1. High level questions

a. Suppose Alice and Bob are communicating over an SSL session. Suppose an attacker, who does not have any of the shared keys, inserts a bogus TCP segment into a packet stream with correct TCP checksum and sequence numbers (and correct IP addresses and port numbers). Will SSL at the receiving side accept the bogus packet and pass the payload to the receiving application? Why or why not?

No, the bogus packet will fail the integrity check (which uses a shared MAC key).

b. Consider an HTTP client that wants to retrieve a Web document at a given URL. The IP address of the HTTP server is initially unknown. What transport and application-layer protocols besides HTTP are needed in this scenario?

Application layer protocols: DNS and HTTP
Transport layer protocols: UDP for DNS; TCP for HTTP

c. Suppose you can access the caches in the local DNS servers of your department. Can you propose a way to roughly determine the Web servers (outside your department) that are most popular among the users in your department? Explain.

We can periodically take a snapshot of the DNS caches in those local DNS servers. The Web server that appears most frequently in the DNS caches is the most popular server. This is because if more users are interested in a Web server, then DNS requests for that server are more frequently sent by users. Thus, that Web server will appear in the DNS caches more frequently.

For a complete measurement study, see:
Craig E. Wills, Mikhail Mikhailov, Hao Shang
"Inferring Relative Popularity of Internet Applications by Actively Querying DNS Caches", in IMC’03, October 27-29, 2003, Miami Beach, Florida, USA

d. The OSPF routing protocol uses a MAC rather than digital signatures to provide message integrity. Why do you think a MAC was chosen over digital signatures?

Digital signatures require an underlying Public Key Infrastructure (PKI) with certification authorities. For OSPF, all routers are in a same domain, so the administrator can easily deploy the symmetric key on each router, without the need of a PKI.

2. TCP congestion control

Consider a TCP system implementing slow start and congestion avoidance with fast retransmit and fast recovery. When a connection is setup the congestion window is initialized to one segment and the slow start threshold to 64 segments. To simplify the problem, assume that the timeout is equal to the RTT (an exact estimate) and specify time in units of RTT, such that one time slot is one RTT.

Packet transmissions are such that at each time slot the sender sends all packets in the congestion window. If ACK’s are received in the next time slot, there is no timeout. In
addition, to simplify matters either the entire window is acknowledged or none of its segments are acknowledged.

For a particular connection, ACK's are received in time slots 1-7, 9-30, 32-40, and 42-50. Timeouts occur in slots 8 and 31; in slot 42, three duplicate ACK's are received for the packets sent in time slot 41.

For the system described, plot both the congestion window and the slow start threshold (on the same graph) versus time (slots). Remember to consider the differences between slow start and congestion avoidance with fast retransmit and fast recovery when changing the congestion window.

![Graph of congestion window and threshold over time]

**Sol:**
In slot 8, the congestion window has expanded to 64 segments. Due to the timeout, the next slot will have a congestion threshold of 64/2 = 32 and the congestion window will be reset to 1. The congestion window grows exponentially until slot 14, when it equals the congestion threshold. It then grows linearly until the timeout in slot 31, when the congestion threshold is set to 49/2 = 24, and the congestion window is reset to 1 again. In slot 42, three duplicate ACKs are received. The congestion window is then halved to 37/2 = 18 and begins to grow linearly.

### 3. Security

Suppose Alice wants to send an e-mail to Bob. Bob has a public-private key pair ($K_B^+, K_B^-$), and Alice has Bob's certificate. But Alice does not have a public, private key pair. Alice and Bob (and the entire world) share the same hash function $H(.)$. 
a. In this situation, is it possible to design a scheme so that Bob can verify that Alice created the message? If so, show how with a block diagram for Alice and Bob.

b. Is it possible to design a scheme that provides confidentiality for sending the message from Alice to Bob? If so, show how with a block diagram for Alice and Bob.

Sol:

(a) No, without a public-private key pair or a pre-shared secret, Bob cannot verify that Alice created the message.

(b) Yes, Alice simply encrypts the message with Bob's public key and sends the encrypted message to Bob.

4. HTTP

Suppose within your Web browser you click on a link to obtain a Web page. The IP address for the associated URL is not cached in your local host, so a DNS lookup is necessary to obtain the IP address. Suppose that $n$ DNS servers are visited before your host receives the IP address from DNS; the successive visits incur an RTT of $RTT_1, \ldots, RTT_n$. Further suppose that the Web page associated with the link contains exactly one object, consisting of a small amount of HTML text. Let $RTT_0$ denote the RTT between the local host and the server containing the object.

1) Assuming zero transmission time of the object, how much time elapses from when the client clicks on the link until the client receives the object?

2) Now, suppose the HTML file references eight very small objects on the same server. Neglecting transmission times, how much time elapses with
   a. Non-persistent HTTP with no parallel TCP connections?
   b. Non-persistent HTTP with the browser configured for 5 parallel connections?
   c. Persistent HTTP?

Sol:

1) The total amount of time to get the IP address is $RTT_1 + RTT_2 + \ldots + RTT_n$.

   Once the IP address is known, $RTT_0$ elapses to set up the TCP connection and another $RTT_0$ elapses to request and receive the small object. The total response time is

   $2 RTT_0 + RTT_1 + RTT_2 + \ldots + RTT_n$
5. P2P

Consider distributing a file of $F=15\text{Gbits}$ to $N$ peers. The server has an upload rate of $u_s = 30$ Mbps, and each peer has a download rate of $d_i = 2$ Mbps and an upload rate of $u$. For $N = 10$, 100, and 1,000 and $u = 300$ Kbps, 700 Kbps, and 2 Mbps, prepare a chart giving the minimum distribution time for each of the combinations of $N$ and $u$ for P2P distribution.

Sol:

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

$$D_{p2p} = \max\{F/u_s, F/d_{\text{min}}, NF/(u_s + \sum_{i=1}^{N} u_i)\}$$

Where, $F = 15$ Gbits = $15 \times 1024$ Mbits

$u_s = 30$ Mbps

$d_{\text{min}} = d_i = 2$ Mbps

Note, $300\text{Kbps} = 300/1024$ Mbps.

<table>
<thead>
<tr>
<th>Peer to Peer</th>
<th>10</th>
<th>100</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>300 Kbps</td>
<td>7680</td>
<td>25904</td>
<td>47559</td>
</tr>
<tr>
<td>700 Kbps</td>
<td>7680</td>
<td>15616</td>
<td>21525</td>
</tr>
<tr>
<td>2 Mbps</td>
<td>7680</td>
<td>7680</td>
<td>7680</td>
</tr>
</tbody>
</table>