Answers to the quiz #2 of introduction to networking

Christian Rinderknecht

June 18, 2025

1 Review questions

1. *Why is it that* FTP *sends control information out-of-band?*

   FTP uses two parallel TCP connections: one for sending control information (such as a request to transfer a file or change the current remote directory) and another connection for actually transferring the file. Because the control information is not sent over the same connection that the file is sent over, FTP sends control information out of band.

2. *Why* FTP *is said to be a stateful protocol?*

   Throughout a session, the FTP server must maintain a state about the user:

   - the control connection must be associated with a specific user account;
   - the server must keep track of a given user's current directory on its file system.

3. *Are the objectives of flow control and congestion control the same?*

   No, their objectives are different. Flow control is about a host not overflowing the other host's receiving buffer, whilst congestion control is about not overflowing the router's queues.

4. *Give a very short description of how the connection-oriented service of the Internet provides reliable transport.*

   This service is TCP. It ensures reliable transport by means of acknowledgement and retransmission. If the sender does not receive the acknowledgement from the destination about a given packet, then this packet is transmitted again.

5. *Describe briefly Web caching at proxies and browsers. Why are they useful?*

Proxies are special nodes in the internet which, among other features, allows several users (hosts), usually on the same Local Area Network (LAN), to share a same cache. This way, for instance, if several users retrieve the same Web objects, only the first one will be actually downloaded from the HTTP server, the following being retrieved from the proxy cache (assuming they are up-to-date or no `GET` request is issued).

Browsers also have caches, implemented as files on the local file system. If the user requests the same object twice of more, the object is taken from the cache, not from the network.

Web caching is useful because it reduces the traffic on the internet.

6. *Compare the common and different features of* SMTP *and* HTTP*.*

   The common feature is that both protocols can use persistent TCP connection (actually, this is mandatory in case of SMTP).

   The differences are as follows:

   (a) An HTTP client pulls files (objects) from a server, whereas an SMTP pushes them (mails) to a server: we say that HTTP is a *pull protocol* whereas SMTP is a *push protocol*.

   (b) SMTP requires the body of each message to be encoded in ASCII, contrary to HTTP, thus accentuated characters or binary data must be encoded and decoded.

   (c) When a file consists of several parts (like text and images), SMTP places all of them in the same message, contrary to HTTP which puts them in separate messages.

7. *What are the components needed to make (use of) cookies?*

   Cookies contain the following components:

   - a cookie header line in the HTTP response;
   - a cookie header line in the HTTP request;
   - a cookie file kept on the user's system and managed by the browser;
   - a database at the server site.

8. *What is the conditional* ***GET*** HTTP *request useful for?*

   Web caching can create a new problem: the cached objects can be *stale*, i.e. the original object (at the server side) may have changed since the last request.

   The mechanism in HTTP which ensures that all retrieved objects are up-to-date despite caching is called conditional `GET`.

9. *List five tasks that a protocol layer can perform. Is it possible that one (or more) of these tasks could be performed by two (or more) layers?*

   Five generic tasks are error control, flow control, segmentation and reassembly, multiplexing and connections set-up. These tasks can be duplicated at different levels. For example, error control is often provided at more than one layer.

10. *What information is used by a process running on one host to identify a process running on another host?*

    The IP address of the destination host and the port number of the destination socket.

2 Problems

1. *Consider sending a file of $M \times L$ bits over a path of Q links. Each link transmits at R bits per second. The network is lightly loaded so that there are no queuing delays. When a form of packet switching is used, the $M \times L$ bits are broken up into M packets, each packet with L bits. Propagation delay is negligible.*

   (a) *Suppose the network is a packet-switched virtual circuit network. Denote the VC set-up time by t_s seconds. Suppose the sending layers add a total of h bits of header to each packet. How long does it take to send the file from source to destination?*

   The time to transmit one packet onto a link is $(L+h)/R$. The time to deliver the first of the M packets to the destination is $Q(L+h)/R$. Every $(L+h)/R$ seconds a new packet from the $M-1$ remaining packets arrives at destination (this is due to pipelining). We must also add the initial setup time, because we use a VC network. Thus the total delay is

   $$t_s + (Q+M-1)\frac{L+h}{R}$$

   (b) *Suppose the network is a packet-switched datagram network and a connectionless service is used. Now suppose each packet has $2h$ bits of header. How long does it take to send the file?*

   Using a datagram network with a connectionless service implies there is no setup needed. Another difference with the previous situation is the header size. Therefore the total delay is simply

   $$(Q+M-1)\frac{L+2h}{R}$$

(c) *Repeat case 1b but assume message switching is used (that is,* $2h$ *bits are added to the message, and the message is not segmented).*

The time to transmit the entire message over one link is $(LM + 2h)/R$. The time to transmit the entire message over Q links (i.e. to destination) is thus

$$Q\frac{LM + 2h}{R}$$

(d) *Finally, suppose that the network is a circuit-switched network. Further suppose that the transmission rate of the circuit between source and destination is* R *bit/s. Assuming* t_s *seconds of set-up and* h *bits of header appended to the entire file, how long does it take to send the file?*

Because there is no store-and-forward delay at the switches, the total delay does not depend on the number of links. Also we must add the setup time. Therefore the end-to-end delay is

$$t_s + \frac{ML + h}{R}$$