1. **DANH SÁCH THÀNH VIÊN**

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **MSSV** | **Thực hiện** |
| Đinh Bùi Huy Phương | 21520090 | Bài P.23, tổng hợp file. |
| Đinh Bạch Kiều Phương | 21520406 | Bài P.24, P.27 |
| Nguyễn Viết Dũng | 21520747 | Bài P.27, P.28 |
| Nguyễn Thị Thanh Mai | 21521112 | Bài P.28, P.31 |
| Lê Đoàn Trà My | 21521149 | Bài P.24, tổng hợp file. |
| Nguyễn Phương Trinh | 21521581 | Bài P.23, P.31 |

1. **BÀI TẬP**

**Bài P.23:**

Consider the GBN and SR protocols. Suppose the sequence number space is of size k. What is the largest allowable sender window that will avoid the occurrence of problems such as that in Figure 3.27 for each of these protocols?

|  |  |
| --- | --- |
| **Tóm tắt**  Xét giao thức GBN và SR. Giả sử khoảng seq num là k. Xác định kích thước cửa sổ lớn nhất để tránh vấn đề xảy ra trong hình 3.27 | **Bài làm**  Vấn đề trong hình 3.27 xảy ra do kích thước cửa sổ quá lớn, khi chồng cửa sổ bên gửi và bên nhận lên nhau, xuất hiện tràn số và quay trở lại giá trị bé nhất. Khi đó, số seq gói tin bên gửi gửi lại có thể sẽ trùng với số seq của gói tin bên nhận đang đợi, dẫn tới trường hợp không phân biệt được gói tin đó là gói mới hay cũ.  Để tránh trường hợp đó, cửa sổ bên gửi và bên nhận khi chồng lên nhau phải chứa không quá k phần tử.  Gọi kích thước cửa sổ là w, gọi gói tin mà bên nhận đang đợi là n  => Cửa sổ bên nhận hiện tại là [n, n + w - 1]    Như vậy, bên nhận đã gửi ack cho gói tin có số seq bé hơn (n - 1)  Giả sử ack của gói tin có seq = n - w bị mất, như vậy gói tin có seq = n - 1 là gói tin có seq lớn nhất nằm trong khoảng gửi lại được  => Cửa sổ bên gửi hiện tại là [n - w; n - 1]    Như vậy, khi chồng 2 cửa sổ lại ta được một khoảng có kích thước là: n + w - 1 - (n - w) + 1 = 2w.    Để thỏa lập luận ở trên, **2w <= k <=> w <= k / 2**  → Vậy **kích thước tối đa** của cửa sổ để tránh vấn đề như hình 3.27 là **k / 2.** |

**Bài P24:**

Answer true or false to the following questions and briefly justify your answer:

a. With the SR protocol, it is possible for the sender to receive an ACK for a packet that falls outside of its current window.

b. With GBN, it is possible for the sender to receive an ACK for a packet that falls outside of its current window.

c. The alternating-bit protocol is the same as the SR protocol with a sender and receiver window size of 1.

d.The alternating-bit protocol is the same as the GBN protocol with a sender and receiver window size of 1.

|  |  |
| --- | --- |
| **Tóm tắt** | **Bài làm** |
| Các câu dưới đúng hay sai? Giải thích ngắn gọn:  **a,** Với giao thức SR (selective repeat), người gửi có thể nhận được ACK cho 1 gói nằm ngoài cửa sổ hiện tại của nó.  **b,** Với GBN (go-back-end), người gửi có thể nhận được ACK cho 1 gói nằm ngoài cửa sổ hiện tại của nó.  **c,** Giao thức alternating – bit giống như giao thức SR với kích thước cửa sổ người gửi và người nhận bằng 1.  **d,** Giao thức alternating – bit giống như giao thức GBN với kích thước cửa sổ người gửi và người nhận bằng 1. | **a,** Đúng.  Trong trường hợp ACK 1, ACK 2 được gửi lại cho bên gửi trễ hơn so với thời gian time out, khi đó:  + Bên gửi: gửi lại các gói tin 1, 2.  + Bên nhận sẽ nhận được các bản sao và gửi lại ACK 1, ACK 2.  Bên gửi sẽ nhận được ACK 1, ACK 2 được gửi lúc đầu và nâng cửa sổ lên 3, 4.  Gói ACK 1, ACK 2 được gửi sau sẽ được bên gửi nhận thêm lần nữa, lúc này các ACK nằm ngoài cửa sổ hiện tại của bên gửi  **b,** Đúng. Vì GBN sử dụng ACK tích lũy.  Ví dụ, trường hợp ACK1 đến sau ACK 2. Khi bên gửi nhận đươc ACK 2 thì nó sẽ nghĩ là gói tin 1 và 2 đã được nhận vì vậy sẽ loại bỏ gói tin 1 và 2 ra cửa sổ hiện tại.  Nếu gói ACK 1 đến, thì ACK 1 đã nằm ngoài cửa sổ hiện tại.  **c,** Đúng.  Trong giao thức alternating – bit có số thứ tự của gói thay đổi giữa 0 và 1.  + Khi bên gửi gửi một gói tin 0, bên nhận sẽ gửi lại ack 0 nếu gói tin đó ko bị lỗi và là gói tin 0.  + Khi nhận được gói ACK 0, thì bên gửi sẽ nâng cửa sổ lên số thứ tự 1.  Điều này tương tự với giao thức SR với kích thước cửa sổ là 1.  **d,** Đúng  Trong giao thức alternating – bit có số thứ tự của gói thay đổi giữa 0 và 1.  + Khi bên gửi gửi một gói tin 0, bên nhận sẽ gửi lại ack 0 nếu gói tin đó ko bị lỗi và là gói tin 0.  + Khi nhận được gói ACK 0, thì bên gửi sẽ nâng cửa sổ lên số thứ tự 1.  Tương tự với giao thức GBN với kích thước cửa sổ là 1. |

**Bài P,27:**

Host A and B are communicating over a TCP connection, and Host B has already received from A all bytes up through byte 126. Suppose Host A then sends two segments to Host B back-to-back. The first and second segments contain 80 and 40 bytes of data, respectively. In the first segment, the sequence number is 127, the source port number is 302, and the destination port number is 80. Host B sends an acknowledgment whenever it receives a segment from Host A.

a. In the second segment sent from Host A to B, what are the sequence number, source port number, and destination port number?

b. If the first segment arrives before the second segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number, the source port number, and the destination port number?

c. If the second segment arrives before the first segment, in the acknowledgment of the first arriving segment, what is the acknowledgment number?

d. Suppose the two segments sent by A arrive in order at B. The first acknowledgment is lost and the second acknowledgment arrives after the first timeout interval. Draw a timing diagram, showing these segments and all other segments and acknowledgments sent. (Assume there is no additional packet loss.) For each segment in your figure, provide the sequence number and the number of bytes of data; for each acknowledgment that you add, provide the acknowledgment number.

|  |  |
| --- | --- |
| **Tóm tắt**  Máy chủ A và B giao tiếp với nhau qua kết nối TCP. B nhận từ A 126 bytes.  Giả sử, A gửi B 2 segment liên tục.  Segment 1 chứa 80 bytes data, segment 2 chứa 40 bytes data.  Trong segment 1, có:  + SeqNum = 127  + Port nguồn = 302  + Port đích = 80  B gửi ACK mỗi khi nhận được segment từ A  **a,** Trong segment 2 được gửi từ A đến B,  SeqNum =?, port nguồn =?, port đích =?  **b,** Giả sử segment1 đến trước segment2, trong ACK đầu tiên gửi về A có:  ACKNum =?, port nguồn =?, port đích =?  **c,** Giả sử segment2 đến trước segment1, trong ACK đầu tiên gửi về A có:  ACKnum =?, port nguồn =?, port đích =? **d,** Giả sử 2 segment tới B đúng thứ tự. ACK cho segment thứ nhất bị mất và ACK cho segment thứ hai tới sau khi bị time-out.  Vẽ sơ đồ thời gian, biểu diễn các gói tin(segment, ack, ...).  Với mỗi segment trong sơ đồ, chỉ rõ SeqNum, số bytes. Với mỗi ACK, AckNum. | **Bài làm**  **a,** Trong segment 2 được gửi từ A đến B, có:  + SeqNum = 127 + 80 = 207.  + Port nguồn = 302.  + Port đích = 80.  **b,** Nếu segment1 đến trước segment2, trong ACK đầu tiên gửi về A có:  + ACKnum = 207.  + Port nguồn = 80.  + Port đích = 302.  **c,** Nếu segment2 đến trước segment1, trong ACK đầu tiên gửi về A có:  + ACKnum = 127.  + Port nguồn = 80.  + Port đích = 302.  (Vì segment có SeqNum = 127 chưa tới nên ACKnum =127 để đợi segment đó)  **d,** |

**Bài P,28:**

Host A and B are directly connected with a 100 Mbps link. There is one TCP connection between the two hosts, and Host A is sending to Host B an enormous file over this connection. Host A can send its application data into its TCP socket at a rate as high as 120 Mbps but Host B can read out of its TCP receive buffer at a maximum rate of 50 Mbps. Describe the effect of TCP flow control.

|  |  |
| --- | --- |
| **Tóm tắt**  A với B kết nối trực tiếp với liên kết 100Mbps.  Giữa A và B có 1 kết nối TCP, A gửi B một tệp rất lớn trên kết nối này.  A gửi dữ diệu tối đa: 120 Mbps  B nhận dữ liệu tối đa: 50 Mbps  Mô tả tác dụng điều khiển luồng TCP? | **Bài làm**  Vì dung lượng liên kết chỉ là 100 Mbps, do đó tốc độ gửi của Máy chủ A ≤ 100 Mbps.  Bộ đệm nhận sẽ bắt đầu đầy, do tốc độ Máy chủ A gửi dữ liệu vào đó nhanh hơn tốc độ Máy chủ B đọc dữ liệu khỏi bộ đệm.  Khi bộ đệm đầy, Máy chủ B sẽ gửi thông báo rvcwnd = 0 đến Máy chủ A để ngừng gửi dữ liệu cho đến khi Máy chủ B có thể đọc dữ liệu khỏi bộ đệm.  Máy chủ B sau đó sẽ gửi một segment TCP với nội dung rvc > 0 đến Máy chủ A, thông báo cho nó tiếp tục gửi dữ liệu.  Bộ đệm sẽ đầy lại và quá trình này sẽ lặp lại cho đến khi tất cả dữ liệu được gửi từ Máy chủ A đến Máy chủ B.  Như vậy, tốc độ gửi dữ liệu trung bình từ Máy chủ A đến Máy chủ B ≤ 50 Mbps |

**Bài P,31**

Suppose that the five measured SampleRTT values (see Section 3.5.3) are 106 ms, 120 ms, 140 ms, 90 ms, and 115 ms. Compute the EstimatedRTT after each of these SampleRTT values is obtained, using a value of α = 0.125 and assuming that the value of EstimatedRTT was 100 ms just before the first of these five samples were obtained. Compute also the DevRTT after each sample is obtained, assuming a value of β = 0.25 and assuming the value of DevRTT was 5 ms just before the first of these five samples was obtained. Last, compute the TCP TimeoutInterval after each of these samples is obtained.

|  |  |
| --- | --- |
| **Tóm tắt** | **Bài làm** *(Đáp án được làm tròn 4 chữ số sau dấu phẩy)* |
| Có 5 giá trị SampleRTT:  + = 106 ms  + = 120 ms  + = 140 ms  + = 90 ms  + = 115 ms  \* Tính EstimatedRTT sau mỗi lần thu SampleRTT, dùng giá trị α = 0,125.  Giả sử, EstimatedRTT = 100ms ngay trước khi thu 5 SampleRTT trên.  \* Tính DevRTT sau mỗi lần thu SampleRTT, dùng giá trị  β = 0.25.  Giả sử, DevRTT = 5ms ngay trước khi thu 5 SampleRTT trên.  \* Tính TCP TimeoutInterval sau mỗi lần thu SampleRTT | ***\* Các công thức:***  + estimatedRTT = (1-α).estimatedRTT + α.sampleRTT  + devRTT = (1-β).devRTT + β. |sampleRTT-estimatedRTT|  + timeoutInterval = estimatedRTT + 4.devRTT  *- Sau khi thu được = 106 ms:*  + **estimatedRTT** = (1-0,125).100 + 0,125.106 = **100,75ms**  + **devRTT** = (1-0,25).5 + 0,25. |106-100,75| = **5,0625ms**  + **timeoutIntervla** = 100,75 + 4.5,0625 = **121ms**  *- Sau khi thu được = 120 ms:*  + **estimatedRTT** = (1-0,125).100,75 + 0,125.120 = **103,1563ms**  + **devRTT** = (1-0,25).5,0625 + 0,25. |120-103,1563| = **8,0078ms**  + **timeoutIntervla** = 103,1563 + 4.8,0078 = **135,1875ms**  *- Sau khi thu được = 140 ms:*  + **estimatedRTT**= (1-0,125).103,1563 + 0,125.140 = **107,7618ms**  + **devRTT** = (1-0,25).8,0078 + 0,25. |140-107,7618| = **14,0654ms**  + **timeoutIntervla** = 107,7618 + 4.14,0654 = **164,0234ms**  *- Sau khi thu được = 90 ms:*  + **estimatedRTT** = (1-0,125).107,7618 + 0,125.90 = **105,5416ms**  + **devRTT** = (1-0,25).14,0654 + 0,25. |90-105,5416| = **14,4345ms**  + **timeoutIntervla** = 105,5416+ 4.14,4345 = **163,2796ms**  *- Sau khi thu được = 115 ms:*  + **estimatedRTT**= (1-0,125).105,5416 + 0,125.115 = **106,7239ms**  + **devRTT**= (1-0,25).14,4345 + 0,25. |115-106,7239| = **12,8949ms**  + **timeoutIntervla** = 106,7239 + 4.12,8949 = **158,3035ms** |