1 phiên Telnet tới server S. Cung cấp số hiệu cổng nguồn và đích có thể có cho:**
   a. Các segments gửi từ A đến S
   b. Các segments gửi từ B đến S
   c. Các segments gửi từ S đến A
   d. Các segments gửi từ S đến B
   e. Nếu A, B là 2 host khác nhau, thì số hiệu cổng nguồn trong segments từ A tới S có được giống như là từ B tới S không?
   f. Nếu chúng là cùng 1 host thì sao?

   Answer:

	source port numbers	**destination port numbers**
a) A→S	467	23
b) B→S	513	23
c) S→A	23	467
d) S→B	23	513
e) Yes		
f) No		

2. **UDP và TCP sử dụng bù 1 cho trường checksum.**

Giả sử ban đầu bạn có 3 byte 8 bit: 01010011, 01010100, 01110100. Bù 1 của tổng của các byte 8 bit là gì? Tại sao UDP lại lấy bù 1 của tổng? Tại sao không chỉ sử dụng tổng? Với việc sử dụng bù 1, làm thế nào để người nhận phát hiện lỗi? Có thể nào lỗi 1 bit sẽ đi không bị phát hiện? Lỗi 2 bit thì sao?

Answer:

0 1 0 1 0 0 1 1 + 0 1 0 1 0 1 0 0 ---------- 1 0 1 0 0 1 1 1 1 0 1 0 0 1 1 1 + 0 1 1 1 0 1 0 0 ---------- 0 0 0 1 1 1 0 0	=> Bù 1 là: 11100011 - Để phát hiện lỗi, người nhận sẽ tính tổng của 4 byte (3 byte gốc + 1 byte checksum) - Nếu tổng có chứa 1 số 0 => có lỗi - Tất cả các lỗi 1 bit đều bị phát hiện, nhưng 2 bit lỗi thì có thể sẽ không phát hiện được (ví dụ chữ số cuối cùng của byte đầu tiên chuyển thành 0 còn chữ số cuối cùng của byte thứ 2 chuyển thành 1)

3. **Giả sử người nhận UDP tính checksum cho gói tin UDP segment và nhận thấy rằng nó trùng khớp giá trị truyền tải trong trường checksum của gói tin. Người nhận có thể hoàn toàn chắc chắn rằng lỗi bit không xảy ra hay không? Giải thích.**

   Answer:

   Không, người nhận không thể hoàn toàn chắc chắn lỗi bit không đã xảy ra

   Bởi: nếu các bit tương ứng của 2 byte trong gói tin là 0 và 1 thì khi chúng đổi giá trị cho nhau (0 1, 1 0) thì tổng của chúng vẫn giống nhau bù 1 của tổng cũng sẽ giống nhau

4. **So sánh phía gửi và phái nhận của giao thức rdt2.2 và rdt3.0**

   Answer:

   Phía gửi dùng giao thức rdt3.0 khác phía gửi dùng giao thức rdt2.2, trong đó có thêm thời gian *timeouts*. Chúng ta đã thấy rằng việc thêm thời gian *timeouts* làm tăng khả năng trùng lặp các gói tin. Tuy nhiên, phía nhận dùng giao thức rdt.2.2 đã có thể xử lý các gói dữ liệu trùng lặp. (Gói tin trùng lặp ở phía nhận trong rdt 2.2 sẽ phát sinh nếu ACK bị lỗi, và người gửi sau đó truyền lại dữ liệu cũ). Do đó phía nhận trong giao thức rdt2.2 làm việc như là phía nhận trong giao thức rdt3.0

5. **Hãy xem xét một giao thức truyền dữ liệu đáng tin cậy mà chỉ sử dụng NAK.**

   Giả sử phía gửi sẽ gửi dữ liệu không thường xuyên. Giao thức chỉ dùng NAK có thích hợp hơn giao thức sử dụng ACKs hay không? Tại sao?

   Bây giờ giả sử phia gửi đã có rất nhiều dữ liệu để gửi và kết nối end-to-end có rất ít mất mát. Trong trường hợp này, giao thức chỉ dùng NAK có thích hợp hơn giao thức sử dụng ACKs không? Tại sao?

   Answer:

   Trong giao thức chỉ dùng NAK, mất gói x chỉ có thể phát hiện được bởi phía nhận khi phía nhận nhận được gói x-1 và ngay sau đó là x+1.

   Nếu phía gửi gửi không thường xuyên, và có 1 khoảng thời gian delay dài giữa 2 gói tin x và x+1 thì chỉ đến khi nào gói x+1 được gửi đi và đến được phía nhận thì phía nhận mới phát hiện được là gói x bị mất => sẽ mất 1 khoảng thời gian dài để khôi phục gói x theo giao thức chỉ dùng NAK

   Mặt khác, nếu dữ liệu được gửi thường xuyên thì việc khôi phục theo giao thức chỉ dùng NAK sẽ xảy ra rất nhanh chóng. Hơn nữa, nếu lỗi không thường xuyên xảy ra thì NAK chỉ thỉnh thoảng mới được gửi (khi cần thiết) còn ACK thì không cần gửi. Do đó làm giảm đáng kể trong thông tin phản hồi

   Giao thức chỉ dùng NAK tối ưu hơn giao thức sử dụng ACK

6. **Xem xét giao thức GBN với kích thước windows phía gửi là 3 và phạm vi của seq num là 1024. Giả sử tại thời điểm t, thứ tự tiếp theo trong gói tin mà phía nhận đang mong đợi là 1 sequence number của k. Cho rằng, môi trường không sắp xếp lại thứ tự của các msg. Trả lời các câu hỏi**

a. **Các tập seq num có thể có bên trong windows phía gửi tại thời điểm t là gì ?**
b. **Tất cả các giá trị có thể có của trường ACK trong tất cả các msg có thể hiện đang truyền quay trở lại phía người gửi tại thời điểm t là gì ?**

Answer:

a. **Ở đây chúng ta có 1 windows với kích thước N=3.**

Giả sử phía nhận đã nhận được gói tin k-1, và đã gửi ACK cho nó và tất cả các gói tin khác. Nếu tất cả ACK đều được nhận ở phía gửi thì sau đó windows phía người gửi sẽ là k, k+N-1].

Giả sử thứ 2 là không có ACK nào được nhận tại phía gửi, khi đó windows phía gửi sẽ chứa N gói tin và phải bao gồm gói k-1, nên windows phái gửi sẽ là k-N, k-1]

Với những lập luận trên thì windows phía gửi sẽ có kích thước là 3 và sẽ bắt đầu tại 1 nơi nào đó trong phạm vi k-N, k]

b. Nếu phía nhận đang chờ gói tin thứ k, và sau đó nó nhận được gói tin thứ k-1 và N-1 gói tin trước đó

=>..... P19

7. Trả lời đúng hoặc sai cho các câu hỏi sau đây và giải thích cho câu trả lời của bạn:

a. Với giao thức *selective repeat*, nó có thể cho phía gửi nhận được một ACK cho một gói tin nằm bên ngoài *windows* hiện tại của nó.

b. Với GBN, nó có thể cho phía gửi nhận được một ACK cho một gói tin nằm bên ngoài *windows* hiện tại của nó.

Answer:

a. Đúng.

Giả sử phía gửi có kích thước windows là 3 và gửi các gói tin 1,2,3

tại t_0. Tại t_1 (t_1>t_0) phía nhận gửi ACK 1,2,3.

Tại t_2 (t_2 > t_1), phía gửi *timeouts* và gửi lại gói tin 1,2,3.

Tại t_3, phía nhận nhận các gói tin trùng lăp, và gửi lại CK 1,2,3.

Tại t_4 phía gửi nhận ACK 1,2,3 mà phía nhận gửi tại thời điểm t_1, và chuyển windows lên 4,5,6.

Tại t_5 phía gửi nhận ACK 1,2,3 mà phía nhận gửi tại t_3. Khi này ACK ngoài khoảng của *windows*

b. Đúng. <giống ý a>

8. Xem xét việc chuyển một tập tin rất lớn của L byte từ Host A đến Host B. Giả sử 1 MSS = 536 byte.

a. Giá trị tối đa của L để TCP seq num không bị cạn kiệt là gì? Ttrường TCP seq num có 4 byte.

b. Với giá trị L có được trong câu a), phải mất bao lâu để truyền tải các tập tin. Giả sử rằng có tổng số 66 byte trường tiêu đề của tầng giao vận, mạng, liên kết được thêm vào segment trước khi gói dữ liệu được gửi qua một liên kết 155 Mbps. Bỏ qua kiểm soát lưu lượng và kiểm soát tắc nghẽn vì thế có thể gửi ra các segment liên tục.

Answer:

Có tất cả 2^{32} = 4294967296 sequence number

a. Sequence number không tăng lên 1 mỗi segment. Thay vào đó, nó gia tăng số lượng các byte dữ liệu đc gửi. Vì vậy kích cỡ của MSS là không liên quan – kích thước tập tin tối đa có thể được gửi từ A đến B chỉ đơn giản là số lượng byte biểu diễn bởi 2^{32} = 4.19 Gbytes

b. Số lượng segment là = 8012999 (lấy nguyên, làm tròn lên)

66byte tiêu đề đc thêm vào cho mỗi segment

=> tổng có = 66 * 8012999 = 528,857,934 bytes của tiêu đề

=> tổng độ lớn phải truyền đi là s = 2^{32} = 4,824*10^9 bytes

=> thời gian để truyền tải là (s * 8) / (155*1000) = 249 (s)

9. **Host A và B giao tiếp qua kết nối TCP và host B đã nhận được từ A tất cả các byte thông qua byte 126. Giả sử rằng host A sau đó gửi thêm 2 segment tới host B back-to-back. Segment đầu tiên và thứ 2 chứa tương ứng 70 và 50 byte dữ liệu. Trong segment đầu tiên sequence number là 127, source port là 302, destination port là 80. Host B gửi ACK ngay khi nó nhận được segment từ A**

   a. **Trong segment thứ 2 được gửi từ host A đến host B thì sequence number, source port, destination port là bao nhiêu?**

   b. **Nếu segment đầu tiên đến trước segment thứ 2, trong acknowledgement của segment đầu tiên đến thì acknowledgement number, source port, destination port là bao nhiêu?**

   c. **Nếu segment thứ 2 đến trước segment đầu tiên, trong acknowledgement của segment đầu tiên đến thi acknowledgement number là bao nhiêu?**

   d. **Giả sử 2 segment được gửi bởi A đến theo thứ tự tại B, acknowledgement đầu tiên là bị mất, acknowledgement thứ 2 đến sau khoảng thời gian time out. Vẽ sơ đồ thời gian hiển thị các segment và acknowledgements đã gửi.**

Answer:

a. Trong segment thứ 2:

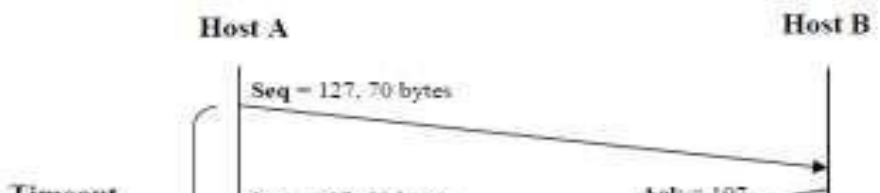
- sequence number là 197
- source port number là 302
- destination port number là 80

b. Nếu segment đầu tiên đến trước segment thứ 2:

- acknowledgement number là 197
- source port number là 80
- destination port number là 302.

c. Nếu segment thứ 2 đến trước segment đầu tiên:

- acknowledgement number là 127 – chỉ ra rằng nó vẫn còn đợi các bytes 127 trở đi d.



10. **Host A và B được kết nối trực tiếp với liên kết 100 Mbps.Có một kết nối TCP giữa hai host, và host A gửi đến host B một tập tin có kích thước lớn qua kết nối này. Host A có thể gửi dữ liệu ứng dụng của nó vào socket TCP của nó với tốc độ cao là 120 Mbps nhưng Host B có thể đọc nhận được bộ đệm với tốc độ tối đa là 60 Mbps. Mô tả ảnh hưởng của kiểm soát lưu lượng TCP.**

Answer:

Host A có thể gửi với tốc độ 100Mbps. Tuy nhiên host A gửi vào bộ đệm nhanh hơn so với host B có thể đọc. Bộ đệm đầy lên với tốc độ 40Mbps. Khi bộ đệm đầy, host B báo hiệu tới host A dừng gửi dữ liệu bằng cách thiết lập **RcvWindow = 0.** Host A sẽ dừng việc gửi dữ liệu cho đến tận khi nhận được segment với **RcvWindow > 0.** Do đó host A sẽ liên tục dừng lại và bắt đầu gửi. Trung bình trong thời gian dài, host A gửi dữ liệu đến host B như là 1 phần của kết nối này và không lớn hơn 60Mbps

11. **Hãy xem xét các thủ tục TCP để ước lượng RTT. Giả sử α = 0,1.**
SampleRTT1 là SampleRTT gần đây nhất. SampleRTT2 là RTT gần đây nhất tiếp theo, và như vậy.
    a. **Đối với một kết nối TCP nhất định, giả sử có bốn acknowledgments đã được trả lại với SampleRTTs tương ứng: SampleRTT4, SampleRTT3, SampleRTT2, và SampleRTT1, Thể hiện EstimatedRTT với bốn SampleRTTs.**
    b. **Khái quát công thức cho n SampleRTTs**
    c. **Đối với công thức trong phần (b) áp dụng**

cho n->∞.

Answer:

a. Kí hiệu $EstimatedRTT^{(n)}$ là ước lượng sau n sample

$EstimatedRTT^{(1)} = SampleRTT_1$

$EstimatedRTT^{(2)} = \alpha .SampleRTT_1 + (1-\alpha) .SampleRTT_2$

$EstimatedRTT^{(3)} = \alpha .SampleRTT_1 + (1-\alpha)[\ \alpha .SampleRTT_2 + (1-\alpha).SampleRTT_3\ ]$

$= \alpha .SampleRTT_1 + (1-\alpha).\alpha .SampleRTT_2 + (1-\alpha)^2.SampleRTT_3$

$EstimatedRTT^{(4)} = \alpha .SampleRTT_1 + (1-\alpha).\alpha .SampleRTT_2 + (1-\alpha)^2.\alpha .SampleRTT_3 + (1-\alpha)^3.SampleRTT_3$

b. $EstimatedRTT^{(n)} = \alpha .\ \sum_{j=1}^{n-1}(1-\alpha)^j.SampleRTT_j + (1-\alpha)^n SampleRTT_n$

c.

$$EstimatedRTT^{(\infty)} = \frac{x}{1-x}\sum_{j=1}^{\infty}(1-x)^j\, SampleRTT_j$$

$$= \frac{1}{9}\sum_{j=1}^{\infty}.9^j\, SampleRTT_j$$

12. Host A gửi một tập tin rất lớn đến host B qua một kết nối TCP. Qua kết nối này là không bao giờ có bất k mất mát gói dữ liệu và không có timeout. Ký hiệu tốc độ truyền dẫn của liên kết kết nối host A đến Internet là R bps. Giả sử rằng quá trình host A có khả năng gửi dữ liệu vào TCP socket với tốc độ S bps, trong đó S = 10 * R. Hơn nữa giả sử rằng bộ đệm nhận là đủ lớn để chứa toàn bộ tập tin, và bộ đệm gửi chỉ có thể giữ một phần trăm của tập tin. Điều gì sẽ ngăn chặn quá trình trong máy chủ liên tục truyền dữ liệu đến socket TCP của nó với tốc độ S bps ?

Answer:

Trong vấn đề này, không có nguy hiểm trong tràn bộ nhớ phía nhận nhận từ khi vùng đệm nhận phía nhận có thể giữ toàn bộ tập tin. Hơn nữa, vì không có mất mát và acknowledgements được gửi trả trước khi timeout, TCP kiểm soát nghẽn không điều tiết phía gửi. Tuy nhiên, quá trình trên host A sẽ không liên tục chuyển dữ liệu cho socket vì vùng đệm gửi sẽ nhanh chóng được lấp đầy. Một khi vùng đệm gửi trở thành đầy, quá trình sẽ chuyển dữ liệu ở tỉ lệ trung bình hoặc R << S.

13. Xem xét việc gửi một tập tin lớn từ một host khác qua kết nối TCP mà không có mất mát.

a. Giả sử TCP sử dụng AIMD kiểm soát tắc nghẽn mà không có khởi đầu chậm chạp. Giả sử tăng cwnd lên 1 MSS mỗi khi tất cả ACKs nhận được và giả định RTT là hằng số, mất bao lâu để cwnd tăng từ 5 MSS lên 11 MSS (giả sử không có sự kiện mất)?

b. Tính thông lượng trung bình cho kết nối này trong khoảng thời gian = 6 RTT?

Answer:

a. Phải mất:

1 RTT để tăng cwnd lên 6 MSS;
2 RTT để tăng cwnd lên 7 MSS;
3 RTT để tăng cwnd lên 8 MSS;
4 RTT để tăng cwnd lên 9 MSS;
5 RTT để tăng cwnd lên 10 MSS;
6 RTT để tăng cwnd lên 11 MSS

b. Trong RTT đầu tiên: 5 MSS đã được gửi; Trong RTT thứ 2: 6 MSS đã được gửi;
Trong RTT thứ 3: 7 MSS đã được gửi; Trong RTT thứ 4: 8 MSS đã được gửi;
Trong RTT thứ 5: 9 MSS đã được gửi; Trong RTT thứ 6: 10 MSS đã được gửi.

Vì thế trong khoảng 6 RTT, 5+6+7+8+9+10 = 45 MSS đã được gửi (và acknowledged).
Do đó, chúng ta có thể nói rằng thông lượng trung bình trong khoảng 6RTT là:

---

14. **Trong giao thức rdt3.0, gói tin ACK từ phía nhận đến phía gửi không phải đánh số, mặc dù chúng có trường trong ACK mà chứa stt của packet mà chúng đã biết. Tại sao các packet ACK không đòi hỏi đánh số (sequence number)**

Answer

Chúng ta thấy rằng, phía gửi cần seq num để phía nhận có thể biết packet đó có trùng với 1 packet đã nhận từ trước không. Đối với ACKs, thì bên gửi không cần thông tin đó (VD: stt của ACK) để biết 1 ACK trùng lặp. Một ACK trùng lặp là r ràng đối với phía nhận của rdt3.0, kể từ khi nó nhận được ACK gốc thì nó chuyển sang trạng thái mới. ACK trùng lặp không phải là cái ACK người gửi cần nên nó bị bỏ qua bởi rdt3.0

15. **Suppose an application uses rdt3.0 as its transport layer protocol. As the stop-and-wait protocol has very low channel utilization (shown in the cross-country example), the designers of this application let the receiver keep sending back a number (more than two) of alternating ACK 0 and ACK 1 even if the corresponding data have not arrived at the receiver. Would this application design increase the channel utilization? Why? Are there any potential problems with this approach? Explain**

Answer

Yes. This actually causes the sender to send a number of pipelined data into the channel.

Yes. Here is one potential problem. If data segments are lost in the channel, then the sender of rdt 3.0 won't re-send those segments, unless there are some additional mechanism in the application to recover from loss