other SNMP/MIB-related RFCs define extensions to the core set of MIB variables—for example, variables that are specific to a particular network technology or to a particular vendor's product. Perkins et al. [PM97] provides a good introduction to SNMP and MIBS.

The National Research Council report on the ossification of the Internet can be found in [NRC01], and a proposal to use overlay networks to introduce disruptive technology was made by Peterson et al., Anderson, Culler, and Roscoe [PACR02]. The original case for overriding BGP routes was made by Savage et al., Collins, Hoffman, Snell, and Anderson [SCH+99]. The idea of using DNS to load-balance a set of servers is described in RFC 1794 [Bri95]. A comprehensive treatment of the issue of web caching versus replicated servers can be found in Rabinovich and Spatscheck's book [RS02]. Wang, Pai, and Peterson explore the design space for redirectors [WPP02].

Finally, we recommend the following live reference to help keep tabs on the rapid evolution of the Web and for a wealth of information related to Web-related standards and history:

- http://www.w3.org/: World Wide Web Consortium.

EXERCISES

1. Discuss how you might rewrite SMTP or HTTP to make use of a hypothetical general-purpose request/reply protocol. Could an appropriate analog of persistent connections be moved from the application layer into such a transport protocol? What other application tasks might be moved into this protocol?

2. Most Telnet clients can be used to connect to port 25, the SMTP port, instead of to the Telnet port. Using such a tool, connect to an SMTP server and send yourself (or someone else, with permission) some forged email. Then examine the headers for evidence the message isn't genuine.

3. What features might be used by (or added to) SMTP and/or a mail daemon such as sendmail to provide some resistance to email forgeries as in the previous exercise?

4. Find out how SMTP hosts deal with unknown commands from the other side, and how in particular this mechanism allows for

the evolution of the protocol (e.g., to "extended SMTP"). You can either read the RFC or contact an SMTP server as in Exercise 2, above, and test its responses to nonexistent commands.

5. As presented in the text, SMTP involves the exchange of several small messages. In most cases, the server responses do not affect what the client sends subsequently. The client might thus implement *command pipelining*: sending multiple commands in a single message.
   (a) For what SMTP commands does the client need to pay attention to the server's responses?
   (b) Assume the server reads each client message with gets() or the equivalent, which reads in a string up to a <LF>. What would it have to do even to detect that a client had used command pipelining?
   (c) Pipelining is nonetheless known to break with some servers; find out how a client can negotiate its use.

6. One of the central problems faced by a protocol such as MIME is the vast number of data formats available. Consult the MIME RFC to find out how MIME deals with new or system-specific image and text formats.

7. MIME supports multiple representations of the same content using the multipart/alternative syntax; for example, text could be sent as text/plain, text/richtext, and application/postscript. Why do you think plaintext is supposed to be the *first* format, even though implementations might find it easier to place plaintext after their native format?

8. Consult the MIME RFC to find out how base64 encoding handles binary data of a length not evenly divisible by three bytes.

9. The POP3 Post Office Protocol only allows a client to retrieve email, using a password for authentication. Traditionally, to *send* email a client would simply send it to its server (using SMTP) and expect that it be relayed.
   (a) Explain why email servers often no longer permit such relaying from arbitrary clients.
   (b) Propose an SMTP option for remote client authentication.
   (c) Find out what existing methods are available for addressing this issue.

10. In HTTP version 1.0, a server marked the end of a transfer by closing the connection. Explain why, in terms of the TCP layer, this was a problem for servers. Find out how HTTP version 1.1 avoids this. How might a general-purpose request/reply protocol address this?

11. Find out how to configure an HTTP server so as to eliminate the 404 not found message and have a default (and friendlier) message returned instead. Decide if such a feature is part of the protocol or part of an implementation or is technically even permitted by the protocol. (Documentation for the apache HTTP server can be found at www.apache.org.)

12. Why does the HTTP GET command on page 712,

```
GET http://www.cs.princeton.edu/index.html HTTP/1.1
```

contain the name of the server being contacted? Wouldn't the server already know its name? Use Telnet, as in Exercise 2, above to connect to port 80 of an HTTP server and find out what happens if you leave the host name out.

13. When an HTTP server initiates a close() at its end of a connection, it must then wait in TCP state FIN_WAIT_2 for the client to close the other end. What mechanism within the TCP protocol could help an HTTP server deal with noncooperative or poorly implemented clients that don't close from their end? If possible, find out about the programming interface for this mechanism and indicate how an HTTP server might apply it.

14. Suppose a very large website wants a mechanism by which clients access whichever of multiple HTTP servers is "closest" by some suitable measure.
    (a) Discuss developing a mechanism within HTTP for doing this.
    (b) Discuss developing a mechanism within DNS for doing this.
    Compare the two. Can either approach be made to work without upgrading the browser?

15. Application protocols such as FTP and SMTP were designed from scratch, and they seem to work reasonably well. What is it about Business to Business and Enterprise Application Integration protocols that calls for a Web Services protocol framework?

16. Choose a Web Service with equivalent REST and SOAP interfaces, such as those offered by Amazon.com. Compare how equivalent operations are implemented in the two styles.

17. Get the WSDL for some SOAP-style Web Service and choose an operation. In the messages that implement that operation, identify the fields.

18. Suppose some receivers in a large conference can receive data at a significantly higher bandwidth than others. What sorts of things might be implemented to address this? (Hint: Consider both the Session Announcement Protocol (SAP) and the possibility of utilizing third-party "mixers.")

19. How might you encode audio (or video) data in two packets so that if one packet is lost, then the resolution is simply reduced to what would be expected with half the bandwidth? Explain why this is much more difficult if a JPEG-type encoding is used.

20. Explain the relationship between Uniform Resource Locators (URLs) and Uniform Resource Identifiers (URIs). Give an example of a URI that is *not* a URL.

21. Find out what other features DNS MX records provide in addition to supplying an alias for a mail server; the latter could, after all, be provided by a DNS CNAME record. MX records are provided to support email; would an analogous WEB record be of use in supporting HTTP?

22. ARP and DNS both depend on caches; ARP cache entry lifetimes are typically 10 minutes, while DNS cache lifetimes are on the order of days. Justify this difference. What undesirable consequences might there be in having too long a DNS cache entry lifetime?

23. IPv6 simplifies ARP out of existence by allowing hardware addresses to be part of the IPv6 address. How does this complicate the job of DNS? How does this affect the problem of finding your local DNS server?

24. DNS servers also allow reverse lookup; given an IP address 128.112.169.4, it is reversed into a text string 4.169.112.128.in-addr.arpa and looked up using DNS PTR records (which form a hierarchy of domains analogous to that for the address domain

hierarchy). Suppose you want to authenticate the sender of a packet based on its host name and are confident that the source IP *address* is genuine. Explain the insecurity in converting the source address to a name as above and then comparing this name to a given list of trusted hosts. (Hint: Whose DNS servers would you be trusting?)

25. What is the relationship between a domain name (e.g., cs.princeton.edu) and an IP subnet number (e.g., 192.12.69.0)? Do all hosts on the subnet have to be identified by the same name server? What about reverse lookup, as in the previous exercise?

26. Suppose a host elects to use a name server not within its organization for address resolution. When would this result in no more total traffic, for queries not found in any DNS cache, than with a local name server? When might this result in a better DNS cache hit rate and possibly less total traffic?

27. Figure 9.17 shows the hierarchy of name servers. How would you represent this hierarchy if one name server served multiple zones? In that setting, how does the name server hierarchy relate to the zone hierarchy? How do you deal with the fact that each zone may have multiple name servers?

28. Use the whois utility/service to find out who is in charge of your site, at least as far as the InterNIC is concerned. Look up your site both by DNS name and by IP network number; for the latter you may have to try an alternative whois server (e.g., whois-h whois.arin.net. . .). Try princeton.edu and cisco.com as well.

29. Many smaller organizations have their websites maintained by a third party. How could you use whois to find if this is the case and, if so, the identity of the third party?

30. One feature of the existing DNS .com hierarchy is that it is extremely wide.

(a) Propose a more hierarchical reorganization of the .com hierarchy. What objections might you foresee to your proposal's adoption?

(b) What might be some of the consequences of having most DNS domain names contain four or more levels, versus the two levels of many existing names?

31. Suppose, in the other direction, we abandon any pretense at all of DNS hierarchy and simply move all the .com entries to the root name server: www.cisco.com would become www.cisco, or perhaps just cisco. How would this affect root name server traffic in general? How would this affect such traffic for the specific case of resolving a name like cisco into a web server address?

32. What DNS cache issues are involved in changing the IP address of, say, a web server host name? How might these be minimized?

33. Take a suitable DNS-lookup utility (e.g., dig) and disable the recursive lookup feature (e.g., with +norecursive), so that when your utility sends a query to a DNS server and that server is unable to fully answer the request from its own records, the server sends back the next DNS server in the lookup sequence rather than automatically forwarding the query to that next server. Then carry out manually a name lookup such as that in Figure 9.18; try the host name www.cs.princeton.edu. List each intermediate name server contacted. You may also need to specify that queries are for NS records rather than the usual A records.

34. Find out if there is available to you an SNMP node that will answer queries you send it. If so, locate some SNMP utilities (e.g., the ucd-snmp suite) and try the following:
    (a) Fetch the entire system group, using something like
    ```
    snmpwalk nodename public system
    ```
    Also try the above with 1 in place of system.
    (b) Manually walk through the system group, using multiple SNMP GET-NEXT operations (e.g., using snmpgetnext or equivalent), retrieving one entry at a time.

35. Using the SNMP device and utilities of the previous exercise, fetch the tcp group (numerically group 6) or some other group. Then do something to make some of the group's counters change, and fetch the group again to show the change. Try to do this in such a way that you can be sure your actions were the cause of the change recorded.

36. What information provided by SNMP might be useful to someone planning the IP spoofing attack of Exercise 17 in Chapter 5? What other SNMP information might be considered sensitive?

37. How would one design a CDN redirection mechanism using only HTTP 302 redirects or only DNS? What are the limitations of each approach? Is a combination of the two mechanisms feasible?

38. What problem would a DNS-based redirection mechanism encounter if it wants to select an appropriate server based on current load information?

39. Imagine a situation in which multiple CDNs exist and want to peer with each other (analogous to the way autonomous systems peer with each at the IP layer) for the purpose of delivering content to a larger set of end users. For example, CDN A might serve content on behalf of one set of content providers and CDN B might serve content on behalf of another set of content providers, where both A and B have a physical footprint that allows them to deliver that content to disjoint sets of end users. Sketch how CDNs A and B can use a combination of DNS redirection and HTTP 302 redirects to deliver content from CDN A's content providers to CDN B's end users (and *vice versa*).

40. Imagine a CDN configured as a *caching hierarchy*, with end users accessing content from edge caches, which fetch the content for a parent cache upon a cache miss, and so on up to a root cache, which ultimately fetches content from an origin server. What metrics would guide provisioning decisions to (a) add more storage capacity to a given cache versus (b) adding an additional level to the caching hierarchy.

41. A multicast overlay effectively *pushes* streaming content from a single source to multiple destinations, with no caching of the stream at the intermediate nodes. A CDN effectively *pulls* content (including videos) down a caching hierarchy, caching it at the intermediate nodes. Show by example how these two can be viewed as duals of each other. Explain why a CDN can be viewed as equivalent to *asynchronous multicast*. (Hint: Think TiVo.)

42. Consider the following simplified BitTorrent scenario. There is a swarm of 2^n peers and, during the time in question, no peers join or leave the swarm. It takes a peer 1 unit of time to upload or download a piece, during which time it can only do one or the

other. Initially, one peer has the whole file and the others have nothing.

(a) If the swarm's target file consists of only 1 piece, what is the minimum time necessary for all the peers to obtain the file? Ignore all but upload/download time.

(b) Let x be your answer to the preceding question. If the swarm's target file instead consisted of 2 pieces, would it be possible for all the peers to obtain the file in less than $2x$ time units? Why or why not?