Solution to COMP5311 Assignment 1

QING Pei, 11500811G

November 8, 2011

1 Problem 1

1.1 a

The second fragment is 40 bytes in size.

```
| IP | data |
| 20 | 1480 |
```

The original IP packet is 20+1480 bytes. This 1500-byte data will be encapsulated when tunneled. Therefore, the payload size exceeds 1480, which is the tunnel MTU, and need fragmentation. So the original packet is broken into 1480- and 20-byte fragments:

After encapsulation, the packet would become:

```
| Tunnel hdr | data |
| 20 | 1480 |
```

and

```
| Tunnel hdr | rest of data |
| 20 | 20 |
```

In the first fragment, the payload length is 1480 bytes, which is

1.2 b

The fragments will be reassembled at the *tunnel endpoint* if no further fragmentation occurs inside the tunnel.

Reassembly of the fragments will only happen at the destination specified in the tunnel header. In this case, the destination is set to the IP of tunnel endpoint.

1.3 c

Still at the *tunnel endpoint.*

If there is no further tunneling, any fragmentation in the tunnel will just take the destination IP in the tunnel header, which is the IP of tunnel endpoint, as the destination. This does not alter the reassembler.

2 Problem 2

a $SEQ < rcv\_nxt$. As long as the first byte of data in the TCP packet has been received before, "some" data are duplicate.

b $SEQ + LEN < rcv\_nxt$. If the last byte of data in the TCP packet has been received before, all the data are duplicate. If the receive buffer is not large enough, $SEQ < rcv\_nxt$ should also be checked.

c *Sender will keep retransmitting this packet if ACK is not sent for duplicated.* If the entire payload is a duplicate, this segment is retransmitted from the sender. The receiver has sent the ACK before, but it has been either corrupted or lost or timed out. Unless the sender receives the ACK for this segment, it will keep retransmitting.

3 Problem 3

a $snd\_nxt = 1$. Originally, server's snd_nxt is 0. After sending the SYN-ACK packet, it increases by 1.

b $snd\_nxt = 1 + 659 = 660$. Originally, client's snd_nxt is 0. After sending the SYN packet, it becomes 1. Sending a data packet will increase the pointer by data length, which is 659 in this case.

c $rcv\_nxt = 1+1430+1430 = 2861$. Receiving SYN increases the pointer by 1. Receiving data packets increases the pointer by data length. 1 SYN and 2 data packets have been received so far.

d $AN = 1 + 1430 + 1430 + 1002 + 1430 + 469 + 215 = 5977$. AN in the last packet is to acknowledge all the data sent from server received by the client so far. SYN packet contributes 1 and data packet contributes data length to this cumulative counter.