Computer Networks - CS132/EECS148 - Spring 2013

Instructor: Karim El Defrawy

Assignment 3 - Solutions

Deadline : May 9th – 9:30pm (hard and soft copies required)

Problem 1 (Problem 6, Chapter 3 - 3 points) - Consider our motivation for correcting protocol rdt2.1. Show that the receiver, shown in figure 3.57 of your book, when operating with the sender, shown in figure 3.11 of your book, can lead the sender and receiver to enter into a deadlock state, where each is waiting for an event that will never occur.

Suppose the sender is in state “Wait for call 1 from above” and the receiver (the receiver shown in the homework problem) is in state “Wait for 1 from below.” The sender sends a packet with sequence number 1, and transitions to “Wait for ACK or NAK 1,” waiting for an ACK or NAK.
Suppose now the receiver receives the packet with sequence number 1 correctly, sends an ACK, and transitions to state “Wait for 0 from below,” waiting for a data packet with sequence number 0. However, the ACK is corrupted. When the rdt2.1 sender gets the corrupted ACK, it resends the packet with sequence number 1. However, the receiver is waiting for a packet with sequence number 0 and (as shown in the home work problem) always sends a NAK when it doesn't get a packet with sequence number 0. Hence the sender will always be sending a packet with sequence number 1, and the receiver will always be NAKing that packet. Neither will progress forward from that state.

Problem 2 (Problem 7, Chapter 3 - 3 points) - In protocol rdt3.0, the ACK packets flowing from the receiver to the sender do not have sequence numbers (although they do have an ACK field that contains the sequence number of the packet they are acknowledging). Why is it that our ACK packets do not require sequence numbers?

To best answer this question, consider why we needed sequence numbers in the first place. We saw that the sender needs sequence numbers so that the receiver can tell if a data packet is a duplicate of an already received data packet. In the case of ACKs, the sender does not need this info (i.e., a sequence number on an ACK) to tell detect a duplicate ACK. A duplicate ACK is obvious to the rdt3.0 receiver, since when it has received the original ACK it transitioned to the next state. The duplicate ACK is not the ACK that the sender needs and hence is ignored by the rdt3.0 sender.

Problem 3 (Problem 24, Chapter 3 - 4 points) - 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.

a) True. Suppose the sender has a window size of 3 and sends packets 1, 2, 3 at t 0 . At t1 (t1 > t 0) the receiver ACKS 1, 2, 3. At t 2 (t 2 > t1) the sender times out and resends 1, 2,

3. At t 3 the receiver receives the duplicates and re-acknowledges 1, 2, 3. At t 4 the sender receives the ACKs that the receiver sent at t1 and advances its window to 4, 5, 6. At t 5 the sender receives the ACKs 1, 2, 3 the receiver sent at t 2 . These ACKs are outside its window.

b) True. By essentially the same scenario as in (a).

Problem 4 (Problem 26, Chapter 3 - 6 points) - Consider transferring an enormous file of L bytes from Host A to Host B. Assume an MSS of 536 bytes.
a - What is the maximum value of L such that TCP sequence numbers are not exhausted? Recall that the TCP sequence number field has four bytes.
b - For the L you obtained in (a), find how long it takes to transmit the file. Assume that a total of 66 bytes of transport, network and data-link header are added to each segment before the resulting packet is sent out over a 155 Mbps link. Ignore flow control and congestion control so A can pump out the segments back to back and continuously.

a)
There are 2^32 = 4,294,967,296 possible sequence numbers.
The sequence number does not increment by one with each segment. Rather, it increments by the number of bytes of data sent. So the size of the MSS is irrelevant -- the maximum size file that can be sent from A to B is simply the number of bytes representable by 2^32 ≈ 4.19 Gbytes.

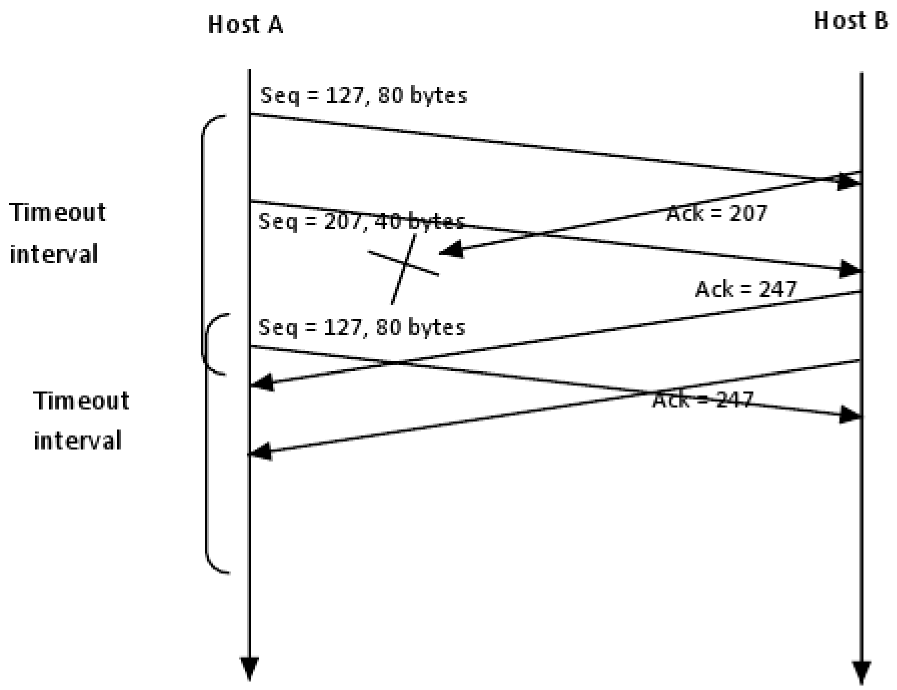
b)
ceil(2^32 / 536) = 8,012,999

The number of segments is. 66 bytes of header get added to each segment giving a total of 528,857,934 bytes of header. The total number of bytes transmitted is 2^32 + 528,857,934 = 4.824 × 10^9 bytes.
Thus it would take 249 seconds to transmit the file over a 155~Mbps link.

Problem 5 (Problem 27, Chapter 3 - 6 points) - Hosts 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 segments, what is the ACK number, the source port number. and the destination port number?
c - If the second segment arrives before the first segment, in the ACK of the first arriving segment, what is the ACK number?

a) In the second segment from Host A to B, the sequence number is 207, source port number is 302 and destination port number is 80.
b) If the first segment arrives before the second, in the acknowledgement of the first arriving segment, the acknowledgement number is 207, the source port number is 80 and the destination port number is 302.
c) If the second segment arrives before the first segment, in the acknowledgement of the first arriving segment, the acknowledgement number is 127, indicating that it is still waiting for bytes 127 and onwards.
d)



Problem 6 (Problem 31, Chapter 3 - 4 points) - Suppose that the five measured SampleRTT values are 106ms, 120ms, 140ms, 90ms, and 115ms. Compute the EstimatedRTT after each of these SampleRTT values is obtained, using a value of alpha = 0.125 and assuming that the value of EstimatedRTT was 100ms just before the first of these 5 samples were obtained. Compute also the DevRTT after each sample is obtained, assuming a value of beta = 0.25 and assuming the value of DevRTT was 5ms just before the first of these five samples was obtained. Last, Compute the TCP TimeoutInterval after each of these samples is obtained.

$\text{EstimatedR TT} = x\text{SampleRTT} + (1 - x)\ \text{EstimatedR TT}$

$\text{DevRTT} = y\ \text{SampleRTT} - \text{EstimatedRTT} + (1 - y)\ \text{DevRTT}$

$\text{TimeoutInterval} = \text{EstimatedRTT} + 4 * \text{DevRTT}$

After obtaining first sampleRTT is $\text{EstimatedRTT} = 0.125 * 106 + 0.875 * 100 = 100.75\text{ms}$

.

$\text{DevRTT} = 0.25 * 106 - 100.75 + 0.75 * 5 = 5.06\text{ms}$

$\text{TimeoutInterval} = 100.75 + 4 * 5.06 = 120.99\text{ms}$

.

After obtaining second sampleRTT = 120ms:

$\text{EstimatedRTT} = 0.125 * 120 + 0.875 * 100.75 = 103.15\text{ms}$

.

$\text{DevRTT} = 0.25 * 120 - 103.15 + 0.75 * 5.06 = 8\text{ms}$

$\text{TimeoutInterval} = 103.15 + 4 * 8 = 135.15\text{ms}$

.

After obtaining Third sampleRTT = 140ms:

$\text{EstimatedRTT} = 0.125 * 140 + 0.875 * 103.15 = 107.76\text{ms}$

.

$\text{DevRTT} = 0.25 * 140 - 107.76 + 0.75 * 8 = 14.06\text{ms}$

$\text{TimeoutInterval} = 107.76 + 4 * 14.06 = 164\text{ms}$

.

After obtaining fourth sampleRTT = 90ms:

$\text{EstimatedRTT} = 0.125 * 90 + 0.875 * 107.76 = 105.54\text{ms}$

.

DevRTT = 0.25 ∗ 90 − 105.54 + 0.75 *14.06 = 14.42ms

TimeoutInterval = 105.54 + 4 *14.42 = 163.22ms

.

After obtaining fifth sampleRTT = 115ms:

EstimatedRTT = 0.125 * 115 + 0.875 * 105.54 = 106.71ms

.

DevRTT = 0.25 ∗ 115 − 106.71 + 0.75 *14.42 = 12.88ms

TimeoutInterval = 106.71 + 4 *12.88 = 158.23ms

Problem 7 (Problem 40, Chapter 3 - 4 points) - Consider Figure 3.58 of your book, assuming TCP Reno is the protocol experiencing the behaviour shown in the figure, answer the following question. In all cases, you should provide a short discussion justifying your answer.
a - Identify the intervals of time when TCP slow start is operating.
b - Identify the intervals of time when TCP congestion avoidance is operating.

a) TCP slowstart is operating in the intervals [1,6] and [23,26]
b) TCP congestion avoidance is operating in the intervals [6,16] and [17,22]

Problem 8 (Problem 44, Chapter 3 - 4 points) - Consider sending a large file from a host to another over a TCP connection that has no loss.
a - Suppose TCP uses AIMD for its congestion control w/o slow start. Assuming cwnd increases by 1MSS every time a batch of ACKs is received and assuming approximately constant round-trip times, how long does it take for cwnd increase from 6MSS to 12MSS (Assuming no loss events)?
b - What is the average throughout (in terms of MSS and RTT) for this connection up through time = 6RTT ?

a) It takes 1 RTT to increase CongWin to 6 MSS; 2 RTTs to increase to 7 MSS; 3 RTTs to increase to 8 MSS; 4 RTTs to increase to 9 MSS; 5 RTTs to increase to 10 MSS; 6 RTTs to increase to 11 MSS; and 7 RTTs to increase to 12MSS.

b) In the first RTT 5 MSS was sent; in the second RTT 6 MSS was sent; in the third RTT 7 MSS was sent; in the fourth RTT 8 MSS was sent; in the fifth RTT, 9 MSS was sent; and in the sixth RTT, 10 MSS was sent. Thus, up to time 6 RTT, 5+6+7+8+9+10 = 45 MSS were sent (and acknowledged). Thus, we can say that the average throughput up to time 6 RTT was (45 MSS)/(6 RTT) = 7.5 MSS/RTT.

Problem 9 (Problem 50, Chapter 3 - 4 points) - Consider a simplified TCP's AIMD algorithm where the congestion window size is measured in number of segments not in bytes. In additive increase, the congestion window size increases by one segment in each RTT. In multiplicative decrease, the congestion window size decreases by half (if the result is not an integer, round down to the nearest integer). Suppose that two TCP connections, C1 and C2, share a single congested link of speed c segments per second. Assume that both C1 and C2 are in the congestion avoidance phase. Connection C1's RTT is 50ms and connection C2's RTT is 100ms. Assume that when the data rate in the link exceeds the link's speed, all TCP connection experience data segment loss.
a - If both C1 and C2 at time t0 have a congestion window of 10 segments, what are their congestion window sizes after 1000ms?
b - In the long run, will these two connections get the same share of the bandwidth of the congested link? Explain.

a) The key difference between C1 and C2 is that C1's RTT is only half of that of C2. Thus C1 adjusts its window size after 50 msec, but C2 adjusts its window size after 100 msec.
Assume that whenever a loss event happens, C1 receives it after 50msec and C2 receives it after 100msec. We further have the following simplified model of TCP. After each RTT, a connection determines if it should increase window size or not. For C1, we compute the average total sending rate in the link in the previous 50 msec. If that rate exceeds the link capacity, then we assume that C1 detects loss and reduces its window size. But for C2, we compute the average total sending rate in the link in the previous 100msec. If that rate exceeds the link capacity, then we assume that C2 detects loss and reduces its window size.
Note that it is possible that the average sending rate in last 50msec is higher than the link capacity, but the average sending rate in last 100msec is smaller than or equal to the link capacity, then in this case, we assume that C1 will experience loss event but C2 will not.

The following table describes the evolution of window sizes and sending rates based on the above assumptions.

	C1	C2		
Time (msec)	Window Size (num. of segments sent in next 50msec)	Average data sending rate (segments per second, =Window/0.05)	Window Size(num. of segments sent in next 100msec)	Average data sending rate (segments per second, =Window/0.1)

0	10	200 (in [0-50]msec]	10	100 (in [0-50]msec)
50	5 (decreases window size as the avg. total sending rate to the link in **last 50msec** is 300= 200+100)	100 (in [50-100]msec]		100 (in [50-100]msec)
100	2 (decreases window size as the avg. total sending rate to the link in **last 50msec** is 200= 100+100)	40	5 (decreases window size as the avg. total sending rate to the link in **last 100msec** is 250= (200+100)/2 + (100+100)/2)	50
150	1 (decreases window size as the avg. total sending rate to the link in last 50msec is 90= (40+50)	20		50
200	1 (no further decrease, as window size is already 1)	20	2 (decreases window size as the avg. total sending rate to the link in **last 100msec** is 80=	20

			(40+20)/2 + (50+50)/2)	
250	1 (no further decrease, as window size is already 1)	20		20
300	1 (no further decrease, as window size is already 1)	20	1 (decreases window size as the avg. total sending rate to the link in **last 100msec** is 40= (20+20)/2 + (20+20)/2)	10
350	2	40		10
400	1	20	1	10
450	2	40		10
500	1 (decreases window size as the avg. total sending rate to the link in last 50msec is 50= (40+10)	20	1	10
550	2	40		10
600	1	20	1	10

650	2	40		10
700	1	20	1	10
750	2	40		10
800	1	20	1	10
850	2	40		10
900	1	20	1	10
950	2	40		10
1000	1	20	1	10

Based on the above table, we find that after 1000 msec, C1's and C2's window sizes are 1 segment each.

b) No. In the long run, C1's bandwidth share is roughly twice as that of C2's, because C1 has shorter RTT, only half of that of C2, so C1 can adjust its window size twice as fast as C2. If we look at the above table, we can see a cycle every 200msec, e.g. from 850msec to 1000msec, inclusive. Within a cycle, the sending rate of C1 is (40+20+40+20) = 120, which is thrice as large as the sending of C2 given by (10+10+10+10) = 40.

Problem 10 (Problem 4, Chapter 4 - 4 points) - Consider the network shown in the figure of Problem 4 of chapter 4 in your book.
a - Suppose that this network is a datagram network. Show the forwarding table in router A, such that all traffic destined to Host H3 is forwarded through interface 3.
b - Suppose that this network is a datagram network. Can you write down forwarding table in routed A, such that all traffic from H1 destined to Host H3 is forwarded through interface 3, while all traffic from H2 destined to Host H3 is forwarded through interface 4? (Hint : this is a trick question)
a) Data destined to host H3 is forwarded through interface 3

Destination Address
H3

b) No, because forwarding rule is only based on destination address.

Link Interface 3

Problem 11 (Problem 13, Chapter 4 - 3 points) - Consider a router that interconnects three subnets: Subnet 1, Subnet 2 and Subnet 3. Suppose all of the interfaces in each of these three subnets are required to have the prefix 223.1.17/24. Also suppose that Subnet 1 is required to support at least 60 interfaces, Subnet 2 is to support at least 90 interfaces, and Subnet 3 is to support at least 12 interfaces. Provide three network addresses (of the form A.B.C.D/X) that satisfy these constraints.

223.1.17.0/26
223.1.17.128/25
223.1.17.192/28

Problem 12 (Problem 17, Chapter 4 - 4 points) - Consider the topology shown in Figure 4.17 in your book. Denote the three subnets with hosts (starting clockwise at 12:00) as Networks A, B, and C. Denote the subnets w/o hosts as Networks D,E and F.
a - Assign network addresses to each of these 6 subnets, with the following constraints. All addresses must be allocated from 214.97.254/23; subnet A should have enough addresses to support 250 interfaces; subnet B should have enough addresses to support 120 addresses; and subnet C should have enough addresses to support 120 addresses. Of course, subnets D, E, and F should each be able to support two interfaces. For each subnet the assignment should take the form of A.B.C.D/X or A.B.C.D/X - E.F.G.H/Y.
b - Using your answer to part (a), provide the following tables (using longest prefix matching). For each of the three routers.

From 214.97.254/23, possible assignments are

a)
Subnet A: 214.97.255/24 (256 addresses)
Subnet B: 214.97.254.0/25 - 214.97.254.0/29 (128-8 = 120 addresses)
Subnet C: 214.97.254.128/25 (128 addresses)

Subnet D: 214.97.254.0/31 (2 addresses)
Subnet E: 214.97.254.2/31 (2 addresses)
Subnet F: 214.97.254.4/30 (4 addresses)

b)
To simplify the solution, assume that no datagrams have router interfaces as ultimate destinations. Also, label D, E, F for the upper-right, bottom, and upper-left interior subnets, respectively.

Router 1

Longest Prefix Match	**Outgoing Interface**
11010110 01100001 11111111	Subnet A
11010110 01100001 11111110 0000000	Subnet D
11010110 01100001 11111110 000001	Subnet F

Router 2

Longest Prefix Match	**Outgoing Interface**
11010110 01100001 11111111 0000000	Subnet D
11010110 01100001 11111110 0	Subnet B
11010110 01100001 11111110 0000001	Subnet E

Router 3

Longest Prefix Match	**Outgoing Interface**
11010110 01100001 11111111 000001	Subnet F
11010110 01100001 11111110 0000001	Subnet E
11010110 01100001 11111110 1	Subnet C

Problem 13 (Problem 19, Chapter 4 - 3 points) - Consider sending a 2400-byte datagram into a link that has an MTU of 700 bytes. Suppose the original datagram is stamped with the identification number 422. How many fragments are generated? what are the values in the various fields in the IP datagram(s) generated related to fragmentation?

The maximum size of data field in each fragment = 680 (because there are 20 bytes IP header).

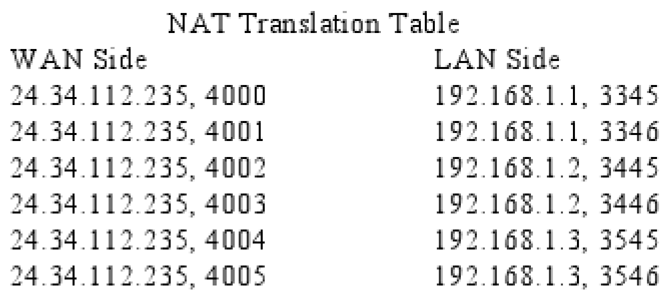
Thus the number of required fragments [2400 − 20 / 680] = 4

Each fragment will have Identification number 422. Each fragment except the last one will be of size 700 bytes (including IP header). The last datagram will be of size 360 bytes (including IP header). The offsets of the 4 fragments will be 0, 85, 170, 255. Each of the first 3 fragments will have flag=1; the last fragment will have flag=0.

Problem 14 (Problem 21, Chapter 4 - 4 points) - Consider the network setup in Figure 4.22 in your book, suppose that the ISP instead assigns the router the address 24.34.112.235 and that the network address of the home network is 192.168.1/24.
a - Assign addresses to all interfaces in the home network.
b - Suppose each host has two ongoing TCP connections, all to port 80 at host 128.119.40.86. Provide the six corresponding entries in the NAT translation table.

a) Home addresses: 192.168.1.1, 192.168.1.2, 192.168.1.3 with the router interface being 192.168.1.4

b)



NAT Translation Table

WAN Side	LAN Side
24.34.112.235, 4000	192.168.1.1, 3345
24.34.112.235, 4001	192.168.1.1, 3346
24.34.112.235, 4002	192.168.1.2, 3445
24.34.112.235, 4003	192.168.1.2, 3446
24.34.112.235, 4004	192.168.1.3, 3545
24.34.112.235, 4005	192.168.1.3, 3546

Problem 15 (Problem 22, Chapter 4 - 4 points) - Suppose you are interested in detecting the number of hosts behind a NAT. You observe that the IP layer stamps an identification number sequentially on each IP packet. The identification of the first IP packet generated by a host is a random number, and the identification numbers of the subsequent IP packets are sequentially assigned. Assume all IP packets generated by hosts behind the NAT are sent to the outside world.
a - Based on this observation and assuming you can sniff all packets sent by the NAT to the outside, can you outline a simple technique that detects the number of unique hosts behind a NAT? justify your answer.
b - If the identification numbers are not sequentially assigned but randomly assigned, would your technique work? justify your answer.

a) Since all IP packets are sent outside, so we can use a packet sniffer to record all IP packets generated by the hosts behind a NAT. As each host generates a sequence of IP packets with sequential numbers and a distinct (very likely, as they are randomly chosen from a large space) initial identification number (ID), we can group IP packets with consecutive IDs into a cluster. The number of clusters is the number of hosts behind the NAT.

For more practical algorithms, see the following papers.

“A Technique for Counting NATted Hosts”, by Steven M. Bellovin, appeared in IMW’02, Nov. 6-8, 2002, Marseille, France. “Exploiting the IPID field to infer network path and end-system characteristics.” Weifeng Chen, Yong Huang, Bruno F. Ribeiro, Kyoungwon Suh, Honggang Zhang, Edmundo de Souza e Silva, Jim Kurose, and Don Towsley. PAM'05 Workshop, March 31 - April 01, 2005. Boston, MA, USA.

b) However, if those identification numbers are not sequentially assigned but randomly assigned, the technique suggested in part (a) won’t work, as there won’t be clusters in sniffed

Extra credit - Include the answers to Wireshark lab problems of Chapter 3 and 4 for extra 20 points.