Solution 1 (Chapter 8 R6): Symmetric key vs public key encryption

Part a:
N*(N-1)/2: Each individual shares a secret key with its N-1 neighbors. There are N individuals. Because symmetric keys are identical, divide by 2 to avoid double-counting keys.

Part b:
2N: N public keys, and N private keys.

Solution 2 (Chapter 2 R20): BitTorrent

Bob will not necessarily provide chunks to Alice. Alice may be opportunistically unchoking her link to Bob.

Solution 3: RIP

Part a:

<table>
<thead>
<tr>
<th>SRC/DST</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>4</td>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Part b:
Suppose the link between B-D fails. Without split-horizon, C will advertise to D that it has a cost 2 link to B. We now have a routing loop.

Solution 4: Wireless

Part a:
False. RTS/CTS may be used to protect data exchange, but is not required. In many cases the
RTS/CTS exchange is too inefficient as these frames are sent at the basic rate. By default, 802.11 relies on CSMA/CA only.

Part b:

No, there would be no advantage to sending the RTS and CTS. If we are trying to prevent collisions, we want to send a small frame that has a lower likelihood of collision. Sending RTS/CTS frames the same size as DATA and ACK would imply that we might as well retransmit the original DATA/ACK rather than go through the RTS/CTS exchange.

Problem 5:

Part a:
UDP offers less overhead than TCP, and can be used in cases where the overhead is unacceptable (such as real-time applications like VoIP). Because UDP is "lighter" than TCP, it offers no guarantees of reliability or connection that TCP would provide. If the application does not need reliability or can handle reliability itself, then UDP would be more useful than TCP.

Part b:
UDP sockets are represented by the tuple (protocol (UDP in this case), IP, port). In this case, both packets would arrive at the UDP socket (UDP, C, 6789). Host C can differentiate between them by checking the source IP in the IP packet and the source port from the UDP header.

Contrast with TCP sockets, represented by (TCP protocol, source IP, source port, destination IP, destination port). Multiple sockets can listen with the same source IP and source port, which is how a HTTP server listening on port 80 can serve multiple clients. Destination IP/port can be * to indicate any destination.

Problem 6:

Part a:
Given a packet forwarding determines which interface to send the packet out from.

Routing is used to populate the table that the forwarder uses for lookup.

Part b:
From the RFC:

*Full Cone*: A full cone NAT is one where all requests from the same internal IP address and port are mapped to the same external IP address and port. Furthermore, any external host can send a packet to the internal host, by sending a packet to the mapped external address.
Restricted Cone: A restricted cone NAT is one where all requests from the same internal IP address and port are mapped to the same external IP address and port. Unlike a full cone NAT, an external host (with IP address X) can send a packet to the internal host only if the internal host had previously sent a packet to IP address X.

Port Restricted Cone: A port restricted cone NAT is like a restricted cone NAT, but the restriction includes port numbers. Specifically, an external host can send a packet, with source IP address X and source port P, to the internal host only if the internal host had previously sent a packet to IP address X and port P.

Symmetric: A symmetric NAT is one where all requests from the same internal IP address and port, to a specific destination IP address and port, are mapped to the same external IP address and port. If the same host sends a packet with the same source address and port, but to a different destination, a different mapping is used. Furthermore, only the external host that receives a packet can send a UDP packet back to the internal host.

Part c:

A /24 is the equivalent of a Class C block.

Problem 7:

It depends.

If packets are coming in at a rate that the router cannot process, then packets must necessarily be dropped. If packets had been coming in before uniformly, doubling queue lengths will likely end up with the router dropping just as many packets, but increase packet latency.

If, on the other hand, traffic had been particularly bursty, and the increased queue length serves to smooth these bursts, then it may be a good idea. As an exaggerated example, imagine a router that is able to process 1 pkt/us that has a queue length of 500K packets. If every second, in the span of 1 us, the router receives 1M packets all at once, it may make sense to double the queue size: the router can handle all the packets over the course of a period, just not bursts of them. In these cases, note that each packet's varying RTT may still affect TCP's RTT estimation, causing undesirable behavior.

Problem 8:

For load balancing, the fields that would most immediately be used would be the source and destination fields in the IP, TCP and UDP headers. The load balancer would modify the source fields to make packets appear as if they originated at the load balancer. It would then modify the destination fields to direct the packets to the appropriate node out of the pool of available nodes.
If the load balancer is also a router, it would also decrement the IP TTL field. Finally, the load balancer would also have to modify the checksum field in the headers after changing the TTL and source and destination addresses.

It is also conceivable that the load balancer could use the window size field as a measure of load on the nodes it controls.

**Problem 9:**

Loss Rate \( = \) Number of Packets Lost / Number of Packets Sent

\[
= \frac{1}{(W/2) + (W/2 +1) + (W/2+2) + \ldots + W) }
\]

\[
= \frac{1}{\sum_{i=0}^{W/2} (W/2 + i) }
\]

\[
\approx \frac{1}{(3 W^2/8) } \quad \text{(summing the arithmetic series and assuming W is large)}
\]

**Problem 10:**

Three IP address blocks that satisfy the constraints are as follows:

- 223.1.17.128/25 for Subnet 2
- 223.1.17.0/28 for Subnet 3
- 223.1.17.64/26 for Subnet 1