TCP & UDP Netzwerkbasics mit Linux erkunden

21. Augsburger Linux-Infotag

26. April 2025

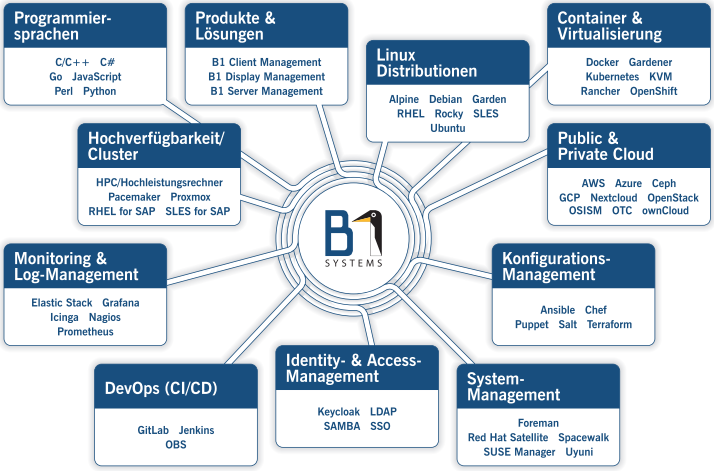
Susanne Schütze
Linux Consultant
B1 Systems GmbH
schuetze@b1-systems.de

Inhaltsverzeichnis

Vorstellung B1 Systems

- gegründet 2004
- spezialisiert auf Linux/Open Source-Themen
- national & international tätig
- ca. 150 Mitarbeiter:innen
- unabhängig von Soft- & Hardware-Herstellern
- Leistungsangebot:
  - Managed Service & Betrieb
  - Beratung & Consulting
  - Support
  - Training
  - Lösungen & Entwicklung
- Standorte in Rockolding, Köln, Berlin & Dresden

Schwerpunkte



whoami

- Susanne Schütze
- 41 Jahre
- Fachinformatikerin für Systemintegration
- bei B1 Systems GmbH seit Juli 2024
- berufliche Themen: Linux Client Management, Development, Ansible, Salt, etc.

Netzwerk-Protokolle und wo sie zu finden sind

Was sind Netzwerk-Protokolle?

- standardisierte Abläufe für die Kommunikation in einem Netzwerk
- die Standardisierung ist in RFCs (*Request For Comments*) festgehalten
- RFCs in diesem Vortrag:
  - RFC: 768, 8085 UDP
  - RFC: 791, 1349, 2474, 6864 IPv4
  - RFC: 4349, 5641 High-Level Data Link Control Frames
  - RFC: 5841 TCP Option to Denote Packet Mood
  - RFC: 6217 Regional Broadcast Using Atmospheric Link Layer
  - RFC: 8200 IPv6
  - RFC: 9293 TCP
  - RFC: 9401 The Addition of Death (DTH) Flag to TCP

Wichtiger Hinweis: Alle am 1.4. herausgegebenen RFCs sind grün markiert.

Open Systems Interconnection Model

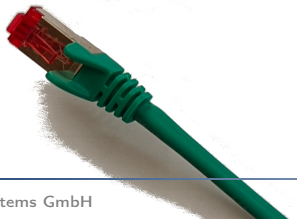
- Netzwerke sind in Schichten aufgebaut
- diese Schichten werden mit einem Modell erklärt
  $\Rightarrow$ OSI (Open Systems Interconnection) Modell
- spezifiziert nach ISO/IEC 7498-1
- besteht aus 7 Schichten/Layern

Open Systems Interconnection Model

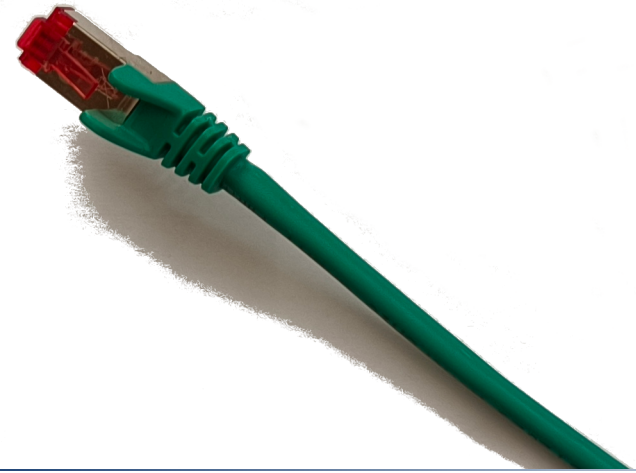
- Netzwerke sind in Schichten aufgebaut
- diese Schichten werden mit einem Modell erklärt
  $\Rightarrow$ OSI (Open Systems Interconnection) Modell
- spezifiziert nach ISO/IEC 7498-1
- besteht aus 8-10 Schichten/Layern

Bitübertragungsschicht / Physical Layer

- verantwortlich für Empfang und Versendung von Daten
- elektrische, Funk- oder optische Signalübertragung
- transportiert unsortierte Rohdaten
- Beispiel-Technologien:
  - WLAN
  - LAN
  - USB
  - Bluetooth



Bitübertragungsschicht / Physical Layer



Physical Layer unter Linux

Anzeige der verschiedenen möglichen Physical Layer

```
$ tcpdump -D
1.enp0s31f6 [Up, Running, Connected]
2.any (Pseudo-device that captures on all interfaces) [Up, Running]
3.lo [Up, Running, Loopback]
4.wlp0s20f3 [Up, Wireless, Not associated]
5.bluetooth0 (Bluetooth adapter number 0) [Wireless, Association
  ↪  status unknown]
6.bluetooth-monitor (Bluetooth Linux Monitor) [Wireless]
7.usbmon4 (Raw USB traffic, bus number 4)
8.usbmon3 (Raw USB traffic, bus number 3)
9.usbmon2 (Raw USB traffic, bus number 2)
10.usbmon1 (Raw USB traffic, bus number 1)
```

Atmospheric Link Layer 1/2

RFC: 6217 alternativer Link Layer

- Beschaffenheit:
  - bestehend aus Stickstoff und Sauerstoff, kann auch Spurenelemente von Argon, Kohlendioxid, Neon, Helium und Schwefeldioxid enthalten
  - üblicherweise ist ein gewisser Teil gasförmiges Dihydrogenmonoxid enthalten
  - Druckverhältnisse von 101,3 Kilopascal auf Meereshöhe
- Ort:
  - in Höhe von 7000-17000 Feet (2133-5181 Meter)
- methodische Umsetzung:
  - Skytyping-Methode stellt Haltbarkeit des Datagramms bei Luftbewegungen sicher
- Chemischer Ansatz:
  - Informationen werden durch Produzieren einer speziellen Form von Rauch mittels Austreten eines Spezialöls durch einen Auspuff realisiert
  - Spezialöl wird in Druckbehältern gelagert und verdampft zu dickem weißem Rauch, dieser produziert eine lesbare Anzeige

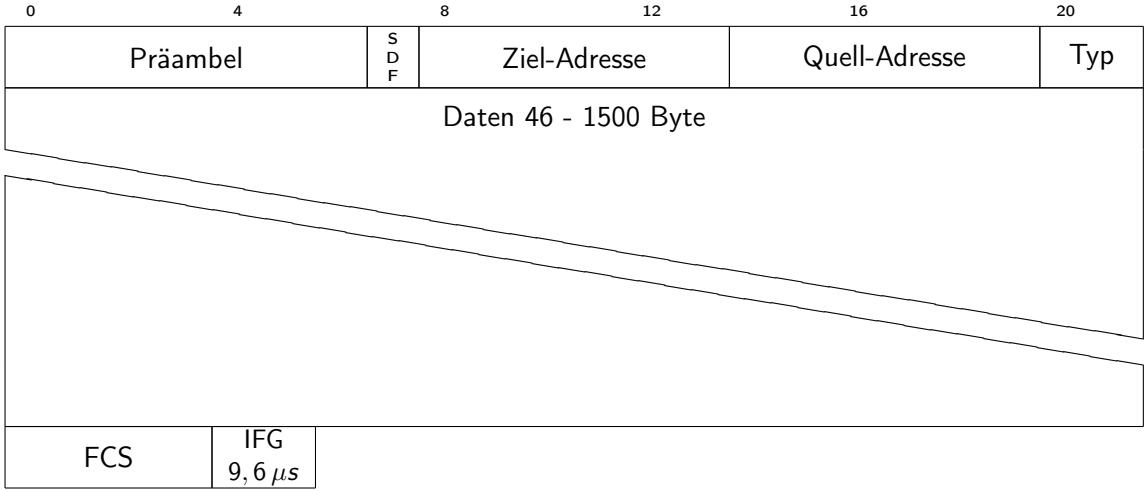
Atmospheric Link Layer 2/2

RFC: 6217

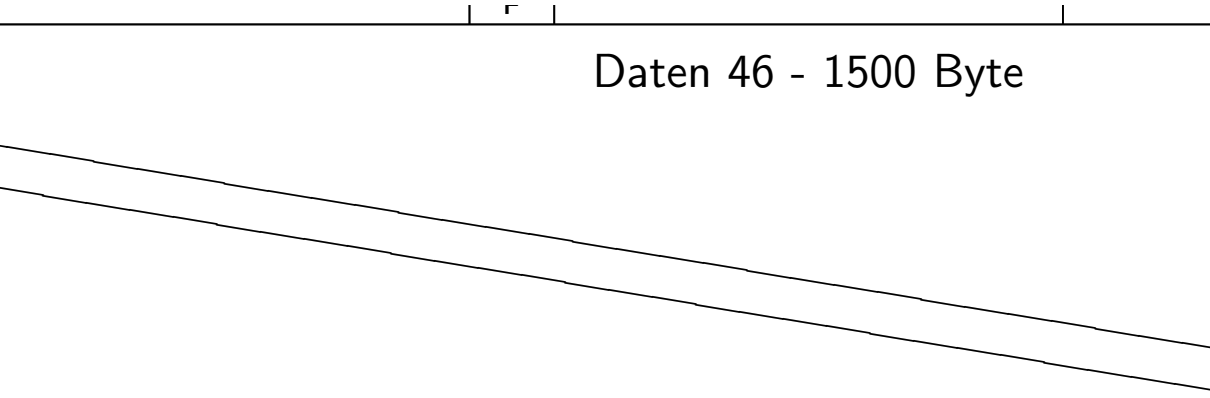
Wie alle Physical Layers hat auch der Atmospheric Link Layer Vor- und Nachteile, abhängig vom Zustand, Wartung, Verbreitung und wirtschaftlichen Faktoren

- bekannte Beeinträchtigungen:
  - ungünstige Gegebenheiten, wie die Ansammlung von Feuchtigkeit, Eiskristallen resultieren in zeitlich begrenzter Übertragungsverhinderung
  - zusätzliche von Sonneneinstrahlung beeinflusste Übertragungen zyklisch vorhersagbar
  - abhängig vom Längengrad Medium für längere Zeit unbenutzbar
  - Übertragungsbedingungen können sich vor, während oder nach der Übertragung verschlechtern, resultiert in erhöhten Fehlerraten
- Vorteile:
  - verhindert erheblich Überlastung des Netzwerk-Equipments
  - keine zusätzlichen Investitionen in physische Infrastruktur erforderlich
  - dieser Data Link Layer skaliert effizient zu dem geographischen LAN und – abhängig von der Sicht – zu einem MAN

Ethernet Frames



Ethernet Frames



Sicherungsschicht / Data Link Layer

- verantwortlich für zuverlässige und weitgehend fehlerfreie Übertragung sowie den Zugriff auf das Übertragungsmedium
- besteht aus zwei Sublayern:
  - LLC (*Logical Link Control*)
  - MAC (*Media Access Control*)
- überträgt die Roh-Daten aus Layer 1 in Frames
- Beispiel: HDLC (*High Level Data Link Control*)-Protokoll

Ethernet-Pakete unter Linux zählen

Tools zu Statistik und Bandbreite

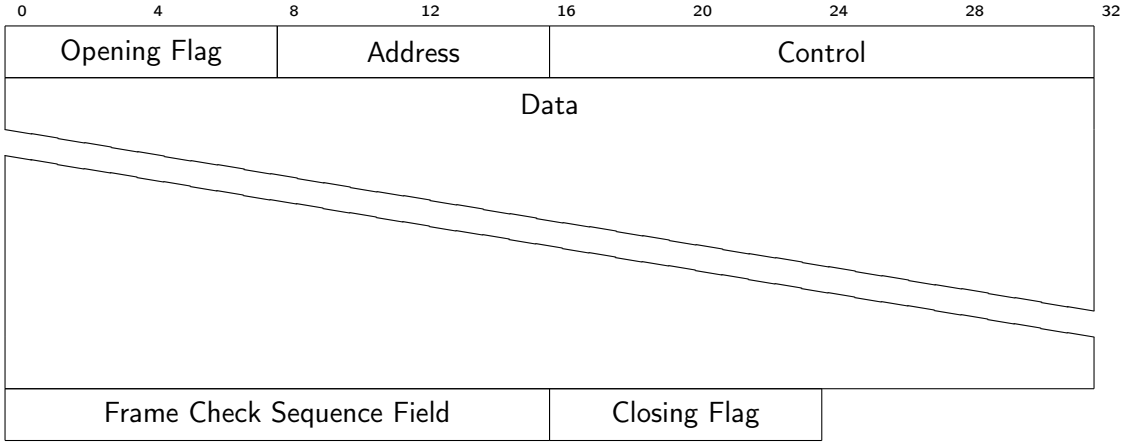
- `bmon`
- `slurm` (bunte Ausgabe)
- `iftop`
- `nethogs`(Bandbreite pro Prozess)
- `nload`
- `ifstat`
- `tcpdump`, `ptcpdump`, `tshark`
  - Packet-Sniffer
  - Layer 2 nur rudimentär abbildbar

Mit Bordmitteln

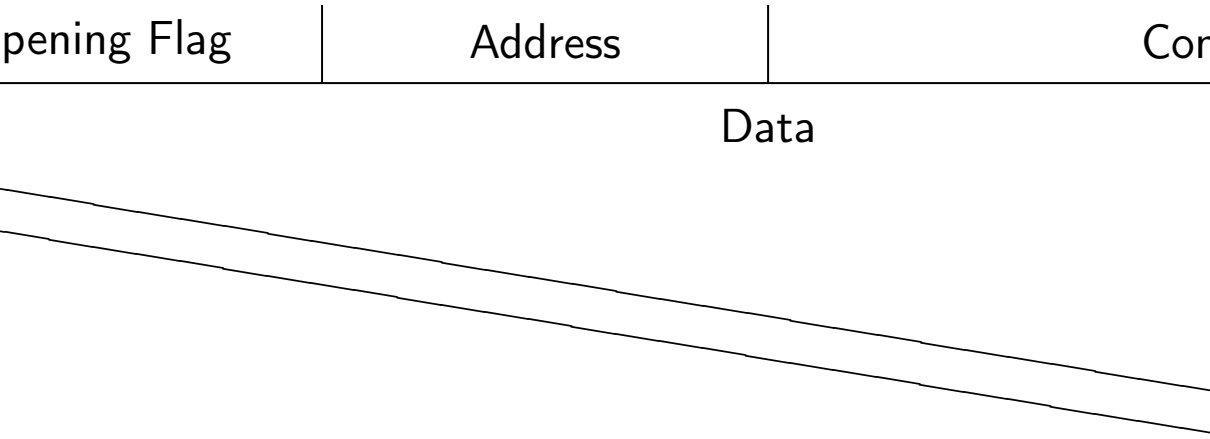
```
1 $ ip -s -h link
2 $ cat /proc/net/dev
```

HDLC-Paket

RFC: 4349



HDLC-Paket



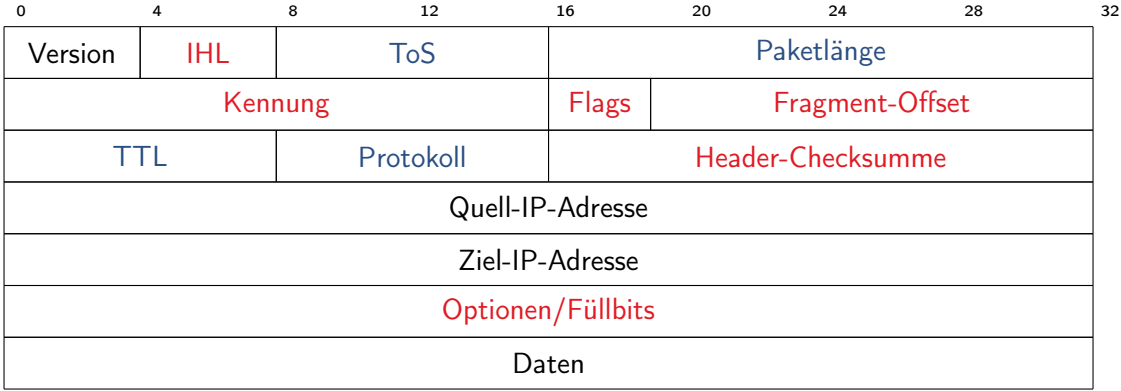
Network Layer/Vermittlungsschicht

- verantwortlich für das Verteilen der Pakete im Netz, findet den besten Pfad
- sorgt für mehr oder weniger eindeutige Adressierung mit IP-Adressen
- setzt die Default-Route für das Gateway (wenn vorhanden)
- teilt Pakete auf, um sie zu verschicken, und fügt sie beim Empfang wieder zusammen
- kann auch mit Brieftauben IPoAC (RFC 1149 / 6214) realisiert werden

`https://media.ccc.de/v/`
`clt24-120-ipv6-grundlagen-in-zusammenhang-mit-ipoac`

Legacy IPv4-Paket

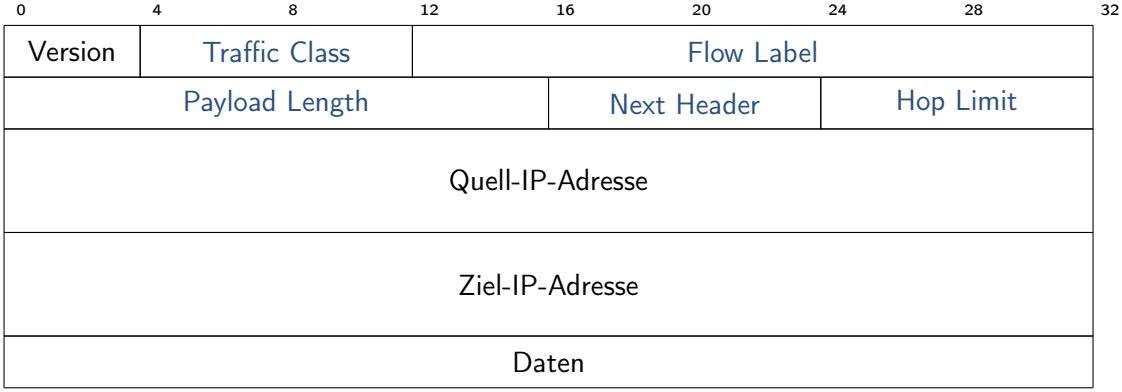
Größe: 20 Byte



Legende: IPv6: Rot fällt weg, Blau ändert sich

IPv6 seziert

RFC 8200 (2017), ursprünglich RFC 1883 (1995); Größe: 40 Byte



Legende: Blau verändert in IPv6

IPv6 unter Linux 1/2

Ein paar IPv6-Befehle

```
$ ip -6 address show
$ ip -6 maddress show # Multicast-Adressen
$ ip -6 neighbour show
$ ip -6 route show table all
```

Vorbereitung zum Anschauen eines IPv6-Frames

```
# tcpdump -s0 -i eth0 -c 100 -w tcpdump.pcap
$ tshark -r tcpdump.pcap -O "ipv6" -Y "frame.number==9"
```

IPv6 unter Linux 2/2

IPv6-Frame

```
Frame 9: 86 bytes on wire (688 bits), 86 bytes captured (688 bits)
Internet Protocol Version 6, Src: 2001:XXXX::5d32:16cd:721d:c060, Dst: 2001:67c:1400:1010::37
    0110 .... = Version: 6
    .... 0000 0000 .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)
        .... 0000 00.. .... = Differentiated Services Codepoint: Default (0)
        .... .... ..00 .... = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    .... 1001 1011 1000 0101 0101 = Flow Label: 0x9b855
    Payload Length: 32
    Next Header: TCP (6)
    Hop Limit: 255
    ...
    Source Address: 2001:XXXX:XXXX:XXXX:5d32:16cd:721d:c060
        [Address Space: Global Unicast]
    Destination Address: 2001:67c:1400:1010::37
        [Address Space: Global Unicast]
...
    [Stream index: 1]
Transmission Control Protocol, Src Port: 33162, Dst Port: 443, Seq: 1, Ack: 542, Len: 0
```

Transportschicht / Transport Layer

- Aufteilung der Daten in Sequenzen
- funktionale sichere End-to-End-Übertragung
- stellt die Reihenfolge der Daten sicher
- reguliert und optimiert den Datenverkehr
- kann bei Engpässen den Datenfluss reduzieren
- hat eigene Adresse (Ports)

Was ist UDP?

Definition

Das *User Datagram Protocol* (UDP) ist ein verbindungsloses Netzwerkprotokoll zur Übertragung von Daten. RFC 3828 (2004 UDP-Lite).

- nicht zuverlässig (Pakete in der Reihenfolge können fehlen)
- ungesichert (Pakete können verloren gehen)
- ungeschützt (Pakete können verfälscht und mitgelesen werden)
- wenn alles schläft und einer spricht...wird UDP benutzt

Vorteile von UDP:

- ziemlich schnell
- braucht weniger Netzwerk-Kapazitäten
- keine End-to-End-Verbindung

Was ist UDP?

Definition

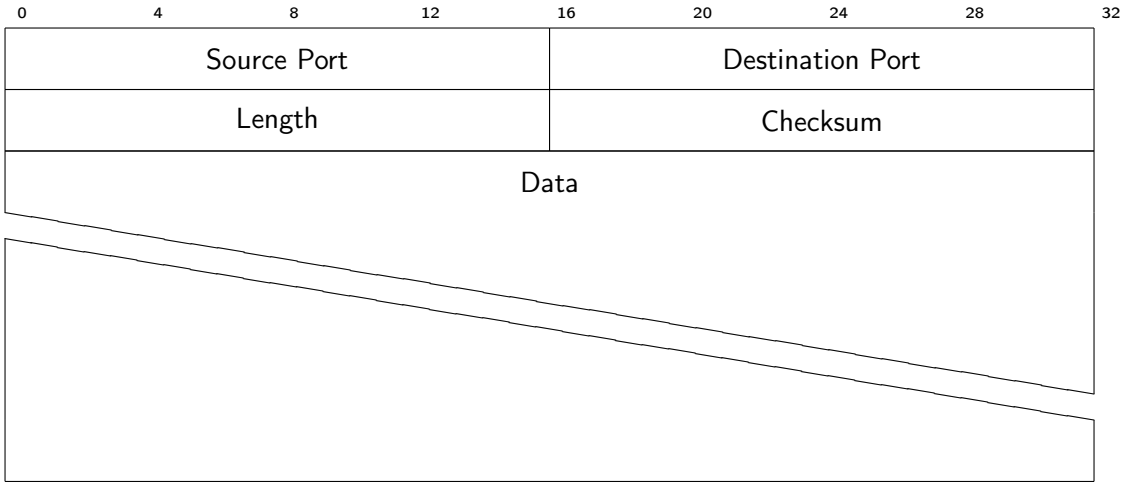
Das *User Datagram Protocol* (UDP) ist ein verbindungsloses Netzwerkprotokoll zur Übertragung von Daten. RFC 3828 (2004 UDP-Lite).

- nicht zuverlässig (Pakete in der Reihenfolge können fehlen)
- ungesichert (Pakete können verloren gehen)
- ungeschützt (Pakete können verfälscht und mitgelesen werden)
- wenn alles schläft und einer spricht...wird UDP benutzt

Vorteile von UDP:

- ziemlich schnell
- braucht weniger Netzwerk-Kapazitäten
- keine End-to-End-Verbindung

Das Innere der UDP-Pakete



UDP unter Linux 1/2

Übersicht über UDP mit ss

```
# ss --udp --all --processes --ipv6
State  Recv-Q Send-Q   Local Address:Port  Peer Address:Port  Process
UNCONN  0   0 [fe80::779d:9ca2:6810:293b]%enp0s31f6:ws-discovery
  ↪  [::]:*  users:(("wsdd",pid=4195,fd=12))
UNCONN  0   0 [ff02::c]%enp0s31f6:ws-discovery  [::]:*
  ↪  users:(("wsdd",pid=4195,fd=10))
UNCONN  0   0         *:54327  *:*  users:(("wsdd",pid=4195,fd=11))
UNCONN  0   0         [::]:mdns       [::]:*
UNCONN  0   0         [::]:llmnr      [::]:*
UNCONN  0   0         [::1]:323       [::]:*
UNCONN  0   0         [::]:33214      [::]:*
```

UDP unter Linux 2/2

UDP-Frame: `tshark -r ipv6dump.pcap "udp" -Y "frame.number==20"`

```
1 Frame 20: 162 bytes on wire (1296 bits), 162 bytes captured (1296 bits)
2 Internet Protocol Version 6, Src: 2001:XXXX:XXXX:XXXX:5d32:16cd:721d:c060, Dst:
  ↪ 2001:67c:1405:1010::5
3 User Datagram Protocol, Src Port: 59630, Dst Port: 10000
4     Source Port: 59630
5     Destination Port: 10000
6     Length: 108
7     Checksum: 0x7675 [unverified]
8     [Checksum Status: Unverified]
9     [Stream index: 0]
10    [Stream Packet Number: 13]
11    [Timestamps]
12        [Time since first frame: 4.831407000 seconds]
13        [Time since previous frame: 0.220012000 seconds]
14    UDP payload (100 bytes)
```

TCP

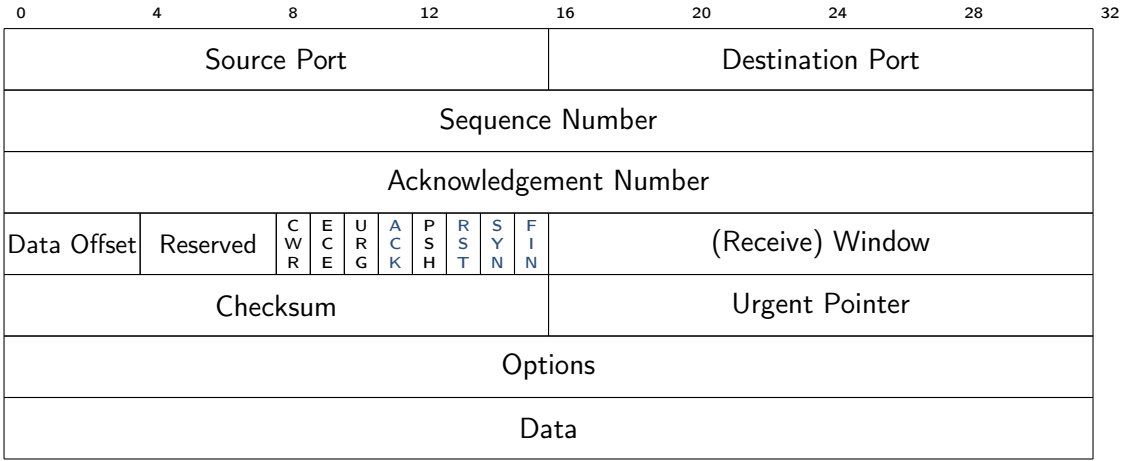
Definition

Das *Transmission Control Protocol* (TCP) ist ein verbindungsorientiert arbeitendes Protokoll. RFC 675, 9293 (Update 2022) und RFC 5841

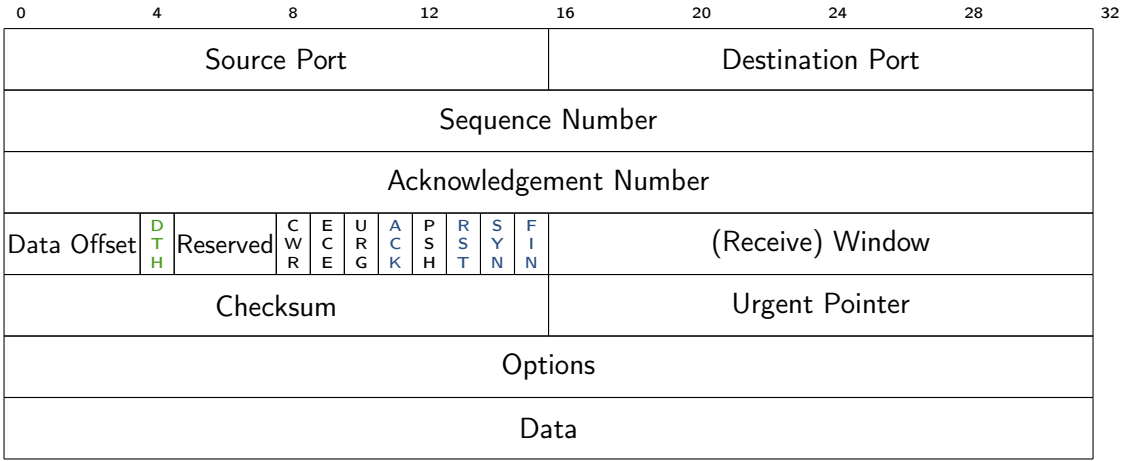
- Verbindungen werden auf- und abgebaut
- Datenübertragung unterliegt Flusskontrolle und Fehlererkennung
- Verbindung kann gleichberechtigt in beide Richtungen genutzt werden

TCP ist relativ komplex und umfangreich. Deswegen kann hier nicht auf alle Details eingegangen werden.

Das Innere der TCP-Pakete



Das Innere der TCP-Pakete mit DTH



Der Verbindung über TCP

Bob →	SYN	„Darf ich mit dir reden?“	Alice	Verbindungsaufbau
Bob	SYN ACK	„Sicher, was ist?“ (Wenn Alice keine Zeit hat schickt sie ein RST ACK)	← Alice	
Bob →	ACK, Daten	„Gut ich höre dich, . . . “	Alice	Verbindung Established

Drei-Wege-Handshake

Der Verbindung über TCP

Bob →	SYN	„Darf ich mit dir reden?“	Alice	Verbindungsaufbau
Bob	SYN ACK	„Sicher, was ist?“ (Wenn Alice keine Zeit hat schickt sie ein RST ACK)	← Alice	
Bob →	ACK, Daten	„Gut ich höre dich, . . . “	Alice	Verbindung Established

Drei-Wege-Handshake

Der Verbindung über TCP

Bob →	SYN	„Darf ich mit dir reden?“	Alice	Verbindungsaufbau
Bob	SYN ACK	„Sicher, was ist?“ (Wenn Alice keine Zeit hat schickt sie ein RST ACK)	← Alice	
Bob →	ACK, Daten	„Gut ich höre dich, . . . “	Alice	Verbindung Established

Drei-Wege-Handshake

TCP unter Linux

Übersicht über TCP mit ss

```
# ss --tcp --all --processes --ipv6
State Recv-Q Send-Q   Local Address:Port  Peer Address:Port    Process
LISTEN  0    4096       [::1]:ipp          [::]:*
LISTEN  0    4096       [::]:llmnr         [::]:*
ESTAB   0     0  [2001:4091:a245:8412:6edc:1299:344e:7739]:56388
  ↪  [2a04:4e42:8e::347]:https
  ↪  users:(("gnome-software",pid=2947,fd=41))
```

Unterschiedliche Gefühle bei TCP

Im RFC 5841 sind alle TCP-Gefühle und ihre Ursachen genauestens spezifiziert.

- :) Happy (Pakete, die mit ACK bestätigt werden)
- :(Sad (erneute Übertragung von Paketen > als 20%)
- :D Amused (Pakete die einen Witz enthalten)
- % (Confused (Antworten auf Out-of-Order Pakete, Pakete im falschen VLAN, falsch geroutete, oder enthaltene Daten sind komplexe philosophische Fragen)
- :o Bored (Pakete mit Buchhaltungsdaten, Schulden, Gutschriften, Telefonbucheinträgen)
- :O Surprised (unerwartete Fehler/Bedienung (unreachable messages))
- :P Silly (zufällige Keepalive-Pakete im Schlagabtausch)
- :@ Frustrated (Pakete, die mehrfach erneut gesendet wurden)
- >:@ Angry (Pakete, die erneut gesendet wurden)
- :| Apathetic (Pakete mit RST)
- ;) Sneaky (Pakete mit cleverem Zusammenhang)
- >:) Evil (Entwickler entscheiden dies durch ihre Sicht auf die Welt)

SYN 1/2

Happy-Paket

```
1 Transmission Control Protocol, Src Port: 35232, Dst Port: 22, Seq: 0,
  ↪ Len: 0
2     Source Port: 35232
3     Destination Port: 22
4     Sequence Number (raw): 1628654193
5     Acknowledgment Number: 0
6     Acknowledgment number (raw): 0
7     Flags: 0x002 (SYN)
8         000. .... .... = Reserved: Not set
9         ...0 .... .... = Accurate ECN: Not set
10        .... 0... .... = Congestion Window Reduced: Not set
11        .... .0.. .... = ECN-Echo: Not set
```

SYN 2/2

Happy-Paket

```
12          .... ..0. .... = Urgent: Not set
13          .... ...0 .... = Acknowledgment: Not set
14          .... .... 0... = Push: Not set
15          .... .... .0.. = Reset: Not set
16          .... .... ..1. = Syn: Set
17      Window: 64440
18      Checksum: 0xa135 [unverified]
19      Urgent Pointer: 0
20      Options: (20 bytes), Maximum segment size, SACK permitted,
        ↪  Timestamps, No-Operation (NOP), Window scale
21          :)
```

RST 1/2

Apathetic-Paket

```
1 Transmission Control Protocol, Src Port: 6000, Dst Port: 58894, Seq:
  ↪ 1, Ack: 1, Len: 0
2     Source Port: 6000
3     Destination Port: 58894
4     Sequence Number (raw): 0
5     Acknowledgment Number: 1    (relative ack number)
6     Acknowledgment number (raw): 3369500539
7     Flags: 0x014 (RST, ACK)
8         000. .... .... = Reserved: Not set
9         ...0 .... .... = Accurate ECN: Not set
10        .... 0... .... = Congestion Window Reduced: Not set
11        .... .0.. .... = ECN-Echo: Not set
```

RST 2/2

Apathetic-Paket

```
12          .... ..0. .... = Urgent: Not set
13          .... ...1 .... = Acknowledgment: Set
14          .... .... 0... = Push: Not set
15          .... .... .1.. = Reset: Set
16      Window: 0
17      Checksum: 0xc8f8 [unverified]
18      Urgent Pointer: 0
19      Options:
20          :|
```

SYN ACK 1/2

Happy-Paket

```
1 Transmission Control Protocol, Src Port: 22, Dst Port: 35232, Seq: 0,
  ↪ Ack: 1, Len: 0
2     Source Port: 22
3     Destination Port: 35232
4     Sequence Number (raw): 596937522
5     Acknowledgment Number: 1    (relative ack number)
6     Acknowledgment number (raw): 1628654194
7     Flags: 0x012 (SYN, ACK)
8         000. .... .... = Reserved: Not set
9         ...0 .... .... = Accurate ECN: Not set
10        .... 0... .... = Congestion Window Reduced: Not set
11        .... .0.. .... = ECN-Echo: Not set
```

SYN ACK 2/2

Happy-Paket

```
12          .... ..0. .... = Urgent: Not set
13          .... ...1 .... = Acknowledgment: Set
14          .... .... 0... = Push: Not set
15          .... .... .0.. = Reset: Not set
16          .... .... ..1. = Syn: Set
17      Window: 64260
18      Checksum: 0x65c4 [unverified]
19      Urgent Pointer: 0
20      Options: (20 bytes), Maximum segment size, SACK permitted,
        ↪  Timestamps, No-Operation (NOP), Window scale
21          :)
```

Out of Order 1/2

Confused-Paket

```
1 Transmission Control Protocol, Src Port: 443, Dst Port: 56388, Seq:
  ↪ 52, Ack: 1, Len: 24
2     Source Port: 443
3     Destination Port: 56388
4     Sequence Number (raw): 3263481265
5     Acknowledgment Number: 1    (relative ack number)
6     Acknowledgment number (raw): 1563477320
7     Flags: 0x018 (PSH, ACK)
8         000. .... .... = Reserved: Not set
9         ...0 .... .... = Accurate ECN: Not set
10        .... 0... .... = Congestion Window Reduced: Not set
11        .... .0.. .... = ECN-Echo: Not set
```

Out of Order 2/2

Confused-Paket

```
12          .... ..0. .... = Urgent: Not set
13          .... ...1 .... = Acknowledgment: Set
14          .... .... 1... = Push: Set
15          .... .... .0.. = Reset: Not set
16          .... .... ..0. = Syn: Not set
17          .... .... ...0 = Fin: Not set
18      Window: 281
19      Checksum: 0x97f3 [unverified]
20      Urgent Pointer: 0
21      Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
22            %(
23    [SEQ/ACK analysis]
24          [TCP Analysis Flags]
25              [Expert Info: This frame is a (suspected) out-of-order segment]
```

FIN 1/2

Happy-Paket

```
1 Transmission Control Protocol, Src Port: 35232, Dst Port: 22, Seq:
  ↪ 5170, Ack: 13125, Len: 0
2     Source Port: 35232
3     Destination Port: 22
4     Sequence Number (raw): 1628659363
5     Acknowledgment Number: 13125    (relative ack number)
6     Acknowledgment number (raw): 596950647
7     Flags: 0x011 (FIN, ACK)
8         000. .... .... = Reserved: Not set
9         ...0 .... .... = Accurate ECN: Not set
10        .... 0... .... = Congestion Window Reduced: Not set
11        .... .0.. .... = ECN-Echo: Not set
```

FIN 2/2

Happy-Paket

```
12          .... ..0. .... = Urgent: Not set
13          .... ...1 .... = Acknowledgment: Set
14          .... .... 0... = Push: Not set
15          .... .... .0.. = Reset: Not set
16          .... .... ..0. = Syn: Not set
17          .... .... ...1 = Fin: Set
18              [Expert Info (Chat/Sequence): Connection finish (FIN)]
19      Window: 662
20      Checksum: 0xa12d [unverified]
21      Urgent Pointer: 0
22      Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
23              :)
```

Das DeaTH-Flag in TCP 1/2

Wenn die Verbindung demnächst abgebrochen wird, kann mit dem DTH-Flag den folgenden Programmen die Möglichkeit gegeben werden, sich auf das abrupte Beenden der Sessions vorzubereiten.

- geht zurück auf einen Brauch in Anime, Manga und leichter Lektüre
- kann von Server und Client gesetzt werden
- keine Auswirkungen auf Sequence oder Acknowledgement Nummer, braucht nicht bestätigt werden
- Empfänger muss nicht speziell darauf reagieren, allerdings sollte die Information an die Anwendungsschicht und somit den User übergeben werden, kein Schließen der Session
- Sende-Zeitpunkt abhängig von Umsetzung
- nicht dazu designed, die TCP-Session zu charakterisieren (z.B. Evil-Bit aus RFC 3514 für IP-Header), beschreibt Schicksal der Session

Das DeaTH-Flag in TCP 2/2

- Anwendungsfälle:
  - jemand bereut seinen Anteil in einem DDoS-Angriff
  - aufhören kryptografischen Schutz zu verwenden (Redirect von HTTPs zu HTTP)
  - gefälschter User-Agent oder Server ändert HTTP-Header, um seine wahre Identität preiszugeben
  - wenn Speicherplatz oder Bandbreite sich verringert
  - AI-Bots, Cyborgs, Zauberanwendungen mit verbotenen Protokollen, die anfangen, stark Fehlermeldungen zu husten
- DTH-Fags sollten nicht verwendet werden:
  - zusammen mit FIN-Flags
  - zusammen mit URG-Flags
  - am Anfang der Session, weil dann davon ausgegangen werden kann, dass sie nicht essentiell ist

Vielen Dank für Ihre Aufmerksamkeit!

Bei weiteren Fragen wenden Sie sich bitte an info@b1-systems.de oder +49 (0)8457 - 931096