Network Systems (201300179/201400431), Test 3

March 20, 2015, 13:45–15:15

Brief answers

---

1. Distance-vector routing

2 pt (a)

1 minute: after the first iteration, each node has learnt about its direct neighbours, and since all nodes have a link to each other, that's enough.

2 pt (b)

2 minutes, since the longest optimal path is between A and C via B, which has two hops.

2 pt (c)

Cost to A is 0, to B is 1, and to C is 2. This same vector is sent to both neighbours.

2 pt (d)

To B, it advertises only itself (with cost 0), because it routes both B and C via this neighbour. To C it advertises all three nodes, exactly as in the previous question.

3 pt (e)

A had not advertised to B any path for C, so after the B-C link breaks, B doesn't know any path to C. B will thus drop C from its table in the first iteration, and no longer advertises to A that it can reach C. A then has no hesitation in choosing its direct link to C (with cost 20), and advertises this to B in the second iteration. Then B will also start using the path via A. Depending on how you count them, 2 or 3 minutes.

---

2. Addressing and routing in the internet

3 pt (a)

So that's $10 \cdot 10^9 \cdot 1000 = 10^{13}$ devices to be addressed. With an HD ratio of not more than 0.87, that means we need about $10^{13/0.87} = 8.76 \cdot 10^{14}$ addresses. That's a bit less than 2^{50}, so 50 bits would suffice.

Of course, using e.g. 0.80 as the HD ratio is also fine.

2 pt (b) Network Address Translators (NATs) do not just change the IP address, but also the (TCP or UDP) portnumber. Why is this necessary?

Because there might be *multiple* computers behind the NAT, setting up a connection to the *same remote computer*, using the *same portnumbers*; if the NAT would only replace their IP addresses by its own global address, and not change anything else, these connections would not be distinguishable.

2 pt (c)

Definitely not in the mobile device, because the home agent needs to have a stable and permanently reachable IP address.
Integration in the fixed host also doesn't work because the home agent should be on the mobile node's home network. (However, the "route optimization" effectively makes the fixed host also do what normally the home agent does, namely send packets to the foreign node.)

---

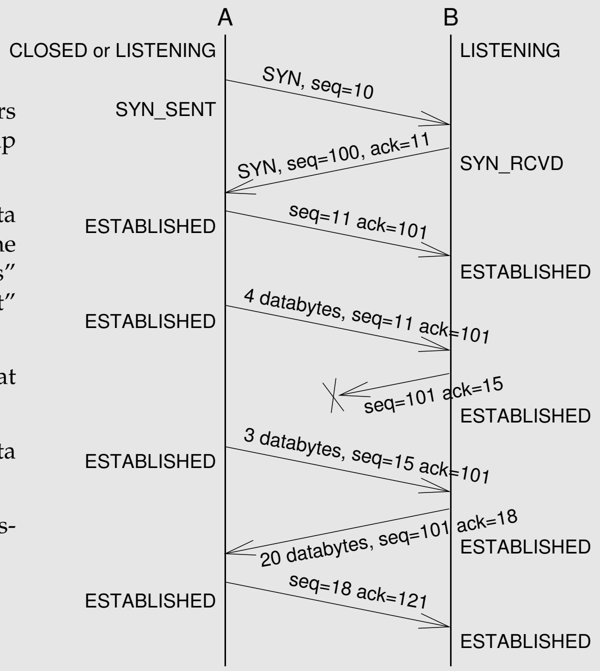
3. TCP

4 pt (a)

See the figure.

Some frequently made errors:

- Forgetting to increment the seq and ack numbers around the 3-way handshake (the SYN takes up one byte of sequence number space).
- Already incrementing the seq number in the data packets; it is the number of the first databyte in the packet, not the last, so the amount of data in "this" data packet only affects the seq nr of the "next" packet going in the same direction.
- Incrementing the seq number after packets that don't contain either data or a SYN (or FIN) flag.
- Incrementing both seq and ack numbers for data transfer going in one direction.
- Incrementing seq numbers twice, once upon transmission and once in the acknowledgement.
- Swapping seq and ack numbers.
- Restarting seq and ack numbers at 0.
- Having packets in which seq or ack are set to 0, to "–", or to "not applicable". All TCP packets have both a valid sequence number and a valid acknowledgement number, with the sole exception of the initial SYN (because then there's no sequence number from the other party to acknowledge yet).



2 pt (b)

No, they are not waiting anymore. TCP's acknowledgements are cumulative, so the lost ack is no problem at all; the next ack, with ack=18, also acknowledges correct receipt of the earlier packet with 4 data bytes.

3 pt (c)

See the diagram at (a).

A's initial state can be either CLOSED or LISTENING. If you mention both or only CLOSED, that's fine for a full score. If you mention *only* LISTENING, the score was a bit lower, because this is a rather exotic situation, namely that side A has first decided to listen for incoming connections and then changed its mind to actively open the connection.

3 pt (d)

No, none of them would have an influence:
Timestamps can be used for accurate RTT measurement (won't make a difference here, with so few packets and no timeouts) or for detecting wrapped sequence numbers (doesn't happen here with so few packets).
Window scaling makes it possible to advertise larger windows that 65535 bytes, but here far fewer bytes are sent so such large windows are not needed.
SACK makes it possible to tell that some later bytes have been received while earlier bytes are still missing; that doesn't happen here because no data packets are lost or reordered.

End of this exam.