TCP versus UDP

TCP und UDP sind beide Transportprotokolle. Sie arbeiten auf der Transportschicht und sorgen dafür, dass Daten zwischen Anwendungen auf verschiedenen Geräten übertragen werden können.

Der wichtigste Unterschied ist:

```
TCP baut vor der Datenübertragung eine Verbindung auf.
UDP sendet Daten ohne vorherigen Verbindungsaufbau direkt los.
```

TCP kurz erklärt

TCP steht für **Transmission Control Protocol**.

TCP ist:

```
- verbindungsorientiert
- zuverlässig
- geordnet
- kontrolliert
- mit mehr Verwaltungsaufwand verbunden
```

Bei TCP wird vor der Übertragung der eigentlichen Nutzdaten zuerst eine Verbindung aufgebaut. Dieser Verbindungsaufbau heißt **3-Wege-Handshake**.

```
Client                    Server

1. SYN  ---------------------->   Verbindungswunsch
2. SYN-ACK <------------------   Bestätigung + eigene Start-Sequenznummer
3. ACK  ---------------------->   Bestätigung

Danach ist die TCP-Verbindung aufgebaut.
Erst danach werden die eigentlichen Nutzdaten übertragen.
```

Wichtig:

```
Nicht ACK ist der erste Schritt.
Der erste Schritt ist SYN.
```

```
ACK ist die Bestätigung im dritten Schritt.
```

Was bedeutet SYN bei TCP?

SYN steht für:

```
Synchronize
```

Auf Deutsch:

```
synchronisieren
```

SYN bedeutet beim TCP-Verbindungsaufbau:

```
Der Client möchte eine neue TCP-Verbindung starten
und seine Start-Sequenznummer mit dem Server synchronisieren.
```

SYN ist also der Verbindungswunsch und gleichzeitig der Start der Synchronisation der Sequenznummern.

Was bedeutet SYN-ACK bei TCP?

SYN-ACK besteht aus zwei Teilen:

```
SYN = Server möchte ebenfalls seine Start-Sequenznummer synchronisieren
ACK = Server bestätigt den SYN des Clients
```

Der Server sagt damit vereinfacht:

```
Ich habe deinen Verbindungswunsch erhalten.
Ich bin bereit.
Hier ist meine eigene Start-Sequenznummer.
```

Was bedeutet ACK bei TCP?

ACK steht für:

```
Acknowledgement
```

Auf Deutsch:

```
Bestätigung
```

Mit ACK bestätigt eine Seite, dass bestimmte Daten oder Steuerinformationen angekommen sind.

Beim 3-Wege-Handshake bestätigt der Client mit ACK die Antwort des Servers.

Merksatz:

```
SYN     = Verbindung starten und Sequenznummer synchronisieren
SYN-ACK = Verbindung bestätigen und eigene Startnummer senden
ACK     = Bestätigung zurücksenden
```

UDP kurz erklärt

UDP steht für **User Datagram Protocol**.

UDP ist:

```
- verbindungslos
- einfach
- schnell bzw. oft latenzärmer
- mit wenig Verwaltungsaufwand verbunden
- ohne eingebaute Zustellgarantie
- ohne eingebaute Reihenfolgekontrolle
- ohne automatische erneute Übertragung verlorener Daten
```

Bei UDP gibt es keinen 3-Wege-Handshake.

```
Client                  Server

Daten ---------------------->

Kein SYN
Kein SYN-ACK
```

```
Kein verbindungsaufbauendes ACK
```

UDP sendet Datagramme direkt los. UDP selbst prüft nicht zuverlässig, ob die Gegenseite bereit ist oder ob die Daten vollständig angekommen sind.

Nutzdaten bei TCP und UDP

Nutzdaten sind die eigentlichen Inhalte, die eine Anwendung übertragen möchte.

Beispiele:

```
- Webseiteninhalte
- Dateien
- Sprache
- Videodaten
- DNS-Anfragen
- Spielinformationen
```

Bei TCP gilt:

```
Erst Verbindungsaufbau,
dann Nutzdaten.
```

Bei UDP gilt:

```
Kein Verbindungsaufbau,
Nutzdaten werden direkt als Datagramm gesendet.
```

Sequenznummern bei TCP

TCP verwendet Sequenznummern.

Eine Sequenznummer ist keine Portnummer.

```
Portnummer     = welcher Dienst?
Sequenznummer = welche Stelle im Datenstrom?
```

TCP nummeriert technisch gesehen die Bytes im Datenstrom. Dadurch kann der Empfänger erkennen:

```
- welche Daten angekommen sind
- ob Daten fehlen
- ob Daten doppelt angekommen sind
- ob Daten in falscher Reihenfolge angekommen sind
```

Beispiel:

```
Segment 1: Sequenznummer 1000, enthält 500 Byte
Segment 2: Sequenznummer 1500, enthält 500 Byte
Segment 3: Sequenznummer 2000, enthält 500 Byte
```

Wenn Segment 2 fehlt, merkt TCP:

```
Nach 1000 müsste 1500 kommen.
Wenn schon 2000 kommt, fehlt der Bereich ab 1500.
```

Fehlende Daten können dann erneut übertragen werden.

UDP hat keine TCP-Sequenznummern

UDP nummeriert die Daten nicht wie TCP mit Sequenznummern.

UDP sendet einzelne Datagramme.

```
Datagramm 1 wird gesendet.
Datagramm 2 wird gesendet.
Datagramm 3 wird gesendet.
```

UDP selbst merkt sich dabei keine Reihenfolge.

Wenn ein Datagramm verloren geht, fordert UDP es nicht automatisch erneut an.

Wenn Datagramme in falscher Reihenfolge ankommen, sortiert UDP sie nicht automatisch.

Zuverlässigkeit

TCP ist zuverlässig, weil es mehrere Kontrollmechanismen verwendet:

- Verbindungsaufbau durch 3-Wege-Handshake
- Sequenznummern
- ACK-Bestätigungen
- erneute Übertragung verlorener Daten
- Reihenfolgekontrolle
- Erkennung doppelt empfangener Daten
- Flusskontrolle

UDP hat diese Zuverlässigkeit nicht eingebaut.

UDP garantiert nicht:

- dass Daten ankommen
- dass Daten vollständig ankommen
- dass Daten in der richtigen Reihenfolge ankommen
- dass verlorene Daten erneut übertragen werden

Wenn eine Anwendung über UDP trotzdem Zuverlässigkeit benötigt, muss sie diese selbst einbauen.

Beispiele:

- eigene Bestätigungen
- eigene Sequenznummern
- eigene Wiederholungen
- Fehlerkorrektur auf Anwendungsebene

Geschwindigkeit und Verwaltungsaufwand

TCP hat mehr Verwaltungsaufwand, weil es Verbindungen aufbaut, Daten bestätigt, Reihenfolgen prüft und verlorene Daten erneut überträgt.

UDP hat weniger Verwaltungsaufwand, weil es diese Funktionen nicht eingebaut hat.

Deshalb ist UDP oft latenzärmer.

Wichtig:

UDP ist nicht automatisch immer schneller im Sinne von höherer Datenrate.
UDP hat aber weniger Verwaltungsaufwand und kann dadurch schneller reagieren.

Besser formuliert:

```
UDP ist oft latenzärmer als TCP,
weil es keinen Verbindungsaufbau und weniger Kontrollmechanismen hat.
```

Typische Anwendungen für TCP

TCP wird verwendet, wenn Daten zuverlässig und vollständig ankommen müssen.

Beispiele:

```
- HTTP/HTTPS über TCP
- SSH
- E-Mail
- Dateiübertragung
- Remote Desktop
- Datenbankverbindungen
```

Hier wäre es problematisch, wenn Daten fehlen oder in falscher Reihenfolge bei der Anwendung ankommen.

Beispiel:

```
Bei einer Dateiübertragung darf kein Teil der Datei fehlen.
Bei SSH müssen Befehle korrekt und in richtiger Reihenfolge ankommen.
Bei Webseiten sollen Inhalte vollständig übertragen werden.
```

Typische Anwendungen für UDP

UDP wird häufig verwendet, wenn geringe Verzögerung wichtiger ist als vollständige Kontrolle.

Beispiele:

```
- DNS
- DHCP
- VoIP
- Live-Streaming
- Online-Gaming
- NTP
```

```
- WireGuard
- QUIC / HTTP/3
```

Beispiel VoIP:

```
Bei einem Telefonat ist es besser,
wenn ein kleines Audiostück kurz fehlt,
als wenn die Sprache stark verzögert ankommt.
```

Beispiel Online-Gaming:

```
Bei schnellen Positionsdaten ist der aktuelle Zustand wichtiger
als ein altes verlorenes Paket nachträglich zu übertragen.
```

DNS als Beispiel für UDP

DNS nutzt sehr häufig UDP auf Port 53.

Ablauf vereinfacht:

```
1. Client fragt per UDP beim DNS-Server an.
2. DNS-Server antwortet per UDP.
3. Die Antwort enthält die passende IP-Adresse zur Domain.
```

Beispiel:

```
Client:     192.168.0.20:53000
DNS-Server: 192.168.0.1:53
Protokoll:  UDP
```

Warum UDP hier sinnvoll ist:

```
- DNS-Anfragen sind meist klein.
- Ein TCP-Verbindungsaufbau wäre zusätzlicher Aufwand.
- Bei Verlust kann die Anfrage einfach erneut gestellt werden.
```

Wichtig:

DNS kann auch TCP verwenden,
zum Beispiel bei größeren Antworten oder Zonentransfers.

QUIC und HTTP/3

QUIC ist ein modernes Transportprotokoll, das auf UDP basiert.

HTTP/3 läuft über QUIC.
QUIC läuft über UDP.
UDP läuft über IP.

Wichtig:

UDP selbst ist einfach und verbindungslos.
QUIC baut darauf eigene Funktionen für Verbindung, Zuverlässigkeit und Verschlüsselung auf.

Das bedeutet:

Nur weil UDP selbst keine TCP-Zuverlässigkeit bietet,
kann ein Protokoll oberhalb von UDP trotzdem eigene Kontrollmechanismen einbauen.

TCP und UDP mit Ports

Sowohl TCP als auch UDP verwenden Ports.

IP-Adresse = welcher Host?
Port = welcher Dienst?
Socket = IP-Adresse + Port

Wichtig:

TCP-Port 443 und UDP-Port 443 sind technisch getrennt.

Beispiel:

```
TCP 443 = HTTPS über TCP
UDP 443 = QUIC / HTTP/3
```

Ein Dienst kann also denselben Port bei TCP und UDP unterschiedlich verwenden.

Firewall-Bezug

Firewalls können sowohl TCP als auch UDP filtern.

Dabei prüfen sie zum Beispiel:

```
- Quell-IP
- Ziel-IP
- Quellport
- Zielport
- Protokoll TCP oder UDP
```

Bei TCP kann eine Stateful Firewall den Verbindungszustand gut erkennen:

```
SYN → SYN-ACK → ACK → Verbindung steht
```

Bei UDP gibt es keinen echten Verbindungszustand wie bei TCP. Eine Stateful Firewall kann sich aber trotzdem kurzzeitig merken, dass ein internes Gerät ein UDP-Datagramm nach außen gesendet hat.

Beispiel:

```
Client sendet DNS-Anfrage per UDP nach außen.
Firewall merkt sich diese Anfrage kurzzeitig.
DNS-Antwort darf zurück.
Unerwartete UDP-Pakete von außen werden blockiert.
```

TCP versus UDP als Tabelle

Merkmal	TCP	UDP
Voller Name	Transmission Control Protocol	User Datagram Protocol
Verbindungsaufbau	Ja, 3-Wege-Handshake	Nein
Erste Nachricht	SYN	Direkt Datagramm/Nutzdaten

Merkmal	TCP	UDP
Zuverlässigkeit	Eingebaut	Nicht eingebaut
Reihenfolgekontrolle	Ja	Nein
Sequenznummern	Ja	Nicht wie TCP
ACK-Bestätigungen	Ja	Nicht durch UDP selbst
Erneute Übertragung	Ja	Nein
Verwaltungsaufwand	Höher	Geringer
Latenz	Oft höher	Oft geringer
Typische Nutzung	Zuverlässige Datenübertragung	Echtzeit oder kleine schnelle Anfragen
Beispiele	HTTPS, SSH, E-Mail, Dateiübertragung	DNS, VoIP, Streaming, Gaming, QUIC

Einfacher Vergleich

TCP ist wie ein Einschreiben:

```
Es wird geprüft, bestätigt und bei Problemen erneut gesendet.
```

UDP ist wie eine Postkarte:

```
Sie wird direkt abgeschickt, aber es gibt keine eingebaute Garantie,
dass sie ankommt oder in welcher Reihenfolge sie ankommt.
```

IHK-sichere Kurzformulierung

TCP ist ein verbindungsorientiertes und zuverlässiges Transportprotokoll. Vor der Datenübertragung wird über den 3-Wege-Handshake eine Verbindung aufgebaut. TCP verwendet Sequenznummern, ACK-Bestätigungen, Reihenfolgekontrolle und erneute Übertragung verlorener Daten. Dadurch eignet sich TCP für Anwendungen, bei denen Daten vollständig und korrekt ankommen müssen.

UDP ist ein verbindungsloses Transportprotokoll mit geringem Verwaltungsaufwand. UDP baut vor dem Senden keine Verbindung auf und bietet keine eingebaute Garantie für Zustellung, Reihenfolge oder erneute Übertragung verlorener Daten. Dadurch eignet sich UDP besonders für Anwendungen, bei denen geringe Verzögerung wichtiger ist als vollständige Kontrolle.

Merksätze

TCP = erst Verbindung aufbauen, dann Nutzdaten senden

UDP = keine Verbindung aufbauen, Daten direkt senden

TCP = zuverlässig, geordnet und kontrolliert

UDP = verbindungslos, schlank und oft latenzärmer

TCP fragt vorher:

"Darf ich eine Verbindung aufbauen?"

UDP sendet direkt:

"Hier sind die Daten."

TCP nutzt SYN, SYN-ACK und ACK für den Verbindungsaufbau.

UDP hat keinen 3-Wege-Handshake.

TCP erkennt fehlende Daten und überträgt sie erneut.

UDP macht das nicht automatisch.

TCP eignet sich für vollständige Daten.

UDP eignet sich für schnelle oder zeitkritische Daten.

Revision #2
Created 1 June 2026 07:52:41 by Admin
Updated 3 June 2026 06:18:12 by Admin