

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Willkommen bei Kommunikations- und Netztechnik!

Von Kupferkabel, Glasfaser und Mikrowelle
über Telefon, Ethernet und TCP
zu E-Mail, Webserver und REST.

☺

Heute: **Transportschicht: Von Anwendung zu Anwendung.**

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Organisatorisches

■ sci-hub und libgen bekannt?

Ziele heute II

■ Sie kennen den TCP-Header

■ Sie können einen Varianzbasierten RTO berechnen (retransmission timeout)

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Wiederholung

■ Routing: Quell- Senken Bäume

■ VC (virtuelle Verbindung) vs. Paket

■ Fluten + Optimierung

■ Routing-Tabellen

■ Warteschlangen verstehen

■ Drosseln

■ IPv4 vs IPv6

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Ziele heute I

■ Sie verstehen, dass die Transportschicht Segmente mit eigenem Header an die Vermittlungsschicht reicht

■ Sie verstehen, dass die Transportschicht Prozesse verbindet und über Ports adressiert und IPs über die Vermittlungsschicht laufen

■ Sie können ein 3-way Handshake-Diagramm aufschreiben

■ Sie können ein 3-way Verbindungsabbau-Diagramm aufschreiben

■ Sie wissen, dass es bei 2 Teilnehmenden immer essentielle Nachrichten gibt, die nicht verloren gehen dürfen

■ Sie wissen, dass Datenverlust bei Servercrash unvermeidbar ist

■ Sie verstehen AIMD (additive increase multiplicative decrease)

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Ziele heute II

■ Sie kennen den TCP-Header

■ Sie können einen Varianzbasierten RTO berechnen (retransmission timeout)

Gründe für Transportschicht

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

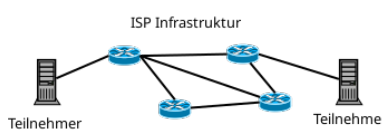
TCP

QUIC

Zusammenfassung

Anhang

Gründe für Transportschicht



■ Transportschicht: läuft auf Computer der Teilnehmer

■ Netzwerkschicht: läuft auf Netzwerk-Hardware der Provider

Aufgaben der Transportschicht

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

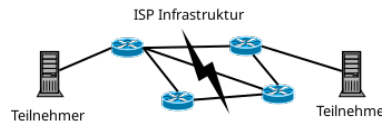
TCP

QUIC

Zusammenfassung

Anhang

Gründe für Transportschicht



■ in der Netzwerkschicht kommt es zu Problemen

■ diese können nicht auf Netzwerkschicht behoben werden

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP


TCP

QUIC

Zusammenfassung

Anhang

Gründe für Transportschicht



Transportschicht macht unzuverlässige Netzwerkschicht zuverlässig

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Aufgaben der Transportschicht

■ Zuverlässigkeit

■ Effizienz

■ Flusskontrolle

■ Überlastkontrolle

Praktisch: Berkeley Sockets

Werden in vielen OS (Operating System) verwendet.

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Gemeinsamkeiten Vermittlungsschicht

■ verbindungslos und -orientiert möglich

■ Adressierung von Hosts

■ Flusskontrolle

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

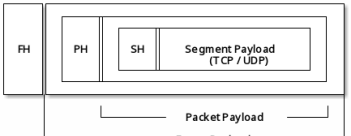
QUIC

Zusammenfassung

Anhang

Segmente: Noch ein Header

Die Transportschicht verschickt Segmente, die in Netzwerkschicht Pakete eingebettet sind.



Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Praktisch: Berkeley Sockets

Werden in vielen OS (Operating System) verwendet.

Function	Bemerkung
socket()	definiere verwendetes Protokoll
bind()	ordne Socket eine Netzwerkadresse zu
listen()	erzeuge Queue, bereit für Verbindung
connect()	blocking; baut Verbindung zu Server auf
accept()	blocking; erzeugt Filedescriptor für Verbindung
send(), receive()	sende und empfangene Daten
close()	beende Verbindung

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Zusammenfassung

■ Kanal zwischen Prozessen

■ Segmente in Paketen

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Eigenschaften von Transport Protokollen

■ Probleme bei Paketen:

- out-of-order
- packet loss
- duplication

■ Aufgaben Transportschicht:

- Fehler-
- Fluss-
- Überlastkontrolle
- Sequenzierung

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Adressierung

■ Netzwerkschicht: Adresse (Bsp: IP)

■ Transportschicht: Ports (Bsp: 80 HTTP)

■ Ports werden verwendet, um eine IP für mehrere Prozesse zu teilen

■ Problem: auf welchen Port soll connected werden?

- well known ports: 22, 25, 80, 443
- portmapper: wie Telefonauskunft

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Problem: doppelte Segmente

Folgendes Szenario:

■ Überweisung per Online Banking

■ Segment mit Überweisung benötigt zu lange

■ wiederholte Übermittlung

■ beide Segmente kommen an

■ Überweisung wird doppelt ausgeführt

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Lösung: Ids?

■ jedes Segment verfügt über eine Id

■ Nachteil: Sender & Empfänger müssen Buch führen

■ Buch muss auch Neustarts überleben

■ -> teuer

Anne Babenhausenheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg

Transporttschicht

Eigenschaften

UDP

TCP

QUIC

Zusammenfassung

Anhang

Beschränkung der Lebenszeit

■ Lebenszeit von Segmenten beschränken

- Hop Counter
- Timestamp

■ -> garantiert, dass Segmente sterben können

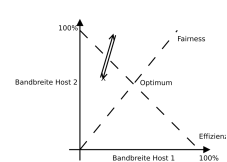
■ Id kann nach Periode T wiederverwendet werden

■ T ist mehrfaches der maximalen Paketlebenszeit

■ im Internet: 120s



Fehlerkontrolle

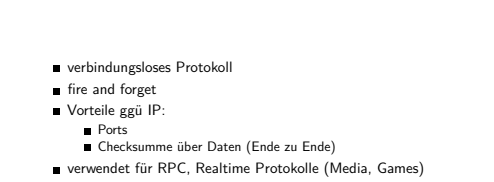


- Gerade durch Arbeitspunkt und Ursprung
- Arbeitspunkt oszilliert um Effizienz
- \rightarrow keine Konvergenz

Arne Babenhauerheide und Carlo Götz

[illegible]

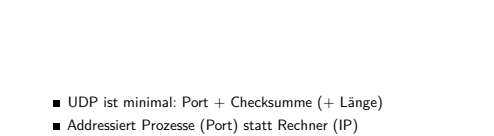
UDP (User Datagram Protocol)



Arne Babenhauserheide und Carlo Götz

Netztechnik 5: Transportschicht							
Einstieg 0000	Transportschicht 00000 000	Eigenschaften 00000000 00000000000000 00000000000000 00000000000000 00000000000000	UDP 0000	TCP 00000000 0000 00000000000000 00000000	QUIC 000	Zusammenfassung 00	Anhang

UDP Zusammenfassung



- UDP ist minimal: Port + Checksume (+ Länge)
- Adressiert Prozesse (Port) statt Rechner (IP)

Arne Babenhäuserheide und Carlo Götz

[illegible]

TCP

Source / Destination Port

- Bsp: 80 (HTTP)
- Port + IP definieren einen Endpunkt
- 2 Endpunkte definieren eine Verbindung

Arne Babenhauerheide und Carlo Götz

Netztechnik 5: Transportschicht

Einstieg ○○○○	Transportschicht ○○○○○ ○○○	Eigenschaften ○○○○○○○○ ○○○○○○○○○○○○○○ ○○○○○○○○○○○○○○○○○○	UDP ○○○○	TCP ○○○○●○○ ○○○ ○○○○○○○○○○○○○○○○○○	QUIC ○○○	Zusammenfassung ○○	Anhang
------------------	----------------------------------	---	-------------	---	-------------	-----------------------	--------

TCP Window Size

- wieviele Bytes dürfen ab ACK Number gesendet werden?
- 0 ist legaler Wert
 - Weiteres Senden möglich nach Segment mit gleicher ACK No und Window Size > 0

Arne Babenhauserheide und Carlo Götz

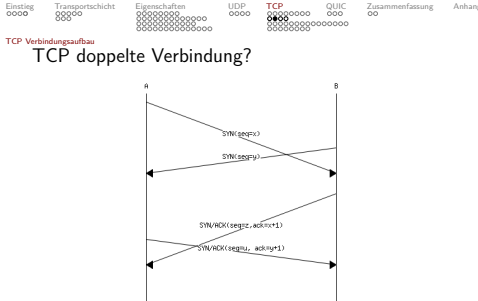
[illegible]

TCP Verbindungsaufbau

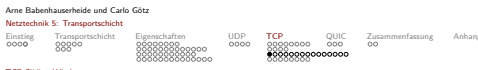
- initiale seq wurde früher mit externer Uhr generiert
 - passiver Teilnehmer muss seq von SYN Segmenten speichern
 - DOS (Denial of Service) möglich durch senden vieler SYN Segmente
 - Lösung SYN Cookies : Generierung initiale seq bei Bedarf

Arne Babenhauerheide und Carlo Götz

Netztechnik 5: Transportschicht

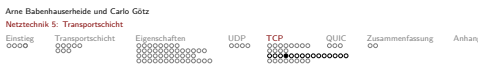


Wieviele Verbindungen werden geöffnet?



TCP Sliding Window

- Window = 0 ist legaler Wert
 - bedeutet: keine weiteren Daten senden
 - außer: urgent data
 - oder: Window Probe
- Sender können Daten buffern vor dem Senden
- Empfänger müssen ACK nicht sofort senden

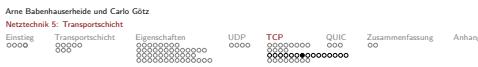


TCP Sliding Window

-

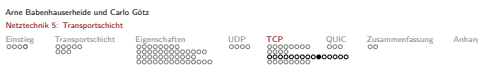
Nagles Algorithmus:

- wenn viele kleine Segmente gesendet werden sollen
- sende erstes Segment
- buffere alle weiteren Segmente
- sende alle gepufferten Segmente sobald ACK eintrifft
- kann mit der TCP_NODELAY Option ausgeschaltet werden
 - besser: TCP_QUICKACK



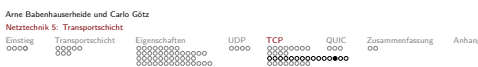
TCP Timer

- RTO (Retransmission Timeout): muss ein Segment erneut gesendet werden?
- Persistence Timer: verhindert Deadlock
 - Empfänger: Window Size=0
 - Window Update geht verloren
 - Persistence Timeout triggert Window Probe
- Keepalive Timer: sende Segmente, um Verbindung offen zu halten
- FIN Timer: beende Verbindung nach Timeout

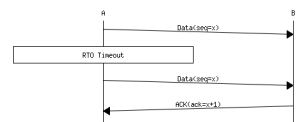


Initialisierung:

- SRTT = RTT
- RTTVAR = RTT / 2
- RTO = SRTT + 4 * RTTVAR
- erfolgt bei erster RTT Messung



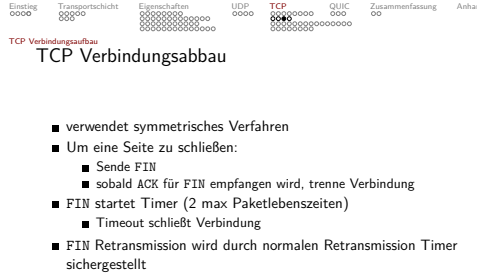
Problem Messung RTT



Auf welches Segment bezieht sich ACK?

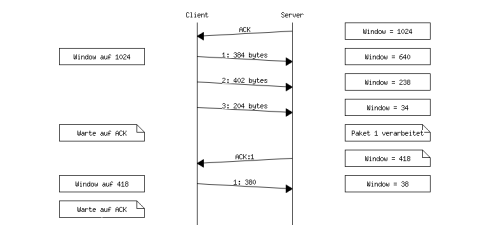
Lösung: Karn Algorithmus

- kein Update der Werte bei wiederholter Übermittlung
- jeder Timeout verdoppelt RTO



TCP Sliding Window

-



Nagle zu Delayed ACK

<https://news.ycombinator.com/item?id=10607422>

Unfortunately by the time I found about delayed ACKs, I had changed jobs, was out of networking, and doing a product for Autodesk on non-networked PCs.



TCP - Retransmission Timer

Problem:

- zu kurz: viele Retransmissions
- zu lang: hohe Latenz bei Packet Loss
- ständige Veränderung durch Überlastkontrolle

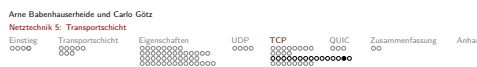
Lösung:

- dynamischer Algorithmus
- Berechne Smoothed Round Trip Time (SRTT)
 - $SRTT_{i+1} = \alpha SRTT_i + (1 - \alpha) RTT_{i+1}$
 - $\alpha = \frac{1}{8}$ Smoothing Factor
- Früher: RTO = 2 * SRTT_{i+1}



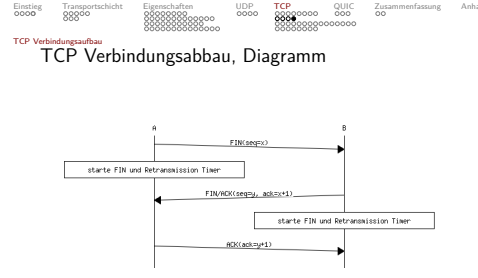
Übung RTO

Berechne RTO in T_{1}. RTT_{0} = 50 ms, RTT_{1} = 30 ms.



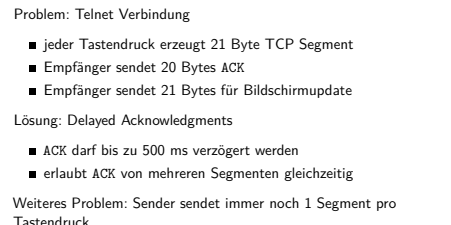
TCP Überlastkontrolle

- TCP verwaltet congestion window
 - wieviele Bytes können sich auf einmal im Netzwerk befinden
- Größe wird mit AIMD angepasst
- zusätzlich zu window der Flusskontrolle
- TCP hört auf zu senden sobald eins der Fenster voll ist
- Packet Loss als binäres (impräzises, implizites) Signal



TCP Sliding Window

-



Silly Window Syndrome

- Anwendung auf Empfängerseite liest immer nur 1 Byte
- wenn Window voll ist, zwingt dies TCP nur noch 1 Byte große Segmente zu verschicken

Lösung:

- Empfänger sendet nur window update sobald mindestens die maximal Segmentgröße Platz ist



Varianz!

Problem: betrachtet nicht die Varianz: wenige Pakete stark verzögert.

Lösung:

- berechne Round Trip Time Variation (RTTVAR)
 - $RTTVAR_{i+1} = \beta RTTVAR_i + (1 - \beta) |SRTT_i - RTT_{i+1}|$
 - $\beta = \frac{1}{4}$
- $RTO_{i+1} = SRTT_{i+1} + 4 RTTVAR_{i+1}$
- keine echte Varianz aber gute Näherung
- SRTT_{i+1} berechnet wie vorher.
- RTO mindestens 1 Sekunde



Lösung RTO

$SRTT_1 = \alpha 50 + (1 - \alpha) 30 = 47,5$
 $RTTVAR_1 = \beta 25 + (1 - \beta) |50 - 30| = 23,75$
 $RTO_1 = 47,5 + 4 * 23,75 = 142,5$

- $\beta = \frac{1}{4}$
- $\alpha = \frac{1}{8}$



ACK Clock

- mehrere kleine Pakete werden gesendet
- werden in Router vor Flaschenhals gepuffert
- Empfänger bestätigt einzelne Pakete
- ACKs treffen bei Sender ein
- Rate der ACKs entspricht optimaler Senderate



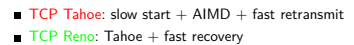
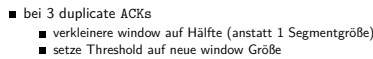
Fast Retransmission

- starte mit kleinem congestion window
- pro Segment, das vor RTO bestätigt wird: vergrößere window um 1 Segment
- exponentielles Wachstum führt zu Überlast
- verwalte zusätzlich einen Schwellenwert (Slow Start Threshold)
- bei Packet Loss: setze Threshold auf Hälfte des congestion windows
- bei Überschreitung der Threshold: verwende additive increase
 - 1 Segment pro RTT (auch hier mit ACK Clock)

Lösung: duplicate ACK



AIMD - Initialisierung



Arne Babenhauserheide und Carlo Göttsch

Netztechnik 5: Transportschicht

Einstieg Transportschicht Eig.

○○○○ ○○○○ ○○

○○ ○○

○○

QUIC

QUIC: Wieso?

- 3-way Aufbau
- 3-way Abbau
- AIMD-Optimierungen:
 - Slow Start ist schnell
 - Fast Retransmission
 - Fast Recovery

Zusammenfassung

- Initialdaten für Setup + Verschlüsselung gleichzeitig \Rightarrow 3 Pakete statt 6
- Bei gleicher Session in Paket 0 schon Daten schicken
- Erstes Paket auf max MTU (Paketgröße) padden, um Pfade mit zu kleinen Paketgrößen zu vermeiden.

- Kanal Zwischen Prozessen
- 3-way Aufbau
- 3-way Abbau
- UDP ist minimal: Port und Länge zu IP dazu
- AIMD-Optimierungen:
 - Slow Start ist schnell
 - Fast Retransmission
 - Fast Recovery

Arne Babenhauer/Heide und
Netztechnik 5: Transportsch
Übungsblatt Transportsch
○○○ ○○○○○○

Verweise I

Nächstes Mal geht es weiter mit Anwendungen.

- 7,4 Hamming Code-Golf:
<https://codegolf.stackexchange.com/questions/45684/correct-errors-using-hamming7-4>

Arne Babenhauerheide und Carlo Götz
Netztechnik 5: Transportschicht
Übungsblatt Transportschicht

Beispiel Aufgabe Multiple Choice

- Bei Multiple Choice Aufgaben reicht eine Lösung nach folgendem Muster:

- ☐ Die letzte Vorlesung war viel zu schnell
- ☐ Sriracha passt zu allem
- ☐ Tabs sind besser geeignet für die Einrückung von Quellcode

Beispiel Lösung Multiple Choice

1, 2

Aufgabe 3

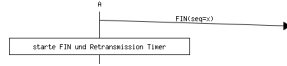
Ekläre die 3 Kriterien: Effizienz, Fairness und Konvergenz.

Aufgabe 1

Kreuze die korrekten Aussagen an:

- 1 ☐ UDP Segmente kommen immer in Absendereihenfolge beim Empfänger an.
- 2 ☐ UDP Segmente können verloren gehen.
- 3 ☐ Erfolgreich empfangene UDP Segmente können beschädigt sein.
- 4 ☐ Segmente können vom Netzwerklayer dupliziert werden.
- 5 ☐ Die function accept () wird in der Regel client-seitig aufgerufen.
- 6 ☐ Ein Telefongespräch wird symmetrisch getrennt.

Aufgabe 5



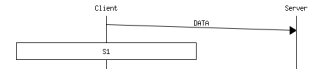
Vervollständige das Sequenzdiagramm für folgende 3 Fälle bis zur Trennung der Verbindung:

- 1 FIN Timer wird ausgelöst.
- 2 Retransmission Timer wird ausgelöst.
- 3 B sendet ein ACK Segment mit seq=y und ack=x+1

Aufgabe 2

Zeichne das Sequenzdiagramm für folgende Client- und Serverkonfiguration. Kommt es zu duplizierten/verlorenen Daten oder ist alles in Ordnung?

- Client: Always retransmit
- Server: First write then ACK
- Eventreihenfolge: Write, Crash, Ack



Aufgabe 5

Zeichne ein Diagramm mit Congestion Window Größe (in Segmenten) auf der y-Achse und Transmission Round (von 0 bis 8) auf der x-Achse für folgende Parameter:

- Threshold = 16 Segmente
- Packet Loss in Transmission Round 6
- Verwendung von Slow Start und Fast Recovery