Systeme II

Albert-Ludwigs-Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Christian Schindelhauer

Sommersemester 2006

17. Vorlesung

05.07.2006

schindel@informatik.uni-freiburg.de

Dienste der Transport-Schicht

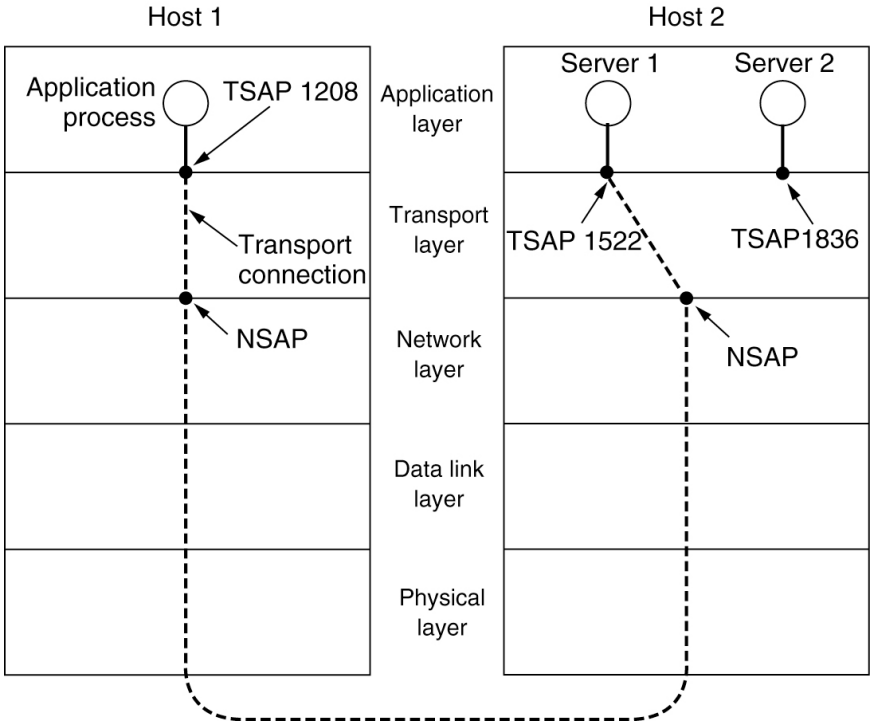
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Verbindungslos oder Verbindungsorientert**
  - Beachte: Sitzungsschicht im ISO/OSI-Protokoll
- **Verlässlich oder Unverlässlich**
  - Best effort oder Quality of Service
  - Fehlerkontrolle
- **Mit oder ohne Congestion Control**
- **Möglichkeit verschiedener Punkt-zu-Punktverbindungen**
  - Stichwort: Demultiplexen
- **Interationsmodelle**
  - Byte-Strom, Nachrichten, „Remote Procedure Call“

Multiplex in der Transportschicht

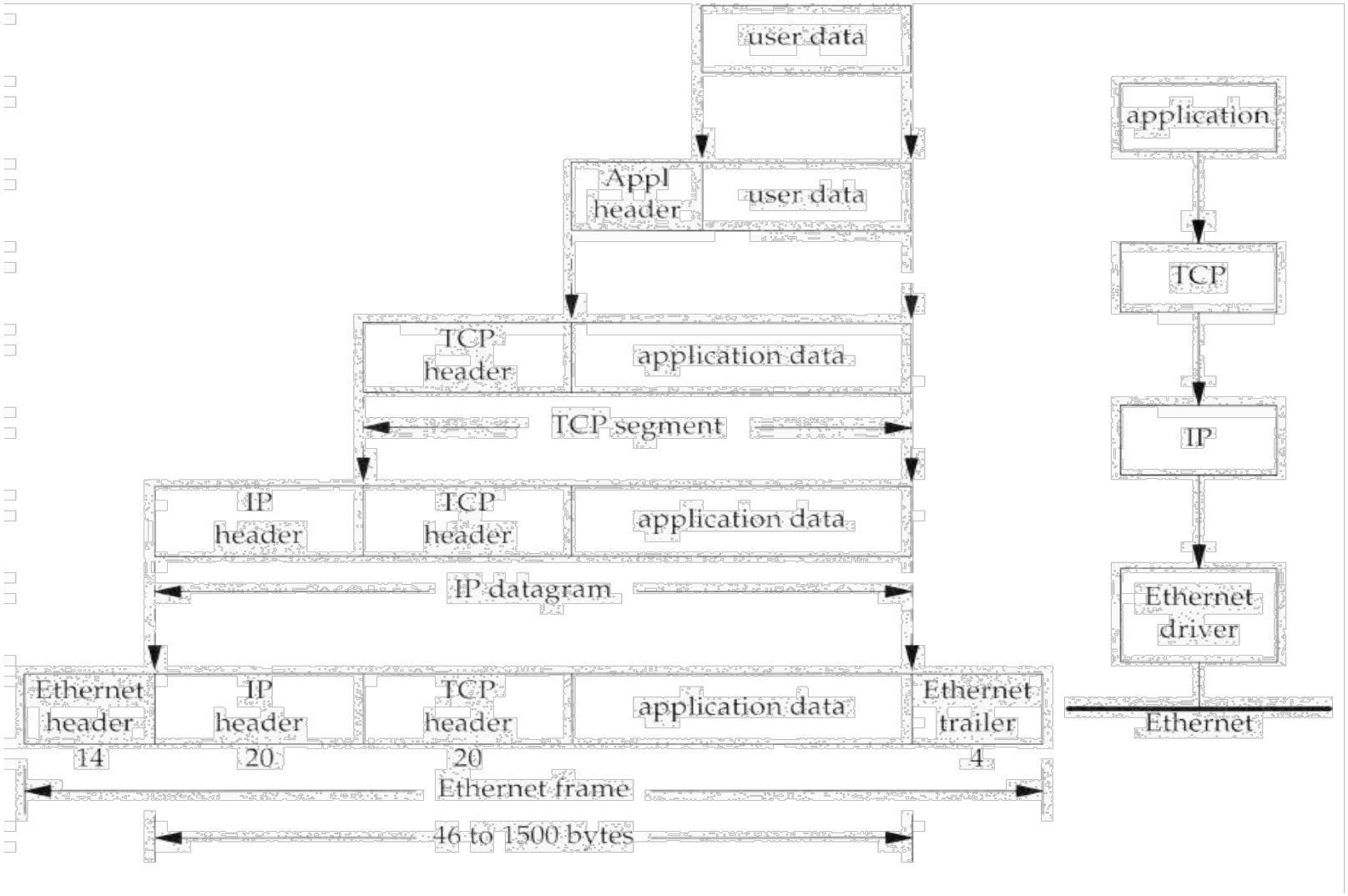
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Die Netzwerkschicht leitet Daten an die Transportschicht unkontrolliert weiter**
- **Die Transportschicht muss sie den verschiedenen Anwendungen zuordnen:**
  - z.B. Web, Mail, FTP, ssh, ...
  - In TCP/UDP durch Port-Nummern
  - z.B. Port 80 für Web-Server



Datenkapselung

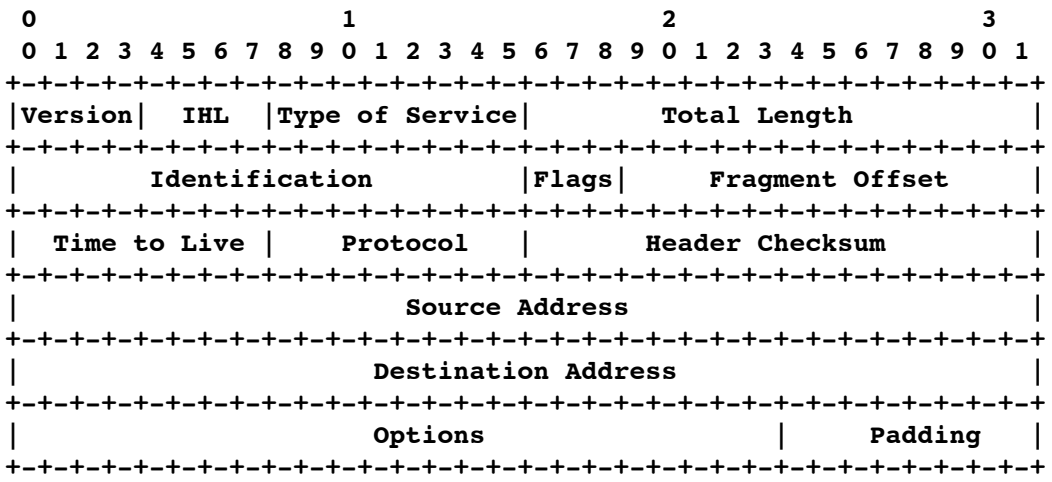
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer



IP-Header (RFC 791)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

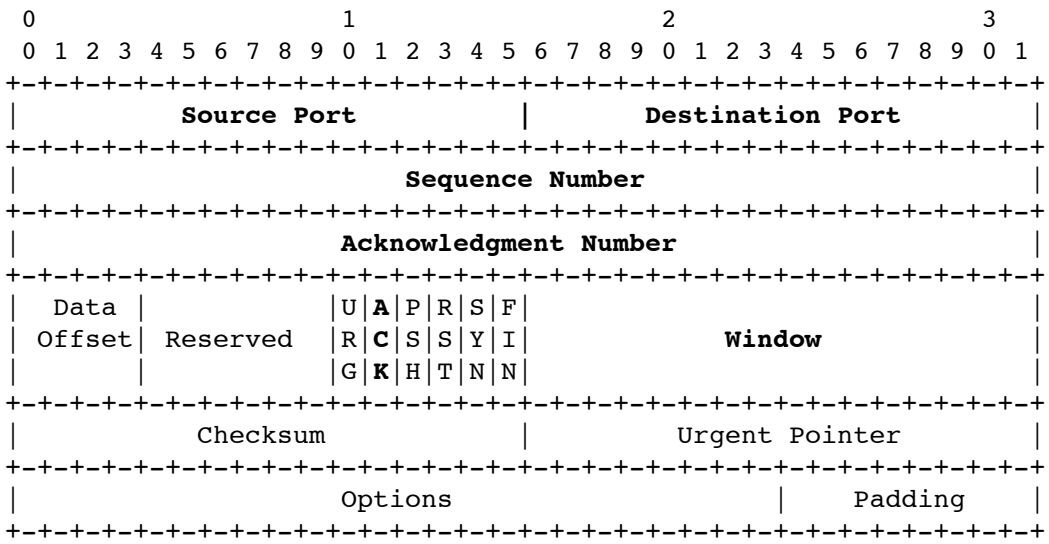
- **Version: 4 = IPv4**
- **IHL: Headerlänge**
  - in 32 Bit-Wörter (>5)
- **Type of Service**
  - Optimiere delay, throughput, reliability, monetary cost
- **Checksum (nur für IP-Header)**
- **Source and destination IP-address**
- **Protocol, identifiziert passendes Protokoll**
  - Z.B. TCP, UDP, ICMP, IGMP
- **Time to Live:**
  - maximale Anzahl Hops



TCP-Header

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Sequenznummer**
  - Nummer des ersten Bytes im Segment
  - Jedes Datenbyte ist nummeriert modulo 256
- **Bestätigungsnummer**
  - Aktiviert durch ACK-Flag
  - Nummer des nächsten noch nicht bearbeiteten Datenbytes
    - = letzte Sequenznummer + letzte Datenmenge
- **Sonstiges:**
  - Port-Adressen
    - Für parallele TCP-Verbindungen
    - Ziel-Port-Nr.
    - Absender-Port
  - Headerlänge
    - data offset
  - Prüfsumme
    - Für Header und Daten



Transportschicht (transport layer)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **TCP (transmission control protocol)**
  - Erzeugt zuverlässigen Datenfluß zwischen zwei Rechnern
  - Unterteilt Datenströme aus Anwendungsschicht in Pakete
  - Gegenseite schickt Empfangsbestätigungen (Acknowledgments)
- **UDP (user datagram protocol)**
  - Einfacher unzuverlässiger Dienst zum Versand von einzelnen Päckchen
  - Wandelt Eingabe in ein Datagramm um
  - Anwendungsschicht bestimmt Paketgröße
- **Versand durch Netzwerkschicht**
- **Kein Routing: End-to-End-Protokolle**

TCP (I)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **TCP ist ein verbindungsorientierter, zuverlässiger Dienst für bidirektionale Byteströme**

- **TCP ist verbindungsorientiert**
  - Zwei Parteien identifiziert durch Socket: IP-Adresse und Port (TCP-Verbindung eindeutig identifiziert durch Socketpaar)
  - Kein Broadcast oder Multicast
  - Verbindungsaufbau und Ende notwendig
  - Solange Verbindung nicht (ordentlich) beendet, ist Verbindung noch aktiv

TCP (II)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **TCP ist ein verbindungsorientierter, zuverlässiger Dienst für bidirektionale Byteströme**

- **TCP ist zuverlässig**
  - Jedes Datenpaket wird bestätigt (acknowledgment)
  - Erneutes Senden von unbestätigten Datenpakete
  - Checksum für TCP-Header und Daten
  - TCP nummeriert Pakete und sortiert beim Empfänger
  - Löscht duplizierte Pakete

TCP (III)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **TCP ist ein verbindungsorientierter, zuverlässiger Dienst für bidirektionale Byteströme**

- **TCP ist ein Dienst für bidirektionale Byteströme**
  - Daten sind zwei gegenläufige Folgen aus einzelnen Bytes (=8 Bits)
  - Inhalt wird nicht interpretiert
  - Zeitverhalten der Datenfolgen kann verändert werden
  - Versucht zeitnahe Auslieferung jedes einzelnen Datenbytes
  - Versucht Übertragungsmedium effizient zu nutzen
    - = wenig Pakete

TCP-Verbindungsaufbau

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

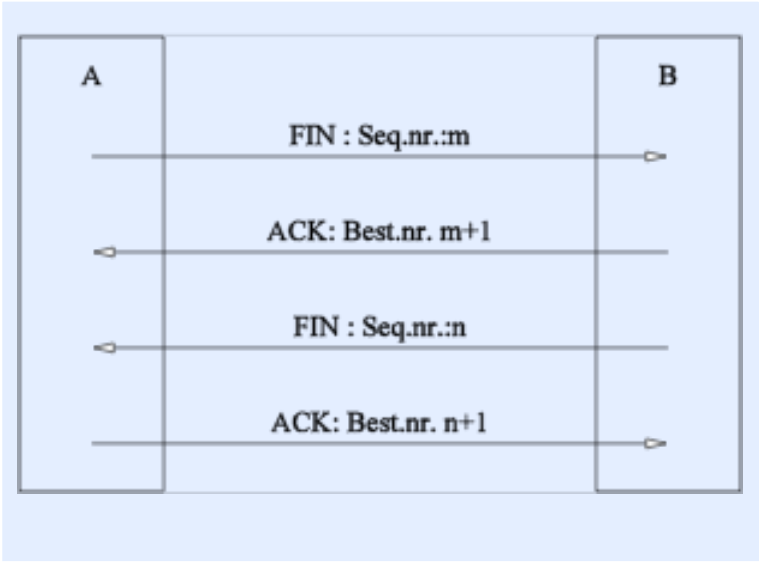
- **In der Regel Client-Server-Verbindungen**
  - Dann Aufbau mit drei TCP-Pakete (=Segmente)
  - Mit ersten SYN-Segment auch Übermittlung der MSS (maximum segment size)

TCP-Verbindungssende

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

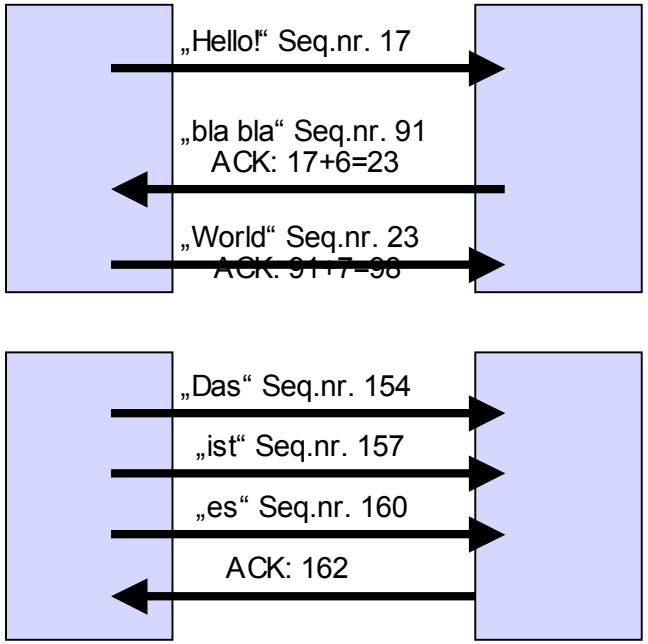
- **Half-Close**
  - –Sender kündigt Ende mit FIN-Segment an und wartet auf Bestätigung
  - –In Gegenrichtung kann weitergesendet werden

- **2 Half-Close beenden TCP-Verbindung**



Bestätigungen

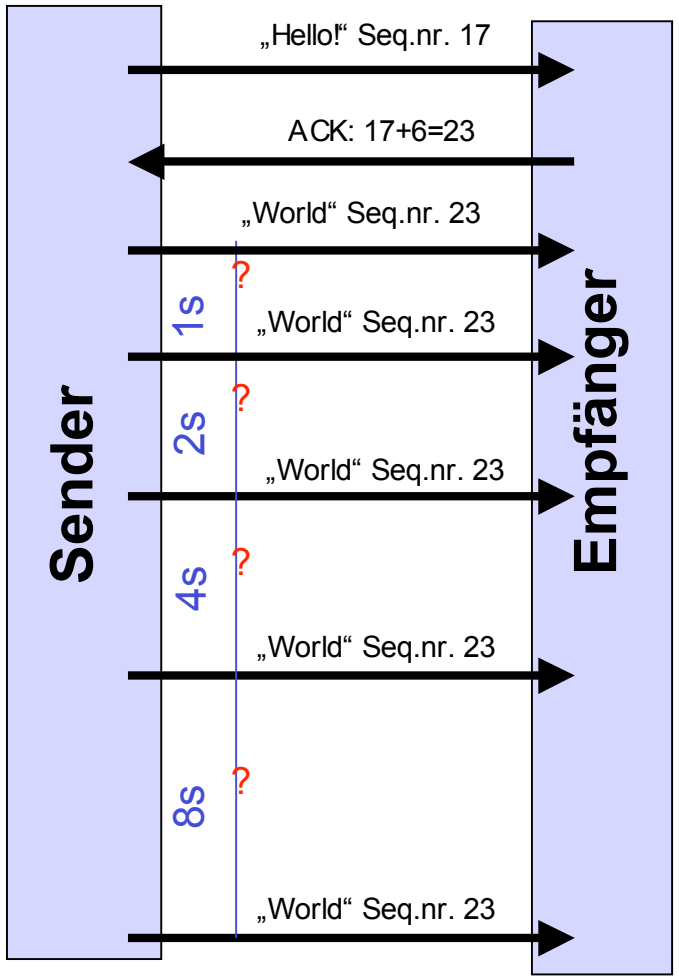
- **Huckepack-Technik**
  - Bestätigungen „reiten“ auf den Datenpaket der Gegenrichtung
- **Eine Bestätigungssegment kann viele Segmente bestätigen**
  - Liegen keine Daten an, werden Acks verzögert



Exponentielles Zurückweichen

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Retransmission Timout (RTO)**
  - regelt Zeitraum zwischen Senden von Datenduplikaten, falls Bestätigung ausbleibt
- **Wann wird ein TCP-Paket nicht bestätigt?**
  - Wenn die Bestätigung wesentlich länger benötigt, als die durchschnittliche Umlaufzeit (RTT/round trip time)
    - 1. Problem: Messung der RTT
    - 2. Problem: Bestätigung kommt, nur spät
  - Sender
    - Wartet Zeitraum gemäß RTO
    - Sendet Paket nochmal und setzt
    - RTO ← 2 RTO (bis RTO = 64 Sek.)
- **Neuberechnung von RTO, wenn Pakete bestätigt werden**



TCP - Algorithmus von Nagle

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Wie kann man sicherstellen,**
  - dass kleine Pakete zeitnah ausgeliefert werden
  - und bei vielen Daten große Pakete bevorzugt werden?

- **Algorithmus von Nagle:**
  - Kleine Pakete werden nicht versendet, solange Bestätigungen noch ausstehen.
    - Paket ist klein, wenn Datenlänge < MSS
  - Trifft die Bestätigung des zuvor gesendeten Pakets ein, so wird das nächste verschickt.
- **Beispiel:**
  - Telnet versus ftp
- **Eigenschaften**
  - Selbst-taktend: Schnelle Verbindung = viele kleine Pakete

Schätzung der Umlaufzeit (RTT/Round Trip Time)

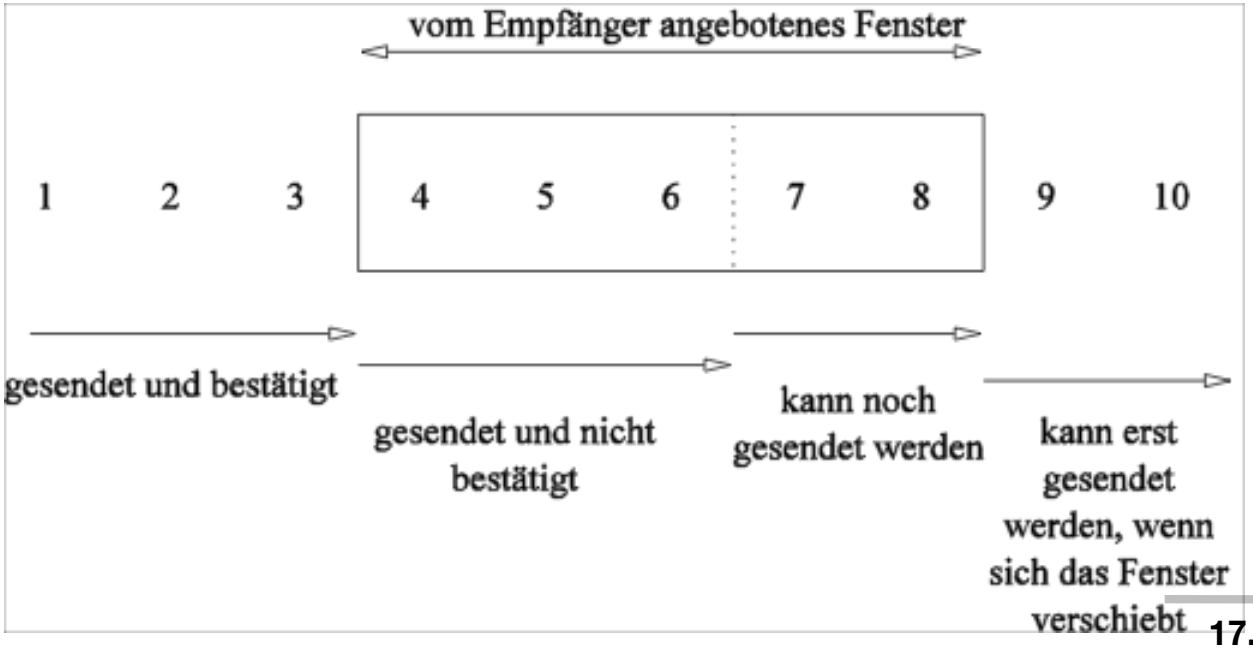
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **TCP-Paket gilt als nicht bestätigt, wenn Bestätigung „wesentlich“ länger dauert als RTO**
  - RTT nicht on-line berechenbar (nur rückblickend)
  - RTT schwankt stark
- **Daher: Retransmission Timeout Value aus großzügiger Schätzung:**
  - RFC 793: (M := letzte gemessene RTT)
    - $R \leftarrow \alpha R + (1- \alpha) M$, wobei $\alpha = 0{,}9$
    - $RTO \leftarrow \beta R$, wobei $\beta = 2$
  - Jacobson 88: Schätzung nicht robust genug, daher
    - $A \leftarrow A + g (M - A)$, wobei $g = 1/8$
    - $D \leftarrow D + h (|M - A| - D)$, wobei $h = 1/4$
    - $RTO \leftarrow A + 4D$
- **Aktualisierung nicht bei mehrfach versandten Pakete**

Gleitende Fenster (sliding windows)

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

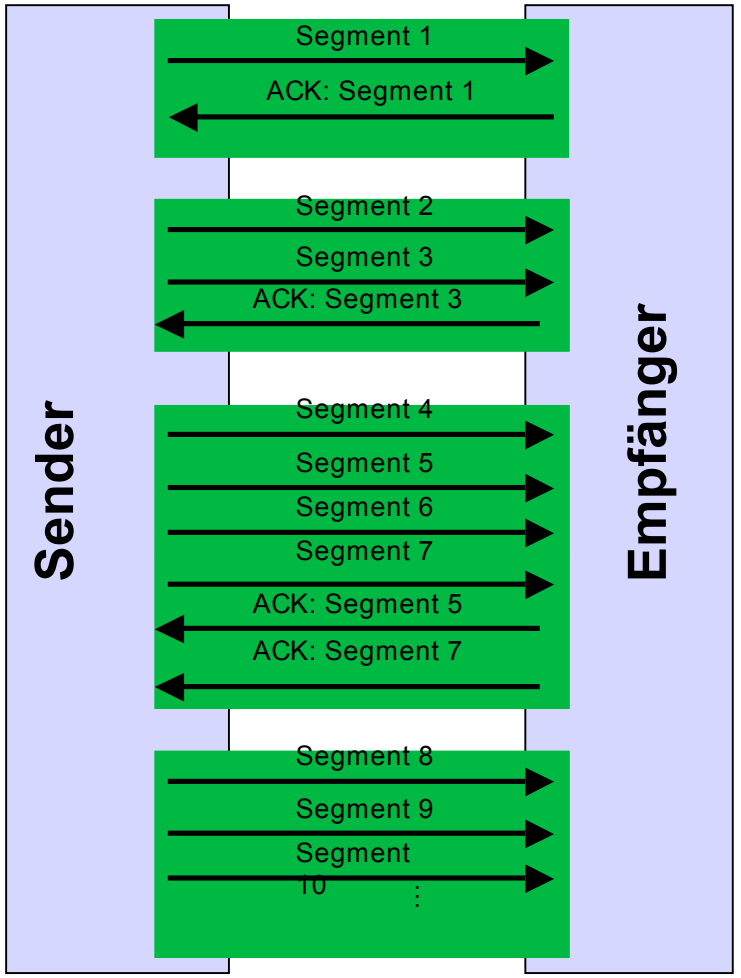
- **Datenratenanpassung durch Fenster**
  - Empfänger bestimmt Fenstergröße (wnd) im TCP-Header der ACK-Segmente
  - Ist Empfangspuffer des Empfängers voll, sendet er wnd=0
  - Andernfalls sendet Empfänger wnd>0
- **Sender beachtet:**
  - Anzahl unbestätigter gesender Daten ≤ Fenstergröße



Slow Start Congestion Fenster

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Sender darf vom Empfänger angebotene Fenstergröße nicht von Anfang wahrnehmen**
- **2. Fenster: Congestion-Fenster (cwnd/Congestion window)**
  - Von Sender gewählt (FSK)
  - Sendefenster: min {wnd,cwnd}
  - S: Segmentgröße
  - Am Anfang:
    - cwnd ← S
  - Für jede empfangene Bestätigung:
    - cwnd ← cwnd + S
  - Solange bis einmal Bestätigung ausbleibt
- **„Slow Start“ = Exponentielles Wachstum**



Ende der 17. Vorlesung

Albert-Ludwigs-Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Systeme II

Christian Schindelhauer

schindel@informatik.uni-freiburg.de