https://matse.paddel.xyz/spicker

Kommunikationssysteme

Patrick Gustav Blaneck, Tim Wende

Letzte Änderung: 1. Juli 2022

Inhaltsverzeichnis

1 Einführung

1.1 Grundbegriffe

Definition: Datenkommunikation

Die *Datenkommunikation* beschäftigt sich mit dem immateriellen Transport digitaler Daten zwischen Endsystemen.

Hierbei sind alle benötigten Verfahren und Regeln Bestandteil der Datenkommunikation.

Vorteile durch Datenkommunikation sind:

- zurückgreifen auf fremde bzw. entfernte Ressourcen und Daten
- Kostensenkung durch gemeinsame Nutzung von Betriebsmitteln
- Informationsgewinn durch entfernten Zugriff

Dazu benötigt es:

- Effiziente Methoden zum Datenaustausch zwischen Kommunikationspartnern
- Absprachen bzw. Regeln zur gemeinsamen Nutzung der Infrastruktur
- Kommunikationsdienste zur Übertragung von Informationen in verteilten Umgebungen

Definition: Kommunikationssystem

Im engeren Sinn ist ein *Kommunikationssystem* eine Einrichtung bzw. eine Infrastruktur für die Übermittlung von Informationen.

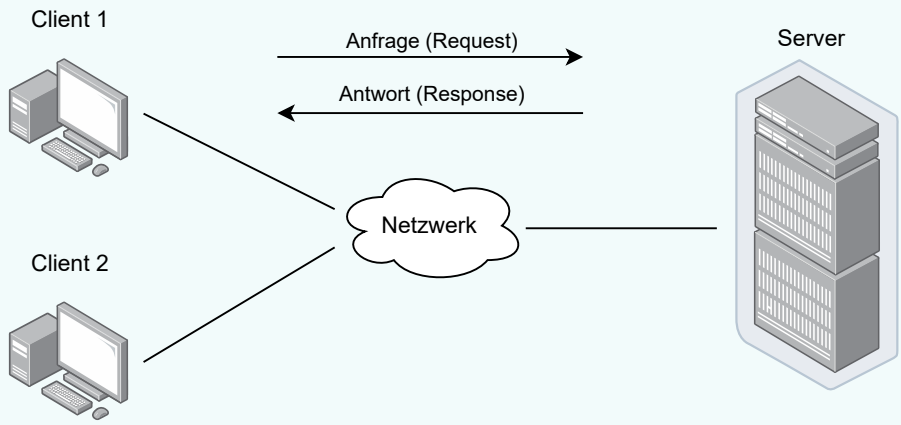
Kommunikationssysteme stellen dazu Nachrichtenverbindungen zwischen mehreren Endstellen her.

1.2 Kommunikationsmodelle

Definition: Klassische Client-Server-Architektur

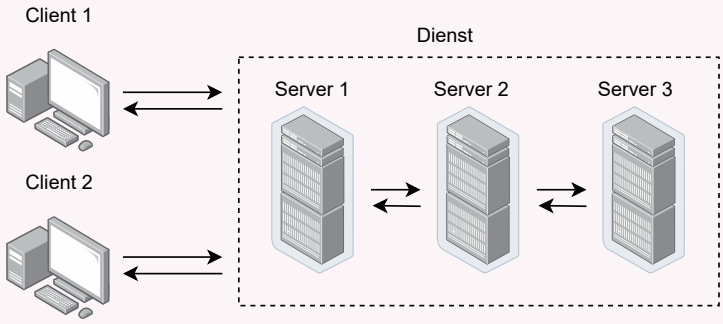
Server sind langlebige Prozesse, welche auf Anfragen von Clients warten und diese abarbeiten.

Clients stellen zu beliebigen Zeiten Anfragen und warten auf die Antwort. Die Rolle ist damit zumeist beendet.



Bonus: Dienst

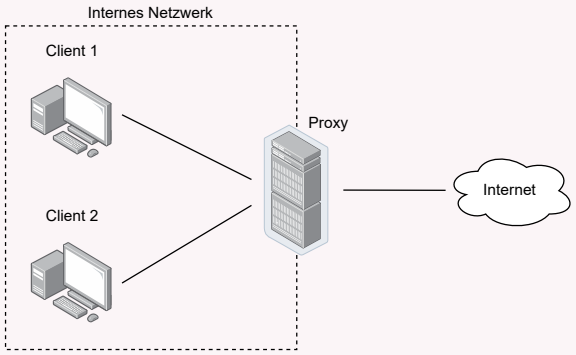
Ein *Dienst* wird von einem Verbund von Servern erbracht, durch den sich erst die Gesamtsicht ergibt. Ggf. merkt der Client nichts von dem Verbund.



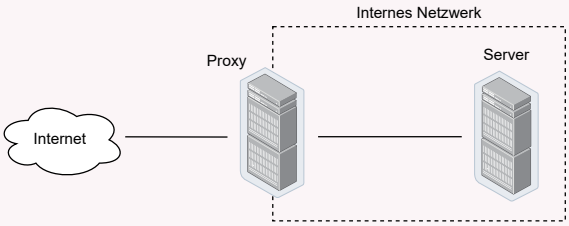
Bonus: Proxy

Ein *Proxy* dient zum Zwischenspeichern oder Anonymisieren von Anfragen auf der Clientseite, oder zum Lastbalanzieren auf der Serverseite.

Forward Proxy:



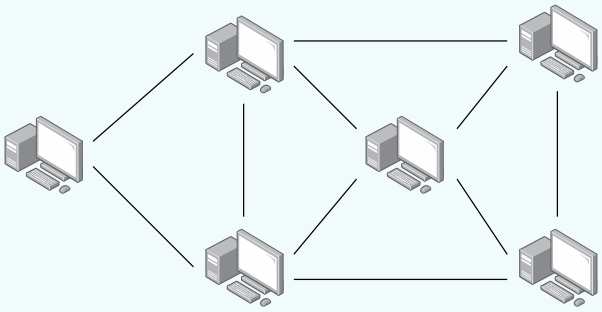
Reverse Proxy:



Definition: Peer-to-Peer

In einem reinen *Peer-to-Peer-Netz* sind alle Computer gleichberechtigt und können sowohl Dienste in Anspruch nehmen, als auch zur Verfügung stellen.

Sobald die Peers, die die gesuchten Objekte halten, in dem P2P-System identifiziert wurden, wird die Datei (in Dateitauschbörsen) direkt, d. h. von Peer zu Peer, übertragen. Es existieren unterschiedliche Verteilungsstrategien, welche Teile der Datei von welchem Peer heruntergeladen werden soll, z. B. BitTorrent.



1.2.1 Klassifikation nach Übertragungstechnik

Definition: Point-to-Point-Verbindung

Bei einer *Point-to-Point-Verbindung* ist ein Paar von Rechnern durch eine direkte Leitung verbunden, ohne dass ein anderer Rechner diese Leitung nutzen kann. Man unterscheidet:

Bezeichnung	Beschreibung	Beispiele
Simplex	Daten können in nur eine Richtung übertragen werden, diese Technik ermöglicht keine Antwort	Rundfunk, Pager
Halbduplex	Daten können abwechselnd, aber nicht gleichzeitig, in beide Richtungen fließen	Wechselsprechanlage, USB bis 2.0
Vollduplex	Daten können in beide Richtungen gleichzeitig übertragen werden.	Gegensprechanlage, USB ab 3.0
Dual-Simplex	ähnlich Vollduplex, aber getrennte Sende- und Empfangswege	PCI Express, Serial ATA

Definition: Multi-Access-Netz

Bei einem *Multi-Access-Netz* teilen sich mehrere Rechner einen Übertragungskanal.

Damit Daten trotzdem an den richtigen Empfänger gesendet werden, müssen sie mit einer Zieladresse versehen werden.

Daten werden in Übertragungseinheiten (Frames) eingeteilt und mit der Empfängeradresse ausgewiesen.[a]

[a] Sollen alle Stationen gleichzeitig eine Nachricht erhalten, so werden Broadcast-Adressen verwendet.

1.2.2 Klassifikation nach Verbindungsart

Definition: Statisches Netz

In einem *statischen Netz* befinden sich fest verdrahtete Point-to-Point-Verbindungen oder Multi-Access-Netze.

Dabei besitzt jeder Knoten eine feste Anzahl von Nachbarn oder einen Zugang zu einem Multi-Access-Netz.

Definition: Dynamisches Netz

In einem *dynamischen Netz* enthalten die Verbindungen konfigurierbare Schaltelemente.

Diese können dynamisch vermitteln durch Leitungsvermittlung[a].

Dabei ist ein ein- oder mehrstufiger Aufbau möglich.

[a] Man sagt „ein Weg wird geschaltet“.

1.2.3 Klassifikation nach Topologieeigenschaften

Definition: Durchmesser

Der *Durchmesser* eines Netzes gibt an, wie groß der maximale Abstand zweier Knoten, d. h. die Anzahl von Kanten (Links), die auf einem kürzesten Pfad (Route) zwischen Knoten genutzt werden muss.

Das Ziel ist ein möglichst kleiner Durchmesser und damit ein möglichst kleiner Zeitbedarf für Übertragungen.

Definition: Bisektionsweite

Die *Bisektionsweite* eines Netzes gibt die minimale Anzahl von Kanten an, die man entfernen muss, um ein Netz mit N Knoten in zwei isolierte Teile mit jeweils $\frac{N}{2}$ Knoten zu teilen.

Das Ziel ist eine möglichst große Bisektionsweite zur Verbesserung der Fehlertoleranz.

Definition: Grad

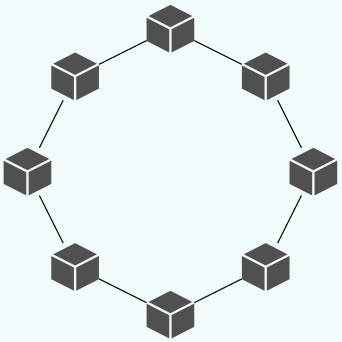
Der *Grad* eines Netzes entspricht dem größten Grad eines Knotens.

Der Grad eines Knotens wiederum ist die Anzahl ausgehender Kanten, d. h. die Anzahl der Nachbarn des Knotens.[a]

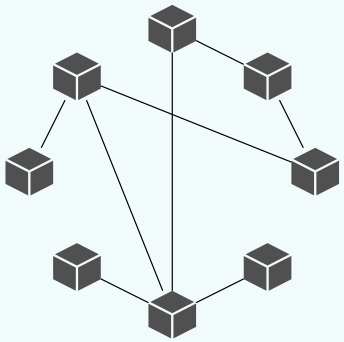
Das Ziel ist ein möglichst kleiner Grad des Netzes, da die Kosten mit dem Grad steigen.

[a] Haben alle Knoten den gleichen Knotengrad, so spricht man von einem *regulären Netz*.

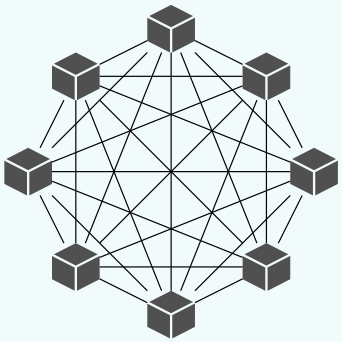
Definition: Netzwerktopologie



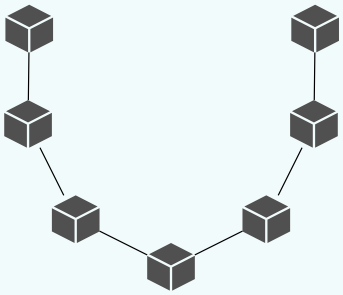
Ring



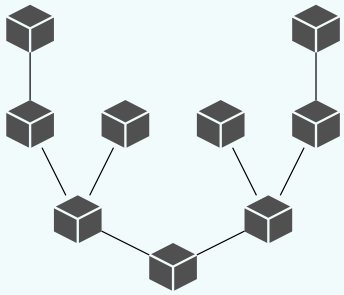
Vermascht



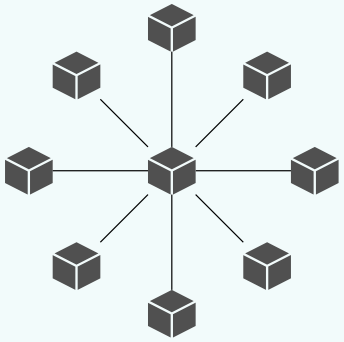
Vollvermascht



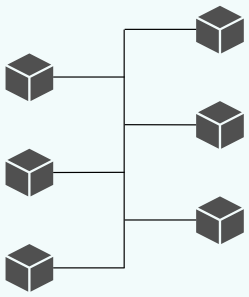
Linie



Baum



Stern



Bus

Topologie	Durchmesser	Bisektionsweite	Grad	Regulär
Ring	$N/2$	2	2	Ja
Vollvermascht[a]	1	$N^2/4$	$N-1$	Ja
Linie	$N-1$	1	1 bzw. 2	Nein
Baum (binär)	$2\log_2 N$	1	1 oder 2	Nein
Stern	2	(1)	1 bzw. $N-1$	Nein
Bus	1	(1)	(1)	(Ja)

Für Details, siehe 16.2.

1.2.4 Klassifikation nach Ausdehnung

Definition: Local Area Network (LAN)

Ein *Local Area Network (LAN)* ist eine Kommunikationsinfrastruktur für einen begrenzten geographischen Bereich (10 Meter bis wenige Kilometer).

Typischerweise erfolgt die Verkabelung eines LANs ab einer gewissen Größe als strukturierte Verkabelung. Ethernet ist der am weitesten verbreitete Standard.

Definition: Metropolitan Area Network (MAN)

Ein *Metropolitan Area Network (MAN)* überbrückt größere Distanzen als ein LAN, z. B. den Bereich einer Stadt.

Üblicherweise verbindet ein MAN zahlreiche LANs und verwendet dazu eine Backbone-Technologie, die meist in Glasfasertechnik realisiert ist. Ein MAN kann eine Ausdehnung von bis zu 100 km haben.

MANs werden oft von international tätigen Telekommunikationsfirmen aufgebaut, die per MAN verkabelte Metropolen wiederum in einem Wide Area Network (WAN) national oder in einem Global Area Network (GAN) international vernetzen.

Definition: Wide Area Network (WAN)

Ein *Wide Area Network (WAN)* ist ein Rechnernetz, das sich im Unterschied zu einem LAN oder MAN über einen sehr großen geografischen Bereich erstreckt.

Die Anzahl der angeschlossenen Rechner entsprechen dem Maximum von IPv4 oder IPv6. WANs erstrecken sich über Länder oder sogar Kontinente.

WANs werden benutzt, um verschiedene LANs, aber auch einzelne Rechner miteinander zu vernetzen.

Einige WANs gehören bestimmten Organisationen und werden ausschließlich von diesen genutzt. Andere WANs werden durch Internetdienstanbieter errichtet oder erweitert, um einen Zugang zum Internet anbieten zu können.

Definition: Global Area Network (GAN)

Unter einem *Global Area Network (GAN)* versteht man ein Netz, das über unbegrenzte geographische Entfernungen mehrere Wide Area Networks verbinden kann. Dies kann zum Beispiel die Vernetzung weltweiter Standorte eines internationalen Unternehmens sein.

Oft wird bei einem GAN Satelliten- oder Glasfaserübertragung eingesetzt.

GAN ist nicht die direkte Bezeichnung für das Internet, da es theoretisch mehrere GANs abgeschottet und unabhängig geben kann, das Internet jedoch eine globale Vernetzung ohne (maßgebliche) Unterteilungen ist. So ist das Internet ein GAN, aber nicht jedes GAN wird Internet genannt.

1.2.5 Klassifikation nach Netzkomponenten

Definition: Switch

Ein *Switch* hat mehrere Anschlüsse, über die Rechner miteinander verbunden werden können.

Er merkt sich, welcher Rechner an welchem Anschluss angeschlossen ist (Adresse der Netzwerkkarte) und kann Daten gezielt an einen Anschluss weiterleiten.

Der Switch kennt nur Rechner, die direkt an ihn angeschlossen sind.

Definition: Router

Will man Daten an weit entfernte Kommunikationspartner schicken, gibt es meist mehrere mögliche Wege, die man nehmen kann.

Router verwalten globale Adressinformationen, kennen kürzeste Wege zu allen Rechnern und können Daten gezielt in andere Netze weiterleiten.

Definition: Backbone

Als *Backbone* bezeichnet man eine Menge von Rechnern, die miteinander verbunden sind, um kleinere Netze miteinander zu koppeln und so den Datenaustausch zu ermöglichen.

2 Schichtenmodelle

Definition: Netzwerkprotokoll

Ein *Netzwerkprotokoll* ist ein Kommunikationsprotokoll für den Austausch von Daten zwischen Computern bzw. Prozessen, die in einem Rechnernetz miteinander verbunden sind (verteiltes System).

Der Austausch von Nachrichten erfordert häufig ein Zusammenspiel verschiedener Protokolle, die unterschiedliche Aufgaben übernehmen.

Um die damit verbundene Komplexität beherrschen zu können, werden die einzelnen Protokolle in Schichten organisiert. Im Rahmen einer solchen Architektur gehört jedes Protokoll einer bestimmten Schicht an und ist für die Erledigung der speziellen Aufgaben zuständig[a].

Protokolle höherer Schichten verwenden Dienste von Protokollen tieferer Schichten.[b] Zusammen bilden die so strukturierten Protokolle einen Protokollstapel.

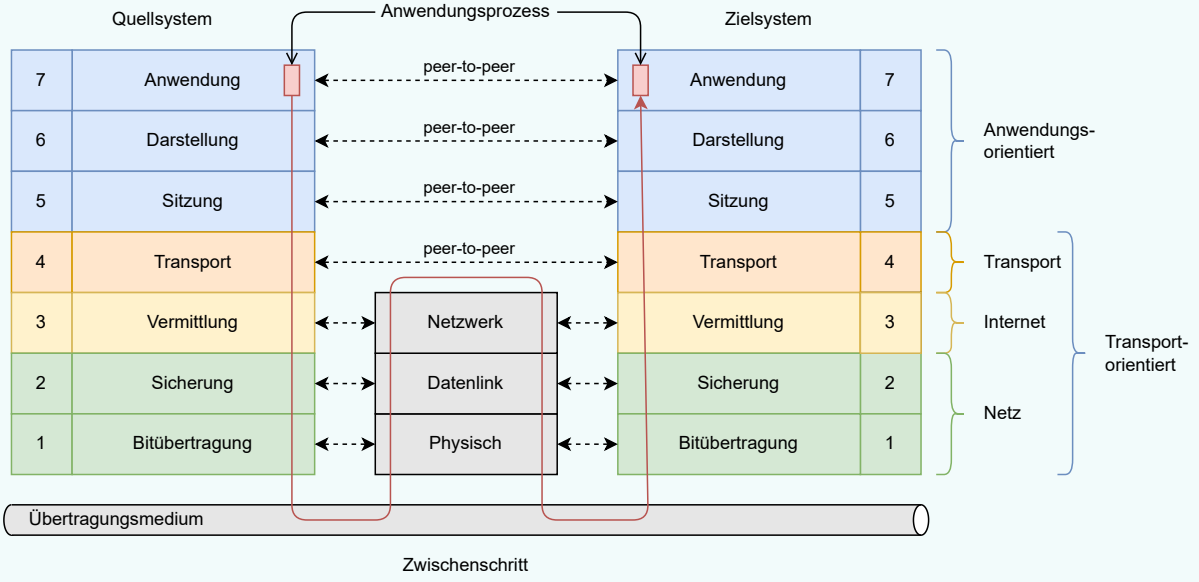
[a] beispielsweise Übermitteln an einen bestimmten Knoten – Schicht 2
[b] Schicht 3 bildet ein logisches Netzwerk und verwendet Schicht 2 für die physische Zustellung.

2.1 ISO/OSI-Referenzmodell

Definition: ISO/OSI-Referenzmodell

Das *ISO/OSI-Referenzmodell* ist ein Referenzmodell für Netzwerkprotokolle als Schichtenarchitektur.

Zweck des OSI-Modells ist es, Kommunikation über unterschiedlichste technische Systeme hinweg zu beschreiben und die Weiterentwicklung zu begünstigen. Dazu definiert dieses Modell sieben aufeinanderfolgende Schichten (layers) mit jeweils eng begrenzten Aufgaben. In der gleichen Schicht mit klaren Schnittstellen definierte Netzwerkprotokolle sind einfach untereinander austauschbar, selbst wenn sie wie das Internet Protocol eine zentrale Funktion haben.



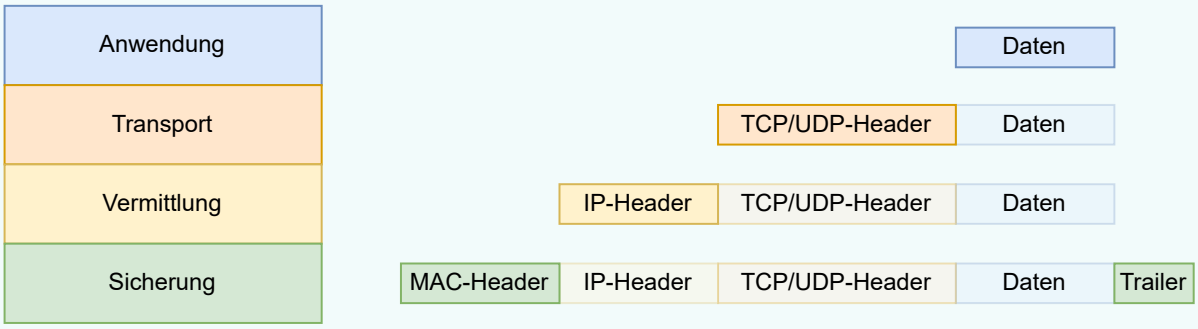
Definition: Interaktion zwischen Schichten

Eine Schicht $(n-1)$ bietet der über ihr liegenden Schicht n Dienste an.

Dabei versieht Schicht n ihre Nachricht mit Kontrollinformationen (Header) und versendet alles zusammen als *Protocal Data Unit (PDU)*.

Zwei Kommunikationspartner auf Schicht n tauschen PDUs aus und nutzen dazu die Dienste der nächsttieferen Schicht $(n-1)$.

Für Schicht $(n-1)$ sind diese PDUs die zu übertragenen Daten.



Definition: Anwendungsschicht (Application Layer)

Dienste, Anwendungen und Netzmanagement.

Die *Anwendungsschicht* (Schicht 7) stellt Funktionen für die Anwendungen zur Verfügung.

Diese Schicht stellt die Verbindung zu den unteren Schichten her.

Auf dieser Ebene findet auch die Datenein- und ausgabe statt. Die Anwendungen selbst gehören nicht zur Schicht.

Definition: Darstellungsschicht (Presentation Layer)

Die *Darstellungsschicht* (Schicht 6) setzt die systemabhängige Darstellung der Daten (z. B. ASCII, Unicode) in eine unabhängige Form um und ermöglicht somit den syntaktisch korrekten Datenaustausch zwischen unterschiedlichen Systemen.

Auch Aufgaben wie die Datenkompression und die Verschlüsselung gehören zur Schicht 6.

Die Darstellungsschicht gewährleistet, dass Daten, die von der Anwendungsschicht eines Systems gesendet werden, von der Anwendungsschicht eines anderen Systems gelesen werden können.

Falls erforderlich, agiert die Darstellungsschicht als Übersetzer zwischen verschiedenen Datenformaten, indem sie ein für beide Systeme verständliches Datenformat, die ASN.1 (Abstract Syntax Notation One), verwendet.

Definition: Sitzungsschicht (Session Layer)

Die *Sitzungsschicht* (Schicht 5) sorgt für die Prozesskommunikation zwischen zwei Systemen.

Um Zusammenbrüche der Sitzung und ähnliche Probleme zu beheben, stellt die Sitzungsschicht Dienste für einen organisierten und synchronisierten Datenaustausch zur Verfügung. Zu diesem Zweck werden Wiederaufsetzpunkte, so genannte Fixpunkte (Check Points) eingeführt, an denen die Sitzung nach einem Ausfall einer Transportverbindung wieder synchronisiert werden kann, ohne dass die Übertragung wieder von vorne beginnen muss.

Definition: Transportschicht (Transport Layer)

Zu den Aufgaben der *Transportschicht* (Schicht 4) zählt die Segmentierung des Datenstroms, die Stauvermeidung (congestion avoidance) und die Sicherstellung einer fehlerfreien Übertragung.

Die Transportschicht bietet den anwendungsorientierten Schichten 5 bis 7 einen einheitlichen Zugriff, so dass diese die Eigenschaften des Kommunikationsnetzes nicht zu berücksichtigen brauchen.

Definition: Vermittlungsschicht (Network Layer)

Die *Vermittlungsschicht* (Schicht 3) sorgt bei leitungsorientierten Diensten für das Schalten von Verbindungen und bei paketorientierten Diensten für die Weitervermittlung von Datenpaketen sowie die Stauvermeidung (congestion avoidance).

Die Datenübertragung geht in beiden Fällen jeweils über das gesamte Kommunikationsnetz hinweg und schließt die Wegsuche (Routing) zwischen den Netzwerkknoten ein.

Da nicht immer eine direkte Kommunikation zwischen Absender und Ziel möglich ist, müssen Pakete von Knoten, die auf dem Weg liegen, weitergeleitet werden. Weitervermittelte Pakete gelangen nicht in die höheren Schichten, sondern werden mit einem neuen Zwischenziel versehen und an den nächsten Knoten gesendet.

Zu den wichtigsten Aufgaben der Vermittlungsschicht zählt das Bereitstellen netzwerkübergreifender Adressen, das Routing bzw. der Aufbau und die Aktualisierung von Routingtabellen und die Fragmentierung von Datenpaketen.

Das Internet Protocol gehört zu dieser Schicht.

Definition: Sicherungsschicht (Data Link Layer)

Aufgabe der *Sicherungsschicht* (Schicht 2) ist es, eine zuverlässige, das heißt weitgehend fehlerfreie Übertragung zu gewährleisten und den Zugriff auf das Übertragungsmedium zu regeln. Dazu dient das Aufteilen des Bitdatenstromes in Blöcke (Frames) und das Hinzufügen von Prüfsummen im Rahmen der Kanalkodierung. So können fehlerhafte Blöcke vom Empfänger erkannt und entweder verworfen oder sogar korrigiert werden.

Eine Datenflusskontrolle ermöglicht es, dass ein Empfänger dynamisch steuert, mit welcher Geschwindigkeit die Gegenseite Blöcke senden darf.

Das Ethernet-Protokoll beschreibt sowohl Schicht 1 als auch Schicht 2, wobei auf dieser als Zugriffskontrolle CSMA/CD zum Einsatz kommt.

Definition: Bitübertragungsschicht (Physical Layer)

Die *Bitübertragungsschicht* (Schicht 1) ist die unterste Schicht.

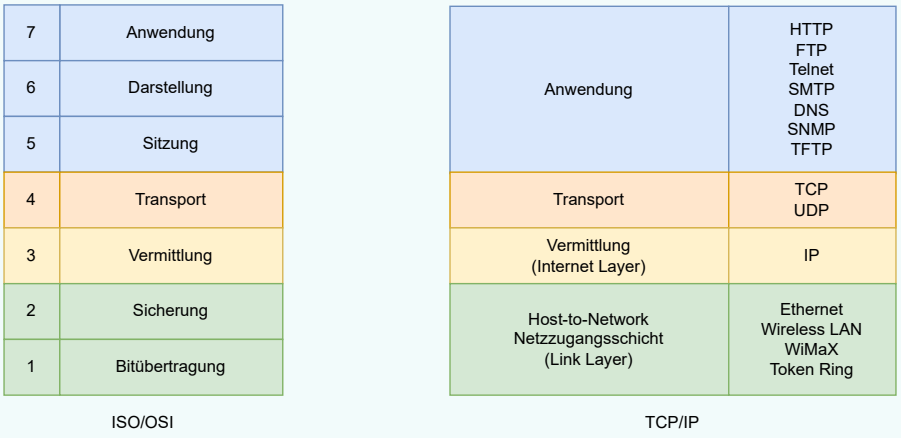
Diese Schicht stellt mechanische, elektrische, physikalische und weitere funktionale Hilfsmittel zur Verfügung, um physische Verbindungen zu aktivieren bzw. zu deaktivieren, sie aufrechtzuerhalten und Bits darüber zu übertragen. Das können zum Beispiel elektrische Signale, optische Signale (Lichtleiter, Laser), elektromagnetische Wellen (drahtlose Netze) oder Schall sein.

Die gemeinsame Nutzung eines Übertragungsmediums kann auf dieser Schicht durch statisches Multiplexen oder dynamisches Multiplexen erfolgen. Dies erfordert neben den Spezifikationen bestimmter Übertragungsmedien (zum Beispiel Kupferkabel, Lichtwellenleiter, Stromnetz) und der Definition von Steckverbindungen noch weitere Elemente.

2.2 TCP/IP-Referenzmodell

Definition: TCP/IP-Referenzmodell

Das *TCP/IP-Referenzmodell* ist eine protokollunabhängige Ausmodellierung des konzeptionellen ISO/OSI-Modells.



Definition: Anwendungsschicht (Application Layer)

Die *Anwendungsschicht* umfasst alle Protokolle, die mit Anwendungsprogrammen zusammenarbeiten und die Netzwerkinfrastruktur für den Austausch anwendungsspezifischer Daten nutzen.

Definition: Transportschicht (Transport Layer)

Die *Transportschicht* ermöglicht eine Ende-zu-Ende-Kommunikation. Das wichtigste Protokoll dieser Schicht ist das Transmission Control Protocol (TCP), das Verbindungen zwischen jeweils zwei Netzwerkteilnehmern zum zuverlässigen Versenden von Datenströmen herstellt.

Definition: Internetschicht (Internet Layer)

Die *Internetschicht* ist für die Weitervermittlung von Paketen und die Wegwahl (Routing) zuständig. Auf dieser Schicht und der darunterliegenden Schicht werden Direktverbindungen betrachtet. Die Aufgabe dieser Schicht ist es, zu einem empfangenen Paket das nächste Zwischenziel zu ermitteln und das Paket dorthin weiterzuleiten. Kern dieser Schicht ist das Internet Protocol (IP) in der Version 4 oder 6.

Definition: Netzzugangsschicht (Network Access Layer)

Die *Netzzugangsschicht* ist im TCP/IP-Referenzmodell spezifiziert, enthält jedoch keine Protokolle der TCP/IP-Familie. Sie ist vielmehr als Platzhalter für verschiedene Techniken zur Datenübertragung von Punkt zu Punkt zu verstehen.

3 Sockets

Definition: Socket

Sockets bilden eine plattformunabhängige standardisierte Schnittstelle (API) zwischen der Netzwerkprotokoll-Implementierung des Betriebssystems und der eigentlichen Anwendungssoftware.

Ein Computerprogramm fordert einen Socket vom Betriebssystem an. Das Betriebssystem hat die Aufgabe, alle benutzten Sockets sowie die zugehörigen Verbindungsinformationen zu verwalten.

Definition: Internet-Socket

Internet-Sockets ermöglichen Kommunikation mittels bestimmter Kommunikationsprotokolle.

Generell kann man unterscheiden zwischen:

- *Stream-Sockets*:
  - kommunizieren über einen Zeichen-Datenstrom
  - verwenden meist TCP
- *Datagramm-Sockets*:
  - kommunizieren über einzelne Nachrichten
  - verwenden meist UDP

Definition: Socket-Kommunikation (Stream-Socket)

Client-seitig:

1. Socket erstellen
2. erstellten Socket mit einer Server-Adresse verbinden, von welcher Daten angefordert werden sollen
3. Senden und Empfangen von Daten
4. evtl. Socket herunterfahren
5. Verbindung trennen, Socket schließen

Server-seitig:

1. Server-Socket erstellen
2. Binden des Sockets an eine Adresse (Port), über welche Anfragen akzeptiert werden
3. auf Anfragen warten
4. Anfrage akzeptieren und damit ein neues Socket-Paar für diesen Client erstellen
5. Bearbeiten der Client-Anfrage auf dem neuen Client-Socket
6. Client-Socket wieder schließen

Definition: Socket-Kommunikation (Datagramm-Socket)

Client-seitig:

1. Socket erstellen
2. An Adresse senden

Server-seitig:

1. Socket erstellen
2. Socket binden
3. warten auf Pakete

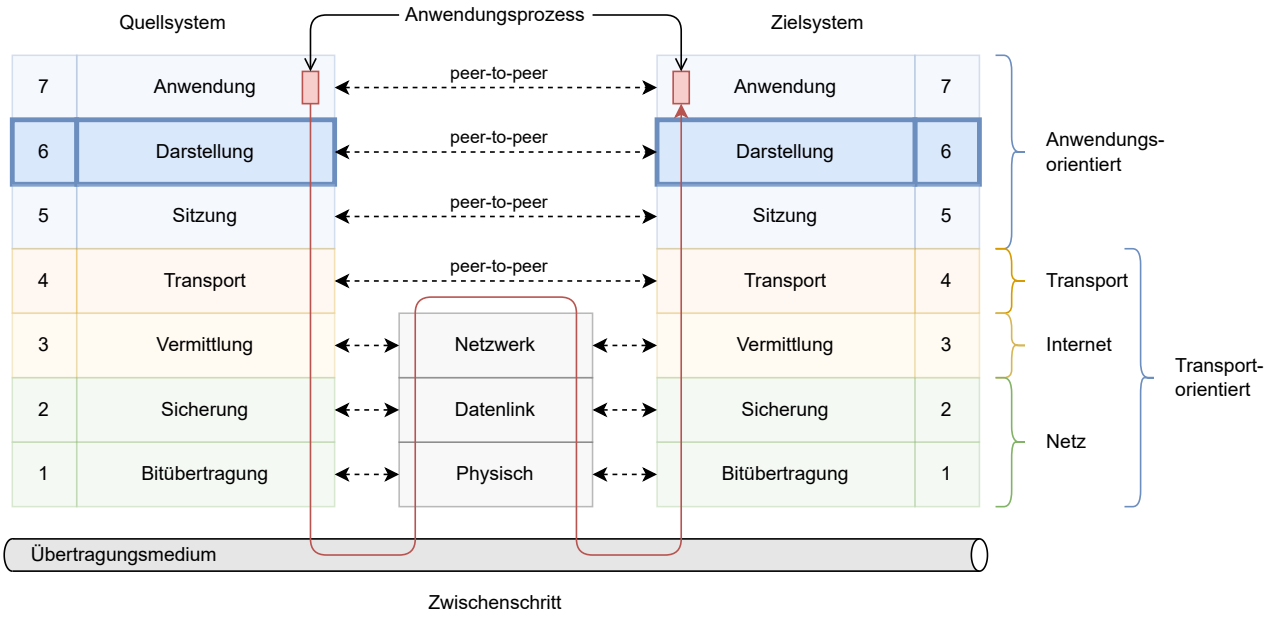
Beispiel: Socket (Client)

```java
import java.io.*;
import java.net.*;

class TCPClient {
    public static void main(String args[]) throws Exception {
        BufferedReader input = new BufferedReader(new InputStreamReader(System.in));

        // Erstelle Client-Socket, baue Verbindung auf
        Socket socket = new Socket("192.168.0.1", 591);

        // Erstelle Datenstrom fuer den Socket
        DataOutputStream sender = new DataOutputStream(
            socket.getOutputStream()
        );

        // Erstelle Datenstrom aus dem Socket
        BufferedReader receiver = new BufferedReader(
            new InputStreamReader(socket.getInputStream())
        );

        String msg = input.readLine();

        // Sende an den Server
        sender.writeBytes(msg + '\n');

        // Empfange vom Server
        String modified_msg = receiver.readLine();

        System.out.println("FROM SERVER: " + modified_msg);

        socket.close();
    }
}
```

Beispiel: Socket (Server)

```java
import java.io.*;
import java.net.*;

class TCPServer {
    public static void main(String args[]) throws Exception {
        // Erstelle Socket
        ServerSocket socket = new ServerSocket(591);

        while (true){
            // Warte auf eingehende Verbindungswuensche
            Socket connection = socket.accept();

            // Starte Thread um auf weitere Anfragen reagieren zu koennen.
            new Thread() {
                public void run() {
                    // Verknuepfe Buffer mit dem Socket
                    BufferedReader receiver = new BufferedReader(
                        new InputStreamReader(connection.getInputStream())
                    );

                    // Verknuepfe Datenstrom mit dem Socket
                    DataOutputStream sender = new DataOutputStream(
                        connection.getOutputStream()
                    );

                    // Lesen vom Socket
                    String msg = receiver.readLine();

                    // Schreiben auf den Socket
                    sender.writeBytes(msg.toUpperCase() + '\n');
                }
            }.start();
        }
    }
}
```

4 Darstellung



Bonus: Probleme bei der Datenübertragung

- Heterogenität in verteilten System:
  - Plattformen (Betriebssystem, Hardware)
  - Programmiersprachen, APIs
- Heterogene Hardware-Architekturen:
  - Zeichencodierung (ASCII, UTF8)
  - Reihenfolge der internen Speicherung von Bytes

Beispiel: Little-Endian und Big-Endian

Dieses Problem betrifft Datentypen, die aus mehreren Byte zusammengesetzt sind.

32 Bit-Ganzzahlen benötigen 4 Byte Speicherplatz[a]:

MSB						**LSB**	
`0xA0`		`0xB0`		`0xC0`		`0xD0`	
16^7	16^6	16^5	16^4	16^3	16^2	16^1	16^0

Als Ausgangswert steht das *höchswertige Byte* („most significant byte“ bzw. MSB) $(16^7 + 16^6)$ am Anfang und das *kleinstwertige Byte* („least significant byte“ bzw. LSB) $(16^1 + 16^0)$ am Ende.

Big-Endian speichert die Bytes in der Reihenfolge, in der sie im Ausgangswert vorliegen:
`0xA0`, `0xB0`, `0xC0`, `0xD0`

Little-Endian speichert die Bytes in der umgekehrten Reihenfolge:
`0xD0`, `0xC0`, `0xB0`, `0xA0`

[a] Hier betrachten wir beispielsweise den Typ *uint*

Beispiel: Übertragung von Datenstrukturen

Häufig müssen nicht nur Werte, sondern ganze Datenstrukturen übertragen werden.

```
class Pokemon {
    uint nummer;
    String name;
}

Pokemon p = new Pokemon(4, "Glumanda");
send(p);
```

Um dieses Objekt zu übertragen muss man die Werte in Bytes umwandeln.

Nummer				Name															
4				Glumanda															
0	0	0	4	4	7	6	C	7	5	6	D	6	1	6	E	6	4	6	1

Der übertragene Bitstream sieht demnach wie folgt aus:

```
0000 0000 [...] 0000 0100 | 0100 0111 [...] 0110 0001
```

Unser Zielsystem weiß, dass ein Pokemon empfangen werden soll. Jedoch sieht die Struktur eventuell auf diesem System wie folgt aus:

```
class Pokemon {
    String name;
    uint nummer;
}

Pokemon p = receive();
```

Demnach teilt dieses System die empfangenen Bytes wie folgt auf:

0	0	0	4	4	7	6	C	7	5	6	D	6	1	6	E	6	4	6	1
Name																Nummer			

Das Pokemon `NULEOTGluman` mit der Nummer `25697` wurde somit empfangen.

Die potentiell entstehenden Probleme lassen sich in zwei Kategorien einteilen:

- Übertragung der grundsätzlichen Struktur der Daten (Serialisierung, Deserialisierung)
- Übertragung des konkreten Datenstroms mit einer vereinbarten Kodierung (Marshalling)

Bonus: Eigenständige Darstellung

Wir gehen mit folgender Idee an unser Problem:

- Definiere eine Menge abstrakter Datentypen und eine Kodierung für jeden dieser Typen.
- Stelle Werkzeuge zu Verfügung, welche die Datentypen der verwendeten Programmiersprache in die abstrakten Typen übersetzt.
- Wenn ein bestimmter Datentyp übertragen werden soll, rufe die Kodierfunktion auf, und übertrage das Ergebnis (*Marshalling*).
- Dekodiere den Bit-String und erzeuge eine neue lokale Repräsentation des empfangenen Typs (*Unmarshalling*).

Beispiel: Datendarstellungsstandards

- ISO:
  - `ASN.1` (Abstract Syntax Notation)
- Sun ONC-RPC (Open Network Computing):
  - `XDR` (eXternal Data Representation)
- OSF-RPC (Open System Foundation):
  - `IDL` (Interface Definition Language)
- Corba:
  - IDL und CDR (Common Data Representation)
  - CDR bildet IDL-Datentypen in Bytefolge ab
- W3C:
  - XML/SOAP
  - Darstellung aller Datentypen als (maschinen-)lesbarer Text.
  - Zu klären: Zeichencodierung
- Java:
  - Objektserialisierung, d.h. Abflachung von Objekten zu einem seriellen Format inkl. Informationen über die Klassen. Deserialisierung ist die Wiederherstellung eines Objektes ohne Vorwissen über die Typen der Objekte.
- JavaScript:
  - JSON (JavaScript Object Notation)

4.1 ASN.1

Definition: ASN.1

Abstract Syntax Notation (ASN.1) ist eine von der ISO genormte Beschreibungssprache zu darstellungsunabhängige Spezifikationen von Datentypen und Werten. Diese Notation findet z.B. zur Definition von Managementobjekten bei SNMP Verwendung.

Elementare Datentypen:
`Boolean`, `Integer`, `Bitstring`, `Octetstring`, `IA5String`, `...`

Strukturelle Datentypen:

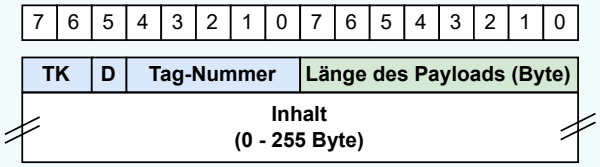
- `Sequence`: Geordnete Liste von Datentypen (Struct in C)
- `Set`: Ungeordnete Menge an Datentypen
- `Sequence OF ...` : Geordnete Liste von Datentypen des gleichen Datentyps (Array in C)
- `Set OF ...` : Ungeordnete Menge des gleichen Datentyps
- `Choice`: Ungeordnete Menge von Datentypen, aus der eine Datentypen ausgewählt werden können (Union in C)

Beispiel: ASN.1

```
Pokemon ::= Sequence {
    Nummer Integer,
    Name IA5String
}
```

Definition: ASN.1 Übertragungssyntax

Zusätzlich zur Bereitstellung einer Datenbeschreibungssprache bietet ASN.1 auch sogenannte *Basic Encoding Rules*, die spezifizieren, wie ASN.1-Objekte über das Netzwerk versendet werden sollten an.



- Bezeichner: 8 bit
  - Typklasse (TK): 2 bit

`00`	Universal
`01`	Application
`10`	Context Specific
`11`	Private

  - Datentyp (D): 1 bit

`0`	Einfach
`1`	Strukturiert

  - Tag-Nummer: 8 bit

`0000 0000`	End-of-Content (EOC)
`0000 0001`	Boolean
`0000 0010`	Integer
	...
`0001 0000`	Sequence, Sequence OF
`0001 0001`	Set, Set OF
	...
`0001 0110`	IA5String
	...

- Länge: 8 bit
- Wert: 0 - 255 Byte

Beispiel: ASN.1

Fassen wir nun einmal unsere Beispiele zusammen erhalten wir:

```
class Pokemon {
    uint nummer;
    String name;
}

Pokemon p = new Pokemon(4, "Glumanda");
```

Bzw. in ASN.1:

```
Pokemon ::= Sequence {
    Nummer Integer,
    Name IA5String
}
```

Nun bauen wir den zu übertragenden Bitstream von innen nach außen auf:

1. `Nummer` ist ein *universaler, einfacher Integer*.
   Demnach lautet der Header für `Nummer`: `00 0 00010`
2. `Name` ist ein *universaler, einfacher IA5String*.
   Demnach lautet der Header für `Name`: `00 0 10110`
3. Nun füllen wir den Payload mit den Werten `4` bzw. `Glumanda`.
4. Daraus erhalten wir die benötigte Länge von 4 Byte für `Nummer` bzw. 16 Byte für `Name`.
5. Unser gesamtes Objekt ist eine *universale, strukturierte Sequence*.
   Demnach lautet der Header für das `Pokemon`: `00 1 10000`
6. Die Gesamtlänge erhalten wir aus der Summe aller Teilobjekte inkl. Header (in Byte).
   4 (Nummer) $+ 16$ (Name) $+ 2 \cdot 2$ (Header) $= 14_{10} \rightarrow 10110_2$

0	0	1	1	0	0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	1	0	1	1	0	0	0	0	1	0	0	0	0
			4								7				
			6								C				
			7								5				
			6								D				
			6								1				
			6								E				
			6								4				
			6								1				

4.2 XML

Bonus: Probleme von HTML

HTML wurde zur Darstellung von Informationen für Menschen und nicht für die Informationsverarbeitung in Programmen entwickelt.

Die Auszeichnungselemente (Markup Elements) sind fest vorgegeben und sind daher nicht zur Beschreibung beliebiger Daten geeignet. Zusätzlich enthält HTML eine problematische Vermengung von Auszeichnungen und Präsentationssemantik.

Aus diesen Gründen sollte eine offene, textbasierte Auszeichnungsspache (Markup Language) entwickelt werden.

Definition: XML

XML (eXtensible Markup Language) ist eine Metasprache zur Strukturierung von Dokumenten und Daten.

Basis von XML sind beliebige Auszeichnungselemente, die geringen syntaktischen Regeln zu folgen haben.

XML ist erst wirklich sinnvoll, wenn es in einer Anwendung konkretisiert wird.

Anwendungen ergeben sich aus:

- *Namensräumen*: Festlegen der semantischen logischen Bedeutung bei Homonymen in den Auszeichnungselementen.
- *Stylesheets*: Definition des Erscheinungsbildes von Auszeichnungselementen.
- *Schemata*: Beschreiben der Dokumentenstruktur eines bestimmten Typs

XML-Dokumente enthalten Daten und Sturkturinformationen über die Daten in einem Dokument (selbstbeschreibend) und haben feste Strukturvorgaben (wohlgeformt). Informationen in XML-Dokumenten haben einen Datentyp (typisiert)

Jedes Dokument entspricht einer Baumstruktur und haben immer ausschließlich eine Wurzel.

Für XML-Dokumente gelten folgende Syntaxregeln:

- Jeder Starttag muss einen Endtag haben:

```
<pokemon> ... </pokemon>
<pokemon />
```

- Verschachtelung nur mit vollständigen Tags möglich.

```
<pokemon> <pokemonname> Glumanda </pokemonname> </pokemon> <!-- OK -->
<pokemon> <pokemonname> Glumanda </pokemon> </pokemonname>  <!-- NOT OK -->
```

Beispiel: XML

```
<?xml version="1.0" encoding="UTF-8"?>
<pokemon nummer="4">
    <name>Glumanda</name>
</pokemon>
```

Definition: XML-Namespace

Elementarbezeichner haben ähnlich wie Variablen in Programmiersprachen einen *Namensraum*. Dieser dient der Trennung der Bedeutung der Bezeichner und ermöglicht Inhalte eindeutig zu referenzieren.

Ein Namespace ist weltweit eindeutig durch eine URI (Uniform Resource Identifier) identifiziert und wird über einen Präfix zugänglich gemacht.

Beispiel: XML-Namespace

```xml
<pokemon nummer="4" xmlns:typ="https://kosy.paddel.xyz/pokemontyp">
    <pokemonname>Glumanda</pokemonname>
    <typ:primaertyp>Feuer</typ:primaertyp>
    <typ:sekundaertyp />
</pokemon>
```

Definition: XML-Schema

Ein *XML-Schema* ist eine XML-Anwendung, mit der die Struktur einer Klasse von XML-Dokumenten beschrieben werden kann.

Dieses Schema enthält in XML notierte Regeln, die erlaubte Elementarbezeichner, Reihenfolgen, Inhalte und Attribute mit Wertebereichen aufzählen.

Hierzu existieren vielfältige vordefinierte Datentypen, die Möglichkeiten zur Definition eigener Datentypen und zur Darstellung komplexer Integritätsbedingungen schaffen.

Beispiel: XML-Schema (primitive Datentypen)

Um primitive Datentypen zu nutzen, kann man sie wie folgt referenzieren:

Schema:

```xml
<xs:element name="pokemonNummer" type="xs:integer" />
```

Alternativ kann man von primitiven Datentypen erben, um diese um weitere Eigenschaften zu erweitern:

Schema:

```xml
<xs:simpleType name="pokemonNummer" base="xs:integer">
    <xs:minInclusive value="1" />
    <xs:maxInclusive value="905" />
</xs:simpleType>
```

Diesen Typ können wir nun nutzen, jedoch zusätzlich den eigentlichen primitiven Datentypen dahinter jederzeit austauschen:

Dokument:

```xml
<pokemonNummer>4</pokemonNummer>
```

Beispiel: XML-Schema (Elementdeklaration)

Um einfache Typen mit Attributen vorzugeben, deklariert man ein Element wie folgt:

Schema:

```
<xs:element name="pokemon" type="pokemonTyp" />

<xs:complexType name="pokemonTyp">
    <xs:attribute name="nummer" type="pokemonNummer" />
</xs:complexType>
```

Nun können wir bereits erste Pokemon anlegen:

Dokument:

```
<pokemon nummer="4" />
```

Beispiel: XML Schemata (komplexe Typen)

Da wir für unser vollwertiges Pokemon noch einen Namen brauchen, müssen wir unserem erstellten Pokemon weiter Kinderobjekte hinzufügen.

Schema:

```
<xs:element name="pokemon" type="pokemonTyp" />

<xs:complexType name="pokemonTyp">
    <xs:attribute name="nummer" type="pokemonNummer" />
    <xs:sequence>
        <xs:element name="name" type="xs:string" />
    </xs:sequence>
</xs:complexType>
```

Jetzt ist unser anfänglich erstelltes Pokemon komplett typisiert.

Dokument:

```
<pokemon nummer="4">
    <pokemonname>Glumanda</pokemonname>
</pokemon>
```

Bonus: XML Schemata (Werkzeuge zur Bindung komplexer Typen)

Attribute eines komplexen Typs kann man wie folgt angeben:

- `xs:sequence`:
  Die Inhalte müssen in exakt der Reihenfolge erscheinen, in der sie angegeben werden.
- `xs:all`
  Die Inhalte dürfen in beliebiger Reihenfolge angegeben werden. Jedes Element muss dabei exakt einmal instanziiert werden. Das hinzufügen weiterer Elemente, die nicht angegeben wurden, ist nicht erlaubt.
- `xs:choice`
  Von den aufgeführten Inhalten muss genau ein frei wählbares instanziiert werden.

Diese Kompositoren können um weitere Tags ergänzt werden.

- `minOccurs` gibt an wie oft ein Element mindestens erscheinen muss
- `maxOccurs` gibt an wie oft ein Element maximal erscheinen darf
- `default` gibt den Wert eines Elements an, wenn es vorhanden jedoch leer ist.

Bonus: Abgeleiteter Typ

Durch *abgeleitete Typen* wird die Basis eingeschränkt oder erweitert.

Mögliche Tags sind:

- `restriction`: Einschränkung des Wertebereichs
- `list`: Auflistung verschiedener Werte
- `union`: Vereinigung mehrerer Typen
- `extension`: Erweiterung in einen komplexen Typen

Beispiel: XML-Schema

Fassen wir nun einmal alle Beispiele zusammen und ergänzen `Team`:

Schema:

```xml
<xs:element name="pokemon" type="pokemonTyp" />
<xs:element name="team" type="teamTyp" />

<xs:simpleType name="pokemonNummer" base="xs:integer">
    <xs:minInclusive value="1" />
    <xs:minInclusive value="905" />
</xs:simpleType>

<xs:complexType name="pokemonTyp" xmlns:typ="https://kosy.paddel.xyz/typ">
    <xs:attribute name="nummer" type="pokemonNummer" />
    <xs:sequence>
        <xs:element name="pokemonname" type="xs:string" />
        <xs:element name="primaertyp" type="typ" default="Normal" />
        <xs:element name="sekundaertyp" type="typ" minOccurs="0" />
    </xs:sequence>
</xs:complexType>

<xs:complexType name="teamTyp">
    <xs:sequence>
        <xs:element name"pokemon" type="pokemonTyp" minOccurs="0" maxOccurs="6" />
    </xs:sequence>
</xs:complexType>
```

Dokument:

```xml
<xs:include schemaLocation="./pokemon-types.xs">

<team>
    <pokemon nummer="4">
        <pokemonname>Glumanda</pokemonname>
        <primaertyp>Feuer</primaertyp>
    </pokemon>
    <pokemon nummer="25">
        <pokemonname>Pikachu</pokemonname>
        <primaertyp>Elektro</primaertyp>
    </pokemon>
</team>
```

Bonus: Objekt Mappings

XML Strukturen werden genutzt, um Objekte zu verknüpfen.

So verschwimmen die Grenzen zwischen XML, Programmiersprachen und Datenbanken:

```xml
<pokemon nummer="4">
    <pokemonname>Glumanda</pokemonname>
</pokemon>
```

```java
public class Pokemon {
    int nummer;
    String name;

    public Pokemon (int nummer, String name) {
        this.nummer = nummer;
        this.name = name;
    }
}
```

```
CREATE TABLE pokemon {
    nummer int,
    name varchar(255)
};
```

Definition: SOAP

SOAP (ursprünglich für *Simple Object Access Protocol*) ist ein Netzwerkprotokoll mit dessen Hilfe Daten zwischen Systemen ausgetauscht und Remote Procedure Calls durchgeführt werden können.

Es ist ein industrieller Standard des WWW-Consortiums (W3C), welcher ein XML definiertes, anwendungsunabhängiges Nachrichtenformat darstellen soll.

Zusätzlich definiert SOAP, wie Anwendungsdaten kodiert werden sollen.

SOAP stellt einige feste Regeln auf:

- Erlaubte Tags und Attribute werden eingeschränkt.
- Es legt fest, wie Daten abzubilden oder zu interpretieren sind.
- Konventionen für Prozeduren bzw. Methodenaufrufe werden dargestellt
- Existierende, jedoch nicht festgelegte, Transportprotokolle werden genutzt.

4.3 Objektserialisierung in Java

Bonus: Objektserialisierung in Java

Mit Hilfe der Klasse `ObjectInputStream` bzw. `ObjectOutputStream` kann man Objekte serialisieren bzw. deserialisieren.

- `writeObject()` (wandelt Objekt in Byte-Stream um)
- `readObject()` (wandelt Byte-Stream in Objekt um)

Im Unterschied zu allgemeinen I/O- bzw. Streaming-Konzepten beruht die Objektkommunikation auf Interfaces. Somit ist ebenfalls eine komplett selbstcodierte Serialisierung möglich.

4.4 JSON

Definition: JSON

JSON-Dokumente beinhalten gültiges JavaScript. Sie haben eine einfache Struktur und sind typisiert aufgebaut.

Mögliche Typen sind:

- `Number`
- `String`: `"..."`
- `Boolean`: `true` / `false`
- `Array`: `[...]`
- `Object`: `{...}`
- `null`

Obacht: JSON deckt nicht alle möglichen JavaScript-Werte ab.

`NaN`, `Infinity` werden zu `null` serialisiert.

`Function`- und `RegeExp`-Objekte werden verworfen.

Beispiel: JSON

```
{
    "pokemonTeam" : [
        {
            "nummer" : 4,
            "name" : "Glumanda"
        },
        {
            "nummer" : 25,
            "name" : "Pikachu"
        }
    ]
}
```

Bonus: JSON Serialisierung

Meistens verwendet man automatische Marshaller bzw. Unmarshaller[a].

Alternativ kann man sich Objekte selber erstellen:

```
public class Pokemon {
    int nummer;
    String name;
}
Pokemon p = new Pokemon(4, "Glumanda");

JSONObject json = new JSONObject(p);
```

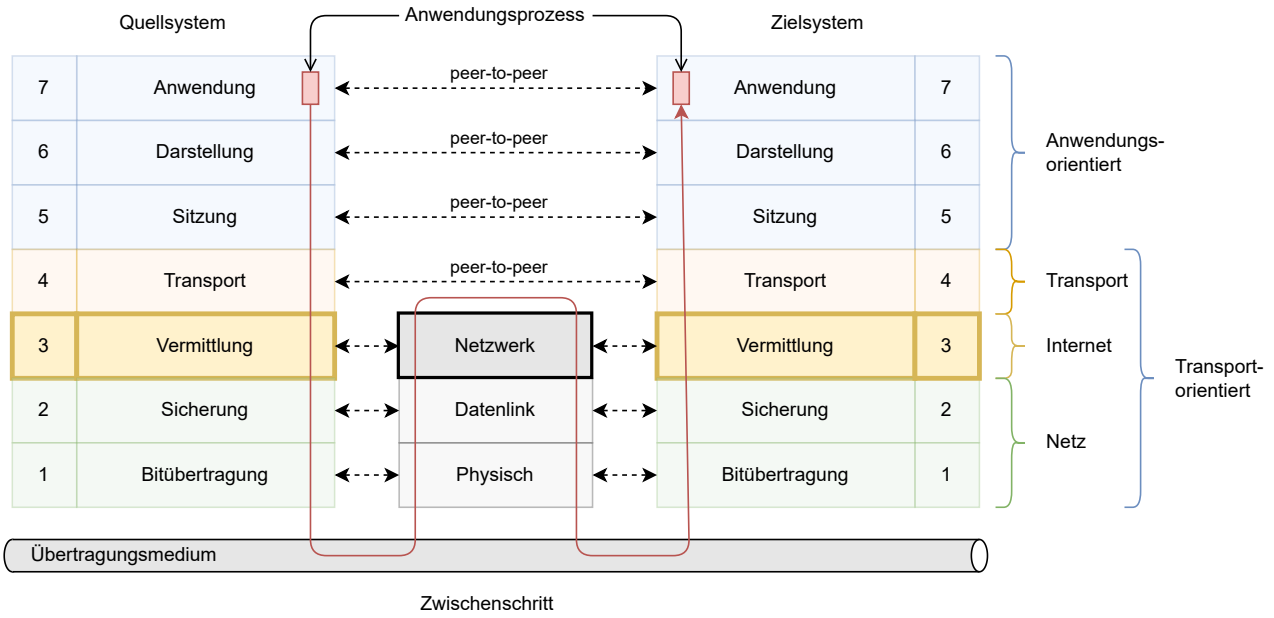
```
{
    "name" : "Glumanda",
    "nummer" : 4
}
```

Dabei werden alle Attribute alphabetisch sortiert.

Eine Formatierung mit Zeilenumbrüchen oder Einrückung findet nicht statt.

[a] wie Jackson (Java) oder Json.NET bzw. System.Text.Json (.NET)

5 Internet und IP-Adressen



5.1 Internet

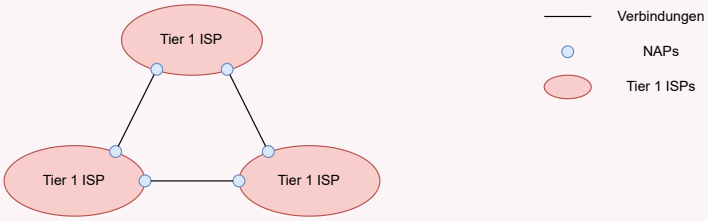
Definition: Internet

Das *Internet*, ist ein weltweiter Verbund von Rechnernetzwerken, den autonomen Systemen. Router werden als koppelnde Elemente zwischen den Teilnetzen genutzt.

Bonus: Aufbau des Internets

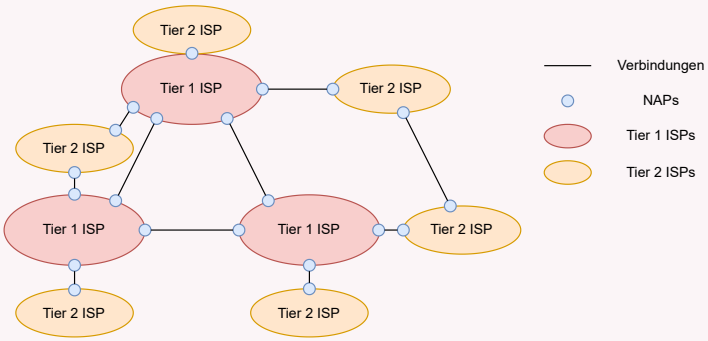
Die Basis des Internets bilden *Tier 1* Internet Service Provider (ISPs). Diese treten gleichberechtigt untereinander auf und sind durch vertraglich regulierte Verbindungen angebunden.

Als Verbindung zwischen den ISPs dienen *Network Access Points* (NAPs) oder *Internet Exchange Points* (IXPs)



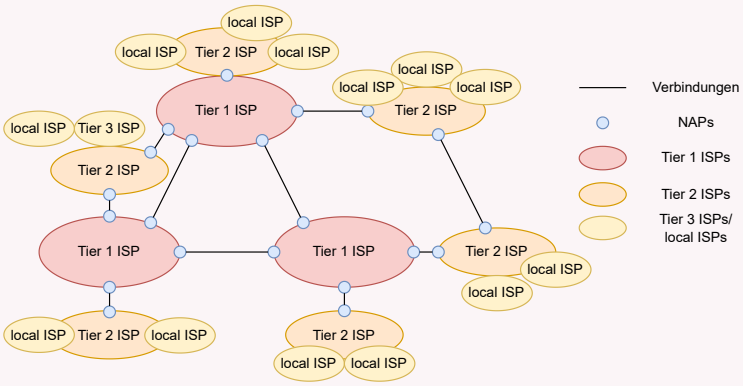
Tier 2 ISPs sind national bzw. regional und sind immer an einen oder mehrere Tier 1 ISPs angeschlossen.

Sie treten dabei gegenüber *Tier 1* ISPs als KundIn auf.

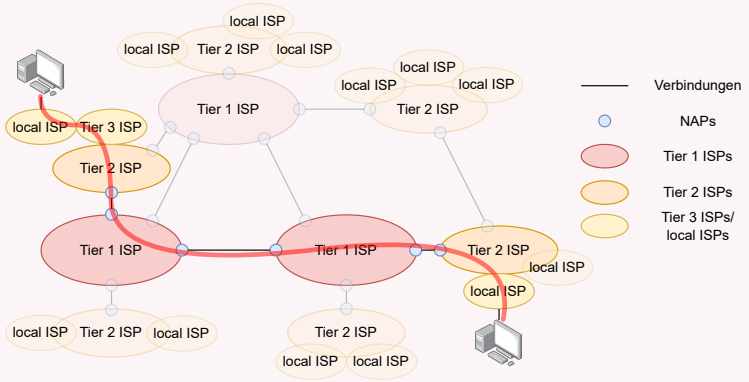


Tier 3 ISPs binden den bzw. die KundIn beispielsweise über DSL direkt an das gesamte Netzwerk an.

Sie treten dabei gegenüber Tier 2 ISPs selber als KundIn auf.



Bonus: Aufbau des Internets



Bonus: Nachrichtenzustellung

Die Herausforderung in diesem großen Netz aus Netzen ist es Datenpakete von einem beliebigen Endgerät zu einem bestimmten Ziel zu versenden.

Damit dies gelingt, wird in der Vermittlungsschicht einer der möglichen Wege zum Zielgerät gewählt. Die Wegwahl findet durch *Routing-Protokolle* statt.

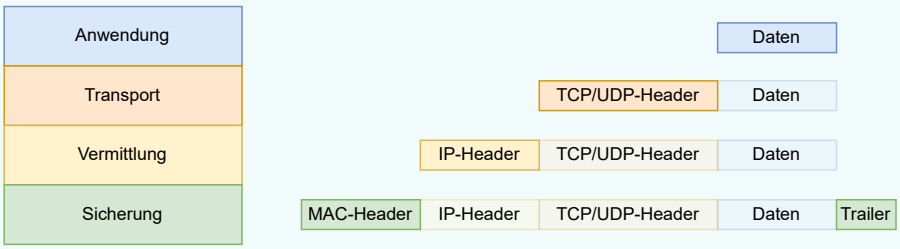
Die Entscheidung erfolgt durch *Routing-Tabellen*. In diesen lassen sich - je nach Protokoll - definierte Wege finden, unter denen man das autonome System des Zielgeräts findet bzw. welcher Weg befolgt werden soll, wenn das Zielgerät komplett unbekannt ist.

7	Anwendung
6	Darstellung
5	Sitzung
4	Transport
3	Vermittlung
2	Sicherung
1	Bitübertragung

In der Vermittlungsschicht wird größtenteils das IP-Protokoll genutzt, um die Pakete zu adressieren und Regeln zur Paketbehandlung aufzustellen.

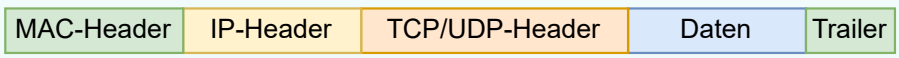
Definition: Kapselung von Protokoll-Headern

Zur erfolgreichen Nachrichtenzustellung erweitern die verschiedenen Protokollinstanzen ein von einer Anwendung produziertes Datenpaket um verschiedene fest definierte Header.



- TCP bzw. UDP-Header dienen zur eindeutigen Prozessadressieren durch Ports.
- TCP sichert darüber hinaus die Datenübertragung
- IP-Header identifizieren ein eindeutiges Quell und Zielsystem

Demnach entsteht folgendes versandfertiges Datagramm:



5.2 IP

Definition: IP

Das *Internet-Protokoll (IP)* bietet eine Ende-zu-Ende Kommunikation zwischen Endgeräten im Internet. Derzeit wird flächendeckend *IPv4* bzw. *IPv6* eingesetzt.

IP ist paketvermittelnd, leitet also in sich geschlossene, unabhängige Dateneinheiten bzw. Datagramme ungesichert zwischen zwei Endpunkten weiter. Jedes Paket wird dabei in dem jeweiligen Router zwischengespeichert, überprüft und dementsprechend aussortiert oder weitergeleitet.

Da die Pakete parallel von den Routern abgearbeitet werden, kann es passieren, dass:

- Datagramme verloren gehen
- Datagramme sich gegenseitig überholen
- Datagramme mehrfach ankommen

Sitzungen werden durch einen exklusiven Kommunikationskanal über TCP simuliert.

Definition: Wegwahl im Internet-Protokoll

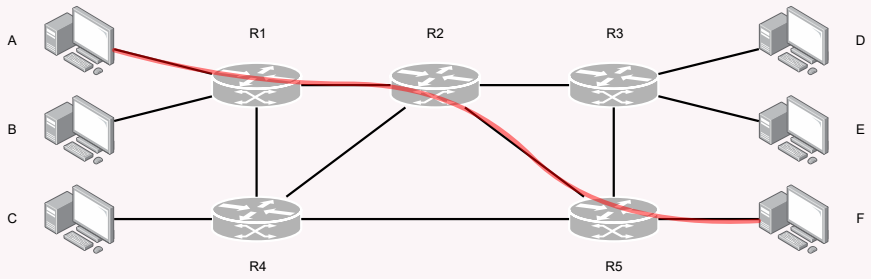
Die IP-Implementierung eines Rechners oder Routers entscheidet eigenständig, wohin dieser ein Datagramm übertragen soll.

Diese Entscheidung wird auf Basis von Routing- bzw. Forwarding-Tabellen getroffen. Dazu werden die Informationen, normalerweise ausschließlich die Zieladresse des Pakets, aus dem IP-Header in der jeweiligen Tabelle abgefragt.

Router sind dabei meist an mehreren Netzen angeschlossen. Sie müssen zusätzlich noch überprüfen, ob sich das Zielgerät in einem anderen Netz befindet als das Quellgerät bzw. wie dieses Netz erreicht werden kann.

Beispiel: Wegwahl im Internet Protokoll

In folgendem Beispiel möchte *Rechner A* an Paket an *Rechner F* senden.



Rechner A

Ziel	Next Hop
*	→ R1

R1

Ziel	Next Hop
A	A
B	B
C	R4
D	R2
E	R2
F	→ R2

R2

Ziel	Next Hop
A	R1
B	R1
C	R4
D	R3
E	R3
F	→ R5

R5

Ziel	Next Hop
A	R2
B	R2
C	R4
D	R3
E	R3
F	→ F

Definition: IPv4-Adresse

Im Internet sind Endgeräte nicht über Namen erreichbar, sondern haben eine *IP-Adresse*.

Diese sind 32 bit bzw. 4 Byte lang und sind unterteilt in einen Bereich, um das Netz zu adressieren und einen Bereich um diesen Rechner in dem angegebenen Netz zu identifizieren. In Kombination ist die IP-Adresse (theoretisch) weltweit eindeutig.

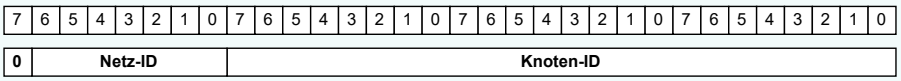
Zur einfachen und übersichtlichen Darstellung werden IP-Adressen in vier Blöcke mit je einem Byte aufgeteilt, von denen jeder eine dezimale Zahl darstellt.

Ein Beispiel ist das Klasse B Netz der RWTH Aachen University `134.130.0.0`.

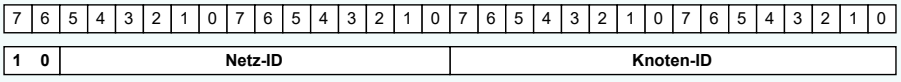
Definition: IPv4-Adressklassen

Um die verfügbaren IP-Adressen fair und sinnvoll aufzuteilen, wurden *Klassen* definiert. Mithilfe dieser Klassen können Anfragen von KundInnen, je nach benötigter Anzahl von IP-Adressen, schnell bearbeitet werden.

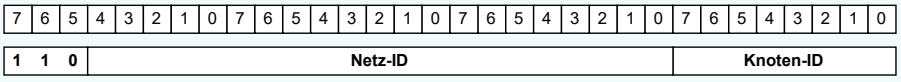
Klasse A: Für Netze mit bis zu 16 Mio. Knoten
(Knoten-ID: 0-127, 50% des IPv4-Adressraums)



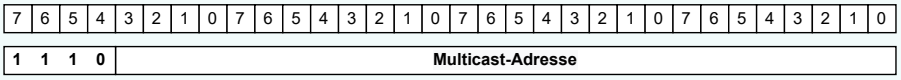
Klasse B: Für Netze mit bis zu 65.536 Knoten
(Knoten-ID: 128-191, 25% des IPv4-Adressraums)



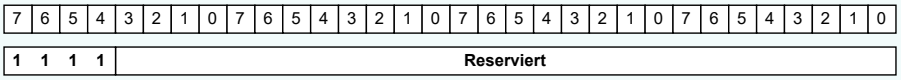
Klasse C: Für Netze mit bis zu 256 Knoten
(Knoten-ID: 192-223, 12.5% des IPv4-Adressraums)



Klasse D: Für Gruppenkommunikation
(Knoten-ID: 224-239, 6.25% des IPv4-Adressraums)



Klasse E: Reserviert für zukünftige Anwendungen
(Knoten-ID: 240-255, 6.25% des IPv4-Adressraums)



In allen Netzen ist die Knoten-ID `0..0` (0) für das Netz selber reserviert.

Des Weiteren wird für Broadcast Nachrichten an alle teilnehmenden Geräte im Netz die Knoten-ID `1..1` (A: 127, B: 191, bzw. C: 223) genutzt.

Bonus: Loopback-Adresse

Um Kommunikation mit Anwendungen auf dem selben Rechner zu führen, kann man die IPv4-Adresse `127.0.0.1` nutzen. Diese virtuelle Netzwerkkarte verfolgt nicht die tieferen Schichten des ISO-OSI-Modells, sondern geht „zurück“ über die Transportschicht an die angesprochene Anwendung.

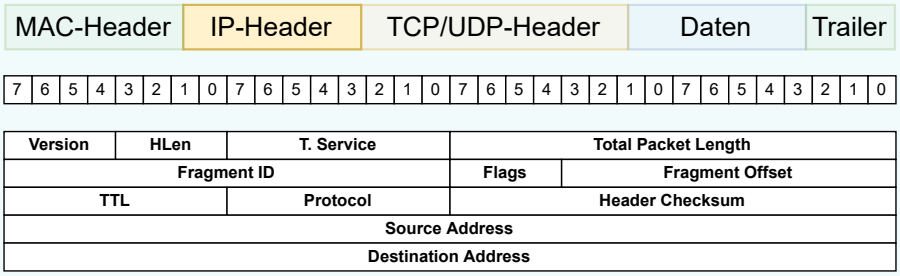
Definition: IP-Subnetze

Wenn wir nun die bestehenden Netzwerkklassen mit unseren neuen bekannten Subnetzmasken zusammenfügen, können wir große Netze in kleine durch Router getrennte Bereiche aufteilen.

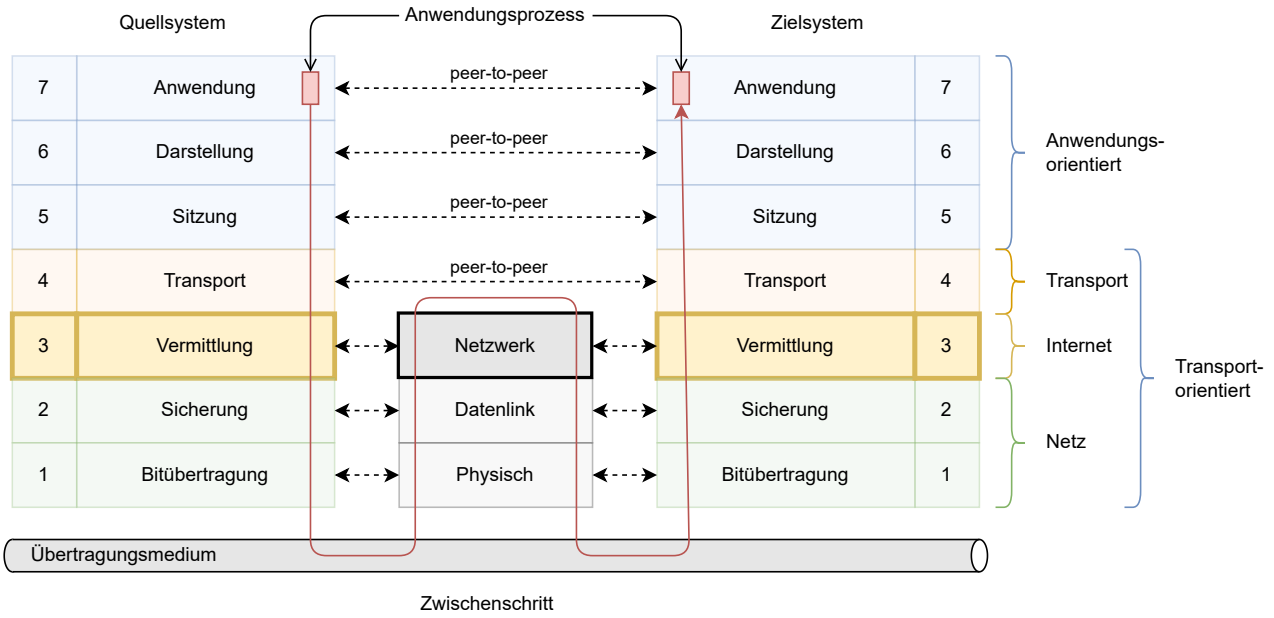
Aufgrund des Präfix erkennt man an der IPv4-Adresse die Adressklasse. Anhand der Subnetzmaske erkennt man die Länge der Netz-ID.

Stimmen die Längen der Netz-ID der Adressklasse und die Gesamtlänge der Netz-ID überein, haben wir keine Subnetze. Ist die Gesamtlänge der Netz-ID länger, können wir errechnen, wie viele Subnetze möglich sind.

Definition: IP-Header



6 CIDR



Definition: CIDR

Classless Inter-Domain Routing (CIDR) beschreibt ein Verfahren zur effizienteren Nutzung des bestehenden 32-Bit-IP-Adress-Raumes für IPv4.

Mit CIDR entfällt die feste Zuordnung einer IPv4-Adresse zu einer Netzklasse (A, B, C, ...), aus welcher aus den ersten beiden Bits des ersten Oktetts die Präfixlänge der jeweiligen Netzklasse hervorging.

Die Präfixlänge ist mit CIDR frei wählbar und muss deshalb beim Aufschreiben eines IP-Subnetzes mit angegeben werden. Dazu verwendet man häufig eine *Netzmaske*.

Bei CIDR führte man als neue Notation so genannte Suffixe ein. Das Suffix gibt die Anzahl der 1-Bits in der Netzmaske an.[a]

[a] Bei IPv6 ist die Notation gleich wie beim CIDR in IPv4 und besteht aus IPv6-Adresse und Präfixlänge.

Bonus: Probleme IPv4

Da nur ca. 4.3 Mrd. (2^{32}) Adressen zur Verfügung stehen, kann nicht jedes Gerät eine eigene weltweit eindeutige IP-Adresse erhalten. Anfänglich hatte niemand damit gerechnet, dass dieses neue Phänomen „Internet“ derart stark wächst.

Um dieses Problem zu lösen, bzw. die resultierenden Auswirkungen aufzuschieben, wurden private Netze eingeführt. Da ebenfalls festgestellt wurde, dass Adressklassen auf Grund hoher Verschwendung an überschüssigen IP Adressen nicht zielführend war[a], konnte man die Grenze zwischen Netz-ID und Knoten-ID nun beliebig wählen. Alternativ erhält man mehrere kleine Netze, die man durch Router verbinden muss.

Jedes IP Netz benötigt ab jetzt ebenfalls eine Subnetzmaske. Diese wird genutzt um eindeutig zwischen Netz und Knoten-ID zu unterscheiden.

Die Subnetzmaske kann entweder in IPv4- oder in CIDR-Schreibweise angegeben werden. In CIDR wird die Anzahl der 1en in der Subnetzmaske hinter der IPv4 Adresse angegeben. Die Stellen der Knoten-ID können weggelassen werden.

Um eine gewisse Struktur in private Netze zu bringen, wurden gewisse Richtlinien geschaffen, um aus den historischen Klassen private Netze zu erstellen. Jedes übersetzte Netz hat 24 frei wählbare Stellen. Im Gegensatz dazu kann man in eigenen privaten Netzen alle Stellen der IP-Adresse frei wählen. Dabei entscheidet die Länge der Subnetzmaske (snm) über die Anzahl der verfügbaren IPv4-Adressen ($2^{32-\text{snm}}$).

Klasse	CIDR-Notation	Anzahl Netze	Anzahl Adressen	Netze
A	`10/8`	$2^0 = 1$	$2^{24} = 16.777.216$	`10/8`
B	`172.16/12`	$2^4 = 16$	$2^{20} = 1.048.576$	`172.16/16` bis `172.31/16`
C	`192.168/16`	$2^8 = 256$	$2^{16} = 65.536$	`192.168.0/24` bis `192.168.255/24`
/24	`192.168.0/24`		$2^{32-24} = 2^8 = 256$	

[a] Wenn man mehr als 254 Endgeräte anschließen möchte, benötigt man bereits ein Klasse B Netz. Es bleiben bis zu 64.281 Adressen ungenutzt

Beispiel: IPv4-Netz

Betrachten wir ein typisches IPv4-Netz mit Subnetzmaske: `192.168.0.0` mit `255.255.255.0`

bzw. in CIDR: `192.168.0.0/24` bzw. `192.168.0/24`

Wenn wir eine beliebige IP-Adresse aus diesem Netz erhalten, können wir mithilfe logischer Gatter zuerst das Netz eindeutig identifizieren, und dann den Host adressieren.

z.B. `192.168.0.4/24`

IP-Adresse	`1100 0000 . 1010 1000 . 0000 0000 . 0000 0100`	`192.168.0.4`
Subnetzmaske	`1111 1111 . 1111 1111 . 1111 1111 . 0000 0000`	`255.255.255.0`
Netz-ID[a]	**`1100 0000 . 1010 1000 . 0000 0000`** `. 0000 0000`	**`192.168.0`**`.0`
Geräte-ID[b]	`0000 0000 . 0000 0000 . 0000 0000 .` **`0000 0100`**	`0.0.0.`**`4`**

[a] Netz-ID = IP-Adresse $\wedge$ Subnetzmaske

[b] Geräte-ID = IP-Adresse $\wedge$ $\overline{\text{Subnetzmaske}}$

Beispiel: IP-Subnetze

Nehmen wir das Klasse B Netz der RWTH Aachen University: `134.130.0.0`[a]

In diesem Klasse B Netz sind folgende Subnetze gegeben:
`134.130.1.0/24`
`134.130.2.0/24`
`134.130.3.0/24`

Durch die Subnetzmaske wissen wir, dass die Netz-ID die ein Byte länger ist, als die Netz-ID der Netzklasse (Klasse B hat `/16`). Dementsprechend können wir bis zu $2^8 = 256$ Subnetze mit jeweils bis zu 254[b] Endgeräten erstellen.

z.B. `134.130.1.4/24`

IP-Adresse	`1000 0110 . 1000 0010 . 0000 0001 . 0000 0100`	`134.130.1.4`
Subnetzmaske	`1111 1111 . 1111 1111 . 1111 1111 . 0000 0000`	`255.255.255.0`
IP-Klassenmaske[c]	`1111 1111 . 1111 1111 . 0000 0000 . 0000 0000`	`255.255.0.0`
Netz-ID[d]	**`1000 0110 . 1000 0010`** `. 0000 0000 . 0000 0000`	**`134.130`**`.0.0`
Subnetz-ID[e]	`0000 0000 . 0000 0000 .` **`0000 0001`** `. 0000 0000`	`0.0.`**`1`**`.0`
Geräte-ID[f]	`0000 0000 . 0000 0000 . 0000 0000 .` **`0000 0100`**	`0.0.0.`**`4`**

[a] Netz-Adresse startet in binär mit `10`. Daher wissen wir, dass es sich um ein Klasse B Netz handelt
[b] Obacht: Die logische Aufteilung in Subnetze ist sinnvoll, jedoch gehen bei jedem Subnetz jeweils zwei IP-Adressen für Netz-Adresse und Broadcast verloren. Dementsprechend sollte man Subnetze ausschließlich für unabhängige Instanzen nutzen. z.B. Institute der RWTH, Dezernate der FH-Aachen etc.
[c] Bekannt durch IP-Klassenpräfix
[d] Netz-ID = IP-Adresse $\wedge$ Subnetzmaske $\wedge$ IP-Klassenmaske
[e] Subnetz-ID = IP-Adresse $\wedge$ Subnetzmaske $\wedge$ $\overline{\text{IP-Klassenmaske}}$
[f] Geräte-ID = IP-Adresse $\wedge$ $\overline{\text{Subnetzmaske}}$

Definition: Longest Prefix Match

Beim *Longest Prefix Match* geht es darum, wie ein Router möglichst effizient eine maximal mögliche Übereinstimmung der Zieladresse mit einer gespeicherten IP-Adresse aus seiner internen Routingtabelle findet.

Der Routenalgorithmus kommt dann zum Einsatz, wenn die Routingtabelle mehrere potentiell zur Zieladresse eines Paketes passende Adressbereiche beinhaltet, und gehört nach der Ablösung Netzklassen durch Adressen und frei wählbare Netzmasken (CIDR) zu den Standardverfahren.

Beispiel: Longest Prefix Match

IPv4 Routingtabelle eines Routers:

Nr.	Netzwerk-Adresse	Subnetzmaske	Ziel
1	`198.51.100.0`	`/24`, `255.255.255.0`	Schnittstelle 1
2	`198.51.100.64`	`/26`, `255.255.255.192`	Schnittstelle 2
3	`198.51.100.128`	`/26`, `255.255.255.192`	Schnittstelle 3

Es wird ein Paket mit der IPv4-Adresse `198.51.100.78` empfangen.

Dann gilt:

Adresse/Netz (CIDR)	Binärdarstellung	Match
`198.51.100.78/32`	`11000110 . 00110011 . 01100100 . 01001110`	
`198.51.100.0/24`	**`11000110 . 00110011 . 01100100 . 0`**`0000000`	25 Bit
`198.51.100.64/26`	**`11000110 . 00110011 . 01100100 . 01`**`000000`	26 Bit
`198.51.100.128/26`	**`11000110 . 00110011 . 01100100`** `. ~~10~~0000000`	-

Die längste Übereinstimmung mit dem jeweiligen vollständigen fixen Adressteil liegt bei Eintrag 2 vor, nämlich 26 Bit.

Weiterleitung des Paketes entsprechend über Schnittstelle 2.

Bonus: Verkleinerung der Routing-Tabelle durch CIDR

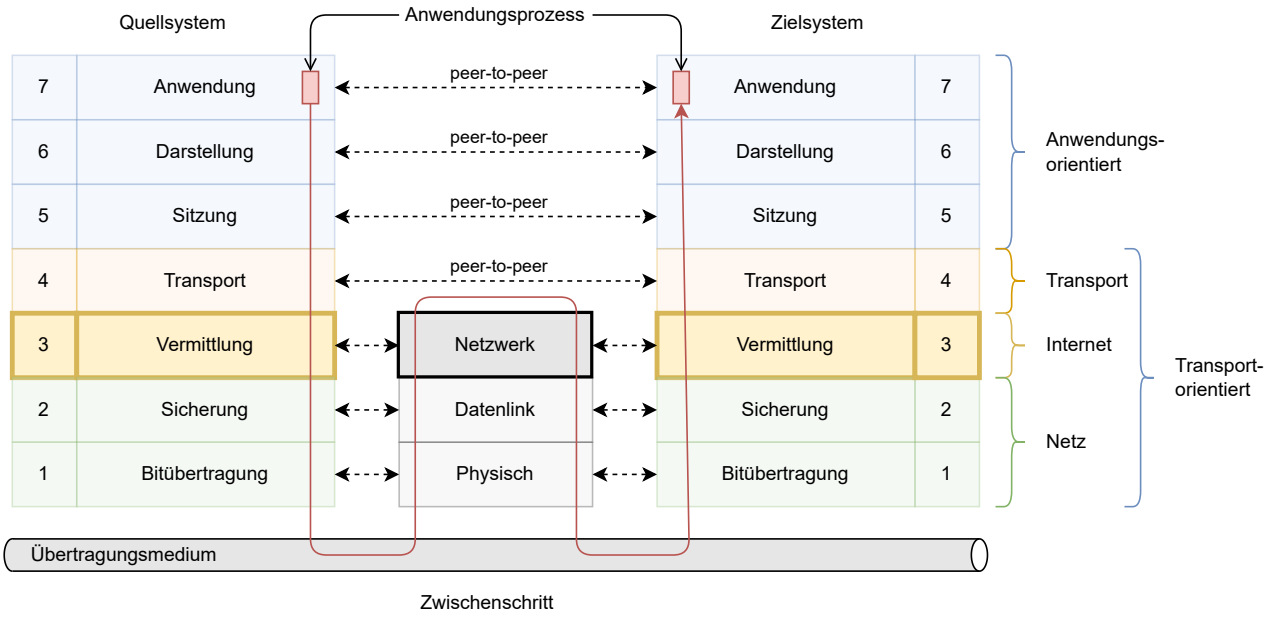
Der Longest Prefix Match erlaubt das Zusammenfassen von Routen.

Zu beachten ist:

- Ein Zusammenfassen ist nur möglich, wenn gleiche Ziele verwendet werden.
- Die relaxierte interpretation der Netzwerkadresse kann ggf. andere, nicht gewollte Einträge umfassen.[a]
- `0.0.0.0/0` ist die Default-Route.

[a] Hier hilft ggf. die Longest Prefix Match-Regel. Hierzu muss aber der Präfix der überschriebenen Regel länger sein.

7 Routingprotokolle



Definition: Routing

Das *Weiterleiten (Routing)* erfüllt die wichtige Aufgabe, einzelne Teilstrecken des Kommunikationsnetzes so zu verbinden, dass eine Ende-zu-Ende-Kommunikation zwischen den angeschlossenen Teilnehmenden möglich wird.

Das Routing kann in zwei Teilprozesse zerlegt werden:

- Das eigentliche Routing (die Control-Plane) ist verantwortlich für Wegewahl, mit der die Ende-zu-Ende-Kommunikation erreicht wird.
- Das Weiterleiten (Forwarding) der Pakete (die Data-Plane) gemäß den Vorgaben der Control-Plane.

Die Forwarding-Entscheidung wird für jedes Paket durchgeführt, unabhängig von vorherigen Entscheidungen, auf Basis der Zieladresse.

Bonus: Routing-Metrik

Eine *Routing-Metrik* ist ein numerischer Wert, mit dessen Hilfe ein Routing-Algorithmus feststellen kann, ob eine Route im Vergleich zu einer anderen besser ist.

Metriken können Informationen wie z. B. Bandbreite, Verzögerung, Hop Count, Pfadkosten, Last, MTU, Verlässlichkeit und Kommunikationskosten berücksichtigen. Falls z. B. die Distanz die ausschlaggebende Metrik bei der Bestimmung einer Route ist, wird im Falle mehrerer möglicher Routen diejenige mit dem kleinsten Wert (d. h. der niedrigsten Distanz) gewählt.

Nicht immer lässt sich aber die beste Route anhand des kleinsten Werts bestimmen, da z. B. eine höhere Bandbreite durch einen höheren Metrik-Wert repräsentiert wird.

In der Routing-Tabelle werden oft nur die bestmöglichen Routen gehalten, während Link-State- oder topologische Datenbanken, aus denen die Routing-Tabelle gewonnen wird, sämtliche Informationen beinhalten.

Definition: Routing-Algorithmen

Routing-Algorithmen benutzen zwei grundlegende Verfahrensweisen:

- Distanzvektor-Protokolle
- Link-State-Routing-Protokolle

Definition: Distanzvektor-Protokoll

„Teile deinen Nachbarn mit, wie für dich die Welt aussieht“: *Distanzvektor-Protokolle* sorgen dafür, dass sich die Router untereinander nur mitteilen, wie gut sie an verschiedene Zielknoten angebunden sind.

Vorgehen:

1. Jeder Router schickt regelmäßig seinen Nachbarn einen Distanzvektor, der die Kosten zu allen anderen Knoten im Netz enthält.
2. Jeder Router aktualisiert daraufhin die eigene Liste mit den Kosten zu den anderen Knoten und wählt dabei den Weg mit den geringsten Kosten.

Durch Auswahl des für ein bestimmtes Ziel optimalen Nachbarn wird die Lösung des Kürzeste-Wege-Problems somit auf mehrere Router verteilt.

Ein Beispiel ist der *Bellman-Ford-Algorithmus*.

Vorteile:

- leicht zu implementieren
- einfache Berechnung
- neue, bessere Routen werden im Netz schnell propagiert

Nachteile:

- Ausfall von Routen bzw. Routern führt zum *Count-to-Infinity-Problem*

Definition: Link-State-Routing-Protokoll

„Teile der Welt mit, wer deine Nachbarn sind“: *Link-State-Routing-Protokolle* sorgen dafür, dass nach einiger Zeit jeder Router die vollständige Topologie des Netzwerks kennt und sich die kürzesten Wege darin selbst ausrechnen kann.

Vorgehen:

1. Nachbarn und deren Netzadressen ermitteln
2. Kosten zu jedem Nachbarn festlegen
3. Paket zusammenstellen, in dem alles steht was bisher gelernt wurde
4. Paket an alle anderen Router senden und von allen anderen Routern derartige Pakete empfangen
5. kürzesten Pfad zu allen Routern berechnen

Dadurch, dass die gesamte Topologie in jedem Router abgebildet ist, kann der Dijkstra-Algorithmus angewandt werden.

Vorteile:

- optimale Lösung kann berechnet werden
- gute Performance bei Änderungen des Netzwerkes

Nachteile:

- höherer Rechenaufwand im Router
- höhere Komplexität bei der Versendung der Link-State-Pakete an alle anderen Router und deren Analyse
- höhere Netzlast

Algorithmus: Wiederholung Dijkstra-Algorithmus

Gegeben: Graph $G = (V, E)$, dessen Bewertungsfunktion die Eigenschaften hat:

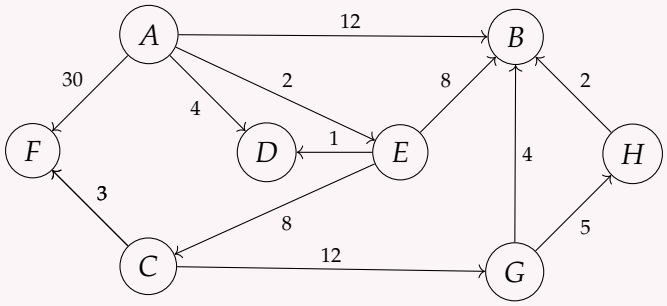
- Jede Kante von v_i nach v_j hat nicht-negative Kosten: $C(i, j) \geq 0$
- Falls keine Kante zwischen v_i und v_j: $C(i, j) = \infty$
- Diagonalelemente: $C(i, i) = 0$

Menge S: die Knoten, deren günstigste Wegekosten von der vorgegebenen Quelle (Startknoten) bereits bekannt sind.

1. Initialisierung: $S = \{\text{Startknoten}\}$
2. Beginnend mit Quelle alle ausgehenden Kanten betrachten (analog Breitensuche). Nachfolgerknoten v mit günstigster Kante zu S hinzunehmen.
3. Jetzt: Berechnen, ob die Knoten in $V \setminus S$ günstiger über v als Zwischenweg erreichbar sind, als ohne Umweg über v.
4. Danach: Denjenigen Knoten v' zu S hinzunehmen, der nun am günstigsten zu erreichen ist. Bei zwei gleich günstigen Knoten wird ein beliebiger davon ausgewählt.
5. Ab Schritt 3 wiederholen, bis alle Knoten in S sind.

Zeitkomplexität (bei Speicherung des Graphen mit Adjazenzmatrix): $\mathcal{O}(|V|^2)$

Beispiel: Dijkstra-Algorithmus



v_i	$d[2]$	$d[3]$	$d[4]$	$d[5]$	$d[6]$	$d[7]$	$d[8]$	$p[2]$	$p[3]$	$p[4]$	$p[5]$	$p[6]$	$p[7]$	$p[8]$
	B	C	D	E	F	G	H	B	C	D	E	F	G	H
A	12		4	**2**	30			A		A	A	A		
E	10	10	**3**	**2**	30			E	E	E	A	A		
D	**10**	10	**3**	**2**	30			E	E	E	A	A		
B	**10**	**10**	**3**	**2**	30			E	E	E	A	A		
C	**10**	**10**	**3**	**2**	**13**	22		E	E	E	A	C	C	
F	**10**	**10**	**3**	**2**	**13**	**22**		E	E	E	A	C	C	
G	**10**	**10**	**3**	**2**	**13**	**22**	**27**	E	E	E	A	C	C	G
H	**10**	**10**	**3**	**2**	**13**	**22**	**27**	E	E	E	A	C	C	G

https://youtu.be/4pBP2hbnGso (Herleitung und Erklärung)

Algorithmus: Wiederholung Bellman-Ford-Algorithmus

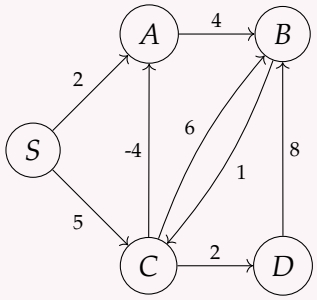
Gegeben: Graph $G = (V, E)$, dessen Bewertungsfunktion die Eigenschaften hat:

- Falls keine Kante zwischen v_i und v_j: $C(i, j) = \infty$
- Diagonalelemente: $C(i, i) = 0$

1. Initialisierung: Startknoten s, Distanz zu $V \setminus \{s\}$ auf ∞ setzen
2. Für jede Kante (i, j):
   - Falls Distanz zu v_i bekannt:
     Falls $d(v_i) + C(i, j) < d(v_j)$, setze $d(v_j)$ auf $d(v_i) + C(i, j)$

Zeitkomplexität: $\mathcal{O}(|V| \cdot |E|)$

Beispiel: Bellman-Ford-Algorithmus



	S	A	B	C	D
0	0	∞	∞	∞	∞
1	0	2	∞	5	∞
2	0	1	6	5	7
3	0	1	5	5	7
4	0	1	5	5	7

Keine Änderungen nach Schritt 3 $\Longrightarrow$ Fertig!

Bonus: Lokales vs. globales Routing

Jeder Rechner braucht eine Routing-Tabelle:

- im Internet:
  - meist Link-State-Verfahren[a] innerhalb autonomer Systeme (Backbones)
  - Distanzvektor-Verfahren[b] zur Koppluing autonomer Systeme
- in Firmen bzw. Stadtnetzen:
  - können State-Link- oder Distanzvektor-Verfahren nutzen
  - meist sinnvoller: statische Konfiguration

[a] z. B. Dijkstra
[b] z. B. Bellman-Ford

8 NAT

Bonus: Konfiguration von IPv4-Adressen

IPv4-Adressen müssen in jedem Endgerät, Router und sonstigen Netzwerkkomponenten konfiguriert werden.

Jeder Knoten muss folgende Daten besitzen:

- IP und Netzwerkmaske
- Standard-Gateway
- DNS-Server

Grundsätzlich existieren zwei Möglichkeiten dazu:

- Manuelle Konfiguration
- Konfiguration über ein Protokoll (z. B. DHCP)

Bonus: Probleme IPv4

Wir hatten bereits geklärt, dass IPv4 nur 2^{32} IP-Adressen zur Verfügung hat.

Da bereits relativ zeitnah nach dem Anpassen der Subnetzgrößen klar wurde, dass die gewonnene Anzahl der daraus resultieren IP-Adressen nicht ausreicht, hat man sich eine provisorische Lösung überlegt.

Definition: NAT

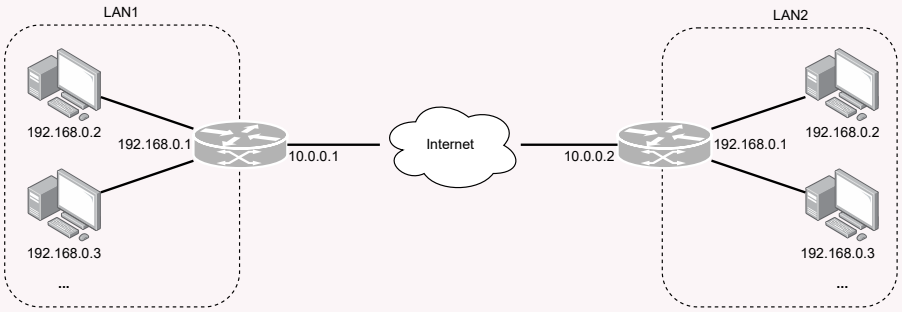
NAT (Network Adress Translation) verfolgt die Idee, dass KundInnen jeweils lediglich eine eindeutige öffentliche IPv4-Adresse erhalten.

Dementsprechend existiert für jedes Netz ebenfalls nur einen Zugangspunkt (Gateway) zu diesem Netz. Das Gateway hat dabei mindestens 2 Netzwerkkarten, um in das große Internet, sowie das lokale Netz zu senden.

Das private Netz nutzt meist einen der folgenden IP-Adressblöcke:

```
10.0.0.0/8:      10.0.0.0     bis  10.255.255.255
172.16.0.0/12:   172.16.0.0   bis  172.31.255.255
192.168.0.0/16:  192.168.0.0  bis  192.168.255.255
```

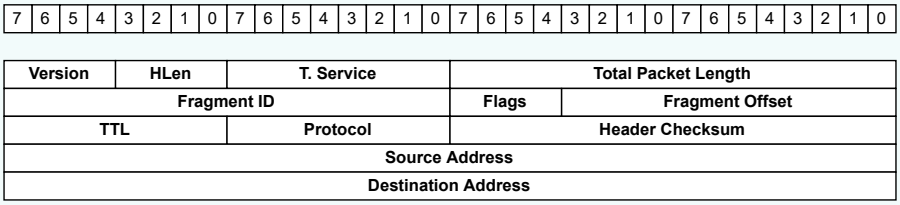
Beispiel: NAT



In diesem Beispiel „verbrauchen“ wir 2 globale IP-Adressen, können jedoch bis zu $(2^{16} - 3)$ Adressen in jeweils 2 Netzen anbinden.

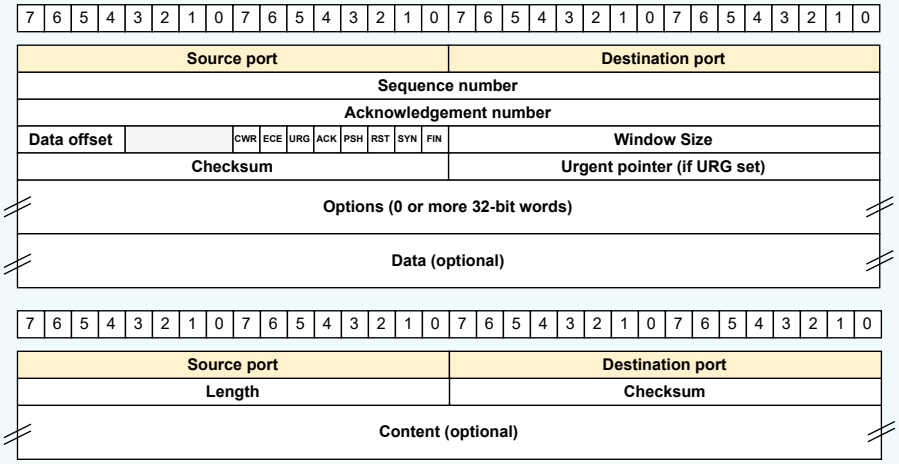
Definition: Adressübersetzung

Zur Identifikation des Absenders bzw. Empfängers müssen im IP-Header freie Felder gefunden werden. Im IP-Header steht jedoch nur noch 1 freies Bit zur Verfügung.



Zweite Idee: Nutze TCP bzw UDP Header. Hier kann man die 16 bit Portnummern nutzen um Clients eindeutig zu adressieren.

Hierbei wird jedoch die Schichtenarchitektur verletzt, da Layer 3 Details über Layer 4 kennt.



Bei TCP bzw. UDP wird sowohl der Destination Port als auch der Source Port mitgesendet.

Der *Destination Port* darf dabei von der sendenden Anwendung nicht verändert werden, da er definiert auf welchen Services zugegriffen wird.

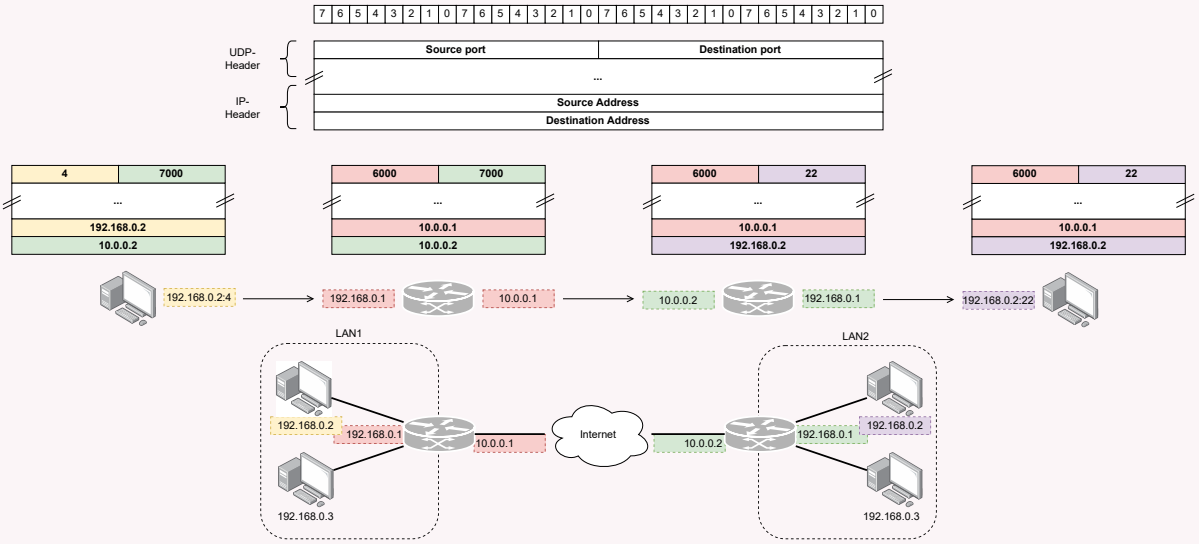
Der *Source Port* wird von Betriebssystem vergeben und ist ggf. nicht eindeutig (z. B. zwei Webserver mit jeweils Port `443`).

Daher ersetzen NAT-Router den Source Port durch eine ID, die den privaten Rechner bzw. die Verbindung identifiziert.

Im Antwortpaket muss der Destination Port extrahiert werden, und der NAT Router ersetzt die Ziel-IP mit der entsprechenden privaten IP und den Zielport mit dem ursprünglichen Port.

Beispiel: Adressübersetzung

In folgendem Beispiel sendet der Rechner `192.168.0.2` in `LAN1` über den Port `4` ein Paket an den Rechner `192.168.0.2` in `LAN2` an den Port `22`. Dieser „versteckt“ sich jedoch hinter dem öffentlichen Router `10.0.0.2` und dem Port `7000`.



Übersetzungstabelle Router (10.0.0.1)

IP-Adresse	Port	NAT-Port
192.168.0.2	4	6000
192.168.0.2	591	6001
192.168.0.3	4	6002

Übersetzungstabelle Router (10.0.0.2)

IP-Adresse	Port	NAT-Port
192.168.0.2	22	7000
192.168.0.3	22	7001

Bonus: NAT Nachteile

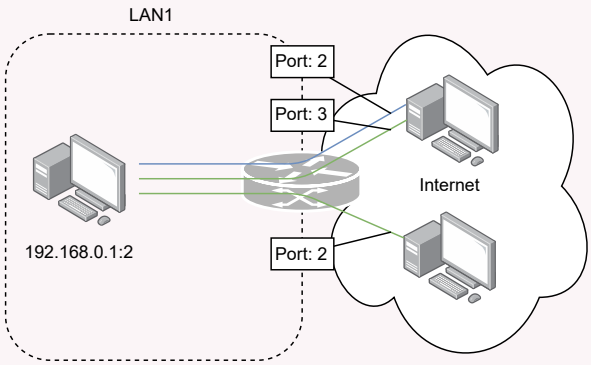
- Nicht jeder Rechner ist eindeutig per IP identifizierbar
- Verletzung des Schichtenmodells
- Übersetzungstabelle kann i. d. R. nur aufgebaut werden, wenn ein interner Knoten die Kommunikation initiiert
- Was ist, wenn nicht TCP oder UDP verwendet wird?
- Was passiert, wenn übergeordnete Protokolle die IP als Payload übertragen?
- Was passiert bei verschlüsselten Verbindungen?

Bonus: Sicherheitsstufen

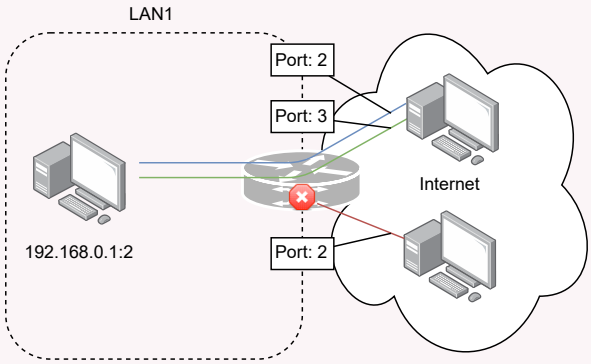
NAT kennt unterschiedliche *Sicherheitsstufen*:

- *Full Cone NAT*:
  Ein Port wird unbegrenzt nach außen freigegeben. Dadurch kann auch von außen ein Rechner die Kommunikation initiieren.
- *Restricted Cone NAT*:
  Ein Port wird nur nach dem Start der internen Kommunikation freigegeben. Die Quell-IP-Adresse wird bei Antworten überprüft.
- *Port Restricted Cone NAT*:
  Die Rückpakete werden zusätzlich auf den Source Port hin überprüft

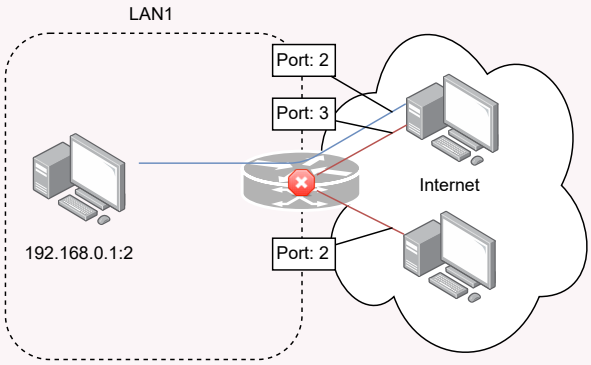
Beispiel: Full Cone NAT



Beispiel: Restricted Cone NAT



Beispiel: Port Restricted Cone NAT



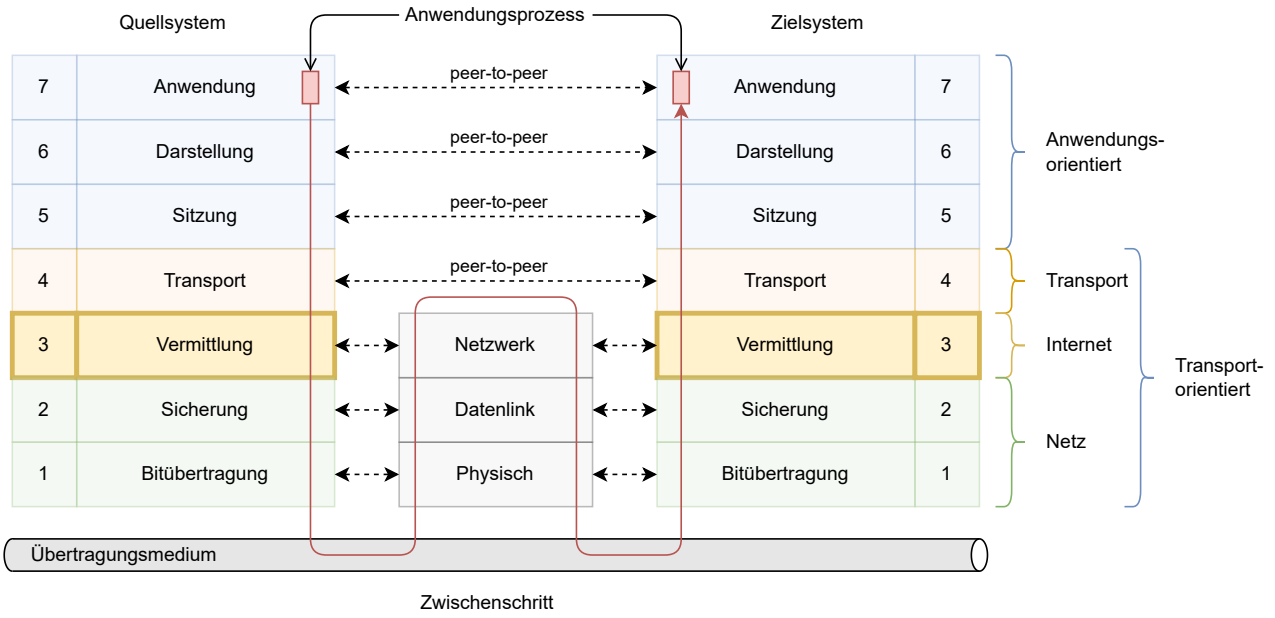
Bonus: Symmetric NAT

Bei *Symmetric NAT* wird für jeden Datenstrom aus dem internen Netz eine eigene Abbildung durchgeführt.

Wenn der Quell-Rechner z. B. vom selben Quell-Port aus mit zwei unterschiedlichen Ziel-Rechnern kommuniziert, wird für jede Verbindung eine eigene NAT-Tabelle angelegt.

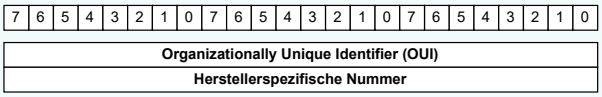
Nur der jeweilige Zielrechner darf jeweils mit den eingetragenen Portnummern antworten.

9 Adressprotokolle



9.1 ARP, RARP

Definition: MAC-Adresse



Die *MAC-Adresse* ist die Hardware-Adresse jedes einzelnen Netzadapters, die als eindeutiger Identifikator des Geräts in einem Rechnernetz dient. Man spricht auch von physischer Adresse oder Geräteadresse.[a]

Die MAC-Adresse enthält neben einer ID lediglich Informationen zum Hersteller, weshalb kein Routing mithilfe der MAC-Adresse möglich ist.

Die Broadcast-Adresse ist definiert als `FF:FF:FF:FF:FF:FF` definiert.

[a] Bei Apple wird sie Ethernet-ID, Airport-ID oder Wi-Fi-Adresse genannt, bei Microsoft Physikalische Adresse.

Definition: ARP

Das *Address Resolution Protocol (ARP)* ist ein Netzwerkprotokoll, das zu einer Netzwerkadresse der Internetschicht die physische Adresse (MAC-Adresse) der Netzzugangsschicht ermittelt und diese Zuordnung gegebenenfalls in den *ARP-Tabellen* der beteiligten Rechner hinterlegt.

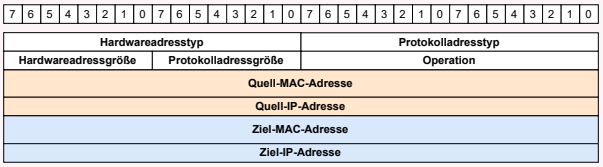
Der Ablauf ist z. B. bei Ethernet wie folgt:[a]

1. Es wird eine *ARP-Request* mit MAC-Adresse und IP-Adresse des anfragenden Rechners als Quelle und der IP-Adresse des gesuchten Empfängers als Ziel an alle Computer des lokalen Netzwerkes gesendet.
2. Empfängt ein Computer ein solches Paket, sieht er nach, ob dieses Paket seine IP-Adresse als Empfänger-IP-Adresse enthält.[b]
3. Wenn dies der Fall ist, antwortet er mit dem Zurücksenden seiner MAC-Adresse und IP-Adresse (*ARP-Reply*) per Broadcast oder als Unicast.
4. Der Empfänger trägt nach Empfang der Antwort die empfangene Kombination von IP- und MAC-Adresse in seine ARP-Tabelle, auch ARP-Cache genannt, ein.

[a] Für ARP-Request und ARP-Reply wird das gleiche Paketformat verwendet.

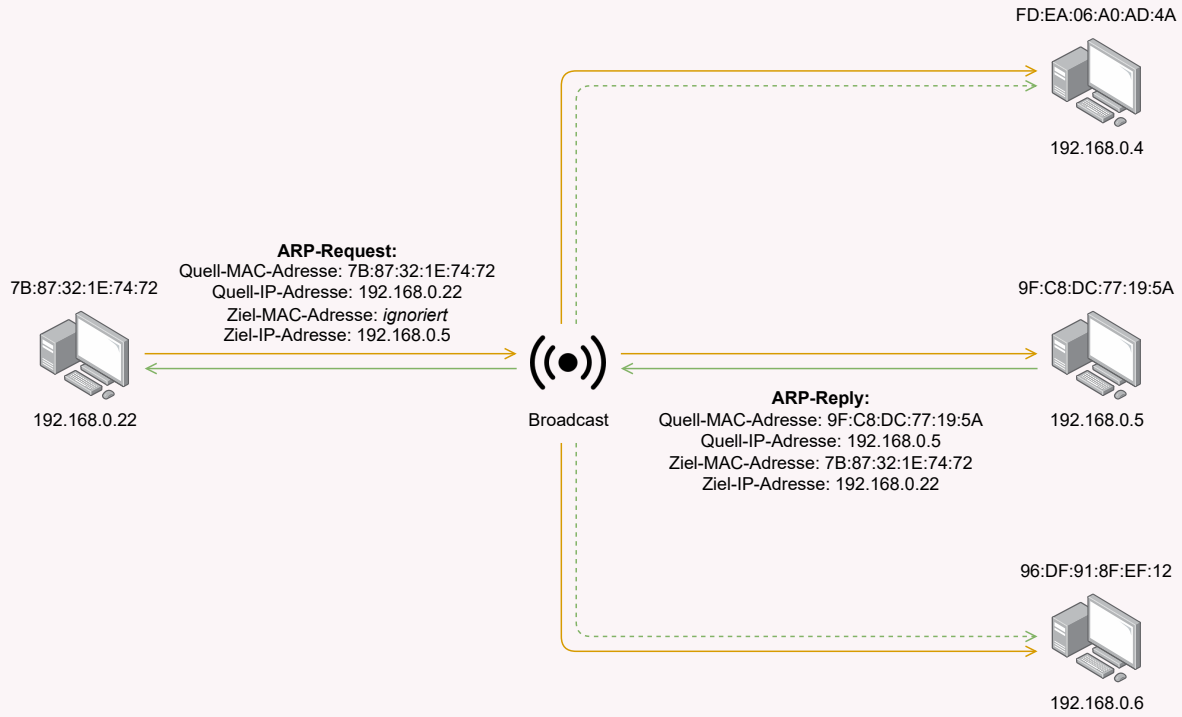
[b] Zusätzlich können die Empfänger des ARP-Requests ebenfalls die Kombination von IP-Adresse und MAC-Adresse des anfragenden Computers in ihre ARP-Tabelle eintragen bzw. einen bestehenden Eintrag aktualisieren. Insbesondere der Rechner mit der im ARP-Request angefragten IP-Adresse sollte diese Eintragung vornehmen, da anzunehmen ist, dass der ARP-Request als Vorbereitung für weitere Kommunikation auf höherer Protokollebene dienen soll, wofür er dann für eventuelle Antworten ebenfalls die MAC-Adresse des Anfragenden benötigt.

Bonus: ARP-Paket



Beispiel: ARP

Der Computer mit MAC-Adresse `7B:87:32:1E:74:72` und IP-Adresse `192.168.0.22` sucht die MAC-Adresse zum Computer mit der IP-Adresse `192.168.0.5`.[a]



[a] Die ARP-Reply wird dann auch über die gestrichelten Routen versandt, wenn Broadcast verwendet wird. Wird Unicast verwendet, gelangt die ARP-Reply nur an den ursprünglichen Sender.

Definition: RARP

Das *Reverse Address Resolution Protocol (RARP)* ist ein Netzwerkprotokoll, das die Zuordnung von Hardwareadressen zu Internetadressen ermöglicht.

RARP sendet dazu ein *RARP-Request*-Broadcast mit der eigenen MAC-Adresse als Inhalt an die am Netzwerk angeschlossenen Rechner. Ein RARP-Server, welcher alle Zuordnungen IP- zu MAC-Adressen kennt, sendet daraufhin eine Antwort mit der IP-Adresse an die anfragende MAC-Adresse (*RARP-Reply*).

Bonus: Probleme von RARP

Die vermeintliche Eindeutigkeit der MAC-Adresse darf nicht als Sicherheitskriterium angewandt werden. Es ist viel zu einfach, MAC-Adressen-Spoofing zu betreiben.

Gültige MAC-Adressen in einem Schicht-2-Netz können durch Abhören des Netzverkehrs ausfindig gemacht werden. Dazu ist lediglich der physische Zugang zum Netzwerk nötig. Die exklusive Vergabe von IP-Adressen nur an registrierte MAC-Adressen über RARP oder DHCP schließt also nicht aus, dass Unberechtigte Zugriff auf das Netzwerk erhalten.

Außerdem sind Ethernet-Broadcasts auf Subnetze beschränkt, so dass RARP nur in einem Subnetz eingesetzt werden kann. Wird ein lokales Netzwerk (LAN) in Subnetze aufgeteilt, muss in jedem dieser Subnetze, in dem RARP-fähige Terminals oder Workstations eingesetzt werden, ein eigener RARP-Server vorhanden sein.

9.2 DHCP

Definition: DHCP

Das *Dynamic Host Configuration Protocol (DHCP)* ermöglicht es, angeschlossene Clients ohne manuelle Konfiguration der Netzschnittstelle in ein bestehendes Netz einzubinden.

Nötige Informationen wie IP-Adresse, Netzmaske, Gateway, Name Server (DNS) und ggf. weitere Einstellungen werden automatisch vergeben, sofern das Betriebssystem des jeweiligen Clients dies unterstützt.

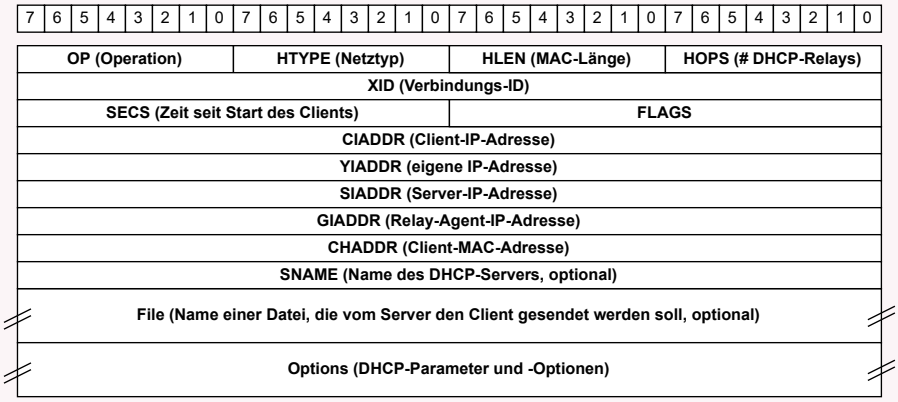
Definition: DHCP-Betriebsmodi

Es gibt drei verschiedene Betriebsmodi eines DHCP-Servers:

- *Statische Zuordnung* bzw. *statisches DHCP*:
  - Am DHCP-Server werden IPs bestimmten MAC-Adressen fest zugeordnet.
  - Adressen werden der MAC-Adresse auf unbestimmte Zeit zugeteilt.
  - Nachteil kann darin liegen, dass sich keine zusätzlichen Clients in das Netz einbinden können.[a]
- *Automatische Zuordnung*:
  - Am DHCP-Server wird ein Bereich von IP-Adressen (range) definiert.
  - IP-Adressen werden automatisch an die MAC-Adressen von neuen DHCP-Clients zugewiesen, was in einer Tabelle festgehalten wird.
  - Im Unterschied zur dynamischen Zuordnung sind automatische Zuordnungen permanent und werden nicht entfernt.
  - Vorteil ist, dass Hosts immer dieselbe IP-Adresse erhalten und eine zugewiesene IP-Adresse keinem anderen Host zugewiesen wird.
  - Nachteil ist, dass neue Clients keine IP-Adresse erhalten, wenn der gesamte Adressbereich vergeben ist, auch wenn IP-Adressen nicht mehr aktiv genutzt werden.
- *Dynamische Zuordnung*:
  - Gleicht der automatischen Zuordnung, allerdings hat der DHCP-Server hier in seiner Konfigurationsdatei eine Angabe, wie lange eine bestimmte IP-Adresse an einen Client verliehen werden darf, bevor der Client sich erneut beim Server melden und eine Verlängerung beantragen muss.
  - Meldet er sich nicht, wird die Adresse frei und kann an einen anderen (oder auch denselben) Rechner neu vergeben werden. Diese vom Administrator bestimmte Zeit heißt *Lease-Time*.

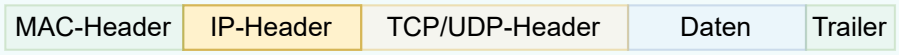
[a] Das kann unter Sicherheitsaspekten erwünscht sein.

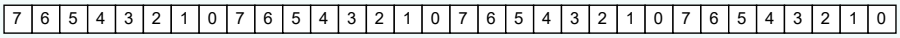
Bonus: DHCP-Paket



9.3 Fragmentierung

Definition: IPv4-Header





Version	HLen	T. Service	Total Packet Length
Fragment ID		Flags	Fragment Offset
TTL		Protocol	Header Checksum
Source Address			
Destination Address			

Definition: IP-Fragmentierung

Die *IP-Fragmentierung* bezeichnet die Aufteilung eines IP-Datenpakets auf mehrere Datenblöcke, falls die Gesamtlänge des Datenpakets größer als die *Maximum Transmission Unit (MTU)* der Netzwerkschnittstelle ist.

Wird ein IP-Datagramm fragmentiert, so wird es erst beim Empfänger wieder zusammengesetzt.[a] Sollte es nötig sein, kann auch ein bereits fragmentiertes Paket weiter fragmentiert werden (etwa bei einem Wechsel der Übertragungstechnik).

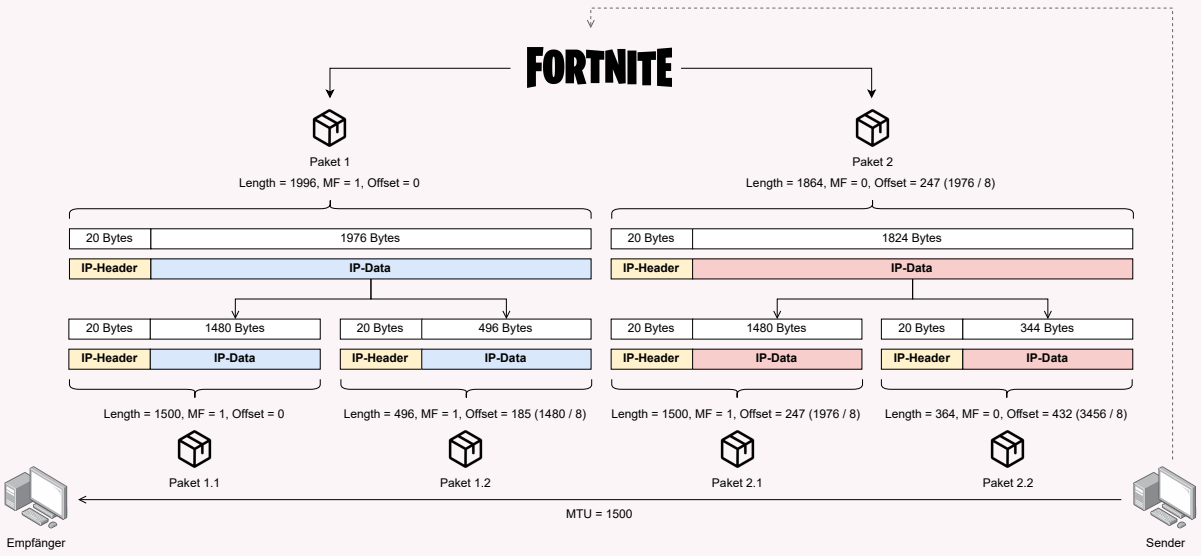
Jedes IP-Datagramm, das fragmentiert wurde, erhält einen neuen Header auf Basis des originalen Headers und spezieller aktualisierter Felder.

In den neuen IP-Headern der Fragmente gibt der *Offset* die Position der in diesem Paket versendeten Daten in Relation zum Originalpaket an (wird in 8-Byte-Blöcken angegeben).

Wichtig: Dadurch müssen alle Fragmente (bis auf das letzte) eine durch 8 teilbare Größe haben.

[a] Ausnahme: ggf. zwischengeschaltete Firewalls, die speziell angewiesen wurden, ein sogenanntes reassembly durchzuführen, bevor die Daten weitergeleitet werden.

Beispiel: IP-Fragmentierung



Bonus: Zusammensetzen der Fragmente

Der Empfänger hat nun die Aufgabe, das Original aus den in den Paketheadern vorhandenen Informationen wieder zusammenzusetzen, indem er alle Fragmente mit gleichem IP-Header (mit Ausnahme der für jedes Fragment separaten Information) nimmt und sie anhand ihres Offsets in die richtige Reihenfolge bringt.

Da jedes einzelne Fragment ein eigenständiges Paket darstellt, kann es auch vorkommen, dass diese Einzelteile nicht geordnet ankommen.

Es ist auch möglich, dass einzelne Fragmente verlorengehen oder defekt sind. Es ist dann Sache des Empfängers, das Paket zu verwerfen und die Daten erneut anzufordern, wodurch eine höhere Netzwerklast entstehen kann.

Definition: Path MTU Discovery

Path MTU Discovery ist ein Verfahren zum dynamischen Erkennen der Maximum Transmission Unit (MTU) und damit der maximalen Paketgröße für einen bestimmten Pfad im Netzwerk.

Im Allgemeinen kann mit dieser Information Overhead vermindert und Fragmentierung von Datenpaketen verhindert werden.

Um in IPv4-Netzen die maximale Größe zu bestimmen, die ein Datenpaket haben sollte, muss die Stelle des Pfades gefunden werden, die die kleinsten Datenpakete zulässt.

Vorgehen[a]:

1. Es wird ein IPv4-Paket versendet, bei dem das `DF`-Bit (Don't Fragment) gesetzt ist und das die Größe der lokal eingestellten Maximum Transmission Unit hat.
2. Kommt das Paket an eine Stelle im Netz, an dem nur eine kleinere MTU verarbeitet werden kann, wird ein ICMP-Error Typ 3 Code 4 (Destination Unreachable Fragmentation Needed, DF Set) zurückgeschickt, der auch die eigene MTU enthält.
3. Der lokale Rechner erhält dieses ICMP-Paket und kann die Größe seiner Nachrichten nun an die zurückgeschickte MTU anpassen.
4. Wiederhole, bis das Paket den gesamten Pfad ohne Fragmentierung durchlaufen kann.

[a] Nur gültig in IPv4, da in IPv6 keine Fragmentierung von weitergeleiteten Paketen auf Routern stattfindet.

Bonus: Jumbo Frames

Der Begriff *Jumbo Frames* bezeichnet nicht standardisierte Frames (d. h. Zusammenfassungen der zu übertragenden Daten), die größer sind als die in der Norm IEEE 802.3 festgelegte Standardgröße von 1518 Bytes.

Für einige Anwendungen können Jumbo Frames sinnvoll sein, da durch sie der Protokoll-Overhead reduziert und die Effizienz verbessert werden können.

Außerdem kann bei den beteiligten Knoten der Verarbeitungsoverhead möglicherweise gesenkt werden, da weniger Frames verarbeitet werden müssen.

Solche Frames sind nicht Standard, und so muss sichergestellt werden, dass alle Netzwerkelemente wie Switches, Router etc. in einem Netz mit diesen Jumbo Frames umgehen können, und getestet werden, ob es einen Geschwindigkeitsvorteil gibt.

9.4 ICMP

Definition: ICMP

Das *Internet Control Message Protocol (ICMP)* dient dem Austausch von Informations- und Fehlermeldungen über das Internet-Protokoll in der Version 4 (IPv4). Für IPv6 existiert ein ähnliches Protokoll mit dem Namen ICMPv6.

ICMP ist Bestandteil von IPv4, wird aber wie ein eigenständiges Protokoll behandelt. Es wird von jedem Router und jedem Rechner erwartet, dass sie ICMP verstehen.

Die meisten ICMP-Pakete enthalten Diagnose-Informationen: Sie werden vom Router zur Quelle zurückgeschickt, wenn der Router Pakete verwirft, etwa weil beispielsweise das Ziel nicht erreichbar ist oder die TTL abgelaufen ist.

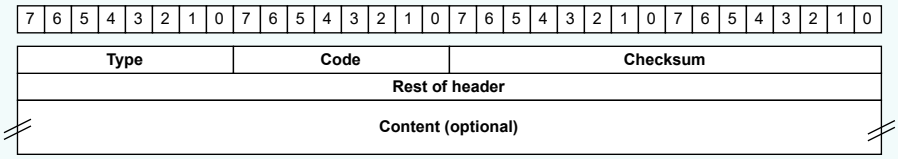
Es gelten folgende Grundsätze:

- ICMP benutzt IP als Kommunikationsbasis, indem es sich selbst als Protokoll einer höheren Schicht interpretiert, d. h. ICMP-Nachrichten werden in IP-Paketen gekapselt.
- ICMP erkennt einige Fehlerzustände, macht aber IP zu keinem zuverlässigen Protokoll.
- ICMP analysiert Fehler in jedem IP-Paket, mit Ausnahme solcher, die eine ICMP-Nachricht tragen.
- ICMP-Nachrichten werden nicht als Antwort auf Pakete an Zieladressen versendet, bei denen es sich um Multicast- oder Broadcast-Adressen handelt.
- ICMP-Nachrichten antworten nur einer eindeutigen Quell-IP-Adresse.

Beispiele für Nachrichtentypen:

- *Destination Unrechable*: Ziel nicht erreichbar
- *Time Exceeded*: Time-to-Live-Feld eines Pakets ist abgelaufen
- *Echo Reply*: Echo-Reply wird angefordert (`ping`)

Definition: ICMP-Header



9.5 IPv6

Definition: IPv6

IPv6-Adressen sind 128 Bit lang (IPv4: 32 Bit) und werden in hexadezimaler Notation mit Doppelpunkten geschrieben, z. B. `3ffe:400:20::a00:2bff:fea3:adcb`.

Die ersten 64 Bit bilden das *Präfix*, die letzten 64 Bit bilden bis auf Sonderfälle einen für die Netzwerkschnittstelle eindeutigen *Interface-Identifier*.

Eine Netzwerkschnittstelle kann unter mehreren IP-Adressen erreichbar sein; in der Regel ist sie dies mittels ihrer *link-lokalen Adresse* und einer *global eindeutigen Adresse*.

Es werden verschiedene Adressklassen durch das Präfix festgelegt:

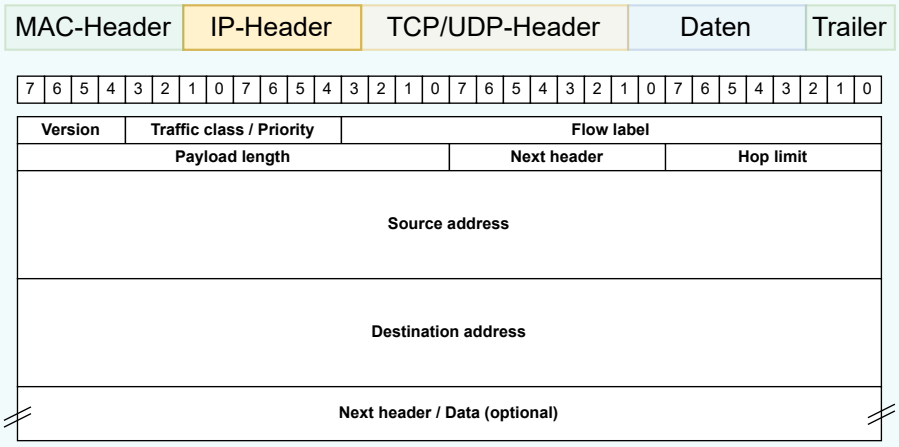
Typ	Präfix (binär)	Präfix (hexadezimal)
Nicht spezifiziert	`0...0`	`::/128`
Loopback	`0...01`	`::1/128`
Link-Local-Unicast	`1111 1110 10`	`fe80::/64`
Unique-Local-Unicast	`1111 110`	`fc00::/7`
Multicast	`1111 1111`	`ff00::/8`
Global-Unicast	alles andere, z. B. auch	
IPv4 mapped	`0...01...1`	`0:0:0:0:0:ffff::/96`
Von IANA vergeben	`001...`	`2000::/3`

Definition: Vorteile bei der Verwendung von IPv6

IPv6 bietet folgende Verbesserungen gegenüber IPv4:

- Effizienteres Routing ohne Fragmentierung von Paketen
- Eingebaute Quality of Service (QoS), die verzögerungsempfindliche Pakete unterscheidet
- Eliminierung von NAT zur Erweiterung des Adressraums von 32 auf 128 Bit
- Eingebaute Sicherheit auf Netzwerkschicht (IPsec)
- Zustandslose Adressen-Autokonfiguration für einfachere Netzwerkverwaltung
- Verbesserte Header-Struktur mit weniger Verarbeitungsaufwand

Definition: IPv6-Header

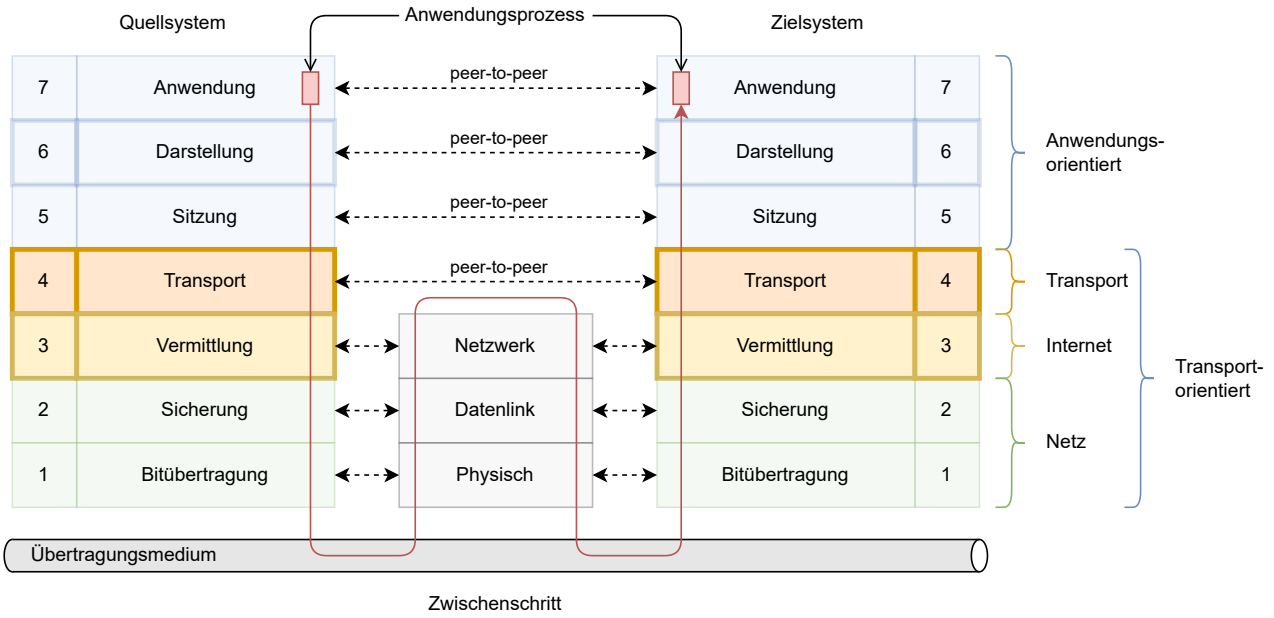


Bonus: IPv6-Erweiterungs-Header

Optionale Angaben folgen in *IPv6-Erweiterungs-Headern*:

Name	Typ	Größe	Beschreibung
Hop-By-Hop Options	0	variabel	Enthält Optionen, die von allen IPv6-Geräten, die das Paket durchläuft, beachtet werden müssen. Wird z. B. für Jumbograms benutzt.
Routing	43	variabel	Durch diesen Header kann der Weg des Paketes durch das Netzwerk beeinflusst werden, er wird unter anderem für Mobile IPv6 verwendet.
Fragment	44	64 Bit	In diesem Header können die Parameter einer Fragmentierung festgelegt werden.
Authentication Header (AH)	51	variabel	Enthält Daten, welche die Vertraulichkeit des Paketes sicherstellen können.
Encapsulating Security Payload (ESP)	50	variabel	Enthält Daten zur Verschlüsselung des Paketes.
Destination Options	60	variabel	Enthält Optionen, die nur vom Zielrechner des Paketes beachtet werden müssen.
Mobility	135	variabel	Enthält Daten für Mobile IPv6.
No Next Header	59	leer	Dieser Typ ist nur ein Platzhalter, um das Ende eines Header-Stapels anzuzeigen.

10 Send and Wait



Bonus: Probleme unsicherer Datenkanäle

Bei unsicheren Übertragungskanälen (wie z. B. IP) können folgende Fehler auftreten:

- Pakete können verloren gehen
- Die Reihenfolge kann vertauscht werden
- Pakete können Fehlerhaft empfangen werden

Bonus: Zuverlässige Transportprotokolle

Wir wollen einen kontinuierlichen Datenstrom in Nachrichteneinheiten (Segmente) unterteilen, die eindeutig identifizierbar sind und im Anschluss sicher versenden. Wir können davon ausgehen, dass sich die beiden Geräte bereits kennen und eine Verbindung aufgebaut haben.

Erforderliche Mechanismen:

- Fehlererkennung und ggf. -behebung
  - Der Empfänger muss Bitfehler in der Übertragungskanälen erkennen
  - Der Empfänger muss das Fehlen eines Segmentes erkennen und darauf reagieren (Obacht: Pakete können sich ebenfalls nur verspäten).
- Empfänger-Feedback
  - Quittierungsnachrichten zwischen Empfänger und Sender
  - *positive Acknowledgment* (`ACK`), welches den korrekten Erhalt bestätigt
  - Optional: *negative Acknowledgment* (`NAK`), welches den Sender über Lücken oder Fehler informiert
- Neuübertragung:
  - Sender überträgt Fehlerhafte Segmente erneut
  - Timer- bzw. `NAK`-gesteuert

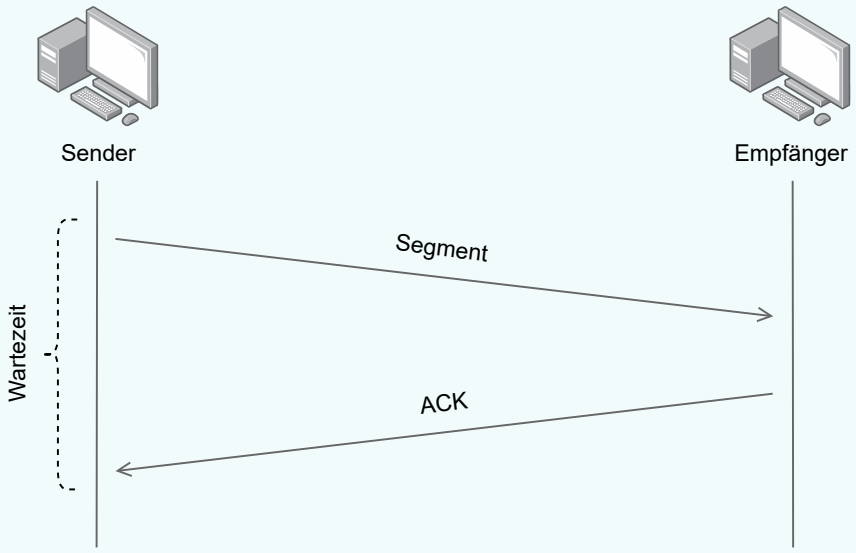
Definition: ARQ-Protokoll

Das *Automatic-Repeat-Request-Protokoll (ARQ)*, auch *Send and Wait-Protokoll*, besteht daraus, dass Empfänger Bestätigungen (`ACK`s) bei korrektem Empfang versenden.

Fehlerhafte Übertragungen werden durch Ausbleiben der Bestätigung erkannt und durch wiederholtes Versenden behoben.

Definition: Einfaches ARQ-Protokoll

Der Sender geht segmentweise vor. Beim Senden startet er einen Timer zur Fehleridentifikation. Bei Erhalt des `ACK` wird das nächste Paket gesendet. Sobald nach Ablaufen des Timers noch kein `ACK` empfangen wurde, wird die Übertragung wiederholt.



Der Datenstrom ist begrenzt auf ein Paket pro Zyklus.

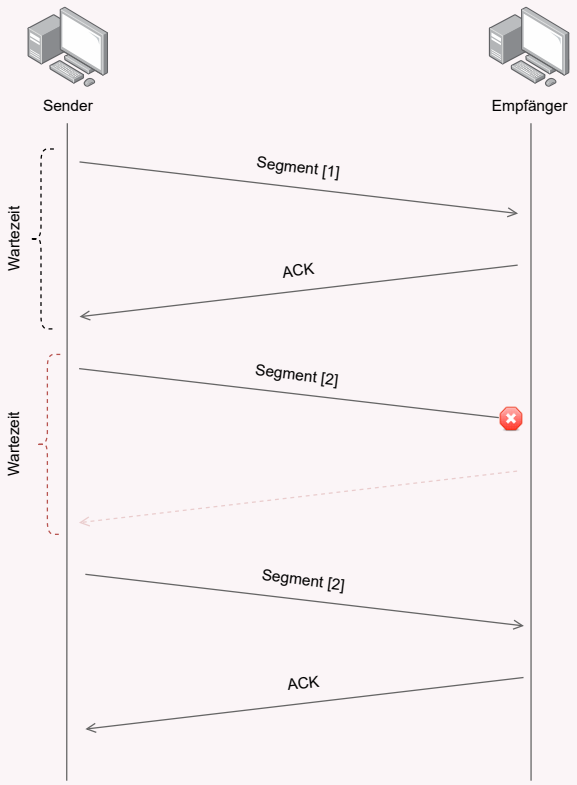
Sei die Länge eines Segments (L) in bits und die Übertragungsrate eines Kanals in $^{\text{bits}}/_{s}$ gegeben, so wird die genutzte Übertragungszeit mit $^{L}/_{R}$ errechnet.

Die Wartezeit bzw. der Timeout (RTT) ist durch die genutzte Übertragungsart in Sekunden gegeben.

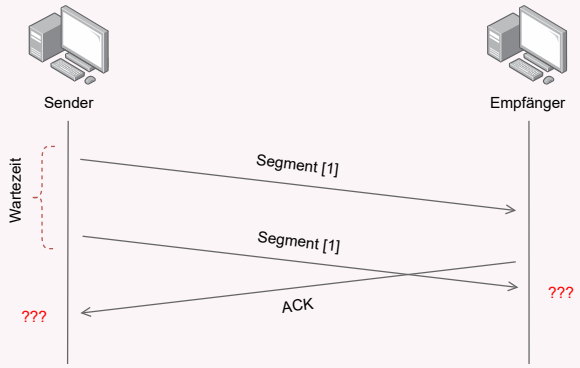
Der Nutzungsgrad ρ wird wie folgt berechnet:

$$\rho := \frac{\text{genutze Übertragungszeit}}{\text{genutze Übertragungszeit} + \text{Wartezeit}} = \frac{\frac{L}{R}}{\frac{L}{R} + RTT}$$

Beispiel: Einfaches ARQ-Protokoll

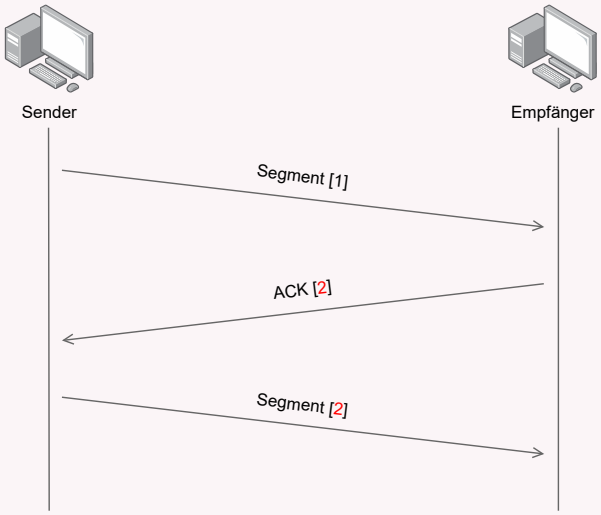


Bonus: Verfahren bei Fehler- bzw. Sonderfällen



Wenn wir länger als erwartet brauchen um das Paket zu versenden, wird immer vor Erhalt der Bestätigung neu versendet. Der Empfänger weiß nicht, welches Segment empfangen wurde und der Sender weiß nicht, für welches Paket er eine Bestätigung erhalten hat.

Um das Problem zu lösen versehen wir jedes Paket mit einer eindeutigen ID. Diese entspricht entweder dem 1. Byte der Nachricht oder wird zyklisch durchnummeriert:



Definition: Pipelining

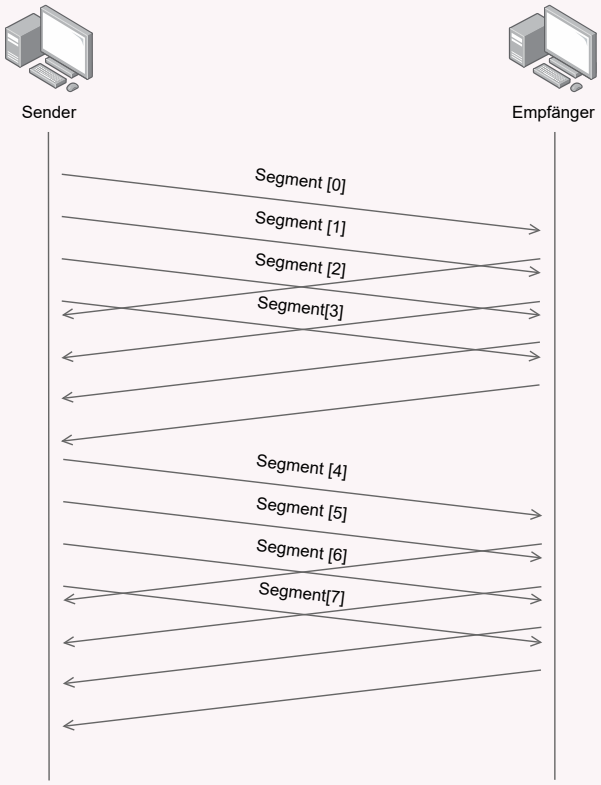
Um das Netz besser auszulasten nutzt man *Pipelining*:

- Senden weiterer Segmente ohne vorheriges `ACK`
- Die Fensterbreite beim Sender richtet sich nach der Anzahl der Segmente, die ohne Bestätigung gesendet werden.
- Die Fensterbreite beim Empfänger richtet sich nach der Anzahl der Segmente, die auch bei Lücken oder Fehlern zwischengespeichert werden.

Sei n die Fensterbreite des Senders. Dann wird der Nutzungsgrad ρ wie folgt berechnet:

$$\rho := n \cdot \frac{\frac{L}{R}}{\frac{L}{R} + RTT}$$

Beispiel: Pipelining



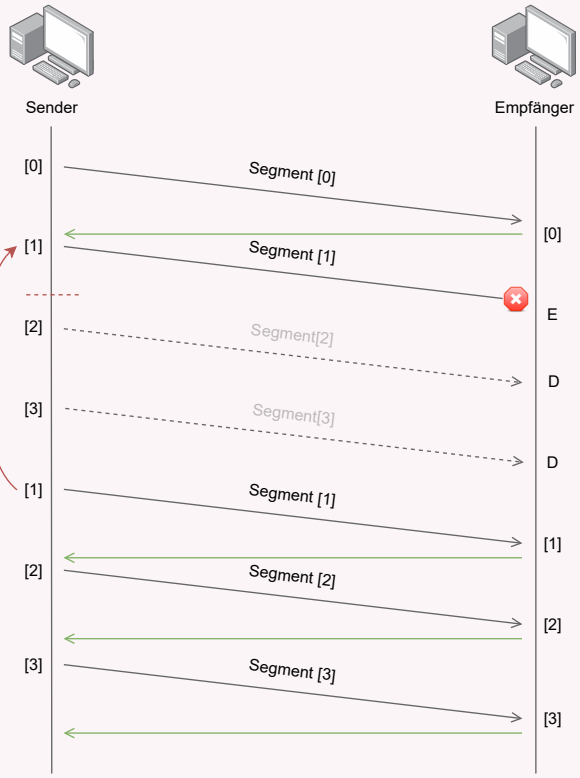
Definition: Go-Back-N

Go-Back-N ist ein Verfahren um nach Fehlern das Senden sinnvoll fortzusetzen.

Sobald der Sender kein `ACK` erhält, läuft er in einen Timeout (Fensterbreite) und springt danach hinter das Paket, welches als letztes erfolgreich versendet wurde.

Alle direkt nach dem Fehler versendeten Segmente werden verworfen.

Beispiel: Go-Back-N



Der Client sendet kein `ACK` auf Fehler (`E`) und verwirft Frames, die unerwartet empfangen werden (`D`).

11 Sliding Window

Definition: Sliding Window

Der Begriff *Sliding Window* bezeichnet bei der Datenflusskontrolle in Rechnernetzen ein Fenster, das einem Sender die Übertragung einer bestimmten Menge von Daten ermöglicht, bevor eine Bestätigung zurückerwartet wird.[a]

Das Sliding Window verfolgt das Ziel, die Kapazitäten der Leitung und des Empfängers optimal auszulasten, das heißt so viele Datenpakete (Datenframes) wie möglich zu senden.

Dabei stellt das Verzögerungs-Bandbreite-Produkt die maximale in der Übertragung befindliche Datenmenge dar, die gesendet werden kann, ohne auf die erste Bestätigung (`ACK`) zu warten.

[a] Netzwerkprotokolle, die auf Sliding Windows basieren, werden *Sliding-Window-Protokolle* genannt.

Definition: Sliding Window (Sender)

Beim Schiebefensterverfahren führt der Sender permanent eine Liste von aufeinanderfolgenden Sequenznummern, die der Anzahl der Frames, die er senden darf, entspricht.

Sobald ein Datenpaket dem Empfänger erfolgreich zugestellt wird, sendet dieser dafür eine Bestätigung (`ACK`) zurück, die den Sender dazu veranlasst, ein weiteres Frame zu übertragen.

Falls der Sender innerhalb des Timeouts jedoch kein `ACK` erhält, versucht er das Frame erneut zu übertragen. Unter der Voraussetzung, dass das Verzögerungs-Bandbreiten-Produkt bereits erreicht ist, kann dann aber kein neues Frame übertragen werden, d. h., es kommt zu einem Stau in der Pipe.

Das Sendefenster verschiebt sich mit jeder eingehenden Bestätigung, indem das bestätigte Frame aus dem Fenster herausfällt und ein neu zu sendendes Frame in das Fenster aufgenommen wird. Dadurch enthält das Fenster immer nur unbestätigte Frames.

Für den Fall, dass Frames während der Übertragung verloren gehen, muss der Sender alle Datenpakete in seinem Speicher halten, um sie erneut übertragen zu können.

Definition: Sliding Window (Empfänger)

Analog zum Fenster des Senders verwaltet auch der Empfänger ein Schiebefenster.

Beide Fenster müssen aber nicht unbedingt die gleiche Größe haben, da diese im Laufe der Zeit durch das Senden und Empfangen von Frames variieren kann.

Die Größe des Sendefensters bestimmt sich durch das vom Empfänger angegebene Maximum sowie durch die Netzbelastung.

Jede Bestätigung für ein erfolgreich übertragenes Frame enthält einen Wert, der angibt, für welche Menge an weiteren Datenpaketen der Empfänger noch Kapazität frei hat.

Definition: Verzögerungs-Bandbreiten-Produkt

Das *Verzögerungs-Bandbreiten-Produkt* ist eine Eigenschaft einer Datenverbindung. Es berechnet sich als das Produkt aus der Verzögerung (Einheit: Sekunde) und der Datenübertragungsrate (Einheit: Bit pro Sekunde).

Das Ergebnis, eine in Bit oder Byte angegebene Datenmenge entspricht der Anzahl der Daten, die sich auf dem Weg befinden, wenn ein oder mehrere Sender den Übertragungskanal füllen.

Ein hoher Durchsatz kann nur dann erzielt werden, wenn der Sender hinreichend große Datenmengen versenden kann, bevor dieser innehalten und auf eine Empfangsbestätigung durch den Empfänger warten muss. Ist die Menge an gesendeten Daten verglichen mit dem Verzögerungs-Bandbreiten-Produkt gering, dann kommt das Protokoll nicht an die maximale Effizienz heran.

Der Nutzungsgrad ρ ist wie folgt definiert:

$$\rho := n \cdot \frac{\frac{L}{R}}{\frac{L}{R} + RTT}$$

mit

- n: Anzahl der Segmente.
- L: Länge eines Segments in Bits.
- R: Übertragungsrate des Kanals in Bits pro Sekunde.
- L/R: Zeit zur Übertragung eines Segments in Sekunden.
- RTT: *Rount Trip Time*
- $B := R\rho$: erzielbare Bandbreite
- $W := n \cdot L$: Fensterbreite in Bits

Die erzielbare Bandbreite $R\rho = B$ ist dann:

$$B := R\rho = n \cdot \frac{L}{\frac{L}{R} + RTT}$$

Es gilt:

$$\frac{L}{R} \ll \text{RTT}, \quad B \leq \frac{W}{\text{RTT}}$$

Beispiel: Sliding Window

Beispiel 1:

Die Anwendung möchte eine Rate von 10 Mbps erreichen. Wir kennen die Round-Trip-Zeit, die bei 8ms liegt. Die Segmente seien 500 Byte groß.

Wie groß muss W sein?

Es gilt:

$$W \geq 10 \text{ Mbps} \cdot 0.008 \text{ s} = 80\,000 \text{ Bit} = 10\,000 \text{ Byte}$$

$$\Longrightarrow \text{ 20 Segmente mit 500 Byte}$$

Beispiel 2:

Die Anwendung möchte eine Rate von 1 Gbps erreichen. Wir kennen die Round-Trip-Zeit, die bei 240ms liegt. Die Segmente seien 1 KB groß.

Wie groß muss W sein?

Es gilt:

$$W \geq 10 \text{ Gbps} \cdot 0.24 \text{ s} = 240\,000\,000 \text{ Bit} = 30\,000\,000 \text{ Byte}$$

$$\Longrightarrow \text{ 30 000 Segmente mit 1 KB}$$

Definition: Ereignisse beim Empfänger

- Empfang eines Out-of-Order-Segments
  - Eigentlich wurde ein anderes Segment erwartet.
  - Je nach Implementierung einfach ignoriert bzw. verworfen.
  - Sender überträgt das Segment nach Ablauf des Timeouts erneut.
- Empfang eines fehlerbehafteten Segments
  - Keine Reaktion, Sender überträgt das Segment nach Ablauf des Timeouts erneut.
  - Falls unterstützt: Verschicken einer „negativen“ Bestätigung `NAK`.

Definition: Ereignisse beim Sender

- Timeout
- Empfang einer nicht erwarteten Bestätigung (*out of order* `ACK`)
- Empfang einer negativen Bestätigung `NAK`

Die Reaktion ist die Identifikation und das erneute Verschicken des fehlerhaften Segments:

- *Go-Back-N*: Ab dem Fehlerfall alles neu versenden.
- *Selective Repeat* bzw. *Selective Reject*: Nur das fehlerhafte Segment neu versenden.

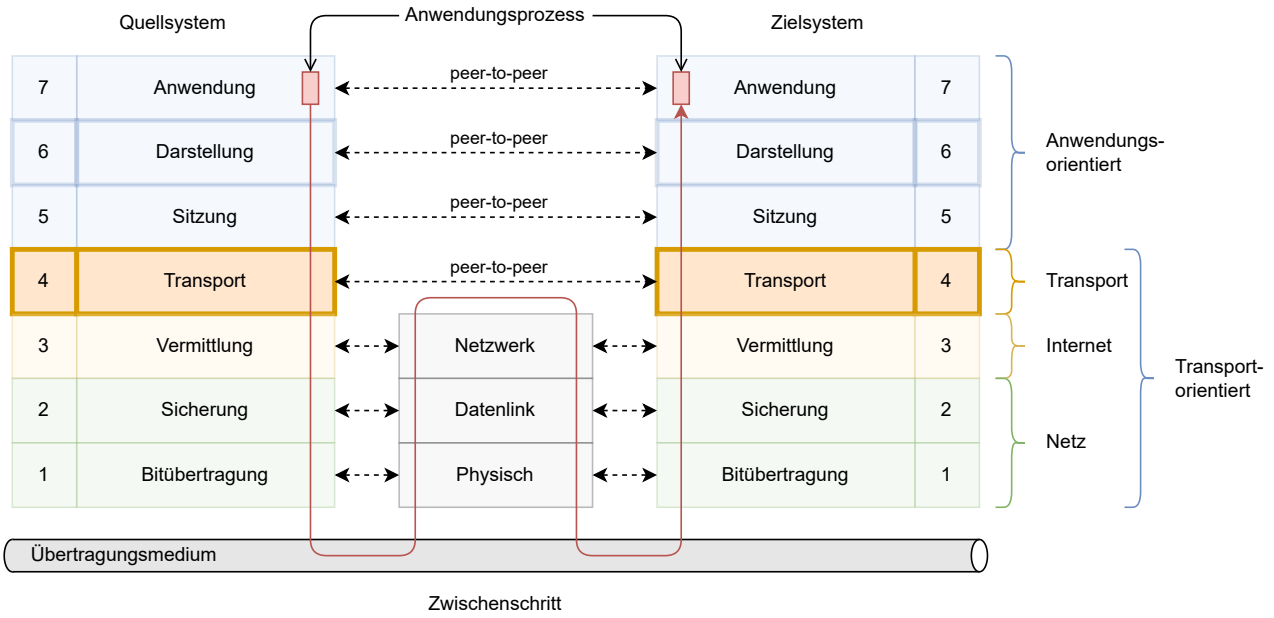
Bonus: Unterschied zum Stop-and-Wait-Algorithmus

Beim Stop-and-Wait-Algorithmus, der ebenso wie das Schiebefensterverfahren ein ARQ-Protokoll ist, wartet der Sender nach der Übertragung eines Frames auf eine Bestätigung, bevor er das nächste Frame überträgt. Kommt innerhalb der Wartezeit keine Bestätigung, überträgt der Sender das Frame erneut.

Im Gegensatz zu diesem Algorithmus, der dem Sender nur jeweils ein ausstehendes Frame auf der Verbindungsleitung gestattet und dadurch ineffizient ist, kann der Sliding-Window-Algorithmus mehrere Frames gleichzeitig übertragen.

Der Stop-and-Wait-Algorithmus ist ein Spezialfall des Sliding-Window-Algorithmus. Beide funktionieren identisch, wenn die Größe des Sendefensters beim Schiebefenster-Algorithmus auf 1 Frame eingestellt ist.

12 TCP



Definition: TCP

Das *Transmission Control Protocol (TCP)* ist ein Netzwerkprotokoll, das definiert, auf welche Art und Weise Daten zwischen Netzwerkkomponenten ausgetauscht werden sollen.

Das Protokoll ist ein *zuverlässiges, verbindungsorientiertes, paketvermitteltes* Transportprotokoll in Computernetzwerken.

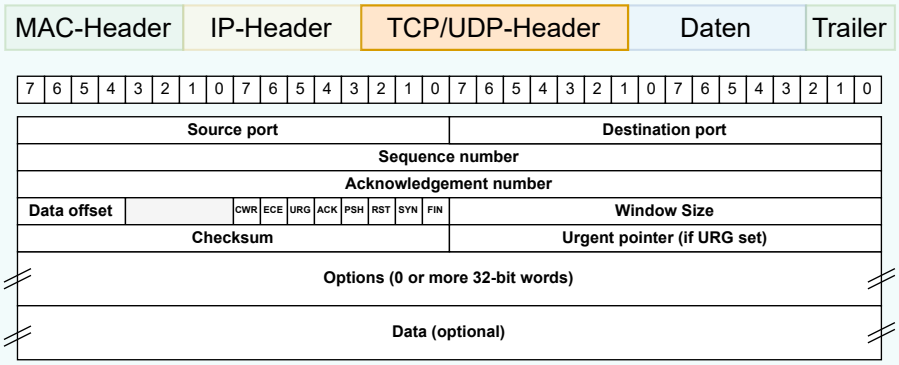
Im Unterschied zum verbindungslosen UDP stellt TCP eine Verbindung zwischen zwei Endpunkten einer Netzverbindung (Sockets) her. Auf dieser Verbindung können in beide Richtungen Daten übertragen werden.

TCP setzt in den meisten Fällen auf das IP (Internet-Protokoll) auf, weshalb häufig (und oft nicht ganz korrekt) auch vom TCP/IP-Protokoll die Rede ist. In Protokollstapeln wie dem OSI-Modell sind TCP und IP nicht auf derselben Schicht angesiedelt.

TCP ist eine Implementierung der Transportschicht.

Aufgrund seiner vielen positiven Eigenschaften (Datenverluste werden erkannt und automatisch behoben, Datenübertragung ist in beiden Richtungen möglich, Netzüberlastung wird verhindert usw.) ist TCP ein sehr weit verbreitetes Protokoll zur Datenübertragung. Beispielsweise wird TCP als fast ausschließliches Transportmedium für das WWW, E-Mail und viele andere populäre Netzdienste verwendet.

Definition: TCP-Header



Definition: Bestätigungen in TCP

Um empfangene Pakete zu bestätigen, wird eine *Huckepack-Technik* verwendet; das bedeutet, dass Bestätigungen im Datenpaket der Rückrichtung erfolgen.

Dabei bestätigt eine Bestätigungsnachricht (`ACK`) alle bis dahin erhaltenen Segmente.[a]

Auch ohne Datenfluss in die Rückrichtung werden Bestätigungen verschickt, wenn auch oft verzögert.[b] Diese verzögerten Bestätigungen heißen *Delayed Acknowledgments*.

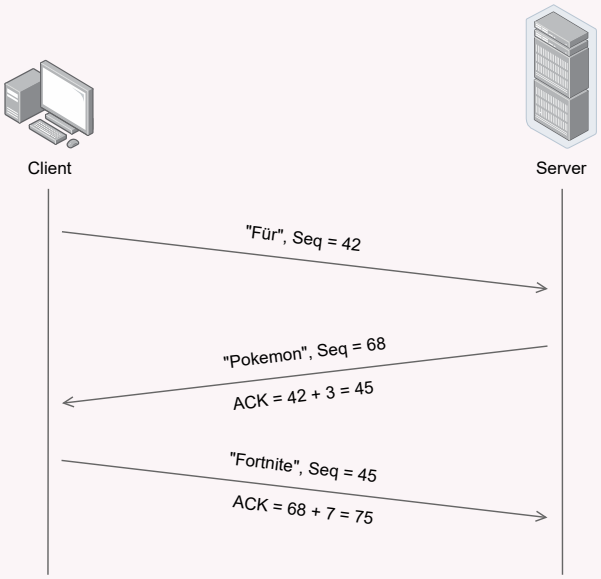
Es gilt:

Ereignis beim Empfänger	Aktion beim Empfänger
Ankunft eines Segments mit der erwarteten Sequenznummer. Alle Daten bis dahin sind schon bestätigt worden.	*Delayed ACK*. Warte bis zu 500ms auf das nächste Segment. Wenn dieses nicht empfangen wird, verschicke die Bestätigung.
Ankunft eines Segments mit der erwarteten Sequenznummer. Allerdings wurde noch keine Bestätigung des voherigen Segments verschickt.	Sofortiges Verschicken einer kumulativen Bestätigung, die beide Segmente bestätigt.
Ankunft eines Segments hinter der erwarteten Sequenznummer. Es wird eine Lücke entdeckt.	Sofortiges Verschicken eines *Duplicate ACK* (`Dup ACK`). Diese enthält erneut die erwartete Segmentnummer.
Ankunft eines Segments, das eine Lücke teilweise oder vollständig füllt.	Sofortiges Verschicken einer Bestätigung.

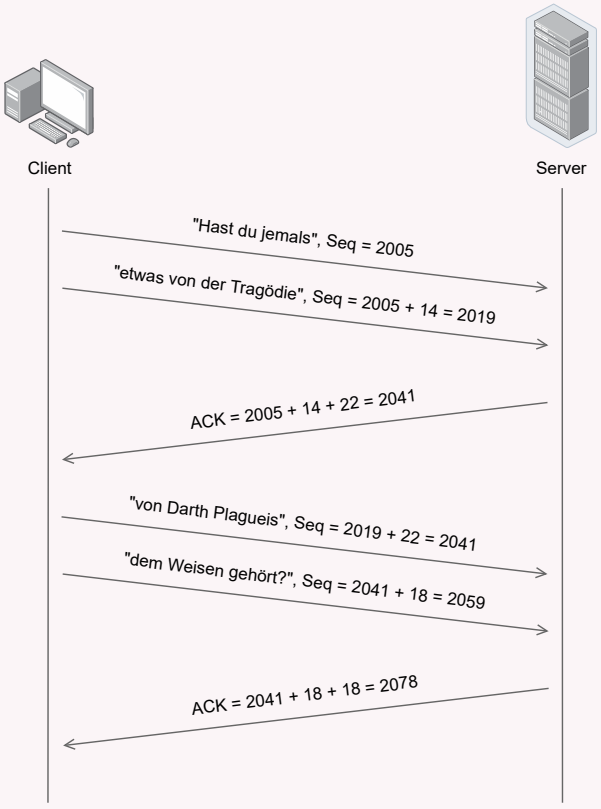
[a] Ein `ACK` enthält also die Sequenznummer des zuletzt erhaltenen Segments.

[b] Nach 500ms muss eine Bestätigung verschickt werden. Nach zwei vollständigen Segmenten sollte ebenfalls eine Bestätigung verschickt werden.

Beispiel: Bestätigungen in TCP



Beispiel: Delayed Acknowledgments



Bonus: Go-Back-N

Go-Back-N stellt ein Verfahren dar, das im Gegensatz zu Stop-and-Wait einen deutlich größeren Durchsatz ermöglicht.

Der Sender kann dabei mehrere Dateneinheiten senden, ohne auf eine Quittung warten zu müssen. Wie viele das sind, hängt von der sogenannten *Fenstergröße* ab.

Beträgt diese n, kann der Sender noch $n - 1$ weitere Dateneinheiten absenden, bevor die Bestätigung für die erste Einheit durch den Empfänger erfolgt sein muss.

Es können vom Empfänger auch mehrere Dateneinheiten auf einmal (kumulativ) bestätigt werden, so zeigt eine Quittung für $n + i$ an, dass alle Einheiten von n bis $n + i$ korrekt empfangen wurden.

Kommt es beim Warten auf die Bestätigungen zu einem Timeout, so übermittelt der Sender alle Dateneinheiten in dem Fenster neu. Er geht also zurück zur letzten unbestätigten Sequenznummer N.

Da es der Fall sein kann, dass lediglich eine Dateneinheit nicht ordnungsgemäß übertragen wurde und dennoch auch alle danach gesendeten erneut übertragen werden, wird an dieser Stelle Übertragungskapazität verschwendet.

Bonus: Selective-Repeat

Bei *Selective Repeat* wird ein fehlerhafter Rahmen verworfen, aber die danach erhaltenen Rahmen werden im Empfänger in einem Puffer abgelegt und bestätigt.

Wenn beim Sender die Zeit abgelaufen ist, wird nur der älteste nicht bestätigte Rahmen erneut übertragen. Wenn dieser Rahmen korrekt ankommt, kann der Empfänger in der Folge alle im Puffer gespeicherten Rahmen an die Vermittlungsschicht übergeben.

Die selektive Wiederholung wird oft mit dem Senden einer negativen Bestätigung (`NAK`, *Negative Acknowledgement*) durch den Empfänger kombiniert, wenn dieser einen Fehler wie einen Prüfsummenfehler oder einen Rahmen außerhalb der Reihenfolge entdeckt.

`NAK`s stoßen die erneute Übertragung an, bevor der entsprechende Timer abläuft und verbessern daher die Leistung.

Definition: Round Trip Time

Die *Round Trip Time (RTT)* gibt die Zeit an, die ein Datenpaket (Datagramm) in einem Rechnernetz benötigt, um von der Quelle zum Ziel und zurück zu reisen.

Es handelt sich also um die Summe aus Laufzeit von Punkt A nach Punkt B und der Laufzeit von Punkt B nach Punkt A.

Die RTT wird zum Beispiel vom Transmission Control Protocol (TCP) laufend gemessen, um zu bestimmen, wann Pakete nach Ausbleiben einer Bestätigung erneut gesendet werden sollten.

Bonus: Slow-Start

Zu Beginn einer Datenübertragung dient der *Slow-Start-Algorithmus* zur Bestimmung des Überlastfenster (*congestion window*), um einer möglichen Überlastsituation vorzubeugen.

Der Algorithmus startet mit einem kleinen Fenster von einer MSS (Maximum Segment Size), in dem Datenpakete vom Sender zum Empfänger übertragen werden.

Der Empfänger sendet nun eine Bestätigung (`ACK`) an den Sender zurück. Für jedes empfangene `ACK` wird die Größe des congestion window um eine MSS erhöht. Da für jedes versandte Paket bei erfolgreicher Übertragung ein `ACK` geschickt wird, führt dies innerhalb einer Roundtrip-Zeit zu einer Verdopplung des Congestion Windows. In dieser Phase gibt es also ein exponentielles Wachstum.

Dieses exponentielle Wachstum wird so lange fortgesetzt, bis der sogenannte *Slow-Start Threshold* erreicht wird.[a]

Danach wird das Congestion Window nur noch um eine MSS erhöht, wenn alle Pakete aus dem Fenster erfolgreich übertragen wurden.[b] Diese Phase wird als Congestion Avoidance Phase bezeichnet.

Das Wachstum wird beendet, wenn das vom Empfänger festgelegte Empfangsfenster erreicht worden ist.

Kommt es zu einem Timeout, wird das congestion window wieder auf 1 zurückgesetzt, und der slow-start threshold wird auf die Hälfte der *Flight Size*[c] herabgesetzt. Die Phase des exponentiellen Wachstums wird also verkürzt, so dass das Fenster bei häufigen Paketverlusten nur langsam wächst.

[a] Die Phase des exponentiellen Wachstums wird auch Slow Start Phase genannt.
[b] Es wächst also pro Roundtrip-Zeit nur noch um eine MSS, also nur noch linear.
[c] Flight Size ist die Anzahl an Paketen, die verschickt, aber noch nicht quittiert wurden.

Definition: TCP-Verbindungsmanagement (Verbindungsaufbau)

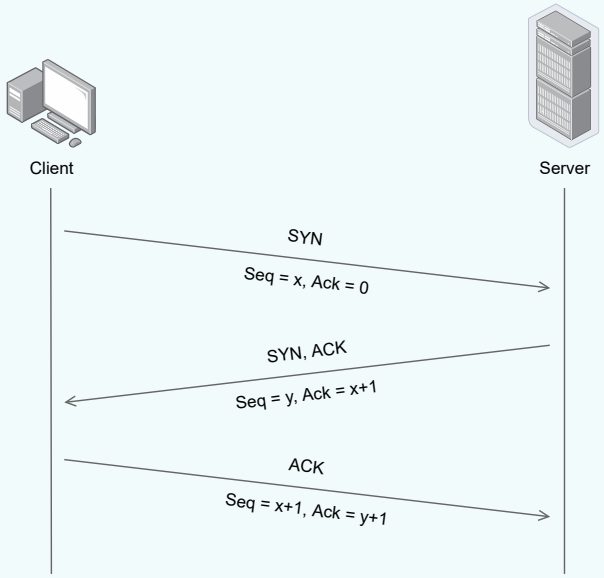
Der Client, der eine Verbindung aufbauen will, sendet dem Server ein `SYN`-Paket mit einer Sequenznummer x. Die Sequenznummern sind dabei für die Sicherstellung einer vollständigen Übertragung in der richtigen Reihenfolge und ohne Duplikate wichtig.

Die Start-Sequenznummer ist eine beliebige Zahl, deren Generierung von der jeweiligen TCP-Implementierung abhängig ist. Sie sollte jedoch möglichst zufällig sein, um Sicherheitsrisiken zu vermeiden.

Der Server empfängt das Paket. Ist der Port geschlossen, antwortet er mit einem `TCP-RST`, um zu signalisieren, dass keine Verbindung aufgebaut werden kann.

Ist der Port geöffnet, bestätigt er den Erhalt des ersten `SYN`-Pakets und stimmt dem Verbindungsaufbau zu, indem er ein `SYN/ACK`-Paket zurückschickt. Zusätzlich sendet er im Gegenzug seine Start-Sequenznummer y, die ebenfalls beliebig und unabhängig von der Start-Sequenznummer des Clients ist.

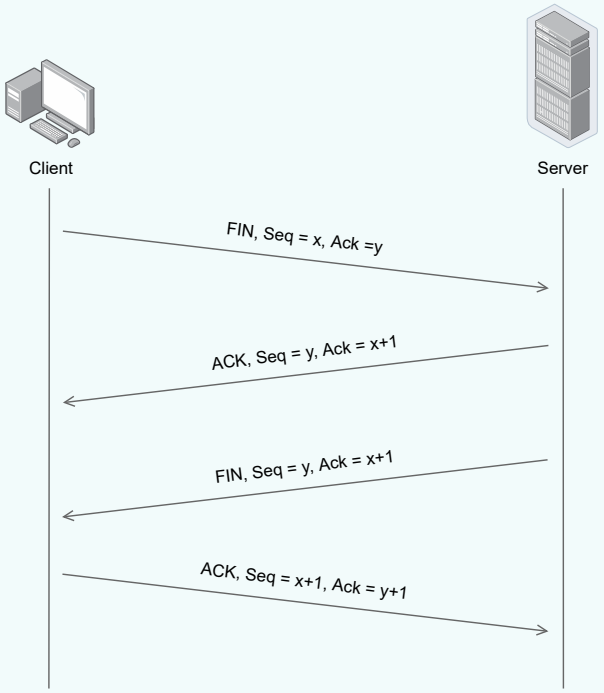
Der Client bestätigt zuletzt den Erhalt des `SYN/ACK`-Pakets durch das Senden eines eigenen `ACK`-Pakets mit der Sequenznummer x+1. Dieser Vorgang wird auch als *Forward Acknowledgement* bezeichnet. Aus Sicherheitsgründen sendet der Client den Wert $y + 1$ im `ACK`-Segment zurück.



Definition: TCP-Verbindungsmanagement (Verbindungsende)

Statt des `SYN`-Bits kommt das `FIN`-Bit zum Einsatz, welches anzeigt, dass keine Daten mehr vom Sender kommen werden. Der Erhalt des Pakets wird wiederum mittels `ACK` bestätigt.

Der Empfänger des `FIN`-Pakets sendet zuletzt seinerseits ein `FIN`-Paket, das ihm ebenfalls bestätigt wird.



Definition: Flusskontrolle

Die *Flusskontrolle (Flow Control)*, ist eine Funktion von Kommunikationsprotokollen zur Anpassung der Übertragungsgeschwindigkeit asynchron arbeitender Endgeräte an die Aufnahmefähigkeit der empfangenden Station.

Die Flusskontrolle regelt die Datenrate zwischen der sendenden und empfangenden Station und sorgt für eine Geschwindigkeitsadaption und dafür, dass die Datenübertragung verlustfrei erfolgt.

Bei Überlast der Übertragungsstrecke oder bei Datenstau veranlasst das empfangende Endgerät das sendende dazu die Datenrate soweit zu reduzieren oder zeitweise auszusetzen, damit die empfangende Station alle Daten der sendenden Station aufnehmen kann.

Eine Überlast tritt dann auf, wenn die Anzahl der Datenpakete, die eine sendende Station dem Netz übergibt, die Aufnahmekapazität der empfangendes Endgerätes überschreitet.

Definition: Flusskontrolle in TCP

TCP implementiert das *Sliding-Window-Protokoll*:

- Bei einer Fenstergröße von n können n Bytes verschickt werden, ohne dass ein `ACK` empfangen werden muss.
- Wenn der Empfang der Daten vom Empfänger bestätigt wurde, so verschiebt sich das Fenster.

Segmente werden durch ihren Byte-Offset im Stream identifiziert (*Sequence Number*), wobei die Startposition beim Verbindungsaufbau zufällig festgelegt wird.

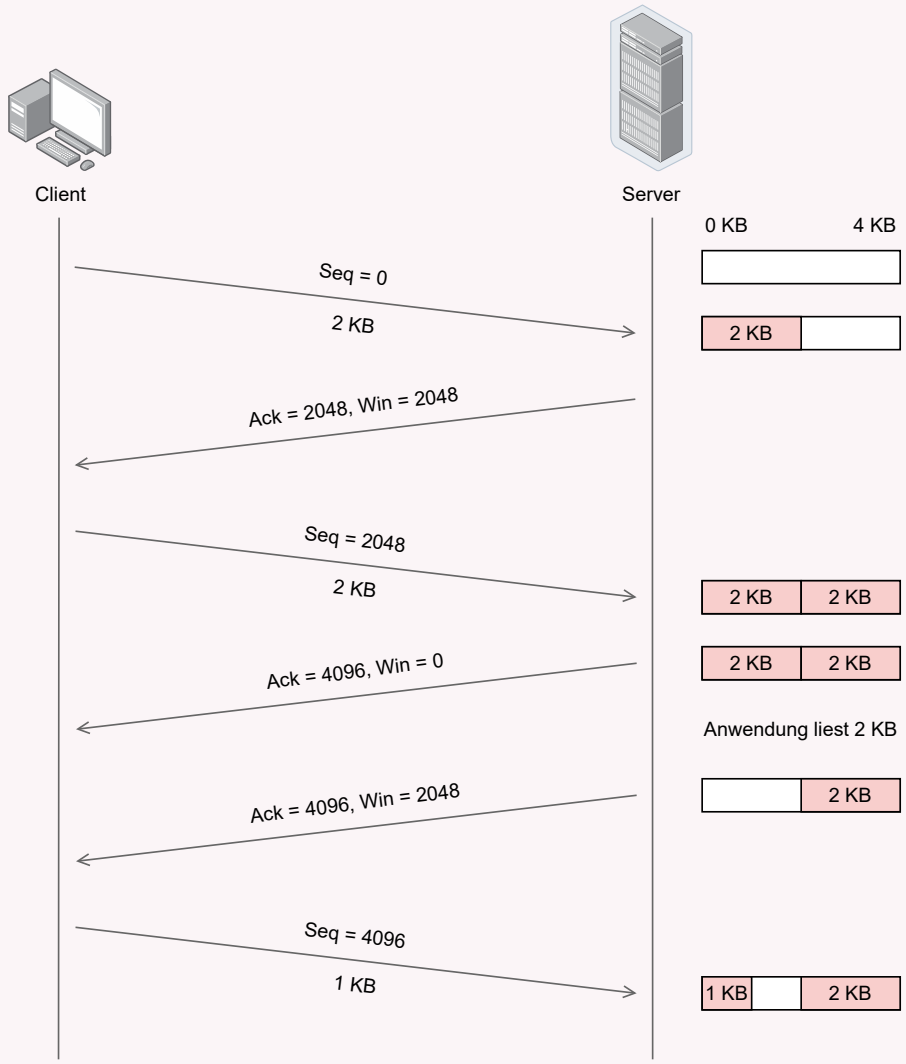
TCP verwendet kumulative `ACKs`. `ACK n+1` sagt aus, dass alle Daten von der vorherigen logischen Position bis zur Position n korrekt empfangen wurden und nun das Segment $n + 1$ erwartet wird.

Definition: Sliding Window in TCP

Da die Anwendung Daten aus dem Puffer liest, ändert sich der Füllstand des Puffers ständig. Deshalb ist es notwendig, den Datenfluss dem Füllstand entsprechend zu steuern. Dies geschieht mit dem *Sliding Window* und dessen Größe.

Mit dem Window-Feld kann er dem Sender mitteilen, dass er keine Daten mehr verschicken soll. Dies geschieht, indem er im Window-Feld den Wert Null einträgt (Zero Window). Der Wert Null entspricht dem freien Speicherplatz im Puffer.

Beispiel: Sliding Window in TCP



Bonus: Silly Window Syndrome

Zuerst sendet der Empfänger ein Zero Window, weil sein Empfangspuffer voll ist. Nachdem die Anwendung einige Bytes (z.B. 4) aus dem Empfangspuffer gelesen hat, sendet der Empfänger ein Window Update mit `Window = 4` an den Sender. Der Sender sendet dadurch ein TCP-Segment mit lediglich vier Bytes Nutzlast.

Dieses Verhalten setzt sich weiter so fort. Dadurch entsteht ein großer Protokoll-Overhead.

Eine Lösung ist, dass der Empfänger ein Zero Window sendet und so lange mit dem Window Update warten soll, bis die Anwendung mindestens die maximale Segmentgröße (*maximum segment size*) aus dem Puffer gelesen hat oder der Puffer halbleer ist – je nachdem, was zuerst eintritt.

Definition: TCP Receive Window

Die *TCP Receive Window (Size)* ist neben der Maximum Segment Size (MSS) ein Parameter, der die Funktion des Netzwerkprotokolls TCP steuert. Sie beschreibt die maximale Datenmenge, die ein Computer empfangen kann, ohne Daten bestätigen zu müssen.

Im Umkehrschluss ist es also die maximale Datenmenge, die ein Computer senden kann, ohne auf eine Empfangsbestätigung des Empfängers warten zu müssen.

Damit ist sichergestellt, dass der Empfangsspeicher (Puffer) des Empfängers nicht überläuft, da dieser nie mehr Daten am Stück empfängt, als er dem Sender durch Übermittlung seiner aktuellen Empfangsfenstergröße erlaubt hat. Erst wenn der Empfänger Daten bestätigt (und damit aus dem Puffer entfernt) hat, sendet der Sender die nächsten Daten.

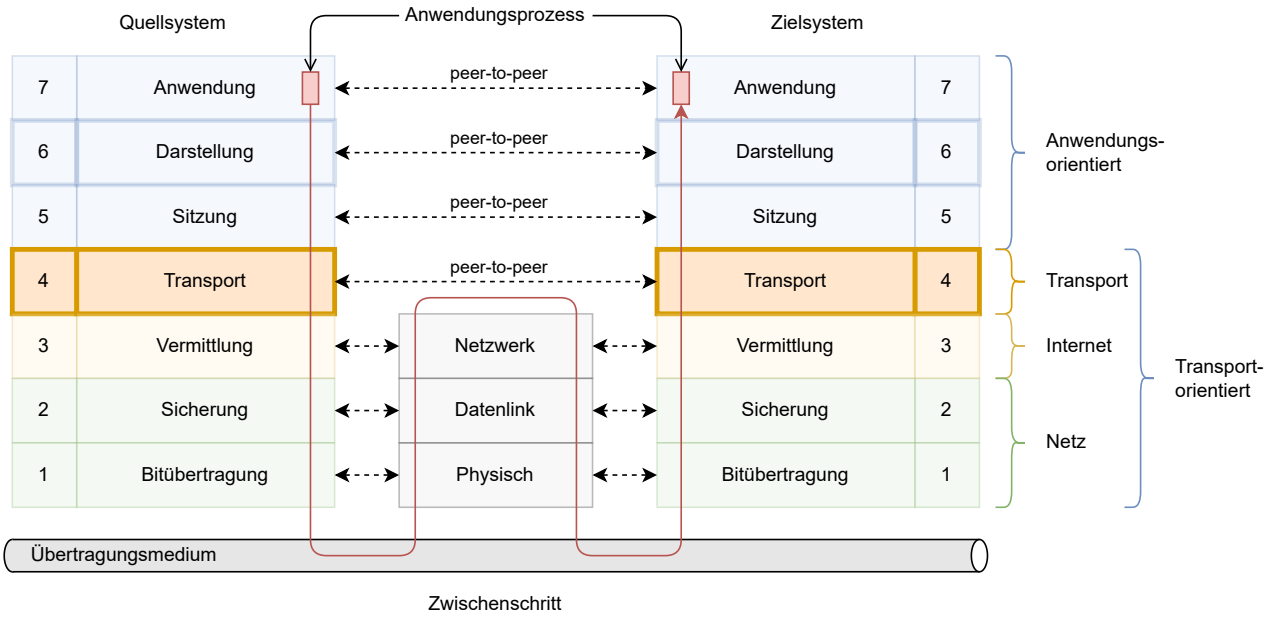
Bonus: Retransmission Timer

Zur Feststellung, wann ein Paket im Netzwerk verloren gegangen ist, wird vom Sender ein Timeout verwendet, bis zu dem das `ACK` der Gegenseite eingetroffen sein muss.

Ein zu niedriger Timeout bewirkt, dass Pakete, die eigentlich korrekt angekommen sind, wiederholt werden; ein zu hoher Timeout bewirkt, dass bei tatsächlichen Verlusten das zu wiederholende Paket unnötig spät gesendet wird.

Aufgrund unterschiedlicher Laufzeiten der zugrundeliegenden IP-Pakete ist nur ein dynamisch an die Verbindung angepasster Timer sinnvoll.

13 UDP



Definition: UDP

Das *User Datagram Protocol (UDP)* ist ein *verbindungsloses, nicht-zuverlässiges und ungesichertes wie auch ungeschütztes* Netzwerkprotokoll, das zur Transportschicht der Internetprotokollfamilie gehört.

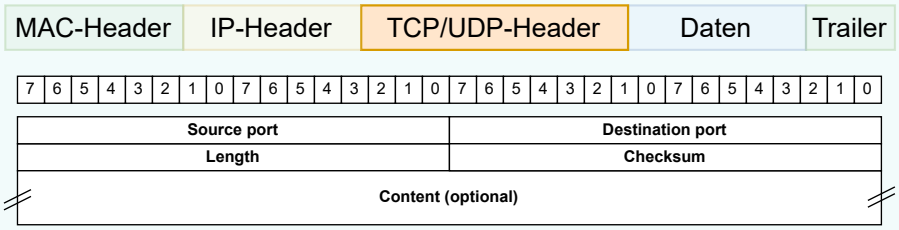
UDP ermöglicht den Versand von Datagrammen in IP-basierten Rechnernetzen.

UDP verwendet Ports, um versendete Daten dem richtigen Programm auf dem Zielrechner zukommen zu lassen. Dazu enthält jedes Datagramm die Portnummer des Dienstes, der die Daten erhalten soll.

Zusätzlich bietet UDP die Möglichkeit einer Integritätsüberprüfung an, indem eine Prüfsumme mitgesendet wird. Dadurch können fehlerhaft übertragene Datagramme erkannt und verworfen werden.

Daneben bietet die ungesicherte Übertragung den Vorteil von geringen Übertragungsverzögerungsschwankungen: Geht bei einer TCP-Verbindung ein Paket verloren, wird es automatisch neu angefordert. Das braucht Zeit, die Übertragungsdauer kann daher schwanken, was für Multimediaanwendungen schlecht ist. Bei VoIP (Discord, MS Teams) oder Streaming (Netflix, YouTube) z. B. käme es zu plötzlichen Aussetzern, bzw. die Wiedergabepuffer müssten größer angelegt werden. Bei verbindungslosen Kommunikationsdiensten bringen verlorengegangene Pakete dagegen nicht die gesamte Übertragung ins Stocken, sondern vermindern lediglich die Qualität.

Definition: UDP-Header

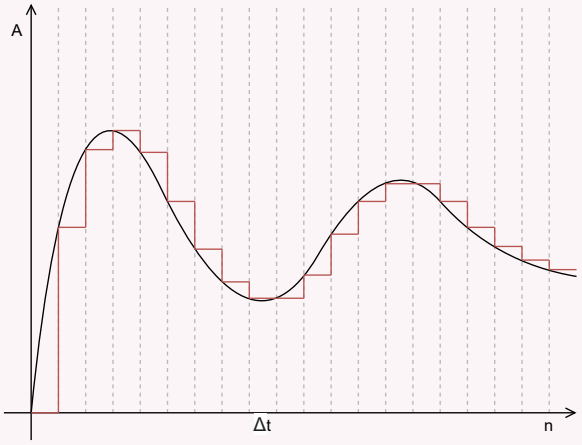


Bonus: Abtastung

Analoge Signale werden regelmäßig gemessen.

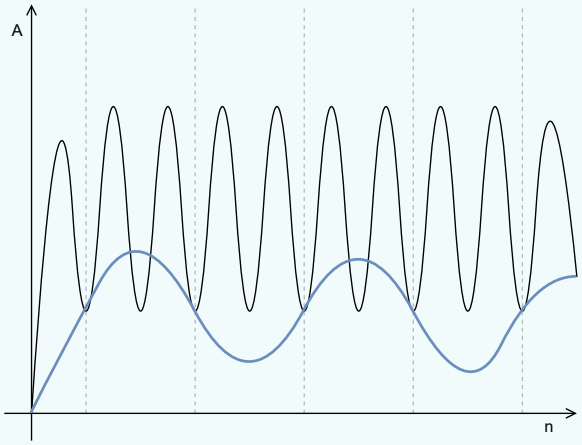
Dabei wird der Wert zu dem n-ten Zeitpunkten $t_n = n \cdot \Delta t$ gemessen.

Bei dem `Hold`-Verfahren wird der letzte gemessene Wert gehalten:



Definition: Sampling Rate

- Wird zu selten abgetastet, erhalten wir eine sehr ungenaue Approximation des ursprünglichen analogen Signals



- Wird zu häufig abgetastet, verbrachen wir Ressourcen, die eventuell für unseren Anwendungsfall nicht genutzt werden müssten

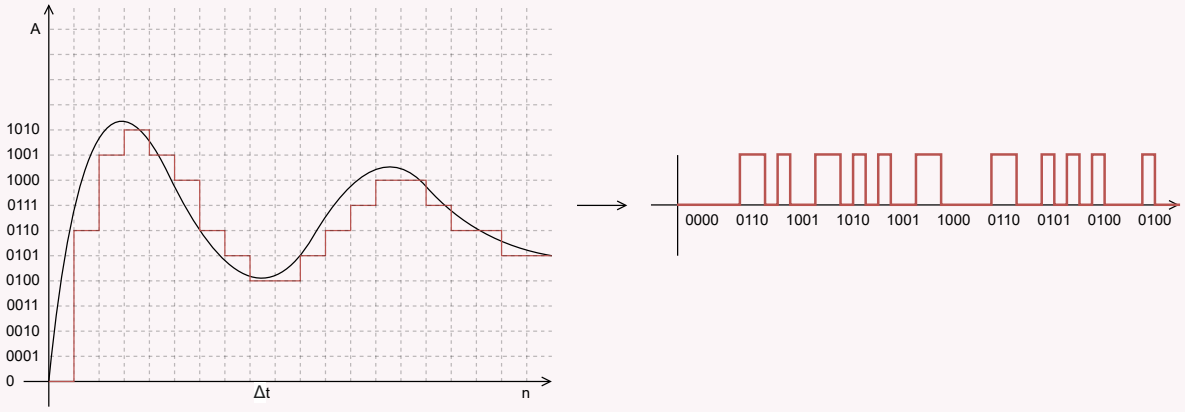
Nach den *Nyquist-Shannon-Abtasttheorem* soll gelten: Abtastrate $> 2 \cdot$ max. Frequenz

Die Qualität des resultierenden Digitalsignals ist dabei abhängig von der Abtastrate und Auflösung.

Die Bitrate ($^{\text{bit}}/_{s}$) sei definiert als:

$$\text{Bitrate} := \text{Kanäle} \cdot \text{Abtastrate (Hz)} \cdot \text{Auflösung (bit)}$$

Bonus: Quantisierung und Kodierung



Das Analoge Signal wird *quantisiert* (umgewandelt) und danach in digitale Werte kodiert.

Definition: RTP

Das RTP-Protokoll liefert Transportschnittstellen, die UDP erweitern:

- UDP liefert Port-Nummern
- IP die Adressen der Endpunkte
- `RTCP` liefert Statistiken auf UDP-Basis
- `RTSP` ist die „Fernbedienung“

7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0

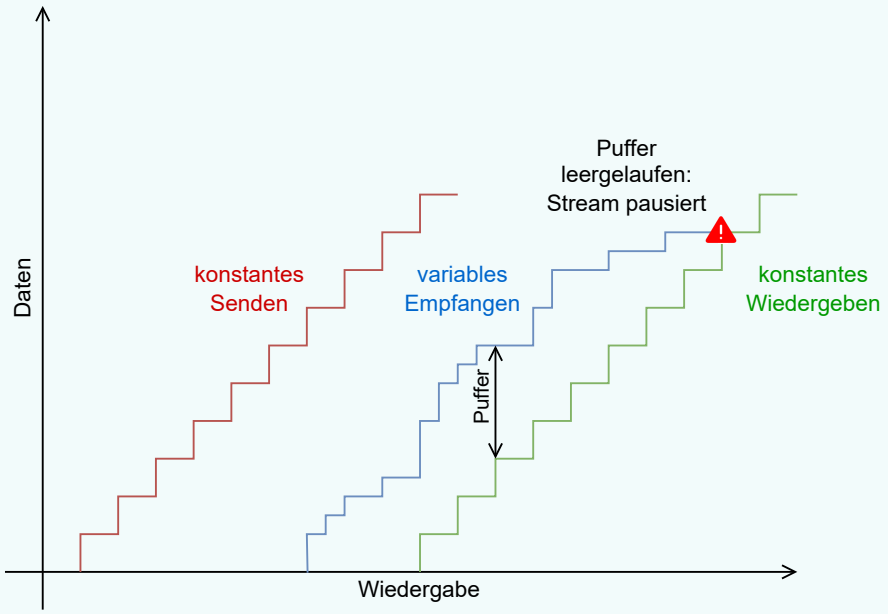
1	0	P	X	CC	M	PT	Sequence Number
Timestamp (in sample rate units)							
Synchronization Source (SSRC) Identifier							
Contributing Source (CSRC) identifiers (optional)							
Header Extension (optional)							

Definition: Streaming

Multimediale Netzanwendungen werden kategorisiert in:

1. Streaming gespeicherter Audio- und Video-Daten
2. Streaming von Live-Audio und -Video
3. Interaktive Audio- und Video-Nutzung

Insbesondere beim Streaming werden die Daten vorzeitig vom Client verwendet, bevor diese vollständig übertragen wurden.



In der Theorie ist Vorspulen nicht möglich, da die Daten noch nicht empfangen wurden, beim Rückspulen bzw. Pausieren kann man jedoch auf seinen lokalen Puffer zurückgreifen.

14 DNS

Definition: DNS

Socket-basierte Kommunikation nutzt IP-Adressen.

Im Gegensatz dazu wollen Endanwender sprechende Bezeichnungen, die leicht zu merken sind und Rückschlüsse auf ihre Bedeutung haben. Des Weiteren können sich IP-Adressen ändern, wenn z. B. ein Dienst umzieht.

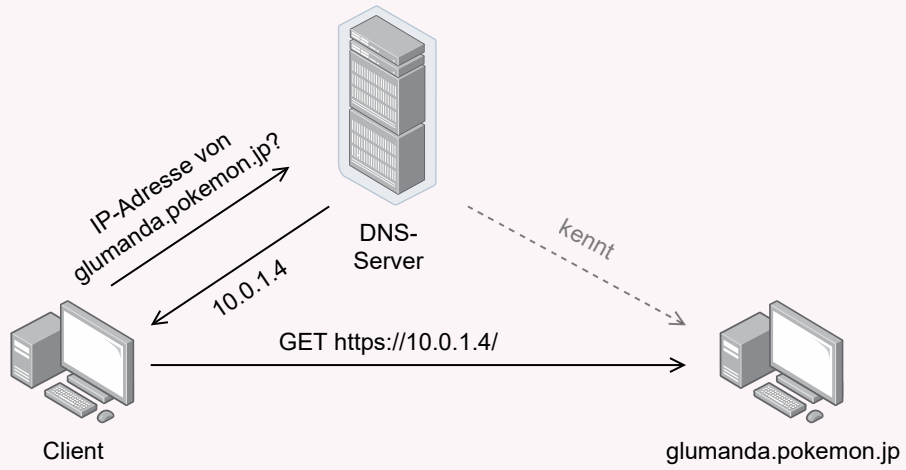
Um diese Anforderungen zu erfüllen, wurde ein System entwickelt um Namen zu übersetzen bzw. aufzulösen.

Domain Name System (DNS) ist eine verteilte Datenbank, welche als Serverprozess auf UDP Port 53 reagiert. Jeder Namensraum ist verantwortlich für sein Segment.

Zur Geschwindigkeitsoptimierung werden angefragte Daten von Clients zwischengespeichert, wenn der DNS Server bei übergeordneten DNS Servern anfragt.

Die IP-Adresse des DNS-Servers muss in jeden Host manuell oder via DHCP eingetragen werden.

Beispiel: DNS



Definition: Domäne

Symbolische Namen bestehen aus dem Hostnamen und der Domäne. Die Domäne besteht aus der *Top Level Domain* und potentiellen *Subdomains*.

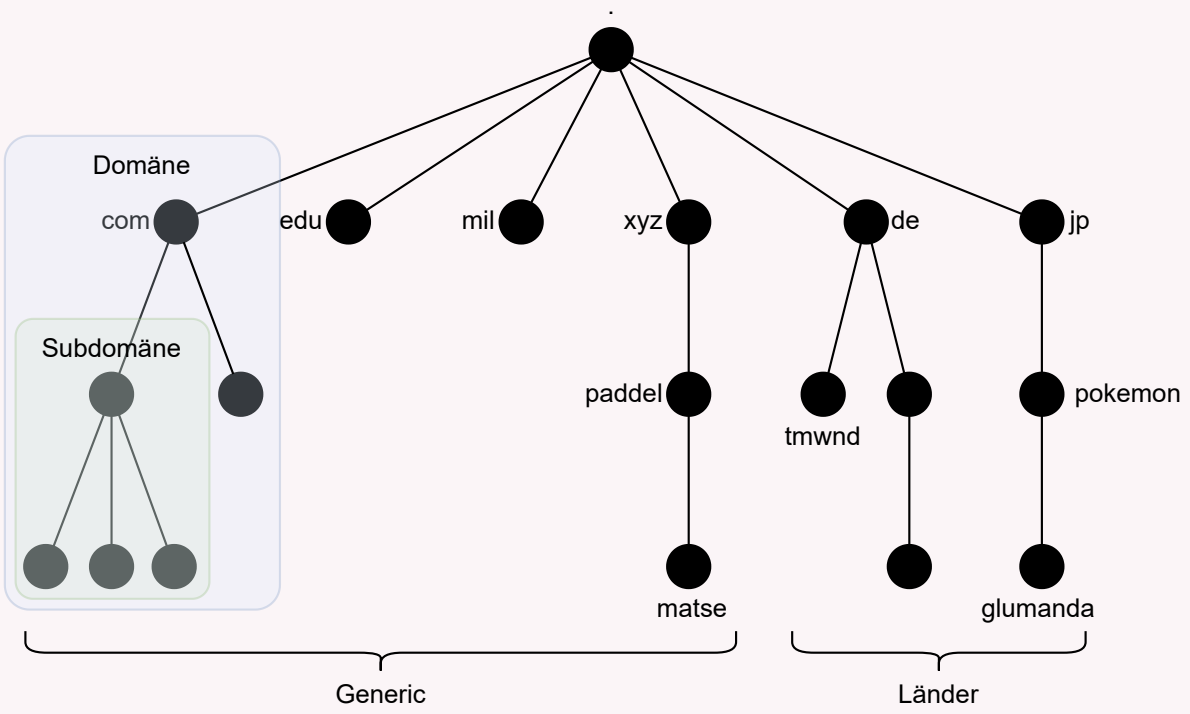
Zur Strukturierung lassen sich die Informationen als Baum darstellen.[a]

Der gesamte Name (*Full Qualified Domain-Name (FQDN)*) darf dabei die Länge von 255 Bytes nicht überschreiten.

Jedes Blatt besitzt eine eindeutige IP-Adresse.

[a] Die einzelnen Labels haben eine Länge von einem bis 63 Bytes.

Beispiel: Domäne



Definition: Zone

Eine *Zone* ist ein autarker und gemeinsam administrierter Bereich im DNS-Namensraum mit eigener Datenbank.

Jede Zone hat einen primären und beliebig viele sekundäre Nameserver.

- Jeder Namensserver kennt nur einen Ausschnitt des gesamten Namensraums
- Jeder Nameserver kennt alle IP-Adressen seiner direkt untergeordneten Sub-Domains.
- Sekundäre Nameserver führen periodisch Updates ihrer Datenbasis durch.
- Der primäre Namensserver kennt den vollständigen Datenbestand.

Zur Einrichtung einer Zone muss man den übergeordneten Knoten davon überzeugen, die Verwaltung zu delegieren.

Definition: Root-Nameserver

Root-Nameserver bilden die Wurzel der hierarchischen DNS-Struktur. Es gibt Weltweit 13 Root-Nameserver.

z. Zt. `{a-m}.root-server.net`

Definition: Namensauflösung

Der DNS-Server beginnt bei dem Nameserver der hintersten bekannten Zone die Suche. Sollte er keine Daten im Cache haben, ist dies der nächste Root-Nameserver.

Der angesprochene Nameserver kann den Client entweder zum nächsten Nameserver weiterleiten (iterativ) oder dort selber anfragen (rekursiv).

Beispiel: Namensauflösung

Beispiel 1:

Ein Client fragt bei seinem lokalen DNS die IP-Adresse des Hosts `glumanda.pokemon.jp` an.

Da der DNS-Server keine Daten gespeichert hat, fragt er den Root-Nameserver nach der IP-Adresse. Dieser leitet ihn weiter auf den DNS-Server von `jp`.

Der Nameserver von `jp` kennt den exakten Host ebenfalls nicht und leitet unseren DNS Server auf `pokemon.jp` weiter. Dieser kennt den Host und antwortet mit der IP-Adresse `10.0.1.4`.

Die IP-Adresse wird von dem DNS an den Client weitergeleitet.

Beispiel 2:

Nach dem ersten Beispiel fragt ein zweiter Client nach der IP-Adresse von `pikachu.pokemon.jp`.

Da unser DNS-Server den Nameserver von `pokemon.jp` zwischengespeichert hat, fragt er direkt dort an und erhält die IP-Adresse `10.0.1.22`.

Diese wird erneut an den Client weitergeleitet.

Bonus: Weitere DNS-Dienste

Neben der Auflösung von Namen ist der DNS-Server zuständig für:

- Die Weiterleitung zu dem Mail-Server der Zone (*Mail Exchange Resource Record* (MX-RR))
- Die Weiterleitung zu IP-Telefon Kommunikationspartnern
- Caching von Anfragen

Bonus: DNS-Datenbank

In der *DNS-Datenbank* befinden sich folgende Typen von Einträgen:

Typ	Bedeutung	Wert
SOA	Start of Authority	Parameter für diese Zone
A	IPv4-Adresse eines Hosts	32 bit Integer
AAAA	IPv6-Adresse eines Hosts	128 bit Integer
MX	Mail-Exchange	Priority, domain willing to accept Mail
NS	Name-Server	Name des Name-Servers des Namensraums
CNAME	Canonical Name	(Sub-) Domain Name
PTR	Pointer	Alias für eine IP-Adresse
SPF	Sender policy framework	Text encoding für Mails Policy
SRV	Service	Host, auf dem der Service läuft
TXT	Text	Kommentare

Beispiel: DNS-Datenbank

Die imaginäre Datenbank des Nameservers von `pokemon.jp`

```
pokemon.jp      86400  IN  NS     pokemon-dns
feuerpokemon    86400  IN  CNAME  pokemon.jp

eich            86400  IN  A      10.0.255.1
esche           86400  IN  A      10.0.255.2

pokemon         86400  IN  MX     1 eich
pokemon         86400  IN  MX     2 esche

glumanda        86400  IN  A      10.0.1.4
glumanda        86400  IN  AAAA   ::4
pikachu         86400  IN  A      10.0.1.22
```

Bonus: DNS-Attacken

DNS ist ein zentraler Bestandteil eines Netzwerks und ein kritischer Dienst.

Da er nur UDP verwendet, ist er häufig Angriffsziel.

- *DNS Spoofing*: Durch Manipulation (Maskierung) werden Benutzer für Phishing-Attacken auf falsche Webseiten gelenkt.
- *Cache Poisoning*: Der Cache des DNS Servers wird vergiftet, in der Hoffnung, dass Clients den Cache unhinterfragt in ihren Cache übernehmen.
- *DDoS*

Um diese Angriffe zu verhindern gibt es verschiedene Sicherheitserweiterungen[a].

[a] z. B. *DNS over TLS*

Bonus: DNS-Filter

DNS-Anfragen können mit einem Proxy gefiltert werden, um z. B. unerwünschte Anfragen abzubrechen (z. B. auf Werbeseiten).

Beispielsweise lässt sich ein Raspberry Pi mittels *Pi-Hole* zu solch einem DNS-Proxy umfunktionieren.

Erwünschte Anfragen könnte man z. B. auf den Google-DNS-Server `8.8.8.8` weiterleiten, oder seinen lokalen Nameserver des Routers nutzen.

15 Anwendungsprotokolle

15.1 HTTP

Definition: HTTP

Das *Hypertext Transfer Protocol (HTTP)* ist ein textbasiertes Protokoll zur Übertragung von Daten auf der Anwendungsschicht über ein Rechnernetz. Es wird hauptsächlich eingesetzt, um Webseiten (Hypertext-Dokumente) aus dem World Wide Web (WWW) in einen Webbrowser zu laden. Es ist jedoch nicht prinzipiell darauf beschränkt und auch als allgemeines Dateiübertragungsprotokoll sehr verbreitet.

HTTP ist ein zustandsloses Protokoll (Informationen aus früheren Anforderungen gehen verloren). Ein zuverlässiges Mitführen von Sitzungsdaten kann erst auf der Anwendungsschicht durch eine Sitzung über einen Sitzungsbezeichner implementiert werden.

Über Cookies in den Header-Informationen können aber Anwendungen realisiert werden, die Statusinformationen (Benutzereinträge, Warenkörbe) zuordnen können. Dadurch werden Anwendungen möglich, die Status- beziehungsweise Sitzungseigenschaften erfordern. Auch eine Benutzerauthentifizierung ist möglich.

Normalerweise kann die Information, die über HTTP übertragen wird, auf allen Rechnern und Routern gelesen werden, die im Netzwerk durchlaufen werden. Über HTTPS jedoch kann die Übertragung verschlüsselt erfolgen.

Definition: HTTP-Anfragen und -Methoden

Die Kommunikationseinheiten in HTTP zwischen Client und Server werden als Nachrichten bezeichnet, von denen es zwei unterschiedliche Arten gibt: die Anfrage (*Request*) vom Client an den Server und die Antwort (*Response*) als Reaktion darauf vom Server zum Client.

Jede Nachricht besteht dabei aus zwei Teilen, dem *Message Header* bzw. *HTTP-Header* und dem *Message Body*.

Der Message Header enthält Informationen über den Message Body wie etwa verwendete Kodierungen oder den Inhaltstyp, damit dieser vom Empfänger korrekt interpretiert werden kann.

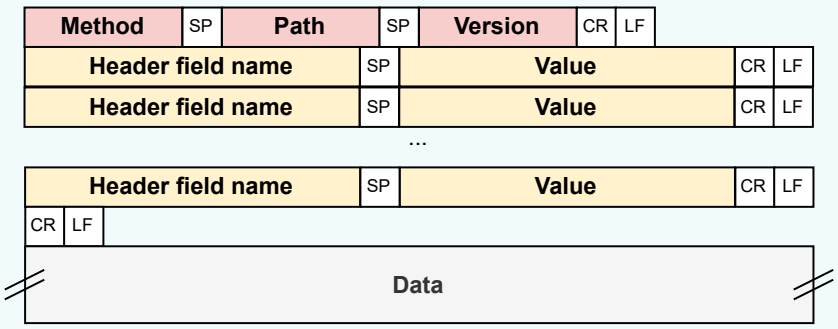
Der Message Body enthält schließlich die Nutzdaten.

HTTP-Anfragemethoden sind:

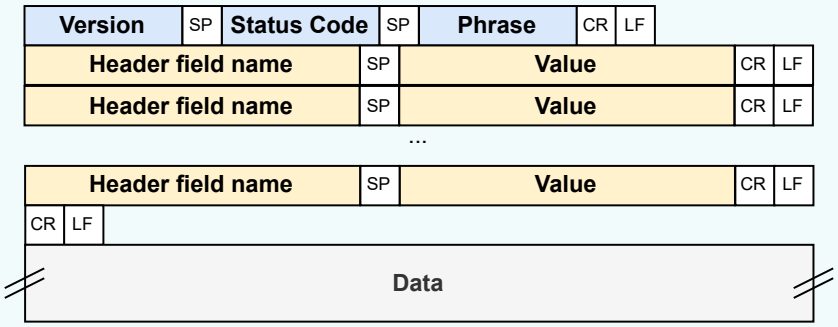
- `GET` fordert eine Ressource an.
- `POST` schickt Daten zur weiteren Verarbeitung zum Server.
- `HEAD` weist den Server an, die gleichen HTTP-Header wie bei `GET`, nicht jedoch den Nachrichtenrumpf mit dem eigentlichen Dokumentinhalt zu senden.
- `PUT` dient dazu, eine Ressource unter Angabe des Ziel-URIs auf einen Webserver hochzuladen.
- `PATCH` ändert ein bestehendes Dokument ohne dieses wie bei PUT vollständig zu ersetzen.
- `DELETE` löscht die angegebene Ressource auf dem Server.
- `TRACE` liefert die Anfrage so zurück, wie der Server sie empfangen hat.
- `OPTIONS` liefert eine Liste der vom Server unterstützten Methoden und Merkmale.
- `CONNECT` wird von Proxyservern implementiert, die in der Lage sind, SSL-Tunnel zur Verfügung zu stellen.

Die Header-Felder werden jeweils durch `CR LF` (*Carriage Return* und *Line Feed*) getrennt, innerhalb der Header-Felder wird durch `SP` (*Space*) getrennt.

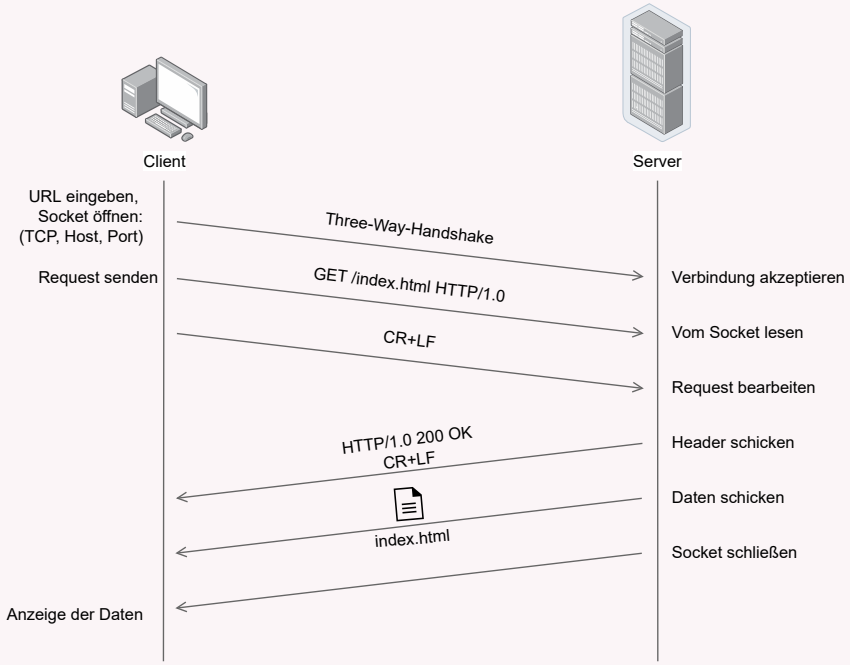
Definition: HTTP-Request



Definition: HTTP-Response



Beispiel: HTTP-GET-Request



15.2 MIME

Definition: MIME

Die *Multipurpose Internet Mail Extensions (MIME)* sind Erweiterungen des Internetstandards RFC 822, der das Datenformat von E-Mails definiert. Dieser sieht nur den American Standard Code for Information Interchange (ASCII) vor.

Die MIME schaffen Kompatibilität für zusätzliche Zeichen wie Umlaute sowie für Multimedia (etwa bei Mail-Anhängen).

Darüber hinaus findet MIME Anwendung bei der Deklaration von Inhalten in verschiedenen Internetprotokollen wie etwa HTTP sowie bei Desktop-Umgebungen wie KDE, Gnome, Xfce oder Aqua[a].

Der MIME-Header besteht aus:

- `MIME-Version`:
  - Damit ein MIME-Dokument mit RFC 2045 konform ist, muss sich dieses Feld im Header der höchsten Ebene befinden und den Wert `1.0` haben.
- `Content-Type`:
  - `Content-Type` hat standardmäßig den Wert `text/plain`.
  - `Content-Type` definiert den Datentyp in jedem Nachrichtenteil als `Typ/Subtyp`.
  - Der MIME-Parser akzeptiert die meisten Werte für `Content-Type` und speichert sie in der logischen Baumstruktur.
- `Content-Transfer-Encoding`:
  - Optional.
- `Content-ID`:
  - Optional.
  - Mit diesem Feld können Nachrichtenteile mit einer Inhalts-ID versehen und so von anderen Teilen der Nachricht referenziert werden.
- `Content-Description`:
  - Optional.
  - Dieses Feld kann Beschreibungen zu Nachrichtenteilen enthalten.

[a] Aqua ist die grafische Benutzeroberfläche von macOS.

Definition: MIME Content-Types

Die wichtigsten Typen sind unter Anderem:

`Content-Type`	Anwendung
`text/plain`	Allgemein für eine typische Mail oder Newsnachricht verwendet.
`text/xml`	Allgemein in Verbindung mit SwA-Nachrichten (SOAP with Attachments) verwendet.
`application/octet-stream`	Verwendet, wenn der Typ der Nachricht unbekannt ist und enthält beliebige Daten als Bytes.
`application/xml`	Für anwendungsspezifische XML-Daten verwendet.
`x-type`	Für vom Standard abweichende Inhaltstypen verwendet. Muss mit `x-` beginnen.
`image/jpeg`	Für Bilder verwendet. `image/jpeg` und `image/gif` sind allgemein gebräuchliche Bildformate.
`multipart/related`	Für mehrere zusammengehörige Teile in einer Nachricht verwendet, insbesondere mit SwA.
`multipart/signed`	Für mehrere zusammengehörige Teile in einer Nachricht, einschließlich Signatur, verwendet.
`multipart/mixed`	Für mehrere unabhängige Teile in einer Nachricht verwendet.

15.3 Sichere Protokolle

Definition: HTTPS

Hypertext Transfer Protocol Secure (HTTPS) ist ein Kommunikationsprotokoll im World Wide Web, mit dem Daten abhörsicher übertragen werden können. Es stellt eine Transportverschlüsselung dar.

HTTPS wird zur Herstellung von Vertraulichkeit und Integrität in der Kommunikation zwischen Webserver und Webbrowser (Client) im World Wide Web verwendet. Dies wird unter anderem durch Verschlüsselung und Authentifizierung erreicht.

Ohne Verschlüsselung sind Daten, die über das Internet übertragen werden, für jeden, der Zugang zum entsprechenden Netz hat, als Klartext lesbar. Mit der zunehmenden Verbreitung von offenen (d. h. unverschlüsselten) WLANs nimmt die Bedeutung von HTTPS zu, weil damit die Inhalte unabhängig vom Netz verschlüsselt werden können.

Die Authentifizierung dient dazu, dass beide Seiten der Verbindung beim Aufbau der Kommunikation die Identität des Verbindungspartners überprüfen können. Dadurch sollen Man-in-the-Middle-Angriffe und teilweise auch Phishing verhindert werden.

Syntaktisch ist HTTPS identisch mit dem Schema für HTTP, die zusätzliche Verschlüsselung der Daten geschieht mittels SSL/TLS: Unter Verwendung des SSL-Handshake-Protokolls findet zunächst eine geschützte Identifikation und Authentifizierung der Kommunikationspartner statt. Anschließend wird mit Hilfe asymmetrischer Verschlüsselung oder des Diffie-Hellman-Schlüsselaustauschs ein gemeinsamer symmetrischer Sitzungsschlüssel ausgetauscht. Dieser wird schließlich zur Verschlüsselung der Nutzdaten verwendet.

Definition: SSL/TLS

Transport Layer Security (TLS), auch bekannt unter der Vorgängerbezeichnung *Secure Sockets Layer (SSL)*, ist ein Verschlüsselungsprotokoll zur sicheren Datenübertragung im Internet.

TLS besteht aus den beiden Hauptkomponenten *TLS Handshake* und *TLS Record*.

Im OSI-Modell ist TLS in Schicht 5 (der Sitzungsschicht) angeordnet. Im TCP/IP-Modell ist TLS oberhalb der Transportschicht (zum Beispiel TCP) und unterhalb Anwendungsprotokollen wie HTTP oder SMTP angesiedelt.

Definition: TLS Handshake Protocol

Das *TLS Handshake Protocol* baut auf dem TLS Record Protocol auf und erfüllt die folgenden Funktionen, noch bevor die ersten Bits des Anwendungsdatenstromes ausgetauscht wurden:

- Aushandeln zu benutzender kryptografischer Algorithmen und Schlüssel. TLS unterstützt auch eine unverschlüsselte Übertragung.
- Identifikation und Authentifizierung der Kommunikationspartner auf Basis asymmetrischer Verschlüsselungsverfahren und Public-Key-Kryptografie.[a]

Insgesamt besteht der TLS Handshake aus vier Phasen:

1. Der Client schickt zum Server ein `ClientHello`, und der Server antwortet dem Client mit einem `ServerHello`. Die Parameter der Nachrichten sind:
   - die Version (die höchste vom Client unterstützte TLS-Protokoll-Version)
   - eine 32 Byte lange Zufallsinformation
   - eine Session-ID
2. Der Server identifiziert sich gegenüber dem Client.
   - Hierzu wird per Certificate ein *X.509-Zertifikat* an den Client geschickt, gefolgt von einem `CertificateVerify` (in einigen TLS Versionen).
   - Die `CertificateVerify` Nachricht enthält eine Unterschrift von zuvor ausgetauschten Nachrichten.
   - Der Client prüft das Zertifikat und die Unterschrift. Bei Misserfolg bricht der Client die Verbindung ab.
   - Außerdem kann der Server optional per `CertificateRequest` ein Zertifikat zur Client-Authentifizierung anfordern.
3. Das zuvor erhaltene Server-Zertifikat enthält den öffentlichen Schlüssel des Servers. Es wird ein Diffie-Hellman-Schlüsselaustausch durchgeführt, um ein gemeinsames *pre-master-secret* zu generieren.
4. Diese Phase schließt den Handshake ab. Aus dem vorhandenen pre-master-secret kann das *master secret* abgeleitet werden, das einen einmaligen *Sitzungsschlüssel (Session Key)* darstellt.
   - Aus dem master secret werden wiederum *Schlüssel* abgeleitet, die zum Ver- und Entschlüsseln der Daten sowie für die Integritätsprüfung verwendet werden.
   - Die Nachrichten, die die Kommunikationspartner sich nun gegenseitig zusenden, werden nur noch verschlüsselt übertragen.

[a] Dieser Schritt ist optional eine Zwei-Wege-Authentifizierung (in diesem Fall wird manchmal von mutual TLS gesprochen), für gewöhnlich authentifiziert sich aber nur der Server gegenüber dem Client.

Definition: TLS Record Protocol

Das *TLS Record Protocol* dient zur Absicherung der Verbindung.

Es setzt direkt auf der Transportschicht auf und bietet zwei verschiedene Dienste, die einzeln oder gemeinsam genutzt werden können:

- Ende-zu-Ende-Verschlüsselung mittels symmetrischer Algorithmen. Der verwendete Schlüssel wird dabei im Voraus über ein weiteres Protokoll (zum Beispiel das TLS Handshake Protocol) ausgehandelt und kann nur einmal für die jeweilige Verbindung verwendet werden.
- Sicherung der Nachrichten-Integrität und Authentizität durch einen Message Authentication Code (MAC).

Bonus: X.509 Zertifikat

X.509 ist ein Standard für eine Public-Key-Infrastruktur zum Erstellen digitaler Zertifikate.

In der elektronischen Kommunikation finden X.509-Zertifikate Anwendung bei den TLS-Versionen diverser Übertragungsprotokolle, wie z. B. beim Abruf von Web-Seiten mit HTTPS oder zum Unterschreiben und Verschlüsseln von E-Mails nach dem S/MIME-Standard.

Struktur eines X.509-v3-Zertifikats:

- Zertifikat
  - Version
  - Seriennummer
  - Algorithmen-ID
  - Aussteller
  - Gültigkeit
    - von
    - bis
  - Zertifikatinhaber
  - Zertifikatinhaber-Schlüsselinformationen
    - Public-Key-Algorithmus
    - Public Key des Zertifikatinhabers
  - Eindeutige ID des Ausstellers (optional)
  - Eindeutige ID des Inhabers (optional)
  - Erweiterungen (optional)
- Zertifikat-Signaturalgorithmus
- Zertifikat-Signatur

Definition: Signatur

Eine *digitale Signatur* ist ein asymmetrisches Kryptosystem, bei dem ein Sender mit Hilfe eines geheimen Signaturschlüssels (dem Private Key) zu einer digitalen Nachricht (d. h. zu beliebigen Daten) einen Wert berechnet, der ebenfalls digitale Signatur genannt wird. Dieser Wert ermöglicht es jedem, mit Hilfe des öffentlichen Verifikationsschlüssels (dem Public Key) die nichtabstreitbare Urheberschaft und Integrität der Nachricht zu prüfen.

Um eine mit einem Signaturschlüssel erstellte Signatur einer Person zuordnen zu können, muss der zugehörige Verifikationsschlüssel dieser Person zweifelsfrei zugeordnet sein.

Bei einer digitalen Signatur wird der private Schlüssel in der Regel nicht direkt auf die Nachricht angewendet, sondern auf deren Hash-Wert, der mittels einer Hashfunktion (wie z. B. SHA-1) aus der Nachricht berechnet wird.

Um Angriffe zu verhindern, muss diese Hashfunktion kollisionsresistent sein, d. h., es muss praktisch unmöglich sein, zwei unterschiedliche Nachrichten zu finden, deren Hash-Wert identisch ist.

Definition: Diffie-Hellmann-Schlüsselaustausch

Der *Diffie-Hellman-Schlüsselaustausch* ist ein Verfahren, mit dem sich ein gemeinsamer Sitzungsschlüssel zwischen zwei Kommunikationspartnern sicher über ein potenziell unsicheres Übertragungsmedium vereinbaren lässt.[a]

Haben sich die Kommunikationspartner auf einen gemeinsamen Sitzungsschlüssel geeignet, verwenden sie diesen zum Ver- und Entschlüsseln der Daten. Der Diffie-Hellman-Schlüsselaustausch kommt häufig zusammen mit Digitalen Signaturen zum Einsatz.

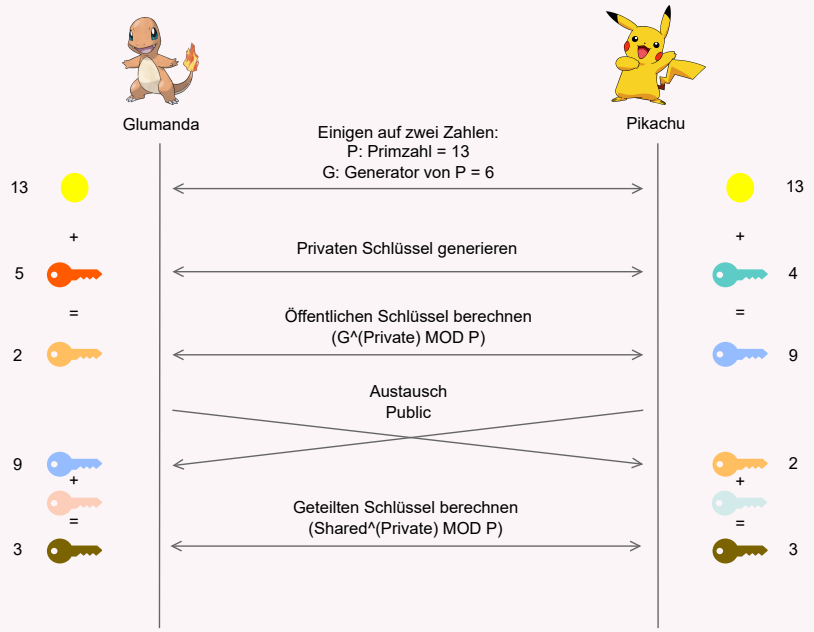
Ablauf:

1. Die beiden Kommunikationspartner einigen sich auf eine große öffentliche Primzahl p und eine kleinere Zahl g.
2. Gleichzeitig wählen beide jeweils eine zufällig Zahl x und y.[b] x und y werden niemals zwischen den Kommunikationspartnern ausgetauscht.
3. Aus ihrer zufälligen Zahl und den beiden öffentlichen Zahlen berechnen beide eine neue Zahl A und B. Diese sind quasi öffentliche Schlüssel.
4. Die Zahlen A und B tauschen die beiden Kommunikationspartner aus.
5. Aus den Zahlen A und B können sie den gemeinsame Schlüssel k jeweils mit ihren zufälligen Zahlen x und y berechnen.
6. Den Schlüssel k können die Teilnehmer anschließend für das Ver- und Entschlüsseln der Daten verwenden.

[a] Streng genommen findet kein Schlüsselaustausch statt, da der gemeinsame Sitzungsschlüssel niemals übertragen wird. Die Schlüssel werden über andere übermittelte Informationen berechnet. Für externe Angreifer, die das Medium abhören, ist das Berechnen des gemeinsamen Sitzungsschlüssel mit vertretbarem Aufwand mathematisch nicht möglich.

[b] Es handelt sich um persönliche Schlüssel, die dem anderen Kommunikationspartner nicht bekannt sind.

Beispiel: Diffie-Hellmann-Schlüsselaustausch



15.4 FTP

Definition: FTP

Das *File Transfer Protocol (FTP)* ist ein zustandsbehaftetes Netzwerkprotokoll zur Übertragung von Dateien über IP-Netzwerke.

FTP ist in der Anwendungsschicht (Schicht 7) des OSI-Schichtenmodells angesiedelt.

Es wird benutzt, um Dateien vom Client zum Server (Hochladen), vom Server zum Client (Herunterladen) oder clientgesteuert zwischen zwei FTP-Servern zu übertragen[a]. Außerdem können mit FTP Verzeichnisse angelegt und ausgelesen sowie Verzeichnisse und Dateien umbenannt oder gelöscht werden.

Das FTP verwendet für die Steuerung und Datenübertragung jeweils separate Verbindungen.

Beim *aktiven FTP* (auch *Active Mode*) öffnet der Client einen zufälligen Port und teilt dem Server diesen sowie die eigene IP-Adresse mittels des `PORT`- oder des `EPRT`-Kommandos mit. Die Datenübertragung auf der Server-Seite erfolgt dabei über Port 20.

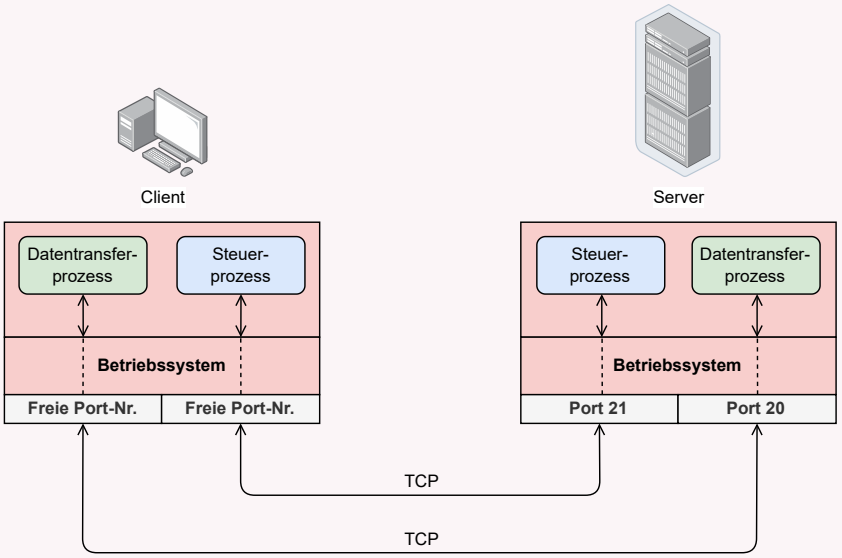
Die Kommunikation mit Befehlen erfolgt ausschließlich auf dem Control Port. Somit bleibt es möglich, dass während der Datenübertragung die Partner noch immer miteinander kommunizieren können.

Beim *passiven FTP* (auch *Passive Mode*) sendet der Client ein `PASV`- oder ein `EPSV`-Kommando, der Server öffnet einen Port und übermittelt diesen mitsamt IP-Adresse an den Client. Diese Technik wird eingesetzt, wenn der Server keine Verbindung zum Client aufbauen kann.[b]

[a] *File Exchange Protocol*

[b] Dies ist beispielsweise der Fall, wenn der Client sich hinter einem Router befindet, der die Adresse des Clients mittels NAT umschreibt, oder wenn eine Firewall das Netzwerk des Clients vor Zugriffen von außen abschirmt.

Beispiel: FTP

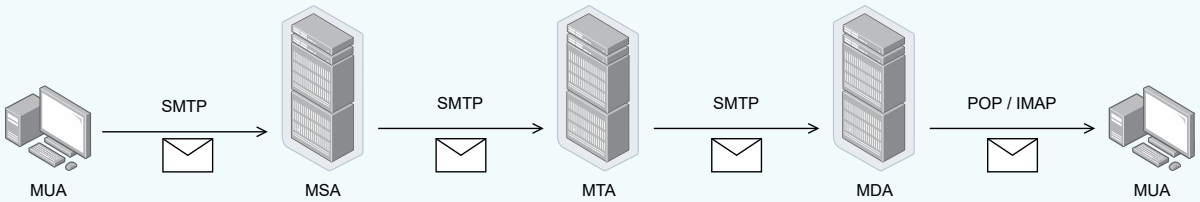


15.5 E-Mail

Definition: E-Mail

Ein E-Mail-System besteht aus zwei Subsystemen:

- *Mail User Agent (MUA)*:
  - E-Mail-Programm auf dem Client-Rechner des Benutzers
  - Empfang und Anzeigen von E-Mails
  - Erstellung neuer E-Mails und Beantwortung früherer E-Mails
  - Speichern und Verwalten empfangener E-Mails
- *Mail Submission Agent (MSA)* bzw. *Message Transfer Agent (MTA)*:
  - Mailserver als langlebiger Prozess
  - MSA nimmt E-Mails entgegen, die von MUA losgeschickt wurden und leitet diese ggf. über mehrere MTA (Mail Relays) zum Ziel (*Mail Delivery Agent (MDA)*) weiter
  - Zwischenspeicherung von Nachrichten für User oder andere MTA



Bonus: POP3

Das *Post Office Protocol (POP)* ist ein Übertragungsprotokoll, über das ein Client E-Mails von einem E-Mail-Server abholen kann.

POP3 ist ein ASCII-Protokoll, wobei die Steuerung der Datenübertragung durch Kommandos geschieht, die standardmäßig an den Port 110 geschickt werden.

POP3 ist in der Funktionalität sehr beschränkt und erlaubt nur das Auflisten, Abholen und Löschen von E-Mails am E-Mail-Server.

Für weitere Funktionen wie hierarchische Mailboxen direkt am Mailserver, Zugriff auf mehrere Mailboxen während einer Sitzung, Vorselektion der E-Mails usw. müssen Protokolle wie IMAP verwendet werden.

Bonus: SMTP

Das *Simple Mail Transfer Protocol (SMTP)* ist ein Protokoll der Internetprotokollfamilie, das zum Austausch von E-Mails in Computernetzen dient. Es wird dabei vorrangig zum Einspeisen und zum Weiterleiten von E-Mails verwendet. Zum Abholen von Nachrichten kommen andere, spezialisierte Protokolle wie POP3 oder IMAP zum Einsatz.

SMTP-Server nehmen traditionell Verbindungen auf Port 25 (`smtp`) entgegen.

Bonus: IMAP

Das *Internet Message Access Protocol (IMAP)* ist ein Netzwerkprotokoll, das ein Netzwerkdateisystem für E-Mails bereitstellt.

IMAP wurde in den 1980er Jahren mit dem Aufkommen von Personal Computern entworfen, um bei der Mail-Kommunikation Abhängigkeiten von einzelnen Client-Rechnern aufzulösen.

Zu diesem Zweck erweitert IMAP die Funktionen und Verfahren des Post Office Protocol (POP) so, dass Benutzer ihre Mails, Ordnerstrukturen und Einstellungen auf den Servern speichern und belassen können.

Während bei der Verwendung von POP die Nachrichten entweder nach dem Abruf gelöscht oder beim nächsten Abruf erneut empfangen werden, ermöglicht IMAP eine zentrale Verwaltung mit Suchfunktion und serverseitiger Gelesen-Markierung.

15.6 Verwaltungsprotokolle

Bonus: Telnet

Telnet (Teletype Network) ist ein heutzutage weniger verbreitetes Netzwerkprotokoll.

Dieses alte und bekannte Client/Server-Protokoll basiert auf einem zeichenorientierten Datenaustausch über eine TCP-Verbindung.

Telnet wird typischerweise zur Fernsteuerung von Computern in Form von textbasierten Ein- und Ausgaben eingesetzt. Hierzu baut der Telnet-Client eine unverschlüsselte Verbindung zu einem Telnet-Server auf.

In dieser Phase wird ein notwendiges Kennwort im Klartext übergeben.

Nach dem Verbindungsaufbau wird das Telnet-Protokoll optional initiiert. Üblicherweise handelt es sich um eine Login-Konsole mit voller Befehlsgewalt.

Durch die hohe Befehlsgewalt und die unverschlüsselte Übertragung gilt dieses Verfahren als unsicher und sollte durch eine SSH-Verbindung ersetzt werden, die auch das Telnet-Protokoll umsetzt, aber verschlüsselt überträgt.

Definition: SSH

Secure Shell oder *SSH* bezeichnet ein kryptographisches Netzwerkprotokoll für den sicheren Betrieb von Netzwerkdiensten über ungesicherte Netzwerke.

Häufig wird es verwendet, um lokal eine entfernte Kommandozeile verfügbar zu machen, d. h., auf einer lokalen Konsole werden die Ausgaben der entfernten Konsole ausgegeben, und die lokalen Tastatureingaben werden an den entfernten Rechner gesendet. Genutzt werden kann dies z. B. zur Fernwartung eines in einem entfernten Rechenzentrum stehenden Servers.

SSH verwendet die Client–Server-Architektur. Eine SSH-Client-Anwendung verbindet sich also mit einem SSH-Server.

Anwendungen sind:

- *Secure System Administration* (Sichere Systemverwaltung)
- *Secure Application Tunneling* (Sicheres Tunneln)
- *Secure Remote Command Execution* (Sichere Ausführung von Kommandos)

Bonus: SSH-Tunnel

Ein *SSH-Tunnel* ist ein gesicherter Kanal, der Netzwerk-Protokolle einbetten und verschlüsselt übertragen kann. Der Tunnel führt dabei in der Regel von einem Rechner innerhalb eines unsicheren Netzwerks zu einem Server/Netz des Vertrauens.

Durch diese Form der Portweiterleitung können TCP-Protokolle wie HTTP (Webseiten) oder SMTP (E-Mail) durch ein fremdes Netz (meist das Internet) hindurch gesichert verwendet bzw. überhaupt erst zugänglich gemacht werden.

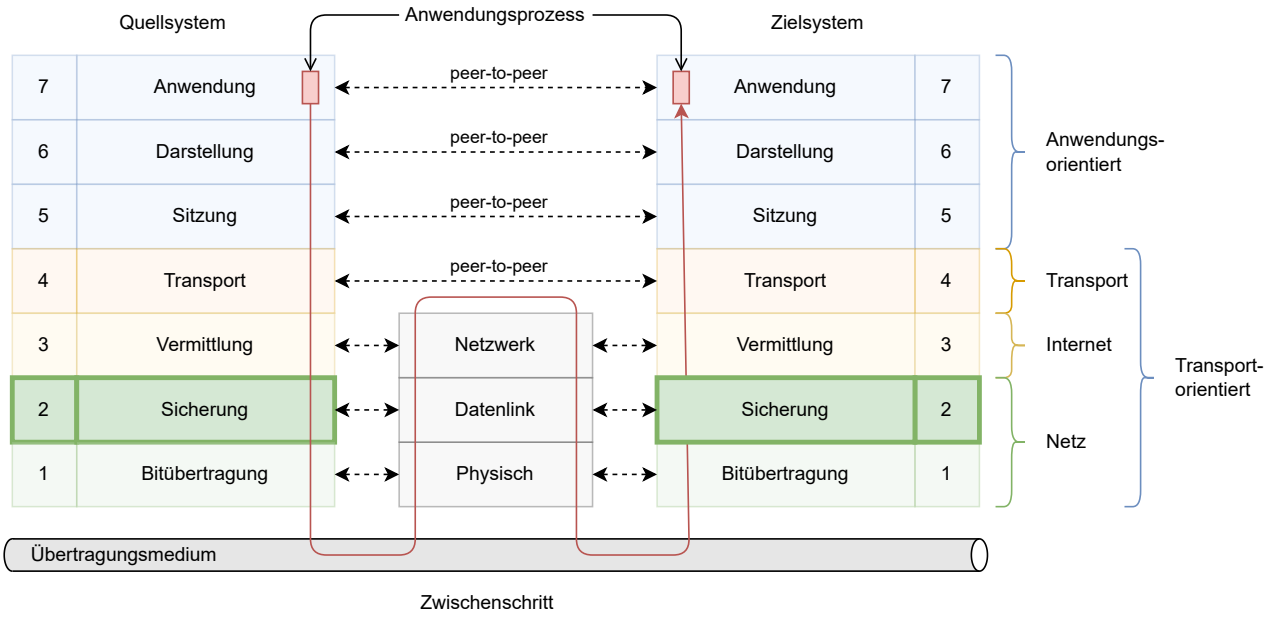
Der Aufbaus des Tunnels erfolgt nach folgenden Muster:

```
ssh benutzer@FernerHost -L <lokaler Port>:<Ziel-Host>:<ferner Port>
```

Bonus: SNMP

Das *Simple Network Management Protocol (SNMP)* ist ein Netzwerkprotokoll um Netzwerkelemente (z. B. Router, Server, Switches, Drucker, Computer usw.) von einer zentralen Station aus überwachen und steuern zu können.

16 Sicherungsschicht



16.1 Übertragungsmedien

Definition: Kupferdoppelader (Twisted Pair)

Bei der *Doppelader* handelt es sich um eine Leitung aus zwei Einzeladern (isolierte Drähte), deren Leiterwerkstoff in der Regel aus Kupfer besteht.

Daten werden als Spannungsniveaus übertragen und sind anfällig für Störungen[a].

Die beiden Einzeladern sind entweder miteinander (*Twisted Pair*) oder mit weiteren Adern verdrillt. Die Verdrillung sorgt dafür, dass elektromagnetische Interferenzen reduziert werden.

Unterscheidung nach Kategorie:

- *CAT 3*:
  - Gemeinsame Umhüllung für vier Kupferdoppeladern.
- *CAT 5*:
  - Wie CAT 3, aber mehr Windungen pro cm
- *CAT 6,7*:
  - Die Paare sind zusätzlich einzeln mit Silberfolie umwickelt.

Unterscheidung nach Abschirmung:

- *UTP Kabel* (Unshielded Twisted Pair):
  - Keine Abschirmung des Kabels
- *STP Kabel* (Shielded Twisted Pair):
  - Abschirmung des Kabels, dadurch günstigere Eigenschaften

[a] Geräte oder andere Kabel in der Umgebung, die elektromagnetische Felder ausstrahlen, verfälschen die Darstellung der Bits auf dem Kupferkabel.

Definition: Koaxialkabel

Koaxialkabel sind zweipolige Kabel mit konzentrischem Aufbau. Sie bestehen aus einem Innenleiter (Seele), der in konstantem Abstand von einem hohlzylindrischen Außenleiter umgeben ist. Der Außenleiter schirmt den Innenleiter vor Störstrahlung ab.

Daten werden als Spannungsniveaus übertragen, sind aber im Vergleich zur Kupferdoppelader besser abgeschirmt und damit weniger störanfällig.

Definition: Glasfaser

Eine *Glasfaser* ist eine aus Glas bestehende lange dünne Faser.

Glasfasern werden unter anderem als Lichtwellenleiter in Glasfasernetzen zur optischen Datenübertragung verwendet. Dies hat gegenüber elektrischer Übertragung den Vorteil einer erheblich höheren maximalen Bandbreite.

Es können mehr Informationen pro Zeiteinheit übertragen werden, außerdem ist das übertragene Signal unempfindlich gegenüber elektrischen und magnetischen Störfeldern und in höherem Maße abhörsicher.

Bonus: Dispersion

Für lange Glasfaserkabel gehört die (Polarisationsmoden-)*Dispersion* mit zu den Faktoren, die die maximale Bandbreite der Datenübertragung begrenzen.

Als (Polarisationsmoden-)Dispersion (PMD) wird der Effekt bezeichnet, bei dem sich Licht unterschiedlicher Polarisation verschieden schnell in einem Lichtwellenleiter ausbreitet und demnach unterschiedlich schnell vorwärts kommen.

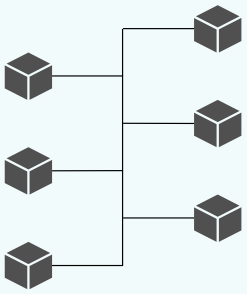
Des Weiteren ist die PMD abhängig von der Verlegung der Glasfaser und verändert ihren Wert, wenn Zug-, Druck- oder Torsionskräfte auf die Glasfaser ausgeübt werden. Ebenfalls können auch Temperaturschwankungen die Glasfaser beeinflussen.

Aufgrund des PMD kann bei der Planung von hochbitratigen Glasfaser-Übertragungsstrecken nicht von der theoretisch möglichen Übertragungskapazität ausgegangen werden. Sondern es müssen sicherheitshalber Reserven berücksichtigt werden. Dies passiert, wenn z. B. die Glasfaser in Erdseilen von Hochspannungsstrecken verlegt werden soll.

16.2 Netztopologien

Definition: Bus

Bei einer *Bus*-Topologie sind alle Geräte direkt mit demselben Übertragungsmedium, dem Bus verbunden. Es gibt keine aktiven Komponenten zwischen den Geräten und dem Medium.

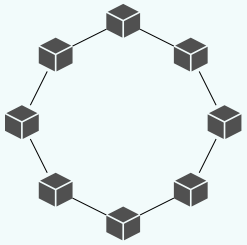


Er ist einfach, preiswert und neue Knoten können einfach angeschlossen werden. Auch ist der Ausfall eines Knotens für die Anderen kein Problem.

Findet eine momentane Datenübertragung zwischen zwei Teilnehmern statt, so müssen die übrigen Teilnehmer zur selben Zeit schweigen, da sie sonst stören würden.

Definition: Ring

Bei der Vernetzung in *Ring*-Topologie werden jeweils zwei Teilnehmer über Zweipunktverbindungen miteinander verbunden, so dass ein geschlossener Ring entsteht.



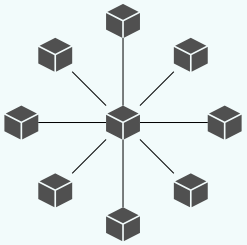
Die zu übertragende Information wird von Teilnehmer zu Teilnehmer weitergeleitet, bis sie ihren Bestimmungsort erreicht. Da jeder Teilnehmer gleichzeitig als Repeater wirken kann (wenn keine Splitter eingesetzt werden), also das Signal wieder verstärkt bzw. auffrischt, können auf diese Art große Entfernungen überbrückt werden.

Wird im Ring nur in eine (Dreh-)Richtung kommuniziert, dann wird bei einem Ausfall von einem der Teilnehmer der Ring unterbrochen.

In einem Ring mit Protection sind alle Leitungen doppelt; dadurch gibt es eigentlich zwei Ringe, ein „Arbeitsweg“ und ein „Ersatzweg“. Die beiden Ringe werden meist in gegensätzlicher „Drehrichtung“ betrieben. Wenn an einer Stelle ein oder beide Ringe unterbrochen wurden, kann immer noch jeder Teilnehmer jeden anderen erreichen.

Definition: Stern

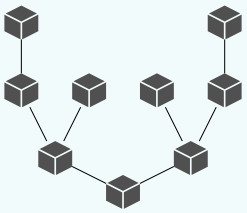
Bei Netzen in *Stern*-Topologie sind an einen zentralen Teilnehmer alle anderen Teilnehmer mit einer Punkt-zu-Punkt-Verbindung angeschlossen. In Computernetzen kann es eine spezialisierte Einrichtung sein, zum Beispiel ein Switch.



In jedem Fall bewirkt eine zentrale Komponente in einem Netz eine höhere Ausfallwahrscheinlichkeit für die einzelnen Verbindungen: ein Ausfall des zentralen Teilnehmers bewirkt unweigerlich den Ausfall aller Verbindungsmöglichkeiten zur gleichen Zeit. Eine geläufige Schutzmaßnahme bei Sternnetzen besteht darin, die zentrale Komponente zu doppeln (Redundanz).

Definition: Baum

Baumtopologien sind dadurch gekennzeichnet, dass sie eine Wurzel (der erste bzw. obere Knoten) haben, von der eine oder mehrere Kanten (Links) ausgehen. Diese führen weiterhin zu einem Blatt (Endknoten) oder rekursiv zu inneren Knoten von Teilbäumen.



Die Baum-Topologie ist nah verwandt mit der Stern-Stern-Topologie, ggf. jedoch mit strengerer hierarchischer Ordnung.

Hierbei müssen Verbindungen zwischen den Verteilern (Hub, Switch) mittels eines Uplinks hergestellt werden.

Häufig wird diese Topologie in großen Gebäuden eingesetzt.

Bei Ausfall eines Verteilers (innerer Knoten) ist der ganze davon ausgehende (Unter)Baum des Verteilers nicht mehr erreichbar

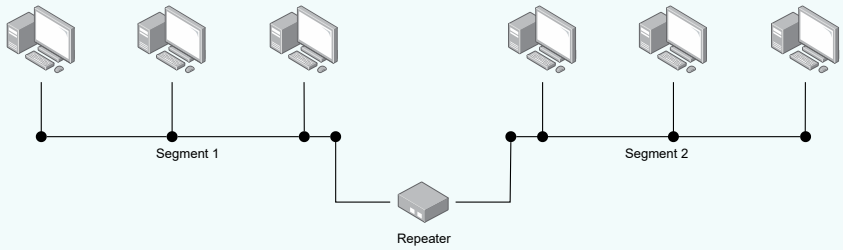
16.3 Netzinfrastruktur

Definition: Repeater

Ein *Repeater* befindet sich in einiger Entfernung zum Sender, empfängt dessen Signale und sendet sie in aufbereiteter Form weiter, wodurch eine größere Distanz überbrückt werden kann.

Beim Einsatz digitaler Übertragungsverfahren kann das Signal zusätzlich vom Repeater dekodiert werden, wodurch Signalstörungen (wie Rauschen oder Verzerrungen der Pulsform) entfernt werden. Anschließend wird das Signal wieder neu kodiert, moduliert und weitergesendet.

In Rechnernetzen sind Repeater Bestandteile der Bitübertragungsschicht (Schicht 1 des OSI-Modells), zur Erweiterung von Netzsegmenten.



Definition: Hub

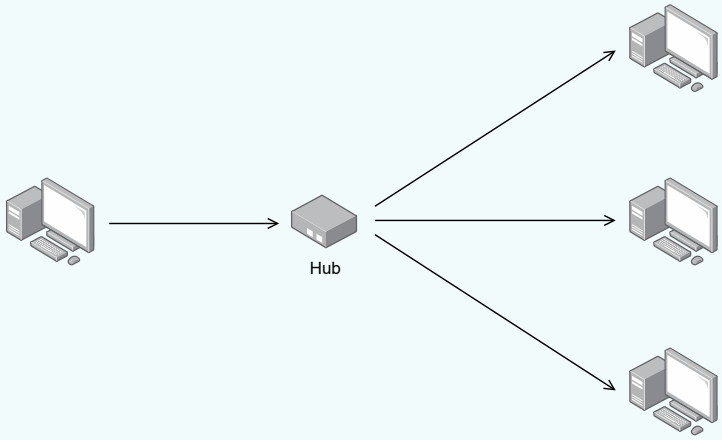
Als *Hub* werden Geräte bezeichnet, die Netzwerkknoten (physisch) sternförmig verbinden.

Das Signal eines Netzteilnehmers wird nicht analysiert, sondern nur die übertragene Bit- bzw. Symbolebene wird regeneriert. Zur Kollisionserkennung trägt ein Hub allerdings meistens bei.

Im Gegensatz zum Switch, der sich zielgerichtet Ports des Empfängers sucht, werden Bits bzw. Symbole an alle anderen Netzteilnehmer weitergeleitet (vergleiche Broadcast).

Aus diesem Grund kann man an jedem Anschluss eines Hubs (im Gegensatz zu denen eines Switches) auch den Datenverkehr zwischen Netzwerkteilnehmern mit Netzwerksniffern analysieren oder mitschneiden.

Ein Hub arbeitet ausschließlich auf Ebene 1 des ISO/OSI-Referenzmodells.



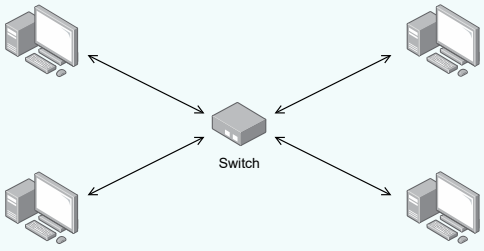
Definition: Switch

Switch bezeichnet ein Kopplungselement in Rechnernetzen, das Netzwerksegmente miteinander verbindet.

Es sorgt innerhalb eines Segments (Broadcast-Domain) dafür, dass die Datenpakete (Frames) an ihr Ziel kommen.

Im Unterschied zu einem auf den ersten Blick sehr ähnlichen Repeater-Hub werden Frames aber nicht einfach an alle anderen Ports weitergeleitet, sondern nur an den, an dem das Zielgerät angeschlossen ist – ein Switch trifft eine Weiterleitungsentscheidung anhand der selbsttätig gelernten Hardware-Adressen der angeschlossenen Geräte.

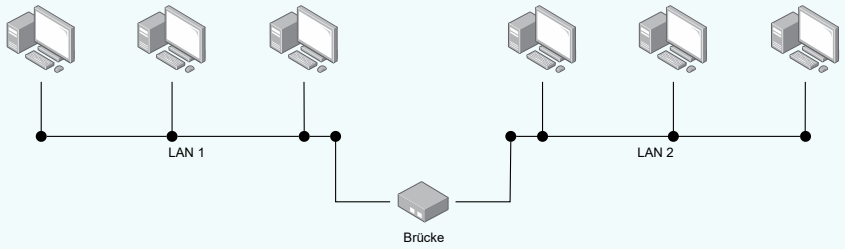
Der Begriff Switch bezieht sich allgemein auf eine *Multiport-Bridge* – ein aktives Netzwerkgerät, das Frames anhand von Informationen aus dem Data Link Layer (Layer 2) des OSI-Modells weiterleitet.



Definition: Bridge

Eine *Bridge* verbindet im Computernetz zwei Segmente auf der Ebene der Schicht 2 (Sicherungsschicht) des OSI-Modells.

Eine Bridge kann auf der Unterschicht MAC oder der Unterschicht LLC arbeiten. Sie wird dann *MAC-Bridge* oder *LLC-Bridge* genannt.



Definition: Router

Router sind Netzwerkgeräte, die Netzwerkpakete zwischen mehreren Rechnernetzen weiterleiten können. Sie werden am häufigsten zur Internetanbindung, zur sicheren Kopplung mehrerer Standorte (Virtual Private Network) oder zur direkten Kopplung mehrerer lokaler Netzwerksegmente, gegebenenfalls mit Anpassung an unterschiedliche Netzwerktechniken (Ethernet, DSL, PPPoE, ISDN, ATM etc.) eingesetzt.

Router treffen ihre Weiterleitungsentscheidung anhand von Informationen aus der Netzwerk-Schicht 3 (für das IP-Protokoll ist das der Netzwerkanteil in der IP-Adresse).

Viele Router übersetzen zudem zwischen privaten und öffentlichen IP-Adressen (Network Address Translation (NAT) bzw. Port Address Translation (PAT)) oder bilden Firewall-Funktionen durch ein Regelwerk ab.

16.4 Dienste der Sicherungsschicht

Definition: Media Access Control

Media Access Control (MAC) ist eine Erweiterung des OSI-Modells.

Dabei wird die Sicherungsschicht (Schicht 2) des OSI-Modells unterteilt in die Unterschichten *Media Access Control* (2a) und *Logical Link Control* (2b), wobei die MAC die untere der beiden ist.

Definition: MAC-Layer

Die Media Access Control (MAC) ist die zweitunterste Schicht im erweiterten OSI-Modell und umfasst Netzwerkprotokolle und Bauteile, die regeln, wie sich mehrere Rechner das gemeinsam genutzte physische Übertragungsmedium teilen.

Sie wird benötigt, weil ein gemeinsames Medium nicht gleichzeitig von mehreren Rechnern verwendet werden kann, ohne dass es zu Datenkollisionen und damit zu Kommunikationsstörungen oder Datenverlust kommt.

Im ursprünglichen OSI-Modell war eine solche Konkurrenz um das Kommunikationsmedium nicht vorgesehen, weshalb die MAC dort nicht enthalten ist.

Definition: LLC-Layer

Logical Link Control (LLC) ist die Bezeichnung für ein Netzprotokoll in Schicht 2b des ISO/OSI-Modells.

Ziel des Protokolls ist die Transparenz unterschiedlicher auf MAC-Ebene eingesetzter Verfahren zur Medienzuteilung.

Daten, welche die OSI-Schicht 3 zur Übermittlung sendet, werden von LLC an den darunterliegenden MAC-Layer weitergegeben. Eingehende Daten werden von der LLC verteilt, indem sie diese an die entsprechenden Instanz-Protokolle der übergeordneten OSI-Schicht 3 weiterleitet.

Bonus: Flusskontrolle

Auch die Sicherungsschicht past die Transmissionsraten den Fähigkeiten des Empfägers an und verhindert so etwaige Pufferüberläufe.

Bonus: Fehlererkennung

Fehler werden durch elektromagnetisches Rauschen und Signaldämpfung verursacht.

Durch den Einsatz redundanter Fehlererkennungsbits kann der Empfänger Fehler erkennen (`NAK` oder Eliminieren des Frames).

Bonus: Fehlerbehebung

Je nach Redundanz der Fehlererkennungsbits können einige Fehler sogar ohne erneute Übertragung korrigiert werden (*Forward Error Correction (FEC)*).

17 Kanalzuteilung, Fehlerkorrektur

17.1 Zugriffsverfahren

Definition: Zugriffsverfahren

Über das *Zugriffsverfahren* wird der Medienzugriff in Netzwerken festgelegt.

Im OSI-Modell wird darunter die Kommunikation zwischen dem Physical Layer (Schicht 1, Bitübertragungsschicht) und dem MAC-Layer (Schicht 2a, Teil der Sicherungsschicht) verstanden.

Über das Zugriffsverfahren wird geregelt, welche Station zu welchem Zeitpunkt welche Datenmenge an wen übertragen darf.

Im LAN ist wichtig festzulegen, wie die beteiligten Stationen und Netzkomponenten auf das Netzwerkkabel zugreifen. Die Regelung des Zugriffs und die damit verknüpfte Übertragung von geeigneten Daten in einem festgelegten Rahmen, gehören zu den Hauptaufgaben des MAC-Layers.

Definition: TDMA

Beim *Zeitmultiplexverfahren* bzw. *Time Division Multiple Access (TDMA)* werden in bestimmten Zeitabschnitten die Daten (Signale) verschiedener Sender auf einem Kanal übertragen.

Definition: FDMA

Das *Frequenzmultiplexverfahren* bzw. *Frequency Division Multiple Access (FDMA)* ist ein nachrichtentechnisches Multiplexverfahren, mit dem gleichzeitig mehrere Signale auf mehrere Träger verteilt übertragen werden können.

Die Träger sind mehreren unterschiedlichen Frequenzen zugeordnet.

Definition: CDMA

Das *Codemultiplexverfahren* bzw. *Code Division Multiple Access (CDMA)* ist ein Multiplexverfahren, das die gleichzeitige Übertragung verschiedener Nutzdatenströme auf einem gemeinsamen Frequenzbereich ermöglicht. Der gemeinsam genutzte Frequenzbereich weist dabei als wesentliche Eigenschaft eine größere Bandbreite auf, als der Nutzdatenstrom belegt.

Zur Unterscheidung werden die Datenströme mit speziellen „Spreizcodes“ codiert, wobei diese Codefolgen zusätzlich bestimmte Eigenschaften wie Orthogonalität aufweisen und in bestimmten Anwendungen auf Pseudozufall basieren, wodurch auf Empfängerseite durch Korrelation mit der Spreizcodefolge die ursprünglichen Nutzdatenströme voneinander getrennt gewonnen werden können.

Gegenüber den klassischen Multiplexverfahren wie dem Frequenzmultiplex und dem Zeitmultiplex erfolgt bei dem Codemultiplex eine Überlagerung der einzelnen Datenströme sowohl im Frequenzbereich als auch im Zeitbereich.

Definition: CSMA

Carrier Sense Multiple Access (CSMA) bezeichnet ein dezentrales, asynchrones Verfahren zum Erlangen des Zugriffsrechts nach dem Konkurrenzverfahren auf Busleitungen.

Trägerprüfung bzw. Carrier Sense bedeutet, dass alle Teilnehmer den Status der Busleitung beobachten und ihre Nachrichten nur senden, wenn gerade kein anderer Teilnehmer sendet, der Kanal also frei ist.

Ist das Medium für eine bestimmte Zeitspanne[a] nicht belegt, wird es als frei betrachtet.

Kollisionen können eintreten, wenn zwei oder mehr Teilnehmer gleichzeitig mit dem Senden beginnen.

[a] 9,6 μs bei 10-Mbps-Ethernet, 960 ns bei 100-Mbps-Fast-Ethernet und 96 ns bei 1000 Mbps

Definition: CSMA/CD

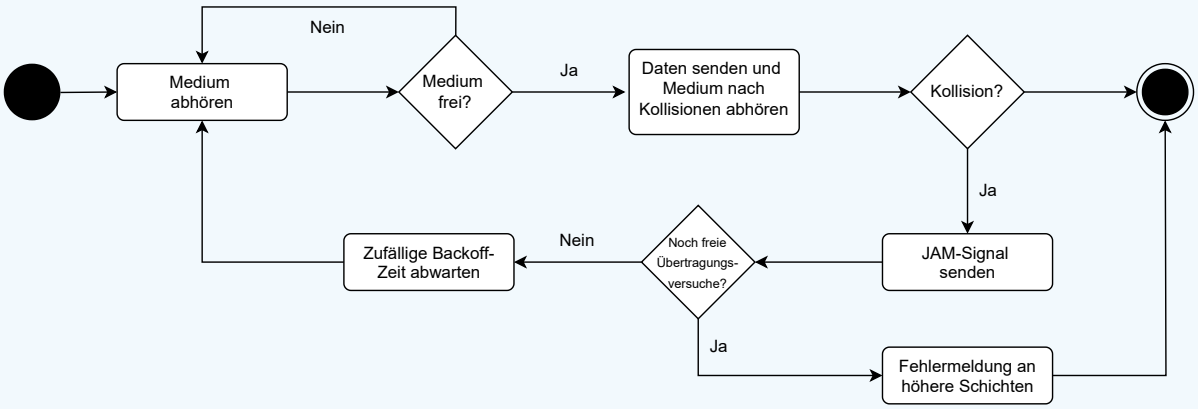
Carrier Sense Multiple Access/Collision Detection (CSMA/CD) ist eine Erweiterung von CSMA. Verwendung findet CSMA/CD z.B. bei Ethernet.[a]

Wenn ein Gerät Daten senden möchte, hält es sich an folgenden Ablauf:

1. *Horchen*: Zuerst muss das Medium überwacht werden, ob es belegt ist.
   - Frei: Wenn das Medium eine bestimmte Zeit lang frei ist, weiter mit Schritt 2.
   - Belegt: Weiter mit Schritt 1.
2. *Senden*: Informationsübertragung, zugleich wird das Medium fortwährend weiter abgehört.
   - Erfolg (keine Kollision bis Übertragungsende): Übertragung ist erfolgreich abgeschlossen und es wird eine Erfolgsmeldung an höhere Netzwerkschichten gemeldet; weiter mit Schritt 5.
   - Kollision: Wird eine Kollision entdeckt, beende die Datenübertragung und sende ein kurzes, definiertes Störsignal (*jam*) auf die Leitung, um sicherzustellen, dass alle anderen Transceiver die Kollision ebenfalls erkennen, dann weiter mit Schritt 3.
3. *Leitung ist belegt*: Überprüfung der Anzahl der Übertragungsversuche:
   - Maximum nicht erreicht: Eine zufällige Zeit (*Backoff*) abwarten, dann wieder bei Schritt 1 beginnen.
   - Maximum erreicht: Weiter mit Schritt 4.
4. *Fehler*: Maximale Anzahl von Übertragungsversuchen wurde überschritten. Ein Fehler wird an die höheren Netzwerkschichten gemeldet, weiter mit Schritt 5.
5. *Ende*: Übertragungsmodus verlassen

[a] Bei Wireless LANs wird ein deutlich anderer Mechanismus namens Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA) (siehe 17.1) benutzt.

Diagramm: CSMA/CD



17.2 Ethernet

Definition: Ethernet

Ethernet ist eine Technik, die Software (Protokolle usw.) und Hardware (Kabel, Verteiler, Netzwerkkarten usw.) für kabelgebundene Datennetze spezifiziert, welche ursprünglich für lokale Datennetze (LANs) gedacht war.

Sie ermöglicht den Datenaustausch in Form von Datenframes zwischen den in einem lokalen Netz (LAN) angeschlossenen Geräten.

Im OSI-Modell ist mit Ethernet sowohl die physische Schicht (OSI Layer 1) als auch die Data-Link-Schicht (OSI Layer 2) festgelegt.

Ethernet verwendet CSMA/CD. Bei Kollisionen wird Binary Exponential Backoff verwendet.

Definition: Binary Exponential Backoff

Der *Binary Exponential Backoff* ist ein Stauauflösungsmechanismus im Ethernet.

Wird von Stationen im Ethernet eine Kollision erkannt, beenden diese Stationen ihre Sendung und versuchen sofort oder nach einer Slot-Time von 51.2 μs[a] erneut ihre Sendung über das Ethernet zu übertragen. Dabei kann es erneut zu einer Kollision kommen, wenn beide Stationen zufällig die gleiche Wahl treffen.

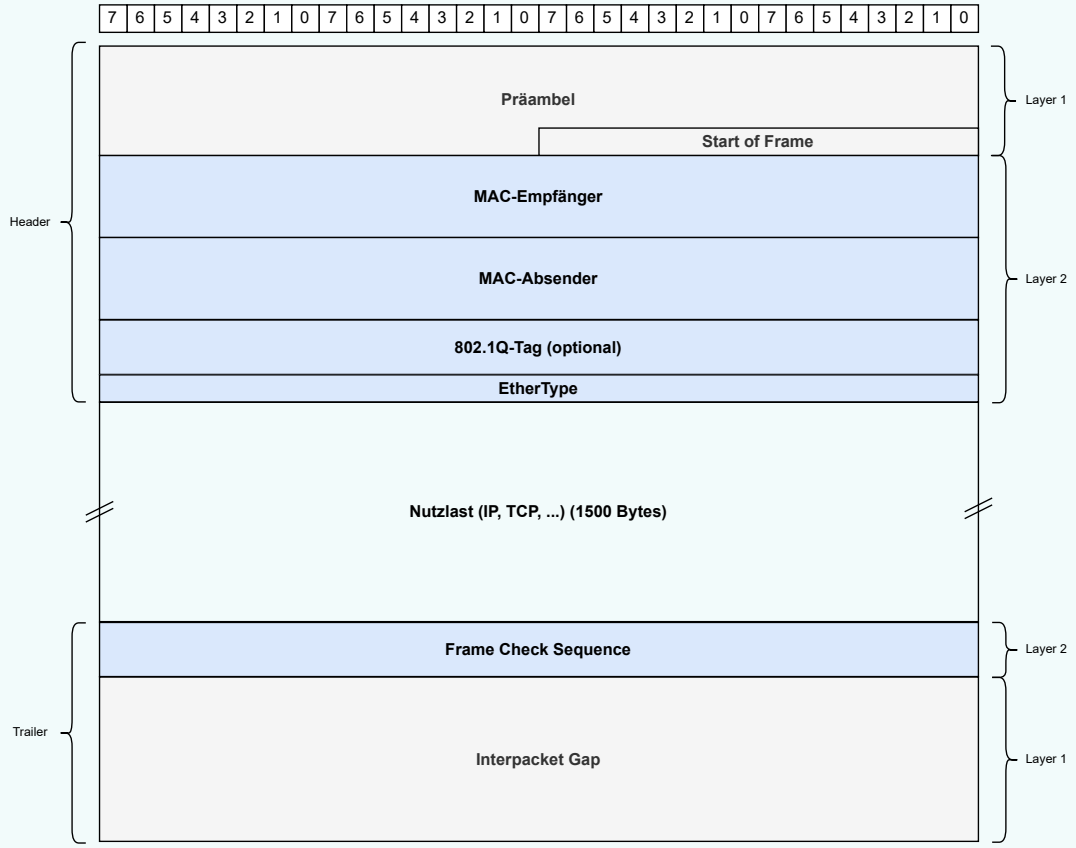
Beim nächsten Versuch wird nun jede der beiden Stationen wieder per Zufallsentscheidung einen neuen Starttermin auswählen, diesmal aber aus vier Möglichkeiten: 0, 1, 2 oder 3 Slot-Times, also 2^2.

Bei einer erneuten Kollision sind es dann $2^3 = 8$ Möglichkeiten, dann 16, 32, 64, 128, 256, 512 und schließlich 1024 (2^{10}), was auch die Maximalgrenze der Möglichkeiten darstellt.

Nach insgesamt 16 erfolglosen Übertragungsversuchen mit Kollision wird mit einer Fehlermeldung des Ethernet-Controllers abgebrochen.

[a] entspricht 512 Bit, gilt nur für 10/100 MBit/s Ethernet, 4,096 μs und 4096 Bit bei 1 GBit/s

Definition: Ethernet-Frame



Bonus: Token-Ring

Token-Ring ist eine Vernetzungstechnik für Computernetzwerke.

Die Token-Ring-Technik wurde praktisch vollständig von den verschiedenen Ethernet-Varianten verdrängt.

Ein Token kreist bei Token-Ring-Netzen über den Ring: Das Token wird stets von einem Knoten an den nächsten weitergereicht.[a]

Möchte nun ein Computer Daten versenden, wartet er, bis das Token ihn erreicht hat, dann hängt er seine Nutzdaten daran an. Zugleich ergänzt er das Token um Steuersignale und setzt außerdem das Token-Bit von 0 (für „freies Token“) auf 1, aus dem Frei-Token wird also ein Datenrahmen.

Jeder Rechner prüft, ob das Paket an ihn adressiert ist, und setzt es anderenfalls zurück auf den Ring. Erhält der vorgesehene Empfänger den an ihn adressierten Datenrahmen, kopiert er die Nutzdaten und quittiert den Datenempfang.

Der Sender erhält die Quittung und sendet das Token mit den nächsten Nutzdaten oder setzt ein Frei-Token auf den Ring.

[a] Selbst im Leerlauf geben die Stationen das Paket fortwährend weiter.

17.3 Fehlererkennung

Definition: Hamming-Abstand

Der *Hamming-Abstand* bzw. *Hamming-Distanz* sind Maße für die Unterschiedlichkeit von Zeichenketten.

Der Hamming-Abstand zweier Blöcke mit fester Länge (sogenannter Codewörter) ist dabei die Anzahl der unterschiedlichen Stellen.

Es wird zur Fehlererkennung und zur Fehlerkorrektur benutzt, indem Dateneinheiten, die über eine Übertragungsstrecke empfangen werden, mit gültigen Zeichen verglichen werden.

Eine etwaige Korrektur der Zeichen erfolgt nach dem Wahrscheinlichkeitsprinzip. Ob eine Fehlererkennung oder -korrektur stattfinden kann, hängt von der Hamming-Distanz ab.

Definition: Wiederholung Hamming-Codierung

Bei der *Hamming-Codierung* werden Paritätsinformationen zu Daten hinzugefügt, um so mögliche Übertragungsfehler zu erkennen.

Hamming-Codewörter haben die Länge $N = 2^k - 1$, wobei k Paritätsbits enthalten sind. Die Bits werden der Einfachheit halber bei Eins beginnend durchnummeriert.

Die Paritätsbits stehen an den Stellen, deren Index eine 2er-Potenz ist.[a]

Sind $p_1, p_2, \ldots, p_k$ Paritätsbits, $d_1, d_2, \ldots, d_{N-k}$ Bits des Datenwortes und $c_1, c_2, \ldots, c_N$ die Bits des zu bildenden Codewortes, hat ein Codewort des so konstruierten Hamming-Codes die folgende Form:

c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}	c_{16}	c_{17}	$\ldots$
p_0	p_1	d_1	p_2	d_2	d_3	d_4	p_3	d_5	d_6	d_7	d_8	d_9	d_{10}	d_{11}	p_4	d_{12}	$\ldots$

Dabei wird jedem Paritätsbit eine spezielle Bitmaske zugewiesen:

Bitmaske	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}	$\ldots$
p_0	p_0	p_1	1	p_2	1	0	1	p_3	1	0	1	0	1	0	1	$\ldots$
p_1	p_0	p_1	1	p_2	0	1	1	p_3	0	1	1	0	0	1	1	$\ldots$
p_2	p_0	p_1	0	p_2	1	1	1	p_3	0	0	0	1	1	1	1	$\ldots$
p_3	p_0	p_1	0	p_2	0	0	0	p_3	1	1	1	1	1	1	1	$\ldots$

Damit gilt:

$$c_1 = p_0 = c_3 \oplus c_5 \oplus c_7 \oplus c_9 \oplus c_{11} \oplus c_{13} \oplus c_{15} \oplus \ldots$$
$$\iff \text{jedes ungerade Datenbit}$$
$$c_2 = p_1 = c_3 \oplus c_6 \oplus c_7 \oplus c_{10} \oplus c_{11} \oplus c_{14} \oplus c_{15} \oplus \ldots$$
$$\iff \text{ein Datenbit rechts von } p_1\text{, zwei überspringen, zwei einberechnen, } \ldots$$
$$c_4 = p_2 = c_5 \oplus c_6 \oplus c_7 \oplus c_{12} \oplus c_{13} \oplus c_{14} \oplus c_{15} \oplus \ldots$$
$$\iff \text{drei Datenbit rechts von } p_2\text{, vier überspringen, vier einberechnen, } \ldots$$

[a] Sehr empfehlenswert: 3Blue1Brown: How to send a self-correcting message (Hamming codes)

Algorithmus: Wiederholung Fehlererkennung beim Hamming-Code

Situation: Empfangen eines Hamming-codierten Datensatzes

1. Erneutes Berechnen der *Paritätsbits*
2. Erkennen, welche neu berechneten Paritätsbits p'_i *verschieden* sind zu den empfangenen Paritätsbits p_i
3. Bitfehler ist in dem Datenbit passiert, in dem gilt:
   $p_i = p'_i \implies$ Bitmaske für p_i ist 0
   $p_i \neq p'_i \implies$ Bitmaske für p_i ist 1

Es wird angenommen, dass nur ein Bitfehler in den Datenbits passiert ist.

Anmerkung: Ist lediglich ein Paritätsbit verschieden, dann ist ein Übertragungsfehler in dem betreffenden Paritätsbit selbst aufgetreten, da alle Datenbits zur Berechnung von mindestens zwei Paritätsbits verwendet werden.

Definition: CRC

Die *zyklische Redundanzprüfung* (*Cyclic Redundancy Check (CRC)*) ist ein Verfahren zur Bestimmung eines Prüfwerts für Daten, um Fehler bei der Übertragung oder Speicherung erkennen zu können. Im Idealfall kann das Verfahren sogar die empfangenen Daten selbständig korrigieren, um eine erneute Übertragung zu vermeiden.

Die Bitfolge der Coderepräsentation der Daten wird durch ein vorher festzulegendes Generatorpolynom (das CRC-Polynom) Modulo 2 geteilt, wobei ein Rest bleibt (CRC-Wert).

Um zu verifizieren, dass die Daten keinen Fehler beinhalten, wird der empfangene Datenblock mit angehängtem CRC-Wert als Binärfolge interpretiert, erneut durch das CRC-Polynom Modulo geteilt und der Rest ermittelt.

Wenn kein Rest bleibt, ist entweder kein Fehler aufgetreten oder es ist ein (unwahrscheinlicher) Fehler aufgetreten, der in Polynomdarstellung das CRC-Polynom als Faktor hat.

Beispiel: CRC (Senden)

Gegeben seien:

- Generator- bzw. CRC-Polynom `110101` ($x^5 + x^4 + x^2 + x^0$).
- Rahmen: `11011`
- Rahmen mit Anhang:[a] `1101100000`

Nun wird der Rahmen mit Anhang von links her durch das Generatorpolynom dividiert:[b]

$$
\begin{array}{cccccccccc}
 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
\oplus & 1 & 1 & 0 & 1 & 0 & 1 & & & \\
\hline
 & & & & & 1 & 1 & 0 & 0 & 0 \\
 & & & & \oplus & 1 & 1 & 0 & 1 & 0 \\
\hline
 & & & & & 0 & 0 & 0 & 1 & 0
\end{array}
\quad
\begin{array}{c}
1 \\
\oplus \\
\hline
0 \\
\hline
1
\end{array}
\quad (Rest)
$$

An den Rahmen ohne Anhang wird nun der Rest angehängt. Dieser muss ebenfalls aus n Bits bestehen. Damit hängen wir nun `00101` an den Rahmen an.

Übertragener Rahmen: `1101100101`

Diese Nachricht kann jetzt beispielsweise über ein Netzwerk übertragen werden. Wenn die Nachricht beim Empfänger eintrifft, kann dieser überprüfen, ob sie korrekt angekommen ist.

[a] Der zu übertragenden Bitfolge werden n Nullen angehängt, wobei n dem Grad des Polynoms entspricht.
[b] Dabei wird ausschließlich `XOR` verwendet!

Beispiel: CRC (Empfangen)

Gegeben seien:

- Generator- bzw. CRC-Polynom `110101` ($x^5 + x^4 + x^2 + x^0$).
- Übertragene Nachricht: `1001100101`

Wir wollen die Nachricht auf Korrektheit prüfen. Es gilt:

$$
\begin{array}{cccccccccc}
 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\
\oplus & 1 & 1 & 0 & 1 & 0 & 1 & & & & \\
\hline
 & & 1 & 0 & 0 & 1 & 1 & 0 & & & \\
 & \oplus & 1 & 1 & 0 & 1 & 0 & 1 & & & \\
\hline
 & & & 1 & 0 & 0 & 1 & 1 & 1 & & \\
 & & \oplus & 1 & 1 & 0 & 1 & 0 & 1 & & \\
\hline
 & & & & 1 & 0 & 0 & 1 & 0 & 0 & \\
 & & & \oplus & 1 & 1 & 0 & 1 & 0 & 1 & \\
\hline
 & & & & & 1 & 0 & 0 & 0 & 1 & 1 \\
 & & & & \oplus & 1 & 1 & 0 & 1 & 0 & 1 \\
\hline
 & & & & & & 1 & 0 & 1 & 1 & 0
\end{array}
\quad (Rest)
$$

Der Rest der Division (`10110`) ist ungleich null. Also ist ein Fehler aufgetreten.

18 Leitungscodes, WLAN

Definition: Basisband

In der Nachrichtentechnik ist das *Basisband* der natürliche Frequenzbereich des Nutzsignals[a].

Der belegte Frequenzbereich wird mit dem Wert der Bandbreite B ausgedrückt.

Die digitalen Informationne werden direkt in physikalische Größen übersetzt und so über die Leitung übertragen.

Hierzu sind Kodierungsverfahren notwendig, die festlegen, wie bei der Übertragung eine 0 bzw. eine 1 repräsentiert werden.

Es kann immer nur je ein Signal übertragen werden.

[a] untere Grenzfrequenz $f_{\min} \approx 0$ Hz

Definition: Breitband

Die digitalen Nutzdaten werden beim *Breitband* nicht direkt übertragen, sonderen einem oder mehreren hochfrequenten Trägern „aufmoduliert“.

Durch die Verwendung verschiedener Trägerwellen (Frequenzen) können dann mehrere Informationen gleichzeitig übermittelt werden.

18.1 Leitungscodes

Definition: Leitungscode

Der *Leitungscode* legt bei der digitalen Telekommunikation fest, wie die zur Informationsübertragung genutzten Symbole auf der physischen Ebene übertragen werden.

Dabei werden bestimmte Pegelfolgen, etwa Lichtintensitäten auf Glasfasern oder Spannungen oder Ströme auf elektrischen Leitungen, binären Bitsequenzen im Datenstrom zugeordnet.

Anforderungen an einen Leitungscode sind:

- *Effizienz*: möglichst hohe Übertragungsraten durch Codewörter
- *Taktrückgewinnung* beim Empfänger (*Synchronisation*): möglichst häufige bzw. regelmäßige Pegelwechsel
- *Gleichstromfreiheit*: positive und negative Signale treten ungefähr gleich oft auf
- *Robustheit*: Erkennen von längeren Sequenzen von 0 bzw. 1 und von fehlerhaften Bits

Bonus: Baudrate vs. Bitrate

Wenn die Zeitdauer (Schrittdauer) eines Symbols bzw. Codeelements T ist, ist die Schrittgeschwindigkeit bzw. *Baudrate*:

$$v_s = \frac{1}{T} \text{ [Baud]}$$

Die dazugehörige Übertragungsgeschwindigkeit bzw. *Bitrate* ist dann:[a]

$$v_u = v_s \log_2 n$$

Bei binären Codeelementen stimmen somit Bitrate und Baudrate überein, falls nur Codeelementen für Daten übermittelt werden.

[a] n = Anzahl diskreter Zustände des Codeelements

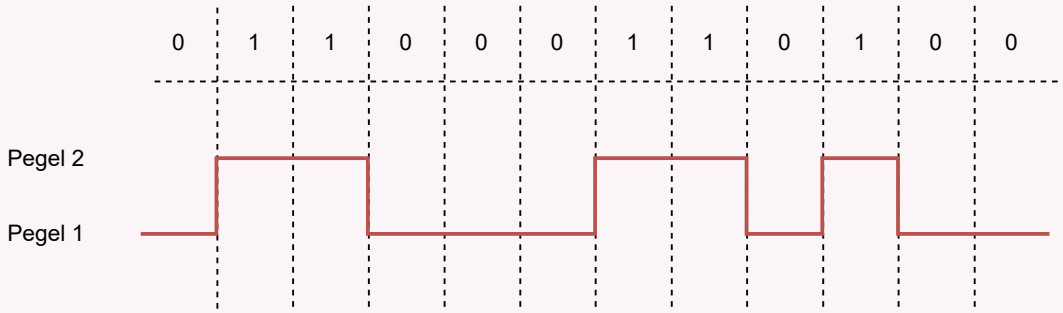
Definition: NRZ-Code

Der *NRZ-Code (Non Return to Zero)* ordnet direkt jedem Bit-Wert einen Leitungszustand zu. Er kann ohne weiteres verwendet werden, wenn in den Nutzdaten keine langen konstanten Folgen auftreten, wie etwa bei ASCII-kodierten Texten.

Von Nachteil ist, dass der Empfänger bei der Übertragung einer längeren Folge gleicher Symbole unsicher wird über die Länge der Folge.

Die NRZ-Kodierung ist im Allgemeinen auch nicht gleichanteilsfrei und damit insbesondere bei magnetischer Datenaufzeichnung problematisch.

Beispiel: NRZ-Code



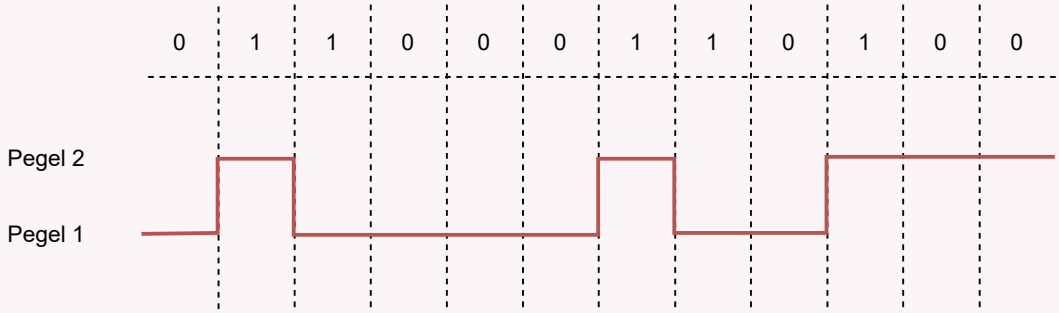
Definition: NRZ-I-Code

Die *NRZI-Kodierung (Non Return to Zero Inverted)* ordnet einem der beiden Bit-Werte den bereits anliegenden Leitungszustand zu, dem anderen Bit-Wert einen Zustandswechsel (Inversion).

Daraus ergibt sich unmittelbar die Polaritätsfreiheit: Ein Verpolen der Übertragungsleitung ändert nicht die Bitfolge.

Beispiel: NRZ-I-Code

Im folgenden Beispiel bewirken Einsen einen Zustandswechsel.



Definition: RZ-Code

Beim *Return-to-Zero-Code (RZ-Code)* handelt es sich um einen Leitungscode, mit dem es möglich ist, Binärzahlen über ein Medium zu übertragen, indem der Sender dessen Zustand zwischen drei Pegelwerten (Sendesymbole, meist als +1, 0 und -1 bezeichnet) wechseln lässt.

Dadurch gibt es beim Übertragen eines Bits garantiert eine Pegeländerung, welche der Empfänger zur Taktrückgewinnung (Synchronisierung) nutzen kann.

Nachteilig gegenüber dem NRZ-Code ist, dass eine doppelt so große Bandbreite benötigt wird.

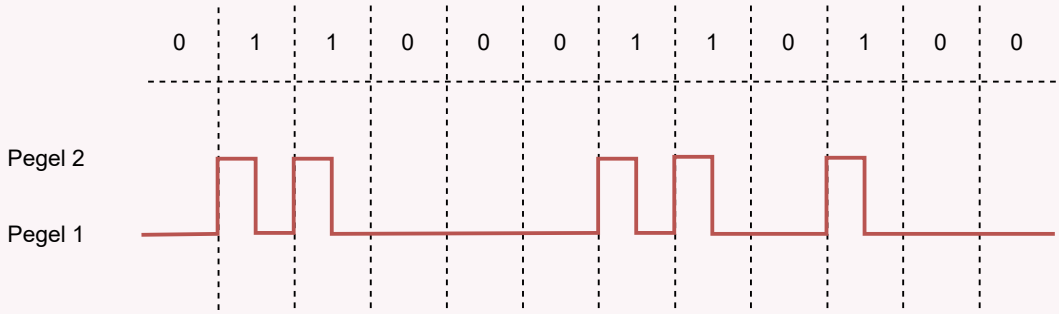
Bonus: RZ-Code (unipolar)

Eine Sonderform stellt die *unipolare RZ-Kodierung* dar.

Der Vorteil besteht darin, dass nur zwei Pegelwerte (+1 und 0) als Symbole benötigt werden und diese Codierung daher mit herkömmlichen Digitalschaltungen leicht realisiert werden kann.

Der Nachteil besteht darin, dass bei der Übertragung einer langen logisch-0-Folge, welche mit konstantem Pegel 0 codiert wird, keine Signaländerung erfolgt und damit eine Synchronisierung seitens des Empfängers unmöglich ist.

Beispiel: RZ-Code (unipolar)



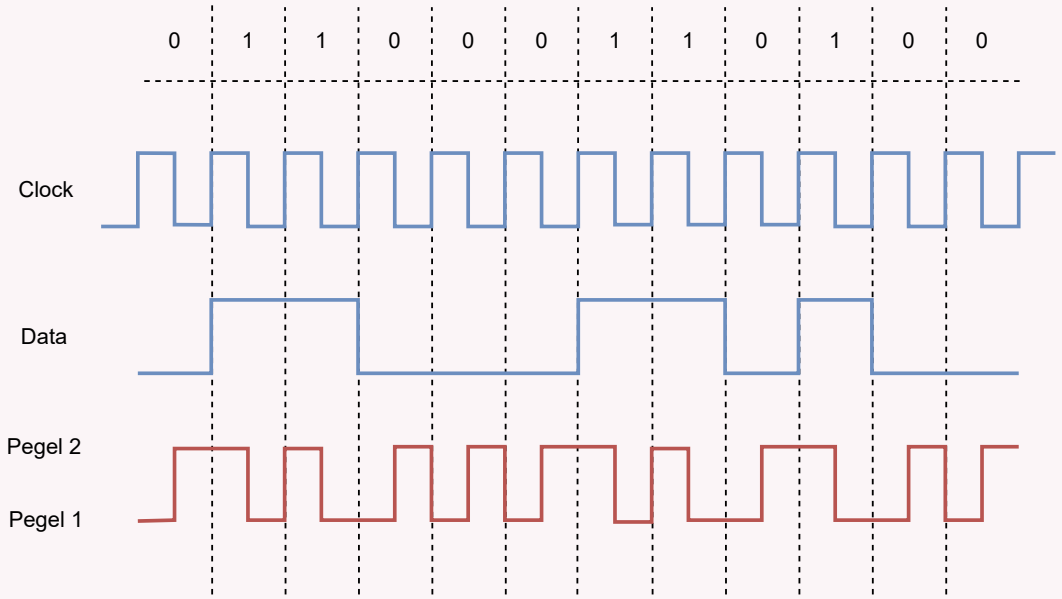
Definition: Manchester-Code

Der *Manchester-Code* ist ein Leitungscode, der bei der Kodierung das Taktsignal erhält.

Es gibt für den Manchester-Code zwei mögliche und gleichwertige Definitionen:

- *G.E. Thomas*: eine fallende Flanke eine logische Eins, eine steigende Flanke eine logische Null.
- *IEEE 802.3*: eine fallende Flanke eine logische Null und eine steigende Flanke eine logische Eins.

Beispiel: Manchester-Code (G.E. Thomas)



Definition: Differentieller Manchester-Code

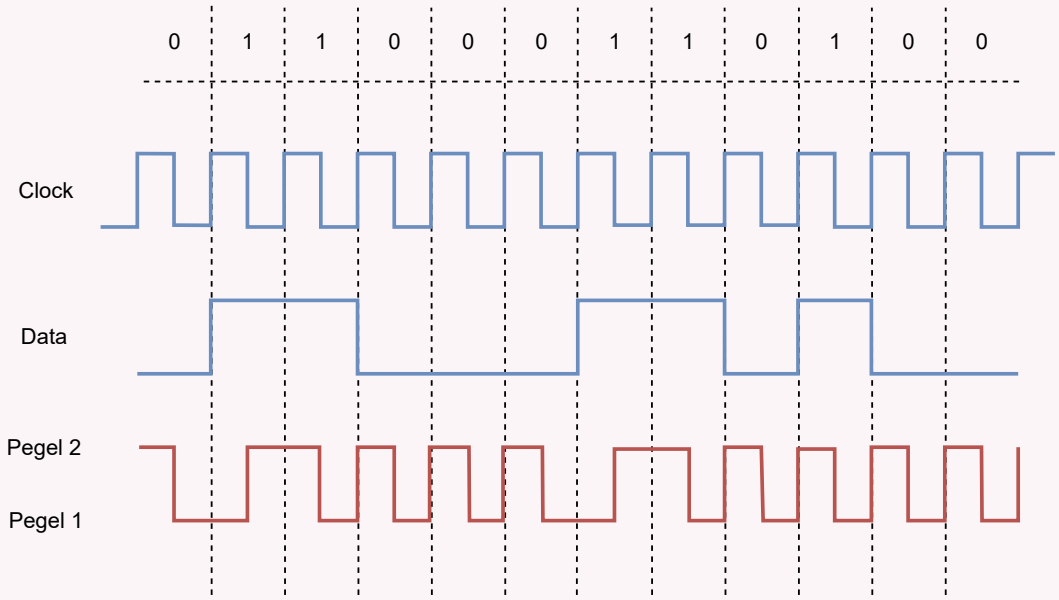
Der *differentielle Manchester-Code* ist ein Leitungscode zur Übertragung von Bitfolgen als Digitalsignal.

Wie beim Manchester-Code enthält jede Bitperiode mindestens eine Flanke, zwecks einfacher und robuster Taktrückgewinnung. Zusätzlich ist es invariant gegen Verpolung.

Zusätzlich gilt:

- Eine Null bewirkt einen Pegelwechsel am Anfang des Symbols.
- Eine Eins bewirkt *keinen* Pegelwechsel am Anfang des Symbols.

Beispiel: Differentieller Manchester-Code



Definition: 4B/5B-Code

Der *4B5B-Code* beschreibt einen Leitungscode, der eindeutig umkehrbar vier Nutzdatenbits auf fünf Codebits abbildet.

Durch das Einfügen eines weiteren Bits erhöht sich die codierte Bitrate gegenüber der Nutzdatenbitrate um 25%.

18.2 Breitbandkommunikation

Definition: Breitbandkommunikation

Bei der *Breitbandkommunikation* wird das digitale Signal auf ein analoges Trägersignal aufmoduliert.

Daten werden physikalisch als elektromagnetische Wellen dargestellt.

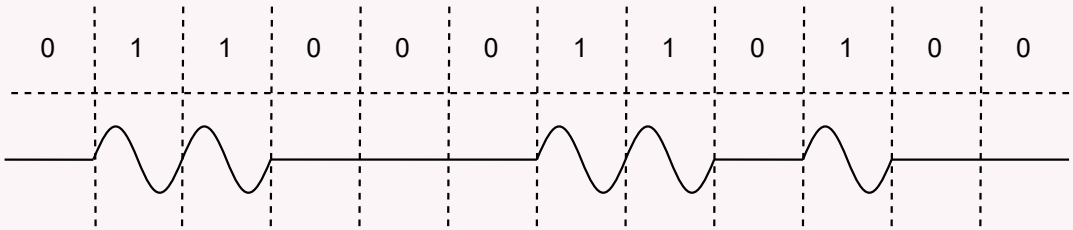
Definition: Amplitudenmodulation

Bei der *Amplitudenmodulation (Amplitude Shift Keying, ASK)* wird

- bei einer 1 eine große Amplitude ausgestrahlt,
- bei einer 0 eine kleine.

Die Amplitudenmodulation ist störanfällig wegen der Abschwächung der Signalstärke.

Beispiel: Amplitudenmodulation

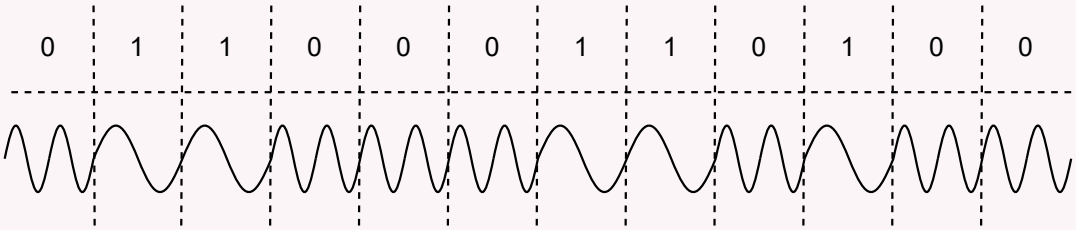


Definition: Frequenzmodulation

Bei der *Frequenzmodulation (Frequence Shift Keying, FSK)* werden für 0 und 1 verschiedene Frequenzen ausgestrahlt.

Das Problem ist hier ein verschwenderischer Umgang mit Frequenzen.

Beispiel: Frequenzmodulation

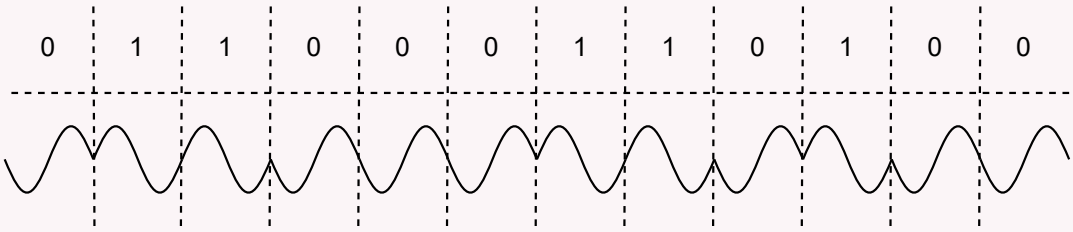


Definition: Phasenmodulation

Bei der *Phasenmodulation (Phase Shift Keying, PSK)* werden für 0 und 1 verschiedene Phasenlagen verwendet.

Das Problem ist komplexe Demodulation, allerdings ist Phasenmodulation störsicher.

Beispiel: Phasenmodulation



Definition: Nyquist-Shannon-Abtasttheorem

Das *Nyquist-Shannon-Abtasttheorem* besagt, dass ein auf $f_{\max}$ bandbegrenztes Signal aus einer Folge von äquidistanten Abtastwerten exakt rekonstruiert werden kann, wenn es mit einer Frequenz von größer als $2 \cdot f_{\max}$ abgetastet wurde.

18.3 WLAN

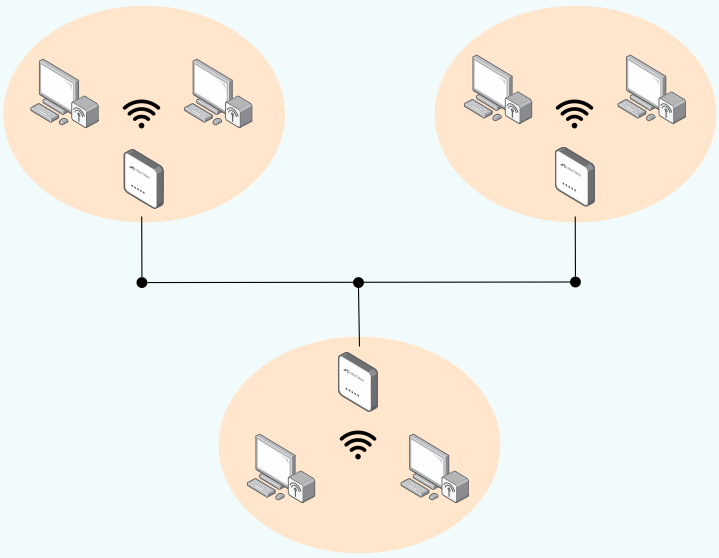
Definition: WLAN

Wireless Local Area Network (WLAN) bezeichnet ein lokales Funknetz, wobei meist ein Standard der IEEE-802.11-Familie gemeint ist.

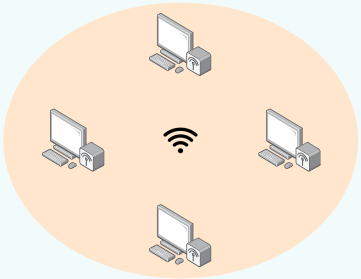
Definition: WLAN-Netze

WLANs können in verschiedenen Modi betrieben werden:

- *Infrastruktur-Modus*:
  - ähnelt im Aufbau dem Mobilfunknetz
  - Wireless Access Point übernimmt Koordination aller Clients



- *Ad-hoc-Modus*:
  - keine Station besonders ausgezeichnet, sondern alle sind gleichwertig
  - lassen sich schnell und ohne großen Aufwand aufbauen
  - für die spontane Vernetzung weniger Endgeräte sind allerdings andere Techniken[a] eher gebräuchlich.

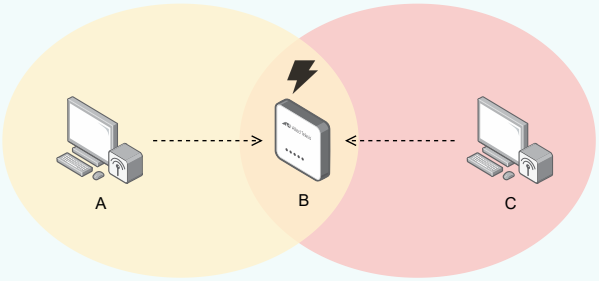


[a] wie Bluetooth

Definition: Hidden-Station-Problem

Eine *Hidden Station* bezeichnet in asynchronen und nicht zentral koordinierten Kommunikationsnetzen, Funknetzen oder Rechnernetzen den unerwünschten Umstand, dass bei einer Übertragung zwischen zwei Teilnehmern (A und B) ein weiterer potentieller Sender (C, das Hidden Terminal) in der Nähe des Empfängers (B) ist, der vom eigentlichen Sender (A) nicht gesehen werden kann.

Dieser potentielle Sender (C) kann die Kommunikation der anderen beiden Knoten (A und B) stören, indem er ebenfalls eine Nachricht an den Knoten in der Mitte (B) sendet, dies kann zu einer Kollision an dem Empfänger (B) führen.



Definition: Exposed-Station-Problem

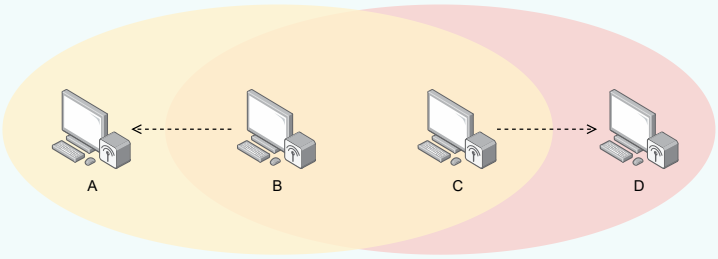
Unter einer *Exposed Station* versteht man, wenn eine Station B an A sendet und nun C an irgendeine andere Station senden möchte, die nicht im Sendebereich von B liegt.

C erkennt die Signale von B und wartet, bis die Übertragung zwischen B und A vorbei ist.

Da die Funkwellen von C aber Station A gar nicht erreichen können, wäre es gar nicht nötig zu warten: bei A könnte gar kein Konflikt auftreten.

Dennoch ist C von der Sendung der anderen beiden Stationen abhängig (ausgeliefert).

Durch das unnötige Warten wird Bandbreite verschwendet.



Definition: CSMA/CA

Der Begriff *Carrier Sense Multiple Access/Collision Avoidance (CSMA/CA)* bezeichnet ein Prinzip für die Kollisionsvermeidung bei Zugriff mehrerer Netzwerkstationen auf denselben Übertragungskanal. Es wird häufig unter anderem bei drahtlosen Netzwerken (Wireless LANs) eingesetzt.

Möchte ein Gerät Daten nach dem CSMA/CA-Verfahren versenden, so ist unter anderem folgender Ablauf möglich:

1. Zuerst wird das Medium abgehört (*Carrier Sense*)
2. Ist das Medium für die Dauer eines DIFS[a] frei, wird eine Backoffzeit aus dem Contention Window ausgewürfelt und nach Ablauf dieser gesendet.
3. Ist das Medium belegt, wird der Backoff bis zum Ablauf des *Network Allocation Vectors (NAV)* gestoppt, bevor er nach einem weiteren DIFS entsprechend weiter läuft.
4. Nach vollständigem Empfang des Paketes wartet der Empfänger ein SIFS[b], bevor das ACK gesendet wird.
5. Eine Kollision durch gleichzeitigen Ablauf des Backoffs führt zu einem ACK-Timeout – nach welchem ein EIFS[c] gewartet wird, bevor sich der gesamte Vorgang wiederholen kann

[a] *DIFS (Distributed Coordination Function Interframe Spacing)*: Die Zeit, die vor dem Senden eines regulären Datenframes vergangen sein muss.

[b] *SIFS (Short Interframe Spacing)*: Die Zeit, die vergangen sein muss vor dem Senden eines Bestätigungsframes (ACKs), eines *Clear to Send*-Pakets (CTSs)

[c] *EIFS (Extended Interframe Spacing)*: Die Zeit, die vor dem Senden nach einer erkannten Kollision vergangen sein muss.

Index

Beispiele