1. An IP datagram using the Strict source routing option has to be fragmented. Do you think the option is copied into each fragment, or is it sufficient to just put it in the first fragment? Explain your answer.
    a. Since the information is needed to route every fragment, the option must appear in every fragment.
2. Many companies have a policy of having two (or more) routers connecting the company to the Internet to provide some redundancy in case one of them goes down. Is this policy still possible with NAT? Explain your answer.
    a. After NAT is installed, it is crucial that all the packets pertaining to a single connection pass in and out of the company via the same router, since that is where the mapping is kept. If each router has its own IP address and all traffic belonging to a given connection can be sent to the same router, the mapping can be done correctly and multihoming with NAT can be made to work.
3. You have just explained the ARP protocol to a friend. When you are all done, he says: ''I've got it. ARP provides a service to the network layer, so it is part of the data link layer.'' What do you say to him?
    a. You say that ARP does not provide a service to the network layer, it is part of the network layer and helps provide a service to the transport layer. The issue of IP addressing does not occur in the data link layer. Data link layer protocols are like protocols 1 through 6 in Chap. 3, HDLC, PPP, etc. They move bits from one end of a line to the other.
4. The Protocol field used in the IPv4 header is not present in the fixed IPv6 header. Why not?
    a. The Protocol field tells the destination host which protocol handler to give the IP packet to. Intermediate routers do not need this information, so it is not needed in the main header. Actually, it is there, but disguised. The Next header field of the last (extension) header is used for this purpose.
5. When the IPv6 protocol is introduced, does the ARP protocol have to be changed? If so, are the changes conceptual or technical?
    a. Conceptually, there are no changes. Technically, the IP addresses requested are now bigger, so bigger fields are needed.
6. Imagine that a two-way handshake rather than a three-way handshake were used to set up connections. In other words, the third message was not required. Are deadlocks now possible? Give an example or show that none exist.
    a. Deadlocks are possible. For example, a packet arrives at A out of the blue, and A acknowledges it. The acknowledgement gets lost, but A is now open while B knows nothing at all about what has happened. Now the same thing happens to B, and both are open, but expecting different sequence numbers. Timeouts have to be introduced to avoid the deadlocks.
7. Why does UDP exist? Would it not have been enough to just let user processes send raw IP packets?
    a. No. IP packets contain IP addresses, which specify a destination machine. Once such a packet arrived, how would the network handler know which process to give it to? UDP

packets contain a destination port. This information is essential so they can be delivered to the correct process.

8. Consider a simple application-level protocol built on top of UDP that allows a client to retrieve a file from a remote server residing at a well-known address. The client first sends a request with file name, and the server responds with a sequence of data packets containing different parts of the requested file. To ensure reliability and sequenced delivery, client and server use a stop-and-wait protocol. Ignoring the obvious performance issue, do you see a problem with this protocol? Think carefully about the possibility of processes crashing.
    a. It is possible that a client may get the wrong file. Suppose client A sends a request for file f1 and then crashes. Another client B then uses the same protocol to request another file f2. Suppose client B, running on the same machine as A (with same IP address), binds its UDP socket to the same port that A was using earlier. Furthermore, suppose B's request is lost. When the server's reply (to A's request) arrives, client B will receive it and assume that it is a reply its own request.
9. Both UDP and TCP use port numbers to identify the destination entity when delivering a message. Give two reasons for why these protocols invented a new abstract ID (port numbers), instead of using process IDs, which already existed when these protocols were designed.
    a. Here are three reasons. First, process IDs are OS-specific. Using process IDs would have made these protocols OS-dependent. Second, a single process may establish multiple channels of communications. A single process ID (per process) as the destination identifier cannot be used to distinguish between these channels. Third, having processes listen on well-known ports is easy, but well-known process IDs are impossible.
10. What is the total size of the minimum TCP MTU, including TCP and IP overhead but not including data link layer overhead?
    a. The default segment is 536 bytes. TCP adds 20 bytes and so does IP, making the default 576 bytes in total.
11. Datagram fragmentation and reassembly are handled by IP and are invisible to TCP. Does this mean that TCP does not have to worry about data arriving in the wrong order?
    a. Even though each datagram arrives intact, it is possible that datagrams arrive in the wrong order, so TCP has to be prepared to reassemble the parts of a message properly.
12. Give a potential disadvantage when Nagle's algorithm is used on a badly-congested network.
    a. Even though the user is typing at a uniform speed, the characters will be echoed in bursts. The user may hit several keys with nothing appearing on the screen, and then all of a sudden, the screen catches up with the typing. People may find this annoying.
13. What is the fastest line speed at which a host can blast out 1500-byte TCP payloads with a 120-sec maximum packet lifetime without having the sequence numbers wrap around? Take TCP, IP, and Ethernet overhead into consideration. Assume that Ethernet frames may be sent continuously.
    a. The goal is to send 2^{32} bytes in 120 sec or 35,791,394 payload bytes/sec. This is 23,860 1500-byte frames/sec. The TCP overhead is 20 bytes. The IP overhead is 20 bytes. The Ethernet overhead is 26 bytes. This means that for 1500 bytes of payload, 1566 bytes must be sent. If we are to send 23,860 frames of 1566 bytes every second, we need a

line of 299 Mbps. With anything faster than this we run the risk of two different TCP segments having the same sequence number at the same time.

14. DNS uses UDP instead of TCP. If a DNS packet is lost, there is no automatic recovery. Does this cause a problem, and if so, how is it solved?
    a. DNS is idempotent. Operations can be repeated without harm. When a process makes a DNS request, it starts a timer. If the timer expires, it just makes the request again. No harm is done.
15. In addition to being subject to loss, UDP packets have a maximum length, potentially as low as 576 bytes. What happens when a DNS name to be looked up exceeds this length? Can it be sent in two packets?
    a. The problem does not occur. DNS names must be shorter than 256 bytes. The standard requires this. Thus, all DNS names fit in a single minimum-length packet.
16. Can a machine with a single DNS name have multiple IP addresses? How could this occur?
    a. Yes. Remember that an IP address consists of a network number and a host number. If a machine has two Ethernet cards, it can be on two separate networks, and if so, it needs two IP addresses.
17. Can a computer have two DNS names that fall in different top-level domains? If so, give a plausible example. If not, explain why not.
    a. It is possible. www.large-bank.com and www.large-bank.ny.us could have the same IP address. Thus, an entry under com and under one of the country domains is certainly possible (and common).
18. The number of companies with a Web site has grown explosively in recent years. As a result, thousands of companies are registered in the com domain, causing a heavy load on the top-level server for this domain. Suggest a way to alleviate this problem without changing the naming scheme (i.e., without introducing new top-level domain names). It is permitted that your solution requires changes to the client code.
    a. There are obviously many approaches. One is to turn the top-level server into a server farm. Another is to have 26 separate servers, one for names beginning with a, one for b, and so on. For some period of time (say, 3 years) after introducing the new servers, the old one could continue to operate to give people a chance to adapt their software.