Chair of Distributed Systems and Security
School of Computation, Information and Technology
Technical University of Munich

Computer Networking and IT Security (INHN0012)

Tutorial 10

Problem 1 Dynamic Routing

We consider the network shown in Figure 1.1. The routers are using RIP as dynamic routing protocol. The tables next to the routers represent the (simplified) routing table of the respective router containing the destination **Dst**, next hop **NH**, and the costs.

Dst	NH	Cost
B	B	0
C	C	0
D	C	1
E	C	2

Dst	NH	Cost
A	A	0
C	C	0
D	C	1
E	C	2

Dst	NH	Cost
A	A	0
B	B	0
D	D	0
E	D	1

Dst	NH	Cost
A	C	1
B	C	1
C	C	0
E	E	0

Dst	NH	Cost
A	D	2
B	D	2
C	D	1
D	D	0

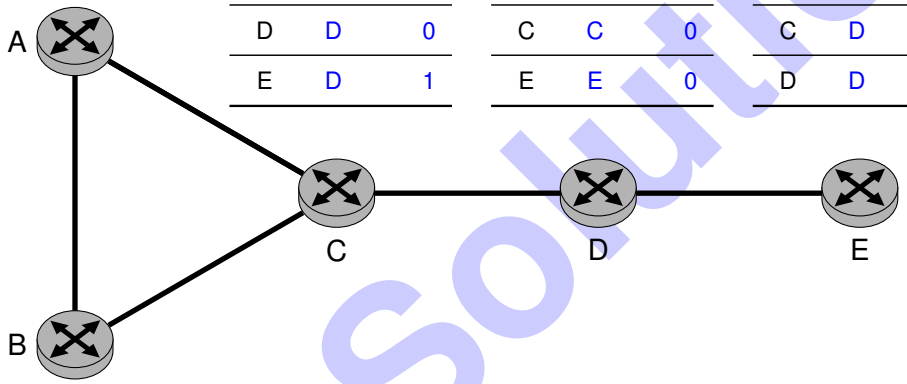

Figure 1.1: Topology and initial routing tables at boot time

a)* Which metric is used by RIP? (Without reason)

Hop Count

b)* RIP is a distance vector protocol. Explain the difference to link state protocols.

The routers only know the next hop and distance for a destination whole link state protocols have detailed view of the network (or parts of it).

c)* RIP is an interior gateway protocol. Explain the difference to exterior gateway protocols.

IGPs are used within a single autonomuous system while EGPs are used between autonomuous systems.

d)* To what extent are networks limited that use solely RIP as routing protocol?

The maximum hop count for RIP is 15, thus the “diameter” of those networks cannot be larger than that.

e)* Which information is contained in a routing update sent by RIP?

Solely the reachable destinations and the cost.(In particular not the next hop.)

f)* Reason whether or not RIP always chooses the shortest path in based on the hop count.

Yes, hop count is RIP's sole metric.

g)* Reason whether or not RIP always chooses the fastest route in terms of bandwidth.

No, the number of hops does not tell anything about available bandwidth.

h) Fill in the routing tables in Figure 1.1 (without intermediate steps) such that the tables represent a converged state.

Assume the link between routers D and E fails. Router D obviously recognizes the fail. Answer the following questions in the given order.

i) Router D sends a periodic update. Describe its immediate effect on the other routers.

C is informed about the fail and will remove the route to E via D.A and B do not receive the update from D.

j) Now, router A sends a periodic update. Describe its immediate effect on the other routers.

Since A still assumes there is a route to E via C, it is included in the update.B will ignore that since it also thinks there is still a route to E via C.
However, C now wrongly assumes that there is a route to E via A with cost 3 and installs this new route.

k) Describe the problem that will now arise and how it can be solved.

Count-to-Infinity: the non-existing route to E will circulate between A, B, and C until the tombstone of 15 is reached.
Possible solutions include split horizon, poison reverse, and triggered updates (where the latter only speed up the process at cost of network traffic).

Problem 2 Sliding Window Protocols

We consider a sliding window protocol whose transmit and receive windows are $w_s = w_r = 2$. Let the sequence number space be $\mathcal{S} = \{0, 1\}$. The error handling is analogous to Go-Back-N. Figure 2.1 shows a data transmission, where the flashes represent segments lost. So the first two ACKs do not reach the sender.

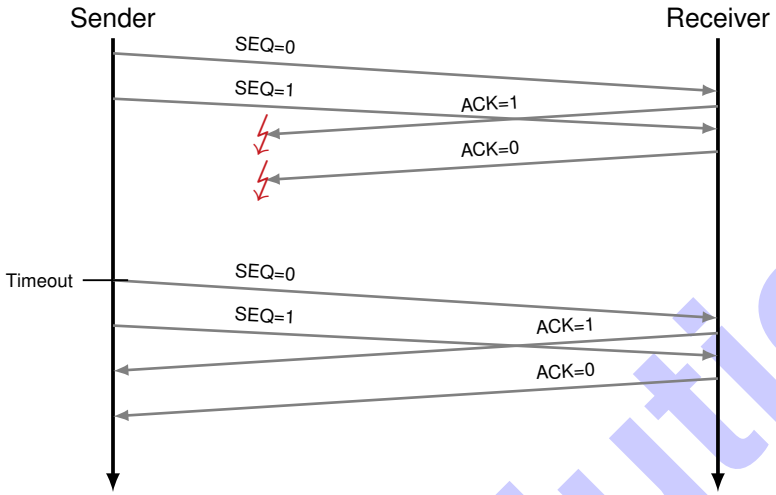

Figure 2.1: Modified Alternating Bit Protocol

a)* Which problem occurs with the above protocol?

After receiving the first two segments, the receiver forwards them to the next higher layer. He does not know after sending the first two ACKs that they did not reach the sender. The transmitter will repeat these first two segments after a timeout.
These carry the same sequence numbers as the first two segments. Unfortunately, however, the receiver expects two **new** segments with just these sequence numbers. So the receiver is not able to distinguish whether it is a repetition or two new segments. It will therefore also forward these two segments to the next higher layer, which has now received the same data **twice**.

b) Adjust $\mathcal{S}$ so that the procedure can work correctly. Give reasons for your answer.

Since the retry procedure is Go-Back-N, the receiver accepts only the next expected segment at a time (regardless of the fact that its receive window is larger).
In this case, already the sequence number space $\mathcal{S} = \{0, 1, 2\}$ is sufficient, since in this way there is always a „ guard distance “ from a sequence number.

In the following, we will analyze the two methods Go-Back-N and Selective Repeat. The sequence numbers $s \in \mathcal{S}$ have a length of 4 bit. Answer the following questions **for both, Go-Back-N and Selective Repeat**.

c)* How many unconfirmed segments may the sender send at a time without risking behaviour as in the previous tasks? Justify your answer with examples. (Note: think of confirmations lost at the most inopportune moments).

Let w_S denote the transmit window. Then applies in general:

- **Go-Back-N:**
  With Go-Back-N, the receiver always accepts only the next expected segment. Segments arriving *out-of-order* are ignored. The worst case for Go-Back-N is that all segments are successfully transmitted, but then all acknowledgements are lost on the way to the sender.This case is shown in figure 2.1. This can be remedied by always sending one segment less than the total number of sequence numbers available. For this reason, $w_s \leq |\mathcal{S}| - 1$ must always hold for the transmit window at Go-Back-N.

- **Selective Repeat:**
  The worst case for Selective Repeat is that w_s segments are successfully transmitted, but subsequently all acknowledgements are lost.Let x be the sequence number of the first segment. In this case, the receiver – having successfully received all segments — will advance its receive window by w_s. However, the transmitter thinks that all segments have been lost. For this reason it will repeat the segments with the sequence numbers $x, x+1, \ldots, x+w_s-1$. The condition now is that none of the sequence numbers of the **repeated** segments must fall within the receiver's current receive window.Otherwise, the receiver would accept a repetition as a new segment.

  For $w_s = 4$ and $|\mathcal{S}| = 8$ you can now quickly see from an example that everything works. If, on the other hand, $w_s = 5$ is chosen, a conflict occurs.

  For $w_s = 3$ and $|\mathcal{S}| = 7$ everything is also fine. However, if $w_s = 4$ is chosen here, a conflict occurs again.

  So, in general, for the transmit window $w_s \leq \left\lfloor \frac{|\mathcal{S}|}{2} \right\rfloor$.

  Note:
  Assuming that there are no cumulative acknowledgements, there is another case that leads to the same result: The confirmation of the first segment is lost, but the rest reach the transmitter. Example: $|\mathcal{S}| = 7$, $w_s = 4$. The transmitter sends segments with sequence numbers $0, 1, 2, 3$. So the receiver expects the sequence numbers $4, 5, 6, 0$ next. The sender repeats the segment with sequence number 0, but this is misinterpreted by the receiver as a new segment.

d)* Justify which upper and lower limits for the receiver's receive window are reasonable for each of the two methods.

For Go-Back-N, a receive window of size $w_r = 1$ is sufficient in principle, since only the next expected segment is always accepted.
With Selective Repeat, on the other hand, the receive window must be at least as large as the transmit window, and of course must not be larger than about half of the sequence number space, i. e. $w_s \leq w_r \leq \left\lfloor \frac{|\mathcal{S}|}{2} \right\rfloor$.Otherwise, the receiver discards u. U. segments that do not arrive in the correct order and do not fall into the same due to the too small size of the receiving window.

e)* For a practical implementation, the receiver needs a receive buffer. How large should this be chosen for each of the two procedures?

Regardless of the method, the receive buffer should always be the size of the maximum allowed send window.
With Selective Repeat this makes sense, because here segments actually have to be buffered on the transport layer until missing segments have arrived.
With Go-Back-N, on the other hand, one could argue that segments have to arrive in the correct order anyway, and therefore they can be forwarded to higher layers immediately. This is only true to a limited extent, because the processing speed of the receiver might not be sufficient to forward incoming segments fast enough.
Regardless of the (somewhat philosophical) question of how large buffers should be chosen for concrete implementations, you should be clear about the difference between the receive window and any receive buffer. For example, the size of the receive window could always correspond to the remaining free memory in the receive buffer while the transmit window is constrained upward by the receive window.