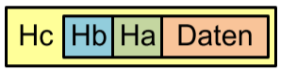
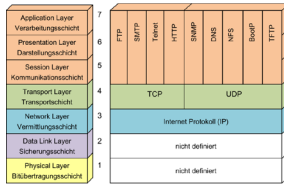
Das Open System Interconnection (kurz: OSI) Referenzmodell

Vorgang: Datenübertragung in einem Schichten-Modell. Jede Schicht packt das bestehende Paket wieder in ein neues Paket und sendet es an die nächst tiefere Schicht weiter. Wenn es in der tiefsten Schicht angekommen ist, wird es mittels einem Medium (z.B. Kabel) übertragen und diese Schicht packt die Pakete wieder aus und sendet es an die nächst höhere Schicht, bis es wieder in der Anwendung landet.
Die übermittlung von Datenpakete dauert ca. 30ms





1. Physical Layer (Bitübertragungsschicht)

Sorgt für die ungesicherte Datenübertragung eines BitStroms zwischen zwei Systeme und definiert:- elektrische Eigenschaften; Codierung; Mechanische Eigenschaften
-> Das eigentliche Übertragungsmedium (z.B. Kabel) liegt unterhalb des Physical Layers und ist nicht Teil von OSI
Hardware: Leitungen, Stecker, Repeater | Sendet: Bits (Leitungscode)

2. Data Link Layer (Sicherungsschicht)

Ist die gesicherte Übertragungsstrecke zwischen direkt verbundenen Knoten. Der Data Link Layer benutzt den Dienst der ungesicherten Datenübertragung des Physical Layers und bietet dann seinerseits einen gesicherten Dienst der nächsthöheren Schicht an. Aufgaben: Realisieren sicherer Verbindung, Massnahmen zur Fehlererkennung oder -korrektur, (Framing) Verpacken von Datenblöcken in Datenrahmen für die Übertragung, (Framing) Auspacken der Datenblöcke aus den empfangen Datenrahmen, Rahmenerkennung, Fluss-Steuerung (Flow-Control)
Hardware: Bridge, Switch | Sendet: Frames

3. Network Layer (Vermittlungsschicht)

Durch die Vermittlungsschicht können zwei untersch. Systeme miteinander kommunizieren
Aufgaben: Nachführen Routing Informationen, Ermittlung des Weges, Forwarding
Hardware: Router, Layer-3-Switch | Sendet: Pakete

4. Transport Layer (Transportschicht)

User Data Protocol (UDP) bspw. Voice-Call
Transmission Control Protocol (TCP) bspw. Video Call
Aufgaben: Fehlerkorrektur, Reihenfolge Datagramme ordnen, Fehlererkennung | Sendet: Segmente

Klassifizierung von Daten

Verbindungslos	Verbindungsorientiert
• Senden ohne Vorbereitungen an beliebige Ziele • Umgehung von Störungen ohne weitere Massnahmen von Aussen • In Transit-Knoten benötigen keine Ressourcen für den Verbindungs-Context und keine Verwaltung dafür.	• Erlaubt garantierte Charakteristik (Durchsatz, Delay, Verlust) auf der Strecke • Kontrolle und gezielte Lenkung von Verkehrsströmen • Reihenfolge der Daten bleibt erhalten • Weiterleitungsentscheid aufgrund der Verbindungs-nummer ist effizienter

	Gesichert	Ungesichert
verbindungsorientiert	TCP	
verbindungslos	(RUDP)*	UDP

5. Session Layer (Kommunikationsschicht)

Auf- & Abbau einer Session -> wird die Transportverbindung unterbrochen, so kann der Session Layer eine neue Verbindung aufbauen.

6. Presentation Layer (Darstellungsschicht)

Umwandlung der Darstellung von Daten; Konvertierung ASCII (oder anderen Codes) & vers. Zahlenarten

7. Application Layer (Anwendungsschicht)

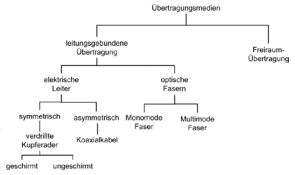
Bindeglied zur eigentliche Anwendung, bestimmt die Protokolle versch. Anwendungen (bspw. Fillezilla, WWW)

Kritik zu OSI

zu komplex, zu teuer; zu ineffizient -> Aus diesem Grund hat es sich nicht durchgesetzt in der prakt. Anwendung
Aus heutiger Sicht fehlen folgende Punkte: 1. Daten-Sicherheit und -Verschlüsselung 2. Hochverfügbarkeit und Redundanz; 3. Netzwerk-Management; 4. Zeitsynchronisation

Übertragungsmedium

Das Übertragungsmedium gilt als Layer 0, welche unterhalb des Physical Layers ist.
Die technische Kommunikation basiert auf Ausbreitung von Signalen. Dabei ist die

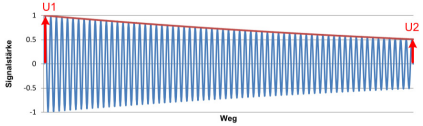


Ausbreitungsgeschwindigkeit eine wichtige unveränderliche Grösse:
- Lichtgeschwindigkeit im Vakuum: ca. 300‘000‘000 m/s (pro ns ca. 30cm) Bsp. Funk
- Ausbreitungsgeschwindigkeit in Materie ist langsamer ca. 200‘000km/s

Signaldämpfung [dB]

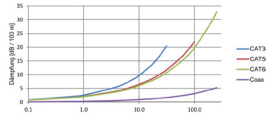
Je höher die Bandbreite (Hz), desto höhere Datenrate (bit/s) können übertragen werden. Dabei hat die Signaldämpfung (in dB) ein wesentlicher Faktor für die erreichbare Distanz. Für die Übertragungsmedien ist die Dämpfung pro Distanz (typischerweise dB pro 100m) massgebend. Diese charakteristische Grösse wird Dämpfungsbelag genannt und wird

Signaldämpfung[dB] = $10 * \log(P_1/P_2) = 10 * \log((U_1/U_2)^2) = 20 * \log(U_1/U_2)$ wobei P = Leistung U = Spannung
Bsp Bild: $U_1/U_2 = 1 / 0.5 = 2$ **Signaldämpfung = 20 * log(2) = 6dB**
-> Eine Dämpfung von 6 dB bedeutet also eine Leistungsabnahme um den Faktor 4 oder eine Spannungsabnahme um den Faktor 2.



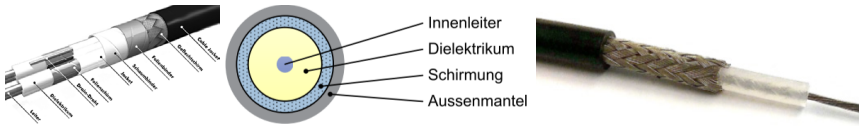
mögliche Störungen bei der Übertragung

- Übersprechen zwischen den Leitungen
(elektromagnetische Felder von den einzelnen Kabel)
- Rauschen des Empfängers
- Einstreuungen durch andere Geräte



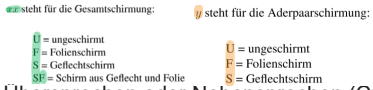
Koaxialkabel (erstes Bild unten = Twinaxial-Kabel)

Eigenen sich für die Übertragung hochfrequenter Signale und sind relativ unempfindlich gegenüber elektromagnetische Strögungen, wobei es mechanisch (knicken/quetschen) heikel ist. Die Grenzfrequenz beträgt 1 GHz



Paarsymmetrische Kabel (Twisted Pair)

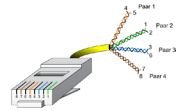
Schon lange im Einsatz (bspw. Hausanschlüsse Telefonie), je nach Qualität kann dies auch für breitbandige Datenübertragung genutzt werden. Gibt geschirmte (Shielded Twisted Pair, STP) und ungeschirmte Kabel (Unshielded Twisted Pair, UTP) und sind in unterschiedliche Kategorien eingeteilt. Die Kategorie 6 hat bspw. einen Frequenzbereich von bis zu 250MHz und ist daher am besten gegeignet für die Gigabit-Ethernet. Neben der Gesamtschirmung können einzelne Aderpaare zusätzlich geschirmt werden. Der Namen der Kabel setzt sich wie folgt zusammen: xx/yTP
TP sind anfällig auf elektromagnetische Störungen. Diese werden als Übersprechen oder Nebensprechen (Crosstalk) bezeichnet. Zwei parallel geführte Leitungen verhalten sich wie Sender und Empfänger. Es gibt kapazitive und induktive Störungen. Diese Störungen können bspw. durch ein komplementäres Signal minimiert werden -> Kapazitiv eingekoppelte Störungen auf TP Kabeln werden durch ein komplementäres Signal minimiert. Der Empfänger subtrahiert die Signale, wobei die eingekoppelten gleichgerichteten Störungen weitgehend aufheben (durch einen Differenzverstärker) induktive Störungen entstehen durch Leiter, diese kann durch die Verdrillung von Kabeln (gegenpolig) nahezu eliminiert werden.



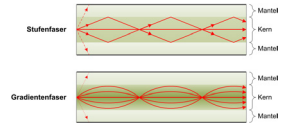
Stecker

Die 4 Adernpaare (8 Drähte, je zwei verdrillt) paarsymmetrischen Kabeln werden typischerweise mit den RJ45-Stecker verwendet. 10 / 100 Mbit: werden 2 Adernpaare gebraucht Gbit: Werden alle 4 Adernpaare gebraucht
Heute ausschliesslich gerade Stecker (Systeme können den rest selber)

Lichtwellenleiter / Glasfaserkabeln



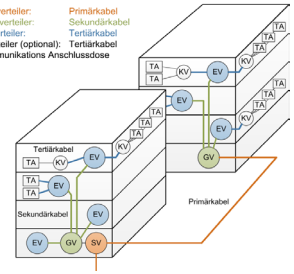
+ Vorteile: Unempfindlich gegenüber elektro.mag.Störungen; kleine Signaldämpfung und daher grössere Distanzen; grössere Bandbreite und somit grosse Übertragungsrate
- Nachteil: Teuer
Die Glasfaser besteht aus einer dünnen, zylindrischen Faser, das Kernglas ist von einem Mantelglas umschlossen mit geringer Dichte. Die mit dem Datensignal modulierten Lichtstrahlen breiten sich im Kernglas aus und werden am Mantelglas totalreflektiert aufgrund der Brechnungsunterschiede. Der Brechnungsindex des Kerns muss leicht höher sein, als der Brechnungsindex des Mantels. Wir unterscheiden bei den Glasfasernkabeln zwischen 2 Typen:
- Monomodefasern: genau in der zentrale Achse des Kerns eine Faser, teurer, grössere Distanz möglich
- Multimodefasern: hat mehrere Fasern, Unterscheidung Stufen- und Gradientenfasern. Stufenfasern haben einen Delay Skew von ca 50 ns/km und zeigen damit ausgeprägte Modendispersion.



Strukturierte Gebäudeverkabelung

Das wichtigste ist, dass die Verkabelungsinfrastruktur Reserven und die Kommunikationsanforderungen der nächsten 10 - 15 Jahren zu berücksichtigen
Primärbereich: max. Länge von 1‘500m
Sekundärbereich: max. 500m
Teritärbereich: max. 90m zuzüglich 10m Anschlusskabel

SV Standortverteiler: Primärkabel
GV Gebäudeverteiler: Sekundärkabel
EV Etagenverteiler: Tertiärkabel
KV Kabelverteiler (optional): Tertiärkabel
TA Telekommunikations Anschlussdose



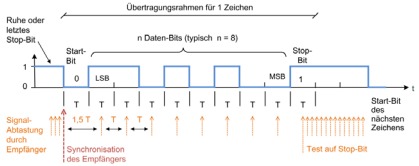
1. Physical Layer (Bitübertragungssschicht)

Der Physical Layer sorgt für die ungesicherte Übertragung eines Bitstroms zwischen zwei Systeme.
Shared Medium: Bspw. Luft im Klassenzimmer
Punkt-Punkt: Bspw. Kabel
Simplex: geht nur in eine Richtung (Bspw. Radio oder TV)
Duplex: geht in beide Richtungen (Bspw. Anruf)
Halbduplex: Zu jedem bestimmten Zeitpunkt gibt es genau einen Sender
Vollduplex: Es kann zu jedem Zeitpunkt frei in jede Richtung gesendet werden
-> Für die Kommunikation wird eig. immer Halb- oder Vollduplex verwendet

Parallele Übertragung

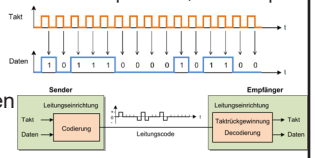
mehrere Bits gleichzeitig über mehrere Leitungen übertragen (Vergleichbar mit mehreren Spuren auf der Autobahn)
--> Wurde vor allem früher verwendet

Serielle Übertragung (asynchron)



Asynchron = Sender und Empfänger haben nicht dieselbe Zeitbasis
Das vom Sender T (Takt) ist unterschiedlich zum T (Takt) des Empfängers --> max. Abweichung 0.5Bit auf 9.5Bit sein
$f_{max/min}$ = Baud / T * Anzahl übertragende Bits = Res -> Res / 2 = kHz
Bit-Dauer T? **1 / Baud = x *10^{-6} (-> usec)**
Zuerst kommt das LSB und am Schluss kommt das MSB -> ASCII: 0*0101011** --> Hex: 0x2B | * MSB; ** LSB |
Vier Bits geben eine Hexadezimalzahl, ein solcher Block wird auch Nibble genannt.

Serielle Übertragung (synchron)

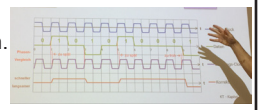


Sender und Empfänger arbeiten im selben Takt
-> Die Gefahr auf Übersprechung ist relativ hoch, weil man Takt und Daten zusammen versendet. Der Empfänger muss das Taktsignal aus den Daten herauslesen

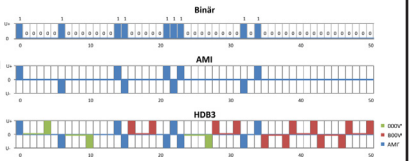
Leitungscodes

Der Empfänger muss das Taksignal aus den Daten herauslesen. Die Leitungscodes sollen - die physiaklisch vorhandene Bandbreite effizient nutzen - Taktrückgewinnung erlauben - Möglichst gleichspannungsfrei sein, um Sender und Empfänger mit Übertragern galvanisch trennen (da fliesst kein Strom und ist komplett voneinander losgelöst) zu können - AMI = Alternate Mark Inversion

AMI garantiert unabhängig von den Daten ein gleichspannungsfreies Signal, jedoch ist es **nicht** für die Taktrückgewinnung geeignet -> Takt muss auch ersichtlich sein, wenn keine Daten gesendet werden



Takrückgewinnung erlaubt das erkennen des lokalen Taktes durch die übertragenen Daten.
HDB3 Taktrückgewinnungsvariante, absichtliche Code-Violation (V-Puls) -> immer vier Nullen hintereinander werden in die entgegengesetze Richtung (gleich wie vorheriger Impuls) gekippt und so hat man trotz einer längeren Null-Sequenz die Möglichkeit auf Taktrückgewinng
-> Gleichspannungsfreiheit geht verloren
Regel zum Bild rechts: Folgen von vier Nullen werden durch 000V (0001) oder B00V (1001) ersetzt. Die Auswahl folgt so, dass zwischen 2 V-Pulsen immer eine ungerade Anzahl Pulsen liegt (entweder B- oder AMI-Puls).



Datenübertragungsrate

- Bit pro Sekunde

Frame-Rate(F) = Bitrate / Total Frame (304 + Nutzdaten [P] * 8)
Nutzbit-Rate (N) = F * P
Bitrate / Datenübertragungsrate / Durchsatz sind Synonyme und bezeichnen Zeiteinheit einer Informationsmenge

Nutzung der Bandbreite

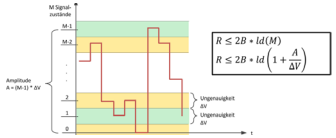
maximale Symbolrate f_s(Baud) ist gemäss Nyquist gleich der doppelten Bandbreite B (Hz) des Übertragungskanals
f_s=2*B -> Gilt für ideales Übertragungsmedium ohne Störung

Maximale Bit-Rate (Hartley's Gesetz)

Anzahl Symbole = unterscheidbare Signalzustände: $M = 1 + \frac{A}{\Delta V}$
Informationsgehalt eines Symbols: $I_s = ld(M) = ld\left(1 + \frac{A}{\Delta V}\right)$
Die maximale Bit-Rate R einer Übertragungsstrecke ist gemäss Hartley, das Produkt der max. Symbolrate 2B und dem Informationsgehalt der Symbole

Kanalkapazität nach Shannon

$$C_s = B \cdot ld\left(1 + \frac{S}{N}\right)$$



2. Data Link Layer(Sicherrungsschicht)

Stellt dem übergeordneten Network Layer eine gesicherte Übertragungsstrecke zwischen zwei direkt verbundenen Teilnehmer zur Verfügung. Die Anforderungen sind je nach Anzahl der Teilnehmer unterschiedlich. Für Punkt-Punkt-Verbindung folgende Aufgaben: 1. sichere Verbindung inkl. Fehlererkennung / -korrektur 2. Verpacken / Auspacken der Datenblöcke in Frames 3. Frame-Erkennung 4. Flow-Control (Sender nicht schneller als Empfänger)

Framing

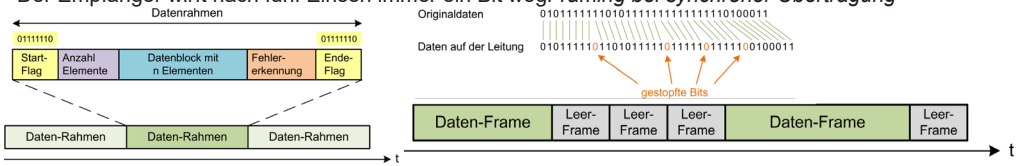
Daten aus dem NetworkLayer werden in Blöcke bestimmer Grösse unterteilt und fügt eigenen Header dazu, diese Frames werden ungesichert über den Physical Layer versendet. Der Empfänger muss diese Frames mit ankommenden Bitstrom erkennen und behandeln. Dabei unterscheidet man zwischen synchron und asynchron.

Unterschied synchron / asynchron

Synchron: Sender und Empfänger haben die gleiche Takrate (gleichzeitig) -> Bits werden auch gesendet, wenn keine Informationen übermittelt werden
Asynchron: Information der Taktrate fehlt beim Empfänger und hat seine eigene Zeitbasis

Bitstuffing bei synchroner Übertragung (nur synchron verwendet) -> Start/Endflag mit 01111110

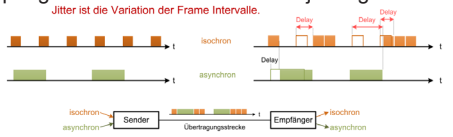
- Der Sender fügt dem Datenstrom nach fünf Einsen immer eine 0 ein
- Der Empfänger wirft nach fünf Einsen immer ein Bit weg*Framing bei synchroner Übertragung*



Frame-Länge

+ Vorteil lange Frames: Gutes Verhältnis von Daten zu Header-Länger -> höhere Netto-Bitrate (auch Overhead)
- Nachteil langer Frames: grössere FehlerWSK; grösserer Datenverlust bei Fehler; grössere WSK unentdeckten Fehler; Übertragungsstrecke länger blockiert -> höherer Jitter (Zeitvariation der Frames = Jitter)
Berechnung Nettobitrate: **Bruttobitrate * (Nutzdaten / Nutzdaten + Header)**
Für einen Lasttest muss man kurze Frames wählen!

Einfluss der Frame-Länger auf Jitter
Lange Frames blockieren die Übertragungsstrecke für die Dauer ihrer Übertragung. Die Blockierung der Übertragungsstrecke führt durch asynchrone Frames zu einer Verklumpung der isochronen Frames -> je länger Frames desto ausgeprägter (Abb. orange: kommunikationsframes typ. klein 30Bit; grün: Datenframes typ. gross)



Einfluss der Frame-Länger auf den Durchsatz
Nettobitrate und Frame-FehlerWSK nehmen zu bei grösseren Frames. FrameWSk hängt von der BitfehlerWSK p_e (BER) des Physical Layers ab. Zuerst wird die WSK berechnet, dass ein Frame Länge N, keinen Fehler enthält:
Frame-**Erfolgs**WSK = **$(1-p_e)^N$** -> daraus folgt -> Frame-**Fehler**WSK = **$1-(1-p_e)^N$**
Für eine BER p_e << 1 lässt sich die opt. Frame-Länge bei einer Header-Länge H nährerungsweise berechnen:
Optimale Frame-Länge ≈ **$\sqrt{(H/p_e)}$**
Wie viel Frames sind nach der Übertragung des N Bits noch fehlerfrei? **$A_n = A(1-p_e)^N$**

Fehlererkennung

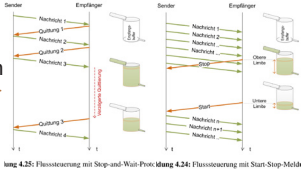
Der Data Link Layer muss fehlerbehaftete Daten nochmals senden (Retransmission)
Hamming-Distanz *(h):* gibt an wie viele Bits im Bitmuser eines gültigen Codes mind. geändert werden müssen, bis der nächste gültige Code vorliegt -> Anzahl erkennbare Fehler **e = h-1**
Parity*:* Jedem Datenelement wird ein zusätzliches Bit (Parity-Bit) zugefügt. Dieses wird so gewählt, dass die Anzahl Einsen inkl. Parity-Bit entweder gerade (Even Parity) oder ungerade (Odd Parity) ist. Even und Odd Varianten sind gleichwertig. Es ist **genau ein Bitfehler erkennbar** mit dieser Variante (h = 2)
Prüfsumme*:* Anzahl *Datenelemente n* eine Summer der *Breite m*, die *Prüfsumme P* kann durch einfache Addition berechnet werden: $P = \left(\sum_{i=0}^{n-1} \text{Element}[i]\right) \mod 2^m$ Mehrbitfehler werden nicht erkannt; Einsatz bei TCP oder UDP
CRC: 1. multipliziere *p mit x^m* (m Nullen an die Nachricht anhängen) 2. Teile das *Ergebnis f* durch das *Generatorpolynom g* und bilde den *Rest r* 3. addiere r zu f -> Das Ergebnis h entspricht dem gesuchten Codewort; h ist durch g teilbar --> Für Ethernet braucht man mind. CRC-32

Fehlerkorrektur

Backward Error Correction: Empfänger kann Fehler erkennen, aber nicht korrigieren. Daten werden nochmals gesendet. -> Funktioniert gut bei kleiner Bitfehlerwahrscheinlichkeit
Forward Error Correction: Empfänger kann Fehler erkennen & korrigieren. Notwendig wenn kein Rückkanal vorhanden ist oder die FehlerWSK gross ist. -> Verfahren bei hohen BER unbrauchbar

Flow Control (Flusssteuerung)

Es könnten Probleme auftreten, wenn der Sender über eine gewisse Zeit mehr Daten sendet, als der Empfänger verarbeiten kann. Der Flow Control erlaubt es den Sender temporär zu stoppen. Wenn Stop-and-Wait-Protokoll verwendet wird, so ist der Flow Control impliziert vorhanden.



ung 4.25: Flusssteuerung mit Stop-and-Wait-Protokoll ung 4.24: Flusssteuerung mit Start-Stop-Meldung

Medium Zugriff

Einer der wichtigste Dienste bei mehreren Teilnehmern ist das Regeln des Zugriffs auf das Übertragungsmedium. Die Verfahren zur Regelung des Zugriffs lassen sich grob in deterministische und nicht-deteministische einteilen.

Wichtige Layer 2 Protokolle

- High Level Data Link Control (HDLC) -> ISO & ITU genormtes Protokoll, ist in vielen Anwendungen anzutreffen
- Point to Point Protocoll (PPP = HDLC Variante) -> zur Anbindung abgesetzten Teilnehmern an Internet Provider

3. Lokale Netzwerke (Local Area Network = LAN) [Teil des Network Layers]

LAN ist ein räumlich begrenztes Netzwerk in dem Daten mit hoher Geschwindigkeit übertragen werden. Es gibt versch. Verbindungsarten, die sich in Bandbreite/Reichweite unterscheiden (WAN, LAN, MAN, PAN, BAN etc)

Topologie

Früher vor allem Nord-Süd-Traffic, heute West-Ost-Traffic, daher in DatenCenters häufig Spine-Leaf-Architektur

Bus: einfachste Topologie Vorteil: einfach & kostengünstig; Nachteil: Unterbruch, Kapazitätbeschr.

Linie: Industrieanwendung (Automation beliebt); Nachteil: Unterbruch

Ring: Linie mit einer Endverknüpfung am Schluss Vorteil: Unterbruch sicher

Stern: Vorteil: Ausfall keinen Einfluss auf Rest Nachteil: Single-Point-of-Failure (in der Mitte)

Baum: Stern-Topologie an Stern-Topologie;
flexible Kontrolle Datenfluss -> wird heute verwendet

Spine-Leaf
-> ist jeweils eine logische Verbindung von Komponenten

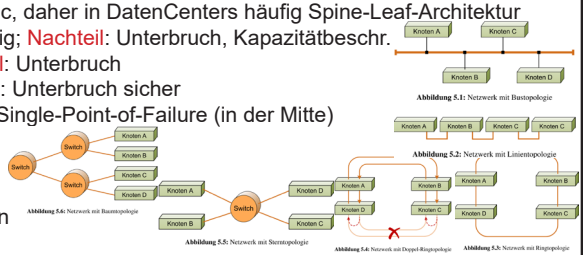


Abbildung 5.1: Netzwerk mit Bustopologie
Abbildung 5.2: Netzwerk mit Linientopologie
Abbildung 5.3: Netzwerk mit Ringtopologie
Abbildung 5.4: Netzwerk mit Doppel-Ringtopologie
Abbildung 5.5: Netzwerk mit Sterntopologie
Abbildung 5.6: Netzwerk mit Baumtopologie

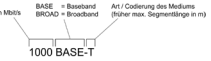
Übertragungsraten

Unicast: Genau ein klar spezifizierter Empfänger; Frame trägt Adresse des Empfängers; *Analogie Briefpost*

Multicast: Gruppe von Empfängern; Frame trägt Multicast-Adresse der Gruppe; *Analogie Mailing-Liste*

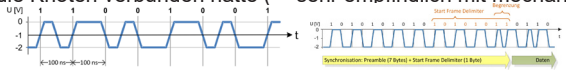
Broadcast: an alle Knoten im LAN; Frame trägt Broadcast-Adresse des LAN; *Analogie Radio-Senderstation*

Anycast: z.B. DNS-Server hat diverse redundante Stellen wo bei einem Ausfall zum Zug kommen, jeder diese Stelle hat die gleiche IP wie der DNS



Ethernet

Ursprünglich für LAN angedacht, mittlerweilen vielen Orten im Einsatz (Industrie, Automobile, Aviation, Railway). Wurde laufend weiterentwickelt Verbilligung (Thick -> Thin Ethernet); Vereinfachung (Koaxial -> TP-Kabel); Leistungssteigerung (10 -> 100'000 Mbit/s). Druch das *Carrier Sense Multiple Access with Collision Detecion* (CSMA/CD) entstand am MIT ein verbessertes Verfahren. Zur erst mit Transceiver und Thick-Wire-Ethernet-Kabel, welches angebohrt werden musste. Danach entstand das Thin-Wire-Ethernet (10BASE2), welches mit den normalen, dünnen Koaxialkabeln direkt die Knoten verbunden hatte (-> sehr empfindlich mit mechanischen Probleme, hat sich nicht durchgesetzt.)



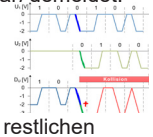
Leitungscodierung

Als Leitungscodierung wird ein Manchester-Code *(Vorteil der Taktrückgewinnung)* eingesetzt, dabei wird eine 1 als positive und eine 0 als negative Flanke übertragen. Für 10Mbit/s wird 10MHz benötigt (Doppelte vom Minimum). *10BASE2 verwendet asymmetrische Signalpegel* von **0V und -2V.** Mit *10BASE-T* ist der Manchester-Code *gleichstromfrei*. Da der Code nicht eindeutig ist, ist eine Bitsynchronisation des Empfängers notwendig. Dabei braucht es 8 Bytes (7 Preamble und 1 Start Frame Delimiter[SFD]), das Ende wird durch eine 1-1-Bitfolge angezeigt.

Datenübertragung und Zugriffskontrolle (CSMA/CD = Carrier Sense Multiple Access / Collision Domain)

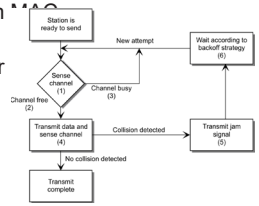
Durch das CSMA/CD Zugriffsverfahren ist jeder Knoten autonom und es braucht kein zentrales Kontrollelement.
Prinzip: Listen before Talk;

Senden ohne Kollision Ein Knoten **wartet** mit dem Senden eines Frames bis der Übertragungskanal **frei** ist. Danach sendet er über den Physical Layer die **Preamble und SFD, damit der Empfänger synchronisieren kann**. Dabei kontrolliert der Physical Layer ob eine Collision stattfindet. Falls ja, überlagen sich die Signale U_1 und U_2, resultierende höhere Signalpegel U_D wird erkannt und an den DataLinkLayer (Collision Detected Signal) gemeldet. DataLinkLayer meldet Beendigung kollisionsfreier Übertragung.



Empfang ohne Kollision Sobald ein Knoten anfängt zu senden, realisieren dies die andern und unternehmen keine Sendungsversuche mehr. Alle Knoten werden nun zu Empfangsknoten, synchronisieren nun die Empfangseinrichtung und übersetzen Zieladresse von Manchester- zu Binärcode. Zieladresse von PhysicalLayer zu DataLinkLayer. Der adressierte Code empfängt nun die restlichen Daten

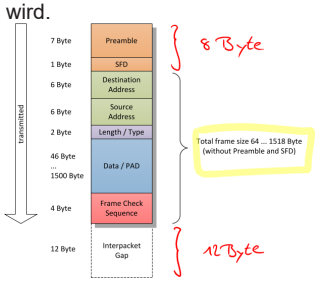
Senden mit Kollision Eine Kollision kann **nur zu Beginn stattfinden**. Um sicher zugehen, dass alle Knoten die Kollision erkannt haben, wird ein Jam-Signal (32Bit mit bel. Inhalt) ausgegeben -> Sendung wird eingestellt. Empfänger kann zwischen gültigen Frame und Jam-Signal unterscheiden, da Prüfsequenz am Ende falsch ist. Danach wird nach dem Binary Exponential Backoff (Vielfaches der Slotzeit t_s)genannten Algorithmus ein erneuter Sendeversuch gestartet.(10MBit/s: t_s = 51.2µs // 100MBit/s: t_s = 5.12µs). Zufällig 0* t_s oder 1* t_s gewartet. Danach Zeitbereich verdoppeln. Nach insgesamt 16 erfolglosen Versuche wird abgebrochen. CSMA/CD wird vom MAC Layer durchgeführt, CPU wird nicht belastet.



Durchsatzbetrachtung von CSMA/CD Bei viel Verkehr, ist die WSK von Kollision grösser -> Reduktion Leistung LAN spätere Untersuchungen zeigten jedoch,
auf falschen Grundannahmen beruhten (Poisson-Verteilung des Verkehrs)

Aufbau des Ethernet-Frames

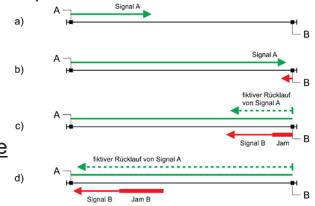
Preamble hat das Format 10101010, welches mit dem Manchester-Code übermittelt wird.
Die Übermittlung erfolgt vom kleinsten (LSB) zum grössten (MSB) Bit.
wichtig! Es wird Byte für Byte (von Links nach Rechts) gesendet,
jedoch innerhalb des Bytes in umgekehrter Reihenfolge (LSB zuerst).
SFD(Start Frame Delimiter), für die Bitsynchronisation, Form: 10101011.
Ethernet Header (total 14 Byte) beinhaltet 6 Byte
Destination Adresse, 6 Byte I/G=1 steht für Group Adress
Source Adresse 6 Byte
Type / Length 2 Byte
46 bis 1518 Byte mit Daten, if (Daten <46 Byte), wird der Rest mit Nullen aufgefüllt.
Am Schluss wird die Frame Check Sequenz von 4 Byte übermittelt (CRC32).
Evtl. noch der Interpacket Gap, danach geht es wieder auf 0 und "Ruhe auf der Leitung



Repeater

Repeater arbeiten auf dem Physical Layer des OSI-Modells, dabei wird Amplitude, Synchronisatinosbits (Preamble), Flanken und kl. zeitlich Verschiebungen (Jiter) restauriert. Repeater schickt mit kl. Verzögerung das Signal ungesehen weiter. (-> Frameinhalt, Adresse etc. wird ignoriert). Multiport Repeater = Hub -> Alle Ports sind gleichberechtigt
Mit Hubs können Stern- und Baumtopologien aufgebaut werden. Über einen Repeater angeschlossene Segmente bilden eine Collision Domain, innerhalb dieser Domain muss jede Collision von allen angeschlossenen Knoten erkannt werden.

Konfigurationsregeln Es gibt Begrenzungen bzgl. Anzahl Repeatern und Kabelsegmenten, die zwischen zwei bel. Knoten eines Netzwerks liegen können. **1. Round Trip Delay**: bel. sendender Knoten muss Kollission erkennen, während er am Senden ist. **2. Interpacket Gap Shrinkage**: Interpacket Gap (IPG) bezeichnet zeitlichen Abstand zwischen zwei folgenden Frames. Min. **IPG = 96-Bit Zeiten**. Beim Durchlaufen eines Repeaters werden verlorene Bits der Preamble regeneriert, Dadurch wird die Preamble länger und IPG kleiner



a) Knote A beginnt zu senden
b) B sendet im letzten Moment (kurz bevor Signal von A eintrifft)
c) B stellt Kollission fest und sendet JAM-Signal
d) Signal B trifft bei Knoten A ein solange dieser noch am Senden ist
-> *Collision Domain* darf nur halb so gross sein, wie die Ausdehnung des kürz. Frame

Falls Netz > doppelte Ausdehnung von Signal, wird der Knoten A die Kollision des Signals A **nicht** bemerkt. -> Frame wird von A **nicht** nochmals versendet.
Dies wird Late Collision genannt, Anwender merkt praktisch nichts, TransportLayer übermittelt fehlende Frames nochmals -> Performance und Prozessoren leiden

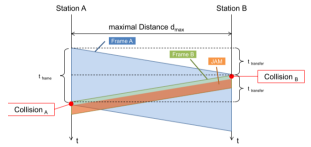
Bedingung Erkennung einer Kollision: $t_{frame} > 2*t_{transfer} \,||\, t_{frame} > 2*(\sum t_{transfer} + \sum t_{forwarding})$

max. Ausdehnung eines Segments: $t_{frame} = Framesize_{min}/Bitrate \;\&\; t_{transfer} = d_{max} / C_{medium}$

Man löst nach d_{max} auf $d_{max} < 0.5 * Framesize_{min}/Bitrate * C_{medium}$

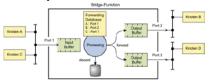
wichtig! Bei Framesize: Preamble und SFD nicht berücksichtigen = 64Byte + Daten

Bedingung Erkennung einer Kollision bei Weiterleitung:
$t_{frame} > 2*(\sum t_{transfer} + \sum t_{forwarding})$



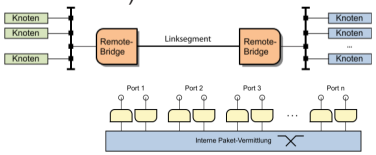
Bridge (=Switch)

Eine Bridge koppelt mehrere Collision Domain miteinander, ohne das CSMA/CD Verfahren zu beinflussen. Die Bridge arbeitet auf dem Data Link Layer und prüft empfangene Frames und leitet diese aufgrund der Zieladresse an andere Ports weiter. Jedoch soll der Sende- und Empfangsknoten nicht merken, dass eine Bridge dazwischen ist -> Frameinhalt darf sich nicht ändern -> Daher auch transparent Bridging genannt. Bridges arbeiten im Promiscuous Mode und leiten die Frames mit der Ursprungsadresse weiter, obwohl jeder Port eine eigene **MAC-Adresse** besitzt. Bridges brauchen keine Konfiguration und auch keine Änderung an den Netzwerkknoten. Durch das Abhören aller Ports und dem merken der Sender-Adresse der empfangenen Frames entsteht eine Forwarding Database. Dadurch lernt die Bridge die einzelnen Adresse mit den versch. Ports. Nach einer gewissen Zeit (Default 600s) wird die Database wieder gelöscht.



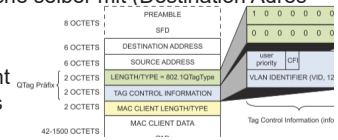
Wenn Knoten A, Knoten B etwas senden möchte, so wird dies von Port1 empfangen und in den Input Buffer gelegt (falls unvollständig oder fehlerhaft -> gelöscht). Bei korrekten Frames wird die Database nach dem Ausgabe Port durchsucht -> via Output Buffer zu Port2.
Falls der adressierte Empfänger (bspw. C) dieselbe PortNr. wie Absender (A) hat, wird das Frame nicht in den Output-Buffer übertragen und gelöscht (discarded). Falls Destination-Adresse nicht gefunden wird, so wird das Frame auf allen Ports ausgegeben (Frame Flooding). Broadcast-Frames gehen an alle Knoten und dürfen von der Bridge nicht gefiltert werden, auch Broadcast-Frames werden am Eingangs-Port nicht ausgegeben.
Da die Bridge einen Teil des Frames ausfiltert, kann sie die Last in Teilnetzen reduzieren -> dient sowohl zur Erweiterung des bestehenden LANs, als auch zur Trennung des Verkehrs und Vergrösserung der Sicherheit

gegenüber Totalausfällen -> gilt vor allem dann wenn viel lokaler Verkehr herrscht.
Durch Vollduplex Betrieb kann der Durchsatz (theoretisch) verdoppelt werden, denn es kann gleichzeitig gesendet und empfangen werden. Voraussetzung ist getrennte oder bidirektionale Übertragungskanäle (heutzutage meist mit Twisted Pair-Verkabelung). Vollduplex auf einer Punkt-Punkt-Verbindung erlaubt über die durch die Kollisionenerkennung limitierte Distanz hinauszugehen. **wichtig!** in einem vollständig vollduplex-fähigen geswitchten Netz (ohne Repeater) treten keine Kollisionen mehr auf, da jede Punkt-Punkt-Verbindung ein unabhängiges LAN mit genau einem Sender und einem Empfänger pro Verbindung darstellt. (-> mikrosegmentierten LAN). Die Frames bleiben solange im Switch zwischen gespeichert, bis der Port frei ist.



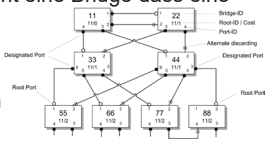
Remote Bridges erlauben die Verbindung über grosse Strecke, bei vollduplex nur durch Eigenschaften Überstragungsstrecke beschränkt. Serverfarmen, wurden häufig mit Remote Bridges aufgebaut
Multiport-Bridges (Switch) problemlose Koppelung von 2+ Segmenten, typischw. 4 - 64 Ports welche sehr schnelle interne Paketvermittlung besitzen
Die Bündelung von mehreren physischen LAN-Ports zu einem schnellen Kanal wird als Link Aggregation oder Port Trunking bezeichnet.

Repeater vs. Bridge
Repeater: Hub in Physical Layer - verstärkt Signal und Weiterleitung an alle Ports
Bridge: Switch in DataLink und Physical Layer - Fehlererkennung mit CRC in FCS - erste Frame in Bridge wird an alle Ports (ausser Sender) versendet, dadurch entsteht eine Learning Bridge, welche selber mit (Destination Adresse und Port; Source Adresse mit Port; TimeStamp für Löschung) gefüllt wird



VLAN
Dem Switch werden die einzelnen Ports einem VLAN zugeordnet, dadurch braucht man nicht n-Switches pro Stockwerk. Möchte man von VLAN_1 zu VLAN_2 etwas senden, muss das Datenpaket über den Router kehren. Mit dem grün/blauen **VLAN-Header tagged** man die Leitungen um eine entsprechende Zuweisung zu machen

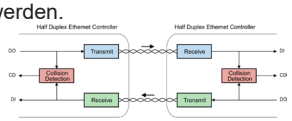
Redundanz Protokolle
Für die Verbesserung der Netzwerkzuverlässigkeit, können Pfade bei Bridges redudant aufgebaut werden werden. Wichtig dabei ist, dass die Frames nicht endlos im Netzwerk herumschwirren (bspw. bei unbekannter Destination Adresse), dies ist die Aufgabe der Redundanzprotokolle.
Ziel des Spanning Tree Protocol ist, dass man alle Segemente in einer loopfreien Topologie verbindet.
Vorgehen **1. Root-Bridge auswählen** (tiefste BridgeID) **2. Baum aufbauen** (ausgehend von der Root-Bridge) **3. Redundante Pfade sperren**. -> Der Algorithmus beruht auf einem speziellen Austausch zwischen den Switches, so genannte Bridges Protocol Data Unit (BPDU). BPDU verwenden spezielle Multicast Adressen (Bridge Group Adresse), diese werden von den Switches nicht weitergeleitet.
Multicast Adressen gibt es 16 Stück zwischen 01-80-C2-00-00-00 bis 01-80-C2-00-00-0F.
Vorgehen: Jeder Knoten preist sich am Anfang als beste Bridge mit Kosten 0 an. Erkennt eine Bridge dass eine andere Bridge die bessere ID hat, teilt der Knoten seiner Nachbarn mittels BPDU mit, welches die beste Root-Bridge ist und zu welchen Kosten er diese erreichen kann.
Die Bridge mit dem tiefsten Identifier wird zur Root Bridge.



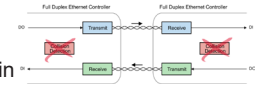
Pfadkosten: Ausgehen von root-Bridge aufsummiert. Falls redundante Pfade entstehen wird der Port mit der besten Kosten zur Root-Pfad.
Alle 2 Sekunden wird ein BPDU versendet, fällt dies aus gibt es eine Rekonfiguration des Netzes. Dies kann zu einem bis zu 30 sekundigen Ausfall führen, aus diesem Grund gibt es Rapid Spanning Tree Protocol (RSTP) bei einer Topologieänderung wird (soweit als möglich) auf die bisherige Pfade verwiesen und parallel die neuen Pfade bestimmt. Wenn neuer Baum bestummen -> umschalten => Ausfallszeit >1 Sekunde
High Availability Seamless Redundancy (HSR) and Parallel Redundancy Protocol (PRP) verbinden die hochverfügbaren Knoten über zwei Ports und schicken die Frames über unabhängige Pfade -> Hochverfügbarkeit
Die blockierten Pfade werden bei STP nicht genutzt, auch in im Extremfall. Shortest Path Bridging erlaubt mehrere Pfade aktiv zu nutzen und unterstützt grössere und flexiblere Layer2-Topologien.

Leistungsmerkmale von Bridges / Switch
Grösse Adresstabelle; Filterrate; Transferrate; Store-and-Forward-Bridge; Cut-Through-Bridges (nur Dest.Addr relv.)

Moderne-Ethernet-Systems
Werden zwei 10BASE-T Geräte miteinander verbunden, so müssen Sendeleitung mit der Empfangsleitung verbunden werden und umgekehrt. Bei normalen Knoten (bspw. PC-Netzwerkkarte) können die Leitungen mit Hilfe von Cross-Over-Kabeln angeschlossen werden. Bei Switches/Hubs sind die Leitungen am Port ausgekreuzt (MDI-X) , damit können die Geräte mit geraden Kabeln an den Repeater-Ports angeschlossen werden.
-> Auto-Crossover erkennt ob die Leitung ausgekreuzt werden muss oder nicht
-> Auto-Polarity erlaubt vertauschte Drähte eines Ader-Paares zu korrigieren



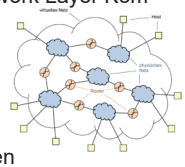
Halbduplex-Betrieb Bei 10BASE-T wird für Senden & Empfangen vers. Adernpaare verwendet. Dadurch entsteht bei Kollisionen keine Spannungsüberhöhung. Deshalb wird DI und DO zusammen überprüft und anhand dieser die CD getätigt.



Vollduplex-Betrieb
Da ein Netz aus Switches grundsätzlich aus Punkt-Punkt-Verbindungen besteht, ist auch ein Vollduplex-Betrieb möglich. Welches einfacher ist, da keine CD durchgeführt werden muss.(Loopfunktion gesperrt)
Medium Dependent Interface (MDI) mechanische und elektrische Schnittstelle zwischen Übertragungsmedium und Physical Layer des Knotens. zwischen 10 und 100 Mbit/s-Systeme = Interoperabilität gewährleistet
Autonegotion zusätzlicher Sub-Layer, damit Port automatisch zwischen 10 oder 100 MBits-Betrieb und anderen Leistungsmerkmalen(Übertragungsgeschw.; Voll-/Halbduplex Betrieb) wählen können (je nach Gegenseite)
Media Independent Interface (MII) unterstützt Datenrate von 10 und 100MBit/s über zwei je vier-Bit-breite Sende- und Empfangskanäle und ist Übertragungsmedium unabhängig. Praxis nur physische Schnittstelle
Reconcilliation Sublayer passt die Signale des MII an den Media Access Control Sublayer an. Die Schnittstelle zwischen MAC-Schicht und Reconciliation Layer ist genau gleich wie 10MBit/s-Systemen
Fast-Ethernet Grundsätzlich identisch zu 10MBit/s-Ethernet, ausser dass der maximale Netzwerkdurchmesser für das Fast-Ethernet um den Faktor 10 kleiner als beim 10MBit/s-Ethernet ist. Das CSMA/CD wurde 1:1 übernommen, daher muss die Grösse der Kollisionsdomäne verkleinert werden.

3.1 Internetprotokolle [Teil des Network Layers]

Internet = Zwischen den Netzen -> Das Internet ist ein virtuelles Netz, welches über einzelne eigene Netze besteht und zusammen verbunden ist. Das Hauptziel war einen Kommunikationskanal zu schaffen, welcher nicht von einer Verbindung abhängig ist. Der Anfang war bei den Universitäten im Jahre 1969. Die Verbindung vers. Netze erfolgt hardwaremässig mittels Router oder Gateway. Die Protokolle des Network Layers verbergen die technischen Details der zugrunde liegenden physischen Netze, Verbindungsstrecke und Router. Das resultierende Network Layer Kommunikationssystem wird darum auch als virtuelles Netz oder Internet bezeichnet.
Aufgaben: Routing (Austausch Routingprotokolle) & Weiterleitung von Pakete (Paket-Forwarding)
flacher Adressraum Vor allem für kleine Netze geeignet, MAC-Adresse identifiziert Gerät zwar eindeutig,sagt jedoch nichts über Position innerhalb des Netzes aus. -> grosse Adresstabellen
hierarchischer Adressraum beschränkte Anzahl von Subnetzen (welche wiederum Subnetz)
Analogie mit Postanschrift. Wechselt ein Host das Netzwerk, muss er eine neue Adresse bekommen



Router
Router verbinden Subnetze miteinander und arbeiten auf dem Netzwork Layer. Router empfangen nur Pakete welche über den Datalink Layer direkt an sie addressiert sind (bspw. MAC-Adresse) -> globales Überstragungssystem mit unabhängiger Addressierung möglich. untersch. Technologien muss der Router in jedem Netz ein Interface und den Protokoll-Stack der entsprechenden Technologie besitzen.

Router vs. Bridges
- Router benutzen immer den optimalen Pfad für die Verbindung zwischen zwei bel. Punkten im Netz
- (Router) verschiedene Netzbereiche können logisch und verkehrsmässig sehr gut voneinander getrennt werden
  Trennung erhöht Sicherheit und verbessert Übersicht & Verwaltung
- Anzahl Stationen via Router vernetzt beinahe unbeschränkt
- Router sind Barriere auf Schicht zwei für Broadcast-Meldungen (Broadcast-Storms)
- Leistungsfähige Router meist teurer als Bridges
- Konfiguration von Router oftmals nicht einfach
- Es gibt Nonroutable Protocols, da keine Schicht 3 verfügen (bspw. Automatisierungsprotokolle)->Bridge notwendig

Routing
Router haben zwei Hauptaufgaben: **1. Routing** - optimalen Weg (Route) im Netz zu bestimmen;
2. Forwarding - die Weiterleitung dieser Datenpakete entlang dieser Routen. Diese Routen können statisch oder dynamisch angegeben werden. Die dynamische erfolgt über die Kommunikation der Router mittels Routing Protocols

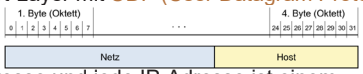
normierte Routing Protokolle:
- Routing Information Protocol (RIP) -> sehr verbreitet -> innerhalb eines Subnetzes routen
- Exterior Gateway Protocol (EGP) -> routing zwischen einzelnen Subnetzen
- Open Shortest Path First (OSPF) -> neuste und leistungsfähigste für sehr grosse Netze auch hirarchisch möglich
- Border Gateway Protocol (BGP)
- [OSI] Intermediate System - End System (IS-ES) -> routet zu Endgeräte, die nicht weiterrouten
- [OSI] Intermediate System - Intermediate System (IS-IS) -> routet zwischen Knoten die selber weiterrouten

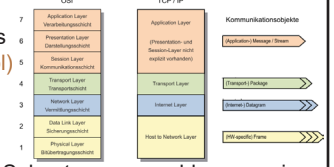
Brouter: Bridging Router oder Layer 3-Switches -> Router mit Bridging Funktionen

TCP/IP (1974 von Vint Cerf und Bob Kahn)

vier Grundsätze: 1. jedes Netzwerk für sich funktionsfähig & keine Änderungen für I-Net notwendig; 2. Kommunikation Basis "best effort" -> Paketsendung fehlgeschlagen = nochmals senden; 3. Verbindung über Blackboxes (Router / Gateway) in denen keine Informationen gesammelt wird 4. keine zentrale Funktionssteuerung notwendig

Das Internet-Protokoll ist heute das Network Layer Protokoll schlecht hin und lässt sich nicht direkt im OSI eingliedern, da es vorgängig entwickelt wurde. Oberhalt des Internet-Layers befindet sich der Transport-Layer mit UDP (User Datagram Protocol) und TCP(Transmission Control Protocol)



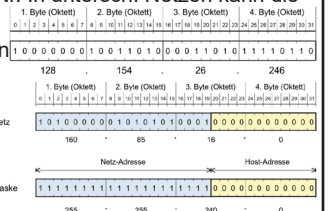


IP Adressierung

Jeder Host besitzt mind. eine Internet-Adresse und jede IP-Adresse ist einem Host zugeordnet bzw. Interface. Es gibt Hosts (Router und Server) die in mehreren Subnetzen angeschlossen sein können -> multi-homed Hosts -> eindeutige 32 Bit grosse Zahl

Die IP-Protokolle verwenden grundsätzlich die Big-Endian-Konvention (höchstwertiges Byte zuerst). Die Bit-Übertragung ist nicht definiert. *Die 32 Bits werden von links nach rechts nummeriert -> MSB Nr. 0 LSB Nr. 31*

Alle Hosts im gleichen Netz haben die gleiche Netz-Nr. aber unterschiedliche Host-Nr. In untersch. Netzen kann die Host-Nr. jedoch gleich sein.



Classless-Routing mit Netzmasken erlaubt eine flexible Zuteilung von Adressräumen in Zweierpotenzschritten. Die Subnetzmaske definiert die Grenze zwischen Netz-& Host-Adresse -> 1 für Netz- und 0 für Host-Adresse. Alternative Schreibweise: 160.85.16.0 SubNMaske 255.255.240.0 -> 160.85.16.0/20 Da die Subnetzmaske keine Lücke aufweisen kann (1 immer links, 0 immer rechts) können nur folgende 9 Werte angenommen werden:

255 (1111'1111), 254(1111'1110), 252(1111'1100), 248(1111'1000), 240(1111'0000) 224(1110'0000), 192(1100'0000) 128(1000'0000), 0 (0000'0000)

Netzwerkadresse: tiefste Nummer, Host-Nr lauter Nullen, Adresse reserviert nicht an Knoten vergeben

Broadcastadresse: höchste Nummer, Host-Nr. lauter Einsen, werden alle Knoten addressiert

Classful-Routing: Durch vordefinierte Grössen in den ersten Adress-Bits kann die Klass und somit die Grösse des Netzes bestimmen werden. Wobei die tiefste (Netzwerkadresse) und die höchste (Broadcastadresse) für Knoten nicht verwendet werden dürfen.
-> Effizientes Routing möglich, in den letzten Jahren (vor allem B-Klasse) an seine Grenzen geraten

Klasse	Adressbereich	Anzahl Netze	Anzahl Knoten	Net-Bit Wert	Anz. Potenz
A	1.0.0.0 – 127.255.255.255	128	16'777'214	0	2^24
B	128.0.0.0 – 191.255.255.255	16'384	65'534	10	2^16
C	192.0.0.0 – 223.255.255.255	2'097'152	254	110	2^8
D	224.0.0.0 – 239.255.255.255	Multicast-Adressen		1110	
E	240.0.0.0 – 247.255.255.255	Reserviert für künftige Nutzung		11110	

IP-Verwaltung: IANA steht zu oberst und unterteilt die Adresseblöcke in die 5 Regionen ein (Schweiz -> Réseaux IP Européens Network Coordination - Network Coordination Center = RIPE-NCC).

Für Privateadressen gibt es reservierte IP-Adressen A-Klasse (10.0.0.0 / 255.0.0.0), B-Klasse (172.16.0.0...172.31.0.0 / 255.255.0.0), C-Klasse (192.168.0.0 ... 192.168.255.0 / 255.255.255.0)

Loopback-Adressen: Kommunikation möglich ohne aufs Netz zu gehen. A-Klasse (127.0.0.0)

Funktion und Routing-Tabelle

Routing = Suche nach geeignetem Weg

Forwarding = Weiterleitung des Paketes -> Oftmals beides als Routing bezeichnet

Routing-Tabelle = bestimmt über welchen Pfad ein Datagram weitergeleitet werden soll (Sortierung nach Länge Netzmaske) -> Anzeige *route -n* oder *netstat -rn* (Unix) *route -print* (Windows)
Flag U = Up // Flag G = Router zu einem Gateway(Router)

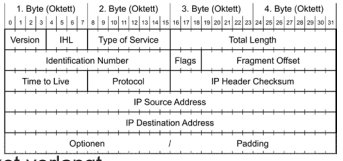
Befehle Routing Tabelle

Anzeige Routing Tabelle: *route -n* oder *netstat -rn* (Unix) *route -print* (Windows)
160.85.19.0 direkt über eth2 erreichbar: # *router add -net 160.85.19.0 netmask 255.255.255.0 dev eth2*
Default-Route Eintrag erstellen: # *route add default gw 160.85.17.1*
Eintrag löschen: # *route del -net 160.85.19.0 netmask 255.255.255.0 dev eth2*

Routingarten

- Flaches Routing: in zentralen Teilen stark vermaschter Netze -> Extremfall jeder Router kennt alle Wege => Falls ein Port ausfällt, so kann der Router direkt über einen anderen Port routen, **Aufwand ist gewaltig**
- hirarchisches (bzw. default) Routing: kleine Routing-Tabellen, jeder Host hat nur **zwei Einträge** (eigene Subnetz und Default-Eintrag für "seinen" Router). => Datagram das nicht im selben Subnetz ist, schickt es an Default-Router, dieser kennt auch nur die angeschlossen Subnetze und so weiter...

IP Protokoll



Ein IP-Datagram besteht aus einem Header und den eigentlichen Nutzdaten. Der Header besteht aus fix 20 Byte + optionalen Teil mit var. Grösse.

Version (4Bit) von IP (IPv4 oder IPv6) ausgelesen.

Internet Header Length *(IHL -> 4 Bit)* Grösse IP Header standardmässig 20 Byte. IHL-Min = 5Bit IHL-Max = 15BitDie Zahl wird immer x4 gerechnet

Type of Service (8Bit) bestimmt die Qualität (Qos) des Dienstes, welches das Paket verlangt

Total Length *(16Bit)* gibt die totale Länge an (inkl. Header und Nutzdaten -> Max. 2^{16} -1 = 65'535 Bytes, empfohlen max 576 Byte, ausser man kann sicherstellen, dass Host genügend grossen Buffer hat

Identifaction *(16Bit)* Sender zugeteilte Wert erlaubt fragementierte Datagramme wieder zusammenzustellen.

Flag *(3Bit)* beinhaltet zwei Kontrollflags für die Fragementierung (DF Don't Fragment // MF More Fragments)

Fragment Offset *(13Bit)* gibt an wo innerhalb des Datagramms ein Fragment hingehört (wird *8 gerechnet)

Time to live *(8Bit)* Bei jedem Hop dekrimentiert um 1, sobald 0 wird das Paket verworfen (Durchsch. 16 Hops)

Protocol *(8Bit)* gibt an ob ICMP (1), TCP(6) oder UDP(17)

IP Header Checksum *(16Bit)* kostet etwas & bringt sehr wenig. Denn IP ist meistens fehlerfrei -> IPv6 weggelassen

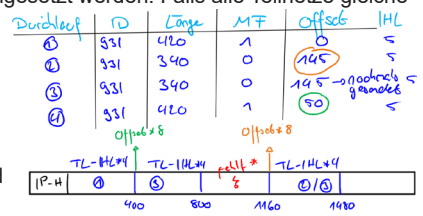
Source Address *(32Bit)* IP-Adresse des Hosts, welcher ursprünglich das Datagram versendet hat

Destination Address *(32Bit)* IP-Adresse des Hosts, der das Datagram schlussendlich erhalten soll

Options und Padding *(var. Bit)* Erweiterungen möglich ohne Design zu ändern Padding -> ganzzahliges Vielfaches von 32Bit zu füllen (mittels Nullen) -> Options max. 5 Linien

Fragmentation and Reassembly

Max. übertragbare Paketgrösse ist von Netz-Technologie des Data Link Layers abhängig -> Maximum Transfer Unit (MTU) -> Dafür müssen IP-Datagramme aufgeteilt und wieder zusammengesetzt werden. Falls alle Teilnetze gleiche oder grössere MTU haben, so ist keine Fragmentierung notwendig.



Wenn eine Fragmentierung stattfindet, so erhält jedes Fragment einen eigenen IP-Header und wird separat gekapselt und versendet. Nach der Übertragung müssen die Datagramme wieder zusammengesetzt werden (Reassembly). Der IP-Standard legt fest, dass die Fragmente erst beim Zielhost wieder zusammengesetzt werden -> Router werden entlastet & Routen können dynamisch gewählt werden. Der Empfänger kann anhand des Feldes Identification das Reassembly vollziehen.

- Falls Fragment Offset und More Fragments beide = 0 -> Datagramm vollständig
- Falls Fragment Offset (erstes Fragment in Buffer) = 0 wird Header in Header-Buffer kopiert
- Falls More Fragments = 0 -> hinterstes Fragment -> mittels Fragment Offset und Total Lenght Wert des Total-Data-Length-Felds berechnet werden

Das Verfahren wird durch einen Timer überwacht (typischerweise 15s) -> falls nicht komplett wird verworfen

Kapselung eines IP-Datagramms

Zwei Aufgaben: 1. IP-Datagramm in Netzwerk-(Hardware-)spezifisches Fram verpacken
2. logische IP-Adresse in Hardware-Adresse umsetzen

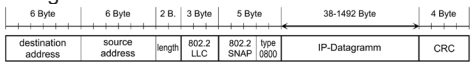


Abbildung 7.28: Kapselung eines IP-Datagramms in einem 802.2/802.3-Frame

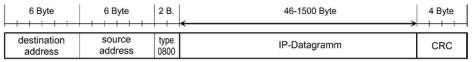


Abbildung 7.27: Kapselung eines IP-Datagramms in einem Ethernet-Frame

Adressauflösung

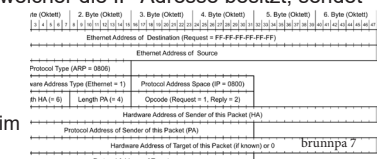
Übersetzung der Protokolladresse in Hardware-Adresse nennt man Adressauflösung und ist immer netzwerkspezifisch. Knoten kann nur Adressen von Knoten im gleichen Netz auflösen -> Bei jedem Netzübergang wird das Datagramm ausgepackt, neu adressiert und wieder gekapselt (relevant für Ziel = Destination Address)

3 Arten:
1. algorithmische Umsetzung (zw. Adressformat und MAC-Adresse algort. Zusammenhang)
2. tabellengesteuert: + vollständige Kontrolle - grosser Aufwand
3. Nachrichtenaustausch: Knoten fordert anderen Knoten auf Informationen zu liefern (zusätzlicher Verkehr)

Address Resolution Protokoll (ARP)

TCP/IP grundsätzliche alle 3 Arten möglich, beste Lösung Mischung aus 2. & 3. Sehr allg. gehalten, hier 4-Byte IP-Adressen auf 6 Byte Ethernet-Adressen

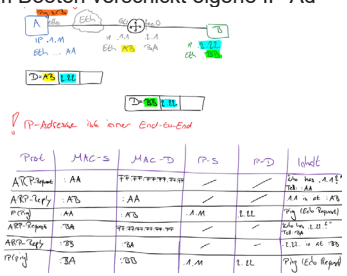
Prinzip: Wenn Knoten von Protokoll zu Hardware-Adresse umsetzen muss, so sendet er via Broadcast-Adresse ein ARP-Request-Paket ("Who has"). Alle Knoten empfangen dies. Der Knoten, welcher die IP-Adresse besitzt, sendet als Antwort ein ARP-Reply-Paket (Unicast) an den fragenden Knoten.



Format: Format für ARP-Request und -Reply prinzipiell gleich. Request -> Ziel-Adresse = Broadcast & Feld "Hardware Address of Target" = 0. Für die IP-Adresse wird die zugehörige MAC-Adresse (Ethernet-Hardware-Adresse) gesucht. ARP-Reply Antwort in Sender-MAC-Adresse zu finden ≈ Source-Adress im Ethernet-Header (nicht zwingend der Fall)

ARP-Cache: Ohne weitere Massnahme, wäre für jede sendendes IP-Datagram das Vorgehen notwendig -> daher führt jeder Knoten ein ARP-Cache, in welchem er die bekannten Protokoll/Hardware-Adresspare einträgt. Nach gewisser Zeit wird die Tabelle wieder gelöscht. => Broadcast-Protokolle werden von Bridges nicht gefiltert, daher kann ARP-Request eine massive Belastung sein.

Gratuitous ARP: überflüssige ARP-Request -> IP-Adresskonflikt zu erkennen beim Booten verschickt eigene IP-Adresse, falls Reply kommt = IP bereits vergeben, falls nicht (zurzeit) kein Konflikt



ARP-Befehle:
Anzeige aller Einträge: *arp -a*
Numerisches Anzeigen aller Einträge (nur UNIX): *arp -an*
Löschen eines Eintrages: *arp -d ip_addr*
Setzen eines Eintrages: *arp -s ip_addr hw_addr*
ARP-Request und Reply auslösen:
arping -c 1 -U 192.168.1.34 oder *arping -c 1 -A 192.168.1.34*

RARP: Wird heute nicht mehr verwendet -> heute DHCP

Internet Control Message Protocol (ICMP)
ICMP wird für die Übertragung von Fehlermeldungen (typw. in Verarbeitung oder Weiterleitung) verwendet.
ICMP-Meldungen werden in IP-Pakete gekapselt IP-Header-Feld Protocol = 1

ICMP-Typ	Bedeutung (Fehler)
3	Destination Unreachable
4	Source Quench
5	Redirect
11	Time Exceeded
12	Parameter Problem

Fehlermeldungen:
- *Destination Unreachable (3)*: Zielhost oder Netz nicht erreichbar. -> Type-Feld immer 3; Code-Feld gibt Grund der Unzustellbarkeit an; Checksum-Feld: Header in Worte (16Bit) zerlegt und mit dem Einerkomplement invertiert wird, danach addiert, wobei Checksum = 0. Die nochmals invertierte Summe (16Bit) ergibt Prüfsumme

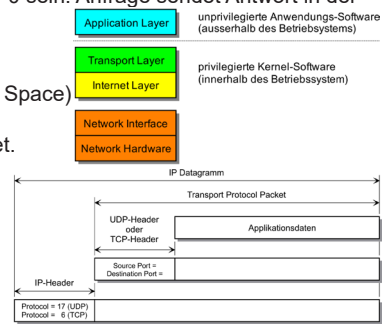
ICMP-Typ	Bedeutung (Information)
0	Echo Reply
8	Echo
13	Timestamp
14	Timestamp Reply
15	Information Request
16	Information Reply

- *Source Quench (4)*: So viele Datagramme im Puffer, dass Speicher erschöpft
->Host aufgefordert Rate zu reduzieren
- *Redirect (5)*: Sendung an entferntes Netz, sendet an Router das an Ziel befördert oder an falschen Router gesendet und Host wird aufgefordert, angebene Route zu ändern (dieser Fall ist üblich).
- *Time Exceeded (11)*: zwei Fälle: 1. TTL = 0 & 2. Wenn Timer abläuft für Reassembly
-> Type-Feld immer 11; rest analog Destination Unreachable
- *Parameter Problem (12)*: Router oder Host findet im IP-Header ungültigen Wert, der nicht verarbeitet werden kann

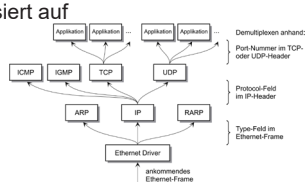
Informationsmeldungen:
- *Echo Request (8) / Reply (0)*: Request kann an ICMP-Software jedes Rechners gesendet werden, darauf folgt eine Reply-Sendung -> Antwort: gleiche Daten wie Anfrage
-> Type-Feld 8 = Echo 0 = Reply; Identifier = Bezug zur Echo; Code-Feld = 0; Checksum analog oben
- *Timestamp Request (13) / Reply (14)*: analog Echo + zusätzlich geben Sender / Empfänger ihre Zeit (ms seit Mitternacht) mit
- *Information Request (15) / Reply (16)*: Sende-&Empfangsadresse kann = 0 sein. Anfrage sendet Antwort in der die Nr. und die 32-Bit Maske des benutzenden Subnetzes enthalten ist.

4 Transport Layer

Der Transport Layer bildet die Schnittstelle zwischen Betriebssystem (Kernel Space) und Applikationen / Anwendung (User Space)
Die Applikationsdaten werden mit einem UDP- bzw. TCP-Header ausgestattet.
-> Transport Protocol Paket
UDP = User Datagram Protocol
TCP = Transmission Control Protocol
Danach wird die Applikationsdatei mit einem IP-Header versehen
-> "Protocol-Feld" steht ob UDP oder TCP



Multiplexen und Demultiplexen
Eine wichtige Aufgabe des Transport Layers ist das Verteilen von eintreffenden Daten auf die vers. Applikationen.Dieser Vorgang nennt man Demultiplexen und basiert auf dem Destination Port des Transport Headers.
-> logischen I/O-Kanel und nicht zu verwechseln mit einem hardwaremässigen



User Datagram Protocol (UDP)
UDP ist ein verbindungsloses, unzuverlässiges Transport-Protokoll und dient im Wesentlichen für Multiplexen und Demultiplexen auf Applikation

UDP-Header:
- UDP Source Port und Destination Port *(je 16Bit)*: Enthalten ein Word zum Verteilen der eintreffenden Datagramme. Der Source Port kann optional verwendet um dem Empfänger mitzuteilen an welchen Port die Antwort erfolgen soll.
- UDP Message Length *(Min. 8 Byte)*: Gibt die Länge (in Bytes) des Transport-Portocol-Pakets an. Dieser besteht aus dem UDP-Header und den Applikationsdaten.
- Checksum: dient zur Fehlererkennung. Zuerst wird ein Pseudo-Header gebildet. Dieser beinhaltet IP-Source- & Destination Address, Protocol als Felder des IP-Headers + die Länge des UDP-Datagramms. Die Berechnung erfolgt nun mit dem Pseudo-Header, UDP-Header und den Daten -> Einerkomplement der Summer aller Einerkomplemente aller 16-Bit-Wörter gebildet wird. Statt Prüfsumme kann auch null angegeben werden -> Zeiteinsparung bei zeitkritschischen Anwendungen

Protokoll	Port	Beschreibung
	0	Reserved
echo	7	Echo
discard	9	Discard
daytime	13	Daytime
time	37	Time
domain	53	Domain Name Server
tftp	69	Trivial File Transfer
www-http	80	World Wide Web HTTP
pop3	110	Post Office Protocol - Version 3
ntp	123	Network Time Protocol
snmp	161	SNMP
xdmcp	177	X Display Manager Control Protocol
imap3	220	Interactive Mail Access Protocol v3
who	513	maintains data bases showing who's
timed	525	timeserver

Ports:
- *Ports 0 - 1023*: System Port oder Well Known Ports und sind für bekannte Applikationen fest reserviert (durch IANA)
- *Ports 1024 - 49151*: User Ports->neue Nr. auf Antrag ohne Beteiligung IETF zugeordnet
- *Ports 49152 - 65535*: Dynamic Port und können frei verfügt werden

Beispiel Anwendungen: DNS, DHCP, NTP

Transmission Control Protocol (TCP)
Hauptaufgabe ist die Bereitstellung einer zuverlässigen Übertragung. TCP muss hierfür Verluste und Verzögerungen im Internet ausgleichen, dabei soll weder das Subnetz noch der Router überlasten.
-> Für die meisten Internet-Anwendungen wird TCP verwendet.

Merkmale von TCP:
- *Verbindungsorientierte Übertragung*: Die Anwendung muss zuerst eine Verbindung aufbauen, bevor es Daten übertragen kann. **Jede TCP-Verbindung hat genau zwei Endpunkte** (keine Multicast-Verbindungen)
- *Hohe Zuverlässigkeit*: TCP garantiert die Übertragung der Daten **ohne Datenverlust** in der richtigen Reihenfolge
- *Vollduplexübertragung*: In einer TCP-Verbindung können die **Daten in beide Richtungen fliessen,** so dass beide Seiten jederzeit Daten senden und empfangen können
- *Stream Schnittstelle*: Die Anwendnung sendet oder empfängt eine fortlaufend, unstrukturierte Bytefolge über die Verbindung. Die Bytes treffen in der **gleichen Reihenfolge** wie versendet beim Empfänger ein
- *Zuverlässiger Verbindungsaufbau*: TCP fordert von zwei kommunikationswilligen Anwendungen die **Vereinbarung über eine Verbindung**. frühere Verbindungen (allenfalls Duplikate) werden nicht akzeptiert
- *Eleganter Verbindungsaufbau*: Verbindung aufbauen -> Daten austauschen -> Abbau der Verbindung. TCP gewährleistet die **zuverlässige Zustellung** aller Daten in beide Richtungen auch beim Abbau der Verbindung.

Ende-zu-Ende-Dienst
TCP ist ein End-to-End-Dienst da es die Verbindung direkt von der Anwendung zum entfernten Host bietet. Die Verbindung wird als virtuell angesehen, da sie von der Software hergestellt wird. Internet bietet weder Hardeware-, noch Software- Unterstützung für die Verbindung -> zwei Hosts tauschen Nachrichten aus um Verbindung zu schaffen.
TCP verwendet IP zur Beförderung von Daten. Kommt das Datagramm beim Zielhost an, gibt IP den Inhalt an TCP ab. -> IP weder liest noch interpretiert die Daten.
Aus Sicht des TCPs ist das Internet ein Kommunikationssystem (virtuelles Netz), in dem Nachrichten transparent ausgetauscht werden, ohne das der Inhalt geändert oder interpretiert wird.

Zuverlässigkeit
Es wäre möglich, dass eine Nachricht der ersten Verbindung dupliziert wird und erst nach dem Aufbau der zweiten Verbindung eintrifft. Nachrichten müssen daher eindeutig einer Verbindung zugeordnet werden können. Weitere grosse Herausforderung ist das Ein-/Ausschalten von Hosts.
Durch die Neuübertargung (Retransmission) von verlorenen Daten, erreicht TCP die Zuverlässigkeit. Bei der Datenübertragung startet der Sender ein Times, trifft das fehlerfreie Segment beim Empfänger ein sendet dieser eine Bestätigung (Acknowledgment). Läuft der Timer ab, so sendet der Sender das Segment (unaufgefordert) nochmals. Falls das Segment fehlerhaft ist, erfolgt keine Rückmeldung des Empfängers an den Sender.
Die Bestätigung in einem LAN kann innerhalb von ms erwartet werden. Über Satellit, kann dies Sekunden sein. Die zum Senden einer Nachricht und Empfangen einer Bestätigung benötigte Gesamtzeit wird Umlaufverzögerung, Round-Trip Time oder Round-Trip Delay genannt. Die Wartezeit bis zur Neuübertragung wird als Retransmission Time-Out bezeichnet.
-> Ziel ist es ausreichend lange zu warte, dass nur Paketverlust zu Timeout führt und unnötige Wartezeit vermieden wird. Neben der Unterscheidung von lokalen und entfernten Ziele muss TCP mit den Verkehrsbedingungen im Netz umgehen können.

Berechnung Retransmission Time-Out:
TCP misst bei jeder aktiven Verbindung die Round-Trip TIme (RTT_n) und bildet daraus einen gewichteten Mittelwert SRTT (Smoothed Round-Trip Time)

$$SRTT_n = (1-\alpha) * SRTT_{n-1} + \alpha * RTT_n \qquad \alpha=0.125$$

Zusätzlich ermittelt TCP die Streuung RTTV AR durch einen gewichteten Mittelwert der Abweichung davon

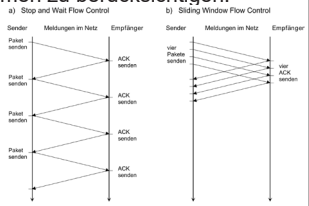
$$RTTV\ AR_n = (1-\beta) * RTTV\ AR_{n-1} + \beta * | SRTT_n - RTT_n | \qquad \beta=0.25$$

Der Retransmission Time-Out (RTO) ist eine Kombination aus der Umlaufverzögerung un dem Mehrfach der Streuung

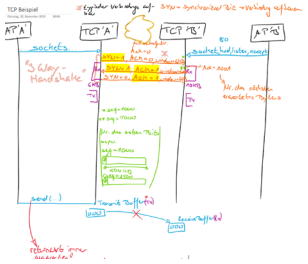
$$RTO_n = SRTT_n + 4 * RTTV\ AR_n$$

Erfahrungen haben gezeigt, dass die adaptive Neuübertragung von TCP gut funktioniert. Bei grosser Streuung setzt TCP den RTO auf einen deutlich über dem Mittel liegenden Wert, um Spitzenaufkommen zu berücksichtigen.

Fluss-Steuerung (Flow-Control)



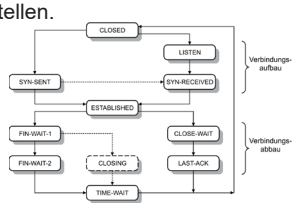
Nicht alle Computer laufen in der gleichen Geschwindigkeit. Wenn ein Computer in einem Netz Daten schneller sendet, als das Ziel aufnehmen kann entsteht ein Datenüberlauf (Data Overrun) -> Datenverlust
Die Fluss-Steuerung (Flow Control) fasst versch. Techniken zusammen, die das löst:
1. Stop-and-Wait-Verfahren: Der Sender wartet immer auf die Quittung des Empfängers.-> Entsteht zwar kein Data Overrun, jedoch wird die Netzbandbreite nur sehr ineffizient genutzt
2. Sliding Window: Sender und Empfänger werden auf eine feste Fenstergrösse programmiert -> max. Datenmenge die vor Ankunft einer Bestätigung gesendet werden kann.



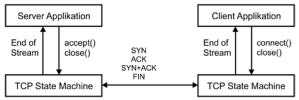
TCP-*Fluss-Steuerung* -> Sliding Window Mechanismus
Beide Seiten benötigen einen Buffer für eingehende Daten. Die zu einem Zeitpunkt verfügbare Anzahl Bytes des Buffers ist bei TCP das Window.
-> Window Advertisement
Sender wir bei Empfang bestätigt und gleichzeitig der aktuelle Stand des Window mitgeteilt.
Wenn Empfänger so schnell arbeiten kann wie Sender und nach mag, zeigt er eine positive Window-Grösse an. Wenn Sender schneller ist, füllt sich sein Buffer was ihn veranlasst ein Window von null (Zero Window) anzuzeigen. -> Sender wartet nun bis er (unaufgefordert) ein positives Window erhält.

Zuverlässiger Verbindungsaufbau und -abbau

Für die Gewährleistung der Zuverlässigkeit benützt TCP Dreiweg-Handshaking (3 Way Handshake) -> mind. drei Nachrichten müssen ausgetauscht werden, um eine eindeutige Verbindung sicherzustellen.

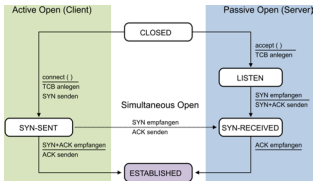


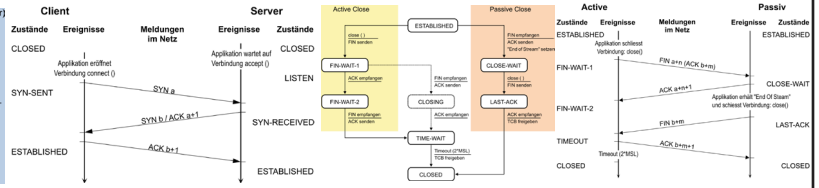
CLOSED Verbindung existiert noch nicht (pränatale Ruhezustand)
LISTEN passiver Zustand Server, wartet auf Verbindungsanforderung des Client
SYN-SENT Client Verbindungsanforderung geschickt, wartet auf Bestätigung
SYN-RECEIVED Server wartet auf Bestätigung seiner Verbindungsbestätigung
ESTABLISHED Verbindung steht (Daten können gesendet / empfangen werden)
FIN-WAIT-1 lok. Appl. will Verbindung schliessend. Abbauanforderung geschickt wartet auf Bestätigung des Remote-Endes
FIN-WAIT-2 Abbauanforderung bestätigt. warten bis Remote-Ende ready für Abbau ist
CLOSE-WAIT Remote-Ende will Verbindung abbauen. warten bis lok. Appl. ready für Abbau ist
LAST-ACK Verbindungsabbau wurde bestätigt. Wird auf Bestätigung der Bestätigung gewartet
CLOSING Remote-Ende gleichzeitigi Verbindungsabbau verlangt, muss nurnoch auf Bestätigung gewartet werden
TIME-WAIT letzte Bestätigung wurde gesendet. Wird fixe Zeit gewartet, dass bei einem Verlust nochmals gesendet werden kann, bevor Verbindung endgültig abgebrochen wird.



Auf beiden Seiten (Server & Client) steht eine unabhängige TCP State Machine (SM), welche durch Events beeinflusst werden. Die beiden TCP SM signalisieren Events mit Meldungen. Diese sind normale TCP Segmente, die im Header die Flags SYN, FIN und/oder ACK gesetzt haben.

Verbindungsaufbau:
Nachrichten (SYN-Segmente) die für den Verbindungsaufbau benutzt werden hab das SYN-Flag im Header gesetzt. Das SYN-Segment wird mit einem ACK-Segment bestätigt und bei Verlust nochmals gesendet. Für den Verbindungsaufbau wird der 3 Way Handshake verwendet. Server und Client legen vor dem eigentliche Verbindungsaufbau eine TCP-spezifische Datenstruktur an, dem Transmission Control Block (TCB). Dabei werden sämtliche wichtige Informationen für die Verbindung übermittelt (Port-Nummer, Pointer zu Sende-/Empfangs-Buffer, Sequenz- / Acknowledge Nummer). Für ein eindeutige Identifkation generieren beide eine 32-Bit Sequenznummer, diese Sequenznummer ist für die Verbindung eindeutig. Alle weiteren Verbindungen haben anderen Sequenznummern.

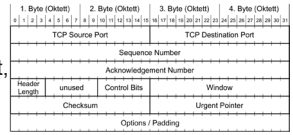




Verbindungsabbau:
Um eine Verbindung abzubauen, schickt eine der beiden Seiten ein FIN-Segment. Nun hat die gesendete Seite ihre ausgehende Verbindung geschlossen, daher ist der aktuelle Zustand half closed (die andereSeite kann noch beliebig Daten senden). Die Ressourcen (TCB) dürfen erst freigegebenwerden, wenn die Verbindung von beiden Seiten geschlossen ist. -> Handshake-Verfahren notwendig. Die Wartezeit beträgt 2-mal die Maximum Segment Lifetime (MSL), was typischerweise 4 Minuten entspricht

TCP Header

TCP verwendet für alle Segmente dasselbe Header-Format. TCP kann in einem einzelnen Segment die Bestätigung für die Eingangsdaten, die Fensteranzeige und Ausgangsdaten übertragen.



- TCP Port und Destination Port: definiert welches Anwendungsprogramm auf dem empfangenden Host die Daten erhalten soll. Der Source Port bezeichnet das sendende Anwendungsprogramm. Falls das Anwendungsprogramm einem Server-Prozess gehört, bezeichnet der Port den entsprechenden Dienst
- Sequenz Number: bezieht sich auf Ausgangsdaten. Gibt SequenzNr. der im Segment enthaltenen Daten an. Empfänger ordnet anhand der SequenzNr. die Segmente, die ausser der Reihe ankommen und benutzt sie zur Berechnung der AcknowledgmentNr.
- Acknowledgment Number: bezieht sich auf Eingangsdaten. AckknowledgmentNr. definiert die SequenzNr. der zu empfangenden Daten
- Data Offset: gibt an wo die Daten beginnen und wo der TCP-Header (inkl. optionale Felder)zuende ist. Der Wert entspricht der Grösse des ganzen TCP-Headers in Doppelwörtern (32-Bit)
- Control Bits: bezeichnet eine Reihe von Flags. unter anderem den Verbindungsaufbau & -abbau oder Gültigkeit des Headers.

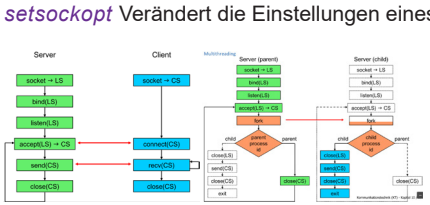
10	11	12	13	14	15
URG	ACK	PSH	RST	SYN	FIN

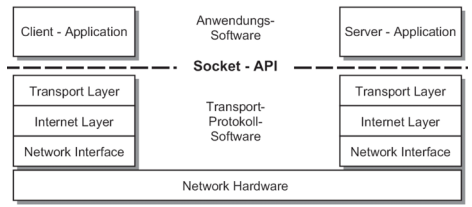
Control Bits	Bedeutung
URG	Urgent-Pointer-Feld enthält gültigen Wert.
ACK	Acknowledgment Feld enthält gültigen Wert.
PSH	Push: Empfänger soll Daten sofort an die Applikation weiterleiten.
RST	Reset: Verbindung rücksetzen.
SYN	Synchronize: Verbindung aufbauen.
FIN	Sender hat keine weiteren Daten zu übertragen.

- Window: zeigt die noch verfügbare Buffer-Grösse an: Wie viele weitere Daten (Bytes) ist der Empfänger moment bereit zu empfangen
- Checksum: dient zur Fehlererkennung. Zuerst wird ein Pseudo-Header gebildet. Dieser beinhaltet IP-Source- & Destination Address, Protocol als Felder des IP-Headers + die Länge des UDP-Datagramms. Die Berechnung erfolgt nun mit dem Pseudo-Header, UDP-Header und den Daten -> Einerkomplement der Summer aller Einerkomplemente aller 16-Bit-Wörter gebildet wird.
- Urgent Pointer: Falls URG-Bits gesetzt, gibt das Feld an, wo in den Daten Informationen mit hoher Prio enthalten sind
- Options: Oft bei Verbindungsaufbau verwendet -> Bspw. Angabe der maximalen Segmentlänge (MSS), Multiplikationsfaktor für die Window-Size oder Selective Acknowledge (SACK) etc.

Schnittstelle zum Transport-Layer (Sockets)

Die Schnittstelle zwischen einer Anwendung und der Transport Protokoll-Software heiss Application Program Interface. Eine API definiert eine Menge von Prozeduren und Datentypen, welche die Anwendung verwenden kann. Die Operationen, die eine Anwendung bei der Interaktion mit Protokoll Software ausführen kann, bestimmen im wesentlichen die Funktionalität, die einer Anwendung zur Verfügung steht, sowie den Komfort um mit Hilfe dieser Funktionalität eine Anwendung zu erstellen.
socket erzeugt Socket-Descriptor(16-Bit Int Wert), der im folgenden für die Netzkommunikation verwendet wird
connect Verbindet einen Client mit einem Server-Dienst
write Sendet Daten über eine Netzwerk-Verbindung
read Liest Daten über eine Netzwerk-Verbindung
close / shutdown Schliesst eine Netzwerk-Verbindung
bind Verknüft ein Socket mit einer lokalen IP-Adresse und einer Port-Nummer
listen Setzt ein Socket eines Servers in den passiven (Warte-)Zustand und bestimmt die Grösse der Warteschlange
accept (Server) wartet auf die nächste Verbindung eines Clients
recv Liest das nächste Datagramm von einer Netzwerk-Verbindung
send Sendet ein Datagramm über eine Netzwerk-Verbindung
getsockopt Liefert die aktuellen Einstellungen eines Sockets
setsockopt Verändert die Einstellungen eines Sockets





Hilftestellungen für Umrechnungen

Sekunden

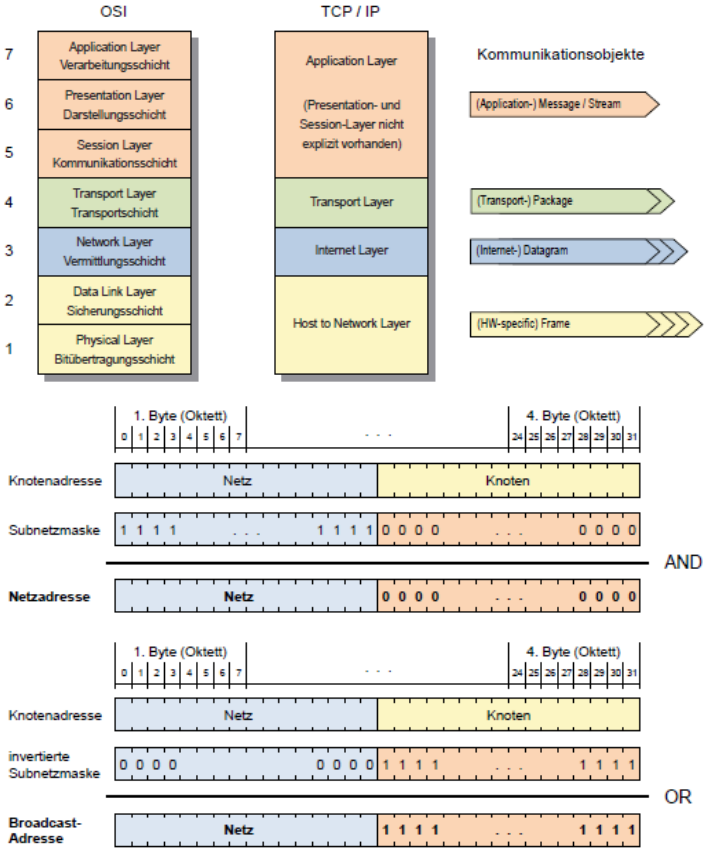
Präfix	Bezeichnung	Dezimal	Wissenschaftlich
1 s	Sekunde	1 s	1
1 ms	Millisekunde	0.001 s	$1 * 10^{-3}$
1 µs	Mikrosekunde	0.000 001 s	$1 * 10^{-6}$
1 ns	Nanosekunde	0.000 000 001 s	$1 * 10^{-9}$

Bit / Byte (immer *1000)

Name	Symbol	Wert
Kilobit/byte	kbit / byte	10^3
Megabit/byte	Mbit /byte	10^6
Gigabit/byte	Gbit/byte	10^9
Terabit/byte	Tbit/byte	10^{12}

Bit in Byte

Byte	Wiss.	Bit
1	1	8
2	2^1	16
4	2^2	32
8	2^3	64
16	2^4	128
32	2^5	256
64	2^6	512
128	2^7	1024
256	2^8	2048
512	2^9	4096
1024	2^{10}	8192
2048	2^{11}	16'384
4096	2^{12}	32'768



Bit in Byte

A	B	AND	A	B	OR
0	0	0	0	0	0
0	1	0	0	1	1
1	0	0	1	0	1
1	1	1	1	1	1

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL