Assignment 3

R1. Suppose the network layer provides the following service. The network layer in the source host accepts a segment of maximum size 1,200 bytes and a destination host address from the transport layer. The network layer then guarantees to deliver the segment to the transport layer at the destination host. Suppose many network application processes can be running at the destination host.

a. Design the simplest possible transport-layer protocol that will get application data to the desired process at the destination host. Assume the operating system in the destination host has assigned a 4-byte port number to each running application process.

At the sender side, protocol STP accepts from the sending process a chunk under 1196 bytes, a destination address, and destination port number. A four-byte header is added to each chunk and the port number of the destination process is added as well. STP gives the network layer the destination address and the resulting segment. It is then delivered to STP at the destination host and the port number is examined. The data is extracted and the passed to the process identified.

b. Modify this protocol so that it provides a “return address” to the destination process.

The segment would now have two header fields, one for source port and the other destination port. The sender side STP does the same, but now has to process the source port throughout the process discussed in part a.

c. In your protocols, does the transport layer “have to do anything” in the core of the computer network?

It does not have anything to do with the core since it lives in the end systems.

R2. Consider a planet where everyone belongs to a family of six, every family lives in its own house, each house has a unique address, and each person in a given house has a unique name. Suppose this planet has a mail service that delivers letters from source house to destination house. The mail service requires that (1) the letter be in an envelope, and that (2) the address of the destination house (and nothing more) be clearly written on the envelope. Suppose each family has a delegate family member who collects and distributes letters for the other family members. The letters do not necessarily provide any indication of the recipients of the letters.

a. Using the solution to Problem R1 above as inspiration, describe a protocol that the delegates can use to deliver letters from a sending family member to a receiving family member.

To send a letter, the family member gives the delegate the letter, the address of the destination, and the name of the recipient. The delegate puts the letter in an envelope and writes the address of the destination on the envelope. On the receiving side, the person receives the letter and looks at the name written on the letter and then gives it to the person.

b. In your protocol, does the mail service ever have to open the envelope and examine the letter in order to provide its service?

No, as it only examines the address on the envelope.

R3. Consider a TCP connection between Host A and Host B. Suppose that the TCP segments traveling from Host A to Host B have source port number x and destination port number y. What are the source and destination port numbers for the segments traveling from Host B to Host A?

Source y, destination x.

R4. Describe why an application developer might choose to run an application over UDP rather than TCP.

A developer would want to avoid TCP’s congestion control, which throttles the sending rate. Therefore, it is better to use UDP for things like video conference apps.

R5. Why is it that voice and video traffic is often sent over TCP rather than UDP in today’s Internet? (Hint: The answer we are looking for has nothing to do with TCP’s congestion-control mechanism.)

Most firewalls block UDP traffic.

R6. Is it possible for an application to enjoy reliable data transfer even when the application runs over UDP? If so, how?

Yes, by putting reliable data transfer into the application layer protocol, which requires heavy work and debugging.

R7. Suppose a process in Host C has a UDP socket with port number 6789. Suppose both Host A and Host B each send a UDP segment to Host C with destination port number 6789. Will both of these segments be directed to the same socket at Host C? If so, how will the process at Host C know that these two segments originated from two different hosts?

Yes, as both segments will be directed to the same socket. For this, it will check the IP addresses to see where each segment came from.

R8. Suppose that a Web server runs in Host C on port 80. Suppose this Web server uses persistent connections, and is currently receiving requests from two different Hosts, A and B. Are all of the requests being sent through the same socket at Host C? If they are being passed through different sockets, do both of the sockets have port 80? Discuss and explain.

For each connection, the server creates a separate socket. Each socket is identified with the source-IP address, source port number, destination IP address, and destination port number. This allows host C to determine which socket it should pass the payload to. This allows the requests to pass through different sockets. The identifier for both of these sockets has 80 for the destination port; however, the identifiers for these sockets have different values for source IP addresses.

R9. In our rdt protocols, why did we need to introduce sequence numbers?

A receiver uses it to find out whether a packet has new data or not.

R10. In our rdt protocols, why did we need to introduce timers?

To handle losses in the channel by checking if a packet is received after a duration of time.

R11. Suppose that the roundtrip delay between sender and receiver is constant and known to the sender. Would a timer still be necessary in protocol rdt 3.0, assuming that packets can be lost? Explain.

Yes, as the roundtrip-delay only lets the sender know if the packet has been lost. You would still need a timer at the sender to detect the loss.

R12. Visit the Go-Back-N Java applet at the companion Web site.

a. Have the source send five packets, and then pause the animation before any of the five packets reach the destination. Then kill the first packet and resume the animation. Describe what happens.

Packet loss caused a time out after all five packets were retransmitted.

b. Repeat the experiment, but now let the first packet reach the destination and kill the first acknowledgment. Describe again what happens.

Loss of ACK did not trigger a retransmission.

c. Finally, try sending six packets. What happens?

Send window is fixed to 5.

R13. Repeat R12, but now with the Selective Repeat Java applet. How are Selective Repeat and Go-Back-N different?

a. When the packet was lost, the four other packets were buffered. After timeout, the sender retransmits the lost packet.

b. Duplicate ACK sent by receiver.

c. Send window size is 5.

Selective sends a duplicate ACK while Go-Back-N used cumulative acknowledgment.

R14. True or false?

a. Host A is sending Host B a large file over a TCP connection. Assume Host B has no data to send Host A. Host B will not send acknowledgments to Host A because Host B cannot piggyback the acknowledgments on data.

False

b. The size of the TCP rwnd never changes throughout the duration of the connection.

False

c. Suppose Host A is sending Host B a large file over a TCP connection. The number of unacknowledged bytes that A sends cannot exceed the size of the receive buffer.

True

d. Suppose Host A is sending a large file to Host B over a TCP connection. If the sequence number for a segment of this connection is m, then the sequence number for the subsequent segment will necessarily be m + 1.

False

e. The TCP segment has a field in its header for rwnd.

True

f. Suppose that the last SampleRTT in a TCP connection is equal to 1 sec. The current value of TimeoutInterval for the connection will necessarily be >= 1 sec.

False

g. Suppose Host A sends one segment with sequence number 38 and 4 bytes of data over a TCP connection to Host B. In this same segment the acknowledgment number is necessarily 42.

False

R15. Suppose Host A sends two TCP segments back to back to Host B over a TCP connection. The first segment has sequence number 90; the second has sequence number 110.

a. How much data is in the first segment?

20 bytes

b. Suppose that the first segment is lost but the second segment arrives at B. In the acknowledgment that Host B sends to Host A, what will be the acknowledgment number?

90

R16. Consider the Telnet example discussed in Section 3.5. A few seconds after the user types the letter ‘C,’ the user types the letter ‘R.’ After typing the letter ‘R,’ how many segments are sent, and what is put in the sequence number and acknowledgment fields of the segments?

3 segments

1 Sequence 43; ACK 80

2 Sequence 80; ACK 44

3 Sequence 44; ACK 81

R17. Suppose two TCP connections are present over some bottleneck link of rate R bps. Both connections have a huge file to send (in the same direction over the bottleneck link). The transmissions of the files start at the same time. What transmission rate would TCP like to give to each of the connections?

R/2

R18. True or false? Consider congestion control in TCP. When the timer expires at the sender, the value of ssthresh is set to one half of its previous value.

False, since it is set to half of the current value.

R19. In the discussion of TCP splitting in the sidebar in Section 3.7, it was claimed that the response time with TCP splitting is approximately 4⋅RTTFE+RTTBE+processing time. Justify this claim.

In the TCP packet exchange diagram between a client and server with a proxy, it can be seen that the response time is approximately 4⋅RTTFE+RTTBE+processing time.