FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Technische Fakultät — Lehrstuhl für Informatik 7 · Rechnernetze und Kommunikationssysteme

Skript der Veranstaltung

Rechnerkommunikation (RK)

gehalten im Sommersemester 2018 von
Prof. Dr. Reinhard German

Vorbermerkung

Dieses Skript enthält den Inhalt der Vorlesung „Rechnerkommunikation“ des Sommersemester 2018 bei Prof. Dr. Reinhard German. Es wurde anhand von Mitschriften in LaTeX gesetzt und erhebt daher weder Anspruch auf Korrektheit noch Vollständigkeit und ist ***offensichtlich inoffiziell***. Bei Unstimmigkeiten und evtl. vorhandenen Fehlern bitte ich um eine Email an untenstehende Adresse. Dieses Skript stellt damit insbesondere **keine** offizielle Veröffentlichung des Lehrstuhl für Rechnernetze und Kommunikationssysteme am Department Informatik der Friedrich-Alexander-Universität Erlangen-Nürnberg dar.

Florian Frank — florian.ff.frank@fau.de
Version vom 2. April 2019

Literaturvorschläge

[Cui+17] Y. Cui u. a. „Innovating Transport with QUIC: Design Approaches and Research Challenges“. In: *IEEE Internet Computing* 21.2 (März 2017), S. 72–76. ISSN: 1089-7801. DOI: 10.1109/MIC.2017.44.

[Gmb19] DATACOM Buchverlag GmbH. *ITWissen.info - Technologiewissen online*. 2019. URL: https://www.itwissen.info/.

[KR17] James F. Kurose und Keith W. Ross. *Computer networking a top-down approach*. seventh , global. 2017, 852 Seiten. ISBN: 978-1-292-15359-9.

[LW04] Alberto Leon-Garcia und Indra Widjaja. *Communication networks fundamental concepts and key architectures*. 2. McGraw-Hill series in computer science. 2004, XXVII, 900 S. graph. Darst. ISBN: 0-07-246352-X.

[PD12] Larry L. Peterson und Bruce S. Davie. *Computer networks a systems approach*. 5th ed. 2012, xxxi, 884 pages. ISBN: 978-0-12-385059-1. URL: http://www.sciencedirect.com/science/book/9780123850591.

[Sch10] Jürgen Scherff. *Grundkurs Computernetzwerke eine kompakte Einführung in Netzwerk- und Internet-Technologien*. 2., überarb. und erw. Aufl. Studium. 2010, XIX, 444 S. ISBN: 978-3-8348-0366-5.

[Sch04] Bernd Schürmann. *Grundlagen der Rechnerkommunikation technische Realisierung von Bussystemen und Rechnernetzen ; für alle Studiengänge: Informatik, Elektrotechnik und Informationstechnik*. 1. Aufl. 2004, XIV, 361 S. ISBN: 3-528-15562-0.

[Sta14] William Stallings. *Data and computer communications*. 10. Always learning. 2014, 907 S. ISBN: 978-1-29-201438-8.

[TW14] Andrew S. Tanenbaum und David Wetherall. *Computer networks*. Pearson new international , fifth. Always learning. 2014, II, 804 Seiten. ISBN: 978-1-292-02422-6.

Inhaltsverzeichnis

KAPITEL 1

EINFÜHRUNG

Der Beginn der Rechnerkommunikation war in den 60er Jahren mit den ersten Konzepten von paketvermittelten Datennetzen für militärische Zwecke. Die Idee des *Internetworkings* kam dann in den 70er Jahren auf, zusammen mit ARPAnet und lokalen Netzen, welche auf Zufallszugriffen basierten. Die Nutzung im akademischen Bereich gewann Überhand und erzeugte Entwicklungen von Protokollen wie TCP/IP, SMTP, DNS, FTP in den 80er Jahren. Ein Jahrzehnt später wurden populäre Anwendungsprotokolle wie HTTP entwickelt und Webbrowser verbreitet. Jetzt wird auch Sicherheit im Netz zum ersten Mal zentrales Thema. Erste Entwicklungen zu drahtlosen Netzen finden statt. In den 2000er Jahren kam es zur großen *Dotcom-Blase* und die Netzwerkkommunikation wuchs immer weiter. Ebenso kamen weitere Anwendungen auf, beispielsweise die Internettelefonie, P2P-Systeme, soziale Netzwerke oder *cloud computing*. Drahtlose Netze wie Bluetooth und ZigBee wurden eingeführt und DSL und drahtloses LAN wurde ausgebaut und verbreitet. In den 2010er Jahren kommt es nun zu einem *internet of things* (Internet der Dinge, IoT), 5G und LTE Ausbau, sowie Software Defined Networking. Die Kommunikation ist also immer noch ein großer Bestandteil von Forschung und Innovation.

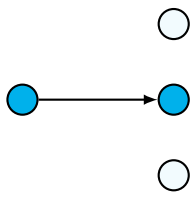
1.1 Klassifikation von Kommunikationssystemen

Der Grundsatz für eine jegliche Kommunikation in Systemen sind Kommunikationssysteme. Diese unterscheiden sich in gewissen Merkmalen, welche wir näher betrachten wollen. Zu diesen Merkmalen zählen unter anderem Kommunikations- und Übertragungsarten, Übertragungsmedien, Entfernungen, Bitraten und die Topologie des Netzwerkes.

1.1.1 Kommunikationsarten

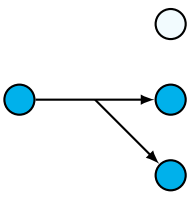
Unicast
(Punkt zu Punkt)

Ein Sender sendet Informationen zu einem Empfänger



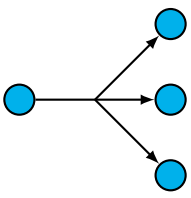
Multicast
(Punkt zu Mehrpunkt, Gruppenruf)

Ein Sender sendet Informationen zu einer Gruppe von Empfängern



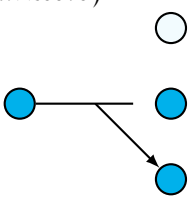
Broadcast
(Rundruf)

Ein Sender sendet Informationen an **alle** Teilnehmer des Netzes



Anycast

Ein Sender sendet Informationen an einen Empfänger aus einer Gruppe möglicher Ziele (*die Verbindung ist hierbei die kürzeste*)

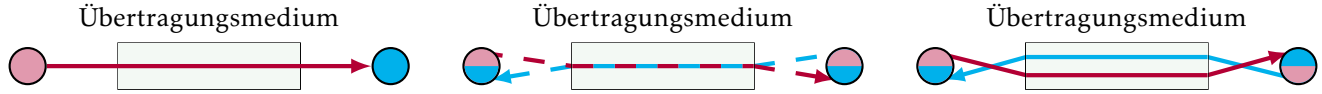


1.1.2 Übertragungsarten

Für die physikalische Anordnung von Datenverbindungen und den logischen Informationsfluss darüber bestehen in Abhängigkeit von den Übertragungssystemen und den gewünschten Anwendungen verschiedene Übertragungsarten, welche wir im Folgenden näher beleuchten wollen.

Übertragungsrichtung Ein wichtiger Bestandteil ist die Übertragungsrichtung. Sie bestimmt welche Kommunikationspartei zu welchem Zeitpunkt Informationen über den Kanal senden und erhalten kann. Wir unterscheiden drei Varianten: Simplex-, Halbduplex- und Duplex-Betrieb.

simplex	**halbduplex**	**(voll-)duplex**
unidirektionale Verbindung	bidirektionale Verbindung mit Umschalten, Kommunikation in beide Richtungen **nicht** gleichzeitig möglich	bidirektionale Verbindung bei der gleichzeitige Kommunikation in beide Richtungen möglich ist



Multiplexverfahren Es sollte ersichtlich sein, dass ein rein sequentieller Betrieb bei geteilten physischen Verbindungen im Allgemeinen keine sinnvolle Idee ist. Mittels Multiplexverfahren schafft man nun Abhilfe insoweit, dass man mehrere logische Verbindungen auf einer physischen Verbindung zulässt. Wir unterscheiden zwischen Frequenz- und Zeitmultipliexverfahren.
Bei ersterem verwenden unterschiedliche Geräte (logische Verbindungen) einen bestimmten, ihnen zugeordneten, Teil des Frequenzspektrums. Man teilt damit ein breites Frequenzband in mehrere schmälere Bänder auf, bei denen ein jedes einem eigenen Übertragungskanal entspricht.
Bei letzterem schafft man reine Pseudoparallelität ähnlich dem Zeitscheibenschedulingverfahren, welche aus vergangenen Vorlesungen bekannt sein sollten. Man teilt die Zeit in mehrere fest definierte Multiplexrahmen auf, welche für jeden Übertragungskanal einen festen Zeitschlitz beinhalten. Die einzelnen Zeitschlitze werden dann **immer noch sequentiell** abgearbeitet, die Geräte wechseln sich somit ab und es entsteht bei kleiner Rahmengröße Pseudoparallelität.

Vermittlungsarten Ebenso wichtig ist die Art und Weise wie ein Übertragungspfad nun zwischen Sender und Empfänger vermittelt wird. Wir unterscheiden zwischen der **Leistungs-** und **Paketvermittlung**.
Bei der **Leistungsvermittlung** wird zwischen Sender und Empfänger eine physikalische Leitung aufgebaut. Dies kann beispielsweise durch Zeit- oder Frequenzmultiplexverfahren geschehen. Die somit zur Verfügung stehende Bitrate muss fest auf die Kanäle aufgeteilt werden. Die Leistungsvermittlung ist per se verzögerungsfrei und liefert volle Transparenz gegenüber allen Protokollen der höheren Schichten. Allerdings gibt es keine Ersatzwege im Fehlerfall, sowie lange Wartezeiten auf Verbindungsauf- sowie -abbau.
Bei der **Paketvermittlung** sendet der Sender die Daten in Paketen, welche einzeln zum Empfänger gelangen. Die Übermittlung dieser findet dann entweder verbindungslos (als Datagramm) oder über eine logische Verbindung statt. Die Pakete können aufgrund ihrer Souveränität in den Vermittlungsknoten zwischengespeichert werden, wodurch ein effizienteres Aufteilen der Bitrate sowie ein Abfangen kurzfristig höherer Datenaufkommen über Puffer ermöglicht wird. Gleichzeitig kann es hierbei allerdings zu (starken) Verzögerungen kommen.

1.1.3 Übertragungsmedium

Wir kommen nun zu einem Punkt des Mediums über welches Informationen übermittelt werden. Hierbei können selbstverständlich feste wie auch flüssige oder gasförmige Materialien verwendet werden. Die Informationen werden dann mit einem Informationsträger wie beispielsweise Druckwellen, Spannungs- oder Lichtimpulsen oder elektromagnetischen Wellen übertragen. Wir wollen leitungsgebundene und drahtlose Übertragungsmedien näher betrachten.
Bei **leitungsgebundenen Übertragungsmedien** ist das Medium eine physische Verbindung über ein Kabel. Man unterscheidet noch zwischen metallischen und nichtmetallischen Leitern. Die Bitraten reichen von einigen Kbps bis hin zu mehreren Gbps, wobei dies von den verwendeten Kabeln abhängt. Die Signalausbreitungsgeschwindigkeit ist ein bisschen geringer als die Lichtgeschwindigkeit. Zudem treten nur kleine Bitfehlerraten auf, bei Glasfaserkabeln ist dies beispielsweise in der Größenordnung 10^{-10}. Beispiele sind symmetrische Kupferkabel wie zum Beispiel die Kupferdoppelader (CuDa), welche besonders im Anschlussbereich eingesetzt werden, sowie Koaxialkabel (COAX) oder verdrillte Kupferkabel. Ebenso leitergebunden sind auch Lichtwellenleiter in Form von Glas- oder Plastikfasern.
Bei **drahtlosen Übertragungsmedien** werden Informationen beispielsweise über Schall, Funktechnik, Infrarot oder sichtbares Licht übertragen. Wegen verschiedener Probleme bei der Ausbreitung von Funkwellen sind die Bitfehlerraten vergleichsweise hoch und reichen von Größenordnungen 10^{-5} bis 10^{-2}. Zudem treten Bitfehler hierbei des Öfteren am Stück in Schüben auf (diese nennt man dann *burst*).

1.1.4 Entfernung

Ein weiteres Unterscheidungsmerkmal sind die Entfernungen zwischen einzelnen Kommunikationsknoten. Wir unterscheiden lokale, Stadt- sowie Weitverkehrnetze.
Bei **lokalen Netzen (LAN)** handelt es sich um Systeme für den Hochleistungsinformationstransfer, welche es einer Anzahl *gleichberechtigter Benutzer* ermöglichen auf einem *räumlich begrenzten Gebiet* partnerschaftlich orientierten Nachrichtenaustausch hoher Güte durchzuführen. Die Ausdehnung ist üblicherweise höchstens 10 km, es mag aber auch LANs geben mit höherer Ausdehnung. Es können hierbei beispielsweise Ausbreitungsverzögerungen von ungefähr $2^1/_2 \ {}^{km}/_{v} \approx 12^1/_2 \ \mu s$ erreicht werden. LANs können drahtgebunden – wie standardisierte lokale Netze Ethernet, Token Ring und FDDI – als auch drahtlos – wie WLANs nach 802.11 – arbeiten.
Bei **Stadtnetzen (MAN)** handelt es sich um Stadtnetze oder Regionalnetze, die sich mit ihren charakteristischen Eigenschaften zwischen den lokalen Netzen (LAN) und den Weitverkehrsnetzen (WAN) positionieren. Bei MANs handelt es sich dabei um Netze mit einer regionalen Ausdehnung, die für unterschiedlichste Übertragungsdienste wie Sprache, Daten, Bewegtbild usw. ausgelegt sind. Die Ausdehnung für ein Stadtnetz liegt bei 100 km und mehr, die Übertragungsgeschwindigkeit bei 100 Mbit/s bis 1.000 Mbit/s. MANs benutzen Lichtwellenleiter als Übertragungsmedium, um die hohen Übertragungsgeschwindigkeiten mit möglichst geringer Fehlerrate zu realisieren.
Bei **Weitverkehrnetzen (WAN)** handelt es sich um Netzwerke, welche für Daten- oder Sprachübertragungen über eine weite Strecke konzipiert sind. Ihnen ist eine Ausdehnung von 1000 km oder auch 10000 km möglich, wobei die Ausbreitungsverzögerung beispielsweise bei 50 ms liegen kann. WANs können kabelgebunden wie auch drahtlos sein.

1.1.5 Bitrate

Wir unterscheiden nun Kommunikationsnetzwerke bezüglich ihrer Übertragungsgeschwindigkeit beziehungsweise ihrer Bitrate. Hierbei handelt es sich um die Anzahl der Bits, welche pro Zeiteinheit übertragen werden. Zusammen mit der Ausbreitungsverzögerung $D = {}^{l}/_{v}$ (l als Länge der Verbindung und v als Ausbreitungsgeschwindigkeit) ergibt sich somit das Datenvolumen auf

einer Kommunikationsstrecke (*Kanalpuffergröße*) durch

$$V_{\text{Data}} = RD = R \cdot D = R \cdot \frac{l}{v} = \frac{D}{1/R} = \frac{l/v}{1/R} = \frac{\textit{Ausbreitungsverzögerung}}{\textit{Bitsendezeit}}.$$

Wir sehen somit, dass die Bitrate in Kombination mit der Fehlersicherung und dem Medienzugriff wichtig für die Effizienz eines Netzwerks ist.

Kanalpuffergröße bei Leistungsvermittlung Die Kanalpuffergröße in Bits gibt die Anzahl an **bereits gesendeten** Bits an, während sich das erste Bit vom Sender zum Empfänger ausbreitet. Man erhält somit auch obige Formel. Ist $RD > 1$, so erreicht zum Zeitpunkt $t = D$ das erste Bit den Empfänger, wobei $R \cdot D$ Bits bereits gesendet sind. Ist andererseits $RD < 1$, so erreicht zum Zeitpunkt $t = D$ der Anfang des ersten Bits den Empfänger, und es sind $R \cdot D \cdot 100\%$ des ersten Bits bereits versendet. Erst zum Zeitpunkt $t = 1/R$ verlässt das Ende des ersten Bits den Sender, dieses erreicht den Empfänger dann zum Zeitpunkt $t = 1/R + D$.

Kanalpuffergröße bei Paketvermittlung Die Kanalpuffergröße in Paketen gibt die Anzahl an **bereits gesendeten *Paketen*** an, während sich das erste Bit vom Sender zum Empfänger ausbreitet. Es ändert sich mit einer Paketgröße L:

$$a = \frac{R \cdot D}{L} = \frac{l/v}{L/R} = \frac{\textit{Ausbreitungsverzögerung}}{\textbf{\textit{Paket}}\textit{sendezeit}}$$

Ist $a > 1$, so erreicht zum Zeitpunkt $t = D$ das erste Bit den Empfänger, wobei a Pakete bereits gesendet sind. Ist andererseits $a < 1$, so erreicht zum Zeitpunkt $t = D$ der Anfang des ersten Bits den Empfänger, und es sind $a \cdot 100\%$ des ersten Pakets bereits versendet. Erst zum Zeitpunkt $t = L/R$ verlässt das Ende des ersten Pakets den Sender, dieses erreicht den Empfänger dann zum Zeitpunkt $t = L/R + D$.

1.1.6 Topologie

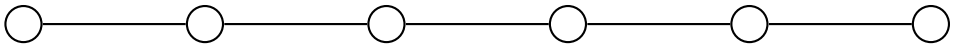
In Kommunikationsnetzwerken ist die Anordnung der einzelnen Teilnehmer ebenso wichtig wie die Verbindung selbst. Die Topologie eines Netzes beschreibt nun die spezifische Anordnung der Geräte und ihrer Verbindungen, die gemeinsam das Netz bilden. Wir unterscheiden im allgemeinen auch hier zwischen physischer und logischer Topologie. Die physische Topologie befasst sich mit den physischen Verbindungen und beschreibt den Aufbau der Netz**verkabelung**, während die logische sich mit logischen Verbindungen beschäftigt und somit den Datenfluss zwischen den Endgeräten beschreibt. Graphisch wird eine Topologie durch einen Graphen dargestellt. Wir unterscheiden verschiedene Typen von Netztopologien: Die Bus-, Ring-, Stern-, Torus- sowie Baumtopologie und ein vollständig vermaschtes Netz.

Bustopologie Bei einer Bustopologie sind alle Geräte direkt mit demselben Übertragungsmedium, dem Bus verbunden. Es gibt keine aktiven Komponenten zwischen den Geräten und dem Medium. Wenn das Übertragungsmedium eines Busses ein Shared Medium ist – also z. B. dieselbe Kupferader von allen Teilnehmern gemeinsam zur Datenübertragung verwendet wird – muss sichergestellt werden, dass immer nur ein Gerät zum selben Zeitpunkt Signale auf das Übertragungsmedium sendet. Es ergeben sich die Vor- und Nachteile:

+ Nur geringe Kosten, da nur geringe Kabelmengen erforderlich sind.
+ Einfache Verkabelung und Netzerweiterung
+ Keine aktiven Netzwerkkomponenten

- — Leichtes Abhöhren von Datenübertragung
- — Eine Störung des Übertragungsmediums blockiert den gesamten Netzstrang
- — Zu einem Zeitpunkt kann nur eine Station Daten senden.

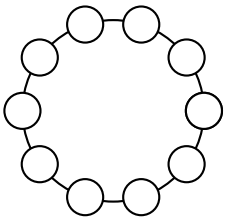
Skizziert:



Ringtopologie Bei der Vernetzung in Ringtopologie 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. Um Überschneidungen zu verhindern, sind bei dieser Art der Vernetzung besondere Adressierungsverfahren nötig. Da jeder Teilnehmer gleichzeitig als Repeater wirken kann (wenn keine Splitter eingesetzt werden), können auf diese Art große Entfernungen überbrückt werden. Vor- und Nachteile:

- + Deterministische Rechnernetzkommunikation ohne Paketkollisionen
- + Garantierte Übertragungsbandbreite
- + Alle Rechner haben gleiche Zugriffsmöglichkeiten
- + Leicht Programmierbar
- — Niedrige Bisektionsweite und Konnektivität
- — hohe Latenzen zu entfernten Knoten
- — Ohne Ringverteiler hoher Verkabelungsaufwand
- — Leichte Abhöhrmöglichkeiten von Datenübertragungen
- — Langsamere Datenübertragung bei vielen angeschlossenen Endgeräten

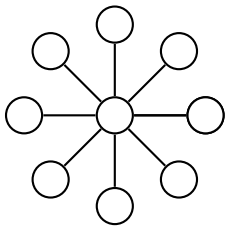
Skizziert:



Sterntopologie Bei Netzen in Stern-Topologie sind an einen zentralen Teilnehmer alle anderen Teilnehmer mit einer Punkt-zu-Punkt-Verbindung angeschlossen. Der zentrale Teilnehmer (Verteiler, beispielsweise ein Hub oder Switch) muss nicht notwendigerweise über eine besondere Steuerungsintelligenz verfügen. Vor- und Nachteile:

- + Ausfall von Endgeräten hat keine Auswirkung auf restliches Netz
- + Leicht erweiterbar und verständlich
- + Eignet sich besonders für Broadcast- und Multicastanwendungen
- — Netzverkehr wird unmöglich wenn Verteiler ausfällt
- — Hoher Kabelaufwand

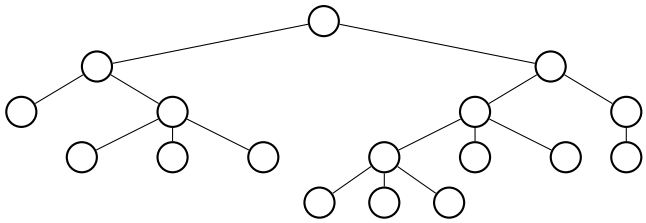
Skizziert:



Baumtopologie 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 („Wurzeln“ weiterer „Äste“). Sie obliegt einer strengen hierarchischen Ordnung, wobei Verbindungen zwischen Verteilern mittels Uplinks hergestellt werden. Vor- und Nachteile:

+ Ausfälle von Endgeräten haben keine Konsequenzen
+ Strukturelle Erweiterbarkeit
— Ausfälle von Verteilern betreffen ganze Unterbäume
— Es kann durch die definierte Bisektionsweite von 1 an der Wurzel zu Engpässen bei der Kommunikation zwischen Baumhälften kommen.
— Bei klassischen Bäumen kommt es zu schlechten Latezeigenschaften

Skizziert:

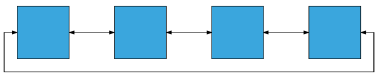


Torustopologie Bei einer Torustopologie wird sich bei dem geometrischen Modell eines Torus bedient und die Eigenschaften auf das Netzwerk übertragen. Eine solche Topologie kann man sich als ein vermaschtes Netz mit in einem Gitterarray der Dimension $N = 2,3,\ldots$ vorstellen, bei dem einzelne Elemente mit ihren nähesten Nachbarn bezüglich jeder Dimension und zusätzlich die Knoten an gegenüberliegenden Ecken einer Dimension verbunden sind. In diesem Gitter hat jeder Knoten genau $2 \cdot N$ Verbindungen. Die Topologie trägt ihren Namen deswegen, da das so entstandene Gitter topologisch homogen mit dem eines N-dimensionalen Torus ist.
Eine eindimensionale Toroustopologie ist äquivalent zur Ringtopologie.
Vor- und Nachteile:

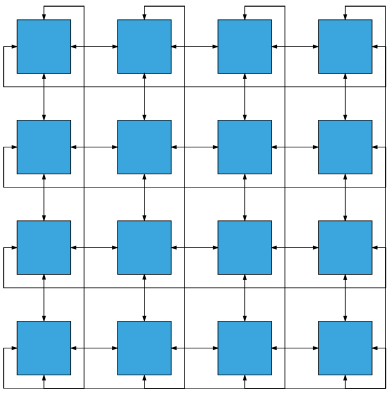
+ Höhere Geschwindligkeit bei geringerer Latenz.
  Dies ist der Verbindung der gegenüberliegenden Kanten und somit hoher Verbindungsanzahl pro Knoten geschuldet. Damit gibt es mehr Verbindungsoptionen zwischen zwei Knoten, was sich meistens wiederum positiv auf die Geschwindigkeit auswirkt.
+ Niedriger Energieverbrauch
  Da Daten im Allgemeinen weniger Zwischenschritte gehen, sinkt auch der Energieverbrauch.
— Hoher Kabelaufwand und hohe Schaltungskomplexität
  Vor allem unterschiedliche Verbindungslängen könnnen hierbei für Probleme sorgen (RC delay). Zudem wird die Designphase bei Entwürfen schwerer, da mehr Verbindungen notwendig sind.

— Hohe Kosten

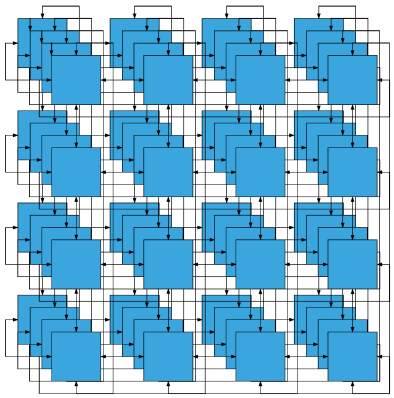
Skizziert:



Eine 1-dimensionale Torustopologie. Sie entspricht genau der Ringtopologie.



Eine 2-dimensionale Torustopologie.

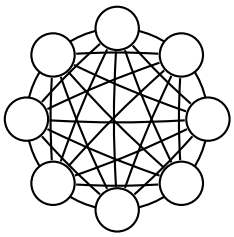


Eine 3-dimensionale Torustopologie.

Vermaschtes Netz In einem vermaschten Netz ist jedes Endgerät mit einem oder mehreren anderen Endgeräten verbunden. Wenn jeder Teilnehmer mit jedem anderen Teilnehmer verbunden ist, spricht man von einem vollständig vermaschten Netz. Bei Ausfall eines Endgerätes oder einer Leitung ist es im Regelfall möglich, durch Umleiten (Routing) der Daten weiter zu kommunizieren. Vor- und Nachteile:

+ Sicherste Variante eines Rechnernetzes
+ Ausfälle von Endgeräten können abgefangen werden
+ Hohe Bisektionsweite und niedrigen Durchmesser
+ je „voller“ die Netze vermascht sind desto weniger Routing wird benötigt
— Hoher Kabelaufwand
— Hoher Energieverbrauch
— Komplexes Routing notwendig, da die Netze nicht regulär oder symmetrisch sind

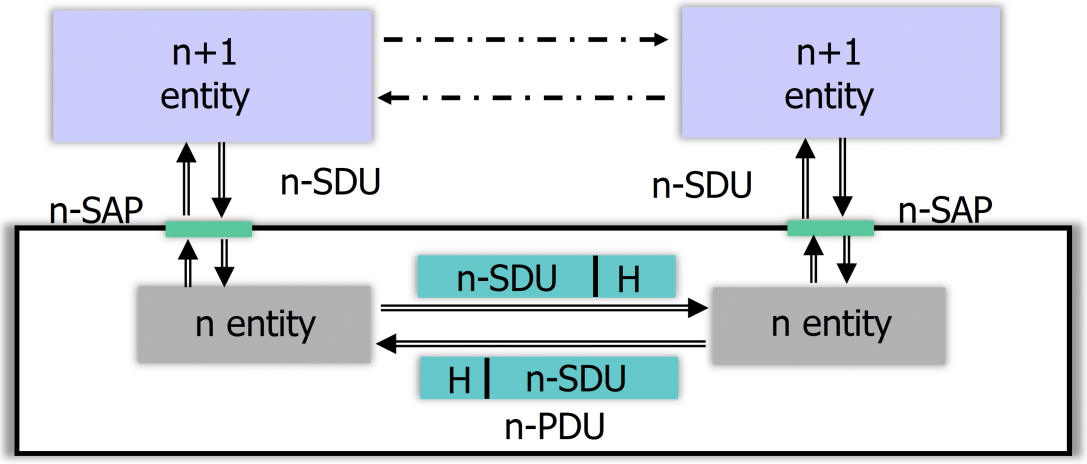
Skizziertes *vollständig vermaschtes Netz*:



1.2 Ein erster Überblick über verwendete Protokolle

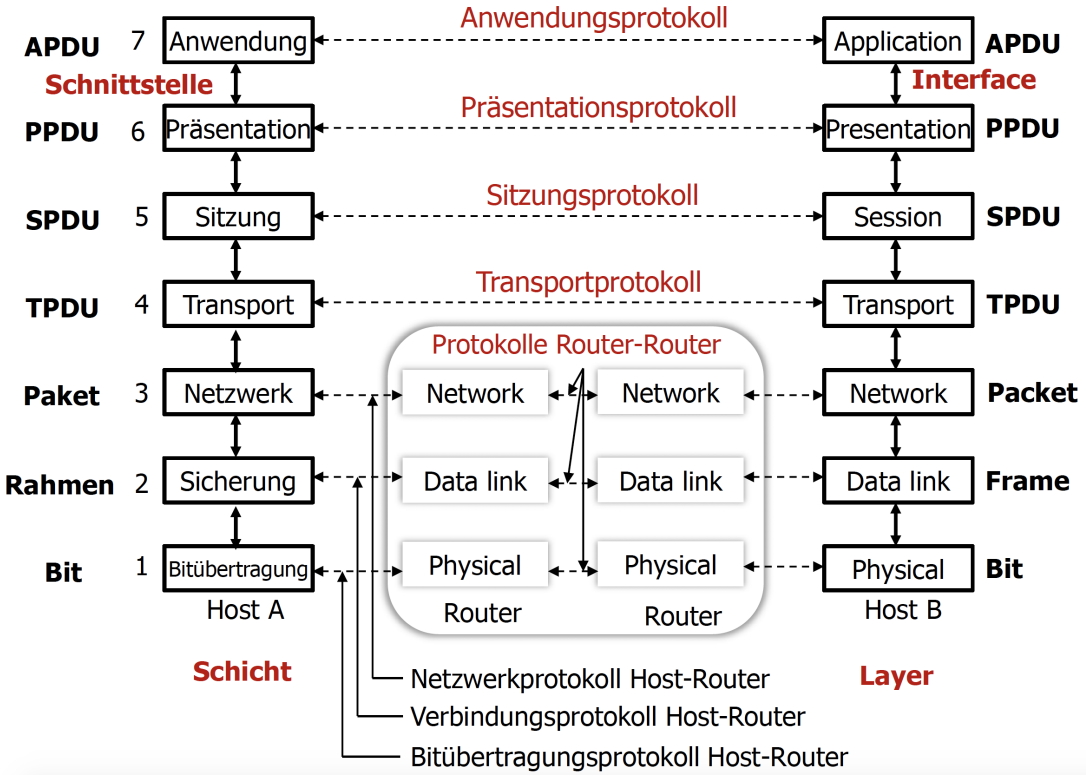
Aufgrund der Komplexität der Netzwerke und ihrer Kommunikation möchte man das Verhalten und das Format der Nachrichten strukturieren. Eine Variante hierfür ist die Verwendung von Protokollen. Zudem werden Systeme in verschiedene Schichten eingeteilt. Hierdurch werden Aufgaben, welche an sich zusammengehören, von derselben Schicht verwaltet und können dann

durch die schichtspezifischen Protokolle kommunizieren. Wir werden zwei solcher Schichtmodelle kennenlernen, das letztere allerdings aufgrund der erhöhten Aktualität bevorzugen. Wir strukturieren in Schichten um einen einfacheren Aufbau des Systems zu erhalten. Dabei nutzen die Instanzen der Schicht $n+1$ ebenso die angebotenen Dienste der Schicht n. Ein solcher Dienst wird zwischen Schichten an Dienstzugangspunkten (*service access points*, SAPs) angeboten; es werden *service data units* (SDUs) übergeben. Die Instanzen der Schicht auf verschiedenen Hosts tauschen dann *protocol data units* (PDUs) aus, wobei eine jede solche einen **Kopf** (*header*) beinhaltet. Als Diagramm:



1.2.1 ISO Open Systems Interconnection

Die *open systems interconnection* (OSI) ist ein Sammelsurium an Protokollen, Diensten und Schichten, deren Basis durch das Referenzmodell gelegt ist, zur Kommunikation zwischen Systemen. Es besteht aus sieben Schichten, welche wie folgt kommunizieren:



Schicht 1 — Bitübertragungsschicht (*physical layer*) Hier werden die nachrichtentechnischen Mittel für die Übertragung von einzelnen **Bits** definiert. Dies können Dinge wie

Zeitsynchronisation, Kodierung oder auch Modulation von Signalen sein. Näheres dazu in Kapitel 6.

Schicht 2 — Sicherungsschicht (*data link layer*) Die Sicherungsschicht (Data Link Layer) fasst Folgen von binären Informationen zu Datenpaketen zusammen bzw. löst größere Einheiten, die von einer höheren Schicht kommen, gegebenenfalls in kleinere Datenpakete auf. Die Sicherungsschicht betrachtet im Wesentlichen Zweipunktverbindungen. Sie beschäftigt sich zudem mit **Rahmen (*frames*)** und elementaren Fehlererkennungsmechanismen. Näheres dazu in Kapitel 5.

Schicht 3 — Netzwerkschicht (*network layer*) Die wichtigste Aufgabe der Netzwerkschicht ist das Routing, sprich die Bestimmung eines optimalen Weges durch ein verzweigtes Netzwerk. Sie beschäftigt sich mit der Übertragung von **Paketen und Datagrammen**, also insbesondere auch mit dem Verbindungsaufbau, der Wegwahl (*routing*), der Vermittlung etc. Sie realisiert dabei eine Ende-zu-Ende-Verbindung zwischen zwei kommunizierenden Stationen. Sie sorgt zudem auch für die Korrektheit des Weges. Näheres dazu in Kapitel 4.

Schicht 4 — Transportschicht (*transport layer*) Die Protokolle der Transportschicht haben Ende-zu-Ende Charakter, da sie eine Verbindung zwischen einzelnen Prozessen in verschiedenen Endsystemen realisieren. Sie beschäftigen sich mit Segmenten. Näheres dazu in Kapitel 3.

Schicht 5 — Sitzungsschicht (*session layer*) Der Sitzungsschicht wird durch die Schichten 1 bis 4 ein universeller Transportservice, welcher Prozess-zu-Prozess-Verbindungen annimmt, bereitgestellt. Eine solche Sitzung bezeichnet die logische Verbindung zwischen zwei Anwendungen der obersten Schicht, welche miteinander kommunizieren.

Schicht 6 — Darstellungsschicht (*presentation layer*) Die Darstellungsschicht sorgt durch spezielle Dienste für eine Transformation der Dateien auf ein vereinbartes Standardformat und für eine einheitliche Interpretation der ausgetauschten Daten.

Schicht 7 — Anwendungsschicht (*application layer*) Die oberste Schicht ist die Anwendungsschicht, die den verteilt realisierten Anwendungen die logisch-kommunikationstechnische Unterstützung in Form bestimmter Dienste wie E-Mail, Dateitransfers oder auch virtuelle Terminals anbietet. Hier kommunizieren Anwendungsprozesse mit anwendungsspezifischen Informationen. Näheres dazu in Kapitel 2.

1.2.2 Schichtenmodell des Internets

Im Gegensatz zu anderen Gebieten hat sich das OSI-Schichtenmodell im Internet nicht durchgesetzt. Stattdessen hat sich ein vereinfachtes Modell durchgesetzt. In diesem wurden Anwendungs-, Darstellungs, sowie Sitzungsschicht zur Anwendungsschicht zusammengefasst. Der Rest verhält sich – für uns – identisch.

Schicht 1 — Bitübertragungsschicht (*physical layer*) Hier werden die binären Formate, sowie Modulationsverfahren definert für eine Übertragung von einzelnen Bits. Es sorgt zudem für die Übertragung auf dem physikalischen Medium.

Schicht 2 — Sicherungsschicht (*link layer*) Die Sicherungsschicht behandelt die Sicherung von Informationen über Fehlererkennungsmaßnahmen, den Medienzugriff sowie den Aufbau von **Rahmen (*frames*)** zwischen benachbarten Geräten.

> Schicht 1 und Schicht 2 werden oft auch zusammen als *network interface layer (NIL)* gefasst.

Schicht 3 – Netzwerkschicht (*internet layer* oder auch *network layer*) Diese Schicht ist hauptsächlich für die Wegewahl (*routing*), Weiterleitung, sowie das Erstellen und Versenden von Datagrammen zwischen Hosts über Router verantwortlich.

Schicht 4 – Transportschicht (*transport layer*) Diese Schicht transportiert **Segmente** zwischen Anwendungen, verwendet dabei typische Protokolle wie das *verbindungslose User Datagram Protocol (UDP)* oder das *verbindungsorientierte TCP*. Es unterstützt die Anwendungsschicht in ihren Sitzungs- und Kommunikationsdiensten.

Schicht 5 – Anwendungsschicht (*application layer*) Hier finden sich Netzwerkanwendungen wieder, welche Protokolle wie SMTP, FTP, HTTP, DNS o.ä. verwenden.

1.2.3 Implementierung und Beschreibung von Protokollen

Weg einer Nachricht Werden Nachrichten zwischen verschiedenen Anwendungen realisiert, so steigen sie die Schichten herab, bis sie schlussendlich über die Bitübertragungsschicht auf physikalischen Medien versendet werden können.[1] Wird von einer höheren Schicht auf eine niedere Schicht gewechselt, so muss ein Nachrichtenkopf (*header*) für die niedere Schicht hinzugefügt werden, welcher erst beim Verlassen der niederen Schicht in eine höhere Schicht wieder entfernt wird.

Implementierungen Meistens können *header* von unteren Schichten nicht direkt manipuliert werden und eigene Routinen auf Schichten unterhalb der Anwendungsschicht definiert werden, da diese Teil des jeweiligen Betriebssystems sind. Die Dienste der Transportschicht können dabei über Betriebssystemaufrufe (*system calls*) genutzt werden, was in höheren Programmiersprachen auf einzelne Pakete und Bibliotheken hinauslief. (Beispiel Java: `javax.net`) Das Betriebssystem besitzt dann verschiedene Realisierungsmöglichkeiten, beispielsweise mit sinnvollem Umgang von Referenzen.

Cross-Layer-Optimierung

> In der Realität wird eine solche Schichtentrennung selbstverständlich nicht eingehalten. So können beispielsweise aus Effizienzgründen Mechanismen der Bitübertragungs- sowie Sicherungsschicht gekoppelt werden.

Beschreibung Protokolle werden von Standardisierungsgremien festgelegt. Es gibt hierbei unterschiedliche Beschreibungsarten, eine *formelle* und **informelle Art**, wobei letztere vor allem bei der IETF verbreitet ist und meist mit Referenzimplementierungen versehen ist. Formale Beschreibungen besitzten dagegen eine Beschreibung des **Formats** der Nachrichten (ASN.1) ähnlich der Beschreibung von Datenstrukturen in Programmiersprachen, verschiedene **Szenarien**, also typische Abläufe des Nachrichtenaustauschs (MSC oder Sequenzdiagramme) und eine Beschreibung des **Verhaltens der Instanzen**. Bei letzterem werden vor allem Automaten verwendet und diese dann über die SDL oder Statecharts dargestellt.

[1] Vereinfacht angenommen, dass niedere Schichtzugriffe nicht wegoptimiert werden können.

Beispiel für ASN.1 Im Folgenden ein einfaches Beispiel für eine ASN.1-Beschreibung eines Nachrichtenformats des Foo-Protokolls, welches Nachrichtentypen für Fragen (`FooQuestion`) und Antworten (`FooAnswer`) bereitstellt:

```
FooProtocol DEFINITIONS ::= BEGIN
    FooQuestion ::= SEQUENCE {
        trackingNumber INTEGER,
        question       STRING
    }

    FooAnswer ::= SEQUENCE {
        questionNumber INTEGER,
        answer         BOOLEAN
    }
END
```

Quellkode 1.1 — Definition von Nachrichtentypen mit ASN.1

Dienstgüte — *quality-of-service (QoS)* Die Dienstgüte eines Systems ist ein Sammelbegriff für quantitative Aspekte von Rechnernetzen und ihren Protokollen. Darunter fallen beispielsweise …

… die **Latenz** eines Netzes, sprich die Zeit, die ein Paket braucht um den Empfänger zu erreichen.

… den **Jitter** eines Netzwerks. Der Jitter ist dabei ein Maß für die Variabilität der Latenz.

… den **Durchsatz** eines Netzes, sprich die übertragenen Daten pro Zeiteinheit, was allerdings **nicht** der Bruttobitrate entspricht.

… die **Verlust- und Fehlerrate** eines Netzes.

… die **Verfügbarkeit** eines Netzes, also der Zeitanteil, in welchem das System einsatzfähig ist.

Die Dienstgüte ist relevant zur Auswahl und Konfiguration von Netzwerkarchitekturen und verwendeten Protokollen, je nach Schwerpunktsetzung. Um die Dienstgüte zu bestimmen gibt es verschiedene Messverfahren, stochastische und deterministische Analysen, sowie Simulationsverfahren, welche durch Werkzeuge assistiert werden.

1.3 Netzwerksicherheit

1.3.1 Grundlegende Begriffe

Netzwerksicherheit

- Begriffe
  - Funktionssicherheit (Safety)
    - Eigenschaft eines IT-Systems, dass es nicht durch technisches Fehlverhalten eine Bedrohung für materielle Güter und die körperliche Unversehrtheit darstellt
  - Informationssicherheit (Security)
    - Schutz eines IT-Systems gegen unautorisierte Nutzung
  - Datenschutz (Privacy)
    - Fähigkeit einer natürlichen Person, die Weitergabe von Informationen, die sie persönlich betreffen, zu kontrollieren

Netzwerksicherheit

- Schutzziele für die Informationssicherheit
  - Authentizität (Authenticity)
    - Echtheit und Glaubwürdigkeit eines Objekts bzw. Subjekts gewährleisten (z.B. einer Nachricht, eines Absenders, Access-Points, Servers, etc.)
  - Datenintegrität (Integrity)
    - zu schützende Daten können nicht unbemerkt verändert werden (z.B. Verhindern der Manipulation von Nachrichten)
  - Informationsvertraulichkeit (Confidentiality)
    - unautorisierte Informationsgewinnung ist unmöglich (z.B. Verhindern des Lesens einer Nachricht)
  - Verfügbarkeit (Availability)
    - keine unautorisierte Beeinträchtigung des Systems möglich (z.B. Verhindern von Denial-of-Service-Angriffen)
  - Verbindlichkeit (Non-Repudiation)
    - kein Abstreiten von Systemnutzung möglich (z.B. Kaufauftrag kann nicht geleugnet werden)
  - Anonymisierung und Pseudomisierung
    - Verändern personenbezogener Daten, so dass Einzelangaben nicht oder nur mit großem Aufwand natürlichen Personen zugeordnet werden können (z.B. Verhindern der Erstellung von Nutzerprofilen durch Dienstanbieter)

Netzwerksicherheit

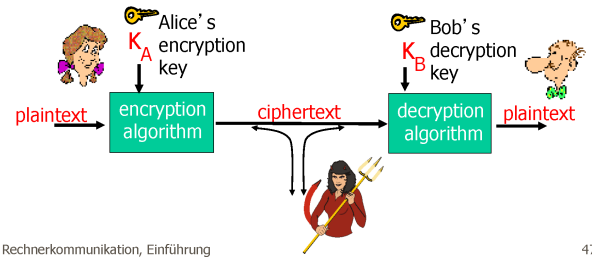
- Begriffe
  - Schwachstelle (Weakness) und Verwundbarkeit (Vulnerability)
    - Schwäche eines IT-Systems, die eine unautorisierte Nutzung ermöglicht
  - Exploit
    - beispielhafte Implementierung zur Ausnutzung von Schwachstellen
  - Angriff (Attack)
    - unautorisierter Zugriffsversuch
    - passive Angriffe: z.B. Abhören (Eavesdropping), Ausspähen von Passwörtern (Sniffing)
    - aktive Angriffe: z.B. Entfernen, Verändern, Einspielen von Datenpaketen, Vorspiegelung falscher Identität (Spoofing, Maskierung), Beeinträchtigung der Verfügbarkeit (Denial-of-Service, DoS)
- Abwehr
  - kryptographische Verfahren: Verschlüsselung, Hashfunktionen, Signierung, Challenge-Response-Verfahren
  - operationelle Sicherheit: Filterung (Firewall), Monitoring (Intrusion Detection)

1.3.2 Kryptosystem

Netzwerksicherheit

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

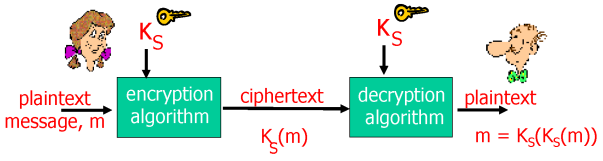
- Kryptografisches System (Kryptosystem)
  - Bestandteile: Klartext, Verschlüsselungsschlüssel, Verschlüsselungsverfahren, Chiffretext, Entschlüsselungsschlüssel, Entschlüsselungsverfahren
  - Kerckhoffs-Prinzip: Sicherheit des Systems darf nicht von der Geheimhaltung des Ver- und Entschlüsselungsverfahrens abhängen



Netzwerksicherheit

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

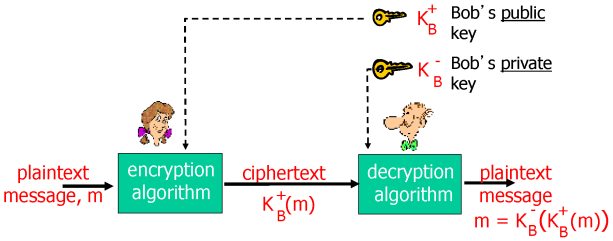
- Symmetrisches Kryptosystem
  - Ver- und Entschlüsselungsschlüssel sind gleich
  - beide Kommunikationspartner benötigen gemeinsamen geheimen Schlüssel (Shared Secret Key), der auf anderem Weg ausgetauscht werden muss
  - Sicherheit hängt von Stärke des Verfahrens und sicherer Verwaltung des Schlüssels ab



Netzwerksicherheit

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

- Asymmetrisches Kryptosystem
  - jeder Kommunikationspartner besitzt geheimen Schlüssel (Private Key) K^- und öffentlichen Schlüssel (Public Key) K^+
  - aus K^+ darf K^- (praktisch) nicht berechenbar sein
  - geheime Schlüssel müssen nur sicher verwaltet, aber nicht ausgetauscht werden



1.3.3 Symmetrische Verschlüsselung

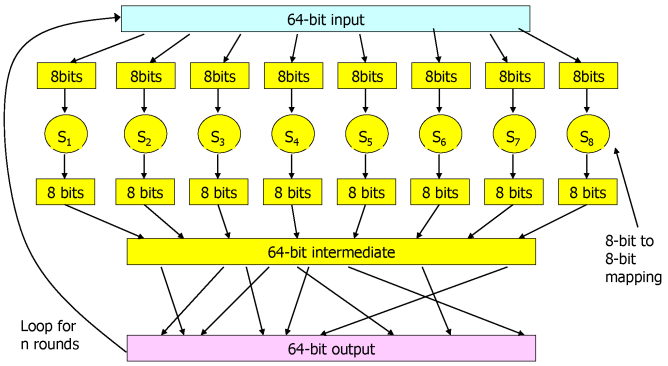
Netzwerksicherheit

- Symmetrische Verschlüsselung
  - Grundprinzipien
    - Permutation: Vertauschung von Zeichen des Klartexts
    - Substitution: Ersetzen von Zeichen des Klartexts
    - Angriffsarten: Brute Force, Ciphertext-Only, Known-Plaintext, Chosen-Plaintext
  - Blockchiffre (block cipher)
    - Eingabeblöcke fester Länge (z.B. 64, 128 Bits)
    - jeder Eingabeblock wird auf gleiche Weise verschlüsselt
    - einsetzbar, wenn Länge des Klartexts bekannt
    - bekannte Blockchiffre (National Institute of Standards and Technology, NIST)
      - Data Encryption Standard (DES), Blockgröße 64 Bits, Schlüssellänge nur 56 (eigentlich 40) Bits
      - Triple-DES (3DES), dreifache Anwendung von DES, Schlüssellänge 168 (eigentlich 112) Bits
      - Advanced Encryption Standard (AES), Blocklänge 128 Bits, Schlüssellänge 128-256 Bits

Netzwerksicherheit

Quelle: Leon-Garcia, Widjaja: *Communication Networks: Fundamental Concepts and Key Architectures*, 2nd ed., McGraw-Hill, 2004.

- Beispielhafter Aufbau einer Blockchiffre:

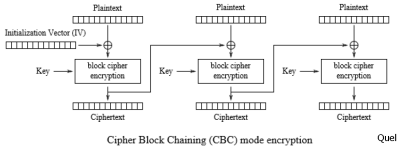


Netzwerksicherheit

- Stromchiffre (stream cipher)
  - Folgen von Klartextzeichen m(i) werden mit variierender Funktion verschlüsselt, z.B.:
    - Erzeugung eines Stroms von Schlüsseln k(i) mit Zufallszahlengenerator
    - Verschlüsselung: Jedes Klartextzeichen wird mit neuem Schlüssel mit XOR bitweise verknüpft, $c(i) = m(i) \oplus k(i)$
    - Entschlüsselung: Jedes Chiffrezeichen wird mit neuem Schlüssel mit XOR bitweise verknüpft, $m(i) = c(i) \oplus k(i)$
    - Kommunikationspartner benötigen Initialwert (Initialization Vector, IV) eines kryptografischen Zufallszahlengenerators
  - bekannter Stromchiffre: RC4, Zeichengröße von 1-256 Bytes, kein IV nötig

Netzwerksicherheit

- Betriebsarten
  - Blockchiffren können mit verschiedenen Betriebsarten als Stromchiffren eingesetzt werden
  - Beispiel: Cipher Block Chaining
    - Verschlüsselung: jeder Eingabeblock wird mit dem vorigen chiffrierten Eingabeblock mit XOR bitweise verknüpft und dann mit dem Blockchiffre verschlüsselt, $c(i) = K_S(m(i) \oplus c(i-1))$
    - Entschlüsselung: jeder chiffrierte Block wird mit dem Blockchiffre entschlüsselt und dann mit dem vorigen chiffrierten Eingabeblock mit XOR bitweise verknüpft, $m(i) = K_S(m(i) \oplus c(i-1))$



Cipher Block Chaining (CBC) mode encryption

Quelle: Wikipedia.

1.3.4 Asymmetrische Verschlüsselung

Netzwerksicherheit

- Asymmetrische Verschlüsselung
  - Grundprinzip
    - Einwegfunktion f mit Falltür: der Funktionswert y = f(x) ist einfach berechenbar, die Umkehrfunktion $x = f^{-1}(y)$ ist praktisch nur berechenbar mit zusätzlicher Information (= Falltür)
    - Verwendung mathematischer Probleme, für die keine effizienten Berechnungsverfahren bekannt sind
      - Faktorisierung (Zerlegung großer Zahlen in Primfaktoren), z.B. RSA-Verfahren, benötigt relativ langen Schlüssel (mindestens 1024 Bits)
      - diskreter Logarithmus (ElGamal)
      - diskreter Logarithmus auf elliptischen Kurven, schwereres Problem, höhere Sicherheit bei gleicher Schlüssellänge (mindestens 256 Bits)

Netzwerksicherheit

- RSA-Verfahren (Rivest, Shamir, Adleman)
  - Schlüsselerzeugung
    - wähle zwei große ungleiche Primzahlen p und q
    - berechne $n = p \cdot q$, $\varphi(n) = (p-1) \cdot (q-1)$
    - wähle zu φ(n) teilerfremde Zahl e mit $1 < e < \varphi(n)$
    - berechne d, so dass $e \cdot d \bmod \varphi(n) = 1$
    - (n,e) ist öffentlicher Schlüssel K^+
    - (n,d) ist privater Schlüssel K^-
  - Verschlüsselung
    - $K^+(m) = m^e \bmod n = c$
  - Entschlüsselung
    - $K^-(c) = c^d \bmod n = m$
  - Beweis für $m = (m^e \bmod n)^d \bmod n$ benötigt Elemente der Zahlentheorie

Netzwerksicherheit

- Bsp. für RSA-Verfahren mit kleinen Zahlen
  - Primzahlen p = 5, q = 7, n = pq = 35, φ(n) = (p-1)(q-1) = 24
  - dann z.B. e = 5, teilerfremd zu 24
  - und z.B. d = 29, ed-1 = 144 ist durch φ(n) = 24 teilbar, daraus folgt $e \cdot d \bmod \varphi(n) = 1$
  - Verschlüsselung: für Klartext $m = (00001000)_2 = 12$ ergibt sich $m^e = 248832$ und $K^+(m) = m^e \bmod n = c = 17$
  - Entschlüsselung: $c^d = 48196857210675091509141182522307169 7$, $K^-(c) = c^d \bmod n = m = 12$
- Vertauschbarkeit von Ver- und Entschlüsselung
  - es gilt auch $K^+(K^-(m)) = (m^d \bmod n)^e \bmod n = m$
  - entspricht einer Verschlüsselung mit dem privaten Schlüssel und einer Entschlüsselung mit dem öffentlichen Schlüssel
  - kann für digitale Signatur verwendet werden

1.3.5 Kryptographische Hashfunktionen und MAC

Netzwerksicherheit

- Kryptografische Hashfunktionen
  - kurzer „Fingerabdruck" H(m) einer variabel langen Nachricht m
  - schwache Hashfunktion
    - leicht zu berechnende Einwegfunktion h = H(m) mit |h| = konstant
    - für gegebenes h ist es praktisch unmöglich, eine Nachricht m' zu finden mit H(m') = h
  - starke Hashfunktion
    - schwache Hashfunktion, für die es zusätzlich praktisch unmöglich ist, zwei Nachrichten m und m' zu finden mit H(m) = H(m')
  - verbreitete Verfahren
    - Message Digest Algorithm 5 (MD5), 1991 von Rivest vorgeschlagen, iterierte Kompressionen, 128 Bit Hashwerte, Kollisionsangriffe bekannt
    - Secure Hash Algorithm (SHA-3), 2012 Gewinner eines Wettbewerbs des NIST, 224-512 Bit Hashwerte

Netzwerksicherheit

- Message Authentication Code (MAC)
  - Hashfunktion $MAC(m, K_{AB})$, die zusätzlich von einem symmetrischen Schlüssel K_{AB} abhängt
  - ermöglicht Überprüfung der Datenintegrität
  - Einsatz
    - Sender A erzeugt $mac = MAC(m, K_{AB})$ und sendet (m, mac)
    - Empfänger erhält (m', mac') und überprüft, ob $MAC(m', K_{AB}) = mac'$
  - Konstruktion von MACs
    - basierend auf Blockchiffren, z.B. bei AES die einzelnen Blöcke mit dem Schlüssel mit XOR verknüpfen
    - Keyed Hash: Schlüssel an Nachricht anhängen und dann Hashfunktion anwenden: $H(m|K_{AB})$
    - Hash MAC (HMAC): Variante von Keyed Hash, komplizierterer Aufbau (NIST)

1.3.6 Digitale Signatur

Netzwerksicherheit

- Digitale Signatur
  - durch asymmetrische Verschlüsselung, bei der zur Verschlüsselung der private Schlüssel und zur Entschlüsselung der öffentliche Schlüssel verwendet wird
  - Einsatz
    - Sender erzeugt $K^-(m)$ mit seinem privaten Schlüssel und sendet dies
    - Empfänger erhält dies und entschlüsselt mit dem öffentlichen Schlüssel des Senders $M = K^+(K^-(m))$
  - Authentizität: Unter der Voraussetzung, dass der öffentliche Schlüssel eindeutig einer Person zuordenbar ist, bezeugt dies die Identität des Senders
  - Datenintegrität: Eine signierte Nachricht kann nicht unbemerkt verändert werden, da sonst keine Entschlüsselung möglich ist
  - Verbindlichkeit: Unter der Voraussetzung, dass der private Schlüssel nicht von anderen verwendet wird, kann der Sender die Signatur nicht zurückweisen

Netzwerksicherheit

- Digitale Signatur (Fortsetzung)
  - asymmetrische kryptografische Operationen sind aufwändig
  - daher Kombination mit Hashfunktion: nur Hashwert der Nachricht wird mit digitaler Signatur versehen
  - Einsatz
    - Sender erzeugt $S = K^-(H(m))$ und sendet (m, S)
    - Empfänger erhält (m', S'), und prüft ob $H(m') = K^+(S')$
  - ermöglicht genauso Authenzität, Datenintegrität und Verbindlichkeit

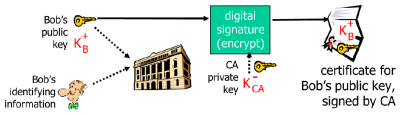
1.3.7 Zertifizierung

Netzwerksicherheit

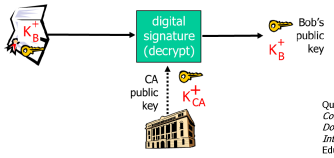
- Zertifizierung
  - digitale Bescheinigung eines öffentlichen Schlüssels
  - verbreiteter Standard für Zertifikate: ITU X.509, enthält
    - Subjektschlüssel: öffentlicher Schlüssel + Algorithmen u. Parameter
    - Signatur: mit privatem Schlüssel des Zertifikatausstellers verschlüsselter Hashwert + Algorithmen u. Parameter
    - Versionsnummer des Standards, Seriennummer (vom Aussteller zu vergeben), Zertifikataussteller (Name der Instanz), Gültigkeitsdauer, Subjektname, …
  - Zertifizierungsstelle (Trust Center, Certification Authority, CA)
    - bietet Dienste zur Generierung von Schüsselpaaren, Suche und Zertifizierung von Benutzern an, privat oder öffentlich
  - Public Key Infrastructure (PKI)
    - hierarchische Organisation von Zertifizierungsstellen
    - Zertifikate können auf Wurzel zurückgeführt werden
    - Umsetzungs- und Vertrauensprobleme
  - alternativ Web-of-Trust (WoT): Vertrauensnetz

Netzwerksicherheit

- Zertifikaterstellung:



- Zertifikatauswertung:

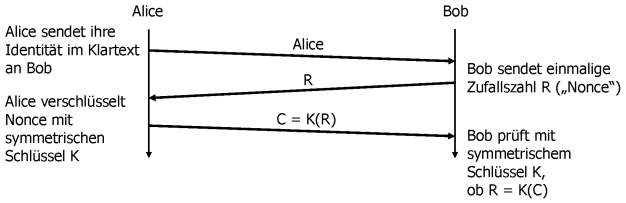


Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

1.3.8 Challenge-Response-Verfahren

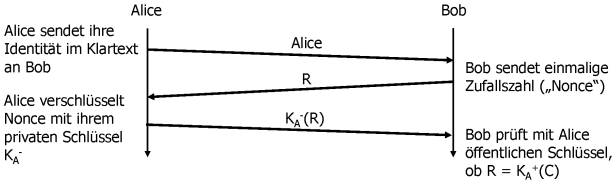
Netzwerksicherheit

- Challenge-Response-Verfahren
  - zur Authentifikation des Kommunikationspartners
  - Verallgemeinerung der Authentifikation durch Wissen (wie z.B. bei Passwörtern)
  - Variante mit symmetrischer Verschlüsselung zur Authentifikation von Alice gegenüber Bob (benötigt Kenntnis über gemeinsamen symmetrischen Schlüssel, „shared secret"):



Netzwerksicherheit

- Challenge-Response-Verfahren (Fortsetzung)
  - Variante mit asymmetrischer Verschlüsselung zur Authentifikation von Alice gegenüber Bob (Bob benötigt öffentlichen Schlüssel von Alice)



  - öffentlicher Schlüssel von Alice muss auf sichere Weise übermittelt werden, sonst „Man-in-the-Middle"-Angriff möglich

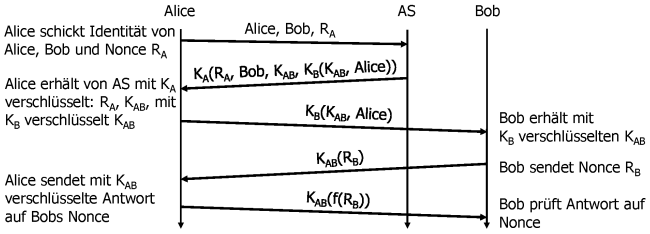
1.3.9 Schlüsselaustausch

Netzwerksicherheit

- Schlüsselaustausch
  - sicherer Austausch von Schlüsseln ist Voraussetzung für den Einsatz von Verfahren zur Verschlüsselung, Authentifikation und Integrität
  - dafür werden Schlüsselaustauschprotokolle benötigt
  - dabei können leicht Sicherheitsprobleme entstehen
  - Needham-Schroeder-Protokolle (1978) stellen Protokoll-Entwurfsmuster dar, auf deren Grundlage die meisten in der Praxis eingesetzten Systeme arbeiten
  - 2 Varianten basierend auf symmetrischer und asymmetrischer Verschlüsselung
  - beide benötigen einen vertrauenswürdigen Authentifizierungs- und Schlüsselverteilungsserver AS

Netzwerksicherheit

- Schlüsselaustausch mit symmetrischer Verschlüsselung
  - AS besitzt symmetrische Schlüssel K_A und K_B für Alice und Bob und erzeugt symmetrischen Sitzungsschlüssel K_{AB}
  - Alice erhält von AS Sitzungsschlüssel K_{AB}, dieser wird ihr zur Weiterleitung an Bob auch mit K_B verschlüsselt geliefert
  - Authentifizierung des AS gegenüber Alice und Alice gegenüber Bob



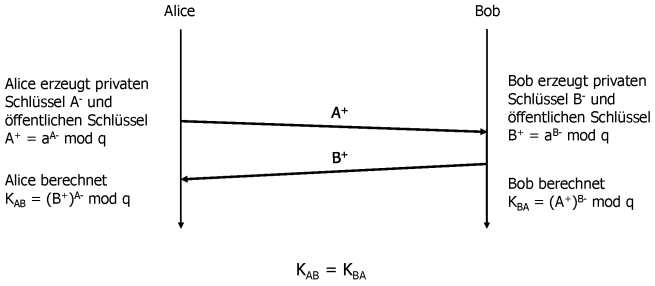
Netzwerksicherheit

- Diffie-Hellman-Schlüsselaustausch (1976)
  - die Kommunikationspartner berechnen dezentral den gemeinsamen symmetrischen Sitzungsschlüssel, dieser muss nicht über das Medium transportiert werden, kein Authentifizierungs- und Schlüsselverteilungsserver nötig
  - mathematische Grundlagen
    - diskreter Logarithmus über einem Galois-Feld zu einer Primzahl q
    - Einheitswurzel: für alle $1 \le m \le q-1$ gibt es ein $1 \le p \le q-1$ mit $m = a^p \bmod q$
  - Verfahren
    - öffentlich bekannt: Primzahl q und primitive Einheitswurzel a
    - Alice und Bob wählen Zufallszahlen A^- und B^- aus {1, …, q-1} als private Schlüssel
    - Alices öffentlicher Schlüssel ist $A^+ = a^{A^-} \bmod q$
    - Bobs öffentlicher Schlüssel ist $B^+ = a^{B^-} \bmod q$
    - Alice und Bob tauschen ihre öffentlichen Schlüssel aus
    - Alice berechnet $K_{AB} = (B^+)^{A^-} \bmod q$, Bob $K_{BA} = (A^+)^{B^-} \bmod q$
    - es gilt $K_{AB} = K_{BA}$, dies ist der gemeinsame Sitzungsschlüssel

Netzwerksicherheit

- Diffie-Hellman-Schlüsselaustausch (Fortsetzung)

bekannt: Primzahl q, primitive Einheitswurzel q



$K_{AB} = K_{BA}$

1.4 Beispiele für Rechen- und Kommunikationsnetze

Es gibt verschiedene Beispiele für Rechnerkommunikation, zu bekannteren gehören unter anderem das IP-Netz, LTE, Bluetooth, LoRaWAN, sowie die Vernetzung im Fahrzeug.

KAPITEL 2

ANWENDUNGSSCHICHT

Eine **Netzwerkanwendung** besteht aus *Anwendungsprozessen* verschiedener Endsysteme, welche mittels *Nachrichten* kommunizieren. Für die **Implementierung** können die *Dienste der Transportschicht* direkt verwendet werden. Ein **Anwendungsprotokoll** standardisiert das Verhalten und sorgt für ein einheitliches Format der Nachrichten. Untere Schichten und der Netzwerkkern benötigen keine Kenntnis über die Anwendung. Beispiele der Anwendungsschicht sind Web-Browser und Web-Server.

Beim **Client-Server-Paradigma** geht es um die typische Verbindung zwischen einem Serverprozess und einem Clientprozess, welche beide gleichzeitig aktiv sein, und eine Verbindung aufgebaut haben müssen um Nachrichten auszutauschen. Dabei ist meist ein leistungsstarker Server dauerhaft aktiv und von - nicht untereinander kommunizierenden - Clients erreichbar.

Im Gegensatz zu dieser **zentralisierte Architektur** gibt es auch wechselnde Rolle von Client und Server (zum Beispiel bei Web-Caching oder SMTP). Außerdem gibt es das System der verteilten Anwendung, wobei mehrere unabhängige Anwendungen gemeinsam als eine Einheit fungieren. Hierfür sind prägnante Beispiele ein WebShop mit Webserver, Applikations-Server oder auch Datenbanken.

Dienste der Transportschicht Im Internet verwendet man meistens eines der zwei dominierenden Transportprotokolle TCP oder UDP. TCP ist verbindungsorientiert und zuverlässig; es hat eine abstrakte Sicht des Versendens eines Bytestroms. UDP auf der anderen Seite ist verbindungslos, denn es versendet einzelne Datagramme, und eher unzuverlässig. Die Dienste werden meist im Betriebssystem realisiert und über **Socket-Schnittstellen**, welche durch eine API von Programmiersprachen angesprochen werden, angeboten. Mit einem solchen Socket kann dann das Transportprotokoll, die IP-Adressen von Sende- und Zielhost, sowie die Portnummern festgelegt werden.

Anforderungen an Anwendungen Wir wollen verschiedene Dinge von Anwendungen anfordern:

- **Verlust** von Daten darf beispielsweise **nicht tolerierbar** im Dateitransfer oder dem Online-Banking sein. In Multimediaanwendungen kann Verlust teilweise toleriert werden.
- Die **Bitrate** ist bei traditionellen Anwendungen (FTP, HTTP, E-Mail) im Allgemeinen nicht festgesetzt. Die Anwendungen verhalten sich aber responsiver und damit für den Nutzer angenehmer, wenn sie eine hohe Bitrate erhalten. Echtzeitmultimediaanwendungen (Netflix, YouTube, ...) brauchen hingegen eine Mindestbitrate.
- Die maximale **Verzögerungszeit** bestimmt wie echtzeitabhängig eine Anwendung ist. Bei sicherheitskritischen Systemen oder bei Steuerungen technischer Geräte braucht es dabei eine Garantie einer maximalen Verzögerungszeit. Spielanwendungen benötigen kurze Zeiten um gut spielbar zu sein.

Diese Anforderungen sind für einige Applikationen in folgender Tabelle zusammengetragen:

Anwendung	Verlust	Bitrate	Verzögerungszeit
Dateitransfer	✘	elastisch	keine harte Grenze
E-Mail	✘	elastisch	keine harte Grenze
Web-Dokumente	✘	elastisch	keine harte Grenze
Echtzeitmultimedia	∼	Audio: Kbps – Mbps Video: 10 Kbps – mehrere 100 Mbps	150ms, wobei Einwegverzögerungen unbemerkt bleiben
Streaming von Multimedia	∼		einige Sekunden
Interaktive Spiele	∼	Kbps – 10 Kbps	einige 100 ms
Automatisierung	✘	Kbps	oft harte Grenzen bei einigen ms
Smartphone Messaging	✘	elastisch	keine harte Grenze

Tabelle 2.1: Quantitative Anforderungen von Anwendungen, entnommen aus [KR17]

Verwendete Protokolle Für einige Anwendungen sind in folgender Tabelle die jeweiligen Anwendungs-, sowie darunterliegende Transportprotokolle zusammengefasst:

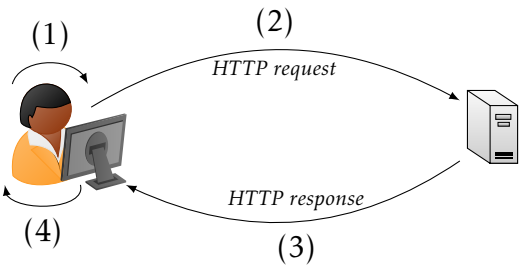
Anwendung	Anwendungsprotokoll	Transportprotokoll
E-Mail	SMTP [RFC 5321]	TCP
Remote Terminal Access	Telnet [RFC 854]	TCP
Zugriff auf Websites	HTTP [RFC 2616]	TCP
Dateitransfer	FTP [RFC 959]	TCP
Multimediastreaming	RTP [RFC 3550], HTTP [RFC 2616] oder proprietäre Protokolle	UDP oder TCP
Internettelefonie	RTP [RFC 3550], SIP [RFC 3261] oder proprietäre Protokolle	UDP oder TCP

Tabelle 2.2: Typische Anwendungs- und Transportprotokolle für typische Anwendungen, entnommen aus [KR17]

2.1 Verbreitete Anwendungen

2.1.1 Hypertext Transfer Protocol — HTTP

HTTP arbeitet nach dem *Request-Response-Verfahren*. Der Web-Client richtet dazu Anfragen (*HTTP-Request*) an den Web-Server, welcher wiederum mit einer Antwort (*HTTP-Response*) antwortet.
Ein vereinfachter Ablauf:



(1) Nutzer gibt URI[1] in Web-Browser ein.

(2) Web-Browser stellt Anfrage an Web-Server für spezifisches **Objekt**.

(3) Web-Server liefert Objekt an Web-Browser zurück.

(4) Web-Browser stellt Objekt in für Nutzer lesbarer Form dar.

Anfragenachrichten Es gibt eine Vielzahl an **Methoden**, welche in HTTP-Anfragen gestellt werden können:

- **GET** — Abruf eines Dokuments
- **HEAD** — Abruf von Metainformationen eines Dokuments
- **POST** — Übergabe von Informationen an Server
- **PUT** — Erstellen von neuen Dokumenten oder Ersetzen von Repräsentationen der Zielressource mit dem Inhalt. *PUT ist idempotent und somit **seiteneffektfrei**!*
- **DELETE** — Löscht die angegebene Ressource.

Die **Kopfzeilen** enthalten mehrere Typ/Wert-Paare. Hier stehen Typen wie der Host, der User-agent (Web-Browser), ...
Der **Rumpf** einer Anfrage ist bei GET leer, kann sonst aber durchaus Inhalt haben.
Ein Beispiel einer Anfrage:

```
GET /somedir/page.html HTTP/1.1
Host: www.example.org
User-agent: Mozilla/4.0
Connection: close
Accept-language: de-de
```

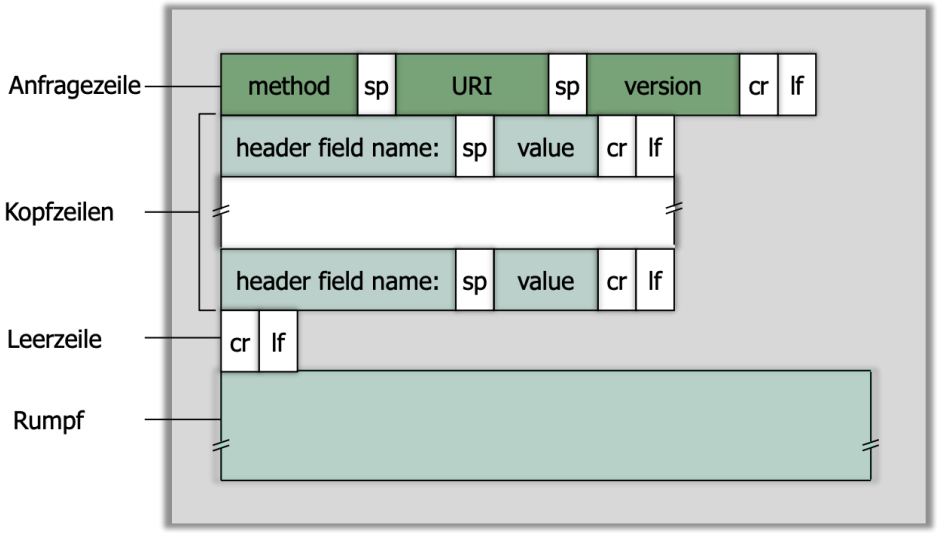


Abbildung 2.1: Format der HTTP-Anfragenachrichten

[1] Die URI (Uniform Resource Identifier) enthält den Web-Server-Namen und den Pfad zu einem Objekt.

Antwortnachrichten Der Server antwortet mit einem Statuscode, der zugehörigen Phrase, Kopfzeilen und einem Rumpf, welcher eventuelle Inhalte beinhaltet.
Beispiele für Statuscodes:

- **200 OK** — Anfrage verlief erfolgreich, das geforderte Objekt ist in dem Antwortrumpf
- **301 Moved Permanently** — Redirektion: Objekt ist zu finden unter der Location: ...
- **400 Bad Request** — Anfragenachricht nicht verstanden
- **404 Not Found** — Objekt nicht gefunden
- **505 HTTP Version Not Supported**

Ein Beispiel einer Antwort:

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 06 Aug 1998 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 22 Jun 1998 ...
Content-Length: 6821
Content-Type: text/html
data data data data data ...
```

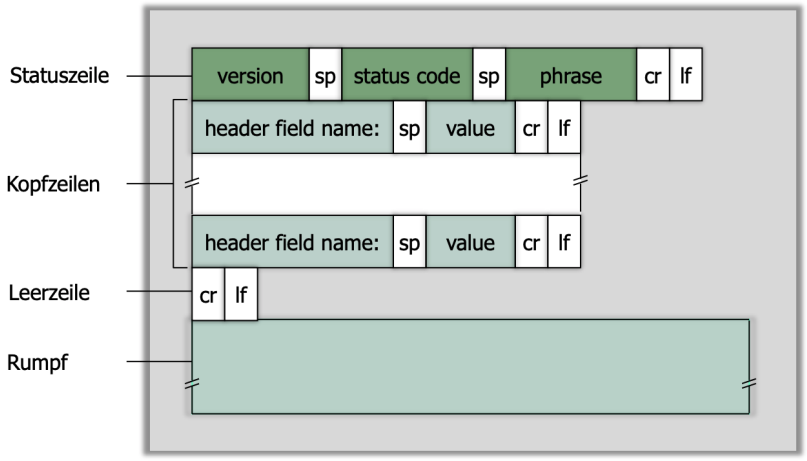


Abbildung 2.2: Format der HTTP-Antwortnachrichten

Ablauf — nicht-persistentes HTTP

(1) Client initiiert TCP-Verbindung zu Server an Port 80.

(2) Server auf Host wartet auf TCP-Verbindungen an Port 80, nimmt TCP-Verbindung an, benachrichtigt Client.

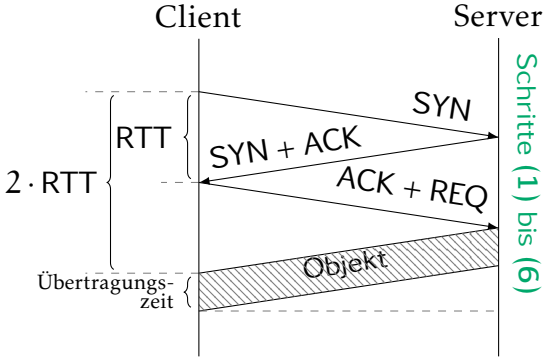
(3) Client übergibt HTTP-Anfrage an TCP-Socket, diese enthält URL mit Verweis auf Objekt.

(4) Server empfängt Anfrage, erstellt HTTP-Antwort mit Objekt und übergibt diese dem TCP-Socket

(5) Server schließt TCP-Verbindung

(6) Client erhält HTTP-Antwort mit dem Inhalt, analysiert ihn, stellt ihn auf dem Bildschirm dar.

(7) Wiederhole Schritte (1) bis (6) für jedes weitere Objekt.



Der TCP-Verbindungsaufbau benötigt eine Rundlaufzeit (*Round-Trip-Time*, RTT). Die Antwortnachricht erfordert dann ebenfalls noch eine RTT, womit sich die Antwortzeit durch

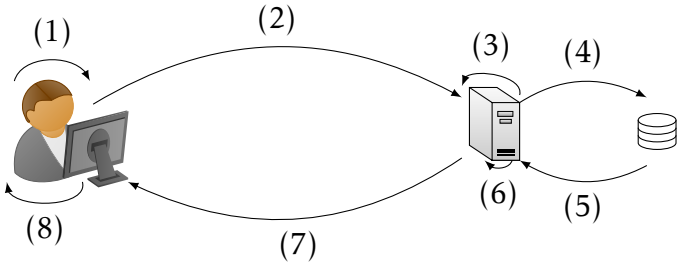
$$\tau_{\text{Antwort}} = 2 \cdot \text{RTT} + \tau_{\text{Senden}} + \text{TCP-Wartezeit}$$

bestimmt.

Ablauf — persistentes HTTP Hier lässt der Server Verbindungen bestehen und alle Objekte werden über **eine TCP-Verbindung** gesendet. Man spart sich somit die erneuten RTTs. Wird dabei *Pipelining* verwendet, so wird eine Anfrage für alle Objekte gestellt, ohne wird weiterhin eine Anfrage für jedes Objekt gestellt.

Dynamische Inhalte — Allgemeines Informationen werden meist entweder im Rumpf von **POST**-Anfragenachrichten oder als Typ/Werte-Paare angehängt an die URL einer **GET**-Anfragenachricht vom Client zum Server gesendet.

Dynamische Inhalte — CGI-Skripte Das Common Gateway Interface (CGI) verarbeitet als externer Prozess die Information und liefert dann eine neue HTML-Seite an den Server. Beispielsweise:



(1) Nutzer füllt ein Formular im Web-Browser aus.

(2) Web-Browser stellt Anfrage an Web-Server mit den übergebenen Daten.

(3) Die Daten werden dem CGI übergeben.

(4) CGI stellt eine Anfrage an die Datenbank.

(5) DB-Antwort wird zurückgegeben.

(6) CGI erstellt HTML.

(7) Web-Server antwortet mit dem erstellten HTML.

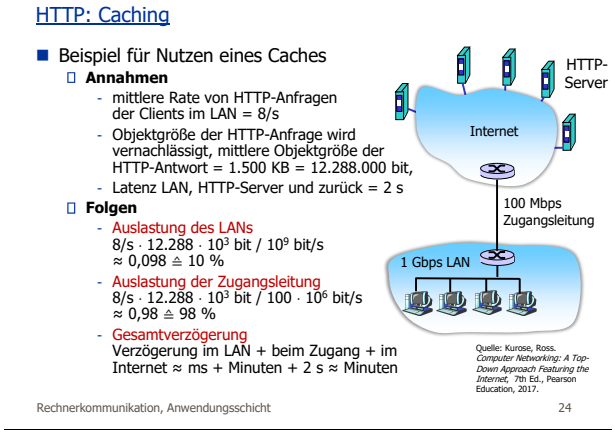
(8) Web-Browser stellt die Daten nutzerlesbar dar.

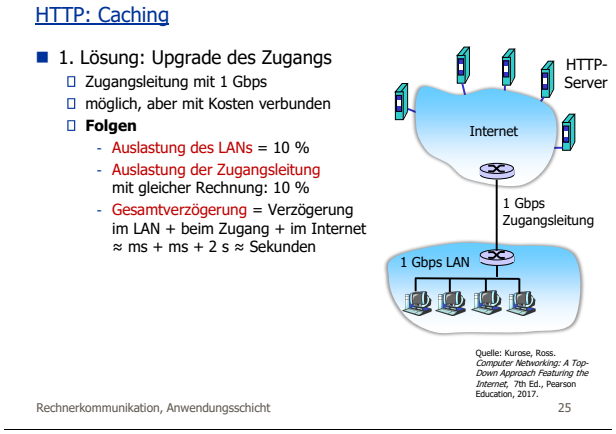
Dynamische Inhalte — Scripting Durch die Interpretation von eingebetteten Skripten können dynamische Inhalte erzeugt werden. Man unterscheidet das **serverseitige** und *clientseitige* Scripting. Bei ersterem ist im HTML Code eingebettet, welcher **vom Server** interpretiert wird und dabei HTML erzeugt. Das Standardbeispiel hierfür ist PHP. Bei letzterem ist im HTML Code eingebettet, welcher *vom Client* interpretiert wird. Das Standardbeispiel hierfür ist JavaScript. Eine Verbesserung ist AJAX (Asynchronous JavaScript And XML), was durch eine Entkopplung der Datenaustaschschicht von der Präsentationsschicht es erlaubt Inhalte dynamisch zu ändern ohne die ganze Seite neuzuladen.

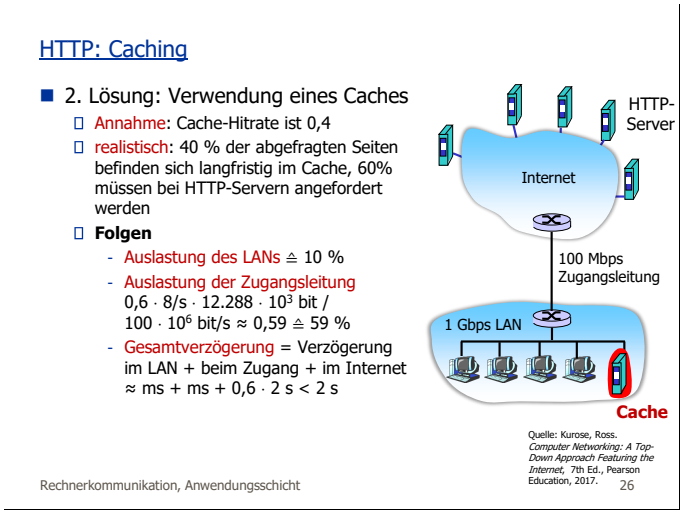
Caching Man möchte nun die Wartezeit des Benutzers und des Netzwerkverkehrs verringern durch die Nutzung von Zwischenspeichern. Dabei übernimmt der Cache für den Web-Browser die Rolle des Servers und für den Server die Rolle des Clients. Dabei ist die Platzierung eines solchen Caches an vielen Stellen möglich.
Der Cache kann dabei beim Server erfragen, ob sein Objekt noch aktuell ist. Dazu stellt der Cache eine Anfrage, welche

`If-modified-since: <date>`

beinhaltet. Ist das Objekt unverändert, so erhält der Cache eine `304 (Not Modified)`-Antwort, andernfalls eine `200 (OK)`-Antwort mit der angeforderten Seite im Rumpf. Ein Rechenbeispiel:







HTTP/2 HTTP/2 ist die Weiterentwicklung von HTTP. Durch die entstandenen Latenzprobleme bei heutigen Websites gab es verschiedene Weiterentwicklung, unter anderem auch das proprietäre SPDY von Google oder HTTP/2. Im Wesentlichen bietet HTTP/2 die gleichen Methoden wie auch HTTP/1.1, besitzt aber ein binäres statt einem textbasiertem Format. Ebenso werden verschiedene Ströme über eine TCP-Verbindung gemultiplext. Es gibt eine Header-Kompression (HPACK), sowie einen Server-Push, welcher es einem Server erlaubt Ressourcen zu einem Client zu schicken, noch bevor er sie angefragt hat. Ebenso ist TLS-Verschlüsselung „de facto“ Standard und die Unterstützung durch die bekannten Browser seit Ende 2015 gesichert.

Authentifizierung Aufgrund der Zustandslosigkeit von HTTP muss die Autorisierung in **jedem Request** erfolgen. Üblicherweise erfolgt diese dann über Benutzername und Passwort. Dazu ist im Request eine weitere Kopfzeile erforderlich: `Authorization: <data>` Fehlt diese Kopfzeile, so sendet der Server keine Daten, sondern in einer Kopfzeile der `401`-Antwort `WWW-Authenticate:` . Diese Nutzerdaten können auch vom Browser gespeichert werden.

Cookies Wie eben festgestellt ist HTTP per se zustandslos, dennoch möchte man manchmal sich den Zustand des Clients merken. Dazu sendet der Server einen „Cookie“ in der Antwort an den Client durch die zusätzliche Kopfzeile `Set-cookie: <ck-nr>`. Der Client speichert sich dann diesen Cookie und überträgt diesen in weiteren Anfragen mit durch die Kopfzeile `cookie: <ck-nr>`. Der Server kann dann das präsentierte Cookie mit den Informationen auf dem Server abspeichern,

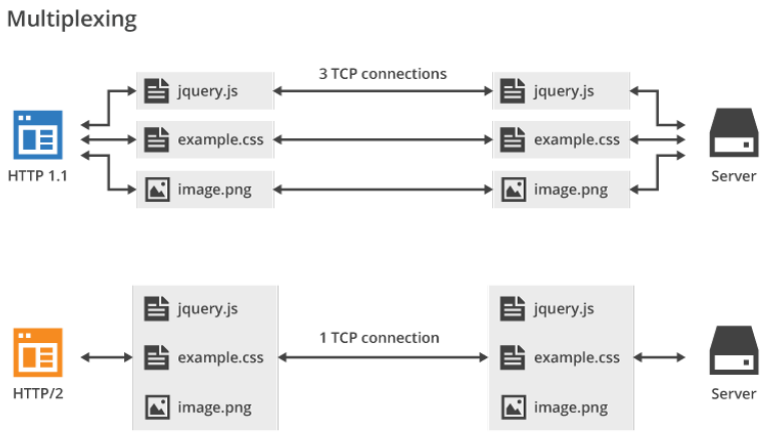


Abbildung 2.3: Multiplexing — HTTP/1.1 im Vergleich zu HTTP/2 (entnommen aus: https://blog.cloudflare.com/content/images/2015/12/http-2-multiplexing.png)

so beispielsweise die Authentifizierung, Benutzereinstellungen oder auch zielgruppengerechte Werbung durch *verfolgende Cookies*, welche Servern es ermöglichen Informationen über den Besuch anderer Server zu erhalten.

2.1.2 File Transfer Protocol — FTP

Das Ziel des Protokolls ist eine Übertragung von Dateien zwischen Hosts. Dabei gibt es eine TCP-Verbindung auf Port 21 allein zur Steuerung und **pro Datei** eine TCP-Verbindung auf Port 20 zur Übertragung. Die Tatsache, dass die Steuerbefehle nicht über die Datenverbindung geschickt werden, sondern über eine extra aufgebaute Verbindung, nennt sich **out-of-band-control**. Dadurch ist die Steuerunug während der Datenübertragung möglich.
Neben FTP ist FTPS (Secure) eine verschlüsselte Variante (Steuerungsport: 990, Datenport: 989), sowie FTPES, wobei hierbei die Ports gleich bleiben, allerdings das AUTH-Kommando zum Verschlüsseln genommen wird.

FTP — Active Mode Hierbei baut der Client eine TCP-Verbindung von einem beliebigen unprivilegierten TCP-Port N zu Port 21 des Servers auf. (*Steuerverbindung*)
Vor der tatsächlichen Dateiübertragung sendet der Client dann den Befehl `PORT N+1` an den Server und wartet auf TCP-Verbindungen auf Port $N+1$. Server baut eine TCP-Verbindung von seinem lokalen Port 20 zu Port $N+1$ des Clients auf (*Datenverbindung*)

> **!** Die Firewall muss Verbindungen vom Server zu Port $N+1$ zulassen.

FTP — Passive Mode Hierbei baut der Client eine TCP-Verbindung von einem beliebigen unprivilegierten TCP-Port N zu Port 21 des Servers auf. (*Steuerverbindung*)
Vor der tatsächlichen Dateiübertragung sendet der Client dann den Befehl `PASV` an den Server. Der Server wartet auf einem beliebigen unprivilegierten Port P auf eingehende Verbindungen vom Client und teilt dem Client **in der Antwortnachricht** des Befehls diesen Port mit. *Der Client baut nun eine TCP-Verbindung von seinem lokalen Port* $N+1$ *zu Port* P *des Servers auf.* (*Datenverbindung*)

2.1.3 E-Mail (SMTP)

SMTP wurde durch die RFC 821 und 1982 spezifiziert. Es dient der Übertragung von ASCII-Nachrichten, obwohl die Nachrichten auch Medien über den MIME-Standard enthalten können.

Versendet werden diese mit SMTP über TCP[2] über Port 587, wobei Mailserver untereinander über Port 25 kommunizieren. Lediglich das Abholen neuer Nachrichten muss mittels POP3, IMAP oder HTTP geschehen.
Drei Phasen zur Übertragung:

(1) Handshaking (2) Nachrichtenübertragung (3) Abschlussphase

Wege einer E-Mail Meist gibt es mehrere Nutzer pro Mail-Server, man spricht hier von *mail user agents (MUA)*, sprich von Clienten zum Verfassen, Editieren und Lesen von Nachrichten. Ebenso können aus- und eingehende Nachrichten auf dem Server gespeichert werden.
Mailserver haben einen Briefkasten (*mailbox*) für (ungelesene) eingegangene Nachrichten für jeden Benutzer, sowie Warteschlangen für ausgehende Nachrichten. Zwischen Mailservern zum Nachrichtenaustausch wird ebenfalls SMTP verwendet, dabei agiert der *sendende Mailserver* als Client, während der *empfangende Mailserver* als Server agiert.

SMTP und Nachrichtenformate SMTP verwendet persistente Verbindung, ebenso müssen Nachrichten (Kopf und Rumpf) als 7-bit-ASCII-Text vorliegen. Dabei sind einige *spezielle Zeichenketten* (`<CR><LF>.<CR><LF>`) verboten, weswegen Nachrichten codiert werden. Diese Codierung erfolgt typischerweise in Base-64 oder Quoted Printable. Der Server nutzt (`<CR><LF>.<CR><LF>`) dann zum Erkennen des Endes einer Nachricht.
In RFC 5321 wird das Nachrichtenformat standardisiert definiert. Nach einer Reihe an Kopfzeilen, welche unter anderem den Empfänger, Absender und den Betreff enthält. **Diese können sich von den SMTP-Befehlen davor unterscheiden!**
Nach einer Leerzeile folgt dann der Rumpf der Nachricht, die „Nachricht“ selbst. Diese ist als **reiner ASCII-Text** zu senden, für andere Codierungen braucht es dann MIME.

Vergleich von SMTP und HTTP

SMTP	HTTP
`push` — Unaufgefordertes Senden	`pull` — Anfordern des Empfängers
Interaktion mit ASCII-Befehlen, Antworten und Statuscodes	
Multipart-Nachrichten für mehrere Objekte in einer Nachricht	Jedes Objekt hat eine eigene Antwortnachricht

Multimediaerweiterung MIME Mit den *Multipurpose Internet Mail Extensions (MIME)* (RFC 2045) werden zusätzliche Kopfzeilen im Nachrichtenkopf eingeführt, durch welche den MIME Content Type bestimmen. Den Beginn macht dabei die MIME-Version durch `MIME-Version:`. Danach kommen weitere Kopfzeilen, sowie die codierten Daten. Ebenso sind Multipart-Nachrichten möglich, also Nachrichten mit unterschiedlichen Codierungen, welche über **eine** Verbindung **gleichzeitig** übertragen werden. Der MIME-Typ ist `multipart/mixed`, man grenzt über das in `boundary` spezifizierte Trennzeichen ab (`-<TRENNZEICHEN>`) und beendet die Nachricht durch weitere zwei Striche (`-<TRENNZEICHEN>-`).
Ein beispielhafter Dialog (entnommen aus SP2-Übung):

```
220 faui03.informatik.uni-erlangen.de ESMTP spoken here
HELO faui06j.informatik.uni-erlangen.de
250 faui03.informatik.uni-erlangen.de
MAIL FROM:<be15piel@stud.informatik.uni-erlangen.de>
250 2.1.0 Ok
```

[2] Dies kann lesbar oder mittels STARTTLS verschlüsselt passieren.

```
RCPT TO:<hansdampf@example.org>
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
From: Max Mustermann <be15piel@stud.informatik.uni-erlangen.de>
To: RK Example <supiduperrk@example.org>
Subject: Hallo Beispiel
Dies ist der Body der Mail. Auch hier werden Zeilen mit \r\n getrennt.
..Diese Zeile beginnt in der Eingabe mit nur einem Punkt.
.
250 2.0.0 Ok: queued as 65AC33E9A9
QUIT
221 2.0.0 Bye
```

POP3 Nach der **Autorisierungsphase**, in welcher sich der Client mittels `user` und `pass` anmeldet (mögliche Antworten des Servers sind `+OK` oder `-ERR` mit jeweiliger Statusmeldung), kommt die **Transaktionsphasse**, in welcher der Client die Nachrichten verwalten kann. Dazu gibt es unter anderem die Befehle `list` für eine Auflistung der Nachrichtennummern, `retr` für die Abholung einer Nachricht mit einer bestimmten Nummer, `dele` zum Löschen einer Nachricht und `quit` zum Beenden der Kommunikation. Dabei wartet der Server auf eine eingehende Verbindung und schließt diese zum Schluss nach `quit` wieder.
POP3 unterscheidet zwischen zwei Modi. Einerseits gibt es den *„download and delete“*-Modus, denn der Nutzer kann eine Nachricht nicht mehrfach abholen und lesen, wenn er einen anderen Mailclient verwendet. Andererseits gibt es noch den *„download and keep“*-Modus, in welchem Kopien der Nachrichten in verschiedenen Clients möglich sind.
POP3 ist über Sitzungen hinweg **zustandslos** und verwendet üblicherweise TCP-Verbindungen auf Port 110. Eine verschlüsselte Variante mit STARTTLS oder SSL/TLS gibt es meist auf Port 995.

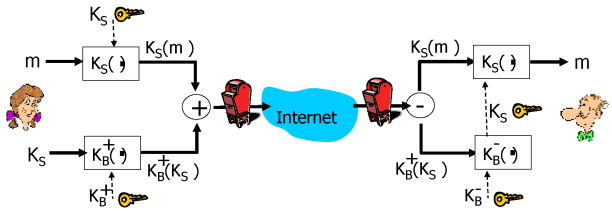
IMAP IMAP besitzt drei Phasen, die **Authorisierungsphase**, die **Transaktionsphase**, sowie die **Abmeldephase**. Während der **Authorisierungsphase** müssen eine Befehls-ID, der Nutzer, sowie das Passwort übergeben werden (`login`). Die Antworten des Servers enthalten *dieselbe* Befehls-ID, sowie ein `OK` bei Erfolg oder ein `NO/BAD` bei Misserfolg.
Während der **Transaktionsphase** wird die Inbox gewählt, sowie Mails verwaltet.
Während der **Abmeldephase** beendet der Client sowie der Server nach einem erfolgreichen `logout` die Verbindung.
IMAP behält im Gegensatz zu POP3 alle Nachrichten an einer zentralen Stelle auf dem Server. Dies erlaubt es dann auch, Nachrichten in Verzeichnisse einzuordnen. IMAP behält dann den Zustand über Sitzungen hinweg, ist also ein *zustandsbasiertes* Protokoll, und verwendet üblicherweise TCP-Verbindungen auf Port 143. Eine verschlüsselte Variante mit STARTTLS oder SSL/TLS gibt es meist auf Port 993.

Sichere E-Mails

E-Mail

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet,* 7th Ed., Pearson Education, 2017.

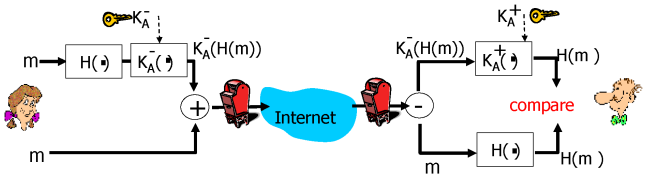
- Sichere E-Mail: Vertraulichkeit
  - Erzeugung eines symmetrischen Schlüssels K_S
  - Verschlüsselung der E-Mail mit K_S
  - asymmetrische Verschlüsselung von K_S mit dem öffentlichen Schlüssel K_B^+ von Bob



E-Mail

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet,* 7th Ed., Pearson Education, 2017.

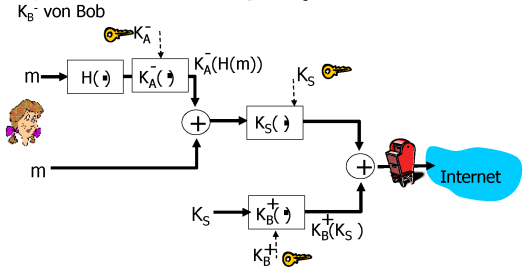
- Sichere E-Mail: Datenintegrität
  - Erzeugung eines Hashwerts der E-Mail
  - Signierung mit dem privaten Schlüssel K_A^- von Alice
  - keine Verschlüsselung der E-Mail



E-Mail

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet,* 7th Ed., Pearson Education, 2017.

- Sichere E-Mail: Vertraulichkeit und Datenintegrität
  - Erzeugung eines Hashwerts der E-Mail
  - Signierung mit dem privaten Schlüssel K_A^- von Alice
  - Verschlüsselung der E-Mail und der Signatur mit K_S
  - asymmetrische Verschlüsselung von K_S mit dem öffentlichen Schlüssel K_B^+ von Bob



E-Mail

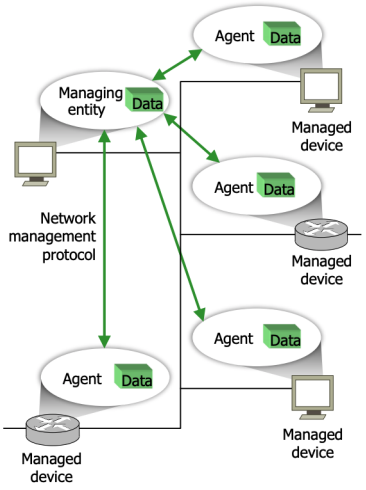
- Sichere E-Mail: Standards
  - S/MIME, RFC 2630, 2632, 2633
    - Erweiterung des Multipurpose Internet Mail Extension (MIME) Standards, bei eine E-Mail aus verschiedenen Teilen mit Kopffeldern bestehen kann
    - Erweiterungen für die meisten Mail-Clients
    - erfordert X.509-Zertifikate, mehrere CAs möglich
  - Pretty Good Privay (PGP)
    - bekannte Public Domain Software
    - Web of Trust: Vertrauensnetz mit transitiven Vertrauensbeziehungen und Algorithmus zur Bestimmung der Vertrauenswürdigkeit (Trust-Level)

2.1.4 Netzwerkmanagement

Zu den Aufgaben des Netzwerkmanagements gehört die Überwachung und Verwaltung eines Netzwerks, welches ein komplexes Hardware/Software-Gebilde sein kann und aus zahlreichen Geräten, Leitungen, Datenstrukturen oder ähnlichem bestehen kann. Dazu gibt es beispielsweise das von der ISO spezifizierte FCAPS-Modell (*fault, configuration, account, performance, and security management*) für Fernmeldekontrollnetze. Eine andere Variante ist das Simple Network Management Protocol (SNMP). Inzwischen ist es in der dritten Version verfügbar, bleibt dabei allerdings einfach und verbreitet. Anwendung findet es für Unternehmen mit einer großen Anzahl an Geräten. Es ermöglicht dem Administrator eine proaktive Überwachung der Geräte im Netzwerk zum einfachen Beheben von Problemen.
SNMP wird von Managementsoftware verwendet, wie zum Beispiel Spiceworks, Getif oder LANView, welche SNMP für eine Kommunikation mit den Geräten im Netzwerk und für das Sammeln von Informationen verwenden. Dabei benötigen Geräte allerdings SW-Agenten, welche dann gemäß SNMP kommunizieren.

SNMP — Organisationsmodell

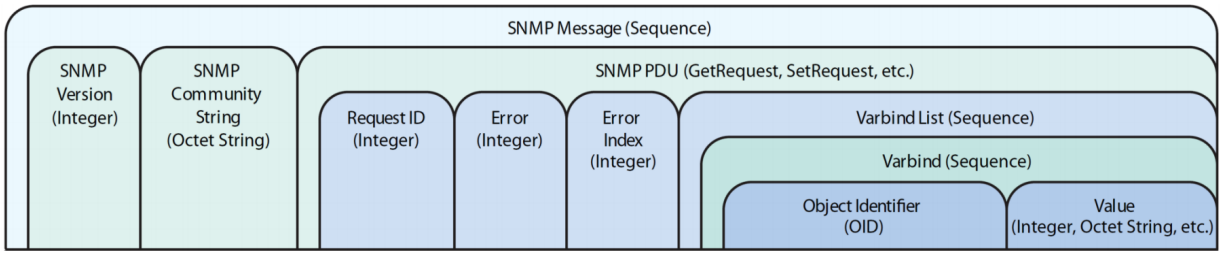
Der Manager (*managing entity*) ist ein Prozess auf einer zentralen Managementstation (entspricht dem Client). Im Netz gibt es dann verschiedene Geräte, die *managed devices*, sowie in diesen Hardware oder Software, sogenannte *managed objects*. Ein Prozess auf einem *managed device*, welcher lokale Aktionen ausführen kann ist der *management agent* (entspricht dem Server).
Dabei wird ein Anfrage/Antwort-Protokoll zwischen Manager und Agent über UDP verwendet. Der Manager entscheidet dann basierend auf Regeln, was zu tun ist.



SNMP — Nachrichtentypen Es gibt verschiedene Nachrichtentypen:

- **Get-Request**s werden vom Manager zum Agent gesendet, um Daten vom Agent zu erhalten.
- **Get-Next-Request**s werden vom Manager zum Agent gesendet, um den nächsten Datensatz zu erhalten. Sie sind damit essentiell für den Zugriff auf sequentielle Datensätze.
- **Get-Bulk-Request**s werden vom Manager zum Agent gesendet, um mehrere Datensätze auf einmal zu erhalten.
- **Set-Request**s werden vom Manager zum Agent gesendet, um den Wert eines Datensatzes zu initialisieren oder zu ändern.
- **Response**s werden vom Agent zum Manger als Antwort auf Get- oder Set-Nachrichten gesendet.
- **Trap**s werden vom Agent zum Manager als unaufgeforderte Nachrichten über Fehlersituationen gesendet.

SNMP — Nachrichtenformat



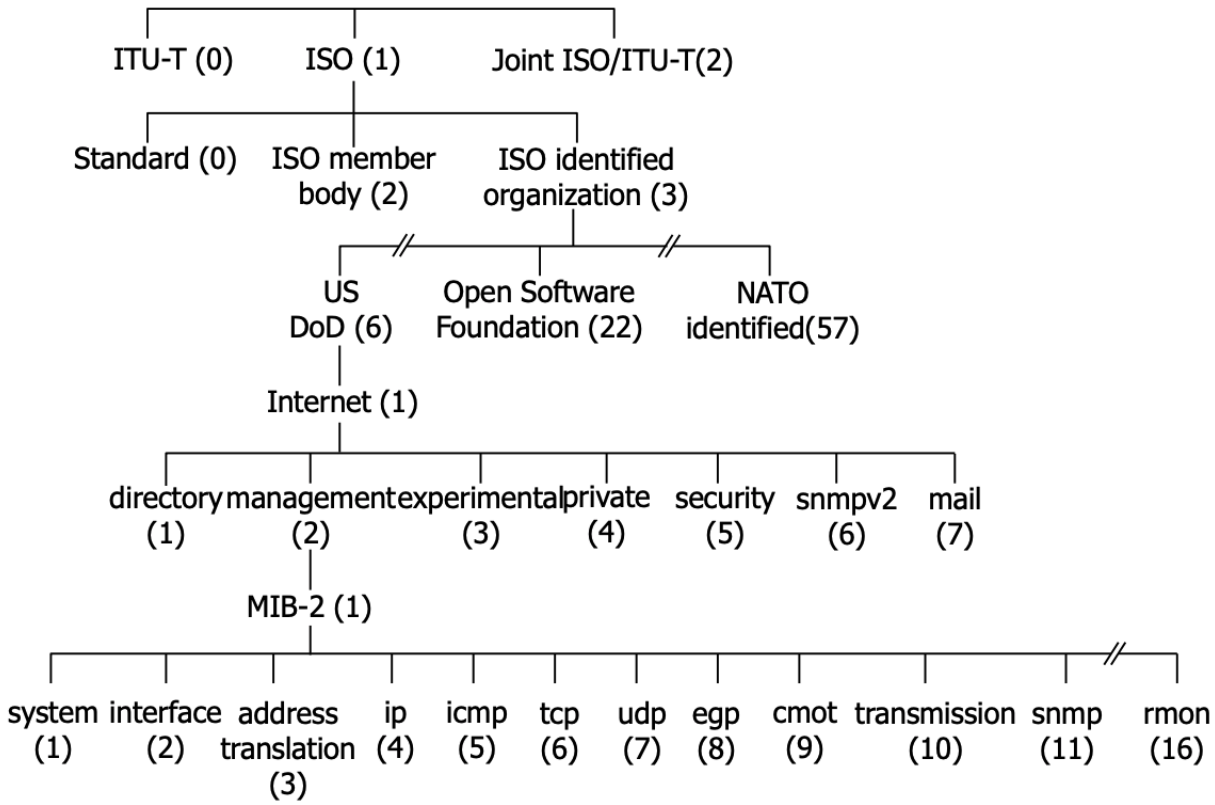
Management Information Base (MIB) MIB-Module enthalten Datenstrukturen, welche wiederum von der IETF genormt sind, für die *managed objects*. Die Syntax wird dabei in *Structure of Management Information (SMI)* der IETF festgelegt, welche wiederum die ASN.1 der ISO benutzt.

- **Basisdatentypen**: `INTEGER, OCTET STRING, OBJECT IDENTIFIER, COUNTER64, ...`
- **Sequenzen**: `SEQUENCE OF (andere MIB Objekte)`

ASN.1 besitzt auch ein Nummerierungsschema zur eindeutigen **Objektidentifizierung (OID)**, womit jedes MIB-Modul eindeutig bezeichnet wird. Das genau binäre Format für die Übertragung wird noch mit den **Basic Encoding Rule (BER)** festgelegt. Beispiel:

```
UDP-MIB DEFINITIONS ::= BEGIN
...
udp OBJECT IDENTIFIER ::= { mib-2 7 }
...
udpInDatagrams OBJECT-TYPE
        SYNTAX Counter32
        MAX-ACCESS read-only
        STATUS current
        DESCRIPTION
                "The total number of UDP datagrams delivered to UDP users."
        ::= { udp 1 }
...
udpGroup OBJECT-GROUP
        OBJECTS { udpInDatagrams, udpNoPorts,
                  udpInErrors, udpOutDatagrams,
                  udpLocalAddress, udpLocalPort }
        STATUS current
        DESCRIPTION
                "The udp group of objects providing for management of UDP
                entities."
        ::= { udpMIBGroups 1 }
END
```

Objekt-Identifizierung (OID) Der Object Identifier (OID) ist ein Konzept das im Simple Network Management Protocol (SNMP) definiert ist. Es ist der Wert, der in bestimmten Modulen der Management Information Base (MIB) zur eindeutigen Identifizierung spezieller SNMP-Objekte benutzt wird. Die OID-Identifikation selbst ist eine Sequenz von Buchstaben und Ziffern, mit der das Objekt eindeutig gekennzeichnet wird.



Netzwerkmanagement

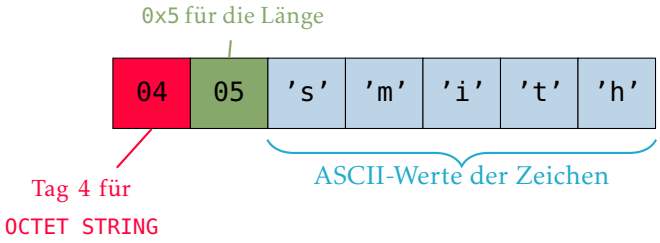
Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

- MIB-Modul für UDP

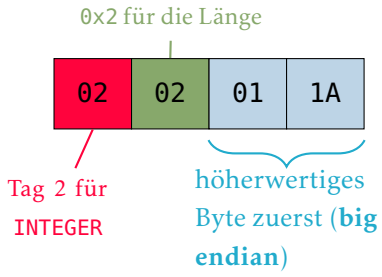
Object Identifier	Name	Type	Description (from RFC 2013)
1.3.6.1.2.1.7.1	udpInDatagrams	Counter32	"total number of UDP datagrams delivered to UDP users"
1.3.6.1.2.1.7.2	udpNoPorts	Counter32	"total number of received UDP datagrams for which there was no application at the destination port"
1.3.6.1.2.1.7.3	udpInErrors	Counter32	"number of received UDP datagrams that could not be delivered for reasons other than the lack of an application at the destination port"
1.3.6.1.2.1.7.4	udpOutDatagrams	Counter32	"total number of UDP datagrams sent from this entity"
1.3.6.1.2.1.7.5	udpTable	SEQUENCE of UdpEntry	"a sequence of UdpEntry objects, one for each port that is currently open by an application, giving the IP address and the port number used by application"

Rechnerkommunikation, Anwendungsschicht 50

Basic Encoding Rules (BER) Die **Basic Encoding Rules (BER)** sind eine Repräsentation zur Übertragung von Informationen. Für jedes Datum wird das Tripel **Tag, Length, Value (TLV)** übertragen. Der Tag ist dabei eine Nummer für den Datumstyp. Wird beispielsweise „smith" übertragen, so soll nach BER



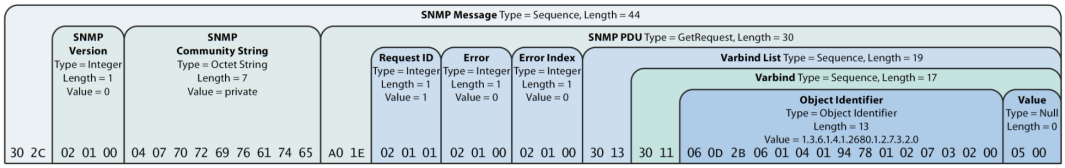
übertragen werden. Wird beispielsweise der Integer 282 (`0x011A`) übertragen, so soll nach BER



übertragen werden.

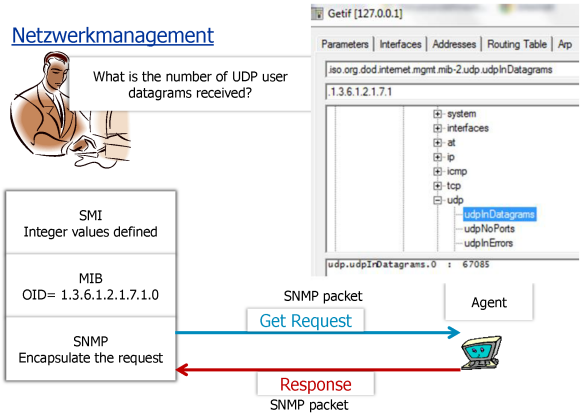
Netzwerkmanagement

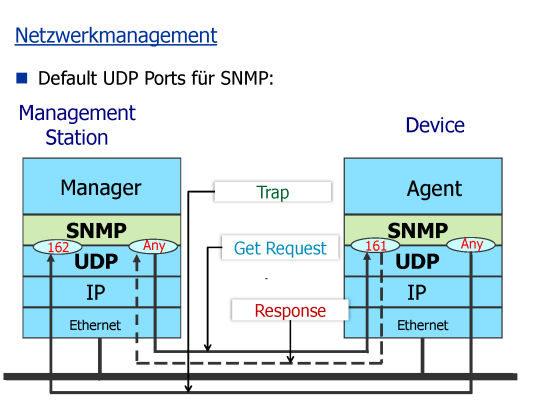
- Format von SNMP Nachrichten
  - Get-Request mit OID und BER:



Quelle: RaneNote: "SNMP: Simple? Network Management Protocol"

  - zum Verständnis, besondere Kodierungsregeln für OIDs:
    - TLV-Tripel beginnt mit Tag-Wert 06_{16}
    - die beiden ersten Werte werden in ein Byte kodiert, indem der erste Wert mit 40 multipliziert wird und der zweite addiert wird, z.B. für 1.3. ergibt sich $1_{10}*40_{10}+3_{10} = 43_{10} = 2B_{16}$
    - Short Form: Werte ≤ 127 werden in ein Byte kodiert, Bit 7 wird auf Null gesetzt
    - Long Form: Werte > 127 werden in mehrere Bytes kodiert: nur Bits 0 bis 6 tragen kodierte Werte, Bit 7 aller Bytes bis auf das letzte wird auf Eins gesetzt, z.B.: $9478_{16} = 10010100\ 01111000_2$, Bit 7 in jedem Byte eliminieren, Bits 0 bis 6 führen zu $00001010\ 01111000_2 = 10_{10}*256_{10}+7_{10}*16_{10}+8_{10}*1_{10} = 2680_{10}$





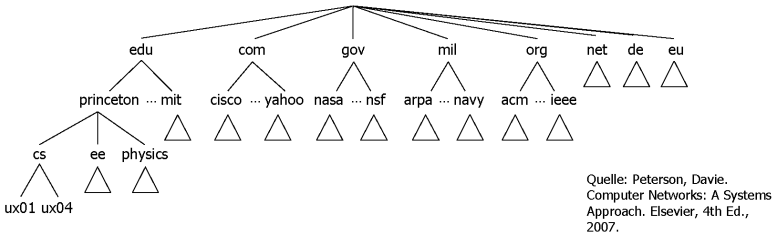
2.1.5 Domain Name System — DNS

Das *Domain Name System (DNS)* ist eine verteilte Datenbank, welche aus vielen Namenservern, welche über ein Anwendungsprotokoll (DNS) kommunizieren, besteht. Im Allgemeinen sind Host- und Domainnamen lesbar. DNS bildet diese Namen dann auf Werte (unter anderem IP-Adressen) ab. Damit leistet DNS eine wesentliche Aufgabe, um Infrastrukturen zu nutzen.

Domain-Struktur

DNS

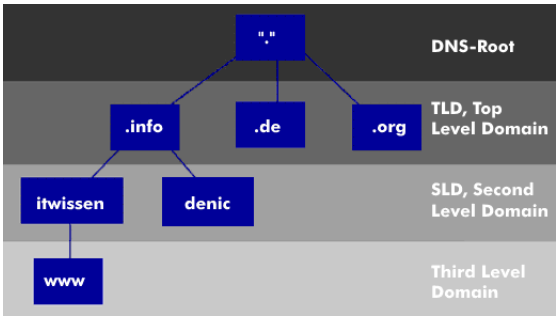
- Domain-Struktur
  - DNS implementiert hierarchischen Namensraum für Internet-Objekte
  - von links nach rechts lesen, von rechts nach links verarbeiten
  - eine Zone wird von einem Name-Server verwaltet
  - die Hierarchie wird durch die Name-Server implementiert



Quelle: Peterson, Davie. Computer Networks: A Systems Approach. Elsevier, 4th Ed., 2007.

Hierarchie von Namensservern

Ganz oben in der Hierarchie steht der **root-name-server**, welcher weltweit verfügbar ist. Direkt darunter sind die **top-level-domain-server** (TLD) für die verschiedenen Top-Level-Domains angeordnet. Danach kommen die **second-level-domain-server**, wonach wiederum weitere Server von niedrigerem Level kommen. Die Hierarchie beendet ein **autoritativer Namensserver**, welcher die unterste Ebene darstellt und nur für eine einzelne Organisation steht.
Insgesamt gibt es 13 **root-name-server** weltweit. Die Lastverteilung erfolgt mittels Anycast, wobei die meisten durch viele Server realisiert werden. Insgesamt gibt es mehrere hundert Server.



DNS — Resource Records **Resource Records** sind die Datensätze der Namensserver. Sie bestehen aus einem 4-Tupel aus Domainname, Wert, Typ und TTL.

DNS — Resource Records: TTL Die **time to live (TTL)** gibt die Gültigkeit des Eintrags an.

DNS — Resource Records: Typ Es gibt verschiedene Typen:

- **Typ = A**: Der Wert ist dann eine IPv4-Adresse. Als Beispiel: `(ns.cisco.com, 128.96.32.20, A, TTL)` **Für IPv6-Adressen gibt es AAAA-Einträge**.

- **Typ = NS**: Der Wert ist dann ein Domainname eines Hosts, auf welchem wiederum ein Namensserver läuft, welcher Namen in der Domain auflösen kann. Als Beispiel: `(princeton.edu, cit.princeton.edu, NS, TTL)`
- **Typ = CNAME** (*canonical name*): Der Wert ist der **kanonische Name** eines Hosts, was **Aliasnamen** ermöglicht. Als Beispiel: `(cic.cs.princeton.edu, cicada.cs.princeton.edu, CNAME, TTL)`
- **Typ = MX** (*mail exchange*): Der Wert ist der Domainname des Hosts, auf welchem ein Mailserver läuft. Als Beispiel: `(cs.princeton.edu, optima.cs.princeton.edu, MX, TTL)`

DNS — Resource Records: Beispiel

DNS: Resource Records

- Bsp: Resource Records
  - Root Name Server

    (princeton.edu, cit.princeton.edu, NS, TTL)
    (cit.princeton.edu, 128.196.128.233, A, TTL)
    (cisco.com, ns.cisco.com, NS, TTL)
    (ns.cisco.com, 128.96.32.20, A, TTL)
    …

  - enthält einen NS-Datensatz für jeden Server der nächsten Ebene und einen A-Datensatz mit der IP-Adresse
  - diese bilden zusammen einen Verweis auf die Server der zweiten Ebene

Quelle: Peterson, Davie. Computer Networks: A Systems Approach. Elsevier, 4th Ed., 2007.

Rechnerkommunikation, Anwendungsschicht 60

DNS: Resource Records

- Server von princeton.edu

  (cs.princeton.edu, optima.cs.princeton.edu, NS, TTL)
  (optima.cs.princeton.edu, 192.12.69.5, A, TTL)
  (ee.princeton.edu, helios.ee.princeton.edu, NS, TTL)
  (helios.ee.princeton.edu, 128.196.28.166, A, TTL)
  (jupiter.physics.princeton.edu, 128.196.4.1, A, TTL)
  (saturn.physics.princeton.edu, 128.196.4.2, A, TTL)
  (mars.physics.princeton.edu, 128.196.4.3, A, TTL)
  (venus.physics.princeton.edu, 128.196.4.4, A, TTL)

- einige Datensätze sind Verweise auf die dritte Ebene, einige lösen die IP-Adressen direkt auf

Quelle: Peterson, Davie. Computer Networks: A Systems Approach. Elsevier, 4th Ed., 2007.

Rechnerkommunikation, Anwendungsschicht 61

DNS: Resource Records

- Server der Domain cs.princeton.edu

  (optima.cs.princeton.edu, 192.12.69.5, A, TTL)
  (cheltenham.cs.princeton.edu, 192.12.69.60, A, TTL)
  (baskerville.cs.princeton.edu, 192.12.69.35, A, TTL)
  (che.cs.princeton.edu, cheltenham.cs.princeton.edu, CNAME, TTL)
  (opt.cs.princeton.edu, optima.cs.princeton.edu, CNAME, TTL)
  (bas.cs.princeton.edu, baskerville.cs.princeton.edu, CNAME, TTL)
  (www.cs.princeton.edu, optima.cs.princeton.edu, CNAME, TTL)
  (cs.princeton.edu, optima.cs.princeton.edu, MX, TTL)

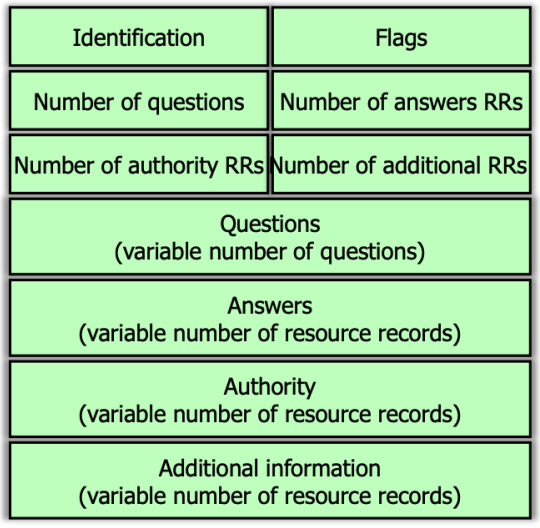
- enthält A-Datensätze für alle Hosts
- Aliasnamen: praktischere Namen, erlaubt Flexibilität, z.B. für Web-Server
- MX-Datensätze: gleicher Zweck speziell für Mail-Server

Quelle: Peterson, Davie. Computer Networks: A Systems Approach. Elsevier, 4th Ed., 2007.

Rechnerkommunikation, Anwendungsschicht 62

DNS — Protokoll

Sowohl Anfrage- als auch Antwortnachrichten haben dasselbe Format, was rechts dargestellt ist. Der Kopf besteht aus der *identification*, einer Zuordnung einer Anfrage zu einer Antwort, sowie *flags*, welche die Art der Anfrage/Antwort spezifizieren, sowie der Anzahl an Fragen, Antworten, Resource Records, welche Autoritäten wiedergeben, sowie der Anzahl an zusätzlichen RRs.

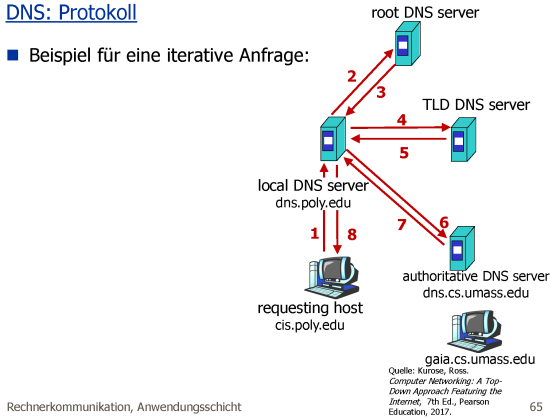


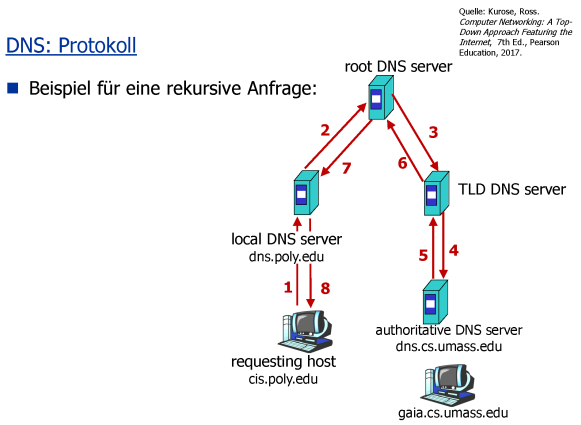
Die *flags* enthalten Sektionen von entweder einem oder vier Bits, welche den Typ der Nachricht, ob der Namensserver eine Autorität darstellt, ob die Anfrage rekursiv war oder nicht, ob die Antwort abgeschnitten wurde und den Status der Anfrage darstellt.
Der Rumpf enthält die *question section*, welche den Domainnamen und Typ des RR (A, AAAA, MX, TXT, etc.) darstellt. Jedes Label im Domainnamen ist mit der Länge als Prefix versehen. Zudem enthält der Rumpf noch die *answers section*, welche die RR des gesuchten Namens enthält. Ebenso die *authority section*, welche Antworten von autoritativen Servern enthält.

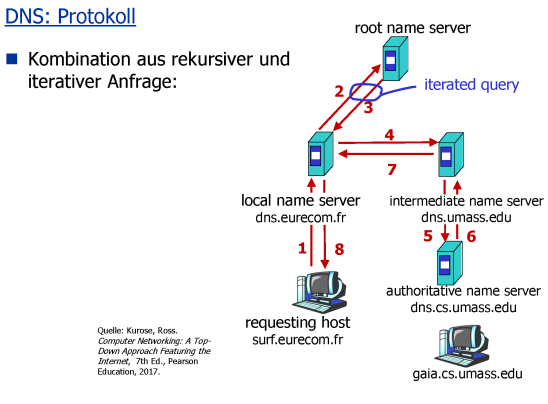
DNS — Anfragearten Es gibt zwei Arten von Anfragearten, die *iterative* und *rekursive*.

- **iterativ**: Die Antwort ist ein anderer Server, welche den Namen eventuell auflösen kann, oder aber es gibt keine Antwort. Dementsprechend nur NS- und A-/AAAA-Datensätze. Die Antwort wird sofort geliefert, es muss keine Information gespeichert werden. Dies ist **vorteilhaft für hochfrequentierte Server**
- **rekursiv**: Die Antwort ist die Auflösung des Namens, welche unter Umständen von anderen Servern geholt wird. Dementsprechend gibt es nur A/AAAA-Datensätze. Bei Anfrage an einen anderen Server muss die Information hier serverseitig zwischengespeichert werden.

DNS — Beispiele für Anfragearten







2.1.6 Content Distribution Networks

Ziel eines *content distribution/delivery networks (CDN)* ist es Wartezeiten beim Laden von Webseiten auch bei **Flash-Crowds**[3] zu reduzieren. Dazu verwendet man statt einem singulärem massiven

[3]Millionen Benutzer greifen auf eine Seite zu

Rechenzentrum mehrere Spiegelserver, welche geographisch verteilt sind und näher beim Nutzer liegen. Damit werden drei Probleme vermieden:

+ Engpässe auf der Strecke zum Nutzer werden minimiert (erste Meile, letzte Meile, sowie die Peering-Punkte).
+ Es findet keine mehrfache Übertragung desselben Inhalts über eine Strecke statt.
+ Ein *single-point-of-failure* (siehe Seite G73) wird verhindert.

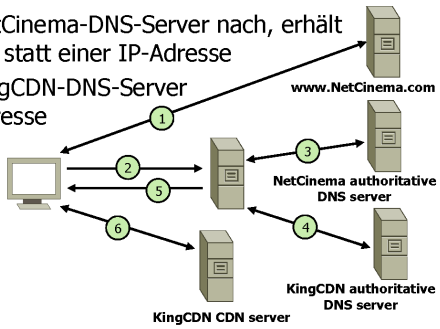
Es wird ein ähnliches Konzept wie auch bei Webcaches verwendet, der Inhalt wird aber abhängig von der Beliebtheit *proaktiv* von anderen Servern und gegebenenfalls auch von der Zentrale repliziert. Bekannte CDNs sind bspw. 3rd-party (Akamai, Level-3) oder auch private (Google mit YouTube).
Anfragen an den Server werden dann verteilt, man unterscheidet:

– **Serverbasierte HTTP-Umleitung**: Der Server liefert aufgrund der IP-Adresse des Clients einen geeigneten anderen Server. **Nachteil:** Dies erfordert eine zusätzliche RTT und es besteht die Gefahr der Überlast für den Server.
– **Clientnahe HTTP-Umleitung**, wird beispielsweise durch Web-Proxys umgesetzt, ist aber schwieriger zu verwirklichen.
– **DNS-basierte Umleitung**: Die DNS-Server bilden den Domainnamen des Dienstes gegebenenfalls auf andere Domainnamen und zuletzt auf die IP-Adresse eines geeigneten Servers ab.
– **URL-Rewriting**: Server liefert Basisseite, die URLs der eingebetteten Objekte werden umgeschrieben, mit dem Domainnamen eines geeigneten anderen Servers.

Kommerzielle CDNs verwenden meist eine Kombination aus DNS-basierter Umleitung und URL-Rewriting. Die Wahl geeigneter Server erfolgt nach geographischer Nähe, geringer Auslastung und weiteren ähnlichen Merkmalen.

Content Distribution Networks

- Bsp. für DNS-basierte Redirection bei 3rd-party-CDN
  - www.NetCinema.com ist Video-Dienst, Video-Dateien werden auf den CDN-Servern von kingcdn.com verteilt
  - 1. HTTP-Anfrage für NetCinema-Seite mit Links zu Videos
  - 2. Nutzer klickt auf http://video.netcinema.com/video1234, DNS-Anfrage für IP-Adresse von video.netcinema.com
  - 3. Lokaler DNS-Server fragt bei NetCinema-DNS-Server nach, erhält Hostnamen video-cdn.kingcdn.com statt einer IP-Adresse
  - 4. Lokaler DNS-Server fragt bei KingCDN-DNS-Server nach, erhält aufgrund seiner IP-Adresse IP-Adresse eines geographisch nahe gelegenen CDN-Servers
  - 5. Lokaler DNS-Server gibt IP-Adresse des auserwählten CDN-Servers an Nutzer-PC weiter
  - 6. HTTP-Anfrage an diesen Server



Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

2.2 Socket-Programmierung

2.2.1 TCP-Verbindungen

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

/*
 * Quellcode eines TCP Servers. Erstellt einen Socket auf Port {port} und akzeptiert eine
 * Verbindung. Anschliessend wird die Verbindung abgearbeitet und anschliessend die Verbin-
 * dung geschlossen. Im Anschluss wird auf eine neue Verbindung gewartet.
 */
public class TCPServer {

    private static int port = 2018;

    public static void main(String args[]) throws IOException {

        // Schritt 1: Erstelle neuen /*nbServerSocket*/ auf Port port
        ServerSocket welcomeSocket = new ServerSocket(port);

        // Schritt 6: Soll der Server mehr als eine Verbindung aufnehmen,
        //            so ist hier eine Schleife von Noeten. Neben true
        //            koennen auch Bedingungen hin.
        while (true) {
            // Schritt 2: Akzeptiere eine neue Verbindung
            /*+ An dieser Stelle erfolgt der TCP-Verbindungsaufbau! */
            Socket socket = welcomeSocket.accept();

            // Schritt 3c: Falls mehrere Zeilen geschickt werden
            //             (beispielsweise durch einen flush()-
            //             Aufruf), so muss hier eine Schleife hin.
            while ( condition_for_looping ) {

                // Schritt 3a: Bekomme einen InputStream zum Lesen,
                //             falls mehrere gewuenscht sind, muss
                //             dies in einer Schleife passieren.
                DataInputStream inFromClient = new DataInputStream(socket.getInputStream());

                // Schritt 3b: Bekomme einen OutputStream zum Schreiben,
                //             falls mehrere gewuenscht sind, muss
                //             dies in einer Schleife passieren.
                DataOutputStream outToClient = new DataOutputStream(socket.getOutputStream());

                // Schritt 4: Arbeit mit den Daten
                // ...
            }

            // Schritt 5: Schliesse die Verbindung
            /*+ An dieser Stelle erfolgt der TCP-Verbindungsabbau! */
            socket.close();
        }
    }
}
```

Quellkode 2.1 — TCP-Server

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

/*
 * Quellcode eines TCP Clients.
 */
public class TCPClient {

    private static String servername = "Hier steht der Servername";
    private static int port = 2018;
```

```
12
13      public static void main(String args[]) throws IOException {
14
15          // Schritt 1: Erstelle neuen Socket auf Port port, welcher eine Verbin-
16          //            dung zum Server servername aufbaut.
17          /*+ An dieser Stelle erfolgt der TCP-Verbindungsaufbau! */
18          Socket socket = new Socket(servername, port);
19
20          // Schritt 2a: Bekomme einen OutputStream zum Schreiben.
21          DataOutputStream outToServer = new DataOutputStream(socket.getOutputStream());
22
23          // Schritt 2b: Bekomme einen InputStream zum Lesen. UTF-8 steht dabei fuer
24          //             das Encoding der Nachrichten.
25          InputStreamReader input = new InputStreamReader(socket.getInputStream(), "UTF-8");
26          BufferedReader inFromServer = new BufferedReader(input);
27
28          // Schritt 3: Arbeite, schicke Daten zum Server, empfange Daten vom Server.
29          // ...
30          /*- Achtung: Tritt folgendes auf, so muss der Server ueber Eingaben loopen. */
31          outToServer.flush();
32
33          // Schritt 4: Schliesse die Verbindung
34          /*+ An dieser Stelle erfolgt der TCP-Verbindungsabbau! */
35          socket.close();
36      }
37  }
```

Quellkode 2.2 — TCP-Client

2.2.2 UDP-Verbindungen

```
1   import java.net.DatagramPacket;
2   import java.net.DatagramSocket;
3
4   public class UDPServer {
5
6       static int port = 2018;
7
8       public static void main(String args[]) {
9
10          // Schritt 1: Erstelle neuen DatagramSocket, welcher auf Port port hoert.
11          DatagramSocket serverSocket = new DatagramSocket(port);
12
13          // Schritt 3c: Falls mehrere Zeilen geschickt werden,
14          //             so muss hier eine Schleife hin.
15          while ( condition_for_looping ) {
16              // Schritt 2: Was soll erhalten werden, wie gross in bytes?
17              byte[] receiveData = new byte[???];
18
19              // ...
20              // Schritt 3a: Baue ein leeres Paket der Laenge receiveData.length
21              DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
22              // Schritt 3b: Erhalte ein Paket von der UDP-Verbindung
23              serverSocket.receive(receivePacket);
24              // ...
25          }
26
27          // Schritt 4: Schliesse den Server-Port.
28          serverSocket.close();
29      }
30
31      public static int byteArrayToInt(byte[] b) {
32          /*- Hier stehen die Konvertierungsprozeduren fuer alle moeglichen Typen */
33      }
34  }
```

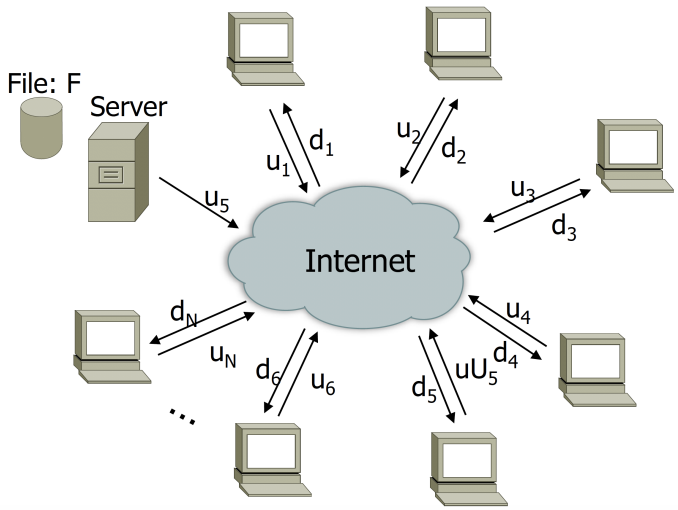
Quellkode 2.3 — UDP-Server

```java
import java.net.*;

/*
 * Quellcode eines UDP Clients.
 */
public class UDPClient {

    private static String servername = "Hier steht der Servername";
    private static final int port = 2018;

    public static void main(String[] args) {
        // Schritt 1: Erstelle neuen DatagramSocket.
        DatagramSocket clientSocket = new DatagramSocket();

        // Schritt 2: Bekomme IP-Adresse des Servers servername.
        InetAddress ipA = InetAddress.getByName(servername);

        // Schritt 3: Was soll gesendet werden und wie gross ist es in bytes?
        byte sendData[] = new byte[???];

        // ...
        // Schritt 4: Erstelle ein Datagramm mit dem zugehoerigen byte-Array,
        //            der IP-Adresse ipA und dem Port port.
        DatagramPacket packet = new DatagramPacket(sendData, sendData.length, ipA, port);

        // Schritt 5: Sende das Datagramm.
        clientSocket.send(packet);
        // ...

        // Schritt 6: Schliesse die Verbindung.
        clientSocket.close();
    }
}
```

Quellkode 2.4 — UDP-Client

2.3 Peer-to-Peer-Systeme

Peer-to-Peer-Netze sind bekannt von Anwendungen zum Filesharing, so sollen die Inhalte nicht nur vom zentralen Server bereitgestellt werden, sondern auch von anderen Peers. Dabei wird die Uploadbitrate der Peers mitgenutzt. Die Poularität von P2P-Netzen ist unter anderem von Anwendungen wie Napster[4] oder diversen Nachfolgern.



Die Peers kommunizieren direkt mittels TCP oder UDP. P2P-Netze bilden ein sogenanntes **Overlay-Netz**, ein logisches Netz aus Peers über dem eigentlichen physikalischen Netz.

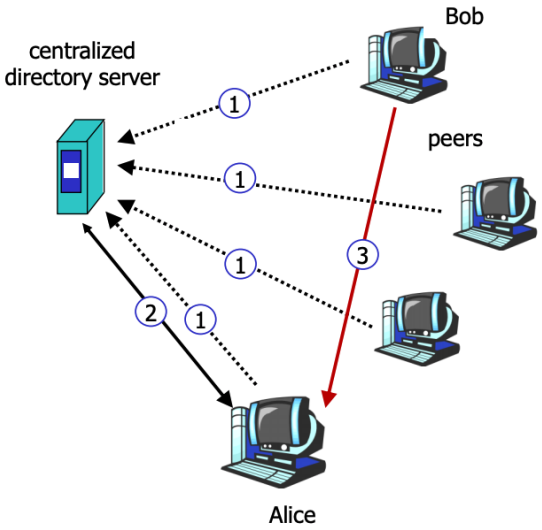
[4] direkter Austausch von Musikdateien, aus juristischen Gründen stillgelegt

2.3.1 P2P — Unstrukturierte Architekturen

Wir unterscheiden zwischen zentralisierten P2P-Netze wie bei Napster, bei welchen der Datenaustausch gemäß P2P erfolgt, das Verzeichnis aber zentral ist, reinen P2P-Netzen wie bei Gnutella 0.4, bei welchen keine Zentrale, dafür aber eine vollständige Vermaschung der Peers stattfindet, sowie hybride P2P-Netze wie bei Gnutella 0.6, bei welchen Inseln mit Serverstruktur existieren, welche wiederum vermascht sind.

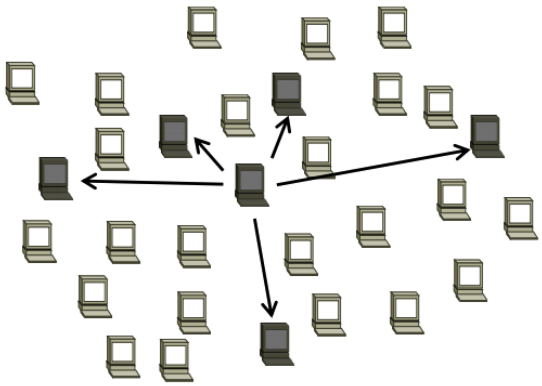
P2P — Zentralisiert

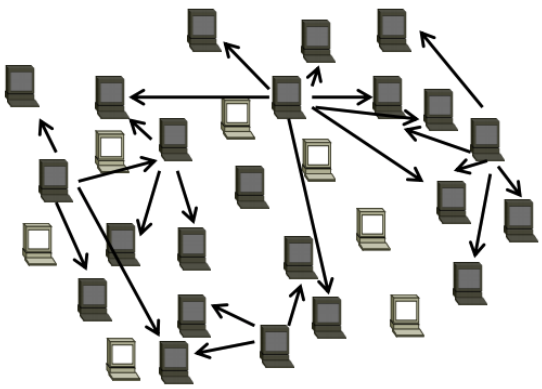
Zentralisierte P2P-Netze haben eine Architektur ähnlich von Napster mit einem zentralisierten Verzeichnis. Bei Eintritt eines Peers informiert dieser den zentralen Server über seine IP-Adresse und seine Inhalte. Die Suche nach Inhalten erfolgt über den zentralen Server, die Datenübertragung direkt zwischen Peers. **Nachteil:** Der zentrale Server stellt hierbei allerdings den juristischen Schwachpunkt, sowie ein Leistungs- und Zuverlässigkeitsproblem.



P2P — Reines P2P

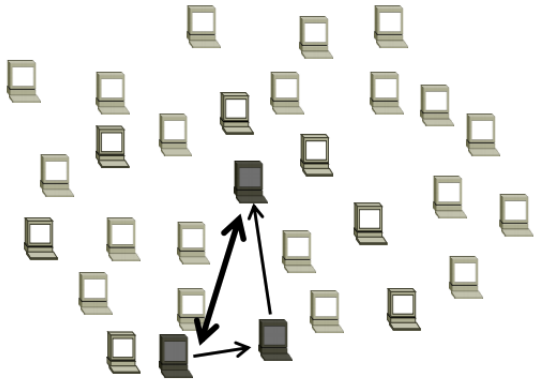
Hierbei findet sich Dezentralität durch das Fluten von Anfragen. Die Peers bilden dabei ein Overlay-Netzwerk über TCP-Verbindungen. Der anfragende Peer sendet eine Anfrage an alle seine Nachbarn im Overlay-Netzwerk, welche dann diese mit den von ihnen angebotenen Inhalten vergleichen. Können sie die Anfrage nicht beantworten kommt es zum **Fluten**; sie wird an mehrere Nachbarn — nicht aber an den Peer selbst — weitergeleitet. Das Fluten wird durch einen maximalen Hopcount begrenzt. Wenn ein Peer den Inhalt anbieten kann, so antwortet er dem anfragenden Peer, welcher wiederum zurückleitet. *Dadurch bleibt die Identität des ursprünglich anfragenden Peers unerkannt*. Die Antwort findet zur Quelle zurück, diese kontaktiert direkt einen der Peers, der die Anfrage beantworten kann, die Übertragung erfolgt dann beispielsweise mittels HTTP.





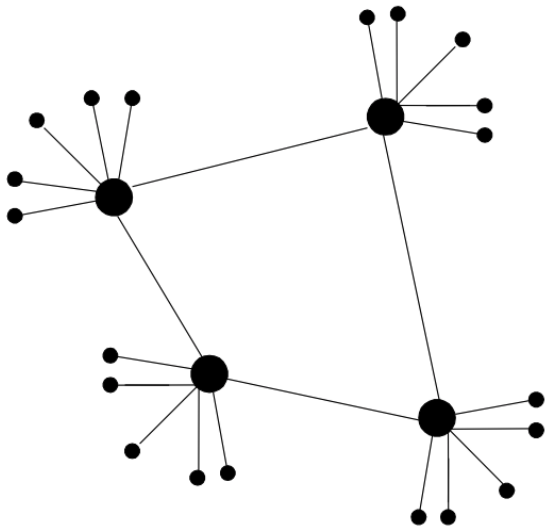
Es wird **kein** zentraler Server benötigt. Die Skalierbarkeit ist wegen des Flutens problematisch.

Beim Eintritt in das Overlay-Netzwerk wird eine Nachricht an eine veröffentlichte Liste von möglichen Peers geschickt.



P2P — Hybrid

Peers bilden hierbei sogenannte Gruppen, wobei einer der Gruppenleiter (*group leader*) ist. Der Gruppenleiter kennt die Inhalte aller Peers aus der Gruppe, man könnte eine Gruppe somit als ein Mininapster ansehen. Das Overlay-Netzwerk ist dann nur zwischen den Gruppenleitern. Der Austausch zwischen den Gruppenleitern ist ähnlich wie bei Gnutella.
Diese Variante besitzt eine bessere Skalierbarkeit und ebenfalls keine zentrale Kontrolle.



2.3.2 P2P — Strukturierte Architekturen

Strukturierte Architekturen sind *distributed hash table* basiert und besitzen keine zentralen Server. Sie sind dafür hochgradig skalierbar. Beispiele hierfür sind Chord und CAN.

P2P: Strukturiert

- Verteilte Hash-Tabellen (Distributed Hash Tables, DHT)
  - dezentrales Verfahren für Speicherung von Datenelementen, bekannt z.B. aus Chord-System
  - Peers bilden ringförmiges Overlay-Netz
  - jedem Peer wird zufälliger Bezeichner p ($0 \le p \le 2^m-1$) aus ringförmigen Bezeichnerraum mit m Bits zugewiesen
  - jedem Datenelement wird mittels Hash-Funktion ein Schüssel k ebenfalls aus diesem Raum zugewiesen
  - Nachfolger von k
    - Datenelement mit Schlüssel k wird auf „Nachfolger" p von k gespeichert (Nachfolger darf Knoten p selbst sein)
    - der nächste Peer im Ring, formal: Peer p mit dem kleinsten $a \ge 0$, so dass $p = succ(k) = (k+a) \bmod 2^m$ existiert (also ringförmig über 2^m-1 hinaus)
    - Auslesen eines Datenelements mit Schlüssel k erfolgt auf Peer succ(k)

P2P: Strukturiert (Beispiel Chord)

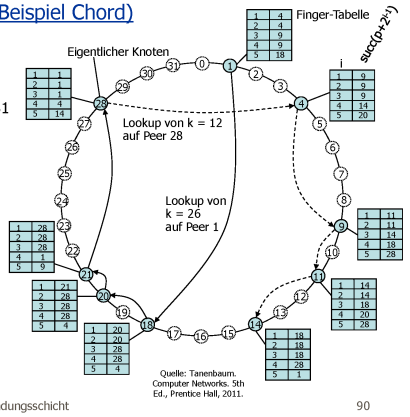
- Routing
  - Finden des gesuchten Eintrags in der DHT
  - Chord verwaltet Ringstruktur über alle Einträge
    - jeweils Vorgänger und Nachfolger
  - Routing entlang des Rings (linearer Aufwand, ineffizient) oder durch Sprungtabellen (Finger-Tabelle, logarithmischer Aufwand)
  - es gibt auch Verweis auf Vorgänger (wird hier vernachlässigt)
- Neuen Knoten einfügen
  - Voraussetzung: ein bekannter Knoten p in der DHT
  - neue ID wählen zwischen p und Nachfolger von p, Aktualisieren der Finger-Tabellen
- Knoten entfernen
  - Daten migrieren, ID entfernen und Finger-Tabellen aktualisieren
- Selbststabilisierung
  - kontinuierliche Überprüfung aller Knoten, evtl. Finger-Tabellen reparieren

P2P: Strukturiert (Beispiel Chord)

- Finger-Tabelle
  - jeder Peer p unterhält Finger-Tabelle FT_p mit m Einträgen
  - i-ter Eintrag der Finger-Tabelle: $FT_p[i] = succ(p+2^{i-1})$ enthält Nachfolger mit exponentiell steigender Distanz
  - Lookup für Datenelement mit Schlüssel k kann auf beliebigem Peer p starten
  - Algorithmus lookup(k,p):
    - wenn k = p, dann gib Peer p als Ergebnis aus
    - wenn k zwischen p (exklusive) und $FT_p[1]$ (inklusive) liegt, dann gib Peer q = $FT_p[1]$ als Ergebnis aus
    - ansonsten wird der Peer in der Finger-Tabelle gesucht, der am nächsten vor k liegt und dort ein erneutes Lookup durchgeführt:
      - wenn k zwischen $FT_p[i]$ (inklusive) und $FT_p[i+1]$ (exklusive) liegt mit 1 < i < m, dann q = $FT_p[i]$
      - wenn k = $FT_p[m]$ ist oder hinter $FT_p[m]$ liegt, dann q = $FT_p[m]$
      - gib von lookup(k,q) gefundenen Peer als Ergebnis aus

P2P: Strukturiert (Beispiel Chord)

- Beispiel
  - m = 5, Bezeichnerraum $0 \le p \le 2^m-1 = 31$
  - farbige Peers gehören zum Overlay



P2P: Strukturiert (Beispiel Chord)

- Lookup von k = 26 auf Peer 1
  - Lookup auf Peer 1 ergibt p = 18 = $FT_1[5] \le 26$
  - Lookup auf Peer 18 ergibt p = 20 = $FT_{18}[2] \le 26 < FT_{18}[3]$
  - Lookup auf Peer 20 ergibt p = 21 = $FT_{20}[1] \le 26 < FT_{20}[2]$
  - Lookup auf Peer 21 ergibt p = 28 = $FT_{21}[1] \ge 26$
  - Ergebnis: succ(26) = 28
- Lookup von k = 12 auf Peer 28
  - Lookup auf Peer 28 ergibt p = 4 = $FT_{28}[4] \le 12 < FT_{28}[5]$
  - Lookup auf Peer 4 ergibt p = 9 = $FT_4[3] \le 12 < FT_4[4]$
  - Lookup auf Peer 9 ergibt p = 11 = $FT_9[2] \le 12 < FT_9[3]$
  - Lookup auf Peer 11 ergibt p = 14 = $FT_{11}[1] \ge 12$
  - Ergebnis: succ(12) = 14

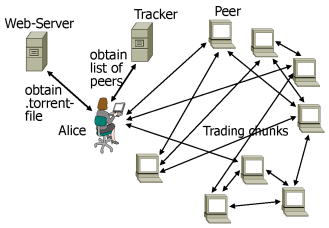
Quelle: Tanenbaum. Computer Networks. 5th Ed., Prentice Hall, 2011.

P2P: Bittorrent

- Bittorrent
  - Torrent: Schwarm von Peers für gleiche Datei, z.B. Tausende
  - Chunks: Teile der zu verteilenden Datei, z.B. 256 KB
  - Tracker: Server, bei dem sich Peers registrieren
  - .torrent-Datei mit Meta-Daten über zu verteilende Datei und Tracker
  - neuer Peer A tritt Schwarm bei
    - A registriert sich bei Tracker und erhält IP-Adressen zufälliger anderer Peers (z.B. 50) des Schwarms
    - A baut TCP-Verbindung zu einigen dieser Peers auf, fragt Liste der Chunks in ihrem Besitz nach und sendet Anfragen für Chunks
    - Rarest First: A fragt die seltensten Chunks der Peers zuerst nach, dadurch gleichmäßige Verteilung
    - Incentive Mechanismus (Tit-for-Tat): A misst Antwortrate der Peers und antwortet an diese in entsprechendem Anteil der Upload-Rate
    - neue Nachbarn werden zufällig dazu genommen
    - und mehr: Pipelining, Random First Piece, Endgame Model, Anti-Snubbing, …

P2P: Bittorrent

- Bittorrent (Fortsetzung)
  - Hybridarchitektur: Tracker ist zentraler Infrastrukturknoten, Rest P2P



Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

  - trackerloser Betrieb: Trackerinformation wird über DHT verteilt, Torrent durch eine ID beschrieben

P2P: Bitcoin

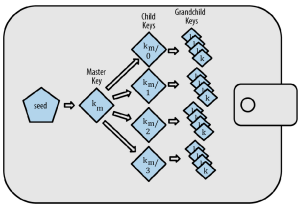
- Bitcoin (BTC)
  - „Kryptowährung": P2P-Zahlungssystem für virtuelle Geldeinheiten ohne vertrauenswürdige zentrale Instanz (wie z.B. einer Bank)
  - vorgeschlagen durch anonymen Autor Satoshi Nakamoto: "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. Open-Source Implementierung Bitcoin Core, 2010
  - Transaktionen von Geldbeträgen an öffentliche Bitcoin-Adressen, digital signiert mit privaten Schlüsseln, Versendung in unstrukturiertem P2P-Netz von Bitcoin-Clients (d.h. Fluten)
  - Miner (Schürfer) sammeln Transaktionen in Blöcken und führen Proof-of-Work durch: Wettbewerb zur Erstellung eines Hashs des Blocks, bei Erfolg Fluten, Verkettung zu einer öffentlichen Blockchain
  - Incentive (Belohnung) beim erfolgreichen Schürfen eines Blocks
  - Blockchain ist verteilte Datenbank, auch verteiltes Hauptbuch (Distributed Ledger) genannt, mit allen Transaktionen seit Beginn
  - erlaubt die Vermeidung von Double-Spending

P2P: Bitcoin

- Adressen
  - abgleitet aus privaten und öffentlichen Schüsseln basierend auf elliptischer Kurven-Kryptographie
  - Schüsselpaar
    - Private Key (Zufallsfahl, 256 Bits), z.B. (hexadezimal): 1E99423A4ED27608A15A2616A2B0E9E52CED330AC530EDCC32C8FFC6A526AEDD
    - ermöglicht die Berechnung des Public Key (Paar von 256 Bits), z.B. 1. Wert: F028892BAD7ED57D2FB57BF33081D5CFCF6F9ED3D3D7F159C2E2FFF579DC341A
  - Abbildung Public Key auf Bitcoin-Adresse
    - zweifache Hashfunktion (SHA-256, RIPMD160) anwenden
    - mit Base58 kodieren (Base64 außer 0OIl\+/, also Alphabet 123456789ABCDEFGHJKLMNPQRSTUVWXYZabcdefghijkmnopqrstuvwxyz)
    - z.B. Public Key oben: 1J7mdg5rbQyUHENYdx39WVWK7fsLpEoXZy
  - Pseudonymität durch Verwendung von Bitcoin-Adressen statt Identitäten, können häufig gewechselt werden
  - eingeschränkte Anonymität, da Verknüpfung der öffentlichen Transaktionsdaten mit identifizierenden Informationen ggfs. möglich ist

P2P: Bitcoin

- Adressen (Fortsetzung)
  - Wallet (Brieftasche): Bitcoin-Client zur Verwaltung von Schlüsseln
  - z.B. Wallet mit hierarchischem Schlüsselbaum, Ableitung aus einem Seed:



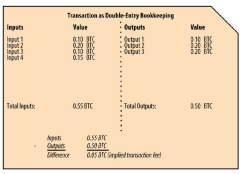
Quelle: A. M. Antonopoulos, Mastering Bitcoin, O'Reilly, 2014.

P2P: Bitcoin

- Transaktionen
  - Transaktion = nicht ausgegebene Gutschriften (Unspent Transaction Outputs, UTXO) aus früheren Transaktionen werden zu neuen Bitcoin-Adressen transferiert, in Satoshi = 10^{-8} BTC
    - Input: UTXO + Unlocking-Skript (z.B. Signierung durch zu Bitcoin-Adresse passenden privaten Schlüssel), erlaubt das Spending (Ausgeben)
    - Output: Transfer zu neuer Bitcoin-Adresse + Locking-Skript (z.B. Bitcoin-Adresse)
    - mehrere Inputs und Outputs möglich, dadurch Zusammenführung und Aufteilung
    - Transaction-Fee: Differenz aus Inputs und Outputs
    - eingeschränkte Skriptsprache erlaubt weitere Transaktionsarten
  - Fluten: An Peers versenden, diese verifizieren Transaktion, überprüfen auf Double-Spending in Blockchain und senden weiter
  - kein Konzept eines Kontos mit Geldbetrag, Wallet kann Summe von UTXOs bilden, Wallet stellt auch Transaktionen zusammen

P2P: Bitcoin

- Transaktionen (Fortsetzung)
  - Bsp. für Transaktion mit mehreren Inputs und Outputs:



  - Bsp. für Kette von Transaktionen:
    - in 2. Transaktion wird UTXO aus 1. Transaktion von Alice signiert (Unlocked, Spent)
    - dann wird Guthaben an Bitcoin-Adresse von Bob transferiert (Locked)

 Quelle: A. M. Antonopoulos, Mastering Bitcoin, O'Reilly, 2014.

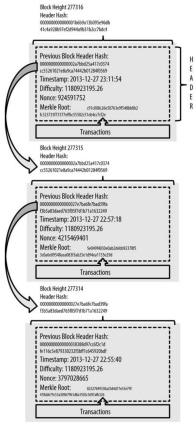
P2P: Bitcoin

- Blockchain
  - Miner erstellen Blöcke mit Transaktionen (1 MB)
  - Block-Header (80 Bytes) bestehen aus: Hash des Headers des letzten Blocks, Wurzel des Merkle-Baums mit Hash über alle Transaktionen im Block, Zeitstempel, Difficulty (Schwierigkeit), Nonce (Zufallszahl)
  - Proof-of-Work: Berechnung von Hash (SHA-256) kleiner als Difficulty (führende Zahl von Nullen), Lösung durch Ausprobieren
  - Wettbewerb der Miner
  - wenn Block gefunden wird: Fluten, Peers verifizieren den Block und berechnen bei Akzeptanz nächsten Block, dadurch wächst die Blockchain, dies geschieht im Mittel alle 10 Minuten
  - ergibt ca. 7 Transaktionen pro Sekunde
  - mögliche Verzweigungen werden weitergeführt, bis sich eine längste Teilkette bildet, diese wird weiterverwendet, dadurch entscheidet Mehrheit der Rechenleistung über Fortsetzung (Konsensbildung)

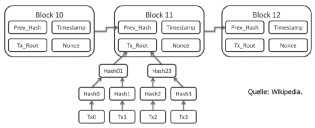
P2P: Bitcoin

- Blockchain (Fortsetzung)

Verkettung von Blockheadern, mit Zahlenbeispielen:



Verkettung von Blockheadern, konzeptionell:



Quelle: Wikipedia.

Block mit Transaktionen, die paarweise in Merkle-Baum gehasht werden, Wurzel wird in Header geschrieben

 Quelle: A. M. Antonopoulos, Mastering Bitcoin, O'Reilly, 2014.

P2P: Bitcoin

- Blockchain (Fortsetzung)
  - Rückverfolgung bis zum 1. Block (Genesisblock) vom 3.1.2009 möglich
  - Immutability (Unveränderlichkeit): nachträgliche Änderung früherer Blöcke fast unmöglich, da in kurzer Zeit gültige nachfolgende Blöcke berechnet werden müssten
  - Irreversibilität von in der Blockchain bestätigten Transaktionen
  - nach 5 Blöcken gilt Transaktion als sicher
  - Difficulty wird abhängig von Geschwindigkeit beim Mining angepasst
  - Incentive durch 1. Transaktion im Block und Transaction Fees (mind. 1000 Satoshis)
  - Geldschöpfung durch 1. Transaktion im Block, initial 50 BTC pro Block, Halbierung alle 210.000 Blöcke, ggw. 12,5 BTC, begrenzt Bitcoin-Menge auf 21 Mio, wird voraussichtlich 2140 erreicht (i.w. 2030)
  - Stand 16.4.2018 (blockchain.info): Höhe der Blockchain: 518470, Größe der Blockchain: ca. 165 GB, Bitcoins im Umlauf: ca. 17 Mio

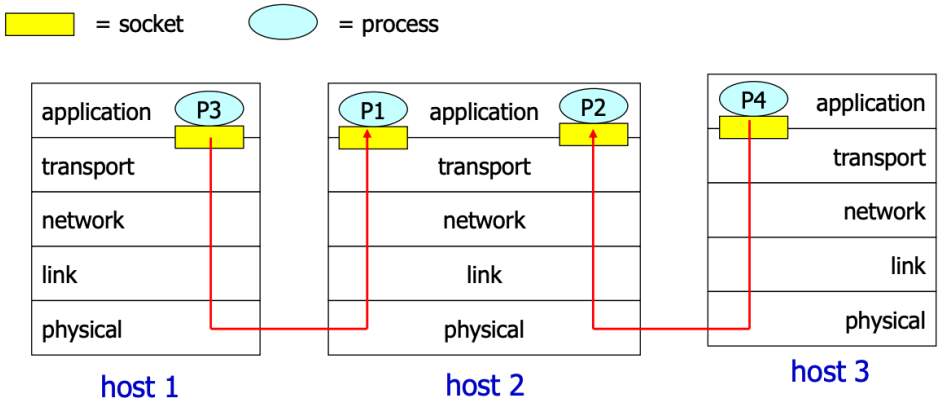
P2P: Bitcoin

- Kontroversen u.a.
  - Angriffe: 51%, Quantencomputer gegen SHA-256 und elliptische Kurven-Kryptografie, …
  - Mining-Pools mit spezialisierter HW nur für Mining
  - Strombedarf in Größenordnung eines kleinen Staats
  - Skalierbarkeit (Visa mehrere Zehntausend Transaktionen pro Sekunde)
  - Verwendung für illegale Zwecke
- Weiterentwicklung der Blockchain-Technologie u.a.
  - Alt Coins: weitere Kryptowährungen basierend auf Forks der Bitcoin-Blockchain und alternativen Blockchains
  - weitere Anwendungen außerhalb des Finanzbereichs
  - Turing-mächtige Skriptsprache, Smart Contracts, z.B. Ethereum
  - Proof-of-Stake: Konsensbildung basierend auf Vermögen
  - Proof-of-Authority: Validierung durch vertrauenswürdige Instanzen
  - Permissioned Blockchain: Zugangsbeschränkung, z.B. Hyperledger

KAPITEL 3

TRANSPORTSCHICHT

Die Aufgabe der Transportschicht ist das Bereitstellen eines Dienstes zur Kommunikation zwischen Anwendungsprozessen:

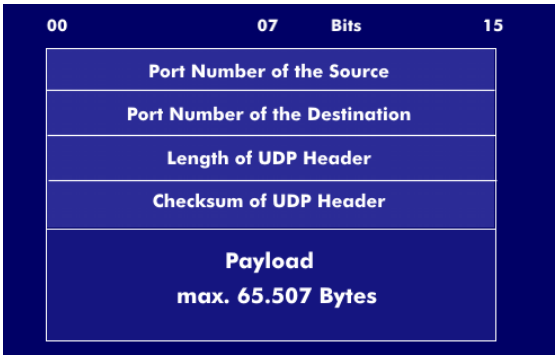


Dabei gibt es verschiedene Dienstmerkmale, unter anderem die Fehlerkontrolle, die Bewahrung der Reihenfolge, eine verbindungslose/verbindungsorientierte Übertragung, Fluss- und Überlastkontrollen, sowie Garantien für die Dienstgüte. Dabei sind zwei Protokolle wesentlich, das *user datagram protocol* (UDP) und das *transmission control protocol* (TCP). Bei ersterem findet die Übertragung verbindungslos und ohne Kontrollmechanismen statt. Zudem wird auch nicht die Reihenfolge gewahrt. Es ist eine Schnittstelle für die einfache Paketvermittlung mittels IP, denn die Verantwortung für Kontrollmechanismen liegen bei der Anwendung selbst. Bei letzterem findet die Übertragung verbindungsorientiert statt mit Fehler-, Fluss- und Überlastkontrolle. Allerdings gibt es keine Dienstgütegarantien. Diese Variante bietet die Abstraktion eines Bytestroms.

3.1 UDP

Ein UDP-Segment besteht aus …

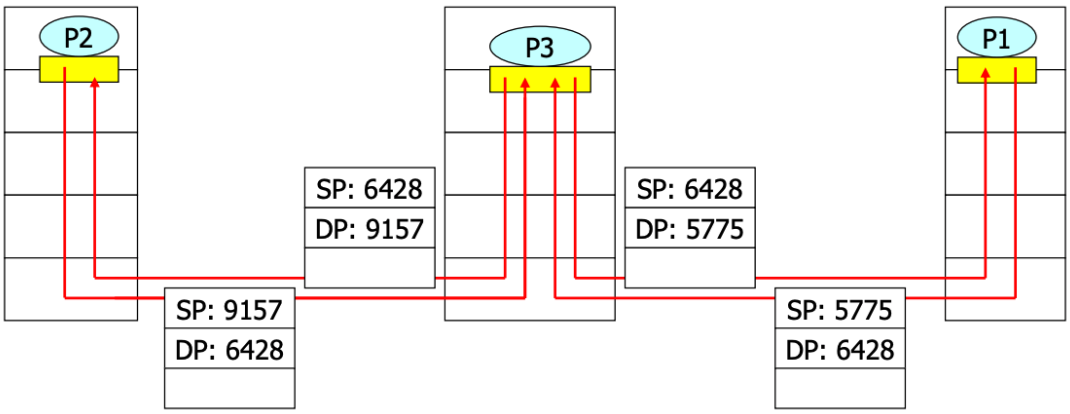
- … der 16 bit langen Quellportnummer
- … der 16 bit langen Zielportnummer
- … der 16 bit langen Segmentlänge
- … der 16 bit langen Prüfsumme
  für eine mögliche Fehlerkontrolle, wobei die Nutzung optional ist und eine Nichtnutzung durch `0x0000` *dargestellt wird*
- … der maximal 65.507 Byte großen Nachricht



Multiplexen und Demultiplexen Das **Multiplexen** bezeichnet das Zusammenführen der Segmente verschiedener Anwendungsprozesse durch die Transportschicht auf dem Quellhost, wohingegen das **Demultiplexen** das Ausliefern der Segmente an verschiedene Anwendungsprozesse durch die Transportschicht des Zielhosts bezeichnet. Der Anwendungsprozess vereinbart mit der Transportschicht auf dem Quellhost die Quellportnummer, welche entweder durch die Anwendung gewählt (Variante ①) oder ein freier Port vom Betriebssystem geliefert wird (Variante ②).

① `new DatagramSocket(6428);` ② `new DatagramSocket();`

UDP auf dem Zielhost erkennt dann an der Zielportnummer – **und nur daran** — zu welcher Anwendung das Segment geliefert werden soll. Ein Anwendungsprozess kann dabei aber mehrere Sockets besitzen. Ein Beispiel:



Berechnung der Prüfsumme und Pseudoköpfe Anders als bei IP fließen bei UDP und TCP das gesamte Paket, also Kopf und Daten, sowie ein Pseudo-Kopf (Pseudo-Header) in die Berechnung der Prüfsumme ein. Bei UDP ist die Prüfsumme optional, bei TCP dagegen obligatorisch. Der Algorithmus zur Berechnung der UDP-Prüfsumme ähnelt stark dem bei IP verwendeten Verfahren:

> *Die UDP-Prüfsumme ist das 16-Bit-Einerkomplement der durch Addition benachbarter 16-Bit-Wörter berechneten Einerkomplementsumme von Pseudo-Kopf, UDP-Kopf und Daten. Bei einer ungeraden Länge des UDP-Datagramms wird ein Null-Byte am Ende angehängt, um immer 16 Bit addieren zu können.*

Verfahren 3.1 (*Berechnung der UDP-Prüfsumme*)

Schritt 1 Setze das Prüfsummenfeld im UDP-Header auf `0x0000 0000 0000 0000`.

Schritt 2 Erzeuge eine vorzeichenlose 32-Bit-Zahl für die Prüfsumme, initialisiere sie mit Nullen.

Schritt 3 Fasse direkt benachbarte Bytes des UDP-Paketes zu 16-Bit-Blöcken zusammen. Falls der letzte Block weniger als 16 Bit hat, dann fülle ihn von hinten mit Nullen auf, bis er 16 Bit hat.

Schritt 4 Speichere das Ergebnis der Addition aller 16-Bit-Blöcke *mit Übertrag* in der Prüfsumme.

Schritt 5 Fasse direkt benachbarte Bytes des Pseudo-Headers zu 16-Bit-Blöcken zusammen.

Schritt 6 Speichere das Ergebnis der Addition dieser 16-Bit-Blöcke und der bisherigen Prüfsumme *mit Übertrag* in der Prüfsumme.

Schritt 7 Fasse direkt benachbarte Bytes der Prüfsumme zu zwei 16-Bit-Blöcken zusammen, addiere diese und speichere das Ergebnis mit Übertrag in der Prüfsumme, bis kein Übertrag mehr bei der Addition entsteht.

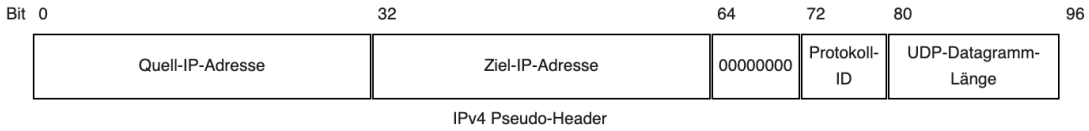
Schritt 8 Die signifikantesten 16 Bit der 32-Bit-Prüfsumme sind nun Nullen. Die weniger signifikanten Bits sind die eigentliche Prüfsumme; speichere diese als vorzeichenlose 16-Bit-Zahl.

Schritt 9 Wenn diese 16-Bit-Zahl nicht nur aus Einsen besteht, dann speichere ihr *Einerkomplement* im UDP-Header (sowohl `0x1111 1111 1111 1111` und das Einerkomplement hiervon, `0x0000 0000 0000 0000`, symbolisieren die Zahl 0). In IPv4-UDP wird `0x0000 0000 0000 0000` auch verwendet, um zu signalisieren, dass keine Prüfsumme berechnet wurde. IPv6-UDP-Pakete mit der Prüfsumme `0x0000 0000 0000 0000` sind **ungültig**.

Der Empfänger prüft zunächst, ob das Prüfsummenfeld des empfangenen Paketes nur aus Nullen besteht. Wenn ja, kann er das Paket als richtig empfangen werten, da keine Prüfsumme vorhanden ist. Wenn nicht, so wendet er den oben beschriebenen Algorithmus auf das empfangene Paket und den zugehörigen Pseudo-Header an, lässt den letzten Schritt weg und addiert die selbst berechnete Prüfsumme auf die im Prüfsummenfeld empfangene, was durch die Einerkomplementdarstellung einer Subtraktion entspricht. Erhält der Empfänger 0 als Ergebnis der Addition (bzw. Subtraktion), so wertet er die empfangenen Daten als mit den gesendeten übereinstimmend.

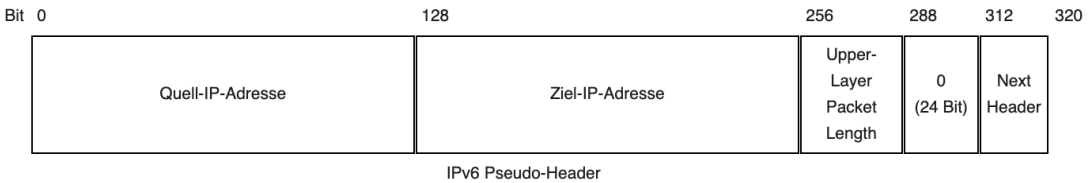
Der bereits erwähnte Pseudo-Kopf umfasst einige Felder des IP-Kopfes. Dadurch soll seitens UDP sichergestellt werden, dass ein Paket auch beim richtigen Empfänger angekommen ist. Fälschlicherweise kann es passieren, dass die IP-Schicht ein Datagramm akzeptiert, obwohl es nicht an den betreffenden Host adressiert ist, oder dessen Nutzlast einem falschen Transportprotokoll (z.B. UDP statt TCP) zuleitet.

Der Pseudokopf für IPv4 hat eine Größe von 96 Bit und setzt sich aus der 32 Bit langen Quell-IP-, sowie Ziel-IP-Adresse zusammen mit anschließenden 8 Bit Leerfeld, 8 Bit Protokoll-ID, wobei UDP hier die ID 17 hat, sowie der Länge des UDP-Datagramms mit 16 Bit:



IPv4 Pseudo-Header

Die Pseudoköpfe (*extension header*) von IPv6 sind in RFC 2460 näher definiert. Er besitzt eine Größe von 320 Bit (40 Oktetts). Er setzt sich aus der Quell- und Ziel-IP-Adresse zusammen, welche jeweils eine Länge von 128 Bit haben, sowie anschließend die *upper-layer packet length* mit 32 Bit, was in diesem Fall die Länge des UDP-Datagrams ist, 24 Bit Leerfeld und zum Schluss 8 Bit für die *Next-Header-ID*, was der Protokoll-ID entspricht:



IPv6 Pseudo-Header

Bitfehlerwahrscheinlichkeiten

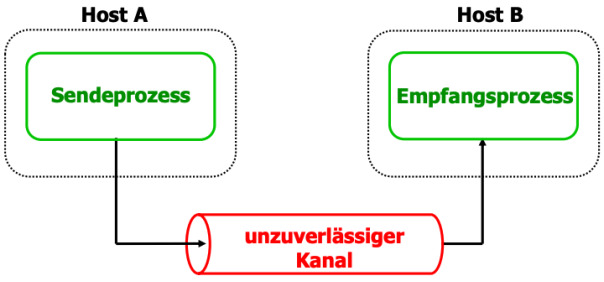
UDP

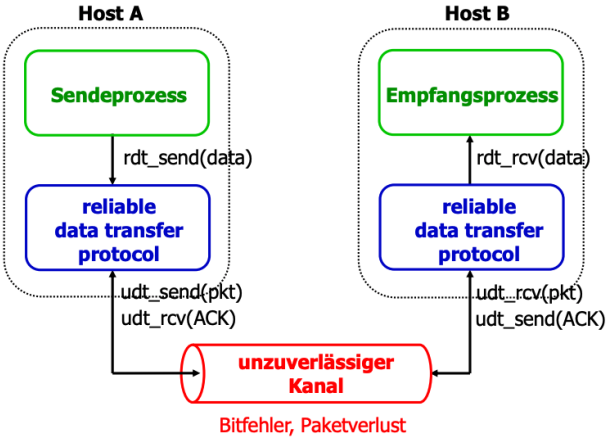
- Bitfehlerwahrscheinlichkeiten
  - sei die Wahrscheinlichkeit eines einzelnen Bitfehlers p = 10^{-7}
  - sei die Segmentlänge L = 10^4 Bits
  - übliche vereinfachende Annahme (um überhaupt rechnen zu können): die Bitfehler der einzelnen Bits sind unabhängig voneinander
  - Wahrscheinlichkeit für mindestens einen Bitfehler im Segment: $1-(1-p)^L \approx 0{,}00099950 \approx 10^{-3}$
  - Wahrscheinlichkeit für zwei Bitfehler im Segment:
    - Anzahl Paare: $\sum_{i=1}^{L-1} i = \frac{(L-1)\cdot L}{2} = \frac{(10^4-1)\cdot 10^4}{2} = \binom{10^4}{2} \approx \frac{10^8}{2}$
    - Wahrscheinlichkeit, dass ein bestimmtes Paar fehlerhaft ist: 10^{-14}
    - Wahrscheinlichkeit, dass ein beliebiges Paar fehlerhaft ist: $10^8 \cdot 10^{-14}/2 = 10^{-6}/2$
  - wie lange dauert es im Mittel, bis ein Segment mit zwei Bitfehlern auftritt bei a) 10 Mbps und b) 10 Gbps?

Rechnerkommunikation, Transportschicht 12

3.2 Fehlerkontrolle

Wie fast überall gibt es durch Rauschen, Pufferüberläufe oder Ausfälle von Komponenten verursachte Bitfehler und Paketverluste bei der Übertragung von Daten. Diese werden durch UDP erstmal nicht erkannt, können jedoch durch ein Protokoll mit Fehlererkennung, Bestätigungen und Sendewiederholungen ausgeglichen werden.





Dabei gibt es drei grundlegende Protokolle für den zuverlässigen Transport. Einerseits das *stop-and-wait* Protokoll, bei welchem der Sender zur Fehlererkennung eine Prüfsumme — besser noch: *cyclic redundancy check (CRC)* — hinzu und der Empfänger eine Bestätigung (ACK, *acknowledgment*) sendet. Kommt diese nach einem Timeout nicht an, so wird das Paket einfach erneut gesendet, gegen mögliche Duplikate geht man mit Sequenznummern (SQN) vor. **Nachteil:** Bei einem großen Bitraten-Verzögerungsprodukt ist der Sender die meiste Zeit blockiert und somit das Protokoll sehr ineffizient. Andererseits gibt es noch Schiebefensterprotokolle (*sliding-window-protocols*), bei welchen mehrere Pakete auf einmal gesendet werden, um den Kanal

zu füllen. Hier gibt es zwei Varianten, *Go-Back-N* und *selective repeat*, welche sich beim Timeout, Bestätigungen und Sendewiederholungen unterscheiden.

3.2.1 Stop-and-Wait

Informelle Beschreibung
Verhalten des Senders:

(1) Sende Paket mit aktueller Sequenznummer und starte den Timeouttimer.

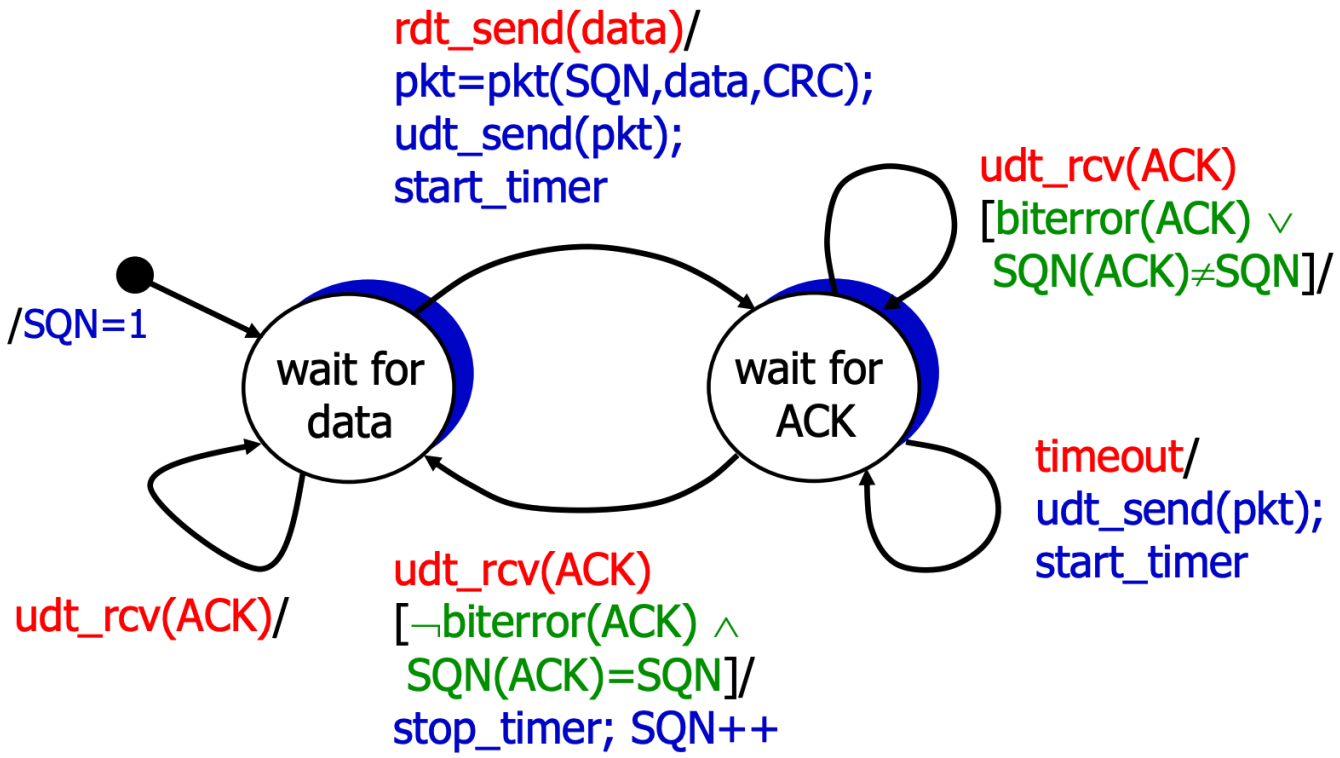
(2) Wenn ein ACK *ohne Bitfehler* und mit aktueller Sequenznummer vor Ablauf des Timeouts zurückkommt, inkrementiere Sequenznummer und gehe zu Schritt (1).

(3) Wenn der Timeout abläuft, sende das Paket erneut, state den Timeouttimer erneut und gehe zu Schritt (2).

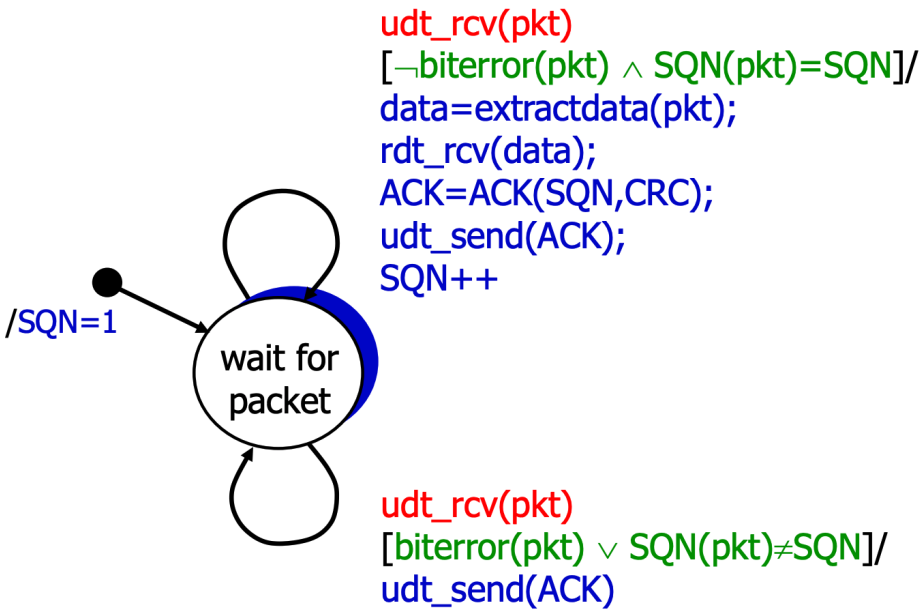
Verhalten des Empfängers: Wenn ein Paket *ohne Bitfehler* und mit aktueller Sequenznummer ankommt, sende ACK mit aktueller Sequenznummer und inkrementiere die Sequenznummer. Andernfalls sende das letzte ACK erneut.

Sequenznummerraum Die Repräsentation der Sequenznummern ist endlich, da `SQN` ein Feld mit n Bits ist und somit höchstens 2^n verschiedene Sequenznummern ermöglicht. Eine Wiederverwendung ist durch *zyklisches Durchlaufen* möglich.
Im Allgemeinen ist für *stop-and-wait* ein Bit zur Darstellung von 2 Sequenznummern ausreichend, man nennt diese Art von Protokoll dann **Alternating-Bit-Protokoll**.

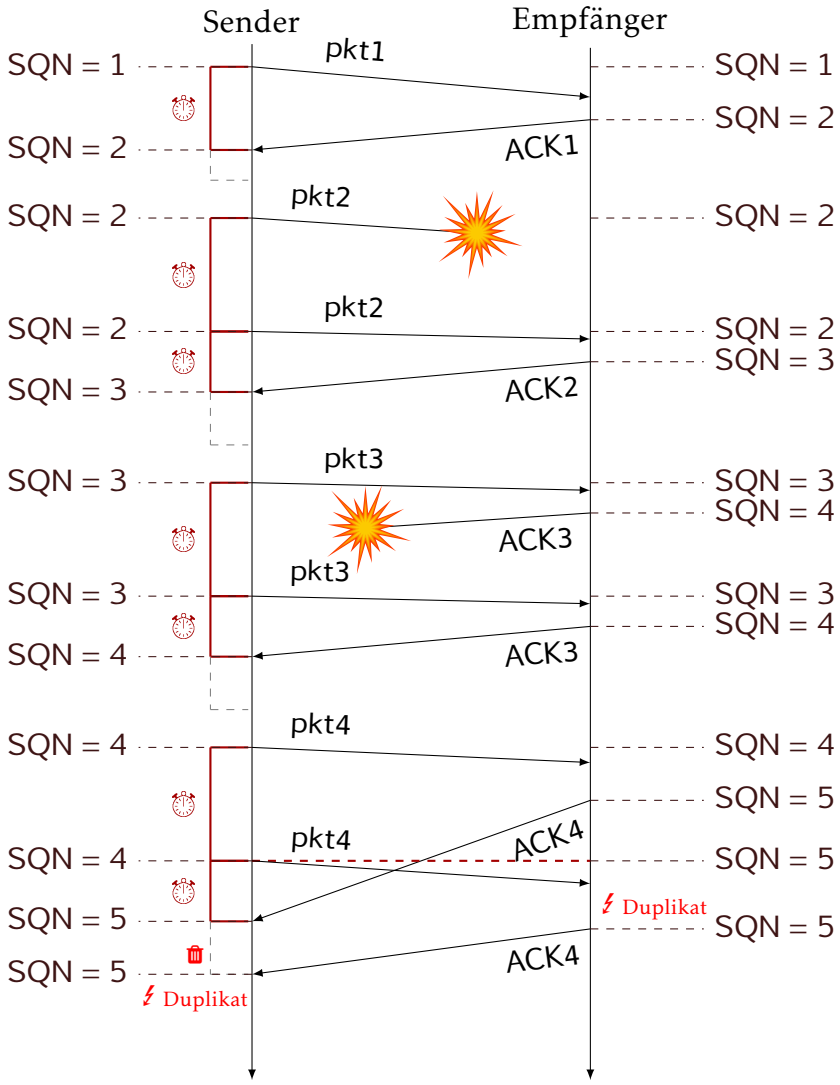
Statechart — Sender



Statechart — Empfänger



Ablauf Ein beispielhafter Ablauf mit den allermeisten Möglichkeiten:

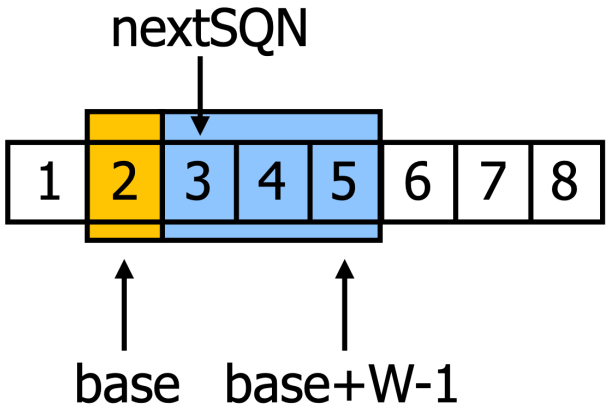


3.2.2 Go-Back-N

Bei Go-Back-N darf der Sender mehrere Pakete — bis zu einer Maximalzahl — vor Erhalt eines ACKs senden. Er startet beim Senden **des jeweils ersten Pakets** einen Timer und puffert die unbestätigten Pakete. Sollte der Timer ablaufen, so werden **alle unbestätigten Pakete erneut gesendet**. Der Empfänger auf der anderen Seite sendet **kumulative ACKs**, also ein ACK mit einer SQN, welche bestätigt, dass alle Pakete *bis zu dieser SQN* erfolgreich empfangen wurden. Der Empfänger akzeptiert dabei nur Pakete in der richtigen Reihenfolge. **Vorteil:** Der Empfänger benötigt somit auch **keinen Puffer**.

Sendepuffer Bei Sendepuffern gibt es drei wichtige Größen:

- **`base`** beschreibt die SQN des **ältesten, aber noch unbestätigten** Pakets
- **`nextSQN`** beschreibt die SQN des nächsten zu verschickenden Pakets
- **`W`** beschreibt die Fenstergröße, die Anzahl der Pakete, welche der Sender vor Erhalt eines ACKs senden darf.



Das Fenster (`[base, base + (W - 1)]`) wird beim Ablauf des Protokolls von links nach rechts verschoben, wegen der kumulativen ACKs hat es **immer** folgende Struktur: Das Intervall `[base, nextSQN - 1]` enthält die bisher noch unbestätigten, aber **bereits gesendeten** Pakete, wohingegen das Intervall `[nextSQN, base + (W - 1)]` noch bisher ungesendete Pakete enthält, welche aber vor Erhalt eines ACKs noch gesendet werden dürfen.

Informelle Beschreibung
Verhalten des Senders:

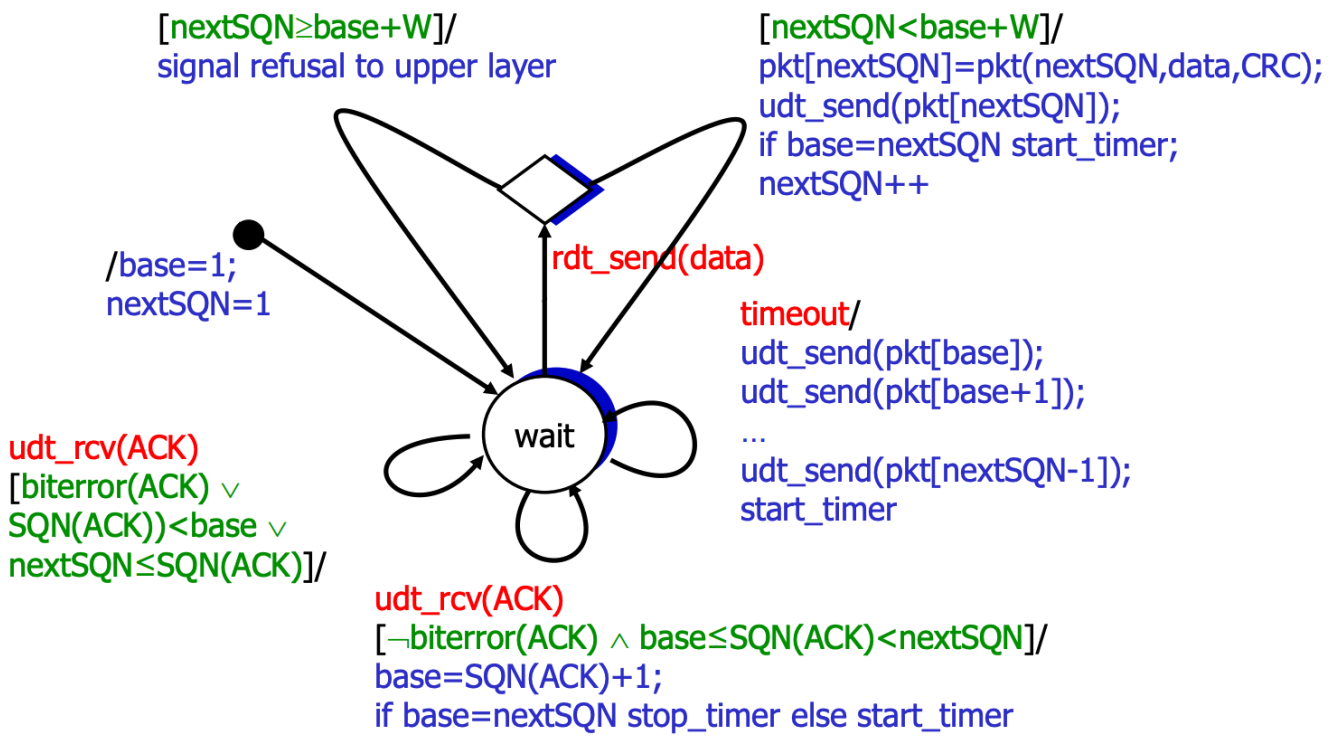
(1) Wenn Daten zum Senden **und ein Platz im Fenster** vorhanden sind, so sende ein Paket mit `nextSQN` und inkrementiere `nextSQN`. *Ist es das **erste** Paket im Fenster, so starte den Timer.*

(2) Wenn ein ACK *ohne Bitfehler* und mit Sequenznummer *im Fenster* vor Ablauf des Timeouts zurückkommt, schiebe das Fenster bis zu dieser Sequenznummer und wenn das Fenster leer ist, so stoppe den Timer, sonst starte den Timer neu.

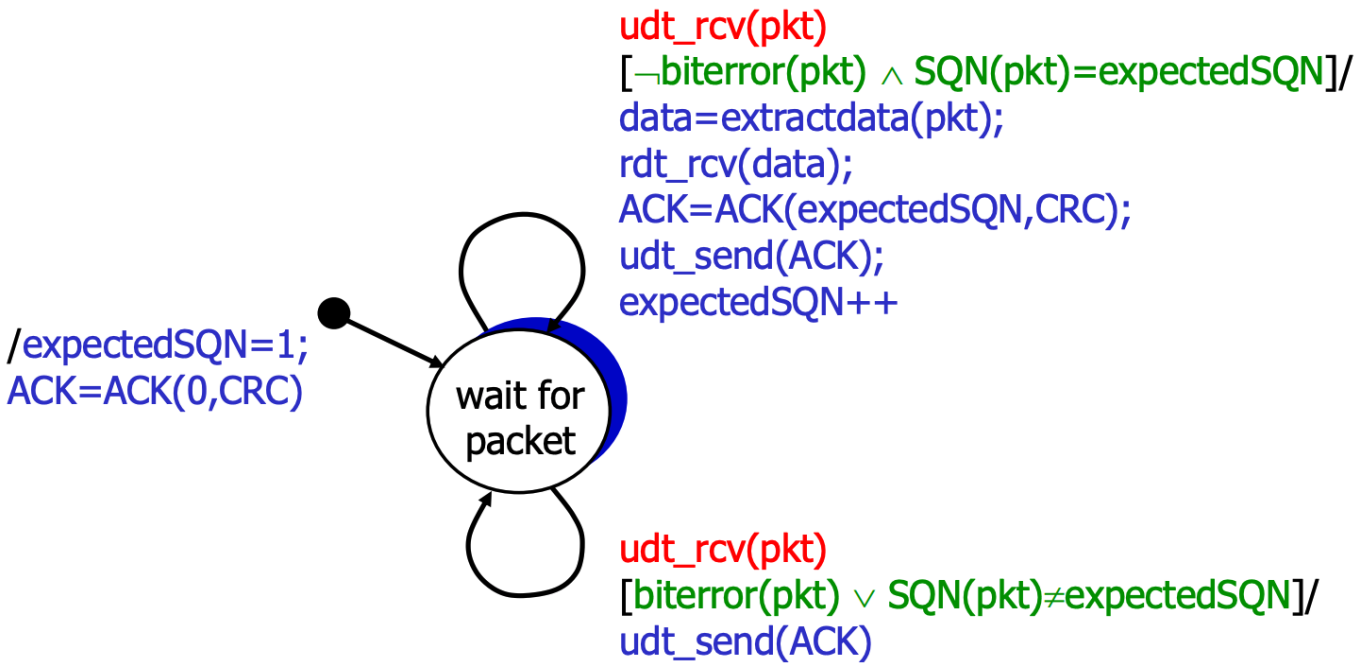
(3) Wenn der Timeout abläuft, sende alle unbestätigten Pakete des Fensters erneut und starte den Timer erneut.

Verhalten des Empfängers: Wenn ein Paket *ohne Bitfehler* und mit aktueller Sequenznummer ankommt, sende ACK mit aktueller SQN und inkrementiere SQN, sonst sende das letzte ACK erneut (identisch zu *stop-and-wait*).

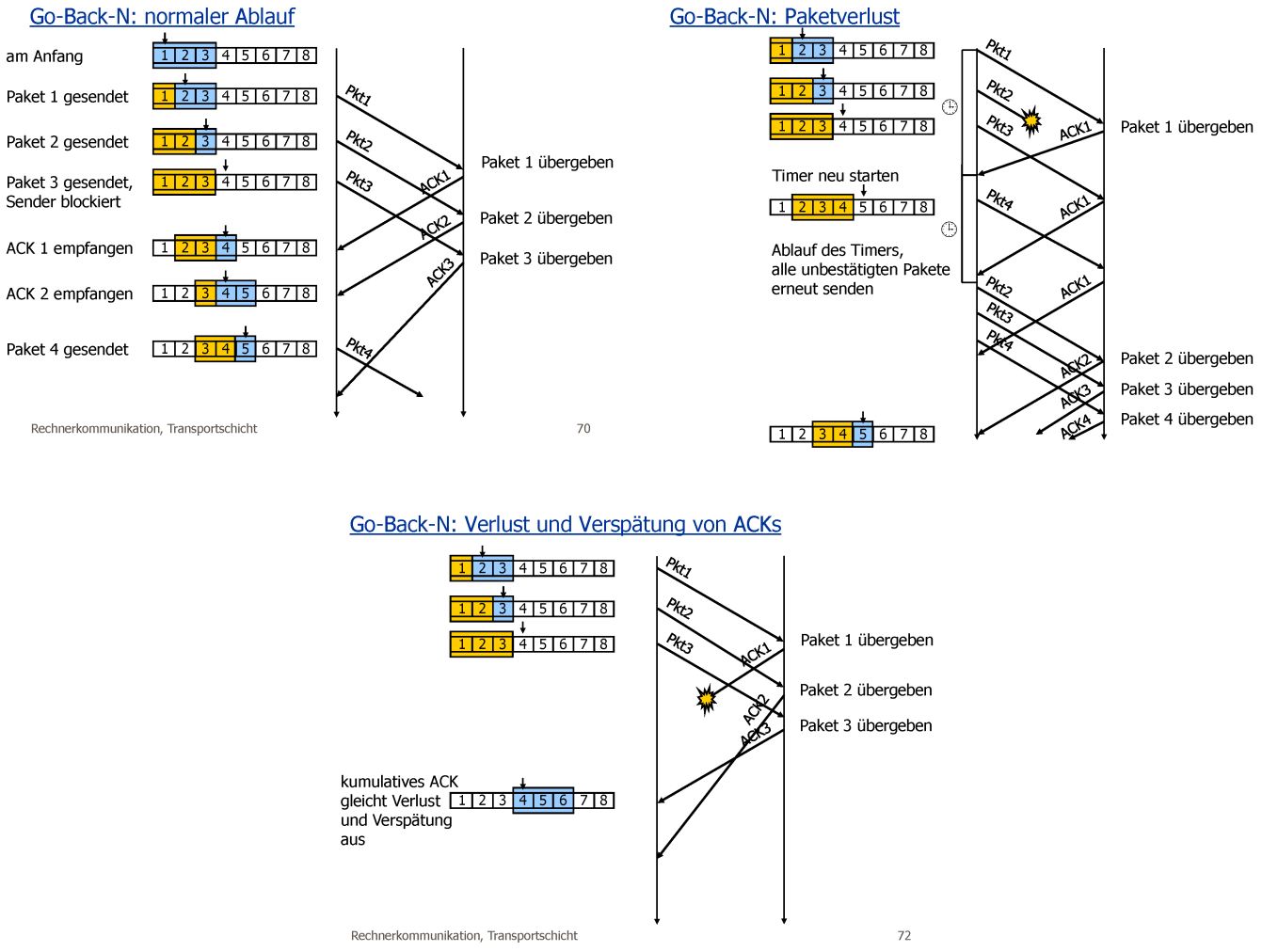
Statechart — Sender



Statechart — Empfänger



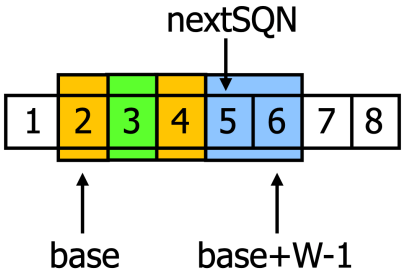
Ablauf Beispielhafter Abläufe mit den allermeisten Möglichkeiten:



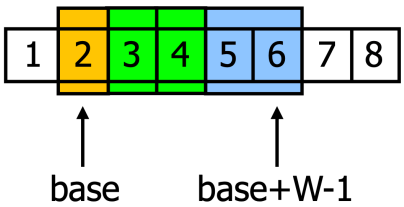
Vorteil: Die kumulativen ACKs gleichen Verlust und Verspätung von ACKs wieder aus!

3.2.3 Selective Repeat

Bei *Selective Repeat* darf der Sender wieder mehrere Pakete — bis zu einer Maximalzahl — vor Erhalt eines ACKs senden. Dabei startet er beim Senden **jedes Pakets** einen Timer und puffert die unbestätigten Pakete. Wenn der Timer für ein *spezifisches Paket* abläuft, wird **eben dieses Paket** erneut gesendet. Der Empfänger schickt hierbei **selektive ACKs**, also ein ACK mit einer SQN, welche bestätigt, dass das Paket mit der Sequenznummer erfolgreich empfangen wurde. **Nachteil:** Der Empfänger benötigt **einen Puffer** zum Ausgleich von Lücken beim Empfang.



Sendepuffer `base`, `nextSQN` und `W` sind identisch zu Go-Back-N gewählt. Das Fenster auf der Senderseite enthält versendete und **unbestätigte**, versendete und **bestätigte** und ungesendete Pakete.



Empfängerpuffer `base`, `nextSQN` und `W` sind identisch zu Go-Back-N gewählt. Das Fenster auf der Empfängerseite enthält Lücken, empfangene Pakete und Platz für unempfangene Pakete. Der Empfänger puffert die empfangenen Pakete.

Informelle Beschreibung
Verhalten des Senders:

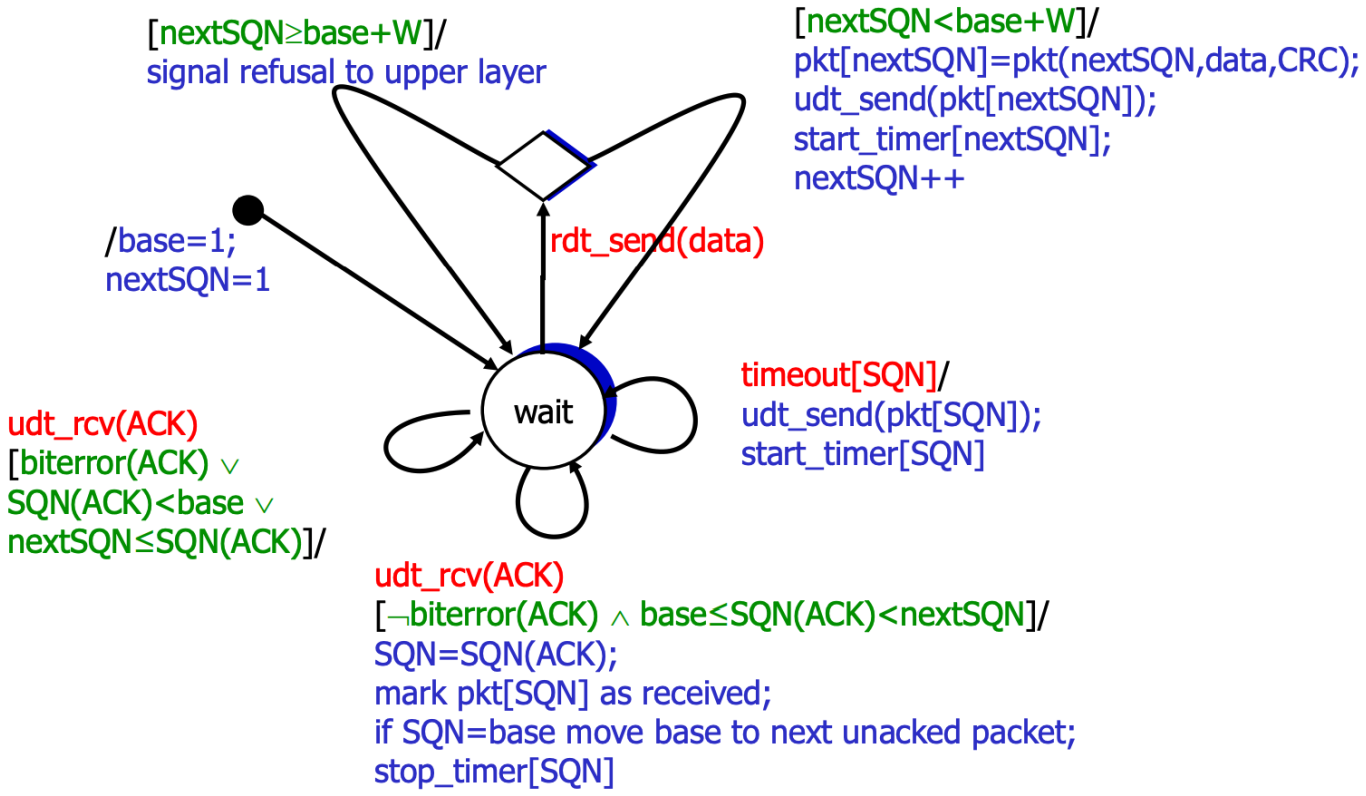
(1) Wenn Daten zum Senden und ein Platz im Fenster vorhanden sind, so sende das nächste Paket mit `nextSQN`, starte den Timer für eben dieses Paket und inkrementiere `nextSQN`.

(2) Wenn ein ACK *ohne Bitfehler* und mit Sequenznummer *im Fenster* vor Ablauf des Timeouts zurückkommt, markiere das Pakte mit der Sequenznummer als bestätigt und schiebe das Fenster bis zur nächsten Lücke.

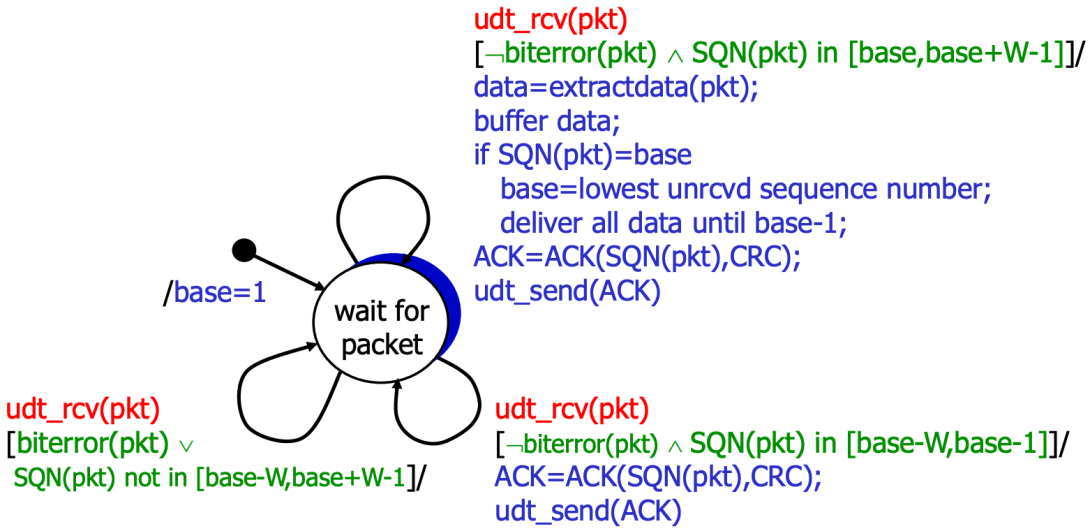
(3) Wenn der Timeout für ein Paket mit Sequenznummer `SQN` abläuft, sende dieses Paket erneut und starte den Timer für dieses Paket erneut.

Verhalten des Empfängers: Wenn ein Paket *ohne Bitfehler* und mit Sequenznummer *im Fenster* ankommt, so sende die ACK mit dieser Sequenznummer, puffere das Paket und schiebe das Fenster bis zur nächsten Lücke. *Wenn ein Paket mit einer Sequenznummer aus einem vorherigem Fenster ankommt, so sende das ACK hierfür erneut.*

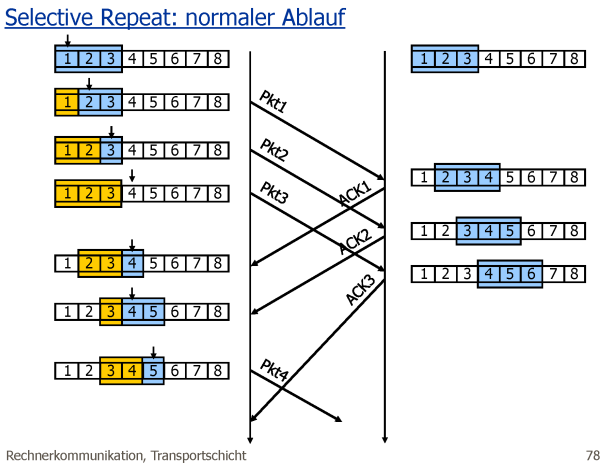
Statechart — Sender

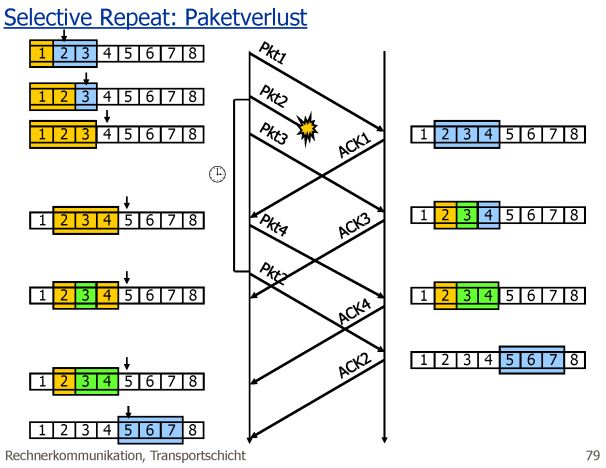


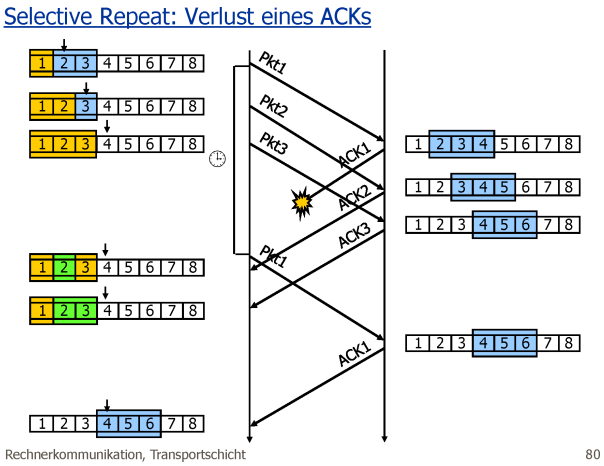
Statechart — Empfänger



Ablauf Beispielhafter Abläufe mit den allermeisten Möglichkeiten:



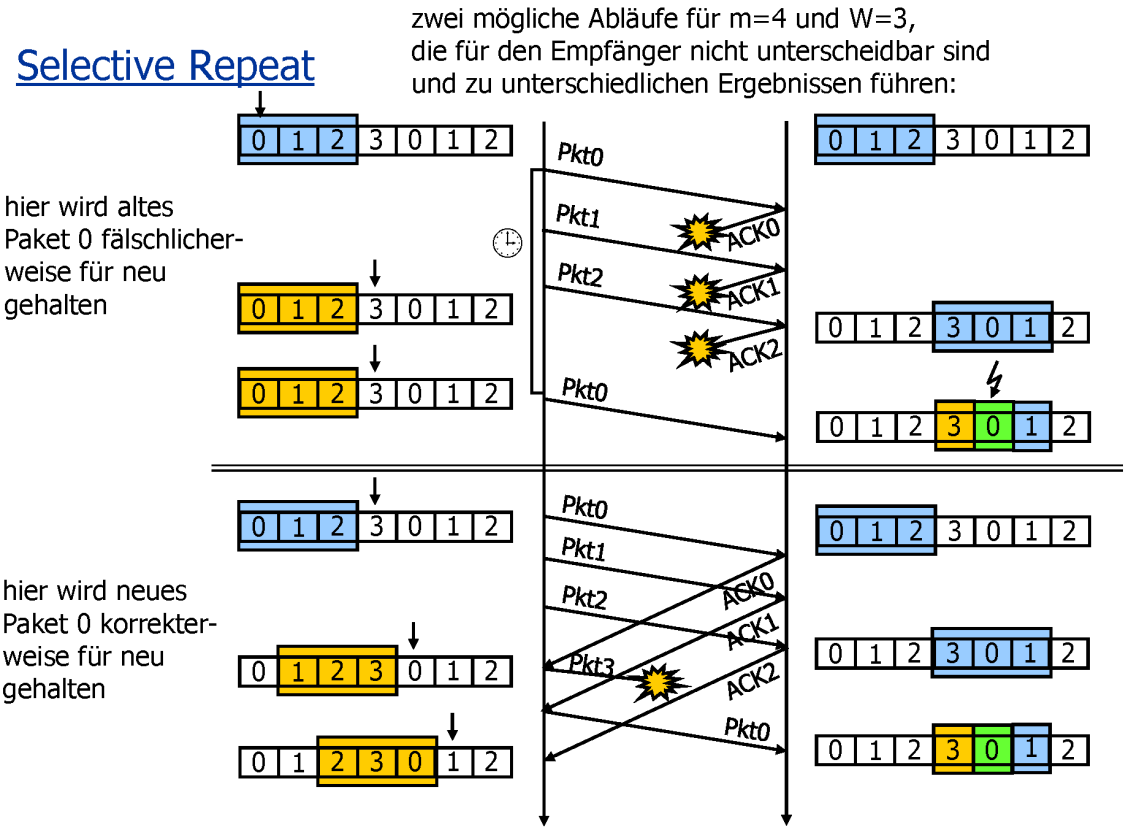




Sequenznummerraum bei Schiebefensterprotokollen Es gibt weiterhin ein endliches Sequenznummernfeld mit m Werten. Durch *zyklisches Durchlaufen* werden Sequenznummern wiederverwendet. **Aber: unterschiedliche Pakete mit gleicher Sequenznummer müssen weiterhin unterschieden werden.** Hinreichende Bedingungen hierfür sind:

- Falls Empfangsfenstergröße $= 1$ gilt, muss $W < m$ gelten.
- Falls Sendefenstergröße $=$ Empfangsfenstergröße $= W > 1$ gilt, so muss $W < (m+1)/2$ gelten.

Beispiel für zu kleinen Sequenznummerraum Im folgenden Beispiel mit $m = 4$ und $W = 3$ ist $W > (m+1)/2$. Damit kann der Empfänger nicht unterscheiden, ob Paket 0 alt oder neu ist:



Vergleich von Go-Back-N und Selective Repeat **Vorteile von Go-Back-N**:

+ kumulative ACKs gleichen ACK-Verluste und -Verspätungen schnell aus, ohne dass die Pakete erneut gesendet werden müssen.
+ Der Sender benötigt nur einen Timer und nicht für jedes Paket einen.
+ Der Empfänger benötigt keinen Puffer.
+ Sender und Empfänger können einfacher realisiert werden, weil keine Lücken in den Fenstern beachtet werden müssen.

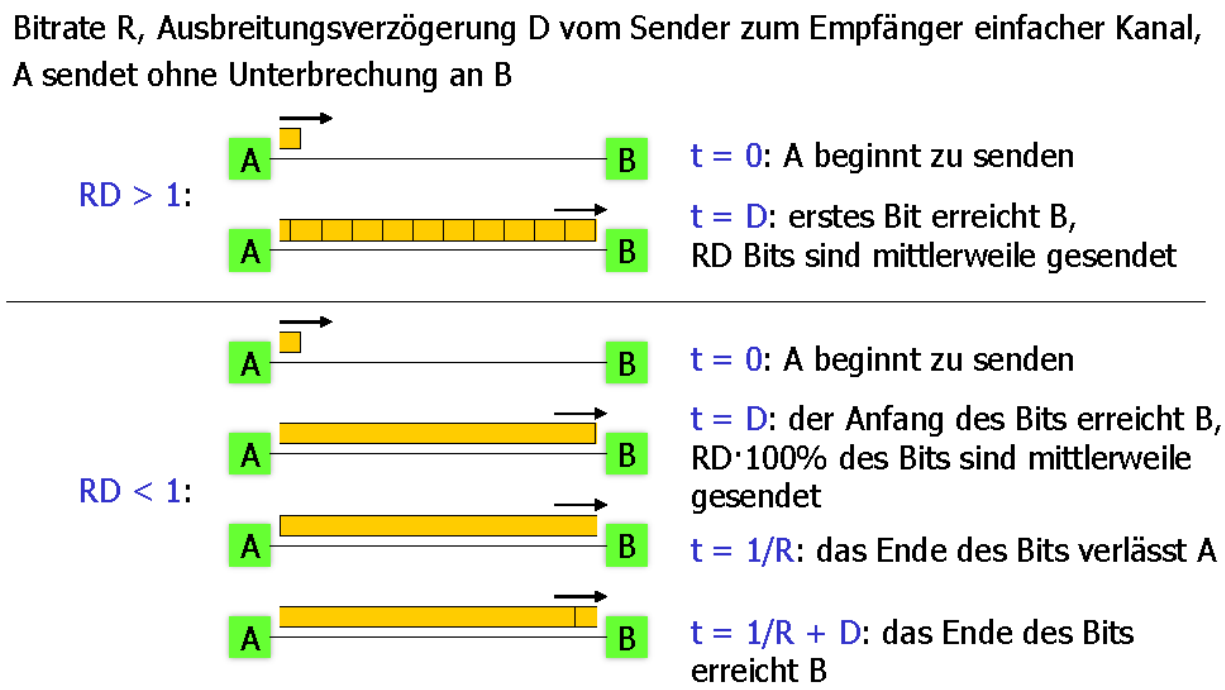
Vorteil von Selective Repeat: Es gibt weniger Wiederholungen von Sendungen, weil wirklich *nur fehlerhafte oder verlorengegangene Pakete* erneut gesendet werden.

3.2.4 Leistungsanalyse

Man möchte sich nun noch typische Fragen für Leistungsanalysen stellen, wie wann eine Senderblockade bei Stop-and-Wait eintritt und wie stark der mögliche Durchsatz verkleinert wird oder wie groß bei Schiebefensterprotokollen das Fenster sein muss, um den Kanal zu füllen. Ebenso will man die Effizienz der Schiebefensterprotokollverfahren vergleichen.

Wir betrachten hier Beispiele für typische Leistungsanalysen mit einigen vereinfachten Annahmen. Genauere Untersuchungen sind dann mit Simulationen möglich.

Kanalpuffergröße Wir betrachten das Produkt aus Bitrate und Verzögerung:



Im Allgemeinen gilt für die Kanalpuffergröße in Bits

$$a = R \cdot D = \frac{D}{1/R} = \frac{\ell/v}{1/R} = \frac{\text{Ausbreitungsverzögerung}}{\text{Bitsendezeit}}.$$

Es ist damit die Anzahl gesendeter Bits während sich das erste Bit vom Sender zum Empfänger ausbreitet.
Das Produkt kann einerseits größer 1 sein, wie im folgenden Beispiel mit $R = 100\text{Mbps}, \ell = 4800\,\text{km}, v = 3 \times 10^8\,\text{m}\,\text{s}^{-1}$:

$$a = RD = 1600 \times 10^3\ \text{Bit} \approx 195\ \text{KiB}$$

Andererseits allerdings auch kleiner 1, wie im Beispiel mit $R = 10\text{Mbps}, \ell = 10\,\text{m}, v = 2 \times 10^8\,\text{m}\,\text{s}^{-1}$:

$$a = RD = 0{,}5\ \text{Bit}$$

Ist die Paketgröße L gegeben, so berechnet sich die Kanalpuffergröße in Paketen durch

$$a = \frac{RD}{L} = \frac{\ell/v}{L/R} = \frac{\text{Ausbreitungsverzögerung}}{\text{Bitsendezeit}}.$$

Es ist damit die Anzahl gesendeter Pakete während sich das erste Bit vom Sender zum Empfänger ausbreitet.
Im Beispiel mit $L = 1500$ Byte und $v = 3 \times 10^8\,\text{m}\,\text{s}^{-1}$ unterscheidet sich a jenachdem, ob wir eine ...

... *Glasfaserverbindung* mit $R = 100\text{Mbit/s}$ und $\ell = 10\,\text{km}$ betrachten, hier ist dann

$$a = \frac{\ell/v}{L/R} \approx \frac{33\,\mu\text{s}}{L/R} \approx 0{,}28.$$

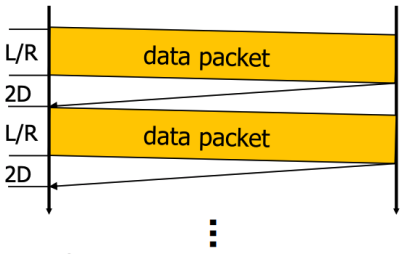
... *Satellitenkommunikation* mit $R = 36\,\text{Mbit/s}$ und $\ell = 72000\,\text{km}$ betrachten, hier ist dann

$$a = \frac{\ell/v}{L/R} \approx \frac{240000\,\mu s}{L/R} \approx 720.$$

Einige Werte für a sind in folgender Tabelle gegeben:

Bitrate	Paketgröße	Entfernung	a
500 Kbps	136 Bits	10 m	0,0001
1 Mbps	1500 Bytes	10 m	0,0000028
1 Mbps	1500 Bytes	1 Km	0,00028
1 Mbps	1500 Bytes	10 Km	0,0028
1 Mbps	1500 Bytes	100 Km	0,028
1 Mbps	1500 Bytes	1.000 Km	0,28
1 Mbps	1500 Bytes	10.000 Km	2,8
1 Mbps	1500 Bytes	36.000 Km	10
10 Mbps	1500 Bytes	10 Km	0,028
10 Mbps	1500 Bytes	100 Km	0,28
100 Mbps	1500 Bytes	100 m	0,0028
100 Mbps	1500 Bytes	10 km	0,28
100 Mbps	1500 Bytes	1.000 km	27,8
1 Gbps	1500 Bytes	100 m	0,028
1 Gbps	1500 Bytes	10 km	2,8
1 Gbps	1500 Bytes	1.000 km	277,8
1 Gbps	1500 Bytes	36.000 km	10.000
100 Gbps	1500 Bytes	100 m	2,8
100 Gbps	1500 Bytes	10 km	277,8
100 Gbps	1500 Bytes	1.000 km	27.777,8
100 Gbps	1500 Bytes	36.000 km	1.000.000

3.2.4.1 Leistungsanalyse bei Stop-and-Wait



Stop-and-Wait ohne Fehler Hierbei wird die ACK-Sendezeit und Bearbeitungszeit erstmal vernachlässigt. Wir erhalten dann einen Durchsatz von

$$\text{Durchsatz} = \frac{L}{L/R + 2D}$$

, wobei D die Ausbreitungsverzögerung, R die Bitrate und L die Paketgröße ist. Normiert durch die Bitrate[1] ergibt sich ein normierter Durchsatz von

$$\text{normierter Durchsatz} = S = \frac{L}{L/R + 2D} \cdot \frac{1}{R} = \frac{1}{1 + 2 \cdot a}.$$

Nachteil: Somit ergibt sich ein schlechterer Durchsatz für größere a, da hier der Kanal nicht gefüllt werden kann.

[1] Ermöglicht besseren Vergleich bei verschiedenen Bitraten.

Stop-and-Wait mit Fehlern Es erfolgt eine Sendewiederholung nach einem Fehler. Wir nehmen **stark vereinfachend** an, dass die Fehler *unabhängig voneinander* mit der Wahrscheinlichkeit p auftreten. Wir haben einen Timeout von 2D. Sei N dann die mittlere Anzahl, mit der jedes Paket gesendet werden muss, so ergibt sich für den Durchsatz und den normierten Durchsatz:

$$\text{Durchsatz} = \frac{L}{N \cdot (L/R + 2D)} \qquad S = \frac{1}{N \cdot (1 + 2 \cdot a)}$$

N berechnet sich dann über den Erwartungswert der geometrischen Verteilung, denn die Wahrscheinlichkeit, dass ein Paket i-mal gesendet werden muss, ist gleich der Wahrscheinlichkeit von $i-1$ fehlerhaften Sendungen gefolgt von einem fehlerfreiem Senden (geometrische Verteilung). Der Erwartungswert ist dann

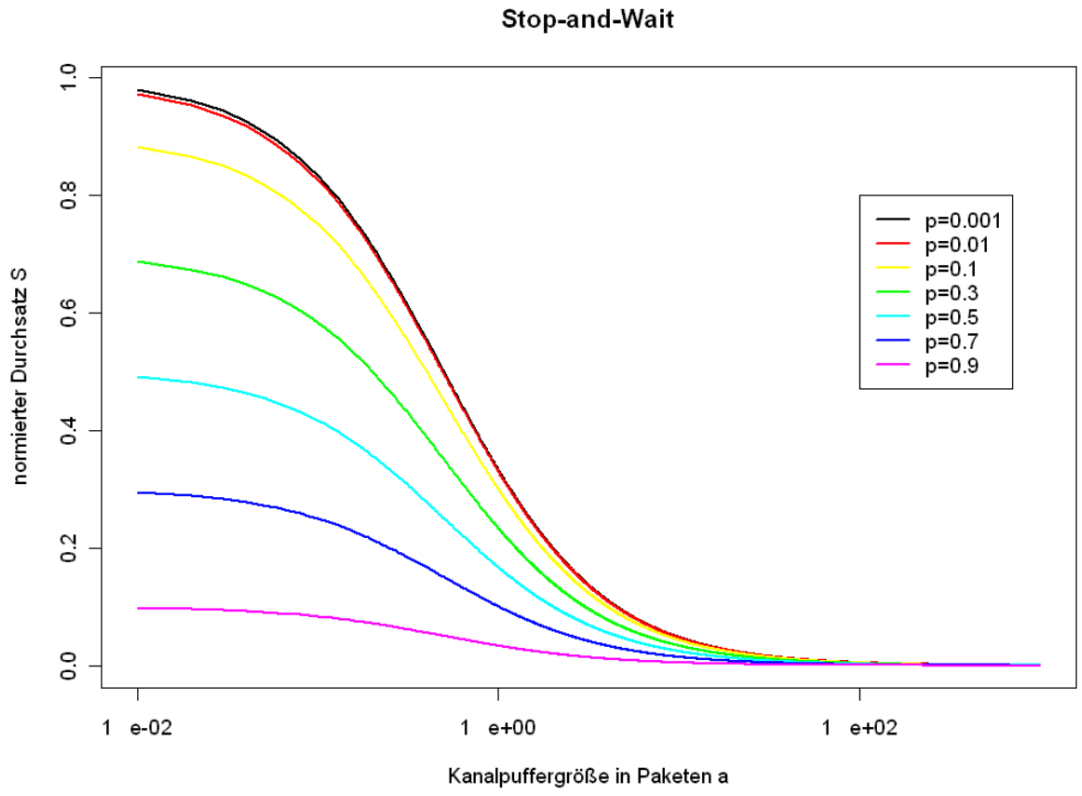
$$N = \sum_{i=1}^{\infty} i \cdot p^{i-1} \cdot (1-p) = (1-p) \cdot \frac{d}{dx} \left(\sum_{i=0}^{\infty} p^i \right) = (1-p) \cdot \left(\frac{1}{1-p} \right)' = \frac{1}{1-p}.$$

Einsetzen ergibt einen normalisierten Durchsatz von

$$S = \frac{1-p}{(1+2 \cdot a)}.$$

Nachteil: Somit ergibt sich ein schlechterer Durchsatz für größere a **sowie p**, da hier der Kanal nicht gefüllt werden kann.

Ergebnisse Fasst man den normierten Durchsatz von Stop-and-Wait als Funktion von a auf und betrachtet eine Schar über p, so stellt man fest, dass für große a der Durchsatz abfällt und für große p ebenfalls kleiner ist.



3.2.4.2 Leistungsanalyse bei Schiebefensterprotokollen

Schiebefensterprotokolle ohne Fehler Wir unterscheiden hier zwei Fälle, entweder ist das Fenster[2] groß genug, um zu senden, bis ACK zurückkommt (Variante 1), oder nicht (Variante 2). Die Bedingung für Variante 1 ist

$$W \geqslant \frac{L/R + 2D}{L/R} = 1 + 2 \cdot a,$$

es ergibt sich ein normierter Durchsatz von

$$S = \frac{W \cdot L}{W \cdot L/R} \cdot \frac{1}{R} = 1.$$

Die Bedingung für Variante 2 ist

$$W < 1 + 2 \cdot a,$$

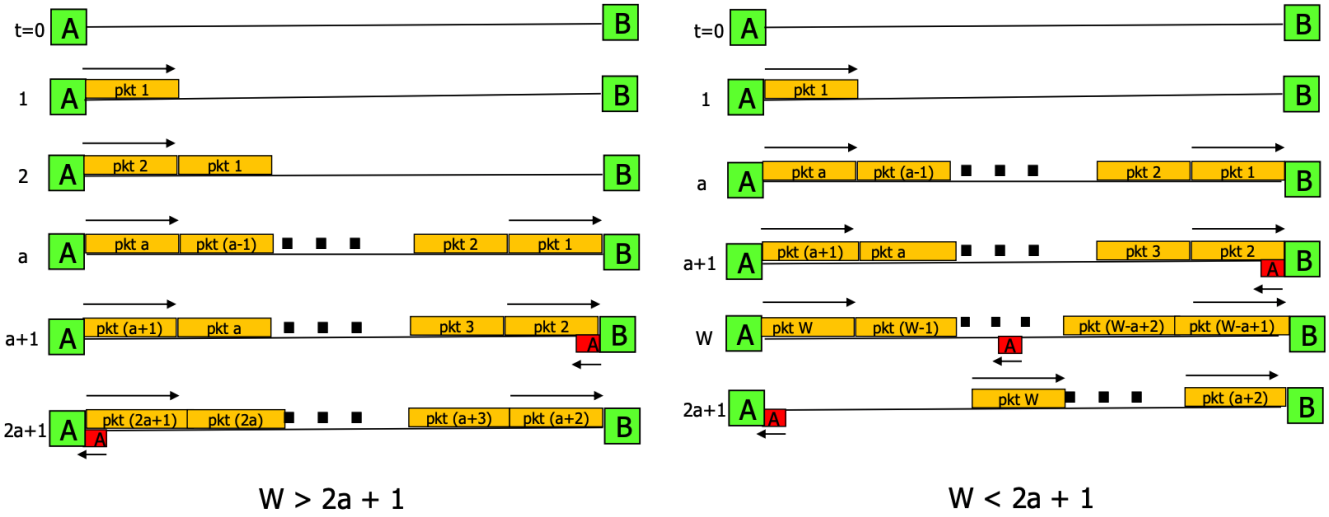
es ergibt sich ein normierter Durchsatz von

$$S = \frac{W \cdot L}{L/R + 2D} \cdot \frac{1}{R} = \frac{W}{1 + 2 \cdot a}.$$

Zusammengefasst:

$$S = \begin{cases} 1 & , \text{falls } W \geqslant 1 + 2a \\ \frac{W}{1+2\cdot a} & , \text{falls } W < 1 + 2a \end{cases}$$

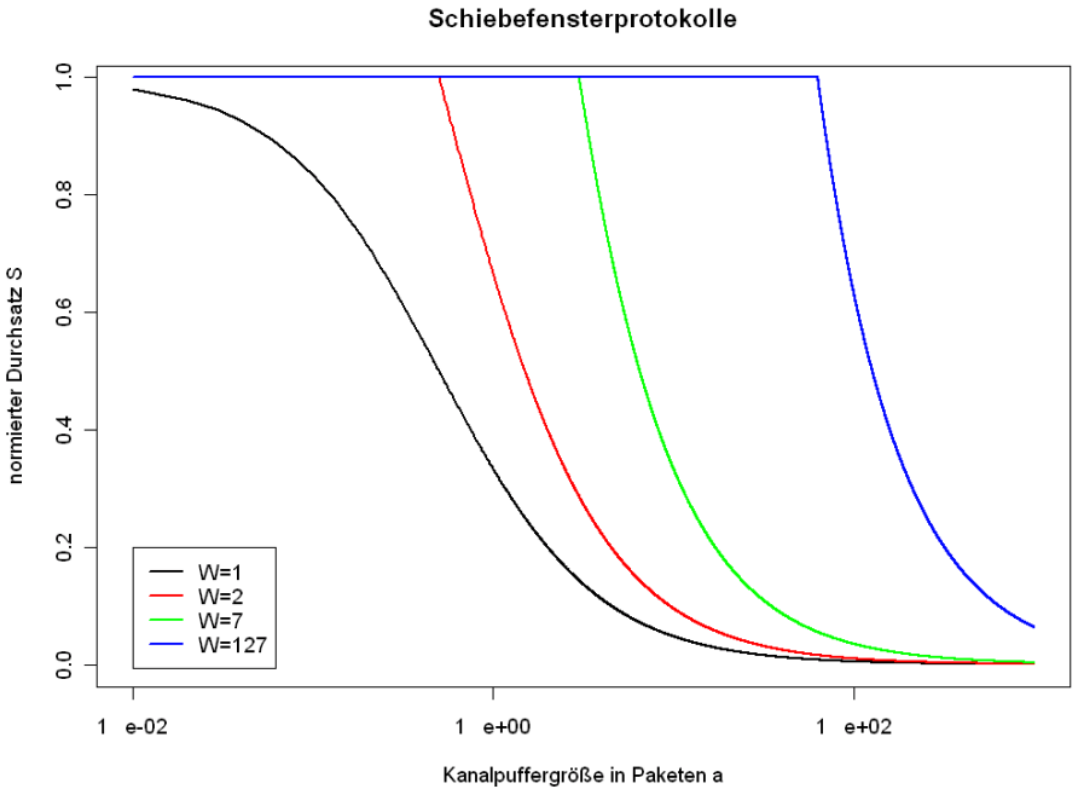
Zeitablauf beim Schiebefensterprotokoll



W > 2a + 1 W < 2a + 1

Ergebnisse I Fasst man auch hier den normierten Durchsatz von Schiebefensterprotokollen als Funktion von a auf und betrachtet eine Schar über W, so erkennt man dass eine Fenstergröße W bis zu einer maximalen Größe $a = (W-1)/2$ reicht um den Kanal vollständig zu füllen, wohingegen anschließend der Durchsatz sinkt.

[2] für eine Fenstergröße mit W Paketen der Größe L



Selective Repeat mit Fehlern Wir nehmen nun wieder unabhängige Fehler mit Wahrscheinlichkeit p an. Es ergibt sich für N

$$N = \frac{1}{1-p}.$$

Der Durchsatz im fehlerfreien Fall muss dann durch N geteilt werden, es ergibt sich:

$$S = \begin{cases} \frac{1}{N} = \frac{1}{1/(1-p)} = 1-p & \text{, falls } W \geqslant 1+2a \\ \frac{W}{N\cdot(1+2\cdot a)} = \frac{W\cdot(1-p)}{1+2\cdot a} & \text{, falls } W < 1+2a \end{cases}$$

$$S = \begin{cases} 1-p & \text{, falls } W \geqslant 1+2a \\ \frac{W\cdot(1-p)}{1+2\cdot a} & \text{, falls } W < 1+2a \end{cases}$$

Go-Back-N mit Fehlern Bei Go-Back-N erfordert jeder Fehler eine Sendewiederholung von K Paketen. Wir nehmen nun an, dass im Fehlerfall das Fenster gefüllt ist und alle Pakete des Fensters erneut gesendet werden müssen. Dann gilt für K:

$$K = \begin{cases} 1+2\cdot a & \text{, falls } W \geqslant 1+2a \\ W & \text{, falls } W < 1+2a \end{cases}$$

Wenn das fehlerhafte Paket i-mal gesendet wird, müssen insgesamt $1+(i-1)\cdot K = (1-K)+K\cdot i$ Pakete gesendet werden. Somit ergibt sich für N:

$$N = \sum_{i=1}^{\infty}((1-K)+K\cdot i)\cdot p^{i-1}\cdot(1-p) = (1-K)(1-p)\cdot\sum_{i=1}^{\infty}p^{i-1} + K(1-p)\sum_{i=1}^{\infty}i\cdot p^{i-1} = 1-K+\frac{K}{1-p}$$

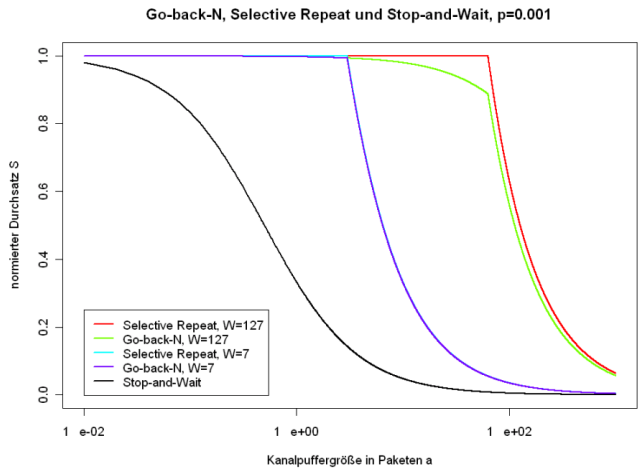
Wir erhalten damit für unser N:

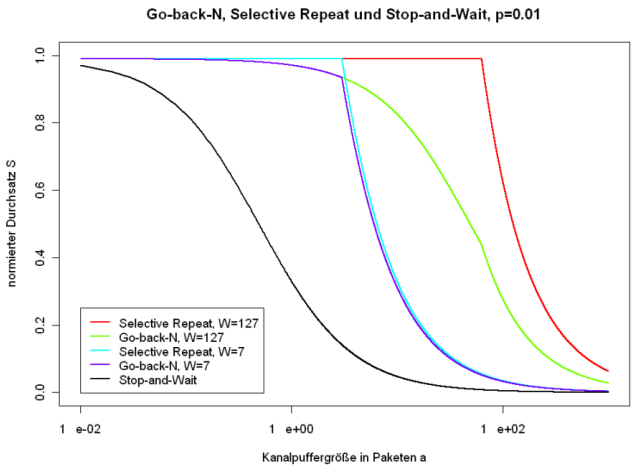
$$N = \begin{cases} \frac{1-p+Kp}{1-p} = \frac{1-p+(1+2a)p}{1-p} = \frac{1+2ap}{1-p} & \text{, falls } W \geqslant 1+2a \\ \frac{1-p+Kp}{1-p} = \frac{1-p+Wp}{1-p} & \text{, falls } W < 1+2a \end{cases}$$

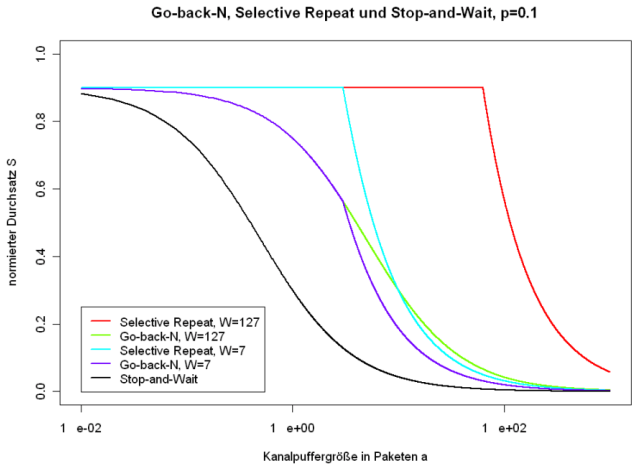
Division des Durchsatzes ohne Fehler durch N ergibt dann:

$$S = \begin{cases} \frac{1-p}{1+2ap} & \text{, falls } W \geqslant 1+2a \\ \frac{W\cdot(1-p)}{(1-p+Wp)\cdot(1+2a)} & \text{, falls } W < 1+2a \end{cases}$$

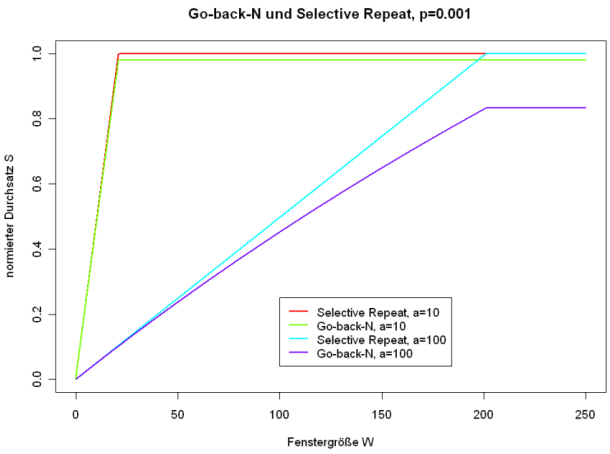
Ergebnisse II Fasst man hier den normierten Durchsatz als Funktion von a auf, so stellen wir fest, dass sich erst für *größere Fenster* (W) oder höhere Verluste (p) ein spürbarer Vorteil von Selective Repeat gegenüber Go-Back-N:

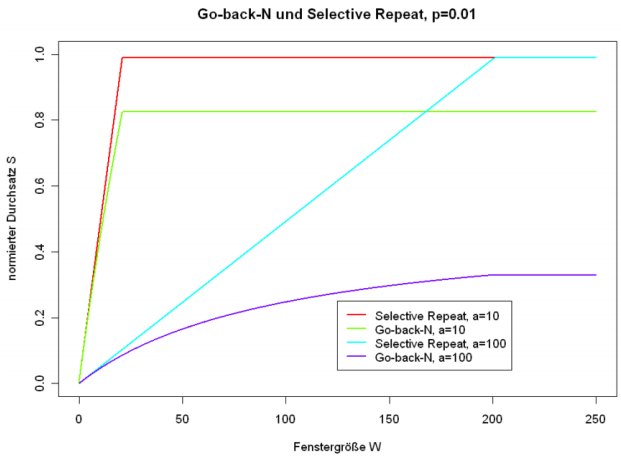


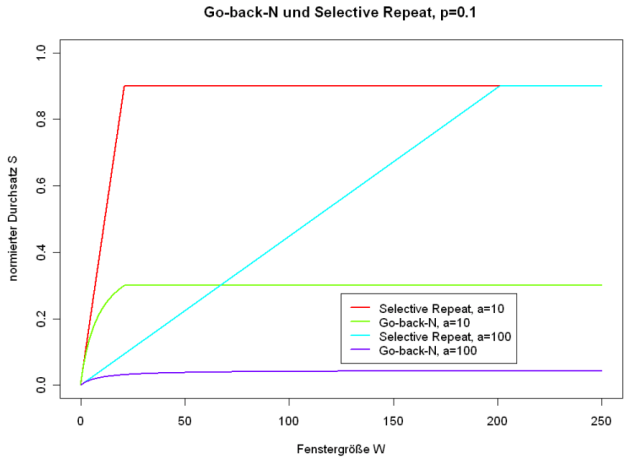




Fasst man hingegen den normierten Durchsatz als Funktion von *W* auf, so stellen wir fest, dass es für große a und große Fenster einen erheblichen Unterschied zwischen Go-Back-N und Selective Repeat gibt, welcher mit größerem p immer deutlicher wird:

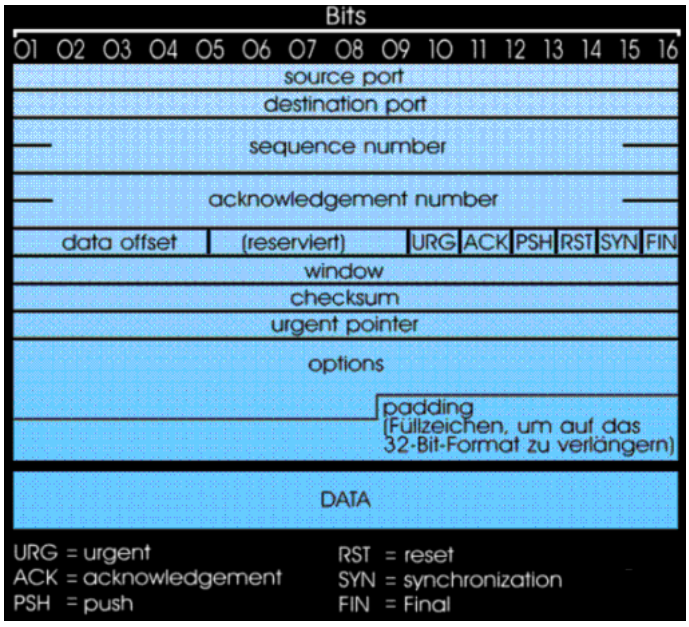






3.3 TCP

Das *Transmission Control Protocol* ist das vertreitete zuverlässige Transportprotokoll im Internet. Es ist unter anderem in den RFCs 793, 1323, 2018 und 5681 definiert und baut auf **Punkt-zu-Punkt**-Verbindungen auf. Es gibt einen Sender und einen Empfänger. Es hat einen **reihenfolgewahrenden** Bytestrom und eine **fensterbasierte Fehlerkontrolle**. Es ist eine **vollduplex** Übertragung, es gibt damit als zwei entgegengesetzte Datenströme. Zudem ist TCP **verbindungsorientiert**, man baut Verbindungen auf als auch ab. Ebenfalls sind Mechanismen wie **Fluss-** und **Überlastkontrolle**, wobei erstere sich bemüht eine Überscheitung der Kapazität des Empfängers und zweitere sich bemüht eine Überlastung des Netzes zu verhindern. Die Pseudoköpfe sind identisch mit denen, welche bei UDP verwendet werden.



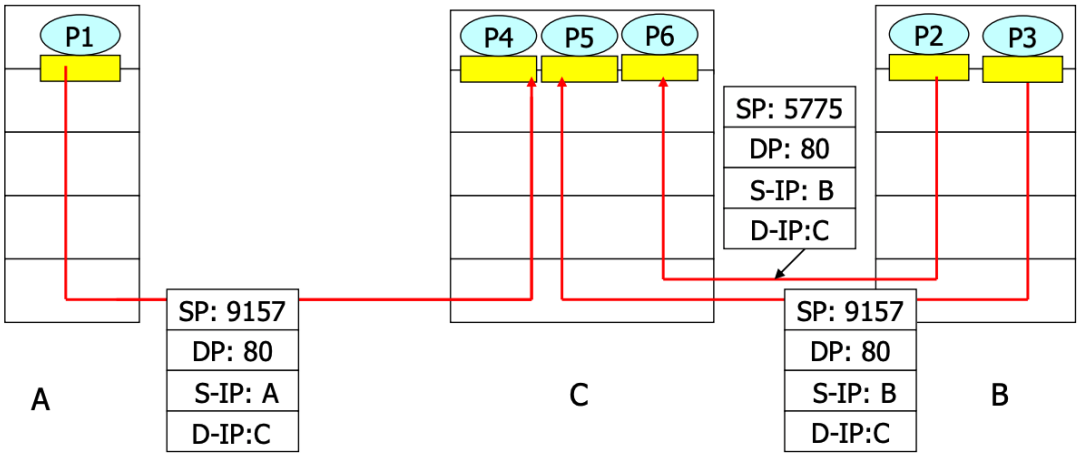
Segmentformat Die Sequenznummer, stellt die Nummer des ersten Bytes des Segments im Bytestrom dar. Ebenso stellt die *acknowledgement number* die Nummer des nächsten erwarteten Bytes im Bytestrom dar. Es gibt verschiedene Flags

mit Steuerinformationen für weitere Informationen hierfür siehe die Glossareinträge. Das Fensterfeld (*window* oder auch *AdvertizedWindow*) enthält die Fenstergröße, welche vor allem bei der Flusssteuerung relevant ist. Die Prüfsumme wird genauso wie bei UDP gebildet.

Multiplexen und Demultiplexen Eine jede TCP-Verbindung ist eindeutig durch das 4-Tupel bestehend aus der Quell- und Ziel-IP-Adresse sowie Quell- und Zielportnummer gekennzeichnet. Dies wird beispielsweise durch die Socket-API (ähnlich wie bei UDP) realisiert:

```
ServerSocket welcomeSocket = new ServerSocket(6789);
```

Zwei Sockets sind unterschiedlich, falls mindestens ein Wert des 4-Tupels sich unterscheidet. Damit können über einen Port viele TCP-Verbindungen laufen. Ein Beispiel:



3.3.1 Fehlerkontrolle

Die Fehlerkontrolle bei TCP ist eine Mischform von Go-Back-N, Selective Repeat und weiterer Elemente. So gibt es ...

- ... einen **Puffer auf Sender- und Empfängerseite**
- ... **einen** Timer
- ... **kumulative ACKs**
- ... Sequenz- und ACK-Nummern, welche sich nicht auf Pakete beziehen. So stellt die Sequenznummer die Position des ersten Bytes des Segments im Bytestrom und die ACK-Nummer die Position des nächsten erwarteten Bytes im Bytestrom dar.

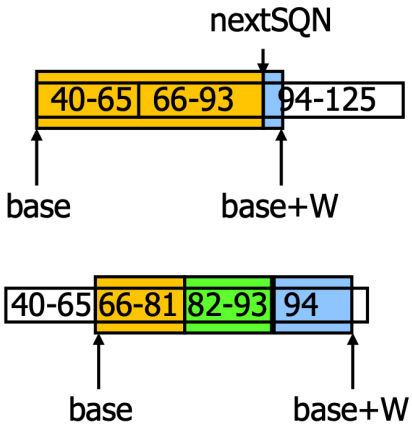
Es gibt diverse Implementierungen, wobei hier nur auf eine vereinfachte „Standardform“ eingegangen wird.

Bitfehler können hierbei genauso behandelt werden wie bei Go-Back-N oder Selective Repeat.

Der Sender darf bei TCP wieder mehrere Segmente vor Erhalt eines ACKs senden — bis zu einer vom Verfahren abhängigen Maximalzahl an Bytes. Er startet dann beim Senden des ersten Segments eines Fensters einen Timer und puffert die unbestätigten Segmente. Wenn der Timer abläuft, sendet er **das erste unbestätigte Segment** des Fensters erneut. Der Empfänger auf der anderen Seite sendet *kumulative ACKs* mit der Position des ersten noch nicht empfangenen Bytes. Das Fenster wird dann auf Sender- und Empfängerseite immer bis zur nächsten Lücke geschoben.

Sendefenster `base` bezeichnet das erste Byte des Fensters, `W` die Länge des Fensters, `base + W` das erste Byte außerhalb des Fensters und `nextSQN` das erste Byte des nächsten noch nicht gesendeten Segments. Das Fenster auf Senderseite enthält dabei versendete und **unbestätigte** und ungesendete Pakete.

Empfängerfenster Die Parameter sind dieselben wie auch beim Sendefenster bereits. Das Fenster auf Empfängerseite enthält dabei Lücken, empfangene Pakete und Platz für unempfangene Pakete.



Informelle Beschreibung *Verhalten des Senders*:

(1) Wenn Daten zum Senden sowie genug Platz im Fenster vorhanden sind, so erstelle ein Segment mit `nextSQN` und versende es mit IP. Erhöhe dann `nextSQN` um die Länge der Daten. Wenn es das erste Paket im Fenster ist, so starte den Timer.

(2) Wenn ein ACK mit ACK-Nr. im Fenster zurückkommt, so schiebe das Fenster bis zu dieser ACK-Nr. Wenn das Fenster leer ist, so stoppe den Timer, andernfalls starte den Timer neu.

(3) Wenn der Timeout abläuft, sende das erste unbestätigte Paket des Fensters erneut und starte den Timer erneut.

Verhalten des Empfängers: Wenn ein Segment ankommt und ...

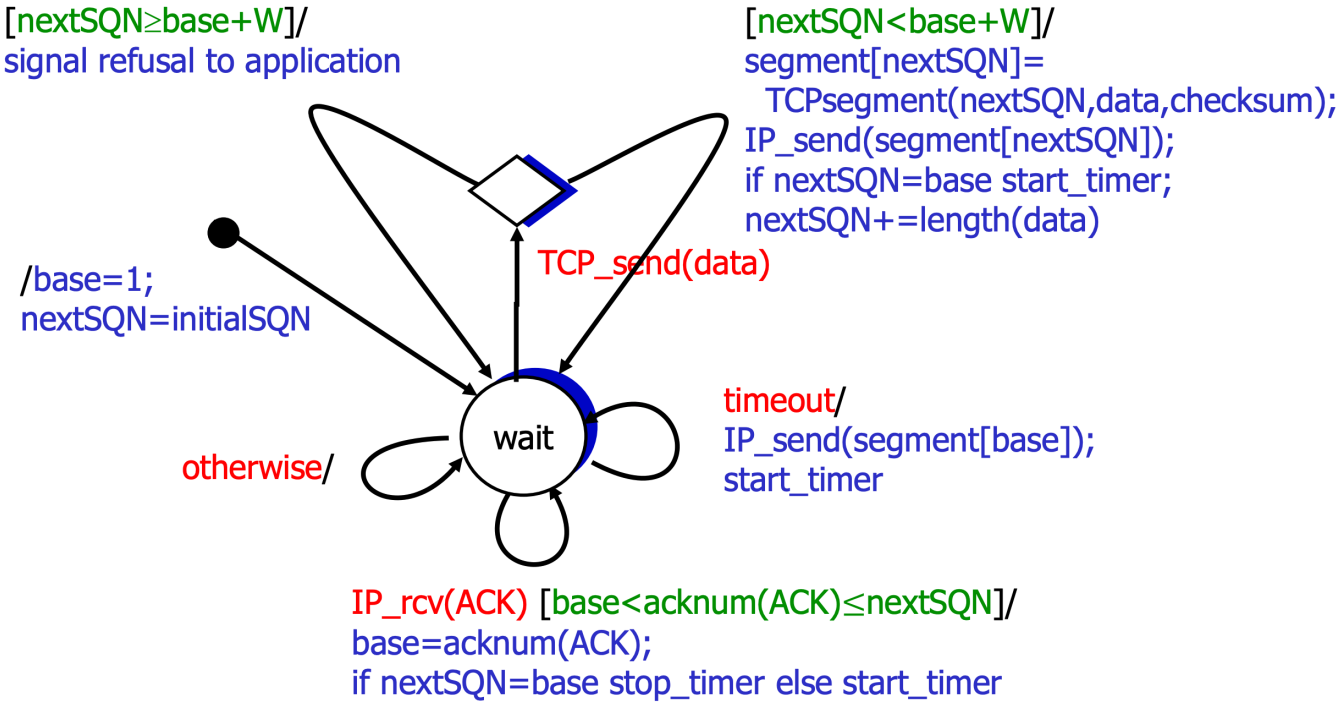
... `SQN = Fensteranfang` gilt *und alle vorherigen Segmente bereits bestätigt sind*, so schiebe den Fensteranfang bis zum nächsten erwarteten Byte und warte ein Timeout (500 ms). Wenn bis dahin kein neues Segment ankommt, schicke ein ACK mit dem Fensteranfang.

... `SQN = Fensteranfang` gilt *und ein vorheriges Segmente* ***noch nicht*** *bestätigt wurde*, so schiebe den Fensteranfang bis zum nächsten erwarteten Byte und schicke sofort ein kumulatives ACK mit dem Fensteranfang.

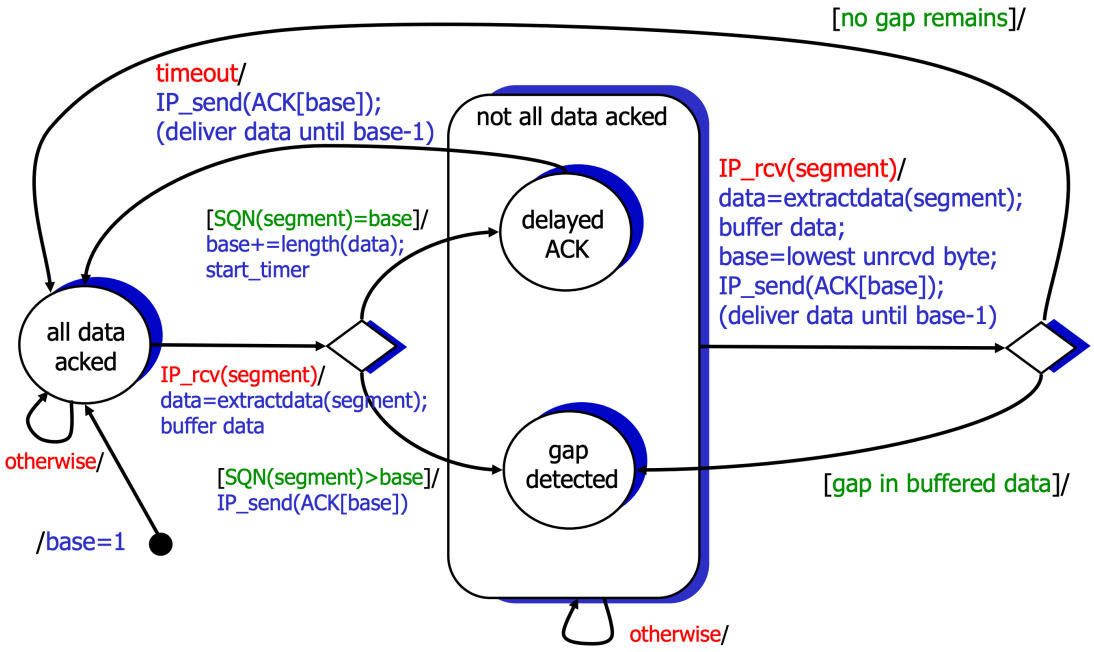
... `SQN > Fensteranfang` gilt, so puffere die Daten und schicke sofort ein kumulatives ACK mit dem Fensteranfang.

... `es eine Lücke` teilweise oder ganz `füllt`, so puffere die Daten, schiebe den Fensteranfang bis zum nächsten erwarteten Byte und schicke sofort ein kumulatives ACK mit dem Fensteranfang.

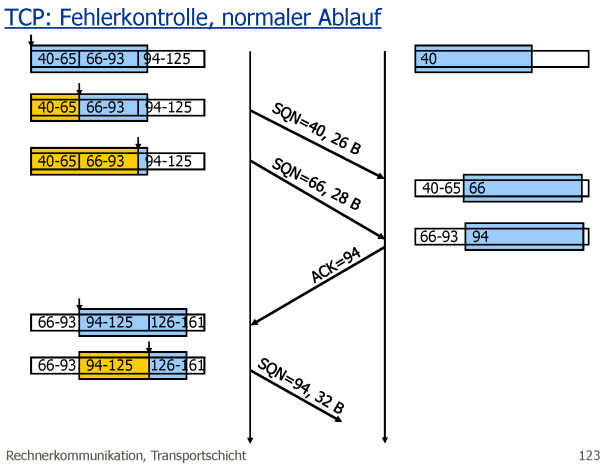
Statechart — Sender

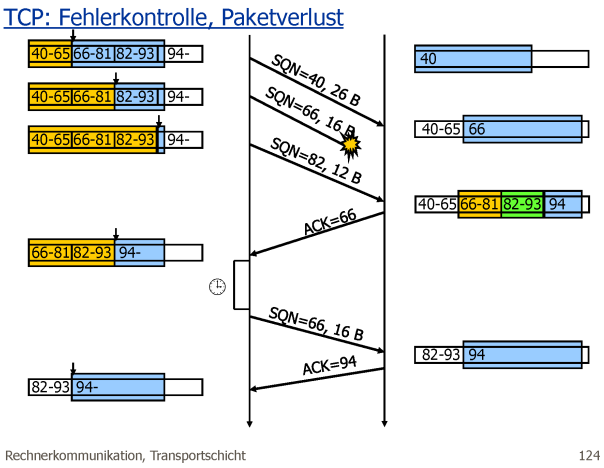


Statechart — Empfänger

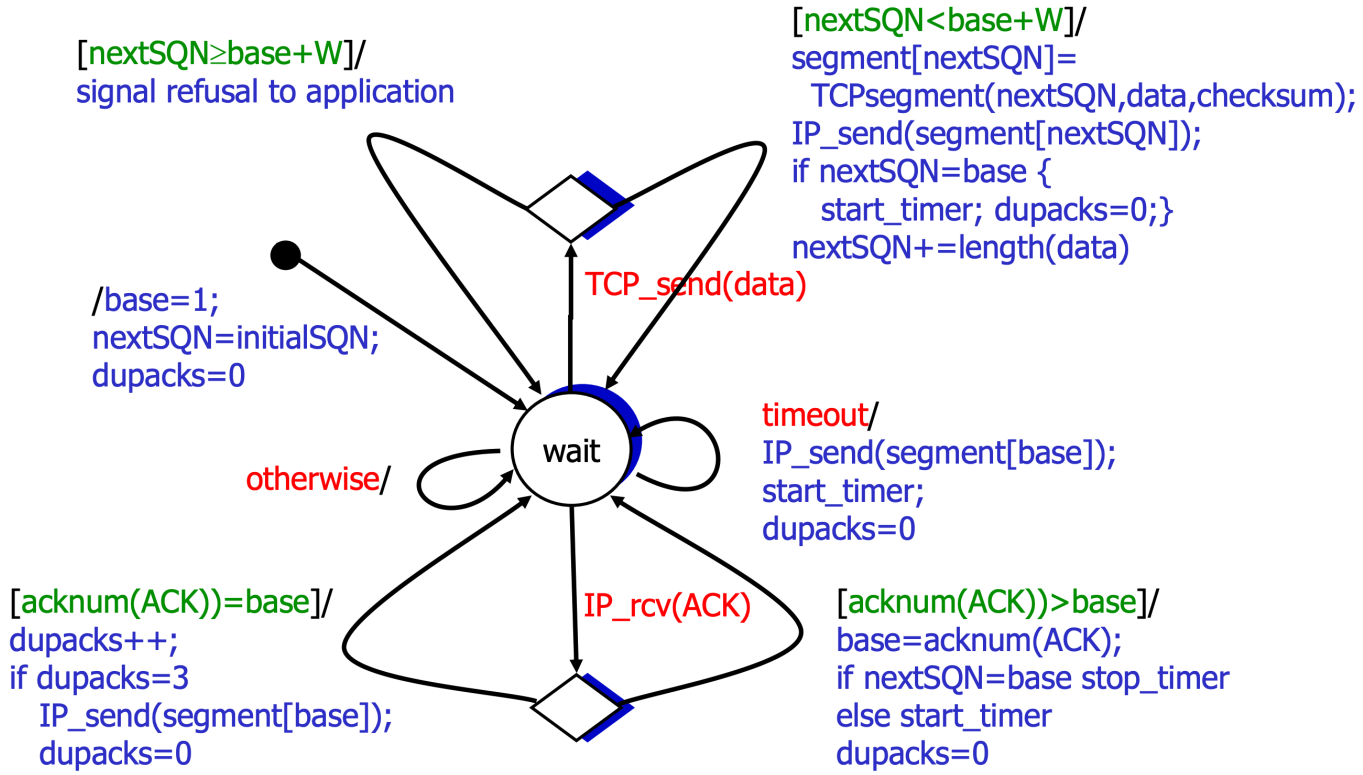


Ablauf Beispielhafte Abläufe mit den allermeisten Möglichkeiten:





Fast Retransmit Wir sehen schnell, dass es relativ lange dauert, bis ein Paketverlust bemerkt wird. Bei mehreren Paketverlusten dauert es noch länger. Wir verwenden deswegen **Fast Retransmit**. Hier wird bei 3 doppelten ACKs, wobei eine doppelte ACKs ACKs mit der gleichen ACK-Nummer sind, eine Sendewiederholung des Segments mit der Sequenznummer ausgelöst. Dazu wird das Statechart vom Sender leicht angepasst, da jetzt ein ACK-Zähler `dupacks` notwendig ist. Beim erwarteten ACK wird der Zähler auf 0 gesetzt, bei doppelten ACKs wird der Zähler inkrementiert und bei einem Wert von 3 der Fast Retransmit ausgelöst.

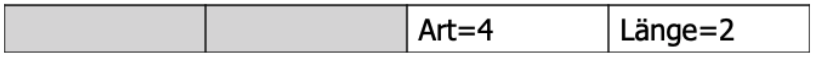


Anmerkungen

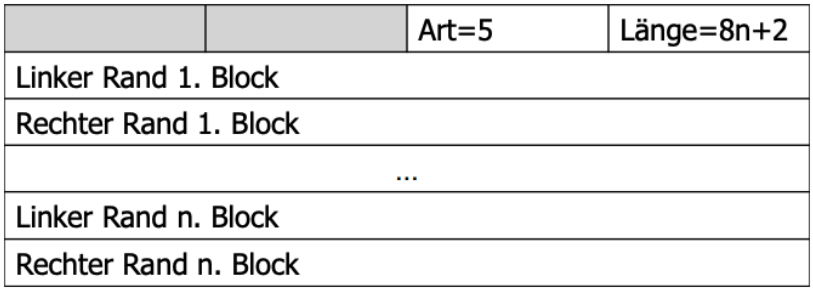
TCP verwendet **vollduplex** Verbindungen. Es werden also zwei logische Verbindungen realisiert, eine davon in jede Richtung.

ACKs reisen hierbei im Huckepack (**piggybacking**). Dabei wartet der Empfänger bei Empfang einer Nachricht und sendet das ACK nicht sofort an den Sender zurück. Der Empfänger wartet bis zum nächsten zu sendenden Datenpaket und fügt das *verspätete ACK* an.
Vorteil: Verbessert Effizienz und macht besseren Gebrauch der verfügbaren Kanalbandbreite.
Nachteil: Es fügt zusätzliche Komplexität hinzu. Ebenso kann der Empfänger den Service lahmlegen falls er nichts zu senden hat, denn wartet er zu lange, so findet eine Neusendung des Bytestroms seiten des Senders statt. Eine Lösungsmöglichkeit ist ein *receiver* und *emitter timeout* auf Seiten des Empfängers und Senders.

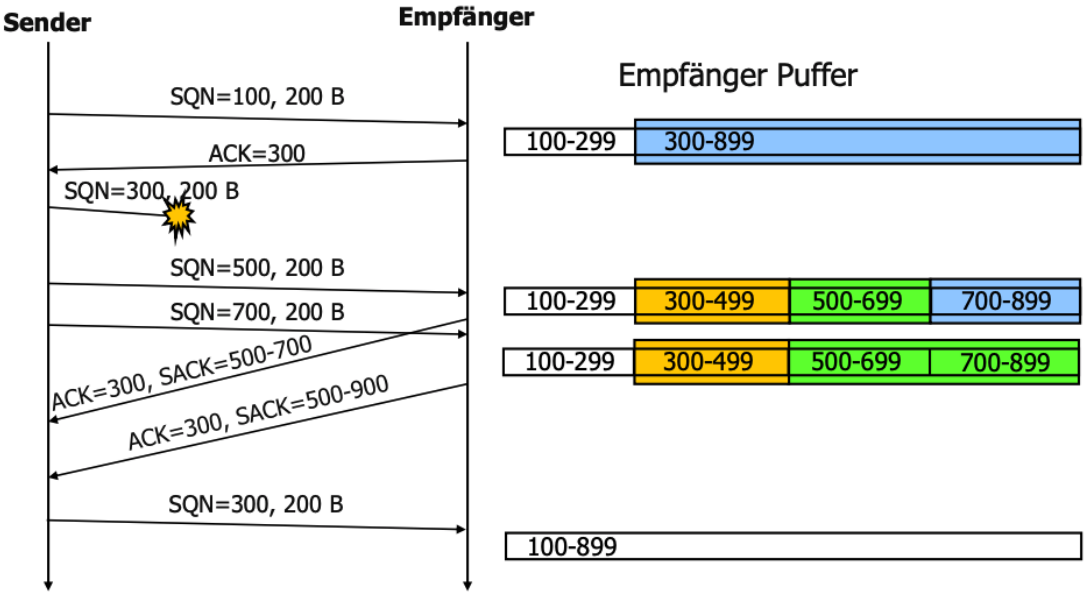
SACK — Selective ACKs Ein Problem, welches jetzt noch nicht gelöst ist, ist, dass **mehrfache Paketverluste** in einem Fenster einen katastrophalen Effekt auf den Durchsatz haben, da der Sender — dem kumulativen ACK geschuldet — für **jedes** fehlende Paket mindestens eine RTT warten muss. Eine Lösung bietet die TCP-Erweiterung der selektiven Acknowledgments, bei welcher zusätzlich im Optionsfeld noch selektive ACKs gesendet werden. Diese informieren den Sender dann über einzelne Pakete, welche im Fenster des Empfängers liefen und nicht durch kumulative ACKs bestätigt werden können, weil es Lücken davor gibt. Der Sender muss diese nach einem Timeout nicht erneut übertragen.
Allgemein gilt aber, dass normale ACKs unverändert weitergeschickt werden (*Kompatibilität*), SACKs für das erste Paket außer der Reihe verschickt werden und der Empfänger *so viele SACKs wie möglich sendet*. Der Sender initiiert dann entsprechende Neuübertragungen.
Wenn das `SYN`-Flag gesetzt ist, kann der Sender dem Empfänger mitteilen, dass SACKs erlaubt sind. Dazu wird das Optionsfeld folgendermaßen gesetzt:



Anschließend kann der Empfänger jeden empfangenen Block, welcher vom Byte des linken Rands bis vor das Byte des rechten Rands geht, als Option übertragen. Das Format ist folgendermaßen aufgebaut:



Dabei ist die Mitteilung der Erlaubnis der SACKs im Verbindungsaufbau, die Übertragung der SACKs im Dateitransfer zu erledigen. Ein Beispiel für SACKs:



Sequenznummerraum Das Sequenznummernfeld ist 32 Bit groß, womit es 2^{32} Sequenznummern gibt. Die Bedingung für Schiebefensterprotokolle ist damit auch erfüllt, betrachtet man die Anzahl an Sequenznummern mit der maximalen Fenstergröße so stellt man fest, dass $2^{32} >> 2^{16}$. Die Zeiten für den Überlauf der Sequenznummern lassen sich auch beispielhaft abschätzen mit 57 Minuten bei 10 Mbps und 34 Sekunden bei 1 Gbps. Wir stellen fest, dass es für hohe Bitraten ein etwas kurzes Zeitintervall ist. Eine TCP-Erweiterung verwendet deswegen Zeitstempel im Optionsfeld für eine weitere Unterscheidung um Verwechslungen von Segmenten zu vermeiden.

3.3.2 Verbindungsauf- und -abbau

Verbindungsaufbau Der Verbindungsaufbau im Client (①) und Server (②) veranlasst einen **3-Wege-Handshake**, am Beispiel mit

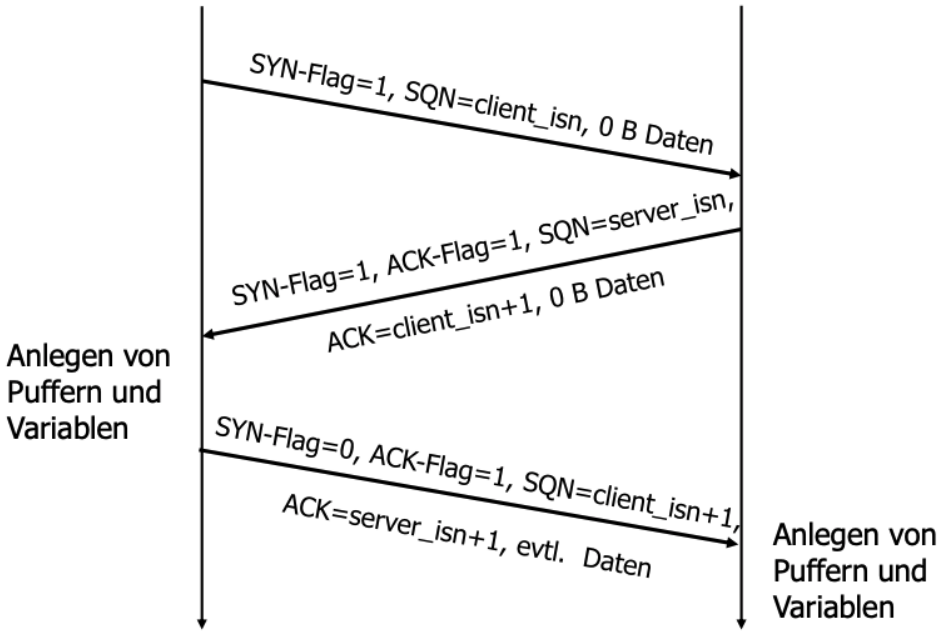
① `new Socket("hostname", "port");` ② `serversocket.accept();`

Weg 1 — **SYN-Segment**: Der Client sendet ein Segment mit `SYN`-Flag auf `1` und einer zufälligen initialen Client-SQN (`client_isn`) **ohne Daten**.

Weg 2 — **SYNACK-Segment**: Der Server sendet ein Segment mit `SYN`-Flag und `ACK`-Flag auf `1`, einer initialen zufälligen Server-SQN (`server_isn`) und (`ACK = client_isn + 1`) **ohne Daten**. Er legt dann Puffer und nötige Variablen an.

Weg 3 — **ACK-Segment**: Der Client sendet ein Segment mit `ACK`-Flag auf `1` einer Sequenznummer von `client_isn + 1`, einer ACK-Nummer von `server_isn + 1` und gegebenenfalls ersten Daten. Er legt dann ebenfalls Puffer und nötige Variablen an.

Graphisch:

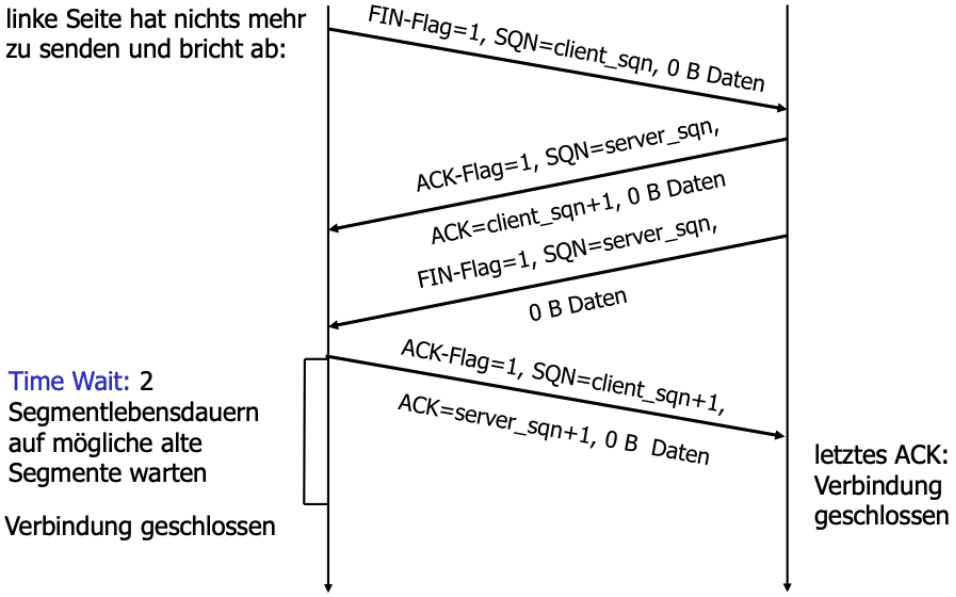


Sequenznummern bei SYN- und FIN-Segmenten

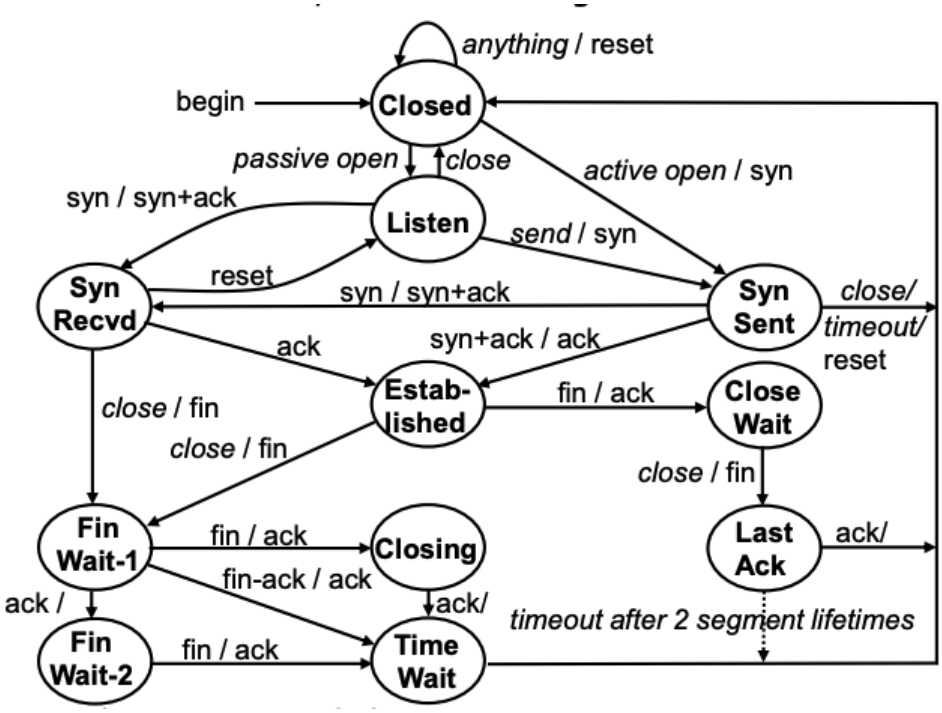
Segmente mit `SYN`-Flag oder `FIN`-Flag auf 1 dürfen **keine Daten** enthalten, die nächste Sequenznummer **muss** aber um eins inkrementiert werden, damit diese Segmente *explizit* bestätigt werden können.

Verbindungsabbau **Jede** Seite kann einen Verbindungsabbau durch ein Segment mit `FIN`-Flag auf 1 veranlassen. Die andere Seite bestätigt daraufhin mit `ACK`-Flag auf 1. **Beide** Seiten **müssen** ihre Hälfte der Verbindung schließen. Hat eine Seite geschlossen, sendet sie keine Daten mehr, **nimmt aber noch welche an**.

Es kommt dann auf der Seite, welche den Verbindungsabbau veranlasst, zum **Time Wait**. Hierbei muss diese Seite zum Schluss noch 2 Segmentlebensdauern, um noch mögliche alte Segmente zu empfangen *und neue TCP-Verbindungen davor zu schützen*. Übliche Werte liegen zwischen 30 Sekunden und 2 Minuten. Solange werden Variablen der Verbindung gehalten und der Socket nicht neu vergeben. Graphisch:



Zustandsmaschine Im RFC 793 ist eine Zustandsmaschine für den Verbindungsauf- und -abbau abgebildet:



3.3.3 Abschätzung der RTT

Der Timeout für Sendewiederholungen muss vom Sender gewählt werden. Dabei ist zu beachten, dass ein ACK frühestens nach RTT zurückkehren kann. Ist der gewählte Timeout zu klein kommt es zu unnötigen Sendewiederholungen, ist er hingegen zu groß, so kann erst spät auf Fehler reagiert werden. Der passende Timeout hängt dabei von der Konfiguration ab und *ändert sich dynamisch*. TCP verwendet **Zeitstempel** für das Segment und die ACK, deren Differenz dann die Messung der aktuellen RTT ist. Dann wird der **Durchschnitt** und die Abweichung aus mehreren Messungen bestimmt und daraus dann der Timeout abgeleitet. **Messungen bei Sendewiederholungen werden nicht verwendet!**

Bestimmung der RTT Jede Messung ergibt ein SampleRTT. Dann wird der **gleitende Durchschnitt** (*exponentially weighted moving average*) gebildet:

$$\mathsf{EstimatedRTT} = (1-\alpha)\cdot \mathsf{EstimatedRTT} + \alpha \cdot \mathsf{SampleRTT}$$

Bei großem α reagiert der Durchschnitt stark auf aktuelle Schwankungen, bei kleinerem α gibt es eine größere Stabilität, aber eine langsamere Reaktion auf mögliche Änderungen. Ein typischer Wert ist $\alpha = 0,125$.

Mittlere Abweichung Ähnlich der Standardabweichung wird wieder als gleitender Durchschnitt die mittlere Abweichung gebildet:

$$\mathsf{DevRTT} = (1-\beta)\cdot \mathsf{DevRTT} + \beta \cdot |\mathsf{SampleRTT} - \mathsf{EstimatedRTT}|$$

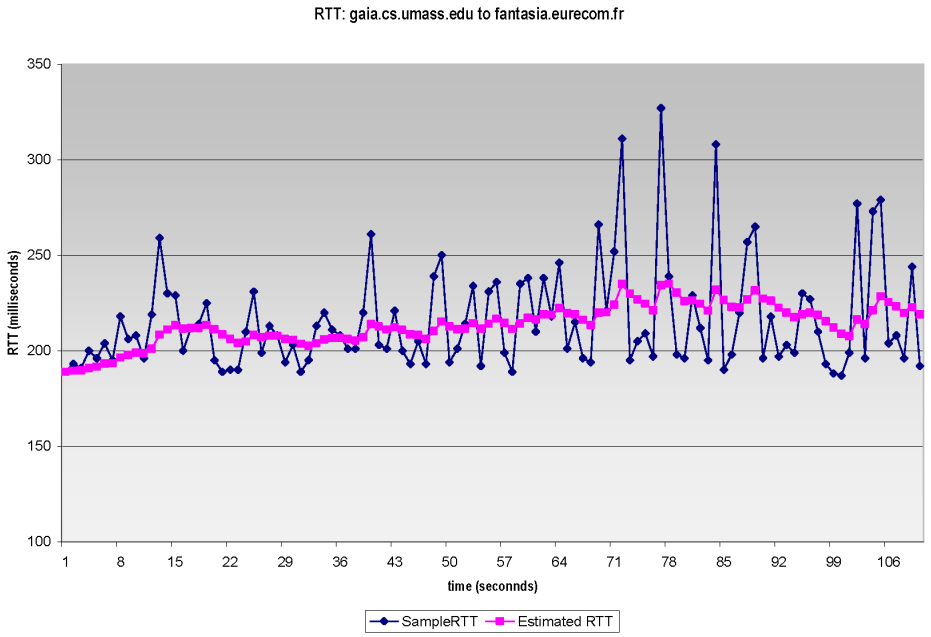
Ein typischer Wert für β ist $\beta = 0,25$.

Timeout Der Timeout berechnet sich dann über die geschätzte RTT und einer aus der Abweichung abgeleiteten Sicherheit:

$$\text{TimeoutInterval} = \text{EstimatedRTT} + 4 \cdot \text{DevRTT}$$

Es kommt zum **Timeout Backoff**, wenn der Timeout ausgelöst wird, da er dann verdoppelt wird und benutzt wird bis ein neues SampleRTT da ist.

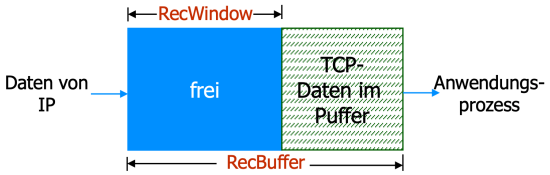
Beispiel



3.3.4 Fluss- und Überlastkontrolle

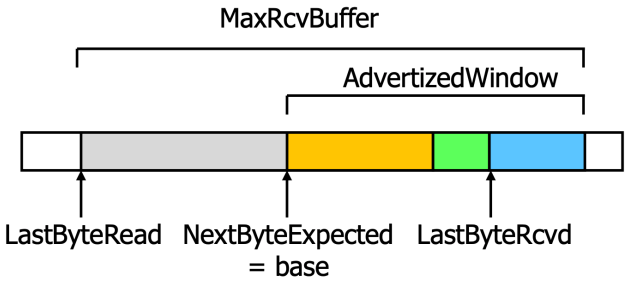
3.3.4.1 Flusskontrolle

Die Flusskontrolle (*flow control*) ist ein Mechanismus mit welchem der **Empfänger** den Sender steuern kann, damit er ihn nicht überlastet aber gleichzeitig so schnell wie möglich sendet. Dies wird üblicherweise über eine Benachrichtigung der **Fenstergröße** getan.

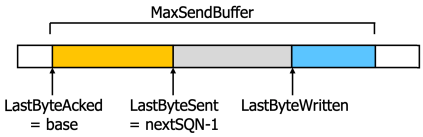


Grundlegendes Die Empfängerseite besitzt einen Puffer, in welchen das IP neue empfangene Daten einfügt. Die Anwendung liest hieraus Daten aus. Der jeweils freie Pufferplatz wird dann der Senderseite mitgeteilt. Die Senderseite besitzt ebenfalls einen Puffer, in welchen die Anwendung Daten schreibt und mit IP so viele Daten entfernt werden, wie es der Puffer der Empfängerseite zulässt. Die Anwendung auf Senderseite blockiert, wenn der Puffer voll ist, wodurch die Empfängeranwendung die Senderanwendung reguliert.

Empfängerpuffer `LastByteRead` ist das letzte an die Anwendung ausgelieferte Byte, `NextByteExpected` ist das nächste erwartete Byte. `LastByteRcvd` ist das letzte empfangene Byte, `MaxRcvBuffer` ist der insgesamt zu Verfügung stehende Pufferplatz. Der freie Pufferplatz, welcher dem Sender dann mitgeteilt wird ist mit `AdvertizedWindow` (`= MaxRcvBuffer - ((NextByteExpected - 1) - LastByteRead)`) beschrieben.



Sendepuffer `LastByteAcked` ist das letzte bestätigte Byte, `LastByteSent` ist das letzte gesendete Byte. `LastByteWritten` ist das letzte von der Anwendung geschriebene Byte und `MaxSendBuffer` ist der insgesamt zur Verfügung stehender Pufferplatz. Das `EffectiveWindow` berechnet sich durch `AdvertisedWindow - (LastByteSent - LastByteAcked)`.
Der Sender sendet nur, falls `EffectiveWindow > 0`, die Anwendung schreibt nur, falls `LastByteWritten` − `LastByteAcked` ⩽ `MaxSendBuffer` gilt.



Ablauf Initial wird das `AdvertizedWindow` möglichst groß eingestellt. Nach `AdvertizedWindow = 0` werden periodisch **Sondersegmente** mit einem Byte gesendet, sonst kommen eventuell nie ACKs mit wieder größerem `AdvertizedWindow` zurück.

Vermeidung des *Silly Window Syndroms*

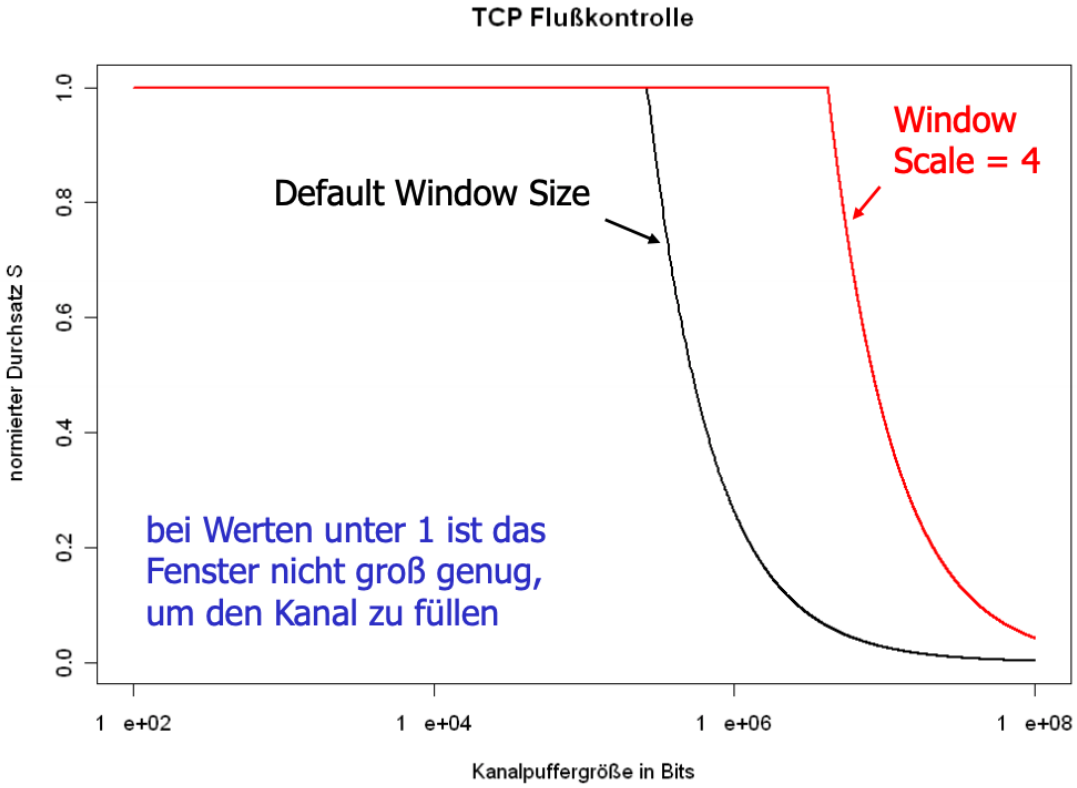
Segmente mit wenig Daten sind ineffizient. Wenn die Puffer voll sind und kleine Segmente gesendet werden, so werden ACKs für diese gesendet und der Sender wird erneut ein kleines Segment senden. Dies wird dann in einer Schleife passieren.

i Abhilfe schafft die MSS (Maximum Segment Size), welche standardmäßig 536 Bytes beträgt. Auf der Empfängerseite wird wenn der Empfänger ein `AdvertizedWindow = 0` bekannt gibt, bis er ein `AdvertizedWindow` ⩾ `MSS` bekannt geben kann gewartet. Auf der Senderseite sendet der Sender ein kleineres Segment als MSS nur, wenn keine weiteren unbestätigten Segmente unterwegs sind.

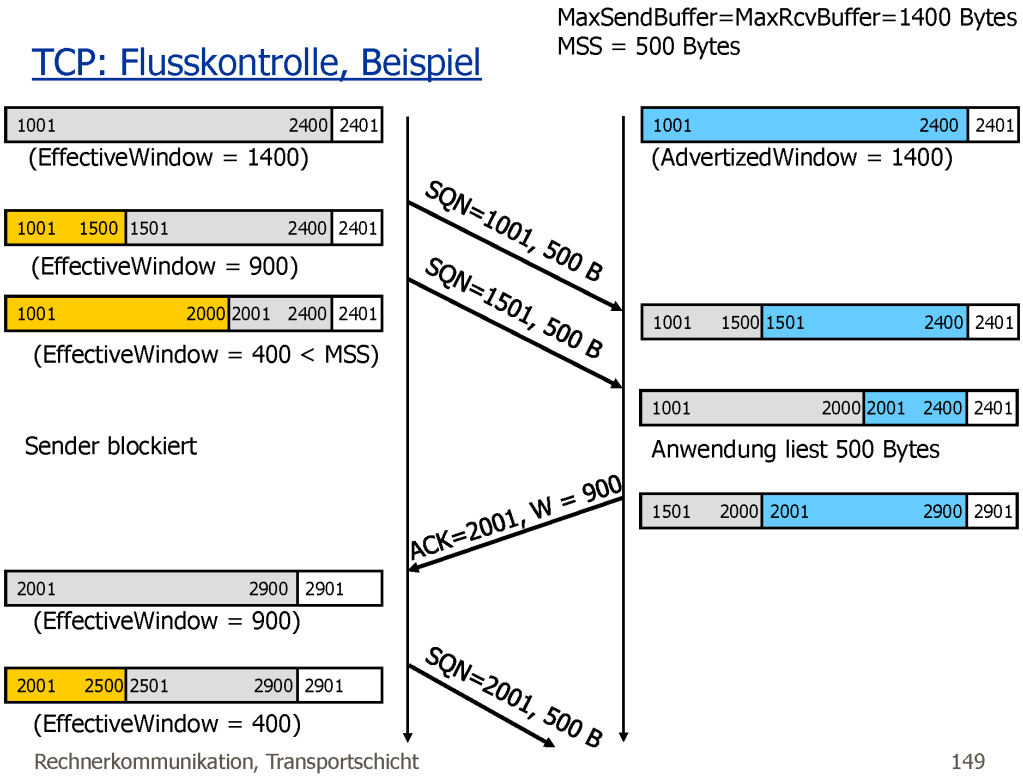
Perfekte Fenstergröße Bei der Frage, ob Fenster groß genug sind, werden zwei Fragen gestellt:

? *Kann es Verwechslungen von Segmenten geben?*
Das `AdvertizedWindow`-Feld ist 16 Bit groß, also kann auch das Fenster 2^{16} Bytes groß sein. Damit ist auch die Bedingung für Schiebefensterprotokolle erfüllt mit $2^{32} >> 2 \cdot 2^{16}$.

? *Kann der Sender den Kanal gefüllt halten?*
In der Graphik unten ist zu sehen, dass für manche Konstellationen das Bitraten-Verzögerungs-Produkt so groß ist, dass das maximale Fenster dafür nicht ausreicht. Als Abhilfe kann im Optionsfeld des ersten Segments ein **Window-Scale-Faktor** $F \leqslant 14$ gesetzt werden, womit sich die Fenstergröße dann aus `AdvertizedWindow` $\cdot 2^F$ ergibt. Es gilt immer noch $2^{32} > 2 \cdot 2^{30}$.



Beispiel



3.3.4.2 Überlastkontrolle

Die Überlastkontrolle (*congestion control*) ist ein Mechanismus, mit welchem der Sender davon abgehalten wird das **Netz** zu überlasten. Es kann durch **explizite Signale** des Netzes an den Sender erfolgen. In TCP gibt es klassischerweise keinen solchen expliziten Mechanismus, der Sender leitet

sich aus den **zurückgekommenen ACKs** Informationen über den Netzzustand ab und reagiert entsprechend. Inzwischen gibt es aber auch mit TCP und IP explizite Überlastbenachrichtigungen mit den *explicit congestion notification (ECN)*.

Grundlegendes Der TCP-Sender versucht aus den zurückkommmenden ACKs Informationen über die mögliche Senderate zu erhalten. Hierzu gibt es das **CongestionWindow**, welches zusammen mit der Flusskontrolle zur Ermittlung des tatsächlichen Sendefensters verwendet wird. Man berechnet somit ...

... $\mathsf{MaxWindow} = \min\left\{\mathsf{CongestionWindow},\ \mathsf{AdvertizedWindow}\right\}$ und ...

... $\mathsf{EffectiveWindow} = \mathsf{MaxWindow} - (\mathsf{LastByteSent} - \mathsf{LastByteAcked})$.

Die Bitrate ergibt sich dann ungefähr aus CongestionWindow pro RTT. Durch eine Vergrößerung des CongestionWindows vergrößert der Sender die Bitrate und versucht sich somit an die mögliche Bitrate anzunähern. Bei einem Verlust, welcher durch drei doppelte ACKs oder einem Timeout zu erkennen ist, wird das CongestionWindow und damit die Bitrate wieder verkleinert. Dazu gibt es drei Mechanismen:

(1) **Slow Start** — Am Anfang erhöht der Sender das CongestionWindow beginnend mit einer MSS exponentiell bis er durch drei doppelte ACKs erfährt, dass ein Segment verlorengegangen ist.

(2) **AIMD (Additive Increase, Multiplicative Decrease)** — Nach einem verlorengegangenem Segment wird das CongestionWindow halbiert und dann linear bis zum nächsten Erhalt von drei doppelten ACKs erhöht. Dies passiert nun „schleifenartig“.

(3) **Konservative Reaktionen nach einem Timeout** — Nach einem Timeout wird Slow Start bis zur **Hälfte des aktuellen CongestionWindows** betrieben, anschließend AIMD.

Slow Start Setze $\mathsf{CongestionWindow} \leftarrow \mathrm{MSS}$. Nach Erhalt eines ACKs setzte

$$\mathsf{CongestionWindow} \leftarrow \mathsf{CongestionWindow} + \mathrm{MSS}.$$

Wenn der Threshold erreicht ist mache mit AMID weiter, zu Beginn ist dieser jedoch unendlich.

AIMD Nach drei doppelten ACKs kommt es zum AIMD. Hier kommt es zu einem *multiplicative decrease*, bei dem der Threshold und das CongestionWindow auf

$$\mathrm{Threshold} = \frac{\mathsf{CongestionWindow}}{2} \qquad \text{und} \qquad \mathsf{CongestionWindow} = \frac{\mathsf{CongestionWindow}}{2}$$

gesetzt werden. Bei Erhalt eines ACKs wird dann das CongestionWindow auf

$$\mathsf{CongestionWindow} = \mathsf{CongestionWindow} + \mathrm{MSS} \cdot \frac{\mathrm{MSS}}{\mathsf{CongestionWindow}}$$

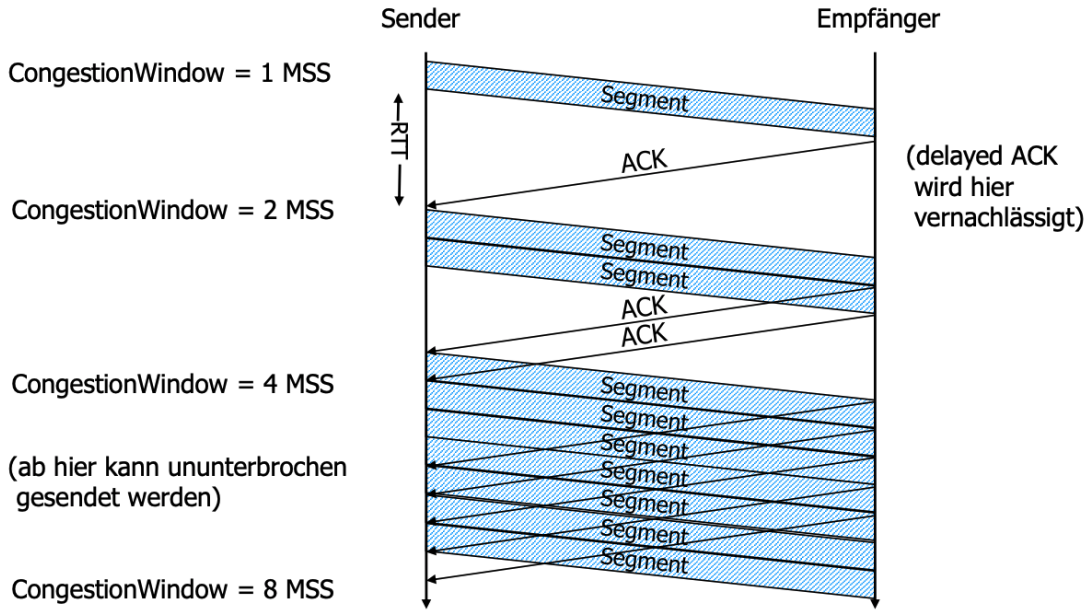
gesetzt. Hierdurch wird ein Wachstum um circa ein MSS pro RTT realisiert.

Timeout Hier wird der Threshold und das CongestionWindow auf

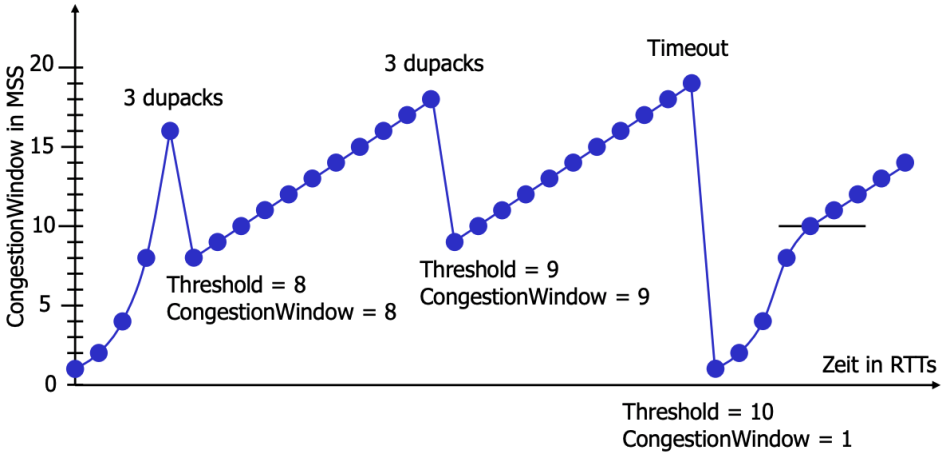
$$\mathrm{Threshold} = \frac{\mathsf{CongestionWindow}}{2} \qquad \text{und} \qquad \mathsf{CongestionWindow} = \mathrm{MSS}$$

gesetzt.

Beispiel für Slow Start



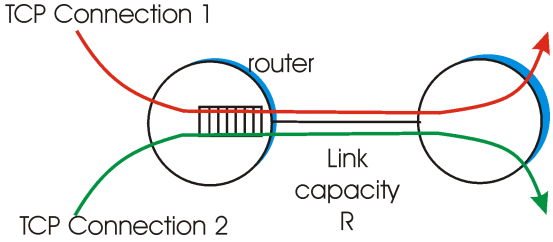
Beispiel für den zeitlichen Ablauf

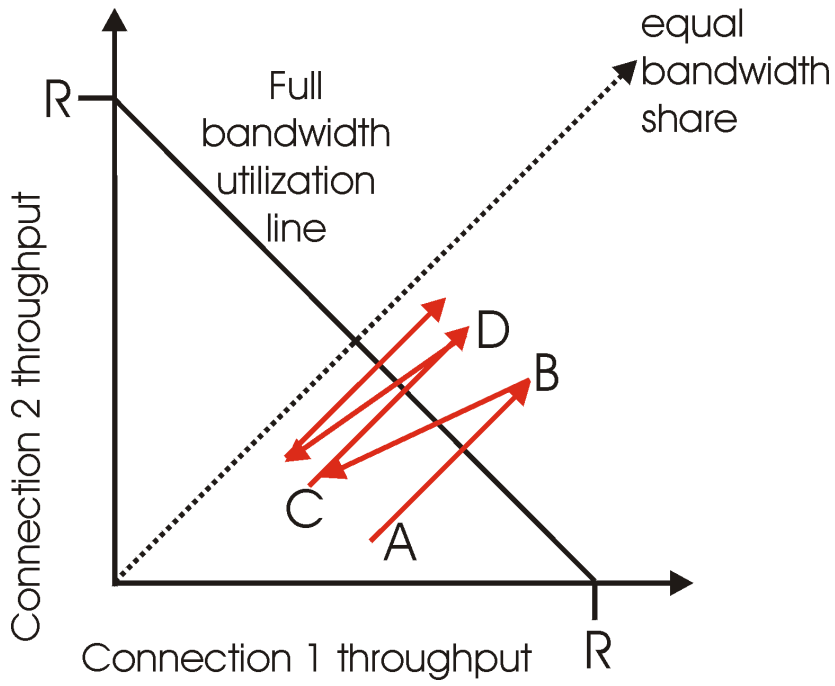


Durchsatz Wir vernachlässigen den Slow-Start am Anfang und nach Timeouts, betrachten also nur AIMD. Das CongestionWindow pendelt zwischen dem maximalen Wert W und der Hälfte $W/2$, die Bitrate also zwischen W/RTT und $W/2 \cdot \text{RTT}$. Im Mittel ergibt sich somit ein Durchsatz von

$$S = 3/4 \cdot \frac{W}{\text{RTT}}.$$

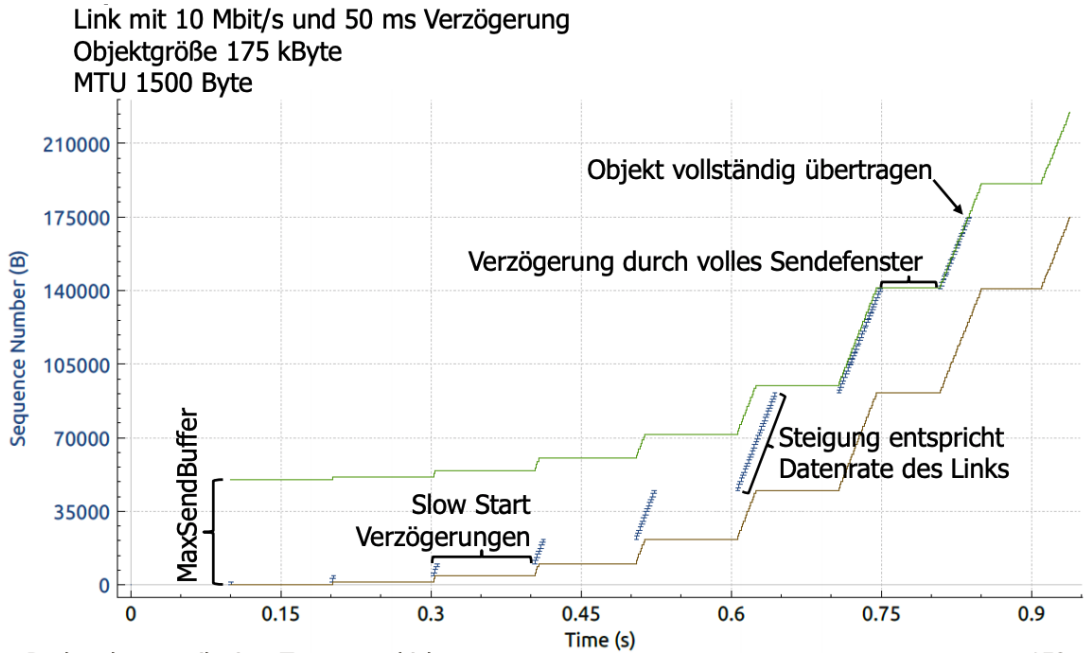
Fairness Wir wollen hier einen einfachen Fall mit zwei TCP-Verbindungen betrachten, welche sich einen Kanal der Bitrate R teilen. Wir nehmen an, dass die Verbindungen dieselbe MSS und RTT haben, große Datenmengen senden müssen und keine anderen TCP oder UDP Verbindungen diesen Kanal benutzen. Wir ignorieren auch die Slow-Start-Phase und ziehen nur AIMD in Betracht. Die Bitraten von Verbindung 1 und 2 sind in der Graphik dargestellt:





Falls TCP die Bitrate fair verteilt, sollte die realisierte Bitrate von Verbindung 1 und 2 auf der 45-Grad-Linie fallen. Angenommen die TCP Fenstergrößen sind so groß, so dass die Bitraten der Verbindungen auf Punkt A in der Abbildung fallen. Da die gemeinsame verwendete Bandbreite geringer als R ist wird kein Verlust auftreten und beide Verbindungen erhöhen ihre Fenstergröße um 1 pro RTT nach AIMD. Dementsprechend ist das Steigen von A nach B auf einer 45-Grad-Linie. Irgendwann (Punkt B) wird die gemeinsame Bandbreite größer als R sein und Paketverlust auftreten. Angenommen der Paketverlust passiert an Punkt B, so werden die Verbindungen ihre Fenster halbierend verkleinern. Die Bitraten sind deswegen an Punkt C angelangt, auf der Hälfte der Strecke zwischen Ursprung und Punkt B. Jetzt beginnt das ganze wieder von vorne, da Punkt C nicht größer R ist. Der Weg konvergiert wie man unschwer erkennt an die „Fairness“-Linie.

Beispiel für Slow Start und Sendefenster



3.3.5 Leistungsanalyse

Die Zeit zum Kopieren eines Objekts mit TCP hängt von der Objektgröße, Bitrate, Ausbreitungsverzögerung und den Verzögerungen durch Protokollmechanismen ab. Insbesondere der Slow-Start kann sich spürbar auswirken. Wir nehmen im Folgenden an, dass **keine** Bitfehler und Verluste, keine schwankenden Bitraten und Verzögerungen, ACK-Sendezeiten oder Bearbeitungszeiten auftreten und dass das Fenster der Flusskontrolle immer groß genug ist. Wir notieren:

- S — MSS in Bit
- R — Bitrate
- W — Fenstergröße in MSS
- O — Objektgröße in Bit
- RTT — Round Trip Time
- υ — Verzögerung

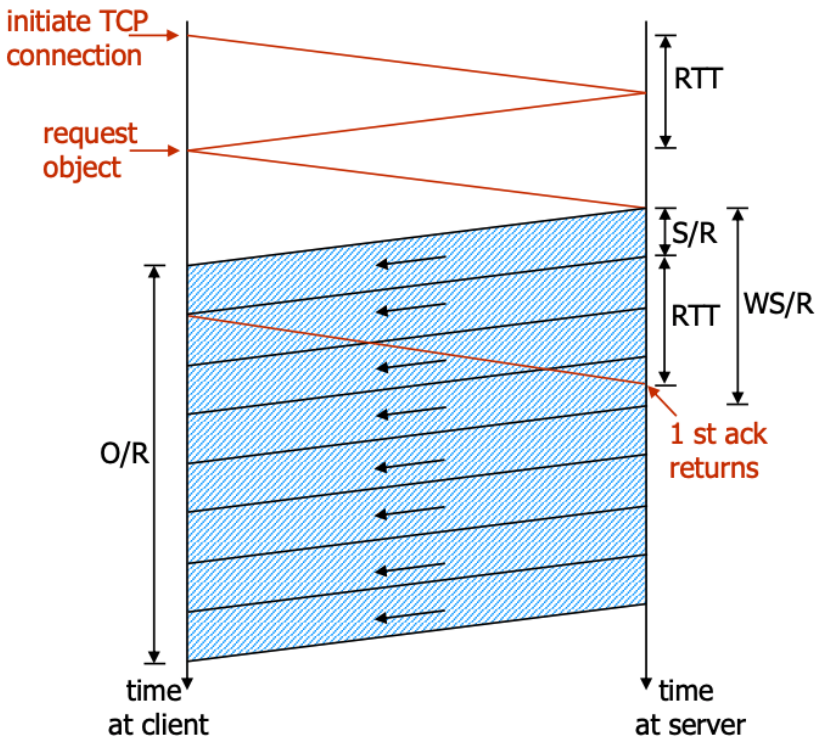
Wir analysieren zuerst für eine feste Fenstergröße und anschließend für wachsende Fenster ähnlich zu Slow-Start.

Leistungsanalyse für feste Fenstergrößen Falls das Fenster den Kanal füllt, also

$$\frac{W \cdot S}{R} > \text{RTT} + \frac{S}{R}$$

gilt, so gilt für die Verzögerung

$$\upsilon = 2 \cdot \text{RTT} + \frac{O}{R}.$$



Falls das Fenster den Kanal nicht füllt, also

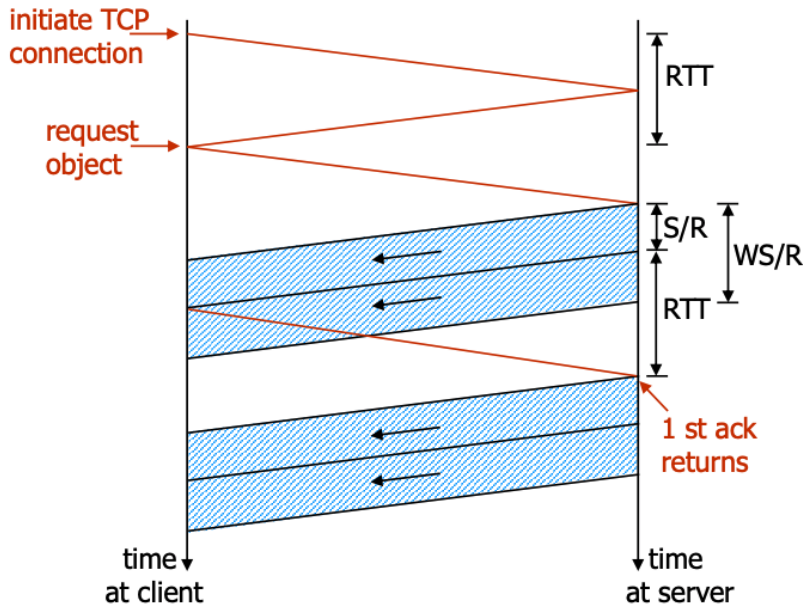
$$\frac{W \cdot S}{R} < \text{RTT} + \frac{S}{R}$$

gilt, so gilt für die Verzögerung

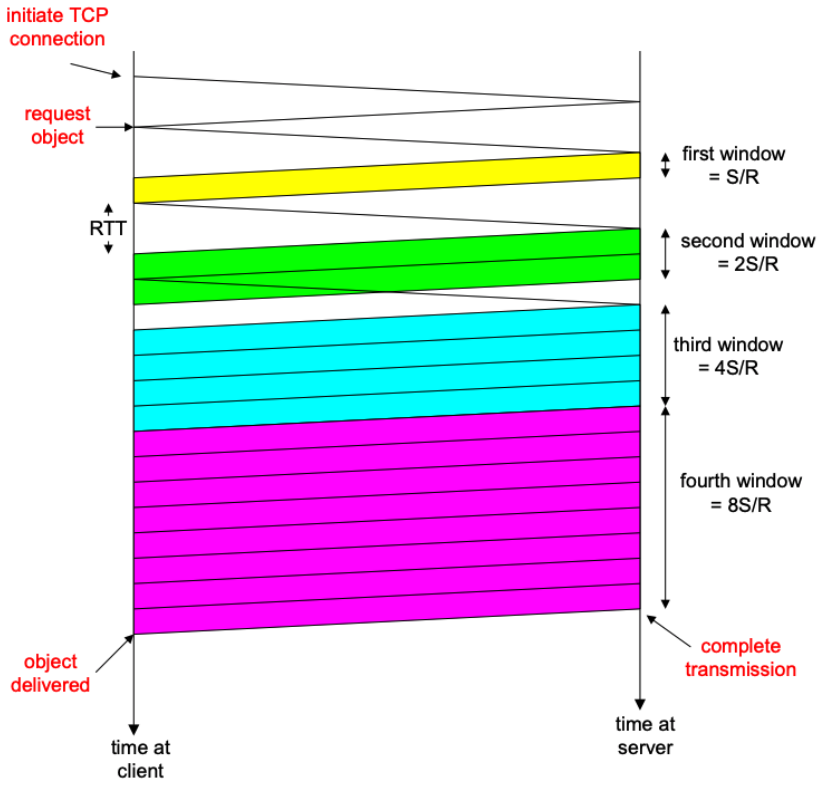
$$\upsilon = 2 \cdot \text{RTT} + \frac{O}{R} + (K-1) \cdot \left[\frac{S}{R} + \text{RTT} - \frac{W \cdot S}{R}\right],$$

wobei K die Anzahl an Fenstern angibt, die das Objekt benötigt:

$$K = \left\lceil \frac{O}{W \cdot S} \right\rceil$$



Leistungsanalyse für wachsende Fenstergrößen Nach den $2 \cdot RTT$ für den Verbindungsaufbau braucht man O/R für das Senden von $\frac{O}{S}$ Segmenten. Dabei haben wir K Fenster und P Slow-Start-Wartezeiten. Für $O/S = 15, K = 4$ und $P = 2$ ergibt sich:



Die Verzögerungszeit berechnet sich über $2 \cdot RTT + O/R +$ Slow-Start-Wartezeit. Die Slow-Start-Wartezeit im k-ten Fenster ergibt sich als

$$\max\left\{\frac{S}{R} + RTT - 2^{k-1} \cdot \frac{S}{R}, 0\right\}$$

durch die Sendezeit von $2^{k-1} \cdot S/R$ und die Zeit vom Sendebeginn bis zum Erhalt des ACKs von $S/R + RTT$. Damit ergibt sich:

$$\upsilon = \frac{O}{R} + 2 \cdot RTT + \sum_{k=1}^{P} \text{Wartezeit}_k = \frac{O}{R} + 2 \cdot RTT + P \cdot \left[RTT + \frac{S}{R} \right] - \left(2^P - 1 \right) \cdot \frac{S}{R}.$$

Sei K die Anzahl der Fenster, die für das Objekt benötigt werden, so gilt

$$K = \min \left\{ k \;\middle|\; 2^0 \cdot S + 2^1 \cdot S + \Lambda + 2^{k-1} \cdot S \geqslant O \right\} = \left\lceil \log_2 \left(\frac{O}{S} + 1 \right) \right\rceil.$$

Sei Q die Anzahl von Slow-Start-Wartezeiten bei einem unendlich großen Objekt, so gilt

$$Q = \max_k \left(\frac{S}{R} + RTT - 2^{k-1} \cdot \frac{S}{R} \geqslant 0 \right) = \left\lfloor \log_2 \left(1 + \frac{RTT}{S/R} \right) \right\rfloor + 1.$$

Somit ergibt sich für P:

$$P = \min \left\{ Q, K - 1 \right\}$$

Wir erhalten als Endergebnis

$$\upsilon = \frac{O}{R} + 2 \cdot RTT + P \cdot \left[RTT + \frac{S}{R} \right] - \left(2^P - 1 \right) \cdot \frac{S}{R}$$

, was ein Produkt aus P und RTT enthält. Wenn RTT also groß und/oder viele Slow-Start-Wartezeiten auftreten, kann die Verzögerung spürbar werden.

Beispiele

TCP: Leistungsanalyse, Beispiele

- Szenario I: S=536 Bytes, RTT=100 ms (intern.), O=100 KB (lang)

R	*O/R*	*Verzögerung*
28 Kbps	28.6 s	28.9 s
100 Kbps	8 s	8.4 s
1 Mbps	800 ms	1.5 s
10 Mbps	80 ms	0.98 s

- Szenario II: S=536 Bytes, RTT=100 ms, O=5 KB (kurz)

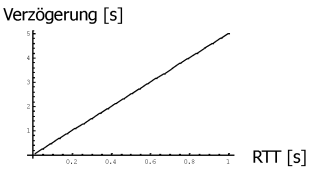
R	*O/R*	*Verzögerung*
28 Kbps	1.43 s	1.73 s
100 Kbps	0.48 s	0.757 s
1 Mbps	40 ms	0.52 s
10 Mbps	4 ms	0.50 s

- Szenario III: S=536 Bytes, RTT=1 s (Überlast), O=5 KB

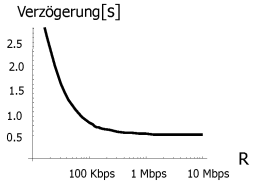
R	*O/R*	*Verzögerung*
28 Kbps	1.43 s	5.8 s
100 Kbps	0.48 s	5.2 s
1 Mbps	40 ms	5.0 s
10 Mbps	4 ms	5.0 s

TCP: Leistungsanalyse, Beispiele

- S=536 Bytes, O=5 KB, R=1 Mbps, RTT wird verändert:



- S=536 Bytes, O=5 KB, RTT=100 ms, R wird verändert: von 10 Kbps bis 10 Mbps (logarithmische Skalierung):



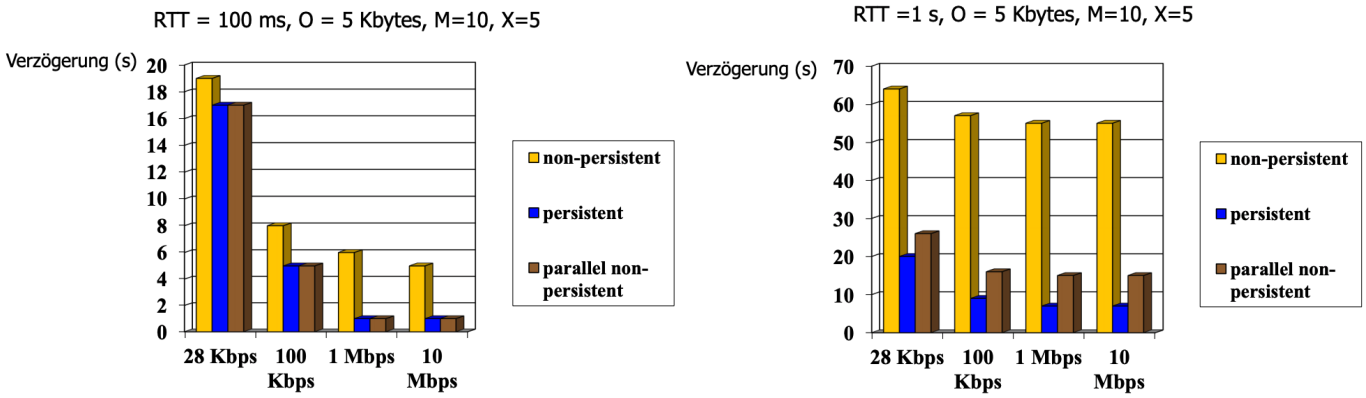
Antwortzeiten bei Web-Seiten

TCP: Leistungsanalyse, Antwortzeiten bei Web-Seiten

- Web-Seite mit:
  - 1 Basis-HTML-Seite (O Bits)
  - M Bilder (auch jeweils O Bits)
- nicht-persistentes HTTP:
  - M+1 sequentielle TCP-Verbindungen
  - Antwortzeit = (M+1)O/R + (M+1)2RTT + Slow-Start-Wartezeiten
- persistentes HTTP:
  - 2 RTT für Basis-Seite
  - 1 RTT für M Bilder
  - Antwortzeit = (M+1)O/R + 3RTT + Slow-Start-Wartezeiten
- nicht-persistentes HTTP mit X parallelen Verbindungen
  - Annahme: M/X ist ganze Zahl
  - 1 TCP-Verbindung für Basis-Seite
  - M/X Mengen von parallelen Verbindungen für Bilder
  - Antwortzeit = (M+1)O/R + (M/X+1)2RTT + Slow-Start-Wartezeiten

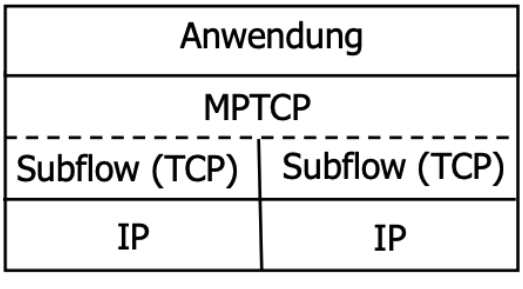
Rechnerkommunikation, Transportschicht 170

Wir stellen fest, dass für geringe Bitraten die Antwortzeit durch die Übertragungszeit dominiert wird. Persistente Verbindungen ergeben dabei nur einen geringen Vorteil gegenüber parallelen Verbindungen. Für große RTTs wird die Antwortzeit durch Slow-Start-Wartezeiten dominiert, persistente Verbindungen ergeben hierbei insbesondere für große Bitraten-Verzögerungs-Produkte Vorteile.



3.3.6 Multipath TCP

Multipath TCP ist eine Erweiterung von TCP um mehrere Pfade für eine Verbindung nutzen zu können. Die Hauptziele sind eine Erhöhung des Durchsatzes, bessere Ressourcenauslastung, bessere Resilienz und eine Abwärtskompatibilität. Dabei erscheint MPTCP als eine Zwischenschicht in der Transportschicht, welche **transparent** arbeitet und wie normales TCP erscheint. Es hat eine zusätzliche API zur Nutzung der Funktionen von MPTCP und TCP Subflows. Zudem werden die Optionsfelder für TCP mit ...

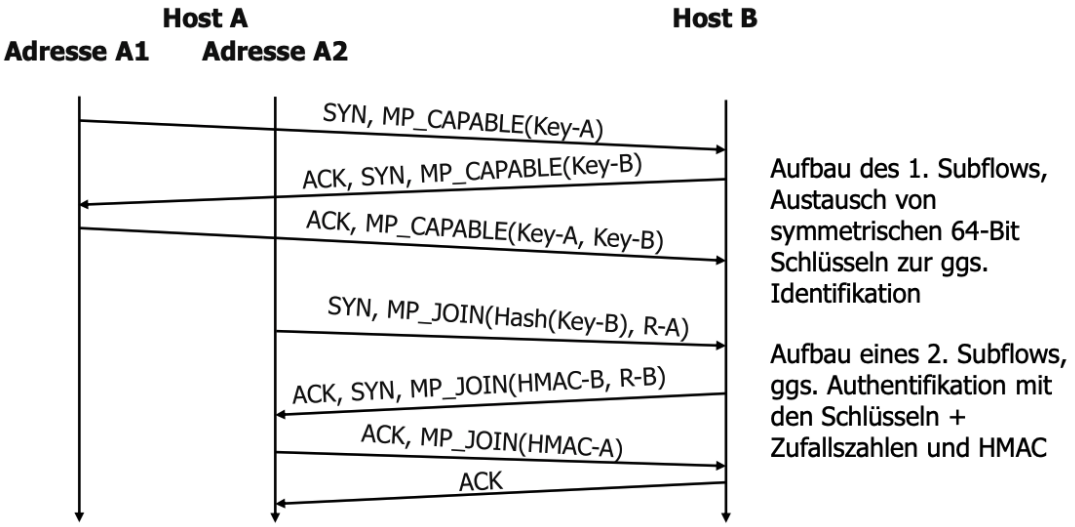


... `MP_CAPABLE` zum Aufbau eines initialen Subflows,

... `MP_JOIN` zum Aufbau weiterer Subflows und

... übergeordneten 64-Bit Data Sequence Numbers genutzt.

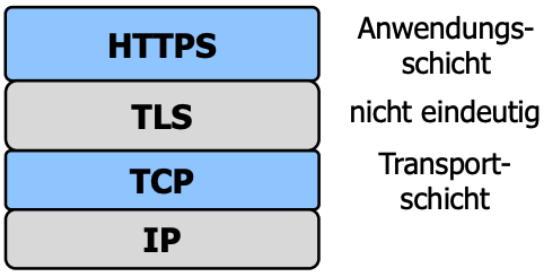
Ein beispielhafter Aufbau von zwei Subflows sei gegeben:



3.4 TLS

3.4.1 TLS 1.2

Ursprünglich aus dem Secure Socket Layer entstanden baut TLS eine Sicherung der Transportschicht und bietet dabei eine allgemeine API für Anwendungen. Es wird insbesondere von HTTP (HTTPS) genutzt und es gibt auch Versionen für UDP-Verbindungen. Nach dem TCP-Verbindungsaufbau folgt ...

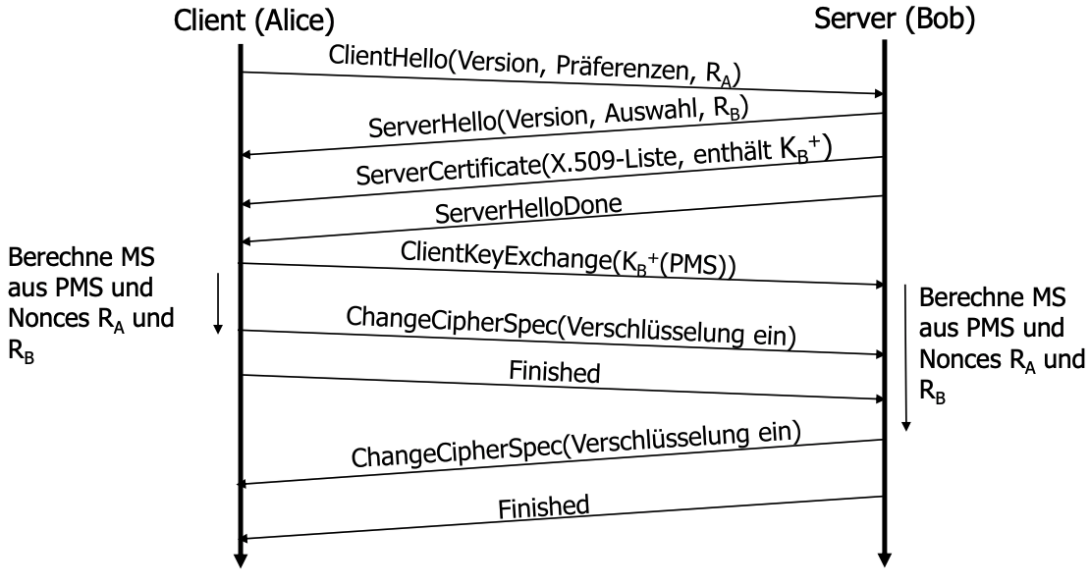


... ein **Handshake**. In diesem werden Algorithmen (Verschlüsselung, MAC) ausgehandelt, Nonces, Zertifikate und Premaster Secrets (PMS) ausgetauscht, das Master Secret (MS) mit zwei symmetrischen Sitzungsschlüsseln, zwei MAC-Schlüsseln und zwei Initialisierungsvektoren für CBC abgeleitet.

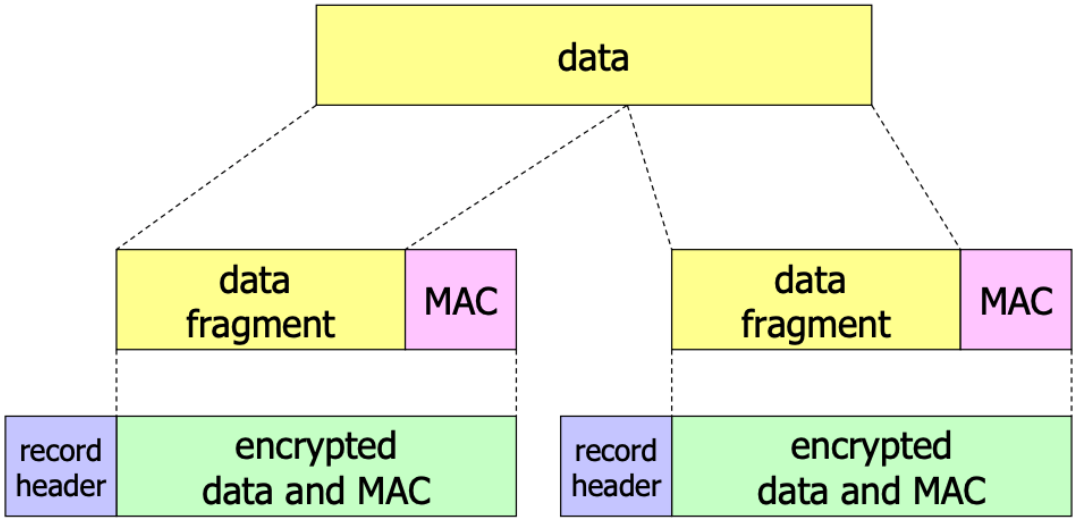
... der Datenaustausch über das **Record-Protokoll**. Hier findet die Fragmentierung, Kompression, Verschlüsselung sowie das Hinzufügen der MAC und eines Headers statt.

... das **Schließen der Verbindung** über ein spezielles Header-Feld authentifiziert durch MAC.

Beispiel für Handshake



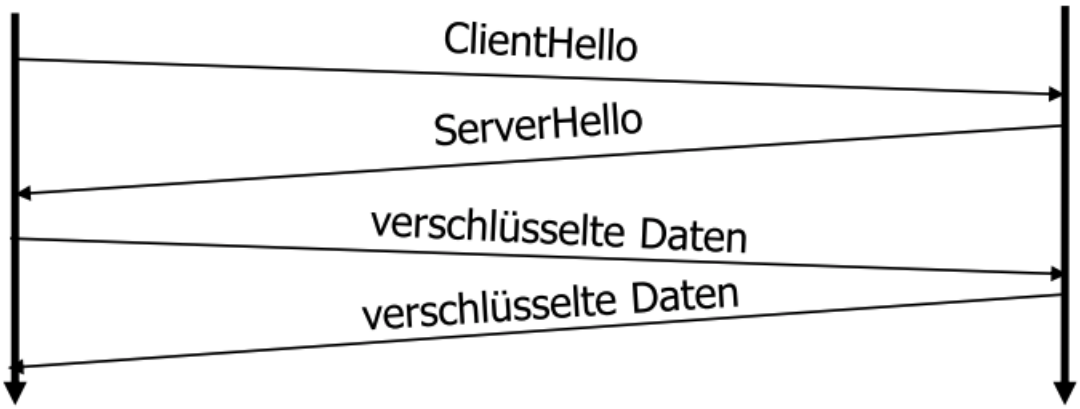
Record Protocol



Dabei stehen im `Record Header` der Type, die Version und Länge und die MAC enthält die Sequenznummer und den Schlüssel.

3.4.2 TLS 1.3

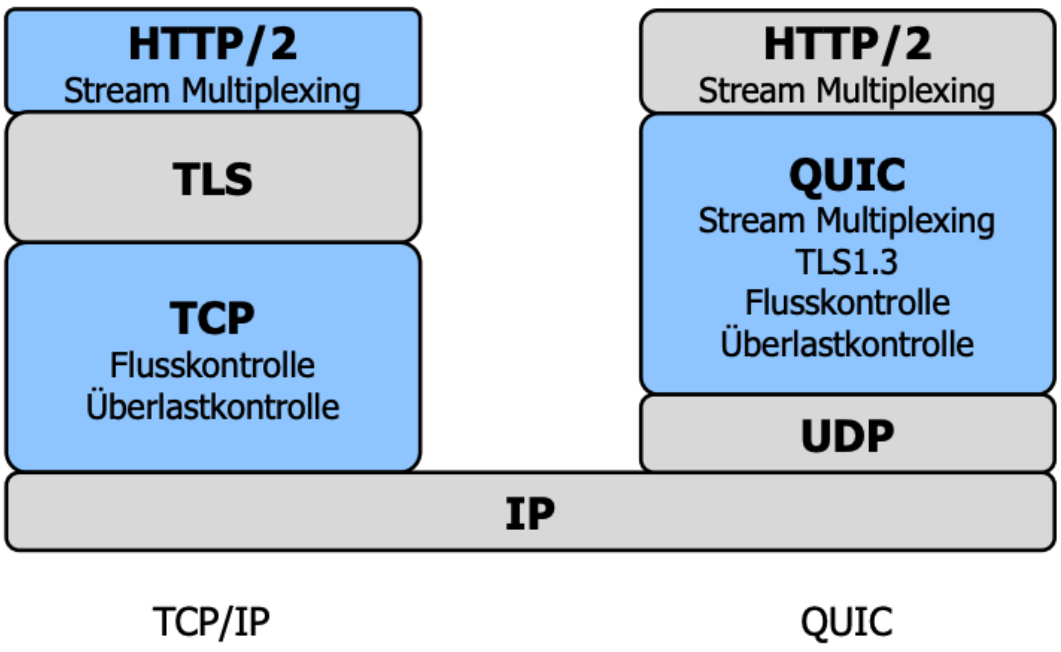
TLS 1.3 ist TLS 1.2 mit verbesserten Algorithmen und kürzeren Handshakes. So gibt es hier nur **Zwei-Wege-Handshakes** zum Austausch aller benötigten Informationen und anschließend folgt bereits das verschlüsselte Senden von Daten. Es basiert auf dem *Diffie-Hellman-Schlüsselaustausch* und/oder einem vorher bereits ausgetauschen Schlüssel (*Pre-Shared Key, PSK*). Durch den **0-RTT Mode** können Daten bei Wiederaufnahme (*resumption*) ohne Handshake sofort verschlüsselt gesendet werden.



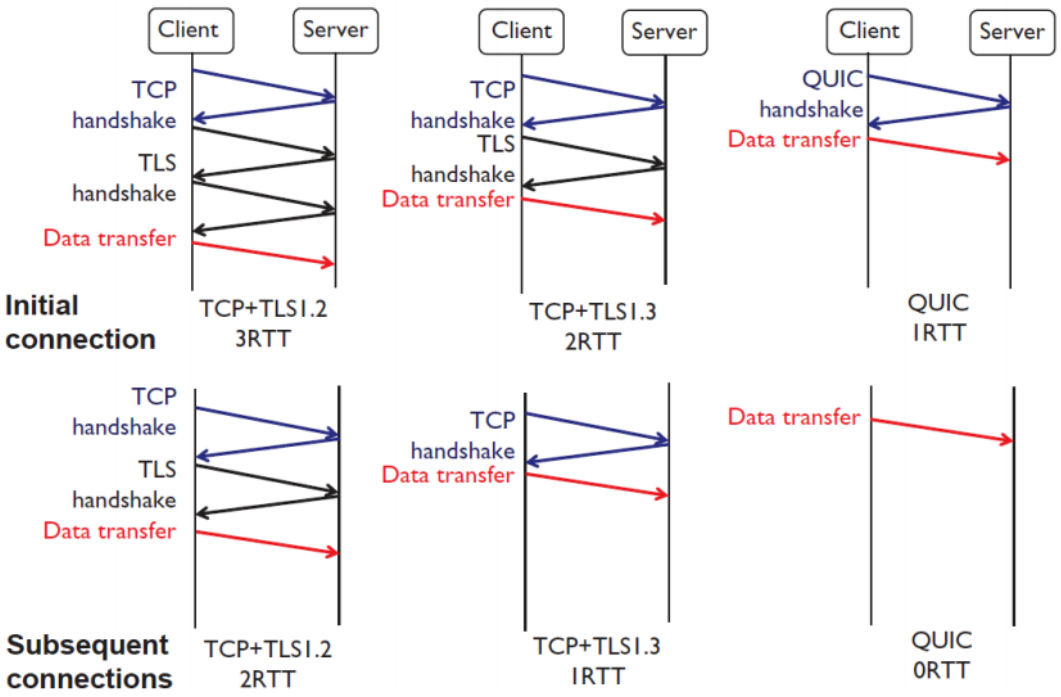
3.5 QUIC

Das **Quick UDP Internet Connections (QUIC)** ist ein zuverlässiges Transportprotokoll, welches von Google entwickelt wurde. Es befindet sich momentan noch in der Standardisierung bei der IETF. Es verwendet **Stream Multiplexing**, also eine verschränkte Übertragung mehrer Objekte und ist ein Gegenstück zu HTTP/2. Es basiert auf UDP, um nicht von Firewalls oder anderen Middleboxes geblockt zu werden und hat weitestgehend verschlüsselte Header, wodurch keine Manipulationsmöglichkeiten durch Middleboxes zugelassen werden. Es verwendet Forward Error Correction und hat die Protokollimplementierung an die Applikation und nicht das Betriebssytem ausgelagert. Ein Multipath QUIC ist ebenso in Vorbereitung.

QUIC im Schichtenmodell



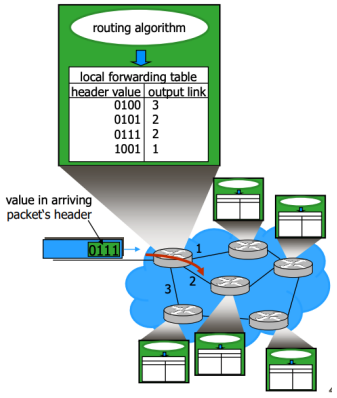
QUIC im Vergleich zu TCP/TLS


Client
Server
TCP
handshake
TLS
handshake
Data transfer
Initial connection
TCP+TLS1.2
3RTT
Client
Server
TCP
handshake
TLS
handshake
Data transfer
TCP+TLS1.3
2RTT
Client
Server
QUIC
handshake
Data transfer
QUIC
1RTT
TCP
handshake
TLS
handshake
Data transfer
Subsequent connections
TCP+TLS1.2
2RTT
TCP
handshake
Data transfer
TCP+TLS1.3
1RTT
Data transfer
QUIC
0RTT

KAPITEL 4

NETZWERKSCHICHT

Die Aufgabe der Netzwerkschicht ist die Kommunikation zwischen Endsystemen, welche über verschiedene Netze verbunden sind. Dabei senden und empfangen die Endsysteme während die Vermittlungseinheiten weiterleiten. Es gibt zwei wichtige Aufgaben die **Weiterleitung (Forwarding)** bei der die Vermittlungseinheit Dateneinheiten auf einer Schnittstelle empfängt und an eine andere weiterleitet, sowie die **Wegewahl (Routing)** ein Verfahren, mit welchen Vermittlungseinheiten entscheiden über welchen Weg Dateneinheiten gesendet werden. Das Zusammenspiel von Weiterleitung und Wegewahl wird schön in folgender Abbildung dargestellt:



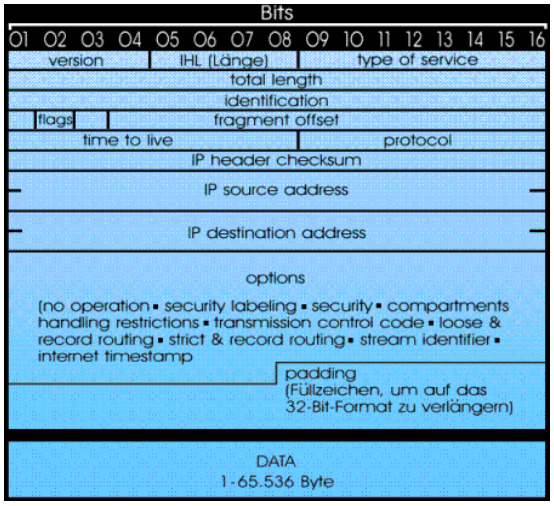
Wir unterscheiden zwei Vermittlungsarten:

Datagrammbasierte Paketvermittlung Hier trägt jedes Datagramm eine globale Adresse, wleche von Routern zur Weiterleitung verwendet wird. Die Vermittlungseinheit beim IP ist der Router. Wohlgemerkt findet hier keine Bereitstellung von Dienstmerkmalen wie Fehlerkontrolle, Bewahrung der Reihenfolge, Unterscheidung zwischen verbindungsorientierten und verbindungslosen Leitungen, Fluss- und Überlastkontrollen, Garantien für die Dienstgüte oder Informationssicherheit statt.

Virtuelle Leitungsvermittlung Jede Dateneinheit[1] erhält eine lokale Kennung, welche beim Weiterleiten von jede Vermittlungseinheit verändert wird. Dies ermöglicht den Aufbau einer virtuellen Leitung und gegebenenfalls die Bereitstellung von Dienstgütemerkmalen. Dies wird beispielsweise bei MPLS verwendet.

4.1 IP

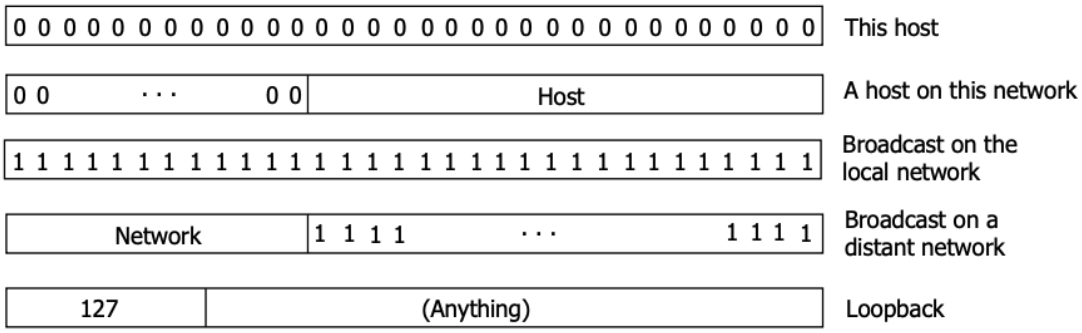
[1] Für diese werden in verschiedenen Protokollen unterschiedliche Namen verwendet



`ver` gibt die Version an, hat 4 Bit, eine 4 steht hier für IPv4. `HL` gibt die Länge des IP Headers *in 32-Bit-Worten* an, ebenfalls mit 4 Bit. `TOS` steht für *type of service* und stellt die ehemaligen `DS/ECN`-Felder dar. `length` gibt die Gesamtlänge des Datagramms in Byte an. **Wichtig:** Das Datagramm kann maximal $2^{16} - 1$ Byte lang sein. `identifier`, `flgs` und `fragment offset` sind für die Fragmentierung von Datagrammen wichtig. Da manche Hosts Datagramme der Länge 65535 Byte nicht akzeptieren gibt es eine Mindestlänge von 537 Byte. Damit trotzdem längere Datagramme gesendet werden können muss hier fragmentiert werden. `TTL` gibt die *time to live* an, ein Hoplimit, welches jeder Router dekrementiert. `upper layer` gibt das höhere Protokoll an, `options` verschiedene Optionen.

4.1.1 Klassenbasierte Adressierung

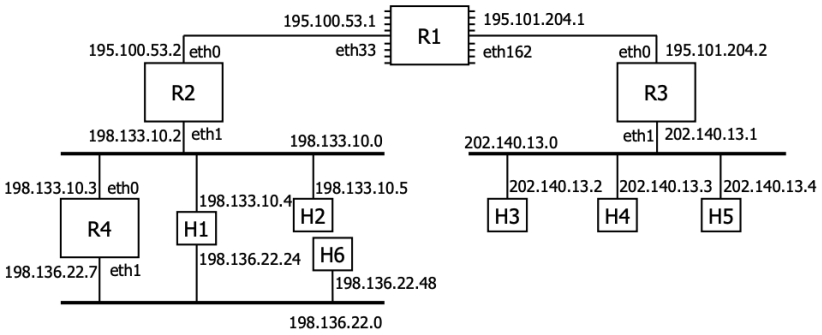
Die IPv4-Adresse kennzeichnet eine Schnittstelle eines Hosts oder eines Routers. Hosts mit mehreren Schnittstellen (*multi-homed*) und Router benötigen dabei mehrere IP-Adressen. Die IPv4-Adresse besteht aus 32 Bit (4 Byte) und lässt sich in einen **Netzwerkteil** und **Hostteil** aufteilen. Die zentrale Kontrolle wird von der *Internet Corporation for Assigned Names and Numbers (ICANN)* geleistet. Autorisierte *address registries* vergeben dann IPs an ISPs und diese die IPs dann wieder weiter. (American Registry for Internet Numbers (ARIN), Reseaux IP Europeens (RIPE), ...)
Wichtig: Es wird die **Dotted-Decimal-Schreibweise** verwendet, sprich eine IPv4-Adresse wird als $d_1.d_2.d_3.d_4$ dargestellt, wobei d_j die Dezimaldarstellung des j-ten Bytes ist.
Man kann nun klassenbasierte Adressen verwenden, ähnlich wie in der Tabelle auf der nächsten Seite darstellt. Dabei stehen Nullen in der Netzmaske für die Bits vom Host und 255 für die Netzwerkbits. Es gibt einige besondere Adressen:



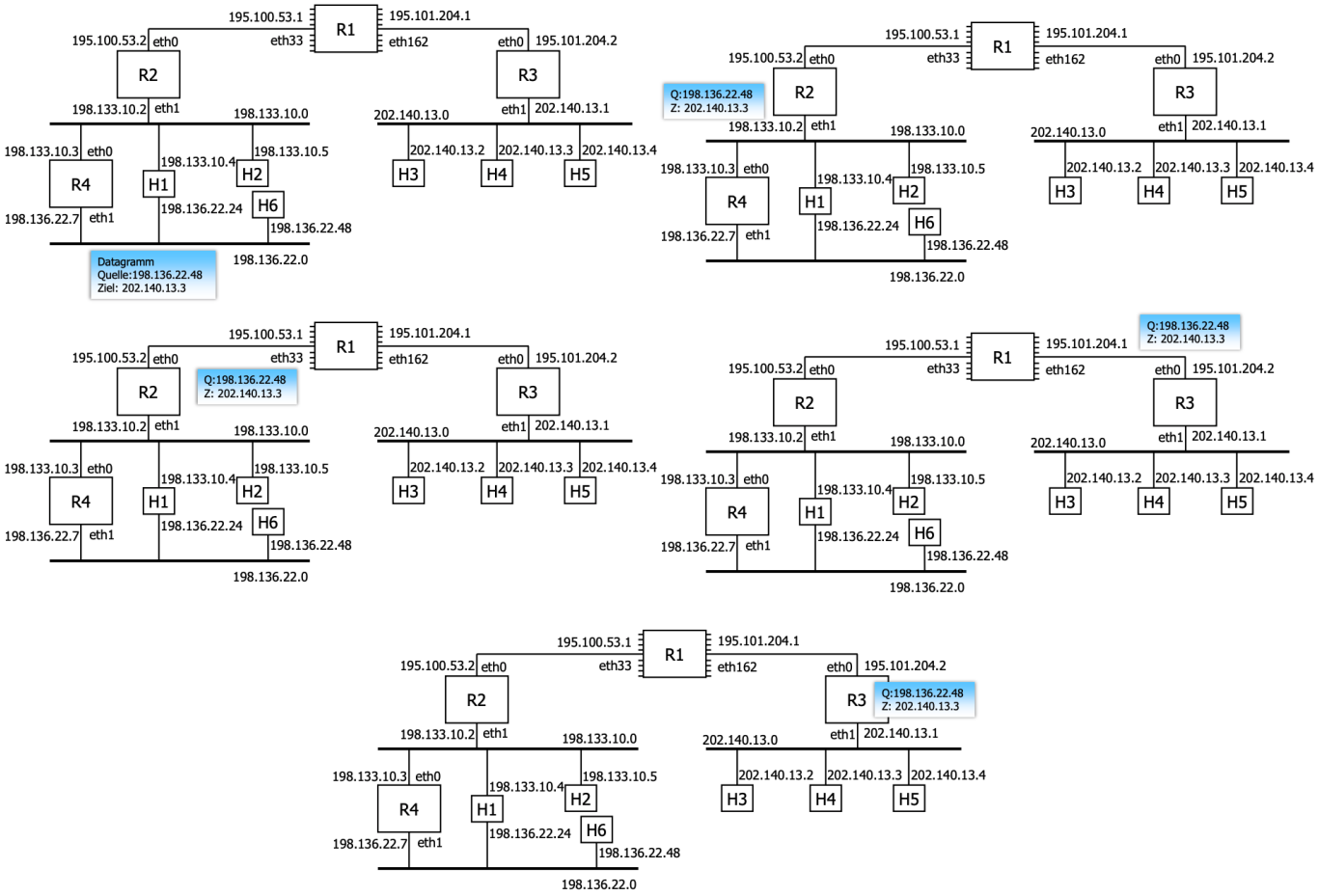
Beispiel für eine klassenbasierte Adressierung Angenommen wir haben ein Netzwerk mit vier Routern R1 bis R4 und 6 Hosts H1 bis H6, welche wiederum die Schnittstellen eth0, eth1, ... anbieten. Dann könnten wir eine Netzwerktopologie haben mit drei Klasse-C-Netzwerken: `198.133.10.0`, `198.136.22.0` und `202.140.13.0`. Jedes Netzwerk ohne Router, also ein Netzwerk was beispielsweise durch Ethernet oder eine Switch-Struktur realisiert wird, wird als Balken abstrahiert.

Netzklasse	Präfix	Adressbereich	Netzmaske	Netzlänge (mit Präfix)	Netzlänge (ohne Präfix)	Hostlänge	Netze	Hosts pro Netz	CIDR Suffix Entsprechung
Klasse A	0...	0.0.0.0 – 127.255.255.255	255.0.0.0	8 Bit	7 Bit	24 Bit	128	16.777.214	/8
Klasse B	10...	128.0.0.0 – 191.255.255.255	255.255.0.0	16 Bit	14 Bit	16 Bit	16.384	65.534	/16
Klasse C	110...	192.0.0.0 – 223.255.255.255	255.255.255.0	24 Bit	21 Bit	8 Bit	2.097.152	254	/24
Klasse D	1110...	224.0.0.0 – 239.255.255.255	Verwendung für Multicast-Anwendungen						
Klasse E	1111...	240.0.0.0 – 255.255.255.255	reserviert (für zukünftige Zwecke)						

Tabelle — Übersicht der Netzklassen



Ein IP-Datagramm findet nun seinen Weg von H6 nach H4 über verschiedene Weiterleitungstabellen:



Die Weiterleitungstabellen sehen so aus:

H6

Ziel	n. HOP	Flag	S.s.
198.136.22.0	—		eth0
default	198.136.22.7	G	eth0

R2

Ziel	n. HOP	Flag	S.s.
195.100.53.0	—		eth0
198.133.10.0	—		eth1
198.136.22.0	198.133.10.3	G	eth1
default	195.100.53.1	G	eth0

R4

Ziel	n. HOP	Flag	S.s.
198.136.22.0	—		eth1
198.133.10.0	—		eth0
default	198.133.10.2	G	eth0

R1

Ziel	n. HOP	Flag	S.s.
198.136.22.0	195.100.53.2	G	eth33
198.133.10.0	195.100.53.2	G	eth33
195.100.53.0	—		eth33
195.100.53.0	—		eth162
202.140.13.0	195.101.204.2	G	eth162

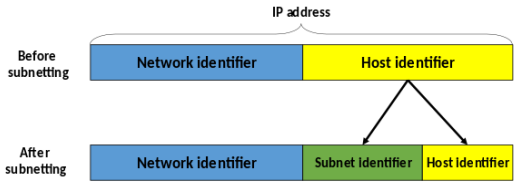
R3

Ziel	nächster HOP	Flag	Schnittstelle
`195.101.204.0`	—		eth0
`202.140.13.0`	—		eth1
`default`	`195.101.204.1`	G	eth0

Vor- und Nachteile einer klassenbasierten Adressierung **Vorteil:** Es gibt **selbstidentifizierende Adressen**, also Adressen bei denen an den ersten Bits erkannt wird, um welche Klasse es sich handelt.
Vorteil: Weiterleitungstabellen benötigen nur einen Netzwerkteil der Adresse und können somit klein gehalten werden.
Nachteil: Es gibt eine feste Zuordnung von Netzwerken. Wenn ein Rechner also „umzieht“, so muss seine IP-Adresse angepasst werden.
Nachteil: C-Netze erlauben nur wenige Hosts, B-Netze hingegen sehr viele Hosts (8 zu 16 Bit). Dies führt unweigerlich zu einer Verschwendung. Größere Organisationen bemühen sich um B-Netze, nutzen diese dann aber oftmals nicht aus.
Nachteil: Der Adressraum ist zu klein. 2011 hat ICANN die letzten verbleibende IPv4-Adressen ausgegeben. Die Lösung hierzu ist IPv6, ein „*Pflaster*“ ist NAT.

4.1.2 Klassenlose Adressierung und Subnetze

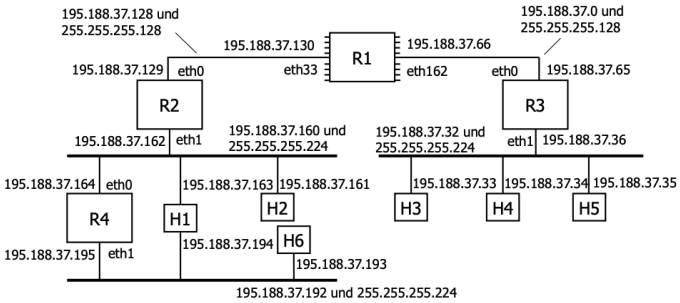
Subnetze Ein Subnetz ist eine logische Unterteilung eines IP-Netzwerks. Dabei wird der Hostanteil weiter in ein Subnetz von variabler Länge und dem Hostanteil aufgeteilt. Entfernte Router benötigen dann nur den weiter klassenbasierten Netzwerkteil.



Die **Subnetzmaske** ist die Bitmaske, die mit einem bitweisen UND auf die IP-Adresse angewandt den Routinganteil erhält. Sie gibt damit mit einem `1` an, welche Bit einer IP-Adresse für den Netzwerkteil und Subnetzteil stehen. Auch hierfür wird die **Dotted-Decimal-Schreibweise** verwendet.

- Beispiel für Subnetz
  - Organisation hat Klasse-B-Adresse (also 16 Bits Hostteil) mit Netzwerkteil: 150.100.0.0
  - es werden Subnetze mit jeweils maximal 100 Hosts erzeugt
  - hierfür reichen jeweils 7 Bits
  - also 16-7=9 Bits für Subnetzteil
  - z.B. IP-Adresse 150.100.12.176 und Maske 255.255.255.128
  - binär = 10010110 01100100 00001100 10110000
  - Maske = 11111111 11111111 11111111 10000000
  - AND = 10010110 01100100 00001100 10000000
  - Subnetzadresse = 150.100.12.128

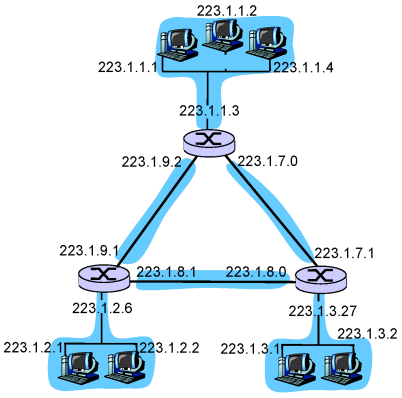
Das obige Beispiel könnte auch mit mehreren Subnetzen realisiert worden sein, hier als Beispiel mit nur einem Klasse-C-Netzwerk `195.188.37.0`:



Wichtig: Bei *Legacy-Equipment* ist es wichtig darauf zu achten, dass das *subnet zero* (Subnetzbitgruppe ist 0) und *all-ones subnet* (Subnetzbitgruppe besteht nur aus 1èn) **nicht** zu verwenden.
Der Hostwert, welcher nur aus Nullen besteht, (*all-zeros host value*) ist reserviert für die Netzwerkadresse des Subnetzes und der Hostwert, welcher nur aus Einsen besteht, für die Broadcastadresse.

CIDR (Classless Inter-Domain Routing) Neben dem **Subnetting** gibt es auch noch das **Supernetting**, bei dem man aufeinanderfolgende klassenbasierte Netze zusammenfasst. Man notiert diese dann als `Präfix/X`, wobei `Präfix` für einen IP-Adressanteil und `X` die Länge des Netzwerkteils steht. Im Ergebnis ist das Subnetz durch diese Schreibweise gegeben. Diese Adressen werden dann als Einträge in die Weiterleitungstabelle geschrieben. **Vorteil:** Aufhebung der starren Zuordnung von klassenbasierten Adressen. **Nachteil:** Es kommt zu größeren Tabellen. Router müssen dabei ein **Longest-Prefix-Match** durchführen. Sie vergleichen die Adresse mit Übereinstimmungen in der Weiterleitungstabelle und wählen diejenige, bei denen die meisten Bits übereinstimmen.

Subnetze topologisch Man kann sich Subnetze als zusammenhängende Bereiche ohne Router vorstellen, wenn man eine gedankliche Abtrennung der Schnittstellen von Routern vornimmt.



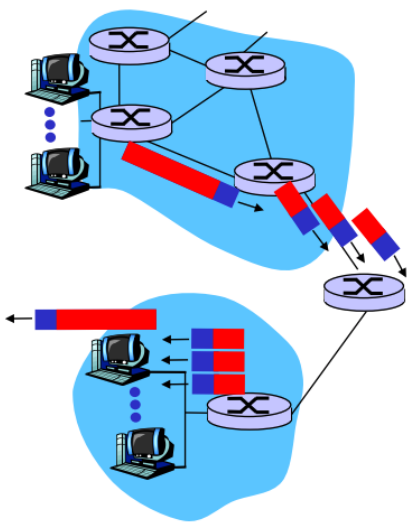
Merkhilfe für Subnetting mit IPv4

	Adressen	Hosts	Netmask	Relativ zur Klassen A, B, C
/30	4	2	255.255.255.252	1/64 C
/29	8	6	255.255.255.248	1/32 C
/28	16	14	255.255.255.240	1/16 C
/27	32	30	255.255.255.224	1/8 C
/26	64	62	255.255.255.192	1/4 C
/25	128	126	255.255.255.128	1/2 C

	Adressen	Hosts	Netmask	Relativ zur Klassen A, B, C
/24	256	254	255.255.255.0	1 C
/23	512	510	255.255.254.0	2 C
/22	1024	1022	255.255.252.0	4 C
/21	2048	2046	255.255.248.0	8 C
/20	4096	4094	255.255.240.0	16 C
/19	8192	8190	255.255.224.0	32 C
/18	16384	16382	255.255.192.0	64 C
/17	32768	32766	255.255.128.0	128 C
/16	65536	65534	255.255.0.0	256 C = B
/15	2^{17}	65534	255.254.0.0	2 B
/14	2^{18}	65534	255.252.0.0	4 B
/13	2^{19}	65534	255.248.0.0	8 B
/12	2^{20}	65534	255.240.0.0	16 B
/11	2^{21}	65534	255.224.0.0	32 B
/10	2^{22}	65534	255.192.0.0	64 B
/9	2^{23}	65534	255.128.0.0	128 B
/8	2^{24}	65534	255.0.0.0	256 B = A
/7	2^{25}	65534	254.0.0.0	2 A
/6	2^{26}	65534	252.0.0.0	4 A
/5	2^{27}	65534	248.0.0.0	8 A
/4	2^{28}	65534	240.0.0.0	16 A
/3	2^{29}	65534	224.0.0.0	32 A
/2	2^{30}	65534	192.0.0.0	64 A
/1	2^{31}	65534	128.0.0.0	128 A
/0	2^{32}	$2^{32}-2$	0.0.0.0	256 A

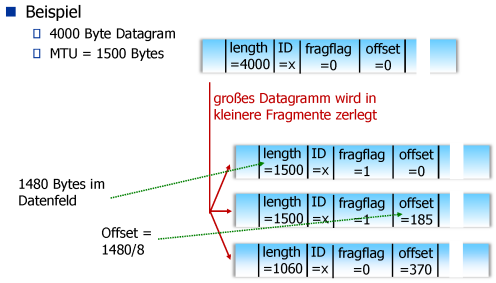
4.1.3 Fragmentierung

Wenn Verbindungen unterwegs eine kleinere **MTU (Maximum Transmission Unit)** erfordern, wird das Datagramm in Fragmente zerlegt und als kleinere Datagramme weitergeleitet. Unter Umständen kann sich dies auf einem Weg mehrmals wiederholen. Dazu gibt es verschiedene IP-Header-Felder:



- `identifier`: Eine Kennzeichnung für zusammengehörige Fragmente
- `flag`: Ist das Datagramm ein Fragment?
- `offset`: Position der Daten des Fragments bei `offset` ·8.

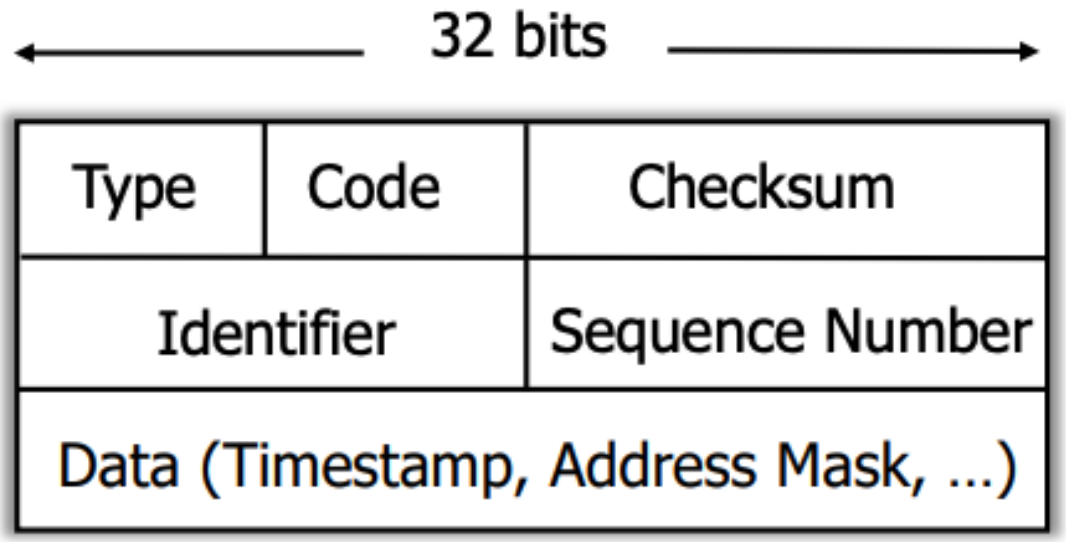
Erst am Ziel werden die Fragmente wieder zusammengesetzt. (*reassembly*)



4.1.4 ICMP

Typ	Code	Bedeutung
0	0	Ping, Echo Antwort
3	0	Netzwerk nicht erreichbar
3	1	Host nicht erreichbar
3	2	Ziel-Protokoll nicht verfügbar
3	3	Ziel-Port nicht erreichbar
3	6	Netzwerk unbekannt
3	7	Ziel-Host unbekannt
4	0	Überlastkontrolle, ungenutzt
8	0	Ping, Echo-Anfrage
9	0	Routen-Bekanntmachung
10	0	Router-Discovery
11	0	Trace-Route
12	0	Schlechter IP-Header

Das *Internet Control Message Protocol* ist für Kontrollnachrichten von Routern an andere Router und Hosts. Beispielsweise findet eine Benachrichtigung über Fehler (unerreichbare Adresse, maximale Zahl von Hops erreicht, ...) oder über den Ping über ICMP statt. Eine Anwendung in `traceroute`, hier werden IP-Pakete mit jeweils inkrementierender TTL geschrieben. Die Router verwerfen diese und antworten dann. Das Format von ICMP ist stark vom Typ abhänging (zweite Zeile der Abbildung). ICMP wird in IP-Datagrammen befördert, bei Bezug auf IP-Pakete enthält der Datenteil den IP-Header und die ersten 64 Bit des Pakets.

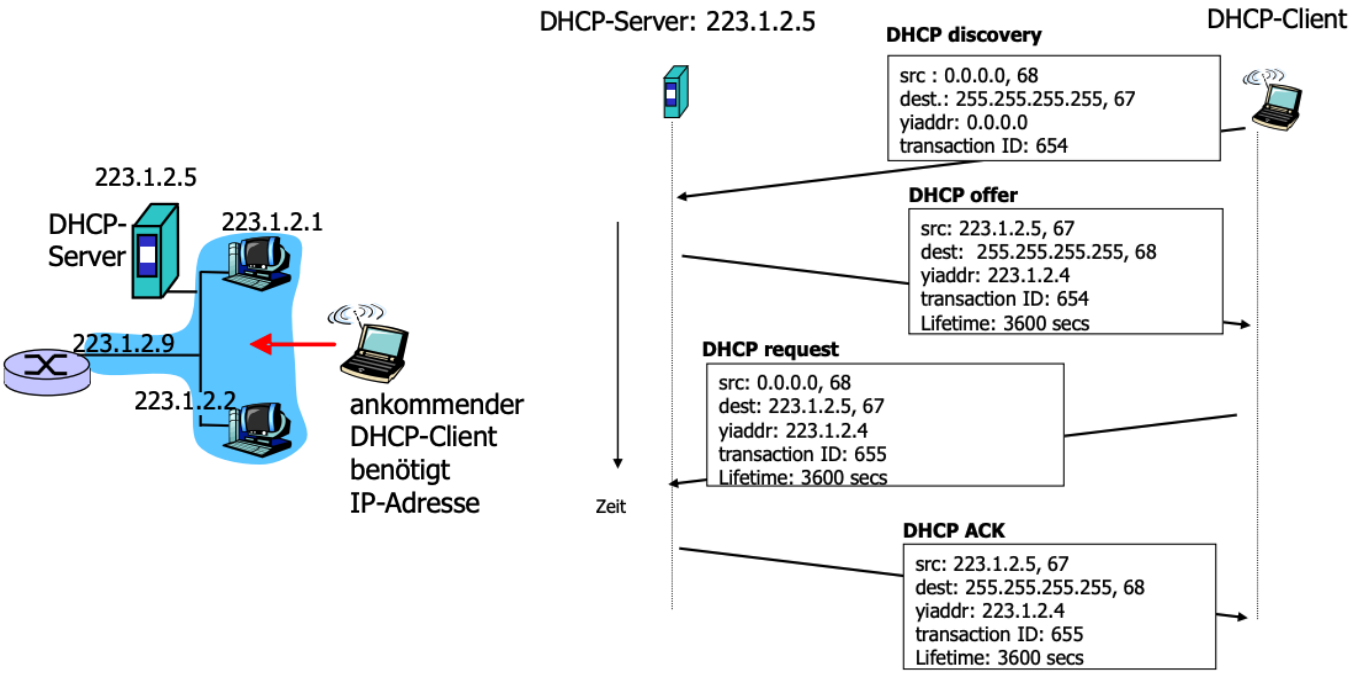


4.1.5 DHCP

Das *Dynamic Host Configuration Protocol (DHCP)* ist ein Client-Server-Protokoll zum automatischen Bezug einer IP-Adresse und weiterer Parameter (*Router, DNS-Server, ...*). Dazu muss es einen **DHCP-Server** im Subnetz (oder ein DHCP-Relay, welches den Server kennt) geben. Dann erfolgt die IP-Adressvergabe in vier Schritten mit UDP-Paketen:

- **DHCP *server discovery*** von `0.0.0.0` an `255.255.255.255` (Broadcast)
- **DHCP *server offers*** von der IP-Adresse des DHCP-Servers an `255.255.255.255` (Broadcast)
- **DHCP *request*** von `0.0.0.0` an die IP-Adresse des ausgewählten DHCP-Servers (Unicast)
- **DHCP *ACK*** vom ausgewählten DHCP-Server an `255.255.255.255`, enthält die IP-Adresse `yiaddr` und gegebenenfalls weitere Parameter.

Gegebenenfalls findet hierbei noch eine Absicherung durch ARP-Requests statt. Außerdem werden alle Nachrichten — bis auch den *DHCP request* — per Broadcast im Subnetz auf der Ebene der Sicherungsschicht gesendet. Eine Beispielkonfiguration:



4.1.6 NAT

Die Adressknappheit von IPv4 wird mit Hilfe von NAT umgangen, indem das Netzwerk intern **global** ungültige Adressen verwendet (bspw. `10.0.0.0/24`) und nur eine global gültige IP-Adresse besitzt. Verbindungen zu internen Hosts werden dann auf Paare abgebildet, die aus ebendieser Adresse und einem Port bestehen. Der NAT-Router muss diese Abbildung dann ausführen, wofür er eine Tabelle besitzt. Er überschreibt Adressen und Ports in den IP-Datagrammen. Die Größe ist dabei durch die Anzahl von Portnummern begrenzt.

Nachteil: Das Schichtenprinzip wird hierbei verletzt, da sich der Router mit Hosts beschäftigt und Ports für Dienste zwischen Transport- und Anwendungsschicht gedacht sind.

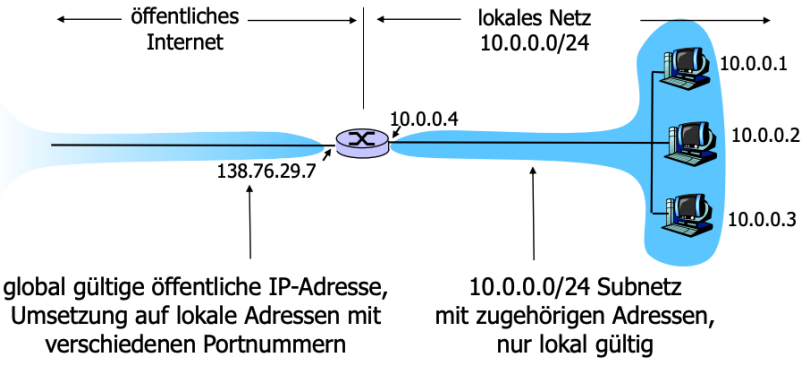
Nachteil: Eingriff in die Ende-zu-Ende-Verbindung

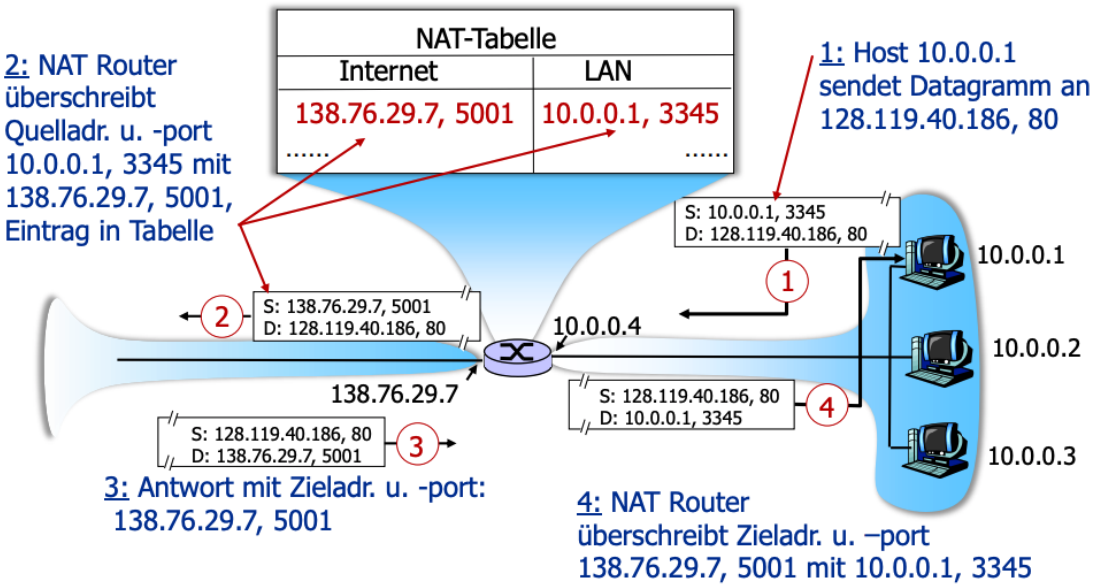
Nachteil: Hosts hinter NAT können nicht als Server auftreten (NAT-Traversal benötigt Tricks)

Vorteil: Interne Änderungen sind ohne externe Auswirkungen möglich und eine bessere Abschirmung wird erzielt.

IPv6 wäre eigentlich besser, NAT ist dennoch stark verbreitet.

Beispiel





4.1.7 IPv6

IPv6 — ursprünglich *Internet Protocol Next Generation (IPng)* — ist initiiert worden wegen des Adressraumproblems von IPv4. Dabei hat man gleich noch weitere Probleme gelöst:

- Header haben nun feste Länge; dies ermöglicht ein schnelles Weiterleiten
- Es gibt keine Fragmentierung mehr, sondern Pakete werden einfach verworfen falls sie größer als die MTU sind.
- Es gibt keine Prüfsumme mehr, die Fehlererkennung erfolgt in höheren Schichten.
- Man kann zusätzliche Optionen außerhalb über Headerketten angeben.
- Die Dienstgüte wird nun unterstützt.
- Informationssicherheit

Notation Gruppen von 16 Bit werden durch vier hexadezimale Ziffern getrennt durch Doppelpunkte repräsentiert. Beispiel:

$$\texttt{4BF5:AA12:0216:FEBC:BA5F:039A:BE9A:2176}$$

Man kann Nullblöcke zusammenfassen

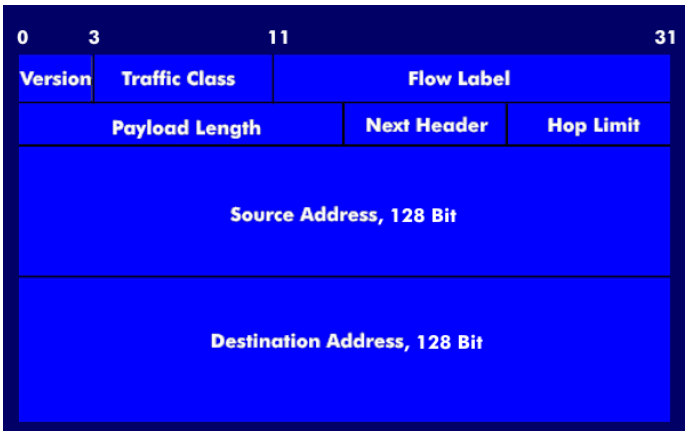
$$\texttt{4BF5:0000:0000:0000:BA5F:039A:000A:2176} \downarrow \texttt{4BF5:0:0:0:BA5F:039A:000A:2176}$$

und mehrere Nullblöcke weglassen (**dies aber nur einmal**)

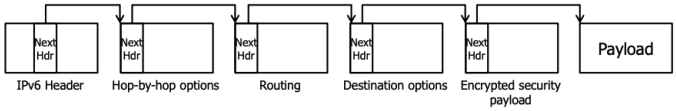
$$\texttt{4BF5:0000:0000:0000:BA5F:039A:000A:2176} \rightsquigarrow \texttt{4BF5::BA5F:039A:000A:2176}.$$

Es gibt auch eine gemischte Notation, da die letzten 32 Bit oft eine IPv4-Adresse ist. Hierzu wird dann die Dotted-Decimal-Schreibweise verwendet. DNS hat hierfür AAAA-Einträge und verwendet längere Prefixe.

Header Konzept Die Header von IPv6:



Es gibt für jede Aufgabe einen eigenen „Extension Header“ fester Größe. Damit kommt es zu einer schnellen Verarbeitung und jedes System sieht nur in die benötigten oder bekannten Header.



Beispiele hierfür sind TCP, UDP, IPsec ESP, IPsec AH, ICMPv6, SCTP.

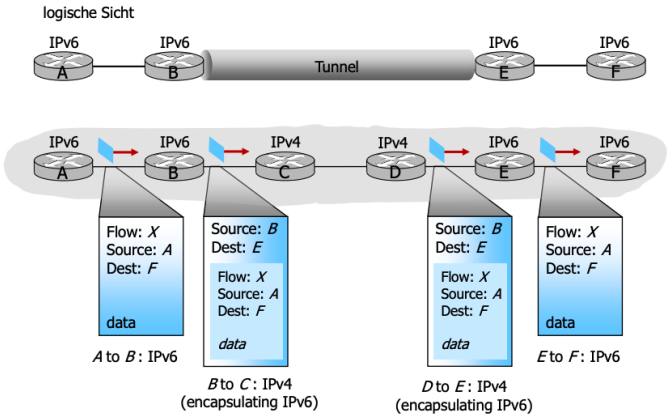
Übergang von IPv4 zu IPv6 Eine gleichzeitige Umstellung ist unmöglich, deswegen gibt es verschiedene Ansätze für den Übergang:

— **Dual Stack**: Ein Endsystem mit IPv6 **und** IPv4-Implementierung, abhängig vom Ziel (*diese Information wird vom DNS geliefert*) wird IPv6 oder IPv4 verwendet. **Nachteil:** Zusätzliche IPv6-Information wie beispielsweise das Flow-Label gehen dabei verloren, dies wird problematisch, wenn Router nur mit IPv4 weiterleiten

— **Tunneling**: Das IPv6-Datagramm wird in ein IPv4-Datagramm eingebettet. Es gibt diverse Tunnelmechanismen wie 6to4, 6in4, 6over4, ISATAP, ...

— **NAT**: Ein Datagramm mit privater IPv4-Adresse wird in ein IPv6-Paket eingebettet und bei der NAT-Übersetzung in ein IPv4-Paket mit einer öffentlichen Adresse verwandelt. Einsatz beispielsweise bei Dual-Stack-Lite.

Ein Beispiel für das Tunneling:

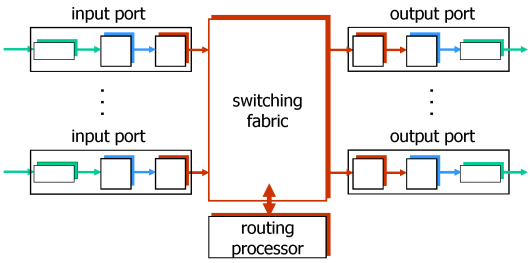


4.2 Aufbau eines Routers

Aufbau eines Routers

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet,* 7th Ed., Pearson Education, 2017.

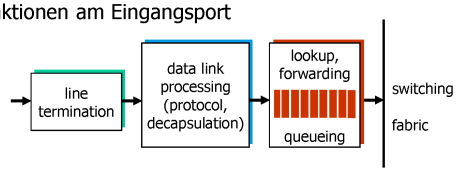
- Aufgaben
  - Weiterleitung, Ausführung von Routingprotokollen
  - prinzipieller Aufbau:



Aufbau eines Routers

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet,* 7th Ed., Pearson Education, 2017.

- Funktionen am Eingangsport

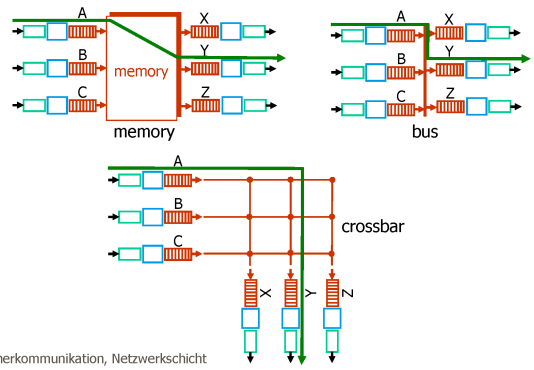


  - Pufferung, falls Pakete schneller von der Leitung kommen, als sie weitergegeben werden können
  - Paketverlust, wenn Puffer überläuft
  - verteiltes Weiterleiten: Port besitzt Kopie der Weiterleitungstabelle
  - effiziente Datenstrukturen für schnelle Suche
  - Empfang eines Pakets mit 64 Byte bei 10 Gbps dauert 51,2 ns, Longest Prefix Match in dieser Zeit benötigt Hardware-Lösung
  - inhaltsadressierbarer Speicher (Content Addressable Memory, CAM)

Aufbau eines Routers

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet,* 7th Ed., Pearson Education, 2017.

- Möglichkeiten für die Switching Fabric



Aufbau eines Routers

- Möglichkeiten für die Switching Fabric
  - Speicher
    - frühe Router waren einfache Rechner: CPU kopiert Paket von Eingangsport in Hauptspeicher, führt Weiterleitungsentscheidung durch und kopiert Paket zum Ausgangsport, heute auch möglich
    - 2 mal internen Bus benutzen, Begrenzung der Leistungsfähigkeit
    - in modernen Routern Direct Memory Access durch Eingangsports
  - Bus
    - ein Bus verbindet alle Ports, Eingangsport versieht Paket mit Markierung und sendet sie über den Bus per Broadcast an alle Ausgangsports, kann nur jeweils für einen Transfer benutzt werden, Wettbewerb
    - üblich für kleine bis mittlere Router

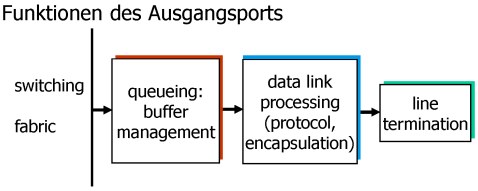
Aufbau eines Routers

- Verbindungsnetzwerk
  - bekannt aus der Verbindung von Prozessoren in Parallelrechnern
  - z.B. Crossbar: jeder Port kann direkt mit jedem anderen verbunden werden (quadratischer Schaltungsaufwand)
  - auch mehrstufige Anordnungen, z.B. Banyan-Netze
  - ermöglicht nebenläufige Weiterleitung

Aufbau eines Routers

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet,* 7th Ed., Pearson Education, 2017.

- Funktionen des Ausgangsports

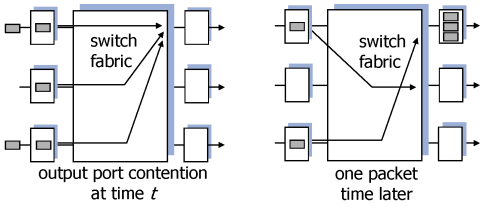


  - Pufferung, wenn Switching Fabric schneller liefert als Pakete auf die Leitung gegeben werden können
  - Paketverlust, wenn Puffer überläuft, z.B. ankommendes Paket (Tail-Drop)
  - Active Queue Management: proaktive Entscheidung, wann Pakete verworfen werden, z.B. mit Random Early Detect (RED)
  - Scheduling: wenn mehrere Pakete gepuffert sind, kann entschieden werden, welches als nächstes gesendet wird, z.B. FIFO oder Weighted Fair Queuing (WFQ), Deficit Round Robin (DRR), ermöglicht Dienstgüte

Aufbau eines Routers

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

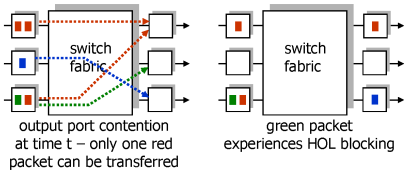
- Pufferungs- und Verlusteffekte
  - wenn Switching Fabric schneller als Anzahl Ports x Leitungsgeschwindigkeit, normalerweise keine Pufferung bei Eingangsports notwendig
  - wenn mehrere gleichzeitig zu einem Ausgangsport schicken, ist dort Pufferung notwendig:



Aufbau eines Routers

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

- Pufferungs- und Verlusteffekte
  - Head-of-the-Line (HOL) Blocking
    - falls Switching Fabric nicht schneller als Anzahl Ports x Leitungsgeschwindigkeit
    - mehrere Eingangsports wollen auf gleichen Ausgangsport
    - manche müssen warten, die dahinter werden blockiert, obwohl ihr Ausgangsport frei wäre



  - noch weitere Effekte, Auslegung durch Simulation und Analyse

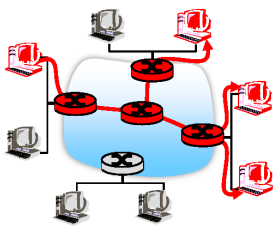
4.3 Routing

Routing bezeichnet ein Verfahren, mit welchem der Router entscheiden kann, über welchen Weg ein Paket gesendet werden soll. Wir unterscheiden zwischen **inter**- und *intradomain*-Routing. Bei **ersterem** wird zwischen verschiedenen Routingdomänen geroutet. Ausgetauschte Routing-Informationen enthalten dabei ganze Pfade, die Auswahl erfolgt durch gewisse Regeln. Bei *letzterem* wird innerhalb einer Routing-Domäne, sprich unter einer administrativen Instanz geroutet. Hier können Verfahren gewählt werden, welche nicht für sehr große Netze skalieren wie das Link-State oder das Distanzvektorrouting.

Routing

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

- Unicast-Routing (Punkt-zu-Punkt)
  - proaktiv: Information über Netztopologie wird ausgetauscht, aktuell gehalten, mit Graph-basierten Verfahren werden Pfade zu allen Zielen bestimmt, bei Sendewunsch werden diese genutzt
  - auf diesen Fall beschränken wir uns hier
- Multicast-Routing (Punkt-zu-Mehrpunkt)
  - Router und Links sollen effizient genutzt werden
  - Erweiterung von Unicast-Routing
  - ebenfalls proaktiv
- Ad-Hoc-Routing
  - dynamische Netztopologie: Pfade veralten schnell
  - Erweiterung von proaktiven Verfahren
  - auch reaktive Verfahren: erst bei Sendewunsch wird Pfad bestimmt
- Datenzentrische Verfahren
  - adresslos, Nachrichten werden aufgrund ihres Inhalts weitergeleitet, z.B. in Sensornetzen



4.3.1 Graphen — Allgemeines

Routing: Graphen

- Graphen sind eine übliche Abstraktion für Netze
- Anwendung von Suchverfahren aus Graphentheorie
- einige Grundbegriffe:
  - Graph G = (N,E)
  - Knoten N (nodes)
  - Kanten $E \subseteq N \times N$ (edges), eine Kante ist ein Paar $(v,w) \in E$, v und w heißen Nachbarn
  - ein Graph ist ungerichtet wenn alle Kanten symmetrisch sind: $(v,w) \in E \Rightarrow (w,v) \in E$ wir betrachten im Folgenden nur ungerichtete Graphen
  - Kosten sind eine Funktion $c: E \rightarrow K$ auf eine geeignete Menge K, verkürzte Schreibweise c(v,w)

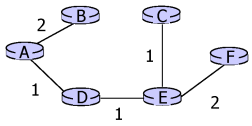
Routing: Graphen

- einige weitere Begriffe
  - ein Pfad ist eine Sequenz $(v_1,v_2,\ldots, v_n)$, so dass alle Paare (v_1,v_2), $(v_2,v_3),\ldots, (v_{n-1},v_n) \in E$
  - die Kosten eines Pfads betragen $c(v_1,v_2)+c(v_2,v_3)+\ldots+c(v_{n-1},v_n)$
  - ein Pfad zwischen zwei Knoten v_1 und v_n heißt kürzester Pfad, wenn es zwischen diesen Knoten keinen Pfad mit geringeren Kosten gibt (es kann mehrere kürzeste Pfade zwischen zwei Knoten geben), seine Kosten heißen Distanz $D(v_1,v_n)$ zwischen v_1 und v_n
  - ein Zyklus ist ein Pfad mit Anfangsknoten = Endknoten
  - ein Graph ist zusammenhängend, wenn es einen Pfad zwischen jedem Knotenpaar gibt

Routing: Graphen

- und noch ein paar ...
  - ein Baum ist ein zusammenhängender Graph, der keine Zyklen enthält
  - ein aufspannender Baum eines Graphen G = (N,E) ist ein Baum B = (N,E´) mit $E' \subseteq E$
  - ein minimaler aufspannender Baum eines Graphen G für einen Startknoten v ist ein aufspannender Baum von G, der für jedes Knotenpaar mit v als Startknoten einen kürzesten Pfad aus G enthält (single-source shortest paths spanning tree, 1-SPST)

Beispiel: minimaler aufspannender Baum für A:



Routing: Graphen

- Eigenschaft der kürzesten Pfade
  - wenn v ein Teil des kürzesten Pfades von x nach y ist, dann setzt sich der kürzeste Pfad von x nach y aus den kürzesten Pfaden von x nach v und v nach y zusammen

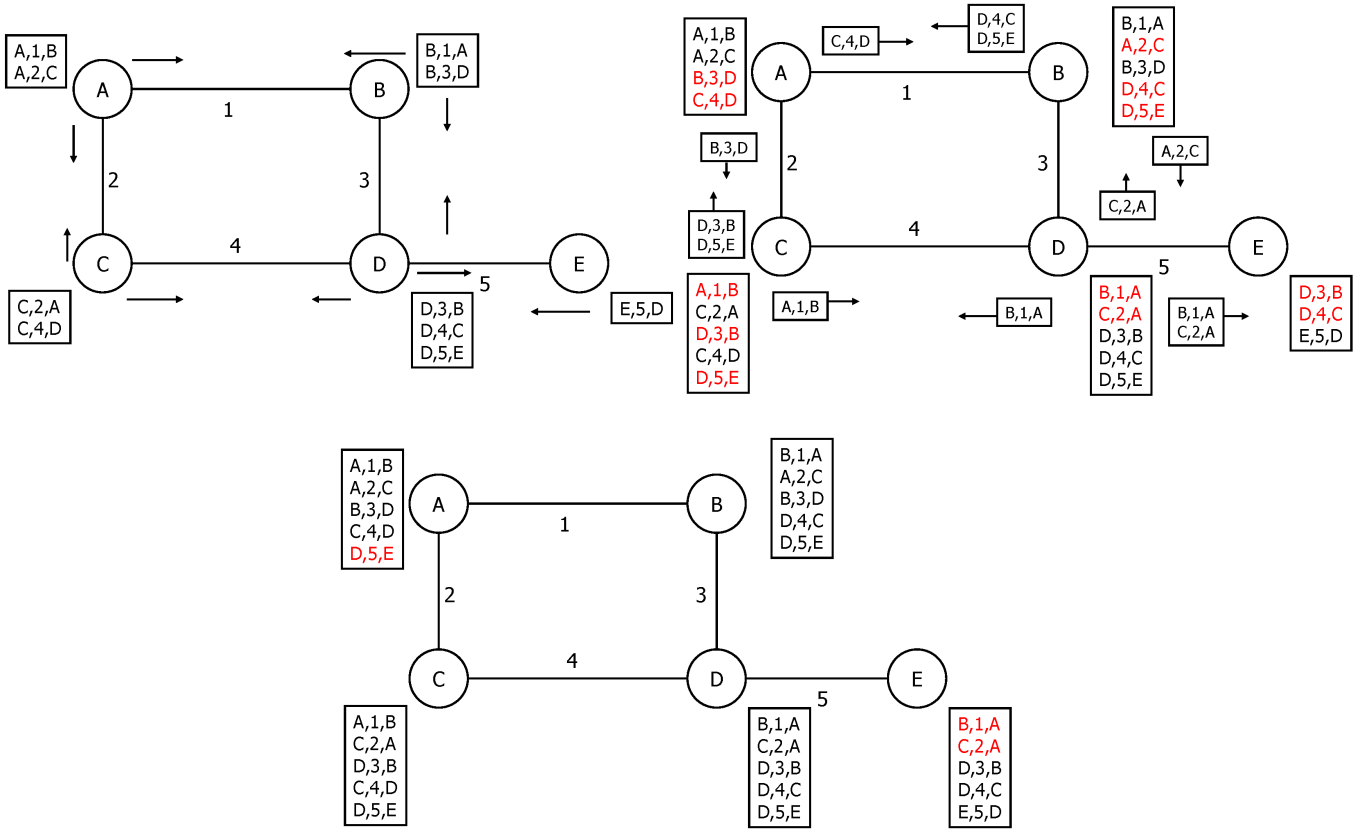
  - führt zu Rekursionsschema: $D(x,y) = \min_v\{D(x,v) + D(v,y)\}$
  - dies wird von den Verfahren zur Suche kürzester Pfade ausgenutzt

4.3.2 Link-State-Routing

Beim Link-State-Routing besitzt jeder Router vollständige Informationen über die gesamte Routing-Domäne, jeder Router berechnet kürzeste Pfade von ihm zu allen anderen Routen mit dem Dijkstra-Verfahren.
Die vollständige Kenntnis der Netztopologie wird durch das Fluten erreicht. Jeder Knoten berechnet daraufhin dann die kürzesten Pfade zu allen anderen Knoten. Wiederholt wird der Prozess bei Änderungen in der Netztopologie, was durch einen Austausch von Hellonachrichten zwischen benachbarten Routern erkannt werden kann.

Fluten Das Fluten erfolgt über **Link-State-Advertisments (LSAs)**. Diese enthalten eine Kennung des Knotens, der LSA erzeugt, Kosten zu Nachbarn und dessen Kennung, sowie einer Sequenznummer und der Lebensdauer. Jeder Knoten erzeugt die LSAs mit den ihm bekannten Informationen über die Verbindungen zu den Nachbarn und sendet sie an alle Nachbarn. Neue von Nachbarn erhaltene LSAs werden an alle Nachbarn weitergeleitet, aber nicht an den Nachbarn, von dem das LSA kam. Zur Erzielung von Zuverlässigkeit gibt es auch Bestätigungen und Sendewiederholungen zwischen Nachbarn, sowie Sequenznummern und Lebensdauern.

Fluten — Beispiel



Suche des kürzesten Weges — Dijkstra Der minimale Spannbaum wird iterativ aufgebaut. Dabei enthält eine Knotenmenge N' enthält immer die Knoten, für die die kürzesten Pfade bereits bekannt sind. N' wird dabei mit dem **Startknoten** u initialisiert. Es wird dann jeweils ein Nachbarknoten v hinzugefügt, der über alle aktuellen Knoten w von N' und die jeweilige Kante zu v den kürzesten Pfad besitzt. Es gilt

$$D(u,v) = \min_{w \in N'} (D(u,w) + c(w,v)).$$

Der Algorithmus endet, wenn $N = N'$ gilt.
Es gibt also eine Knotenmenge N mit einem Startknoten u. N' ist dann die Menge von Knoten, zu denen kürzeste Pfade von u aus bereits bekannt sind. Die Kosten $c(v,w)$ zwischen zwei Knoten v und w sind die positiven Verbindungskosten, wenn v und w Nachbarn sind, 0 für $v = w$ und ∞ sonst. Die Distanz $D(v)$ gibt die Kosten des aktuell bekannten kürzesten Pfads von u nach v aus. Die Vorgängerfunktion $p(v)$ gibt die Vorgänger von v auf dem aktuell bekannten kürzesten Pfad von u nach v aus.

Verfahren 4.1 (*Dijkstra-Verfahren*)

Schritt 1 Initialisierung: $N' \leftarrow \{u\}$ und $D(v) \leftarrow c(u,v) \forall\, v \in N$.

Schritt 2 **Wiederhole** bis $N' = N$

Schritt 2a Finde ein $w \in N \backslash N'$, so dass $\forall\, v \in N \backslash N'. D(w) \leqslant D(v)$.

Schritt 2b $N' \leftarrow N' \cup \{w\}$

Schritt 2c Mache für alle $v \in N \backslash N'$, **falls** $D(w) + c(w,v) < D(v)$:

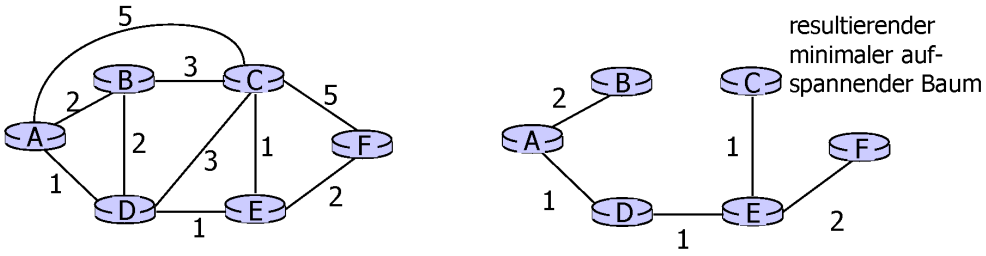
$$D(v) \leftarrow D(w) + c(w,v) \qquad \text{und} \qquad p(v) = w.$$

Suche des kürzesten Weges — Dijkstra: Beispiel

Beispiel für den Ablauf des Dijkstra-Verfahrens:

		jeweils D(·), p(·)				
Schritt	N′	B	C	D	E	F
0	A	2,A	5,A	1,A	∞,-	∞,-
1	A,D	2,A	4,D	″	2,D	∞,-
2	A,D,B	″	4,D	″	2,D	∞,-
3	A,D,B,E	″	3,E	″	″	4,E
4	A,D,B,E,C	″	″	″	″	4,E
5	A,D,B,E,C,F	″	″	″	″	″

(Einträge mit ″ können sich nicht mehr verändern, da Knoten in N′)



Suche des kürzesten Weges — Praxis Die Routing-Tabelle enthält für jedes Ziel v den nächsten HOP auf dem kürzesten Weg. Dies kann aus dem Vektor $p(v)$ mit der definierten Funktion

$$\texttt{nexthop}(v) = \begin{cases} v & \text{, falls } p(v) = u \\ \texttt{nexthop}(p(v)) & \text{sonst} \end{cases}$$

bestimmt werden. In der Praxis sammelt nun jeder Knoten die LSAs und berechnet daraus dann direkt die Routing-Tabelle. Dafür wird die als **Forward-Search-Algorithmus** bekannte Variante genutzt. Dazu werden Einträge der Form (Ziel, Kosten, nächster HOP) in zwei Listen verwaltet, der *bestätigtenListe*, welche N' entspricht und der *vorläufigenListe*, welche den Nachbarn von Knoten aus N' entspricht. `nexthop(v)` ist dann der nächste Hop, um einen Knoten v vom Startknoten u aus zu erreichen. Die Werte für $c(w,v)$ werden aus den LSAs gelesen, die Werte für $D(w)$ und `nexthop(w)` werden aus den Einträgen der Listen gelesen.

Verfahren 4.2 (*Dijkstra-Verfahren*)

Schritt 1 Initialisierung: bestätigteListe $\leftarrow \langle (u,0,-) \rangle$ und vorläufigeListe $\leftarrow \langle\rangle$.

Schritt 2 **Wiederhole**

Schritt 2a $w \leftarrow$ letzter in bestätigteListe eingetragener Knoten

Schritt 2b Mache für alle Nachbarn v von w

Schritt 2b – i Falls $v \notin$ bestätigteListe, vorläufigeListe, füge

$$(v, D(w) + c(w,v), \texttt{nexthop}(w))$$

der vorläufigeListe hinzu.

Schritt 2b – ii Falls $v \in$ vorläufigeListe und $D(w) + c(w,v) < D(v)$, so ersetze den Eintrag für v in vorläufigeListe durch

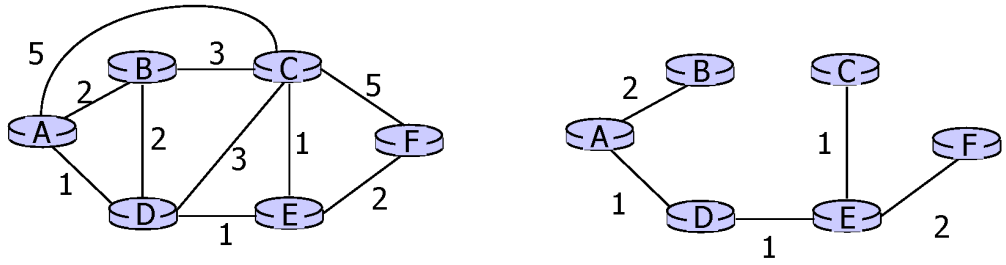
$$(v, D(w) + c(w,v), \texttt{nexthop}(w)).$$

Schritt 2c Verschiebe den Eintrag mit den geringsten Kosten von der vorläufigeListe in die bestätigteListe.

solange bis die vorläufigeListe leer ist.

Suche des kürzesten Weges — Praxis: Beispiel

Schritt	bestätigteListe	vorläufigeListe
0	(A,0,-)	
1	(A,0,-)	(B,2,B), (C,5,C), (D,1,D)
2	(A,0,-), (D,1,D)	(B,2,B), (C,4,D), (E,2,D)
3	(A,0,-), (D,1,D), (B,2,B)	(C,4,D), (E,2,D)
4	(A,0,-), (D,1,D), (B,2,B), (E,2,D)	(C,3,D), (F,4,D)
5	(A,0,-), (D,1,D), (B,2,B), (E,2,D), (C,3,D)	(F,4,D)
6	(A,0,-), (D,1,D), (B,2,B), (E,2,D), (C,3,D), (F,4,D)	



OSPF (Open Shortest Path First)

Routing: Link-State-Routing

- OSPF (Open Shortest Path First)
  - verbreitetes Intradomain-Routing-Protokoll, OSPFv2 in RFC 2328, OSPFv3 für IPv6 in RFC 5340
  - Neighbor Discovery: Router erkundet seine Nachbarn durch Austausch von Hello-Nachrichten (z.B. alle 10 s), Point-to-Point bzw. Broadcast bei Anschluss an Multi-Access-Netzwerk (z.B. Ethernet), Kennzeichnung von Router durch eindeutige ID (z.B. Loopback-IP-Adresse)
  - Synchronizing Database State: Austausch von Database-Description- und Link-State-Request-Nachrichten zwischen Nachbarn
  - Advertizing Link State: Versenden von Link-State-Update-Nachrichten mit LSAs an Nachbarn (periodisches Fluten, z.B. alle 30 min.)
  - Designated Routers: bei Anschluss an Multi-Access-Netzwerk Auswahl eines Routers basierend auf Priorität und ID zur Verringerung der Kommunikationsbeziehungen
  - kürzeste Wege mit Dijkstra-Verfahren

Routing: Link-State-Routing

- weitere Mechanismen von OSPF
  - Authentifizierung (in OSPFv3 entfernt, auf IPv6 verlagert)
  - OSPF Areas: 2 Hierarchieebenen für große Domänen
    - Area Zero ist Backbone, Area 1 bis N darunter
    - in jeder Area eigenes Routing (Aufbau Link State, Dijkstra)
  - Type-of-Service (TOS)
    - verschiedene Kostenmetriken sind möglich, Standard ist Hop-Count, weiter möglich ist z.B. max. Durchsatz, min. Verzögerung
    - werden manuell gesetzt, für jede Metrik wird minimaler aufspannender Baum berechnet, ermöglicht zentrale Ermittlung und Verteilung von Gewichten
    - wenn in IP-Paket im TOS-Feld dieser Wert gesetzt ist, wird der entsprechende Baum verwandt
    - kaum genutzt
  - Equal Cost Multiple Path (ECMP)
    - mehrere kürzeste Wege gleicher Länge mit geändertem Dijkstra-Verfahren bestimmen, Verkehr über diese Wege verteilen

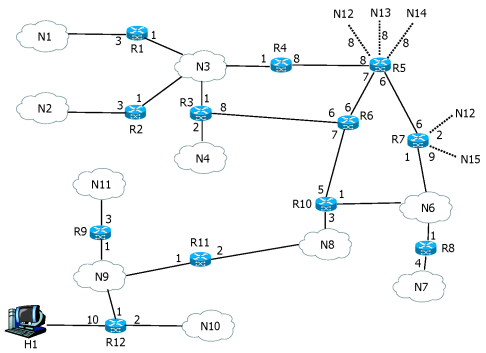
Routing: Link-State-Routing

- Repräsentation eines Netzwerks als Graph in OSPF
  - Knoten
    - Router
    - Multi-Access-Netzwerk: Transit (mit Durchgangsverkehr), Stub (sonst)
  - Kanten
    - direkte Verbindungen zwischen Routern (Punkt-zu-Punkt)
    - Verbindungen zwischen Routern und Netzwerken (Multi-Access)
  - Kosten nur für Ausgangsports der Router, Kanten von Multi-Access-Netzwerken zu Routern besitzen keine Kosten (Kosten = 0)
- Bsp.: Netzwerk mit
  - Punkt-zu-Punkt-Verbindungen: z.B. zwischen R6 und R10
  - Multi-Access-Verbindungen: z.B. R1, R2, R3, R4 an N3
  - Stub-Netzwerk nur an einem Router: z.B. N7
  - Host direkt an Router: H1
  - Verbindungen zu anderen Netzwerken: N12 – N15

Routing: Link-State-Routing

Quelle: W. Stallings. *Data and Computer Communications*, 10th Ed., Pearson Education, 2014.

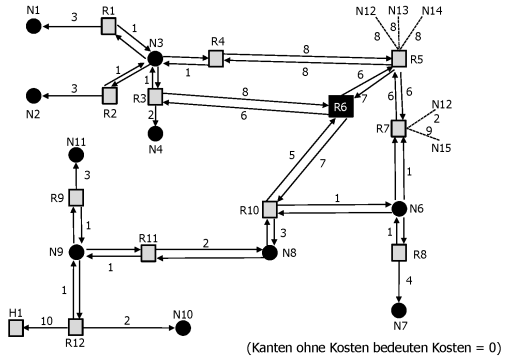
- Bsp.: Netzwerk



Routing: Link-State-Routing

Quelle: W. Stallings. *Data and Computer Communications*, 10th Ed., Pearson Education, 2014.

- Bsp.: Abbildung des Netzwerks auf gerichteten Graph

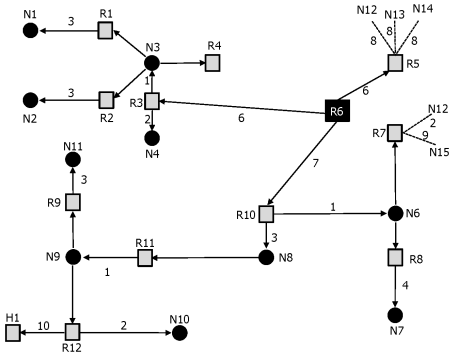


(Kanten ohne Kosten bedeuten Kosten = 0)

Routing: Link-State-Routing

Quelle: W. Stallings. *Data and Computer Communications*, 10th Ed., Pearson Education, 2014.

- Bsp.: resultierender 1-SPST für R6



Routing: Link-State-Routing

Quelle: W. Stallings. *Data and Computer Communications*, 10th Ed., Pearson Education, 2014.

- Bsp.: Routing-Tabelle für R6

Ziel	Next Hop	Distanz
N1	R3	10
N2	R3	10
N3	R3	7
N4	R3	8
N6	R10	8
N7	R10	12
N8	R10	10
N9	R10	11
N10	R10	13
N11	R10	14
H1	R10	21
R5	R5	6
R7	R10	8
N12	R10	10
N13	R5	14
N14	R5	14
N15	R10	17

4.3.3 Distanzzvektor-Routing

Jeder Router kenn nur die Kosten zu seinen direkten Nachbarn und die von ihm erreichbaren Ziele. Die kürzesten Pfade werden verteilt mit dem Bellman-Ford-Verfahren berechnet.

Bellman-Ford-Verfahren Die Suche nach den kürzesten Wegen wird hier verteilt durch alle beteiligten Knoten durchgeführt. Jeder Knoten teilt seinen Nachbarn mit, mit welchen minimalen Kosten er andere Knoten erreichen kann. Anfangs können nur die Kosten zu den Nachbarn bekanntgegeben werden, mit jedem Austausch werden dann längere Pfade bekannt. Hierbei wird für den ersten Teil eine Kante verwendet, es gilt wieder

$$D(u,v) = \min_w (c(u,w) + D(w,v)).$$

Wenn ein Knoten eine kürzere Entfernung ermittelt, so sendet er diese Information an alle Nachbarn. Der Algorithmus endet, wenn sich keine Veränderung mehr ergibt (**Konvergenz**).
Es werden wieder identisch Kosten $c(v,w)$ gebildet. Die Distanz $D_u(v)$ ist die bekannte kürzeste Entfernung von u nach v. Die Distanzen eines Knotens fasst man in einem Distanzvektor D_u zusammen. Jeder Knoten u besitzt dann die Distanzen $D_u(v)$ für alle $v \in N$, **lokale** Kopien $D_w(v)'$ der Distanzen $D_w(v)$ von allen Nachbarn w zu allen Knoten $v \in N$, sowie den nächsten Hop $\texttt{nexthop}_u(v)$, um v von u aus zu erreichen für alle Knoten $v \in N$.

Verfahren 4.3 (*Bellman-Ford-Verfahren*)

Schritt 1 Initialisierung:

Schritt 1a Für alle $v \in N$ setze

$$D_u(v) \leftarrow \begin{cases} c(u,v) & \text{, falls } v \text{ Nachbar von } u \\ \infty & \text{sonst} \end{cases}$$

und

$$\texttt{nexthop}_u(v) = \begin{cases} v & \text{, falls } v \text{ Nachbar von } u \\ - & \text{sonst} \end{cases}$$

Schritt 1b Für alle Nachbarn w setze $D_w(v) = \infty$ für alle $v \in N$ und sende D_u an w.

Schritt 2 **Wiederhole**

Schritt 2a **Warte** auf Änderungen der Kosten zu Nachbar w (D_u ändert sich) oder Erhalt eines Distanzvekotrs D_w von Nachbar w (D'_w ändert sich).

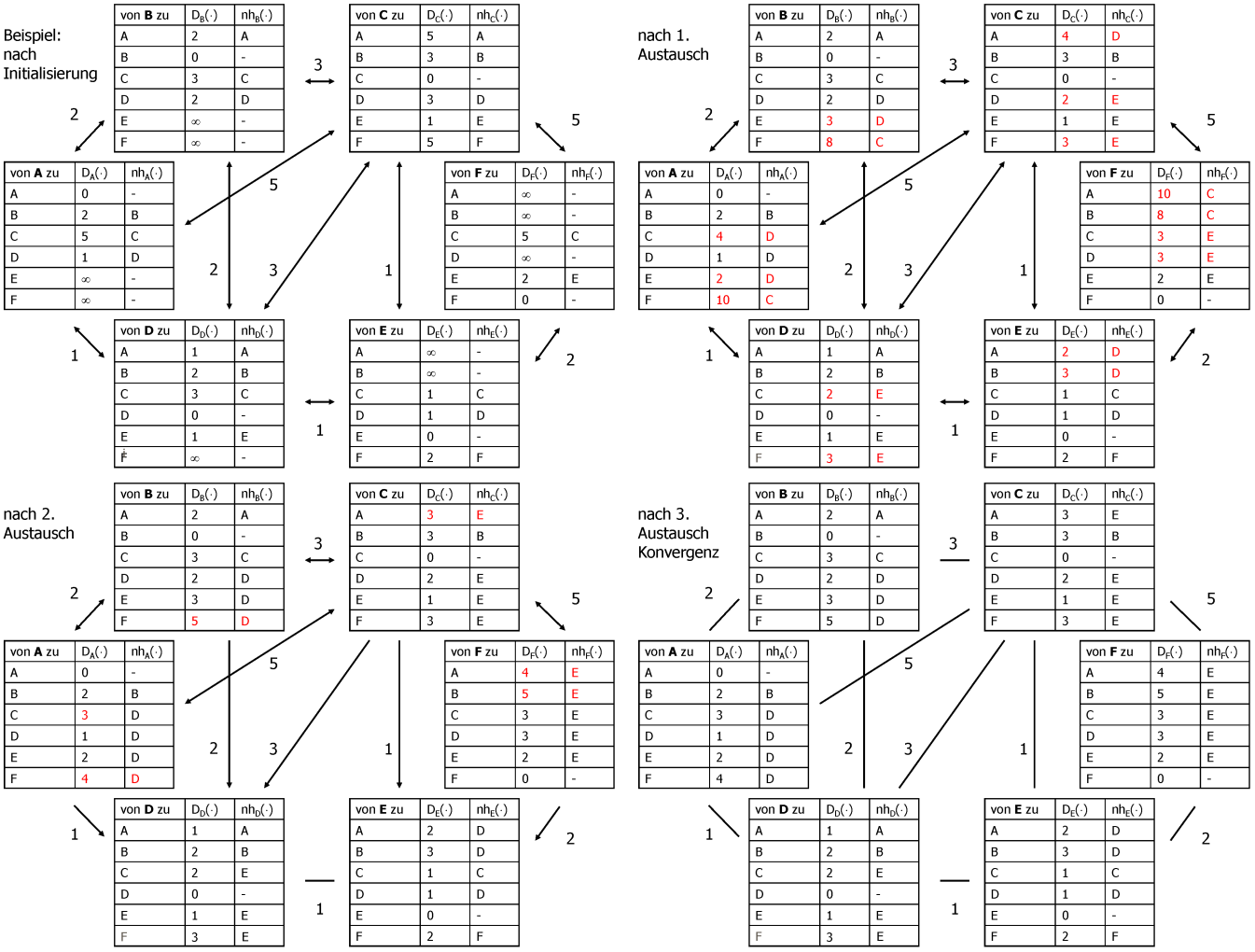
Schritt 2b Für alle $v \in N$ setze

$$D_u(v) \leftarrow \min_w \left(c(u,w) + D_w(v)'\right) \qquad \text{und} \qquad \texttt{nexthop}_u(v) \leftarrow w \text{ von eben}$$

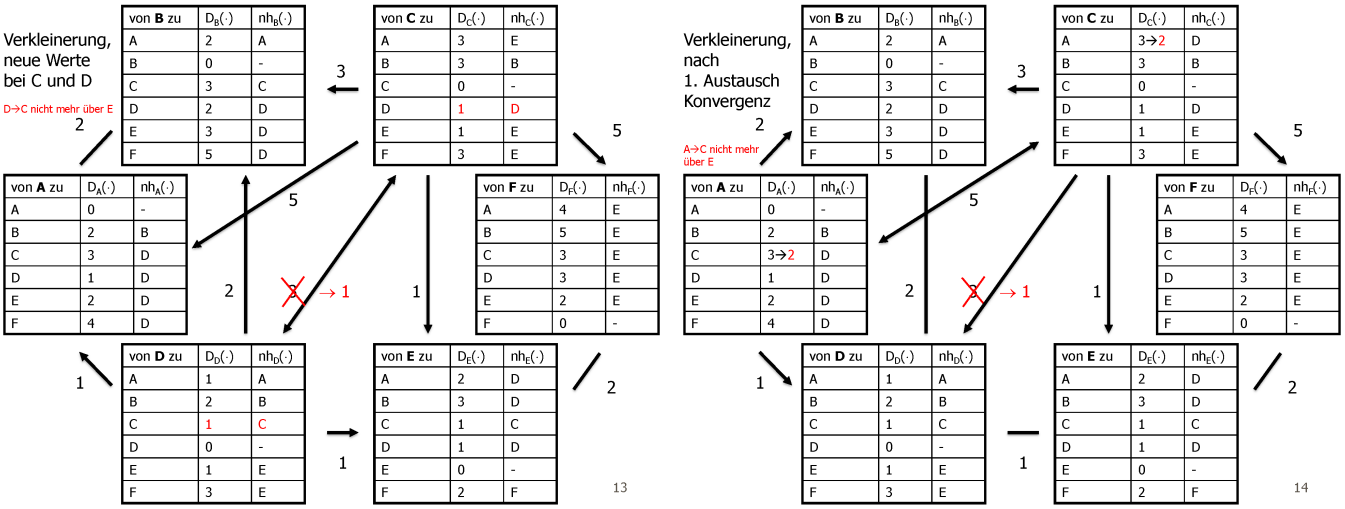
Schritt 2c Wenn eine Änderung in D_u voliegt, so sende D_u an alle direkten Nachbarn.

solange bis Konvergenz erreicht ist.

Beispiel



Verhalten bei Änderungen an der Netztopologie Der Algorithmus fuktioniert weiterhin bei Topologieänderungen und wenn die Informationen in asynchroner Weise ausgetauscht werden. Bei einer Verkleinerung der Verbindungskosten konvergiert das Verfahren schnell (**good news travel fast**), bei einer Vergrößerung der Verbindungskosten können jedoch durch Zyklen in den Pfaden Probleme entstehen (**bad news travel slowly**).

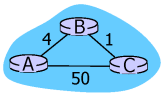


Routing: Distanzvektor-Routing

- Beispiel für Vergrößerung
  - Ausgangspunkt:

von **B** zu	$D_B(\cdot)$	$nh_B(\cdot)$
A	4	A
B	0	-
C	1	C

von **A** zu	$D_A(\cdot)$	$nh_A(\cdot)$
A	0	-
B	4	B
C	5	B



von **C** zu	$D_C(\cdot)$	$nh_C(\cdot)$
A	5	B
B	1	B
C	0	-

16

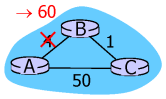
Routing: Distanzvektor-Routing

- Beispiel für Vergrößerung
  - nach Vergrößerung der Kosten

von **B** zu	$D_B(\cdot)$	$nh_B(\cdot)$
A	4→6	**C**
B	0	-
C	1	C

„zu C 1 und von C zu A 5"

von **A** zu	$D_A(\cdot)$	$nh_A(\cdot)$
A	0	-
B	4→51	C
C	5→50	C



von **C** zu	$D_C(\cdot)$	$nh_C(\cdot)$
A	5	B
B	1	B
C	0	-

17

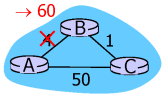
Routing: Distanzvektor-Routing

- Beispiel für Vergrößerung
  - nach 1. Austausch

von **B** zu	$D_B(\cdot)$	$nh_B(\cdot)$
A	6	C
B	0	-
C	1	C

„zu B 1 und von B zu A 6"

von **A** zu	$D_A(\cdot)$	$nh_A(\cdot)$
A	0	-
B	51	C
C	50	C



von **C** zu	$D_C(\cdot)$	$nh_C(\cdot)$
A	5→7	B
B	1	B
C	0	-

18

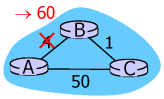
Routing: Distanzvektor-Routing

- Beispiel für Vergrößerung
  - nach 2. Austausch

von **B** zu	$D_B(\cdot)$	$nh_B(\cdot)$
A	6→8	C
B	0	-
C	1	C

„zu C 1 und von C zu A 7"

von **A** zu	$D_A(\cdot)$	$nh_A(\cdot)$
A	0	-
B	51	C
C	50	C



von **C** zu	$D_C(\cdot)$	$nh_C(\cdot)$
A	7	B
B	1	B
C	0	-

19

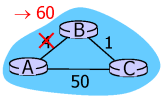
Routing: Distanzvektor-Routing

- Beispiel für Vergrößerung
  - nach 3. Austausch

von **B** zu	$D_B(\cdot)$	$nh_B(\cdot)$
A	8	C
B	0	-
C	1	C

„zu B 1 und von B zu A 8"

von **A** zu	$D_A(\cdot)$	$nh_A(\cdot)$
A	0	-
B	51	C
C	50	C



von **C** zu	$D_C(\cdot)$	$nh_C(\cdot)$
A	7→9	B
B	1	B
C	0	-

- usw. bis $D_B(A) = 51$ und $D_C(A) = 50$ erreicht ist ...
- Zyklischer Pfad zwischen B und C

20

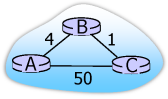
Count-to-infinity-Problem und Poisoned Reverse Die veraltete Information in den verteilten Routing-Tabellen enthält im letzten Beispiel einen zyklischen Pfad. Die langsame Iteration endet somit erst, wenn die Kosten des alternativen Pfads erreicht sind. Abhilfe schaffen einerseits eine Beschränkung des größten Kostenwerts, was aber auch die Größe des Netzwerks einschränkt und **Poisoned Reverse**. Wenn der kürzeste Weg von u nach v über den nächsten Hop w führt, sendet u an w die Kosten **von unendlich** für die Entfernung von u nach v. („*sich selbst dem Nexthop auf diesem Pfad nicht anbieten*")

Routing: Distanzvektor-Routing

- Beispiel für Vergrößerung mit Poisoned Reverse
  - Ausgangspunkt:

von **B** zu	$D_B(\cdot)$	$nh_B(\cdot)$
A	4	A
B	0	-
C	1	C

von **A** zu	$D_A(\cdot)$	$nh_A(\cdot)$
A	0	-
B	4	B
C	5	B



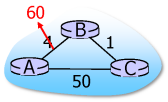
von **C** zu	$D_C(\cdot)$	$nh_C(\cdot)$
A	5	B
B	1	B
C	0	-

Routing: Distanzvektor-Routing

- Beispiel für Vergrößerung mit Poisoned Reverse
  - nach Vergrößerung der Kosten

von **B** zu	$D_B(\cdot)$	$nh_B(\cdot)$
A	60	A
B	0	-
C	1	C

von **A** zu	$D_A(\cdot)$	$nh_A(\cdot)$
A	0	-
B	51	C
C	50	C



C bietet sich nicht als Nexthop für den Pfad an

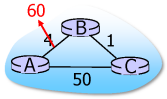
von **C** zu	$D_C(\cdot)$	$nh_C(\cdot)$
A („∞“ ←)	5	B
B	1	B
C	0	-

Routing: Distanzvektor-Routing

- Beispiel für Vergrößerung mit Poisoned Reverse
  - nach 1. Austausch

von **B** zu	$D_B(\cdot)$	$nh_B(\cdot)$
A	60	A
B	0	-
C	1	C

von **A** zu	$D_A(\cdot)$	$nh_A(\cdot)$
A	0	-
B	51	C
C	50	C



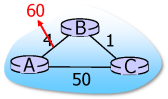
von **C** zu	$D_C(\cdot)$	$nh_C(\cdot)$
A	50	A
B	1	B
C	0	-

Routing: Distanzvektor-Routing

- Beispiel für Vergrößerung mit Poisoned Reverse
  - nach 2. Austausch Konvergenz

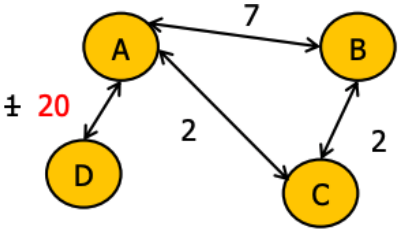
von **B** zu	$D_B(\cdot)$	$nh_B(\cdot)$
A	51	C
B	0	-
C	1	C

von **A** zu	$D_A(\cdot)$	$nh_A(\cdot)$
A	0	-
B	51	C
C	50	C

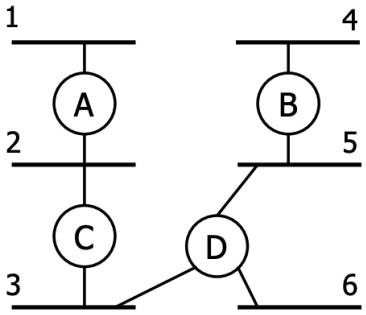


von **C** zu	$D_C(\cdot)$	$nh_C(\cdot)$
A	50	A
B	1	B
C	0	-

Poisoned Reverse — Grenzen Mit Poisoned Reverse können Zyklen der Länge 2 vermieden werden, nicht jedoch längere Zyklen.



Routing Information Protocol (RIP) Das *Routing Information Protocol* ist ein frühes und verbreitetes Protokoll und Teil von BSD Unix. Es verwendet Distanzvektorrouting. Router teilen hier in Advertisements über UDP den Nachbarn mit, welche Netzwerke sie mit welchen Kosten erreichen können. Diese Advertisments werden periodisch (alle 30 Sekunden) und bei Änderungen gesendet. Die verwendete Kostenmetrik ist die Anzahl von Hops, welche auf 15 beschränkt ist, 16 für Poisoned Reverse.



Beispiel:
Router C teilt A mit, dass er
Netzwerk 2 und 3 mit Kosten 0,
Netzwerk 5 und 6 mit Kosten 1,
Netzwerk 4 mit Kosten 2
erreichen kann

Vergleich Distanzvektor- und Link-State-Routing

Link-State-Routing	Distanzvektor-Routing
Es ist ein **zentrales Verfahren**.	Es ist ein **verteiltes Verfahren**.
Es gibt effiziente Implementierungen in $\mathcal{O}(n\log(n))$ (naiv in $\mathcal{O}(n^2)$).	Es hat Konvergenzprobleme bei Zyklen.
Es beschränkt die Skalierbarkeit.	Es beschränkt die Skalierbarkeit.
Robustheit: Nur die weitergegebene Topologie kann fehlerhaft sein.	**Robustheit**: Router können fehlerhafte Pfade weitergeben, wodurch eine Fehlerfortplanzung möglich ist.
Der Nachrichtenaustausch ist in $\mathcal{O}(ne)$ bei e Kanten.	Eine Fehlfunktion eines Routers wirkt sich auf andere aus.

! Dynamische — von der aktuellen Netzlast abhängige — Metriken führen zu instabilem Verhalten und haben sich nicht bewährt. Deswegen sind meistens statische Metriken im Einsatz.

4.3.4 Interdomain-Routing

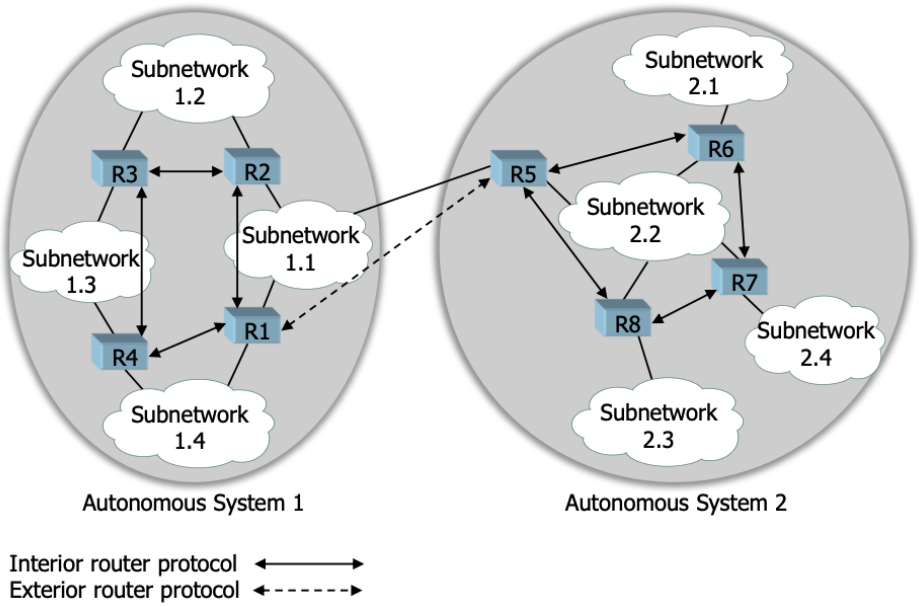
Innerhalb einer Routingdomäne wird über die *Interior Gateway Protokolle (IGP)*, zwischen Routingdomänen über die *Exterior Gateway Protokolle (EGP)* geroutet. Das Standardbeispiel für EGP ist das *Border Gateway Protocol (BGP)*. Die Routingdomänen sind **autonome Systeme (AS)**. Man unterscheidet drei Typen von AS:

— **Stub AS**: Diese AS haben nur eine Verbindung zu anderen AS.

— **Multihomed AS**: Diese AS haben mehrere Verbindungen zu anderen AS, befördern aber **keinen** Durchgangsverkehr.

— **Transit AS**: Diese AS haben mehrere Verbindungen zu anderen AS und befördern auch Durchgangsverkehr.

Die Transit AS sind durch eine zentral vergebene AS-Nummer (16 Bit) gekennzeichnet. Ein Router eines AS ist dann ein Gateway und führt das EGP aus.



Pfadbasiertes Routing Hierbei werden ganze Pfade ausgetauscht und es findet keine Berücksichtigung von Kosten statt.[2] Router geben dann nur die Pfade bekannt, welche andere Router nutzen sollen. Der Router sucht sich aus den bekannten Pfaden einen Pfad nach individuellen Regeln aus. Zudem können Zyklen erkannt werden und Pfade auch annulliert werden.

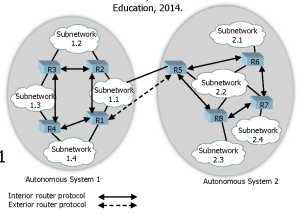
Funktion von BGP Die Gateways tauschen Advertisements in BGP-Sessions über TCP aus. Diese Advertisements enthalten dann Pfade zu Netzwerken, welche den Gateways bekannt sind. Ein Pfad besteht aus einer Liste von Netzwerken in einem erreichbaren AS, einer Sequenz von AS-Nummern zu diesem AS und der IP-Adresse des sendenden Gateways. Beim Weiterleiten eines Pfades fügt ein Router sein AS am Anfang des Pfads an und setzt sich dann als den nächsten HOP ein. Dadurch erlaubt er, dass der Verkehr an die Liste von Netzwerken über ihn geleitet wird. Pfade, in denen er selbst in der Sequenz auftaucht, werden verworfen.

Beispiel zu BGP

Routing: Interdomain-Routing

- Beispiel für den Ablauf von BGP
  - R1 erzeugt Routingtabelle für AS1 mit IGP
  - R1 schickt Advertisement an R5 (in AS2)
    - Netzwerkliste: alle Subnetze in AS1
    - Sequenz von AS-Nummern: {AS1}
    - Nächster Hop: IP-Adresse von R1
  - Annahme: R5 ist Nachbar von R9 in AS3
  - R5 leitet die Information von R1 an R9 in Advertisement weiter
    - Netzwerkliste: alle Subnetze in AS1
    - Sequenz von AS-Nummern: {AS2,AS1}
    - Nächster Hop: IP-Adresse von R5
  - R9 entscheidet, dass dies sein bevorzugter Pfad zu den Subnetzen in AS1 ist und leitet an seinen Nachbarn den Pfad {AS3, AS2, AS1} weiter

Quelle: W. Stallings. *Data and Computer Communications*, 10th Ed., Pearson Education, 2014.

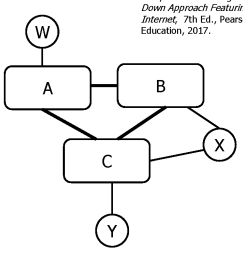


Rechnerkommunikation, Netzwerkschicht 117

Routing: Interdomain-Routing

- Beispielkonfiguration
  - A, B, C: Providernetz, Transit AS
  - X: Kundennetz, Multihomed AS
  - W, Y: Kundennetze, Stub AS
  - X möchte keinen Durchgangsverkehr, sendet also keine Advertisements (und kann dies auch gar nicht, da es keine AS-Nummer besitzt)
  - A versendet den Pfad AW an B
  - B versendet den Pfad BAW an X
  - B versendet den Pfad BAW nicht an C, weil B es nicht möchte, dass C Verkehr durch B leitet

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.



Rechnerkommunikation, Netzwerkschicht 118

[2] Dies wäre aufgrund der Größe der beteiligten Netze und der uneinheitlichen Metriken nicht möglich.

4.4 IPsec

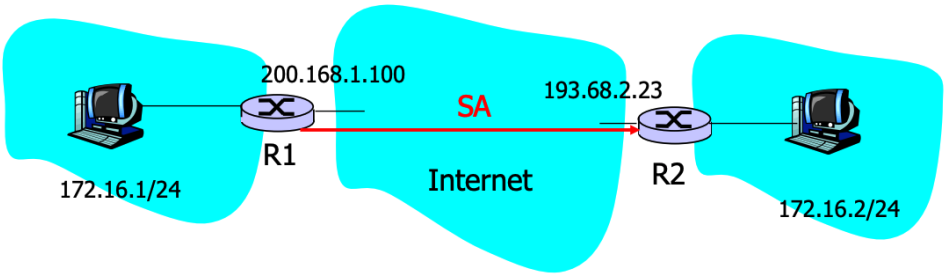
Das *IP security protocol* schafft Sicherheit auf Netzwerkschicht. Es bietet eine **Authentizität** des Senders, **Datenintegrität** der Nachrichten, **Vertraulichkeit** des Inhalts und von Protokollinformationen, sowie einen grundlegenden Schutz gegen Replay-Angriffe. Ebenso werden **VPNs** ermöglicht.
Es gibt verschiedene Varianten:

- **Authentication Header (AH)**: Ermöglicht Authentifikation und Integrität, aber **keine Vertraulichkeit**
- **Encapsulation Security Payload (ESP)**: Ermöglicht Authentifikation und Integrität **und Vertraulichkeit**
- **Transport Mode**: Ohne Tunneling, keine neuen IP-Header
- **Tunnel Mode**: Mit Tunneling und neuen IP-Headern

Am stärksten verbreitet ist ESP im Tunnel Mode.

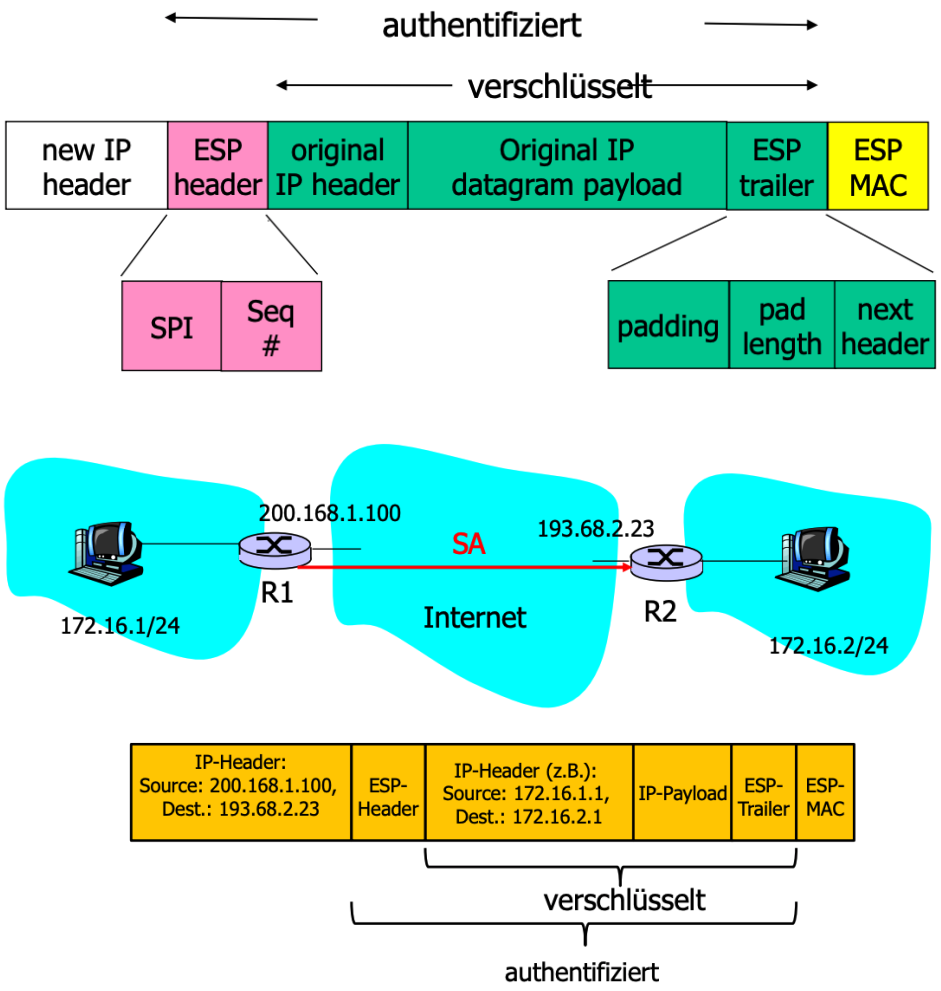
Security Associations IPsec erfordert das Einrichten von Security Associations (SA):

- **Security Association (SA)**: Eine logische unidirektionale Verbindung zwischen IPsec-Systemen.
- **Security Parameter Index (SPI)**: Eine 32-Bit-Kennung für SAs.
- **Security Association Database (SAD)**: Datenbank auf IPsec-System für SAs mit IP-Quell- und -Zieladresse, Algorithmen und Schlüssel für Verschlüsselung und MAC.
- **Security Policy Database (SPD)**: Datenbank auf IPsec-System für Behandlung von Paketen, z.B. Art der SA für IP-Ziel-Adresse.



Schlüsselaustausch Der Schlüsselaustausch bei IPsec erfolgt über IKEv2 Protokoll (RFC 7296). Beide Kommunikationspartner besitzen hierbei Zertifikate für öffentliche Schlüssel. Der Schlüsselaustausch erfolgt dann in zwei Phasen:
In **Phase 1** wird mit Diffie-Hellmann-Verfahren ein symmetrischer Schlüssel erzeugt. Damit wird dann eine bidirektionale authentifizierte und vertrauliche IKE SA aufgebaut. Ebenso wird ein *Master Secret (MS)* ausgetauscht.
In **Phase 2** w erden für die einzelnen IPsec SAs Algorithmen und Parameter ausgehandelt. Die Phase benötigt nur symmetrische Verschlüsselung, wobei die verwendeten Schlüssel mit Hilfe des MS erzeugt werden.

ESP-Paketformat im Tunnelmodus Das Orginal-IP-Datagramm wird mit ESP-Trailer auf die für die Verschlüsselung benötigte Länge aufgefüllt. Dies wird dann mit dem Schlüssel für die SA verschlüsselt. Der ESP-Header enthält SPI, damit der Empfänger die SA in seiner SAD findet. Dies wird dann mit MAC authentifiziert und in das ESP MAC geschrieben. Dies erhält dann einen neuen IP-Header mit IP-Adressen der IPsec-Systeme und mit 50 für IPsec-ESP im Protokollfeld.



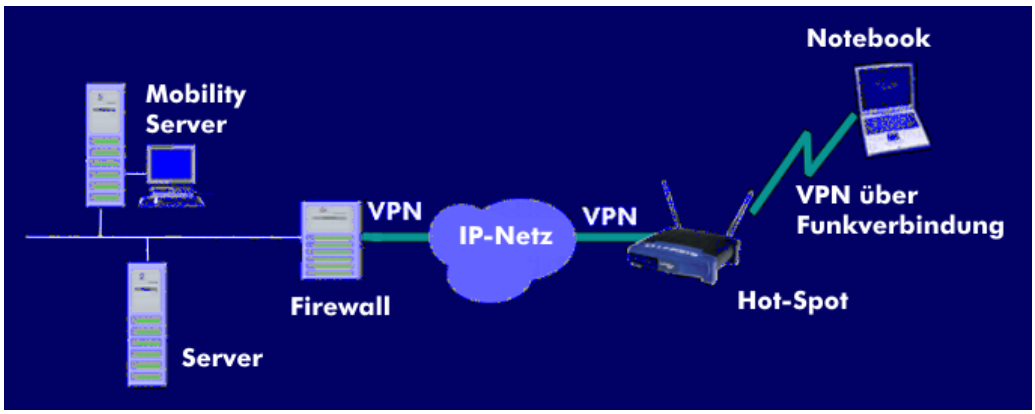
4.5 Netzvirtualisierung

Privates Netz Unternehmen können durch Eigentum oder Miete von Leitungen zwischen Standorten ein eigenes in sich geschlossenes Netz betreiben. **Vorteil:** Auslastung liegt in eigener Hand **Vorteil:** Hohe Sicherheit (getrennt von anderen Netzen, es ist ein physikalisches Eindrigen erforderlich) **Nachteil:** Hohe Kosten

4.5.1 VPNs mittels IP-Tunneling

Virtual Private Networks setzen auf einem öffentlichem Netz ein Overlay auf, welches wie ein privates Netz erscheint. Dabei gibt es verschiedene Varianten:

— **End-to-Site**: Es findet eine virtuelle Integration eines Hosts in ein anderes Netz statt. Beispielsweise die Anbindung eines mobilen Hosts an ein Unternehmensnetz, welches wie ein Host im Netzwerk erscheint.

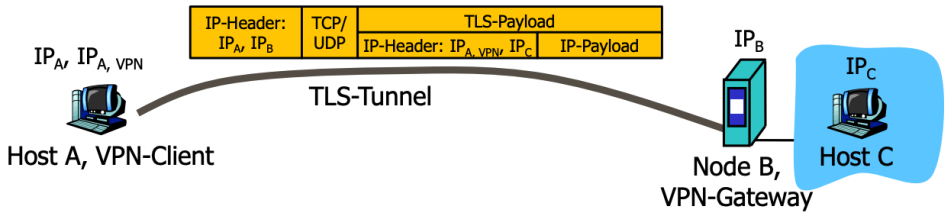


— **Site-to-Site**: Es findet eine Verbindung von Netzen beispielsweise an unterschiedlichen Standorten statt.

— **End-to-End**: Es findet eine Verbindung von Hosts zu einem virtuellen Netz statt.

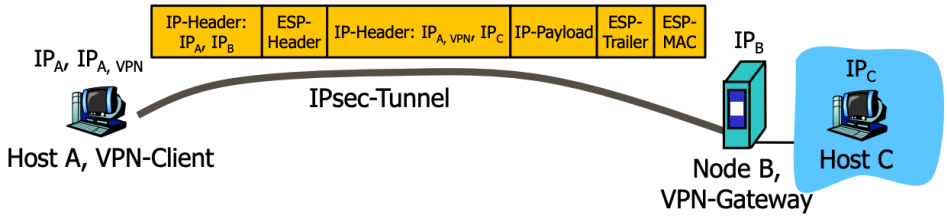
IP-Tunneling Als **Tunneling** bezeichnet man die Einbettung von Paketen eines Protokolls in Pakete eines Protokolls der gleichen oder einer höheren Schicht. Wir unterscheiden zwischen TSL-VPN, bei dem ein Tunneling von IP in TLS stattfindet, und IPsec-VPN, bei dem ein Tunneling von IP in IPsec stattfindet. Dies kann wahlweise mit Authentifikation, Integrität und Verschlüsselung stattfinden.

TSL-VPN



A baut sich hierbei eine sichere Transportverbindung mit TLS zu B auf. Dabei wird AUthentifizierung, Datenintegrität und Verschlüsselung ausgehandelt. Es findet ein Tunneling von DHCP durch TLS statt, um A eine zusätzliche *im VPN gültige* IP-Adresse $IP_{A,\,VPN}$ zuzuweisen. Dann erfolgt das Tunneling von IP-Paketen durch TLS, womit A wie ein Teil des Netzwerks erscheint.

IPsec-VPN

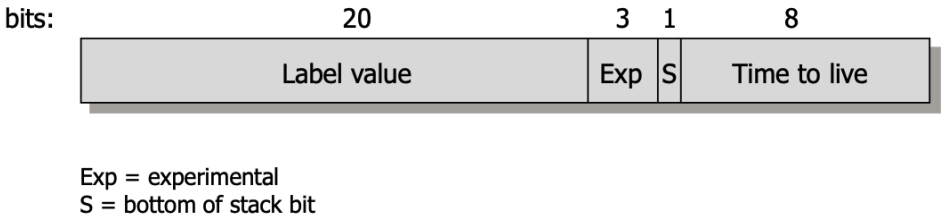


A baut sich hier eine Security Association (SA) zu B auf. Dabei wird die Protokollversion (hier: ESP) ausgehandelt und die IP-Adresse $IP_{A,\,VPN}$ zugewiesen. Danach findet ein Tunneling von IP-Paketen durch ESP statt, wodurch A wie ein Teil des Netzwerks erscheint. Dies ermöglicht zusätzliche Authentifizierung, Datenintegrität und Verschlüsselung und es gibt diverse Konfigurationsmöglichkeiten.

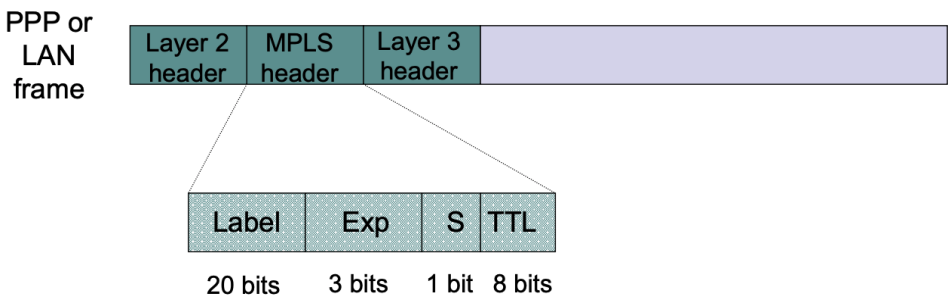
4.5.2 Multiprotocol Label Switching (MPLS)

Multiprotocol Label Switching (MPLS) findet fast bei jedem großen ISP Einsatz. Es kann mit mehreren Protokollen der Schicht 2 und Schicht 3 zusammenarbeiten. Es kommt zu einer virtuellen Leitungsvermittlung. Die Pakete erhalten ein **Label**, die Weiterleitung erfolgt über **Label-Switching**, jeder Router erkennt den Eingangsport und das Label, sieht in einer Tabelle nach, welches das neue Label ist und auf welchen Ausgangsport das Paket weitergeleitet wird. Es findet eine Signalisierung zum Aufbau eines **Label-Switched Path (LSP)** statt.
Vorteil: Es kann **schneller weitergeleitet** werden mit Labeln statt IP-Adressen (Vermeidung von Longes Prefix Match)
Vorteil: Traffic Engineering, zur optimalen Verkehrsführung können LSPs gezielt aufgebaut werden.
Vorteil: VPNs durch LSPs als Tunnel zwischen Nutzern.
Vorteil: Für LSPs sind Dienstgüteeigenschaften (QoS) möglich.

MPLS Header (*Shim Header*)



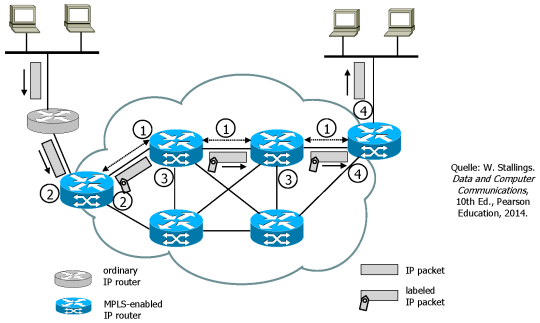
Label Value sind 20 Bit für den Wert des Labels, Exp sind 3 Bit für eine experimentelle Nutzung, beispielsweise für DiffServ-Klassen. MPLS Header können im Allgemeinen gestapelt werden, die Weiterleitung erfolgt mit dem obersten Label (links), ist S 1, so kennzeichnet dies den untersten Wert (rechts, *bottom of stack*). TTL gibt die Lebensdauer an, eine einfache Übernahme und Dekrementierung aus dem IP-Paket. Die Position des MPLS Headers ist zwischen Schicht 2 und Schicht 3:



MPLS Beispiel

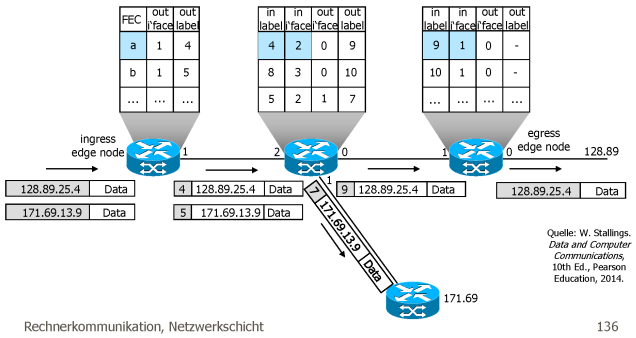
Multiprotocol Label Switching (MPLS)

- Weiterleiten von IP-Paketen mit Labels in einer MPLS-Domain über einen LSP:



Multiprotocol Label Switching (MPLS)

- Weiterleitungstabellen in jedem Label-Switching Router (LSR):

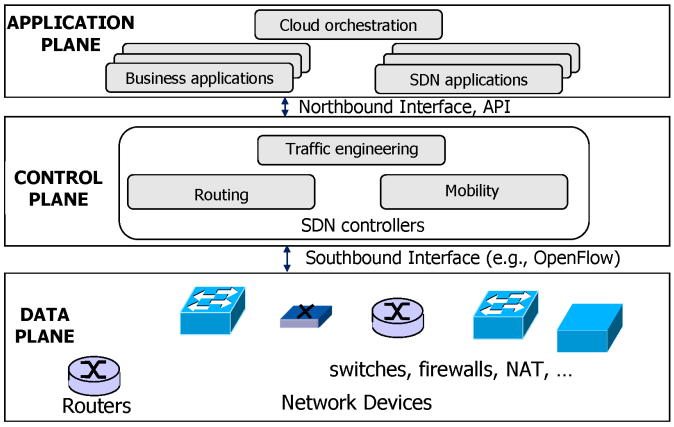


4.5.3 Software-Defined Networking (SDN)

Mit dem RFC 7426 gib es das Industriekonsortium der Open Networking Foundation (ONF). Es hat die Einführung und Weiterentwicklung des SDNs gewidmet. Bei diesem SDN möchte mann die **Data Plane** (zur Weiterleitung von Daten), die **Control Plane** (für die Steuerfunktionen) und die **Application Plane** trennen. Bislang waren diese Funktionen in Netzwerkgeräten integriert und über mehrere Geräte verteilt. Als Beispiel für ein Protokoll zwischen der Control und Data Plane ist OpenFlow zu nennen.
Der **SDN-Controller** instruiert dann den **OpenFlow-Switch**, wie Pakete aufgrund von Attributen (*Header-Felder, MPLS-Label, ...*) und internen Variablen behandelt werden sollen (*weiterleiten, verwerfen, zu Controller senden, ...*). Über eine API kann eine SDN-Anwendung programmiert werden.

Vorteil: Einsatz von preiswerter Hardware auf Data Plane
Vorteil: Vermeidung der „Protokollsuppe“
Vorteil: Zentrale Sicht auf das Netzwerk
Vorteil: Flexible Programmierung von Netzen
Vorteil: Offene Schnittstellen
Vorteil: Open Source in Netzwerken
Vorteil: schnelle Reaktion auf Netzsituationen

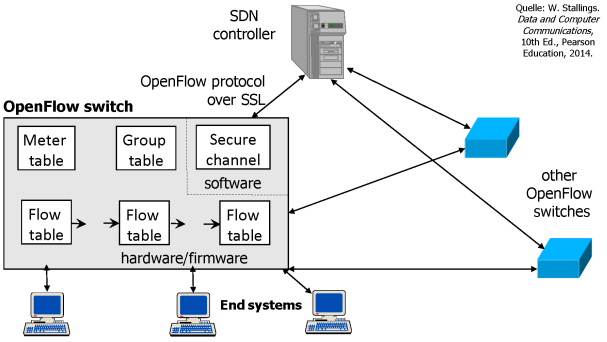
SDN Architektur



Beispiel OpenFlow

Software-Defined Networking (SDN)

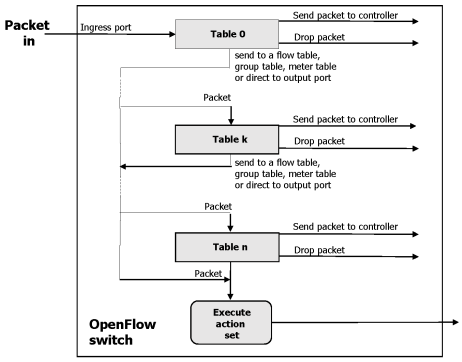
- Architektur eines OpenFlow-Switch:



Quelle: W. Stallings. *Data and Computer Communications*, 10th Ed., Pearson Education, 2014.

Software-Defined Networking (SDN)

- Behandlung eines Pakets in einem OpenFlow-Switch:

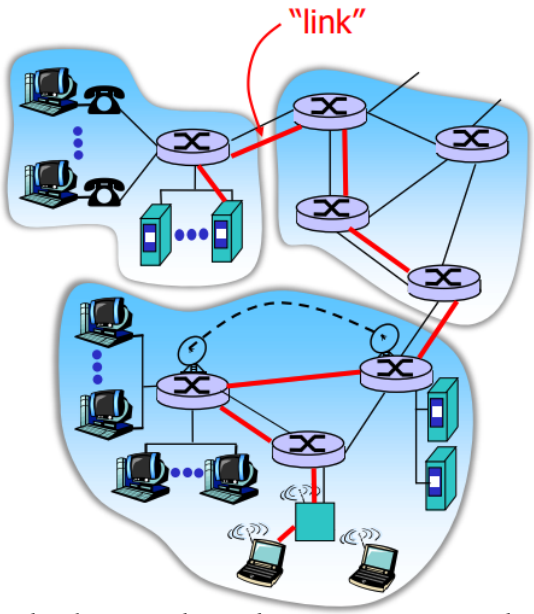


Quelle: W. Stallings. *Data and Computer Communications*, 10th Ed., Pearson Education, 2014.

KAPITEL 5

SICHERUNGSSCHICHT

Die Aufgabe der Sicherungsschicht ist der Transfer der Rahmen zwischen Knoten und Links. Hosts und Router sind dabei Knoten, Kommunikationskanäle Links und Dateneinheiten sind Rahmen (*frames*). Die Dienste der Sicherungsschicht sind:



— Das Ein- und Auspacken von Datagrammen der Netzwerkschicht in Rahmen.

— Adressierung: Rahmen enthalten eine **physikalische Adresse** der Knoten

— Datensicherung

— Medienzugriff (Medium Access, MAC)

— eventuell Flusskontrolle

Die Funktionalität befindet sich meistens im Netzwerk**adapter** beziehungsweise in der Netzwerkkarte. Diese enthalten die Schnittstellen zum Systembus des Knotens und zum Netzwerk. Die Funktionen können in HW/SW oder im FPGA implementiert sein. Es gibt zwei Varianten von Ein-/Ausgabe:

— **Programmed I/O (PIO)**: Hierbei transferiert die CPU Daten zwischen dem Speicher und Adapter unter Verwendung von Statusregistern und Unterbrechungen (siehe GRa)

— **Direct Memory Access (DMA)**: Hierbei schreibt und liest der Adapter selbst (siehe GRA)

Der Gerätetreiber für die Adapter finden sich im Betriebssystem.

Einführung

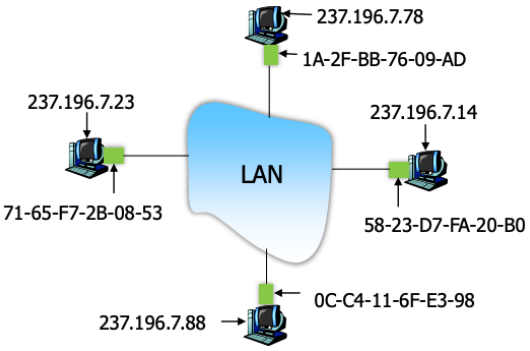
- Entwicklung von LANs, MANs und PANs (u.a.)
  - Local Area Networks (LANs)
    - klassisches Ethernet mit Bustopologie, Koaxialkabel, IEEE 802.3
    - Token Ring mit Ringtopologie, IEEE 802.5
    - Switched Ethernet (Stern- und Baumtopologie, Hubs, Switches, Twisted Pair, Glasfaser), IEEE 802.3
    - Drahtlose LANs, IEEE 802.11
  - Metropolitan Area Networks (MANs)
    - Fiber Distributed Data Interface (FDDI): Doppelring, ISO 9314
    - Resilient Packet Ring (RPR): optischer Doppelring, IEEE 802.17
    - Metropolitan Ethernet (Metro Ethernet Forum)
    - Worldwide Interoperability for Microwave Access (WiMAX): Fixed (Versorgung von Hotspots, drahtloser Hausanschluss), Mobile (Versorgung mobiler Nutzer), IEEE 802.16
    - Mobilfunk gemäß Long Term Evolution (LTE), ggw. 5. Generation in Vorbereitung (5G)

Einführung

- Personal Area Networks (PANs)
  - Bluetooth, IEEE 802.15.1: „Kabelersatz" für die Verknüpfung von Geräten, Daten- und Sprachkanäle
  - ZigBee, IEEE 802.15.4: preiswerter, geringe Datenraten, lange Batterielebensdauern, kleine Codegröße, für Heim-, Gebäude-, Industrieautomatisierung, eingebettete Geräte
  - Ultrabreitband (Ultra-Wideband, UWB): große Datenraten (> 500 Mbps) über kurze Entfernungen, Hindernisse (wie Wände) können durchdrungen werden, IEEE 802.15.3a, 802.15.4a
- Außerdem
  - Near Field Communication (NFC)
  - Sensornetze
  - Powerline
  - Satellitenkommunikation
  - ...

5.1 Adressierung

Die **physikalische Adresse** (oder auch MAC-Adresse, LAN-Adresse) besteht aus 48 Bit (6 Byte oder 12 Hexadezimalziffern). Sie ist in den ROM des jeweiligen Adapters eingebrannt. Die IEEE verwaltet diese, sie können von Herstellern gekauft werden. **Wichtig:** Sie ist global eindeutig und besitzt keine Strukturierung. Sie entspricht nicht der IP-Adresse.



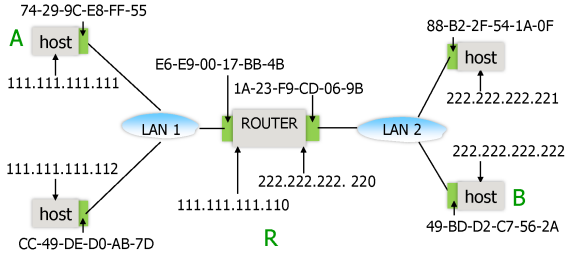
Herausfinden der MAC-Adresse über das ARP Um zu einer gegebenen IP-Adresse die zugehörige physikalische Adresse herauszufinden verwendet man das *Address Resolution Protocol (ARP)*. Dabei besitzt jeder Knoten eine ARP-Tabelle mit Zuordnungen (IP-Adresse, physikalischer Adresse, TTL). Dabei stellt die *Time To Live (TTL)* die Dauer der Gültigkeit des Eintrags dar.
Angenommen Knoten A möchte nun an Knoten B einen Rahmen schicken, kennt aber nur dessen IP-Adresse. Dann sendet A eine ARP-Anfrage als Broadcast (Adresse `FF-FF-FF-FF`) mit seiner physikalischen Adresse und der IP-Adresse von B. B erkennt sich als Ziel an IP-Adresse in der ARP-Anfrage und sendet in der ARP-Antwort seine physikalische Adresse an die physikalische Adresse von A. A speichert daraufhin die Zuordnung der Adressen von B in seiner ARP-Tabelle. Somit gilt Autonomität und der „Soft State“.

Beispiel

Adressierung

Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

- Beispiel
  - A in LAN1 sendet IP-Datagramm von A über R zu B in LAN2
  - A kennt IP-Adresse von B
  - R benötigt für jedes LAN eine ARP-Tabelle



Adressierung

- A erzeugt Datagramm mit IP-Quelladresse A und IP-Zieladresse B
- A findet R in seiner Routingtabelle
- A benutzt ARP um die physikalische Adresse des Adapters von R an LAN1 zu finden
- A erzeugt einen Rahmen mit sich als physikalischer Quelladresse, physikalische Zieladresse ist der Adapter von R an LAN1 (die IP-Zieladresse im eingepackten Datagramm bleibt B!)
- der Adapter von A sendet den Rahmen auf LAN1
- R's Adapter in LAN1 empfängt den Rahmen und packt das Datagramm aus, liest die IP-Zieladresse B, findet in der Routingtabelle heraus, dass B in LAN2 ist
- R benutzt ARP um die physikalische Adresse von B zu finden
- R erzeugt einen Rahmen mit seinem Adapter in LAN2 als physikalischer Quelladresse und B als physikalischer Zieladresse (die IP-Quelladresse bleibt A!)
- R's Adapter in LAN2 versendet den Rahmen
- B's Adapter empfängt den Rahmen und liefert das Datagramm aus

5.2 Datensicherung

Es gibt verschiedene Fehlercharakteristika. Beispielsweise können thermisches Rauschen, elektromagnetische Einstrahlung (Übersprechen, Motoren und Zündanlagen), sowie radioaktive Einstrahlung alle für Fehler verantwortlich sein. Typische Bitfehlerwahrscheinlichkeiten sind zwischen 10^{-3} bei Funkverbindungen und 10^{-12} bei Glasfaserverbindungen. Oftmals treten Fehler auch als **Burts** auf. Um Daten zu sichern bietet sich nun eine **Fehlererkennung** sowie **-korrektur** an.

Fehlererkennung Den Nutzdaten werden hierbei zusätzlich noch Prüfdaten zugefügt, um Fehler beim Empfänger zu erkennen und eine Sende Wiederholung zu veranlassen. Beispiele hierfür sind die **Paritätsprüfung** und die **zyklische Redundanzprüfung**.

Fehlerkorrektur Die Nutzdaten werden hierbei redundant kodiert, der Empfänger kann dann Fehler erkennnen und korrigieren. Bei n Bit Nutzdaten in m Bit gesendete Daten muss $m >> n$ gelten. Dies wird beispielsweise mit Block- und Faltungscodes erreicht. Die Redundanz hierbei ist deutlich größer als für Fehlererkennung und wird typisch in stärker gestörten Kanälen und bei hohen Latenzanforderungen verwendet. Es gibt zudem noch hybride Verfahren wie beispielsweise Hybrid ARQ.

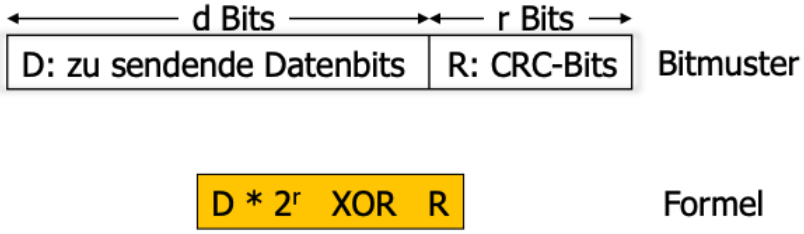
Zyklische Redundanzprüfung (CRC) Bei der zyklischen Redundanzprüfung (CRC) werden Bitfolgen als Koeffizienten eines binären Polynoms interpretiert, so entspricht die Bitfolge

$$(b_n b_{n-1} \dots d_1 d_0)$$

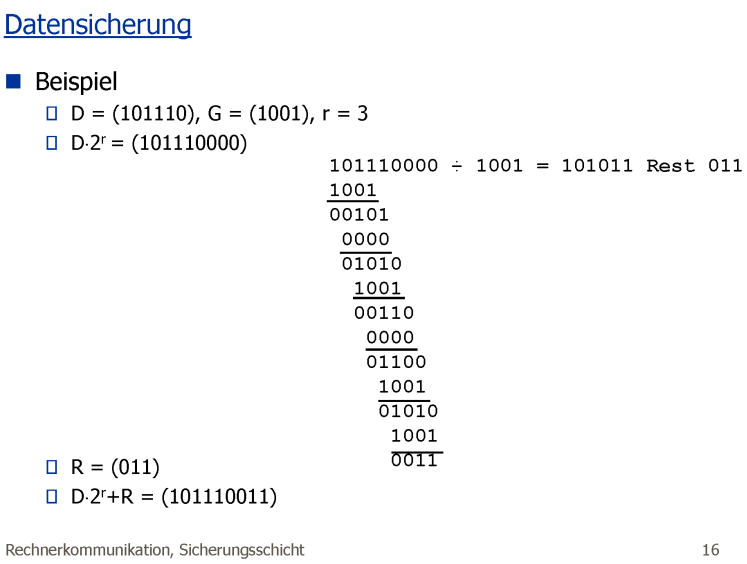
dem Polynom

$$\sum_{i=0}^{n} b_i x^i.$$

Algebraisch gesehen befinden wir uns im Polynomring über $\mathbb{Z}_2$. Es gibt dabei verschiedene Operationen. Die **Addition** und **Subtraktion** ist realisiert durch ein XOR, wohingegen die **Multiplikation** mit einer Potenz x^n einer Schiebeoperation um n Bit entspricht. Die **Division** wird analog zu GTI durchgeführt.
Bei Nutzdaten D mit d Bit und Prüfdaten R mit r Bit wird das Wort (D, R) gesendet. Es gibt dann ein Generatorpolynom G mit $r+1$ Bit. Der Sender wählt nun R so, dass (D, R) **ohne** Rest durch G teilbar ist, dazu ist R der Rest der Operation $D \cdot 2^r \div G$. Der Empfänger teilt (D, R) durch G, dabei ist kein Fehler in der Übertragung aufgetreten, falls der Rest 0 ist.



CRC — Beispiel



CRC — Implementierung Die euklidische Division kann leicht und effizient durch Schieberegister und XOR-Gatter in Hardware durchgeführt werden. Es gibt dann auch verbreitete genormte Generatorpolynome, wie zum Beispiel

— CRC-8 (ITU-T): $x^8 + x^2 + x + 1$

— CRC-16 (IBM): $x^{16} + x^{15} + x^2 + 1$

— CRC-16 (ITU-T): $x^{16} + x^{12} + x^5 + 1$

— CRC-32 (IEEE 802.3): $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$.

CRC — Eigenschaften bei der Fehlererkennung Man erkennt alle Einzelbitfehler, wenn die Koeffizienten von x^r und x^0 gleich Eins sind. Man erkennt alle Doppelbitfehler, wenn G einen unzerlegbaren Faktor mit mindestens drei Termen enthält. Man erkennt jede ungerade Bitfehlerzahl, falls G den Faktor $x + 1$ enthält. Jeder Fehlerburst der Länge $< r$ wird erkannt.

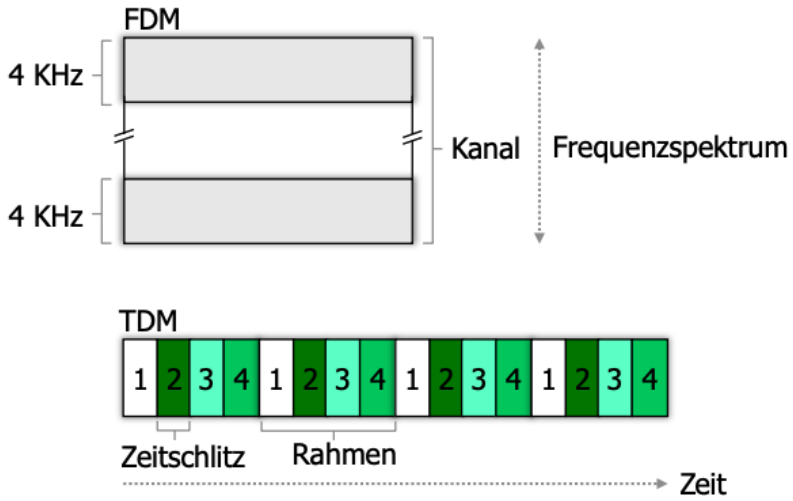
5.3 Medienzugriff

Es gibt zwei unterschiedliche Arten von Verbindungsstrategien. Entweder kann man **Punkt-zu-Punkt-Verbindungen** verwenden. Bei diesen greifen nur die zwei Endpunkte auf das Medium zu beispielsweise über das PPP zwischen Router und ISP über die Telefonverbindung oder die Verbindung zweier Router über ein anderes Netz. Es ist **keine** komplizierte Koordination notwendig.
Bei Medien mit Mehrfachzugriff benötigt man eine verteilte Koordination des Medienzugriffs (*Medium Access Control*, *MAC*). Beispiele sind ein gemeinsamer Bus oder Funkkanal sowie der Internetzugang über Kabel.

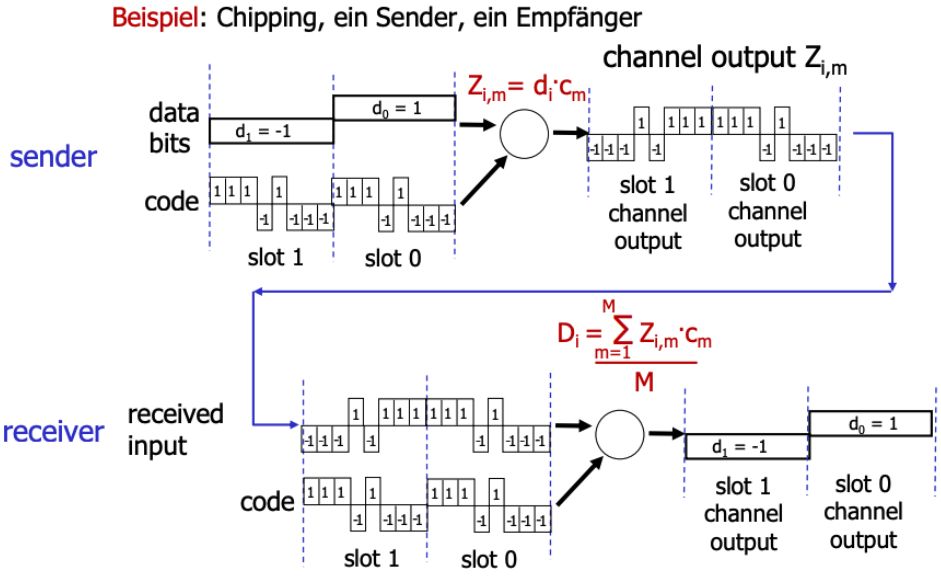
5.3.1 Feste Kanalaufteilung

Bei der festen Kanalaufteilung wird das Medium durch ein geeignetes Multiplexverfahren in feste Kanäle für Knotenpaare aufgeteilt. Bekannte Verfahren sind das Frequenz-, Zeit-, Raum- oder Codemultiplexverfahren. Diese Variante war vor allem für Sprachkommunikation früher

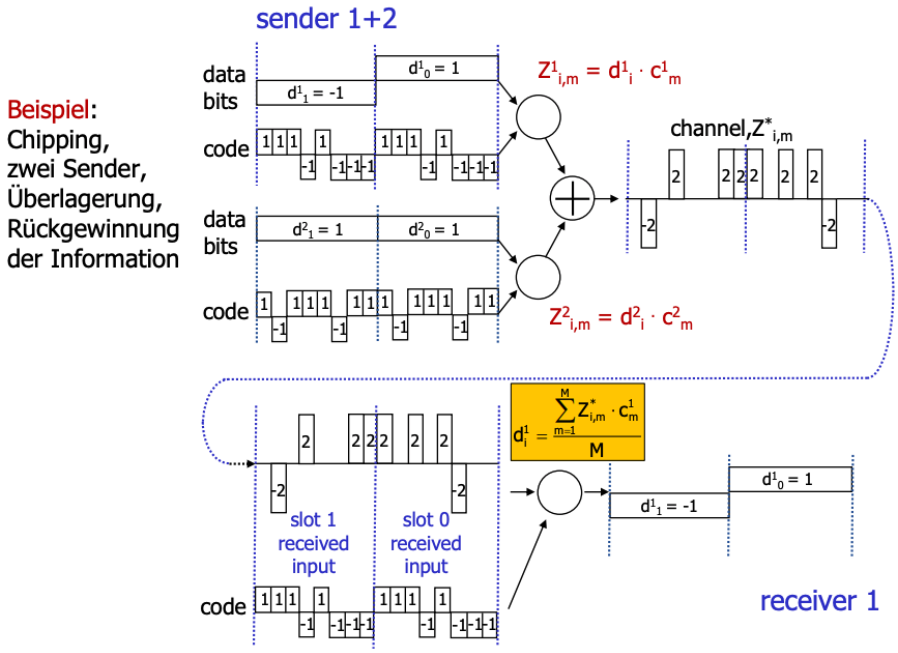
verbreitet. **Nachteil:** Daten werden typischerweise sporadisch versendet, es kommt zu einer ineffiziente Nutzung des Mediums.



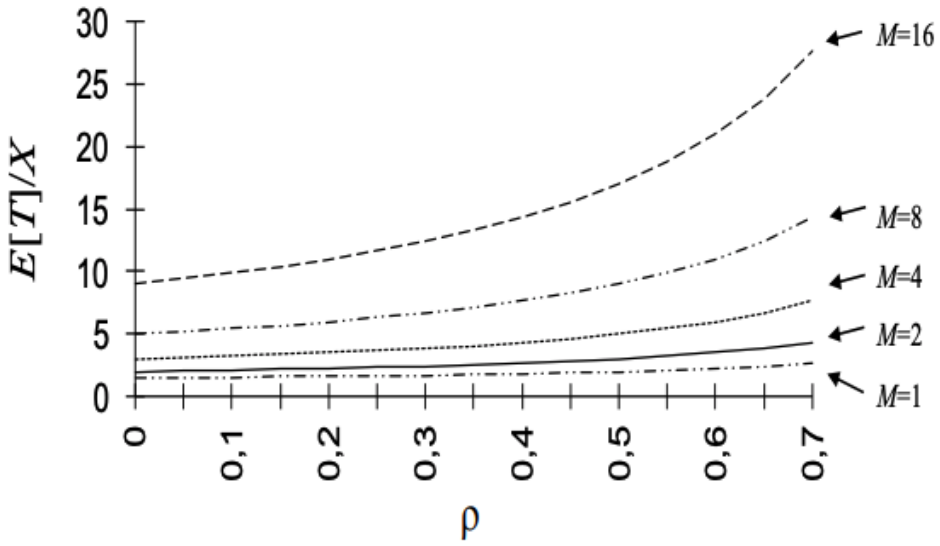
Codemultiplexverfahren Beim Codemultiplexverfahren wird die **Spreiztechnik** verwendet, bei der der Sender jedes Bit mit einem **Chipping-Code** multipliziert. Dadurch erzeugt er ein Signal mit höherer Frequenz und sendet dieses dann auf das Medium.[1] Die gespreizten Signale überlagern sich auf dem Medium woraus der Empfänger mit dem Chipping-Code das einzelne gesendete Signal extrahieren.
In einer anderen Variante (**Frequenzsprungverfahren**) springt der Sender während des Sendens eines Bits oder nach mehreren Bit zwischen verschiedenen Frequenzen, was ebenfalls die Überlagerung vieler Signale auf dem Kanal erlaubt. Durch Kenntnis des Sprungmusters kann dann das Signal empfangen werden. Seinen Ursprung hat diese Technik im Militär, ist aber durch den Mobilfunk bekannt geworden.



[1] Er benutzt dazu das volle Spektrum und die gesamt Zeit.



Effizienz fester Kanalaufteilung Wir wissen, dass die Datenkommunikation **bursty** (schubartig) ist, wodurch eine feste Kanalaufteilung ineffizient ist. Als Beispiel wird ein Medium mit jeweils gleicher Last ρ bei einer Aufteilung in M Kanäle betrachtet, wobei die Zwischenankünfte und die Paketlänge exponentiell verteilt sind. Die mittlere Wartezeit durch Pufferung kann mit der Warteschlangentheorie berechnet werden, sie wächst mit der Kanalzahl:



5.3.2 Zufallszugriff

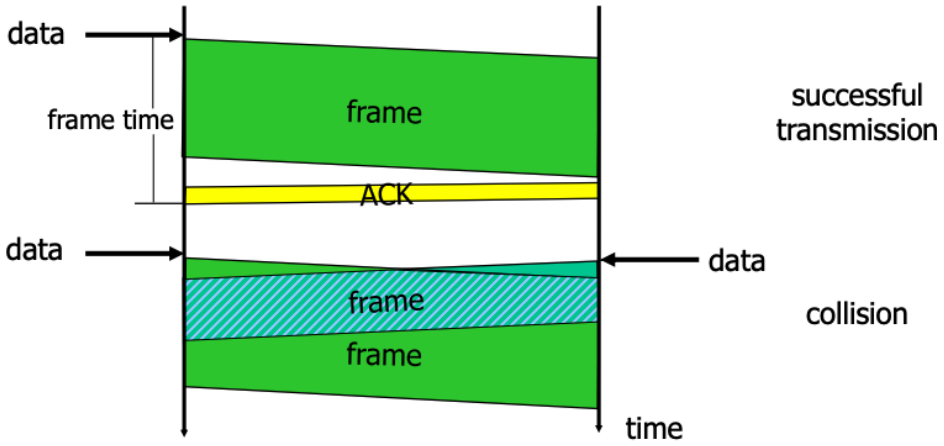
Bei Zufallszugriffsverfahren greifen Stationen zufällig auf das Medium zu, wobei eventuelle gleichzeitige Übertragungen (Kollisionen) beachtet werden müssen. In der Urform wird dies von ALOHA umgesetzt, daraus hat sich MAC bei Ethernet und WLAN abgeleitet.
Wenn ein Knoten einen Rahmen zum Senden hat, so sendet er mit der vollen Bitrate des Mediums. Wenn mehrere Knoten gleichzeitig senden, überlagen sich die Signale auf dem Medium und *zerstören sich normalerweise gegenseitig*. Es kommt zu einer Kollision, welche durch eine Sendewiederholung behoben wird.
Grundidee: Kollisionen treten *bei schwacher Last* sehr selten auf.
Es gibt unterschiedliche Verfahren zur Vermeidung und Erkennung von Kollisionen: ALOHA

und slotted ALOHA, sowie das Carrier Sense Multiple Access (CSMA) mit Collision Detection (CSMA/CD in Ethernet) oder Collision Avoidance (CSMA/CA in WLANs).

5.3.2.1 ALOHA

ALOHA ist in den 70er Jahren zur Vernetzung von Rechnern der Uni Hawaii genutzt worden und hat einen gemeinsamen Funkkanal für alle Knoten. Das Verfahren beschreibt, wenn die MAC-Schicht eines Knotens von der Netzwerkschicht ein Datagramm erhält, wird der Rahmen **sofort gesendet**. Wenn der Empfänger ihn fehlerlos erhält, so sendet er eine **positive Bestätigung (ACK)** zurück. Wenn nach einem **Timeout** kein ACK zurückkommt, wartet der Sender eine *zufällige Wartezeit (**Backoff**)* und wiederholt dann das Senden. Kollisionen werden damit wie Fehler bei der Fehlerkontrolle behandelt. **Wichtig:** Das Protokoll ist einfach, verteilt und es gibt keine Absprachen zwischen Knoten.

Beispielhafter Ablauf Hier ist das Produkt aus Bitrate und Verzögerung klein ($a < 1$), deswegen die große Ähnlichkeit zu Stop-and-Wait.

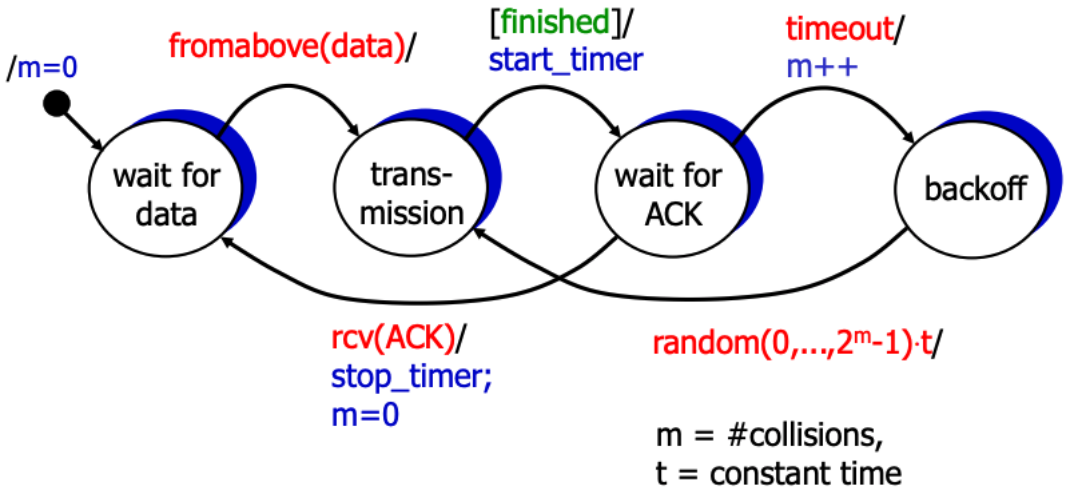


Binärer exponentieller Backoff Die Idee ist es die Backoffzeit an die aktuelle Last anzupassen. Bei niedriger Last sind vermutlich nur wenige Knoten an der Kollision beteiligt, deswegen reicht eine Auswahl von K — der Backoffkonstante — aus wenigen Möglichkeiten. Bei einer höheren Last gibt es mehr kollidierende Knoten, weswegen man K aus einer Vielzahl an Möglichkeiten auswählen möchte und deswegen auch eine größere mittlere Backoffzeit hat.
Die m-te Kollision macht deswegen eine *gleichverteilte Auswahl* von K aus

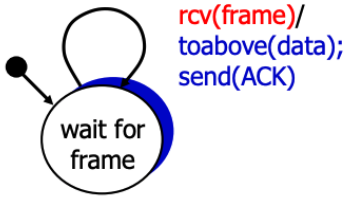
$$\{0, 1, \ldots, 2^m - 1\}.$$

Die Backoffzeit berechnet sich dann durch $K \cdot t$, wobei t eine vorher feste Zeitkonstante ist. Nach einer maximalen Zahl M von Kollisionen bricht die MAC-Schicht dann ab und meldet einen Fehler an die Netzwerkschicht.

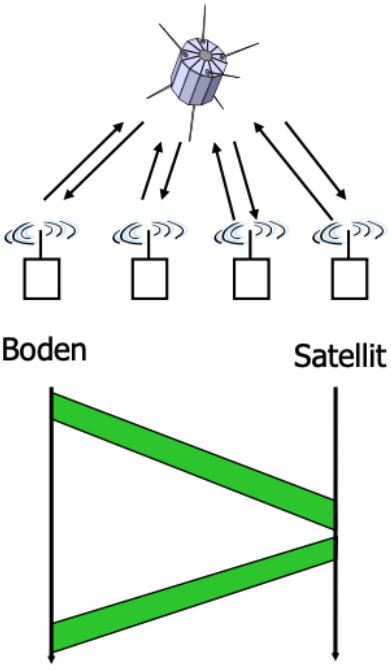
Statechart — Sender



Statechart — Empfänger

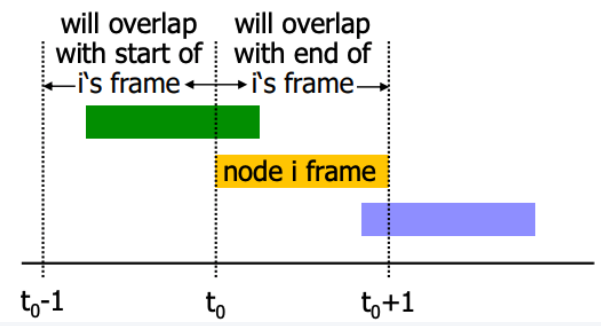


ALOHA über Satellit Bodenstationen haben keine Möglichkeit zur Koordination, deswegen senden sie in einem Uplink-Kanal an Satellit und der Satellit reflektiert in einem Broadcast-Downlink-Kanal. Die Kollisionen treten hierbei im **Uplink-Kanal** auf. Die Verzögerung ist sehr groß mit einer RTT von 0.27 Sekunden (!) bei einem geostationärem Satellit, weswegen das Produkt von Bitrate und Verzögerung auch sehr groß ist. Die sendende Bodenstation erkannt an den ausbleibenden ACKs Kollisionen.



Leistungsanalyse Es tritt eine Verschwendung durch Kollisionen und Backoffzeiten auf. Aus der Literatur ist eine einfache Analyse sehr bekannt, welche durch die Simulationen bestätigt. Dazu treffen wir vereinfachende Annahmen, mit welchen alle Rahmen konstante Länge besitzen, zum Senden wird eine Slotzeit benötigt, welche als Zeiteinheit verwendet wird.**Wichtig:** Jeder Knoten besitzt dabei die gleiche Wahrscheinlichkeit p, einen Rahmen zu senden (das erste Mal oder als Wiederholung) (**wesentliche Einschränkung**) Der Nah-/Ferneffekt ist ebenso unberücksichtigt, da bei Funkkanälen Signale bei Kollisionen nicht immer zerstört werden. Ebenso treten keine Verluste und Bitfehler auf.

Bedingung für eine Kollision



Wenn ein Knoten zum Zeitpunkt t_0 mit dem Senden beginnt und ein weiterer Knoten im Intervall $[t_0 - 1, t_0 + 1]$ das Senden beginnt, kommt es zu einer Kollision.

Seien nun N Stationen gegeben. Die Wahrscheinlichkeit, dass ein ausgewählter Knoten in einem beliebigen Slot **ohne Kollision** sendet ist

$$\underbrace{p}_{\text{Knoten sendet}} \cdot \underbrace{(1-p)^{N-1}}_{\substack{\text{kein Knoten sendet} \\ \text{in } [t_0-1,t_0]}} \cdot \underbrace{(1-p)^{N-1}}_{\substack{\text{kein Knoten sendet} \\ \text{in } [t_0,t_0+1]}}.$$

Die Wahrscheinlichkeit, dass ein **beliebiger** Knoten in einem beliebigen Slot ohne Kollision sendet ist

$$N \cdot p \cdot (1-p)^{2(N-1)} = \text{durchschn. Anzahl erfolgr. Slots} = S.$$

Sei $G = Np$ der **Offered Load**, sprich die Rate von Sendeversuchen von Rahmen in einem Slot. Damit ist $p = G/N$. Durch das Einsetzen in S ergibt sich

$$S = G \cdot \left(1 - \frac{G}{N}\right)^{2(N-1)}.$$

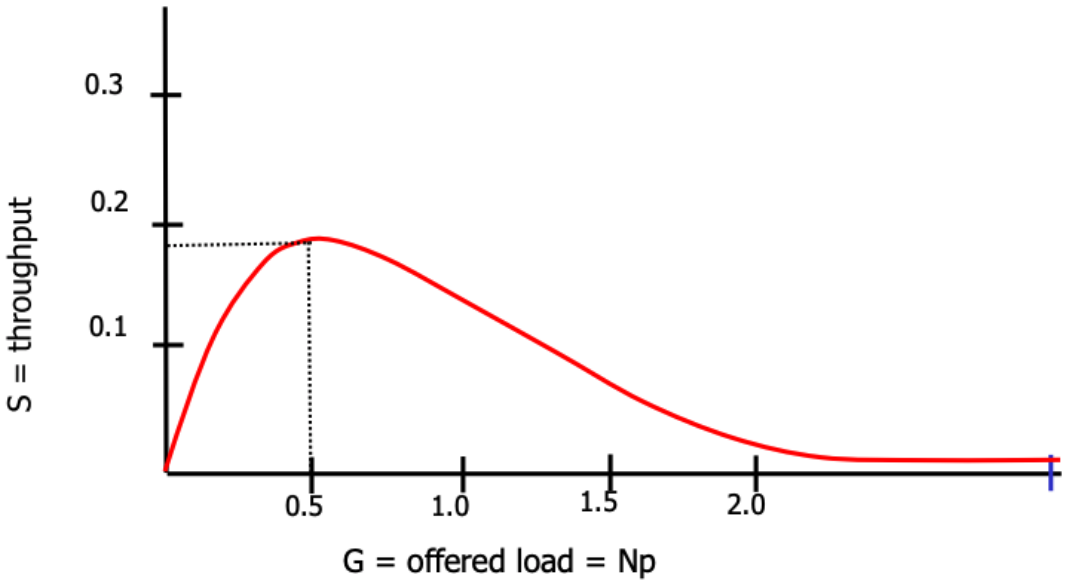
Es gilt eine schnelle Konvergenz für große N, denn es gilt die Definition von $\exp(x)$ über den Limes. Damit ergibt sich für große N

$$\lim_{N\to\infty} S = \lim_{N\to\infty} G \cdot \left(1 - \frac{G}{N}\right)^{2(N-1)} = G \cdot e^{-2G}.$$

Das Maximum S_{max} folgt dann aus der Nullstelle der Ableitung

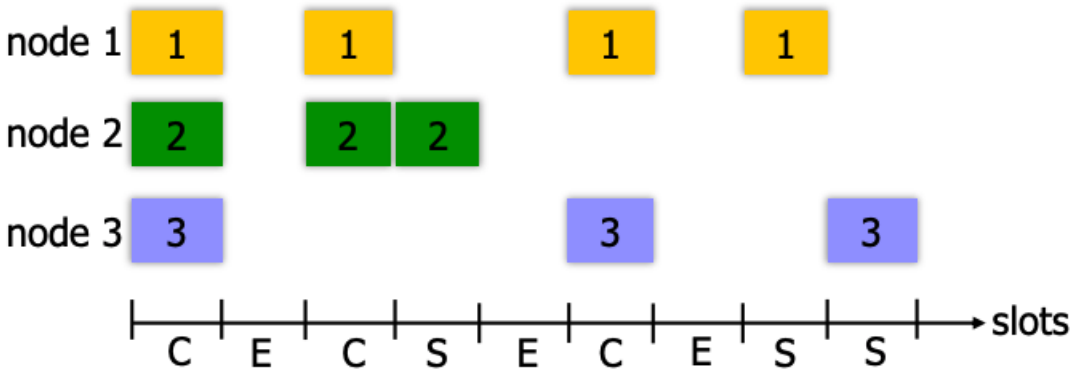
$$\frac{d}{dG}\left(Ge^{-2G}\right) = (1-2G) \cdot e^{-2G} \stackrel{!}{=} 0$$

mit $S_{max} = e/2 \approx 0,18$ bei $G = 1/2$.
Wichtig: Selbst bei optimaler Einstellung der Last kann maximal **18%** Durchsatz erreicht werden!



5.3.2.2 Slotted ALOHA

Bei *Slotted ALOHA* synchronisieren alle Knoten ihre Slots (beispielsweise durch ein zentrales Zeitsignal). Der Sendebeginn ist dabei nur zu Beginn eines Slots, das Kollisionsintervall verkürzt sich somit auf einen Slot.



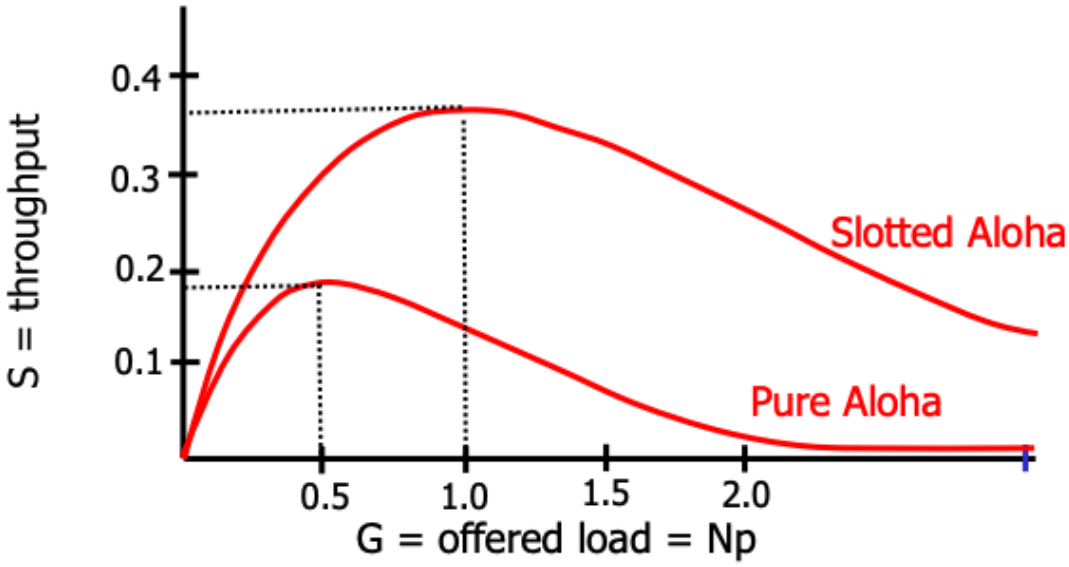
Die Wahrscheinlichkeit, dass dann von allen Knoten ein beliebiger Knoten in einem beliebigen Slot ohne Kollision sendet ist

$$N \cdot p \cdot (1-p)^{N-1}.$$

Identische Überlegungen führen zu

$$S = G \cdot e^{-G}$$

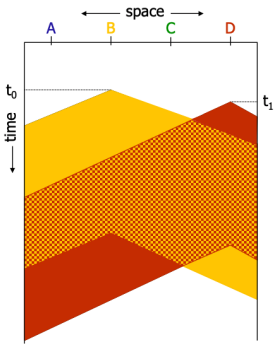
und $S_{max} = 1/e \approx 0{,}37$ bei $G = 1$. Im Vergleich:



5.3.2.3 CSMA

Beim *Carrier Sense Multiple Access (CSMA)* Verfahren prüfen die Knoten **vor** dem Senden, ob das Medium belegt ist (*listen before talking*). Die reduziert dann die auftretenden Kollisionen. Die Voraussetzung dafür ist, dass die Ausbreitungsverzögerung kleiner ist als die Rahmensendezeit.[2] **Wichtig:** Kollisionen sind hierbei **immer noch möglich**. Wenn ein anderer Knoten startet, bevor sich das Signal auf dem Medium zu ihm ausgebreitet hat.

CSMA — allgemein Wenn die MAC-Schicht eines Knotens von der Netzwerkschicht ein Datagramm erhält, **überprüft sie das Medium** (*listen before talking*) und sendet den Rahmen **nur**, wenn es **frei** ist. Andernfalls wird gewartet. Wenn der Empfänger ihn fehlerlos erhält, so sendet er eine positive Bestätigung (ACK) zurück. Wenn nach einem **Timeout** kein ACK zurückkommt, so wartet der Sender eine zufällige Wartezeit (**Backoff**) und wiederholt dann das Senden.
Es gibt drei Varianten:



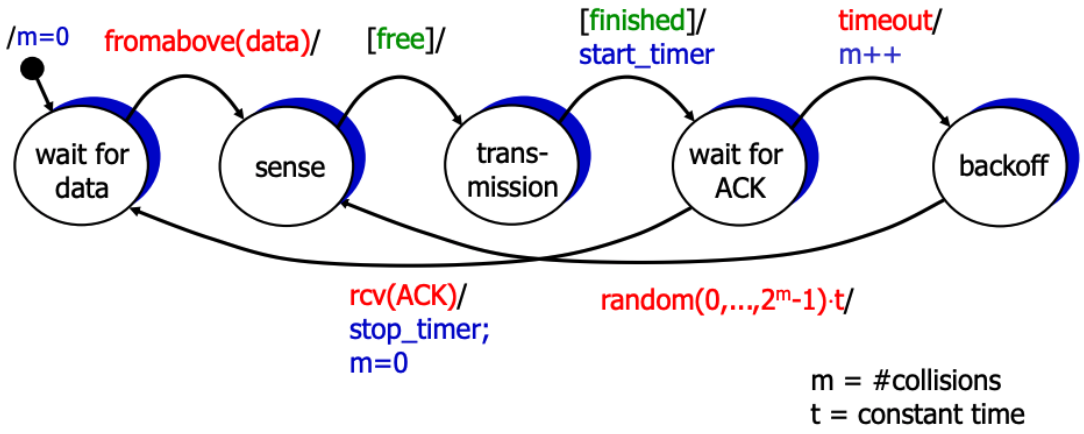
— **1-persistentes CSMA** Wenn das Medium belegt ist, wartet der Knoten bis es frei ist und sendet dann sofort. Es führt zu geringen Wartezeiten, schafft aber mögliche neue Kollisionen, wenn mehrere Knoten auf ein freies Medium warten.

— p-**persistentes CSMA** Wenn das Medium belegt war und wieder frei ist, sendet der Knoten jeweils mit Wahrscheinlichkeit p oder wartet noch einen Slot mit Wahrscheinlichkeit $1 - p$. Diese Lösung ist ein Kompromiss aus den anderen beiden.

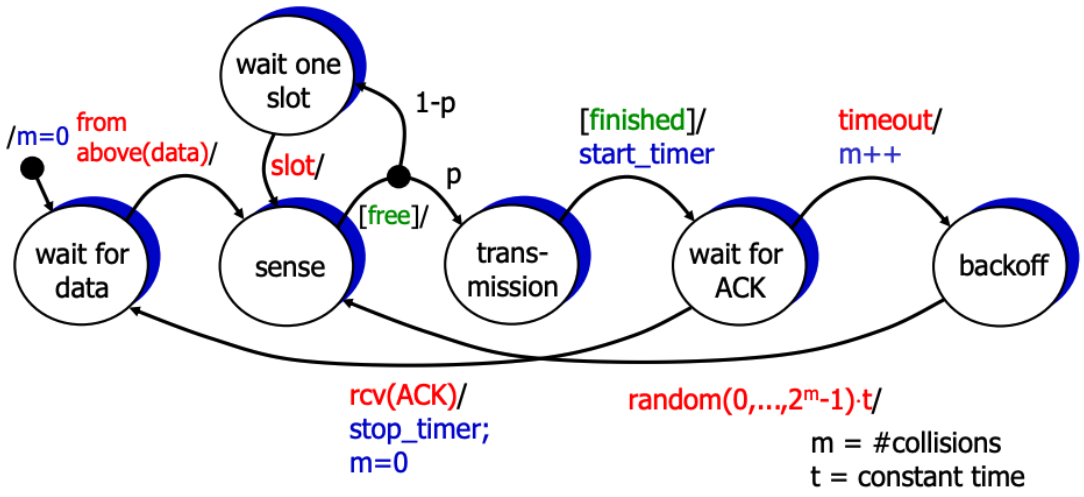
— **nicht-persistentes CSMA** Wenn das Medium belegt ist, geht der Knoten in Backoff. Das führt zu weniger Kollisionen, aber gleichzeitig zu längeren Wartezeiten.

[2] Andernfalls ist die Information des belegten Kanals veraltet und das Verfahren sinnlos!

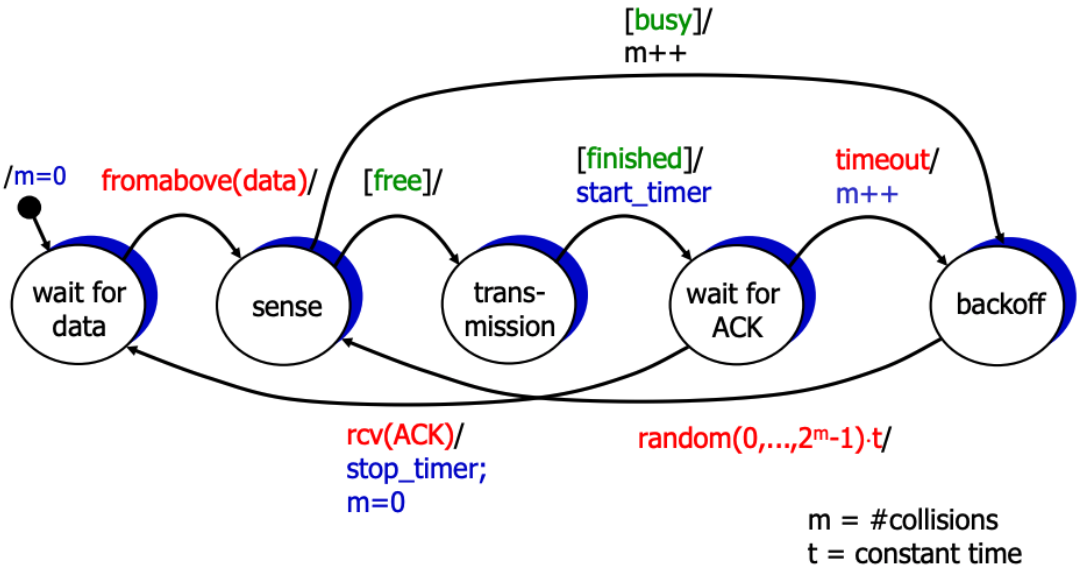
CSMA Statechart: 1-persistent



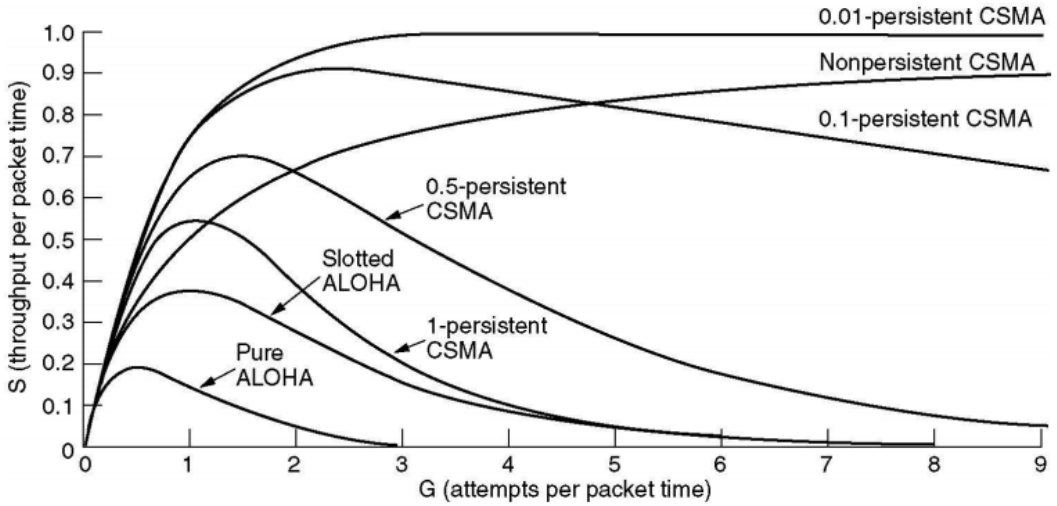
CSMA Statechart: p-persistent



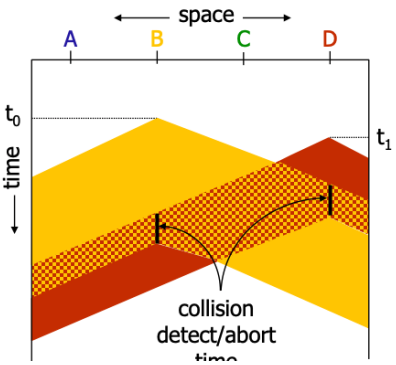
CSMA Statechart: nicht-persistent



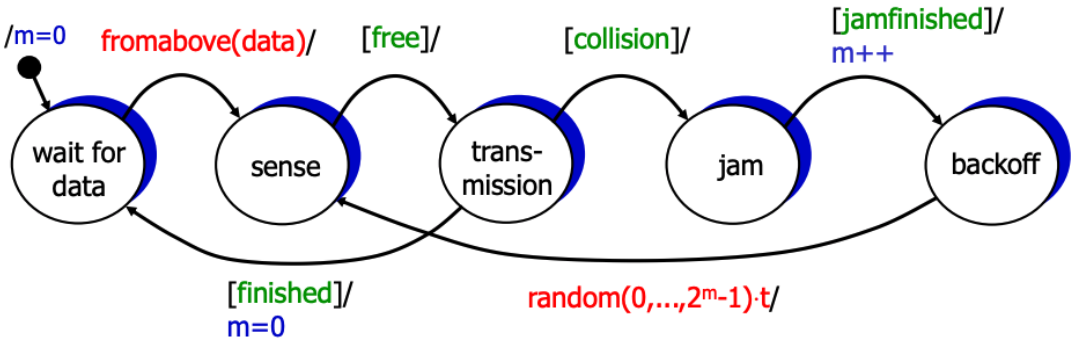
Durchsatz für verschiedene CSMA-Verfahren Die Herleitung findet sich in [TW14] und ist hier irrelevant.



Kollisionserkennung mit CSMA/CD Die Knoten besitzen hierbei Hardware, um während des Sendens Kollisionen zu erkennen (*listen while talking*). Nach der Kollisionserkennung wird das Senden **abgebrochen** (weniger Verschwendung) und ein **Jamming-Signal** wird gesendet, damit alle Knoten die Kollision sicher erkennen. **Wichtig:** Es gibt **keine ACKs** und ist mit allen CSMA-Varianten kombinierbar.



CSMA/CD Statechart: 1-persistent

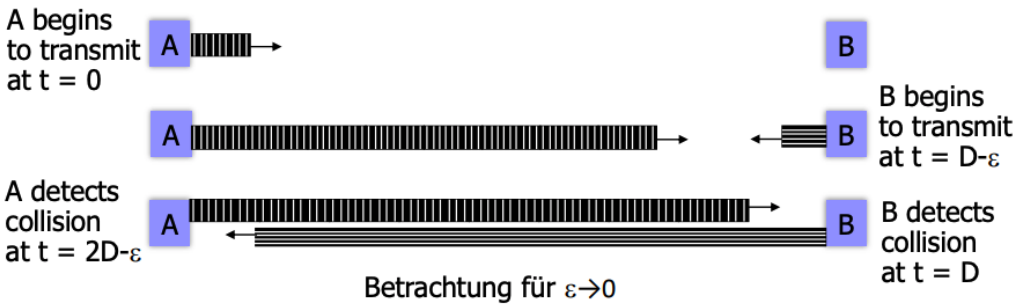


m = #collisions
t = constant time

CSMA/CD — minimale Rahmengröße Sei D die maximale Ausbreitungsverzögerung zwischen zwei Knoten, so dauert es höchstens 2D bis eine Kollision von allen Knoten bemerkt wird. Bei einer Bitrate R **muss** die minimale Rahmengröße L groß genug sein, so dass

$$\frac{L}{R} > 2D,$$

damit eine Kollision **sicher erkannt** wird.



CSMA/CD — Leistungsanalyse Es kommt zu einem Wechsel von Sende-, Leerlauf- und Wettbewerbsphasen. Dabei dauert die Sendephase L/R Zeiteinheiten. Kollisionen werden nach Intervallen der Länge 2D behoben, dabei wird die Zeit für das Jamming vernachlässigt. Die Wettbewerbsphase wird in Slots der Länge 2D eingeteilt. Hierbei gibt es N Knoten, von denen **ein jeder** mit einer Wahrscheinlichkeit p versucht in einem Slot zu senden. Der Wettbewerb ist beendet, wenn genau ein Knoten sendet. Es ergibt sich eine Erfolgswahrscheinlichkeit von

$$p_{\text{Erfolg}} = N \cdot p \cdot (1-p)^{N-1}.$$

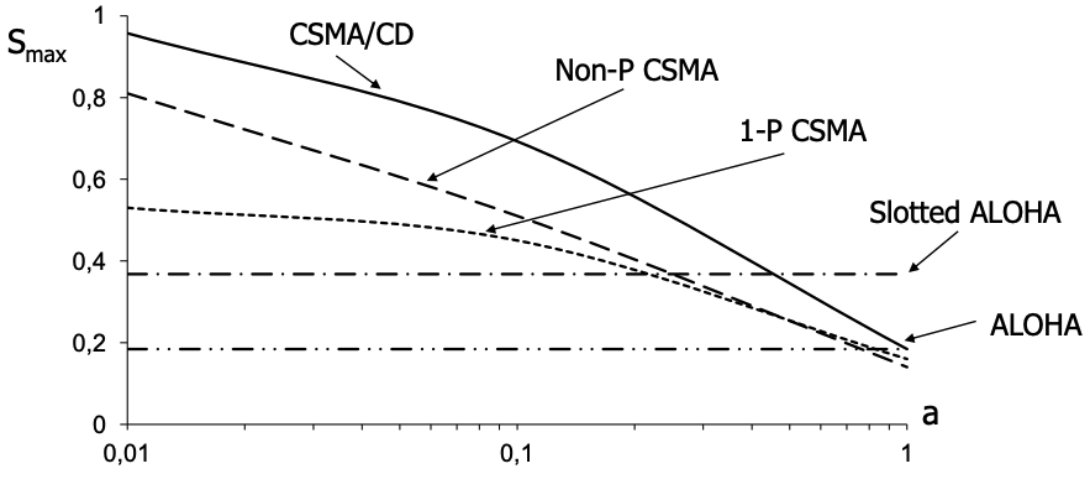
Wie bei Slotted ALOHA kann man wieder herleiten, dass für $p = 1/N$ die Erfolgswahrscheinlichkeit maximal wird, dann ist $p^{\max}_{\text{Erfolg}} = 1/e$. Im Mittel ist ein Knoten also einmal pro $1/p^{\max}_{\text{Erfolg}} = e$ Slots erfolgreich.
Damit ist die Wettbewerbsphase im Mittel $e-1 \approx 1{,}718$ Slots lang. Für einen maximalen Durchsatz wird die Leerlaufphase auf Null gesetzt, womit das System nur zwischen Sende- und Wettbewerbsphasen alterniert. Nach dem Senden dauert es aber noch eine Ausbreitungsverzögerung D, bis das Ende überall bemerkt wird. Es ergibt sich

$$S_{\max} = \frac{\text{Sendephase}}{\text{Sendephase} + \text{Ausbreitung} + \text{Wettbewerbsphase}} = \frac{L/R}{L/R + D + (e-1)\cdot 2D},$$

also

$$S_{\max} = \frac{1}{1+[1+2\cdot(e-1)]\cdot a} \approx \frac{1}{1+4{,}4\cdot a}.$$

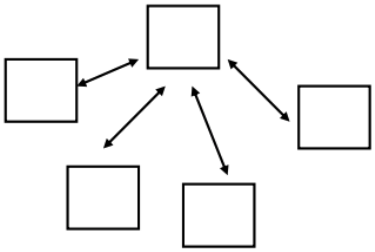
$a = RD/L$ ist die Kanalpuffergröße in Rahmen, wie bereits in der Transportschicht definiert.
Für kleines a ist also CSMA/CD am besten, der maximale Durchsatz von ALOHA und Slotted ALOHA ist unabhängig von a und somit besser für großes a:



5.3.3 Zyklische Zuteilung

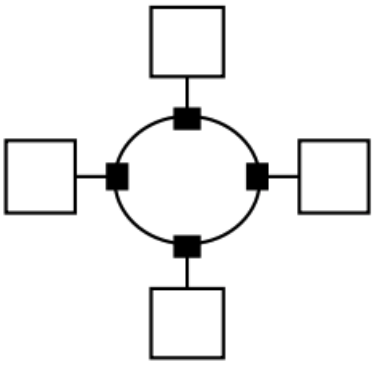
Bei der zyklischen Zuteilung erfolgt Polling durch einen zentralen Knoten. Verteilt tritt dann die Sendeerlaubnis durch ein rotierendes Bitmuster auf. Beispiele sind unter anderem der Token Ring, FDDI, USB und Profibus.

Variante 1 — Polling Beim **Polling** wird die Sendeerlaubnis den Knoten sukzessive zugewiesen durch (i) einen *zentralen Knoten*. (ii) einen *zufällig ausgewählten Knoten*. (iii) ein *verteiltes Protokoll* Die Reihenfolge kann hier zyklisch oder auch anders passieren[3]. Die **Zykluszeit** ist die Zeit bis die Sendeerlaubnis zu dem Knoten zurückkommt. Sie beträgt für jeden Knoten die Sendezeit für die Sendeerlaubnmis, die Ausbreitungszeiten, Verarbeitungszeiten und die Sendezeit für Daten.



Nachteil: Es kommt zu einem Overhead. Zusätzlich ist der zentrale Knoten ein „Single-Point-of-Failure“.

Variante 2 — Token Ring Beim **Token Ring** sind die Knoten ringförmig vernetzt. Jeder Knotenadapter hat einen Eingang und einen Ausgang mit zwei Modi:

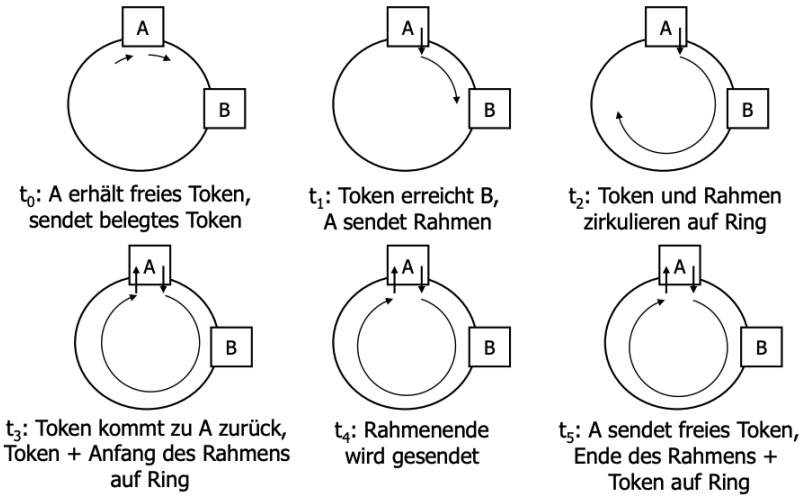


(1) **Listen Mode** — Die Bit vom Eingang werden mit Pufferung (typischerweise ein Bit) weitergereicht. Der Knoten erhält eine Kopie.

(2) **Transmit Mode** — Die Bit vom Eingang werden an den Knoten geleitet, die Bit zum Ausgang kommen vom Knoten.

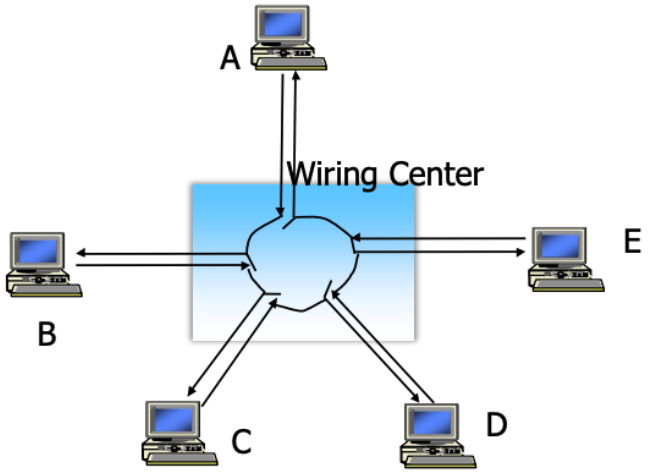
Ein Bitmuster (das *Token*) zirkuliert dabei auf dem Ring. Es gibt zwei Zustände **frei** und **belegt**, welche durch eine 0 oder 1 kodiert werden. Wenn ein Knoten ein freies Token empfängt und ein Sendewunsch vorliegt, verändert er das Token durch Umsetzen des Bits in belegt und sendet Daten. Der Empfänger erhält daraufhin die Daten. Nach einem Ringumlauf **entfernt der Sender** das belegte Token und die Daten wieder vom Ring und sendet das freie Token weiter.

Beispiel zu Variante 2

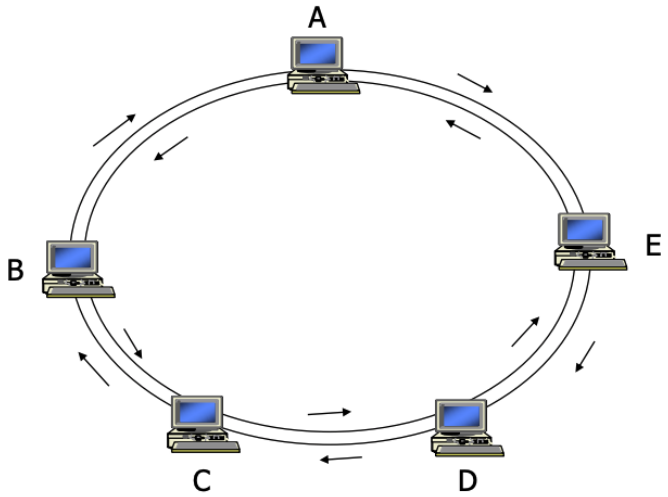


[3] beispielsweise prioritätsgesteuert

Verkabelung — Ringe mit sternförmiger Topologie

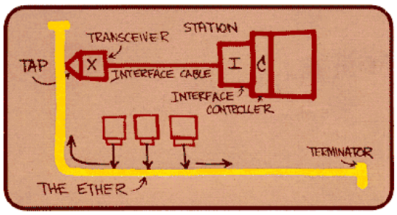


Verkabelung — gegenläufige Ringe Man macht dies für eine Fehlertoleranz bei Ausfällen.

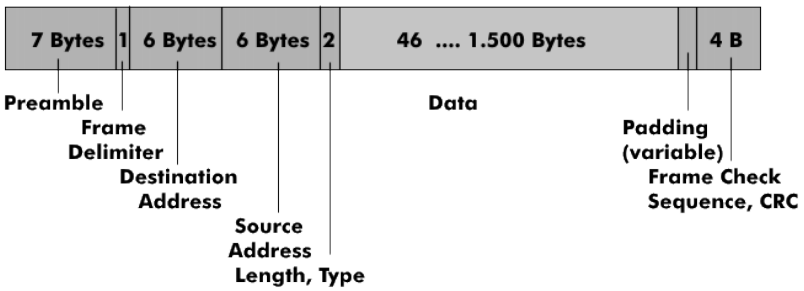


5.4 Ethernet

Ethernet ist die dominierende Technologie für kabelgebundene LANs, welche sich insbesondere gegen Token Ring oder ATM durchgesetzt hat. Das Rahmenformat blieb über die Jahre hinweg aus Kompatibilitätsgründen gleich.

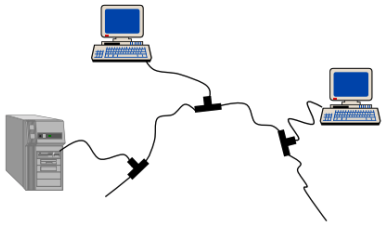


Rahmenformat Nach der Präambel von 8 Bytes, die sich aus 7 Bytes mit einem Muster von `10101010` zur Synchronisation der Taktfrequenz vom Empfänger mit dem Sender und ein Byte mit Muster `10101011` zur Anzeige des Beginns der Zieladresse zusammensetzt, folgt die **physikalische** Quell- und Zieladresse. Anschließend kommt der zwei Byte lange Type, eine Nummer für das Protokoll der Nutzdaten (IP, Appletalk, ...), und die eigentlichen Nutzdaten, welche eine Länge von mindestens 46 und maximal 1500 Byte haben. Abschließend folgt der CRC mit 4 Bytes (CCITT-32). **Wichtig:** Es ergibt sich somit eine Gesamtgröße von minimal 64 Byte (ohne Präambel).



Medienzugriff Bei Ethernet wird 1-persistentes CSMA/CD verwendet. Das Jam-Signal ist 32 bis 48 Bit lang und es kommt zu einem binären exponentiellen Backoff. Dabei ist nach der m-ten Wiederholung eine gleichverteilte Auswahl von K aus $[0, 2^n - 1]$ mit $n = \min\{m, 10\}$ zu treffen, nach 16 Versuchen wird aufgegeben. Es kommt zu Backoffzeiten von $K \cdot 512$ Bitzeiten.
Wichtig: Der Medienzugriff ist **verbindungslos** (es ist kein Handshaking erforderlich) und **unzuverlässig**, da keine Bestätigungen gesendet werden.
Die minimale Sendezeit muss **immer noch größer** sein als die zweifache maximale Ausbreitungszeit im Medium.

Ursprüngliche Bus-Topologie (10Base5) Hierbei ist der Bus ein Koaxialkabel, an welches Knoten über Transceiver angeschlossen sind. Die Datenrate beträgt 10 Mbps, die maximale Segmentgröße 500 Meter mit maximal 4 Repeatern. Somit ist die **maximale Entfernung** 2500 Meter.



Die maximale RTT (mit Zeit in den Repeatern) beträgt knapp 50µs. Die Zeit zum Senden eines Bits beträgt $1/10$µs.
Die minimale Rahmengröße ist 64 Bytes (512 Bit), dementsprechend ist die minimale Rahmensendezeit bei 51,2µs.
Der maximale Durchsatz ist immer noch durch

$$S_{max} = \frac{1}{1 + 4{,}4 \cdot a}$$

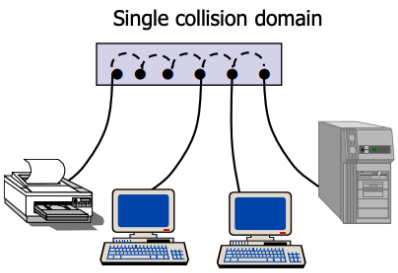
bestimmbar. Somit ergeben sich zum Beispiel:

$$\left.\begin{matrix} 64 \\ 640 \\ 6400 \end{matrix}\right\} \text{ Byte Rahmen} \longrightarrow a = \begin{cases} 4.8 \cdot 10^{-1} \longrightarrow S_{max} \approx 32\% \\ 4.8 \cdot 10^{-2} \longrightarrow S_{max} \approx 83\% \\ 4.8 \cdot 10^{-3} \longrightarrow S_{max} \approx 98\% \end{cases}$$

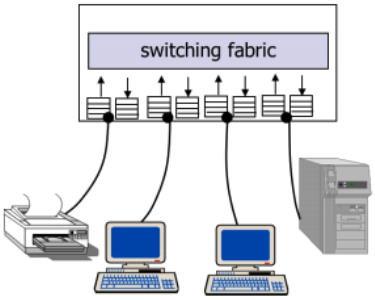
Repeater **Repeater** dienen dem Auffrischen von Signalen und operieren auf physikalischer Schicht, sind somit für die Medienzugriffsschicht *transparent*.

Bridge Eine **Bridge** verbindet zwei Ethernet-Segmente miteinander. Bei *jedem* Empfang eines Rahmens an einem Eingangsport wird entschieden an welchen Ausgangsport der Rahmen weitergeleitet wird und mittels CSMA/CD auf das Medium dieses Segments gegeben. **Wichtig:** Die Bridge operiert auf der Medienzugriffsschicht und schafft eine Aufteilung in verschiedene Kollisionsdomänen.

Sterntopologie mit Hub (10BaseT) Ein **Hub** ist ein Repeater mit vielen Ports. Er besitzt *keine* Pufferung, hat aber eine Managementfunktion. Alle Knoten werden an einen zentralen Hub angeschlossen, das Signal auf jedem eingehenden Port wird auf jeden ausgehenden Port weitergegeben. **Wichtig:** Es gibt nur eine Kollisionsdomäne, man verwendet weiterhin CSMA/CD.
Für die Verkabelung werden UTP-Kabel (*unshielded twisted pair*) mit RJ-45 Anschlüssen verwendet. Die Bitrate ist bei 10 Mbps, die Entfernung des Hub-Knotens ist maximal 100 Meter oder 200 Meter bei Kategorie 5-Kabeln.



Sterntopologie mit Switch Ein **Switch** ist eine Bridge mit vielen Ports, bei denen eine Pufferung an jedem Port stattfindet. Es hat voll-duplex Verbindungen und verwendet die Store-and-Forward-Techniken. **Wichtig:** Die Knoten führen noch CSMA/CD durch, **Kollisionen treten aber nicht mehr auf.** Es ist eine Kaskadierung, Heterogenität von Bitraten, sowie die Kombination mit Hubs möglich.



Fast Ethernet

Ethernet

- Fast Ethernet
  - Sterntopologie, Hubs, Switches
  - Bitrate 100 Mbps
  - 2 Modi: mit CSMA/CD für Hubs, ohne CSMA/CD für Switches
  - Rahmenformat gleich
  - Entfernung Hub-Knoten
    - Twisted Pair: max. 100 m (100BaseT)
    - Glasfaser: max. 2.000 m (100BaseFX)
  - Kodierung 4B/5B-NRZI (mit Modifikationen für jedes Medium)
  - Kaskadierung, Kombination Switches/Hubs möglich
  - Kombination 10BaseT/100BaseT möglich: Switches mit Dual-Speed-Anschlüssen, die sowohl 10BaseT als auch 100BaseT beherrschen

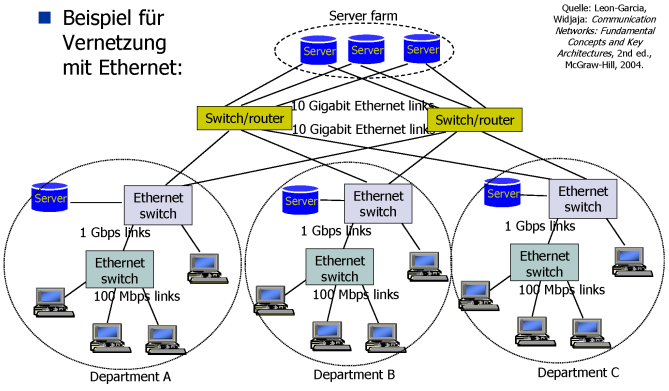
Ethernet

- Gigabit Ethernet
  - Bitrate 1 Gbps, gleiches Rahmenformat, Kodierung 8B/10B
  - Hubs (Buffered Distributers) mit Kollisionen, minimale Rahmengröße 512 Bytes (um Bedingung für Sendezeit und Ausbreitungszeit zu genügen)
  - Switches ohne CSMA/CD
  - 1000BaseT: Twisted Pair, 100 m
  - 1000BaseSX: Multimode-Glasfaser (550 m)
  - 1000BaseLX: Singlemode-Glasfaser (5 Km)
- 10 Gigabit Ethernet
  - Bitrate 10 Gbps, gleiches Rahmenformat, Kodierung 64B/66B
  - CSMA/CD aufgegeben, nur Switches
  - Entfernungen Multimode bis 300 m, Singlemode bis 40 Km
- 40 und 100 Gigabit Ethernet
  - Bitrate 40 und 100 Gbps, gleiches Rahmenformat, Kodierung 64B/66B
  - Multilane Distribution: mehrere physikalische Kanäle
  - Entfernungen Multimode bis 100 m, Singlemode bis 40 Km
- Terabit Ethernet
  - Standardisierungsaktivitäten für Bitraten > 100 Gbps

Beispiel einer möglichen Vernetzung

Ethernet

- Beispiel für Vernetzung mit Ethernet:



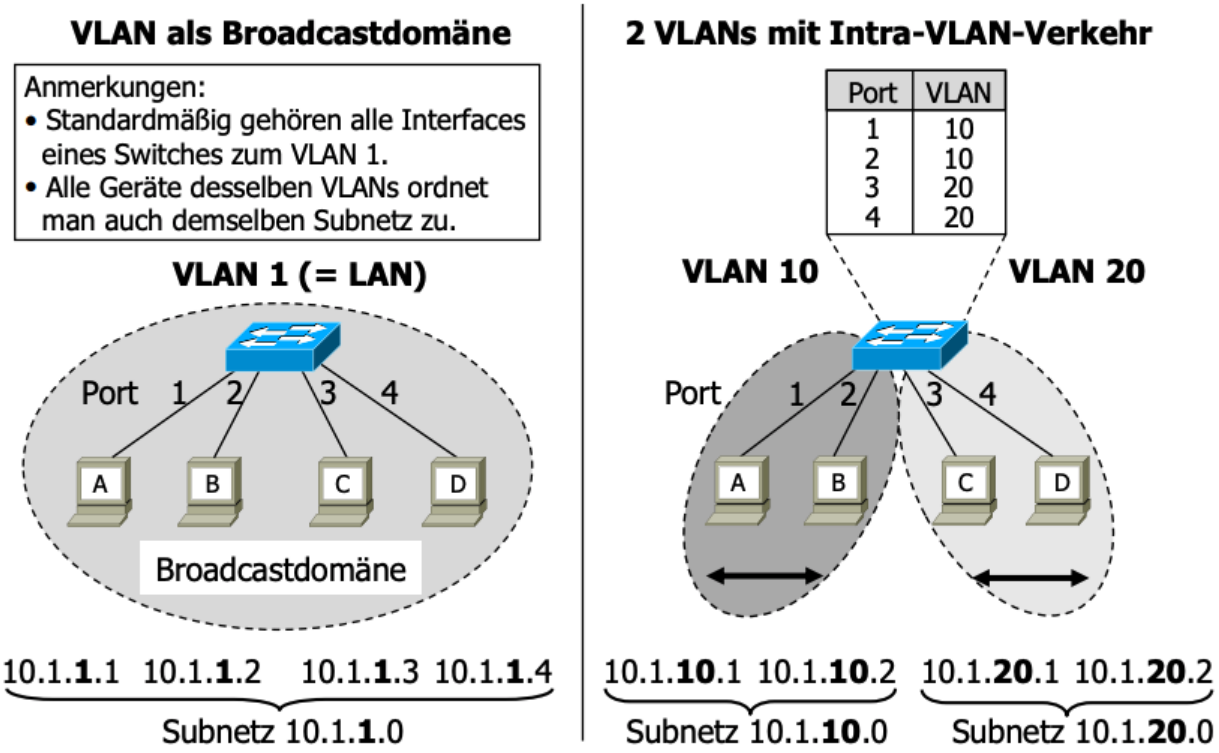
Quelle: Leon-Garcia, Widjaja: *Communication Networks: Fundamental Concepts and Key Architectures*, 2nd ed., McGraw-Hill, 2004.

Selbstlernendes Ethernet Sobald ein Switch einen Rahmen erhält, muss er eine Entscheidung treffen, wohin der Rahmen weitergeleitet werden soll. Wenn die physikalische Zieladresse an dem Port ist, von dem der Rahmen kommt, wird er *verworfen*. Wenn der Port der physikalischen Adresse unbekannt ist, wird der Rahmen einfach an alle Ports geflutet. Für einen eingehenden Rahmen wird dann die Zuordnung von physikalischer Adresse und Portnummer in einer Tabelle gespeichert. Damit kann die Zuordnung spezifischer erfolgen. Es ist *soft state* und hat eine TTL.

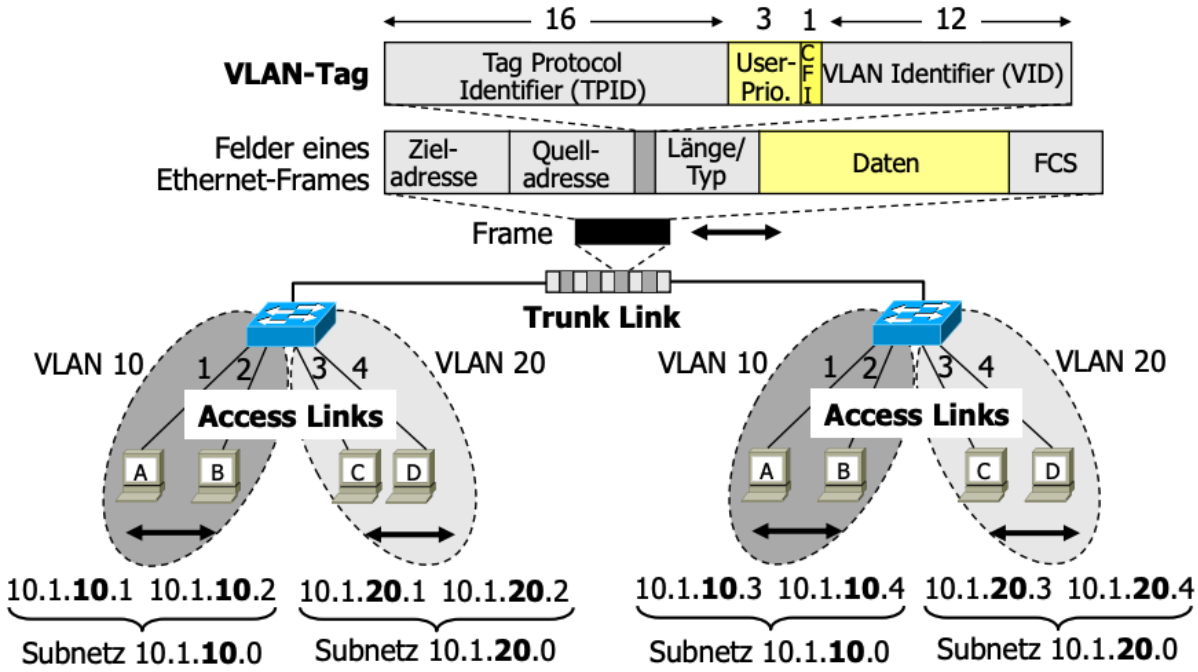
Spanning Tree Protocol Es ist klar, dass mit Switches zyklische Strukturen aufgebaut werden können, jedoch funktioniert dann das Selbstlernen nicht mehr. Deswegen führen alle Switches in einem LAN einen verteilten Algorithmus durch, so dass die Netztopologie einem Baum entspricht. Die *Idee* ist es einen Root Switch mit der kleinsten MAC-Adresse zu bestimmen und jeden Switch den kürzesten Pfad zum Root Switch bestimmen zu lassen. Jeder Switch leitet dann nur auf den Ports auf dem kürzesten Pfad weiter. Aufgrund hoher Konvergenzzeiten ($>$ 30 Sekunden) wurde das Protokoll weiterentwickelt hin zu einem *Rapid Spanning Tree Protocol (RSTP)*.

Virtuelle LANs (VLANs) Man erkennt, dass über Switches verbundene Knoten eine Broadcastdomäne für beispielsweise DHCP doer ARP bilden. Mit *VLANs* möchte man nun Netzwerke in scheinbar verschiedene LANs mit verschiedenen Subnetzen aufteilen. **Wichtig:** Dadurch soll eine *Lastoptimierung* sowie eine *Anpassung der logischen Netztopologie an Unternehmensorganisationen* stattfinden. Die Konfiguration der Switche findet dann über Management-Software basierend auf Ports, MAC-Adressen oder Protokollinformationen.

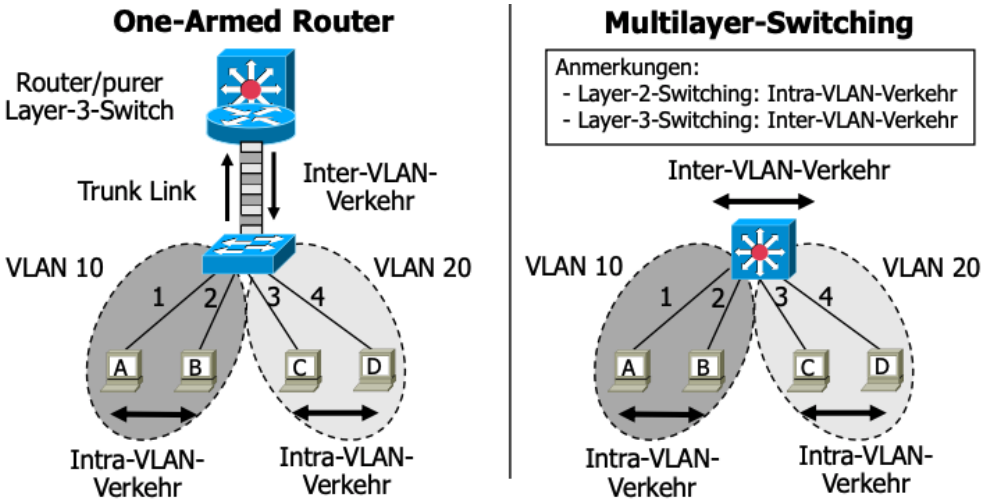
Portbasierte VLANs Hierbei bilden die Endgeräte an bestimmten Ports das VLAN. Die Rahmen werden vom Switch nur innerhalb des VLAN weitergeleitet, Inter-VLAN-Verkehr geht hier über den Router.



VLAN Tagging Eine andere Möglichkeit ist es VLAN-Tags im Ethernet-Rahmen unterzubringen. Diese Rahmen mit den VLAN-Tags werden dann zwischen Switches ausgetauscht. **Wichtig:** Dies ermöglicht VLANs über mehrere Switches.



VLAN Multilayer-Switching Während Intra-VLAN-Verkehr mit reinem Layer-2-Switching möglich ist braucht es für Inter-VLAN-Verkehr Layer-3-Switching (über IP). Dazu gibt es das **Multilayer-Switching**, bei der Layer-3-Funktionalität in einen Layer-2-Switch integriert wird.



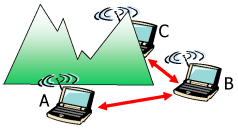
5.5 Drahtlose LANs

Charakteristika von Funkkommunikationen Es könnte **Dämpfung**, also eine umgebungsabhängige Abnahme der Signalstärke quadratisch mit der Entfernung, auftreten, ebenso wie Interferenzen durch andere Sender (drahtlose Netze, schnurlose Telefone, Mikrowellenöfen, Motoren, ...). Es findet eine **Mehrwegausbreitung** statt. Die Funkwellen werden reflektiert, phasenverschobene Wellen überlagern sich und schwächen sich kurzfristig ab beziehungsweise löschen sich aus (*short-range fading*). **Wichtig:** Es tritt eine höhere Fehlerrate, insbesondere als **Bursts**, auf.

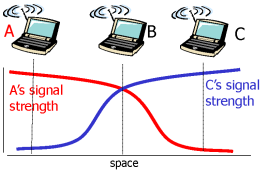
WLAN

Drahtlose LANs

- Hidden-Terminal Problem:
  - A, B hören sich
  - C, B hören sich
  - A, C hören sich nicht,
    A und C wissen nichts von möglichen Kollisionen bei B



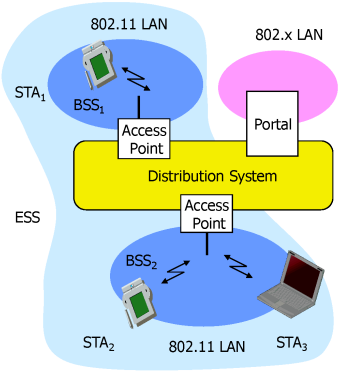
- genauso möglich durch Signaldämpfung:



Drahtlose LANs

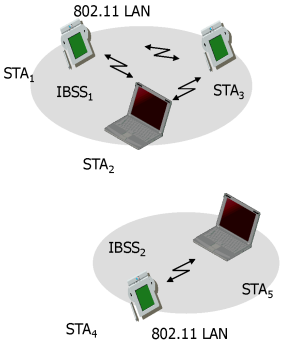
- Drahtlose LANs nach IEEE 802.11
  - 90er Jahre Beginn der Standardisierung: ETSI HIPERLAN und IEEE 802.11
  - 802.11-1997 mit Infrarot und Funk (FHSS, DSSS) bei 1 und 2 Mbps (Brutto)
  - Amendments in diversen Task Groups, 11b (11 Mbps), 11g (54 Mbps), 11n (600 Mbps), 11i (Sicherheit), 11e (Quality-of-Service), 11p (Vehicular), 11ac (Very High Throughput < 6 GHz), 11ad (Wireless Gigabit bei 60 GHz, WiGig) …
  - Aufnahme in konsolidierte Standards: 802.11-2003, 802.11-2007, 802.11-2012 (2.793 Seiten), 802.11-2016 (3.534 Seiten)
  - Wi-Fi-Alliance: Industrieforum zur Zertifizierung interoperabler Produkte
  - Betriebsmodi: Infrastrukturnetz (Access Points, Distribution Service), Ad-Hoc-Netz, Mesh Network (drahtlose Multi-Hop-Weiterleitung)
  - Adressierung: 4 physikalische Adressen
  - Medienzugriff: CSMA/CA (Collision Avoidance)
    - Kollisionserkennung würde 2. Antenne benötigen, die während des Sendens empfängt, schwieriger, teurer
    - 2 Varianten: Basic Access, RTS/CTS-Austausch
  - Energiesparen: Schlafphasen, synchronisiertes Aufwachen

Drahtlose LANs: Architektur eines Infrastrukturnetzes



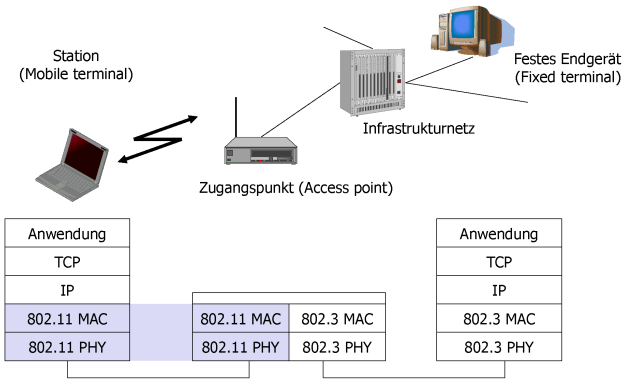
- Station (STA)
  - System mit Zugriffsfunktion auf das drahtlose Medium und Funk-kontakt zum Access Point
- Zugangspunkt, Access Point (AP)
  - Station, die sowohl in das Funk-LAN als auch das verbindende Netz (Distribution System) integriert ist
- Basic Service Set (BSS)
  - Gruppe von Stationen, die dieselbe Funkfrequenz nutzen
  - Basic Service Set Identifier (BSSID): MAC-Adresse des AP
  - SSID: Zeichenkette
- Portal
  - Übergang in ein anderes Netz
- Distribution System (DS)
  - Verbindung der APs über Schicht 2, gleiches IP-Subnetz
  - Extended Service Set (ESS): Vereinigung von BSSs mit gleicher SSID

Drahtlose LANs: Architektur eines Ad-Hoc-Netzes



- Station (STA)
  - System mit Zugriffsfunktion auf das drahtlose Medium
- Independent Basic Service Set (IBSS)
  - Gruppe von Stationen, die dieselbe Funkfrequenz nutzen
  - zufällige MAC-Adresse als BSSID

Drahtlose LANs: Protokollarchitektur bei Infrastrukturnetz



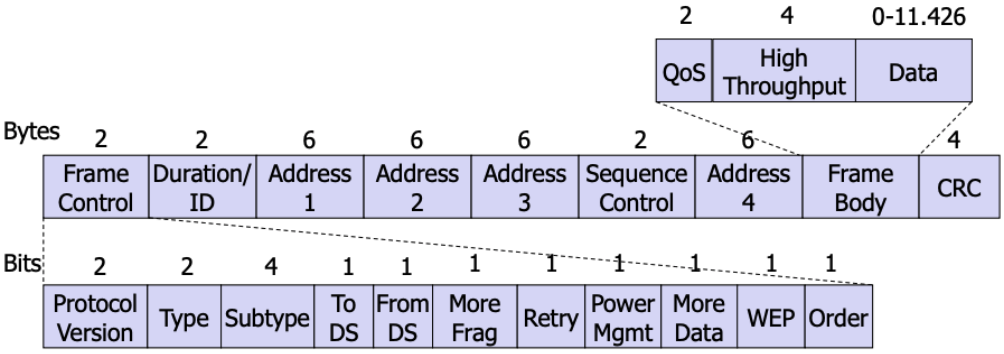
Drahtlose LANs

- Physikalische Varianten u.a.

	Frequenz	max. Bitrate	PHY-Technologien, Bemerkungen
802.11-1997	2,4 GHz	1-2 Mbps	Frequency-Hopping Spread Spectrum & Direct Sequence Spread Spectrum
802.11b (1999)	2,4 GHz	11 Mbps	Complementary Code Keying (CCK) & QPSK
802.11a (1999)	5 GHz	54 Mbps	OFDM
802.11g (2003)	2,4 GHz	54 Mbps	Orthogonal Frequency Division Multiplexing (OFDM) & CCK für Rückwärtskompatibilität mit 802.11b
802.11n (2009)	2,4 und 5 GHz	600 Mbps	OFDM, Multiple Input Multiple Output (MIMO), Frameaggregation
802.11ac (2012)	5 GHz	3,2 Gbps	Bandwidth Expansion (max. 160 MHz), Multiuser MIMO, Frameaggregation
802.11ad (2014)	60 GHz	6,76 GHz	2.160 MHz Bandbreite, Millimeterwellen, keine Objektdurchdringung, Beamforming

5.5.1 MAC

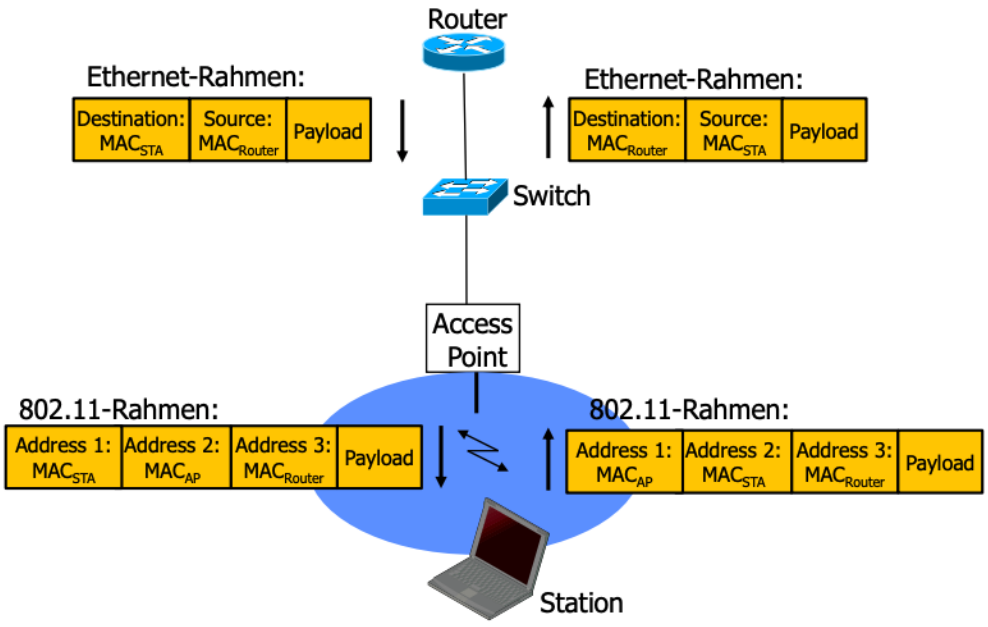
Rahmenformat Die **Rahmensteuerung (*frame control*)** besteht aus dem Typen (Management (00), Steuerung (01) oder Daten (10)) und einem Subtypen (Association Request, Data, ACK, RTS, CTS, . . .). Anschließend folgt die Belegungsdauer des Mediums in µs — für RTS/CTS oder eine ID. Vier physikalische Adressen bieten Platz für Empfänger, Sender und weitere. Sequenznummern sind notwendig für ARQ.



Interpretation der Adressen Die Stationen und Zugriffspunkte werden durch physikalische Adressen identifiziert. Zudem gibt es BSSIDs (*service set IDs*), welche die BSS identifizieren. Dies sind zufällige physikalische Adressen bei Ad-Hoc-Netzen und Adressen der Zugriffspunkte bei Infrastrukturen. Abhängig von der Rahmenart werden dann die DS-Bits gesetzt und Adressen unterschiedlich interpretiert:

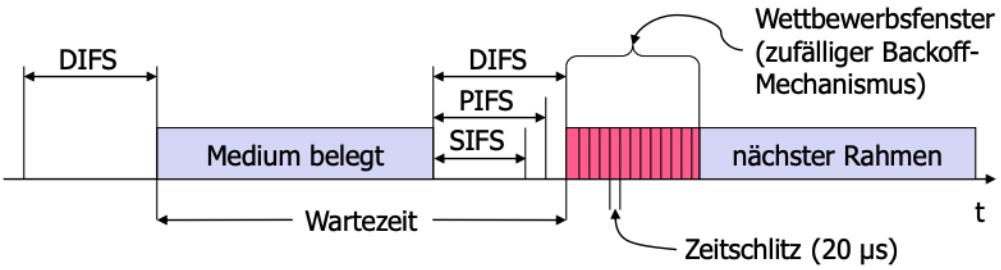
Rahmenart	To DS	From DS	Address 1 (Empfänger)	Address 2 (Sender)	Address 3 (zusätzlich)	Address 4 (zusätzlich)
Ad-Hoc-Netz STA → STA	0	0	Empfänger-STA	Sender-STA	(zufällige) BSSID	—
Infrastrukturnetz STA → AP	0	0	AP	Sender-STA	Ziel (bspw. Router)	—
Infrastrukturnetz STA → AP	0	0	Empfänger-STA	AP	Quelle (bspw. Router)	—
4 Adressen, ausschließlich *Mesh Network*	0	0	Empfänger akt. HOP	Sender akt. HOP	Quelle	Ziel

Beispiel für eine solche Verwendung der Adressen

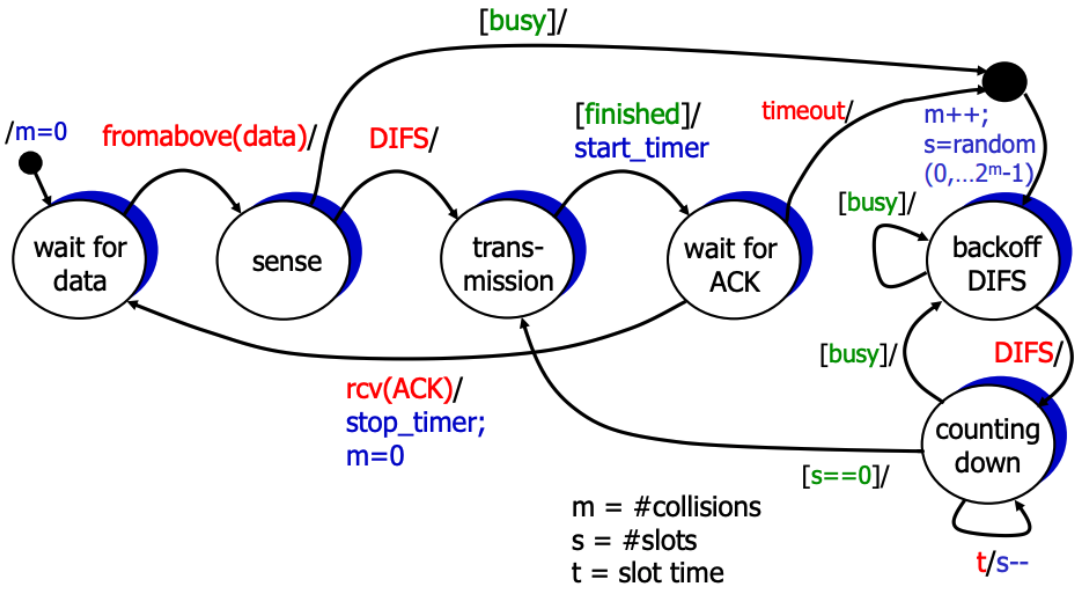


Medienzugriff Wenn die MAC-Schicht einer Station von der Netzwerkschicht ein Datagramm erhält, überprüft sie das Medium (gemäß CSMA/CA, *listen before talking*). Wenn es eine Zeitdauer **DIFS** (*distribution interframe space*) frei ist, wird der Rahmen gesendet, andernfalls geht es in Backoff. Wenn der Empfänger das Datagramm fehlerlos erhält, wartet er eine Zeitdauer **SIFS** (*short interframe space*) und sendet dann eine positive Bestätigung zurück. Wenn nach einem Timeout kein ACK zurückkommt geht der Sender in Backoff.
Es kommt zum **Truncated Binary Exponential Backoff**, abhängig von der Anzahl der Kollisionen würfelt der Sender eine zufällige Wartezeit, **reduziert sie solange das Medium frei ist**[4] und wiederholt dann die Sendung.
Damit ist es eine **Variante von nicht-persistentem CSMA/CD**, was neu ist sind DIFS und SIFS.

Medienzugriff — Zeitablauf

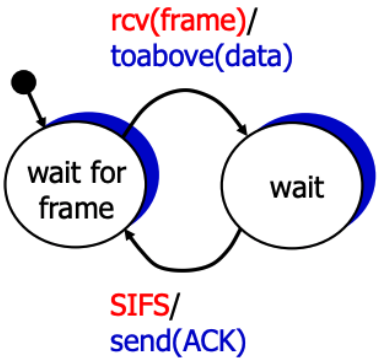


Statechart — Sender

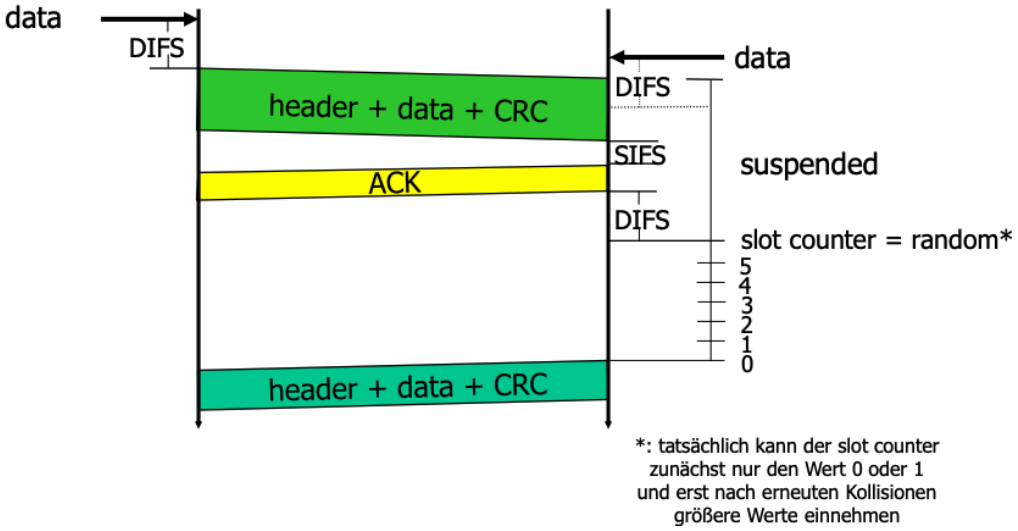


Statechart — Empfänger

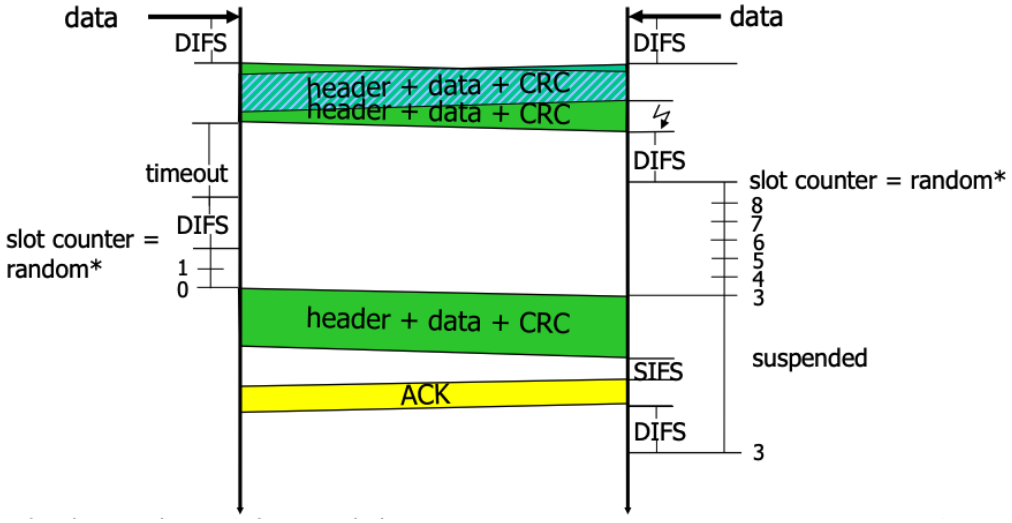
[4] In anderen Worten wird nur gewartet solange das Medium frei ist.



Medienzugriff — Beispiel Wollen zwei Stationen zu ähnlicher Zeit senden, so kann durch das DIFS eine Kollision vermieden werden.



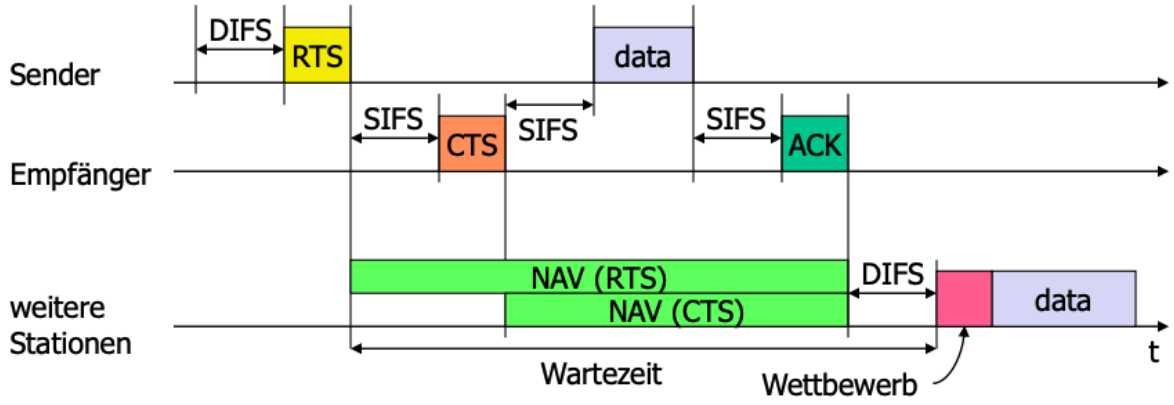
Wollen zwei Stationen *fast gleichzeitig* senden **und ist der Unterschied kleiner als die Ausbreitungsverzögerung**, entsteht eine Kollision, welche anschließend aufgelöst wird.



RTS/CTS-Austausch Hierbei kann zusätzlich ein vorheriger Austausch von kurzen Reservierungsnachrichten erfolgen. Der Sender sendet dabei ein *Request-to-Send (RTS)* mit der Länge des Rahmens, der Empfänger (eine Station im Ad-Hoc-Netz und ein Zugriffspunkt bei einem Infrastrukturnetz) erteilt dann ein *Clear-to-Send (CTS)*, in welcher sich auch die Länge findet. Der Medienzugriff findet hierbei ebenfalls über Basic Access statt.

Nachteil: Es ist deutlich größerer Overhead.
Vorteil: Wenn keine Kollision auftritt, ist das Medium reserviert, wenn eine Kollision auftritt, dauert diese nicht lange, da die RTS kurz sind.
Vorteil: Das Hidden-Termina-Problem ist damit teilweise gelöst, denn die Stationen, die den Sender nicht hören, erfahren vom Empfänger eine Reservierung.

RTS/CTS-Austausch — Zeitablauf



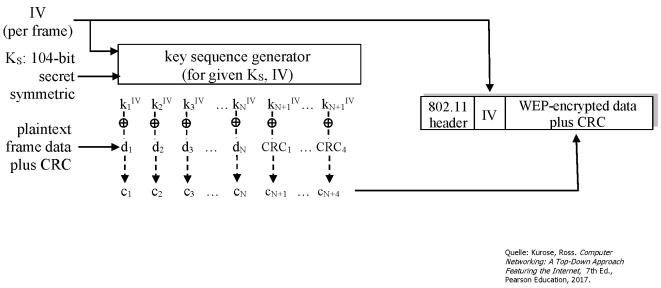
5.5.2 Sicherheit

Drahtlose LANs: Sicherheit

- Wired Equivalent Privacy (WEP)
  - im initialen WLAN 802.11-1997-Standard: Ziel war ähnliche Sicherheit wie bei leitungsgebundener Kommunikation
  - Schlüsselaustausch: nicht festgelegt
  - Authentifikation zwischen Station und Zugangspunkt: symmetrisches Challenge-Response-Verfahren
  - Verschlüsselung
    - symmetrischer 40-Bit oder 104-Bit Schlüssel K_S in Station und Zugangspunkt
    - für jedes Paket wird 24-Bit Initialisierungsvektor IV erzeugt und im Klartext im Rahmen gesendet
    - für Daten wird 4-Byte Integrity Check Value (ICV) mittels CRC berechnet
    - aus K_S+IV wird mit RC4 pseudozufällige Bitsequenz erzeugt, Daten + ICV werden mit XOR bitweise verknüpft
    - wg. nur 2^{24} möglichen IVs sind Angriffe möglich, weitere Sicherheitsprobleme bestehen

Drahtlose LANs: Sicherheit

- Versenden eines Frames mit WEP (Encapsulation):



Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

Drahtlose LANs: Sicherheit

- Weiterentwicklung der Sicherheit
  - Arbeitsgruppe IEEE 802.11i, aufgenommen in 802.11-2007
  - WiFi-Alliance: WiFi Protected Access (WPA)
    - 1. Version vor Fertigstellung von 802.11i
    - erfordert nur Aktualisierung der Firmware von Stationen, aber neue Zugangspunkte
    - gegenseitige Authentifikation über Pre-Shared Key (PSK) oder über Authentifikations-Server (z.B. RADIUS)
    - Temporal Key Integrity Protocol (TKIP): verbesserte Version der WEP Encapsulation
      - RC4 mit wirklich neuem 128-Bit Schlüssel pro Frame
      - statt CRC Frame-Authentifikation: Message Integrity Code (MIC)

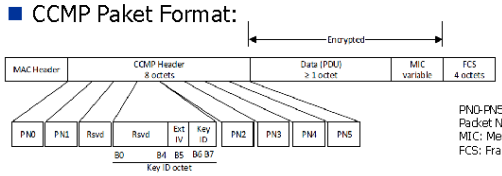
Drahtlose LANs: Sicherheit

- Weiterentwicklung der Sicherheit (Fortsetzung)
  - WEP und TKIP weiter in 802.11-2016, aber „obsolet"
  - WPA2
    - gegenseitige Authentifikation über Pre-Shared Key (PSK) oder über Authentifikations-Server (z.B. RADIUS, DIAMETER)
    - Encapsulation: Counter Mode with Cipher Block Chaining Message Authentication Protocol (CTR with CBC-MAC Protocol, CCMP)
      - Counter (Packet Number), wird bei jedem Frame inkrementiert, dient Erstellung einer Nonce pro Frame
      - Verwendung von AES
  - seit 2017 Sicherheitslücke in WPA2 bekannt (Krack), WPA3 von WiFi Alliance angekündigt

Quelle: IEEE802.11-2016

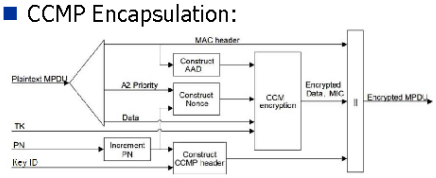
Drahtlose LANs: Sicherheit

- CCMP Paket Format:



PN0-PN5:
Packet Number, 48 Bit
MIC: Message Integrity Code
FCS: Frame Check Sequence

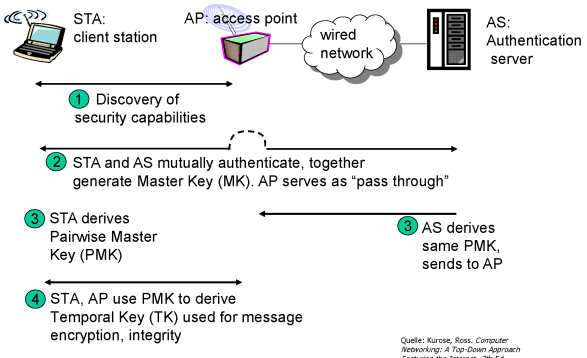
- CCMP Encapsulation:



TK: Temporal Key
(neu für jede Session)
PN: Packet Number
Key ID:
A2: 2. Adresse im MAC-Header
Priority: ggfs. QoS-Priorität aus MAC-Header
AAD: Additional Authentication Data

Drahtlose LANs: Sicherheit

- Authentifizierung über Authentifikations-Server:



Quelle: Kurose, Ross. *Computer Networking: A Top-Down Approach Featuring the Internet*, 7th Ed., Pearson Education, 2017.

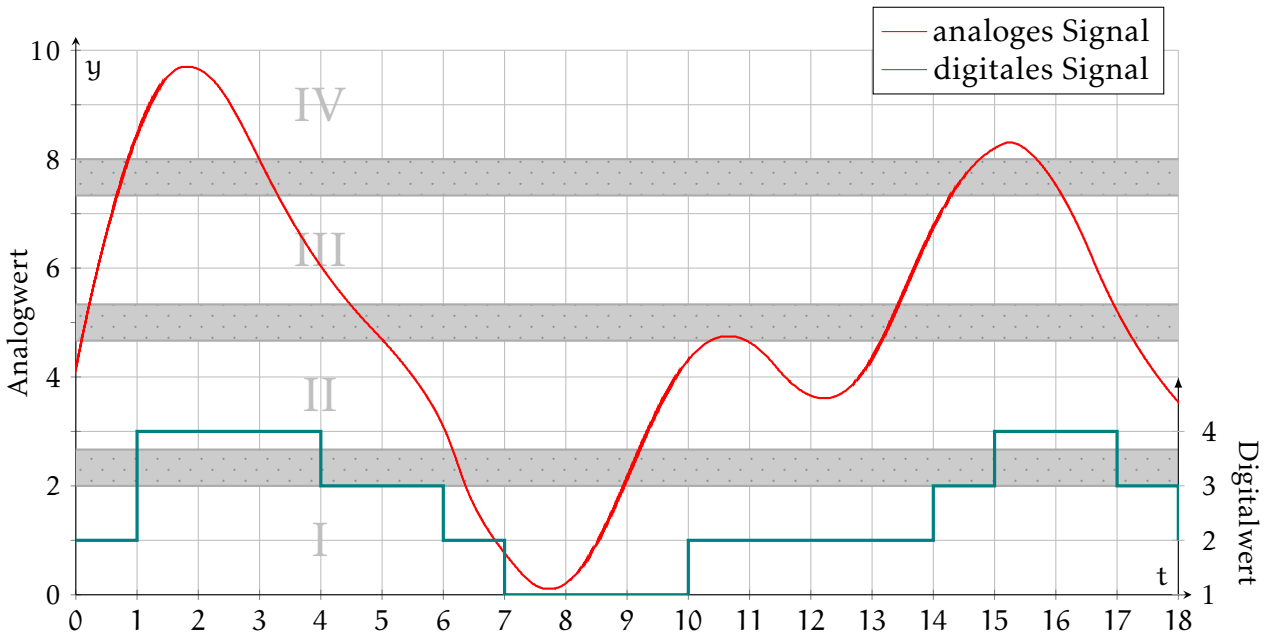
KAPITEL 6

PHYSIKALISCHE SCHICHT

6.1 Signale und Übertragungssysteme

6.1.1 Signale

Ein **Signal** ist die Darstellung von Informationen durch physikalische Größen, wie beispielsweise Strom, Spannung, Lichtwellen oder elektromagnetische Feldstärken. Wir kategorisieren Signale in **analoge Signale**, welche kontinuierlich[1] im Werte- und Zeitbereich sind, **digitale Signale**, welche diskret[2] im Werte- und Zeitbereich sind, sowie **binäre Signale**, welche Digitalsignale mit nur zwei Werten sind.



Leitungskodierung Unter der **Leitungskodierung binärer Signale** versteht man die Zuordnung von Signalwerten zu Nullen und Einsen einer Bitsequenz bei der Übertragung im Basisband. Signal!Basisbandübertragung Bei einer **Basisbandübertragung** werden Signale direkt auf das Medium gesendet. Wir wollen dann verschiedene Eigenschaften betrachten:

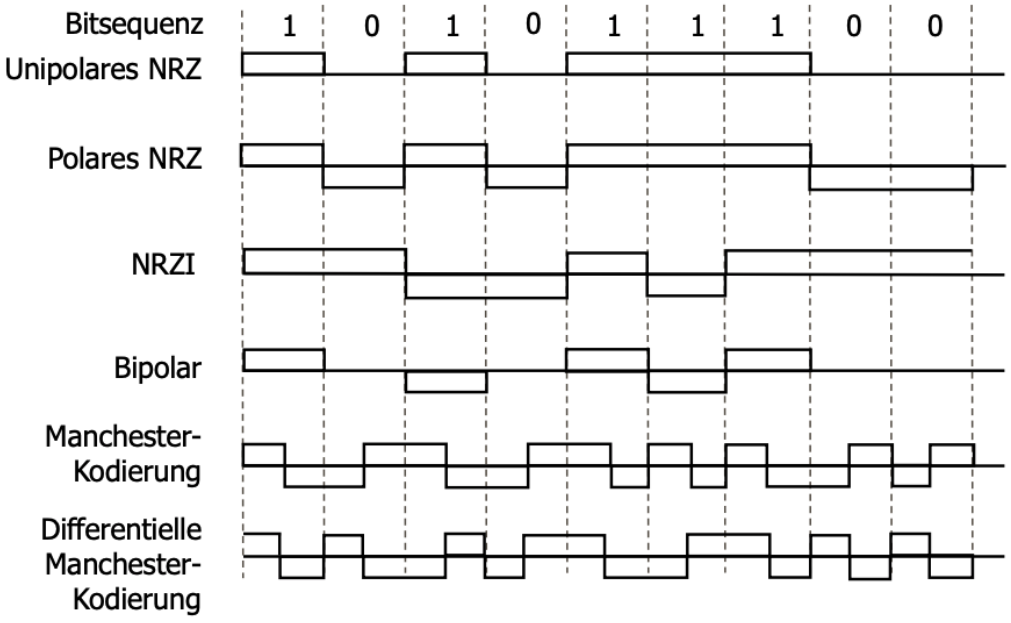
- Wir sprechen von einem **selbsttaktendem Signal**, wenn der Empfänger den Sendertakt direkt aus dem Signal gewinnen kann.
- Wir sprechen von einem **gleichstromfreien Signal**, wenn kein Gleichanteil im elektrischen Signal zu finden ist.

[1] Es treten beliebige Werte auf
[2] Es treten nur endlich viele verschiedene Werte auf, sind idealisiert.

- Der **Bandbreitenbedarf eines Signals** ist die Breite des Frequenzbands, welche benötigt wird um das Signal zu übertragen.

Beispiele für Leitungskodierungen

Verfahren	Beschreibung	selbstaktend	gleichstromfrei	Anmerkungen
NRZ[3]	„1“ ist HI, „0“ ist LO	✘	unipolar: ✘ polar: ✓	
NRZI[4]	„1“ ist ein Wechsel „0“ ist kein Wechsel	✘	✘	
Bipolar	„0“ ist ein Nullpegel „1“ ist alt. LO u. HI	✘	✓	
Manchester	Pegelwechsel in Mitte HI $\overset{\text{„0“}}{\underset{\text{„1“}}{\longleftrightarrow}}$ LO	✓	✘	dopp. BB Verw. bei Eth.
Diff. Manch.	Pegelwechsel in Mitte Taktwechsel am Flankenanf.	✓	✓	höhere BB Verw. bei TR
4B/5B	4-Bit-Block durch 5-Bit-Wort, diese durch NRZI	✓	✘	1,25-fache BB Verw. bei FDDI



Fourierreihe Die **Fourierreihe** beschreibt ein periodisches Signal $s(t)$ mit Periodendauer T als Summe von Sinus- und Kosinusschwingungen verschiedener Frequenzen:

$$s(t) = \frac{c}{2} + \sum_{n=1}^{\infty} a_n \cdot \sin(2\pi \cdot n \cdot f \cdot t) + \sum_{n=1}^{\infty} b_n \cdot \cos(2\pi \cdot n \cdot f \cdot t)$$

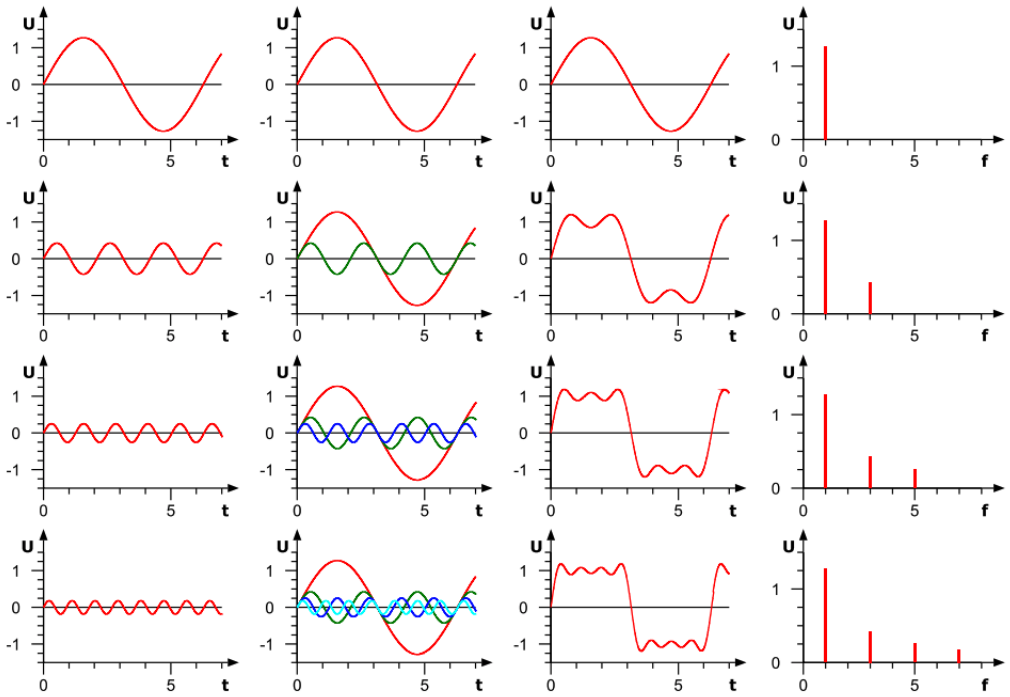
[3]Non-Return-To-Zero
[4]NRZ-Inverted

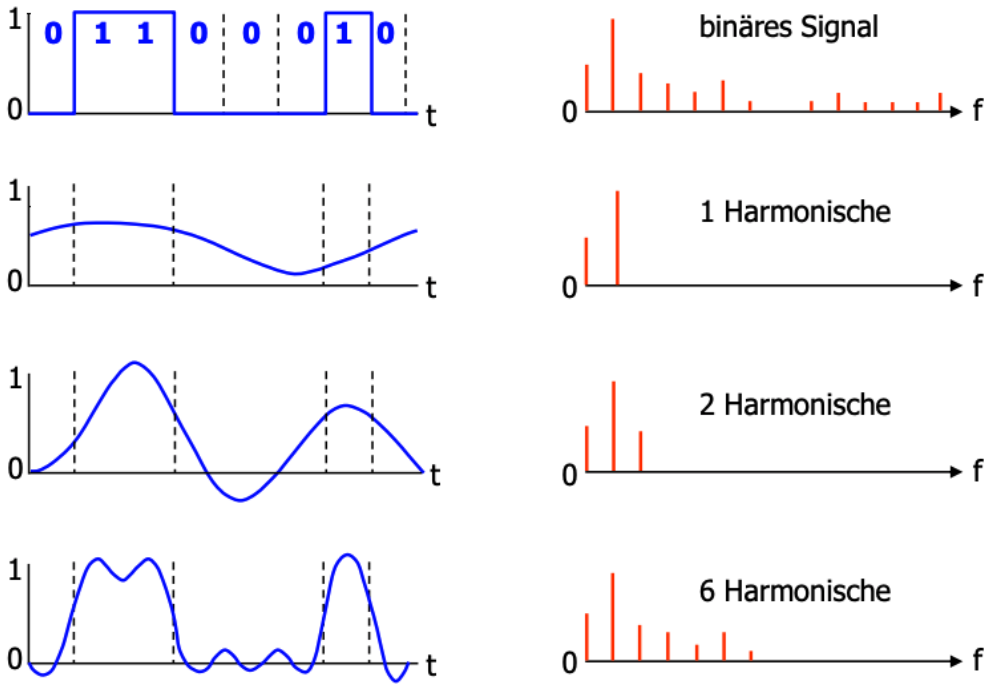
Die Koeffizienten a_n und b_n sind die Sinus-/Kosinus-Amplituden der n-ten **harmonischen Schwingung**. Der konstante Anteil des Signals wird durch c ausgedrückt. $f = 1/T$ ist die Frequenz der *ersten* harmonischen Schwingung. Die Koeffizienten sind aus dem Signal durch

$$a_n = \frac{2}{T} \cdot \int_0^T s(t) \cdot \sin(2\pi \cdot n \cdot f \cdot t)\, dt \qquad b_n = \frac{2}{T} \cdot \int_0^T s(t) \cdot \cos(2\pi \cdot n \cdot f \cdot t)\, dt \qquad c = \frac{2}{T} \cdot \int_0^T s(t)\, dt$$

bestimmbar. Eine äquivalente Darstellung ergibt sich durch **Amplituden-** und **Phasenkoeffizienten**:

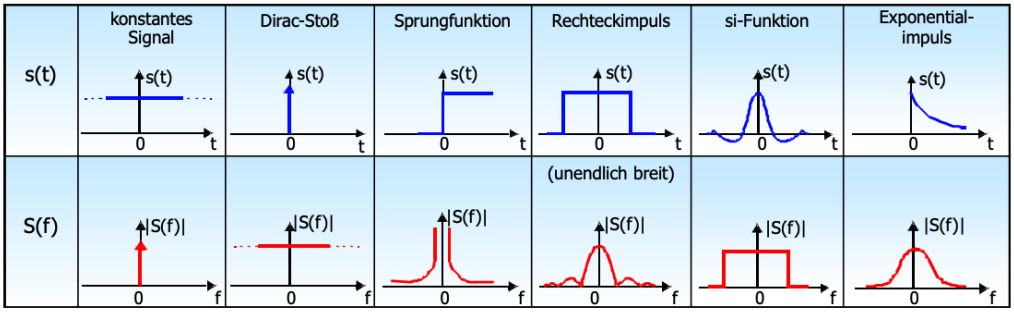
$$d_n = \sqrt{a_n^2 + b_n^2} \qquad \text{und} \qquad \varphi_n = \arctan\left(\frac{b_n}{a_n}\right).$$





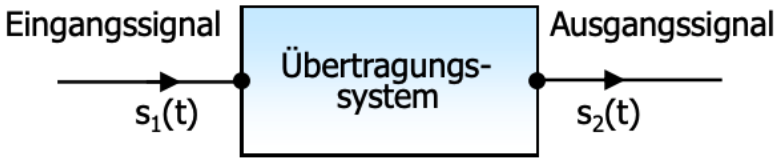
Fouriertransformationen Während **Fourierreihen** zur Darstellung periodischer Signale gedacht sind, verwendet man **Fouriertransformationen** zur Darstellung aperiodischer Signale. Sie sind ein Grenzfall der Fourierreihe mit $T \to \infty$. Für $T \to \infty$ wird aus dem Linienspektrum ein kontinuierliches Spektrum und die Addition geht in eine Integration über.

Beispiel für Signale und Aplitudenspektren Bei Signalen mit Impulsen und diskreten Sprüngen stellt man fest, dass ie ein unendlich breites Spektrum besitzen.

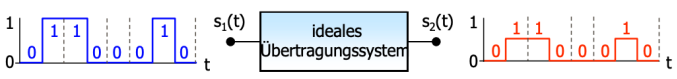


6.1.2 Übertragungssysteme

Die Systemtheorie beschäftigt sich mit einem mathematischen Modell zur Beschreibung des Übertragungsverhaltens einer komplexeren Anordnung. Dabei werden Eingangssignale s_1 in Ausgangssignale s_2 in einem **Übertragungssystem** umgewandelt.



Bandbreitenbeschränkung Jedes Übertragungssystem benötigt dabei zur Signalübertragung Energie. Man sagt ein Übertragungssystem sei *ideal*, wenn es alle Frequenzen gleichermaßen dämpft. *Reale* Übertragungssysteme hingegen dämpfen verschiedene Frequenzen unterschiedlich. Dabei kommt es zu einer **Verzerrung** des Signals. *Meistens werden die Frequenzen bis zu einer Grenzfrequenz* f_C ***unverändert*** *übertragen und höhere Frequenzen dafür stark abgeschwächt (**Tiefpass**)*. Gründe hierfür sind die Eigenschaften des Übertragungsmediums oder Filter zur Bandbreitenbegrenzung bei Frequenzmultiplexverfahren.



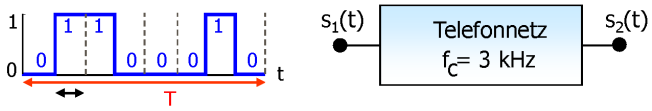
Bandbreitenbeschränkung — Beispiel: Telefonnetz

Quelle: Bernd Schürmann. Grundlagen der Rechnerkommunikation, Vieweg Verlag, 2004.

Signale und Übertragungssysteme

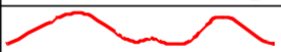
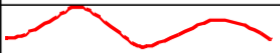
- Beispiel: Telefonnetz
  - Bandbreitenbeschränkung ca. f_c = 3.000 Hz (Frequenz der höchsten harmonischen Schwingung)
  - Bitrate B in Bits/s
  - Annahme: 1 Byte stellt periodisches Signal dar
  - also ist T = 8 Bits/B die Zeit, um 1 Byte zu senden
  - also ist f = 1/T = B/8 Bits die Frequenz der 1. harmonischen Schwingung in Hz
  - alle harmonischen Schwingungen sind Vielfache von f, also ist die Anzahl der harmonischen Schwingungen
    ⌊3.000 Hz/f⌋ = ⌊3.000 Hz/(B/8 Bits)⌋ = ⌊24.000 Bits/B⌋

Übertragung eines Bytes



Rechnerkommunikation, Physikalische Schicht 15

Beziehung Datenrate und Anzahl an harmonischen Schwingungen Wir stellen fest, dass **je höher die Bitrate bei gegebener Bandbreite, desto weniger Frequenzen werden übertragen und desto größer ist die Verzerrung.**

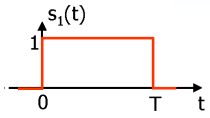
bps	T [ms]	f [Hz] (1. Harmonische)	übertragene Harmonische	etwaiges Aussehen des Ausgangssignals
300	26,67	37,5	80	
2.400	3,33	300	10	
4.800	1,67	600	5	
9.600	0,83	1200	2	
19.200	0,42	2400	1	
38.400	0,21	4800	0	

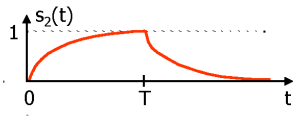
Faltungsintegral

Übertragungssysteme

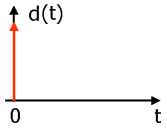
- Faltungsintegral
  - Ausgangssignal kann durch Überlagerung der Transformationen der Komponenten des Eingangssignals berechnet werden
  - diese Operation entspricht im Zeitbereich dem Faltungsintegral:

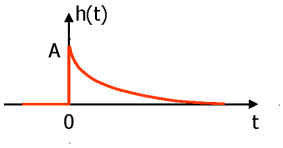
$$s_2(t) = \int_{-\infty}^{\infty} s_1(\tau) \cdot h(t-\tau) d\tau$$





  - dabei ist h(t) die Impulsantwort, die Form des Ausgangssignals für einen Impuls der Länge 0 (Dirac-Stoß):

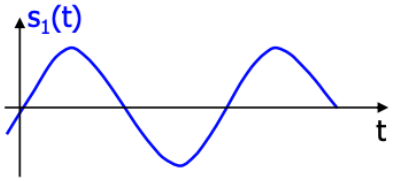


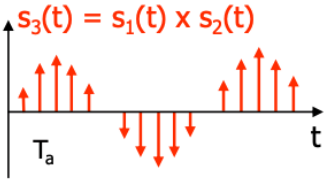


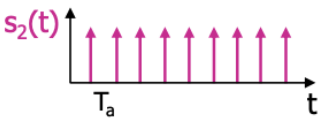
Rechnerkommunikation, Physikalische Schicht 17

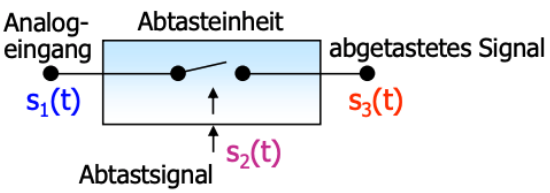
6.2 Abtasttheoreme

Wenn wir nun ein analoges Signal s_1 gegeben haben, wobei alle Schwingungen innerhalb einer Frequenzbandbreite f liegen, können wir das Signal mit dem Abtastsignal (periodisch mit Abtastfrequenz f_a) s_2 abtasten. Das abgetastete Signal s_3 ergibt sich dann durch Multiplikation.

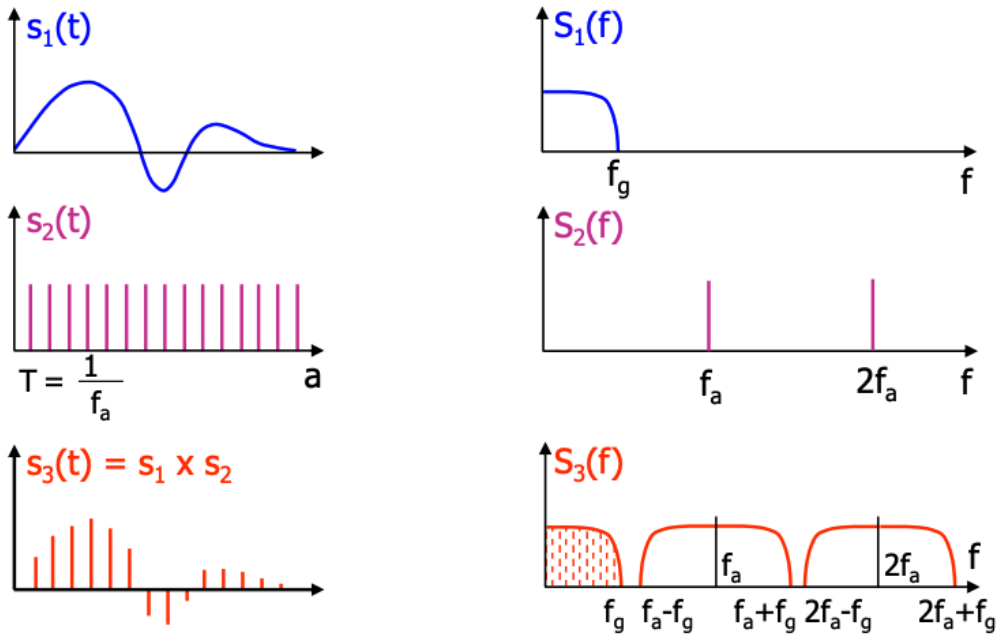








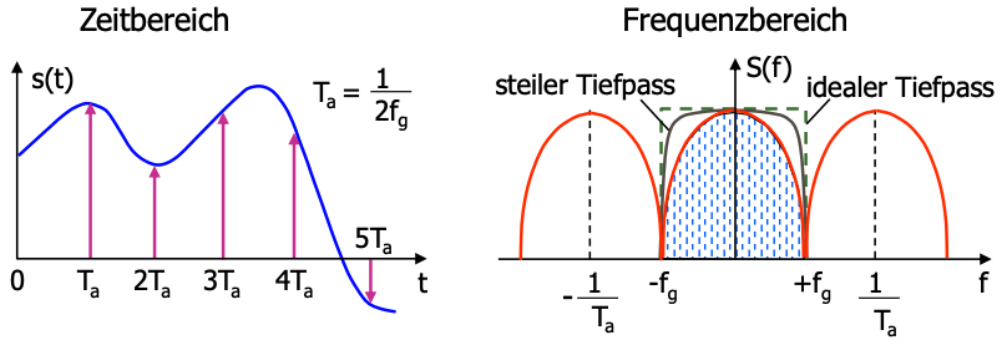
Wir stellen fest, dass aus einem kontinuierlichen, bandbegrenzten Signal s im Frequenzbereich durch Abtastung ein unendlich breites und in f_a periodisches Spektrum.



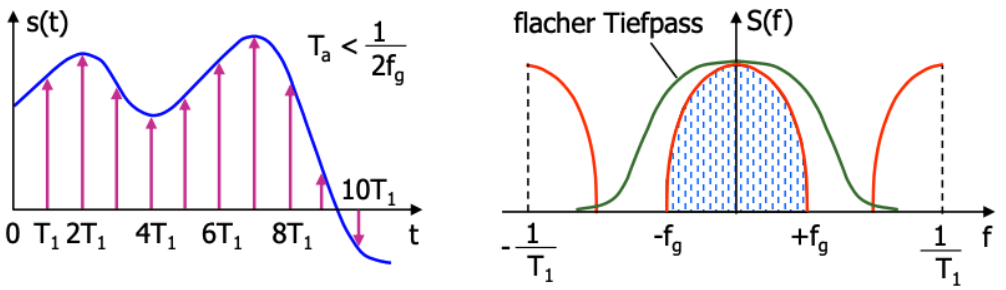
Um eine eindeutige Rückkonstruktion in das kontinuierliche Signal zu garantieren muss gemäß des **Abtasttheorems** gelten, dass die Abtastfrequenz f_a mindestens der doppelten Signalfrequenz f_g entspricht. Allgemein lautet das Theorem:

> Das analoge Signal s_1 mit einem Spektrum von 0 bis f_g wird durch das abgetastete Signal s_2 vollständig beschrieben, wenn $f_a \geqslant 2 \cdot f_g$ gilt.

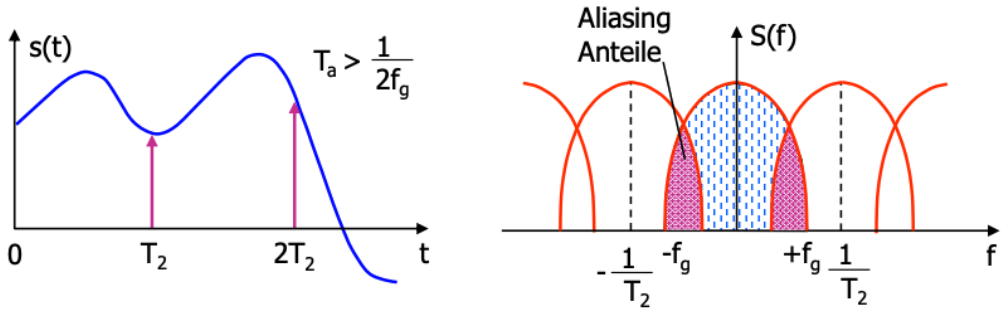
Verschiedene Beispiele Im ersten Beispiel wird das Abtasttheorem genau eingehalten.



Je größer jedoch die Abtastfrequenz ist, desto größer sind die Abstände der Amplitudenspektren.



Ist die Abtastfrequenz zu niedrig, so überlagern sich die Ausgangsspektren und sind nicht mehr zu trennen (Alias-Effekt)



Maximale Datenrate Das obige Telefonbeispiel hat gezeigt, dass eine Mindestbandbreite notwendig ist, damit der Empfänger das binäre Signal erkennen kann. Durch eine Umkehrung des Abtasttheorems lässt sich dann eine theoretische Obergrenze für die Datenrate eines Kanals abgeleitet werden.

Nyquist-Theorem Das Abtasttheorem sagt aus, dass das Signal der Bandbreite B durch Abtastwerte der Frequenz 2B wiederhergestellt werden kann, mehr Abtastwerte sind unnötig. Damit *kann* ein Signal der Bandbreite B nur Abtastwerte der Frequenz 2B repräsentieren. Ein Signal mit V diskreten Werten repräsentiert dabei $\log_2 V$ Bit. Die maximale Datenrate bestimmt sich dann durch

$$\text{max. Datenrate} = 2B \cdot \log_2(V) \qquad \text{Einheit: } \left[\frac{\text{Bit}}{\text{s}}\right]$$

Beispiel — Telefonie: Ein Telefon mit 3 KHz und 6000 Abtastungen pro Sekunde hat $V = 2$, womit sich eine maximale Datenrate von 6 kpbs ergibt.
Beispiel — ISDN: Eine ISDN-Leitung mit 4 KHz und 8000 Abtastungen pro Sekunde hat $V = 256$, womit sich eine maximale Datenrate von 64 kpbs ergibt.

Shannon-Theorem In Wirklichkeit wird diese Maximalgrenze aber unter anderem durch Rauschen verursachte Störsignale nicht erreicht. Wir definieren deswegen einen **Rauschabstand** als das Verhältnis von Signal- und Rauschleistung S/N.
Hierbei werden wieder übliche logarithmische Einheiten wie dB (*Dezibel*) mit $10\log_{10}(S/N)$ verwendet. So entspricht ein Rauschabstand von $S/N = 100/0{,}1 = 1000$ der logarithmischen Größe 30 Dezibel.
Shannon erweiterte daraufhin das Nyquist-Theorem für rauschbehaftete Kanäle. Für einen Kanal mit Bandbreite B und Rauschabstand S/N gilt dann

$$\text{max. Datenrate} = B \cdot \log_2(1 + S/N) \qquad \text{Einheit: } \left[\frac{\text{Bit}}{\text{s}}\right].$$

Beispiel: Mit einer Bitrate von $B = 3{,}4$ kHz und einem Rauschabstand von 40 Dezibel ergibt sich eine maximale Bitrate von 45,2 kbps.

6.3 Modulation

Unter **Modulation** versteht man die Änderung von Parametern[5] eines Trägersignals durch ein *moduliertes* oder *aufgeprägtes* Signal. Unter **Demodulation** versteht man die Rückgewinnung

[5] Dinge wie Amplituden, Frequenzen oder Phasen

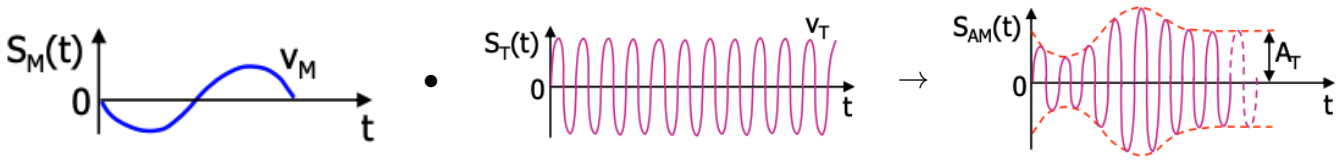
des modulierenden Signals aus dem modulierten Informationsträger. **Wichtig:** Ein Beispiel für die Modulation und Demodulation in einer Einheit ist ein **Modem**. Wir unterscheiden drei Modulationsarten.

6.3.1 Analog-Analog-Wandlung

Bei der **Analog-Analog-Wandlung** möchte man ein analoges modulierendes in ein analoges Trägersignal umwandeln. Dies erlaubt dann eine Breitbandübertragung mit dem Frequenzmultiplexverfahren. Ein Beispiel hierfür sind Fernsehsignale.

Amplitudenmodulation Das Trägersignal hierbei ist eine hochfrequente Sinusschwingung, welche mit dem modulierendem Signal multipliziert wird.

$$s_{AM}(t) = s_M(t) \cdot s_T(t)$$



Frequenzmodulation Das Trägersignal ist identisch der Amplitudenmodulation, hier wird die Frequenz moduliert.

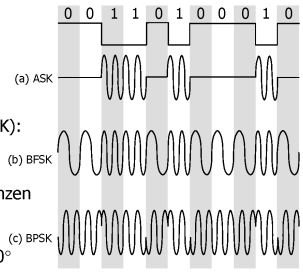
Phasenmodulation Das Trägersignal ist identisch der Amplitudenmodulation, hier wird die Phase moduliert.

6.3.2 Digital-Analog-Wandlung

Bei der **Digital-Analog-Wandlung** möchte man ein digitales modulierendes Signal in ein analoges Trägersignal wandeln, wie es zum Beispiel bei Daten über Funkkanäle notwendig ist.

Modulation

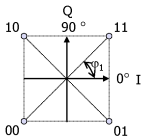
- Verfahren zur Digital-Analog-Wandlung
  - Amplitudentastung (Amplitude Shift Keying, ASK)
    - „1" = hohe Amplitude
    - „0" = niedrige Amplitude
    - Verwendung für digitale Daten über Glasfaser
  - Binäre Frequenzumtastung (Binary Frequency Shift Keying, BFSK):
    - „1" = hohe Frequenz
    - „0" = niedrige Frequenz
    - geeignet für hohe Trägerfrequenzen
  - Binäre Phasenumtastung (Binary Phase Shift Keying, PSK)
    - „1" = Phase 0°, „0" = Phase 180°
    - robust gegenüber Rauschen



Quelle: W. Stallings, *Data and Computer Communications*, 10th Ed., Pearson Education, 2014.

Modulation

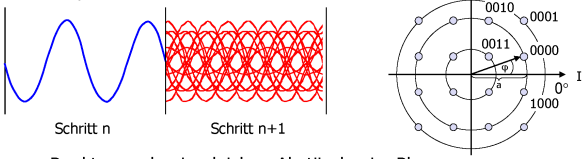
- Multiple FSK (MFSK): mehrere Frequenzen, Signalwert repräsentiert mehr als ein Bit
- Differential BPSK (DBPSK): „1" = Phasenwechsel, „0" = kein Wechsel, benötigt keine genaue Synchronisierung
- Quadrature PSK (QPSK)
  - Phasen $\varphi_1 = 45°$, $\varphi_2 = 135°$, $\varphi_3 = 225°$, $\varphi_4 = 315°$
  - Signalwert repräsentiert 2 Bits
  - im Phasenraum entspricht jeder Punkt einer Amplitude und Phase $\varphi_1 \ldots \varphi_4$
- Multilevel PSK (MPSK)
  - Verallgemeinerung zu n Phasen
  - Signal enthält $\log_2 n$ Bits



Quelle: Bernd Schürmann. Grundlagen der Rechnerkommunikation, Vieweg Verlag, 2004.

Modulation

- Quadraturamplitudenmodulation (Quadrature Amplitude Modulation, QAM)
  - Kombination von ASK und PSK
  - Signal kann zwischen verschiedenen Phasen und Amplituden wechseln:

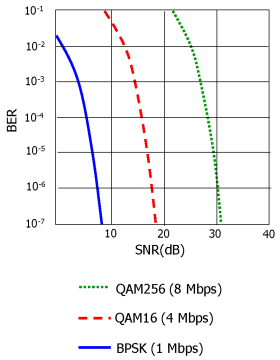


  - Punkte werden in gleichen Abständen im Phasenraum verteilt
  - Konstellationen mit 16, 64, 256 Punkten
  - verbreitete Verwendung

Quelle: Bernd Schürmann. Grundlagen der Rechnerkommunikation, Vieweg Verlag, 2004.

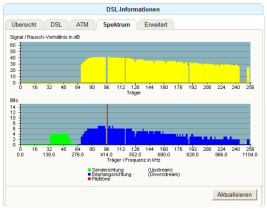
Modulation

- mit Anzahl von Signalwerten steigt neben Bitrate auch die Fehlerwahrscheinlichkeit:



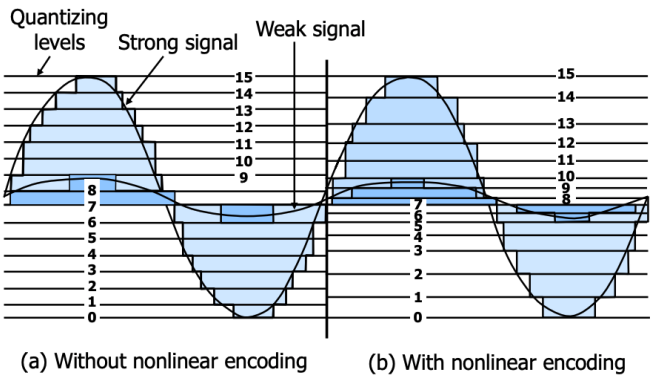
Modulation

- Mehrträgerverfahren: Discrete Multitone Transmission (DMT), Orthogonal Frequency Division Multiplexing (OFDM)
  - Frequenzband wird in Subkanäle zerteilt
  - zu übertragendes Signal wird in Symbole aufgeteilt, die in den Subkanälen parallel übertragen werden
  - Modulation in jedem Subkanal, z.B. QAM, jeweils angepasst an Rauschabstand in Subkanal
  - z.B. ADSL mit 255 Subkanälen:
  - Überlappung der Subkanäle durch orthogonale Schwingungen möglich
  - Verwendung u.a. in 802.11a/g/n/ac, Bluetooth 3.0, WiMAX, LTE, 5G
  - große Leistungssteigerung



6.3.3 Analog-Digital-Wandlung

Bei der **Analog-Digital-Wandlung** möchte man ein analoges modulierendes Signal in ein digitales Trägersignal wandeln, wie es zum Beispiel bei Sprachaufnahmen gemacht wird. Ein Verfahren zur Analog-Digital-Wandlung ist die **Pulscodemodulation**. Hierbei wird ein analoges Signal zu diskreten Zeitpunkten gemessen (**Abtastung**) und anschließend eine **Quantisierung** der Messwerte durchgenommen (*analoge Messwerte werden in digitale abgebildet*). Zum Schluss erfolgt die **Codierung** der quantisierten Werte. Hierbei werden diskrete Messwerte in digitale Signale umgewandelt und übertragen (*pulse codes*). Am Beispiel für ISDN, hierbei werden 8 Bit alle 125 Mikrosekunden übertragen, dementsprechend gibt es 256 Quantisierungsstufen und 8000 Abtastungen die Sekunde. Es werden 64 kbps erreicht. Die Quantisierung kann dabei mit einer linearen oder nicht-linearen Skala passieren:



6.4 Übertragungsarten

6.4.1 Übertragung über elektrische Leiter

Eigenschaften elektrischer Leiter Man unterscheidet **Bandbreiten** (*bandwidth*), **Wellenwiderstände** (*impedance*)[6], **Dämpfung** (*attenuation*)[7] und **Nebensprechen** (*crosstalk*)[8].

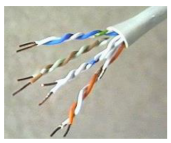
Arten von Kabeln Wir unterscheiden zwei Arten von Kabeln, **Koaxialkabel** und **Twisted-Pair-Kabel**.

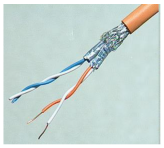
Koaxialkabel Koaxialkabel haben einen Kupfer-Innenleiter und einen Kupfergeflecht-Außenleiter. Die Dämpfung ist bei circa 0,15 $^{dB}/_{m}$ bei 100 MHz. Der Wellenwiderstand ist bei circa 50 Ω, deswegen sind Abschlusswiderstände notwendig. Sie sind abstrahlungsarm, werden deswegen beim klassischen Ethernet und bei Audio/Video-Übertragungen verwendet.

Twisted-Pair-Kabel (TP) Hier sind es verdrillte Kupferleiterpaare, deren Dämpfung bei circa 0,05 $^{dB}/_{m}$ bei 10 MHz lieft. Der Wellenwiderstand ist bei 100 bis 300 Ω, deswegen sind Abschlusswiderstände notwendig. Die Verdrillung und Schirmung reduziert die Störanfälligkeit. Sie sind relativ preiswert und sehr verbreitet.

Signalübertragung über elektrische Leitungen

- Bauarten von Twisted-Pair-Kabeln
  - Sternvierer: 2 Adernpaare, alle 4 Adern sind zu Spiralform verseilt, keine Schirmung, klassische Telefonkabel
  - UTP (Unshielded TP): ungeschirmt, meist 4 einzeln verseilte Adernpaare
  - S/UTP (Screened/Unshielded TP): Gesamtschirm, meist 4 einzeln verseilte Adernpaare
  - STP (Shielded TP): ungeschirmt, meist 4 einzeln verseilte Adernpaare mit Einzelschirm
  - S/STP (Screened/Shielded TP): Gesamtschirm, meist 4 einzeln verseilte Adernpaare mit Einzelschirm
  - FTP (Foilded TP) bzw. S/FTP (Screened/Foiled TP): Schirmung aus Metallfolie

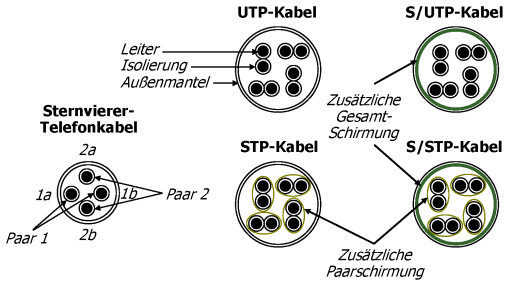




Quelle: www.wikipedia.de

Signalübertragung über elektrische Leitungen

- Bauarten von Sternvierer- und TP-Kabeln im Querschnitt:



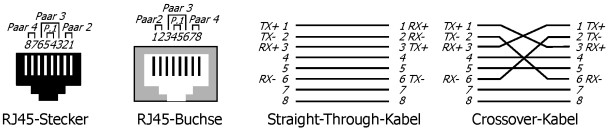
Quelle: J. Scherff: Grundkurs Computernetze, 2. Auflage, Vieweg Verlag, 2010.

Signalübertragung über elektrische Leitungen

- TP-Kabelstandards
  - Normen: amerik.: TIA/EIA 568, europ.: CENELEC EN 50173, internat.: ISO/IEC 11801
  - Kategorie 1: UTP, bis 100 KHz, Telefon und ISDN-Hausanschluss, Datenrate R < 1 Mbps
  - Kat 2: UTP, bis 1 MHz, ISDN-Primärmultiplexanschluss, R ≤ 4 Mbps
  - Kat 3: UTP, bis 16 MHz, Ethernet, Token Ring, R ≤ 4 Mbps
  - Kat 4: UTP, bis 20 MHz, Ethernet, Token Ring, R ≤ 16 Mbps
  - Kat 5: UTP, bis 100 MHz , Fast Ethernet, R ≤ 100 Mbps
  - Kat 5e: UTP, bis 100 MHz, Gigabit Ethernet
  - Kat 6/6a: UTP, STP, bis 250 MHz, Gigabit Ethernet
  - Kat 7/7a: STP, S/FTP, bis 600 MHz, 10 Gigabit Ethernet
  - Kat 8: S/FTP, bis 2 GHz, 40 Gigabit Ethernet

Signalübertragung über elektrische Leitungen

- TP-Steckverbindungen
  - RJ45 (Registered Jack Connectors, TIA/EIA 568)
  - Paarbelegung unterschiedlich (z.B. Ethernet Paare 2 und 3)
    - Straight-Through-Kabel: durchgeschleift, z.B. für Anschluss an Switch
    - Crossover-Kabel: Paare 2 und 3 gekreuzt, z.B. für Verbindung von Rechnern
    - Rollover-Kabel: alle Paare paarweise gekreuzt, z.B. für Router-Konsole
  - für Kat-7 und Kat-8 TP neue Steckverbindungen wg. Frequenzen



RJ45-Stecker RJ45-Buchse Straight-Through-Kabel Crossover-Kabel

Quelle: J. Scherff: Grundkurs Computernetze, 2. Auflage, Vieweg Verlag, 2010.

[6] frequenzabhängig und komplex, da sich für Hochfrequenzverhalten aufgrund von Reflexionsvermeidung alle Komponenten einen gleichen Wellenwiderstand haben.

[7] Signalabschwächung durch das Kabel, hängt maßgeblich von der Differenz zwischen Sende- und Empfangspegel ab.

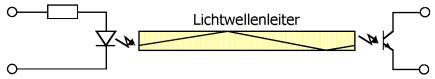
[8] Dämpfung durch andere Leiter

6.4.2 Übertragung über Lichtwellenleiter

Signalübertragung über Lichtwellenleiter

Quelle: Bernd Schürmann. Grundlagen der Rechnerkommunikation, Vieweg Verlag, 2004.

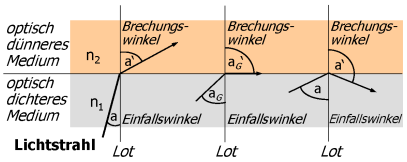
- Eigenschaften von Lichtwellenleitern
  - hohe Datenübertragungsrate (Gbps)
  - geringe Dämpfung (z.B. 0,5 dB/km möglich), daher große Entfernungen (mehrere 1000 km) möglich
  - keine Störung durch elektromagnetische Felder, kein Nebensprechen
  - geringe Bitfehlerrate
  - Abhörsicherheit?
  - Einsatz heute überall außer im Anschlussbereich
- Prinzip
  - Sender wandelt elektrisches Signal mit LED/Laserdiode in optisches Signal (Amplitudenmodulation: „1" = Puls, Frequenzmultiplex möglich)
  - Empfänger wandelt optisches Signal mit Fotodiode in elektrisches Signal



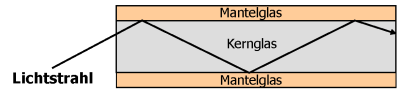
Quelle: Bernd Schürmann. Grundlagen der Rechnerkommunikation, Vieweg Verlag, 2004.

Signalübertragung über Lichtwellenleiter

- Führung des Lichtstrahls
  - Kern ist optisch dichteres, Mantel optisch dünneres Medium
  - Brechung zum optisch dichteren Medium
  - Totalreflexion bei Einfallswinkel > Grenzwinkel
  - kein Licht tritt aus, Leiter „führt" Lichtstrahl

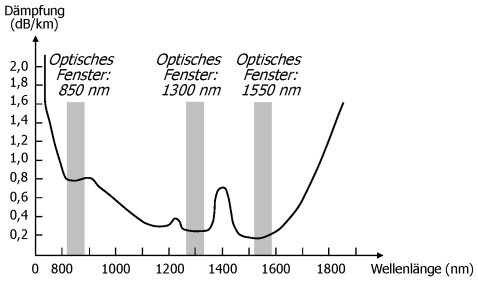


Quelle: J. Scherff: Grundkurs Computernetze, 2. Auflage, Vieweg Verlag, 2010.



Signalübertragung über Lichtwellenleiter

- Dämpfungsverhalten von Lichtwellenleitern
  - drei optische Fenster werden zur Signalübertragung genutzt



Quelle: J. Scherff: Grundkurs Computernetze, 2. Auflage, Vieweg Verlag, 2010.

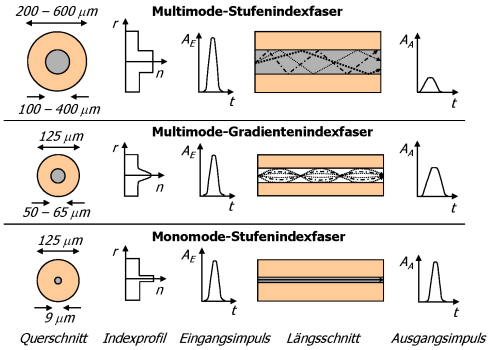
Signalübertragung über Lichtwellenleiter

- Moden in Lichtwellenleitern
  - Modus: Ausbreitungsweg eines Lichtimpulses, niedrig: fast parallel, hoch: zickzackförmig
  - Dispersion: Lichtimpuls teilt sich in verschiedene Moden auf, Verschlechterung des Empfangssignals
  - Bandbreitenlängenprodukt B·l: max. Impulsfrequenz bei Leiterlänge
    - z.B. bei 1 GHz · km sind möglich: 2 GHz bei 500 m, 1 GHz bei 1 km, 500 MHz bei 2 km
  - Glasfasertypen
    - Multimode-Stufenindexfaser: großer Durchmesser, sprunghafter Anstieg des Brechungsindex, ≤ 100 MHz · km
    - Multimode-Gradientenindexfaser: kontinuierlich ansteigender Brechungsindex, ca. 1 GHz · km
    - Monomode-Stufenindexfaser: kleiner Kerndurchmesser, stufenförmig ansteigender Brechungsindex, geringe Dispersion durch nur einen Modus, ≥ 10 GHz · km

Signalübertragung über Lichtwellenleiter

- Überblick über die Glasfasertypen:

Quelle: J. Scherff: Grundkurs Computernetze, 2. Auflage, Vieweg Verlag, 2010.



Signalübertragung über Lichtwellenleiter

- Verbindungstechnik und Stecker
  - Spleißtechnik: punktgenaue Verbindung von Glasfasern
    - Fusions-Spleiß (Lichtbogen)
    - Klebe-Spleiß
    - Crimp-Spleiß (klemmen)
  - Steckertechnik
    - Glasfasern sind am Faserende in Adernhülse (Ferrule) eingebettet, um punktgenaue Justierung beim Steckvorgang zu ermöglichen
    - Verspleißung
    - Breakoutkabel zur einfachen Montage
    - größere Vielfalt an standardisierten Steckern, z.B.
      - FSMA (Field Installable Subminiature Assembly), runder Schraubstecker, früher Standard
      - SC (Subscriber Connnector), EN 50173, Anschluss Endgeräte, weit verbreitet
      - MT-RJ: soll SC ersetzen, ähnlich zu RJ45

6.5 Strukturierte Verkabelung

Weil eine bedarfsorientierte Verkabelung zu unüberschaubarer Komplexität führt, gibt es mehrere Standards für eine strukturierte Verkabelung: ISO/IEC 11801 und CENELEC EN 50173. Man unterscheidet **hierarchische Verkabelungsbereiche**:

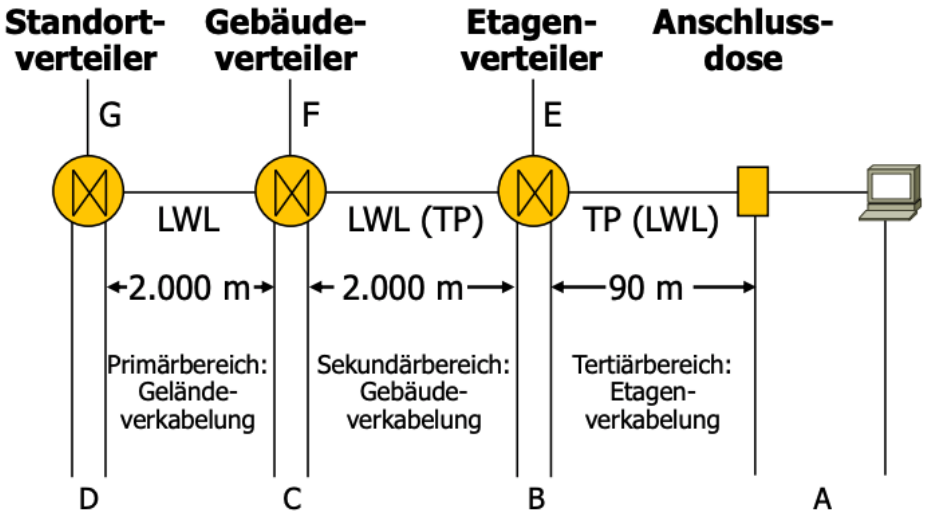
Primärbereich (Reichweite: Je nach LWL-Typ *maximal 2000 Meter*, **Glasfaser**)
Der **Primärbereich** ist eine Geländeverkabelung zwischen Gebäuden und eine Anbindung an

WAN über den Standortverteiler (*Campus Distributor, CD*).

Sekundärbereich (Reichweite: Je nach LWL-Typ *maximal 2000 Meter*, **Glasfaser** *empfohlen*, alt. TP mit *maximal 100 Metern*)
Der **Sekundärbereich** ist eine Gebäudeverkabelung mit dem zentralen Gebäudeverteiler (*Building Distributor, BD*) und verbindet Etagen über Steigleitungen.

Tertiärbereich (Reichweite: *maximal 90 Meter + 10 Meter*, TP *empfohlen*, *Glasfaser möglich*)
Im **Tertiärbereich** findet eine Etagenverkabelung von dem Etagenverteiler an Anschlussdosen (*Telecommunication Outlet, TO*) statt. Es werden Geräteanschlusskabel von *maximal 10 Meter Länge* (TP) verwendet, genauso wie **Patchkabel** zwischen Netzwerkkomponenten.

Bereiche und maximale Längen

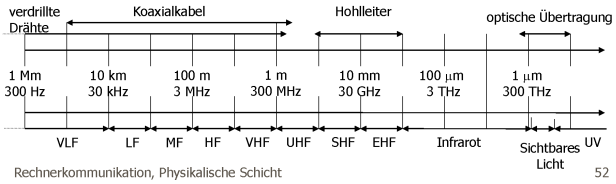


6.6 Funkübertragung

6.6.1 Grundlegende Eigenschaften

Funkübertragung

- Grundlegende Eigenschaften
  - kabellose Signalübertragung durch elektromagnetische Wellen
  - Frequenz f, Wellenlänge l, Lichtgeschwindigkeit c = 300.000 km/s
  - Zusammenhang: $c = l \cdot f$
  - z.B. 50 Hz u. 6.000 km, 1 MHz u. 300 m, 300 MHz u. 1m, 1 GHz u. 30 cm
  - regulierte Zuteilung der Funkfrequenzen (internat. ITU-R, USA FCC, Deutschland Bundesnetzagentur)
  - Industrial, Scientific, Medical (ISM): 3 Bänder, die mit allgemeiner Zulassung verwendet werden können, bei 433 MHz, 2,4 GHz, 5 GHz
  - elektromagnetisches Spektrum:



Funkübertragung

- Für Datenkommunikation relevant
  - Radiowellen
    - f ≈ 10 kHz .. 10 MHz
    - omnidirektionale Ausbreitung, Mbps möglich
    - große Entfernungen, Durchdringung von Hindernissen, störanfällig (z.B. Regen)
  - Mikrowellen
    - f ≈ 10 MHz .. 1 GHz: omnidirektionale Ausbreitung, Einsatz für PANs und LANs
    - f ≈ 1 GHz .. 100 GHz: geradlinige Ausbreitung möglich, Einsatz im Nahbereich und für Richtfunkstrecken und Satellitenübertragung, Gbps möglich
  - Infrarot
    - f ≈ $3 \cdot 10^{11}$ Hz .. $2 \cdot 10^{14}$ Hz, geradlinige Ausbreitung, Gbps möglich, bis ca. 1 km, Einsatz im Nahbereich und für optische Richtfunkstrecken, benötigt Sichtverbindung

6.6.2 Funksysteme

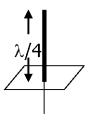
Funkübertragung

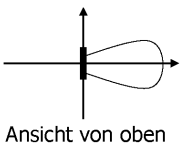
- Funksysteme
  - terrestrischer Funk: auf Erdoberfläche
    - Rundfunk (Radio + Fernsehen): Broadcast, Radiowellen, Sendereichweite 50 km bis zu Erdumrundung
    - Mobilfunk (Telefonie, Daten): Funksysteme mit Zellen und mobilen Teilnehmern, Anbindung an Vermittlungssystem über Basisstationen
    - Richtfunk: Verbindung von Gebäuden durch Richtfunkstrecken, optisch oder durch Mikrowellen
  - Satellitenfunk
    - als Fernmelde- (Daten, Sprache, Video) oder Fernsehsatelliten
    - Nutzung von Mikrowellen, höhere Frequenzbereiche
    - geostationär (ca. 36.000 km Höhe, 270 ms Latenz), MEO (medium earth orbit), LEO (low earth orbit)

6.6.3 Antennen

Funkübertragung

- Antennen
  - Abstrahlung und Aufnahme elektromagnetischer Felder
  - isotroper Punktstrahler (theoretische Bezugsantenne)
    - strahlt Wellen kugelförmig aus
    - Leistung auf konzentrischen Kugeln gleich
    - Dämpfung: Oberfläche $4\pi r^2$, Leistungsdichte S nimmt also mit $1/r^2$ ab
  - reale Antennen
    - z.B. Dipol, Parabol
    - besitzen Hauptstrahlrichtungen
    - Antennengewinn G: Verhältnis Leistungsdichte in Hauptstrahlrichtung zu isotropem Punktstrahler





gerichtete Antenne

Ansicht von oben

Quelle: Jochen H. Schiller, Vorlesungsunterlagen "Mobile Communications", FU Berlin

6.6.4 Rauschen

Funkübertragung

- Rauschen
  - Summe aller Störeffekte, die nur mit statistischen Methoden beschreibbar sind
  - z.B. atmosphärische Strahlung, thermisches Rauschen in Empfängerelektronik, Ungenauigkeiten bei Analog-/Digitalwandlung
  - verbreitetes Modell: Additive White Gaussian Noise (AWGN)
  - Varianz der Gauß-Verteilung ergibt Rauschleistungsdichte N
  - verbreitetes Maß: Signal-to-Noise-Ratio SNR = S/R

6.6.5 Interferenz

Funkübertragung

- Interferenz
  - Signale, die aufgrund nicht perfekter räumlicher, zeitlicher oder spektraler Trennung das Nutzsignal störend überlagern
    - Intersysteminterferenz: von anderen Systemen
    - Intersymbolinterferenz (ISI): durch Verschiebung aufeinanderfolgender Symbole beim Empfänger
  - Ursachen
    - Abschattung durch Hindernisse
    - Spiegelung (Reflexion) an Flächen größer als Wellenlänge
    - Beugung (Diffraction) an scharfen Kanten
    - Streuung (Scattering) an kleinen Hindernissen
    - Brechung (Refraction) in Abhängigkeit der Dichte eines Mediums
    - Dopplereffekt bei mobilem Sender/Empfänger
  - Abhilfe: Diversitätstechnologien (Zeit, Frequenz, Raum)

6.6.6 Ausbreitung

Funkübertragung

- Ausbreitung über Sichtverbindung (Line-of-Sight, LOS)
  - typisch bei Richtfunkübertragung
  - Dämpfung i.w. gleich der Freiraumausbreitungsdämpfung, andere Interferenzen nicht signifikant
  - Fresnelzone: Ellipsoid um direkte Verbindung zwischen Sender und Empfänger mit Abstand d, breitester Durchmesser

$$D_{Fresnel} = \sqrt{d \cdot \lambda}$$

  - Bedingung für LOS-Verbindung: Ellipsoid mit Durchmesser $D = 0{,}6 \cdot D_{Fresnel}$ ist frei von Hindernissen
  - Dämpfung der Empfangsleistung:

$$P_{Rx} = P_{tx}\left(\frac{\lambda}{4\pi d}\right)^2 G_{Tx} G_{Rx} = P_{tx}\left(\frac{c}{4\pi d f}\right)^2 G_{Tx} G_{Rx}$$

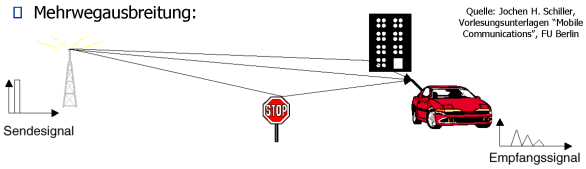
→ quadratisch in Abstand und Frequenz

Funkübertragung

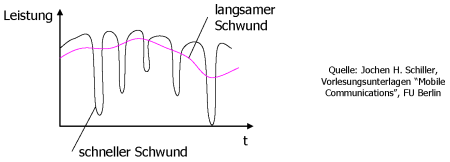
- Ohne Sichtverbindung (Non-Line-of-Sight, NLOS)
  - typisch bei Mobilkommunikation mit Benutzermobilität
  - mittlere Übertragungsdämpfung durch Hindernisse größer als bei Freiraumübertragung, proportional d^g mit $2 \le g \le 5$
  - Abschattungsdämpfung
    - zusätzliche variable Dämpfung durch Hindernisse wie Gebäude, Berge
    - langsamer Schwund (Slow Fading)
  - Mehrwegausbreitung
    - durch Reflexion, Beugung und Streuung entstehen unterschiedliche Ausbreitungswege zum Sender
    - diese sind phasenverschoben und überlagern sich beim Empfänger
    - hierdurch kann Signal zeitweise stark abgeschwächt werden
    - schneller Schwund (Fast Fading)

Funkübertragung

- Mehrwegausbreitung:



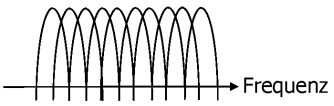
- typischer Verlauf der Signalstärke bei einem Empfänger in Bewegung:



6.6.7 Mehrträgerverfahren

Funkübertragung

- Mehrträgerverfahren: Orthogonal Frequency Division Multiplexing (OFDM)
  - zu übertragendes Signal wird wie bei QAM in Symbole aufgeteilt
  - je N Symbole werden so auf höhere Frequenzen moduliert, dass die Signale orthogonal zueinander sind (Integral des Produkts gleich 0)
  - diese N Signale werden in parallelen Subkanälen übertragen



  - durch Orthogonalität können Subkanäle enger liegen als bei FDM
  - damit verringert sich Bitrate pro Subkanal und damit ISI
  - realisiert durch fortgeschrittene Signalverarbeitung (Fast Fourier Transformation und Inverse FFT)
  - Adaptivität an Störverhältnisse im Subkanal möglich
  - insgesamt große Leistungssteigerung bei kleiner Fehlerrate
  - auch als Multiplextechnik: Kombination von Signalen, OFDMA

6.6.8 Multiantennentechnik

Funkübertragung

- Multi-Antennentechnik
  - Anordnung mehrerer Sende- bzw. Empfangsantennen (NxM), z.B. N, M = 2, 4, 8, Abstand in Größenordnung mehrerer Wellenlängen
  - ermöglicht Erhöhung der Zuverlässigkeit, Kapazität, Energieeffizienz, Reduktion der Interferenz
  - Spatial Diversity: Erhöhung der Zuverlässigkeit durch Überlagerung der gleichen Signale verschiedener Antennen
  - Spatial Multiplexing: Erhöhung der Kanalkapazität durch parallele Übertragung zwischen verschiedenen Antennen, Multiple Input Multiple Output (MIMO)
  - Beamforming: jede Sendeantenne sendet gleiches Signal mit jeweils leichtem Phasenversatz, dadurch konstruktive Interferenz in Hauptstrahlrichtung, destruktive Interferenz in anderen Richtungen, analog zum Empfang aus Haupteinfallrichtung, Abstand in Größenordnung einer Wellenlänge benötigt
  - Massive MIMO: einige 100 oder 1000 Sendeantennen, einige 10 Empfänger

6.6.9 Beispiele und Ultrabreitband

Funkübertragung

- Ultrabreitband (Ultra-Wideband, UWB)
  - Übertragung von sehr kurzen Impulsen (z.B. 1 ns) mit sehr großer Bandbreite (> 500 MHz) z.B. im 2,4-GHz-ISM-Band
  - diese erreichen fast Rechtecksignal und benötigen deswegen großes Spektrum, für das aber kleine Leistung reicht (ähnlich wie Rauschen)
  - ermöglicht große Datenraten (> 500 Mbps) über kurze Entfernungen, Hindernisse (wie Wände) können durchdrungen werden
  - Entfernungsmessung auch möglich

Funkübertragung

- Beispiele für die Verwendung von Funktechnologie
  - GSM
    - FDMA+TDMA, GMSK (Gauss-Filter+FSK), 900 MHz, 13 Kbps/Kanal
    - EDGE: 8PSK 69,2 Kbps/Kanal
  - UMTS, Release 99: CDMA-DSSS, QPSK, bis 384 Kbps/Kanal
  - HSPA+: 64-QAM und MIMO bis zu 337,5 Mbps (DL), 23 Mbps (UL)
  - LTE: OFDM und MIMO bis zu 300 Mbps (DL), 75 Mbps (UL)
  - LTE-Advanced: OFDM und MIMO bis zu 3 Gbps (DL), 1,5 Gbps (UL)
  - WLAN
    - 802.11-1997: DSSS, DQPSK, 2,4 GHz, bis 2 Mbps
    - 11g-2003: DSSS, OFDM, 2,4 GHz, bis 54 Mbps
    - 11n: OFDM bis 100 Mbps, MIMO bis 600 Mbps
  - Bluetooth
    - 802.15.1: FHSS, GFSK, 2,4 GHz, bis 723,2/57,6 Kbps (ursprünglich, diverse weitere Varianten)
  - ZigBee
    - 802.15.4: DSSS, BPSK bei 868 MHz (20 Kbps) und 915 MHz (40 Kbps), QPSK bei 2,4 GHz (250 Kbps)

Anhang A

Merkblatt

Paketverzögerung

1. Bitrate R in $\frac{b}{s}$
2. Paketgröße L in Bit
3. Ausbreitungsverzögerung $D = \frac{l}{v}$, wobei l die Länge der Verbindung und v die Signalausbreitungsgeschwindigkeit ist
4. Kanalpuffergröße in Bit: $R \cdot D = R \cdot \frac{l}{v} = \frac{\frac{l}{v}}{\frac{1}{R}}$
   $\Longrightarrow$ Anzahl der gesendeten Bits während 1. Bit sich vom Sender zum Empfänger ausbreitet (*Einführung, 21*)
   a) $RD > 1$:
      i. $t = D$: erstes Bit erreicht Empfänger, $R \cdot D$ Bits versendet
   b) $RD < 1$:
      i. $t = D$: Anfang des Bits empfangen, $R \cdot D \cdot 100\%$ des Bits versendet
      ii. $t = \frac{1}{R}$: Bit komplett gesendet
      iii. $t = \frac{1}{R} + D$: Bit komplett empfangen
5. Kanalpuffergröße in Paketen $a = \frac{R \cdot D}{L} = \frac{\frac{l}{v}}{\frac{L}{R}}$
6. Sendezeit leitungsverb. : $R_{eff} = \frac{R}{x}$ mit x den Slots pro Sekunde/Frequenz
7. Paketverzögerung: (*Übung 1, 11 — Übung 2, 2*)

$$d = \underbrace{d_{trans}}_{\text{Übertragung, Bits auf Link, } \frac{L}{R}} + \underbrace{d_{prop}}_{\text{Ausbreitung im Medium } \frac{l}{v}} + \underbrace{d_{proc}}_{\text{Verarbeitung (Bitfehler, Links bestimmen)}} + \underbrace{d_{queue}}_{\text{Warteschlange (Wartezeit auf Link)}}$$

8. Paketverzögerung mit mehreren Links: (*Übung 2.6*)

$$\left(\sum_{i=1}^{E} d_{trans_i} + d_{prop_i} + d_{proc_i} + d_{queue_i} \right) + d_{proc}$$

9. Übertragung in Paketen über mehrere Links: (*Übung 1, 25*)

$$d_{trans} = n_{links} \cdot \frac{L}{R} + (n_{pakete} - 1) \cdot \frac{L}{R}$$

(wenn leitungsvermittelt: $n_{links} = 1$)

10. mittlere Warteschlangenverzögerung (*Übung 2.4*)

$$d = \frac{d_{ges}}{N}$$
$$d_{ges} = \sum_{i=1}^{N} (i-1) \cdot \frac{L}{R} = \frac{N \cdot (N-1)}{2} \cdot \frac{L}{R}$$

11. Verkehrsintensität: (*Übung 1, 20 — Übung 2.5*)

    λ: durchschnittliche Paketankunftsrate (Pakete pro Sekunde)

$$\rho = \frac{L \cdot \lambda}{R} \begin{cases} d_{queue} \text{ fällt} & \text{falls } \rho < 1 \\ d_{queue} \text{ steigt} & \text{falls } \rho = 1 \\ d_{queue} \to \infty, \text{ da mehr Pakete ankommen als abgearbeitet werden} & \text{falls } \rho > 1 \end{cases}$$

12. Wahrscheinlichkeit, dass n Nutzer gleichzeitig senden: (*Übung 1.3c*)
    $P_n = \binom{max}{n} \cdot (P_{user})^n \cdot (1 - P_{user})^{max-n}$

13. Wahrscheinlichkeit, dass mehr als n Nutzer gleichzeitig senden: (*Übung 1.3d*)
    $\sum_{n}^{max} P_i = 1 - \sum_{i=0}^{n} P_i$

TCP-Leistungsanalyse

1. Verbindungsaufbau: $2 \cdot RTT$ (*Transport, 134*) für:

   a) $C \to S$: `SYN` (seqnum: client_isn)

   b) $S \to C$: `SYN + ACK` (seqnum: server_isn, acknum: client_isn + 1)

   c) $C \to S$: `ACK + Request` (seqnum: client_isn + 1, acknum: server_isn + 1)

2. Verbindungsabbau: (*Transport, 136*)

   a) $C \to S$: `FIN` (seqnum: client_sqn)

   b) $S \to C$: `ACK` (seqnum: server_sqn, acknum: client_sqn + 1)

   c) $S \to C$: `FIN` (seqnum: server_sqn)

   d) $C \to S$: `ACK` (seqnum: client_sqn + 1, seqnum: server_sqn + 1)

   e) C wartet noch 2 Segmentlebensdauern auf mögliche alte Segmente („time wait“), S ist nach dem Erhalt des letzten ACKs fertig

3. festes Fenster, keine Wartezeiten: (*Transport, 162 — Übung 4, 9 — Übung 5, 4*)

$$d = \underbrace{2 \cdot RTT}_{\text{Verbindungsaufbau}} + \underbrace{\frac{O}{R}}_{\text{Datei übertragen}}$$

   tritt ein falls gilt:

$$\underbrace{\frac{WL}{R}}_{\text{Senden eines Fensters der Größe } WL} \geqslant \underbrace{\frac{L}{R}}_{\text{Senden Segment}} + \underbrace{RTT}_{\text{ACK von Server bei Client}}$$

4. festes Fenster, mit Wartezeit (also $\frac{WL}{R} < \frac{L}{R} + RTT$): (*Transport, 163 — Übung 4, 8 — Übung 5, 6*)
   Wartezeit pro Fenster:
   $$\frac{L}{R} + RTT - \frac{WL}{R}$$
   K Fenster $\Rightarrow (K-1)$-mal Warten, Anzahl benötigter Fenster $K := \left\lceil \frac{O}{WL} \right\rceil \Rightarrow (K-1)$-mal Warten
   $$d = 2RTT + \frac{O}{R} + (K-1)\left(\frac{L}{R} + RTT - \frac{WL}{R}\right)$$

5. Dynamisches Fenster: (*Transport, 164 — Übung 5, 8*)
   $$d = 2RTT + \frac{O}{R} + d_{SSW}$$
   warten, wenn $\frac{WL}{R} \leqslant \frac{L}{R} + RTT \Rightarrow$ Wartezeit nach K. Fenster: $\max\{\frac{L}{R} + RTT - 2^{K-1} \cdot \frac{L}{R}; 0\}$
   Wartezeiten bis Fenster Q: $\max\{k : T \cdot \frac{L}{R} + RTT - 2^{k-1} \cdot \frac{L}{R}; 0\}$
   Anzahl Fenster K um O zu übertragen: $K = \min\{k : \sum_{i=1}^{k} x^{i-1} \cdot L >= 0\}$ (*Transport, 166 — Übung 5, 9*)

   $$Q = \left\lfloor \log_2\left(1 + \frac{RTT}{\frac{L}{R}}\right)\right\rfloor + 1 \qquad \text{bis zu welchem Fenster Wartezeiten}$$
   $$K = \left\lceil \log_2\left(\frac{O}{L} + 1\right)\right\rceil \qquad \text{Anzahl Fenster um O zu übertragen}$$
   $$\underbrace{P}_{\text{Anzahl SSW}} = \min\{\ \underbrace{Q}_{\substack{\text{Nicht mehr warten,}\\ \text{O groß genug}}}\ ;\ \underbrace{K-1}_{\substack{\text{Warten,}\\ \text{wenn O zu klein}}}\ \} \qquad \text{Wartezeit bei endlich großem Objekt O}$$
   $$d = 2RTT + \frac{O}{R} + P \cdot \left(RTT + \frac{L}{R}\right) - \left(2^P - 1\right) \cdot \frac{L}{R}$$

6. mehrere Links T: (*Übung 5.2*)
   $$d = 2 \cdot RTT_T + \frac{O}{R} + (T-1) \cdot \frac{L}{R} + P_T \cdot \left(RTT_T + \frac{TL}{R}\right) - \left(2^{P_T} - 1\right) \cdot \frac{L}{R}$$
   $$Q_T = \left\lfloor \log_2\left(T + \frac{RTT}{\frac{L}{R}}\right)\right\rfloor + 1$$
   $K =$ wie normal

7. um Faktor 2 erhöhen, also Fenstergröße W^3: (*Übung 5.1*)
   $$d = 2 \cdot RTT + \frac{O}{R} + P \cdot \left(\frac{L}{R} + RTT\right) - \frac{P-1}{2} \cdot \frac{L}{R}$$
   $$Q = \left\lfloor \log_3\left(1 + \frac{RTT}{\frac{L}{R}}\right)\right\rfloor + 1$$
   $$K = \left\lceil \log_3\left(\frac{2O}{L} + 1\right)\right\rceil$$

8. Verzögerung HTTP mit M Objekten und 1 Basisseite:

a) nicht-persistent: (*Übung 5, 18 — Aufgabe 5.1*)

$$(M+1)\left(2\cdot RTT+\frac{O}{R}+SSW\right)$$

b) persistent mit Pipelining: Q und K konstant (*Übung 5, 18 — Aufgabe 5.3*)

$$2\,RTT+\underbrace{RTT}_{\text{Anfrage Bilder}}+(M+1)\cdot\frac{O}{R}+\text{gem. SSW}$$

$$\text{gem. SSW}=SSW+RTT-\max\left\{\frac{L}{R}+RTT-\left(2^{K-1}\cdot\frac{L}{R}\right);0\right\}$$

c) nicht-persistent mit X parallelen Verbindungen ($\frac{M}{X}$ ganze Zahl):
in $SSW_{parallel}$ R durch $\frac{R}{X}$ ersetzen (*Übung 5, 18 — Aufgabe 5.2*)

$$\left(\frac{M}{X}+1\right)\cdot 2\cdot RTT+(M+1)\cdot\frac{O}{R}+SSW_{normal}+SSW_{parallel}$$

9. Flusskontrolle: (*Transport, 144*)

a) effective window $>0\Rightarrow$ senden

b) LastByteWritten $-$ LastByteAcked $\leqslant$ MaxSendBuffer $\Rightarrow$ Anwendung schreibt (*Transport, 147*)

c) Freier Speicher bei Empfänger:
advertised window = maxRcvBuffer $-$ ((NextByteExpected -1) $-$ LastByteRead)
(*Transport, 146*)

d) Falls mehr als 16 Bit für Fenstergröße benötigt werden: Scaling Factor $F\leqslant 14$, dann gilt
window = advertised window $\cdot\, 2^F$ (*Transport, 150*)

10. Überlastkontrolle:

a) max window = min{congestion window; advertised window} (*Transport, 152*)

b) effective window = max window $-$ (LastByteSent $-$ LastByteAcked) (*Transport, 152*)

c) Bitrate $R\approx\frac{\text{congestion window}}{RTT}$

d) Slow-Start: (*Transport, 153f*)

i. anfangs congestion window = MSS, nach jedem ACK congestion window+ = MSS (also exponentiell), bis 3 doppelte ACKs eintreffen oder Threshold (anfangs unendlich) erreicht

ii. Multiplicative Decrease (3 doppelte ACKs):
Threshold $=\frac{\text{congestion window}}{2}$; congestion window/ $=2$

iii. Additive Increase: Mit jedem ACK: congestion window+ $=\text{MSS}\cdot\frac{\text{MSS}}{\text{congestion window}}$

iv. nach Timeout: Threshold $=\frac{\text{congestion window}}{2}$; congestion window = MSS (dann wird Slow-Start bis zum Threshold und danach AIMD durchgeführt)

Bitfehlerwahrscheinlichkeiten

(*Transport, 12*)

1. mindestens 1 Bitfehler im Segment: $1-(1-p)^L$
2. 2 Bitfehler im Segment: $P_{2\_Fehler} := \#_{Paare} \cdot P_{bestimmtes\ Paar\ fehlerhaft} = \frac{(L-1)\cdot L}{2} \cdot P_{bestimmtes\ Paar\ fehlerhaft}$
3. mittlere Anzahl Pakete bis 2 Bitfehler im gleichen Segment: $\frac{1}{P_{2\_Fehler}}$

Fehlerkontrolle

1. Stop-And-Wait: (*Transport, 18*)
   a) (Leitungs-)Durchsatz: (*Transport, 92*)
   $$\frac{L}{N\cdot\left(\frac{L}{R}+2D\right)}$$
   b) Normierter Durchsatz: (*Transport, 92*)
   $$S=\frac{1}{N(1+2a)}$$
   c) Falls keine Fehler: $N=1$ (*Transport, 91 — Übung 4, 3*)
   d) Mittlere Anzahl der Sendeversuche pro Paket bei Fehlerwahrscheinlichkeit p: (*Transport, 93*)
   $$N=\sum_{i=1}^{\infty} i\cdot p^{i-1}\cdot(1-p)=\frac{1}{1-p}\Rightarrow S=\frac{1-p}{1+2a}$$
2. Selective-Repeat (mit Pakete auf Kanal $a=\frac{R\cdot D}{L}$):
   a) 1. Fall: Fenster groß genug, dass bis zum ACK-Empfang gesendet werden kann, also $W>1+2a$:
      i. ohne Fehler: (*Transport, 95 — Übung 4, 9*)
      $$S=\frac{W\cdot L}{W\cdot\frac{L}{R}}\cdot\frac{1}{R}=1$$
      ii. mit Fehlerwahrscheinlichkeit p: (*Transport, 99 — Übung 4, 16*)
      $$S=1-p$$
   b) 2. Fall: Es muss auf ACK gewartet werden, also $W<1+2a$:
      i. ohne Fehler: (*Transport, 95 — Übung 4, 8*)
      $$S=\frac{W\cdot L}{\frac{L}{R}+2D}\cdot\frac{1}{R}=\frac{W}{1+2a}$$
      ii. mit Fehlerwahrscheinlichkeit p: (*Transport, 99 — Übung 4, 16*)
      $$S=\frac{W\cdot(1-p)}{1+2a}$$

3. Go-Back-N (ohne Fehler wie Selective-Repeat) mit Fehlerwahrscheinlichkeit p:

   a) Mittlere Anzahl Sendeversuche pro Paket:

$$N = \sum_{i=1}^{\infty} ((1-K) + K_i) \cdot p^{i-1} \cdot (1-p) = \frac{1-p+K \cdot p}{1-p}$$

   b) $W \geqslant 1+2a$: $(K = 1+2a)$ (*Transportschicht, 101 — Übung 4, 18*)

$$S = \frac{1-p}{1+2ap}$$

   c) $W < 1+2a$: $(K = W)$ (*Transportschicht, 101 — Übung 4, 18*)

$$S = \frac{W \cdot (1-p)}{(1-p+W \cdot p) \cdot (1+2a)}$$

Sequenznummernraum mit $m = 2^n$ Werten

(*Transport, 61 — Transport, 81*)

1. Falls Empfangsfenstergröße = 1: $W < m$ hinreichend
2. Falls Sendefenstergröße = Empfangsfenstergröße > 1: $W < \frac{m+1}{2}$

Leistungsanalyse Medienzugriff

(*Sicherung, 19*)

1. (Slotted-)ALOHA: (*Sicherung, 28*)

   a) Wahrscheinlichkeit für Senden ohne Kollision bei Sendewahrscheinlichkeit p: (*Sicherung, 36*)

$$\underbrace{p}_{\text{Knoten sendet}} \cdot \underbrace{P}_{\text{kein anderer Knoten sendet in } [t_0-1, t_0]} \cdot \underbrace{P}_{\text{kein anderer Knoten sendet in } [t_0, t_0+1]}$$
$$= p \cdot (1-p)^{N-1} \cdot (1-p)^{N-1} = p \cdot (1-p)^{2N-2}$$

   b) Normalisierter Durchsatz: (*Sicherung, 36*)

$$S = N \cdot p(1-p)^{2(N-1)}$$

   c) Sendeversuche pro Slot: $G = N \cdot p \Rightarrow p = \frac{G}{N}$ (*Sicherung, 37*)

      i. ALOHA: $G = 0{,}5$ (*Sicherung, 37*)

      ii. Slotted-ALOHA: $G = 1$ (*Sicherung, 39*)

2. CSMA (*Sicherung, 41 — Übung 8, 4*)

   a) notwendige Voraussetzung: Ausbreitungsverzögerung d_{prop} < Rahmensendezeit $\frac{L}{R}$ (*Sicherung, 42*)

   b) verwendet ACKs

c) bei fehlendem ACK nach Timeout ⇒ Backoff (warten) ⇒ Resend

d) Varianten: (*Sicherung, 43 — Übung 8, 6*)

i. 1-persistent: Bei belegtem Medium warten bis frei, dann sofort senden

ii. nicht-persistent: Bei belegtem Medium Backoff

iii. p-persistent: Wenn Medium wieder frei, Senden mit Wahrscheinlichkeit p oder einen weiteren Slot abwarten $(1-p)$

3. CSMA/CD (*Sicherung, 48 — Übung 8, 11 — Übung 8.2*)

a) mit allen CSMA-Varianten kombinierbar

b) keine ACKs

c) verwendet „listen while talking“

d) Normierter Durchsatz: (*Sicherung, 52*)

$$\begin{aligned} S_{max} &= \frac{Sendezeit}{Sendezeit + Ausbreitungszeit + Wettbewerbszeit} \\ &= \frac{\frac{L}{R}}{\frac{L}{R} + D + (e-1)\cdot 2D} = \frac{1}{1+4{,}4a} \end{aligned}$$

e) Minimale Rahmengröße L: (*Sicherung, 50*)

$$\frac{L}{R} > 2\cdot D \Rightarrow L > 2\cdot R\cdot D$$

f) Erfolg

$$\begin{aligned} p &= \frac{1}{N} \\ P_{erfolg} &= N\cdot p(1-p)^{N-1} \\ (P_{erfolg})^{max} &= \frac{1}{e} \end{aligned}$$

⇒ alle e Slots erfolgreicher Sendeversuch

4. Ethernet (*Sicherung, 63*)

a) 1-persistentes CSMA/CD

b) Hub: eine Kollisionsdomäne, Signal wird auf alle Ports weitergeleitet (*Sicherung, 66*)

c) Bridge: Aufteilung in verschiedene Kollisionsdomänen, enthält „Switching Fabric“ (*Sicherung, 67*)

d) VLAN entweder port- oder tag-basiert (*Sicherung, 72*)

Verteilte Hash-Tabellen

(*Anwendung, 87*)

1. jeder Peer hat Bezeichner $p \in [0, 2^m)$ bei m Bits
2. jedes Datenelement hat Schlüssel k aus gleichem Raum
3. Speicherung und Auslesen von k auf Peer $p = succ(k) = (k+a) \mod 2^m$ mit minimalem a

4. Effizienzsteigerung durch Fingertabellen auf jedem Peer p: (*Anwendung, 89 — Übung 3.1*)
   a) m Einträge:
   $$FT_p[i] = \left(p + 2^{i-1}\right) \mod 2^m$$
   b) Lookup von k beginnend auf beliebigem Peer p:
   $$\text{lookup}(k,p) = \begin{cases} p & ,p = k \\ FT_p[1] & ,p < k \leqslant FT_p[1] \\ \text{lookup}(k, FT_p[i]) & ,FT_p[i] \leqslant k < FT_p[i+1] \text{ mit } 1 < i < m \\ \text{lookup}(k, FT_p[m]) & ,FT_p[m] \leqslant k \end{cases}$$

Cyclic Redundancy Check

(*Sicherung, 14*)

1. Nutzdaten D mit d Bits, Prüfdaten R mit r Bits, Generatorpolynom G mit $r+1$ Bits
2. R ist Rest bei $(D \cdot 2^r)/G$
3. (D,R) ist folglich $(D \cdot 2^r) \oplus R$
4. Korrekt falls $(D,R)/G = 0$

Shannons Theorem

1. Maximale Datenrate R bei Bandbreite B: $R = B \cdot \log_2(1 + \frac{S}{N})$ (*Physikalisch, 24*)
2. x dB $= 10\log_{10}\frac{S}{N} \Rightarrow \frac{S}{N} = 10^{\frac{x}{10}}$ (*Physikalisch, 24*)

Mathematisches

1. Binomialkoeffizient:
   $$\binom{n}{k} = \frac{n!}{k! \cdot (n-k)!}$$
2. Gaußsche Summenformel:
   $$\sum_{k=1}^{n} k = \frac{n \cdot (n+1)}{2}$$
3. Quadratische Pyramidalzahl:
   $$\sum_{k=1}^{n} k^2 = \frac{n \cdot (n+1) \cdot (2n+1)}{6} = \frac{2n^3 + 3n^2 + n}{n}$$
4. Geometrische Reihe für $|q| < 1$:
   $$\sum_{k=0}^{\infty} \left(a_0 \cdot q^k\right) = \frac{a_0}{1-q}$$

ANHANG B

PROGRAMMIERBEISPIELE

B.1 HTTP Web-Server

```
1  import java.io.* ;
2  import java.net.* ;
3  import java.util.* ;
4
5  public final class WebServer {
6
7      public static void main(String argv[]) throws Exception
8      {
9          // set the port number:
10         int port = 12345;
11
12         // establish the listen socket:
13         ServerSocket listenSocket = new ServerSocket(port);
14
15         // process HTTP service requests in an infinite loop:
16         while (true) {
17             // listen for a TCP connection request:
18             Socket connectionSocket = listenSocket.accept();
19
20             // construct an object to process the HTTP request message:
21             HttpRequest request = new HttpRequest(connectionSocket);
22
23             // create a new thread to process the request:
24             Thread thread = new Thread(request);
25
26             // start the thread:
27             thread.start();
28         }
29     }
30 }
31
32 final class HttpRequest implements Runnable {
33
34     final String EOL = "\r\n";
35     Socket socket;
36
37     // constructor:
38     public HttpRequest(Socket socket) throws Exception {
39         this.socket = socket;
40     }
41
42     // implement the run() method of the runnable interface:
43     public void run() {
44         try {
45             processRequest();
46         }
47         catch (Exception e) {
48             System.out.println(e);
49         }
50     }
51
52     private void processRequest() throws Exception {
53         // get a reference to the sockets input and output streams:
54         InputStream is = socket.getInputStream();
55         DataOutputStream os = new DataOutputStream(socket.getOutputStream());
```

```
56
57          // Set up input & output stream filters for ASCII-HTTP-Header.
58          BufferedReader br = new BufferedReader(new InputStreamReader(is));
59          BufferedWriter bw = new BufferedWriter(new OutputStreamWriter(os, "ASCII"));
60
61          // get the request line of the HTTP request message:
62          String requestLine = br.readLine();
63
64          // display the request line:
65          System.out.println();
66          System.out.println(requestLine);
67
68          // get and display the header lines:
69          String headerLine = null;
70          while ((headerLine = br.readLine()).length() != 0) {
71              System.out.println(headerLine);
72          }
73
74          // extract the filename from the request line:
75          StringTokenizer tokens = new StringTokenizer(requestLine);
76          tokens.nextToken();  // skip over the method, which should be "GET"
77          String fileName = tokens.nextToken();
78
79          // prepend a "." so that file request is within the current directory:
80          fileName = "." + fileName;
81
82          if (fileName.contains("../")) {
83              fileName = "./index.html";
84          } else if (fileName.endsWith("/")) {
85              fileName = fileName + "index.html";
86          }
87
88          // open the requested file:
89          FileInputStream fis = null;
90          boolean fileExists = true;
91          try {
92              fis = new FileInputStream(fileName);
93          } catch (FileNotFoundException e) {
94              fileExists = false;
95          }
96
97          // construct the response message:
98          String statusLine = null;
99          String contentTypeLine = null;
100         String entityBody = null;
101         if (fileExists) {
102             statusLine = "HTTP/1.0␣200␣OK" + EOL;
103             contentTypeLine = "Content-Type:␣" + contentType(fileName) + EOL;
104         } else {
105             statusLine = "HTTP/1.0␣404␣Not␣Found" + EOL;
106             contentTypeLine = "Content-Type:␣" + contentType(".html") + EOL;
107             entityBody = getNotFoundBody();
108         }
109
110         // send the status line:
111         bw.write(statusLine);
112
113         // send the content type line:
114         bw.write(contentTypeLine);
115
116         // send a blank line to indicate the end of the header lines:
117         bw.write(EOL);
118         bw.flush();
119
120         // send the entity body:
121         if(fileExists) {
122             sendBytes(fis, os);
123             fis.close();
124         } else {
125             bw.write(entityBody);
126             bw.flush();
127         }
128
129         // Close streams and socket.
130         os.close();
131         br.close();
```

```
132         socket.close();
133     }
134 
135     private static void sendBytes(FileInputStream fis, OutputStream os) throws Exception {
136         // construct a 1K buffer to hold bytes on their way to the socket:
137         byte[] buffer = new byte[1024];
138         int bytes = 0;
139 
140         // copy requested file into the sockets output stream:
141         while((bytes = fis.read(buffer)) != -1 ) {
142             os.write(buffer, 0, bytes);
143         }
144     }
145 
146     private static String contentType(String fileName) {
147         if(fileName.endsWith(".htm") || fileName.endsWith(".html")) {
148             return "text/html; charset=utf-8";
149         }
150         if(fileName.endsWith(".jpg") || fileName.endsWith(".jpeg") || fileName.endsWith(".jfif"))
     {
151             return "image/jpeg";
152         }
153         if(fileName.endsWith(".gif")) {
154             return "image/gif";
155         }
156         return "application/octet-stream";
157     }
158 
159     private String getNotFoundBody() {
160         return /*- Hier steht der HTML-Code der 404-Seite ... */;
161     }
162 }
```

Quellkode B.1 — WebServer

B.2 Alternating-Bit-Protokoll

```
1  import fau.cs7.nwemu.AbstractHost;
2  import fau.cs7.nwemu.NWEmu;
3  import fau.cs7.nwemu.NWEmuMsg;
4  import fau.cs7.nwemu.NWEmuPkt;
5  
6  class ABPCommonHost extends AbstractHost {
7      protected int seqnum, acknum;
8  
9      @Override
10     public void init() {
11         seqnum = 0;
12         acknum = 0;
13     }
14 }
15 
16 class ABPSendingHost extends CommonHost {
17     // Wir halten das zuletzt gesendete Paket aktuell um es im Fehlerfall
18     // nochmals schicken zu koennen.
19     NWEmuMsg lastMsg = null;
20     // Solange wir keine ACK erhalten haben und uns **in einem Sende-
21     // vorgang ** befinden, ist isSending wahr.
22     boolean isSending = false;
23 
24     private static final double TIMEOUT = 1000.0;
25 
26     /**
27      * @return true, falls das Paket gueltig ist und es versendet
28      *         haette werden koennen, false, andernfalls.
29      */
30     @Override
31     public Boolean output(NWEmuMsg nwEmuMsg) {
32         if(isSending)
33             return false;
```

```
34
35          isSending = true;
36          lastMsg = nwEmuMsg;
37
38          if(nwEmuMsg == null) {
39              sysLog(1, "ERROR:␣nwEmuMsg␣IS␣NULL!");
40              return false;
41          } else if(nwEmuMsg.data == null) {
42              sysLog(1, "ERROR:␣DATA␣of␣nwEmuMsg␣IS␣NULL!");
43              return false;
44          }
45          send();
46          return true;
47      }
48
49      /**
50       * Baut das zu sendende Paket auf. Als Pruefsumme wird einfach
51       * die Summe aller Daten und Metadaten genommen.
52       */
53      private void send() {
54          NWEmuPkt packet = new NWEmuPkt();
55          packet.checksum = 0;
56          for(int i = 0; i < NWEmu.PAYSIZE; i++) {
57              packet.payload[i] = lastMsg.data[i];
58              packet.checksum += packet.payload[i];
59          }
60          packet.seqnum = seqnum;
61          packet.checksum += packet.seqnum;
62
63          sysLog(0, "Sending␣Host:␣output(" + lastMsg + ")␣->␣toLayer3(" + packet + ")");
64          toLayer3(packet);
65          startTimer(TIMEOUT);
66      }
67
68      @Override
69      public void input(NWEmuPkt nwEmuPkt) {
70          if(nwEmuPkt.seqnum == seqnum) {
71              // Beim ABP gibt es nur zwei SQNern, deswegen wird hier in IZ_2 gerechnet.
72              seqnum = (seqnum + 1) % 2;
73              isSending = false;
74              sysLog(2, "Sending␣Host:␣Received␣ACK␣is␣fine.");
75          } else {
76              sysLog(2, "ERROR:␣seqnum␣wrong␣(" + nwEmuPkt.seqnum + ")");
77              send();
78          }
79          sysLog(0, "Sending␣Host:␣input(" + nwEmuPkt + ")");
80          stopTimer();
81      }
82
83      @Override
84      public void init() {
85          sysLog(0, "Sending␣Host:␣init()");
86          super.init();
87      }
88
89      @Override
90      public void timerInterrupt() {
91          send();
92      }
93  }
94
95  class ABPReceivingHost extends CommonHost {
96      /**
97       * Fall Emnpfaenger: Ueberpruefe auf einen Bitfehler und sende
98       *                   zugehoerige ACK.
99       */
100     @Override
101     public void input(NWEmuPkt nwEmuPkt) {
102         // Schritt 1: Ueberpruefe auf Bitfehler
103         int checksum = nwEmuPkt.checksum - seqnum;
104         for(byte b : nwEmuPkt.payload) {
105             checksum -= b;
106         }
107         // Falls kein Bitfehler aufgetreten ist, sollte checksum = 0
108         // gelten!
```

```
109         // Schritt 2: Baue Antwortpaket fuer ACK
110         NWEmuPkt pkt = new NWEmuPkt();
111         if(nwEmuPkt.seqnum == seqnum && checksum == 0) {
112             NWEmuMsg msg = new NWEmuMsg();
113             for(int i = 0; i < NWEmu.PAYSIZE; i++) {
114                 msg.data[i] = nwEmuPkt.payload[i];
115             }
116             sysLog(0, "Receiving␣Host:␣Sending␣message␣to␣Layer␣5.");
117             toLayer5(msg);
118 
119             pkt.seqnum = seqnum;
120             seqnum = (seqnum + 1) % 2;
121         } else {
122             // Hier ist was falsch gelaufen, entweder checksum != 0 oder
123             // falsche Reihenfolge der Pakete (andere SQN).
124             sysLog(0, "Receiving␣Host:␣Package␣either␣sequentially␣" +
125                     "incorrect␣or␣checksum␣(" + checksum + ")␣failure.");
126             pkt.seqnum = (seqnum + 1) % 2;
127         }
128         toLayer3(pkt);
129     }
130 
131     @Override
132     public void init() {
133         sysLog(0, "Receiving␣Host:␣init()");
134         super.init();
135     }
136 }
```

Quellkode B.2 — Alternating-Bit-Protokoll

B.3 Go-Back-N

```
1  import fau.cs7.nwemu.AbstractHost;
2  import fau.cs7.nwemu.NWEmu;
3  import fau.cs7.nwemu.NWEmuMsg;
4  import fau.cs7.nwemu.NWEmuPkt;
5  
6  
7  import java.nio.ByteBuffer;
8  import java.util.LinkedList;
9  import java.util.zip.CRC32;
10 
11 
12 
13 class CommonHostGBN extends AbstractHost {
14     int seqnum, acknum;
15 
16     static final int BUFFER_SIZE = 8;   // Wie gross der Packetpuffer ist
17     static final double TIMER = 1000.0; // Timeout
18 
19     @Override
20     public void init() {
21         seqnum = 0;
22         acknum = 0;
23     }
24 
25     /**
26      * Anstatt einer einfachen Summe wurde hier ein CRC verwendet. Glueklicherweise
27      * besitzt Java dafuer vorgefertigte Methoden ...
28      */
29     int calculateChecksum(int seqnum, int acknum, byte[] payload) {
30         CRC32 crc32 = new CRC32();
31         crc32.update(ByteBuffer.allocate(4).putInt(seqnum));
32         crc32.update(ByteBuffer.allocate(4).putInt(acknum));
33         crc32.update(payload);
34         return (int) crc32.getValue();
35     }
36 }
37 
38 
39 
```

```
40 |
41 | class Sender extends CommonHostGBN {
42 |     // Puffer fuer unbestaetigte Pakete
43 |     private LinkedList<NWEmuPkt> buffer;
44 |     // Nummer des zuletzt bestaetigten Pakets
45 |     private int s;
46 |
47 |     @Override
48 |     public void init() {
49 |         this.buffer = new LinkedList<>();
50 |         this.acknum = 0;
51 |         this.seqnum = 0;
52 |         this.s = 0;
53 |     }
54 |
55 |     @Override
56 |     public Boolean output(NWEmuMsg nwEmuMsg) {
57 |         // Falls der Puffer voll ist, sende kein neues Paket
58 |         if (seqnum + 1 > s + BUFFER_SIZE) {
59 |             sysLog(2, "WARNING: buffer of sender is full!");
60 |             return false;
61 |         }
62 |
63 |         // Falls der Puffer noch mindestens einen Platz frei hat,
64 |         // sende ein Paket. Dieses hat dann die SQN = sqnum.
65 |         NWEmuPkt pkt = new NWEmuPkt();
66 |         pkt.seqnum = seqnum;
67 |         pkt.acknum = 0;
68 |         System.arraycopy(nwEmuMsg.data, 0, pkt.payload, 0, NWEmu.PAYSIZE);
69 |         pkt.checksum = calculateChecksum(pkt.seqnum, pkt.acknum, pkt.payload);
70 |         // Fuege das neue Paket zum Puffer hinzu, es ist unbestaetigt und wird gesendet.
71 |         buffer.add(pkt);
72 |
73 |         sysLog(0, "Sending Host: output(" + nwEmuMsg + ") -> toLayer3(" + pkt + ")");
74 |         toLayer3(pkt);
75 |         // Beim Starten des ersten Pakets wird der Timer gestartet.
76 |         // Ersten in dem Sinne des ersten seit der letzten Bestaetigung.
77 |         if(s == seqnum) {
78 |             startTimer(TIMER);
79 |         }
80 |         seqnum++; // Aufgrund mehrerer Pakete ohne Bestaetigung wird hier SQN inkr.
81 |         return true;
82 |     }
83 |
84 |     @Override
85 |     public void input(NWEmuPkt p) {
86 |         // Wenn ein ACK ohne Bitfehler und mit SQN im Fenster zurueckkommt ...
87 |         if(p.checksum != calculateChecksum(p.seqnum, p.acknum, p.payload) ||
88 |                 (p.acknum < s) || (seqnum <= p.acknum)) {
89 |             // Wenn nicht dann ist was kaputt.
90 |             sysLog(3, "WARNING: checksum fail!");
91 |             return;
92 |         }
93 |         // ... schiebe das Fenster bis zu dieser SQN ...
94 |         s = p.acknum + 1;
95 |         // ... stoppe den Timer ...
96 |         stopTimer();
97 |         if(s < seqnum) {
98 |             // ... und wenn das Fenster nicht leer ist starte ihn neu.
99 |             startTimer(TIMER);
100|         }
101|     }
102|
103|     @Override
104|     public void timerInterrupt() {
105|         sysLog(1, "{s:" + s + ", length: " + buffer.size() + ", seqnum: " + seqnum + "}");
106|         // Wenn der Timeout ablaeuft sende alle unbestaetigten Pakete erneut
107|         // und starte den Timer erneut
108|         for(int i = s; i < seqnum; i++) {
109|             toLayer3(buffer.get(i));
110|         }
111|         startTimer(TIMER);
112|     }
113| }
114|
```

```
115 class Receiver extends CommonHostGBN {
116
117     @Override
118     public void init() {
119         super.init();
120         this.acknum = 0;
121         this.seqnum = 0;
122     }
123
124     @Override
125     public void input(NWEmuPkt p) {
126         // Wenn ein Paket ohne Bitfehler und mit aktueller SQN ankommt ...
127         if(p.checksum != calculateChecksum(p.seqnum, p.acknum, p.payload) ||
128                 p.seqnum != seqnum) {
129             // Wenn nicht dann ist was kaputt.
130             sysLog(3, "WARNING: checksum or seqnum failure");
131             return;
132         }
133
134         // ... gib Paket nach oben weiter ...
135         NWEmuMsg msg = new NWEmuMsg();
136         System.arraycopy(nwEmuPkt.payload, 0, msg.data, 0, NWEmu.PAYSIZE);
137         sysLog(0, "Receiving Host: Sending message to Layer 5.");
138         toLayer5(msg);
139
140         // ... sende ACK mit aktueller SQN
141         NWEmuPkt ackResp = new NWEmuPkt();
142         ackResp.seqnum = 0;
143         ackResp.acknum = seqnum;
144         ackResp.flags = 0;
145         ackResp.checksum = calculateChecksum(0, seqnum, ackResp.payload);
146         toLayer3(ackResp);
147
148         // ... und inkr. SQN.
149         seqnum++;
150     }
151
152 }
```

Quellkode B.3 — Go-Back-N

ANHANG C

Lösungen der Präsenzübungen

C.1 Übung 1

Aufgabe 1.1

Wie lange dauert es, eine Datei der Größe 640 000 Bits von Rechner A zu Rechner B über ein leitungsvermitteltes Netz zu übertragen?
Alle Links haben eine Bitrate R von 1,536 Mbps und nutzen das TDMA-Verfahren mit 24 Slots pro Sekunde. Der Verbindungsaufbau von einem Ende zum anderen dauert $d_{con} = 500\text{ms}$.

Lösungsvorschlag

Beim TDMA-Verfahren wird die verfügbare Sendezeit gleichmäßig auf alle Nutzer aufgeteilt, damit teilt sich auch die Bitrate gleichmäßig auf.

$$
\begin{aligned}
O &= 640\,000\text{ Bits} = 640\text{ Kb} = 0{,}64\text{ Mb} \\
R_{gesamt} &= 1{,}536\ \frac{\text{Mb}}{\text{s}} \\
R_{Nutzer} &= \frac{R_{gesamt}}{24} = \frac{1{,}536\ \frac{\text{Mb}}{\text{s}}}{24} = 0{,}064\ \frac{\text{Mb}}{\text{s}} \\
t_{Übertragung} &= \frac{O}{R_{Nutzer}} = \frac{0{,}64\text{ Mb}}{0{,}064\ \frac{\text{Mb}}{\text{s}}} = 10\text{s} \\
d_{gesamt} &= d_{con} + d_{Übertragung} = 0{,}5\text{ s} + 10\text{ s} = 10{,}5\text{ s}
\end{aligned}
$$

Aufgabe 1.2

Wie oben, nur FDMA statt TDMA.

Lösungsvorschlag

Die verfügbare Bandbreite wird gleichmäßig aufgeteilt, damit auch die verfügbare Bitrate pro Nutzer. Analog zu oben.

Aufgabe 1.3

Es besteht eine Verbindung Verbindung mit 1 $\frac{\text{Mb}}{\text{s}}$ sowie 35 Nutzer mit jeweils 100 $\frac{\text{Kb}}{\text{s}}$ Bedarf, wobei jeder Nutzer nur zu 10% der Zeit aktiv ist.

a) Wie viele Nutzer sind bei Leitungsvermittlung möglich?

b) Mit welcher Wahrscheinlichkeit sendet ein gegebener Nutzer?

c) Mit welcher Wahrscheinlichkeit senden zu einem gegebenen Zeitpunkt genau n Nutzer?

d) Mit welcher Wahrscheinlichkeit senden mehr als 10 Nutzer gleichzeitig?

Lösungsvorschlag

a) Bei Leitungsvermittlung wird jedem Nutzer dauerhaft eine Leitung zugesichert, unabhängig davon ob er sie gerade ausreizt oder überhaupt nutzt.

$$\begin{aligned} R_{\text{gesamt}} &= 1\ \frac{\text{Mb}}{\text{s}} \\ R_{\text{Nutzer}} &= 100\ \frac{\text{Kb}}{\text{s}} = 0,1\ \frac{\text{Mb}}{\text{s}} \\ N_{\text{Nutzer}} &= \frac{R_{\text{gesamt}}}{R_{\text{Nutzer}}} = \frac{1\ \frac{\text{Mb}}{\text{s}}}{0,1\ \frac{\text{Mb}}{\text{s}}} = 10 \end{aligned}$$

b) $p_{\text{nutzer}} = 10\% = 0,1$

c) $p_{n\ \text{nutzer}} = \binom{35}{n} \cdot p_{\text{Nutzer}}^{n} \cdot (1 - p_{\text{Nutzer}})^{35-n} = \binom{35}{n} \cdot 0,1^{n} \cdot 0,9^{35-n}$

d) $p_{\text{mehr als 10 Nutzer}} = 1 - \sum_{i=0}^{10} p_{i\ \text{Nutzer}} \approx 1 - 0,9996 = 0,0004$

Aufgabe 1.4

Moderne Containerschiffe können im Mittel 14 000 TEU bei einer Maximalgeschwindigkeit von 60 km/h transportieren. Ein TEU hat die Größe von 2,5m · 2,5m · 6m.
Die Hochseeleitung Apollo verläuft zwischen USA und Großbritannien. Sie erstreckt sich über eine Länge von 12315km und besitzt eine Datenrate von 3,2 Tb/s.
Nehmen Sie an, dass das Schiff mit 2TB Festplatten der Größe 0,1m · 0,2m · 0,05m voll beladen wird.
Welches Medium besitzt die höhere Datenrate, wenn das Schiff genau die gleiche Strecke wie das Seekabel zurück legt?

Lösungsvorschlag

$$
\begin{aligned}
R_{\text{Apollo}} &= 3{,}2\ \frac{\text{Tb}}{\text{s}} \\
v_{\text{Schiff}} &= 60\frac{\text{km}}{\text{h}} \\
s_{\text{Strecke}} &= 12\,315\ \text{km} \\
d_{\text{Schiff}} &= \frac{s_{\text{Strecke}}}{v_{\text{Schiff}}} = \frac{12\,315\ \text{km}}{60\ \frac{\text{km}}{\text{h}}} = 205{,}25\ \text{h} = 738\,900\ \text{s} \\
V_{\text{Festplatte}} &= 0{,}1\ \text{m} \cdot 0{,}2\ \text{m} \cdot 0{,}05\ \text{m} = 0{,}001\ \text{m}^3 = 1 \cdot 10^{-3}\ \text{m}^3 \\
V_{\text{Schiff}} &= 14\,000 \cdot 2{,}5\ \text{m} \cdot 2{,}5\ \text{m} \cdot 6\ \text{m} = 525000\ \text{m}^3 = 525 \cdot 10^3 \text{m}^3 \\
N_{\text{Festplatten}} &= \frac{V_{\text{Schiff}}}{V_{\text{Festplatte}}} = \frac{525 \cdot 10^3\ \text{m}^3}{1 \cdot 10^{-3}\ \text{m}^3} = 525 \cdot 10^6 \\
O_{\text{Festplatte}} &= 2\ \text{TB} = 2 \cdot 8\ \text{Tb} = 16\ \text{Tb} \\
O_{\text{Schiff}} &= N_{\text{Festplatten}} \cdot O_{\text{Festplatte}} = 525 \cdot 10^6 \cdot 16\ \text{Tb} = 84 \cdot 10^8\ \text{Tb} \\
R_{\text{Schiff}} &= \frac{O_{\text{Schiff}}}{d_{\text{Schiff}}} = \frac{84 \cdot 10^8\ \text{Tb}}{738\,900\ \text{s}} \approx 11\,368{,}3\ \frac{\text{Tb}}{\text{s}}
\end{aligned}
$$

Die Datenrate des Schiffes ist wesentlich höher als die der Unterseeleitung.

Aufgabe 1.5

Nehmen Sie an: Sie wohnen in Rosenbach und besitzen eine 1Mbit/s DSL Internetverbindung. Sie benötigen für die Wegstrecke Universität-Rosenbach 20 Minuten. In der Universität steht Ihnen eine 100Mbit/s Internetverbindung zur Verfügung.
Ab welcher Dateigröße lohnt sich die Fahrt in die Universität?

Lösungsvorschlag

$$
\begin{aligned}
R_{\text{Zuhause}} &= 1\ \frac{\text{Mb}}{\text{s}} \\
R_{\text{Universität}} &= 100\ \frac{\text{Mb}}{\text{s}} \\
d_{\text{Download}} &= \frac{O}{R} \\
d_{\text{Fahrt}} &= 2 \cdot 20\ \text{min} = 40\ \text{min} = 2400\ \text{s} \\
d_{\text{Zuhause}} &\overset{!}{\geqslant} d_{\text{Universität}} \\
\frac{O}{1\ \frac{\text{Mb}}{\text{s}}} &\geqslant \frac{O}{100\ \frac{\text{Mb}}{\text{s}}} + 2400\ \text{s} \\
\frac{O}{1\ \frac{\text{Mb}}{\text{s}}} - \frac{O}{100\ \frac{\text{Mb}}{\text{s}}} &\geqslant 2400\ \text{s} \\
\frac{100 \cdot O - O}{100\ \frac{\text{Mb}}{\text{s}}} &\geqslant 2400\ \text{s} \\
\frac{99 \cdot O}{100\ \frac{\text{Mb}}{\text{s}}} &\geqslant 2400\ \text{s} \\
99 \cdot O &\geqslant 240\,000\ \text{Mb} \\
O &\geqslant 2424\ \text{Mb} \\
O &\geqslant 303\ \text{MB}
\end{aligned}
$$

C.2 Übung 2

Aufgabe 2.1

Führen Sie einen `traceroute`-Aufruf zu einem beliebigen Server durch.

Lösungsvorschlag

```
traceroute heise.de
```

Aufgabe 2.2

a) Betrachten Sie eine Anwendung, die mit einer konstanten Rate Daten versendet, d.h. der Sender überträgt alle k Zeit- einheiten N Bits an Daten. Dabei soll k klein und konstant sein. Zudem soll die Anwendung nach dem Start relativ lange aktiv sein.
Ist für diese Anwendung ein paketvermitteltes oder ein leitungsvermitteltes Netz geeigneter? Warum?

b) Nehmen Sie an, es wird ein paketvermitteltes Netz verwendet, der gesamte Verkehr in diesem Netz stammt von Anwendungen mit obiger Charakteristik und die Summe der Datenraten der Anwendungen ist kleiner als die Kapazität eines jeden Links im Netz.
Ist in diesem Fall eine Überlastkontrolle nötig? Warum?

Lösungsvorschlag

a) Da näherungsweise konstant die gleiche Menge an Datenverkehr herrscht, ist der Vorteil des „statistischen Multiplexen“ der Paketvermittlung eher gering. Damit bietet sich Leitungsvermittlung an, denn es entfällt viel Overhead der Paketvermittlung.

b) Nein, denn auch im schlimmsten Fall (alle Teilnehmer senden gleichzeitig) wird die Maximalkapazität des Netzes nicht vollständig ausgeschöpft.

Aufgabe 2.3

In dieser Aufgabe soll das Senden von Sprache von Rechner A an Rechner B über ein paketvermitteltes Netz betrachtet werden (wie bei Internet-Telefonie). Die analogen Sprachdaten werden von Host A zur Echtzeit in einen digitalen Datenstrom mit 64 kbps konvertiert und dann in Pakete der Größe 48 Bytes gruppiert.
Zwischen Host A und B befindet sich ein Link mit einer Bitrate 1 Mbps und mit einer Ausbreitungsverzögerung von 2 ms.
Sobald A ein Paket zusammengestellt hat, wird es an B gesendet. Wenn B das Paket vollständig empfangen hat, werden die enthaltenen Digitaldaten in ein Analogsignal konvertiert.
Wie lange dauert es von der Erzeugung eines Bits aus dem Analogsignal bei A bis zur Rückverwandlung in ein Analogsignal bei B?

Lösungsvorschlag

$$
\begin{aligned}
d_{\text{gesamt}} &= d_{\text{proc}} + d_{\text{queue}} + d_{\text{trans}} + d_{\text{prop}} \\
d_{\text{queue}} &= 0 \\
d_{\text{proc}} &= \frac{48\ \text{B}}{64\ \frac{\text{Kb}}{\text{s}}} = \frac{384\ \text{b}}{64\ \frac{\text{Kb}}{\text{s}}} = 0{,}006\ \text{s} \\
d_{\text{trans}} &= \frac{L}{R} = \frac{48\ \text{B}}{1\ \frac{\text{Mb}}{\text{s}}} = \frac{384\ \text{b}}{1\ \frac{\text{Mb}}{\text{s}}} = 0{,}000384\ \text{s} \\
d_{\text{prop}} &= 2\ \text{ms} = 0{,}002\ \text{s} \\
d_{\text{gesamt}} &= 0{,}006\ \text{s} + 0{,}000384\ \text{s} + 0{,}002\ \text{s} \approxeq 0{,}008384\ \text{s}
\end{aligned}
$$

Aufgabe 2.4

Betrachten Sie die Warteschlangenverzögerung in einem Puffer eines Routers (vor einem ausgehenden Link). Nehmen Sie an, alle Pakete haben eine Länge von L Bits, die Übertragungsrate ist R bps und alle NL/R Sekunden kommen wieder N Pakete zum Versenden dazu.
Bestimmen Sie die mittlere Warteschlangenverzögerung eines Pakets.

Lösungsvorschlag

Das erste Paket kann direkt versendet werden, das zweite muss $\frac{L}{R}$ Sekunden warten, das dritte $2 \cdot \frac{L}{R}$ Sekunden warten, etc. Damit ergibt sich eine Formel für die Wartezeit d_i des i. Pakets:

$$d_i = (i-1) \cdot \frac{L}{R}$$

Wir bilden nur den Durchschnitt d_{Mittel} als das arithmetische Mittel:

$$d_{Mittel} = \frac{\sum_{i=1}^{N}(i-1) \cdot \frac{L}{R}}{N} = \frac{(\sum_{i=1}^{N} i) \cdot \frac{L}{R} - N \cdot \frac{L}{R}}{N} = \frac{\left(\frac{N \cdot (N+1)}{2}\right) \cdot \frac{L}{R} - N \cdot \frac{L}{R}}{N} = \left(\frac{N+1}{2} - 1\right) \cdot \frac{L}{R} = \frac{N-1}{2} \cdot \frac{L}{R}$$

Aufgabe 2.5

Betrachten Sie die Warteschlangenverzögerung in einem Puffer eines Routers.
Gegeben seien die Verkehrsintensität

$$\rho = \frac{L \cdot \lambda}{R}$$

und die Warteschlangenverzögerung

$$d_{queue} = \frac{L \cdot \rho}{R \cdot (1-\rho)}$$

für $\rho < 1$.

a) Geben Sie eine Formel für die Gesamtverzögerung an, d.h. Warteschlangen- plus Übertragungsverzögerung.

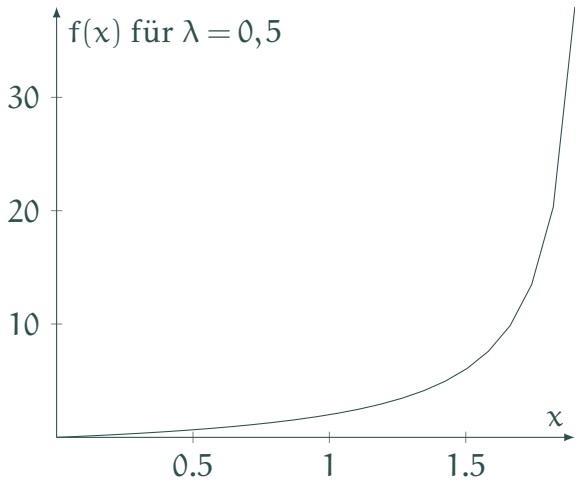
b) Zeichnen Sie die Gesamtverzögerung als Funktion von L/R.

Lösungsvorschlag

a)

$$\begin{aligned} d_{gesamt} &= d_{trans} + d_{queue} \\ d_{trans} &= \frac{L}{R} \\ d_{gesamt} &= \frac{L}{R} + \frac{\rho \cdot L}{R \cdot (1-\rho)} = \frac{L}{R} \cdot \left(1 + \frac{\rho}{1-\rho}\right) = \frac{L}{R} \cdot \left(\frac{1-\rho}{1-\rho} + \frac{\rho}{1-\rho}\right) = \frac{L}{R} \cdot \frac{1-\rho+\rho}{1-\rho} \\ &= \frac{L}{R} \cdot \frac{1}{1-\rho} = \frac{L}{R} \cdot \frac{1}{1-\frac{\lambda L}{R}} = \frac{\frac{L}{R}}{1-\lambda \cdot \frac{L}{R}} \end{aligned}$$

b) Es gelte $x := \frac{L}{R}$, dann gibt $f(x) := \frac{x}{1-\lambda \cdot x}$ die Gesamtverzögerung als Funktion an.



Aufgabe 2.6

a) Verallgemeinern Sie die Formel für die Ende-zu-Ende-Verzögerung einer homogenen Verbindung $d_{\text{end-end}} = E \cdot (d_{\text{proc}} + d_{\text{trans}} + d_{\text{prop}}) + d_{\text{proc}}$ für heterogene Verarbeitungsraten, Übertragungsraten und Ausbreitungsverzögerungen. E bezeichnet die Anzahl der Links zwischen Start- und Zielknoten.

b) Wiederholen Sie die die Verallgemeinerung, aber nehmen Sie nun auch an, dass bei jedem Knoten zusätzlich eine Warteschlangenverzögerung von d_{queue} auftritt.

Lösungsvorschlag

a)

$$\sum_{i=1}^{E} \left(d_{\text{proc},i} + d_{\text{trans},i} + d_{\text{prop},i}\right) + d_{\text{proc}}$$

b)

$$\sum_{i=1}^{E} \left(d_{\text{proc},i} + d_{\text{trans},i} + d_{\text{prop},i} + d_{\text{queue},i}\right) + d_{\text{proc}}$$

C.3 Übung 3

Aufgabe 3.1

a) Füllen Sie die Fingertabelle von Knoten 8 aus (mit $m = 4$).

b) Es wird nun ein neuer Peer $p = 9$ eingefügt. Vervollständigen Sie dessen Fingertabelle und die seiner direkten Nachbarn.

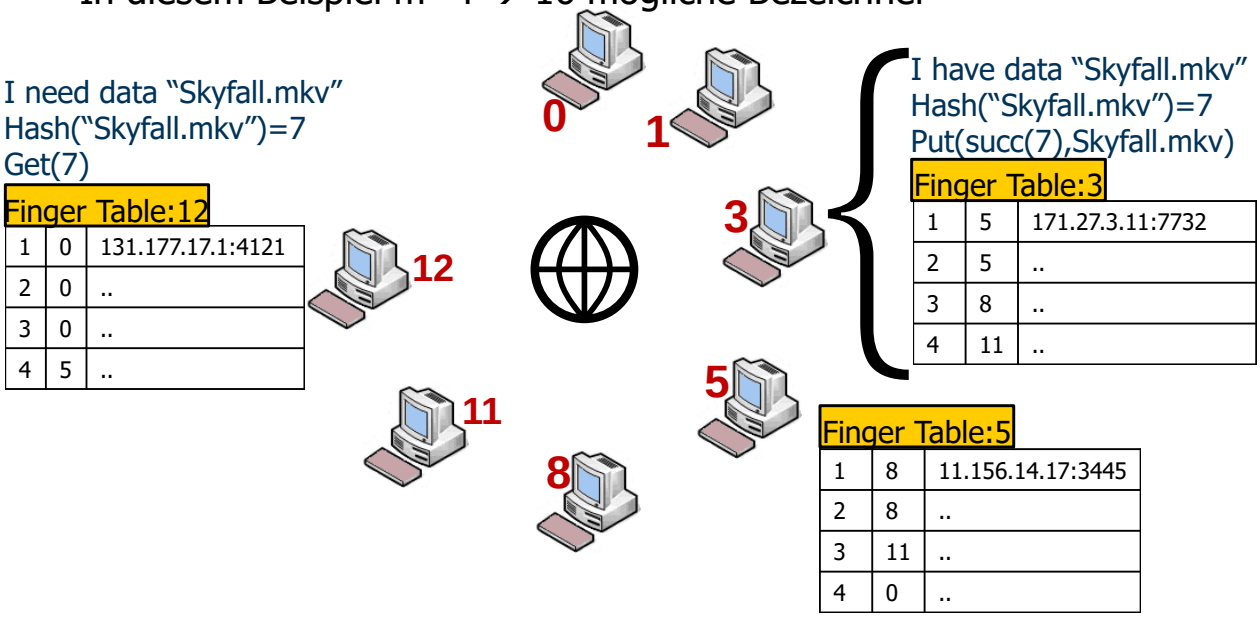
c) Welchen Weg wird die Abfrage nach „Skyfall.mkv“ folgen?

Lösungsvorschlag

a) Fingertabelle von Peer $p = 8$:

Structured P2P: Distributed Hash Tables (DHT)

- Siehe Anwendungsschicht, Folien 87ff.
- Dateien bzw. Referenzen auf Knoten (Peers) abbilden
- Peers haben nur Teilsicht über das ganze System
- In diesem Beispiel m=4 → 16 mögliche Bezeichner



i	p
1	$succ(8+2^{1-1}) = succ(8+1) = 11$
2	$succ(8+2^{2-1}) = succ(8+2) = 11$
3	$succ(8+2^{3-1}) = succ(8+4) = 12$
4	$succ(8+2^{4-1}) = succ(8+8) = 0$

b) Fingertabelle von Peer $p = 9$:

i	p
1	$succ(9+2^{1-1}) = succ(9+1) = 11$
2	$succ(9+2^{2-1}) = succ(9+2) = 11$
3	$succ(9+2^{3-1}) = succ(9+4) = 0$
4	$succ(9+2^{4-1}) = succ(9+8) = 1$

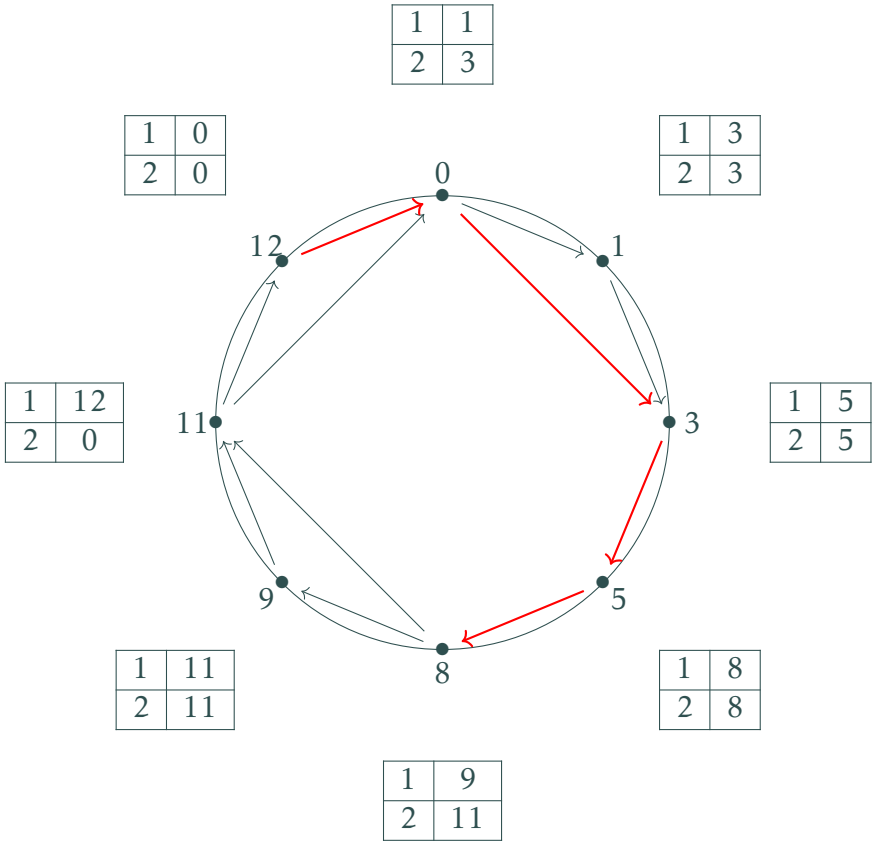
Fingertabelle von Peer $p = 8$:

i	p
1	$succ(8+2^{1-1}) = succ(8+1) = 9$
2	$succ(8+2^{2-1}) = succ(8+2) = 11$
3	$succ(8+2^{3-1}) = succ(8+4) = 12$
4	$succ(8+2^{4-1}) = succ(8+8) = 0$

Fingertabelle von Peer $p = 11$:

i	p
1	$succ(11+2^{1-1}) = succ(11+1) = 12$
2	$succ(11+2^{2-1}) = succ(11+2) = 0$
3	$succ(11+2^{3-1}) = succ(11+4) = 0$
4	$succ(11+2^{4-1}) = succ(11+8) = 3$

c) Es werden nun nur $\frac{m}{2} = \frac{4}{2} = 2$ Einträge in der Fingertabelle pro Peer gespeichert.



Pfad der Abfrage von „Skyfall.mkv“ mit Hash $k = 7$ von Knoten $p = 12$:
$12 \Rightarrow 0 \Rightarrow 3 \Rightarrow 5 \Rightarrow 8$

C.4 Übung 4

Aufgabe 4.1

Betrachten Sie eine Halbduplex-Punkt-zu-Punkt-Verbindung, für die das Stop-and-Wait-Verfahren eingesetzt wird.

a) Was geschieht mit der Leitungsauslastung (= normierter Durchsatz S), wenn die Größe der Nachrichten (Objekte) erhöht wird? Die anderen Parameter, inkl. der Paketgröße, sollen nicht verändert werden.

b) Welche Auswirkung auf die Auslastung der Leitung kann man beobachten, wenn die Anzahl der Pakete bei konstanter Nachrichtengröße erhöht wird?

c) Welche Auswirkung auf die Auslastung der Leitung hat eine Vergrößerung der Pakete?

Lösungsvorschlag

a) Keine Veränderung (L, R, D konstant $\rightarrow S$ konstant)

b) Durchsatz kleiner, bessere Auslastung (L kleiner $\rightarrow a$ größer $\rightarrow S$ kleiner)

c) Bessere Auslastung, denn dann wird $a = \frac{R \cdot D}{L}$ kleiner, damit $S = \frac{1}{1+2a}$ größer.

Aufgabe 4.2

Ein Kanal hat eine Datenrate von $4{,}0\ \frac{\text{Kb}}{\text{s}}$ und eine Ausbreitungsverzögerung $D = 20$ ms. Für welchen Bereich von Paketgrößen hat das Stop-And-Wait-Verfahren eine Effizienz von mindestens 50%?

Lösungsvorschlag

$$\begin{aligned} S = \frac{1}{1+2a} = \frac{1}{1+2 \cdot \frac{R \cdot D}{L}} &\geqslant 0{,}5 \\ \frac{1}{1+2 \cdot \frac{4\ \frac{\text{Kb}}{\text{s}} \cdot 0{,}02\ \text{s}}{L}} &\geqslant \frac{1}{2} \\ 1 + \frac{8\ \frac{\text{Kb}}{\text{s}} \cdot 0{,}02\ \text{s}}{L} &\leqslant 2 \\ \frac{160\ \text{Bit}}{L} &\leqslant 1 \\ 160\ \text{Bit} &\geqslant L \end{aligned}$$

Eine obere Grenze kann nicht angegeben werden (?)

Aufgabe 4.3

Warum werden beim Alternating-Bit-Protokoll keine NAK0- und NAK1-Nachrichten benötigt?

Lösungsvorschlag

Bei fehlerhaftem Empfang wird das letzte ACK zurückgesendet, damit weiß der Sender, dass das letzte Paket fehlerhaft übertragen wurde. Dieses „doppelte“ ACK übernimmt hier implizit die Aufgabe der NAK-Nachrichten.

Aufgabe 4.4

Auf einer Satellitenverbindung mit $R = 1\ \frac{\text{Mb}}{\text{s}}$ und $D = 0{,}27$ s sollen Pakete der Größe $L = 1000$ Bits eingesetzt werden.
Wie hoch ist die maximale Auslastung der Verbindung bei

a) Stop-And-Wait-Fehlerkontrolle?

b) Schiebefenster-Fehlerkontrolle mit einer Fenstergröße von 7 Paketen?

c) Schiebefenster-Fehlerkontrolle mit einer Fenstergröße von 127 Paketen?

d) Schiebefenster-Fehlerkontrolle mit einer Fenstergröße von 255 Paketen?

Lösungsvorschlag

$$a = \frac{R \cdot D}{L} = \frac{1\ \frac{\text{Mb}}{\text{s}} \cdot 0{,}27\ \text{s}}{1\ \text{Kb}} = \frac{270\ \text{Kb}}{1\ \text{Kb}} = 270$$

a)

$$\begin{aligned} S &= \frac{1}{1+2\cdot a} = \frac{1}{1+2\cdot 270} \\ &= \frac{1}{1+540} \\ &= \frac{1}{541} \approx 0,18\% \end{aligned}$$

b)

$$\begin{aligned} W &= 7 \\ 1+2\cdot a &= 1+2\cdot 270 = 541 \\ W &< 1+2\cdot a \\ S &= \frac{W}{1+2\cdot a} = \frac{7}{541} \approx 1,3\% \end{aligned}$$

c)

$$\begin{aligned} W &= 127 \\ W &< 541 \\ S &= \frac{127}{541} \approx 23\% \end{aligned}$$

d)

$$\begin{aligned} W &= 255 \\ W &< 541 \\ S &= \frac{255}{541} \approx 47\% \end{aligned}$$

Aufgabe 4.5

Die Abbildung unten stellt drei Hosts dar. Zwischen A und B liegen 2000 km, zwischen B und C 500 km. Pakete werden bei Host A erzeugt und teüber B nach C versendet. Bestimmen Sie die minimale Übertragungsrate zwischen B und C, bei der die Puffer bei Knoten B nicht überlaufen, wenn folgende Voraussetzungen gelten:

- Die Datenrate zwischen A und B beträgt 100 kbps.
- Die spezifische Ausbreitungsverzögerung beträgt 10 µs/km bei beiden Verbindungen.
- Die Leitungen unterstützen Vollduplex-Betrieb.
- Alle Datenpakete sind 1000 Bits groß. ACK-Pakete haben eine vernachlässigbare Größe.
- Zwischen Host A und B wird ein Schiebefensterprotokoll mit einer Fenstergröße von 3 Paketen verwendet.
- Zwischen Host B und C wird Stop-And-Wait verwendet.
- Es treten keine Fehler auf.

Lösungsvorschlag

$$
\begin{aligned}
D_{AB} &= 2000\ \text{km} \cdot 10\frac{\mu s}{\text{km}} = 0{,}02\ \text{s} \\
D_{BC} &= 500\ \text{km} \cdot 10\frac{\mu s}{\text{km}} = 0{,}005\ \text{s} \\
R_{in} &= \frac{W \cdot L}{\frac{L}{R_{AB}} + 2 \cdot D_{AB}} \\
&= \frac{3 \cdot 1\ \text{Kb}}{\frac{1\ \text{Kb}}{100\ \frac{\text{Kb}}{\text{s}}} + 2 \cdot 0{,}02\ \text{s}} \\
&= \frac{3\ \text{Kb}}{0{,}01\ \text{s} + 0{,}04\ \text{s}} \\
&= \frac{3}{0{,}05}\frac{\text{Kb}}{\text{s}} \\
&= 60\ \frac{\text{Kb}}{\text{s}} \\
R_{in} = R_{out} &= \frac{L}{\frac{L}{R_{BC}} + 2 \cdot D_{BC}} \\
&= \frac{1\ \text{Kb}}{\frac{1\ \text{Kb}}{R_{BC}} + 2 \cdot 0{,}005\ \text{s}} \\
60\ \frac{\text{Kb}}{\text{s}} &= \frac{1\ \text{Kb}}{\frac{1\ \text{Kb}}{R_{BC}} + 0{,}01\ \text{s}} \\
60\ \frac{\text{Kb}}{\text{s}} \cdot \frac{1\ \text{Kb}}{R_{BC}} + 0{,}6\ \text{Kb} &= 1\ \text{Kb} \\
60\ \frac{\text{Kb}}{\text{s}} \cdot \frac{1\ \text{Kb}}{R_{BC}} &= 0{,}4\ \text{Kb} \\
60\ \frac{\text{Kb}}{\text{s}} \cdot \frac{1}{0{,}4} &= R_{BC} \\
150\ \frac{\text{Kb}}{\text{s}} &= R_{BC}
\end{aligned}
$$

Aufgabe 4.6

Nehmen Sie an, dass das Selective-Repeat-Schema mit $W = 4$ zur Übertragung benutzt wird. Veranschaulichen Sie anhand eines Beispiels, dass Sequenznummern mit 3 Bits genügen.

Lösungsvorschlag

Es muss gelten $W > \frac{m+1}{2}$. Da $W = 4$ und $m = 3$ gilt, folgt $4 > \frac{3+1}{2} = 2$, also reichen 3 Bit für den Sequenznummernraum.

Aufgabe 4.7

Knoten tauschen Pakete von fester Größe L Bits auf einem Kanal mit Datenrate von $R\ \frac{b}{s}$, Ausbreitungsgeschwindigkeit v und einer Länge l aus.
Bestimmen Sie eine Formel für die minimale Größe des Sequenznummernfeldes (Anzahl der benötigten Bits) in Abhängigkeit von R , l, v und L , bei der die maximale Auslastung der Verbindung berücksichtigt wird.
Dazu können Sie annehmen, dass ACK-Pakete eine vernachlässigbare Größe besitzen und die Verarbeitung in den Knoten unmittelbar geschieht (d.h. keine Zeit benötigt).

Lösungsvorschlag

$$n = \log_2(SQN) S = 1 \rightarrow \frac{W}{1+2a} \geqslant 1$$
$$W \geqslant 1+2a = 1+2 \cdot \frac{DL}{R} = 1+2 \cdot \frac{\frac{l}{v} \cdot R}{L}$$
$$SQN \geqslant 2W = 2+4 \cdot \frac{\frac{l}{v} \cdot R}{L} n = \log_2(2+4 \cdot \frac{\frac{l}{v} \cdot R}{L})$$

Aufgabe 4.8

Ein Webserver empfängt üblicherweise relative kleine Nachrichten (Requests) von den Clients, überträgt aber möglicherweise sehr große Objekte als Antwort an die Clients.
Welches Schiebefensterprotokoll, d.h. Selective Repeat oder Go-back-N, würde besonders populäre Webserver am wenigsten belasten? Warum?

Lösungsvorschlag

Selective Repeat, da mit diesem Protokoll nur einzelne fehlerhafte und nicht empfangene Pakete nachgeschickt werden und es somit keine "kumulativen Nachsendungen"gibt. Die Nachrichten von den Clients (also ACKs) sind im Vergleich dazu relativ klein, somit ist der Vorteil von kumulativen im Vergleich zu selektiven ACKs sehr gering. (Begruendung ist sehr vage)

Aufgabe 4.9

Vergleichen Sie die Leitungsauslastung (= normierter Durchsatz S) als Funktion der Fehlerwahrscheinlichkeit p für folgende Fehlerkontrollmechanismen:

- Stop-and-Wait
- Go-back-N mit $W = 7$
- Go-back-N mit $W = 127$
- Selective-Repeat mit $W = 7$
- Selective-Repeat mit $W = 127$

Beachten Sie für die Kanalpuffergröße a folgende Werte:

- 0.1
- 1
- 10

Welche Technik ist für die jeweiligen Werte von a die beste?

Lösungsvorschlag

Es sind bekannt:

$$S_{\text{Stop-And-Wait}}(p, a) = \frac{1-p}{1+2a}$$
$$S_{\text{Go-Back-N},W}(p, a) = \begin{cases} \frac{1-p}{1+2 \cdot a \cdot p} & W \geqslant 1+2a \\ \frac{W \cdot (1-p)}{(1-p+W \cdot p) \cdot (1+2a)} & W < 1+2a \end{cases}$$
$$S_{\text{Selective-Repeat},W}(p, a) = \begin{cases} 1-p & W \geqslant 1+2a \\ \frac{W \cdot (1-p)}{1+2a} & W < 1+2a \end{cases}$$

	$a = 0,1$	$a = 1$	$a = 10$
$S_{\text{Stop-And-Wait}}$	$\frac{1-p}{1,2}$	$\frac{1-p}{3}$	$\frac{1-p}{21}$
$S_{\text{Go-Back-N,7}}$	$\frac{1-p}{1+0,2\cdot p}$	$\frac{1-p}{1+2\cdot p}$	$\frac{7\cdot(1-p)}{(1+6\cdot p)\cdot 3}$
$S_{\text{Go-Back-N,127}}$	$\frac{1-p}{1+0,2\cdot p}$	$\frac{1-p}{1+2\cdot p}$	$\frac{1-p}{1+20\cdot p}$
$S_{\text{Selective-Repeat,7}}$	$1-p$	$1-p$	$\frac{1-p}{3}$
$S_{\text{Selective-Repeat,127}}$	$1-p$	$1-p$	$1-p$

Für $a = 0,1$ funktioniert auch Stop-And-Wait mit zufriedenstellender Effizienz, die anderen beiden Verfahren jeweils etwas besser. Bei $a = 1$ zeigt Go-Back-N bereits deutliche Vorteile, Selective-Repeat ist noch einmal besser. Für sehr große $a = 10$ eignet sich Selective-Repeat mit großer Fenstergröße W am Besten.

C.5 Übung 5

Aufgabe 5.1

Nehmen Sie an, TCP würde sein Überlastfenster (Congestion Window) für jedes empfangene ACK-Segment um zwei statt um eins zu vergrößern. Dann bestünde das erste Fenster aus einem Segment, das zweite aus drei Segmenten, das dritte aus neun Segmenten, etc.

a) Bestimmen Sie K (Anzahl Fenster bei tatsächlicher Objektgröße) als Ausdruck in O und L.

b) Bestimmen Sie Q (Wartezeiten unendlich großes O) als Ausdruck in RTT, L und R.

c) Bestimmen Sie die Gesamtverzögerung unter Verwendung von O, R, L, RTT und $P = \min\{Q; K-1\}$.

Lösungsvorschlag

a)

$$\begin{aligned} K &= \min\{k | \sum_{i=1}^{k} 3^{i-1} \cdot L \geqslant O\} \\ &= \min\{k | \sum_{i=1}^{k} 3^{i-1} \geqslant \frac{O}{L}\} \\ &= \min\{k | \frac{1}{2} \cdot (3^k - 1) \geqslant \frac{O}{L}\} \\ &= \min\{k | 3^k \geqslant \frac{2O}{L} + 1\} \\ &= \lceil \log_3 \left(\frac{2O}{L} + 1 \right) \rceil \end{aligned}$$

b)

$$\begin{aligned} Q &= \max\{k|\frac{L}{R} + RTT - 3^{k-1}\frac{L}{R} \geqslant 0\} \\ &= \max\{k|3^{k-1}\frac{L}{R} \leqslant \frac{L}{R} + RTT\} \\ &= \max\{k|3^{k-1} \leqslant 1 + \frac{RTT}{\frac{L}{R}}\} \\ &= \max\{k|k-1 \leqslant \log_3\left(1 + \frac{RTT}{\frac{L}{R}}\right)\} \\ &= \lfloor \log_3\left(1 + \frac{RTT}{\frac{L}{R}}\right)\rfloor + 1 \end{aligned}$$

c)

$$\begin{aligned} P &= \min\{Q, K-1\} \\ &= \min\left\{\lfloor \log_3\left(1 + \frac{RTT}{\frac{L}{R}}\right)\rfloor + 1, \lceil \log_3\left(\frac{2O}{L} + 1\right)\rceil - 1\right\} \\ D_{gesamt} &= 2\,RTT + \frac{O}{R} + \sum_{k=1}^{P}\left(\frac{L}{R} + RTT + 3^{k-1}\frac{L}{R}\right) \\ &= 2\,RTT + \frac{O}{R} + P\cdot\left(RTT + \frac{L}{R}\right) - \frac{1}{2}\cdot\left(3^P - 1\right)\frac{L}{R} \end{aligned}$$

Aufgabe 5.2

Wiederholen Sie die Analyse für T Links zwischen Client und Server. Hier tritt nun ein Store-and-Forward-Delay auf.
Nehmen Sie an, dass dieses Delay für ACK-Segmente und Segmente zum Verbindungsaufbau vernachlässigbar klein ist und nur für Segmente berücksichtigt werden muss, die Daten enthalten und im Netzwerk keine Überlast auftritt und die Pakete daher keine Warteschlangenverzögerungen erfahren.
Tipp: Die Zeit vom Absenden des ersten Bits des ersten Segmentes beim Server bis zum Empfang des zugehörigen ACKs beträgt $T \cdot \frac{L}{R} + RTT$.

Lösungsvorschlag

K: unabhängige Wartezeiten/Links $\Rightarrow$ K bleibt gleich

$$\begin{aligned} Q_T &= \max\left\{k|T\cdot\frac{L}{R} + RTT - 2^{k-1}\frac{L}{R} \geqslant 0\right\} = \cdots = \lfloor \log_2\left(T + \frac{RTT}{\frac{L}{R}}\right)\rfloor + 1 \\ P_T &= \min\{Q_T, K-1\} \\ D_{gesamt} &= 2\,RTT_T + \frac{O}{R} + (T-1)\frac{L}{R} + P_T\cdot\left(RTT_T + T\cdot\frac{L}{R}\right) - \left(2^{P_T} - 1\right)\frac{L}{R} \end{aligned}$$

C.6 Übung 6

Aufgabe 6.1

Knoten: $V = \{A, B, C, S, P\}$
Kanten: $E = \{(A, B), (B, C), (A, S), (A, P), (B, S), (B, P), (C, P)\}$
Kosten: $c(A, S) = c(C, P) = 1, c(A, B) = c(B, C) = 2, c(B, S) = c(B, P) = 4, c(A, P) = 7$

a) Zeichnen Sie den Graphen G.

b) Führen Sie aus Sicht des Knoten S das Dijkstra-Verfahren für die Minimierung des Graphen G durch.

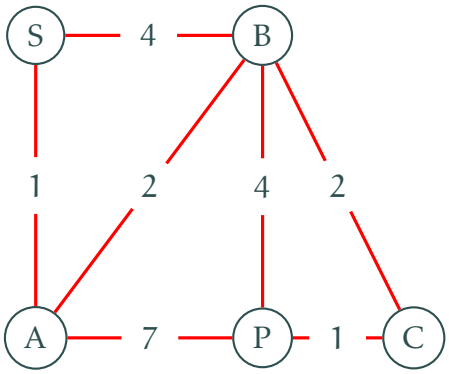
c) Zeichnen Sie den aus dem Dijkstra-Verfahren für Knoten S resultierenden minimalen Spannbaum.

d) Verfolgen Sie den Ablauf des Forward-Search-Algorithmus für den Knoten S.

e) Geben Sie basierend auf dem Forward-Search-Algorithmus für den Knoten S die Forwarding-Tabelle an.

Lösungsvorschlag

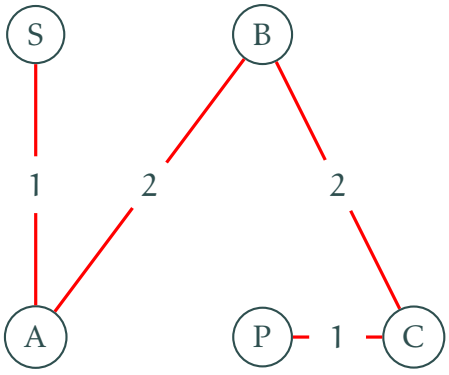
a) Graph G:



b) Dijkstra-Verfahren für G aus Sicht des Knoten S:

Schritt	V'	A	B	C	P
0	S	1,S	4,S	∞, -	∞, -
1	S,A	"	3, A	∞, -	8,A
2	S,A,B	"	"	5,B	7,B
3	S,A,B,C	"	"	"	6,C
4	S,A,B,C,P	"	"	"	"

c) Minimaler Spannbaum:



d) Forward-Search-Algorithmus:

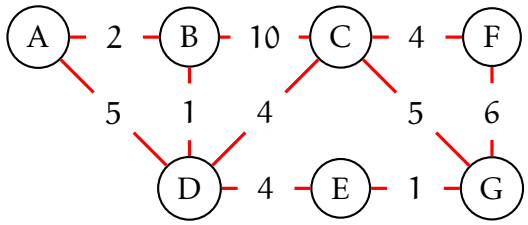
Schritt	bestätigt	vorläufig
0	(S, 0, -)	-
1	(S,0, -)	(A, 1, A), (B,4,B)
2	(S,0,-), (A, 1, A)	(B,3,A), (P,8,A)
3	(S,0,-), (A,1,A),(B,3,A)	(P,7,A), (C,5,A)
4	(S,0,-),(A,1,A),(B,3,A), (C,5,A)	(P, 6, A)
5	(S,0,-),(A,1,A),(B,3,A), (C,5,A), (P, 6, A)	-

e) Forwarding-Tabelle für Knoten S:

Knoten	Kosten	nextHop
A	1	A
B	3	A
C	5	A
P	6	A
S	0	-

Aufgabe 6.2

Graph G:



a) Führen Sie aus der Sicht des Knoten A das Dijkstra-Verfahren zur Minimierung des Graphen G durch.

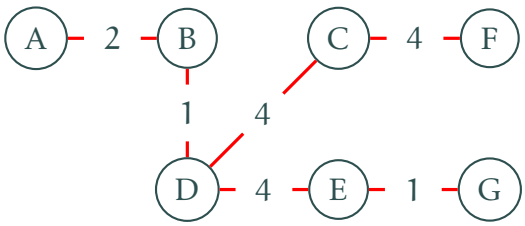
b) Zeichnen Sie den resultierenden minimalen Spannbaum.

Lösungsvorschlag

a) Dijkstra-Algorithmus von Knoten A des Graphen G:

Schritt	V'	B	C	D	E	F	G
0	A	2,A	∞, -	5,A	∞, -	∞, -	∞, -
1	A,B	"	12, B	3,B	∞, -	∞, -	∞, -
2	A,B,D	"	7,D	"	7,D	∞, -	∞, -
3	A,B,D,C	"	"	"	7,D	11,C	12,C
4	A,B,D,C,E	"	"	"	"	11, C	8,G
5	A,B,D,C,E,G	"	"	"	"	11,C	"
6	A,B,D,C,E,G,F	"	"	"	"	"	"

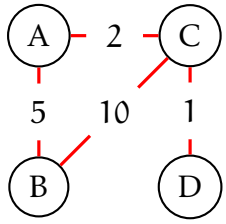
b) Minimaler Spannbaum:



C.7 Übung 7

Aufgabe 7.1

Führen Sie den Distanzvektor-Algorithmus für den Graph G durch:



Lösungsvorschlag

- Initialisierung:
  - Knoten A

Knoten	D()	nh()
B	5	B
C	2	C
D	∞	-

  - Knoten B

Knoten	D()	nh()
A	5	A
C	10	C
D	∞	-

  - Knoten C

Knoten	D()	nh()
A	2	A
B	10	B
D	1	D

  - Knoten D

Knoten	D()	nh()
A	∞	-
B	∞	-
C	1	C

- Schritt 1:
  - Knoten A

Knoten	D()	nh()
B	5	B
C	2	C
D	3	C

  - Knoten B

Knoten	D()	nh()
A	5	A
C	7	A
D	11	C

  - Knoten C

Knoten	D()	nh()
A	2	A
B	7	A
D	1	D

  - Knoten D

Knoten	D()	nh()
A	3	C
B	11	B
C	1	C

- Schritt 2:
  - Knoten A

Knoten	D()	nh()
B	5	B
C	2	C
D	3	C

  - Knoten B

Knoten	D()	nh()
A	5	A
C	7	A
D	8	C

  - Knoten C

Knoten	D()	nh()
A	2	A
B	7	A
D	1	D

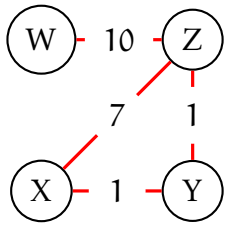
  - Knoten D

Knoten	D()	nh()
A	3	C
B	8	B
C	1	C

  - Schritt 3: Konvergenz erreicht

Aufgabe 7.2

Führen Sie den Distanzvektor-Algorithmus für den Graph G durch:



Lösungsvorschlag

- Initialisierung:
  - Knoten W

Knoten	D()	nh()
X	∞	-
Y	∞	-
Z	10	Z

  - Knoten X

Knoten	D()	nh()
W	∞	-
Y	1	Y
Z	7	Z

  - Knoten Y

Knoten	D()	nh()
W	∞	-
X	1	X
Z	1	Z

– Knoten Z

Knoten	D()	nh()
W	10	W
X	7	X
Y	1	Y

- Schritt 1:
  – Knoten W

Knoten	D()	nh()
X	17	Z
Y	11	Z
Z	10	Z

  – Knoten X

Knoten	D()	nh()
W	17	Z
Y	1	Y
Z	2	Y

  – Knoten Y

Knoten	D()	nh()
W	11	Z
X	1	X
Z	1	Z

  – Knoten Z

Knoten	D()	nh()
W	10	W
X	2	Y
Y	1	Y

- Schritt 2:
  – Knoten W

Knoten	D()	nh()
X	12	Z
Y	11	Z
Z	10	Z

  – Knoten X

Knoten	D()	nh()
W	12	Y
Y	1	Y
Z	2	Y

  – Knoten Y

Knoten	D()	nh()
W	11	Z
X	1	X
Z	1	Z

  – Knoten Z

Knoten	D()	nh()
W	10	W
X	2	Y
Y	1	Y

- Schritt 3: Konvergenz erreicht

C.8 Übung 8

Aufgabe 8.1

Zeichnen Sie ein Ausbreitung-Zeit-Diagramm, in dem die Knoten A und B sich in einem 10 Mbps Ethernet-Segment befinden, keine weiteren Knoten aktiv sind, die Rahmengröße 512 Bit ist und das Propagation-Delay 225 Bit-Zeiten, A und B gleichzeitig Daten senden zum Zeitpunkt t = 0 Bit-Zeiten, die Frames kollidieren und A & B unterschiedliche Werte für die Backoff-Konstante K wählen: $K_A = 0$ und $K_B = 1$, die Backoff-Zeit = 512 Bit-Zeiten ist, A und B ihre Übertragung beginnen und das Jam-Signal 48 bit lang ist.
Beantworten Sie dabei folgende Fragen:

a) Zu welcher Bitzeit beginnt A mit der Sendewiederholung?

b) Zu welcher Bitzeit ist der Sendewiederholungsbeginn von B geplant?

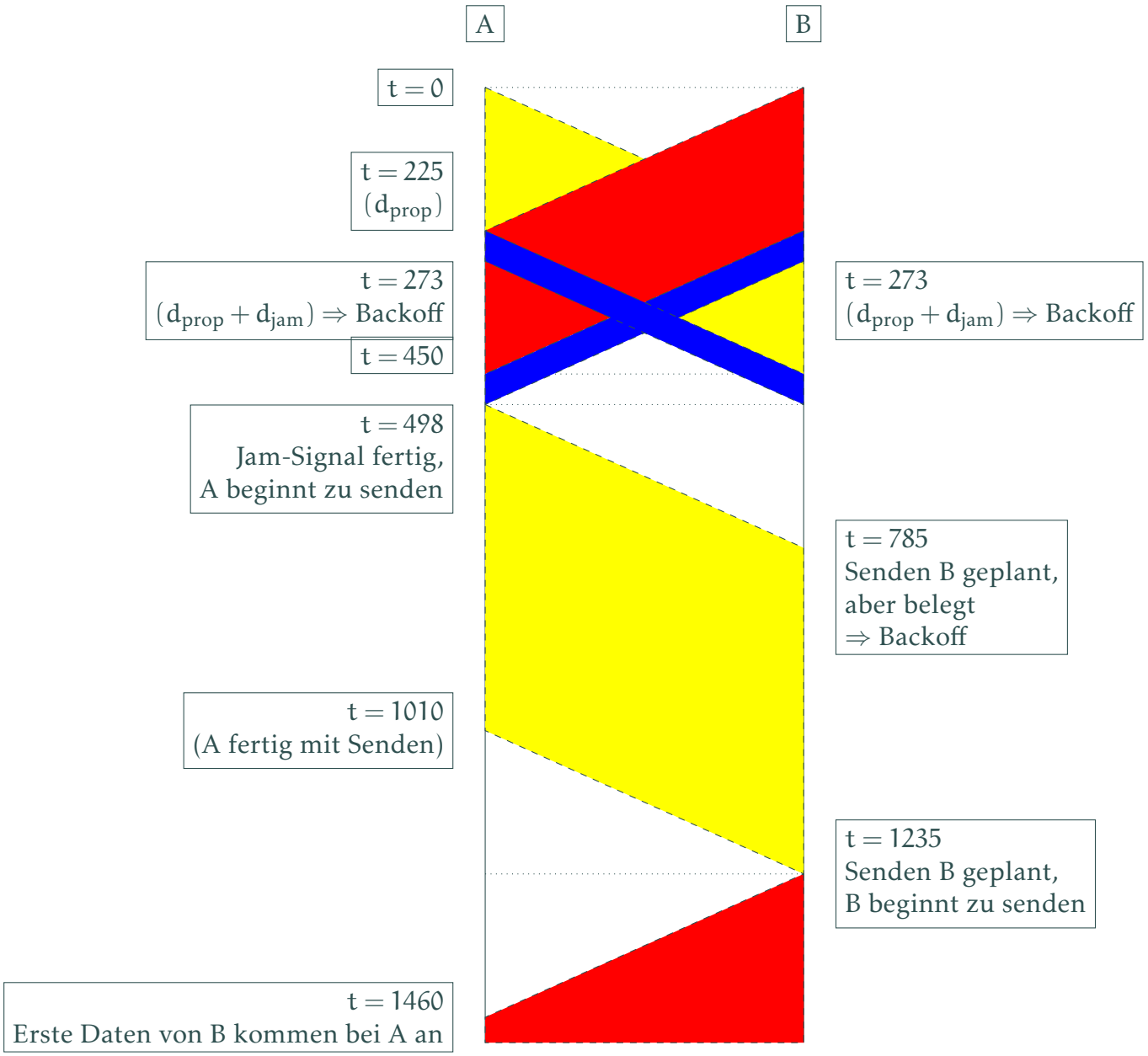
c) Können die Sendewiederholungen von A und B kollidieren?

d) Wann führt B die Sendewiederholung durch?

e) Welches Verhalten ergäbe sich, wenn A im direkten Anschluss an den ersten Rahmen einen weiteren senden würde?

Lösungsvorschlag

Diagramm:



Antworten:

a) $t = 498 = d_{prop} + d_{jam} + d_{prop}$

b) $t = 785 = d_{prop} + d_{jam} + d_{backoff}$

c) Wenn beide die gleiche Backoff-Konstante wählen schon, dies ist aber hier nicht der Fall.

d) $t = 1235 = d_{prop} + d_{jam} + d_{backoff} + d_{backoff}$

e) Dann würde B senden, zeitgleich aber das Signal von A empfangen ⇒ Kollision ⇒ Jamming ⇒ Backoff

Aufgabe 8.2

Betrachten Sie ein 100 Mbps Ethernet-Netzwerk mit 100BASE-T4, Halbduplex, CSMA/CD, wobei zwei Knoten mit einem Hub in der Mitte verbunden sind. Die Frame-Länge beträgt 64 Bytes, die Ausbreitungsgeschwindigkeit $c = 2 \cdot 10^8 \frac{m}{s}$.

a) Wie groß ist die Ausbreitungsverzögerung zwischen zwei Knoten, um eine Effizienz von 0,5 zu erreichen?

b) Stellt diese Ausbreitungsverzögerung sicher, dass Knoten A erkennen kann, dass ein weiterer Knoten sendet, während A selbst sendet? Warum bzw. warum nicht?

c) Wie groß ist die Kabellänge zwischen einem Knoten und dem Hub?

Lösungsvorschlag

a)

$$\begin{aligned} S = 0{,}5 &= \frac{1}{1 + 4{,}4 \cdot a} \\ &= \frac{1}{1 + 4{,}4 \cdot \frac{R \cdot D}{L}} \\ 2 &= 1 + 4{,}4 \cdot \frac{R \cdot D}{L} \\ \frac{1}{4{,}4} &= \frac{R \cdot D}{L} \\ \frac{L}{4{,}4 \cdot R} &= D \\ \frac{64 \cdot 8 \text{ Bits}}{4{,}4 \cdot 100 \frac{Mb}{s}} &= D \\ 0{,}000001164 \text{ s} &= D \end{aligned}$$

b) Damit die Überlagerung sicher erkannt werden kann, muss gelten:

$$\frac{L}{R} > 2 \cdot D$$

Im konkreten Fall gilt

$$\frac{64 \cdot 8 \text{ Bits}}{100 \frac{Mb}{s}} = 0{,}00000512 \text{ s} > 0{,}000002327 \text{ s} = 2 \cdot D$$

Damit kann die Überlagerung sicher erkannt werden.

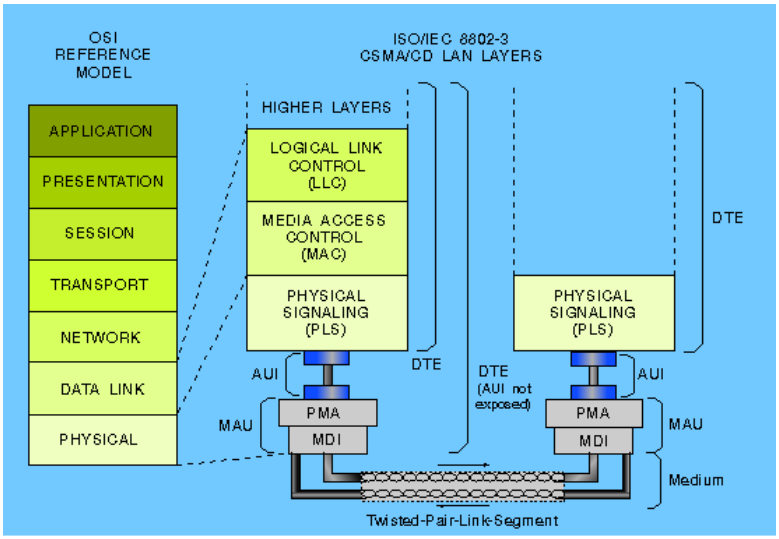
c)

$$\begin{aligned} c &= \frac{2l}{D} \\ \frac{D \cdot c}{2} &= l \\ \frac{0{,}000001164 \text{ s} \cdot 2 \cdot 10^8 \frac{m}{s}}{2} &= l \\ 116{,}36 \text{ m} &= l \end{aligned}$$

GLOSSAR

10Base-T

10Base-T ist der Teil des 802.3-Standards, der den Einsatz einer sternförmigen Verkabelungsstruktur für Ethernet erlaubt. 10Base-T spezifiziert ein CSMA/CD-Netz mit 10 Mbit/s auf UTP-Kabel mit RJ45-Stecker. Der Bus konzentriert sich bei dieser 802.3-Version in einem Hub. Alle Stationen sind mit diesem Hub sternförmig über Vierdraht-Leitungen verbunden. Es werden zwei Adernpaare des TP-Kabels verwendet: Receive und Transmit.
Der Aderndurchmesser liegt zwischen 0,4 mm (AWG 26) und 0,6 mm (AWG 22) mit einer Impedanz von 85 bis 111 Ohm und einer Einkopplungsimpedanz von 100 Ohm. Die Ausbreitungsgeschwindigkeit auf dem UTP-Kabel ist <0,59c. Die zulässige Dämpfung darf im Frequenzbereich zwischen 5 MHz und 10 MHz Werte von 11,5 dB nicht überschreiten. Die Nahnebensprechdämpfung errechnet sich aus der Formel <26-15 log(f/10) und liegt bei 5 MHz bei 30,5 dB und bei 10 MHz bei 26,0 dB. 10 Base-T hat eine Bitfehlerrate, die unterhalb von 1: 100.000.000 liegt.



Die max. Entfernung zwischen zwei MAUs ohne Zwischenverstärker wurde auf 100 m festgelegt, wobei der Hub ebenfalls eine Ansammlung von Transceivern (MAU) darstellt, die über den internen Bus des Hubs zusammengeschaltet werden. In diesen 100 m sind allerdings Wandsteckdosen und Rangierverteiler sowie die Entfernungen, z.B. zwischen Endgeräten und Steckdosen, inbegriffen.

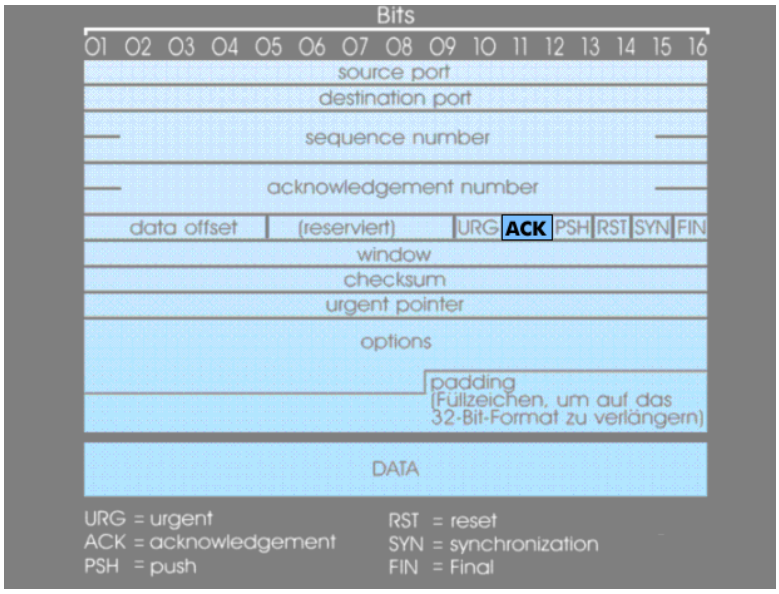
Abrechnungsmanagement (*accounting*)

Das Abrechnungsmanagement, Accounting Management (AM), ist eine Funktion des OSI-Managements und des Fault, Configuration, Account, Performance, and Security Management (FCAPS), mit dem Accounting-Informationen zur Systembenutzung gesammelt werden. Die Abrechnungsinformationen zeigen an wer, wann und für welchen Zeitraum welche Netzwerk-Ressource benutzt hat.
Anhand dieser Daten können die Kosten für die Benutzung des Netzwerks

und der Services anwenderbezogen zugeordnet werden. Anwendungsbeispiele für das Abrechnungsmanagement sind Billing- und Rating-Funktionen. Das Abrechnungsmanagement wird auch in Telekommunikationsnetzen eingesetzt. Bei dieser Managementfunktion geht es um die Kostenzuordnung zu den Benutzern, die Verwaltung der Benutzerkonten, um Benutzerstatistiken und die Rechnungserstellung. *Siehe auch*: *FCAPS (S. G20)*.

ACK (acknowledgment flag)

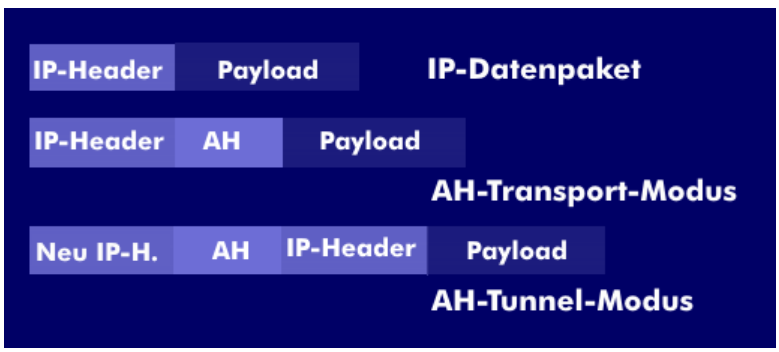
Das Acknowledgement Frame (ACK) wird in der Datenübertragung für die Flusskontrolle in einer Ende-zu-Ende-Verbindung benutzt. Als positive Bestätigung wird das ACK-Frame übertragen um den Empfang von einem oder mehreren Datenpaketen zu bestätigen.
Das ACK-Flag oder -Frame wird in verschiedenen Protokollen wie dem TCP-Protokoll oder bei Fibre Channel (FC) benutzt. So beispielsweise als Bit-Indikator im Control-Flag-Feld des TCP-Headers. Ein gesetztes Flag zeigt an, dass die im Bestätigungsfeld eingetragene Acknowledgement Number relevant ist.



Bei Fibre Channel wird es in den FC-Dienstklassen Class 1 und Class 2 benutzt um den Empfang von einem oder mehreren Datenpaketen zu bestätigen.

AH (authentication header)

Der Authentication Header (AH) stellt die Datenintegrität von den IP-Protokollen IPv6 und IP Security Protocol (IPsec) durch die Prüfsumme sicher und sorgt für die Authentifizierung des Datenursprungs.



Mit dem Authentifizierungs-Header lässt sich also feststellen, ob die Nachricht von dem angenommenen Absender stammt und ob ihr Inhalt unverändert ist. Der AH-Header, der in RFC 2402 spezifiziert ist, schützt den gesamten Nachrichteninhalt durch Verschlüsselung des IP-Paketes. Für die Verschlüsselung wird der HMAC-Algorithmus, Hash-Based Message

Authentication Code (HMAC); verwendet. Ein zusätzliches Nummernfeld im AH-Header wehrt Replay-Angriffe ab.

Aktives FTP

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. Dies ist typischerweise ein Port des Clients, der jenseits von 1023 liegt, kann aber auch ein anderer Server sein, der seinerseits in den Passive Mode geschaltet wurde, also auf eine Verbindung wartet (so genanntes FXP). Heutzutage ist FXP jedoch bei den meisten FTP-Servern aus Sicherheitsgründen standardmäßig deaktiviert. Die Datenübertragung auf der Server-Seite erfolgt dabei über Port 20. Die Kommunikation mit Befehlen erfolgt ausschließlich auf dem Control Port. Man spricht auch von der Steuerung „Out of Band“. Somit bleibt es möglich, dass während der Datenübertragung die Partner noch immer miteinander kommunizieren können. *Siehe auch*: *FTP (S. G25) und Out-of-Band-Control (S. G57)*.

American National Standards Institute (ANSI)

Das American National Standards Institute (ANSI) ist eine private, gemeinnützige, amerikanische Organisation zur Koordinierung der Entwicklung freiwilliger Normen in den Vereinigten Staaten. Es ist das einzige US-amerikanisches Mitglied in der Internationalen Organisation für Normung. Das deutsche Pendant ist das Deutsche Institut für Normung e. V. (DIN), das österreichische Pendant das Austrian Standards Institute und das schweizerische Pendant die Schweizerische Normen-Vereinigung.

Anwendungsschicht (application layer) (OSI)

Die Anwendungsschicht, Application Layer (APL), ist die 7. Schicht des OSI-Referenzmodells. Aufgabe der Anwendungsschicht ist die Bereitstellung von Anwendungsdiensten mit entsprechenden Datenstrukturen und Protokollen.

Bei der Entwicklung der Standards wurden zunächst zwei Richtungen verfolgt. Zum einen die Bereitstellung von anwendungsorientierten Grunddiensten für Standardanwendungen wie Dateitransfer, elektronische Post und Filesharing. Zum anderen die Bereitstellung von Grundfunktionen innerhalb der Anwendungsschicht, die von speziellen Anwendungen genutzt werden können. Daher gibt es zwei Klassen von Dienstelementen innerhalb der Anwendungsschicht. Allgemein verwendbare Anwendungs-Dienstelemente (ASE), die Common Application Service Elements (CASE), und anwendungsspezifische Elemente, die Specific Application Service Elements (SASE). Dienstelemente von SASE sind u.a. File Transfer, Access, and Management (FTAM) und Virtual Terminal Service (VTS). Beispiel einer ITU-T-Empfehlung ist das Message Handling System (MHS). *Siehe auch*: *ITU-T (S. G42) und OSI (S. G41)*.

Anycast

Anycast ist eine Adressierungsart in Computernetzen, bei der einer ganzen Gruppe von Rechnern eine gemeinsame Adresse zugeteilt ist, unter der man aber nur den Rechner erreicht, der über die kürzeste Route, und somit am schnellsten, erreichbar ist. Diese Technik kommt gemäß OSI-Modell in der Vermittlungsschicht zum Einsatz.

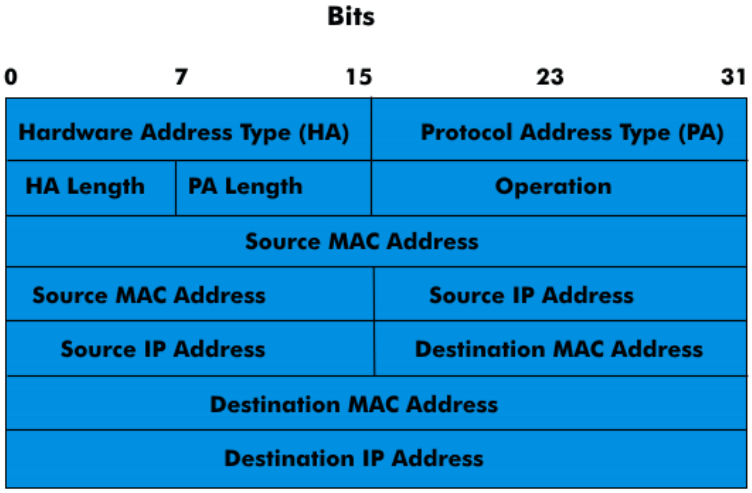
Die Bezeichnung Anycast steht im Gegensatz zum Multicast bei der mit einer sendenden Station eine ganze Gruppe von Empfangsstationen angesprochen werden kann.

Die Anycast-Adressierung eignet sich besonders für das Aktualisieren der Routingtabellen einer Hostgruppe. IPv6 bestimmt hierbei den am nächsten liegenden Router und sendet diesem die Datenpakete, ähnlich dem Unicast. Der angesprochene Router sendet die Datenpakete weiter im Anycast bis alle Routingtabellen aktualisiert sind. Die Anycast-Adressierung von IPv6 ist in RFC 2373 näher beschrieben. *Siehe auch*: *Kommunikationsart (S. G43)*.

ARP (address resolution protocol)

Das Address Resolution Protocol (ARP) ist ein typisches ES-IS-Protokoll, End System to

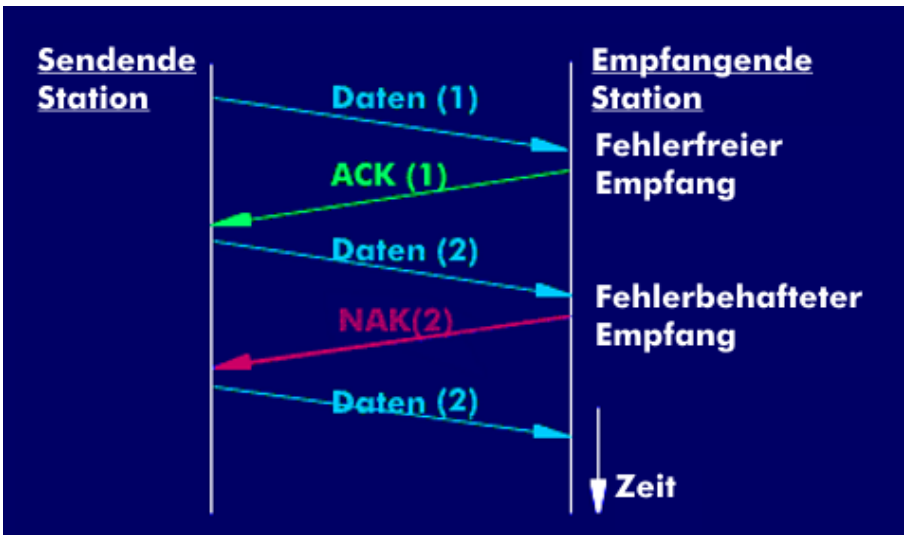
Intermediate System (ES-IS), mit dem Netzwerkadressen auf Hardware-Adressen abzubilden. So können beispielsweise MAC-Adressen in die zugehörigen IP-Adressen umgewandelt werden, damit überhaupt eine Kommunikation auf der Vermittlungsschicht mittels des IP-Protokolls stattfinden kann.
Das ARP-Protokoll legt zu diesem Address-Mapping Adresstabellen an, die die MAC-Adressen den Netzwerkadressen zuordnen. Vor dem Verbindungsaufbau über das Ethernet fragt das IP-Protokoll beim Address Resolution Protocol nach der Ethernet-Adresse der zugehörigen Ziel-Internet-Adresse an. Das ARP-Protokoll vergleicht seine Adresstabellen mit der Anfrage.



Hat ARP keinen Eintrag in seiner Tabelle, so wird über eine Anfrage an alle Netzknoten (Broadcast) die Ethernet-Adresse der zugehörigen Internetadresse erfragt. Nur Netzknoten mit einem Eintrag zu dieser IP-Adresse antworten auf die Anfrage. Die Antwort auf den ARP-Broadcast wird in der ARP-Adresstabelle gespeichert.
Da wegen der unterschiedlichen Adresslängen von MAC-Adressen (48 Bit) und IP-Adressen (32 Bit) kein unmittelbares Mapping möglich ist wie bei anderen Protokollen, wird beim ARP-Protokoll die Mapping-Tabelle auf Basis eines dynamischen Abfrage-Algorithmus angelegt und regelmäßig überprüft.
ARP ist ein nichtroutbares Protokoll, das in RFC 926 und 1577 beschrieben ist.

ARQ (automatic repeat request)

Automatic Repeat Request (ARQ) ist ein Quittungsbetrieb für die gesicherte Datenübertragung, bei der die Datensenke, also die empfangende Station, Fehler erkennt und um Übertragungswiederholung nachsucht. Im Fehlerfall oder bei Nichteintreffen der Datenpakete wird eine automatische Wiederholung der Sendung ausgelöst.
Im normalen, fehlerfreien Betrieb sendet die Datensenke nach jedem Datenpaket eine positive Bestätigung (ACK). Nach dessen Empfang sendet die Datenquelle das nächste Datenpaket. Ist ein Datenpaket fehlerhaft oder ganz ausgeblieben, dann sendet der Empfänger eine negative Bestätigung (NAK) und die Datenquelle sendet erneut das gleiche Datenpaket.



Verfahrensmäßig unterschiedet man zwischen dem Continuous ARQ, dem Stop-and-Wait ARQ, dem Selective Repeat (SR) und dem Hybrid Automatic Repeat Request (HARQ).
Zu den ARQ-Protokollen gehören Protokolle mit Flusskontrolle und quittierter Empfangsbestätigung für die empfangenen Datenpakete. Man unterscheidet zwischen zeichenorientierten Protokollen, bei denen nach jedem übertragenen Zeichen auf ein ARQ gewartet wird, und bitorientierten, bei denen die Quittierung durch die Steuerung der Fenstergröße, wie bei High Level Data Link Control (HDLC) und Synchronous Data Link Control (SDLC) geregelt wird.

ASN.1 (abstract syntax notation one)

Eine abstrakte Syntax der Darstellungsschicht des OSI-Referenzmodells beschreibt die Menge und Art der Datentypen, die von der Anwendungsschicht in Form einer Application Protocol Data Unit (APDU) an die Darstellungsschicht übergeben werden, damit diese die APDUs an das korrespondierende Endsystem weiterleitet.
Im ISO-Standard 8824 wird eine Notation beschrieben, mit der es möglich ist, abstrakte Syntaxen zu definieren. Diese Sprache wird Abstract Syntax Notation One (ASN.1) genannt. Mit ASN.1 lassen sich relativ einfache hierarchisch strukturierte Datentypen beschreiben. ASN.1 bietet eine Verständigungsmöglichkeit zweier Kommunikationspartner im Rahmen der Darstellungsschicht. Beim Datentransfer wandelt jeder Kommunikationspartner seine lokale Darstellung in eine gemeinsame um.
Das TCP/ IP-Protokoll benutzt anstelle von ASN.1 das External Data Representation (XDR) um beispielsweise die Darstellungsprobleme beim Remote Procedure Call (RPC) zu lösen.
Als ein Beispiel dient im Folgenden das Foo-Protokoll. Der Typ `FooQuestion` steht dabei für eine Frage, der Typ `FooAnswer` für eine Antwort. Eine Antwort ist über die Fragennummer mit der jeweiligen Frage verbunden, wobei nur Ja/Nein-Fragen zugelassen sind. Es ergibt sich in ASN.1-Notation:

```
1  FooProtocol DEFINITIONS ::= BEGIN
2      FooQuestion ::= SEQUENCE {
3          trackingNumber INTEGER,
4          question       STRING
5      }
6
7      FooAnswer ::= SEQUENCE {
8          questionNumber INTEGER,
9          answer         BOOLEAN
10     }
```

```
11 | END
```

Siehe auch: *OSI (S. G41)*.

Ausbreitungsverzögerung

Mit einer Länge einer Verbindung ℓ und einer Signalausbreitungsgeschwindigkeit von v ergibt sich eine Ausbreitungsverzögerung von

$$D = \frac{\ell}{v}.$$

Baumtopologie

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 („Wurzeln“ weiterer „Äste“). Sie obliegt einer strengen hierarchischen Ordnung, wobei Verbindungen zwischen Verteilern mittels Uplinks hergestellt werden. Vor- und Nachteile:

+ Ausfälle von Endgeräten haben keine Konsequenzen
+ Strukturelle Erweiterbarkeit
— Ausfälle von Verteilern betreffen ganze Unterbäume
— Es kann durch die definierte Bisektionsweite von 1 an der Wurzel zu Engpässen bei der Kommunikation zwischen Baumhälften kommen.
— Bei klassischen Bäumen kommt es zu schlechten Latezeigenschaften

Siehe auch: *Topologie (S. G82)*.

BER (basic encoding rules)

Basic Encoding Rules (BER) sind Grundregeln für die Codierung von Daten, die in Abstract Syntax Notation One (ASN.1) beschrieben werden. Mit diesen Regeln können Datenelemente codiert werden, die zur Spezifizierung von ASN.1-Elementen als Byte-String dienen. Der String umfasst Datenfelder für den Typ, die Länge und den Wert. Das Typ-Feld zeigt die Objekt-Klasse an, das Längenfeld die Anzahl der Bytes, mit denen der Wert codiert wurde, und das Datenfeld für den Wert zeigt die Information, die mit ASN.1 in Verbindung stehen. Die Typklasse wird von den beiden hochwertigsten Bits, Bit 8 und Bit 7, bestimmt. Es gibt folgende Typklassen: Universell (00), Anwendung (01), Kontext (10) und Privat (11).
Die Distinguished Encoding Rules (DER) sind eine Untermenge der Basic Encoding Rules und zeigen u.a. an, wie die Längenparameter der Datenobjekte codiert sind. Die Packed Encoding Rules (PER) sind die effizienteren Nachfolgeregeln der Basic Encoding Rules.

BGP (border gateway protocol)

Das Border Gateway Protocol (BGP) ist ein Exterior Routing Protocol (ERP), das die Routinginformationen zwischen autonomen Systemen (AS) überträgt. Beim Border Gateway Protocol (BGP) informieren sich die Router untereinander über die verfügbaren Verbindungswege. Die Stärke des BGP-Protokolls liegt darin, verschiedene optionale Routing-Pfade in einer einzigen Routingtabelle zu vereinen.
Das Border-Gateway-Protokoll ist wie das Exterior Gateway Protocol (EGP), das vom BGP-Protokoll abgelöst wird, ein Path-Vector-Protokoll für das Routing zwischen autonomen Systemen (AS). BGP unterstützt eine Metrik und kann intelligente Routing-Entscheidungen treffen. Es kann mit Open Shortest Path First (OSPF) als internem Routing-Protokoll zusammenarbeiten. Insbesondere wird die Routen-Aggregierung von Classless Interdomain Routing (CIDR) unterstützt.
Die BGP-Information enthält alle Daten über den kompletten Pfad zwischen den autonomen Systemen. Anhand dieser Information erstellt das Protokoll einen Graphen, der die Vernetzung der verschiedenen Autonomen Systeme darstellt und eine Schleifenbildung

des Routings ausschließt. Das Routing-Update, bei dem ein BGP-Router mit anderen BGP-Systemen in Verbindung steht, wird mittels TCP-Protokoll übertragen. Die vom BGP-Protokoll verwendete Metrik basiert auf Informationen, die der Netzverwalter den Routern bei deren Konfiguration zuweist, sowie auf den physikalischen und übertragungstechnischen Parametern. Da jeder BGP-Router über Routen-Informationen von anderen, insbesondere den benachbarten BGP-Routern verfügt, baut sich jeder BGP-Router eine Datenbank für die Routen zu allen erreichbaren Autonomen Systemen auf.
Das BGP-Protokoll ist in den RFCs 1163 und 1771 beschrieben. 1991 wurde im RFC 1269 die Border Gateway Protocol (Version 3) Management Information Base (MIB) veröffentlicht. Derzeit wird die Version 4 eingesetzt, die CIDR unterstützt. Sie ist im RFC 4271 beschrieben und eignet sich für Gigabit-Ethernet.
Das BGB-Protokoll gibt es auch als Interior-BGP (IBGP) und als Exterior-BGP (EBGP), das das Routing zwischen zwei autonomen Systemen (AS) ausführt.

Bitrate

Die Bitrate wird auch als Übertragungsgeschwindigkeit, Übertragungsrate oder Datenrate bezeichnet. Es handelt sich um die Anzahl der Bits, die pro Zeiteinheit (in der Regel 1 Sekunde) übertragen werden. Die Bitrate wird in bit/s (Bits pro Sekunde) bzw. in den entsprechenden Präfixen als Kilobit pro Sekunde (kbit/s), Megabit pro Sekunde (Mbit/s), Gigabit pro Sekunde (Gbit/s) oder Terabit pro Sekunde (Tbit/s) angegeben. In der amerikanischen Schreibweise werden die Schreibweisen bps, kbps, Mbps und Tbps benutzt.

Bitübertragungsschicht (physical layer) (OSI)

Die Bitübertragungsschicht, Schicht 1 im OSI-Referenzmodell, legt die elektronischen, funktionalen und prozeduralen Parameter und Hilfsmittel für die physikalische Verbindung zwischen Einheiten an einem Netz fest.
Die wichtigste Funktion der Bitübertragungsschicht ist die Aufrechterhaltung einer physikalischen Verbindung. Es werden insbesondere die Struktur der Bits, die Bedeutung der Bits und die Methoden zur Übertragung einzelner Bits festgelegt.
Als Dienst für die höheren Schichten können die Bereitstellung der physikalischen Verbindung und der transparente Transport von Dateneinheiten genannt werden. Bei großen Datenübertragungsstrecken können durch die Qualität des Übertragungsmediums verursachte oder atmosphärisch bedingte Übertragungsfehler auftreten. Diese wirken sich vielleicht auf die Wellenform des zu übertragenden Signals oder die gesamte Biteinheit negativ aus. Durch geeignete Baugruppen - Verstärker, Regeneratoren - können solche Fehler beseitigt werden. Können Fehler nicht behoben werden, ohne dass die Kommunikation der nächsten darüber liegenden Schicht davon betroffen wäre (z.B. Leitungsausfall für längere Zeit, starke elektromagnetische Störungen etc.), ist es ein Dienst der Bitübertragungsschicht, Fehlerinformationen an diese weiterzugeben. *Siehe auch*: *OSI (S. G41)*.

Broadcast

Ein Broadcast (engl. Sendung, Übertragung, Rundfunk, Ausstrahlung, hier Rundruf) in einem Rechnernetz ist eine Nachricht, bei der Datenpakete von einem Punkt aus an alle Teilnehmer eines Nachrichtennetzes übertragen werden. In der Vermittlungstechnik ist ein Broadcast eine spezielle Form der Mehrpunktverbindung. Ein Broadcast-Paket erreicht alle Teilnehmer eines lokalen Netzes, ohne dass sie explizit als Empfänger angegeben sind.
Jeder Empfänger eines Broadcasts entscheidet selbst, ob er im Falle seiner Zuständigkeit die erhaltene Nachricht entweder verarbeitet oder andernfalls stillschweigend verwirft. Broadcasts gibt es auf verschiedenen Schichten des OSI-Referenzmodells. Allen gemein ist, dass Broadcasts von einer höheren Schicht an die unteren Schichten entsprechend angepasst werden müssen.
Siehe auch: *Kommunikationsart (S. G43)*.

Bustopologie

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. Wenn das Übertragungsmedium eines Busses ein Shared Medium ist – also z. B. dieselbe Kupferader von allen Teilnehmern gemeinsam zur Datenübertragung verwendet wird – muss sichergestellt werden, dass immer nur ein Gerät zum selben Zeitpunkt Signale auf das Übertragungsmedium sendet. Vor- und Nachteile:

+ Nur geringe Kosten, da nur geringe Kabelmengen erforderlich sind.
+ Einfache Verkabelung und Netzerweiterung
+ Keine aktiven Netzwerkkomponenten
— Leichtes Abhöhren von Datenübertragung
— Eine Störung des Übertragungsmediums blockiert den gesamten Netzstrang
— Zu einem Zeitpunkt kann nur eine Station Daten senden.

Siehe auch: *Topologie (S. G82)*.

ccTLD (country code top level domain)

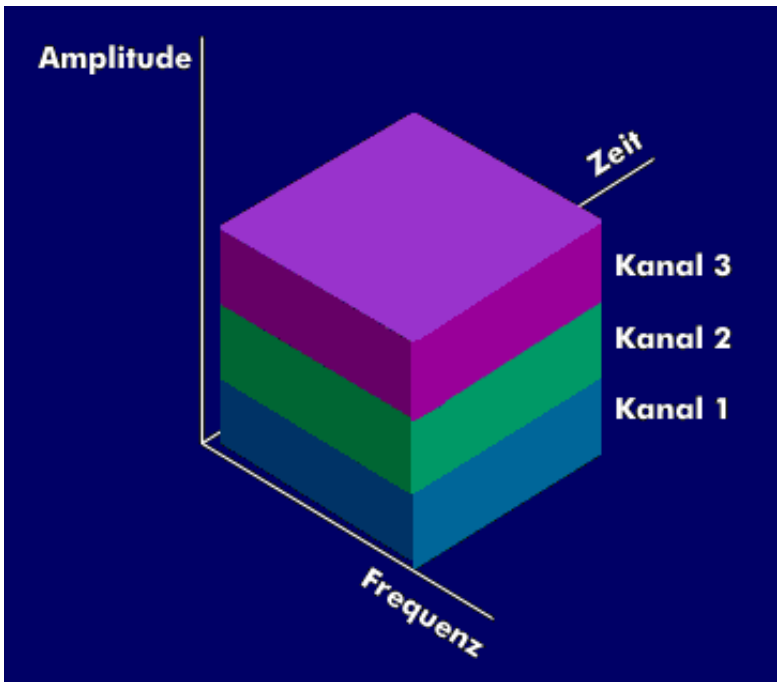
Bei denTop-Level-Domains (TLD) unterscheidet man zwischen den geografischen oder landesbezogenen Country Code Top Level Domains (ccTLD) und den organisatorischen und generischen, den Generic Top Level Domains (gTLD). Länder-Domänen sind Ländern oder geografischen Regionen zugeordnet und bestehen aus zwei Buchstaben. So steht beispielsweise die Top Level Domain *.de für Deutschland, *.fr für Frankreich, *.at für Österreich und *.ca für Kanada. Die Regeln und Richtlinien für die Registrierung von ccTLD-Domains variieren relativ stark.

Domain ccTLD	Land	Domain ccTLD	Land
.at	Österreich	.it	Italien
.au	Australien	.jp	Japan
.ca	Kanada	.li	Lichtenstein
.ch	Schweiz	.nl	Niederlande
.de	Deutschland	.no	Norwegen
.fi	Finnland	.se	Schweden
.fr	Frankreich	.ru	Russland
.gb	Großbritannien	.uk	United Kingdom
.ie	Irland	.us	USA

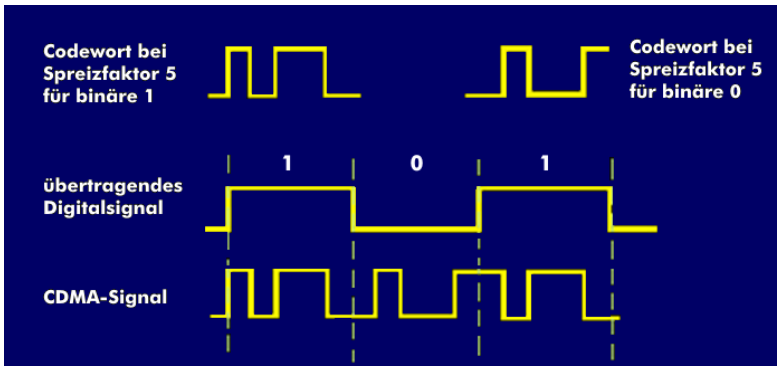
Siehe auch: *DNS (S. G15)*.

CDMA (code division multiple access)

Code Division Multiple Access (CDMA) ist ein funktechnisches Mehrfachzugangsverfahren, das mehreren Benutzern den Zugriff auf einen Funk-Übertragungskanal ermöglicht.
In diesem Verfahren, das in Mobilfunksystemen eingesetzt wird, belegen alle Benutzer denselben Frequenzbereich, jedoch wird das Nutzsignal für jeden Benutzer unterschiedlich codiert. Für die Übermittlung der Bits wird für den Nutzer Äëin anderer Code verwendet wie für den Benutzer "B".



Die Codierung basiert auf einer Spreizung des Nutzdatenkanals. Dabei werden die einzelnen Bits eines schmalbandigen Nutzsignals durch längere Bitkombinationen ersetzt. Ersetzt man ein Bit durch eine Bitkombination von beispielsweise zehn Bits, dann erreicht man eine Spreizung um Faktor 10. Man benötigt zwar eine höhere Übertragungsbandbreite, kann aber den Übertragungskanal gleichzeitig für mehr Nutzkanäle verwenden. Die Daten der einzelnen Benutzer sind im Übertragungskanal klar voneinander unterscheidbar.



Das CDMA-Verfahren bietet den Vorteil, dass die zur Verfügung stehende Bandbreite besser genutzt wird und einer größeren Zahl von Teilnehmern der Zugriff auf das Übertragungsmedium ermöglicht werden kann als bei Frequency Division Multiple Access (FDMA) oder Time Division Multiple Access (TDMA).
Das CDMA-Verfahren ist von der Telecommunications Industry Association (TIA) unter IS-95 genormt worden. Diese Norm basiert auf 1,25-MHz-Bändern, über die 128 Kanäle von 9,6 kbit/s übertragen werden können. Für die Signalspreizung wird ein Walsh-Code mit insgesamt 64 verschiedenen Codes verwendet.
CDMA gibt es in einer von der internationalen Fernmeldeunion (ITU) standardisierten Version CDMA2000.

CDMA/FDD (code division multiple access/frequency diversity duplex)

Code Division Multiple Access/ Frequency Division Duplex (CDMA/FDD) wird in Wireless Local Loops (WLL) eingesetzt, und zwar bei MMDS für exklusive, drahtlose Duplex-Verbindungen im Anschlussbereich. Die CDMA/FDD-Technik entspricht IS-95 CDMA. Bei diesem Verfahren sind alle Endanwender an das gleiche breitbandige Frequenzband angeschlossen; der gewünschte Endanwender wird durch einen einzigartigen Code

identifiziert. Nachteilig ist bei diesem Verfahren, dass es bei breitbandigen Übertragungen zu Bandbreitenengpässen kommen kann.

CDN (content distribution network)

Ein Content Distribution Network (CDN) ist ein Value Added Network (VAN) eines Internet Service Provider (ISP) oder eines Carriers, das an strategischen Stellen intelligente Router und Einrichtungen für die Lastverteilung und den Lastausgleich enthält. Die Aufgabe eines CDNs ist es, den Inhalt in der Nähe der Anwender abzulegen, damit dieser möglichst schnell auf den gewünschten Content zugreifen kann. Ein Content Distribution Network kann über das Internet, es kann aber auch über ein völlig unabhängiges Kernnetz konfiguriert werden. Ein Content Distribution Network setzt sich zusammen aus zentral verwalteten Cache- und Speichersystemen, die verteilt in einem Netzwerk eingesetzt werden. Durch die Verteilung sollen die Web-Inhalte näher an den Anwender gebracht, zur Verfügung stehenden Bandbreiten effizienter genutzt und der Datenzugriff beschleunigt werden. Konzeptionell besteht ein Content Distribution Network aus verschiedenen Komponenten wie den Knoten, in denen die Daten gespeichert werden und die für die Verbreitung des Contents sorgen ebenso wie für die Integration der Dienste, den Storage and Delivery Nodes. Diese Verteilung erfolgt über das Internet Content Adaption Protocol (ICAP).

Als weitere Komponente ist der zentrale Controller/Manager zu nennen, der für die Ablage der Inhalte, für die Verwaltung, der User Account und die Performance-Überwachung sorgt. Darüber hinaus gibt es noch den Request Manager der Content-Anfragen von Anwendern an die als nächstes verfügbare Storage Node verteilt.

Von der Konfiguration her gibt es für CDN-Netzwerke drei Grundstrukturen: Das Edge Distribution, die Edge Hierarchy und das "Hub and Spoke". Das Edge Distribution eignet sich für kleine CDNs. Dabei wird der Content von einem zentralen Speichersystem auf die Edge Devices oder auf Point of Presence (POP) übertragen und dort für den schnelleren Zugriff abgelegt. Bei der Edge Hierarchy, das für größere CDN-Netzkonfigurationen geeignet ist, werden dem zentralen Speichersystem Hub-Caches nachgeschaltet, die die Verteilung des Contents übernehmen. Und bei der CDN-Konfiguration Hub and Spoke, das in global tätigen Unternehmen eingesetzt werden kann, werden die Inhalte des zentralen Speichersystems auf beliebig viele dezentrale Speichersysteme ausgelagert. Der Content wird von den Hub-Systemen in Cache-Systemen gespeichert, die als Spoke bezeichnet werden.

CDNs müssen den Content für alle Arten an Endgeräten bereitstellen. Das können mobile Endgeräte sein wie Handys, Handhelds oder PDAs, aber ebenso Computer oder Personal Computer.

CGI (common gateway interface)

Das Common Gateway Interface (CGI) ist ein standardisiertes, plattformunabhängiges Interface, dass den Informationsaustausch zwischen Webservern oder HTTP-Servern und einem externen Programm regelt. So kann das CGI-Interface die Interaktion zwischen dem Anforderungsformular vom Web-Browser mit dem Datenbankprogramm des Servers definieren. Das CGI die Ausführung externer Programme von WWW-Servern unterstützt, wird es zur Erstellung von dynamischen Webseiten genutzt. Über das Common Gateway Interface können andere Systeme, wie Datenbanken, mit HTTP-Servern kommunizieren.

CGI-Programme müssen für den Datenaustausch den CGI-Spezifikationen entsprechen, wobei die Programme dank der Plattform-Unabhängigkeit in beliebigen Programmiersprachen wie "C", Perl oder Java, geschrieben werden können. Benutzereingaben, wie beispielsweise der Klick auf einen Hyperlink, werden über den Browser abgeschickt und vom Webserver mittels CGI an das auszuführende Programm weitergeleitet. Nach der Bearbeitung gibt das Programm die Ergebnisdaten mittels CGI an den Server zurück.

Eine schnellere Variante von Common Gateway Interface (CGI) ist FastCGI. *Siehe auch*: *FCGI (S. G21)*.

CIDR (classless interdomain routing)

Die Klasseneinteilung der 32 Bit umfassenden IP-Adressen ist wenig effizient. Das starre Adressenschema der klassischen IP-Adresse in Netzwerk- und Hostteil verhindert eine flexible Anpassung und schränkt den gesamten Adressraum wesentlich ein. Da die Anzahl an IP-Adressen begrenzt ist, hat man bei der IPv4-Adresse das Verfahren des Classless Interdomain Routing (CIDR) eingeführt, das den zur Verfügung stehenden 32 Bit umfassenden Adressraum effizienter ausnutzt. Dieses CIDR-Verfahren wird auch in der IPv6-Adresse verwendet.
Das CIDR-Verfahren löst die starren IP-Adressstrukturen und basiert auf Subnetzmasken. Die Subnetzmaske teilt die IP-Adresse in den Netzwerkteil und den Hostteil auf. Zwischen ihr und der IPv4-Adresse besteht ein unmittelbarer mathematischer Zusammenhang aus der Adressbereich bestimmt werden kann.

IPv4-Adresse: 168.122.34.3/24	Dezimal	Binär
IP-Adresse	168.122.34.3	10101000.01111010.00100010.00000011
Netzmaske	255.255.255.0	11111111.11111111.11111111.00000000
Netzwerkadresse*	168.122.34.0	10101000.01111010.00100010.00000000
Hostadressen*	3	00000000.00000000.00000000.00000011

* Netzwerkadresse durch ADD-Funktion von IP-Adresse und Netzmaske
* Hostadresse durch ADD-Funkion von IP-Adresse und negierter Netzmaske

In der Notation werden diese Subnetzmasken durch einen Suffix angegeben, der mit einem Schrägstrich an die IP-Adresse angehängt wird. Das Suffix bestimmt die Anzahl der 1-Bit-Werte in der IP-Adresse, beginnend beim ersten Bit im ersten Byte. Ist der Suffix /1, beginnt das erste Byte der Subnetzmaske mit einer 1. In binärer Schreibweise würde die Subnetzmaske so aussehen: 10000000 00000000 00000000 00000000. In der Dotted Decimal Notation stellt sich die Subnetzmaske so dar: 128.0.0.0. Class-A-Adressen haben den Suffix /8 und entsprechen dezimal 255.0.0.0. Class-B-Adressen haben /16 und entsprechen 255.255.0.0. usw. Das Suffix kann Werte zwischen /0 und /32 annehmen. Mit dem CIDR-Schema können insgesamt 4.294.967.296 (32 Bit) Adressen abgedeckt werden. Eine CIDR-Notation kann folgendermaßen aussehen: 168.122.12.3/24, wobei die IP-Adresse 168.122.12.3 ist und die Subnetzmaske 255.255.255.0.

CIDR-Zonen	C-Adressräume
Europa	194.0.0.0 bis 195.255.255.255
Nordamerika	198.0.0.0 bis 199.255.255.255
Mittel-, Südamerika	200.0.0.0 bis 201.255.255.255
Asien, paz. Raum	202.0.0.0 bis 203.255.255.255
Reserve	204.0.0.0 bis 223.255.255.255

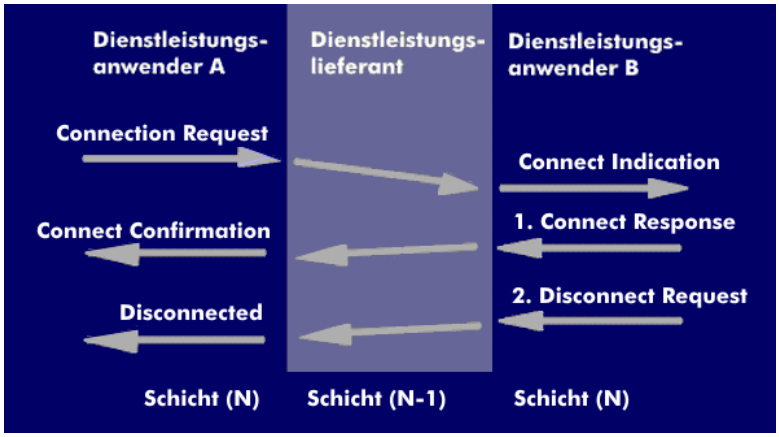
Benötigt eine Domäne eine bestimmte Anzahl an Adressen, dann teilt ihr CIDR die entsprechende Anzahl an Blöcken zu. Bei 1000 benötigten Adressen wären das vier Blöcke. Um die weltweite Nutzung der C-Adressräume zu sichern, wurden die weltweiten Adressbereiche in vier Zonen eingeteilt, wodurch jede Region 32 Millionen Adressen erhält und weitere 320 Millionen in Reserve gehalten werden können.
Das CIDR-Verfahren reduziert die in Routern gespeicherten Routingtabellen. Mit diesem Suffix kann ein großer Internet Service Provider (ISP) bzw. ein Betreiber eines großen Teils des Internets gekennzeichnet werden. Dadurch können auch darunter liegende Netze zusammengefasst werden; sogenanntes Supernetting. Die Methode wird u.a. im Border Gateway Protocol (BGP) eingesetzt und ist in RFC 1518 beschrieben.

Client-Server-Paradigma

Beim **Client-Server-Paradigma** geht es um die typische Verbindung zwischen einem Serverprozess und einem Clientprozess, welche beide gleichzeitig aktiv sein, und eine Verbindung aufgebaut haben müssen um Nachrichten auszutauschen. Dabei ist meist ein *leistungsstarker* Server *dauerhaft aktiv* und von - *nicht untereinander kommunizierenden* - Clients erreichbar. Die Clients sind dabei aber nur *manchmal verbunden*.
Das Client-Server-Paradigma ist eine typische zentralisierte Architektur mit Zentrum beim Server.

CONF (confirmation)

Die Bestätigung, Confirmation (CONF), gehört zu den Dienstelementen, auch Dienstprimitive genannt, der Schichten des OSI-Referenzmodells. Sie dient dem Dienstleitungsanbieter als Bestätigung für einen vorherigen Request.



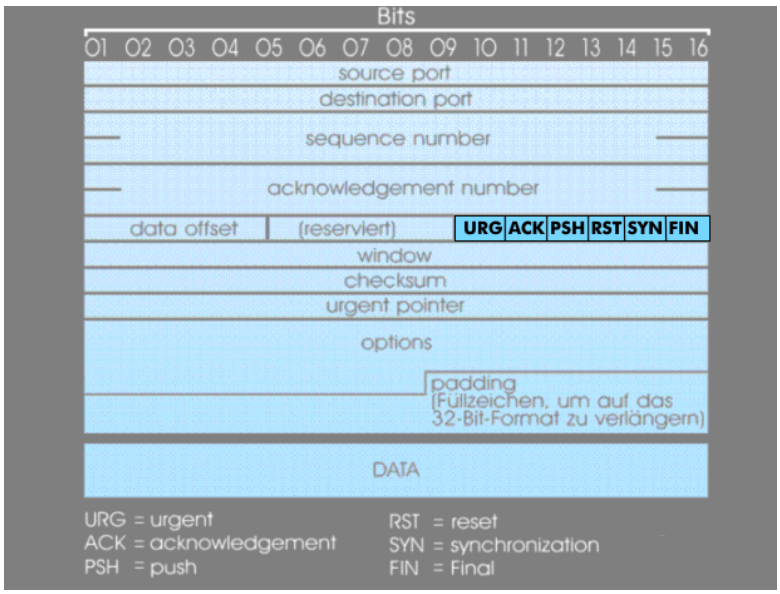
Bei der Kommunikation sendet ein Dienstleistungsanwender eine Anfrage (Request) über den Dienstleistungslieferanten an den Kommunikationspartner. Dieser beantwortet die Anfrage mit einer Antwort (Response), die über den Dienstleistungslieferanten als Bestätigung (Confirmation) dem anfragenden Dienstleistungsteilnehmer zugestellt wird.

continuous ARQ

Continuous ARQ ist eine Variante der Übertragungssicherung nach Automatic Repeat Request (ARQ). Bei diesem Verfahren handelt es sich um ein Kreditverfahren, dabei wird dem Data Communication Equipment (DCE) ein Kredit über die Fenstergröße eine bestimmte Anzahl von Datenblöcken eingeräumt, die er unquittiert senden kann. Durch diese Maßnahme erhöht sich der Datendurchsatz gegenüber Stop-and-Wait ARQ mit Einzelblockquittierung. Beispiele für Continuous ARQ sind High Level Data Link Control (HDLC) nach X.25 und Synchronous Data Link Control (SDLC).

Control-Flag-Field

Das Control-Flag-Feld ist ein Datenfeld im TCP-Header, das dem Datenfeld für die Header-Länge folgt. Das Control-Flag-Feld enthält sechs Bit-Indikatoren, die den Verbindungszustand steuern, also den Verbindungsaufbau, die Aufrechterhaltung der Verbindung und den Verbindungsabbau. Darüber hinaus behandeln die Flags die Abarbeitung von Daten und deren Dringlichkeiten.



Das Control-Flag-Feld umfasst sechs Flags: Urgent-Flag (URG), Bestätigungs-Flag (ACK), Push-Flag (PSH), Reset-Flag (RST), Synchronisations-Flag (SYN) und Final-Flag (FIN).

Cross-Layer-Optimierung

Ein reines Schichtenmodell wie es beispielsweise OSI beschreibt ist in der Praxis aufgrund von Effizienz nicht strikt umgesetzt. Stattdessen werden manche Dienste und Mechanismen verschiedener Schichten *über diese* hinweg *optimiert*, man spricht hierbei von einer *Cross-Layer-Optimierung*. *Siehe auch*: *OSI (S. G41) und TCP/IP (S. G80)*.

Darstellungsschicht (presentation layer) (OSI)

Aufgabe der Darstellungsschicht - sie bildet Schicht 6 im OSI-Referenzmodell - ist die Codierung und Darstellung der Informationen, die zwischen offenen Systemen ausgetauscht werden.

Instanzen der Anwendungsschicht vereinbaren zunächst, wie die Daten, die ausgetauscht werden sollen, zu strukturieren sind und welche Datentypen und -werte benutzt werden. Diese Vereinbarung in Form eines Befehlssatzes wird abstrakte Transfersyntax genannt. Eine entsprechende Beschreibungssprache ist Abstract Syntax Notation One, ASN.1, ein Normenvorschlag der ISO. Die Darstellungsschicht hat die Aufgabe, die Dateieinheiten unter Erhaltung ihres Informationsgehalts zu übertragen. Die Instanzen der Darstellungsschicht treffen Vereinbarungen über eine konkrete Transfersyntax. Die Zuordnung zwischen abstrakter und konkreter Transfersyntax wird als Darstellungskontext bezeichnet. Beispiele für realisierte Protokolle der Darstellungsschicht sind die Dokumentarchitektur T.73 von CCITT und EHKP-6 für Bildschirmtext. *Siehe auch*: *OSI (S. G41)*.

DDNS (dynamic domain name system)

Normalerweise werden IP-Adressen einem Rechner fest zugeordnet. Bei der dynamischen Adressvergabe wechselt die IP-Adresse allerdings bei jeder neuen Sitzung. Die DynDNS ist ein DNS-Netzwerkdienst bei dem sich die IP-Adressen in Echtzeit ändern.

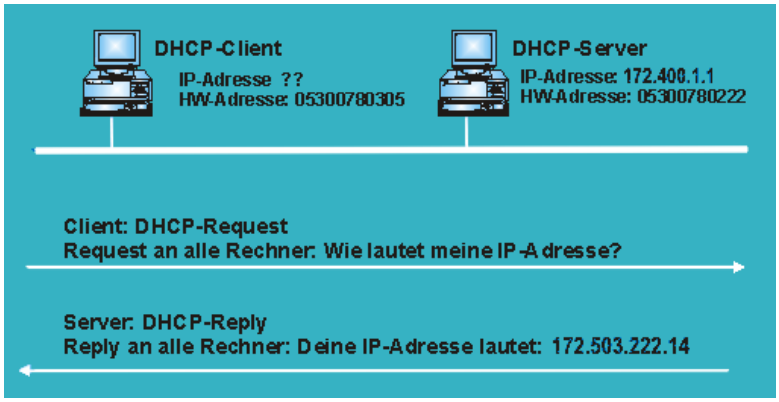
Bei der Verbindung eines Rechners mit dem Internet muss die IP-Adresse der Gegenseite bekannt sein, damit diese eine Verbindung zum Zielrechner aufbauen kann. Ändert sich aber IP-Adresse wie beim Dynamic Domain Name System (DDNS), dann muss der Nameserver des DDNS-Providers über die ständig wechselnden IP-Adressen informiert werden und diese verwalten. Die wechselnden IP-Adressen für den lokalen Rechner werden von einem Client-Programm generiert und dem Nameserver des DDNS-Providers mitgeteilt, der daraufhin eine Verbindung zur neuen IP-Adresse aufbauen kann.

Die dynamische Adressänderung hat den Vorteil, dass sie sehr flexibel ist und Netzwerkressourcen besser nutzen kann. Sie wird dann sinnvoll eingesetzt, wenn der Anschluss über verschiedene Netze und Komponenten erfolgt oder wenn die Rechner vom

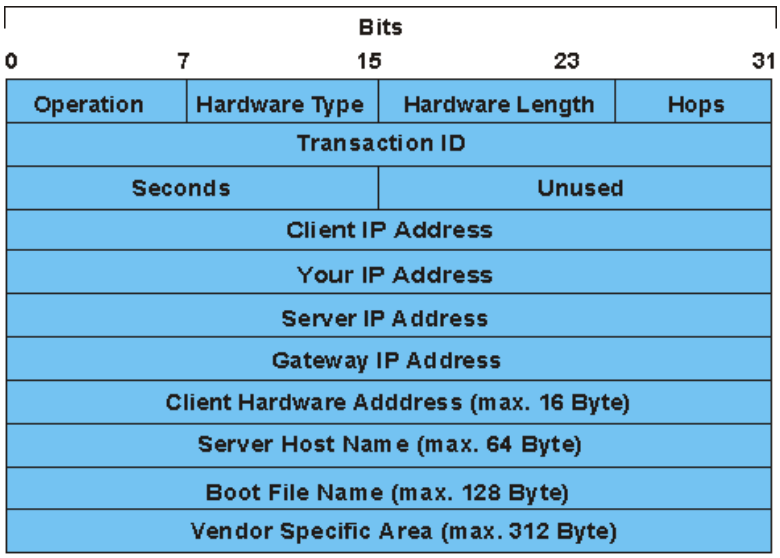
Netz abgeschaltet werden. Angewendet wird das DynDNS-Verfahren u.a. in DSL-Routern und bei der Internettelefonie. *Siehe auch*: *DNS (S. G15)*.

DHCP (dynamic host configuration protocol)

Das DHCP-Protokoll ist ein Client-Server-Protokoll, das den Aufwand für die Vergabe von IP-Adressen und sonstigen Parametern reduziert. Mittels Dynamic Host Configuration Protocol (DHCP) kann ein Netzwerkadministrator alle TCP/IP-Konfigurations-Parameter zentral verwalten und warten.
Das DHCP-Protokoll dient der dynamischen und automatischen Endgeräte-Konfiguration z.B. der Vergabe von IP-Adressen unter IPv4 und IPv6. Die entsprechenden IP-Adressen werden von den angeschlossenen DHCP-Clients beim DHCP-Server angefordert. Die Adressen werden einem Adresspool entnommen, der in einem DHCP-Server residiert.
Die Zuweisung der IP-Adresse kann automatisch, dynamisch oder manuell erfolgen. Bei der automatischen Zuweisung wird dem Client bei der ersten Anmeldung am Netz eine freie Adresse mitgeteilt. Bei dieser Art der Adress-Zuweisung kann jede Adresse nur einmal vergeben und nicht für andere Clients genutzt werden. Dies ist bei der dynamischen Zuweisung anders. Bei dieser Zuweisungsart wird die Adresse temporär für eine bestimmte Zeit vergeben. Wird die Adresse vom Client nicht mehr benötigt, kann der Server wieder über sie verfügen und sie an einen anderen Client vergeben. Über die manuelle Adressenvergabe kann der Netzadministrator einem Client eine Adresse zuweisen.



Da das DHCP-Protokoll mehrere Kommandos des Bootstrap-Protokolls (BootP) benutzt und die im Bootstrap-Protokoll definierten BootP-Relay-Agents unterstützt, kann das DHCP-Protokoll als Erweiterung des älteren, statischen Bootstrap-Protokolls angesehen werden, das durch DHCP abgelöst wird. DHCP gilt als sicherer und leichter zu handhaben.
Das DHCP-Protokoll kennt mehrere Nachrichtentypen, mit denen der gesamte Informationsaustausch zwischen Client und Server gesteuert wird. Die Anfrage des Clients erfolgt mittels eines Broadcast (DHCP Discover), um das Netz auf verschiedene DHCP-Server hin zu testen. Als Antwort verschickt der DHCP-Server eine Broadcast- oder Unicast-Nachricht (DHCP Offer), in der dem Client eine Konfiguration unterbreitet wird. Akzeptiert der DHCP-Client die angebotenen Konfigurationsparameter, sendet er mittels Broadcast einen DHCP-Request. Darauf hin sendet der Server mittels DHCP-ACK die Konfigurationsparameter und die IP-Adresse. Lehnt ein DHCP-Server eine Anfrage bezüglich bestimmter Konfigurationsparameter ab, sendet er ein NAK-Signal an den Client. Wird die Netzwerkadresse vom Client nicht mehr benötigt, sendet dieser ein DHCP Release an den Server.



Bestimmte Nachrichten müssen sowohl Server- als auch Client-seitig als Broadcast verschickt werde, damit alle anderen Clients und Server über die Konfiguration informiert sind und nicht unnötig Adressen und Konfigurationen reservieren.
Der Aufbau des DHCP-Frame-Formats ist identisch mit dem des BootP-Datenrahmens bis auf das Flag- und Optionsfeld, die bei BootP nicht oder anders benutzt werden. Das DHCP-Protokoll ist in den RFCs 1541, 1542 und 2131 beschrieben.

Dienst (Service)

Ein Dienst ist in der OSI-Terminologie eine Funktionssammlung einer Schicht, die diese einer übergeordneten Schicht am sogenannten Dienstzugangspunkt anbietet. Ein Dienst wird immer der direkt übergeordneten Schicht angeboten. Die Dienste der einzelnen Schichten werden von den unterschiedlichen Aufgaben dieser Schichten geprägt. So gibt es auf der Netzwerkebene die Netzwerkdienste, auf der Transportschicht die Transportdienste und auf der Anwendungsschicht die Anwendungsdienste. Sie ist insofern ähnlich der öffentlichen Schnittstelle einer Softwarekomponente.

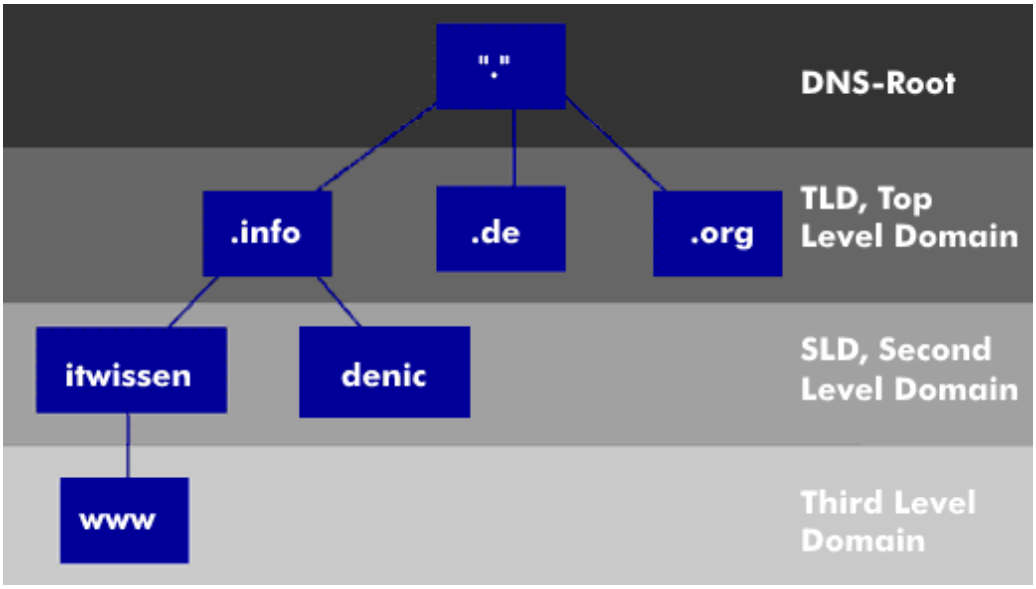
DMA (direct memory access)

Direct Memory Access (DMA) ist ein Protokoll für den direkten Speicherzugriff. Beim DMA-Protokoll erfolgt der Informationsaustausch zwischen Arbeitsspeicher und Massenspeicher ohne die Zentraleinheit (CPU) in Anspruch zu nehmen. DMA hat eine eigene Steuerlogik, den DMA-Controller, für die Steuerung des Systembusses, und eignet sich besonders für den schnellen Datentransfer von großen Datenmengen.
Beim DMA-Transfer wird die DMA-Schnittstelle von der Zentraleinheit initialisiert, mit einem DMA-Request an die CPU erfolgt die Bus-Freigabe. Danach nutzt die DMA-Schnittstelle die Busse in gleicher Art und Weise wie die Zentraleinheit. DMA kennt vier Betriebsarten: den Byte Mode mit der byteweisen Übertragung, den Burst Mode, den Halt Mode, bei dem die CPU bis zur Beendigung des DMA-Transfers angehalten wird, und den Transparent Mode, bei dem die DMA-Schnittstelle und die Zentraleinheit zeitmultiplex arbeiten.
Beim ISA-Bus können bis zu 16 MB des Arbeitsspeichers für Direct Memory Access (DMA) adressiert werden. Beim EISA-Bus und bei der Micro-Channel-Architektur ist der gesamte Speicherbereich adressierbar.
Eine Weiterentwicklung von DMA für ATA/ IDE ist das Ultra-DMA, das Datentransferraten von 33 MB/s und 66 MB/s unterstützt, ebenso wie die PIO-Modi 1, 3 und 4. Eine Alternative zu DMA sind das Programmed Input Output (PIO), bei dem allerdings alle übertragenen Daten über den Prozessor laufen, und Remote DMA (RDMA).

DNS (domain name system)

Ein Domain Name System (DNS) ist ein Namensdienst, ein Online verteiltes Datenbanksystem, in dem der Domainname in die IP-Adresse - IPv4-Adresse oder IPv6-

Adresse - umgesetzt wird. So wird beispielsweise der Domainnamen itwissen.info in die IPv4-Adresse 213.133.101.238 umgesetzt. Wird umgekehrt aus der IP-Adresse der Domainnamen abgeleitet, dann spricht man von Reverse DNS (rDNS).
Im Domain-Name-System sind die Domainnamen hierarchisch in einer Baumstruktur gegliedert, die Domain-Name-Server kennen immer nur die Domain-Adressen der hierarchisch höheren und niedrigeren Nameserver innerhalb einer Domäne. Die einzelnen Hierarchieebenen repräsentieren einzelne Namensteile und sind im Domainnamen durch Punkte voneinander getrennt. Bei itwissen.info sind es die zwei Hierarchieebenen ïnfoünd ïtwissen", die eine als Top-Level-Domain (TLD), die andere als Second-Level-Domain (SLD).



Die DNS-Server und Nameserver bieten einen hierarchisch geordneten Namensraum, damit Unternehmen, Institutionen, Behörden usw. ihre Domain-Namen selbst bestimmen können. DNS unterstützt auch verschiedene Verzeichnislisten zwischen der elektronischen Post (E-Mail) und IP-Adressen. Die Namensdarstellung umfasst verschiedene Domain-Ebenen. Die erste Ebene stellt die DNS-Root dar, die zweite Ebene ist die der Top-Level-Domain (TDL) und repräsentiert neben den geografischen Zuordnungen, bezeichnet als Country Code Top Level Domain (ccTLD), auch Organisationsformen. Die TLD wird als Generic Top Level Domain (gTLD) bezeichnet. Länder-Domänen sind beispielsweise "de"für Deutschland, "fr"für Frankreich und "ca"für Kanada.
Organisationen und deren gTLD-Domainen sind beispielsweise: "com"für kommerzielle Organisationen, ïnt"für internationale Organisationen, ëdu"für Bildungseinrichtungen, "gov"für US-amerikanische Regierungsorganisationen, mit "mil"für US-Militärorganisationen und mit örg"für andere nicht profitorientierte Organisationen. Die folgende Hierarchieebene ist die Second Level Domain (SLD), die unter der Top Level Domain nur einmal vorkommen darf, gefolgt von der Third-Level-Domain. Die Second-Level-Domain steht in der Regel für den Dienst, was allerdings nicht zwingend ist. So beispielsweise www.
Da das Domain Name System die Informationen ohne Authentifizierung über das UDP-Protokoll überträgt, hat die Internet Engineering Task Force (IETF) das gesicherte Domain Name System Security Extension (DNSsec) entwickelt. Neben der fest zugeordneten Internet-Adresse gibt es noch die dynamisch zugeordnete. Man spricht dann von dynamischer DNS oder DynDNS oder Dynamic Domain Name System (DDNS). Eine der bekanntesten und am häufigsten eingesetzten Implementierungen des Domain Name System (DNS) ist Berkeley Internet Name Domain (BIND).

DNSsec (domain name system security extension)

Beim Domain Name System (DNS) findet die Kommunikation zwischen dem Nameserver und dem DNS-Client über das verbindungslos arbeitende User Datagram Protocol (UDP)

statt. Dieses verbindungslos arbeitende Transportprotokoll sieht keine Authentifizierung der Nachrichtenquelle vor. Dadurch kann die Identität des Absenders vom Empfänger nicht überprüft werden. Es kann also nicht sichergestellt werden, ob die Nachricht tatsächlich von dem entsprechenden DNS-Server stammt.
Wird das Datenfeld für den Eintrag der Absenderadresse manipuliert und eine andere IP-Adresse eingetragen, gefährdet dies alle Internet-Anwendungen. Um entsprechende Manipulationen auszuschließen, hat die Internet Engineering Task Force (IETF) das Domain Name System Security Extension (DNSsec) entwickelt. Es handelt sich dabei um eine Protokollerweiterung des Domain Name System, die mit dem Public-Key-Verfahren arbeitet und die Nachrichtenquelle authentifiziert. Damit ist sichergestellt, dass die Antwort des Nameservers den Angaben entspricht, die ihr zugeordnet sind. Die Entwicklung von DNSsec reicht zurück in das Jahr 1994 und ist verankert in der RFC 2535. Darüber hinaus gibt es diverse RFCs, die sich mit DNSsec befassen. Die 2005 vorgestellte Version fasst die diversen RFCs unter der Bezeichnung DNSsec-bis zusammen.
DNSsec arbeitet mit kryptografischen Domainnamen und zielt auf die Bereiche Schlüsselverteilung, Authentifizierung der Ursprungsdaten und Transaktion der Authentifizierung ab. Bei Anfragen an den DNSsec-Server sendet dieser einen umfangreicheren DNS-Record, der mit einem Private Key unterzeichnet ist. Der Anfragende kann damit den Response auf Echtheit und Unverfälschtheit überprüfen. Die Antwort erhält außerdem ein Zertifikat mit dem der Empfänger auch den Absender und damit den Informationsinhalt verifizieren kann. Die digitale Signatur der Datenpakete basiert auf einer Hashfunktion, die vom Empfänger erzeugt wird und mit der Signatur verglichen werden kann.

drahtloses Übertragungsmedium

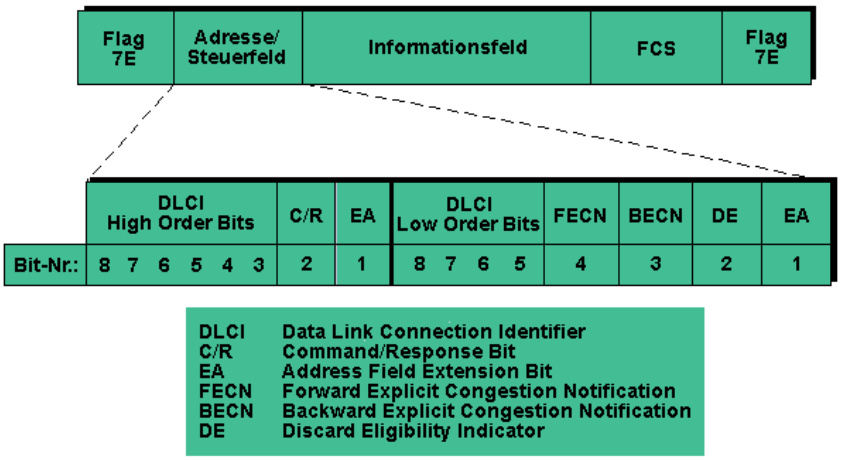
Drahtlose Übertragungsmedien sind beispielsweise Schall, Funktechnik, Infrarot und sichtbares Licht. Dabei treten hohe Bitfehlerraten wegen verschiedener Probleme mit der Ausbreitung von Funkwellen in der Größenordnung von 10^{-5} bis 10^{-2} auf, außerdem treten diese of in Schüben (*bursts*) auf. Beispiele für diese Technik sind die Schallübertragung, IR-LANs und die komplette Mobil- und Satellitenkommunikation, WLANs, Bluetooth oder RFID. *Siehe auch*: *Übertragungsmedium (S. G83)*.

Duplex-Betrieb (Übertragungsart)

Der (Voll-)Duplex-Betrieb ist eine Übertragungsart, bei der zwei gleichberechtigte Datenstationen gleichzeitig über eine Datenverbindung senden und empfangen können. Die Duplex-Übertragung wird deswegen auch als Gegenbetrieb bezeichnet.

ECN (explicit congestion notification)

Explicit Congestion Notification (ECN) ist ein Mechanismus im IP-Protokoll und in Frame-Relay (FR), der dann zum Einsatz kommt, wenn das Netzwerk überlastet ist. Der Mechanismus benutzt zwei Bit-Werte zur Anzeige der Überlastung, diese befinden sich im IP-Header im DSCP-Feld, Differential Service Code Point. Das eine Bit ist die Backward Explicit Congestion Notification (BECN), das andere die Forward Explicit Congestion Notification (FECN).



Im Frame-Relay-Header wird die Backward Explicit Congestion Notification (BECN) in der der Überlast entgegengesetzten Richtung übertragen und soll den sendenden Knoten darüber informieren, dass die Übertragungsstrecke überlastet ist. Die Forward Explicit Congestion Notification (FECN) wird in den Frame-Relay-Header eingefügt und in der Richtung übertragen, in der die Überlast auftritt. Der FECN soll den Empfangsknoten über die Überlast informieren.

Der Überlastmechanismus wird durch das Setzen eines Bits im Header ausgelöst.

EGP (exterior gateway protocol)

Das Exterior Gateway Protocol (EGP) ist auf der Vermittlungsschicht des OSI-Referenzmodells angesiedelt und baut auf dem IP-Protokoll auf. Das EGP-Protokoll wird zur Kommunikation zwischen Routern benutzt und dient dem Verbund mehrerer komplexer Netze, die in sich eine abgeschlossene Welt bilden und nur gelegentlich mit anderen Netzen kommunizieren.

Ein solches Netz wird in TCP/IP-Terminologie als autonomes System (AS) bezeichnet. Es bildet mit anderen autonomen Systemen im Verbund ein „Netz von Netzen“. In jedem autonomen System des Netzwerkverbundes wird nun mindestens ein Edge Router (ER) als Exterior-Gateway eingerichtet, der das autonome System mit den anderen autonomen Systemen verbindet.

Das Exterior-Gateway-Protokoll beruht im Wesentlichen auf drei Mechanismen:

1. Es unterstützt die sogenannte Nachbar-Akquisition, d.h., es gibt einen Mechanismus durch den ein Edge-Router andere Edge-Router benachbarter, autonomer Systeme kennenlernt und mit ihnen Routing-Informationen auf der Basis des EGP austauscht. Dabei handelt es sich um Router-Router Multicast zwischen Exterior Gateways.
2. EGP-Nachbarn testen in Intervallen, ob ihr Nachbar immer noch existiert (d.h., auf Anfrage antwortet). Dies verhindert, dass in endloser Folge Datenpakete über ein autonomes System geleitet werden, das in Wirklichkeit gar nicht mehr erreichbar ist.
3. EGP-Nachbarn tauschen in Intervallen Informationen darüber aus, welche Netze sie erreichen können, und aktualisieren so ihre Routingtabellen entsprechend der gerade aktiven Topologie.

Das Exterior-Gateway-Protokoll erkennt einen Nachbar-Gateway und dessen Aktivierung; es kann EGP-Nachbar-Gateways testen, ob sie antworten, und periodisch „Routing Update Messages“ zwischen EGP-Nachbar-Gateways austauschen.

Der EGP-Header besteht aus dem 8 Bit langen Versionsfeld, den gleichlangen Datenfeldern EGP-Type, Code-Feld und Statusfeld sowie aus vier Feldern mit 16 Bit Länge: dem Prüfsummenfeld, dem Datenfeld Autonomes-System, dem Sequenznummernfeld und einem reservierten Datenfeld. Das EGP-Version-Feld gibt die verwendete Version des EGP-Headers

an. Das EGP-Type-Feld definiert, um welchen Meldungstyp es sich handelt, und ermöglicht dem Router, bei der Übertragung über Netze die Art der EGP-Message anzugeben.
In dem EGP-Code-Feld wird die Art des AGP-Typs genauer spezifiziert. Es wird angegeben, ob es sich um ein Kommando oder um eine Antwort auf ein Kommando handelt. Im Statusfeld werden die Abruf- und Bestätigungsmodi behandelt. Beschrieben ist das EGP-Protokoll in RFC 904.

ESP (encapsulating security payload)

Der Encapsulating Security Payload (ESP) dient der Verschlüsselung von IP-Datenpaketen und benutzt ebenso wie der Authentication Header (AH) als Algorithmus den Hash-Based Message Authentication Code (HMAC). Der ESP-Header verwendet kryptografische Verfahren wie den Data Encryption Standard (DES) und verschlüsselt alle Daten komplett, die von einem ESP-Header und einem ESP-Trailer eingeschlossen werden. Am Ende eines Paketes kann optional ein ESP-Authentifikationsblock für zusätzliche Authentizität sorgen.
Der Encapsulating Security Payload, der in RFC 2406 spezifiziert ist, authentifiziert im Transportmodus nur den IP-Inhalt, nicht aber den IP-Header. Dieser Modus wird vor allem innerhalb eines sicheren Netzwerks eingesetzt.
Im Tunnelmodus wird dagegen der IP-Header verschlüsselt, um interne Adressinformationen vor unberechtigtem Zugriff zu schützen. Dieser Modus ist im IPsec-Framework für das sichere Tunneling zwischen zwei Firewalls bei Virtual Private Networks (VPN) vorgeschrieben.

Ethernet

Ethernet ist aus einem Projekt der Unternehmen Digital Equipment, Intel und Xerox in den siebziger Jahren hervorgegangen, das unter der Bezeichnung DIX bekannt wurde. Dieses Projekt zielte auf die gemeinsame Nutzung eines Übertragungsmediums durch mehrere gleichberechtigte Datenstationen und war für den lokalen Bereich konzipiert. Die Struktur dieses lokalen Netzes war die eines Busses an den alle Datenstationen angeschlossen werden konnten. Das Buskonzept war in seiner Ausdehnung, in der Anzahl der anschließbaren Stationen und in Datenrate auf 3 Mbit/s begrenzt und daher nur für den lokalen Bereich geeignet. Aus dem DIX-Projekt wurde 1982 Ethernet. Die Spezifizierung und Standardisierung übernahm das IEEE, das die Spezifikationen DIX Ethernet V2.0 veröffentlichte.
Das von der IEEE-Arbeitsgruppe 802.3 spezifizierte Ethernet ist ein lokales Netz an dessen Übertragungsmedium gleichberechtigte Datenstationen angeschlossen werden. Die Datenstationen arbeiten mit einem stochastischen Zugangsverfahren bei dem sie Signale auf dem Übertragungsmedium erkennen und dann, wenn kein anderes Signal vorhanden ist, es für die eigene Datenübertragung nutzen. Das Zugangsverfahren ist CSMA/CD und kann zu Kollisionen zwischen den übertragenen Datensignalen führen.
Das Ur-Ethernet nach 10Base-5 arbeitet mit einer Übertragungsgeschwindigkeit von 10 Mbit/s. Alle Datenstationen sind gleichberechtigt und werden direkt an das Übertragungsmedium angeschlossen. Die Anzahl an Datenstationen ist auf max. 256 Stationen pro Ethernet-Segment begrenzt, die Längenausdehnung wird durch die Signallaufzeit und die Framelänge eingeschränkt. Mit zunehmender Segmentlänge wird die Laufzeit zwischen den beiden am weitesten voneinander entfernten Stationen größer, was sich negativ auf die Zeit auswirkt, die zum Erkennen einer Kollision benötigt wird. Alle drei Parameter, die Anzahl der Stationen, die Framelänge und die Segmentlänge, bestimmen die Performance von Ethernet. Je kürzer die Framelänge und je höher die Anzahl der sendebereiten Stationen, desto geringer wird die Performance. Die Framelänge des Ethernet-Frames beeinträchtigt die Leistungsfähigkeit insofern, als dass das Verhältnis von Nutzdaten zur Framelänge bei niedriger werdender Rahmengröße immer geringer wird.
Die zulässige Netzausdehnung eines Ethernet ist abhängig von der Ethernet-Variante und damit von der Übertragungsgeschwindigkeit auf dem Netz. Generell verringert sich die Ausdehnung mit steigender Übertragungsgeschwindigkeit. Betrachtet man das

klassische Ethernet mit 10-Mbit/s-Übertragungsgeschwindigkeit, dann ermittelt sich die zulässige Netzausdehnung aus der Umlaufverzögerung auf dem Übertragungsmedium. Diese wird durch die Laufzeit zwischen den beiden entferntesten Stationen bestimmt. Bei einer kabelspezifischen Signallaufzeit von 0,77c ergibt sich für ein 500-m-Segment eine Laufzeitverzögerung von 2,165 µs. Ausgehend von der minimalen Paketlänge eines Ethernet-Frames von 64 Byte und der Übertragungsrate von 10 Mbit/s, ergibt sich als Hin- und Rücklaufzeit zwischen den beiden entferntesten Stationen eine Zeit von 51,2 µs, die einfache Laufzeit beträgt also 25,6 µs. Diese Umlaufverzögerung entspricht einer Ausdehnung von etwa 5 km. Unter Berücksichtigung der Signallaufzeiten in den Transceiver-Kabeln und der Verzögerungszeiten der aktiven Komponenten sind damit Ethernet-Ausdehnungen von ca. 3.000 m möglich. *Siehe auch*: *LAN (S. G46)*.

Europäische Institut für Telekommunikationsnormen (ETSI)

Das ETSI ist eine der drei großen Normungsorganisationen in Europa. ETSI ist zuständig für die europäische Normung im Bereich Telekommunikation. Zusammen mit dem Europäischen Komitee für elektrotechnische Normung (CENELEC) und Europäischen Komitee für Normung (CEN) bildet ETSI das europäische System für technische Normen.

FC-Dienstklasse

Um eine möglichst effiziente Datenübertragung für die unterschiedlichen Verkehrsarten zu gewährleisten, unterscheidet Fibre-Channel (FC) sechs verschiedene Service-Klassen für die Datenübertragung zwischen Endgeräten, von denen aber nur einige in Produkten implementiert sind.

Bei der Dienstklasse **Class 1** (Dedicated Service) handelt es sich um einen verbindungsorientierten, quittungslosen Dienst zwischen zwei Node-Ports. Die Class-1-Verbindung wird dediziert zwischen zwei Ports festgelegt und stellt dem Benutzer die volle Bandbreite zur Verfügung. Die Übertragung erfolgt quittungslos. Die Verbindung kann nicht für andere Zwecke genutzt werden.

Class 2 (Multiplex Service) ist ein verbindungsloser Dienst, der mit Datagrammen arbeitet und die Frames von einem oder von mehreren N-Ports oder NL-Ports multiplext. Dieser Dienst arbeitet mit Quittungsmechanismen und Ende-zu-Ende-Flusskontrolle.

Class 3 (Datagram Service) ist vergleichbar Class 2, arbeitet allerdings quittungslos nach dem Best-Effort-Prinzip mit Link-Flusskontrolle.

Class 4 (Virtual Circuit Fractional Bandwidth) ist ein verbindungsorientierter Dienst mit Bandbreitenzuteilung. Für die Übertragung wird nur ein Teil der zur Verfügung stehenden Bandbreite genutzt. Die restliche Bandbreite kann für andere Zwecke genutzt werden. Der Class-4-Dienst benötigt eine Maximum an Zeit um die Datenpakete zwischen Sender und Empfänger zu übertragen.

Class 6 ist ein verbindungsorientierter Dienst für Multicast-Services. Diese Dienstklasse unterstützt unidirektionale Multicast-Verbindungen.

Service-Klasse	Service-Merkmale
Class 1	Verbindungsorientierter Dienst ohne Quittung.
Class 2	Verbindungsloser Dienst mit Quittung.
Class 3	Verbindungsloser Dienst ohne Quittung zwischen den N-Ports.
Class 4	Verbindungsorientierter Dienst mit Bandbreitenzuordnung.
Class 6	Verbindungsorientierter Dienst für Multicast-Services.
Class F	Verbindungsloser Dienst zwischen E_Ports zur Steuerung und Konfiguration der Fabric.

FCAPS (fault, configuration, account, performance, and security management)

Fault, Configuration, Account, Performance, and Security (FCAPS) steht für das ISO-Netzwerkmanagement von Fernmelde-Kontrollnetzen (TMN). FCAPS bildet dabei das funktionale Modell, das die internationale Fernmeldeunion (ITU) unter M.3010 standardisiert hat.

Die FCAPS-Struktur ist in eine logische Architektur, Logical Layer Architecture (LLA), bestehend aus fünf den Managementschichten: Network Element Layer (NEL), Element Management Layer (EML), Network Management Layer (NML), Service Management Layer (SML) und Business Management Layer (BML).

FCAPS bietet Mehrwertdienste, die zur Verbesserung der Kommunikations- und Informationsflüsse beitragen. Diese von Dienstleistern erbrachten Leistungen umfassen das Fehlermanagement (Fault), Konfigurationsmanagement (Configuration), Abrechnungsmanagement (Accounting), Leistungsmanagement (Performance) und das Sicherheitsmanagement (Security).

Das FCAPS-Verfahren wird u.a. im Cloud-Computing für das Application Performance Management (APM) bei virtualisierten Anwendungen eingesetzt.

FCGI (fast common gateway interface)

Fast Common Gateway Interface (FCGI) ist eine Weiterentwicklung des Common Gateway Interface (CGI). Das Protokoll bildet die Schnittstelle zwischen interaktiven Programmen mit einem Webserver. Das Ziel von FastCGI ist es, den Overhead zu reduzieren, der das Interfacing des Webservers mit CGI-Programmen begleitet. Dadurch kann der Webserver gleichzeitig mehr Webseiten-Anfragen bearbeiten.

FastCGI ist eine Programmierschnittstelle, die bestimmte Schwachpunkte des CGI-Protokolls, wie dei eingeschränkte Effizienz und Skalierbarkeit bei hoher Belastung, behebt und Webanwendungen beschleunigt. Benutzeranfragen, die ein bestimmtes Anwendungsprogramm verwenden, werden mit FastCGI etwa zehnmal schneller bearbeitet als mit dem CGI-Protokoll. Als Plugin für den Webserver erfodert der Einsatz von FastCGI nur geringfügige Änderungen an vorhandenen Serveranwendungen.

FCGI wurde erstmals in den 90er Jahren vorgestellt und war eine Antwort auf Netscapes eigenes NSAPI für die Entwicklung von Web-Anwendungen. Einige Webserver-Hersteller haben die Open-Market-Entwicklung FCGI in ihre Produkte implementiert. Verglichen mit anderen Techniken, die auch auf die Beschleunigung und Vereinfachung von der Server-Subprogramme setzen, folgt FCGI den CGI-Paradigmen. *Siehe auch*: *CGI (S. G10)*.

FDDI (fiber distributed data interface)

FDDI war 1989 der erste internationale Standard für ein Hochgeschwindigkeitsnetz. Fiber Distributed Data Interface (FDDI) ist eine ISO-Norm für einen 100-Mbit/s-Token-Ring, vorgeschlagen von ANSI in der Arbeitsgruppe X3T9.5. Das FDDI-Protokoll ist eines der wenigen Zugangsverfahren, das speziell für eine hohe Übertragungsgeschwindigkeit und für die Verwendung von Glasfaser entwickelt wurde.

FDDI kann trotz der hohen Datenrate nur begrenzt für Echtzeit-Applikationen eingesetzt werden. Im Standardbetrieb treten bei 50 aktiven Stationen Übertragungsverzögerungen von bis zu 200 ms auf, was den maximal tolerierbaren Wert von 10 ms weit übersteigt. Es besteht allerdings im so genannten Synchronbetrieb die Möglichkeit, bestimmten Stationen eine feste Übertragungsbandbreite zuzuordnen. Die resultierende Übertragungsverzögerung kann dadurch auf 8 ms bis 16 ms reduziert werden. *Siehe auch*: *LAN (S. G46)*.

Fehlermanagement (*fault*)

Das Fehlermanagement (FM) ist ein Funktionsbereich des OSI-Managements und des Fault, Configuration, Account, Performance, and Security Management (FCAPS) und umfasst alle Aufgaben zur Fehlerprophylaxe, Fehlererkennung und Fehlerbehebung. Durch die ständige Beobachtung des Netzes und der angeschlossenen Systeme werden Änderungen in den Netzparametern erkannt und daraus Rückschlüsse auf zu erwartende Fehler gezogen.

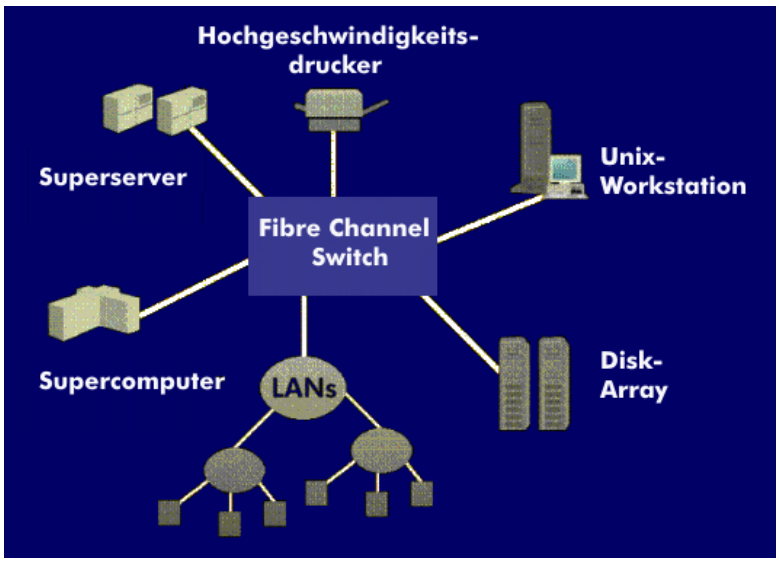
Die wichtigsten Aufgabenbereiche des Fehlermanagements sind die Statusanzeige, Fehlererkennung, Fehlerkorrektur, Alarmauslösung und Problemanalyse sowie die Wiederherstellungsverfahren für die Fehlerbehebung.
Die Fehlererkennung und Fehlerkorrektur werden qualifiziert und quantifiziert durch die Fehlerrate und die Fehlerdauer. Die Bewertung erfolgt über die verschiedenen mittleren Reparatur- und Ausfallzeiten:

- Mean Time between Failure and Repair (MTFR) ist die durchschnittliche Zeit vom Auftreten eines Fehlers bis zu dessen Behebung.
- Mean Time between Failure and Disclosure (MTFD) ist die durchschnittliche Zeit vom Auftreten eines Fehlers bis zu dessen Entdeckung.
- Mean Time between Disclosure and Diagnosis (MTDD) ist die durchschnittliche Zeit zwischen der Entdeckung und der Diagnose eines Fehlers.
- Mean Time between Diagnosis and Repair (MTDR) ist die durchschnittliche Zeit zwischen der Diagnose und der Fehlerbehebung.
- Mean Time between Failures (MTBF) ist die durchschnittliche Zeit zwischen dem Auftreten zweier Fehler.

Siehe auch: *FCAPS (S. G20)*.

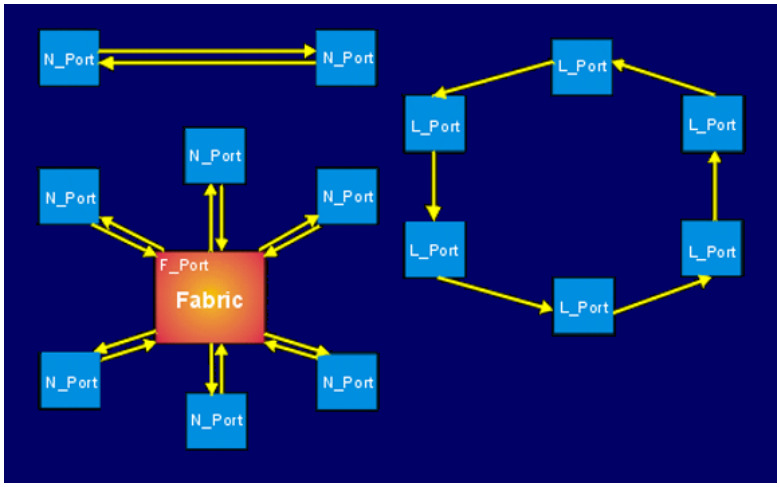
Fibre-Channel

Fibre-Channel (FC) ist eine High-Speed-Übertragungstechnik, die für den schnellen Datentransfer zwischen Workstations, Mainframes, Supercomputern, Speichereinrichtungen und Displays entwickelt und von der ANSI standardisiert wurde.
Der Fibre Channel hat weder eine kanaltypische Architektur noch eine netzwerktypische, er kombiniert die Vorteile beider Architekturen in einer einzigen I/O-Schnittstelle miteinander. Ein solcher Kanal verfügt über eine hohe Flexibilität und ermöglicht leistungsstarke, serielle Hochgeschwindigkeitsübertragungen unter Benutzung des FCP-Protokolls oder anderer gängiger Protokolle.
Das FC-Schichtenmodell kennt daher Kanal-Protokolle wie Small Computer System Interface (SCSI), Internet SCSI (iSCSI), Single Byte Command Code Set (SBCCS), High Performance Parallel Interface (HIPPI), Enterprise System Connection (ESCON) und Fibre Channel Connection (FICON), sowie Netzwerkprotokolle wie FDDI, IP-Protokoll und auch ATM.



Die von Fibre-Channel unterstützten Übertragungsraten reichen von 133 Mbit/s bis hin zu 10 Gbit/s. Die FC-Technologie kann durch einen Switch mit Multipoint-Adressierungsmöglichkeit auch in den LAN-Bereich adaptiert werden. Dabei unterstützt

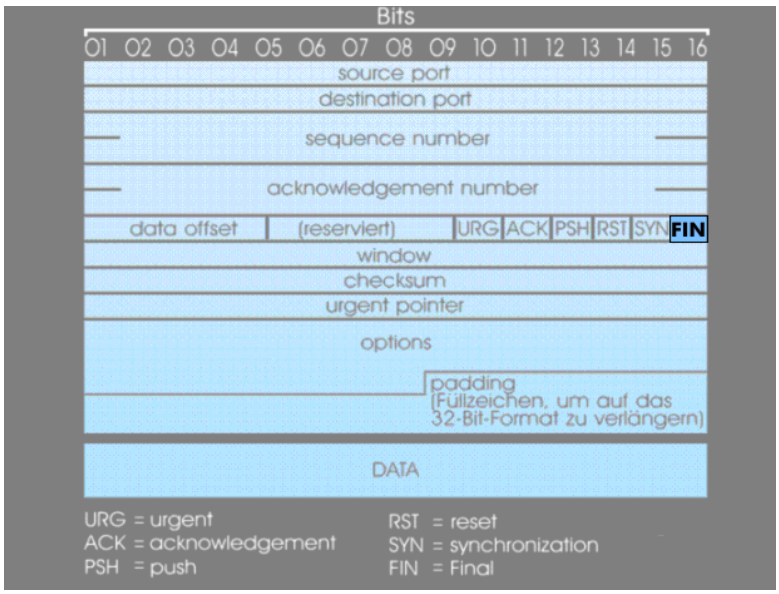
Fibre-Channel das Kanal-Interface als auch das Netzwerk-Interface.
Die FC-Topologie basiert auf einem Link, der aus zwei unidirektionalen Glasfaserstrecken aufgebaut ist. Darüber hinaus können auch Kupferleitungen als FC-Übertragungsmedien genutzt werden. Mit Glasfaser können Entfernungen von mindestens 10 km überbrückt werden, mit Kupferkabel über 30 m. Mit Einführung der kupferbasierten Übertragung wurde "Fiber Channelïn "Fibre Channelümbenannt.



Fibre-Channel basiert auf Port-Verbindungen und kennt mehrere FC-Dienstklassen. Bei den Port-Verbindungen kann es sich um eine Punkt-zu-Punkt-Verbindung zu einem Port einer Switching-Fabric, dem eines Hubs oder einer Loop handeln. Das bedeutet, dass als Fibre-Channel-Topologie die reine Punkt-zu-Punkt-Verbindung, die sternförmige Topologie mit FC-Switch und die vermittelte Ringtopologie, die Fibre Channel Arbitrated Loop (FC-AL), zur Verfügung stehen.
Fibre-Channel hat sich im High-End-Bereich des Hochgeschwindigkeitstransfers etabliert, es ist weniger für den universellen LAN-Einsatz mit wechselnden Verkehrsprofilen geeignet, sondern ideal für Speichernetze (SAN) und wegen der vorhersagbaren Übertragungszeiten auch für Video-Übertragungen. Der Fibre-Channel wurde vom ANSI Committee X3T11 im Jahre 1994 unter der Norm X.3230 standardisiert. Mit diesem Standard wurde der 100 MB/s schnelle Fibre-Channel als zuverlässige Interconnect-Möglichkeit mit der für Disk-I/O erforderlichen Bandbreite und Durchsatzraten außerhalb der elitären Mainframe-Welt definiert. Die entsprechenden Übertragungsraten liegen bei 1,0625 Gbit/s, 2,125 Gbit/s und 4,25 Gbit/s.
Die Fibre Channel Industry Association (FCIA) hat im Jahre 2000 einen neuen Standardisierungsvorschlag für einen 10 Gbit/s schnellen Fibre-Channel ausgearbeitet: 10GFC, 10 Gigabit Fibre-Channel. Ein weiteres interessantes FC-Konzept ist Fibre Channel over Ethernet (FCoE).

FIN (final flag)

Die Final Flag ist ein Datenfeld im Control-Flag-Feld des TCP-Headers. Ein gesetztes Final-Flag zeigt an, dass der Sender die Verbindung abgebaut hat und keine weiteren Daten übertragen wird. Beim TCP-Protokoll erfolgt der Verbindungsabbau auf ähnliche Art und Weise wie der Verbindungsaufbau über den 3-Wege-Handshake, wobei beim Verbindungsabbau nicht das Synchronisations-Flag (SYN) benutzt wird, sondern das Final-Flag (FIN).



Flusskontrolle

Die Flusskontrolle 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. Da die Flusskontrolle auf unterschiedlichen Ebenen und zwischen verschiedenen Netzkomponenten stattfinden, so zwischen Anwendungsprozessen, Ports, Endgeräten, Netzknoten und Subnetzen, handelt es sich um unterschiedliche hard- oder softwaremäßige Steuerungsverfahren.

Die Flusskontrolle ist eine Funktion der Vermittlungsschicht, sie kann aber auch funktional auf der Vermittlungsschicht oder auf höheren Schichten angesiedelt sein und den Datentransport, Ende-zu-Ende-Verbindungen und auch die Anwendung einschließen. Bei der Flusskontrolle vereinbaren die in Kontakt stehenden Stationen ihre Kontrollparameter untereinander. Diese geben beispielsweise die Anzahl der Datenpakete an, die eine sendende Station senden darf, ohne vom Empfänger eine Empfangsbestätigung erhalten zu haben. Sie kann mit Quittungsbetrieb arbeiten, entweder mit Einzelblockquittierung oder mit der gemeinsamen Quittierung von mehreren Datenpakete in einem Zeitfenster.

Alle Übertragungsprotokolle arbeiten mit Flusskontrolle, allerdings mit unterschiedlichen Algorithmen. Bei ATM ist es die Generic Flow Control (GFC), bei Fibre-Channel gibt es das Credit-Verfahren, im High Level Data Link Control (HDLC) und im Synchronous Data Link Control (SDLC) basiert sie auf der Fenstertechnik mit veränderlicher Fenstergröße, beim TCP-Protokoll wird die Flusskontrolle im Window-Datenfeld geregelt und benutzt ein Verfahren mit verschiebbarer Fenstergöße und bei Lossless Ethernet nach einem Prioritätenprinzip, dem Priority Flow Control (PFC).

Fragmentierung

In IP-Netzen handelt es sich beim Fragmentieren um die Unterteilung von großen Datenpaketen in mehrere kleinere Datenpakete. So kann beim Routen von Datenpaketen über mehrere Netze die Paketlänge die maximal verarbeitbare Paketlänge der Router oder die zulässige Datenpaketlänge der Netze übersteigen. Dies ist beispielsweise beim Übergang

von Ethernet mit einer Paketlänge von 1.518 Bytes zu X.25 mit einer Paketlänge von 512 Bytes der Fall. In einem solchen Fall werden die Datenpakete in Teile zerlegt (fragmentiert) und einzeln über das Netz geschickt.
Das Fragmentieren gehört zu den Standardfunktionen des IP-Protokolls. Fragmente werden jeweils mit einem vollständigen IP-Header versehen und als unabhängige Datenpakete übertragen. Die Fragmente können unterschiedliche Wege über das Netz nehmen und die Datenstation in unterschiedlicher Reihenfolge erreichen. Die Datenstation muss in der Lage sein, die Fragmente zu sortieren und diese in der richtigen Reihenfolge an die höhere Protokollschicht zu übergeben. Dies wird durch die Angabe eines Offsets in dem eigens hierfür vorgesehenen Fragment-Offset-Feld im IP-Header erreicht. Der Vorgang des Zusammensetzens der einzelnen Datenpakete wird Reassembling genannt.

Frequenzmultiplexverfahren (FDM)

Beim Frequenzmultiplex wird ein breites Frequenzband in mehrere schmälere Frequenzbänder unterteilt. Jedes dieser Bänder entspricht einem eigenen Übertragungskanal.
Siehe auch: *Multiplexverfahren (S. G54)*.

FTP (file transfer protocol)

Das File-Transfer-Protokoll (FTP) dient dem Dateitransfer zwischen verschiedenen Systemen und der einfachen Dateihandhabung. Das FTP-Protokoll basiert auf dem TCP-Protokoll und kennt sowohl die Übertragung zeichencodierter Informationen als auch von Binärdaten. In beiden Fällen muss der Benutzer eine Möglichkeit besitzen zu spezifizieren, in welcher Form die Daten auf dem jeweiligen Zielsystem abzulegen sind. Die Dateiübertragung wird vom lokalen System aus gesteuert, die Zugangsberechtigung für das Zielsystem wird für den Verbindungsaufbau mittels User-Identifikation und Passwort überprüft.
Will ein Client mit dem Server kommunizieren, baut der Benutzer über den Interpreter im Client eine Verbindung zum Interpreter im Server auf. Über diese Steuerverbindung kommunizieren Client und Server über einen Satz festgelegter Kommandos. Es gibt mehrere Kommandosätze für die Zugriffssteuerung, die Befehle für den Datentransfer und die FTP-Services.
Zum Austausch der Nutzdaten baut der Server eine zweite Verbindung zum Client auf, und zwar wird diese Verbindung von dem Datenübertragungsprozess (DTP) gesteuert. Über diese Verbindung werden die Nutzdaten übertragen. Beim Verbindungsabbau bestätigt der Server-Interpreter das Ende des Datentransfers über die Steuerverbindung an den Interpreter des Clients.
Das FTP-Protokoll ist eines der ältesten Protokolle. Die erste Version ist aus dem Jahr 1971, die aktuelle Version aus 1985. Da das FTP-Protokoll die Steuer- und Nutzdaten unverschlüsselt überträgt, also auch den Benutzernamen und das Passwort, hat es Schwachstellen in der Datensicherheit und ist es anfällig auf Angriffe.
Neben dem FTP-Protokoll gibt es weitere Protokolle für den ungesicherten und gesicherten Dateitransfer. Da ist das aus 1984 stammende Simple File Transfer Protocol (SFTP) zu nennen, das sich ebenso wie das Trivial File Transfer Protocol (TFTP) eine geringere Funktionalität hat, als das FTP-Protokoll und ebenfalls ungesichert sind. Als Alternativen für den gesicherten Dateitransfer stehen Secure FTP, Secure Copy Protocol (SCP), FTP over SSL (FTPS) und das SSH File Transfer Protocol (SFTP) zur Verfügung.

FTP-Befehle	Funktionen
`ascii`	Umschalten auf Text
`binary`	Umschalten auf Binärdaten
`cd`	Verzeichniswechsel auf dem Server
`dir`	Verzeichnisinhalt anzeigen
`get`	Datei downloaden
`lcd`	Verzeichniswechsel auf dem Client
`mget`	Mehrere Dateien downloaden
`mput`	Mehrere Dateien uploaden
`prompt`	Umschalten der Bestätigungsanfrage
`put`	Datei uploaden
`pwd`	Verzeichnisname auf dem Server anzeigen
`quit`	Sitzung beenden

Reale FTP-Befehle	Funktionen
`ABOR`	Abbrechen einer Datenübertragung
`CWD`	Wechseln des Arbeitsverzeichnisses
`DELE`	Löschen von Dateien
`LIST`	Auflisten der entfernten Dateiinformationen
`MDTM`	Modifikationsdatum einer Datei
`MKD`	Erstellen eines Verzeichnisses
`MODE`	Setzen des Transfermodus
`NLST`	Ausgabe der Verzeichnisinhalte
`PASS`	Übergabe des Passworts
`PASV`	Passiven Modus aktivieren
`PORT`	Angabe des Ports für die Datenübertragung im aktiven Modus
`PWD`	Verzeichnisname anzeigen
`QUIT`	Sitzung beenden
`RETR`	Download einer entfernten Datei
`RMD`	Entfernen eines Verzeichnisses
`RNFR`	Umbenennen einer Datei. Mit `RNFR` wird die Datei angegeben,
`RNTO`	mit `RNTO` der neue Name
`SITE`	Ausführen eines seitenspezifischen Kommandos
`SIZE`	Größe einer Datei
`STOR`	Upload einer Datei
`TYPE`	Setzen des Transfertyps
`USER`	Übergabe des Nutzernamens

Geocast

Geocast ist ein Kommunikationsmechanismus, der sich auf das Übermitteln einer Nachricht in einem Netz an ein räumlich abgegrenztes Gebiet bezieht. Geocast ist eine spezielle Form des Multicast, bei dem zwischen der symbolischen und geometrischen Adressierungsart unterschieden wird. Bei geometrischer Addresierung wird mittels absoluter Koordinaten das Gebiet beschrieben, bei symbolischer mit einem Alias. Des Weiteren enthält das neue Internet-Protokoll (IPv6) Erweiterungen, die Geocast unterstützen. *Siehe auch*: *Kommunikationsart (S. G43)*.

gTLD (generic top level domain)

Die Generic Top Level Domain (gTLD) ist eine Kategorie von Top Level Domains (TLD), die von der Internet Corporation for Assigned Names and Numbers (ICANN) vergeben werden. Bei den generischen Top Level Domains unterscheidet man zwischen den allgemeinen und den gesponserten Top Level Domains.
Die allgemeinen TLDs werden auch als ungesponserte TLDs, Unsponsored Top Level Domain (uTLD), bezeichnet, das sind die normalen generischen TLDs wie *.arpa, *.org für nichtkommerzielle Organisationen und Verbände oder *.info für Informationsanbieter.
Zu den gesponserten TLDs, Sponsored Top Level Domain (sTLD), gehörten bisher die TLDs *.gov für die Regierungsstellen der USA, *.pro für spezielle Berufsgruppen, *.edu für Bildungseinrichtungen, *.aero für die Lufttransportindustrie und einige weitere. Gesponserte TLDs können jetzt auch von Unternehmen oder von weltweit agierenden Organisationen bei der ICANN beantragt werden. So könnten zukünftige gesponserte sTLDs auf Nokia oder Deutsche Bank ausgestellt werden als *.nokia oder *.db. Die ICANN verlangt dafür allerdings relativ hohe Anmeldegebühren und jährliche Verwaltungsgebühren.

Domain gTLD	Organisationsform
.aero	Lufttransportindustrie
.arpa	Arpanet
.biz	Business für große und kleinere Unternehmen
.com	Kommerzielle Domain
.coop	Kooperationen, Genossenschaften
.edu	Schulen, Universitäten, Bildungseinrichtungen
.gov	Regierungsstellen der Vereinigten Staaten von Amerika
.info	Informationsdienste
.int	International tätige Institutionen
.mil	Militär der Vereinigten Staaten von Amerika
.museum	Museen
.name	Privatpersonen
.net	Netzspezifische Dienste und Angebote
.org	Nichtkommnerzielle Unternehmen und Projekte
.pro	Professionals, spezielle Berufsgruppen

Siehe auch: *DNS (S. G15) und sTLD (S. G75)*.

Halbduplex-Betrieb (Übertragungsart)

Als Halbduplex-Betrieb bezeichnet man eine Übertragungsart der Datenübertragung, bei der die Datenstationen über den gleichen Kanal senden und empfangen können, allerdings nicht gleichzeitig. Halbduplex erlaubt die wechselseitige Nutzung einer Übertragungsleitung in beiden Richtungen und wird deswegen auch als Wechselbetrieb bezeichnet.

HARQ (hybrid automatic repeat request)

Hybrid Automatic Repeat Request (HARQ) wird bei High Speed Downlink Packet Access (HSDPA) und bei Mobile-WiMAX zur Effizienzsteigerung benutzt. Die HARQ-Technik dient der fehlerfreien Übertragung von Datenpaketen und kann fehlerhaft empfangene Datenpakete erneut anfordern. Dabei wird das als fehlerhaft erkannte Datenpaket gespeichert und nach dem erneuten Senden wird es decodiert und in Verbindung mit den vorher empfangenen Paketkopien betrachtet.
HARQ arbeitet in einem Stop-and-Wait-Mechanismus, der auf Bestätigungen (CONF) und negativen Bestätigungen (NAK) basiert, bei der die Empfangsstation über einen Kanal mit der Sendestation kommuniziert. Im Falle von High Speed Downlink Packet Access (HSDPA) und High Speed Uplink Packet Access (HSUPA) ist es der E-HICH-Kanal über den die Sendestation mit der Empfangsstation kommuniziert.

Zur Erhöhung der Round Trip Time (RTT) findet das wiederholte Übertragen der Datenpakete auf dem Physical Layer statt.

Header-Prüfsummen-Feld

Im IP-Header ist die Header Checksum (HCS) ein 16 Bit langes Datenfeld, das die Prüfsumme aus den Daten des IP-Headers und des eventuell benutzten Optionsfeldes enthält; aber ohne den Payload. Mit diesen Daten kann der IP-Header auf Fehler hin überprüft werden. Beim Fragmentieren von IP-Datagrammen verändert sich die Prüfsumme des Headers, ebenso bei der Dekrementierung der TTL-Zeit. Die Prüfsumme muss dann neu errechnet werden.
Fehlerbehaftete IP-Datenpakete werden einfach verworfen und von der sendenden Datenstation neu angefordert.

HTTP (hypertext transfer protocol)

Das Hypertext Transfer Protocol (HTTP) ist ein allgemeines, statusloses, objektorientiertes Protokoll zur Datenübertragung im World Wide Web (WWW). Das HTTP-Protokoll ist in RFC 2616 aus dem Jahr 1999 beschrieben und definiert einen Satz von Anfragen und Antworten, Request/Response, mit denen ein Web-Client und ein Webserver während einer HTML-Sitzung miteinander kommunizieren. Als einfaches, zustandsloses Protokoll geht es beim Request-Response-Verfahren um die Übertragung einer einzelnen Webseite oder um Teile einer Webseite. Wobei sich das HTTP-Protokoll nicht den dazugehörigen Benutzer oder andere Einträge merkt. Bei jedem neuen Eintrag muss die komplette Webseite neu übertragen werden.
Beim HTTP-Protokoll richtet der Web-Client, das ist der Browser, eine Anfrage an den Webserver. Bei jeder Anfrage nach einem neuen Dokument stellt der Browser über das TCP/ IP-Protokoll eine Verbindung zum Webserver her, über die die nachgefragten HTML-Dokumente übertragen werden. In einigen Erweiterungen des HTTP-Protokolls geht es um Bild-, Audio- oder Video-Dateien. Nach der Übermittlung der Daten an den Browser, wird die Datenverbindung wieder aufgehoben.
Das HTTP-Protokoll dient der Adressierung der Objekte über die Internetadresse (URL), es wickelt die Interaktion zwischen Client, dem User Agent, und Server, Origin Server, ab und sorgt für die Anpassung der Formate zwischen Client und Server. Die Kommunikation erfolgt über das Request-Response-Verfahren bei dem der User Agent über die vorher hergestellte TCP-Verbindung einen Request an den Server sendet. Dieser antwortet mit einem Response, der immer einen HTTP-Status-Code enthält. Aus einem solchen Status-Code kann hervorgehen, dass eine Transaktion abgeschlossen ist oder dass der entsprechende Server nicht erreicht werden kann. Werden Dokumente wegen falscher oder ungültiger Internetadressen nicht gefunden, beinhaltet der Status-Code eine Fehlermeldung, die vom Browser angezeigt wird. Bei fehlerhaften URLs wird ein Bad Request mit dem Fehlercode Error 400 angezeigt, für nicht gefundene Webseiten oder Fehlerseiten ist es der Fehlercode Error 404. Bei einem Serverfehler wird der HTTP-Status-Code 500 angezeigt.
Für den zuverlässigen Nachrichtentransport gibt es das HTTP-R-Protokoll, wobei das "R"für Reliable steht. In der mit Secure Socket Layer (SSL) verschlüsselten Version heißt es HTTPS (Secure). Das Serverprogramm zu HTTP ist httpd. Es beantwortet Anfragen von WWW-Clients.
Da das HTTP-Protokoll bei der Übermittlung von Webseiten für jedes Request nach einem Webseitendetail eine neue TCP-Verbindung aufbauen muss, hat es eine relativ lange Latenzzeit, was die Geschwindigkeit bei der Webseiten-Übertragung einschränkt. Diesen Nachteil beheben das von Google entwickelte SPDY-Protokoll (Speedy), die Version HTTP 2.0, HTTP/2 und das QUIC-Protokoll.

HTTP-Status-Codes

Die Kommunikation zwischen Web-Browsern und Webservern erfolgt über das http-Protokoll, wobei der Browser eine Anfrage an den Webserver richtet, dass dieser ihm

eine bestimmte Webseite als HTML-Dokument zur Verfügung stellt. Der Webserver sendet daraufhin das ihm vorliegende Dokument mit einem HTTP-Status-Code an den Browser. Aus dem HTTP-Status-Code geht hervor, ob die Anfrage erfolgreich oder erfolglos ausgeführt wurde. In beiden Fällen enthält der HTTP-Status-Code Detailinformationen zu der Übertragung.
HTTP-Status-Codes sind dreistellige Ziffern zwischen 100 und 599, die in fünf Gruppen gegliedert sind, und aus denen der Status der Verbindung hervorgeht. Status-Codes der Gruppe 1xx betreffen allgemeine Informationen wie "die Anfrage wird noch bearbeitet,öder SZeitüberschreitung beim Ladevorgang."Bei den Status-Codes der Gruppe 2xx geht es um die Durchführung der Bearbeitung. Typische Meldungen diese Status-Gruppe sind "die Anfrage wurde erfolgreich bearbeitetöder "der Client soll das Dokument neu aufbauen."Bei den Status-Codes der Gruppe 3xx geht es um die Weiterleitung der Anfrage auf andere Internetadressen. Mit diesen Status-Codes wird die erfolgreiche Bearbeitung sichergestellt. In den HTTP-Status-Codes 4xx geht es um Client-Fehler, die bei einer Anfrage auftreten können. Hierzu gehören Bad Request für eine fehlerhafte Anfrage, Error 404 Not Found für eine nicht gefundene Ressource oder Method not Allowed für Anfragen, die nur mit anderen http-Methoden durchgeführt werden dürfen. In der letzten Status-Code-Gruppe 5xx geht es um Serverfehler, die allerdings nicht eindeutig von Clientfehlern zu trennen sind. Zu den 5xx-Status-Codes gehören u.a. Internal Server Error, Bad Gateway, Service Unavailable.

Status-Codes	Bedeutung
1xx	Information
2xx	Anforderung erfolgreich 200: OK, 202: Accepted, 204: No Count
3xx	Redirektion auf andere Adresse 301: Moved Permanently, 302: Moved Temporarily, 304: Not Modified
4xx	Fehler auf Clientseite 400: Bad Request, 401: Unauthorized, 403: Forbidden, 404: Not Found
5xx	Fehler auf Serverseite 500: Internal Server Error, 501: Not Implemented, 502: Bad Gateway, 503: Service Unavailable

Siehe auch: *HTTP (hypertext transfer protocol) (S. G28) und HTTP/2 (S. G29).*

HTTP/2

Das HTTP-Protokoll dient der Übertragung von Webseiteninhalten. Es handelt sich um ein als Request-Response-Verfahren bei dem der Web-Client und der Webserver über das TCP-Protokoll miteinander kommunizieren. Da sich seit der Standardisierung des HTTP-Protokolls im Jahr 1999 der Webseiteninhalt markant verändert hat und aus vielen, manchmal mehreren hundert Details wie Texten, Grafiken, Fotos und Videos besteht, hat sich die Übertragungszeit für die Übermittlung einer Webseite verlängert, zumal das HTTP-Protokoll nicht auf eine Verringerung der Latenzzeit hin optimiert ist.
Wegen dieser längeren Webseiten-Ladezeit hat die Internet Engineering Task Force (IETF) mit HTTP 2.0, HTTP/2, und der HTTP-Header-Kompression HPACK eine Version mit einer kürzeren Latenzzeit standardisiert. Zur Verkürzung der Latenzzeit trägt auch die Server-Push-Technologie und das parallele Laden mehrerer Seitenelemente über eine einzelne

TCP-Verbindung bei.
HTTP/2 hat die gleiche Syntax wie HTTP 1.1, ebenso die HTTP-Status-Codes, HTTP-Headerfelder und die Uniform Resource Identifier (URI). Durch die Änderungen von HTTP/2 werden existierende Web-Anwendungen nicht beeinträchtigt. Allerdings können neue Anwendungen von den kürzeren Ladezeiten profitieren. *Siehe auch*: *HTTP (hypertext transfer protocol) (S. G28)*.

ICMP (Internet control message protocol)

Das Internet Control Message Protocol (ICMP) ist ein Protokoll zur Übertragung von Statusinformationen und Fehlermeldungen in IP-, TCP- und UDP-Protokollen zwischen IP-Netzknoten. Besonders Gateways und Hosts benutzen ICMP, um Berichte über Probleme mit Datagrammen zur Originalquelle zurückzuschicken. Die Meldungen des ICMP-Protokolls sind in zwei Klassen definiert: In Fehlermeldungen und Informationsmeldungen.
Fehlermeldungen können über den Status der Verbindung informieren, wie "DEE nicht erreichbar", Wegumleitung, "Ressourcen nicht mehr nutzbar", SZeit abgelaufenöder Parameterprobleme. Und bei den Informationsmeldungen kann es sich Angaben für die Zeitmessung, Adressmaske, Router-Findung oder Echo handeln, mit denen festgestellt werden kann, ob ein Bestimmungsort erreichbar ist und antwortet.
Das ICMP-Protokoll wird von dem IP-Protokoll wie ein Protokoll höherer Schichten behandelt und ist integraler Bestandteil des IP-Protokolls. Daher setzt sich der ICMP-Header auch aus einem IP-Header mit nachfolgenden ICMP-Daten zusammen.

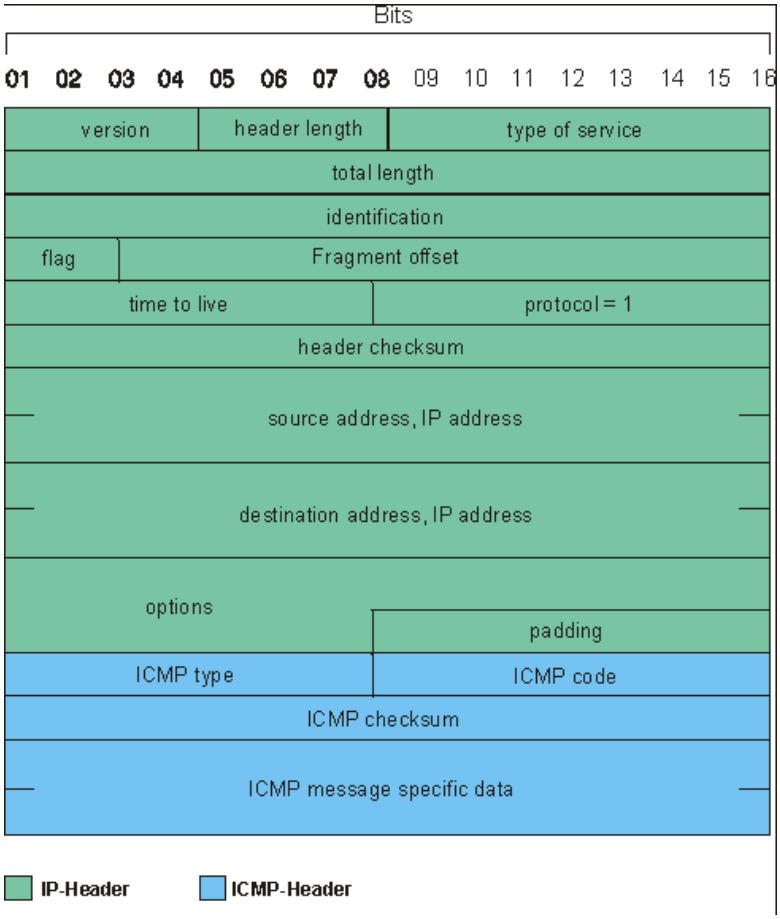
Typ	Code	Bedeutung
0	0	Ping, Echo Antwort
3	0	Netzwerk nicht erreichbar
3	1	Host nicht erreichbar
3	2	Ziel-Protokoll nicht verfügbar
3	3	Ziel-Port nicht erreichbar
3	6	Netzwerk unbekannt
3	7	Ziel-Host unbekannt
4	0	Überlastkontrolle, ungenutzt
8	0	Ping, Echo-Anfrage
9	0	Routen-Bekanntmachung
10	0	Router-Discovery
11	0	Trace-Route
12	0	Schlechter IP-Header

Das Datenformat von ICMP kennt den ICMP-Typ (8 Bit), den ICMP-Code (8 Bit) und die ICMP-Prüfsumme (16 Bit), gefolgt von der ICMP-Meldung (224 Bit). Je nach Meldung werden weitere Datenfelder hinzugefügt. So beispielsweise bei der Meldung Parameter-Probleme das 1 Byte lange Pointer-Feld oder im Falle des Echo-Request das Identification-Feld (8 Bit) und das Datenfeld für die Sequenznummer (8 Bit). Im Falle des Timestamp-Request addieren sich weitere drei vier Byte Datenfelder für den Timestamp hinzu: Originate Timestamp (4 Byte), Receive Timestamp (4 Byte) und Transmit Timestamp (4 Byte). Ähnliches gilt für die Bestimmung von Routern.
Das ICMP-Protokoll ist in RFC 792 veröffentlicht.

ICMP-Header

Der ICMP-Header ist als Erweiterung des IP-Headers und integraler Bestandteil des IP-Protokolls anzusehen. Er wird innerhalb des Datenrahmens des IP-Datagramms übertragen und besteht aus dem Type-Feld, dem Code-Feld, die jeweils ein Byte umfassen, dem zwei Byte langen Prüfsummenfeld (ICMP Checksum) und dem Datenfeld für die ICMP-Nachrichten.
Im Internet Control Message Protocol (ICMP) sind mehrere Pakettypen festgelegt, die je nach Funktion in das Type-Feld eingetragen werden. Das Type-Feld und das Code-Feld spezifizieren gemeinsam die Funktion einer Nachricht. Das Code-Feld wird allerdings nur

bei bestimmten Pakettypen für Einträge benutzt. Beispielsweise dann, wenn eine Station oder wenn das adressierte Netzwerk nicht erreichbar sind (Type 3), wenn das Datagramm umgeleitet werden muss (Type 5), die Lebensdauer des Datagramm abgelaufen ist (Type 11) oder wenn Parametrisierungsprobleme auftreten.



Die verschiedenen Pakettypen unterstützen die verschiedensten Funktionen. So dient beispielsweise der Pakettype 8 einer Ping-Anforderung und der Pakettype 0 einer Ping-Antwort. Die anfragende Station trägt in das Type-Feld die Ziffer 8 ein, die antwortende Station die Ziffer 0. Mit Ping können Knoten dahingehend überprüft werden, ob sie IP-Datagramme empfangen können. Ein Paket von Type 3 wird von einem IP-Router zurückgesendet, wenn dieser den Knoten nicht erreichen oder das Datagramm nicht weiterleiten kann. Zur Flusskontrolle werden Pakete von Type 4 (Source Quench) verwendet. Wenn ein Router die ankommenden Datagramme nicht verarbeiten kann, sei es wegen einer zu hohen Datenrate oder weil der Puffer überläuft, dann vernichtet der Router die Datagramme und sendet jeweils eine ICMP-Nachricht in der im Type-Feld die Ziffer 4 eingetragen ist an die sendende Station, damit diese den Datenstrom drosselt.

Type	Funktion
0	Echo Reply [RFC792]
1	Unassigned
2	Unassigned
3	Destination Unreachable [RFC792]
4	Source Quench [RFC792]
5	Redirect [RFC792]
6	Alternate Host Address
7	Unassigned
8	Echo [RFC792]
9	Router Advertisement[RFC1256]
10	Router Solicitation[RFC1256]
11	Time Exceeded [RFC792]
12	Parameter Problem [RFC792]
13	Timestamp [RFC792]
14	Timestamp Reply [RFC792]
15	Information Request [RFC792]
16	Information Reply [RFC792]

Type	Funktion
17	Address Mask Request [RFC950]
18	Address Mask Reply [RFC950]
19	Reserved (for Security)
20 - 29	Reserved (for Robustness Experiment)
30	Traceroute[RFC1393]
31	Datagram Conversion Error[RFC1475]
32	Mobile Host Redirect
33	IPv6 Where-Are-You
34	IPv6 I-Am-Here
35	Mobile Registration Request
36	Mobile Registration Reply
37	Domain Name Request [RFC1788]
38	Domain Name Reply [RFC1788]
39	SKIP
40	Photuris [RFC2521]
41	ICMP messages utilized by experimental [RFC4065]
42 - 255	Reserved

Zur Überprüfung von Routingfunktionalitäten und zur Änderung von Routingtabellen werden ICMP-Pakete der Typen 5, 9 und 10 verwendet. Diese Pakettypen werden sowohl

für den Anschluss von neuen Geräten als auch bei Konfigurationsänderungen benutzt. Ein Information Request entspricht dem Typ 15 und ein Information Reply dem Typ 16. Typ 17 ist als Address Format Request definiert und Typ 18 als Address Format Reply.
Die Prüfsummenbildung für das gesamte ICMP-Paket erfolgt im Prüfsummenfeld, einem 2 Bytes langem Feld. An weiteren Kontroll-Informationen gibt es ein Feld für die ICMP-Identifikation. Hierin wird der Kennwert zur Zuordnung eines Fragments zu einem Datagramm festgelegt. Anhand des Identifikationsfeldes ermittelt der Zielknoten, zu welchem Datagramm ein empfangenes Datagramm gehört.
ICMP ist unter RFC 792 beschrieben, die Version ICMPv6 unter RFC 1885.

ICMPv6 (Internet control message protocol version 6)

Internet Control Message Protocol (ICMP) in der Version 6 (ICMPv6) ist ein Protokollzusatz für das IPv6-Protokoll. ICMPv6 wird ebenso wie das klassische ICMP-Protokoll für die Fehlermeldung und -behandlung benutzt und mittels Datagrammen über Routingpfade und Netzwerke übertragen.
Bei der Übertragung kann der ICMPv6-Header verfälscht werden. Stellt ein Empfänger fest, dass die übertragenen Daten nicht korrekt sind, meldet er dies der absendenden Station. Für diesen Zweck bietet ICMPv6 die unterschiedlichsten Fehler- und Informationsnachrichten, die als Ziffern im Type-Feld eingetragen werden.
Es gibt Fehlernachrichten, diese sind mit den Type-Nummern 0 bis 127 gekennzeichnet, und Informationsnachrichten, die den Ziffernbereich von 128 bis 255 belegen. Die meisten Type-Bezeichnungen sind in den TCP/IP-Standarddokumenten (RFC) beschrieben und können einen informellen und experimentellen Charakter haben.

Type	Funktion
	Fehlernachrichten
1	Destination Unreachable [RFC4443]
2	Packet Too Big [RFC4443]
3	Time Exceeded [RFC4443]
4	Parameter Problem [RFC4443]
100	Private experimentation [RFC4443]
101	Private experimentation [RFC4443]
102 -126	Unassigned
127	Reserved for expansion of ICMPv6 error messages [RFC4443]
	Informationsnachrichten
128	Echo Request [RFC4443]
129	Echo Reply [RFC4443]
130	Multicast Listener Query [RFC2710]
131	Multicast Listener Report [RFC2710]
132	Multicast Listener Done [RFC2710]
133	Router Solicitation [RFC4861]
134	Router Advertisement [RFC4861]
135	Neighbor Solicitation [RFC4861]
136	Neighbor Advertisement [RFC4861]
137	Redirect Message [RFC4861]
138	Router Renumbering

Type	Funktion
	Informationsnachrichten
139	ICMP Node Information Query [RFC4620]
140	ICMP Node Information Response RFC4620]
141	Inverse Neighbor Discovery Solicitation Message [RFC3122]
142	Inverse Neighbor Discovery Advertisement Message [RFC3122]
143	Version 2 Multicast Listener Report [RFC3810]
144	Home Agent Address Discovery Request Message RFC3775]
145	Home Agent Address Discovery Reply Message [RFC3775]
146	Mobile Prefix Solicitation [RFC3775]
147	Mobile Prefix Advertisement [RFC3775]
148	Certification Path Solicitation Message [RFC3971]
149	Certification Path Advertisement Message [RFC3971]
150	ICMP messages utilized by experimental mobility protocols such as Seamoby [RFC4065]
151	Multicast Router Advertisement [RFC4286]
152	Multicast Router Solicitation [RFC4286]
153	Multicast Router Termination [RFC4286]
154	FMIPv6 Messages [RFC5568]
155 - 199	Unassigned
200	Private experimentation [RFC4443]
201	Private experimentation [RFC4443]
255	Reserved for expansion of ICMPv6 informational messages [RFC4443]

Neben dem 8 Bit umfassenden Type-Feld besteht der ICMPv6-Header aus dem ebenso langen Code-Feld, dem 16 Bit umfassenden Prüfsummenfeld und dem ICMPv6-Nachrichtenfeld. Die beiden Datenfelder, das Type-Feld und das Code-Feld, spezifizieren gemeinsam die Funktion der ICMP-Nachricht.
Das ICMPv6-Protokoll wird auch im Neighbour Discovery Protocol (NDP) und im Address Resolution Protocol (ARP) benutzt.

IEEE 802.11

Der 802.11-Standard für WLANs stellt sich in den verschiedenen Versionen vollkommen unterschiedlich dar, und zwar hinsichtlich der Übertragungsrate, Frequenzbereiche, Modulationsverfahren, Kanalzahl usw., was nicht zuletzt auf die Entwicklungszeit des Standards und die technologischen Fortschritte zurückzuführen ist. Für den Anwender stellen sich die 802.11-Standards unübersichtlich dar, da sie größtenteils inkompatibel zueinander sind.
Der Basisstandard 802.11 wurde in den 90er Jahren erarbeitet und 1997 verabschiedet. Er hat eine Übertragungsrate von 2 Mbit/s, die 1999 im Standard 802.11a auf bis zu 54 Mbit/s erhöht wurde. Zwischenzeitlich wurden verschiedene 802.11-Standards mit Übertragungsraten von mehreren hundert Mbit/s und mit den Standards 802.11a und 802.11ad sogar solche mit mehreren Gbit/s entwickelt.

WLANs nach 802.11 übertragen die Signale mittels Infrarot (IR) oder über Mikrowellen in den Frequenzbändern bei 2,4 GHz, 5 GHz und im 60-GHz-Band. Im 2,4-GHz-Band, dem ISM-Band, in dem auch Bluetooth und HomeRF sendet, arbeitet der Basisstandard 802.11 mit Spreizbandtechnik (DSSS) und nach dem Frequenzsprungverfahren (FHSS). Dieses Frequenzband wird auch von den Standards 802.11b und 802.11g benutzt, das 5-GHz-Band hingegen von 802.11a und Hiperlan. Auch bei der Modulation und Codierung arbeiten die verschiedenen Standards mit unterschiedlichen Verfahren.
So benutzt 802.11b den Barker-Code und das Complementary Code Keying (CCK), 802.11g hingegen neben den genannten Verfahren auch noch Orthogonal Frequency Division Multiplex (OFDM) und letztendlich 802.11a nur das OFDM-Multiplexing. Aus diesen verschiedenen Verfahren resultieren völlig unterschiedliche Übertragungsraten, die zwischen 1 Mbit/s und 54 Mbit/s liegen. Weitere Unterschiede zeigen sich in der Kanalzahl, die zwischen drei 20-MHz-Kanälen (802.11b, 802.11g) und 19 Kanälen bei 802.11a differiert. Höhere Datenraten im Gigabit-Bereich werden mit den Standards 802.11ac und 802.11ad erzielt.
Dem Standard nach wird ein 802.11-LAN aus mindestens zwei Funkstationen gebildet, die als Basic Service Set (BSS) bezeichnet werden. Die BSS-Stationen sind über den Zugangspunkt mit dem Distribution System (DS) verbunden.
802.11 berücksichtigt in 802.11e und 802.11r Sprachdienste wie VoIP und VoWLAN und hat Mechanismen für das Roaming und den schnellen Wechsel zwischen verschiedenen Access Points (AP), damit dieser unterbrechungs- und verzögerungsfrei erfolgt. Das Problem liegt dabei in der Authentifizierung, die einige Zeit in Anspruch nehmen kann. Um diese Zeit zu verkürzen wenn der Teilnehmer einen Access Point verlässt, sich an einem anderen anmeldet und wieder zu dem vorherigen zurückkehrt, wurde der PMK-Mechanismus entwickelt, was für Pairwise Master Key steht. In diesem Fall wird die erneute Authentifizierung durch eine Bestätigung der PMKID auf wenige Datenpakete reduziert. *Siehe auch*: *LAN (S. G46) und WLAN (S. G91)*.

IGP (interior gateway protocol)

Interior Gateway Protocol (IGP) ist eine Gattungsbezeichnung, die man auf jedes Protokoll anwenden kann, das Informationen über Wegewahl und Erreichbarkeit in einem autonomen System (AS) verbreitet. IGP ist ein IP-Protokoll zum Austausch von Routing-Informationen in autonomen Systemen.
Obwohl es keinen einzigen Standard für das IGP-Protokoll gibt, ist das RIP-Protokoll das populärste. Das IGP-Protokoll ist topologieunabhängig. Da unterschiedliche Topologien und Netzwerke vorhanden sind, gibt es mehrere Interior-Gateway-Protokolle. Gateways können gleichzeitig verschiedene Routing-Protokolle benutzen, wenn sie die Verbindung zwischen autonomen Systemen und einem übergeordneten Backbone-Netzwerk sind. Hierfür stehen neben dem erwähnten Routing Information Protocol (RIP) das Hello-Protokoll, das Interior Gateway Routing Protocol (IGRP) und Open Shortest Path First (OSPF) zur Verfügung.

IKE (Internet key exchange)

Das im Internet Security Association and Key Management Protocol (ISAKMP) eingesetzte Internet Key Exchange (IKE) ist ein Schlüssel-Protokoll zur Verwaltung und zum Austausch der Schlüssel des IP Security Protocol (IPsec). Das IKE-Protokoll bietet ein standardisiertes Verfahren für die Authentifizierung von IPsec-Kommunikationspartnern sowie zur Erzeugung von gemeinsam genutzten Schlüsseln. Bei der Verwendung von Public-Key-Verfahren können dafür digitale Zertifikate und Sicherheitsinfrastrukturen (PKI) eingesetzt werden. Andererseits können auch Pre-shared Keys (PSK) eingesetzt werden.
Beim IKE-Protokoll erfolgt die Abwicklung zwischen zwei Instanzen in zwei Phasen. In der ersten Phase baut es eine Kommunikationsverbindung mit relativ schwachen Sicherheitsmechanismen auf, die der Absicherung und Authentisierung der weiteren Managementvorgänge dient. In der zweiten Phase verhandeln die beiden Instanzen über das

zu verwendende Sicherheitsprotokoll und etablieren dieses. In dieser Phase werden auch die erforderlichen Schlüssel generiert. Beim Verbindungsaufbau wird eine RSA-Verschlüsselung mit Public-Key-Verschlüsselung benutzt, über die ein symmetrischer Schlüssel wie der DES-Algorithmus oder der RC4-Algorithmus generiert wird.

Für den Aufbau einer gesicherten Verbindung müssen mehrere Parameter ausgetauscht werden, die die Art der Verschlüsselung, den Algorithmus, den Schlüssel und dessen Gültigkeitsdauer vorschreiben. Alle diese Parameter werden in der Security Association (SA) beschrieben.

Die Version2 IKEv2 ist eine verbesserte Variante des IKE-Protokolls und löst dieses ab. IKEv2 basiert auf IPsec und stellt darüber eine gesicherte Verbindung für das Virtual Private Network (VPN) her. Das Internet Key Exchange Protocol stammt von der Internet Engineering Task Force und ist in den RFCs 2409 und 4306 beschrieben.

IMAP (Internet message access protocol)

Das Internet Message Access Protocol (IMAP) wurde 1988 entwickelt und dient dem verbindungslosen Zugriff von Mail User Agents (MUA) auf Message Transfer Agents (MTA), um dort E-Mails zu verwalten, zu öffnen oder diese abzuholen.

Dieses effiziente E-Mail-Protokoll löst das Post Office Protocol (POP) ab und ermöglicht es, Nachrichten nur nach Bedarf zu übermitteln. Dazu werden zuerst nur die Kopfzeilen übertragen und nach der Auswahl die Anhänge. Mittels IMAP können vom Arbeitsplatzrechner aus auf dem Mail-Server hierarchische Mailboxen eingerichtet und verwaltet werden. Dazu gehört das Suchen, Löschen, Kopieren, Speichern und Herunterladen von Nachrichten sowie das Ändern des Nachrichtenstatus.

IMAP hat den Vorteil, dass die Speicherung der Nachrichten unabhängig von der verwendeten Client-Software funktioniert. Es ist prädestiniert für den Einsatz in Unternehmen, weil es auch in Groupware-Umgebungen eingesetzt werden kann.

Eine interessante Funktion bildet die Selektionsmöglichkeit der E-Mails im Mail-Server. Dadurch entfällt das Kopieren und nachfolgende Selektieren, was besonders bei der mobilen Kommunikation über Laptops und Handhelds Kosteneinsparungen mit sich bringt.

IMAP wird von allen wichtigen Mail-Servern unterstützt. Da IMAP bereits in der vierten Version vorliegt, heißt die aktuelle Bezeichnung IMAP4.

Institute of Electrical and Electronics Engineers (IEEE)

Das IEEE ist ein weltweiter Berufsverband von Ingenieuren hauptsächlich aus den Bereichen Elektrotechnik und Informationstechnik. Er ist Veranstalter von Fachtagungen, Herausgeber diverser Fachzeitschriften und bildet Gremien für die Standardisierung von Techniken, Hardware und Software. Wissenschaftlichen Beiträgen in Zeitschriften oder zu Konferenzen des IEEE wird im Allgemeinen eine besonders hohe fachliche Güte unterstellt. Mit Veröffentlichungen wie der Zeitschrift IEEE Spectrum setzt sich die Organisation auch für eine fachübergreifende Information und die Diskussion der gesellschaftlichen Auswirkungen neuer Technologien ein.

International Telecommunication Union, Radiocommunication Sector (ITU-R)

Die ITU-R ist einer von drei Sektoren der Internationalen Fernmeldeunion (ITU) mit Zuständigkeit für internationale Angelegenheit auf dem Gebiet Funkkommunikation. *Siehe auch*: *ITU (S. G34)*.

Internationale Fernmeldeunion (ITU)

Die Internationale Fernmeldeunion (englisch International Telecommunication Union, ITU) mit Sitz in Genf ist eine Sonderorganisation der Vereinten Nationen und die einzige Organisation, die sich offiziell und weltweit mit technischen Aspekten der Telekommunikation beschäftigt. Die Ziele der ITU sind Abstimmung und Förderung der internationalen Zusammenarbeit im Nachrichtenwesen durch:

- Internationale Regelungen für die Nutzung von Frequenzen

- Internationale Zuweisung und Registrierung von Sende- und Empfangsfrequenzen
- Internationale Zuweisung von Rufzeichenblöcken, das ITU-Präfix
- Koordinierung der Entwicklung von Telekommunikationsanlagen
- Koordinierung von Bemühungen zur Störungsbearbeitung im internationalen Funkverkehr
- Vereinbarungen von Leistungsgarantien und Gebühren

Die ITU teilt sich auf in den Entwicklungssektor (ITU-D) den Radiokommunikationssektor (ITU-T) früher auch „Comité Consultatif International des Radiocommunications (CCIR)“ genannt und den *Siehe auch*: *ITU-R (S. G34) und ITU-T (S. G42)*.

Internationale Organisation für Normung (ISO)

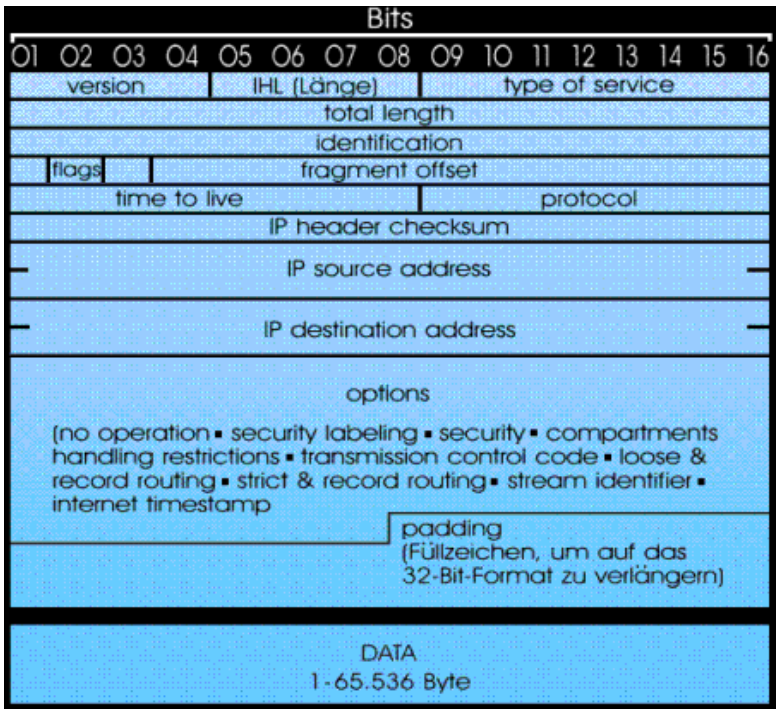
Die internationale Organisation für Normung (ISO) ist die interationale Vereinigung von Normungsorganisationen und erarbeitet interantionale Normen in allen Bereichen mit Ausnahme der Elektrik und Elektronik für die die internationale elektrotechnische Kommision (IEC) verantwortlich ist, sowie der Telekomunikation für die die internationale Fernmeldeunion (ITU) zuständig ist. *Siehe auch*: *Internationale Fernmeldeunion (ITU) (S. G34)*.

Internet Engineering Task Force (IETF)

Die IETF ist eine Organisation, die sich mit der technischen Weiterentwicklung des Internets befasst, um dessen Funktionsweise zu verbessern. Ihr Auftrag ist die Erstellung hochqualitativer, relevanter technischer Dokumente, welche die Art und Weise beeinflussen, wie Menschen das Internet weiterentwickeln, benutzen und verwalten. Diese Dokumente umfassen Internetprotokollstandards, Beschreibungen momentan bekannter Verfahren sowie verschiedener Dokumente mit eher informativem Charakter.

IP (Internet protocol)

Die Aufgabe des Internetprotokolls (IP) besteht darin, Datenpakete von einem Sender über mehrere Netze hinweg zu einem Empfänger zu transportieren. Die Übertragung erfolgt auf der Vermittlungsschicht, sie ist paketorientiert, verbindungslos und nicht garantiert. Die IP-Datagramme werden auch bei identischen Sendern und Empfängern vom IP-Protokoll als voneinander unabhängige Datenpakete transportiert. Das IP-Protokoll garantiert weder die Einhaltung einer bestimmten Reihenfolge noch eine Ablieferung beim Empfänger, d.h. Datagramme können z.B. wegen Netzüberlastung verloren gehen. Empfangsquittungen gibt es auf IP-Schicht nicht.
Die maximale Länge von IP-Datenpaketen ist auf 65.535 Bytes beschränkt. Da bestimmte Netze, ebenso wie einige Internetworking-Komponenten, diese Paketlänge nicht verarbeiten können, spezifiziert IP die Mindestpaketlänge mit 576 Bytes. Es kann also durchaus die Notwendigkeit bestehen, IP-Datenpakete in kleinere Datenpakete aufzuspalten, zu Fragmentieren. Das Wiederzusammensetzen nennt man Reassemblieren. Jedes IP-Fragment hat selbst wieder das Format eines gewöhnlichen IP-Paketes. Durch diese Maßnahme kann eine geforderte Transportleistung der Transportschicht an die zur Verfügung stehende Netzleistung angepasst werden. Das Transmission Control Protocol (TCP) und das User Datagram Protocol (UDP) setzen als Transportprotokolle auf dem IP-Protokoll auf.



Da das Internet-Protokoll auf der Vermittlungsschicht aufsetzt und diese für das IP-Routing durch ein Netzwerk zuständig ist, erlaubt das Protokoll auch die Identifikation von Stationen anhand der IP-Adresse. Damit die Datenpakete nicht unendlich durch das Netz irren und dieses verkehrsmäßig belasten, hat der IP-Header ein Time-to-Live-Feld (TTL), in dem die Lebensdauer eines Datagramms festgelegt wird. Dieses Datenfeld setzt die Zeitdauer fest, nach der ein Datenpaket verworfen wird.

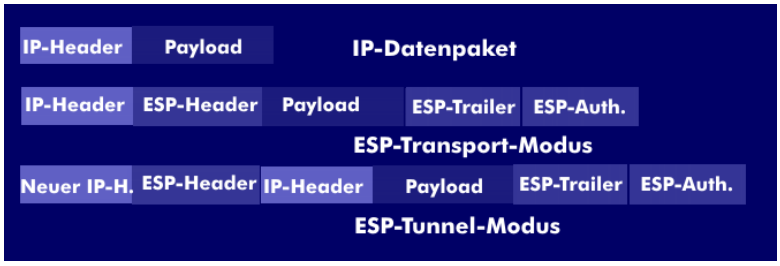
Es existieren verschiedene Netzwerkprotokolle, die jeweils im Zusammenhang mit einer ganzen Protokollfamilie entstanden sind. Die bekanntesten davon sind das Datagramm-orientierte Internet-Protokoll der TCP/IP-Protokolle; das Internetwork Datagram Protocol (IDP) von Xerox aus der XNS-Familie; das von der Open Systems Interconnection (OSI) standardisierte Protokoll ISO-IP sowie das Internetwork Packet Exchange Protocol (IPX) von Novell.

IPsec (IP security protocol)

IP Security Protocol (IPsec) ist ein Standardisierungsvorschlag der Internet Engineering Task Force (IETF), in dem Verfahren und Protokolle für einen herstellerübergreifenden sicheren und geschützten Datenaustausch mittels des IP-Protokolls festgelegt werden. Die Normungsaktivitäten laufen seit 1995. Der Normenrahmen von IPSec definiert die Vorgehensweise für die Datenintegrität, die Vertraulichkeit der Inhalte sowie die Verwaltung der kryptografischen Schlüssel. Die Bestandteile von IPSec sind der Authentication Header (AH), die Encapsulating-Security-Payload (ESP), die Security Association (SA), der Security-Parameter-Index (SPI) und der Internet Key Exchange (IKE).

Zu Beginn der Kommunikation wird zwischen den an der Kommunikation beteiligten Rechnern das benutzte Verfahren geklärt und ob die Datenübertragung im Tunneling erfolgen soll oder nicht. Daher unterscheidet das IPSec-Framework je nach Art der Verschlüsselung zwischen dem Authentication Header-basierten Transportmodus und dem ESP-basierten Tunnelmodus. Im ersten Fall, dem verschlüsselten Authentication Header (AH), bleibt der ursprüngliche IP-Header erhalten, die Quell- und Zieladressen bleiben ungeschützt. Der AH-Header liegt zwischen den IP-Headern und den Headern der Transportprotokolle. Die Authentisierung erfolgt entweder mittels des MD5-Algorithmus oder mit dem Secure Hash Algorithm (SHA), wobei die Authentisierung nur die Datenfelder des AH-Headers umfasst, die während der Übertragung unverändert bleibt. Dazu gehört der Nachrichteninhalt des AH-Headers, wodurch Veränderungen bemerkt werden können.

Im Gegensatz dazu bietet der Tunnelmodus, basierend auf der verschlüsselten ESP-Payload, eine höhere Sicherheit für das übertragene Datenpaket, da ja der gesamte Rahmen verschlüsselt wird. Das Datenpaket bekommt einen neuen Header in dem die Quell- und Zieladresse versteckt sind und nur die Tunnelendpunkte erkennbar.

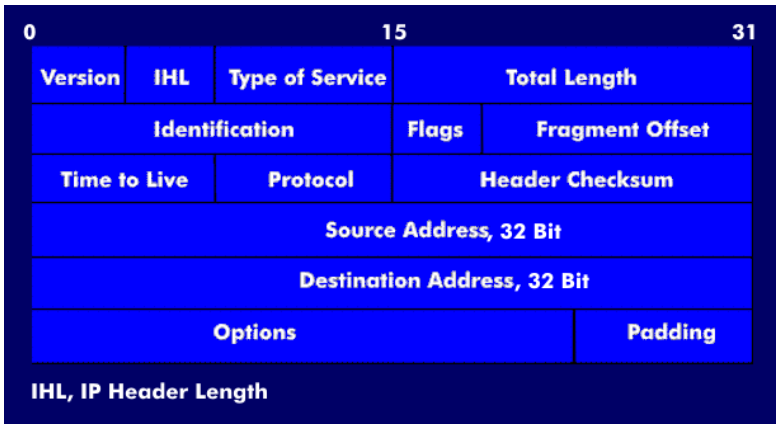


Das IPsec-Protokoll wurde speziell für die Verbindung zwischen zwei LANs entwickelt. Dabei schützt IPsec die Datenpakete des IP-Protokolls vor möglichen Modifikationen und vor Ausspähungen. IPsec beeinflusst weder die Kommunikationsprotokolle noch die Anwendungsprogramme, sodass das Internetworking über Router nicht beeinträchtigt wird. Authentisierungsverfahren, die mittels IPsec erstellt wurden, können zwischen den Daten von zugelassenen und nicht zugelassenen Kommunikationspartnern unterscheiden. Die Autorisierungsverfahren basieren auf den MD5-Hash-Algorithmen mit 128 Bits, die Verschlüsselung auf dem DES-Algorithmus mit 56 Bits in Cipher Block Chain (CBC).
Mit IPSec kann jeglicher IP-Datenverkehr geschützt werden: Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Internet Control Message Protocol (ICMP), Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP), Simple Network Management Protocol (SNMP), Telnet, usw. IPsec ist in den RFCs 1825 bis 1829 und 2401 beschrieben und standardisiert.

IPv4 (Internet protocol version 4)

Internet Protocol Version 4 (IPv4) wurde von der Internet Engineering Task Force (IETF) im Rahmen der IPnG-Aktivitäten entwickelt und bereits 1981 im RFC 791 beschrieben. Ziel dieser Aktivitäten war es, Folgeprotokolle für das klassische IP-Protokoll zu entwickeln, die einen wesentlich größeren Adressierungsbereich haben und gleichzeitig kompatibel mit dem IP-Protokoll sind. Diese Entwicklungen führten im Jahre 1995 zu der Standardisierung von IPv6, die die klassische IP-Version (IPv4) aus dem Jahre 1970 ergänzen sollte.
Die Version 4 hat einen Adressierungsbereich für die IPv4-Adresse von 32 Bit mit dem 4,3 Milliarden IPv4-Adressen adressiert werden können. Im Gegensatz dazu haben die IPv6-Adressen einen Adressierungsbereich von 128 Bit.
Der gesamte IPv4-Header entspricht dem klassischen IP-Header. Er umfasst 20 Oktette, 12 Felder inklusive 3 Flags und eine begrenzte Anzahl von Optionen. Er besteht aus einem 4 Bit langen Versionsfeld mit dem die Protokollversion gekennzeichnet wird, dem ebenfalls 4 Bit umfassenden IHL-Feld in dem die Anzahl der 32-Bit-Worte im IP-Header angegeben wird, dem 8 Bit langen Dienstleistungsfeld, Type of Service (ToS), in dem die Dienstleistung mit den Übertragungsparametern stehen, und dem 16 Bit langen Total-Length-Feld für die Gesamtlänge. In das Total-Length-Feld wird die Länge des gesamten Datagramms in Oktetten eingetragen, einschließlich des Headers und der Daten. Es folgt das Identifikator-Feld mit

16 Bit für die Identifizierung einzelner Datenpakete, das drei Bit lange Flag-Feld für die Anzeige der Fragmentierung, dem das Datenfeld Fragment Offset (13 Bit) folgt.



Das folgende Time-To-Live-Feld ist 8 Bit lang und sorgt für eine definierte Lebensdauer des Datenpaketes. Das Protokolltyp-Feld (8 Bit) kennzeichnet das nächst höhere Protokoll, das im Zusammenhang mit IPv4 benutzt wird. Wird das Internet Control Message Protocol (ICMP) benutzt, wird in das Datenfeld eine 1 eingetragen, beim Internet Group Management Protocol (IGMP) eine 4, beim TCP-Protokoll eine 6 und beim UDP-Protokoll die 17. Danach folgt das Header-Prüfsummen-Feld mit 16 Bit mit dem der IPv4-Header auf Fehler hin überprüft wird, das folgende Quelladressfeld und das Zieladressfeld haben jeweils 32 Bit. Abgeschlossen wird der IPv4-Header 24 Bit umfassenden Optionsfeld und dem Datenfeld für das Padding mit 8 Bit.

IPv4-Adresse

Durch die rasant steigende Anzahl an Internetteilnehmern war der Adressraum an klassischen IP-Adressen begrenzt, zumal man bei den IP-Adressen eine Klasseneinteilung in Netzwerk- und Hostadressen beachten musste. Diese feste IP-Klasseneinteilung hat die Flexibilität der Adressenvergabe stark eingeschränkt, wodurch ein Großteil der IP-Adressen nicht genutzt werden konnte. Sie wurde daher bereits 1993 aufgehoben und durch das Classless Interdomain Routing (CIDR), das mit Subnetzklassen arbeitet und eine effizientere Adressennutzung unterstützt, ersetzt.
Angewandt wird das CIDR-Verfahren in IPv4-Adressen und später auch in IPv6-Adressen. IPv4 hat dank des Classless Interdomain Routing einen theoretischen Adressenumfang von 4.294.967.296 IP-Adressen, der sich aus den Adressumfang von 32 Bit ergibt. Diese IPv4-Adressen können wie die klassischen IP-Adressen über die Subnetzmaske in eine Netzwerkkennung (net-id) und Hostkennung (host-id) aufgeteilt werden. Im Gegensatz zur klassischen IP-Adresse kann die Aufteilung nicht nur in einige wenige Klassen erfolgen, sondern durch die insgesamt 32 Subnetzmasken in 32 Klassen. In der umfangreichsten Klasse können alle Adressen (2exp32) als Hostadresse vergeben werden, wie im Internet, bei einem Suffix von /16, was der Class-B-Adresse der klassischen IP-Adresse entspricht, sind es 65.534 (2exp16) Hostadressen und 16.384 (2exp14) Netzwerkadressen. Die Subnetzmaske sieht bei einer IP-Adresse mit dem Suffix /16 folgendermaßen aus: 255.255.0.0.
Aus der klassenlosen IPv4-Adresse können durch die Subnetzmaske und die logische Verknüpfungen der IP-Adresse mit der Subnetzmaske die Netzwerkadresse und die Hostadresse ermittelt werden.

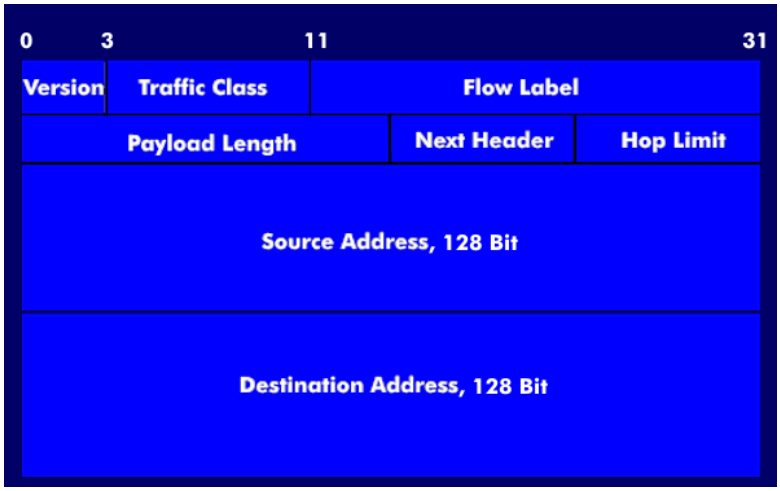
IPv4-Adresse: 168.122.34.3/24	Dezimal	Binär
IP-Adresse	168.122.34.3	10101000.01111010.00100010.00000011
Netzmaske	255.255.255.0	11111111.11111111.11111111.00000000
Netzwerkadresse*	168.122.34.0	10101000.01111010.00100010.00000000
Hostadressen*	3	00000000.00000000.00000000.00000011

* Netzwerkadresse durch ADD-Funktion von IP-Adresse und Netzmaske
* Hostadresse durch ADD-Funkion von IP-Adresse und negierter Netzmaske

Obwohl das CIDR-Verfahren bei den IPv4-Adressen eine wesentlich höhere Flexibilität hat, war auch dieser Adressraum der Dynamik des Internet nicht gewachsen, so dass man mit IPv6 ein Adressierungskonzept entwickelte, dass einen Adressraum von 128 Bit abdeckt, was einer 39-stelligen Dezimalzahl entspricht.

IPv6 (Internet protocol version 6)

IPv6 (Internet Protocol Version 6) ist eine von der Internet Engineering Task Force (IETF) erarbeitete IP-Protokollversion, die als Ergebnis der IPnG-Arbeiten im Dezember 1995 in den Internet-Standard übernommen wurde. Die Version 6 des IP-Protokolls ist eine kontinuierliche Weiterentwicklung des IPv4-Protokolls.
Die wesentlichen Unterschiede zwischen den beiden Versionen liegen in den wesentlich erweiterten Adressierungsmöglichkeiten von IPv6, der Vereinfachung des Headerformats, der verbesserten Unterstützung von Optionen und Erweiterungen, in neuen Möglichkeiten der Dienstgüte, der Vereinfachung des Routing und nicht zuletzt in den verbesserten Sicherheitsaspekten.
Der IPv6-Header besteht aus einem 4 Bit langen Versionsfeld für die IP-Version, einem 8 Bit langen Prioritätsfeld (Traffic Class) für die Dienstgüte und einem 20 Bit langen Flow-Label-Feld, das von der Datenquelle zur Kennzeichnung von Datenpaketen verwendet wird, die ein spezielles Handling mit einer bestimmter Dienstgüte erfahren sollen. Darüber hinaus enthält der IPv6-Header das 16 Bit lange Payload-Length-Feld, das Next-Header-Feld (8 Bit) für die Eintragung des Folgeprotokolls, das 8 Bit lange Feld für das Hop-Limit-Feld, das dem Time-to-Live-Feld in IPv4 entspricht und dem jeweils 128 Bit langen Quelladressfeld und ebenso langen Zieladressfeld.



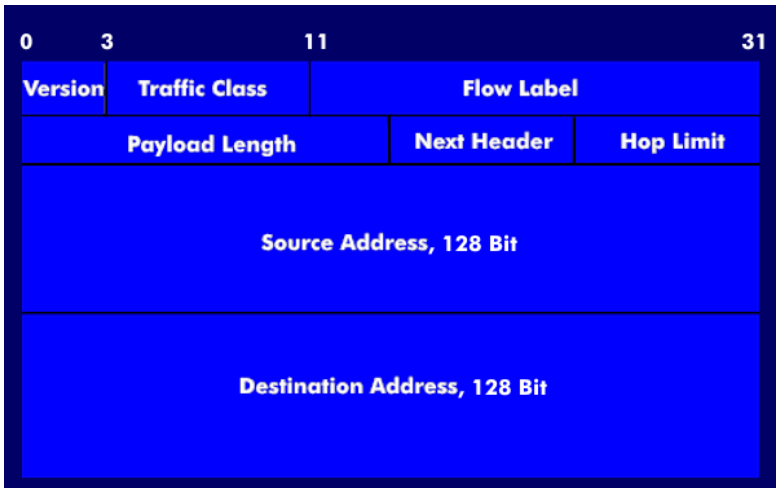
Bedingt durch den auf 128 Bits erweiterten Adressraum kann eine schier unendliche Anzahl an adressierbaren Knoten angesprochen werden, die Adressenanzahl entspricht der einer 39-stelligen Dezimalzahl. Die einfachste Form einer IPv6-Adresse umfasst als Knotenadresse 128 Bits. Mittels eines Format-Präfixes (FP) sind bestimmte Adressformate voreingestellt. So die Formate für Unicast, Multicast und für einige protokollabhängige Adresserweiterungen.

Prioritätsklassen	Datenverkehr
0	Nicht charakterisiert
1	Verkehrsschwacher Datenverkehr
2	Hintergrundverkehr
3	Reserviert
4	Massendatenverkehr wie FTP
5	Reserviert
6	Interaktiver Datenverkehr wie Telnet
7	Steuerungsverkehr für Switches/Router

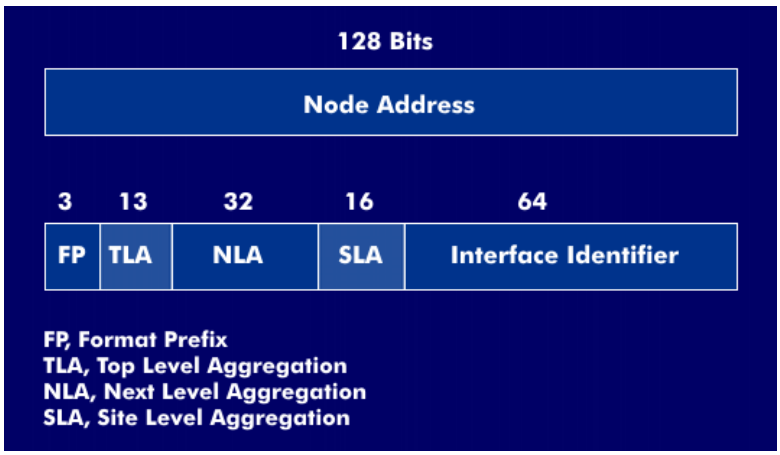
Um eine optimale Durchleitung der Datenpakete sowie eine gesonderte Bearbeitung der Datagramme durch das Netz zu gewährleisten, gibt es für IPv4 Optionen und für IPv6 optionale Erweiterungs-Header. Es handelt sich dabei um Header deren Reihenfolge festgelegt ist, die miteinander verkettet werden und dem IPv6-Header folgen. An Erweiterungs-Headern gibt es den Hop-by-Hop-Options-Header, Routing-Header, Fragment Header, Authentication Header, Privacy Header und den Destination-Options-Header. Erweitert werden kann der IPv6-Header um den Header von Internet Control Message Protocol, ICMPv6, für Fehlermeldungen und -behandlungen.
Bei der Migration vom IPv4-Protokoll auf das IPv6-Protokoll werden beide Internet-Protokolle zuerst parallel auf allen Netzknoten betrieben. Diese Betriebsart nennt sich Dual Stack und wird solange beibehalten, bis auf das IPv4-Protokoll verzichtet werden kann. Der Status von IPv6 wird u.a in den RFCs 1881, 1887, 2375 und 2462 beschrieben, die Adressstruktur in RFC 2373 und RFC 2374.

IPv6-Adresse

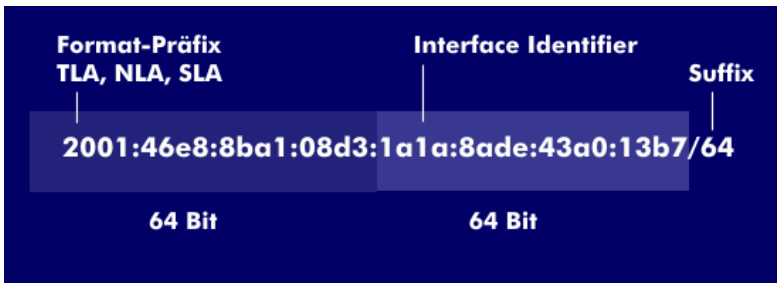
Die IPv6-Adresse unterscheidet sich im Wesentlichen durch den enormen Adressraum von 128 Bit von der IPv4-Adresse, die einen Adressenumfang von 32 Bit hat. Bedingt durch diesen erweiterten Adressraum kann eine schier unendliche Anzahl an adressierbaren Knoten angesprochen werden, die Adressenanzahl entspricht der einer 39-stelligen Dezimalzahl.
Anders als bei der IPv4-Adresse werden die IPv6-Adressen nicht in vier Dezimalzahlen dargestellt, sondern mit acht alphanumerischen Kombinationen des Hexadezimalsystems. Statt der Trennpunkte werden die Zahlengruppen der IPv6-Adresse durch Doppelpunkte getrennt und führende Nullen werden unterdrückt. So stellt sich eine IPv6-Adresse folgendermaßen dar: 221:4033:a450:fb3:8a8b:56:ad:1234.



Damit man in diesem Adressierungsbereich nicht den Überblick verliert, werden die 128 Bit in acht Blöcken zu jeweils 16 Bit unterteilt. Die ersten vier Blöcke, also 64 Bit, werden für das Routing genutzt und bezeichnen den Netz-Präfix, die folgenden 64 Bit kennzeichnen die Interface Identifier (Interface ID) für die Schnittstellenkennzeichnung in Subnetzen.



Die einfachste Form einer IPv6-Adresse umfasst als Knotenadresse 128 Bits. Mittels eines Format-Präfixes (FP) sind bestimmte Adressformate voreingestellt. So die Formate für Unicast, Multicast und für einige protokollabhängige Adresserweiterungen. Die Adressstruktur ist geprägt durch mehrere Levels, die die Adressen von Areas, Subnetzen und Interfaces repräsentieren. Durch diese Hierarchie-Level oder -Layer werden das Routing und die Autokonfiguration von Adressen wesentlich vereinfacht.
Benötigen die Router im Internet-Backbone mit IPv4 noch die komplette IP-Adressierung und alle Routen, so reduziert sich der Aufwand in IPv6 auf den jeweils relevanten Adressteil, der in der jeweiligen Level Aggregation, der Top Level Aggregation (TLA), der Next Level Aggregation (NLA) und der Site Level Aggregation (SLA), eingetragen ist. Diese Aggregationsfelder belegen zusammen mit dem Format-Präfix 64 Bit der IPv6-Adresse. Die restlichen 64 Bit werden von dem Interface Identifier belegt, einer weltweit eindeutigen Identifizierung von Knoten und Schnittstellen, der im Extended Unique Identifier (EUI) formatiert ist.



Die Notation der IPv6-Adresse erfolgt wie bei der IPv4-Adresse im Classless Interdomain Routing (CIDR) mit einem Suffix, dessen Wert mit einem Schrägstrich getrennt wird und der die Anzahl der Subnetzklassen bestimmt.

ISO Open Systems Interconnection (OSI)

Open Systems Interconnection (OSI) ist eine von der internationalen Standardisierungs-Organisation (ISO) ins Leben gerufene Inittiative für die herstellerübergreifende Kommunikation von Systemen, Geräten und Komponenten. Die offene Kommunikation beschreibt international vereinbarte Standards, mit denen offene Systeme arbeiten, und definiert die Regeln für die Implementierung dieser Normen.
Als offene Kommunikationssysteme bezeichnet man eine Kombination von Netzwerk-Hardware sowie Netzwerk- und Systemsoftware, die den uneingeschränkten Informationsaustausch zwischen diesen Geräten auf der Basis gemeinsamer Protokollvereinbarungen und Schnittstellen unabhängig von der sonstigen Bauart und Ausstattung dieser Geräte erlaubt. Systeme, die OSI-Protokolle implementieren, sind ein Beispiel hierfür. Die Basis für die offene Kommunikation wurde mit dem OSI-Referenzmodell gelegt. *Siehe auch*: *ISO (S. G35)*.

ISP (Internet service provider)

Internet Service Provider (ISP) sind Firmen, Unternehmen oder Organisationen, die spezielle Dienste, Inhalte oder Hosting für den Internetzugang, die Präsentation und den Betrieb von Websites anbieten. Sie sind, was den Internetzugang betrifft, vergleichbar den Internet Access Providern (IAP), bieten in der Regel darüber hinaus aber weitere Internet-Services an.

ISPs unterscheiden sich in Bezug auf die angebotene Leistungen und die damit in Zusammenhang stehende technische Infrastruktur und Netzinfrastruktur. So bieten einige ISPs Dienstleistungen im Anschlussbereich an, andere im Backbone-Bereich, die Backbone-ISPs. Zur Unterschiedung der Größe ihrer Netzstruktur, werden Internet Service Provider in Rangordnungen, im Englischen: Tier, eingeteilt. Es gibt drei Tier-ISP-Ränge: Tier-1-ISPs betreiben große IP-basierte Netze, die weltweit mit anderen Internet Regions verbunden sind und jedes andere Netzwerk via Internet erreichen können, ohne dass sie Traffic hinzukaufen müssen. Da es im Internet keine Institution gibt, die die Tier-Klassen definiert, werden die größten ISPs - in Deutschland die Deutsche Telekom, in den USA sind es AT&T, Verizon, Sprint u.a. - als Tier-1-ISPs bezeichnet.

Tier-2-ISPs sind kleinere Internet Service Provider, die einen IP-Backbone zur Verfügung stellen, der mit den Tier-1-Netzen verbunden ist. Sie haben Peering-Vereinbarungen für den IP-Transit. Tier-3-ISPs sind kleine regionale Provider. Sie sorgen für den Internetanschluss und arbeiten ebenfalls mit Peering-Abkommen für die Transitkosten.

Internet Access Provider (IAP) stellen mit ihren Vermittlungsknoten, den Points of Presence (PoP), Zugangsknoten für das Internet zur Verfügung, an die der Kunde mittels Festverbindungen oder Modem-Strecken seine Rechner anbinden kann. Je nach Einzugsgebiet des Knotenrechners unterscheidet man den regionalen und nationalen Internet Service Provider durch die Tier-Ränge. Die Backbone-ISPs bieten Dienstleistungen für die regionalen und nationalen Internet Service Providern. Sie verfügen über Hochgeschwindigkeitsinfrastrukturen und unterstützen den Datenverkehr über größere Entfernungen. Sie sind über Internet-Knoten, den Internet Exchange (IX) oder den Internet Exchange Points (IXP) mit dem internationalen Carriern verbunden.

Der Begriff Internet Service Provider (ISP) wird häufig synonym zu Online Service Provider (OSP) verwendet, obwohl es zwischen beiden Dienstanbietern Unterschiede gibt.

ITU Telecommunication Standardization Sector (ITU-T)

Die meisten Empfehlungen (Standards) werden innerhalb der ITU von der ITU-T (Telecommunication Standardization Sector) verabschiedet. Diese Empfehlungen werden im Gegensatz zu nationalen Normen wie DIN, RS oder ANSI weltweit anerkannt. *Siehe auch*: *ITU (S. G34)*.

Jitter

Jitter ist eine zufallsbedingte zeitliche oder amplitudenmäßige Schwankung der Signalfrequenz, dessen Phasenlage oder Pegel. Die Internationale Fernmeldeunion (ITU) definiert Jitter als: "Kurzzeitige Schwankungen der signifikanten Zeitpunkte eines Taktsignals, abweichend von ihren idealen zeitlichen Positionen".

Mit Jitter bezeichnet man in der Datenübertragung die Phasenschwankungen und damit die zeitlichen Änderungen von Signalfrequenzen. Es handelt sich um Schwankungen von fixierten Zeitpunkten z.B. dem Zeitpunkt des Übergangs von einer Signalamplitude eines Digitalsignals auf eine andere. Dadurch können sich Bezugszeitpunkte verschieben, was zu Fehlinterpretationen der Signale, zu Paketverlusten und damit zur Verschlechterung der Übertragungsqualität bei Echtzeitanwendungen führen kann.

Jitter tritt speziell bei hohen Frequenzen auf, es wird durch Rauschen und Übersprechen, durch Einstreuungen, Flankenverzerrungen und minimale Pegelschwankungen verursacht und kann mit dem Augendiagramm gemessen werden. Angegeben wird der Jitter in Parts per Million (ppm) oder in Nanosekunden (ns). Die Jittermessung ist von der Audio Engineering Society (AES) unter dem Standard AES17 standardisiert worden.

Beim Jitter differenziert man zwischen dem zufälligem Jitter, Random Jitter (RJ), dem totalen Jitter, Total Jitter (TJ), dem deterministischen Jitter, deterministic Jitter (DJ) und dem nicht-korrelierten Jitter, Bounded Uncorrelated Jitter (BUJ). Es gibt diverse Methoden um Jitter zu kompensieren indem man senderseitig Flankenverzerrungen vorkompensiert oder empfangsseitig den Jitter durch mathematische Algorithmen korrigiert.
Damit ist der Jitter ein Maß für die Variablilität der Latenz.

Kanalpuffergröße (Bits)

Die Kanalpuffergröße in Bits gibt die Anzahl an **bereits gesendeten** Bits an, während sich das erste Bit vom Sender zum Empfänger ausbreitet. Als Formel ergibt sich:

$$R \cdot D = R \cdot \frac{l}{v} = \frac{D}{1/R} = \frac{l/v}{1/R} = \frac{\textit{Ausbreitungsverzögerung}}{\textit{Bitsendezeit}}.$$

Ist $RD > 1$, so erreicht zum Zeitpunkt $t = D$ das erste Bit den Empfänger, wobei $R \cdot D$ Bits bereits gesendet sind. Ist andererseits $RD < 1$, so erreicht zum Zeitpunkt $t = D$ der Anfang des ersten Bits den Empfänger, und es sind $R \cdot D \cdot 100\%$ des ersten Bits bereits versendet. Erst zum Zeitpunkt $t = 1/R$ verlässt das Ende des ersten Bits den Sender, dieses erreicht den Empfänger dann zum Zeitpunkt $t = 1/R + D$.

Kanalpuffergröße (Paket)

Die Kanalpuffergröße in Paketen gibt die Anzahl an **bereits gesendeten *Paketen*** an, während sich das erste Bit vom Sender zum Empfänger ausbreitet. Es ergibt sich mit einer Paketgröße L:

$$a = \frac{R \cdot D}{L} = \frac{l/v}{L/R} = \frac{\textit{Ausbreitungsverzögerung}}{\textbf{\textit{Paket}}\textit{sendezeit}}$$

Ist $a > 1$, so erreicht zum Zeitpunkt $t = D$ das erste Bit den Empfänger, wobei a Pakete bereits gesendet sind. Ist andererseits $a < 1$, so erreicht zum Zeitpunkt $t = D$ der Anfang des ersten Bits den Empfänger, und es sind $a \cdot 100\%$ des ersten Pakets bereits versendet. Erst zum Zeitpunkt $t = L/R$ verlässt das Ende des ersten Pakets den Sender, dieses erreicht den Empfänger dann zum Zeitpunkt $t = L/R + D$.

Kommunikationsart

In einem Kommunikationsnetzwerk werden Endgeräte und Netzknoten, sprich Mitglieder des Netzwerks, über verschiedene Verbindungstypen miteinander in Verbindung gebracht. Man unterscheidet an dieser Stelle zwischen physikalischen Verbindungen und logischen Verbindungen. Dabei können über eine physische mehrere logische Verbindungen aufgebaut werden. Diese Verbindungen können dann wieder über ein leitungsgebundenes oder drahtloses Übertragungsmedium aufgebaut werden.

Konfigurationsmanagement (*configuration*)

Das Konfigurationsmanagement, Configuration Management (CM), ist ein Funktionsbereich des OSI-Managements und des Fault, Configuration, Account, Performance, and Security Management (FCAPS). Es dient im weitesten Sinne dazu, das gesamte vernetzte und verteilte System verfügbar zu machen, während das Fehler- und Leistungsmanagement dazu dient, das System in einem verfügbaren Zustand zu halten bzw. diesen wiederherzustellen.
Wichtige Aufgabenbereiche des Konfigurationsmanagements sind die Bestandsführung in Form der Erfassung der bestehenden Software-Konfigurationen und die Softwareverteilung. Des Weiteren die Verwaltung von Änderungen bei dem ermittelt wird, welche Veränderungen durch die Softwareverteilung entstehen. Darüber hinaus die Dokumentation der Installationen und die Verzeichnisdienste.
Zu den Aufgaben des Konfigurationsmanagements gehören weiterhin die Definition von Betriebsmitteln, das Initialisieren und Terminieren derselben, das Einstellen und Ändern von Parametern sowie das Sammeln von Zustandsdaten und das Sichern des Normalbetriebs. Je nach Netzgröße und Systemvielfalt wird man unterschiedliche Anforderungen an das

Konfigurationsmanagement stellen. Dieser Managementbereich bietet heute die meisten Tools. *Siehe auch*: *FCAPS (S. G20)*.

Latenz

Der Begriff Latenz wird synonym für Verzögerungszeit verwendet. Im Allgemeinen handelt es sich um das Zeitintervall vom Ende eines Ereignisses oder Befehls bis zum Beginn der Reaktion auf dieses Ereignis oder wie bei der Computational Latency um das Ergebnis der Aufgabe. Hier bezeichnet es die Zeit bis zu dem ein Paket den Empfänger erreicht hat.

Leitungsgebundenes Übertragungsmedium

Man unterscheidet hierbei zwischen metallischen und nichtmetallischen Leitern. Die Bitraten reichen hierbei von einigen Kbps bis mehreren Gbps und die Signalausbreitungsgeschwindigkeit ist etwas geringer als die Lichtgeschwindigkeit. Zudem treten nur kleine Bitfehlerraten (10^{-10}) auf.
Beispiele für leitergebundene Kabel sind symmetrische Kupferkabel wie die Kupferdoppelader (CuDa), die im Anschlussbereich eingesetzt werden oder Koaxialkabel (COAX) und die verdrillten Kupferkabel. Ebenso leitergebunden sind auch Lichtwellenleiter in Form von Glas- oder Plastikfasern. *Siehe auch*: *Übertragungsmedium (S. G83)*.

Leistungsmanagement (*performance*)

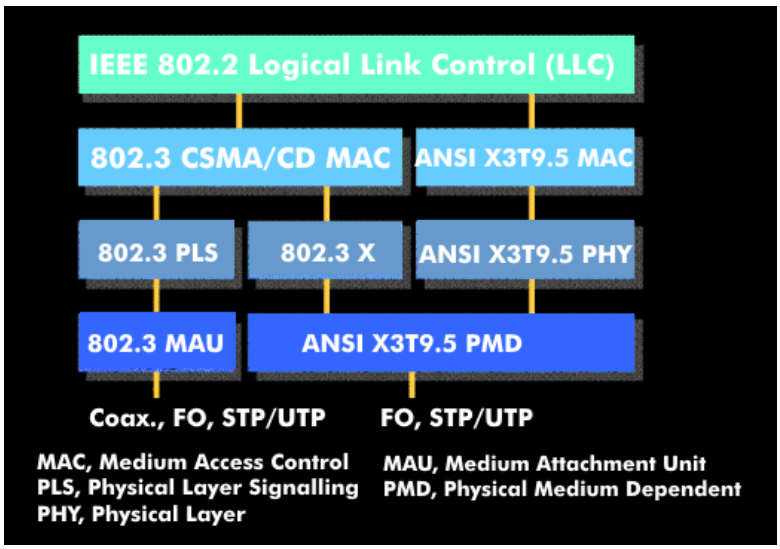
Das Leistungsmanagement oder Performance Management (PM) ist ein Funktionsbereich des OSI-Managements und des Fault, Configuration, Account, Performance, and Security Management (FCAP), zur Optimierung der Leistungsfähigkeit eines Netzes.
Das Leistungsmanagement ermittelt die Systembelastung und zeigt Leistungsengpässe an, es steht in direktem Zusammenhang mit der Netzwerkauslegung, der Netzwerkerweiterung und dem Fehlermanagement. Wichtige Parameter für das Leistungsmanagement sind das Verhalten der Antwortzeit, die Verweilzeit und die Verzögerungszeit, die theoretischen Leistungsgrenzen und die Netzlast. Diese Parameter werden durch viele übertragungstechnische Eigenschaften beeinflusst, so beispielsweise von der Flusskontrolle, den Zugangsverfahren, dem Dämpfungsverhalten oder den Paketverlusten. Das Leistungsmanagement gibt Aufschluss über die Netzauslastung und dient zur Ermittlung der Leistungstrends für die weitere Netzplanung.
Zu den Aufgabenbereichen gehören das Sammeln statistischer Daten sowie die Aufzeichnung der Ereignisse zur Bewertung und Verbesserung des Leistungsverhaltens von Betriebsmitteln. *Siehe auch*: *FCAPS (S. G20)*.

Leitungsvermittlung

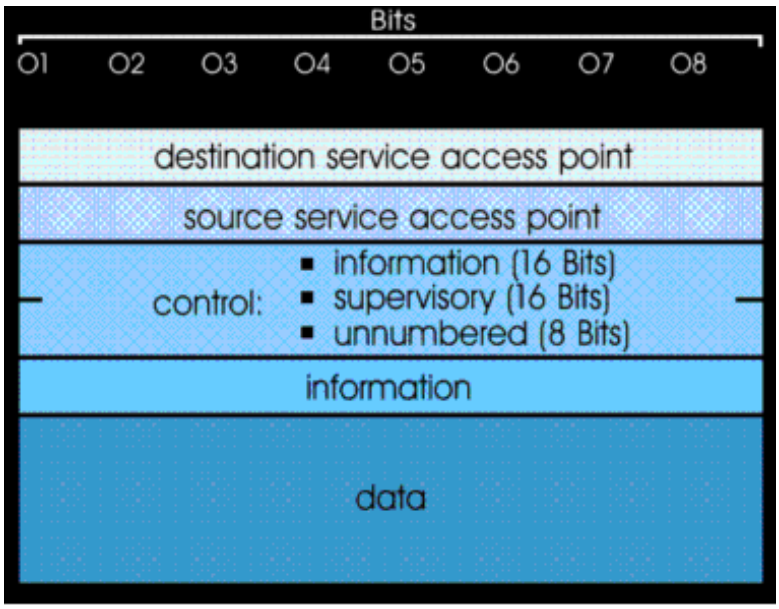
Bei der Leitungsvermittlung wird zwischen Sender und Empfänger eine physikalische Leitung geschalten. Es wird damit durch beispielsweise Frequenz- oder *Siehe auch*: *FDM (S. G25)*, *TDM (S. G92) und Vermittlungsart (S. G88)*.

LLC (logical link control)

Logical Link Control (LLC) ist ein OSI-Protokoll, das von der IEEE-Arbeitsgruppe 802 entwickelt wurde und für alle LAN-Subsysteme im Rahmen des Standards IEEE 802 gleich ist. Es handelt sich um die Steuerung der Datenübertragung auf der oberen Teilschicht der Sicherungsschicht, die im Ethernet-Schichtenmodell in die Sublayers Logical Link Control und Medium Access Control (MAC) unterteilt wurde.
IEEE 802.2-LLC umfasst die Adressierung der Endsysteme sowie die Fehlerprüfung. Entsprechend ihrer architektonischen Einbettung besteht die LLC-Spezifikation aus der Teilnehmerschnittstelle, der LLC-Protokollspezifikation und der MAC-Schnittstelle. Die Teilnehmerschnittstelle beschreibt die Dienste, die die LLC-Schicht von der unterhalb liegenden MAC-Teilschicht anfordern kann.



Logical Link Control kennt vier Dienstformen: Typ 1 kennzeichnet einen verbindungslosen, ungesicherten Datagrammdienst. Typ 2 kennzeichnet einen verbindungsorientierten Dienst. Typ 3 bezeichnet einen verbindungslosen Datagrammdienst mit Bestätigung und Typ 4 für Punkt-zu-Punkt-Verbindungen.
Wird der LLC-Typ 1, der Unacknowledged Connectionless Mode Service benutzt, wird eine von der Vermittlungsschicht an LLC übergebene Dateneinheit einfach auf das Übertragungsmedium gegeben, unabhängig davon, ob die Empfangsstation empfangsbereit ist oder vorhergehende Datenpakete akzeptiert hat. Der LLC-Typ-1-Dienst vertraut auf das Vorhandensein geeigneter Software höherer Schichten die Vollständigkeit, Fehlerfreiheit, Reihenfolgeerhalt von Datenpaketen u.ä. sicherstellt.
Beim LLC-Typ-2-Dienst, dem Connection Mode Service, kann ein Teil dieser Aufgaben in der LLC-Teilschicht erledigt werden, da eine logische Verbindung explizit aufgebaut wird und ein verbindungsbezogener Status gehalten werden kann. Der Dienst trennt die Phasen Verbindungsaufbau, Datentransfer, Verbindungsabbau.
Mit dem Typ-3-Dienst, dem Acknowledged Connectionless Mode Service, kann ein einfaches Polling anderer LAN-Stationen erzielt werden sowie eine Quittierung von Sendungen über das LAN hinweg, ohne die Komplexität eines verbindungsorientierten Status zu haben.
Der LLC-Typ 4 wurde speziell für sehr schnelle Punkt-zu-Punkt-Verbindungen entwickelt. Die Vollduplex-Verbindung ermöglicht zwei völlig voneinander unabhängige Verbindungen.
Der Standard IEEE 802.2 umfasst für jedes Dienstprimitiv die Festlegungen bezüglich der allgemeinen Funktionen, die Parameterliste, die Semantik, den zulässigen Aufrufzeitpunkt, die ausgelöste Wirkung beim Empfänger und gegebenenfalls weitere Bemerkungen. Das LLC-Protokoll lehnt sich an das bitorientierte HDLC-Protokoll an, das z.B. als Leitungsprozedur bei X.25 Verwendung findet.



Der Aufbau der Steuerinformation des Control-Feldes und die Verwendung der LLC-Frame-Typen, d.h. der Schicht-2b-Protokollelemente, ist im wesentlichen identisch dem bei HDLC, mit folgenden Ausnahmen: Logical Link Control benutzt nur den Asynchronous Balanced Mode (ABM), geht also nicht von unsymmetrischen Konfigurationen aus, wie beim Normal Response Mode (NRM) unterstellt; somit kann jede Station Leitstation (Primary Node) sein. LLC unterstützt einen Datagramm-Dienst durch Nutzung des Unnumbered Information Frame.

LLC erlaubt das Multiplexen auf der Schicht 2 dadurch, dass pro Station mehrere LLC-Dienstzugangspunkte zugelassen sind. Dies unterstützt z.B. den Anschluss von Terminalservern am lokalen Netz (LAN), die Adressenfelder haben zwar die gleiche Länge wie bei HDLC, sind aber anders codiert. Das erste Bit im Zieladressfeld entscheidet, ob es sich um eine Individual- oder Gruppenadresse handelt. Das erste Bit im Quellenadressfeld gestattet die Unterscheidung von Kommandos und Antworten. LLC benutzt eine 32 Bit lange zyklische Redundanzprüfsumme. Damit sinkt auch die Restfehlerwahrscheinlichkeit für verfälschte, aber nicht als solche erkannte Bits bei einer Übertragung in die Größenordnung von 2exp-32. LLC sieht im Addendum 1 zu 802.2 eine Flusskontrolle über eine dynamische Änderung der Fenstergröße vor.

lokales Netz (LAN)

Bei LANs handelt es sich um Systeme für den Hochleistungsinformationstransfer, welche es einer Anzahl *gleichberechtigter Benutzer* ermöglichen auf einem *räumlich begrenzten Gebiet* partnerschaftlich orientierten Nachrichtenaustausch hoher Güte durchzuführen. Die Ausdehnung ist üblicherweise höchstens 10 km, es mag aber auch LANs geben mit höherer Ausdehnung. Es können hierbei beispielsweise Ausbreitungsverzögerungen von ungefähr $2^1/_2 \, {}^{\mathrm{km}}/_{\nu} \approx 12^1/_2 \, \mu\mathrm{s}$ erreicht werden. LANs können drahtgebunden – wie standardisierte lokale Netze Ethernet, Token Ring und FDDI – als auch drahtlos – wie WLANs nach 802.11 – arbeiten.

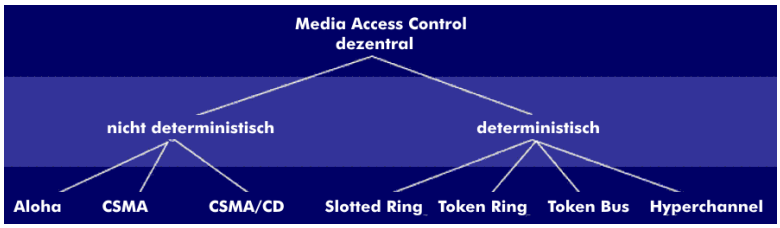
LSP (label switched path)

Der Label Switched Path (LSP) ist ein feststehender Weg durch ein MPLS-Netz, Multi-Protocol Label Switching (MPLS), der beim Ingress Label Edge Router (LER) beginnt und beim Egress Label Edge Router endet. Auf diesem Pfad werden die Datenpakete mittels Label-Switching von einem Label Switch Router (LSR) zum nächsten Label Switch Router weitergeschaltet. Der Pfad wird zuerst über das Label Distribution Protocol (LDP) oder das Resource Reservation Protocol (RSVP) aufgebaut und anschließend von den IP-Paketen genutzt. Die Weiterleitung der Datenpakete erfolgt über das Label selbst und nicht mehr über die IP-Adresse.

Innerhalb der Label Switched Paths stellt Multi-Protocol Label Switching (MPLS) Mechanismen für das Traffic-Engineering (TE) und die Dienstgüte (QoS) zur Verfügung. Der Label Switched Path kann als flexibler Tunnel betrachtet werden, über den ein Bandbreitensharing möglich ist. Die Bandbreite des Tunnels wird durch die Committed Information Rate (CIR) und die Peak Information Rate (PIR) festgelegt. Außerdem kann die Bandbreite dynamisch zugeordnet werden, ebenso können die Mindestbandbreite und die Bandbreite für die Spitzenlast konfiguriert werden.

MAC (medium access control)

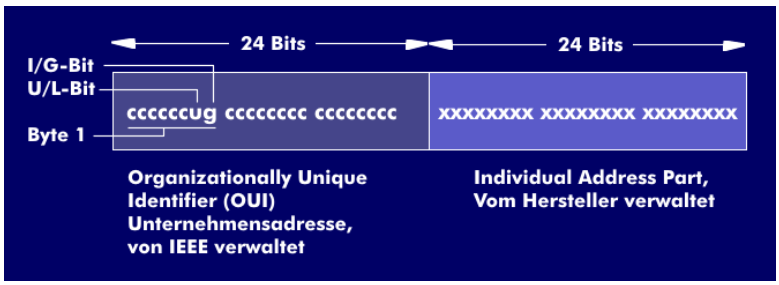
Im LAN-Schichtenmodell ist die Sicherungsschicht in zwei Teilschichten untergliedert: Medium Access Control (MAC) und Logical Link Control (LLC). Auf dem Medium Access Control (MAC) werden die medienspezifischen Zugangsverfahren realisiert mit denen die angeschlossenen Stationen auf das Medium zugreifen.
In IEEE 802 sind die Zugangsverfahren für lokale Netze spezifiziert. Es gibt deterministische, kollisionsfreie Reservierungsverfahren wie Token Ring und stochastische, kollisionsbehaftete Zugangsverfahren wie CSMA/CD. Während Token Ring als Zugangsverfahren das Token-Verfahren benutzt, arbeitet Ethernet mit dem nicht-deterministischen CSMA/CD mit Kollisionserkennung.



Die Medienzugangsverfahren umfassen die Frame-Aufbereitung beim Senden (Encapsulation) und Empfangen (Decapsulation), die Frame-Übergabe an das Zugangsmanagement (Senderseite), das Zugangsmanagement und die Fehlerkorrektur. Die Arbeitsweise der MAC-Schicht basiert auf der Kooperation unterschiedlicher Prozesse. Die Dienste, die die MAC-Schicht der LLC-Schicht anbietet, sind Transmit Frame und Receive Frame; die Dienste der MAC-Schicht der ihr untergeordneten Bitübertragungsschicht zur Verfügung stellt, sind Receive Bit, Transmit Bit und Wait.

MAC-Adresse

Die MAC-Adresse, auch als Ethernet-Adresse bezeichnet, ist eine eindeutige, unverwechselbare Adresse für alle Workstations, Server oder Rechner, die über Ethernet-Adapterkarten an ein Ethernet angeschlossen sind. Es handelt sich um eine 48 Bit lange Hardware-Adresse, die zur eindeutigen Identifikation eines Knotens im Netzwerk dient. Die MAC-Adresse besteht aus einer Herstellerkennung und einer Adapterkennung. Die MAC-Adresse ist Bestandteil der Netzwerkkarte, sie wird in einem PROM gespeichert, kann nicht verändert werden und wird in der Regel auf diese aufgedruckt.
Die 24 Bit umfassende Herstellerkennung der MAC-Adresse, das ist der Organization Unique Identifier (OUI), wird von der IEEE vergeben und verwaltet. Jeder Hersteller hat seinen eigenen OUI-Code. Neue Hersteller werden von IEEE registriert und können ihre Adresszuordnungen abfragen. Ein Teil des nachfolgenden Adressbereichs kann für Privatpersonen oder kleine Firmen vergeben werden. Dieser 12 Bit umfassende Adressbereich ist der Individual Address Block (IAB).



Das niederwertigste Bit (LSB) des ersten Byte der Herstellerkennung dient der Unterscheidung zwischen einer individuellen Adresse und einer Gruppenadresse, Individual/Group (I/G). Wird das I/G-Bit auf 0 gesetzt, handelt es sich um eine Individualadresse für Unicast, wird das I/G-Bit auf 1 gesetzt, ist es eine Multicastadresse.

MAC-Adresse	Funktion
Byte 1 LSB: I/G-Bit 7. Bit: U/L-Bit Bit 1 bis 6	 I/G = 0, Individual, I/G = 1, Group U/L = 0, Universal, U/L 0 1, Local OUI-Kennzeichnung
Byte 2 und 3	Organizationally Unique Identifier (OUI)
Byte 4 bis 6	Individual Address Part

Das siebte Bit ist das U/L-Bit, was für Universal/Local steht. Mit ihm wird festgelegt, ob es sich um eine universelle oder um eine lokal verwaltete MAC-Adresse handelt, wobei die lokal vergebenen Adressen nicht an einer netzübergreifenden Kommunikation beteiligt sein dürfen. Wird das U/L-Bit auf 0 gesetzt hat die Adresse eine eindeutige Firmenkennung. Es handelt sich dann um eine Universally Administered Addresses (UAA), die vom IEEE verwaltet wird. Bei einer 0 wird die Adresse lokal verwaltet. Es ist dann eine Locally Administered Address (LAA). Die weiteren Bits des ersten Byte dienen ebenso wie die restlichen 3 Byte der Adapterkennung. Wobei die zweiten 24 Bits vom jeweiligen Hersteller vergeben und verwaltet.

Um weltweit eine eindeutige Adresszuordnung zu gewährleisten hat IEEE mit dem Extended Unique Identifier (EUI) den Adressbereich für die MAC-Adressen von 48 Bit auf 64 Bit erweitert. Das entsprechende Format nennt sich EUI-64.

MDA (message delivery agent)

Ein Message Delivery Agent (MDA) oder auch Mail Delivery Agent ist ein E-Mail-Client in einem E-Mail-System. Der Message Delivery Agent hat Zustellfunktionen, die der Mail Transfer Agent (MTA) dazu benutzt, um Nachrichten in den Mailboxen der Message User Agents (MUA) abzulegen.

Der E-Mail-Client erstellt, empfängt und liest die Nachrichten und überträgt sie mittels Simple Mail Transfer Protocol (SMTP) über das Internet. Die Nachrichten vom Message User Agent (MUA) werden mittels Post Office Protocol (POP) oder Internet Message Access Protocol (IMAP) übertragen und vom MDA-Agent an den Message Transfer Agent (MTA) weitergeleitet. Empfangsseitig stellt der MDA-Agent die E-Mails auf dem lokalen System zu.

MIB (management information base)

Die Management Information Base (MIB) ist eine Datenbank, die alle für ein Management-System relevante Daten in relationaler oder objektorientierter Form enthält. Der Begriff MIB wird meist im Zusammenhang mit dem Simple Network Management Protocol (SNMP) benutzt. Die SNMP-MIB ist ein einheitlicher, hierarchisch aufgebauter, protokollunabhängiger Raum für Datenobjekte. Die MIB enthält eine Vielzahl an Managed

Objects (MO), die in Gruppen zusammengefasst sind. Diese Gruppen erfüllen alle Anforderungen an die Knotenrechner im Internet.
Alle Objekte in einer Management Information Base werden einheitlich in Abstract Syntax Notation One (ASN.1), der Abstract Syntax Notation One, die ursprünglich für die Definition abstrakter Transfersyntaxen in der Darstellungsschicht des OSI-Referenzmodells entworfen wurde, formuliert, wodurch eine Normung der abstrakten Darstellungen gegeben ist. Objekte in der abstrakten Syntax ASN.1 können später auch von anderen Management-Protokollen benutzt werden.
Die aus der TCP/IP-Welt bekannte MIB I reflektiert Protokolle des Internet. Diese MIB wurde 1990 durch die MIB 2 abgelöst und wird seither nicht mehr verwendet. MIB II hat wesentlich mehr Elemente, die die verschiedenen Bereiche wie Host-MIB, Bridge-MIB, Router-MIB usw. beinhalten. In MIB II sind die Objekttypen in elf Gruppen angeordnet, die jede für sich eigene MIB-Bäume bilden. Es git folgende MIB-Groups: System, Interface, Address Translation, Internet, Internet Control Message Protocol (ICMP), User Datagram Protocol (UDP), Exterior Gateway Protocol (EGP), CMIP over TCP/IP (CMOT), Transmission und Simple Network Management Protocol (SNMP). Die Gruppenbildung bietet eine höhere Flexibilität für die Implementierung der MIBs auf den Agents.

MIME (multipurpose Internet mail extensions)

Die Multipurpose Internet Mail Extension (MIME) ist eine Erweiterung des Simple Mail Transfer Protocols (SMTP), um die Beschränkungen des SMTP-Protokolls in Bezug auf Textnachrichten, Grafiken, Audio und andere binäre Dateien aufzuheben. Das Simple Mail Transfer Protocol (SMTP) unterstützt nur Textnachrichten im ASCII-Zeichensatz mit 7 Bit. Dagegen können mit den Multipurpose Internet Mail Extensions (MIME) Texte im 8-Bit-Code, Grafiken, Audio und Video übertragen werden. Zu diesem Zweck teilt MIME die verschiedenen Dateitypen in Haupt- und Untergruppen, genannt Medientypen und Subtypen.
Die einzelnen MIME-Typen der verschiedenen Medientypen können in einem einzigen Nachrichtenkörper zusammengefasst werden. Der Empfänger muss allerdings in der Lage sein, diese wieder zu trennen. Zu den MIME-Medientypen gehören: text, image, video, audio, message, text, zu den MIME-Subtypen, die vom Medientyp durch einen Schrägstrich getrennt sind und die den MIME-Typ genauer definieren, gehören: msword, html, javascript, jpg, gif, mpeg.
Voraussetzung für den Empfang und die korrekte Darstellung dieser multimedialen elektronischen Post ist, dass das Mail-Programm des Empfängers ebenfalls den MIME-Standard unterstützt. Das MIME-Protokoll wird auch zur Kommunikation zwischen zwei Kommunikationspartnern im Internet eingesetzt, zwischen Programmen, die entfernt voneinander auf arbeiten. So können beispielsweise die unterschiedlichen Daten eines Webservers für Grafiken und Audio direkt mit MIME auf einem Browser dargestellt werden. Bei der Übertragung wird die Datei im Header beschrieben. Damit die Daten für alle Computer und Betriebssysteme interpretierbar werden, werden sie vor dem Versenden codiert. Die im ASCII-Zeichensatz und im erweiterten Zeichensatz verwendete 8-Bit-Codierung wird daher auch in einen 7-Bit-Code und einen 6-Bit-Code umgerechnet. Zur Erhöhung der Datensicherheit gibt es mit S/MIME eine MIME-Version mit Verschlüsselung. Multipurpose Internet Mail Extension (MIME) ist von der Internet Engineering Task Force spezifiziert und in RFC 2049 beschrieben.
Der erste Teil dieser Spezifikation (RFC 2045) führt grundlegende zusätzliche Felder im Kopf von E-Mails ein:

```
MIME-Version
Content-Type
Content-Transfer-Encoding
```

Das Content-Transfer-Encoding gibt an, ob die Übertragung nach Internetstandard RFC 6152 erfolgen soll, dies stattgefunden hat, oder eine Kodierung für Internetstandard RFC 822 erfolgt ist, die beim Empfänger wieder rückgängig gemacht werden muss:

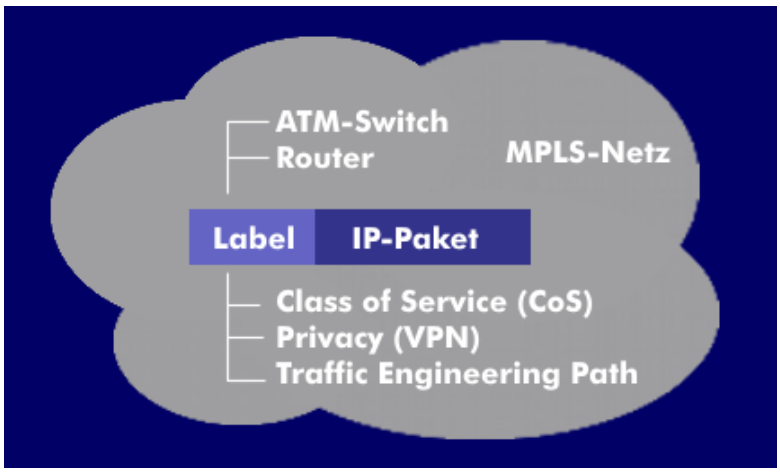
- `7bit` – keine Kodierung, Text enthält nur ASCII-Zeichen
- `8bit` – keine Kodierung, Text enthält auch Nicht-ASCII-Zeichen, Übertragung mittels Extended SMTP
- `binary` – keine Kodierung, binärer Inhalt
- `quoted-printable` – Kodierung von Steuerzeichen und Nicht-ASCII-Zeichen durch Ersetzung mit deren Hexadezimalwert
- `base64` – Kodierung durch Transformation in eine 6-Bit-Darstellung

Was kein Text ist, erfordert sofern der ESMTP-Server nicht Binärdaten nach RFC 3030 (BDAT-Befehl) akzeptiert auf jeden Fall Kodierung, die dann grundsätzlich nach Base64 erfolgt. Nichts weiter als beliebigen Text enthaltende E-Mails bedürfen hingegen keiner Umformung. Ein Beispiel:

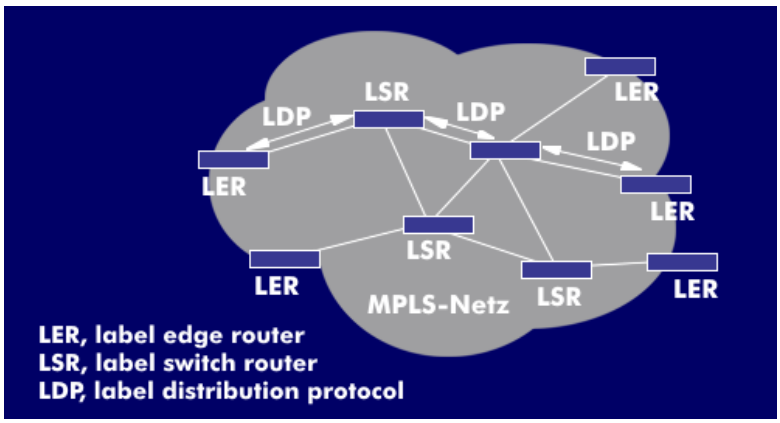
```
From: <alice@example.org>
To: <bob@example.org>
Subject: Umlaute dank MIME
MIME-Version: 1.0
Content-Type: text/plain; charset=iso-8859-15
Content-Transfer-Encoding: 8bit

Wären die drei zusätzlichen Kopfzeilen nicht, wäre diese Zeile nicht leserlich.
```

MPLS (multi-protocol label switching)

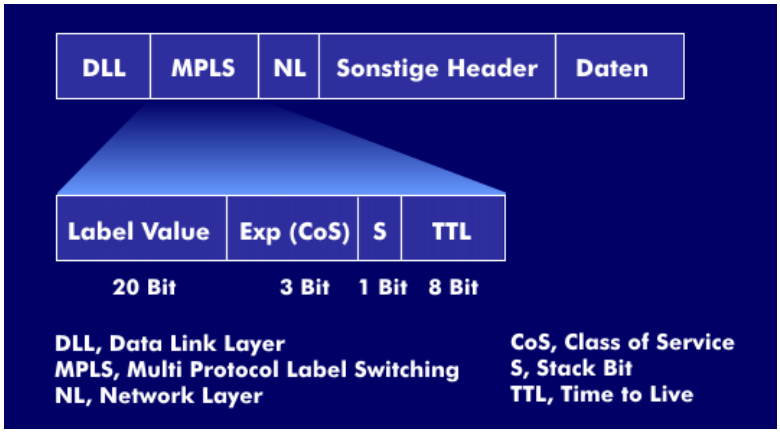
Multi-Protocol Label Switching (MPLS) ist eine Technologie, deren Konzept auf der Übermittlung von IP-Daten über ATM basiert. Mit MPLS lassen sich komplexe vermaschte virtuelle Verbindungen (VC) für einzelne Virtual Private Networks (VPN) umgehen und tausende von VPNs auf vorhandener Technik aufbauen.
Die MPLS-Technologie dient der Erhöhung der Leistungsfähigkeit von Weitverkehrsnetzen und soll dazu beitragen, den stark wachsenden IP-Verkehr zu bewältigen. Des Weiteren wird die Dienstgüte in IP-Netzen bereitgestellt und der Transport von Echtzeitdaten über paketbasierte Netze vereinfacht.
Das MPLS-Switching ist von der Internet Engineering Task Force (IETF) standardisiert und bietet ein vereinfachtes Management und eine verbesserte Organisation des Datenverkehrs in Internetworking-Systemen und VPNs sowie die Unterstützung von skalierbaren und verwaltbaren QoS-Anwendungen. Mit MPLS lässt sich jeder Anwendung eine differenzierte Dienstgüte zuordnen.
Die technische Basis von des MPLS-Switching ist ein Label-basierter Weiterleitungsmechanismus. Dem Verfahren nach werden die IP-Pakete entsprechend ihrer Priorität mit Etiketten, sogenannten Labels, versehen und über einen Label Switched Path (LSP) übertragen. MPLS weist den Datenpaketen beispielsweise Verbindungen mit der erforderlichen Bandbreite zu und räumt ihnen Vorrang vor weniger wichtigen Daten ein.



Die Zuordnung eines Datenpaketes zu einem Stream wird beim MPLS-Switching nur bei Eintritt in das Netzwerk durchgeführt. Wenn ein Datenpaket zum nächsten Knoten gesendet wird, wird das Label mitgeschickt. Dadurch ist bei den nächsten Knoten keine Analyse der Layer-3-Information mehr erforderlich. Das Label wird als Tabellenindex genutzt, in dem der nächste Knoten und ein neues Label definiert sind. Das alte Label wird ersetzt und das Datenpaket zum nächsten Knoten gesendet. Am Ende des Netzwerkes werden die eingehenden Labels aus den Datenpaketen ausgewählt und verarbeitet.



Die zentralen Komponenten eines MPLS-Netzwerks sind die Label Switch Router (LSR). Sie teilen die zu transportierenden IP-Datagramme in Forwarding Equivalence Classes (FEC) ein, wobei sie sich an den Layer-3-Zieladressen und anderen Merkmalen (Labels) orientieren. Anhand der Labels leiten die nachgeschalteten Router die ankommenden Datenpakete weiter. Dieser Prozess wird als Label-Switching bezeichnet.
Der erste LSR-Router in einem MPLS-Netz ist der Ingress Label Switch Router, der letzte der Egress Label Switch Router. Der gesamte Pfad zwischen Ingress Label Switch Router und Egress Label Switch Router heißt Label Switched Path (LSP).



Die IETF hat für die Zuweisung der Labels zwei Signalisierungsprotokolle vorgesehen: das für MPLS entwickelte Label Distribution Protocol (LDP) mit Constraint Routing LDP (CR-LDP) sowie das um Traffic-Engineering (TE) ergänzte Resource Reservation Protocol (RSVP-TE).
Der MPLS-Header besteht aus dem 20 Bit umfassenden Datenfeld Label Value, das die Basis für die Weiterleitung der Datenpakete durch das MPLS-Netz bildet. Dieser Wert dient als Index in der Weiterleitungstabelle. Diesem Datenfeld folgt das 3 Bit lange Exp-Datenfeld, was für Experimental Bits steht, und das normalerweise Diffserv unterstützt. Das folgende 1 Bit umfassende S-Datenfeld (Stack-Bit) zeigt an, dass der untere Boden des Stapels erreicht ist. Und das 8 Bit umfassende Time-to-Live-Feld schützt das MPLS-Netz gegen eine Schleifenbildung. Es wird bei jedem Hop dekrementiert.
Die komplette MPLS-Architektur ist in den RFCs 3031 und 3032 beschrieben.

MPTCP (multipath TCP)

Multipath-TCP (MPTCP) ist eine TCP-Erweiterung, die von einer Arbeitsgruppe der Internet Engineering Task Force (IETF) standardisiert wurde. Bei MPTCP können über eine TCP-Verbindung mehrere Pfade eingerichtet werden um so die Ressourcen besser zu nutzen, die Redundanz, den Datendurchsatz und die Ausfallsicherheit zu erhöhen und auf Fehler ausgeglichener zu reagieren. Die von MPTCP gebotene Redundanz ermöglicht inverses Multiplexing, bei dem ein breitbandiger Kanal in mehrere unabhängige Einzelkanäle geteilt werden kann. Multipath-TCP ist rückwärts kompatibel zu normalem TCP.
Multipath-TCP ist für WLANs von besonderem Interesse, bei dem sowohl WiFi als auch Mobilfunknetze genutzt und damit die Qualität und Leistungsfähigkeit von Smartphones erhöht werden können. Bei dieser Technik können beim inversen Multiplexing mit einer Art Link-Handover Verbindungen hinzugeschaltet oder entfernt werden, und zwar so, dass die TCP-Ende-Verbindung nicht unterbrochen wird. Die MPTCP-Technik benötigt keine spezielle Hardware, sie wurde direkt in den Protokollstack des IP-Protokolls implementiert und zwar durch Abstraktion in der Transportschicht, ohne einen speziellen Mechanismus auf der Netzwerkschicht oder dem Link-Layer. Das Link-Handover wird direkt in den Endpunkten implementiert. Bei der Übertragung handeln Clients und Server selbstständig eine TCP-Verbindung aus. Die Datenpakete mit unterschiedlichen IP-Absendern können über verschiedene Leitungen übertragen werden.
Multipath-TCP bringt Vorteile für die Leistung von Rechenzentren mit sich. Im Unterschied zur Link-Aggregation nach 802.3ad kann Multipath-TCO eine einzelne TCP-Verbindung über mehrere Schnittstellen ausgleichen. MPTCP stellt die gleiche Benutzerschnittstelle dar, wie normales TCP. Es modifiziert TCP so, dass es für Anwendungen wie eine normale Schnittstelle erscheint, obwohl es die Daten über verschiedene Sub-Verbindungen verteilt.
Es gibt verschiedene Implementierungen für MPTCP, so für FreeBSD, für Linux Kernel, Citrix und Apple iOS 7.

MSC (message sequence chart)

Der Message Sequence Chart (MSC) ist ein ITU-T-Standard für die grafische Darstellung von Signalsequenzen in der Vermittlungstechnik. Er ist als standardisierter Protokolltest in der ITU-Empfehlung Z.120 von 1999 sowie in dessen Anhängen "Formal Semantics of Message Sequence Chartsünd SStatic Semantics of Message Sequence Charts"beschrieben.
Mit der Message Sequence Chart kann das Kommunikationsverhalten von Vermittlungssystemen beschrieben werden. Eine MSC-Chart stellt den zeitlichen Ablauf der Informationen zwischen den vermittelnden Komponenten und den angeschlossenen Netzen und Geräten dar.
Bei Verwendung der Unified Modeling Language erfüllt das Sequenzdiagramm die Aufgabe des MSC.

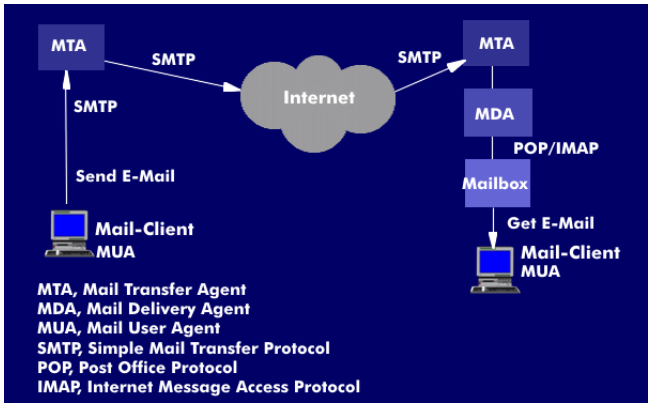
MSTP (multiple spanning tree protocol)

Multiple Spanning Tree Protocol (MSTP) ist ein von der IEEE-Arbeitsgruppe 802.1s, später

802.1Q, standardisiertes Spanning-Tree-Protokoll mit nicht nur einem Spanning Tree, der sich über das gesamte Netzwerk erstreckt, sondern mit mehreren kleineren Spanning Trees, die über ein größeres physikalisches Netz aufgebaut werden können. Bei großen Netzen hat ein einzelner Spanning Tree den Nachteil, dass die Rekonfigurationszeiten relativ lange dauern. Dies kann durch mehrere Spanning Trees mit kürzeren STP-Instanzen vermieden werden.
Beim MSTP-Verfahren hat jede einzelne Root-Bridge eine eigene Spanning Tree Instanz (MSTI), die vollkommen unabhängig von anderen ist. Das in 802.1s behandelte MSTP ist als Weiterentwicklung des Rapid Spanning Tree Protocol (RSTP) anzusehen, es unterstützt bis zu 64 MSTIs, mit einigem Aufwand sogar bis 4.000.
Beim MSTP-Verfahren wird eine Root-Bridge bestimmt und die geringsten Streckenkosten zwischen der Root-Bridge und den angebotenen Root-Ports der einzelnen Brücken. Die Root-Bridge sendet Bridge Protocol Data Units (BPDU) an alle Bridges und stellt über die in den BPDU-Datenpaketen enthaltenen Konfigurationsdaten die Netzkonfiguration fest.

MTA (mail transfer agent)

Die Bezeichnung Mail Transfer Agent (MTA) wird synonym mit Message Transfer Agent (MTA) benutzt. Es handelt sich um einen Mail-Server, der Mitteilungen von einem Mail User Agent (MUA) oder von einem anderen Mail-Server empfängt und diese für die Weiterleitung mittels Simple Mail Transfer Protocol (SMTP) bereitstellt.
Den eigentlichen Datentransport an den Empfänger oder einen weiteren Mail-Server übernimmt der Message Delivery Agent (MDA).
Bei diesem Softwareprogramm zur Vermittlung von E-Mails versucht jeder Message Transfer Agent den empfangenden Mail-Server direkt zu erreichen und zwar über den Domainnamen. Über den DNS-Server wird dann die IP-Adresse des Mailbox-Hostes ermittelt.
MHS-Systeme wurden bereits in den ersten RFCs in den 80er Jahren beschrieben. So auch die Implementierung des Mail Transfer Agent (MTA) und des Simple Message Transfer Protocols (SMTP), die erstmals in RFC 821 beschrieben wurden.



MTU (maximum transmission unit)

Die Maximum Transmission Unit (MTU) ist die größtmögliche Framelänge, die über ein vorhandenes physikalisches Übertragungsmedium bzw. über einen LAN- oder WAN-Pfad ohne Fragmentierung gesendet werden kann. Wenn größere Framelängen auftreten, werden sie entsprechend den verwendeten Protokollregeln fragmentiert oder das Frame wird verworfen. Weitverkehrsnetze (WAN) haben in aller Regel geringere MTU-Größen als lokale Netze (LAN).
Je größer die MTU-Einheit, desto besser ist das Verhältnis von Nutzdaten zu Headerdaten. Bei einem TCP-Header mit 40 Byte und einer MTU-Einheit von 650 Byte, ist das Verhältnis 6,1 %, bei einer größeren MTU-Einheit von 1.500 Byte sinkt das Verhältnis auf 2,66 %. Da aber die Framelänge unmittelbar in die Übertragungszeit eingeht, spielt die MTU-Einheit

vorallem bei Echtzeitanwendungen wie VoIP oder Online-Games eine wesentliche Rolle für die Verzögerungszeit und das Antwortzeitverhalten.
Bei IPv6 muss das IP-Netz Datenpakete von mindestens 1.280 Byte unterstützen. Allerdings sollten die Netzwerke so konfiguriert werden, dass sie 1.500 Byte oder sogar mehr übertragen können, damit auch verkapselte Ethernet-Payloads übertragen werden können, ohne Fragmentierungen zu beeinträchtigen. Dagegen stellt 802.15.4 eine Maximum Transmission Unit (MTU) von lediglich 127 Byte zur Verfügung. Das bedeutet, dass bei 6LoWPAN, bei dem IPv6-Protokolle über Low Power WPANs übertragen werden, eine Anpassung durch Fragmentierung erfolgen muss.

MUA (message user agent)
Die Bezeichnung Message User Agent (MUA) und Mail User Agent (MUA) werden synonym verwendet. Es handelt sich dabei um einen E-Mail-Client, in den der Benutzer Nachrichten eingibt, E-Mails erstellt, sendet, empfängt und liest. Die Nachrichten werden mittels Simple Mail Transfer Protocol (SMTP) zum Mail-Server übertragen und mit dem Post Office Protocol (POP) oder dem Internet Message Access Protocol (IMAP) abgeholt.
Der Message User Agent ist ein Diensterbringer, der dem Benutzer die Dienste des Message Transfer Service (MTS) zugänglich macht. Er kommuniziert mit dem Mail Transfer Agent (MTA) als Benutzer der Nachrichtendienste. Mail User Agents können Nachrichten aus Mailboxen lesen und auch auf diese antworten.

Multicast
Multicast bezeichnet in der Telekommunikation eine Nachrichtenübertragung von einem Punkt zu einer Gruppe und ist daher eine Form der Mehrpunktverbindung. Die Technik kommt gemäß OSI-Modell in der Vermittlungsschicht zum Einsatz. Ihr Vorteil besteht darin, dass zeitgleich Nachrichten an mehrere Teilnehmer oder an eine geschlossene Teilnehmergruppe übertragen werden können, ohne dass sich die hierfür verwendete Datenübertragungsrate beim Sender mit der Zahl der Empfänger multipliziert. Der Sender braucht beim Multicasting nur dieselbe Datenübertragungsrate wie ein einzelner Empfänger. Handelt es sich um paketorientierte Datenübertragung, findet die Vervielfältigung der Datenpakete an jedem einzelnen Verteiler (Router, Switch oder Hub) auf der Route statt.
Der Unterschied zu Broadcast besteht darin, dass beim Broadcast Inhalte verbreitet werden (hier: ganz überwiegend sog. Content), die – mit geeigneter Empfangsausrüstung – jeder ansehen kann, wohingegen beim Multicast vorher eine Anmeldung beim Sender erforderlich ist.
Eine spezielle Ausprägung von Multicast ist der Geocast, bei der nur in einen räumlich abgegrenzten Bereich gesendet wird.
Man spricht im Allgemeinen bei einem Multicast auch von einer Punkt-zu-Mehrpunkt-Verbindung. *Siehe auch*: *Kommunikationsart (S. G43)*.

Multiplexverfahren
Multiplexverfahren sind Techniken mit denen mehrere Signale gleichzeitig simultan oder sequenziell über vorhandene Übertragungswege übertragen werden. Multiplextechniken können auf drahtgebundenen Übertragungswegen und auch in der Funktechnik eingesetzt werden. Durch sie werden die Übertragungswege effizienter genutzt, weil gleichzeitig mehrere Übertragungskanäle über einen Übertragungsweg übertragen werden. Die Vorteile der Multiplextechniken liegen in der Wirtschaftlichkeit, der verbesserten Frequenzökonomie und der Erhöhung der Datenübertragungsraten.

NAT (network address translation)
Das NAT-Verfahren (Network Address Translation) ist ein Verfahren des Adress-Mappings mit dem IP-Adressen eines Netzwerks in IP-Adressen eines anderen Netzwerks übersetzt werden. Das Verfahren wird dazu verwendet, um Computern eines privaten Netzes einen gemeinsamen Zugang über das IP-Protokoll zum Internet zu ermöglichen. Beim NAT-Verfahren werden die IP-Adressen eines privaten Netzwerks registriert und öffentlich

registrierten IP-Adressen zugeordnet.Der Vorteil des NAT-Verfahrens liegt darin, dass alle Rechner, die innerhalb eines Unternehmensnetzes miteinander kommunizieren, nur über eine IPv4-Adresse Zugang zum Internet haben. Dadurch wird die Anzahl an öffentlichen IPv4-Adressen begrenzt, was für weniger Verwaltungsaufwand und mehr Sicherheit sorgt.
Die internen Netzwerkadressen sind spezielle Netzwerkadressen in einem dafür reservierten Adressbereich. Sie sind nur im internen lokalen Netz nutzbar, öffentlich nicht bekannt und werden nicht über den NAT-Router, der sich zwischen dem internen und dem öffentlichen Netz befindet, geroutet. Über sie können interne Computer auf die internen Netzwerkressourcen wie Drucker oder Server zugreifen. Lediglich die Computer, die eine Kommunikation zu externen, öffentlichen Rechnern aufbauen, erhalten beim Routing im NAT-Router einen Tabelleneintrag. Bei der Übertragung der Datenpakete werden die Quelladressen der internen Computer im NAT-Router durch die öffentliche IPv4-Adresse ersetzt. Diese Adresse bezieht der NAT-Router aus der NAT-Tabelle. Dieses Verfahren wird auch als Source NAT (SNAT) bezeichnet. Erfolgt die Übertragung vom Internet aus, so handelt es sich um eine Destination NAT (DNAT).
Das NAT-Verfahren reduziert die Anzahl an öffentlichen IPv4-Adressen und verbessert die Sicherheit, da die ein- oder ausgehenden Anfragen mittels Authentifizierung überprüft werden können. Die Neztwerkadressen-Übersetzung kann zwischen Firmennetzen und Internet eingesetzt werden, ebenso zwischen Netzen mit IPv4 und IPv6 und umgekehrt. Dadurch wird die Integration einer IPv4-Infrastruktur in IPv6-Umgebungen möglich und IPv6-Dienste können mit IPv4-Systemen interagieren.
Neben dem Network Address Translation (NAT) gibt es noch mehrere NAT-Varianten wie das Network Address Port Translation (NAPT), das dynamische Address-Mapping, Dynamic Network Address Translation (DNAT) und das Static Network Address Translation (SNAT).

NDP (neighbour discovery protocol)

Das Neighbour-Discovery-Protokoll (ND) dient dem Internet Control Message Protocol (ICMP) in der Version ICMPv6 bzw. dem IPv6-Protokoll zur schnellen Ermittlung der Netzwerkadressen. Darüber hinaus wird das Neighbour Discovery Protocol von Routern und Rechnern genutzt, um am Netzwerk angeschlossene Router zu ermitteln und den aktuellen Netzzustand sowie Änderungen im Netz automatisch zu erkennen. Neben der Erkennung der Netzwerkadressen dient das Neighbour-Discovery-Protokoll zur Ermittlung der verfügbaren Router, der Link-Parameter, des nächsten Hop, einer Umleitung bei optimalen Bedingungen und von nicht mehr erreichbaren Stationen.

Netzwerkanwendung

Eine **Netzwerkanwendung** besteht aus *Anwendungsprozessen* verschiedener Endsysteme, welche mittels *Nachrichten* kommunizieren. Für die **Implementierung** können die *Dienste der Transportschicht* direkt verwendet werden. Ein **Anwendungsprotokoll** standardisiert das Verhalten und sorgt dabei für ein einheitliches Format der Nachrichten. Tieferliegende Schichten, sowie der Netzwerkkern, benötigen keine Kenntnis über die Anwendung. Es herrscht *große Dynamik* bei *einfacher Verbreitung*.
Beispiele der Anwendungsschicht sind Web-Browser und Web-Server.

Netzwerkschicht (network layer) (OSI)

Die Netzwerkschicht ermöglicht den Verbindungsaufbau zwischen zwei beliebigen Teilnehmern durch geeignete Adressierung. Die dafür auszuführenden Funktionen sind: Vermittlung, Verbindungsaufbau und -abbau, Rücksetzung, Unterbrechung, Fehlererkennung sowie transparenter Datentransport zwischen den Netzwerkendpunkten.
Die grundlegende Aufgabe der Vermittlungsschicht ist es, Dienste bzw. Funktionen bereitzustellen, die es ermöglichen, die gesicherten Systemverbindungen miteinander zu verknüpfen. Hierbei sind nicht nur homogene Netze bzw. Führungen durch ein einziges Netz zu berücksichtigen, sondern es sind auch Endsystemverbindungen zu ermöglichen, die über mehrere unterschiedliche Netze geführt werden können.

Die Vermittlungsschicht bietet der darüber liegenden Transportschicht diverse Dienstleistungen an. So werden Dateneinheiten transparent durch das Netzwerk transportiert, wobei bei verbindungsorientierter Kommunikation auf die Einhaltung der richtigen Reihenfolge geachtet wird. Die Flusskontrolle erlaubt dem Endgerät oder dem Vermittlungssystem, den Datenfluss bei verbindungsorientierter Kommunikation zwischen Endgerät und Vermittlungssystem zu kontrollieren. Stoppt ein Kommunikationspartner den Datenfluss zu seinem Gerät, da er nicht mehr empfangsbereit ist, werden alle Daten solange zwischengepuffert, bis der Kommunikationspartner den Datenfluss wieder freigibt. Vereinbarungen über Leistungsmerkmale benötigt der Anwender, um verschiedene Kommunikationsfunktionen zu aktivieren, wie Rufumleitung, geschlossene Benutzergruppen usw. *Siehe auch*: *OSI (S. G41)*.

Object Management Group (OMG)

Die OMG ist ein 1989 gegründetes Konsortium, das sich mit der Entwicklung von Standards für die herstellerunabhängige systemübergreifende objektorientierte Programmierung beschäftigt.

OID (object identifier)

Der Object Identifier (OID) ist ein Konzept das im Simple Network Management Protocol (SNMP) definiert ist. Es ist der Wert, der in bestimmten Modulen der Management Information Base (MIB) zur eindeutigen Identifizierung spezieller SNMP-Objekte benutzt wird. Die OID-Identifikation selbst ist eine Sequenz von Buchstaben und Ziffern, mit der das Objekt eindeutig gekennzeichnet wird.

In einer Event Management Information Base (MIB) kann ein Netzwerkmanagementsystem oder der Benutzer spezielle Objekte beobachten und auf bestimmte Ereignisse hin triggern. Dieser Trigger, der durch voreingestellte Pegel oder gesetzte Filter ausgelöst werden kann, spezifiziert den Object Identifier (OID) des überwachten Objekts in der MIB-Datenbank.

ONF (open networking foundation)

Die Open Networking Foundation (ONF) ist eine Benutzer-orientierte Organisation, die sich der Einführung und Förderung von Software Defined Networking (SDN) widmet und dafür offene Standards entwickelt. Der Ansatz von SDN-Networking unterscheidet sich von konventionellen Ansätzen dadurch, dass das physikalische Netzwerk von der Netzwerksteuerung entkoppelt ist und diese softwaremäßig erfolgt. Dadurch können extrem dynamische, verwaltbare und kosteneffektive Netzwerke aufgebaut werden.

Durch die Implementierung von Software Defined Networks anhand offener Standards entstehen Netzwerke von außerordentlicher Dynamik. Die Open Networking Foundation zielt auf einen offenen, kollaborativen Entwicklungsprozess, der von den Endbenutzern ausgeht. Ein von der ONF entwickelter Standard ist OpenFlow, ein weiterer das OpenFlow Configuration and Management Protocol.

OSPF (open shortest path first)

Open Shortest Path First (OSPF) ist ein Interior Routing Protocol (IRP) und beschreibt wie Router untereinander die Verfügbarkeit von Verbindungswegen zwischen Datennetzen propagieren. Es unterstützt hierarchische Netzstrukturen, zeichnet sich durch ein schnelles dynamisches Verhalten in Bezug auf die Änderungen in der Netztopologie aus, optimiert das Routing hinsichtlich der Übertragungskosten, hat eine dynamische Lastverteilung, einen geringen Overhead und kann die Dienstleistungsmerkmale, Type of Service (TOS), im Routing berücksichtigen.

Eine Kostenzuordnung für die einzelnen Nutzer kann anhand diverser Leitungsparameter wie Tarifierung, genutzte Bandbreite, Lastaufkommen u.a. vorgenommen werden. Diese Parameter können auch für die Metrik genutzt werden, wodurch die Routenerstellung flexibel und differenziert erfolgen kann.

OSPF arbeitet nach dem Link-State-Algorithmus (LSA) kann große Entfernungen mit mehr als 14 Zwischensystemen überbrücken und Subnetze in Gruppen zusammenfassen. Insgesamt

kann OSPF Datenpakete über 65.000 Router leiten.
Der dem OSPF-Protokoll zugrunde liegende Routing-Algorithmus ist der SPF-Algorithmus (Shortest Path First). Das Routing des OSPF-Protokolls nutzt zur Optimierung eine Datenbank, in der die angrenzende Topologie abgelegt ist. Aufbauend auf dieser Topologie generiert sich jeder Router eine hierarchische Baumtopologie, den Shortest-Path-Baum, in dem jedes Ziel mit der kürzesten Route eingetragen ist. Die Baumstruktur ist in die Ebenen Netze, eine Gruppe von Netzen, die sogenannte Area, von Backbones, die die Areas miteinander verbinden, und autonome Systeme, die eine Zusammenfassung aller über das Backbone verbundenen Netze darstellen, untergliedert.
Die Kommunikation zwischen den Routern erfolgt über einen Authentisierungs-Mechanismus, an dem nur autorisierte Router teilnehmen können. Routing-Informationen anderer Routing-Protokolle werden transparent weitergeleitet.
Das OSPF-Protokoll baut direkt auf dem IP-Protokoll auf und ist eine Weiterentwicklung einer frühen Version des Intermediate System to Intermediate System Protocol (IS-IS).
Der Header des OSPF-Protokolls kennt neben den Datenfeldern für die Version, den Typ und die Paketlänge, der Prüfsumme und der Quelladresse auch ein 4 Oktett langes Datenfeld für die ID der Area sowie mehrere Datenfelder für die Authentisierung, wobei der Authentisierungstyp festlegt, ob überhaupt eine Authentisierung stattfinden soll.
Neben dem klassischen OSPF gibt es zwei weitere Versionen: OSPFv2 für IPv4 und OSPFv3 für IPv6. OSPFv3 basiert auf der Vorgängerversion OSPFv2 und behält die meisten darin vorhandenen Routing-Mechanismen bei. Die erweiterten Funktionen betreffen die Adressierungssemantik, die von den OSPF-Datenpaketen und den Link-State-Algorithmen (LSA) entfernt und durch neue Link-State-Algorithmen für den Transport der IPv6-Adressen ersetzt wurden. Das klassische IP-basierte OSPF läuft unter OSPFv3 auf Link-Basis und nicht mehr auf IP-Basis. Die Authentifizierung wird nicht mehr vom OSPFv3-Protokoll vorgenommen, sondern durch den Authentication Header (AH) von IPv6.
Open Shortest Path First (OSPF) ist eine Spezifikation der Internet Engineering Task Force (IETF) und wird in diversen RFCs beschrieben. So in RFC 2328. Die Version 3, OSFPv3, ist in den RFC 2740 beschrieben.

Out-of-Band-Control

Steuerbefehle beim FTP gehen nicht über die normale Datenverbindung, sondern über extra aufgebaute Verbindungen.
Vorteil: Steuerung während einer Datenübertragung ist möglich. *Siehe auch*: *FTP (S. G25)*.

Paketvermittlung

Die Paketvermittlung ist ein Verfahren zur Datenübertragung in Rechnernetzen. Längere Nachrichten werden in einzelne Datenpakete aufgeteilt und entweder verbindungslos (als Datagramm) oder über eine virtuelle Verbindung übermittelt.
Hierbei durchqueren die Pakete das Netz als unabhängige und eigenständige Einheiten und können so in den Vermittlungsknoten zwischengespeichert werden. Dadurch wird die Bitrate effizienter aufgeteilt und ein kurzfristisges höheres Datenaufkommen kann über Puffer abgefangen werden.
Nachteile finden sich in den dadurch entstehenden Verzögerungen, Paketverlusten und Pufferüberläufen. Ebenso ist der Ablauf intransparent, da der Anwender keinerlei Informationen über den sich dynamisch ändernden Übertragungsweg erhält. *Siehe auch*: *Vermittlungsart (S. G88)*.

Passives FTP

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. Hier wird auf der Client-Seite ein Port jenseits 1023 verwendet und auf der Server-Seite der vorher an den Client übermittelte Port.
Vorteil gegenüber dem aktiven Modus: Diese Technik wird eingesetzt, wenn der Server keine

Verbindung zum Client aufbauen kann. 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. Die Firewall erlaubt in diesem Fall die Datenverbindung, weil sie ihren Ursprung innerhalb der geschützten Zone hat.

Siehe auch: *FTP (S. G25) und Out-of-Band-Control (S. G57)*.

Peer-to-Peer-Netz

Peer-to-Peer-Netze (P2P) sind Rechnernetze bei denen alle Rechner im Netz gleichberechtigt zusammen arbeiten. Das bedeutet, dass jeder Rechner anderen Rechnern Funktionen und Dienstleistungen anbieten und andererseits von anderen Rechnern angebotene Funktionen, Ressourcen, Dienstleistungen und Dateien nutzen kann. Die Daten sind auf viele Rechner, in der Regel auf die der Nutzer, verteilt. Das Peer-to-Peer-Konzept ist ein dezentrales Konzept, ohne zentrale Server, wie das Internet. Jeder Rechner eines solchen Netzes kann mit mehreren anderen Rechnern verbunden sein.

Einfache Peer-to-Peer-Netze organisieren sich selbst, als Self Organized Networks (SON), sie arbeiten dezentral und haben keinen Server. In einem solchen dezentralen Peer-to-Peer-Netzwerk stellen sich die Workgroup-Mitarbeiter gegenseitig Betriebsmittel und Ressourcen zur Verfügung. Da es sich um hunderte Nutzer-Rechner handeln kann, ist es kaum möglich die einzelnen Nutzer-Rechner zu ermitteln, zumal ihren Namen verschlüsselt sind. Das Peer-to-Peer-Networking wird durch verschiedene Netzwerkbetriebssysteme unterstützt, so von Novells NetWare Lite, Windows for Workgroups (WFW) und die Workstation-Version von Windows NT.

Peer-to-Peer-Networking eignet sich für kleinere Arbeitsgruppen zwischen 5 und 20 Arbeitsplätzen. In einer solchen Umgebung kann jeder Mitarbeiter seinen Partnern den Zugang zu Betriebsmitteln wie Festplatten oder Drucker erlauben. Die einzelnen Peers teilen sich die Daten im Filesharing.

Eine Weiterentwicklung des Peer-to-Peer-Netzes mit zentralen Serverkomponenten sind die Super-Peer-Netzwerke. In einer solchen Konfiguration werden besonders leistungsfähige Peers zu Super-Peers zusammengeschlossen, die die Serverdienste erbringen und das Netzwerk organisieren. Sie sorgen für das Routing der Daten zu den dezentralen Clients und bilden in sich ein Backbone-Netzwerk.

Im Internet gibt es diverse Super-Peer-Konfigurationen, bekannt als anarchische Netze wie Skype für die Internettelefonie, sowie das Netz-im-Netz-Konzept Gnutella, Tauschbörsen wie Kazaa, Napster und BitTorrent für Filesharing und Downloads.

PIO (programmed input output)

Programmed Input Output (PIO) ist ein älteres Übertragungsprotokoll für die Kommunikation zwischen der Zentraleinheit (CPU) und Peripheriegeräten wie Festplattenlaufwerke, Wechselplattenlaufwerken und Diskettenlaufwerken. Bei PIO ist die Zentraleinheit des Rechners enorm belastet, da sie den Informationsaustausch zwischen Arbeitsspeicher und Massenspeicher steuert, was sich in der Datentransferrate bemerkbar macht.

Der PIO-Mode legt die Übertragungsraten der Integrated Drive Electronics (IDE) fest. Es gibt mehrere PIO-Modi mit Übertragungsraten zwischen 3,33 MB/s und 16,6 MB/s: PIO-Mode 0 bietet bei einer Zykluszeit von 600 ns Übertragungsraten bis zu 3,33 MB/s, Mode 1 bei einer Zykluszeit von 383 ns bis zu 5,22 MB/s, Mode 2 bei einer Zykluszeit von 240 ns bis zu 8,33 MB/s, Mode 3 erreicht 11,11 MB/s bei einer Zykluszeit von 180 ns und Mode 4 16,6 MB/s bei 120 ns Zykluszeit.

PIO Mode 4 wird von neueren CD-Laufwerken unterstützt. Das Nachfolgeprotokoll von PIO ist das Direct Memory Access (DMA) als Ultra-DMA (UDMA). Eine weitere PIO-Mode-Variante ist der geplante Mode 5 mit 20 MB/s.

POP (post office protocol)

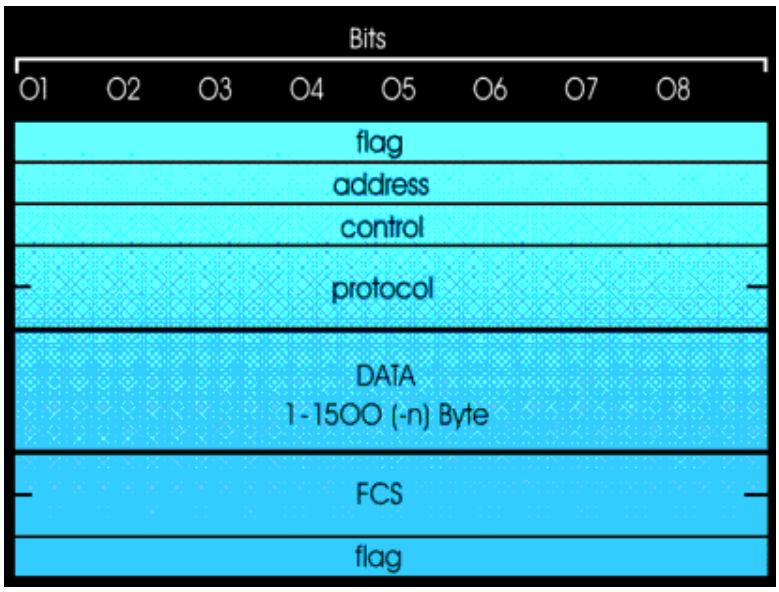
Das Post Office Protocol (POP) ist ein in mehreren RFCs spezifiziertes Standard-Protokoll zum Transfer von E-Mails vom Mail-Server auf den Mail-Client des Benutzers. Das POP-Protokoll arbeitet zusammen mit dem SMTP-Protokoll und wurde von dem effizienteren IMAP-Protokoll abgelöst.
Die meisten PC-basierten Mail-Clients verwenden das Post Office Protocol (POP), um E-Mails vom Server auf den Arbeitsplatzrechner zu laden. Die E-Mails werden vom Absender im POP-Server des Absender-Providers zwischengespeichert, von dort in den POP-Server des Empfänger-Providers geladen und solange zwischegespeichert, bis sie vom Empfänger abgeholt werden. Zur Übertragung der Mail baut der Arbeitsplatzrechner die Verbindung zum POP-Server auf und holt die Mail ab.

POP-Befehl	Funktion
USER name	Übertragung des Benutzernamens
PASS string	Übertragung des Passwortes
QUIT	Abbruch
STAT	Status-Informationen der Mailbox
LIST msg	Informationen über eine best. Nachricht
RETR msg	Übertragung einer Nachricht über die Nummer
DELE msg	Löschung einer Nachricht über die Nummer
NOOP	Keine Operationen
LAST	Nummer der letzten Nachricht
RSET	Rücks. aller Veränderungen der POP-Transaktionen

Das Post Office Protocol existiert in den Versionen POP2 und POP3. Die aktuelle Version POP3 hat den Vorteil, dass die E-Mails zwischengespeichert werden, bis der Benutzer sie abholt. Die auf seinen Arbeitsplatzrechner geladenen E-Mails können anschließend offline, also ohne bestehende Internetverbindung, gelesen werden. Zur Authentifizierung benutzt POP3 den in Klartext geschriebenen Benutzernamen und das Passwort.
Eine verbesserte Sicherheitsstufe hat das POP3S-Protokoll. Dabei handelt es sich um ein erweitertes POP3-Protokoll mit Secure Socket Layer (SSL) und Transport Layer Security (TLS). Ein weiteres POP-Protokoll mit erhöhter Sicherheit ist das Authenticated Post Office Protocol (APOP), das die Authentifizierung kryptografisch verschlüsselt, um sie so gegen unerlaubtes Mithören zu schützen.

PPP (point-to-point protocol)

Das Point to Point Protocol (PPP) ist als Protokoll für die Einwahl ins Internet über leitungsvermittelte Netze konzipiert. Es wurde für die Verkapselung von Datagrammen über serielle Leitungen entwickelt und unterstützt die Übertragung von LAN-Protokollen wie dem IP-Protokoll, dem IInternetwork Packet Exchange Protocol (IPX), DECnet, Appletalk und dem Connectionless Network Protocol (CLNP) mittels Brücken und Routern.
Das PPP-Protokoll soll die Einschränkungen der Interoperabilität aufheben, die durch die von Brücken und Routern verwendete Encapsulation-Technik bei der Übertragung über Weitverkehrsverbindungen verursacht werden. Unabhängig von der Art und Anzahl identischer LAN-Protokolle, die zwei Kommunikationseinheiten unterstützen, war eine Kommunikation nur möglich, wenn beide Einheiten das gleiche WAN-Protokoll hatten. Dieses Problem löst das PPP-Protokoll, indem es herstellerübergreifend Router und Brücken bei WAN-Verbindungen unterstützt.
Das PPP-Protokoll kann Daten über synchrone und asynchrone Wähl- und Standleitungen übertragen. Es ist dadurch in der Lage unabhängig vom jeweiligen physikalischen Interface zu arbeiten. Die einzige Voraussetzung, die beim Einsatz des PPP-Protokolls gefordert wird, besteht in einer vollkommen transparenten, voll duplexfähigen Datenleitung.



Das Datenformat des PPP-Protokolls hat festgelegte Datenfelder und beginnt und endet jeweils mit einem 8 Bit langen Flag. Dieses Flag hat immer die Bitkombination 0111 1110. Im Anschluss an das Flag folgt das 8 Bit lange Adressenfeld, das auf 111 111 gesetzt ist, da das PPP-Protokoll derzeit noch keinen Adressmechanismus unterstützt. Das folgende 8 Bit lange Control-Feld definiert das Unnumbered-Information-Kommando (UI). Die binäre Struktur für das Control-Feld ist 000 001. Es werden nur Datenpakete mit diesen Werten bearbeitet. Das folgende 2 Byte lange Protocol-Feld bestimmt die Weiterbehandlung der Datenpakete. Dabei kann es sich um festgelegte Netzwerkprotokolle handeln, aber auch um Routing-Protokolle. Im folgenden Informationsfeld sind die Header und Daten des im Protokollfeld definierten Netzwerkprotokolls enthalten. Den Abschluss vor dem Flag bildet das 16 oder 32 Bit lange Prüfsummenfeld (FCS).

Das PPP-Protokoll basiert auf dem Data-Encapsulation-Protokoll, dem Link Control Protocol (LCP) und dem Network Control Protocol (NCP). Das PPP-Protokoll ist u.a. in diversen RFCs beschrieben. Für die Übertragung über mehrere unabhängige Übertragungskanäle oder für die Bündelung von mehreren Kanälen gibt es das Multilink-PPP-Protokoll (ML-PPP).

Protokoll (Protocol)

Damit Kommunikationspartner überhaupt miteinander kommunizieren können, müssen sie bestimmte Vereinbarungen und Regeln einhalten. Diese Regeln sind die Protokolle. Ein einzelnes Protokoll arbeitet immer einen bestimmten Funktionsbereich ab, der bei der Datenkommunikation der Funktionalität einer Schicht des OSI-Referenzmodells entsprechen kann.

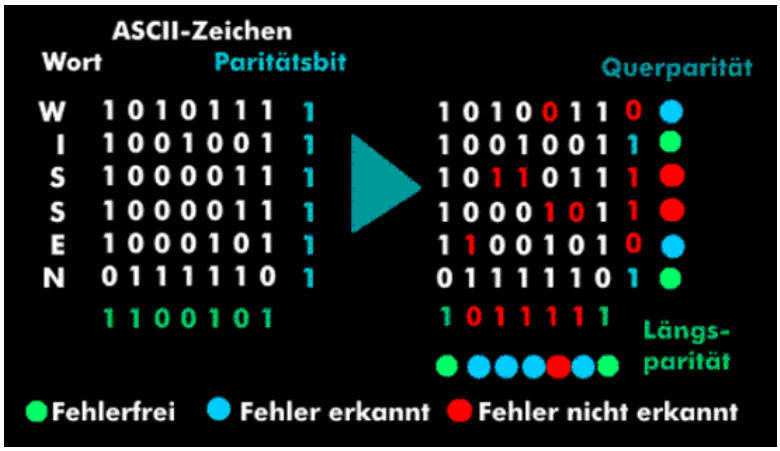
Was die Kommunikation betrifft, so gibt es für die Vermittlungsschicht die Netzwerkprotokolle, für die Transportschicht die Transportprotokolle und für die Anwendungsschicht die Anwendungsprotokolle, wie beispielsweise die File-Transfer- und E-Mail-Protokolle. Darüber hinaus gibt es Routing-Protokolle und Gateway-Protokolle für das Internetworking und Sicherheitsprotokolle für die Verschlüsselung, Authentisierung und Authentifizierung. Protokolle der unteren Schichten des OSI-Referenzmodells werden als Zugangsverfahren bezeichnet. Sind mehrere Protokolle aufeinander abgestimmt und unterstützen mehrere Funktionalitäten, spricht man von einem Protokollstack.

Es liefert eine Beschreibung wie Instanzen miteinander interagieren um einen gewissen Dienst zu realisieren. Dabei definieren sie Verhaltensregeln oder Nachrichtenformate. Wichtig ist allerdings, dass sie lediglich eine Beschreibung der Implementierung liefern, nicht aber die Implementierung selbst. *Siehe auch*: *Dienst (S. G15)*.

Prüfsumme

Die Prüfsummenbildung ist ein Verfahren zur Fehlererkennung, das in den meisten

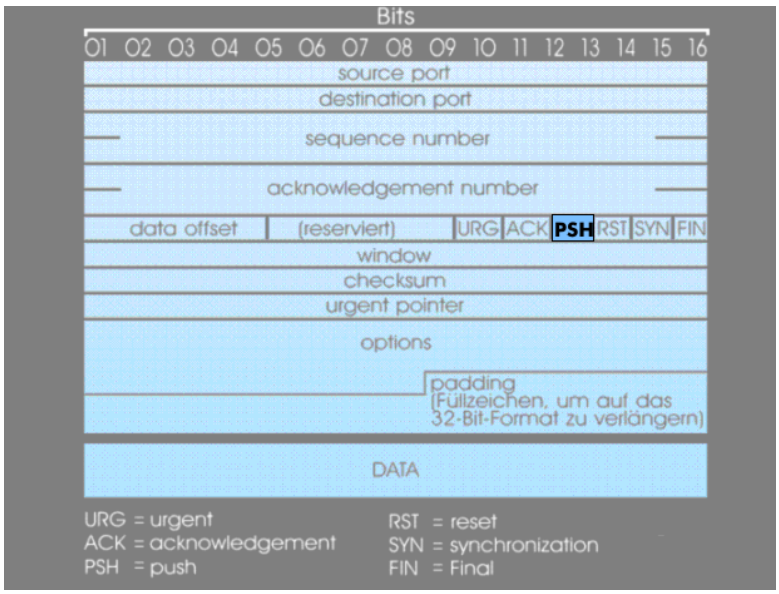
Netzwerkprotokollen und in Codesystemen implementiert ist. Bei Kommunikations- und Transportprotokollen kann sich die Prüfsumme, englisch: Checksum, allein auf die Nutzdaten beziehen, sie kann aber ebenso aus den Nutzdaten, dem Header und dem Trailer berechnet werden und wird im Datenpaket mit übertragen. Bei Codesystemen wie Strichcodes oder 2D-Codes geht es darum die Fehlerquote beim Lesen eines Codes so gering als möglich zu halten.
Bei kommunikationstechnischen Protokollen kann die Prüfsumme durch Paritätsprüfung aus der Summe aller übertragenen Bits ermittelt werden. Zu diesem Zweck haben viele Protokolle ein Prüfsummenfeld in das die Prüfsumme eingetragen wird. Bei anderen Prüfsummenverfahren kommen komplizierte mathematische Algorithmen zum Einsatz, so beispielsweise bei der zyklischen Blockprüfung, dem Cyclic Redundancy Checksum (CRC), beim Modulo-Verfahren oder bei der Fehlerkorrektur, dem Error Correcting Code (ECC).



Je nach Prüfsummenverfahren wird die Blockprüfsumme durch eine bestimmte Zahl, dem Generatorpolynom geteilt und der nichtteilbare Restbetrag wird invertiert und der übertragenen Blockprüfsumme angefügt. Das Ergebnis ist eine einzigartige Prüfsumme, mit der geprüft wird, ob die Nachricht einwandfrei übertragen wurde. In der Regel wird die Prüfsummenzahl am Ende der Nachricht mit übertragen. Die Empfangsstation nutzt den gleichen Algorithmus, errechnet die Prüfsumme und vergleicht die Zahl mit der übertragenen Prüfsumme. Sind die Zahlen identisch, ist die Nachricht fehlerfrei, im anderen Fall ist sie Fehler-behaftet und wird zurückgewiesen.
Beim TCP-Protokoll enthält die Prüfsumme ein 16-Bit-Einerkomplement, das ist die Summe der Einerkomplemente aller 16-Bit-Wörter aus Header und Daten.

PSH (push flag)

Die Push-Flag ist ein Bit-Indikator des Control-Flag-Feldes im TCP-Header, der die interne Steuerung der Datenpakete übernimmt.



Die Flag wird immer während des Verbindungsaufbaus übermittelt. Bei einem gesetzten Flag müssen die Daten mit dem Data Stream Push Service weiterverarbeitet werden, d.h. das TCP-Protokoll muss die Daten an eine höhere Schicht weiterleiten. Bei nicht gesetztem Flag können die Daten in die Sende- bzw. Empfangsqueue gestellt werden.

PTR (pointer)

Bei IP-Adressen können mit einem Domain Name Pointer Rückschlüsse auf den Hostnamen gezogen werden. Damit lässt sich aus der IP-Adresse eines Rechners auf dessen Besitzer schließen. Dieses Verfahren wird auch als Reverse-DNS bezeichnet. Wenn die Reverse-DNS-Einträge gepflegt würden, könnte das eine aktive Hilfe gegen Spams sein. Da dadurch die Authentifizierung vereinfacht würde.

QoS (quality of service)

Unter Dienstgüte, Quality of Service (QoS), versteht man alle Verfahren, die den Datenfluss in lokalen Netzen (LAN) und Weitverkehrsnetzen (WAN) so beeinflussen, dass der Dienst mit einer festgelegten Qualität beim Empfänger ankommt. Es handelt sich also um die Charakterisierung eines Dienstes, der für den Nutzer unmittelbar »sichtbar« ist und dessen Qualität er messen kann. Technisch handelt es sich um eine Parametrisierung von Protokollen zur Bestimmung des Übertragungsverhaltens für bestimmte Dienste.

Im Zusammenhang mit der Dienstgüte sei auf die historische Entwicklung der Netze und Dienste hingewiesen, die sich von der Übertragung zeitunkritischer Daten, wie sie bei Filetransfer oder der Übertragung von E-Mails anfallen, hin zu Netzen mit zeitkritischen Daten entwickelt haben: Internettelefonie, Webkonferenzen und Live-Streaming bestimmen maßgeblich die Datenstruktur.

Um den Anforderungen an die Übertragung von zeitkritischen Daten gerecht zu werden, mussten die die Übertragung beeinflussenden Faktoren wie die Latenz, der Jitter, der Paketverlust, die Bandbreite oder die Verfügbarkeit in definierten Grenzen gehalten werden.

QUIC (quick UDP Internet connection)

Quick UDP Internet Connection (QUIC) ist wie das SPDY-Protokoll ein Netzwerkprotokoll für die schnelle Übermittlung von Webseiten. Beide Protokolle wurden von Google entwickelt und sollen Webseiten schneller zwischen Webserver und Web-Client übermitteln als das HTTP-Protokoll. Während das SPDY-Protokoll auf dem TCP-Protokoll aufsetzt, arbeitet das QUIC-Protokoll mit dem UDP-Protokoll.

Das QUIC-Protokoll unterstützt Multiplex-Verbindungen zwischen zwei Endpunkten, dem Server und dem Client, mit dem UDP-Protokoll. Die Verbindungen sind wie beim TLS- oder SSL-Protokoll verschlüsselt. Zur Vermeidung von Überlastsituationen wurden die Latenzzeiten für die Verbindungen und den Transport durch einen TCP

Congestion Avoidance Algorithm reduziert. QUIC verbessert die Leistungsfähigkeit von verbindungsorientierten Webanwendungen, die mit dem TCP-Protokoll übertragen werden. Darüber hinaus kann QUIC Datenströme multiplexen, Redundanzen in den Daten eliminieren und die Daten durch Kompression reduzieren. Außerdem kann es wie HTTP vielfache Anfragen des Webclients gemeinsam beantworten. Clientseitig wird das QUIC-Protokoll von dem Betriebssystem Google Chrome OS - ab Version 29 - unterstützt.

Reassembling

Reassembling ist ein Begriff aus dem OSI-Bereich. Werden die Fragmente eines Datagramms als einzelne unabhängige Datenpakete über die Netze gesendet, dann müssen die einzelnen Fragmente beim Empfänger wieder zusammengesetzt werden. Diesen Mechanismus nennt man Reassemblierung.
Die Reassemblierung oder das Zusammensetzen stellt die umgekehrte Funktion des Fragmentierens dar. Hierbei werden mehrere Dienst-Dateneinheiten (SDU) zu einer Protokoll-Dateneinheit (PDU) zusammengefasst. Eine solche Reassemblierung muss allerdings bestimmte Voraussetzungen erfüllen, so müssen die einzelnen Datenpakete über die gleichen Quell- und Zieladressen verfügen, sie müssen die gleiche Protokolltypkennung und die gleiche Identifikationsnummer besitzen. Um unnötige Verzögerungen zu vermeiden, ist auch eine zeitliche Begrenzung vorgesehen, in der die einzelnen Datenpakete für die Reassemblierung zur Verfügung stehen müssen. Das Reassemblieren wird mit dem Empfang des letzten Fragments beendet. Das Reassembling findet auf der Vermittlungsschicht statt.

Request for Comments (RFC)

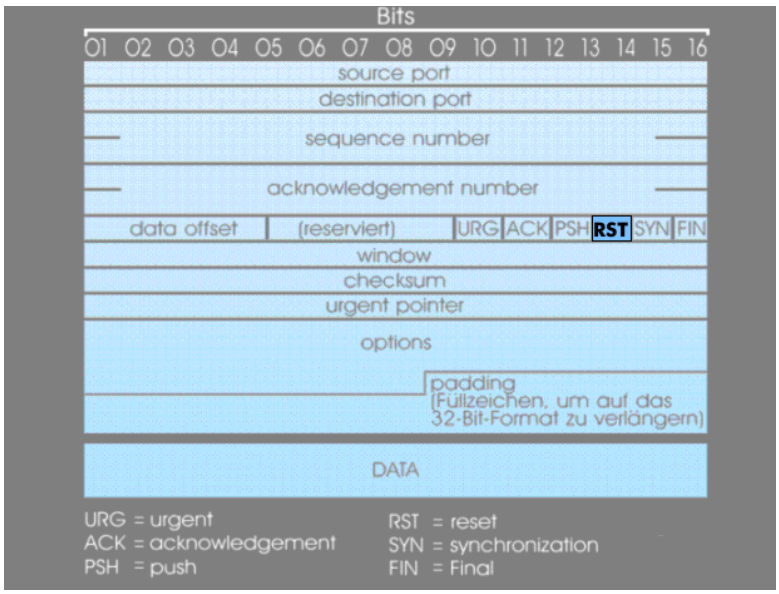
Die RFC sind eine Reihe technischer und organisatorischer Dokumente des RFC-Editors zum Internet (ursprünglich Arpanet), die am 7. April 1969 begonnen wurden. Bei der ersten Veröffentlichung noch im ursprünglichen Wortsinne zur Diskussion gestellt, behalten RFC auch dann ihren Namen, wenn sie sich durch allgemeine Akzeptanz und Gebrauch zum Standard entwickelt haben. Wichtige Beispiele sind Standards zu UDP, POP3, TCP, IP, ICMP, SMTP etc.

Request-Response-Verfahren

Das Request-Response-Verfahren ist eine Anfrage-Antwort-Interaktion mit denen Computer miteinander kommunizieren. Ein anderes Verfahren ist der Dialogbetrieb. Das Request-Response-Verfahren ist ein Nachrichtenaustausch, der üblicherweise in Client-Server-Architekturen eingesetzt wird.
Beim Request-Response-Verfahren sendet die anfragende Station eine Anfrage, nämlich den Request, an einen anderen Computer. Die Anfrage bezieht sich in aller Regel auf einen kleinen Teil einer Aufgabe. Sie wird vom zweiten Computer empfangen, bearbeitet und ausgeführt und mit dem Response, das ist die Antwort, beantwortet.
Wie das Request-Response-Dienstmodell arbeitet, verdeutlicht das HTTP-Protokoll. Dabei richtet ein Web-Browser einen HTTP-Request an den Webserver und fragt diesen nach einem HTML-Dokument. Der Webserver überträgt in seinem HTTP-Response das gewünschte Dokument an den Web-Client. Bei der erwähnten Anwendung wird die Verbindung zwischen den beiden Kommunikationspartner solange offen gehalten, bis die Antwort beim Client eingegangen ist, oder bis die Zeitbegrenzung überschritten ist.

Reset-Flag

Die Reset-Flag (RST) ist ein Bit-Indikator im Control-Flag-Feld des TCP-Headers. Ein gesetztes Flag zeigt an, dass der Sender die Verbindung beenden will.



Daraufhin informiert das TCP-Protokoll die Applikation über die Beendigung der Verbindung. Nach Abbau der Verbindung verbleibt die Connection für die Zeit der doppelten Lebensdauer des Segmentes in einem "Time WaitZustand.

Ringtopologie

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. Um Überschneidungen zu verhindern, sind bei dieser Art der Vernetzung besondere Adressierungsverfahren nötig. Da jeder Teilnehmer gleichzeitig als Repeater wirken kann (wenn keine Splitter eingesetzt werden), können auf diese Art große Entfernungen überbrückt werden. Vor- und Nachteile:

+ Deterministische Rechnernetzkommunikation ohne Paketkollisionen
+ Garantierte Übertragungsbandbreite
+ Alle Rechner haben gleiche Zugriffsmöglichkeiten
+ Leicht Programmierbar
— Niedrige Bisektionsweite und Konnektivität
— hohe Latenzen zu entfernten Knoten
— Ohne Ringverteiler hoher Verkabelungsaufwand
— Leichte Abhöhrmöglichkeiten von Datenübertragungen
— Langsamere Datenübertragung bei vielen angeschlossenen Endgeräten

Siehe auch: *Topologie (S. G82)*.

RIP (routing information protocol)

Beim Routing arbeitet man mit Routingtabellen, die bei statischem Routing manuell angelegt werden, bei dynamischem Routing von den Routern erlernt und danach angelegt werden. Routing Information Protocol (RIP) arbeitet mit Distance-Vector-Algorithmus und ist das am häufigsten verwendete Interior Routing Protocol (IRP). Es wurde auf der Basis des XNS RIP entwickelt. Es hat sich als Standardmodul der Berkeley Software Distribution (BSD) unter Unix 4.x etabliert.

Beim RIP-Protokoll schicken alle Router in Intervallen ihre eigenen Routingtabellen als Broadcast an die anderen Router. Die Entfernung zu anderen Netzwerken wird dabei in Relation, d.h. aus der Sichtweise der eigenen Routing-Tabelle angegeben. Auf der Basis der empfangenen Tabellen berechnen die Router die kürzesten übermittelten Entfernungen zu

jedem Zielnetz und nehmen den Nachbar-Router, der diese Entfernung bekannt gegeben hat, als Ziel-Router zur Weiterleitung. Die maximale Entfernung darf 15 Hops betragen, der Wert 16 wird Infinity genannt und besagt, dass der Wert nicht erreichbar ist.
Das RIP-Protokoll gibt es in zwei Versionen: die Version 2 von RIP, RIPv2, baut auf dem Distance-Vector-Algorithmus auf und benutzt wie RIPv1 das User Datagram Protocol (UDP) für den Transport der Routingtabellen. Die Header beider Protokolle haben identische Längen. Die nicht benutzten Datenfelder des Headers in der Version 1 nutzt RIPv2 für die Angabe der Subnetze (Subnet Mask, 4 Oktetts) und den Eintrag für den nächsten Hop (Next Hop, 4 Oktetts). Mit dieser Angabe wird die nächste Adresse festgelegt, die das Datenpaket durchlaufen muss. In dem Datenfeld Metric Count wird der Hop-Zähler geführt. Bei jedem Durchlaufen des Datenpaketes durch einen Router wird der Metric Count um eine Zählereinheit höher gesetzt. Der maximale Wert beträgt 16 und bedeutet, dass das Netzwerk nicht mehr erreichbar ist.
Das Routing Information Protocol ist von der Internet Engineering Task Force (IETF) spezifiziert und in RFC 1058 beschrieben. Für IPv6 ist es unter RFC 2080 und in der Version RIP2 in RFC 2453 beschrieben.

RR (resource record)

Als Resource Records (RR) werden Informationseinträge im Domain Name System (DNS) und in einer Zonendatei für die Verwaltung einer Domain bezeichnet. Resource Records haben eine feste Struktur: "Domain (NAME) - Klasse (CLASS) - Typ (TYPE) - Eintrag (RDATA)".
In der RR-Struktur gehört Start of Authority Resource Record (SOA-RR) zu den RR-Typen. SOA-RR ist nur einmal in einer Zonendatei enthalten und zwar zu Beginn des Strukturbereichs "Typ". Der Name Server Resource Record (NS-RR) gehört auch zu den RR-Typen und kann mehrmals vorhanden sein, nämlich jeweils für die verschiedenen Nameserver (NS). Weitere wichtige Resource Records sind der MX-Record für den Namen des Mail-Servers, der PTR-Record für die IP-Adresse, der A-Record für die IPv4-Adresse und der AAAA-Record für die IPv6-Adresse. Diese beiden Records werden beim Dual Stack von IPv4 auf IPv6 in der Zonendatei gesetzt. Bei Dual Stack handelt sich dabei um das Migrationskonzept für den Übergang vom IPv4-Protokoll auf das IPv6-Protokoll, bei dem beide IP-Protokolle parallel arbeiten und von allen Netzknoten unterstützt werden.
Neben den erwähnten RR-Typen gibt es diverse weitere für die IP-Adressen der Hosts, für Zertifikate, komplette Domainnamen, für Verschlüsselungsverfahren, Informationen über den Host, Mailboxen, für die Vermittlung und Weiterleitung und von Mails, für Zugangspunkte im Netzwerk, für digitale Zertifikate und viele mehr.
Damit die Resource Records global lesbar werden, sind in mehreren RFCs Syntax-Regeln für die RR-Typen festgelegt. Zu diesen Regeln gehört, dass zur Beschreibung der RR-Typen keine Leerzeilen zulässig sind, dass Kommentare durch Anführungszeichen kenntlich gemacht werden oder dass längere Resource Records in Klammern gesetzt werden.

RSTP (rapid spanning tree protocol)

Rapid Spanning Tree (RSTP) ist ein von der IEEE-Arbeitsgruppe 802.1w standardisiertes Redundanzverfahren mit einer geringen Reaktionszeit. Geht man beim Spanning-Tree-Protokoll im Falle einer Reorganisation der Netzstruktur noch von Strukturierungszeiten aus, die bis hin zum mehrstelligen Sekundenbereich liegen, so reduzieren sich diese Zeiten bei RSTP auf einige hundert Millisekunden, die allerdings nicht garantiert werden können, was für Service Level Agreements (SLA) besonders wichtig wäre. Die Fehlerreaktionszeit liegt bei etwa 500 ms, die Initialisierung bei ca. 400 ms.
Kritikpunkte an RSTP sind die fehlende Fehlerverwaltung und dass diese Technik in Ringtopologien nicht eingesetzt werden kann.
RSTP ist abwärtskompatibel zum Spanning-Tree-Protokoll und wird u.a. beim zeitversetzten Internetfernsehen eingesetzt. Bei RSTP arbeitet die Netzkonfiguration auch bei Ausfall der bevorzugten Verbindung weiter und erst wenn eine neue logische Baumstruktur berechnet

ist, wird das Netzwerk auf diese umgestellt.
Ein weiteres von IEEE 802.1s spezifiziertes Spanning-Tree-Protokoll ist das Multiple Spanning Tree Protocol (MSTP) bei dem größere Netze in viele Spanning-Tree-Instanzen unterteilt werden.

Schicht (Layer)

In der Kommunikation werden die Funktionalitäten in einzelnen Schichten eines Schichtenmodells festgeschrieben. Schichtenmodelle gibt es für alle Netzwerkarchitekturen und Netzwerkbetriebssysteme wie die SNA-Architektur, DECnet, Transdata, dem Department of Defense (DoD), NetWare, Vines, Windows NT oder das standardisierte OSI-Referenzmodell. Diese Modelle bestehen aus mehreren Schichten, auf denen Elemente mit vergleichbaren Funktionen residieren. Jede Schicht beschreibt die Funktionen der Elemente. Die Schichten erleichtern Entwicklern die Implementierung von Diensten, da die Funktionen und Schnittstellen eindeutig definiert sind.
Schichten fassen mehrere gleichartige Instanzen zusammen.

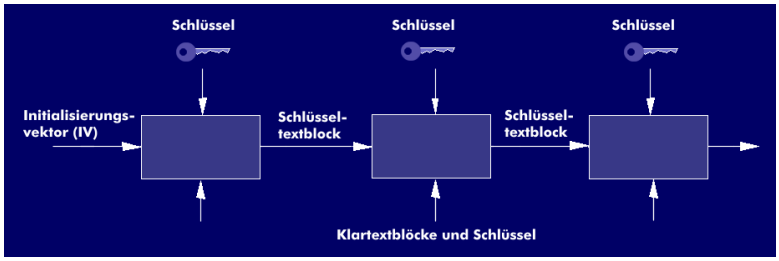
Schichtenarchitektur

System von Schichten, bei denen die Funktion der einzelnen Schichten und das Prinzip der Interaktion untereinander festgelegt ist. *Siehe auch*: *Schicht (S. G66)*.

Schlüsselblock-Kette (CBC)

Cipher Block Chain (CBC) ist ein Verschlüsselungsverfahren. Bei diesem Blockchiffre wird ein Datenblock eines zu chiffrierenden Klartextes durch eine XOR-Verknüpfung mit dem vorherigen Textblock verknüpft, bevor er mit einem Verschlüsselungsalgorithmus wie dem DES-Algorithmus verschlüsselt wird.
Es besteht somit eine Abhängigkeit der Chiffrierblöcke von allen vorhergehenden Chiffrierblöcken. Durch die Verkettung der Datenblöcke können diese weder verändert noch vertauscht oder gelöscht werden, ohne, dass dies erkannt würde. Damit dieser Verkettungsprozess auch für den ersten zu verschlüsselnden Block funktioniert, wird ein auf beiden Seiten abgesprochener Startwert, der Initialisierungsvektor (IV), benötigt. Dieser Initialisierungsvektor ersetzt im ersten Schritt den rückgekoppelten verschlüsselten Vorgängerblock.

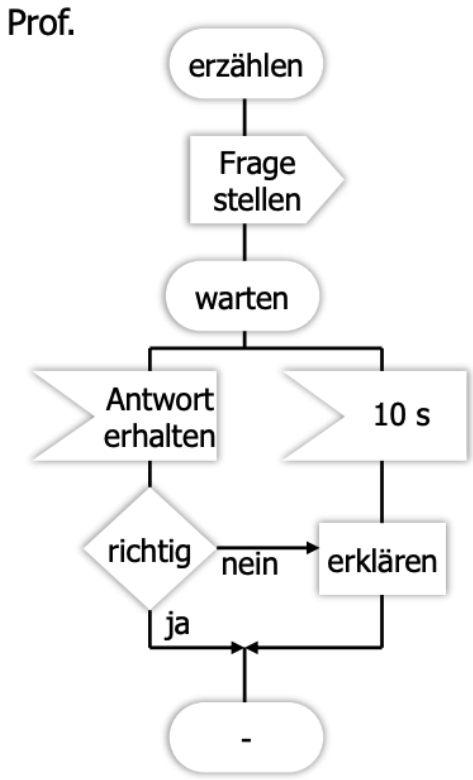
	Blockchiffre	Verschlüsselungsart
ECB	Electronic Code Book	Jeder Datenblock wird einzeln verschlüsselt. Blockchiffre ohne Rückkopplung.
CBC	Cipher Block Chain	Verknüpfung eines Datenblocks mit dem vorhergehenden Chiffratblock.
CFB	Cipher Feedback	Kleine Klartextinkremente werden in Geheimtext gewandelt.
OFB	Output Feedback	Blockchiffre mit einer Rückkopplung ähnlich CBC.



Es gibt zwei verschiedene CBC-Verfahren: Das CBC-MAC (Message Authentication Code), bei dem ein MAC-Code mit Hilfe einer Blockchiffre konstruiert wird und den CBC-Modus, bei dem ein Block vor der Verschlüsselung mit dem vorherigen, bereits chiffrierten Block in einer XOR-Funktion verknüpft wird.

SDL (specification and description language)

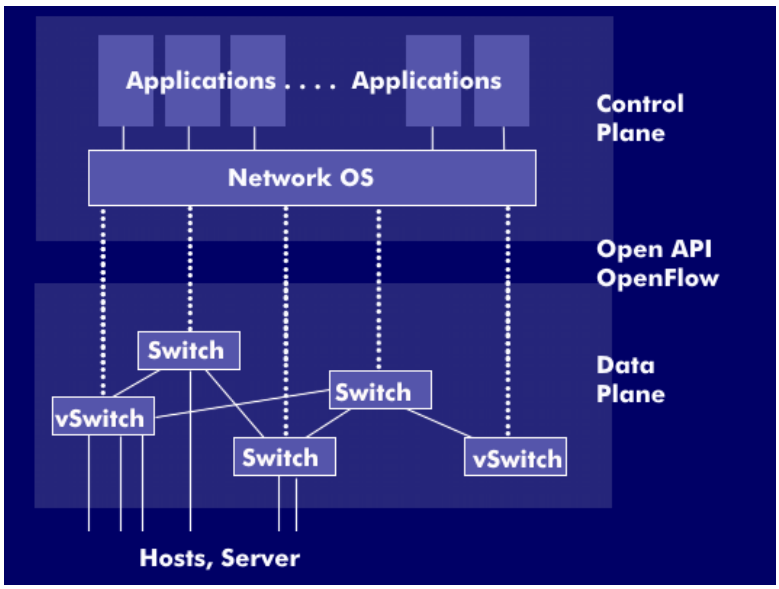
Die Specification and Description Language (SDL) ist eine Spezifikations- und Beschreibungssprache der internationalen Fernmeldeunion (ITU) mit dem der das Verhalten von Vermittlungssystemen und deren Entwicklung unterstützt werden. Die SDL-Beschreibungssprache stammt aus den 70er Jahren und wurde von der ITU unter den Standards Z.100ff standardisiert.
Die Specification and Description Language dient der Beschreibung von Funktionen, Protokollen und Diensten innerhalb und zwischen Vermittlungsstellen und Endeinrichtungen. Die Beschreibungssprache basiert auf grafischen Elementen mit denen Echtzeitsysteme beschrieben werden. Es gibt grafische Elemente für die Beschreibung der Architektur und andere für die Beschreibung des Verhaltens. Die Verwendung der grafischen Symbole und deren Reihenfolge sind durch Regeln festgelegt. Es gibt zwei syntaktische Ausprägungen, das sind das Graphical Representation (SDL/GR) und das Phrase Representation (SDL/PR). Dabei handelt es sich um verschiedene Darstellungsformen der gemeinsamen Semantik. SDL wurde ständig weiterentwickelt und hat u.a. in OSDL die objektorientierte Programmierung und die abstrakte Syntax ASN.1 einbezogen.
Mit der SDL-Beschreibungssprache können die hierarchische Systemstruktur beschrieben werden, ebenso die Systemzustände und die Ablaufsteuerung. Die Systemstruktur von der SDL-Sprache besteht aus Blöcken, in denen Prozesse ablaufen. Die Blöcke sind untereinander verbunden und tauschen über die Verbindung Informationen miteinander aus. Die ablaufenden Prozesse sind unabhängig voneinander und können gleichzeitig ablaufen.
Ein Beispiel für eine Verhaltensbeschreibung einer Proffesoreninstanz in einer Vorlesung sei in der folgenden Abbildung gegeben:



SDN (software defined networking)

Software Defined Networking (SDN) ist ein intelligenter Ansatz für das Networking, bei dem die Netzwerksteuerung softwaremäßig erfolgt und von der Hardware entkoppelt ist. Für die Entkopplung stehen in den Netzwerkkomponenten zwei Ebenen zur Verfügung: die Control Plane (CP) und die Data Plane (DP). Die Entscheidungen für den Transport der Datenpakete werden auf der Control Plane getroffen, die unabhängig ist von der darunter liegenden Data Plane, die für die Weiterleitung der Datenpakete zum Empfangsort verantwortlich ist.

Bei konventionellen Netzwerken werden die Datenpakete von den Switches nach den Vorgaben der proprietären Firmware weitergeleitet. Die Datenpakete gelangen über die gleichen Verbindungswege zum Empfangsort. In intelligenten Switches werden die verschiedenen Datenpakete analysiert und erreichen ihr Ziel über unterschiedliche Verbindungen.



Anders ist es beim Software Defined Networking (SDN) bei dem der Administrator durch Traffic-Shaping die Verbindungswege optimieren kann. Falls erforderlich kann der Administrator die Vorgaben der Switches in Bezug auf Priorisierung oder durch Ausfiltern von bestimmten Datenpaketen ändern. Diese Technik bietet sich besonders für das Cloud-Computing mit vielen verschiedenen Clouds an, da der Administrator die Verkehrslast mittels Traffic-Shaping flexibel und effizient verwalten kann. Hinzu kommt, dass preiswertere Switches eingesetzt werden können. Durch die Trennung der Control Plane von der Data Plane können verschiedene Anwendungen wie die Netzwerkvirtualisierung wesentlich einfacher realisiert werden.

Eine der bekanntesten Spezifikationen für Software Defined Networking ist der offene Standard OpenFlow von der Open Networking Foundation (ONF) über den die Administratoren Routingtabellen kontrollieren können. Weitere Infrastrukturprotokolle sind Virtual Extensible LANs (VXLAN) und Network Virtualization using Generic Routing Encapsulation (NVGRE).

Security Association Database (SAD)

Die Security Association Database (SAD) ist eine Datenbank auf einem IPsec-System für aktive SAs im ein- und ausgehenden Verkehr mit IP-Quell- und -Zieladresse, Algorithmen und Schlüssel für Verschlüsselungen und MAC.

Sicherheitsmanagement (*security*)

Das OSI-Sicherheitsmanagement ist ein Funktionsbereich des OSI-Managements und des Fault, Configuration, Account, Performance, and Security Management (FCAPS) und hängt mit der Zielspezifikation der Benutzerverwaltung unmittelbar zusammen. Sicherheitspolitische Aspekte müssen ethische und gesetzliche Komponenten ebenso berücksichtigen wie rechtliche, organisatorische und wirtschaftliche Voraussetzungen.

Das Sicherheitsmanagement (SM) umfasst den Schutz von Informationen. Dies kann sich auf den Schutz von Objekten, von Diensten und Ressourcen auswirken. Zu den Sicherheitsmaßnahmen gehören u.a. die Authentifizierung, die Passwortverwaltung und die Zugriffsberechtigung auf Netze und LAN-Segmente. Sicherheitsbetrachtungen müssen unter der Prämisse geplant werden, dass Informationen einen Wert darstellen, der quantifiziert und qualifiziert werden kann.

Die Datenbasis des OSI-Sicherheitsmanagements bildet die Security Management Information Base (SMIB). Die OSI-Sicherheitsarchitektur kennt drei Management-Kategorien: System Security Management, Security Services Management und Security Mechanismen Management. Die Abwicklung des Sicherheitsmanagements zwischen den Endsystemen erfolgt über Sicherheitsprotokolle. Dabei müssen die Sicherheitsprotokolle und die übertragenen Management-Informationen geschützt werden. *Siehe auch*: *FCAPS (S. G20)*.

Sicherheitsvereinbarung

Security Associations (SA) sind Sicherheitsvereinbarungen, die zwei mittels IPsec miteinander kommunizierende Instanzen vor der Kommunikation untereinander austauschen. Diese Sicherheitsvereinbarungen werden für den Authentication Header (AH) und den Encapsulated Security Payload (ESP) jeweils individuell getroffen. Sie gelten für die unidirektionale Kommunikation, also nur für eine Übertragungsrichtung. Da eine Kommunikation bidirektional erfolgt, sind mindestens zwei Sicherheitsvereinbarungen für die Übertragung erforderlich: eine für beispielsweise für die Verschlüsselung eines Datenpakets, die zweite für die Authentifizierung.

Security Associations sind die grundlegende individuelle Basis jeder IPSec-Verbindung. Sie definieren exakt, wie der Host oder das Security Gateway eine Verbindung zur Zielkomponente aufbauen und erhalten muss. Eine Security Association ist stets einzigartig und wird durch drei wesentliche Komponenten beschrieben: Den Security Parameter Index (SPI) die IP-Zieladresse und den Security Protocol Identifier.

Zur Vereinfachung des Verfahrens benutzt man so genannte Domain of Interpretation (DOI), in der die verschiedenen Parameter festgelegt sind.

Sicherungsschicht (data link layer) (OSI)

Die Aufgabe der Sicherungsschicht, der Schicht 2 des OSI-Referenzmodells, betrifft die fehlerfreie Übertragung von Frames über Übertragungsabschnitte, die zwei Stationen ohne Zwischenschaltung von Vermittlungsknoten miteinander verbinden. Diese Funktion wird als Bridging bezeichnet.

Ungesicherte Systemverbindungen werden durch diese Schicht zu gesicherten Systemverbindungen modifiziert. Sie abstrahiert von den physikalischen Verbindungswegen. Ihre Aufgabe ist es, die Verbindungen zu verwalten. Dazu gehört auch das Anfordern und Freigeben. Sie fasst Folgen von Informationen zu Datenpaketen zusammen bzw. löst (größere) logische Einheiten, die von einer oberen Ebene kommen, zu (kleineren) Datenpaketen auf. Fehler werden im Zusammenhang mit ganzen Datenpaketen betrachtet. Zudem werden hier Fehler, die auf der Bitübertragungsschicht nicht feststellbar sind, erkannt und zum Teil korrigiert.

Die Kommunikation auf der Schicht 2 kann sowohl verbindungslos als auch verbindungsorientiert erfolgen. Bei der verbindungslosen Kommunikation müssen alle zu übertragenden Einheiten mit Quell- und Zieladressen versehen werden (Beispiel: Token-Ring-Frame), während bei der verbindungsorientierten Kommunikation vor dem eigentlichen Datenaustausch logische Verbindungen zwischen den kommunizierenden Endgeräten aufgebaut werden müssen. *Siehe auch*: *Token Ring (S. G82) und OSI (S. G41)*.

Simplex-Betrieb (Übertragungsart)

Der Simplex-Betrieb ist eine Übertragungsart, bei der die Kommunikation nur in einer Richtung möglich ist. Bei dieser Betriebsart kann die eine Datenstation nur senden, während die andere nur empfangen kann, womit keine Rückmeldung oder Fehlerkorrektur möglich ist. Eine anderer Name wäre Richtungsbetrieb.

Sitzungsschicht (session layer) (OSI)

Eine Sitzung des Session Layers ist die logische Verbindung zwischen zwei Arbeitseinheiten der obersten Ebene, die miteinander kommunizieren. Eine Sitzung muss dann errichtet werden, wenn ein Anwenderprozess eines Rechners mit einem Anwenderprozess eines anderen Rechners in Verbindung treten will.

Die Sitzungsschicht ermittelt dann die Adresse der entsprechenden Sitzungs-Arbeitseinheit des Partnerrechners und fordert eine entsprechende Verbindung von der Transportschicht an. Für die Steuerung der Kommunikation werden den Anwendungsprozessen Dienste bereitgestellt, die ihnen das Organisieren und Synchronisieren ihres Dialogs ermöglichen.
Die Dienste sind in Funktionseinheiten zusammengefasst und werden für den Sitzungsaufbau vereinbart. Von der internationalen Standardisierungs-Organisation (ISO) wird eine Zusammenfassung der Funktionseinheiten in drei Klassen vorgeschlagen: Basic Combined Subset (BCS), Basic Synchronized Subset (BSS) und Basic Activity Subset (BAS). *Siehe auch*: *OSI (S. G41)*.

SMTP (simple mail transfer protocol)

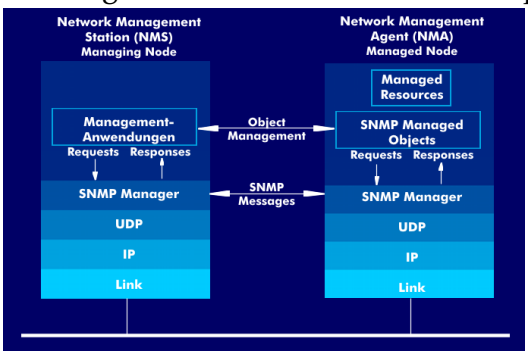
Das SMTP-Protokoll (Simple Mail Transfer Protocol) ist der Internet-Standard zur Verteilung von E-Mails. Das SMTP-Protokoll ist ein älteres Anwendungsprotokoll aus dem Jahr 1982. Erstmals wurde es in RFC 821 beschrieben, später, im Jahr 2001 wurde eine erweiterte Version unter RFC 2821 veröffentlicht. Das Protokoll ist textorientiert und setzt auf dem TCP-Protokoll auf.
Die SMTP-Nachricht besteht aus dem Header und den Nutzdaten. Der Header enthält u.a. Datum, Bezug, Empfänger, Absender, Kopien-Empfänger, wobei der Benutzer für jeden dieser Einträge durch einen »Prompt« angesprochen wird. Der Nutzdatenteil besteht typischerweise aus freiem ASCII-Text. Nachrichten mit mehreren Empfängern auf einem Ziel-Host werden nur einmal zum Ziel übertragen und dort verteilt.
Da das SMTP-Protokoll selbst keine Nachrichten abrufen kann, wird diese Aufgabe vom POP-Protokoll (POP3) oder vom IMAP-Protokoll übernommen. E-Mails werden in den Message User Agent (MUA), das ist der Mail-Client, eingegeben und mit dem SMTP-Protokoll an den Mail Transfer Agent (MTA) ausgeliefert. Der Mail User Agent (MUA) ist der SMTP-Client, der Mail Transfer Agent der SMTP-Server. Wenn der Mail Transfer Agent die Nachricht an einen Server senden will, agiert er als Mail User Agent, also als Client. Das SMTP-Protokoll ist ein Client-Server-Protokoll, das die Nachrichten vom SMTP-Client über eine TCP-Verbindung zum SMTP-Server überträgt. Sobald der Sever die komplette Nachricht empfangen hat, sendet er eine Bestätigung an den Client. Das Übertragen der Nachrichten wird durch SMTP-Befehle mit einer speziellen Syntax unterstützt.
SMTP definiert nicht wie eine Nachricht von einem oder zu einem Benutzer vermittelt wird und es ist auch nicht festgelegt, wie empfangene Nachrichten vom Benutzer präsentiert oder zwischengespeichert werden. Diese Aufgaben werden für das SMTP-Protokoll von anderen Applikationsprogrammen durchgeführt. Im Vergleich zu X.400 handelt es sich bei SMTP um ein funktional ärmeres Protokoll aus dem Jahr 1986, das von der Internet Engineering Task Force (IETF) spezifiziert und im RFC 821 erstmals beschrieben wurde. Mit entsprechenden SMTP-Erweiterungen, dem Extended SMTP (ESMTP), befassen sich die RFCs 1891 und 2821.

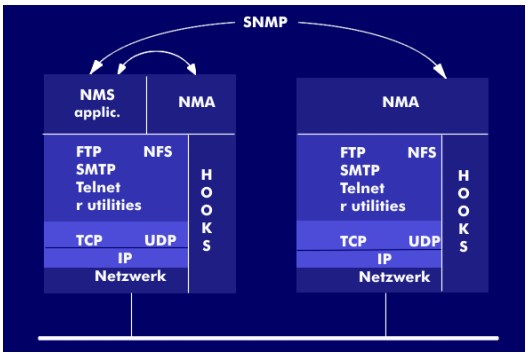
SNMP (simple network management protocol)

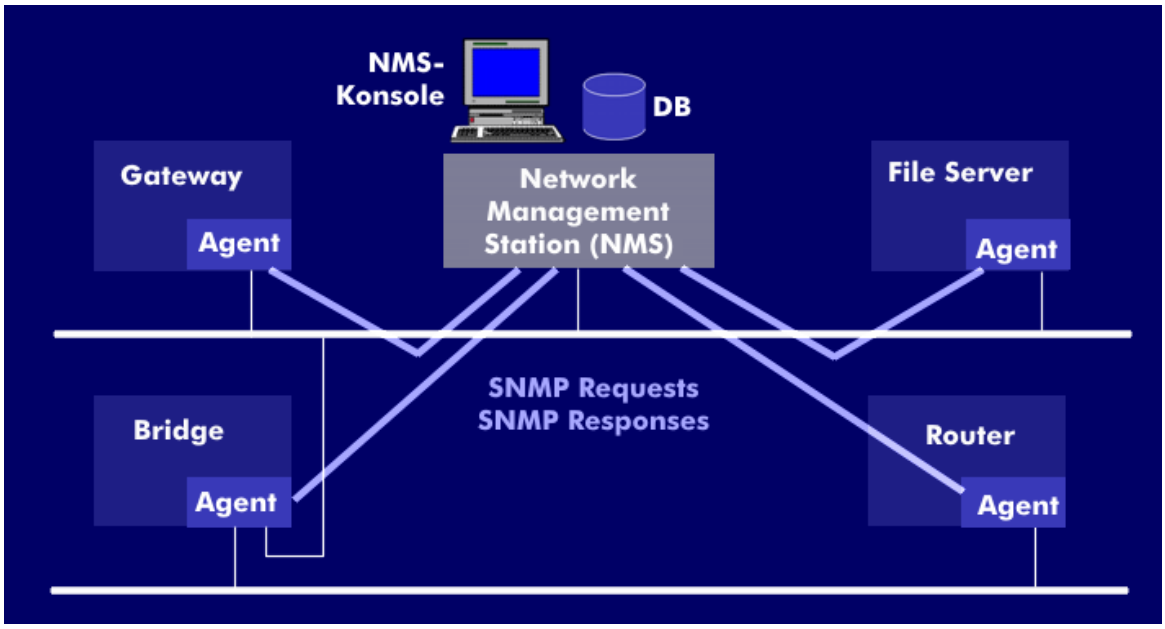
Das Simple Network Management Protocol (SNMP) erlaubt ein zentrales Netzwerkmanagement für viele Netzwerkkomponenten. Die primären Ziele des SNMP-Protokolls sind die Verringerung der Komplexität der Management-Funktionen, die Erweiterbarkeit des Protokolls und die Unabhängigkeit von irgendwelchen Netzwerkkomponenten.
Es gibt drei Managementbereiche, in denen dieses Protokoll eingesetzt werden kann:

- Beim Monitoring können Ereignisse aufgezeichnet und Netzwerkstatistiken gesammelt werden.
- Das Controlling unterstützt das Ändern von Geräteparametern und -variablen.
- Bei der Administrierung werden Informationen aufgezeichnet, die die Entwicklung des Netzwerk-Aufbaus beschreiben.

Das Architekturmodell von SNMP ist sehr einfach. Nach diesem Modell teilt sich ein Netzwerk auf in die Stationen für das Netzwerkmanagement, die Network Management Stations (NMS), und die Netzwerkkomponenten, die Network Management Agents (NMA). Die NMS-Stationen führen Anwendungen zur Überwachung und Steuerung der NM-Agenten aus. Die Netzwerkkomponenten Router, Host, Bridge oder Terminalserver haben Netzwerk-Management-Agenten, die die Management-Funktionen ausführen. Mittels SNMP findet zwischen der Managementstation (NMS) und den Netzwerkkomponenten (NMA) eine Kommunikation statt. Diese Kommunikation umfasst die Übertragung relevanter Management-Daten, die von den Netzwerkkomponenten gesammelt wurden und in der MIB abgelegt sind und zur Managementstation übertragen werden, ebenso wie die Steuerungsdaten für die Netzwerkkomponenten.







Beispiele solcher Informationen sind Netzwerkadressen, physikalische Adressen, Timer, Zähler und Protokollparameter. Der Agent reagiert auf SNMP-Anfragen des Managers durch die MIB-Werte abgefragt oder auch gesetzt werden. Letzteres bewirkt das Steuern des Netzwerkknoten. Zusätzlich kann der Agent von sich aus aufgrund bestimmter Ereignisse eine sogenannnte Trap Message (Ereignismeldung) senden.

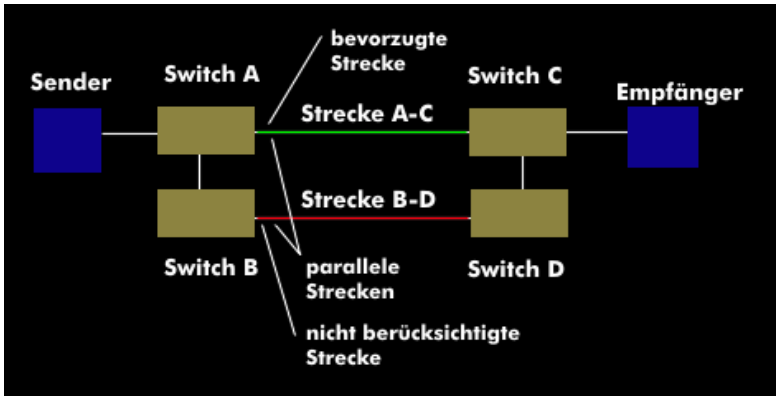
SNMP verwendet Abstract Syntax Notation One (ASN.1) als Präsentationssprache. Dank der Flexibilität von ASN.1 können Netzwerkobjekte je nach Bedarf zu der Management Information Base (MIB) hinzugefügt werden. Eine SNMP-Version mit erweiterten Eigenschaften und Definitionen ist das Nachfolger-Protokoll SNMPv2.

Die grundlegenden Dokumente zu SNMP sind in den RFCs 1157, 1441 und 3410.

Spanning-Tree-Protokoll

Das Spanning-Tree-Protokoll (STP) ist ein Redundanzverfahren zur Schleifenunterdrückung und damit zur Pfadoptimierung in Netzwerken. Bei diesem Verfahren werden physikalisch

redundante Netzwerkstrukturen ermittelt und in einer zyklenfreien Struktur abgebildet. Diese Maßnahme reduziert die aktiven Verbindungswege einer beliebig vermaschten Netzwerkstruktur und führt sie in eine Baumtopologie, daher die Bezeichnung Spanning Tree.
Mathematisch betrachtet ist eine Baumtopologie so geartet, dass alle vernetzten Punkte nur durch einen Weg miteinander verbunden sind. Außerdem sind alle vernetzten Punkte von allen anderen vernetzten Punkten aus erreichbar, zudem gibt es zwischen zwei beliebigen vernetzten Punkten keine Zyklen.
Erst wenn der bevorzugte Verbindungsweg unterbrochen ist, wird das Netzwerk neu organisiert und die optionalen Verbindungsstrecken aktiviert. Zur Überprüfung der Verbindungsstrecken kommunizieren die Switches oder Bridges regelmäßig mit Konfigurations-Datenpaketen, den Bridge Protocol Data Units (BPDU). Fällt eine Strecke aus kommen die BPDU-Datenpakete nicht mehr an und das SPA-Protokoll sorgt für eine Reorganisation.



Das Spanning-Tree-Protokoll wurde von Digital Equipment (DEC) entwickelt und später in abgewandelter Form von IEEE 802.1d übernommen. Der Algorithmus ist in entsprechenden Brückentypen implementiert, wobei jede Brücke im Rahmen bestimmter Optimalitätskriterien den Weg hin zur Wurzel der Baumtopologie berechnet. Als Berechnungsparameter können Entfernungen, Kapazitäten, Kosten oder Verkehrsbelastungen herangezogen werden.
Da das konventionelle Spanning-Tree-Verfahren eine lange Konvergenzzeit von ca. 30 Sekunden für die Rekonfiguration der Verbindungen benötigt, hat die IEEE-Arbeitsgruppe 802.1 mehrere Spanning-Tree-Verfahren spezifiziert, so das Rapid Spanning Tree Protocol (RSTP) mit verkürzter Reaktionszeit und als Weiterentwicklung das Multiple Spanning Tree Protocol (MSTP) mit dem die Rekonfigurationszeiten noch weiter verkürzt werden.

SPDY (speedy)

Das SPDY-Protokoll (Speedy) wurde von Google für die Beschleunigung des Webseiten-Aufrufs vorgeschlagen. Der Aufruf von Webseiten erfolgt durch die Anfrage des Web-Clients an den Webserver. Die Übermittlung der Webseiten vom Webserver zum Web-Client erfolgt über das Hypertext Transfer Protocol (HTTP) und das TCP-Protokoll. Da das HTTP-Protokoll nicht auf minimale Latenzzeiten optimiert ist, und sich die Webseiten in den letzten zehn Jahren stark verändert haben, bildet es einen Engpass in der Webseiten-Übermittlung.
Da Webseiten aus mehreren Details wie Texte, Grafiken oder Videos bestehen, werden die Webseitendetails jeweils über einzelne TCP-Verbindungen zum Web-Browser übertragen. Hinzu kommt, dass Browser gleichzeitig nur eine begrenzte Anzahl an Verbindungen unterstützen. Beides, der vielfache Verbindungsaufbau und die begrenzte Anzahl an Verbindungen reduzieren die Latenzzeit.
Das Google SPDY-Protokoll, was für Speedy steht, setzt auf TCP/IP, soll aber das HTTP-Protokoll ablösen. Die Geschwindigkeitserhöhung wird durch die Reduzierung

des Overheads, durch das Interleaving der Anfragen des Clients und durch persistente Verbindungen erreicht.
Was die Datenreduzierung betrifft, so sind die Header-Informationen komprimiert, die Anfragen des Web-Clients werden nicht sequenziell übertragen, sondern in einer Multiplextechnik mit Interleaving ineinander verschachtelt, und was die persistente Verbindungen betrifft, so sind diese dauerhaft und werden nicht für jeden Request neu aufgebaut. Durch diese Maßnahmen und durch Priorisierung der Requests erfolgt die Übertragung von Webseiten zum Web-Client zwischen 10% und 50% schneller.
Einen ähnlichen Ansatz verfolgen HTTP 2.0, kurz HTTP/2, und das QUIC-Protokoll, Quick UDP Internet Connections (QUIC), mit denen ebenfalls ein schnellerer Seitenaufruf realisiert wird. HTTP/2 komprimiert den HTTP-Header mit HPACK und das QUIC-Protokoll, das auf das UDP-Protokoll setzt und die Beschränkungen des SPDY-Protokolls durch das darunter liegende TCP-Protokoll aufhebt.

SPI (security parameter index)

Der Security Parameter Index (SPI) ist eine Nummer, die gemeinsam mit der IP-Zieladresse und einem Sicherheitsprotokoll, eine bestimmte Sicherheitsorganisation identifiziert. Bei Benutzung des IKE-Protokolls, Internet Key Exchange (IKE), zur Bestimmung der Sicherheitsorganisation, ist das SPI für jede Sicherheitsorganisation eine statistisch ausgewählte Nummer. Bei anderen Sicherheitsprotokollen kann die SPI manuell spezifiziert werden.

SPoF (single point of failure)

Mit Single Point of Failure (SPoF) werden Systemkomponenten oder Systempfade bezeichnet, durch die im Fehlerfall das System nicht mehr betriebsbereit ist. Das trifft immer dann zu, wenn eine Komponente eine zentrale Funktion im Gesamtsystem übernimmt, nicht redundant vorhanden ist und bei Ausfall die Funktionen der anderen Komponenten beeinträchtigt.
SPoF-Komponenten sind verantwortlich für die zuverlässige Funktionalität des Gesamtsystems. Es sind die Schwachpunkte innerhalb eines Systems oder Netzwerks, die einen unmittelbaren Einfluss auf die Verfügbarkeit und Hochverfügbarkeit von Systemen haben. SPoF-Komponenten können Switches sein oder Router, nicht gesicherte Back-End-Server oder Cluster und auch nur ein defekter Stecker.
Zur Verhinderung von Ausfällen und aus sicherheitstechnischen Gründen werden in manipulationssicheren Systemen, wie sie bei Blockchains erforderlich sind, die Daten auf die angeschlossenen Computer von den Peer-to-Peer-Netzen verteilt und gleichzeitig aktualisiert. Dadurch wird sichergestellt, dass die Blockchain nicht manipuliert werden kann.

SR (selective repeat)

Das Automatic Repeat Request (ARQ) ist ein Quittungsbetrieb für die Datenblockübertragung, das mehrere Verfahren für die Fehlermeldung kennt. Neben dem kontinuierlichen Betrieb gibt es noch Stop-and-Wait ARQ, das Selective Repeat (SR) und das Hybrid Automatic Repeat Request (HARQ).
Beim Selective Repeat meldet der Empfänger fehlerhafte Blöcke an den Sender, die dieser dann erneut überträgt. Nach der Übertragung des fehlerhaften Datenblocks sendet der Sender in der ursprünglichen Reihenfolge weiter.

SSL (secure socket layer)

Das Secure Socket Layer (SSL) ist ein Protokoll zur Authentifizierung und Verschlüsselung von Internetverbindungen. Das SSL-Protokoll kann in Verbindung mit dem Simple Mail Transfer Protocol (SMTP), E-Mail, Telnet, dem FTP-Protokoll und Hypertext Transfer Protocol (HTTP), bzw. HTTPS eingesetzt werden und setzt auf TCP/IP auf. Der Datenaustausch findet beim SSL-Protokoll auf der Transportschicht statt. Sobald beide Kommunikationspartner über eine SSL-Verbindung in Kontakt, kann die Verbindung weder abgehört noch kann die Datenübertragung manipuliert werden.

Das SSL-Protokoll wurde von Netscape entwickelt und sorgt für eine komplexe 128-Bit-Verschlüsselung der Daten, die im Internet übertragen werden. Das SSL-Verfahren verschlüsselt mit öffentlichen Schlüsseln, die von einer dritten Partei nach X.509 bestätigt werden. Die Sicherheit wird dadurch garantiert, dass der Schlüssel zur Dechiffrierung nochmals individuell festgelegt werden muss und nur beim Anwender gespeichert ist - im Internet nicht übertragen wird.

Die Entwickler des SSL-Protokolls haben das Protokoll in zwei Ebenen angelegt: Die eine Ebene ist für die Verschlüsselung von Daten zuständig. Sie erlaubt verschiedene symmetrische Algorithmen, darunter der Data Encryption Standard (DES), Triple-DES oder Rivest Cipher 4 (RC4) und setzt voraus, dass beide Kommunikationspartner einen gemeinsamen, geheimen Chiffrierschlüssel besitzen, der jeweils für eine Verbindung generiert wird. Die Echtheit der Daten wird zudem durch einen Prüfsummencheck wie den Secure Hash Algorithm (SHA) oder Message Digest No. 5 (MD5) verifiziert. Auf der zweiten Ebene findet mit dem Transport Layer Security (TLS) Handshaking der Austausch der privaten Schlüssel statt. Die Server und Clients einer Kommunikationsverbindung authentifizieren sich, handeln einen Verschlüsselungsalgorithmus aus und schicken einander die codierten Sitzungsschlüssel.
Secure Socket Layer (SSL) ist in RFC 2246 beschrieben und entspricht seit der Version 3.1 dem TLS-Protokoll. Beide Protokolle sind risikobehaftet, weil die mit einer asymmetrischen Verschlüsselung verschlüsselten Daten mit dem Master-Key dechiffriert werden können.übergegangen.

Stadtnetz (MAN)

Metropolitan Area Networks (MAN) sind Stadtnetze oder Regionalnetze, die sich mit ihren charakteristischen Eigenschaften zwischen den lokalen Netzen (LAN) und den Weitverkehrsnetzen (WAN) positionieren. Bei MANs handelt es sich um Netze mit einer regionalen Ausdehnung, die für unterschiedlichste Übertragungsdienste wie Sprache, Daten, Bewegtbild usw. ausgelegt sind.
Die Ausdehnung für ein Stadtnetz liegt bei 100 km und mehr, die Übertragungsgeschwindigkeit bei 100 Mbit/s bis 1.000 Mbit/s. MANs benutzen

Lichtwellenleiter als Übertragungsmedium, um die hohen Übertragungsgeschwindigkeiten mit möglichst geringer Fehlerrate zu realisieren.
In IEEE sind Stadtnetze in 802.6 standardisiert. Hierbei handelt es sich von der Topologie her um einen Doppelbus, bestehend aus zwei unidirektionalen, gegenläufigen Bussen. Alle Stationen eines solchen Netzes sind an beide Busse angeschlossen. Der Datenverkehr kann vollduplex in jede Richtung laufen, wobei je nach Zieladresse der eine oder der andere Bus für die Übertragung gewählt wird. Dank dieser Topologie können verschiedene Netzkonzepte realisiert werden: Punkt-zu-Punkt-Verbindungen, Bustopologien oder ein Looped Bus, also eine logische Ringtopologien auf einem physikalischen Bus.

Statistisches Zeitmultiplexverfahren (STDM)

Beim statistischen Zeitmultiplexverfahren werden die Daten von den einzelnen Eingangskanälen übernommen, in einen dynamisch arbeitenden Speicher eingelesen und anschließend blockweise über die Übertragungsleitung transportiert. Dabei ist eine strikte Aufteilung von jedem Übertragungskanal auf jeden Rahmen nicht mehr gegeben, es kann vorkommen, dass ein Kanal in einem Block verhältnismäßig weniger oft auftritt. *Siehe auch*: *Multiplexverfahren (S. G54) und TDM (S. G92)*.

Sterntopologie

Bei Netzen in Stern-Topologie sind an einen zentralen Teilnehmer alle anderen Teilnehmer mit einer Punkt-zu-Punkt-Verbindung angeschlossen. Der zentrale Teilnehmer (Verteiler, beispielsweise ein Hub oder Switch) muss nicht notwendigerweise über eine besondere Steuerungsintelligenz verfügen. Vor- und Nachteile:

- \+ Ausfall von Endgeräten hat keine Auswirkung auf restliches Netz
- \+ Leicht erweiterbar und verständlich
- \+ Eignet sich besonders für Broadcast- und Multicastanwendungen
- — Netzverkehr wird unmöglich wenn Verteiler ausfällt
- — Hoher Kabelaufwand

Siehe auch: *Topologie (S. G82)*.

sTLD (sponsored top level domain)

Sponsered Top Level Domains (sTLD) gehören zu den generischen TLDs, zu denen auch die ungesponserten TLDs, die Unsponsored Top Level Domain (uTLD) gehören. Es gibt bereits diverse gesponserte TLDs, wie *.pro für spezielle Berufsgruppen, *.edu für Bildungseinrichtungen, *.aero für die Lufttransportindustrie oder *.mil für das Militär der Vereinigten Staaten von Amerika.
Die gesponserten Top Level Domains werden an Unternehmen und weltweit tätige Organisationen vergeben und können bei der Internet Corporation for Assigned Names and Numbers (ICANN) beantragt werden. Sie sind relativ teuer, sowohl was die Beantragung betrifft, als auch die jährlichen Gebühren. So könnten die gesponserten Unternehmens-TLDs den Firmennamen beinhalten und .*ebay, *.nokia oder *.vw heißen.
Die sponsered TLDs stehen unter Kontrolle der Inhaber, sie sind nicht verpflichtet die Domains anderen Interessen für deren Nutzung zur Verfügung zu stellen. Außerdem werden dadurch die Möglichkeiten des Phishings weitestgehend ausgeschlossen. *Siehe auch*: *DNS (S. G15) und gTLD (S. G26)*.

stop and wait ARQ

Stop-and-Wait ARQ ist eine Variante der Übertragungssicherung nach dem Automatic Repeat Request (ARQ). Bei Stop-and-Wait ARQ wird für jeden einzelnen Datenblock eine Quittung erteilt, bevor der nächste Datenblock übertragen werden kann. Das Verfahren hat einen geringeren Datendurchsatz als beispielsweise Continuous ARQ, dafür aber eine geringe Fehleranfälligkeit. Beispiele für Stop and Wait ARQ sind Medium Speed Version (MSV) und das Bisynchron-Verfahren (BSC).

Struktur der Managementinformation (SMI)

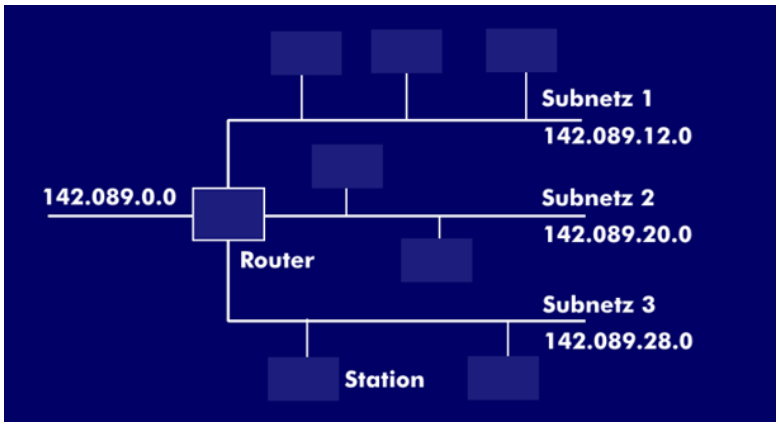
Structure of Management Information (SMI) sind Regeln mit denen Objekte definiert werden, auf die mittels eines Netzwerkmanagement-Protokolls zugegriffen werden kann. Die SMI-Information ist Teil des Simple Network Management Protocols (SNMP), in dem die Regeln für die Definition der Managed Objects in der Management Information Base (MIB) von SNMP beschrieben sind. Die Objekte werden in einer Baumstruktur, der Management Information Tree (MIT) gespeichert und in der abstrakten Beschreibungssprache ASN.1 beschrieben. Als Komponente des Netzwerkmanagements können mit der SMI-Information verwaltete Objekte dargestellt werden. In OSI-Netzwerken können Objekte mittels Management Information Base (MIB) gemanagt werden.

In RFC 1155 sind die Regeln für die Definition der Managed Objects in der Management Information Base (MIB) beschrieben. *Siehe auch*: *MIB (S. G48)*.

Subnetz

In IP-Netzen sind Subnetze Teilnetze mit eigener Subnetz-Adresse. Subnetze können über Router oder Switches zu einem größeren Netz zusammengefügt werden. Das Konzept der Subnetze bietet sich dann an, wenn sich Unternehmensstrukturen ändern und das bestehende Unternehmensnetz aus organisatorischen Gründen in mehrere Subnetze unterteilt werden muss. Dabei kann einzelnen, organisatorisch miteinander verbundenen Unternehmensteilen ein eigenes Subnetz zur Verfügung gestellt werden.

Die Bildung von Subnetzen, das Subnetting, kann auch aus Sicherheitsgründen erfolgen, um einen höheren Durchsatz zu erzielen, um Netzüberlastung zu vermeiden oder um die Lastverteilung zu verbessern. In IP-basierten Netzen wird der zur Verfügung stehende Adressraum auf die Subnetze aufgeteilt. Dazu wird die IP-Adresse über eine Subnetzmaske in die neuen Subnetz-Adressen umgesetzt.



Subnetze werden von Netzwerkadministratoren so strukturiert, dass sie ein hierarchisches Routing haben und in der Adressierung gegenüber den angeschlossenen Netzen autark sind.

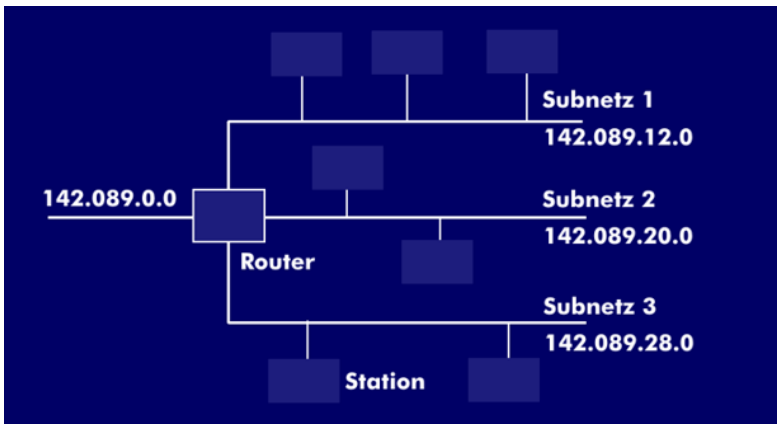
Subnetzmaske

Subnetzmasken sind Adressierungsmasken mit denen IP-Adressen in die Adressräume von Subnetzen transformiert werden. Bei der Bildung von Subnetzen, dem Subnetting, geht es darum, autarke Netze mit eigenem Adressbereich zur Verfügung zu stellen. Zu diesem Zweck trennt man den IP-Adressraum und den Adressraum für die Subnetze und teilt den neu entstehenden Subnetz-Adressraum auf die Subnetze auf.

Diese Aufteilung des IP-Adressraums hat den Vorteil, dass jedes Subnetz über die IP-Adresse angesprochen werden kann. Dieses Verfahren wird beim Classless Interdomain Routing (CIDR) eingesetzt, das wiederum bei der Bildung von IPv4-Adressen und IPv6-Adressen benutzt wird.

IP-Adressen bestehen aus den Adressen für die Netzwerkkennung und die Hostkennung und sind in Adressklassen eingeteilt. Je nach Adressklasse ist die Netzwerk- oder die Hostkennung größer. Die Subnetzmaske bestimmt an welcher Stelle die 32-stellige IP-Adresse für die

Subnetze geteilt wird und weist damit den Subnetzen einen mehr oder weniger großen Adressraum für die Arbeitsstationen zu. Der zugewiesene Subnetz-Adressraum kann eine feste oder variable Länge haben, Man spricht dann von Fixed Length Subnet Mask (FLSM) und von Variable Length Subnet Mask (VLSM). Ebenso wie die IP-Adresse besteht die Subnetz-Adresse aus einem mehr oder weniger umfangreichen Netzwerkteil und einem Hostteil. Über diese Subnetz-Adressen können die Hosts und Arbeitsstationen innerhalb eines Subnetzes adressiert werden.



Die Subnetzmaske sieht der IP-Adresse sehr ähnlich. Ebenso wie diese besteht sie aus 32 Bit, unterteilt in 4 mal 8 Bit. Die Schreibweise kann dual oder auch dezimal sein. Um die IP-Adresse auch für die Subnetze nutzen zu können, wird der Netzadressenteil der Subnetzmaske in allen Bits auf 1 gestellt. Bei einer Klasse-C-Adresse stehen damit die ersten 24 Bit auf 1, sie sind damit nicht variabel, und die folgenden 8 Bit für die Stationen stehen auf 0 und sind damit variabel. Die binär dargestellte Standard-Subnetzmaske würde in binärer Schreibweise folgendermaßen aussehen:

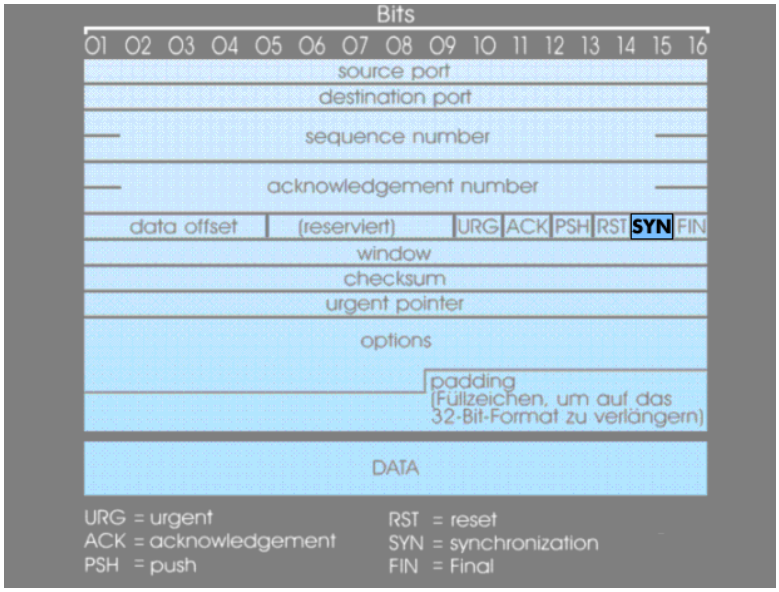
`11111111.11111111.11111111.00000000`, und in dezimaler: `255.255.255.0`.

Der variable Adressenanteil umfasst in diesem Beispiel 256 adressierbare Stationsadressen. Die erste Adresse 0000 0000 (Netzwerk) und die letzte Adresse 1111 1111 (Broadcast) sind vergeben. Es verbleiben somit 254 Stationsadressen. Für die anderen IP-Adressklassen können auf die gleiche Weise Standard-Subnetzmasken erstellt werden.
Subnetzmasken können nicht nur für die IP-Klassen, sondern für alle 24-Bit-Kombinationen der Hostadressen erstellt werden. Mit den Subnetzmasken hat man die Adressierungsmasken für die Erstellung der Subnetz-Adressen. Am Beispiel der Standard-Subnetzmaske für Klasse C soll das verdeutlicht werden, und zwar soll die Subnetzmaske für vier Subnetze aufgeteilt werden. Da ja nur der Hostanteil der Standard-Subnetzmaske variabel ist, wird dieser für die Subnetz-Adressierung benutzt. Bei vier Subnetzen müssen zwei Bit für die Adressierung belegt werden. Das bedeutet, aus der 0000 0000 werden die beiden Bits mit der höchsten Wertigkeit benutzt, der Hostanteil der Subnetzmaske sieht damit folgendermaßen aus: 1100 0000.
Die komplette Subnetzmaske sieht in dezimaler Darstellung folgendermaßen aus: 255.255.255.192. Für die Hostadressen verbleiben 6 Bit, mit denen jeweils 64 Stationen adressiert werden können. Die vier Subnetze teilen sich den früheren Adressenbereich von 254 Hostadressen, indem jedem Subnetz 64 Adressen zugeordnet werden. In diesem Beispiel würde bei der Subnetzmaske 255.255.255.192 dem ersten Subnetz der Hostbereich von .0 und .63, dem zweiten der von .64 bis .127, dem dritten der von .128 bis .191 und dem vierten der zwischen .192 und .255 zugewiesen. Jeweils bezogen auf die Hostadresse, ohne Berücksichtigung der Netzwerkadressen, da diese unverändert bleiben.
Mathematisch werden die Subnetz-Adressen über logische Verknüpfungen ermittelt. Der Netzwerkteil kann über eine AND-Verknüpfung zwischen der IP-Adresse und der

Subnetzmaske bestimmt werden. Der Hostteil wird durch Negierung der Subnetzmaske und anschließender AND-Verknüpfung ermittelt.

Synchronisations-Flag

Das Synchronisations-Flag (SYN) ist ein 1 Bit umfassender Indikator im Control-Flag-Feld des TCP-Headers. Mit einer gesetzten Flag informiert der Sender den Empfänger darüber, dass er eine Verbindung aufbauen möchte. Wobei das Transmission Control Protocol (TCP) simultan mehrere Verbindungen eröffnen kann. Danach sendet der Sender ein Synchronisation-Paket zum Empfänger.



Der Verbindungsaufbau wird von Host 1 eingeleitet, der im TCP-Header die SYN-Flag setzt. Damit zeigt er gegenüber Host 2 an, dass er eine Verbindung unter einer bestimmten Sequenznummer aufbauen möchte. Host 2 bestätigt den Verbindungsaufbau mit einem Bestätigungs-Flag (ACK). Danach werden die einzelnen Segmente, das sind im TCP-Protokoll Informationseinheiten der Transportschicht, nacheinander gesendet und bestätigt.

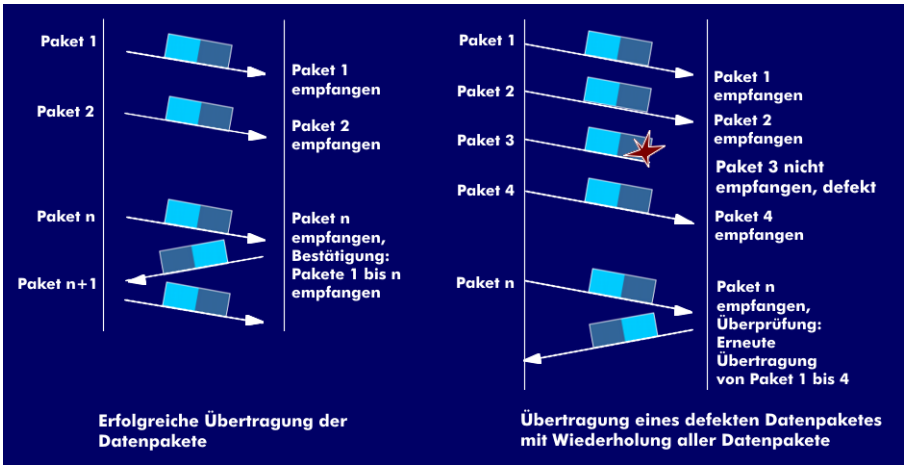
TCP (transmission control protocol)

Das Transmission Control Protocol (TCP) ist ein äußerst zuverlässiges, verbindungsorientiertes Transportprotokoll für den Einsatz in paketvermittelten Netzen. Das Protokoll baut auf dem IP-Protokoll auf, unterstützt die Funktionen der Transportschicht und stellt vor der Datenübertragung eine gesicherte Verbindung zwischen den Instanzen her. Beim TCP-Protokoll werden die Daten der höheren Schichten segmentiert und in einzelnen Datenpaketen versendet, die einen Umfang von bis zu 65 kB haben können. Das TCP-Protokoll ist ein Protokoll der Transportschicht. Das auf der Netzwerkschicht residierende Internet Protocol (IP) fragmentiert die TCP-Datensegmente in kleinere Datenpakete. Jedes Oktett eines Segments wird mit einer Sequenznummer versehen, um empfangsseitig die richtige Reihenfolge wieder herstellen zu können.

Die wesentlichen Dienstleistungen, die das TCP-Protokoll in Verbindung mit dem IP-Protokoll für die Anwendungsprozesse bereitstellt, sind die Ende-zu-Ende-Kontrolle, das Verbindungsmanagement, die Flusskontrolle, die Zeitkontrolle und das Multiplexen von Verbindungen sowie die Fehlerbehandlung.

Die Ende-zu-Ende-Kontrolle arbeitet mit einer positiven Bestätigung (ACK), bei der alle Datenpakete bestätigt, nicht empfangene erneut angefordert und gesendet werden. Durch diesen Mechanismus ist eine zuverlässige Datenübermittlung gewährleistet. Das Verbindungsmanagement sorgt für einen gesicherten Verbindungsaufbau mittels Handshake-Verfahren. Darüber hinaus sorgt das Verbindungsmanagement für die einwandfreie Bereitstellung der Verbindung während der Übertragungsphase und für einen korrekten

Verbindungsabbau.
Da alle übertragenen Datenpakete im TCP-Header fortlaufend nummeriert und bestätigt werden, verhindert die Flusskontrolle Paketverluste. Die Zeitüberwachung durch die Fenstertechnik dient dazu, dass übertragene Datenpakete innerhalb eines bestimmten Zeitraums bestätigt werden. Findet innerhalb dieses Zeitraums keine Bestätigung statt, werden die Datenpakete erneut gesendet. Um mehrere Prozesse gleichzeitig über das TCP-Protokoll zu betreiben, werden für das Multiplexen mehrere Ports zur Verfügung gestellt. Treten Fehler auf, tritt der Fehlermechanismus in Funktion und fordert die fehlerhaften Datensegmente von den höheren Schichten erneut an.



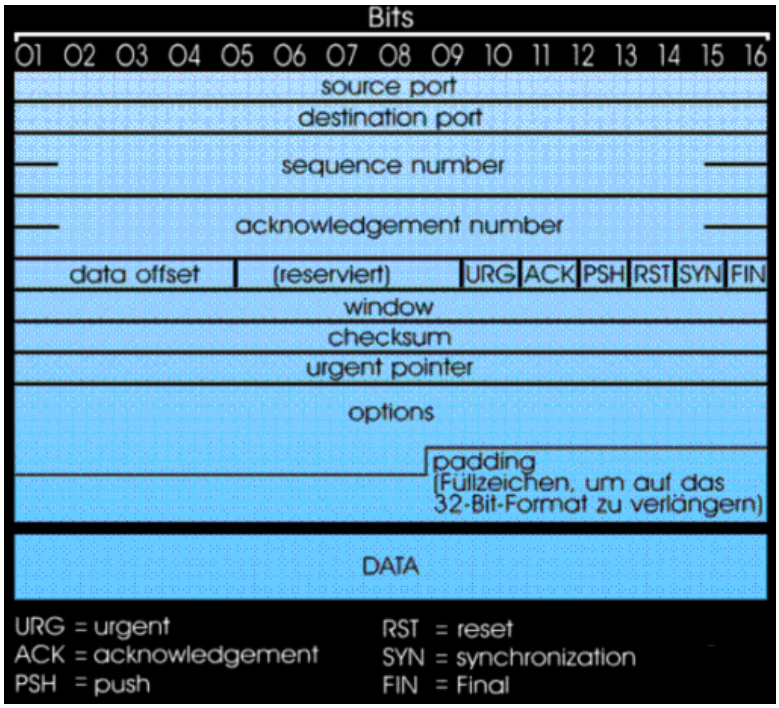
Eine TCP-Übertragung lässt sich in drei Phasen gliedern: die Initialisierungsphase, die Phase der Nutzdatenübertragung und die Phase des Verbindungsabbaus.
In der Initialisierungsphase erfolgt der aktive oder passive Verbindungssaufbau in der eine Eins-zu-Eins-Verbindung hergestellt wird, die während der gesamten Dauer des Datentransfers aufrechterhalten wird. Diese Phase, die durch einen Zwei- oder Drei-Weg-Handshake eingeleitet wird, dient auch der Synchronisation der Kommunikationspartner.
In der Datenübertragungsphase, die nach dem Verbindungsaufbau beginnt, erfolgt über die aufgebaute virtuelle Verbindung. Diese Phase ist geprägt durch die Übertragung der Datenblöcke und die Empfangsbestätigung der Sequenznummern durch den Kommunikationspartner. Die Phase des Datentransfers wird durch mehrere Timer überwacht, um beispielsweise unbestätigte Datenblöcke nachzusenden, um die Fenstergröße umzustellen, den Verbindungsabbau einzuleiten oder aber einen erneuten Verbindungsaufbau zu initiieren. Die Datentransferphase wird durch eine Flusssteuerung und durch verschiedene Algorithmen optimiert. Diese Algorithmen sind Steuerungsmechanismen für die Datenmenge, den Datenfluss und die Netzauslastung. Bekannte Algorithmen sind der Karn's Algorithmus und der Nagle-Algorithmus.
Die dritte und letzte Phase, der Verbindungsabbau, kann einerseits nach der Übertragung aller Daten erfolgen, andererseits durch einseitigen Abbruch der Verbindung durch ein höheres Protokoll.
Das TCP-Protokoll ist in RFC 793 veröffentlicht.

TCP Rate Control

TCP Rate Control ist ein Traffic-Shaping-Algorithmus zur dynamischen Bandbreitenzuordnung in Weitverkehrsnetzen. Der Algorithmus wird von Traffic-Shapern genutzt und begrenzt die Anzahl der Pakete, die ein Knoten pro Sekunde übertragen kann. Bei TCP Rate Control handelt es sich um einen Algorithmus, der auf einem statischen Weitverkehrsmodell basiert und Durchschnittswerte der Auslastung und der Fehlerrate verwendet. Kurzzeitige Änderungen des Verkehrsflusses bleiben bei diesem Algorithmus unberücksichtigt.

TCP-Header

Der TCP-Header ist der Header des Transmission Control Protocol (TCP). Er ist 20 Byte lang und setzt unmittelbar auf dem IP-Header auf. Den einzelnen Datenfeldern des TCP-Headers sind folgende Funktionalitäten zugeordnet: Die Portnummer des Senders bzw. des Empfängers, über die das Multiplexing der TCP-Pakete erfolgt. In dem Datenfeld Sequenznummer sind die Sequenznummern der gesendeten Pakete für die virtuelle Verbindung eingetragen. Mit der Bestätigungsnummer werden empfangene TCP-Datenpakete bestätigt.
Das Feld Header-Länge (IHL) gibt die Länge des Headers in 32-Bit-Worten an. Diesem Feld folgt das Control-Flag-Feld, in dem insgesamt sechs Flags gesetzt werden können: Urgent-Flag (URG), Push-Flag (PSH), Bestätigungs-Flag (ACK), Reset-Flag (RST), Synchronisations-Flag (SYN) und Final-Flag (FIN). Diese Flags dienen der Bestätigung (CONF), dem Pointer (PTR), dem unverzögerten Weiterleiten, dem Rücksetzen einer Verbindung, der Synchronisation der Sequenznummer und dem Hinweis für die Beendigung der Übertragung. Das folgende 16 Bit lange Datenfeld Window dient der Flusskontrolle. Beim TCP-Protokoll ist es ein Sliding Window, eine verschiebbare Fenstergröße, bei dem die Aufnahmefähigkeit der empfangenden Station durch Geschwindigkeitsanpassung gesteuert wird.



Über die TCP-Prüfsumme erfolgt eine Prüfsummenkontrolle der Header-Daten, über den Urgent Pointer die Prioritätensteuerung für Dringlichkeitsdaten und die Maximum Segment Size (MSS) dient der Bestimmung der maximalen Länge des TCP-Pakets.
Darüber hinaus stellt das TCP-Protokoll an das Protokoll der Vermittlungsschicht ganz bestimmte Anforderungen. So muss dieses in der Lage sein, Datagramme über einen Verbund von Netzen hinweg zu senden, den Partnern eine eindeutige Adressierung zuzuordnen, Datenpakete nach den jeweiligen Netzkonventionen zu segmentieren und zu reassemblieren sowie Informationen über die Paketreihenfolge und Sicherheitsmerkmale zu übermitteln.

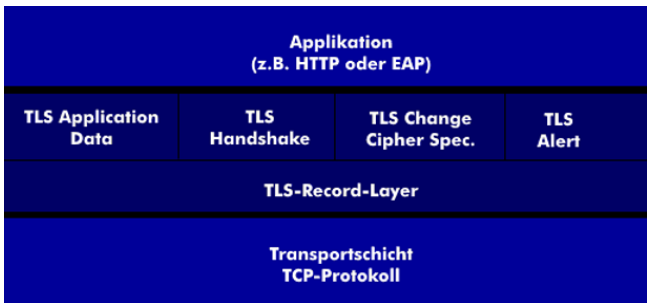
TCP/IP (transmission control protocol/internet protocol)

Die TCP/ IP-Protokolle wurden schon in den 70er Jahren von der Research Project Agency (DARPA) des US-Verteidigungsministeriums (DoD) mit Unterstützung des National Bureau of Standards (NBS) entwickelt. Ziel war die Schaffung möglichst Code-kompakter Protokolle für die Interface Message Processor (IMP) des Arpanet.
Die Entwicklung der TCP-Protokolle wurde wesentlich durch die Erfahrungen im DARPA Internet (früher ARPAnet) beeinflusst, ebenso durch Spezifikationen und sogenannte

Requests For Comments (RFC).
Der TCP/IP-Protokollstack besteht aus dem Network Interface Layer (NIL), der für die Übertragung der Datenpakete auf dem Medium und deren Empfang sorgt, dem darüber liegenden Internet Layer, der für die Adressierung, die Paketierung und die Routing-Funktionen verantwortlich ist und dem darüber liegenden Transport Layer, der den Application Layer in ihren Sitzungs- und Kommunikationsdiensten unterstützt. Das Kernprotokoll des Internet Layers ist das Internet Protocol (IP), weitere sind das Address Resolution Protocol (ARP), das Internet Control Message Protocol (ICMP) und das Internet Group Management Protocol (IGMP) und für das Routing das OSPF-Protokoll (Open Shortest Path First) und das Routing Information Protocol (RIP). Die Kernprotokolle des Transport Layers sind das verbindungsorientierte TCP-Protokoll und das verbindungslose User Datagram Protocol (UDP).
Alle Protokolle aus dem TCP/IP-Umfeld unterstützen direkt die Funktionalität der Vermittlungsschicht und der Transportschicht sowie verschiedene Anwendungen, die auf den TCP/IP-Protokollen aufsetzen. Zu den Anwendungen gehören Telnet, das File Transfer Protocol (FTP), Simple Mail Transfer Protocol (SMTP), r-Befehle, X-Window, Simple Network Management Protocol (SNMP) und diverse weitere.
Im Gegensatz zum OSI-Schichtenmodell wurden hier die Schichten 5 bis 7 zu einer Anwendungsschicht zusammengefasst. *Siehe auch*: *OSI (S. G41)*.

TLS (transport layer security)

Transport Layer Security (TLS) ist eine Weiterentwicklung des Secure Socket Layers (SSL) durch die Internet Engineering Task Force (IETF), die das SSL-Protokoll 1999 in Transport Layer Security umbenannte. Der aktuelle Standard ist in RFC 5246 beschrieben und stammt aus dem Jahr 2008.
Das TLS-Protokoll ist abwärtskompatibel zum SSL-Protokoll, es wird vorwiegend im Web-Umfeld eingesetzt und dort vor Allem zur Absicherung von HTTP-Verbindungen und für kommerzielle Transaktionen. TLS-Security bildet eine generische Sicherungsschicht oberhalb der Transportschicht und benutzt das TCP-Protokoll als verbindungsorientiertes Transportprotokoll. TLS arbeitet mit einer 128-Bit-Verschlüsselung, es wird für die Verschlüsselung von Mails eingesetzt. Um die Integrität der E-Mails zu überwachen und nichtautorisierte Zugriffe auf den Mail-Server zu verhindern, nutzt Transport Layer Security eine zertifikatbasierte Authentifizierung.



Zur Authentifizierung der Daten unterstützt das TSL-Protokoll den Hashed Message Authentication Code (HMAC) und erzeugt das Schlüsselmaterial. Das TLS-Protokoll nutzt den TLS-Record-Layer, der für die Verschlüsselung der Anwendungsdaten sorgt, mit den darauf aufsetzenden Protokollen Alert, Change Cipher Spec., Handshake und Application Data. Über das TLS-Handshake-Protokoll einigen sich die Peers darauf mit welchen Algorithmen verschlüsselt und authentifiziert werden soll. TLS arbeitet mit vier verschiedenen Schlüsseln: jeweils einen zum Ver- und Entschlüsseln und je einen zur Authentifizierung der ankommenden und abgehenden Datenpakete.
Das TLS-Protokoll wird nicht nur im Web-Umfeld mit HTTP eingesetzt, sondern auch in Verbindung mit anderen Anwendungsprotokollen wie für den Abruf von E-Mails über das

Post Office Protocol (POP) oder über das Internet Message Access Protocol (IMAP). In WLANs wird TLS in Verbindung mit dem Extensible Authentication Protocol (EAP), EAP-TLS, für den sicheren Austausch der Authentifizierungsdaten eingesetzt. Da das EAP-TLS von beiden Kommunikationspartnern Zertifikate verlangt, ist man auf EAP-TTLS, dem Tunneled TLS, übergegangen.

Token Ring

Das IBM Token-Ring-Netzwerk entspricht den internationalen Standards für Token-Ring-LANs von der European Computer Manufacturers Association (ECMA) und IEEE (IEEE 802.2 und 802.5). Der Token Ring ist ein offenes lokales Netz (LAN), das den Anschluss heterogener Geräte erlaubt. Das IBM Token-Ring-Netzwerk ist ein Starshaped Ring, also ein Ringnetzwerk, das aus Sicherheits-, Fehlertoleranz- und Redundanzgründen aus einer Reihe ringförmig gekoppelter Sterne gebildet wird, sich logisch aber wie ein Ring verhält. Bedingt durch die rasante Entwicklung des Ethernet wurde Token Ring nicht weiterentwickelt. *Siehe auch*: *LAN (S. G46)*.

Topologie

Die Topologie eines Rechnernetzes beschreibt die spezifische Anordnung der Geräte und Leitungen, die das Rechnernetz bilden. Es wird zwischen physikalischer und logischer Topologie unterschieden. Die physikalische Topologie beschreibt den Aufbau der Netzverkabelung; die logische Topologie den Datenfluss zwischen den Endgeräten. Topologien werden grafisch mit Knoten und Kanten dargestellt.

Torustopologie

Bei einer Torustopologie wird sich bei dem geometrischen Modell eines Torus bedient und die Eigenschaften auf das Netzwerk übertragen.
Eine solche Topologie kann man sich als ein vermaschtes Netz mit in einem Gitterarray der Dimension $N = 2,3,\ldots$ vorstellen, bei dem einzelne Elemente mit ihren nähesten Nachbarn bezüglich jeder Dimension und zusätzlich die Knoten an gegenüberliegenden Ecken einer Dimension verbunden sind. In diesem Gitter hat jeder Knoten genau $2 \cdot N$ Verbindungen. Die Topologie trägt ihren Namen deswegen, da das so entstandene Gitter topologisch homogen mit dem eines N-dimensionalen Torus ist.
Eine eindimensionale Toroustopologie ist äquivalent zur Ringtopologie.
Vor- und Nachteile:

+ Höhere Geschwindligkeit bei geringerer Latenz.
  Dies ist der Verbindung der gegenüberliegenden Kanten und somit hoher Verbindungsanzahl pro Knoten geschuldet. Damit gibt es mehr Verbindungsoptionen zwischen zwei Knoten, was sich meistens wiederum positiv auf die Geschwindigkeit auswirkt.
+ Niedriger Energieverbrauch
  Da Daten im Allgemeinen weniger Zwischenschritte gehen, sinkt auch der Energieverbrauch.
− Hoher Kabelaufwand und hohe Schaltungskomplexität
  Vor allem unterschiedliche Verbindungslängen könnnen hierbei für Probleme sorgen (RC delay). Zudem wird die Designphase bei Entwürfen schwerer, da mehr Verbindungen notwendig sind.
− Hohe Kosten

Siehe auch: *Topologie (S. G82)*, *Ringtopologie (S. G64) und Vermaschtes Netz (S. G87)*.

Transportschicht (transport layer) (OSI)

Die Transportschicht ist die Schicht, in der eine direkte logische Ende-zu-Ende-Kommunikation zwischen zwei Teilnehmern realisiert wird. Der transparente Datentransport über die Netzwerkverbindungen kann durch Mechanismen für die Flusskontrolle gesteuert

werden.

Die Aufgabe der Transportschicht, die vom Transportprotokoll mit ihren Transportklassen ausgeführt wird, besteht darin, zwei miteinander kommunizierenden Anwendungsprozessen eine transparente, lückenlose und gesicherte Ende-zu-Ende-Datenübertragung bereitzustellen, ohne Rücksicht auf die in den Schichten 1 bis 3 verwendeten Medien. Die folgenden Elemente bzw. Eigenschaften sind charakteristisch: Transparenz, Fehlerfreiheit, Netzwerkunabhängigkeit, Ende-zu-Ende-Transportservice, Kostenoptimierung und Transportadressierung. Es werden zwei Arten von Transportdiensten unterschieden: verbindungsorientiert und verbindungslos. Zu den verbindungslosen Transportdiensten zählt z.B. der Datagrammdienst.

Folgende Dienste werden den höheren Schichten angeboten: Die Transportschicht unterstützt die verbindungsorientierte oder verbindungslose Kommunikation. In der Transportschicht werden normale und Vorrang-Dateneinheiten transferiert.

Erlaubt ein Netzwerk nur den Transport von kleinen Dateneinheiten, so kann die Schicht 4 die von der Schicht 5, der Kommunikationssteuerungsschicht, ankommenden großen Datenpakete mittels Segmentierung in mehrere kleine Dateneinheiten aufteilen, diese über das Netzwerk senden und in der Transportschicht des Zielsystems wieder zur ursprünglichen Form zusammensetzen, dem Reassembling.

Sollen sehr kleine Dateneinheiten der Schicht 5 durch das Netzwerk transportiert werden und wäre diese Übertragung ineffizient, kann die Station durch Verketten, Concatenation, eine größere Einheit bilden, die sie durch das Netzwerk transportiert. Die Gegenstelle trennt diese Dateneinheiten wieder in ihre ursprünglichen kleinen Einheiten.

Die Transportschicht unterstützt die Auswahl von Dienstgüteparametern. Dafür wurden bei den OSI-Transportprotokollen 5 Serviceklassen, die Transportklassen 0 bis 4 mit unterschiedlichen Leistungsmerkmalen festgelegt. Die Transportschicht ist in der Lage, für eine bestimmte Anwendungsbeziehung zwischen mehreren Wegen zu wählen bzw. Dateneinheiten aus Gründen des Datendurchsatzes über mehrere Wege an ein Ziel zu übersenden. Bei verbindungsorientierter Kommunikation stellt die Transportschicht durch die Vergabe von Folgenummern die richtige Reihenfolge der empfangenen Dateneinheiten wieder her. *Siehe auch*: *OSI (S. G41)*.

Übertragungsmedium

Übertragungsmedien sind Einrichtungen zur Übermittlung von Informationen. Als Medien können feste, flüssige und gasförmige Materialien verwendet werden. Die Informationen werden mit einem Informationsträger wie beispielsweise Druckwellen, Spannungs- oder Lichtimpulse oder elektromagnetische Wellen übertragen.

Überlastkontrolle

Wenn in einem Netzwerk die Anzahl der Datenpakete einen bestimmten Wert übersteigt, dann wird das Netz überlastet und seine Leistung sinkt. Dies ist dann der Fall, wenn der Puffer in der empfangenden Datenstation überläuft und der Speicherplatz für die Zwischenspeicherung knapp wird.

Durch die Überlast benötigt das Netzwerk mehr Zeit um die Überlastsituationen, Staus und Konflikte zu behandeln und zu entfernen. Überlastkontrollen, Congestion Control, vermeiden solche Engpässe.

Überlastkontrollen sollten verteilt implementiert oder auch in Subnetze verlegt werden. Solche Kontrollmechanismen sorgen dafür, dass an den Übergangsstellen zwischen den Subnetzen genügend Speicherkapazität für die Datenpufferung vorhanden ist.

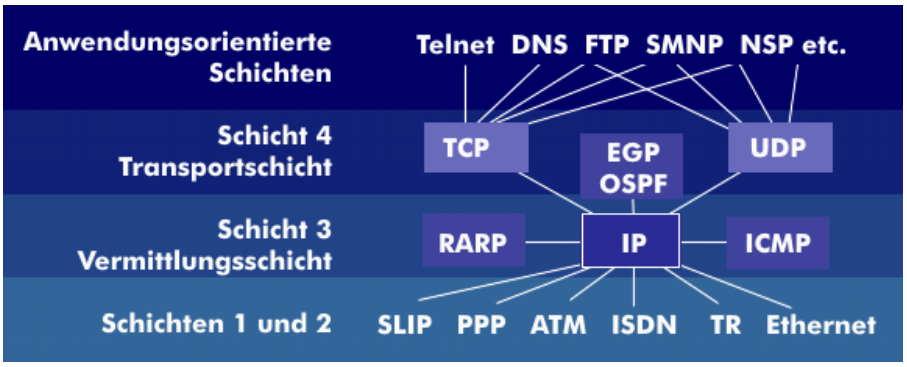
Eine Möglichkeit der Überlastkontrolle ist die Reservierung von Pufferspeichern, andere Verfahren arbeiten mit Flusskontrolle und steuern über eine Fenstertechnik den Datenstrom und wieder andere Verfahren basieren auf der Abweisung von Datenpaketen, sobald der Pufferspeicher voll ist. Letztgenanntes Verfahren zur Überlastkontrolle nennt man Retransmission-Verfahren. Es hat den Nachteil, dass das Netz zusätzlich durch die

abgewiesenen Datenpakete belastet wird.

UDP (user datagram protocol)

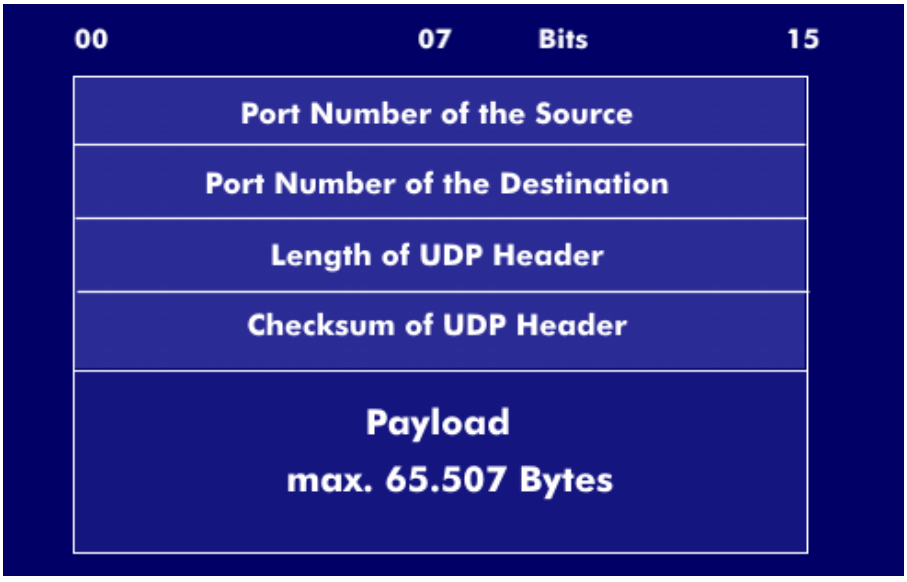
Das User-Datagram-Protokoll (UDP) ist ein verbindungsloses Transportprotokoll für den Datenaustausch zwischen Rechnern. Das UDP-Protokoll wurde definiert damit Anwendungsprozesse direkt Datagramme versenden können und damit die Anforderungen an transaktionsorientierten Verkehrs erfüllen. Das UDP-Protokoll baut direkt auf dem darunter liegenden IP-Protokoll auf und zeichnet sich durch einen geringen Overhead und kurze Latenzzeiten aus.

Das UDP-Protokoll hat einen minimalen Protokollmechanismus und garantiert weder die Ablieferung eines Datagramms beim Zielpartner, noch sind Vorkehrungen gegen eine Duplizierung oder eine Reihenfolgevertauschung getroffen. Der Funktionsumfang des UDP-Protokolls ist daher gegenüber dem TCP-Protokoll eingeschränkt. Er beschränkt sich auf den Transportdienst, das Multiplexen von Verbindungen und die Fehlerbehandlung.



Bei dem Transportdienst ist die korrekte Datenübermittlung an den Empfänger nicht sichergestellt, da er ohne Bestätigungsmechanismus arbeitet. Verlorene Datenpakete können daher nicht erneut gesendet werden. Im Gegensatz zum TCP-Protokoll, das verbindungsorientiert arbeitet, baut das verbindungslos arbeitende UDP-Protokoll keine aktive Verbindung zwischen den Stationen auf, sondern schickt die einzelnen Datenpakete völlig unabhängig voneinander ins Netz.

Durch Verwendung von Ports können in einem Rechner mehrere Kommunikationsziele ausgewählt werden. Die Ports werden durch Portnummern identifiziert, wobei diese Nummern teilweise für bestimmte Prozesse reserviert sind.



Da das UDP-Protokoll nur unter bestimmten Implementierungen eine minimale Fehlerbehandlung hat, obliegt es den höheren Schichten Fehler zu erkennen und eine

Korrektur vorzunehmen. Der UDP-Header umfasst lediglich 8 Bytes, im Gegensatz zum TCP-Header, der 40 Bytes umfasst. Zu den Datenfeldern des UDP-Header gehören die Quelladresse und Zieladresse, das Längenfeld, in dem die UDP-Headerlänge eingetragen ist, und das Prüfsummenfeld.
Das UDP-Protokoll ist u.a. im TCP/IP-Standarddokument RFC 768 beschrieben. Es wird überall dort eingesetzt wo eine geringe Verzögerungszeit und ein deterministisches Verhalten erforderlich sind.

UML Statechart

Stop-and-Wait

- Beschreibung durch UML-Statecharts
  - ein Statechart befindet sich immer in einem Zustand, der schwarze Punkt kennzeichnet den initialen Zustand
  - ein Zustandsübergang findet statt, wenn das Ereignis ausgelöst wurde und die Bedingung erfüllt ist
  - wenn ein Zustandsübergang stattfindet, wird die Aktion durchgeführt
  - zur Steigerung der Flexibilität gibt es auch Variablen

Rechnerkommunikation, Transportschicht 19

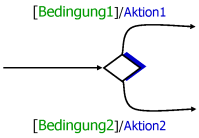
Stop-and-Wait

- Bemerkungen zu den Statecharts
  - Statecharts stellen eine Variante von endlichen Automaten dar
  - Ereignisse, Bedingungen und Aktionen werden oft durch Pseudocode beschrieben, man erhält eine halbformale Beschreibung
  - das Verhalten von Protokollen wird oft durch solche oder ähnliche Automaten dargestellt
  - es gibt auch Werkzeuge, die dies unterstützen: Protokolle können so spezifiziert werden und daraus der Code generiert werden sowie Analysen, Simulationen und Tests durchgeführt werden
  - man kann daraus gut Implementierungen ableiten: eine große Fallunterscheidung für die möglichen Ereignisse in den verschiedenen Zuständen
  - hier werden Statecharts einfach zur genauen Darstellung des Stop-and-Wait-Protokolls und später von weiteren Protokollen verwendet
  - die Darstellung ist durch Kurose/Ross motiviert, unterscheidet sich aber von den Automaten in dem Buch

Rechnerkommunikation, Transportschicht 20

Go-Back-N

- Beschreibung durch Statecharts
  - neues Element: Verzweigung

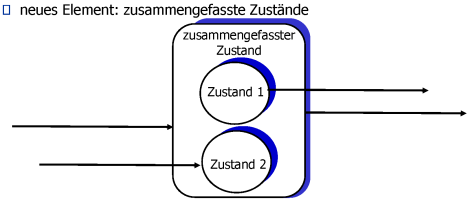


  - Zustand, in dem keine Zeit verbracht wird („Pseudozustand")
  - abgehende Zustandsübergänge werden mittels Bedingungen gewählt, auslösende Ereignisse sind hier nicht möglich

Rechnerkommunikation, Transportschicht 67

TCP: Fehlerkontrolle

- Beschreibung durch Statecharts
  - neues Element: zusammengefasste Zustände



  - grafische Vereinfachung: Zustandsübergänge, die an einem zusammengefassten Zustand beginnen (enden), gelten für jeden inneren Zustand
  - Zustandsübergänge können auch direkt an inneren Zuständen beginnen (enden)

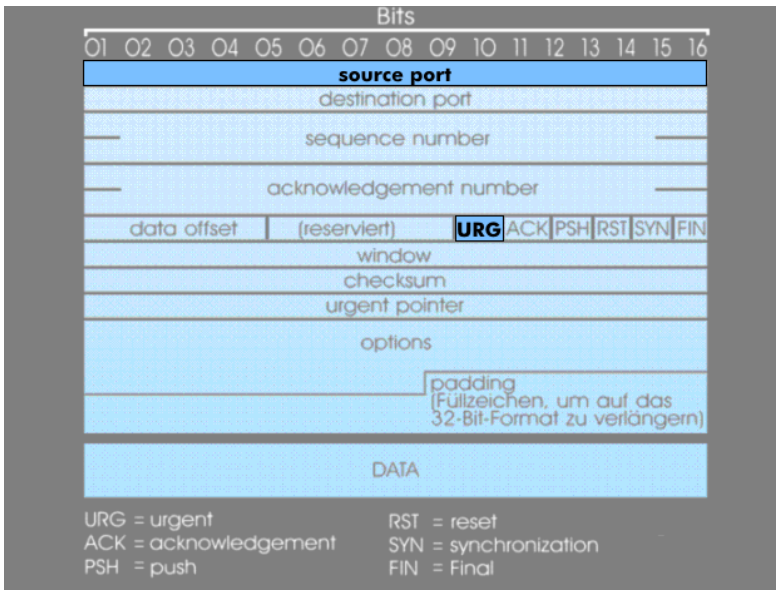
Rechnerkommunikation, Transportschicht 120

Unicast

Unicast ist in der Telekommunikation die Adressierung einer Nachricht an einen einzigen Empfänger. Dabei handelt es sich um Adressierungsschemata der Vermittlungsschicht im OSI-Modell. Der Unicast entspricht einer Punkt-zu-Punkt-Verbindung. Ein typisches Beispiel für Unicast ist das Telefonieren mit **einem** anderern Teilnehmer. *Siehe auch*: *Kommunikationsart (S. G43)*.

URG (urgent flag)

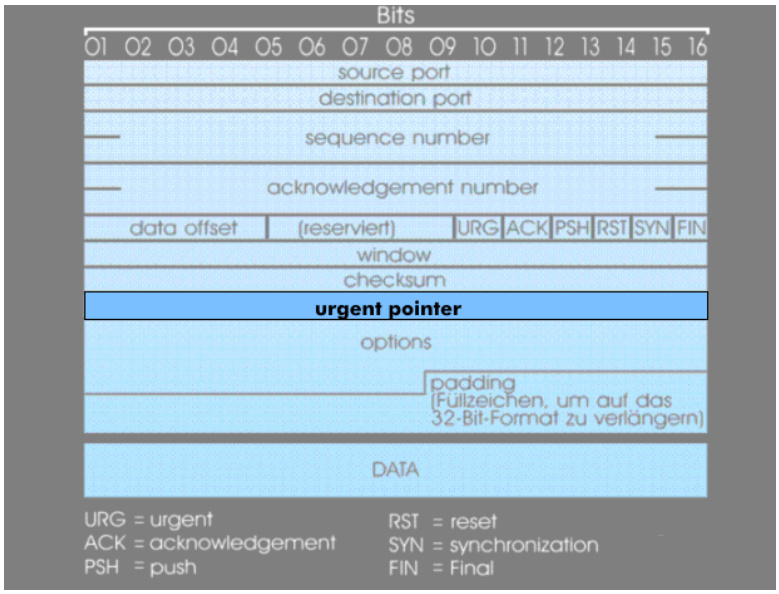
Die Urgent-Flag (URG) ist ein Bit-Indikator im Control-Flag-Feld des TCP-Headers. Ist ein Urgent-Flag gesetzt, müssen die gekennzeichneten Daten mit dem Urgent Data Signalling Service weiterverarbeitet werden.



Das Transmission Control Protocol (TCP) unterstützt Urgent-Daten von jeder Länge und informiert das Anwendungsprogramm darüber, wie viel Urgent-Daten sich noch im Empfangspuffer befinden.

Urgent Pointer

Urgent Pointer ist ein 16 Bit langes Datenfeld im TCP-Header. Durch das Urgent-Flag wird signalisiert, dass im TCP-Header Dringlichkeitsdaten sind.



Der Urgent-Pointer auf das Ende der als dringlich gekennzeichneten Daten. Der Wert des Pointers ist ein positiver Offset der Sequenznummer. Dringende Daten werden immer als erste Informationen im TCP-Header übertragen.

URI (Uniform Resource Identifier)

Der Uniform Resource Identifier (URI) ist eine Kennzeichnung mit der alle im globalen Web vorhandenen Ressourcen und Objekte eindeutig kennzeichnet werden, einschließlich des benutzten Protokolls und Dienstes. So ist eine URI für eine E-Mail - mailto:florian.ff.frank@fau.de - anders aufgebaut als die für eine Grafik auf einer Webseite - https://wwwcip.informatik.uni-erlangen.de/yq53ykyr/img/fau-logo-tech.png - oder als die für einen Dateitransfer - ftp:/dokumente/smtp.txt. Bei den im Uniform Resource Identifier gekennzeichneten Objekten kann sich um eine Textdatei oder eine Grafik handeln, um eine Excel-Datei, eine Audio- oder Videosequenz. Jedes einzelne Objekt hat einen eigenen,

einmalig vorkommenden Uniform Resource Identifier. Die URI ist gekennzeichnet durch das Protokoll - http, mailto, ftp, gopher, news - durch den Dienst wie World Wide Web (WWW) und durch eine gültige Adresse. Ein Uniform Resource Identifier oder auch Universal Resource Identifier kann ein Uniform Resource Locator (URL) oder ein Uniform Resource Name (URN) sein. Er kennzeichnet einen bestimmten Inhalt im Web: ein Ton-, Video-, Bild- oder Textdokument. Die Bezeichnungen URI und URL werden synonym verwendet.

Verfügbarkeit

Die Verfügbarkeit ist ein Maß für die Wahrscheinlichkeit, dass ein System zu einem bestimmten Zeitpunkt eine geforderte Leistung erbringt. In der Informationstechnik und der Datenübertragung ist die Verfügbarkeit von Daten ist ein wirtschaftlicher Aspekt.

Die Verfügbarkeit kann sich gleichermaßen auf Systeme und deren Funktionalität und auf Daten oder Informationen beziehen. Zwischen der Verfügbarkeit und der Leistungssteigerung besteht eine Wechselwirkung. Da bei Ausfall eines Systems ein hoher wirtschaftlicher Schaden entstehen kann, sollten Datenverarbeitungssysteme und Kommunikationswege möglichst redundant ausgeführt sein.

Die Verfügbarkeit wird bestimmt durch die Mean Time Between Failures (MTBF) und die Mean Time To Repair (MTTR). Sie ergibt sich aus dem Quotienten von MTBF/(MTBF + MTTR) oder aus dem Verhältnis der Uptime zur Summe aus Downtime und Uptime. Sie wird in mehrere Klassen zwischen einfacher Verfügbarkeit, über Hochverfügbarkeit bis hin zur Non-Stop-Verfügbarkeit eingeteilt, wobei die einfache Verfügbarkeit bei 99,5% liegt, die Hochverfügbarkeit bei 99,9 % und höher und die Non-Stopp-Verfügbarkeit bei 100%. Eine Hochverfügbarkeit von 99,95% entspricht einer Ausfallzeit von 4,38 Stunden pro Jahr.

Neben dieser Klassifizierung gibt es noch die Bewertung durch den Availability Level (AL). Dieser Kennwert beschreibt die Einschränkungen bei Ausfall, Abbruch, Unterbrechnung von Transaktionen und anderen Beeinträchtigungen. Es gibt vier AL-Level 1 bis 4 in denen die Beeinträchtigungen bewertet sind.

Die Verfügbarkeit umfasst Hardware und Software, Carrier-Netze und den Datentransport, das Betriebssystem und die Datenspeicherung, die Stromversorgung und die Netzwerkkomponenten, die Sicherheit vor unberechtigten Zugriffen und vor Sabotage usw., daher steht bei der Verfügbarkeit an erster Stelle die Redundanz, die optimal organisiert sein muss. Die Verfügbarkeit von Carrier-Netzen wird durch ein Prädikat ausgezeichnet, den Carrier Grade. Dieses Prädikat ist an mehrere spezifizierte Regeln geknüpft, so an die Gesamtverfügbarkeit, die 99,999% betragen muss, was einer Ausfallzeit von wenigen Minuten pro Jahr entspricht, an die Bereitstellung neuer Dienste innerhalb vorgegebener Zeiten, an Dienstgütevereinbarungen (SLA), an die mechanische Robustheit u.v.a.m.

Vermaschtes Netz

In einem vermaschten Netz ist jedes Endgerät mit einem oder mehreren anderen Endgeräten verbunden. Wenn jeder Teilnehmer mit jedem anderen Teilnehmer verbunden ist, spricht man von einem vollständig vermaschten Netz. Bei Ausfall eines Endgerätes oder einer Leitung ist es im Regelfall möglich, durch Umleiten (Routing) der Daten weiter zu kommunizieren.

Vor- und Nachteile:

- \+ Sicherste Variante eines Rechnernetzes
- \+ Ausfälle von Endgeräten können abgefangen werden
- \+ Hohe Bisektionsweite und niedrigen Durchmesser
- \+ je „voller“ die Netze vermascht sind desto weniger Routing wird benötigt
- — Hoher Kabelaufwand
- — Hoher Energieverbrauch
- — Komplexes Routing notwendig, da die Netze nicht regulär oder symmetrisch sind

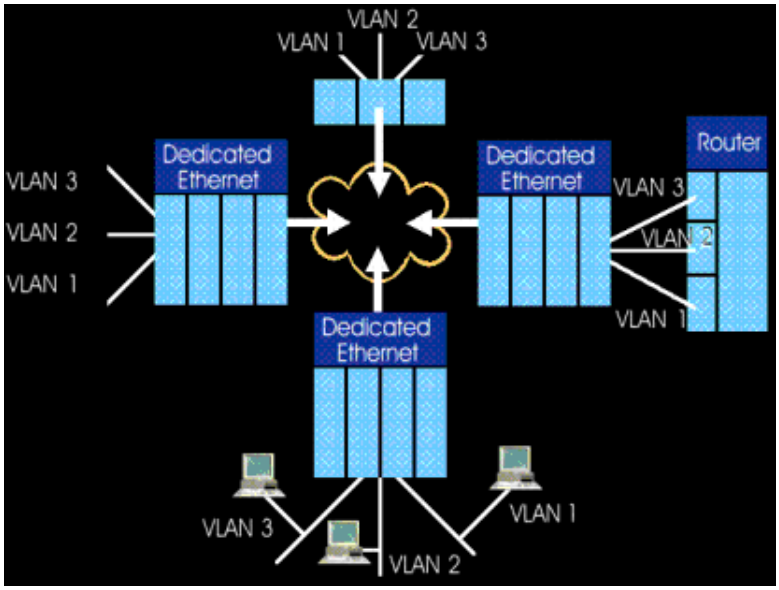
Siehe auch: *Topologie (S. G82)*.

Vermittlungsart

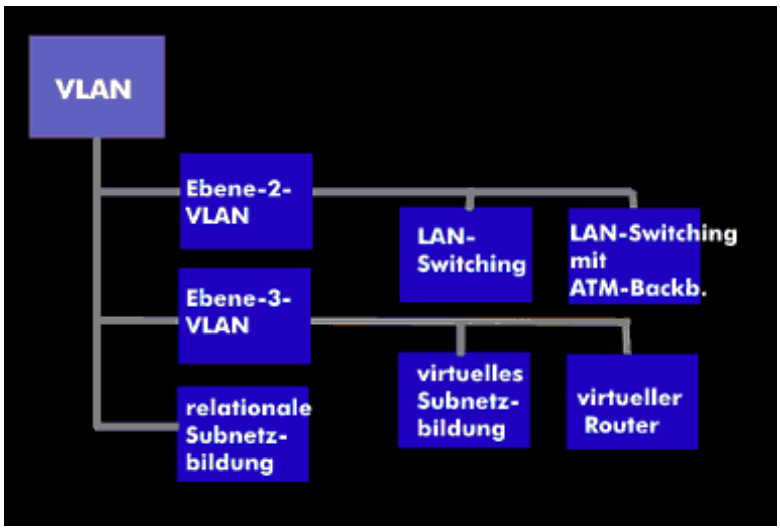
Unter der Vermittlung versteht man ganz allgemein die Art und Weise, mit der ein Übertragungspfad zwischen Sender und Empfänger vermittelt wird.

VLAN (virtual LAN)

Virtuelle Netze oder virtuelle LANs (VLAN) sind ein technologisches Konzept zur Implementierung logischer Workgroups innerhalb eines Netzes. Die Realisierung eines solchen virtuellen Netzes erfolgt mittels LAN-Switching oder mittels virtuellem Routing auf der Sicherungsschicht oder auf der Vermittlungsschicht. Virtuelle Netze werden durch eine Menge von Switching Hubs aufgebaut, die ihrerseits durch einen Backbone miteinander verbunden sind. Die IEEE-Arbeitsgruppe 802.1Q hat sich dieses Themas angenommen.
Im Backbone von virtuellen Netzen werden klassische Netzwerkprotokolle eingesetzt. Daher bestehen konstruktiv keine Unterschiede zwischen dem lokalen Bereich und einem größeren Unternehmensnetz, das z.B. aus mehreren lokalen Bereichen besteht. Virtuelle Netze benutzen sogenannte »Membership Rules« zur Definition der Zugehörigkeit von Stationen zu logischen Workgroups und implementieren Schaltnetzwerke zur Verbindung der Mitglieder der logischen Workgroups. Dieser Ansatz erlaubt die Zugehörigkeit zu einer logischen Workgroup, unabhängig vom physischen Ort des Arbeitsplatzrechners.



Virtuelle Netze bilden eine nach bestimmten Kriterien frei definierbare Broadcast-Domäne auf Basis physikalischer LAN-Segmente. Es besteht also eine Verbindung auf dem Medienzugangsverfahren (MAC) mit oder ohne zusätzliche Auswertung der Information von der Vermittlungsschicht.
VLANs vereinigen die Vorteile, die man normalerweise mit durch Brücken verbundene Netze assoziiert, wie leichtes Hinzufügen bzw. Wegnehmen oder Ändern einer Station, zusammen mit dem Vorzug der logischen Systemtrennung und Strukturierung mittels Routern, ohne jedoch die Durchsatzprobleme von Brücken und die schwierige Konfiguration großer Netze mit Routern hinnehmen zu müssen. Virtuelle Netze werden mit Switches des im Dedicated Ethernet vorkommenden Typs und mit Hochgeschwindigkeits-Backbone-Technologie implementiert.



Eine Methode zur Definition eines virtuellen Netzes ist die Zuordnung von Ports. Alle Stationen, die an einem bestimmten Port eines Ethernet-Switching-Systems liegen, werden als Teil des virtuellen Netzes aufgefasst, und eine Menge von Ethernet Switch Ports im physikalischen Gesamtnetz bilden das gesamte virtuelle Netz. Das kann nur unter der Voraussetzung transparent funktionieren, dass die Ethernet-Switches mit einer skalierbaren Hochgeschwindigkeitstechnologie untereinander verbunden sind. Ein virtuelles Netz hat, wenn die richtige Backbone-Technologie verwendet wird, kaum Ausdehnungsbeschränkungen.

Der gesamte Verkehr im virtuellen Netz wird mit Packet Switching realisiert, d.h. die Adressierung erfolgt im flachen Adressraum der Schicht-2-Adressen (Data-Link-Adressen). Allerdings ist der Adressraum auf 4.096 VLAN-IDs begrenzt.

Die Switching-Systeme besitzen einen Lern-Algorithmus, der dem traditioneller Brücken ähnlich ist. Das bedeutet, dass eine Station den physischen Ort einfach wechseln kann und dennoch Mitglied des virtuellen Netzes bleibt, ohne dass eine Rekonfiguration in der Endstation erforderlich wäre. So können ortsunabhängige Workgroups geschaffen werden. Außerdem hat die Orientierung an den Data-Link-Adressen den Vorteil der Protokolltransparenz: im Gegensatz zu routerbasierten Techniken können auch innerhalb einer Arbeitsgruppe unterschiedliche Protokolle der Schichten 3 bis 7 benutzt werden. Durch diese Konstruktion verbleibt der Verkehr innerhalb einer logischen Workgroup bzw. innerhalb eines virtuellen Netzes. Broadcasts auf einem virtuellen Netz werden keinesfalls auf ein anderes virtuelles Netz weitergeleitet.

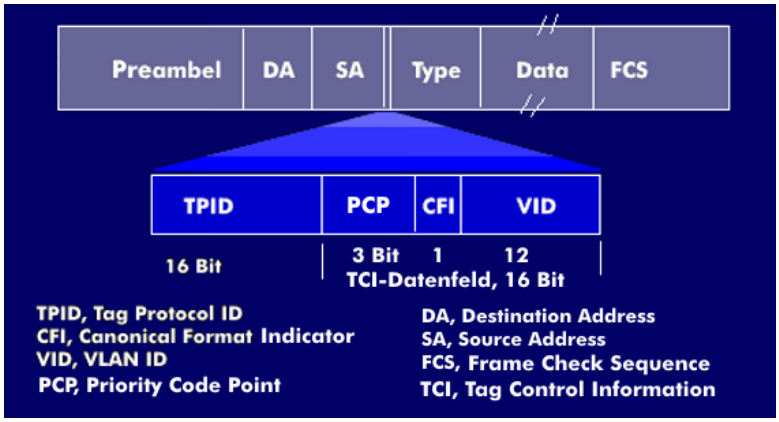
Die virtuellen Netze erscheinen als vollständig unabhängige Switching-Fabrics. Von daher schirmen die virtuellen Netze ihren eigenen Verkehr ab, was wiederum ein Routing zwischen den virtuellen Netzen wünschenswert macht, jedoch ohne die Kopplung von logischer Workgroup und physischen Orten hinnehmen zu müssen. Dies ist aber bei den virtuellen Netzen im Grunde genommen durch konventionelle Router zu erreichen. Durch die Ortstransparenz des virtuellen Netzes kann man an eine Stelle des physischen Netzes einen Router stellen und diesen mit so vielen physischen Ports verbinden, wie es virtuelle Netze gibt. Danach muss man nur jeden dieser Ports einem virtuellen Netz zuordnen, und der Router ist für alle Teilnehmer dieses virtuellen Netzes erreichbar.

Eine Erweiterung der VLAN-Technik bietet das Infrastrukturprotokoll Virtual Extensible LANs (VXLAN), das 24 Bit für den Adressbereich bereitstellt.

VLAN-Tagging

VLAN-Tagging ist ein Verfahren um VLAN-Informationen innerhalb der Ethernet-Frames auf MAC-Ebene zu kennzeichnen. Das VLAN-Tagging unterstützt bestehende 802.x MAC-Services, bietet Priorisierungsmechanismen, beschränkt sich auf portbasierte VLANs und unterstützt jedes virtuelle Netz mit einem Spanning-Tree.

Bei Einsatz des Tagging bleibt die Kompatibilität und Interoperabilität zu existierenden lokalen Netzen (LAN) erhalten. Diese Forderung ist nicht unproblematisch, da beim Tagging der Header für die VLAN-Information erweitert wird, gleichzeitig aber das MAC-Frame erhalten bleibt. Mit der Erweiterung des Ethernet-Frames befasst sich die Arbeitsgruppe IEEE 802.3as: Frame Extension, mit dem VLAN-Tagging die Arbeitsgruppen 802.1Q, 802.1ad und 802.1ah.
Um ein funktionierendes Tagging-Verfahren zu gewährleisten, muss ein eingehendes Frame analysiert und mit den entsprechenden VLAN-Informationen versehen an die Transitsysteme weitergegeben werden. Diese Systeme müssen wissen, an welcher Position des MAC-Frames sich der Tag befindet, und diesen auswerten können. Da die traditionellen MAC-Frames einen unterschiedlichen Aufbau haben, ist die Kenntnis des Tag-Platzes entscheidend für die Auswertung.



Beim VLAN-Tagging wird ein zusätzliches Frame in das Ethernet-Frame eingefügt, und zwar zwischen das Datenfeld für die Quelladresse (SA) und das Type-Feld. Das VLAN-Tag-Feld besteht aus dem 16 Bit langen Datenfeld, dem Tag Protocol Identifier (TPID), und dem ebenfalls 16 Bit langen Datenfeld Tag Control Information (TCI), das wiederum aus drei Datenfeldern besteht: dem Priority Code Point (PCP) in dem die Priorität des Frames festgelegt wird, dem Canonical Format Indicator (CFI) für Portsteuerung von Ethernet-Switches und dem 12 Bit langen Datenfeld für den VLAN Identifier (VID).

VPN (virtual private network)

Die Bezeichnung Virtual Private Network (VPN) wird in mehreren Bedeutungen verwendet. Ganz allgemein spricht man von einem virtuellen privaten Netz wenn innerhalb eines öffentlichen Netzes kundenspezifische logische Teilnetze gebildet werden. Das können Netze der Sprachkommunikation sein, das Internet, Frame Relay, ATM oder ISDN, aber auch Anschlussnetze mit DSL-Techniken. Die heute gebräuchliche Interpretation für Virtual Private Networks (VPN) sind die IP-VPNs, bei denen die Teilnehmer über IP-Tunnel verbunden sind.
Generell handelt es sich bei einem Virtual Private Network (VPN) um ein geschlossenes, logisches Netz, das auf unterschiedlichen Schichten des OSI-Referenzmodells aufsetzt und für eine bestimmte Benutzergruppe etabliert wird. VPN-Netze setzen in der Regel auf den Schichten 2 oder 3 des OSI-Referenzmodells auf und verwenden Tunneling-Mechanismen für den IP-Verkehr.
Ein VPN nutzt immer die öffentlich zugänglichen Übertragungsnetzwerke, bei denen die Verbindungen durch einen öffentlichen Carrier bereitgestellt werden. Der Anwender bildet sich über diese Übertragungswege praktisch sein privates Netz. Er verfügt über Sicherheitsmechanismen, wie die Identifikation und die Authentifikation der Netzteilnehmer, sodass sich Unbefugte keinen Zugang zum virtuellen privaten Netz verschaffen können. Die Dienstleistungen werden über ein öffentliches Netz erbracht, wodurch der Anwender auch die Ressourcen des Carriers nutzen kann.

Ein VPN hat gegenüber echten privaten Netzen, wie einem Corporate Network, die Vorteile der wesentlich höheren Flexibilität und Leistungsfähigkeit. So kann der Anwender seinen Bandbreitenbedarf definieren oder auch den Bandbreitenbedarf dynamisch konfigurieren (Bandwidth on Demand). Netzwerkrelevante Eigenschaften, wie eigene Adressierung, Netzwerkmanagement, dynamische Konfiguration und Routenoptimierung, können je nach Carrier sowohl vom diesem als auch vom Anwender durchgeführt werden. Sind in einem VPN Mobilgeräte eingebunden, so spricht man auch von mobilen virtuellen privaten Netzen, Mobile VPNs.

Kommunikationsgrundlage der IP-basierten VPNs ist das Point to Point Protocol (PPP) mit IP-Tunneling. Die Anbindung von Außenstellen kann dabei entweder über Wählzugänge im analogen Telefonnetz oder im ISDN-Netz erfolgen, aber auch über das GSM-Netz und bei großem Datenaufkommen auch über Standleitungen. Man spricht in diesem Zusammenhang von Remote-Access-VPNs, über die Heimarbeitsplätze und mobile Datenendgeräte auf das Unternehmensnetz zugreifen können. Die Clients eines solchen virtuellen Netzes bauen über das PPP-Protokoll eine Verbindung zum Access-Router auf und werden mittels des Challenge Handshake Authentication Protocols (CHAP) authentifiziert. Die Übertragung erfolgt dann über den IP-Tunnel.

Die IETF beschäftigt sich mit dem Thema Ethernet und VPNs. Sie gliedert ihre Aktivitäten in Bezug auf den IP-Traffic (Schicht 3), Ethernet (Schicht 2) und die SDH-Hierarchie sowie den asynchronen Übertragungsmodus (ATM) auf Schicht 1.

Danach ist ein Layer-3-VPN ein vom Netzbetreiber eingerichtetes VPN, das mit Routing in der Schicht 3 arbeitet. Ethernet als Transporttechnik wurde nicht berücksichtigt. Auf der Sicherungsschicht werden Ethernet-Dienste in IP/MPLS-Netzen realisiert. Dabei handelt es sich um Virtual Private LAN Services (VPLS), Virtual Private Wire Services (VPWS) und IP-only-VPN.

Bezüglich der Bitübertragungsschicht geht es der IETF-Arbeitsgruppe um die Frage wie VPNs direkt auf Transportnetzen, basierend auf Synchronous Digital Hierarchy (SDH), Optical Transport Hierarchy (OTH) und Ethernet ohne IP/ MPLS, realisiert werden können.

Weitverkehrnetz (WAN)

Weitverkehrsnetze, Wide Area Networks (WAN), sind für die Sprach- oder Datenübertragung über weite Strecken konzipiert. Diese Netze sind flächendeckend aufgebaut und können uneingeschränkt für die geschäftliche und private Kommunikation genutzt werden. Die Konzeption solcher Netze wird im Wesentlichen durch das Dienstangebot geprägt. So eignet sich das klassische analoge Fernsprechnetz (POTS) ebenso wie ISDN für die Telefonie. Dagegen wurden die öffentlichen Datenpaketnetze für Datenübertragungsdienste konzipiert. Weitverkehrsnetze können eine Ausdehnung von 1.000 km oder auch 10.000 km haben.

Die für den Anwender ganz entscheidenden Übertragungsraten haben sich in den letzten Jahren rasant nach oben entwickelt. Waren vor Jahren noch Übertragungsgeschwindigkeiten von 64 kbit/s Stand der Technik, sind mit ISDN 2 Mbit/s und über E3 sogar 34 Mbit/s möglich. Durch die Nutzung von ATM können dem Anwender Datenraten von 155 Mbit/s oder auch 622 Mbit/s zur Verfügung gestellt werden, die durch einen möglichen Einsatz von 10-Gigabit-Ethernet bis in den Höchstgeschwindigkeitsbereich gesteigert werden könnten.

WANs werden in Europa meist von den Telekommunikationsverwaltungen oder Netzbetreibern betrieben, so dass insbesondere der Übergang zwischen zwei Netzwerken für den Anwender von besonderer Bedeutung ist.

wireless local acess network (WLAN)

Wireless LANs (WLAN) sind drahtlose lokale Netze (LAN), die ihre Daten mit Funk übertragen. Wenn man von WLANs spricht, meint man die von der Arbeitsgruppe IEEE 802.11 standardisierten. Die Entwicklung der WLANs begann Mitte der 90er Jahren mit dem Standard 802.11, der mit 1 Mbit/s und 2 Mbit/s noch relativ geringe Datenraten hatte. Dank verbesserter und ausgefeilter Modulationsverfahren und Codierungen konnten die

Datenraten bei 802.11b auf 11 Mbit/s gesteigert werden, bei 802.11g und 802.11a auf 54 Mbit/s und bei 802.11n auf bis zu 600 Mbit/s. Mit 802.11ac und 802.11ad sogar auf über 6 Gbit/s. *Siehe auch*: *LAN (S. G46) und IEEE 802.11 (S. G32)*.

World Wide Web Consortium (W3C)

Das W3C ist das Gremium zur Standardisierung der Techniken im World Wide Web.

WWW (world wide web)

World Wide Web (WWW oder W3), kurz Web, ist ein auf Hypertext basierendes Informationssuchsystem im Internet. Es wurde 1992 vom Kernforschungsinstitut CERN in der Schweiz entwickelt. Später wurde es auf Hypermedia erweitert und kommt damit der Arbeitsweise des menschlichen Gehirns am nächsten. Hypertextdokumente sind Textdateien, die über Suchwörter mit einem oder mehreren Textdokumenten vernetzt sind. Diese werden in Hypertext Markup Language (HTML) programmiert und mittels Hypertext Transfer Protocol (HTTP) im Internet übertragen. Diese Hypertextseiten sind die Webseiten.
World Wide Web (WWW), das im Allgemeinen mit dem Internet assoziiert und synonym benutzt wird, ist nur einer von mehreren Internet-Diensten, die auf dem IP-Protokoll aufsetzen. Vorher gab es bereits den File Transfer und das Telnet-Protokoll, E-Mail, Remote Procedure Call und einige mehr.
Mit dem Einzug der Multimediatechnik und dem damit verbundenen Verzweigungssystem Hypermedia erweitern sich die Verzweigungsmöglichkeiten über Textdokumente hinaus hin zu Bild-, Ton- und Videodateien. Hypertext-Seiten werden durch die Internetadresse (URL) eindeutig identifizierbar. Das gilt auch für alle Testdokumente ebenso wie für die multimedialen Elemente, bei denen es sich um Bild-, Video- und Audioinformationen handeln kann. Diese Dokumente werden auf Webservern abgelegt und sind untereinander durch Hyperlinks verbunden.
World Wide Web ist das derzeit flexibelste Informationssuchsystem im Internet, da es alle Dateiformate berücksichtigt und aufgrund seines Aufbaus auch Client für Gopher, Veronica, WAIS oder Archie sein kann. Mit den Web-Browsern stehen außerdem hervorragende WWW-Client-Programme mit grafischer Benutzeroberfläche für die Navigation und die Darstellung der Dokumente auf dem Bildschirm zur Verfügung.

Zeitmultiplexverfahren (TDM)

Beim Zeitmultiplex erfolgt die Datenübertragung der einzelenen Verbindungen in einem definierten Multiplexrahmen, in dem für jeden Übertragungskanal ein fester Zeitschlitz vorhanden ist. Die einzelnen Schlitze werden nacheinander abgearbeitet, die Geräte wechseln sich damit ab. Damit entsteht ein pseudoparalleles Abarbeiten der Verbindungen. *Siehe auch*: *Multiplexverfahren (S. G54)*.

Zyklische Blockprüfung (CRC)

Zyklische Blockprüfung, Cyclic Redundancy Checksum (CRC), ist ein Fehlererkennungsverfahren, bei dem auf der Basis von Binärzahlen Prüfzeichen durch die Summenbildung der Datengruppen vor ihrer Übertragung gebildet werden.
Die zyklische Blockprüfung basiert auf der Division von Polynomen. Bei diesem Verfahren werden die n Bits eines Datenblocks als Koeffizienten eines Polynoms U(x) vom Grad $n-1$ interpretiert. Das erzeugte Polynom G(x) ist abhängig vom Grad k. Ein Polynom mit $k = 16$, wie es die internationale Fernmeldeunion (ITU) vorsieht, sieht folgendermaßen aus:

$$x^{16} + x^{12} + x^5 + 1.$$

Bezeichnung	Polynom
CRC-12	$x^{12}+x^{11}+x^3+x^2+x+1$
CRC-16/IBM	$x^{16}+x^{15}+x^2+1$
CRC-16/ITU	$x^{16}+x^{12}+x^5+1$
CRC-32 (IEEE 802)	$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}$ $x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$

Vom Prinzip her werden bei der zyklischen Blockprüfung die zu überwachenden Bits nacheinander in ein rückgekoppeltes Schieberegister geschoben. Die Länge wie auch die Anzahl und Lage der Rückkoppelungsanzapfungen sind je nach Verfahren angegeben; so nennt man ein Prüfsummenverfahren mit vierstelligem Register CRC-4. Das Prüfsummenverfahren erkennt Einzelfehler zuverlässig, mehrere Fehler mit großer Wahrscheinlichkeit. Der Empfänger prüft den CRC-Wert eines jeden empfangenen Datenpaketes und entfernt die Prüffunktion vor der Freigabe des Datenpaketes an die Empfangsstation. Für jedes zu übertragende Bitmuster werden entweder 16 oder 32, manchmal auch 64 Prüfbits berechnet und hinter dem Informationsteil im FCS-Feld übertragen.
Die Code-ungebundene Fehlersicherung (CRC) erfolgt mittels eines geeigneten Generatorpolynoms, z.B. $x^{16}+x^{12}+x^5+1$, wobei die Binärzeichen des zu sichernden Datenblocks als Koeffizient des Polynoms verwendet und durch das genannte Polynom Modulo 2 dividiert werden. Der nach der Division verbleibende Rest stellt die Blockprüfzeichenfolge dar und wird mit dem zu übertragenden Datenblock ausgesendet. In der Empfangsstation werden Übertragungsfehler bei Anwendung der obigen Regel mit sehr hoher Wahrscheinlichkeit entdeckt, weil bei der Modulo-2-Division der festgelegten Folge der Binärzeichen durch das Generatorpolynom ein konstanter Rest entstehen sollte und eine Abweichung durch einen Übertragungsfehler sofort in eine andere Restklasse führt.
Die Sicherheit des Verfahrens basiert auf einer möglichst »günstigen« Restklassenzerlegung, die ihrerseits vom gewählten Generatorpolynom abhängt. Je mehr unterschiedliche Restklassen erzeugt werden, desto geringer ist die Wahrscheinlichkeit, dass eine originale Bitfolge und eine verfälschte Bitfolge nach der Operation in die gleiche Restklasse fallen. Die Anzahl der Restklassen wird allerdings auch durch die Anzahl der im Datenpaket für die Darstellung des Restes verfügbaren Bits beschränkt.

FORMELSAMMLUNG

E.1 Wichtige Formeln

Rechnungen

PAKETVERZÖGERUNGEN

1. Ausbreitungsverzögerung d_{prop}
2. Übertragungsverzögerung d_{trans}
3. Verarbeitungsverzögerung d_{proc}

⇒ FehlerPrüfung / Bestimmung AusgangsLink

4. Warteschlangenverzögerung d_{queue}

⇒ Wartezeit auf den ausgehenden Link zur Übertragung

RECHENGRÖSSEN

→ l = Länge
→ v = Ausbreitungsgeschwindigkeit
→ O = Daten
→ R = Datenrate
→ L = Paketgröße

AUSBREITUNGSVERZÖGERUNG d_{prop}

$$d_{prop} = \frac{l}{v}$$

ÜBERTRAGUNGSVERZÖGERUNG d_{trans}

$$d_{trans} = \frac{L}{R}$$

ÜBERTRAGUNGSZEIT D

$$d = \frac{l}{v} + \frac{O}{R}$$

KANALPUFFERGRÖSSE A

$$a = v \cdot R$$

Verzögerung an einem Knoten

$$d_{prop} + d_{trans} + d_{proc} + d_{queue}$$

Sendezeit

Gegeben:
→ Datei Größe $O = N \cdot L$ auf Pfad mit E sequentiellen Links
→ Bei Paketvermittlung N Paketen der Größe L
→ Ausbreitung/Warteschlangenverzögerung vernachlässigen

Sendezeit — h Bits header und d sec. Verbindungsaufbau

$$t = d + (N + E - 1) \cdot \frac{L + h}{R}$$

Sendezeit — Verbindungslos und 2h Header

$$t = d + (N + E - 1) \cdot \frac{L + 2 \cdot h}{R}$$

Sendezeit — verbindungslos, ohne segmentierung, 2h Header

$$t = \frac{(L \cdot N + 2h) \cdot E}{R}$$

Sendezeit — leitungsvermittelt,ohne segmentierung, h Header

$$t = d + \frac{N \cdot L + h}{R}$$

Latenzzeitenberechnung

d_{prop} = Ausbreitungsverzögerung und d_{trans} = Übertragungsverzögerung

Cut-Through

Vorteil: Sehr schnell
Nachteil: Fehler können auftreten
→ Zeit s um ein Paket zu verschicken:

$$s = d_{prop} + d_{trans}$$

Store-And-Forward

Vorteil: keine Fehler

Nachteil: Langsam

→ Zeit s um ein Paket zu verschicken (E = Links):

$$s = E \cdot d_{trans} + d_{prop}$$

Übertragungszeit minimieren/maximieren:

Gegeben:

→ Host A schickt Datei Größe O an Host B

→ Zwischen den Host sind E-1 unbelastete Hosts

→ Host a segmentiert in die Größe L und fügt h Bits an

→ Übertragungsrate der Links: R bps

→ Gesucht: L, damit Übertragungsverzögerung minimal

Lösung:

Übertragungszeit insgesamt:

$$U(L) = \frac{h \cdot O}{L \cdot R} + \frac{O}{R} + \frac{E \cdot L}{R} + \frac{E \cdot h}{R} - \frac{L - h}{R}$$

→ Ü insgesamt als Ü(L) betrachten und ableiten

→ Ü'(L) = 0 und nach L auflösen

→ Gesucht: Übertragungszeit maximieren

Lösung:

Ein Weg dauert wegen Store and Forward am längesten, wenn $O = L \Rightarrow d_{max} = E \cdot O/R$

HTTP-Verzögerungszeiten

Gegeben:

→ M Grafik Dateien

→ Objekte haben Größe O Bits

→ Link zw. Client und Server mit RTT und Rate R

→ ausreichendes Überlastfenster, Segmentgröße L

→ keine Sendewiederholung

Fragen:

1. Wenn eine Webseite aus genau einem Objekt besteht, dann entsteht durch nichtpersistentes und persistentes HTTP unterschiedliche Antwortzeiten.
falsch, Persistent kann sofort ACK und Request nach erhalt der Basisseite senden, nicht persistent schickt zuerst ACK und SYN, bevor er requestet

2. Betrachten Sie die Übertragung eines Objekts der Größe O von einem Server zum Browser über TCP. Wenn $O > L$, wobei L die maximale Segmentgröße ist, muss der Server mindestens einmal warten.
richtig, da statt $O = L$, $L < O$ gilt

3. Nehmen Sie an, eine Webseite bestünde aus zehn Objekten mit jeweils der Größe O

Bits. Für persistentes HTTP ist der RTT-Anteil an der Antwortzeit 20RTT.
falsch, RTT liegt unter 20 RTT

4. Nehmen Sie an, eine Webseite bestünde aus zehn Objekten mit jeweils der Größe O Bits. Für nichtpersistentes HTTP mit 5 parallelen Verbindungen ist der RTT-Anteil an der Antwortzeit 12RTT
falsch, RTT-Anteil der Antwortzeit liegt unter 12 RTT

Formeln:
1. Latenzformel:
→ nicht persistentes HTTP
→ sequentielle Verbindungen

$$delay = (M+2) \cdot (2RTT + \frac{O}{R} + P \cdot (RTT + \frac{L}{R}) - (2^P - 1) \cdot \frac{L}{R})$$

2. Latenzformel:
→ nicht persistentes HTTP
→ parallele Verbindungen
→ x parallele Verbindungen

$$delay = (M+2) \cdot \frac{O}{R} + 2(\frac{M+1}{x} + 1)RTT + P(RTT + \frac{L}{R})$$
$$-(2^P - 1)\frac{L}{R} + \frac{M+1}{x}(P'(RTT + \frac{xL}{R}) - (2^{P'} - 1)\frac{xl}{R})$$

Grundwissen

Größenordnungen:

Name	Byte	Name	Bit
Kilobyte	10^3 Byte	Kibibyte	2^{10}
Megabyte	10^6 Byte	Mebibyte	2^{20}
Gigabyte	10^9 Byte	Gibibyte	2^{30}
Terabyte	10^{12} Byte	Tebibyte	2^{40}
Petabyte	10^{15} Byte	Pebibyte	2^{50}
Exabyte	10^{18} Byte	Exbibyte	2^{60}

Prozentrechnung:
Prozentuale Änderung:
→ Gegeben: Ausgangswert x , Neuer Wert y
→ Frage: Um wie viel Prozent hat sich x verändert?
→ Rechnung: Veränderung = (x/100)*|x-y|

E.2 Wichtige Größen, Konstanten und Tabellen

Wichtige Konstanten

Name	Wert
Elementarladung e	$e = 1.602176 \times 10^{-19}\,\mathrm{C}$
magnetische Feldkonstante μ_0	$\mu_0 = 4\pi \times 10^{-7}\,\mathrm{V\,s\,A^{-1}\,m^{-1}}$
elektrische Feldkonstante ε_0	$\varepsilon_0 = \frac{1}{\mu_0 c^2} = 8.854 \times 10^{-12}\,\mathrm{A\,s\,V^{-1}\,m^{-1}}$
Lichtgeschwindigkeit im Vakuum	$c = 2.99792 \times 10^{8}\,\mathrm{m\,s^{-1}}$

SI-Vorsätze zur Bezeichnung von Zehnerpotenzen

Vorsatz	Name	Faktor
a	atto	10^{-18}
f	femto	10^{-15}
p	piko	10^{-12}
n	nano	10^{-9}
µ	mikro	10^{-6}
m	milli	10^{-3}
c	zenti	10^{-2}
d	dezi	10^{-1}

Vorsatz	Name	Faktor
E	Exa	10^{18}
P	Peta	10^{15}
T	Tera	10^{12}
G	Giga	10^{9}
M	Mega	10^{6}
k	Kilo	10^{3}
h	Hekto	10^{2}
da	Deka	10^{1}

Wichtige physikalische Größen und ihre Einheiten

Größe	**Symbol**	**SI-Einheit**	**Name**
Kraft	F	$1\,\mathrm{N} = 1\,\mathrm{kg\,m\,s^{-2}}$	Newton
Energie/Arbeit	E oder W	$1\,\mathrm{J} = 1\,\mathrm{kg\,m^2\,s^{-2}} = 1\,\mathrm{N\,m} = 1\,\mathrm{W\,s} = 1\,\mathrm{V\,A\,s}$	Joule
Leistung	P	$1\,\mathrm{W} = 1\,\mathrm{kg\,m^2\,s^{-3}} = 1\,\mathrm{V\,A}$	Watt
Frequenz	f	$1\,\mathrm{Hz} = 1\,\mathrm{s^{-1}}$	Hertz
Spannung	U	$1\,\mathrm{V} = 1\,\mathrm{J\,C^{-1}}$	Volt
Ladung	Q	$1\,\mathrm{C} = 1\,\mathrm{A\,s}$	Coulomb
Widerstand	R	$1\,\Omega = 1\,\mathrm{V\,A^{-1}}$	Ohm
Letifähigkeit	G	$1\,\mathrm{S} = 1\,\mathrm{A\,V^{-1}}$	Siemens
Induktivität	L	$1\,\mathrm{H} = 1\,\mathrm{V\,s\,A^{-1}}$	Henry
magnetische Flussdichte	B	$1\,\mathrm{T} = 1\,\mathrm{Wb\,m^{-2}} = 1\,\mathrm{V\,s\,m^{-2}}$	Tesla
Kapazität	C	$1\,\mathrm{F} = 1\,\mathrm{C\,V^{-1}} = 1\,\mathrm{A\,s\,V^{-1}}$	Farad

Griechisches Alphabet

A	B	Γ	Δ	E	Z	H	Θ	I	K	Λ	M	N	Ξ	O	Π	P	Σ	T	Υ	Φ	X	Ψ	Ω
α	β	γ	δ	ε	ζ	η	ϑ	ι	κ	λ	µ	ν	ξ	ο	π	ρ	σ	τ	υ	φ	χ	ψ	ω

ANHANG F

INDEX

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

Z

ANHANG G

SONSTIGES