TCP

TCP

GRUNDLEGENDE EIGENSCHAFTEN VON TCP

DIE HAUPTAUFGABEN VON TCP SIND

- sicherstellen, dass Daten nicht verändert werden
- sicherstellen, dass Daten nicht verloren gehen
- sicherstellen, dass Daten nicht dupliziert werden
- sicherstellen, dass Daten in der richtigen Reihenfolge eintreffen
- Fluss- bzw. Staukontrolle
- Multiplexing
- Segmentierung

FÄHIGKEITEN TCP

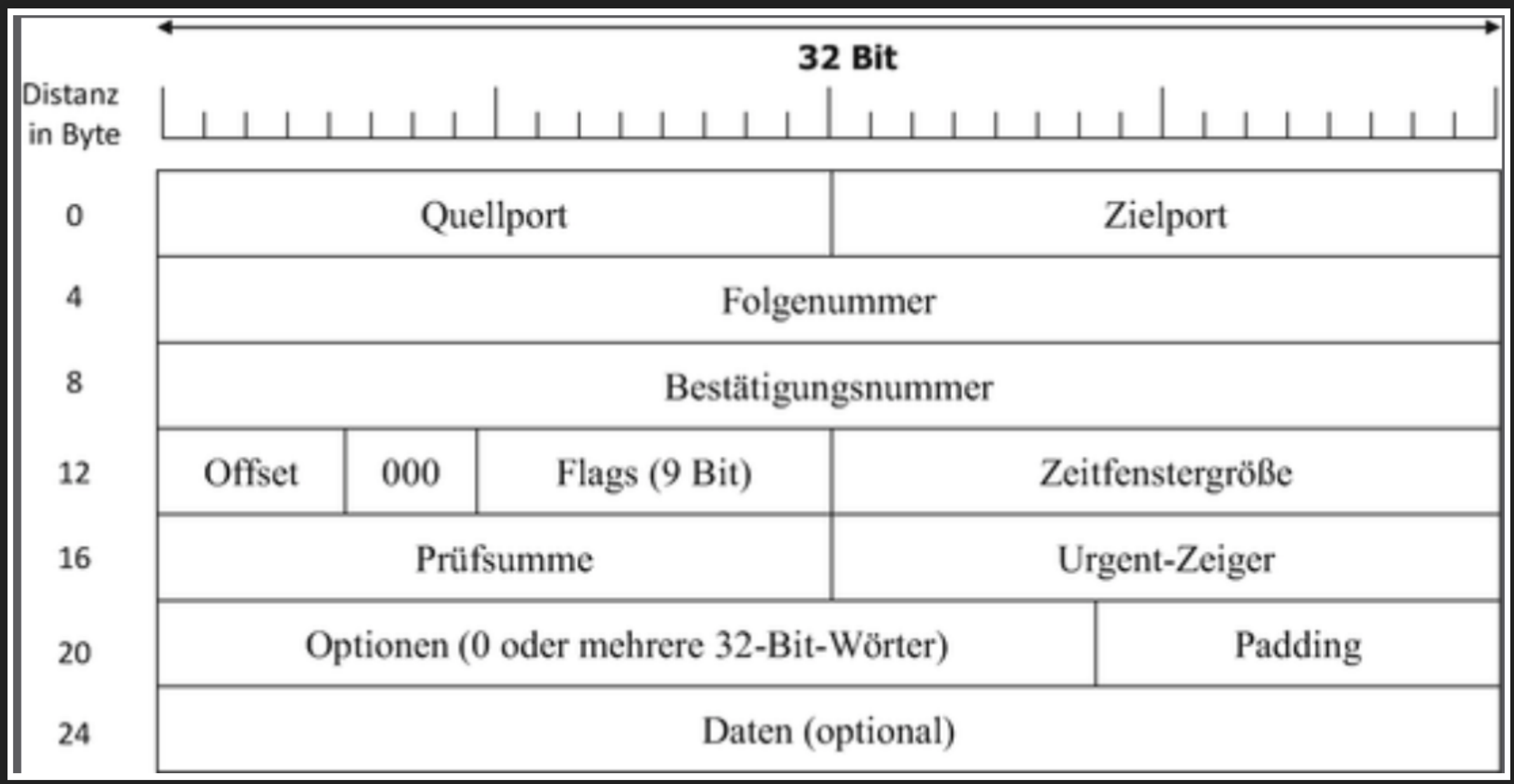
TCP ermöglicht Kommunikation über vollduplex fähige Verbindungen, das heißt beide Kommunikationspartner können jederzeit senden und empfangen, auch gleichzeitig. TCP übernimmt die segmentierung für die Anwendung im gegensatz zu ihrem OSI Equivalent OSI TP4

WELL KNOWN PORTS

- Port 22, Telnet
- Port 20 und 21, ftp
- Port 25, SMTP
- Port 80, HTTP
- Port 443, HTTPS

AUFBAU EINES TCP PAKETES

TCP HEADER



Header

HEADER TEIL 1

- **Quellport:** Port des Senders
- **Zielport:** Port des Empfängers
- **Folgenummer:** Eindeutige Nummerierung des Paketes im Datenstrom
- **Bestätigungsnummer:** Eindeutige Nummerierung des nächsten Paketes im Datenstrom
- **Offset:** Größe des Headers in 32-Bit Worten (da keine fixe Länge)
- **Reserved:** Hat bisher noch keine Nutzung

Folge und Bestätigungsnummer dienen Flusskontrolle, sowie Synchronisation

HEADER TEIL 2

- **Flags:** Steuerkennzeichen zu besonderen Aufgaben (folgt noch)
- **Zeitfenstergröße:**
- **Prüfsumme:** Prüfsumme für das Gesamtpaket, also Header * Daten
- **Urgent-Zeiger:** Zeigt auf Position, an der sich wichtige Daten befinden
- **Optionen:** Optionale Angaben über Verbindungsparameter (folgt noch)
- **Padding:** Auffüllen des verbleibenden Platzes zum Alignment

FLAGS

Flag Bit	Bedeutung
CWR	Nutzung in der Staukontrolle
ECE	Nutzung in der Staukontrolle
URG	Urgent Feld wird genutzt
ACK	Bestätugungsnummer (Acknowledgement) wird genutzt
PSH	TCP Empfängersegment darf nicht puffern, sondern muss direkt weiterleiten

Flag Bit	Bedeutung
RST	Ungültiges TCP Segment / Verbindung abgelehnt / Senderprozess abnormal beendet
SYN	Verbindung aufgebaut (Synchron)
FIN	Verbindung abgeschlossen (Finished)

RÜCKBLENDE TSAP/NSAP

TSAP und NSAP sind Addressen, die ermöglichen Daten an ein spezielles Ziel zu schicken. Sie sind vergleichbar mit teilen einer Adresse. Dabei könnte NSAP für Stadt und Postleitzahl stehen, während TSAP für Straße und Hausnummer steht.

TSAP

- Steht für Transport Service Access Point
- Dient als Teil Adresse für Anwendung zum Verschicken sowie Empfangen
- Dient zur feineren Aufteilung vom NSAP
- In TCP gegeben durch Port

NSAP

- Steht für Network Service Access Point
- Dient als Teil der Adresse zum Verschicken sowie Empfangen
- Dient zur Findung des Adressaten vom Transport Layer
- In TCP gegeben durch IP-Adresse

VERBINDUNGSAUFBAU

- Verbindet Socket zu Socket
- Socket bestehend aus IP-Addresse und Port (TSAP und NSAP)
- 3-Wege Handshake:
  - TCP-Instanz richtet Verbindungskontext ein inklusive initialer Folgenummer
  - Passiver Partner wartet mit listen aufruf an Socket Schnittstelle
  - Bei Empfang baut TCP-Instanz Connect-Response PDU (Protocol / Payload Data Unit)
  - Dabei werden SYN und ACK Flag gesetzt und Bestätigungsnummer um 1 erhöht
  - Ebenfalls wird die Folgenummer gesetzt

Verbindungsaufbau

VERBINDUNGSABBAU

- Verbindung kann theoretisch beliebig lange andauern
- Abbau geschieht ähnlich dem Aufbau in 4-Wege Handshake
  - Aktiver Partner sendet Paket mit gesetzem FIN-Flag
  - Passiver Partner antwortet mit gesetzten ACK-Flag
  - Wird beim Passiven Partner durch die Anwendung auch beendet, sented er Paket mit gesetzem FIN
  - Aktiver Partner bestätigt mit Paket mit gesetztem ACK

Verbindungsabbau

TIMER MANAGEMENT

- Timer wichtig für Performance des Protokolls
- In verschiedenen Schichten sind Verzögerungen unterschiedlich gut einzuschätzen

RETRANSMISSION TIMER

- Wichtigster Timer für Leistungsfähigkeit
- Timer started bei absenden des Paketes
- Paket nicht bestätigt vor Ablauf -> Timer verdoppelt & nochmal senden
- Timerlänge ist lastabhängig und wird nach jedem Paket neu berechnet
- Algorithmus dazu heißt Karn-Algorithmus und ist spezifiziert in RFC 6298

KEEPALIVE TIMER

- Timer wird gestartet
- Bis Ablauf des Timers keine Nachricht -> Keepalive Paket senden
- Keine Antwort nach mehreren Versuchen -> Verbindung abbauen
- Ist optional
- Standardzeit für Keepalive Timer: 2h

TIME-WAIT TIMER

- Definiert inRFC 793
- Läuft doppelte Paketlebensdauer
- Stellt sicher, dass im Netzwerk befindliche Pakete noch empfangen werden
- Nach ablauf wird Verbindungsabbau endgültig durchgeführt
- Erweitert in RFC 6191 um Zeit zu verringern, da genauere Vorhersage möglich

CLOSE-WAIT TIMER

- Gibt die Zeit an, die passive TCP-Instanz maximal wartet bis close
- Schließt nach Ablauf halboffene Verbindung
- Da eigentlich Aufgabe der Anwendungsschicht nicht explizit definiert
- Unterschiedliche Betriebssysteme behandeln die somit unterschiedlich

PERSISTENCE TIMER

- Verhindert endloses Warten nach Empfang eines TCP-Segmentes mit Fenstergröße 0 (siehe Sliding Window)
- Normalerweise muss nach Empfang eines solchen Segmentes auf ein nächstes gewartet werden
- Persistence Timer startet nach Empfang des 0-Größe Segmentes
- Läuft er ab wird ein Byte mit Nutzdaten gesendet

TCP-PROTOKOLLOPTIONEN

- Sind vollständig optional
- Wurden früher kaum genutzt
- Werden an das Ende des TCP-Headers angefügt
- Aufbau unterscheidet je nach Option
- Grober Aufbau: Typ / Länge / Optionsdaten

MAXIMUM SEGMENT SIZE OPTION

- Vereinbarung über maximale Segmentgröße
- Paket mus SYN-Flag gesetzt haben
- Kann beim Verbindungsaufbau genutzt werden
- Abkürzung: MSS
- Typ wird hierfür auf 2 gesetzt
- Standardgröße beträgt 536+20 Byte

TCP WINDOW SCALE OPTION

- Vereinbarung über maximale Fenstergröße für Sliding Window
- Zeitfenstergrößen Feld im TCP Header kann zu klein sein für Leitungen mit hoher Bandbreite
- Unabhängig von Senderichtung
- Paket muss SYN-Flag gesetzt haben
- Kann beim Verbindungsaufbau genutzt werden
- Typ wird hierfür auf 3 gesetzt
- Optionsdaten sind 3 Byte lang

SACK-PERMITTED-OPTION & SACK-OPTION

SACK PERMITTED OPTION

- Ermöglicht Veränderung am Verfahren der Übertragungswiederholung
- Standardverfahrenn ist Go-Back-N
- Typ wird hierfür auf 4 gesetzt
- Paket muss SYN-Flag gesetzt haben
- Kann beim Verbindungsaufbau genutzt werden

SACK OPTION

- Erweitert Sack Permission Option Liste von Sequenznummerbereichen mitzusenden
- Diese Bereiche beschreiben erhaltene Pakete
- Liste besteht aus Nummernpaaren (von, bis)
- Typ wird hierfür auf 4 gesetzt
- Paket muss SYN-Flag gesetzt haben
- Kann beim Verbindungsaufbau genutzt werden

TIMESTAMPS OPTION

- Hilft bei verbesserter Berechnung der Rond Trip Time
- Besteht aus Zwei Zeitstemplen mit je 4 Byte
- Erster Zeitstempel TSval (Time Stamp Value)
- Zweiter Zeitstempel TSer (Time Stamp Echo Reply)
- Typ wird hierfür auf 8 gesetzt

WRAPPED SEQUENCES

PROBLEM

- Da Wertebereich für Sequenznummern begrenzt, kann Überlauf vorkommen
- Nach Überlauf wird bei 0 wieder angefangen zu zählen
- Bei Netz mit hohem Durchsatz besteht Gefahr multipler Nutzung der selben Sequenznummer (Kollision)
- Bei einem Netzt mit 1Gbit/s könnte dies nach frühestens 34s vorkommen

PROTECT AGAINST WRAPPED SEQUENCES

- Problem wird gemildert durch fiktiven Zeitgeber bei 3-Wege Handshake
- Maximale Segmentlebensdauer wurde auf 2 Minuten verkürzt (RFC 792)
- Nach Crash oder bei neuer Verbindung muss 2*MSL abgewartet werden, bevor neue initiale Sequenznummer zugewiesen wird
- Möglichkeit der TCP-Option TSopt, zur angabe von Timestamp mit Paket

SICHERHEIT

Im Sinne der Informationssicherheit liegen keine Kontrollmechanismen vor. Diese werden sowohl der Anwenderschicht, als auch der Vermittlungsschicht überlassen.

SYN-FLOODING

- Angreifer sendet viele Connect Requests mit falschen Quelladdressen
- Täuscht somit Verbindungswünsche vor
- Angegriffener TCP-Server öffnet halboffene Verbindungen -> DoS
- Heutige Implementationen haben Timer, die Resourcen wieder freigeben
- Dies vermindert SYN-Flooding Auswirkungen, negiert aber nicht

SEQUELZNUMMERNANGRIFF

- Angreifer versucht Pakete in Verbindung einzuspeisen
- Dafür müssen Sequenz sowie Bestätigungsnummern bekannt sein
- Als gegenmaßnahme dienen Paketfilter, sowie randomisierte Initialwerte

SESSION HIGHJACKING

- Angreifer versucht Verbindung zu übernehmen
- Dazu gibt er sich als einer der Partner aus
- Angreifer kann Pakete empfangen und bestätigen, sodass Synchronisation zerstört wird

UDP

GRUNDLEGENDE EIGENSCHAFTEN UDP

- Keine Empfangsbestätigungen
- UDP Datagramme können verloren gehen
- Eingehende Pakete werden nicht sortiert
- Keine Fluss- oder Staukontrolle
- Adressierung wie bei TCP durch Socket aus Tupel mit Internetadresse und UDP-Port

VORTEILE GEGENÜBER TCP

- Keine Verbindungsaufbau Phase notwendig
- Potentiell höhere Performance
- Multi- / Broadcasting möglich

STEUERINFORMATIONEN

Header deutlich vereinfachtere Version des TCP Headers:

- **UDP-Quell-Portnummer:** Nummer des sendenden Ports
- **UDP-Ziel-Portnummer:** Nummer des empfangenden Ports
- **Länge:** Größe des UDP-Segments inklusive des Headers in Bytes (Notwendig, da keine fixe Länge)
- **Prüfsumme (optional):** Prüfsumme für Header + Daten
- **Daten:** Nutzdaten des Datagramms

DATENÜBERTRAGUNG

DATAGRAMME

- Die Kommunikation über UDP erfolg mittels Paketen, die Datagramme genannt werden.
- Datagramme können parrallel in beide Richtungen gesendet werden
- UDP übernimmt bei Bedarf (De)Segmentierung
- Bestätigungen oder Anfragen zum erneuten Senden müssen von Anwendung gesendet werden
- UDP Instanz kann an alle Rechner einer Gruppe oder eines Subnetzes senden. Wie dies in der Netzwerkschicht abläuft liegt außerhalb der Spezifikation des UDP

Prüfsummen

- UDP nutzt sehr simples Prüfverfahren
- Wird erläutert in RFCs 1071, 1141 und 1624
- Kann nur 1bit Fehler detektieren
- Es gab die Überlegung CRC als Prüfsumme zu nutzen, wurde aber nie umgesetzt

QUELLEN

- (Postel 1981a)
- (Postel 1981b)
- (Braden, Borman, and Partridge 1988)
- (Mallory and Kullberg 1990)
- (Rijsinghani (Ed.) 1994)
- (Gont 2011)
- (Paxson et al. 2011)
- (Duke et al. 2015)
- (Tanenbaum and Wetherall 2010)
- (Mandl 2018)

Braden, R.T., D.A. Borman, and C. Partridge. 1988. “Computing the Internet checksum.” RFC. Internet Request for Comments. Fremont, CA, USA: RFC Editor; RFC Editor; RFC 1071 (Informational); RFC Editor. https://doi.org/10.17487/RFC1071.

Duke, M., R. Braden, W. Eddy, E. Blanton, and A. Zimmermann. 2015. “A Roadmap for Transmission Control Protocol (TCP) Specification Documents.” RFC. Internet Request for Comments. Fremont, CA, USA: RFC Editor; RFC Editor; RFC 7414 (Informational); RFC Editor. https://doi.org/10.17487/RFC7414.

Gont, F. 2011. “Reducing the TIME-WAIT State Using TCP Timestamps.” RFC. Internet Request for Comments. Fremont, CA, USA: RFC Editor; RFC Editor;