Rechnernetze & Verteilte Systeme

Ludwig-Maximilians-Universität München

Sommersemester 2019

Prof. Dr. D. Kranzlmüller

MNM TEAM

Fragestunde

Termin: 26.07.2019

Fragen bitte per Mail an rnvs-fragen@nm.ifi.lmu.de

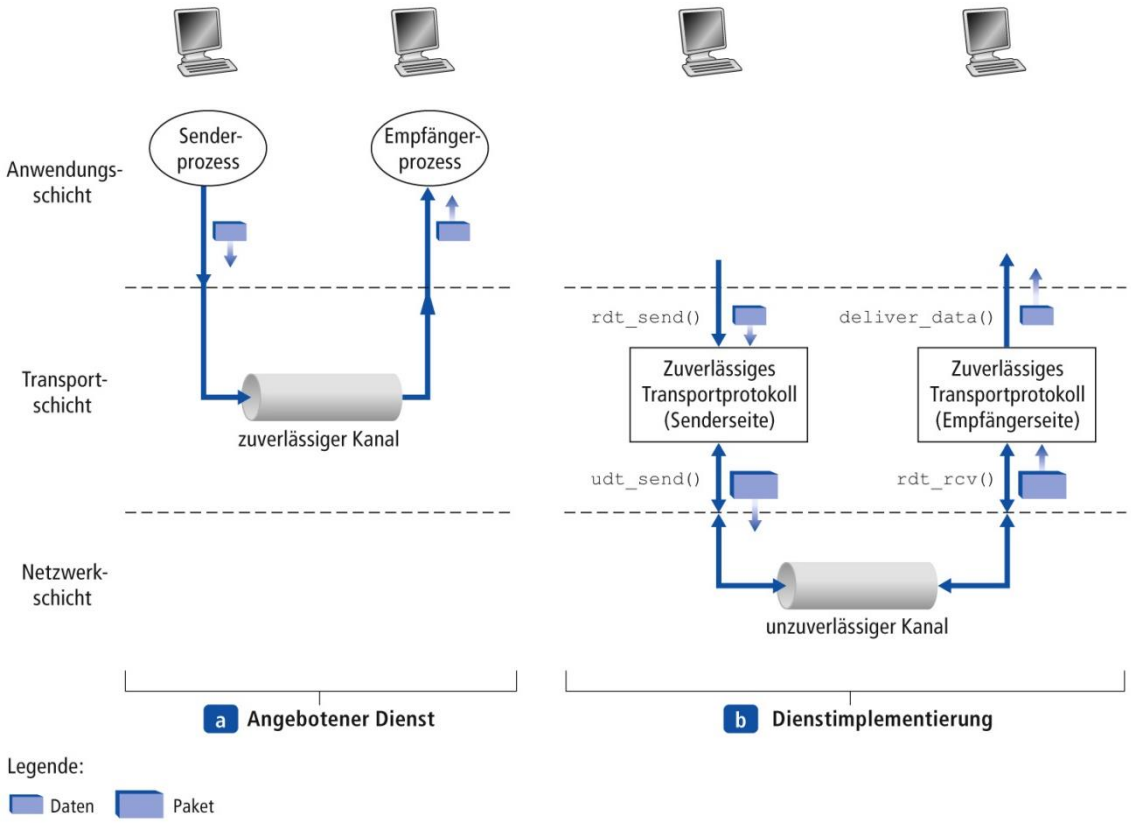
Kapitel 4: Transportschicht

(Engl.: Transport layer)

MNM TEAM

Wiederholung zu TCP

Konzeptionelle Sicht



Fenstertechnik (Sliding Window)

- Ziel: Optimierung des Durchsatzes (Bytes / Sekunde) bei verbindungsorientierten Protokollen.
- Zwei grundlegende Verfahren
  - Go-Back N:
  - Selective Repeat

Go-Back N

- Einfaches Pipelining

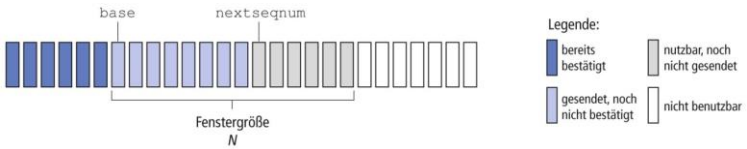


Abbildung 3.19: Sequenznummern aus Sicht des Senders bei Go-Back-N.

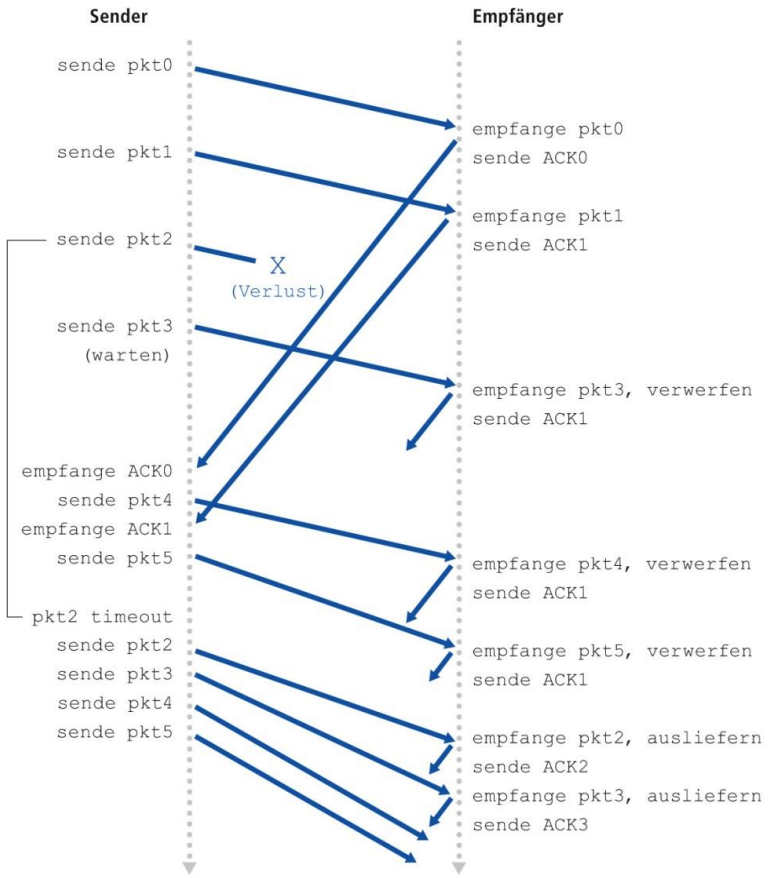


Abbildung 3.22: Arbeitsweise von Go-Back-N.

Selective Repeat (SR)

- Bereits bestätigte Segmente innerhalb des Sendefensters werden nicht erneut versandt.

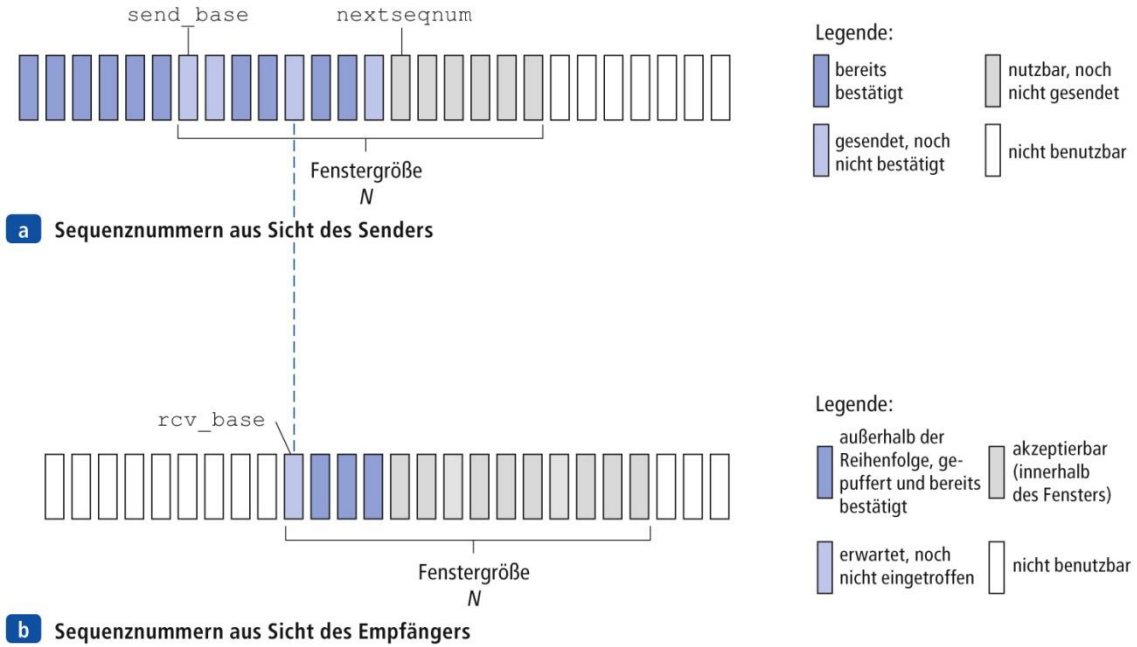


Abbildung 3.23: Sender- und Empfängersicht des Bereichs der Sequenznummern bei Selective Repeat (SR).

SR Ablauf

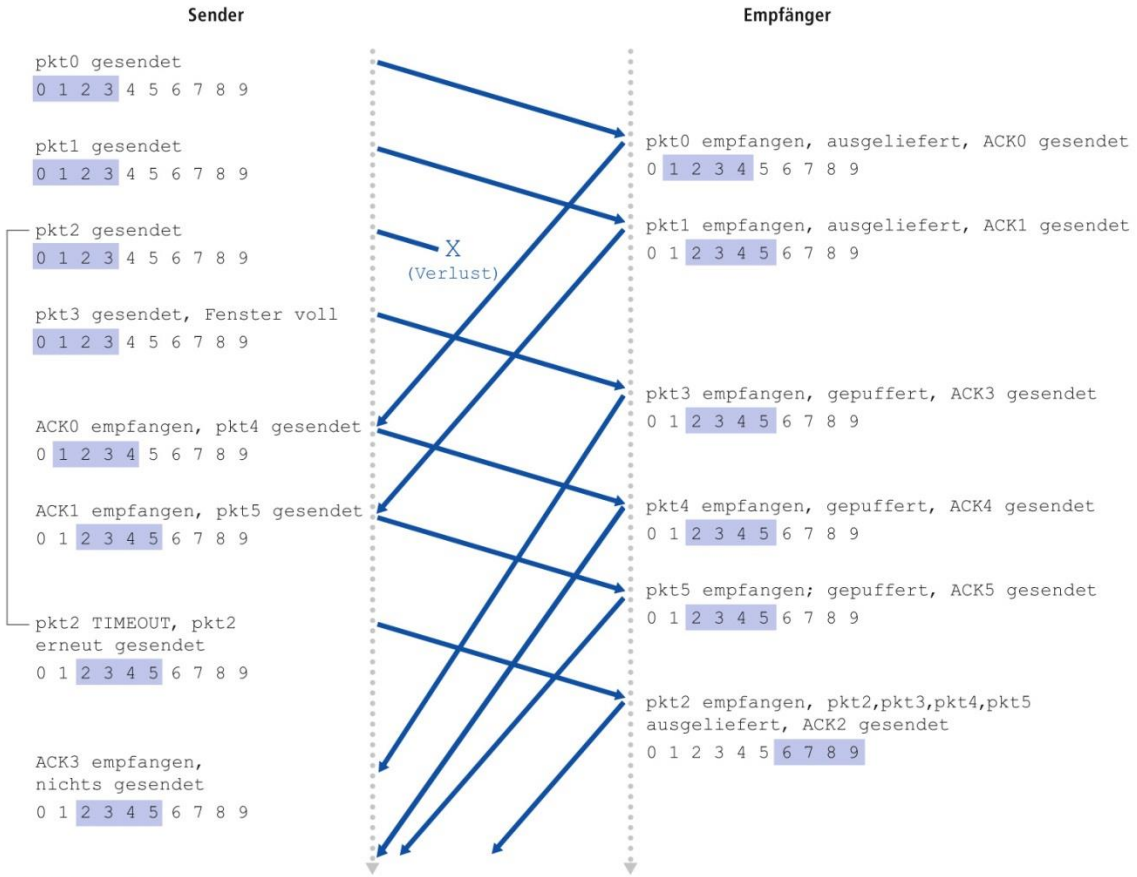


Abbildung 3.26: SR-Arbeitsweise.

SR: Empfängerdilemma

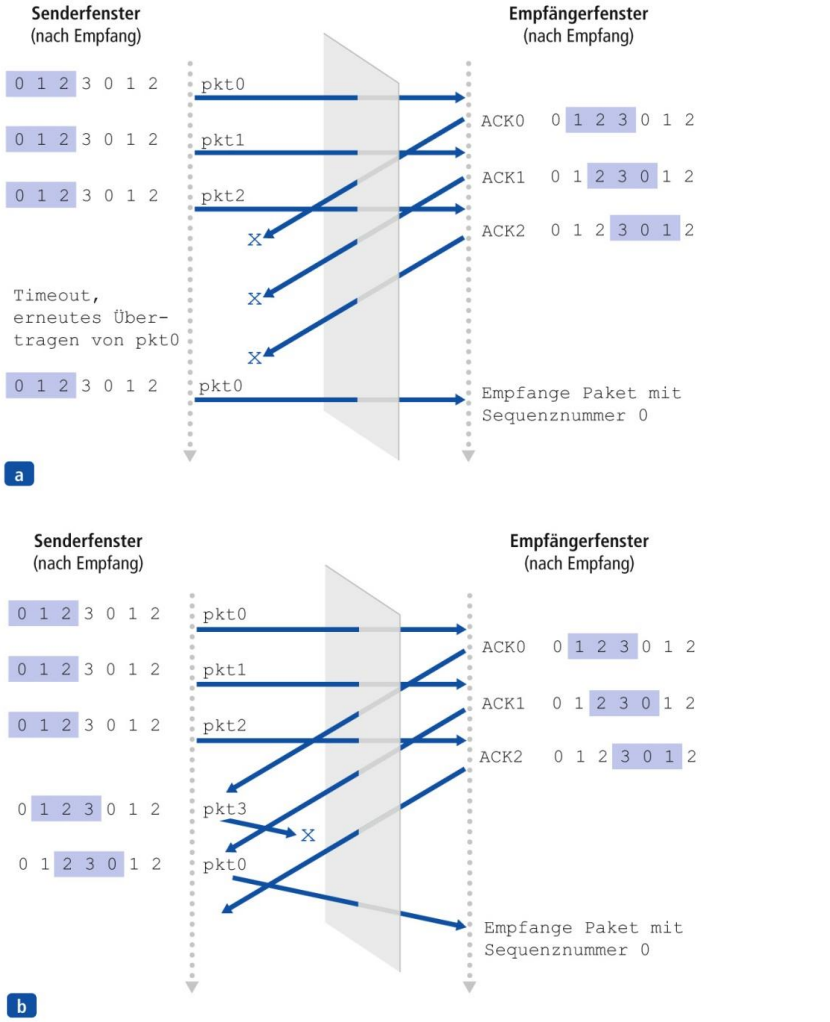


Abbildung 3.27: Dilemma des SR -Empfängers bei zu großen Fenstern: neues Paket oder Übertragungswiederholung?

Zusammenfassung relevanter Datenübertragungs-Mechanismen

- Timer: Wird verwendet, um ggf. verloren gegangene Segmente erneut zu übertragen.
- Sequenznummer: fortlaufende Nummerierung von Bytes in Segmenten.
- Acknowledgement (ACK): Quittungen über erfolgreich erhaltene Pakete.
- Pipelining, Fenstertechnik: maximale Anzahl unbestätigter Segmente auf dem Sendekanal.

Aufbau eines TCP Segments

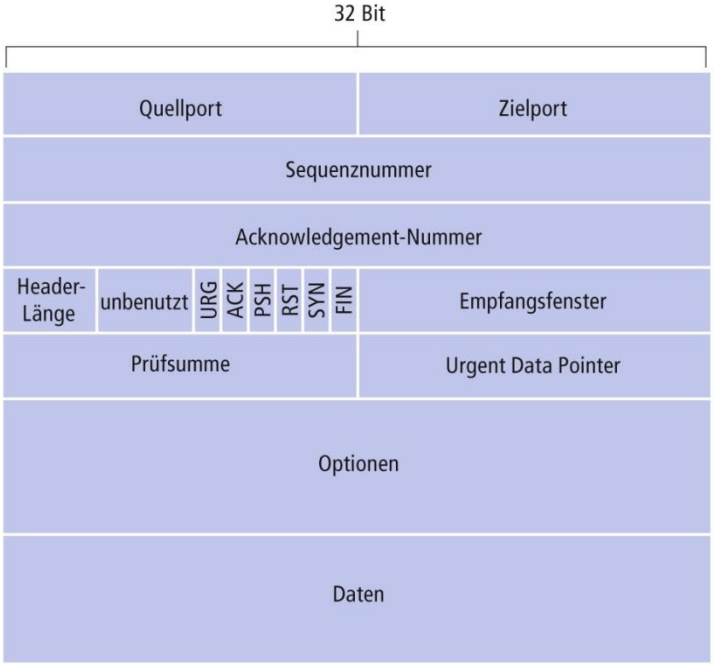


Abbildung 3.29: TCP-Segmentstruktur.

Zusammenspiel von Sequenz- und ACK-Nummern

- Beispiel: Host A verbindet sich via Telnet zu Host B

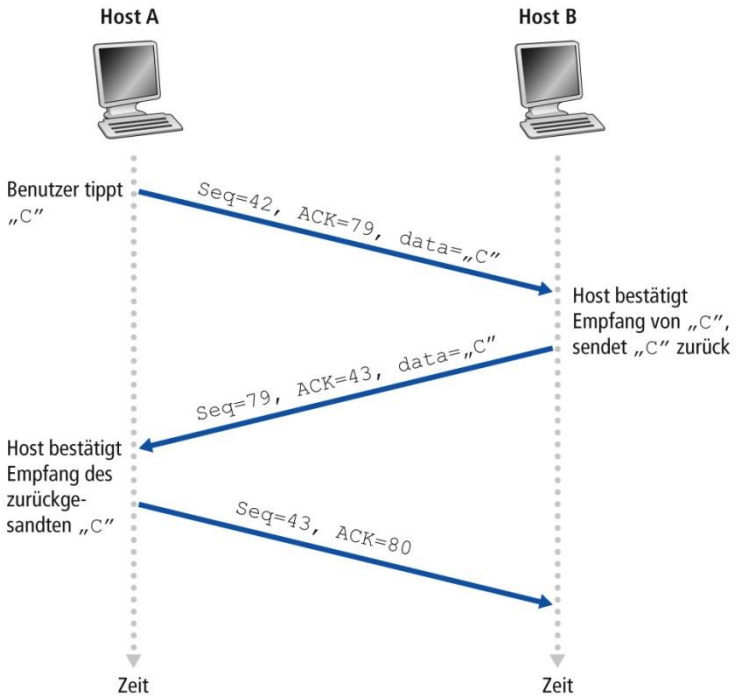


Abbildung 3.31: Sequenznummern und Acknowledgment-Nummern für eine einfache Telnet-Anwendung über TCP.

Kommunikationsablauf (1)

- Szenario: Erneuter Versand aufgrund von verloren gegangenem ACK

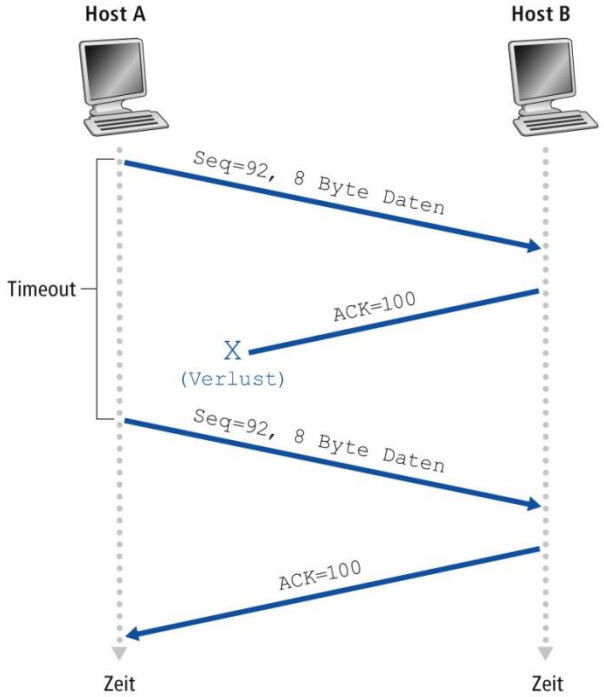


Abbildung 3.34: Erneute Übertragung aufgrund eines verloren gegangenen Acknowledgments.

Kommunikationsablauf (2)

- Szenario: ACK kommt zu spät zum Empfänger.
  - Timeout: Segment 92 wird erneut versendet
  - Segment 100 wird **nicht** erneut versendet, da das ACK innerhalb des nächsten Timeout Intervalls ankommt.

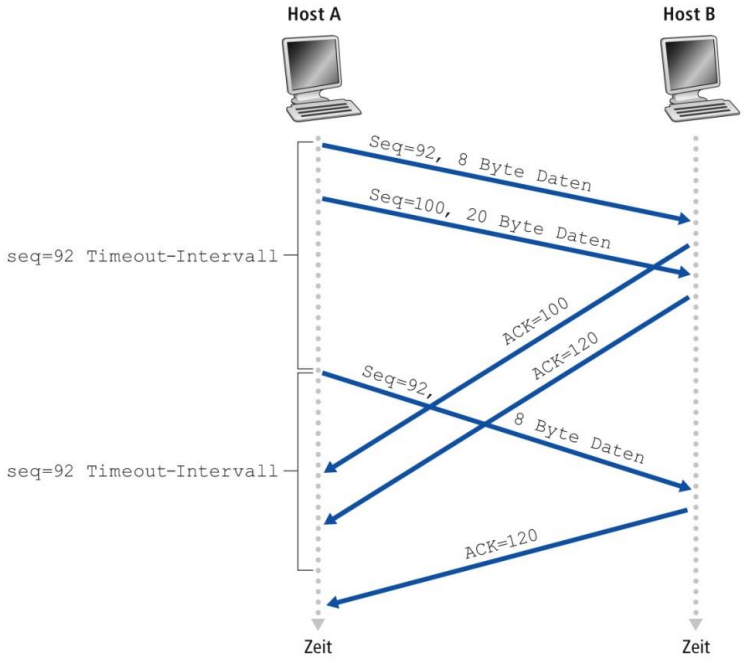


Abbildung 3.35: Segment 100 wird nicht erneut übertragen.

Kommunikationsablauf (3)

- Szenario: Kein erneutes übertragen aufgrund des kumulativen ACK.

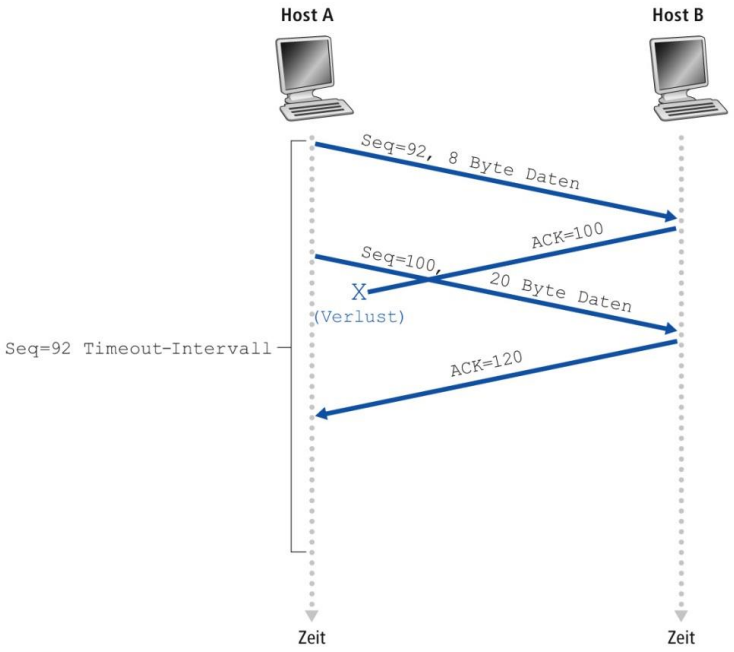


Abbildung 3.36: Das kumulative Acknowledgment verhindert die erneute Übertragung des ersten Segments.

Fast Retransmit

- Timeout können vergleichsweise lange sein
  - Regelmäßiges Anpassen durch Sampling und gewichteten Durchschnitt (siehe RFC 2988)
- Weiterer zuverlässiger Indikator: ACK-Duplikate.
- Optimierung bei TCP: Wenn zu einem ACK 3 weitere duplizierte ACKs ankommen (innerhalb des Timeout-Intervalls), wird das entsprechende Segment erneut übertragen.

Fast Retransmit: Beispiel

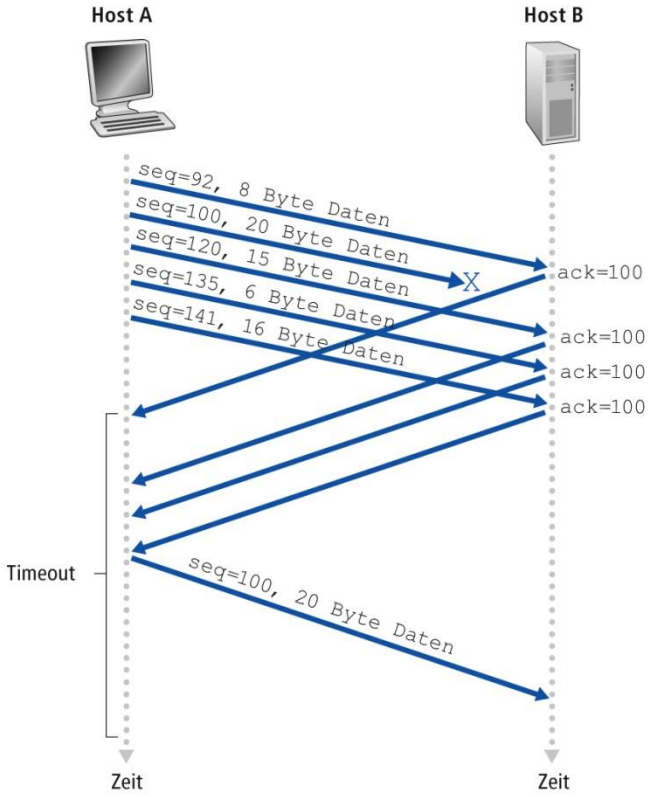


Abbildung 3.37: Fast Retransmit: erneute Übertragung des fehlenden Segments, bevor der Timer des Segments abläuft.

TCP Verbindungsaufbau

- Segmentaustausch beim 3-Wege Handschlag

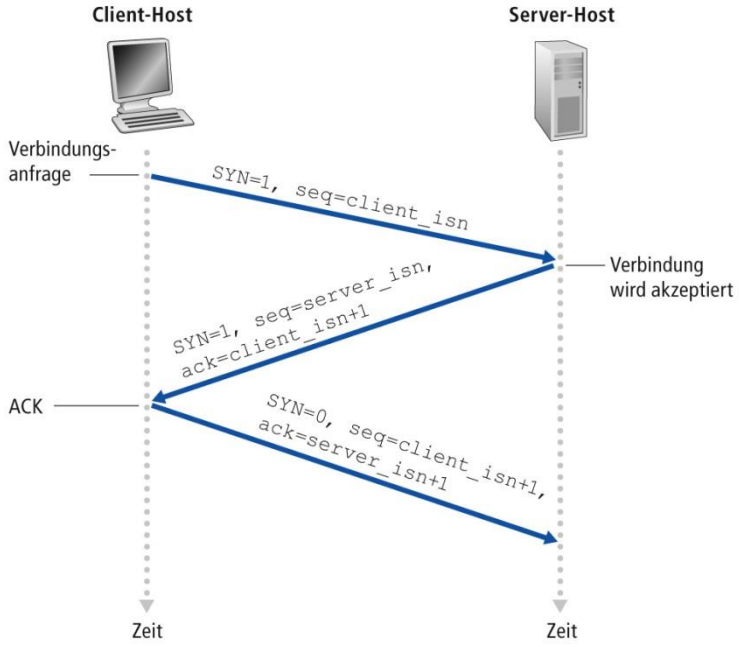


Abbildung 3.39: Segmentaustausch beim Drei-Wege-Handshake.

TCP Verbindungsabbau

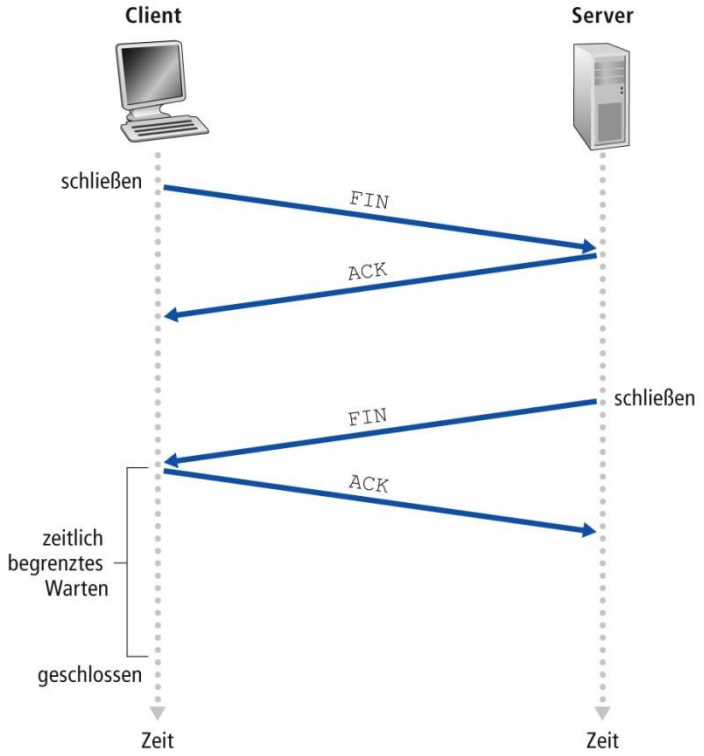


Abbildung 3.40: Schließen einer TCP-Verbindung.

TCP Staukontrolle

- **Überlastung der Netze**: Zu viele Transportinstanzen auf vielen Endsystemen speisen gleichzeitig zu viele Pakete ins Netz ein.

➔ Überlastung der Router auf der Vermittlungsschicht

- Wie kann auf Transportschicht festgestellt werden, dass eine Überlastung der Transitnetze besteht?
- Wie kann man darauf reagieren?

TCP Ansatz

- Ein verlorenes Segment (Timeout, ACK-Duplikate) impliziert eine „Stau“-Situation im Netz.

  → Senderate verringern.
- Ein angekommenes ACK impliziert einen guten Datenfluss.

  → Senderate erhöhen.
- Kontinuierliches Probing: Senderate kontinuierlich erhöhen bis ein Segment verloren geht. Anschließende Reduktion.

Relevante Parameter

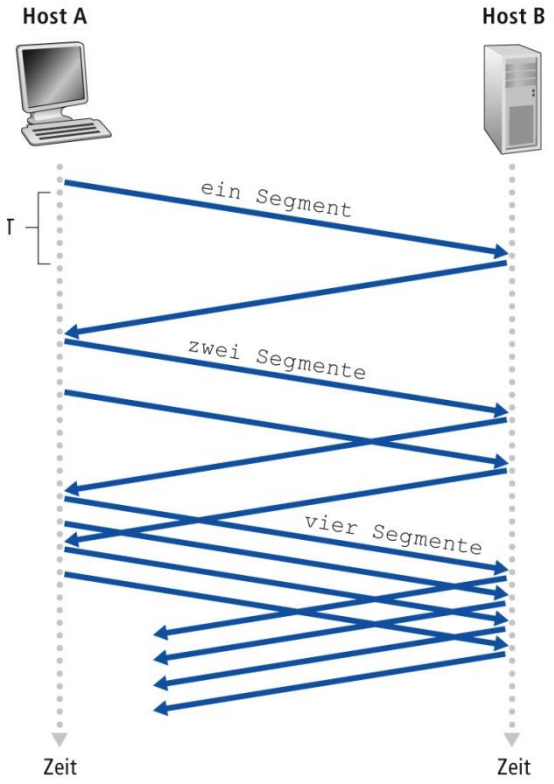
- TCP definiert folgende Parameter:
  - CongWindow (congestion window): steuert die Einspeisung ins Netz wie folgt:
    LastByteSent – LastByteAcked ≤ CongWindow
  - Effektives Fenster: min(RcvWindow, CongWindow)
  - MSS (maximum segment size): maximale Datenmenge pro Segment.
  - Threshold: obere Schranke für das schnelle Wachstum des Überlastfensters (CongWindow).
  - RT (retransmission timer): nach Ablauf des Timer werden nicht bestätigte Segmente neu gesendet.

Stauvermeidung: 3 Phasen

1. Slow Start (exponentielles Wachstum von CongWin)
2. Stauvermeidung (lineares Wachstum)
3. Bei Paketverlust (Threshold anpassen)
   a. TCP Tahoe: CongWin zurücksetzen und mit Schritt 1 neu beginnen.
   b. TCP Reno: CongWin auf Threshold setzen und lineares Wachstum fortsetzen → Fast Recovery

Slow Start

- Die Übertragung beginnt mit einem kleinen Überlastfenster (CongWin = 1 MSS).
- Bis zum Erreichen eines *Threshold* inkrementieren wir CongWin bei jeder erhaltenen Quittung um eins ➔ CongWin verdoppelt sich nach jeder Übertragungsrunde
- Eine Übertragungsrunde entspricht in etwa einer RTD
- Bei Paketverlust: *Fast Retransmit*, *Threshold* auf (*CongWin* / 2) setzen und *CongWin* auf den ursprünglichem initialen Wert (bspw. 1) zurücksetzen.



Stauvermeidung

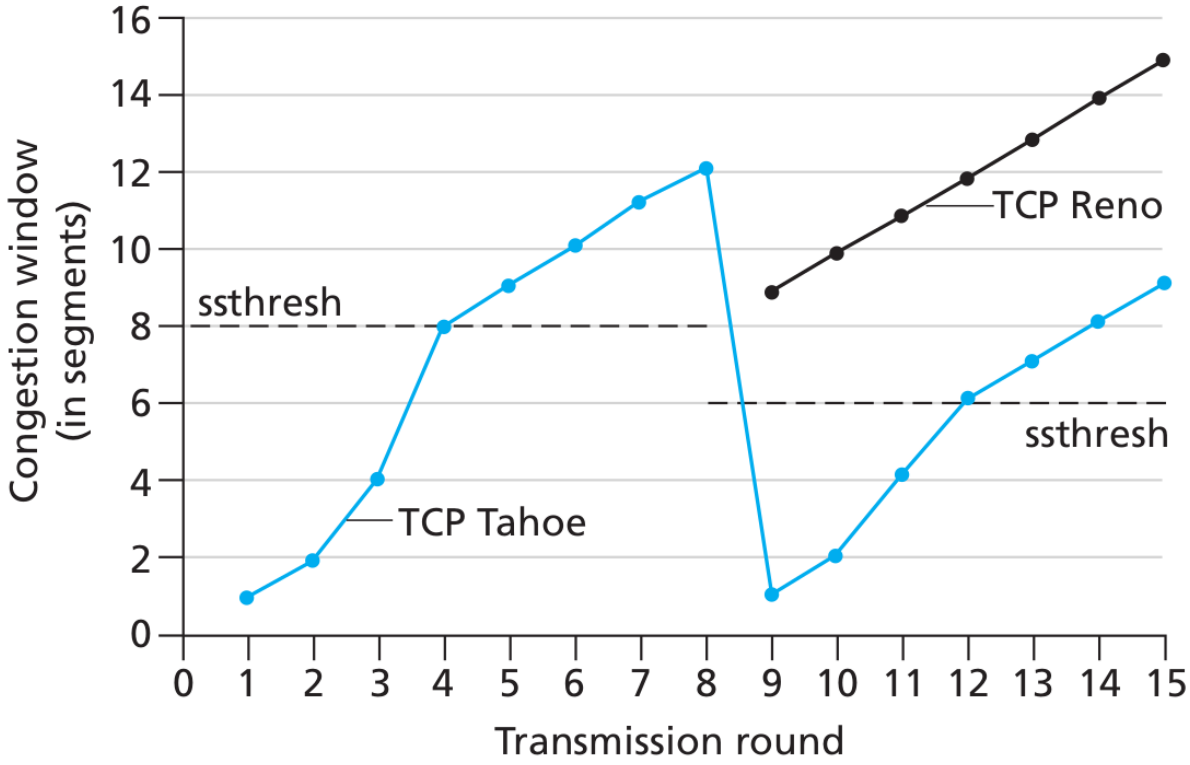
- Bei Erreichen des Threshold Übergang zu linearem Wachstum.
- Falls ein ACK ankommt: Erhöhe CongWin um $\frac{1}{CongWin}$ MSS. → Erhöht CongWin um 1 pro RTD.
- Bei Paketverlust:
  - Timeout: Zurücksetzen von *CongWin* auf 1, *Threshold* auf (CongWin / 2)
  - Bei 3 duplizierten ACKs: Fast Recovery

Fast Recovery

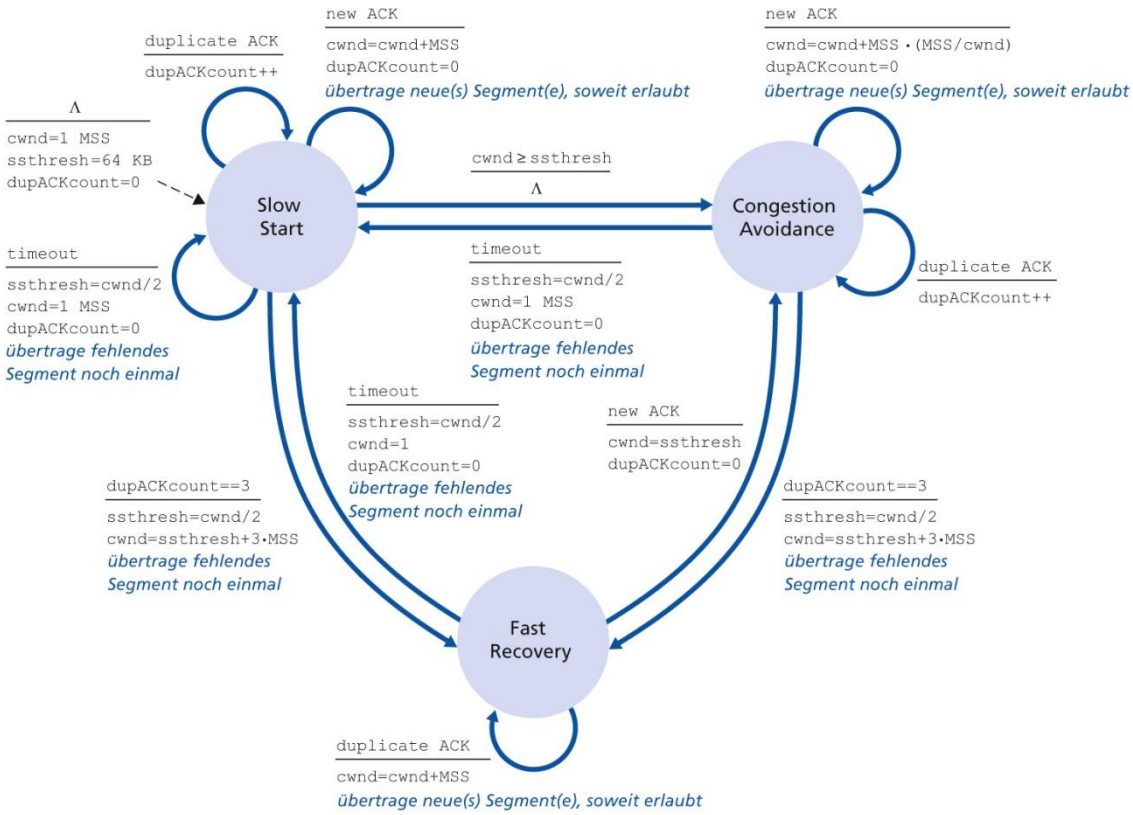
- Positive Annahme: Duplizierte ACKs zeigen, dass Segment beim Empfänger ankommen.

  → Slow Start zu konservativ.
- Optimierung durch Fast Recovery
  - Setze *Threshold* ebenfalls auf CongWin / 2.
  - CongWin = *Threshold + 3 MSS* (ACKs berücksichtigen)
- Falls ein ACK für das ursprünglich verlorene Segment eintritt, Übergang zu *Stauvermeidung*.

Staukontrolle (Ablauf)



Stauvermeidung (Zustandsdiagramm)



TCP Retrospektive

- Stauvermeidung folgt dem ***Additive Increase Multiple Decrease*** *(AIMD)* Prinzip.
- Wenn Slow Start ignoriert wird, linearer Anstieg (Additiv)
- Bei Paketverlust Halbierung des CongWin um Faktor 2 (Multiple Decrease).

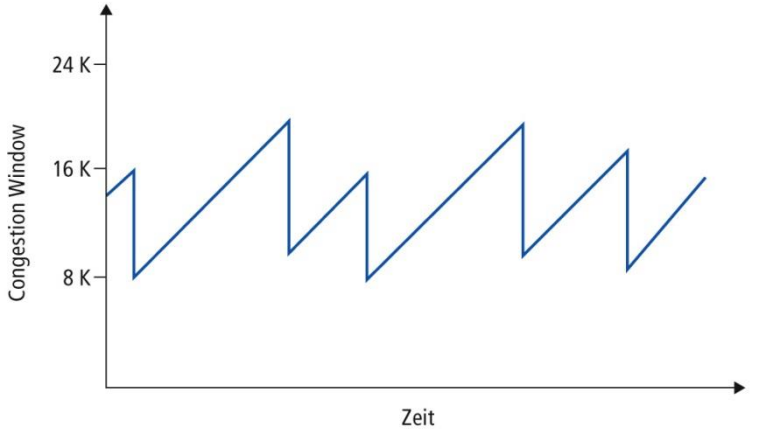


Abbildung 3.54: Überlastkontrolle: Additive-Increase, Muliplicative-Decrease.

Approximation des Durchsatzes

- Gegebene Variablen
  - *w:* aktueller Wert von CongWin
  - *W:* Wert von CongWin bei Paketverlust
  - *RTD:* Round Trip Delay

- Angenommen *W* bleibt konstant über die Verbindung: Durchsatz pendelt immer zwischen W/(2 RTD) und W / RTD.

→ Ungefährer Durchsatz bei TCP: (0.75 W) / RTD.

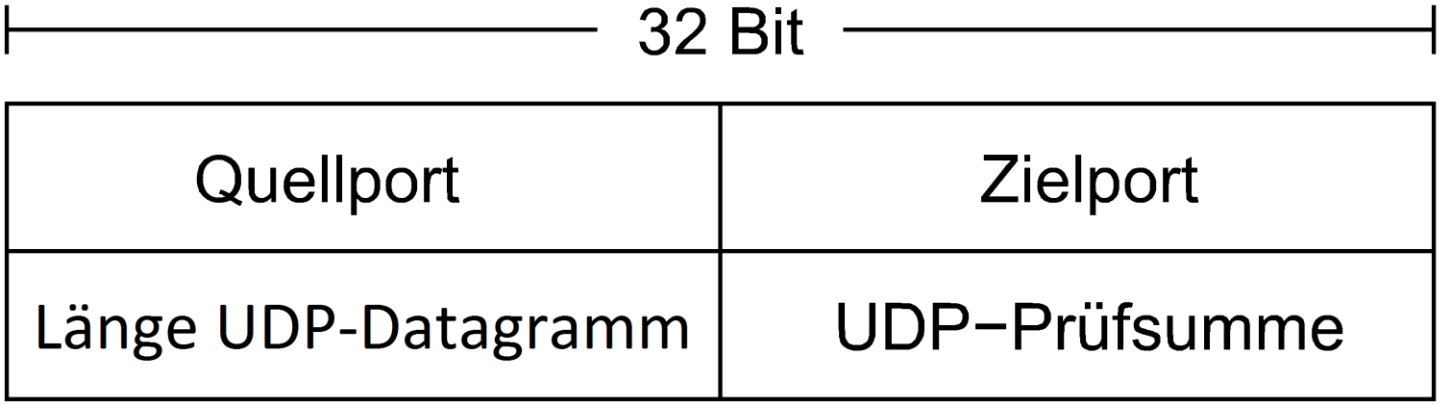
Kapitel 4.6
User Datagram Protocol (UDP)

Das verbindungslose Transportprotokoll des Internets

Überblick UDP

- Verbindungsloses, unzuverlässiges Internet Transportprotokoll (RFC 768)
  - kein Verbindungsaufbau, -abbau, oder -status
  - unregulierte Senderate
  - keine Sequenznummern
    ➔ keine Reihenfolgesicherung, Duplikats- oder Verlusterkennung
- 8 Byte Header (vgl. mit mindestens 20 Byte bei TCP)
- Unterstützt das Multiplexen über Anwendungen mit Hilfe von UDP-Ports (wie bei TCP)
- Bsp. Protokolle, die UDP verwenden: TFTP, DNS, RPC, SNMP
- UDP-Nutzer (Anwendungen) sind über UDP-Sockets adressierbar.

Der UDP Header (PCI)

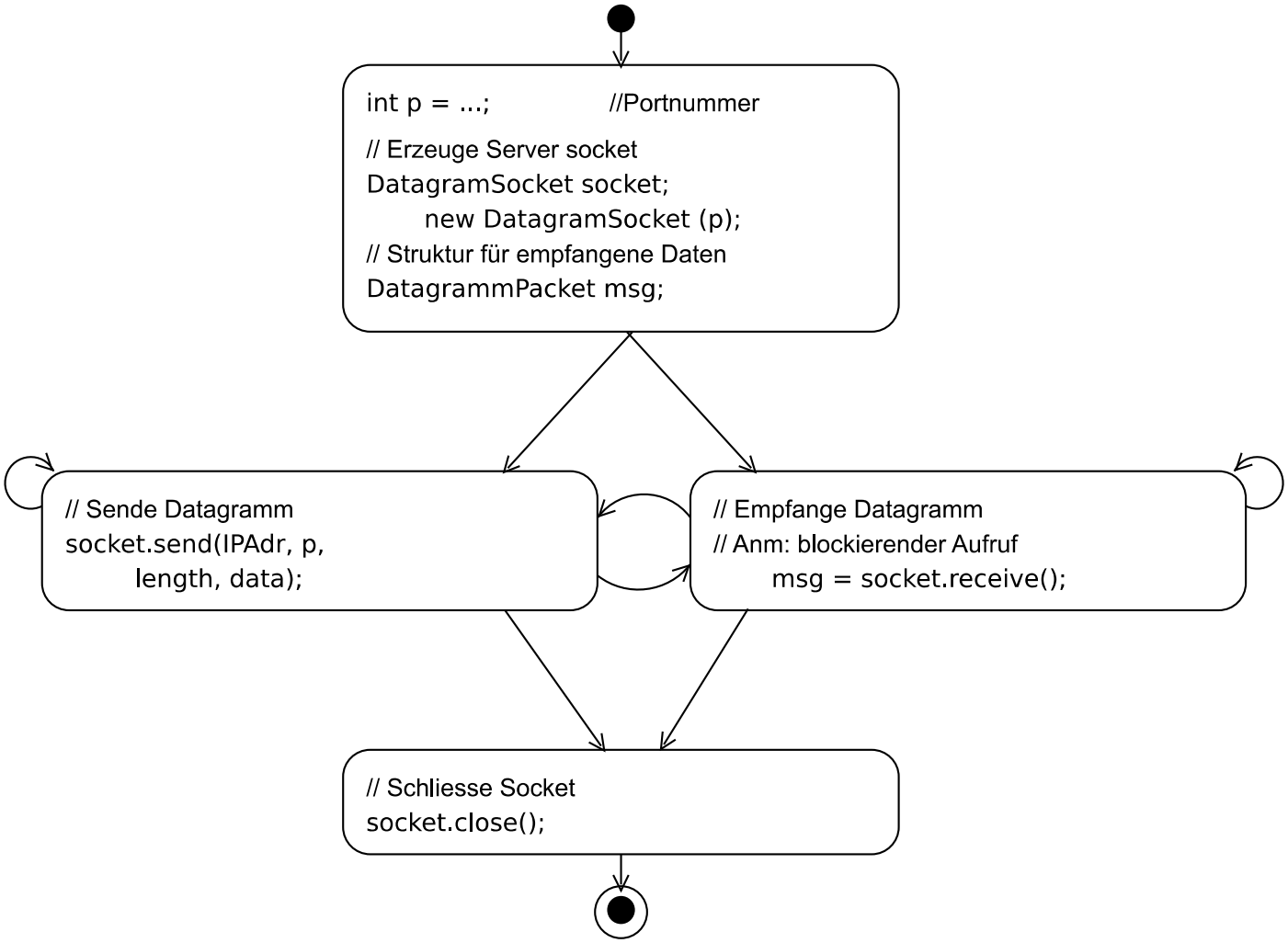


- **Quellport/Zielport** (je 16 Bit): zur Adressierung
- **Datagrammlänge** (16 Bit): Maximallänge von 65515 Bytes (bei Verwendung von IPv4)
- **Prüfsumme** (16 Bit): (optionale) Prüfsumme über IP-Pseudoheader und UDP-Datagramm

Verwendung von UDP

- Programmierung mit UDP Sockets
  - Senden/Empfangen isolierter Datagramme
  - Sicht auf lokales Socket
  - Senden/Empfangen zu/von mehreren Hosts (IP-Adresse und Port des Empfängers werden i.d.R. beim Senden jedes einzelnen Datagramms angegeben)
- Reihenfolgesicherung, Verlust- und Duplikatsbehandlung, sowie die Steuerung der Senderate obliegen dem Anwendungsprogramm.

Beispiel: UDP-Socket-Programm



Zusammenfassung Transportschicht

- Netzunabhängiger Transport von Nachrichten zwischen Endsystemen
- Verschattung der Wege durchs Netz
- Fehlerbehandlung Ende-zu-Ende
- Anpassung der Übertragungsqualitäten
- Splitting/Multiplex über Anwendungen/Prozesse
- Verbindungsloser oder -orientierter Dienst
- z.B. im Internet: TCP (verbindungsorientiert), UDP (verbindungslos)

Fragen zu Kapitel 4

- Was sind Einflussgrößen für die Größe des Sequenznummernraumes?
- Wozu kann die Fenstertechnik (Schiebefensterprotokoll) eingesetzt werden?
- Wie wird Eindeutigkeit der Sequenznummern sichergestellt?
- Wie wirken sich zu kleine Fenster aus, wie zu kleine Sequenznummernräume?
- Unter welchen Voraussetzungen genügt ein Zwei-Wege-Handschlag für einen Verbindungsaufbau?
- Warum ist für einen sicheren Verb.-Aufbau auf einer Transportschicht i.A. ein Drei-Wege-Handschlag erforderlich?
- Warum sollte ein Verbindungsabbau i.A. beidseitig geschehen?
- Was ist der Unterschied zwischen Flusssteuerung und Staukontrolle?
- Ist TCP ein „Go-Back N“ oder Selective Repeat Protokoll (SR)?