1. 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.

   Implementers are free to add new subtypes to MIME, but certain default interpretations may apply. For example, unrecognized subtypes of the application type are to be treated as being equivalent to application/octet-stream. New experimental types and subtypes can be introduced; names of such types are to begin with X- to mark them as such. New image and text subtypes may be formally registered with the IANA; senders of such subtypes may also be encouraged to send the data in one of the "standard" formats as well, using multipart/alternative.

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

   The base64 encoding actually defines 65 transmission characters; the 65th, "=", is used as a pad character. The data file is processed in input blocks of three bytes at a time; each input block translates to an output block of four 6-bit pieces in the base64 encoding process. If the final input block of the file contains one or two bytes, then zero-bits are first added to bring the data to a 6-bit boundary (if the final block is one byte, we add four zero bits; if the final block is two bytes, we add two zero bits). The two or three resulting 6-bit pieces are then encoded in the usual way, and two or one "=" characters are appended to bring the output block to the required four pieces. In other words, if the encoded file ends with a single =, then the original file size was $\equiv 2 \pmod 3$; if the encoded file ends with two =s then the original file size was $\equiv 1 \pmod 3$.

3. 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 to do this.

   A mechanism within HTTP would of course require that the client browser be aware of the mechanism. The client could ask the primary server if there were alternate servers, and then choose one of them. Or the primary server might*tell* the client what alternate to use. The parties involved might measure "closeness" in terms of RTT, in terms of measured throughput, or (less conveniently) in terms of preconfigured geographical information.

   (b) Discuss developing a mechanism within DNS to do this.

   Within DNS, one might add a WEB record that returned multiple server addresses. The client resolver library call (*e.g.* gethostbyname()) would choose the "closest", determined as above, and return the single closest entry to the client application as if it were an A record. Also, some CDNs today use DNS resolution to try to direct a client to a nearby CDN node. This is usually done without the clients knowledge.

   Can either approach be made to work without upgrading the browser?

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

   Consider the GetObject operation of Amazons S3 (Simple Storage Service)Web Service. GetObjects input message is a GetObjectRequest, and its output message is a GetObjectResponse:

```
<wsdl:operation name="GetObject">
<wsdlsoap:operation soapAction=""/>
<wsdl:input name="GetObjectRequest">
<wsdlsoap:body use="literal"/>
</wsdl:input>
<wsdl:output name="GetObjectResponse">
<wsdlsoap:body use="literal"/>
</wsdl:output>
</wsdl:operation>
```

   A GetObjectRequest message consists of a GetObject element

```
<wsdl:message name="GetObjectRequest">
<wsdl:part element="tns:GetObject" name="parameters"/>
</wsdl:message>
of type GetObjectResult
<xsd:element name="GetObjectResponse">
<xsd:complexType>
<xsd:sequence>
```

```
<xsd:element name=“GetObjectResponse” type=“tns:GetObjectResult”/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
```

which has the following fields:

```
<xsd:complexType name=“GetObjectResult”>
<xsd:complexContent>
<xsd:extension base=“tns:Result”>
<xsd:sequence>
<xsd:element name=“Metadata”
type=“tns:MetadataEntry” minOccurs=“0” maxOccurs=“unbounded”/>
<xsd:element name=“Data”
type=“xsd:base64Binary” nillable=“true”/>
<xsd:element name=“LastModified” type=“xsd:dateTime”/>
<xsd:element name=“ETag” type=“xsd:string”/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
```

5. 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.inaddr.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 IP *address* is genuine. Explain the insecurity in converting the source address to a name as above and then comparing this name to a list of trusted hosts. (Hint: Whose DNS servers would you be trusting?)

   The lookup method here requires trusting of the remote sites DNS PTR data, which may not be trustworthy. Suppose, for example, that it is known that cicada. cs.princeton.edu trusts gnat.cs.princeton.edu. A request for authentication might arrive at cicada from, say, IP address 147.126.1.15, which is not part of the princeton.edu domain. If cicada followed the strategy of the exercise here, it would look up the string 15.1.126.147.in-addr.arpa in the DNS PTR data. This query would eventually reach the DNS server for PTR zone 1.126.147.in-addr.arpa, which if suborned or malicious might well return the string gnat.cs.princeton.edu regardless of the fact that it had no connection with princeton.edu. Hostname strings returned by DNS servers for PTR searches are arbitrary, and need not be related to the servers assigned domain name.

   A forward DNS lookup to confirm the result of the reverse DNS lookup would, however, be reasonably safe.

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

   There is little if any relationship, formally, between a domain and an IP network, although it is nonetheless fairly common for an organization (or department) to have its DNS server resolve names for all the hosts in its network (or subnet), and no others. The DNS server for cs.princeton.edu could, however, be on a different network entirely (or even on a different continent) from the hosts whose names it resolves. Alternatively, each x.cs.princeton.edu host could be on a different network, and each host that is on the same network as the cs.princeton.edu nameserver could be in a different DNS domain.
   If the reverse-mapping PTR records are used, however, then the same nameserver can handle both forward and reverse lookups only when DNS zones do correspond to groups of subnets.

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

   DNS records contain a TTL value, specified by the DNS server, representing how long a DNS record may be kept in the client cache. RFC 1034 puts it this way:

   If a change can be anticipated, the TTL can be reduced prior to the change to minimize inconsistency during the change, and then increased back to its former value following the change.

8. Consider distributing a file $F = 15$ Gbits to N peers. The server has an upload rate of $\mu_s = 30$ Mbps, and each peer has a download rate of $d_i = 2$ Mbps and an upload rate of μ. For $N = 10$, 100, and 1000 and $\mu = 500$ Kbps, 800 Kbps, and 2 Mbps, prepare a table giving the minimum distribution time for each of the combinations of N and μ for both client-server distribution and P2P distribution.

   For calculating the minimum distribution time for client-server distribution, we use the following formula:

$$D_{cs} = \max\{NF/\mu_s, F/d_{min}\}$$

Similarly, for calculating the minimum distribution time for P2P distribution, we use the following formula:

$$D_{P2P} = \max\{F/\mu_s, F/d_{min}, NF/(\mu_s + \sum_{i=1}^{N} \mu_i)\}$$

9. Consider an overlay network with N active peers, with each pair of peers having an active TCP connection. Additionally, suppose that the TCP connections pass through a total of M routers.

   (a) How many nodes and edges are in the corresponding overlay network?

   There are N nodes in the overlay network. There are$N(N-1)/2$ edges.

10. Suppose Bob joins a BitTorrent, but he does not want to upload data to any other peers (so called free-riding).

    (a) Bob claims that he can receive a complete copy of the file that is shared by the swarm. Is Bob's claim possible? Why or why not?

    Yes. His first claim is possible, as long as there are enough peers staying in the swarm for a long enough time. Bob can always receive data through optimistic unchoking by other peers.

    (b) Bob further claims that he can further make his "free-riding" more efficient by using a collection of multiple computers (with distinct IP addresses) in the computer lab in his department. How can he do that?

    His second claim is also true. He can run a client on each machine, and let each client do free-riding, and combine those collected chunks from different machines into a single file. He can even write a small scheduling program to let different machines only asking for different chunks of the file.

11. In this problem, we are interested in finding out the efficiency of a BitTorrent-like P2P file sharing system. Consider two peers Bob and Alice. They join a torrent with M peers in total (including Bob and Alice) that are sharing a file consisting of N chunks. Assume that at a particular time t, the chunks that a peer has are uniformly at random chosen from all N chunks, and no peer has all N chunks. Answer the following questions.

    (a) What is the probability that Bob has all the chunks that Alice has, given that the numbers of chunks that Bob and Alice have are denoted by n_b and n_a?

    (b) Remove part of the conditioning in part (a) to find out the probability that Bob has all the chunks that Alice has, given that Alice has n_a chunks.

    (c) Suppose that each peer in BitTorrent has 5 neighbors. What is the probability that Bob has data that is of interest to at least one of his five neighbors?

    See the paper posted in the resources folder for the correct answers to the above questions.

12. A circular DHT consists of peers 1, 4, 5, 6, 9, 10, 12, and 15. Suppose that a new peer 7 wants to join the DHT and peer 7 initially only knows peer 15's address. What steps are taken?

    Peer 7 would first send peer 15 a message, saying "what will be peer 7's predecessor and successor?" This message gets forwarded through the DHT until it reaches peer 6, who realizes that it will be 7's predecessor and that its current successor, peer 9, will become 7's successor. Next, peer 6 sends this predecessor and successor information back to 7. Peer 7 can now join the DHT by making peer 9 its successor and by notifying peer 6 that it should change its immediate successor to 7.

13. In the circular DHT example above, suppose that peer 4 learns that peer 6 has left.

    (a) How does peer update its successor state information?

    (b) Which peer is now its first successor?

    (c) Which peer is now its second successor?

    Peer 4 learns that peer 6 has just left the system, so Peer 4 asks its first successor (peer 5) for the identifier of its immediate successor (peer 9). Then peer 4 will make peer 9 as its second successor. Note that peer 4 knows that peer 6 was originally the first successor of peer 5, so peer 4 would wait until peer 5 finishes updating its first successor.

14. As DHTs are overlay networks, they may not necessarily match the underlay physical network well in the sense that two neighboring peers might be physically very far away; for example, one peer could be in Asia and its neighbor could be in North America.

    (a) If we randomly and uniformly assign identifiers to newly joined peers, would this assignment scheme cause such a mismatch?

    Yes, that assignment scheme of keys to peers does not consider underlying network at all, so it very likely causes mismatch.

    (b) How would such a mismatch affect the DHT's performance?

    The mismatch may potentially degrade the search performance. For example, consider a logical path p_1 (consisting of only two logical links): $A \longrightarrow B \longrightarrow C$, where A and B are neighboring peers, and B and C are neighboring peers. Suppose that there is another logical path p_2 from A to B (consisting of 3 logical links): $A \longrightarrow D \longrightarrow E \longrightarrow C$. It might be the case that A and B could be very far away physically, and B and C could be very far away physically. But A, D, E, and C are very close physically. In other words, a shorter logical path corresponds to a longer physical path than does a longer logical path.