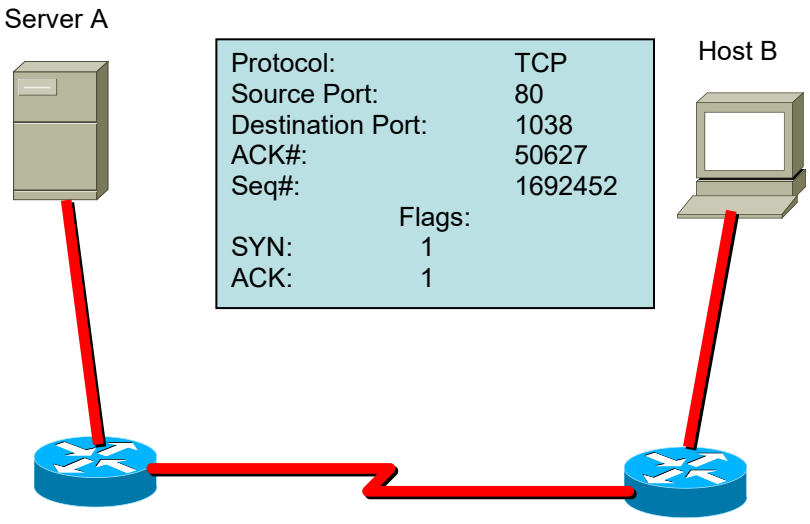
Informatik Bachelor, LV „Rechnernetze“, SS 2025

Dozent: Prof. Dr. Michael Massoth Datum: 04.06.2025	**Übung-09**
Vorname: David **Nachname:** Schirrmeister	**Paketanalyse, UDP und TCP** **Punkte (von 16):**

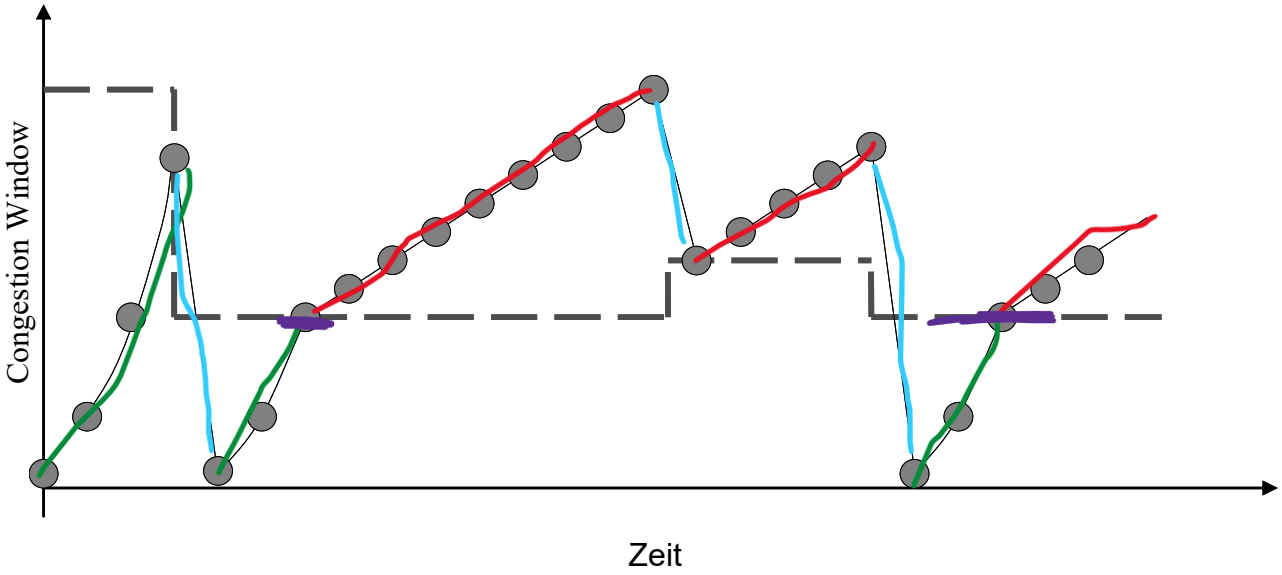
Aufgabe 1: **(3 P = 1+1+1 P)**

Betrachten Sie die nachfolgende Abbildung. Welche Fakten kann man aus den gegebenen Informationen ableiten? (**Wählen Sie 3 Antworten**.)



- ☒ Die Nachricht gehört zum TCP-Verbindungsaufbau (3-Wege-Handshake).
- ☒ Am Quellport (Source Port) erkennt man, dass eine HTTP-Verbindung vorliegt.
- ☒ Die Daten fließen vom Server A zum Host B.
- ☐ Der Zielport (Destination Port) lässt auf eine TELNET-Verbindung schließen.
- ☐ Die Daten gehören zu einer UDP-Übertragung.
- ☐ Die Nachricht gehört zum TCP-Verbindungsabbau.

Aufgabe 2: (5 P = 1+1+2+1 P) Transportschicht



a) Mit welchem Protokoll und welchem Dienst ist diese Abbildung assoziiert? (1 P)

TCP

Dienst: Congestion Control

b) Beschriften Sie die Diagramm-Achsen. (1 P = 0,5+0,5 P)

c) Markieren Sie die beiden unterschiedlichen algorithmischen Phasen, und benennen Sie diese Phasen. (2 P = 1+1 P)

SlowStart

CongestionAvoidance

d) Tragen Sie im Diagramm die Ereignisse ein, die den jeweiligen Phasenwechsel ausgelöst haben. (1 P)

Cwnd erreicht ssthresh

Paketverlust

Aufgabe 3: (4 P = 2+2 P) Silly Window Problem von TCP

Wie kann das Silly Window Problem von TCP gelöst werden?
Denken Sie dabei sowohl an die Senderseite, als auch an die Empfängerseite.

Das Silly Window Problem tritt auf, wenn der Empfänger kleine Fenstergrößen (z. B. 1 Byte) meldet, die sofort vom Sender genutzt werden, wodurch Ineffizienz durch viele kleine Pakete entsteht.
Lösungen:
Senderseitig

- Der Sender wartet, bis er entweder:
  - das gesamte Fenster füllen kann oder
  - eine maximale Segmentgröße (MSS) senden kann.
- Das verhindert, dass er viele kleine Pakete schickt.

Empfängerseitig:

- Der Empfänger meldet kein kleines Fenster zurück (nur wenn es groß genug ist).
- Erst wenn ausreichend Puffer frei ist (z. B. MSS oder halbes Fenster), wird wieder ein ACK mit Fenstergröße > 0 geschickt.

<u>Aufgabe 4:</u> (4 P = 2+2 P) Guter Retransmission-Timer für adaptive Neuübertragung von TCP-Segmenten

Für die Neuübertragung von verloren gegangenen oder beschädigten TCP-Segmenten muss ein guter Retransmission-Timer bestimmt werden. Wie wird dieser adaptive Retransmission-Timer berechnet?

1. Ermittlung der RTT (Round Trip Time):
    - RTT wird bei jeder bestätigten Übertragung gemessen.
2. Glättung des RTT-Werts (smoothed RTT):
    - SRTT = (1 − α) * SRTT + α * RTT_sample
    - Typisch: α = 1/8
3. Berechnung der Abweichung (RTT-Varianz):
    - RTTVAR = (1 − β) * RTTVAR + β * |SRTT − RTT_sample|
    - Typisch: β = 1/4
4. RTO-Berechnung:
    - RTO = SRTT + max(G, 4 * RTTVAR)
      (G = Clock Granularity, z. B. 1 ms)

Besonderheiten:

- Exponential Backoff: Wenn Timeout auftritt, wird RTO verdoppelt.
- Initial RTO: Standardmäßig auf z. B. 1 Sekunde gesetzt.