Transport Layer

- Arten von Diensten
  De-/Multiplexing der draüberliegenden Schichten
- Datenübertragung

→End2End Protokolle, stellt logische Verbindung zwischen 2 Geräten dar
Prinzipiell: TCP, UDP

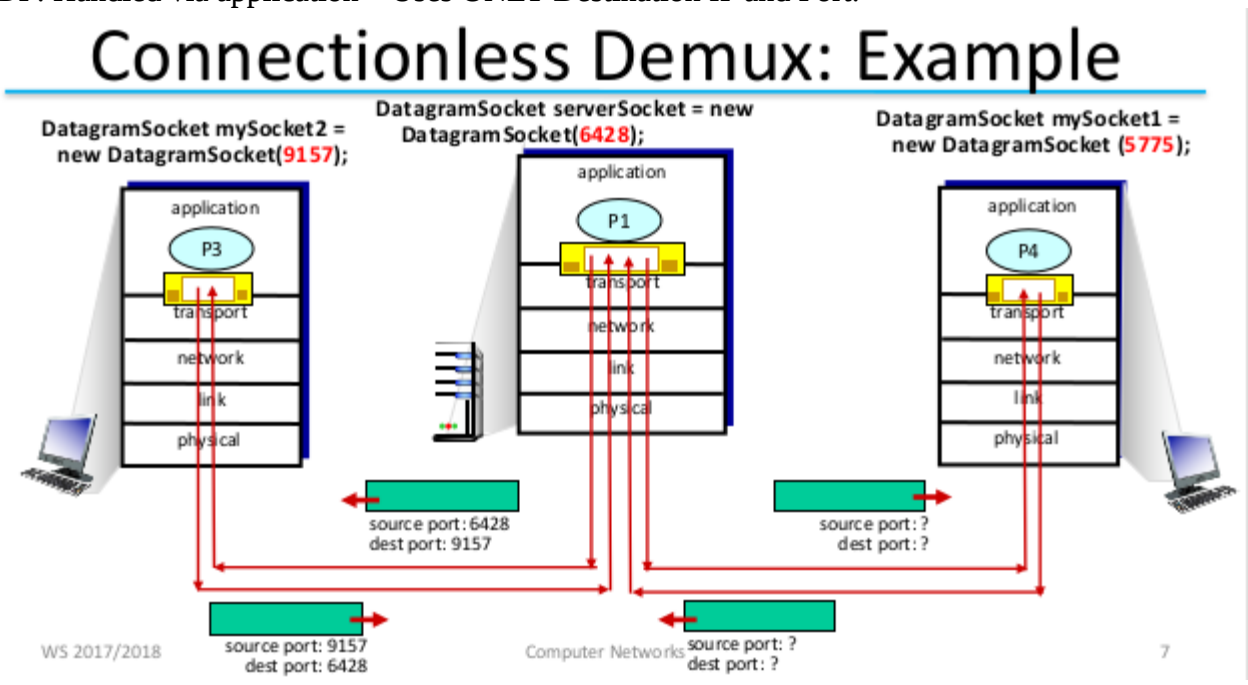
Zuverlässige Datenübertragung – TCP
- Verbindungsaufbau via Handshake
- Congestion Control bei Datenüberfluss
- Flow Control bei Missübertragungen o.ä.
- Impliziertess Connection Setup

Unzuverlässige Datenübertragung – UDP
Best-Effort IP: Darunterliegendes Netzwerk überträgt Nachricht möglichst schnell.

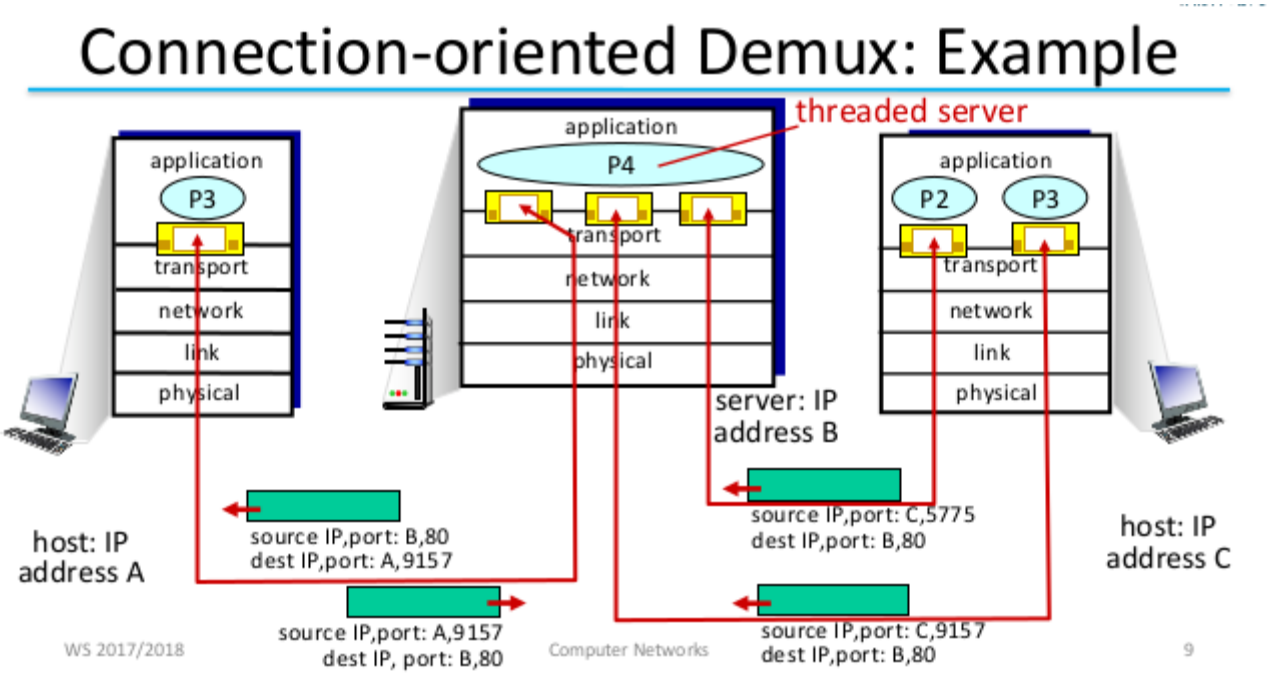
Services not available via TCP/UDP:
- Verzögerungen und Bandbreitengarantien =: Quality of Service “QOS”
- Kundenkontakt von Provider =: Quality of Experience “QOE”

Multiplexing
- Sender/Server: Behandelt Sockets und fügt Transport layer header hinzu für spätere Demultiplex
- Klient: Nutze Socket und Header zum Abfragen/Erhalten von Daten

Demultiplexing:
  source port # dest port #
    Other header Fields
    Application Data
→ Jeder Header beinhaltet source & destination Port und IP Addresse.
TCP: Jeder Socket nutzt Subkanal zwischen 2 Knoten die durch (src IP, src Port, Dest IP, dest Port) kommunizieren.
UDP: Handled via application – Uses ONLY Destination IP and Port.



* Flippen von SRC port & Dest Port <=> Rückführbarkeit von UDP-Packs



UDP [RFC 768]

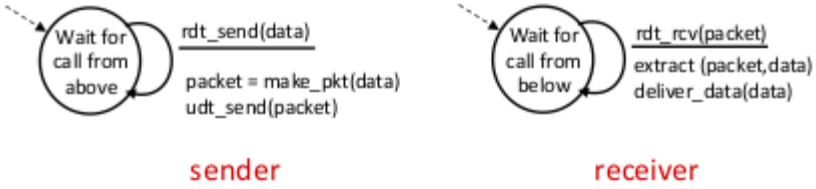
0	16	32 [Bit]
SRC Port	Dest Port	
length	Checksum	
Application data		

- No connection establishment
- No connection state at sender or receiver.
- Minimum header size
- No flow & congestion control ↔ Dataverlust möglich.
- Reference: QUIC

- Segment size: 8 byte to 2^16 bytes - Limited by IP (Darunterliegende Schicht) and NIC's MTU
- Protokollnr.: 17

Reliable Data Transfer

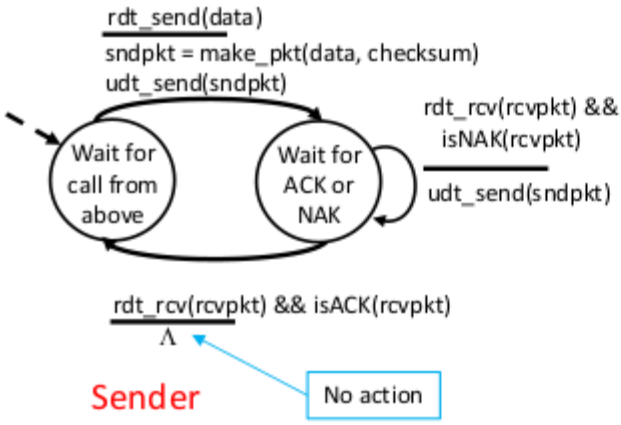
→ Notwendigkeit von Kontrolinfo, Steuerinfo
- Es gibt keine Bit-Fehler
- Es gibt kein Packetverlust
- Sender sendet Daten, Empfänger erhält Daten

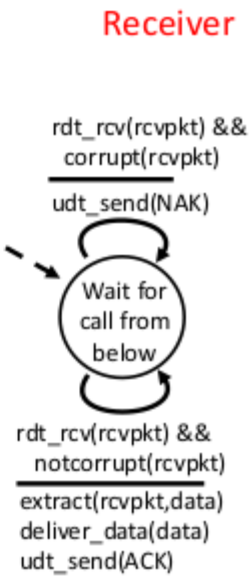


→ Channel with Bit Errors:
- Checksumme zum Entdecken von Bitfehlern
- ACK-Flag: Received Packet was ok

- NAK-Flag: Received Packet was not ok
    → Mit Retransmission des Packets

=> Empfänger sendet Feedback zum Sender.





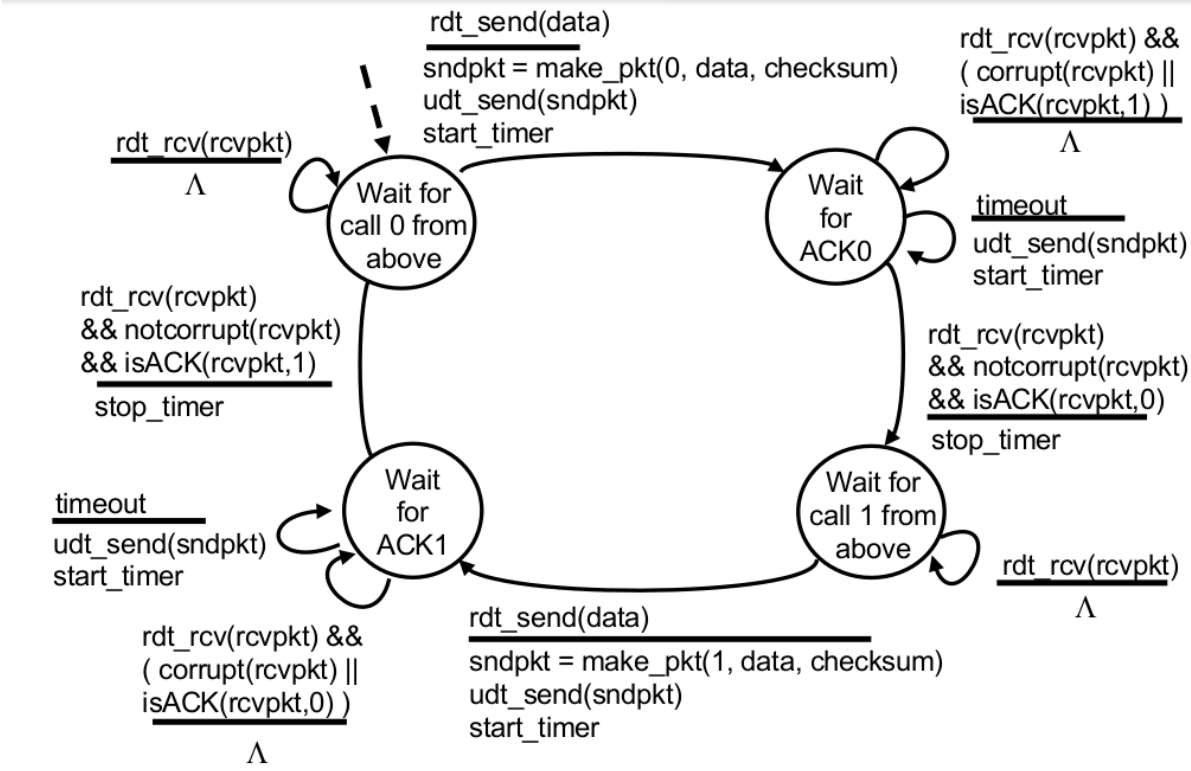
→ Fatal Flaw:

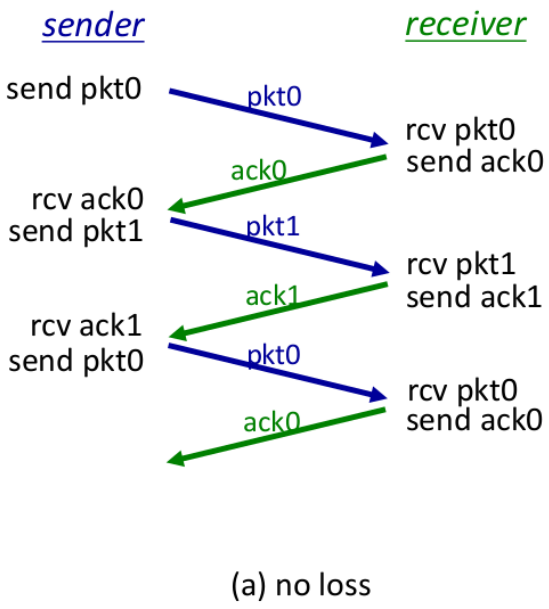
- ACK/NAK Corrupted ↔ Sender fügt Sequenznummer hinzu.
    Wenn ein Empfänger ein Duplikates Packet enthält, so verwirft er dieses.
- “Stop and Wait”: Sender sendet Packet und wartet auf Response

Reliable Transfer Protokoll v3.:

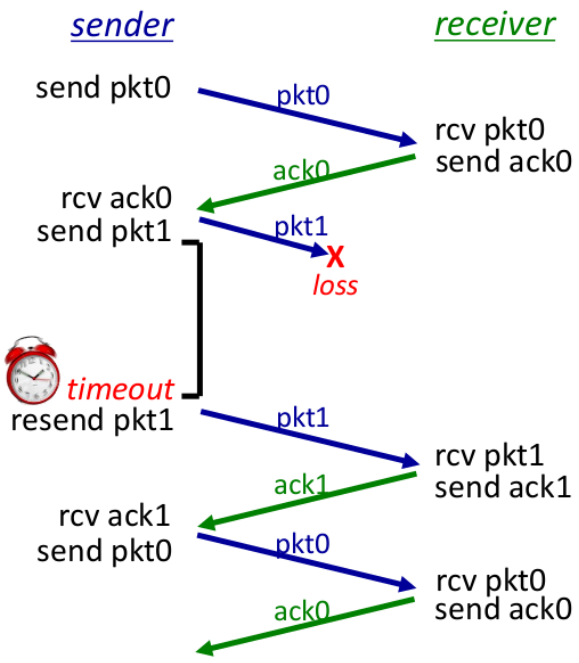
→ Checksumme
→ Sequenznummer
→ ACK/NAK

→ Retransmission: Sender wartet einige Sekunden auf das ACK/NAK. Wenn keine Response kommt, so retransmit Packet.
    → Duplikat-handling via Sequenznummer

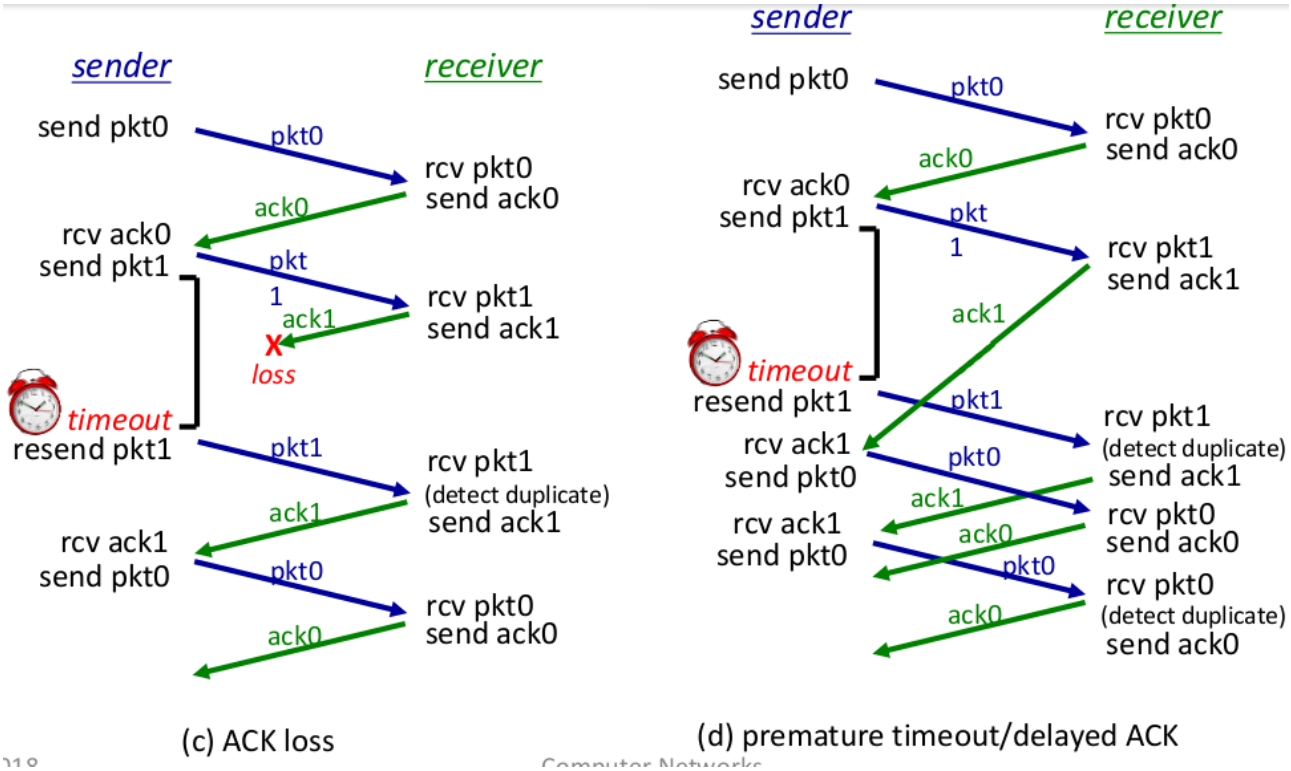




(a) no loss



(b) packet loss



Performance von RDT:

E.g.: 1 Gbps link, 15 ms propagation delay, 8000 bit packet:

$$D_{trans} = \frac{L}{R} = \frac{8000\ bits}{10^9\ bits/sec} = 8\ microsecs$$

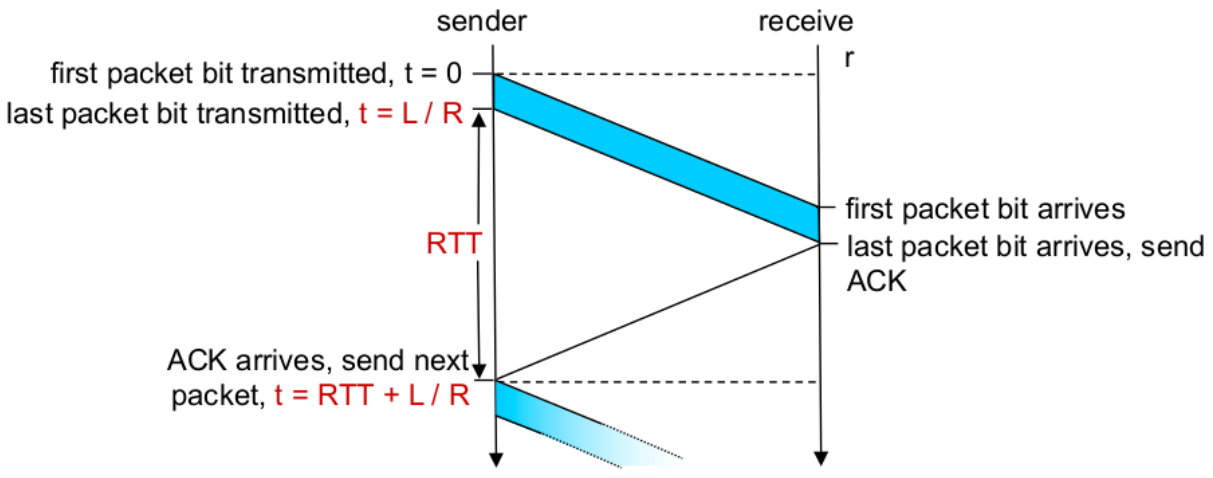
R ... transmission rate [bits/s]
L ... packet size [bits]
D ... duration [s]
RTT ... round-trip-time [s]
U ... utilization

U_{sender}: *utilization* – fraction of time sender busy sending

$$U_{sender} = \frac{L/R}{RTT + L/R} = \frac{.008}{30.008} = 0.00027$$

If RTT=30 msec, 1KB pkt every 30 msec: 33kB/sec throughput over 1 Gbps link

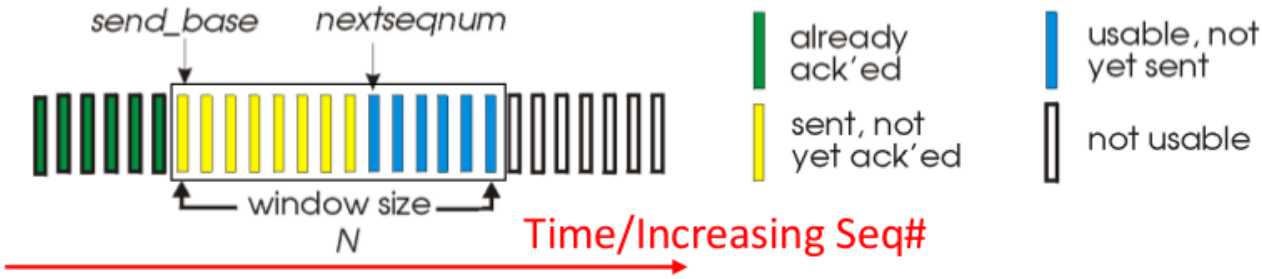
.. mit Graphen:



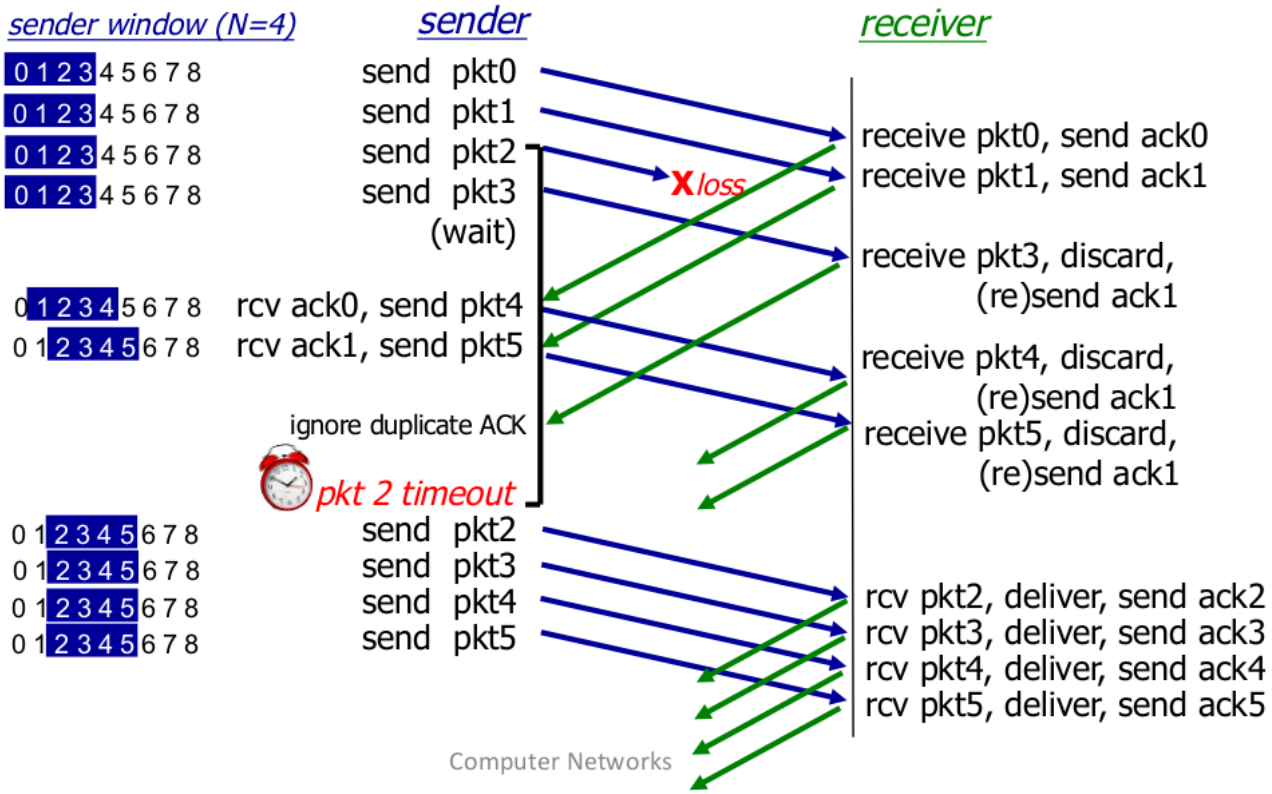
$$U_{sender} = \frac{L/R}{RTT + L/R} = \frac{.008}{30.008} = 0.00027$$

→ Pipeline Protokolle
á go-Back-N oder selective Repeat
Vgl.: http://www.ccs-labs.org/teaching/rn/animations/gbn_sr/

→ Go-Back-N:
Sender kann N Packete durchschicken und Sender schickt kumulative ACKs.
- Bei Datenverlust, so erfolgt keine Bestätigung. Lediglich diese, bis zum Loch werden bestätigt.
Sender:

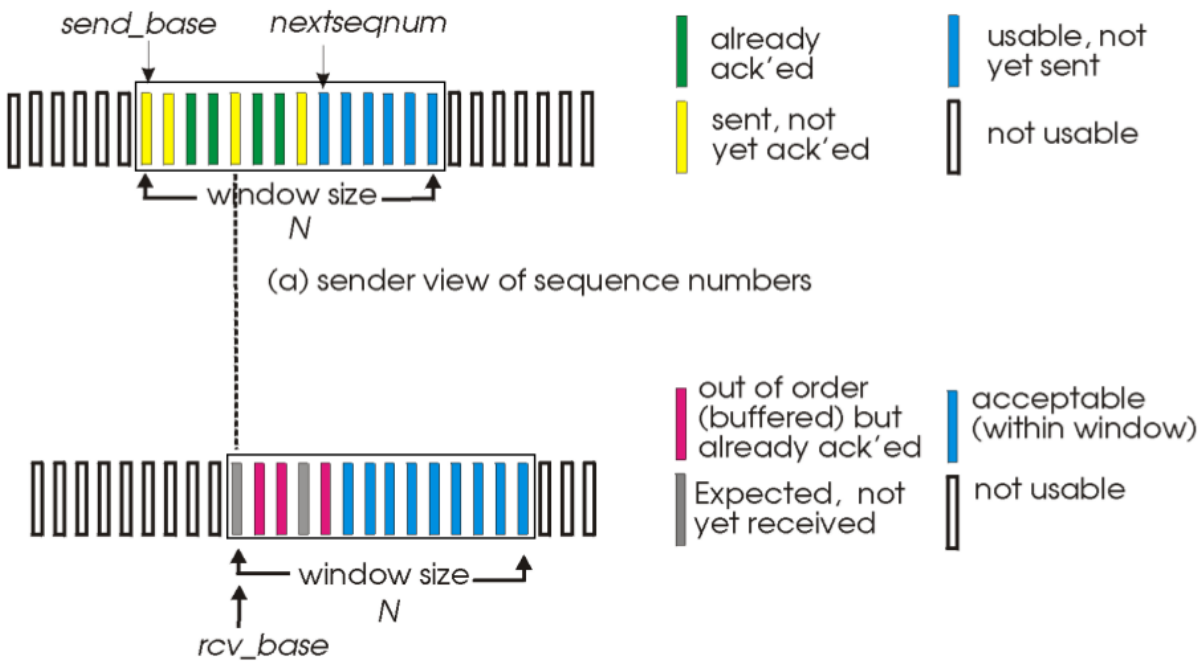


* Bei Timeout werden die Gelben packete wieder gesendet.
Empfänger:
Bei Packetverlust werden alle weiteren Packete verworfen und der Rest wird ACK'd



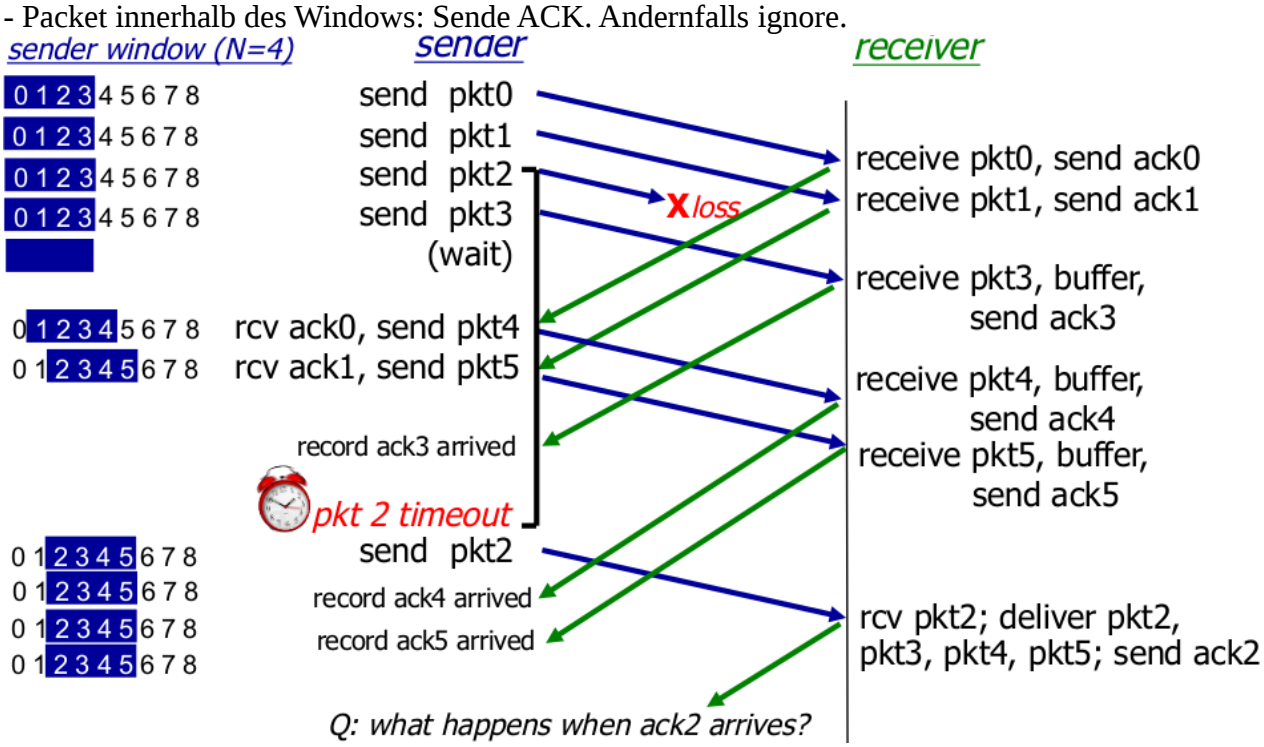
* Es gibt keinen Empfangsbuffer

→ Selective Repeat:
Sender sendet ebenso N Packete, wobei jedes Packet einzeln bestätigt wird.
Jedes Packet bekommt Timer



Sender:
- Bei Seq#: Sende Packet
- Timeout: Re-Send Packet
- ACK: Mark packet n as received
- If letztes Packet mit kleinster ACK bestätigt wird, so wird Window um 1 nach vorne erhöht.

Empfänger:
- Bei Empfang: Sende ACK
- Bei Lücke: Packet n in Buffer
- Packet innerhalb des Windows: Sende ACK. Andernfalls ignore.



→ Ack2: Senderseitig geht das window weiter bis auf 6.

Problem bei Selective Repeat:

Bei Verlust der ACK gibt es schwerwiegende Folgefehler: