Next Week on Communication Networks

Reliable Transport Project
Go-Back-N is a reliable transport protocol using a **sender window**, **ACKs** and **timeout**.
Go-Back-N is a reliable transport protocol using a sender window, ACKs and \textbf{timeout}.

\begin{center}
0 1 2 3 $\boxed{4\ 5\ 6\ 7}$ 8 9 10 11 \ldots \\
\textbf{go back N segments and resend all} \\
\text{(window size)}
\end{center}
The receiver typically uses cumulative acknowledgements.

\[ \text{ACK } X \quad = \quad \text{X is the next expected segment} \]
\[ \quad \text{every segment up to } X \text{ is ACKed} \]
GBN waits for a timeout before segments are retransmitted.

- **Sent segments:** 0 2 3 4 5
- **Receiver behavior:** 0 - 2 3 4 5
- **Out-of-order segments are dropped by the receiver**
- **Sent ACKs:** 1 - 1 1 1 1 1

**Retransmission:**
GBN waits for a **timeout** before segments are retransmitted.

- **Sent segments:** 0 → 2 3 4 5
- **Receiver behavior:** 0 → 2 3 4 5
- **Sent ACKs:** 1 → 1 1 1 1 1
- **Retransmission:**
  - **Sender window:** 1 2 3 4 5
  - **Time:**
  - **Timeout:**

The diagram illustrates the sequence of events, showing the sender window and the receiver's behavior, including the acknowledgment of received segments and the retransmission process due to a timeout.
Let’s see how it works in practice visually.
Retransmitting the whole window is wasteful.

*How can we do better?*
Selective Repeat does not wait for timeout and can *increase performance*.

<table>
<thead>
<tr>
<th>Sent segments:</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Receiver behavior:</td>
<td>0</td>
<td>–</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Sent ACKs:</td>
<td>1</td>
<td>–</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Retransmission:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Out-of-order segments are *buffered at the receiver*.
Selective Repeat does not wait for timeout and can increase performance.

Sent segments: 0 ↪ 2 3 4 5

Receiver behavior: 0 – 2 3 4 5

Sent ACKs: 1 – 1 1 1 1

Retransmission: No timeout (yet). **When** should we retransmit?

Out-of-order segments are **buffered at the receiver**
Selective Repeat does not wait for timeout and can increase performance.

- **Sent segments:**
  - 0 → 2 3 4 5

- **Receiver behavior:**
  - 0 → 2 3 4 5
  - Out-of-order segments are buffered at the receiver

- **Sent ACKs:**
  - 1 → 1 1 1 1 1

- **Retransmission:**
  - No timeout (yet). **When** should we retransmit?
    - Full ACK info (complex)
    - Duplicate ACKs (simple)
Fast Retransmit is based on duplicate ACKs.

- **Sent segments:** 0 2 3 4 5
- **Receiver behavior:** 0 2 3 4 5
- **Sent ACKs:** 1 1 1 1 1
- **Retransmission:** 1

Out-of-order segments are buffered at the receiver.

3 duplicate ACKs
Fast Retransmit is based on duplicate ACKs

Sent segments: $0 \rightarrow 2 \ 3 \ 4 \ 5 \ 1$

Receiver behavior: $0 \ - \ 2 \ 3 \ 4 \ 5$

Sent ACKs: $1 \ - \ 1 \ 1 \ 1 \ 1 \ 1$

Retransmission: $1$

3 duplicate ACKs

receiver delivers buffered segments
This Weeks Exercise

GBN and DNS
The first part of the exercise is all about Go-Back-N.

First

Warm-up questions (8.1) prepare you for the following question (8.2).
Discuss with your classmates!

Second

8.2 is an exam question from 2017.
Read the instructions carefully!
The last three questions encourage you to have a look at DNS responses.

8.3 Hierarchy

DNS is organized into **hierarchical zones**.

*How are they connected?*

8.4 Caching

The load on **authoritative DNS** servers is reduced by caching on **local DNS** servers.

*How is this cache managed?*

8.5 One Query, Many Responses

DNS can **return multiple IP addresses**.

*Why is this useful?*
The last three questions encourage you to have a look at DNS responses.

Unix

dig ethz.ch

Windows
	nslookup ethz.ch
	nslookup -debug ethz.ch

provides more information