Entwurf und Implementierung verteilter Systeme

Rechnernetze 2010/11

Dr. Günter Kolousek

26.9.2010

Inhaltsverzeichnis

Vorwort

Dieses Dokument behandelt die Grundlagen der Rechnernetze, die für einen Entwickler von verteilten Systemen von Relevanz sind. D.h. es werden die Grundlagen in einem Detailierungsgrad präsentiert und aus einem Blickwinkel betrachtet, der für Entwickler verteilter Systeme zum Zwecke des Verständnisses notwendig ist.

Das Verständnis der Grundlagen ist deshalb wichtig, damit der Entwickler die nötigen Vorgaben zum Entwurf des Netzwerkes und der Wahl der Netztechnologie liefern kann oder auch, um die Wünsche und Anforderungen der Netzbetreiber zu berücksichtigen.

Unbedingt notwendig sind diese Grundlagen natürlich auch direkt für den Entwurf und die Implementierung verteilter Systeme. Diese Kenntnisse sind unabdingbar bei

- der Festlegung der Systemarchitektur,
- dem Entwurf einer Softwarearchitektur,
- der Implementierung.

Konkrete Beispiele für solche Tätigkeiten sind:

- Entwurf und Entwicklung von Protokollen. Solche Protokolle können z.B. nötig sein, um auf technische Geräte (Messgeräte, Boardcomputer in Autos oder Flugzeugen, Steuerungsrechner in Produktionsanlagen) oder bestehende Informationssysteme (engl. legacy systems) zuzugreifen.
- Implementierung von Clients für bestehende Protokolle und im speziellen Internetprotokolle, wie z.B. für das APP (atom publishing protocol, Protokoll zum Publizieren und Verwalten von Inhalten im Web).
- Erstellung serverseitiger Programme beispielsweise basierend auf einer Dämon-Struktur oder einer Super-Server-Struktur. Beispiele sind http Server, Server für betriebswirtschaftliche Anwendungen (z.B. SAP NetWeaver), Spieleserver.
- Erstellung einer Gefahrenanalyse bzw. Implementierung von Sicherheitsmaßnahmen in den Bereichen Erstellung von Sicherheitspolicies, Implementierung von Firewalls, Konzepten der Standortvernetzung...
- Entwurf, Implementierung und in Betriebnahme von Webapplikationen und Webservices z.B. im Bereich E-Commerce oder Webcontentmanagement.
- Entwurf, Implementierung und Inbetriebnahme von verteilten Anwendungen wie verteiltes Dokumentenmanagement, Workflowanwendungen mit RFID Integration, rechenintensive verteilte Anwendungen (Wettervorhersage, Simulationen bei der Entwicklung neuer Autotypen) oder zuverlässige Anwendungen mit vielen Geräten wie bei der Bargeldauszahlung in einem Netz von Bankomaten und den zugehörigen Servern.

Weitere Beispiele finden sich auf der Seite 17.

Struktur dieses Dokumentes

Dieses Dokument gliedert sich in sechs große Teile:

1. Grundlagen
2. Technologien
3. TCP/IP
4. Dienste und Anwendungen
5. Sicherheit

Der erste Teil dieser Unterlage erläutert die Grundlagen der Kommunikation und der Rechnernetze, danach werden im zweiten Teil konkrete Netztechnologien — wie die im LAN dominierende Ethernet-Technologie — vorgestellt. Der dritte Teil beschäftigt sich mit der heute wichtigsten Netzsoftware TCP/IP. Im vierten Teil werden heute häufig benutzte Dienste und Anwendungen des Internets besprochen und anschließend werden die notwendigen Grundlagen und Anwendungen rund um das Thema Sicherheit behandelt. Der letzte Teil behandelt verschiedene Aspekte, die bei der Entwicklung verteilter Systeme von Relevanz sind.

Teil I.

Grundlagen

Dieser Teil beschreibt die grundlegenden Mechanismen der Kommunikation und der Rechnernetze. Es wird in der „Einleitung" zuerst ein Überblick über die Themen Netzwerktechnik und Rechnernetze gegeben als auch der Begriff des verteilten Systemes eingeführt. Danach werden im Kapitel „Grundlagen und Basiskonzepte" die grundlegenden Begriffe und Konzepte vorgestellt, die in weiterer Folge für das Verständnis absolut notwendig sind. Im anschließenden Kapitel „Nachrichtenübertragung" werden die technischen Grundlagen für die Übertragung von Nachrichten in einer Punkt-zu-Punkt Verbindung behandelt. Abschließend werden die verschiedenen Arten und Aspekte der Netzarchitekturen präsentiert.

1. Einleitung

1.1. Überblick

In diesem Abschnitt wird ein grober Überblick über das Gebiet der verteilten Systeme und im speziellen der Rechnernetze gegeben.

Was ist überhaupt ein Netz? In einer ersten und sehr allgemeinen Definition verstehen wir darunter die Kommunikation mittels Protokolle zwischen zwei oder mehreren Kommunikationspartnern. Diese allgemeine Definition wird in weiterer Folge konkretisiert. Vorerst wollen wir uns diesem Thema jedoch mittels Beispielen annähern.

Je nach Betrachter kann man ein Netz aus verschiedenen Blickwinkeln betrachten. Beispiele sind:

- Werden einfach zwei Geräte mittels serieller Leitungen (Terminals, Modem,...) verbunden kann das aus der Sicht der Benutzer schon ein Netz darstellen.
- Aus der Sicht der Endbenutzer ist ein Telekommunikationsnetz mit seinen Diensten Telefon, Telefax, Telex (TELEprinter EXchange), GSM oder UMTS ebenfalls ein Netz.
- Ebenfalls aus Benutzersicht tritt das Fernsehnetz mit seinen Diensten Fernsehen, Kabelfernsehen und Teletext als Netz auf.
- In dieser Unterlage wollen wir uns jedoch ausschließlich mit Rechnernetzen beschäftigen, also solchen Netzen in denen Rechner im Vordergrund stehen. Aber selbst dieses Gebiet ist sehr breit gefächert, wie die folgenden Beispiele zeigen:
  - Ein Netz, das verschiedene Geräte für eine Industriesteuerung vernetzt stellt ganz spezielle Anforderungen an die Datenübertragung und die Ausfallsicherheit und wird als Feldbus bezeichnet.
  - Netzverbindungen zum Anschluss von Peripheriegeräten an einen Rechner wie z.B. der Gerätebus USB sollen viele verschiedene Geräte anbinden können und möglichst preisgünstig sein.
  - Ein Computersystem verbindet mehrere Rechner oder Prozessoren miteinander, um meist rechenintensive Aufgaben zu erledigen. Man kann prinzipiell unterteilen in Multiprozessor-Rechner (mehrere Prozessoren in einem Rechner) und Multicomputersysteme, die mehrere Rechner homogen (also alle gleichartige) oder heterogen (also verschiedenartige) zu einem Netz verbinden.
  - Allgemeine Rechner-zu-Rechner Kommunikation wie z.B. über das Internet. In diesem Sinne handelt es sich hierbei um ein heterogenes Rechnernetz.

In weiterer Folge werden wir uns nur mit Rechnernetzen also Netzen im Sinne der allgemeinen Rechner-zu-Rechner Kommunikation beschäftigen. Rechnernetze sind dadurch gekennzeichnet, dass

- die Kommunikationspartner beliebige Geräte sein können und nicht auf spezielle Anforderungen wie bei Telefon oder bei Geräteanbindungen eingeschränkt sind.
- die Kommunikation nicht auf spezielle Daten eingeschränkt ist. D.h. beliebige Daten wie z.B. WWW, E-Mail, Streaming-Audio und -Video (Download und Konferenzen), Chat, File-Sharing, verteiltes Rechnen oder elektronischer Handel.

Im folgenden Text werden Rechnernetze noch von weiteren Gesichtspunkten betrachtet, um die vielfältigen Aufgabenstellungen, die an Rechnernetze gestellt werden, zu veranschaulichen.

Behandlung der Daten

Betrachtet man die Kommunikation etwas näher so erkennt man, dass es auch verschiedene Arten gibt, wie Daten behandelt werden können. Das heißt, dass es verschiedene Arten der Anwendung von Daten gibt. Für Videodaten, gibt es z.B. folgende Arten: Video-Download, Video-on-Demand, Video-Konferenz. Jede dieser Anwendungen stellt besondere Anforderungen an das Netz:

- Ein Video-Download soll aus Benutzersicht lediglich möglichst schnell erfolgen. Ansonsten gibt es keine besonderen Bedingungen, die eingehalten werden müssen.
- Bei Video-on-Demand ist es z.B. akzeptabel 10s auf den ersten Frame zu warten, danach erwartet man sich jedoch eine synchrone Übertragung der Audio- und Videodaten ohne nennenswerte Verzögerungsschwankungen.
- Bei einer Video-Konferenz sind – im Vergleich zu Video-on-Demand – 10s Delay nicht akzeptabel und die Kommunikationsrichtung findet in beide Richtungen statt.

Charakteristisch bei Audio- oder Videosignalen ist auch, dass kleine Fehler oder ausgelassene Daten nicht stören! In einer reinen Datenkommunikation, müssen fehlerhafte Daten erkannt werden und danach entweder korrigiert oder wieder neu angefordert werden. Diese neuerliche Übertragung von Daten ist bei Audio- oder Videoanwendungen nicht notwendig und außerdem unerwünscht!

Zielstellung der Kommunikation

Je nach Zielstellung der Kommunikation kann man unterscheiden z.B. in:

- Datenübertragung (z.B. ftp).
- Zugriff auf nicht lokale Informationen, wie Daten (z.B. aus einem Datenbanksystem), Information (z.B. WWW) oder Programmen (z.B. mittels telnet oder besser ssh).
- Nachrichtenaustausch, kann entweder asynchron (z.B. E-Mail) oder synchron (z.B. Chat, Videokonferenz, Telefon) erfolgen.
- Allseitige Erreichbarkeit (kabellos, kabelgebunden).
- Broadcast-Kommunikation (z.B. Fernsehen, Rundfunk).
- Neue Arbeitsformen (z.B. Telelearning, Teleworking).
- Steuerung entfernter Prozesse.

Art der Verbindungen

- Direkte Verbindungen
  - Punkt-zu-Punkt Verbindung (z.B. serielle RS-232)
  - Mehrfachzugriffsverbindung (z.B. Ethernet, Bus)
- Vermitteltes Netzwerk
  - Leitungsvermitteltes Netzwerk (z.B. Telefonie)
  - Paketvermitteltes Netzwerk (z.B. Internet)

Nutzergruppen

Unterschiedliche Perspektiven aus denen Rechnernetze betrachtet werden können:

- Netzbenutzer: will möglichst einfachen Zugriff auf Netzressourcen.
- Netzbetreiber: ist interessiert an Nutzungsgebühren, Wartbarkeit, Skalierbarkeit. Je nach Betreiber kann man die Netze einteilen:
  - Öffentliche Netze (Telefon, Daten, Mobilfunk).
  - nicht öffentliche Netze (Sondernetze: Polizei, Verkehrswesen, Energieversorger).
  - Private Netze
    - Corporate Networks: Unternehmensnetze auf Basis gemieteter oder unternehmenseigener Übertragungskanäle.
    - Virtuell private Netze: Vernetzung von Unternehmensstandorten über fremde Übertragungskanäle, Kommunikation wie in einem lokalen Netz.
- Netzdesigner: ist interessiert an kostengünstigem Design, effektive Nutzung der Ressourcen. Der Netzdesigner benötigt Vorgaben vom Netzbetreiber aber auch Informationen über die Dienste, die im Netz angeboten werden sollen.
- Softwareentwickler ist interessiert an Diensten, die die Anwendung benötigt und die im Netzwerk zur Verfügung stehen. Der Softwareentwickler muss vom Netzdesign etwas verstehen, um dem Netzdesigner gegenüber seine Anforderungen formulieren zu können. Außerdem muss er die Auswirkungen des Netzdesigns auf die Anwendung abschätzen und auch die Anforderungen des Netzbetreibers beim Entwurf der verteilten Anwendung berücksichtigen.

Netzausdehnung

Je nach Netzausdehnung kann man unterscheiden in

PAN Personal Area Network, ca. 1m

LAN Local Area Network, ca. 10m – 1km

MAN Metropolitan Area Network, ca. 10km

WAN Wide Area Network, bis ca. 1000km

GN Global Network, weltumspannend, Zusammenschluss mehrerer WANs, ca. 10000km

1.2. Einsatzziele von Rechnernetzen

Die Rechner arbeiten in einem Rechnernetz autonom, aber trotzdem im Verbund an der Lösung einer oder mehrerer Aufgaben. Je nach Ziel des Verbundes kann man Rechnernetze unterscheiden in:

Datenverbund Das Ziel eines Datenverbunds ist die logische Verbindung räumlich getrennter (persistenter) Datenbestände über das Netz. Hier gibt es prinzipiell zwei Vorgehensweisen:

- Die Partitionierung eines Datenbestandes auf mehrere Standorte. Ziele: höhere Sicherheit, niedrigere Zugriffszeiten, bessere Auslastung
- Die Verteilung eines Datenbestandes über mehrere Standorte. Ziele: höhere Verfügbarkeit, niedrigere Zugriffszeiten

Geht es um einen schnellen Zugriff auf große Datenmengen, dann wird heute ein sogenanntes SAN (Storage Area Network) verwendet. Es werden die Computer mittels eines speziellen Netzes mit den Speicherressourcen verbunden. Diese Netze unterscheiden sich dadurch von einem LAN, dass die Netze auf die Übertragung von blockorientierten Daten optimiert sind. Zwischen Computer und Festplatte werden entweder ATA oder SCSI verwendet. In SANs wird meistens SCSI oder Fibre Channel verwendet.

Funktionsverbund In einem Funktionsverbund fasst man Geräte oder Systeme zur Realisierung spezieller Funktionen zusammen, so dass die anderen Knoten die neuen Funktionen nutzen können. D.h. es wird eine Verteilung von speziellen Funktionen auf spezielle Knoten vorgenommen. Beispiele sind: Superrechner, Abteilungsdrucker, Gatewaydienste wie Faxserver oder ein Internet-Zugang.

Leistungsverbund Bei einem Leistungsverbund wird eine geforderte Rechenleistung von mehreren Rechnern erbracht. D.h. es wird vorausgesetzt, dass eine Aufgabe in mehrere Teilaufgaben zerlegt werden kann, deren Abarbeitung auf mehrere Rechner verteilt werden kann.

Liegt das Hauptziel in der Bereitstellung einer möglichst großen Rechenleistung, dann verwendet man entweder Multiprozessorsysteme oder homogene Multicomputersysteme. Beide Arten haben einen homogenen Aufbau, d.h., dass die einzelnen Teile aus denen sie aufgebaut sind bzgl. einer gewissen Sichtweise gleich sind. Als („kostengünstige") Alternative bieten sich Grids an, die eine Vielzahl an – meist heterogenen – Rechnern miteinander verbinden. Diese sind jedoch nur für gut zerlegbare Aufgabenstellungen geeignet, da die Verbindungen zwischen den Rechnern die eigentliche Begrenzung der Rechenleistung darstellen.

Bei einem Multiprozessorsystem sind mehrere gleiche Prozessoren mit je einem eigenem Cache über einen Bus mit einem gemeinsamen Hauptspeicher verbunden. Das Betriebssystem muss in der Lage sein, mehrere Prozessoren zu verwalten. Die Schwierigkeit liegt in der gemeinsamen effizienten Nutzung des gemeinsamen Speichers.

Homogene Multicomputer sind in gewisser Weise wesentlich einfacher: Sie bestehen aus mehreren unabhängigen, aber „gleichen" Computern mit jeweils einem eigenen

Speicher, die über ein effizientes Verbindungsnetzwerk miteinander verbunden sind. Die Schwierigkeit liegt hier darin, eine wirklich effiziente Verbindung herzustellen. Das Verbindungsnetzwerk kann entweder wiederum ein Bussystem sein oder ein auf Schaltern basierendes Verbindungssystem.

In einem auf Schaltern basierenden Multicomputer werden die Nachrichten zwischen den Prozessen über ein Verbindungsnetzwerk geroutet und nicht mit Hilfe von Broadcasts über einen Bus übertragen. Eingesetzt wird häufig ein Maschennetz oder dreidimensionale Strukturen wie Würfel (engl. cube) oder vierdimensionale Strukturen (Hypercubes).

Das Betriebsystem für einen homogenen Multicomputer muss in der Lage sein die Ressourcen zu verwalten ohne, dass es einen gemeinsam genutzten Speicher gibt. Auf Grund der gleichartigen Hardware kann das Betriebssystem optimal auf die Hardware zugeschnitten werden.

Lastverbund Das Ziel eines Lastverbund ist die gleichmäßige Auslastung mehrer Ressourcen. Das heißt, dass ein stoßweiser Anfall einer Last auf verschiedene Rechner verteilt wird. In der Regel handelt es sich um Rechenleistung.

Verfügbarkeitsverbund Unter einem Verfügbarkeitsverbund versteht man die Schaffung fehlertoleranter Systeme, die auch bei Ausfall einzelner Knoten noch eine Mindestleistung erbringen können. Z.B. Server im Standby-Betrieb.

Allgemeiner Verbund Liegt der Schwerpunkt auf einem möglichst generellen Einsatz ohne das spezielle Anforderungen vorliegen, dann werden heterogene Multicomputersysteme eingesetzt. Diese Computersysteme sind aus Teilen zusammengesetzt, die sich wesentlich von einander unterscheiden. Bei diesen Teilen geht es um Prozessortyp, Speichergrößen oder I/O Bandbreite. D.h. die Kommunikation zwischen den einzelnen Rechnern ist nur mehr mittels Nachrichten möglich. Eine gemeinsame Nutzung der Ressourcen ist nicht mehr einfach möglich.

In weiterer Folge werden wir nur mehr heterogene Rechnersysteme hauptsächlich im Kontext eines algemeinen Verbandes betrachten.

1.3. Begriff und Beispiele eines verteilten Systems

Ein *verteiltes System* ist eine Menge voneinander unabhängiger Computer, die dem Benutzer wie ein einzelnes, zusammenhängendes System erscheinen. Der Zweck ist als eine Transparenz zu schaffen, die es ermöglicht die technischen Problemen der Netze und der Kommunikation zu verstecken. Eine genauere Betrachtung und Definition ist im Abschnitt **??** auf Seite ?? zu finden.

Beispiele verteilter Anwendungen:

- Rechenintensive Anwendungen werden mittels Parallelrechner, Cluster oder Grid-Computing gelöst:
  - Simulationen in der Wettervorhersage, für Sturm- und Tsunamiwarnungen oder Vorhersage von Vulkanaktivitäten.

  - Bei der Konstruktion neuer Autotypen werden Kosten gespart, da auf aufwändige Crashversuche weitgehend verzichtet werden kann.
  - In der Bioinformatik wird die aufwändige DNA-Sequenzanalyse mit Hilfe solcher Systeme gelöst.
  - Im Data-Mining geht es darum, in sehr großen Datenvorkommen Muster zu erkennen und zu strukturieren. Ein populäres Beispiel ist SETI@home bei dem es darum geht, aus den aus dem Weltraum empfangenen Signalen nach Hinweisen auf Leben zu durchsuchen. Es ist insoferne populär, da es sich der Internetbenutzer bedient, die ihre lokale, freie Rechnerkapazität zur Verfügung stellen.

- Integration von Legacy-Anwendungen und Einbindung neuer Systeme zu einem Gesamtsystem.
- Zusammenführung von Datenbeständen von verschiedenen Standorten mittels verteilten Datenbanken.
- Verteiltes Dokumentenmanagement.
- Fehlertolerante Anwendungen: Prozesssteuerung in einer Fabrik, einem Flugzeug oder einem Atomkraftwerk. Kennzeichen solcher Anwendungen ist, dass diese speziell auf den teilweisen Ausfall einzelner Komponenten oder den vollständigen Ausfall des gesamten Systems ausgelegt sind.
- Computer Supported Cooperative Work (CSCW): Groupware- und Workflowanwendungen. Aktuell ist auch die Einbindung von RFID in den Produktionsablauf und die Workflowsysteme.
- Teleworking, Distance Learning, Ausbildung: Tele-Medizin, Fernuniversität, Simulation von Flugsituationen in der Pilotenausbildung (Ausfall einzelner Komponenten, Wetter).
- Informationssysteme und -dienste: Hotelbuchung, Flugreservation.
- E-Commerce: Business-to-Business (B2B), Business-to-Consumer (B2C) wie z.B. Home-Banking.

Um solche Anwendungen entwickeln zu können, benötigt man entsprechende Techniken und vor allem aber ein Rechnernetz.

2. Grundlagen und Basiskonzepte

In diesem Kapitel werden die Grundlagen und Basiskonzepte der Rechernetze dargestellt. Diese Konzepte kommen alle auch im größeren Zusammenhang der Entwicklung verteilter Systeme vor.

Prinzipiell lässt sich sagen, dass diese Konzepte meistens in verschiedenen Sichtweisen und Anwendungsfällen vorkommen. Z.B. treten sie entweder auf einer logischen oder einer physischen Ebene auf oder sie betreffen entweder die Hardware oder die Software.

Eine Definition der grundlegenden Konzepte ermöglicht es uns, diese für die Erläuterungen der Grundlagen der Rechnernetze, der Beschreibung der Anwendungen von Rechnernetzen als auch der Entwicklung von verteilten Systemen konsistent anzuwenden. Es ermöglicht hiermit eine klare Beschreibung der weitergehenden Konzepte.

2.1. Grundlegende Begriffe

System Wir verstehen unter einem System eine abgegrenzte oder abgrenzbare Menge von Elementen, die miteinander in Relationen stehen. Die Struktur des Systems ergibt sich aus der Gesamtheit aller Relationen zwischen den Elementen, aus denen globale Systemeigenschaften entstehen.

Ein System lässt sich durch die Definition einer Systemgrenze von seiner Umwelt abgrenzen. Diese Systemgrenze stellt die Schnittstelle zur Umwelt dar.

Zur weiteren Strukturierung lässt sich ein System in Subsysteme unterteilen. Ein Subsystem ergibt sich damit aus der Zusammenfassung mehrerer Elemente, die zusammen die Schnittstelle des Subsystems definieren und bereitstellen. Ein Subsystem kann wiederum in weitere Subsysteme und Elemente gegliedert sein.

Komponente Eine Komponente (engl. component) ist eine *austauschbares* Element in einem System, das ihre innere Struktur nach außen kapselt und ihre öffentliche Funktionalität über Schnittstellen zur Verfügung stellt. Damit ist eine Komponente unabhängig verwendbar.

Objekt Unter einem Objekt versteht man allgemein einen Gegenstand oder das Ziel des Denkens oder Handelns. In der objekt-orientierten Programmierung versteht man darunter ein Element, das zur Laufzeit existiert. Objekte sind die Basisbausteine eines Programmes, die Nachrichten empfangen, Daten verarbeiten und auch selbst Nachrichten versenden können. Die drei grundlegenden Eigenschaftes eines Programmes sind, dass jedes Objekt:

- einen Zustand aufweist

- ein Verhalten hat und
- eine eindeutige Identität besitzt.

Zerlegung Eine Zerlegung (engl. decomposition) eines Problems in kleinere Teilprobleme stellt in der Regel eine Möglichkeit dar, die Komplexität des Gesamtproblems besser zu handhaben bzw. sogar die Komplexität zu verringern. Das Prinzip der Zerlegung wird eben auch bei Systemen oder Komponenten vorgenommen. In diesem Fall wird die Gesamtaufgabe in einzelne Teilaufgaben zerlegt, die von bestimmten Systemen, Komponenten oder Objekten erledigt werden, sodass das Gesamtproblem gelöst wird.

Kapselung Kapselung (engl. encapsulation oder auch information hiding) bedeutet, dass die innere Struktur von außen nicht einsehbar ist.

Abstraktion Bei einer Abstraktion (engl. abstraction) werden aus einer bestimmten Sicht (engl. view) die wesentlichen Merkmale (engl. feature) einer »Elementes« (beispielsweise ein Objektes, eine Komponente oder ein Systems) betrachtet. Abhängig von der Sicht können ganz unterschiedliche Merkmale abstrahiert werden.

Je nach betrachteter Menge von Merkmalen erhält man daher unterschiedliche Abstraktionen. Das heißt, dass die Abstraktion von der verwendeten Sichtweise abhängt.

Hierarchie Eine Hierarchie (engl. hierarchy) stellt eine *geordnete* Reihenfolge von Abstraktionen dar. D.h., man versucht eine Abstraktion in weiterer Folge durch eine oder mehrere Abstraktionen weiter zu verfeinern.

Schnittstelle Die Schnittstelle (engl. interface) definiert detailiert wie mit einem Element oder einem System interagiert werden kann. Damit wird einerseits eine Abstraktion festgelegt und andererseits wird eine Kapselung erreicht (wenn auf das Element nur über ihre Schnittstelle zugegriffen werden kann).

Schicht Ordnet man mehrere Abstraktionen in einer Hierarchie an, dann wird jeder Teil dieser Hierarchie eine Schicht genannt. Jede Schicht dieser Hierarchie hat eine klar definierte Schnittstelle zu seiner unten angrenzenden Schicht und eine klar definierte Schnittstelle zu der oben angrenzenden Schicht (siehe 2.4.1 auf Seite 25).

Nachricht Eine Nachricht (engl. message) stellt einen Informationsfluss von einem Kommunikationspartner zum anderen dar. Unter Information wollen wir hier – ohne eine Definition anzugeben – die Struktur (Syntax), den Inhalt (Semantik) und die Bedeutung (Pragmatik) einer Nachricht beschreiben. Eine Nachricht wird durch die Auswahl aus einer Menge von vorgegebenen Möglichkeiten (dem Zeichenvorrat) unter Verwendung von Regeln (Kodierung) gebildet.

Jede Nachricht wird von einem Sender (der Quelle) über einen Kanal zu einem Empfänger übertragen (der Senke).

Kommunikation Elemente kommunizieren miteinander, um gemeinsam ein bestimmtes Verhalten an den Tag zu legen. Dies erreichen sie dadurch, dass sie Nachrichten austauschen. Der Austausch der Nachrichten ist in einem Protokoll festgelegt.

Protokoll Der Austausch von Nachrichten muss gewissen Regeln entsprechen, damit die Kommunikationspartner einander verstehen. Protokolle sind präzise Festlegungen aller Regeln (z.B. Abfolge von Nachrichten, Formate der zu übertragenden Daten, den

zeitlichen Einschränkungen,...), die für eine Kommunikation notwendig sind. D.h. Protokolle definieren eine Sprache zwischen Kommunikationspartnern.

Protokolle sind der Grund warum Komponenten austauschbar sind: Wenn beide Kommunikationspartner das Protokoll verstehen, dann sind die Schnittstellen dieser Kommunikationspartner zueinander kompatibel. Daher kann eine Komponente durch eine beliebig andere Komponente ersetzt werden, solange die neue Komponente wieder eine Schnittstelle aufweist, die zu der Schnittstelle der anderen Komponente kompatibel ist.

2.2. Kommunikation

Der Begriff der Kommunikation als Austausch von Nachrichten stellt die grundlegende Sichtweise für das Thema der verteilten Systeme dar.

Signale Auf der pyhsikalischen Ebene werden Nachrichten in Form von Signalen (elektrisch, optisch, Funk) über eine entsprechendes Übertragungsmedium versendet. Ein Signal ist die physikalische Darstellungsform einer Nachricht. Es besteht aus einer diskreten oder kontinuierlichen Folge von Werten eines Signalparameters (z.B. Spannungswert, Stromwert oder Feldstärke).

Kommunikationsrichtung Wir unterscheiden zwischen:

Simplex Nur eine vorgegebene Richtung.

Half-Duplex Ebenfalls nur eine Richtung, aber diese kann sich ändern.

Duplex Beide Richtungen gleichzeitig.

Netzprozess Unter einem Netzprozess oder kurz Prozess versteht man eine unabhängige Ausführung eines Programmes im Kontext eines Rechnernetzes. Zwei oder mehrere Netzprozesse können miteinander kommunizieren bzw. als Gesamtheit eine Ausführung einer verteilten Anwendung darstellen. Dabei kann es sich um eine beliebige Anzahl an Hosts handeln, die einen oder mehrere Betriebssystemprozesse ausführen.

Anzahl der Kommunikationspartner

Unicast 1:1 Kommunikation.

Multicast 1:n Kommunikation, d.h. Gruppenruf: alle Teilnehmer einer Gruppe.

Anycast 1:1 Kommunikation, beliebiger Teilnehmer aus einer Gruppe.

Broadcast 1:n Kommunikation allerdings Rundruf: alle Teilnehmer im Netz.

Adresse Die Fragestellung, die man sich mit einer beliebigen Anzahl von potenziellen Kommunikationspartnern stellen muss: Wie finden sich zwei Kommunikationspartner, sodass sie überhaupt miteinander kommunizieren können? Zur Identifizierung eines Kommunikationsparters führen wir dazu das Konzept der Adresse ein, die einen Kommunikationspartner eindeutig identifiziert. Wir gehen vorerst davon aus, dass diese Adresse bekannt ist und sich auch nicht ändert.

Rolle Man kann oft die beiden Kommunikationspartner einteilen bzgl. Diensterbringer (engl. service provider) und Dienstnutzer (engl. service user). Der Diensterbringer wird meistens als Server bezeichnet und der Dienstnutzer meistens als Client. Der Server bietet einen gewissen Dienst (engl. service) an, den beliebige Clients nutzen können.

Art der Kommunikation Man unterscheidet zwischen verbindungsorientierter (engl. connection-oriented) und verbindungsloser (engl. connectionless) Kommunikation:

verbindungsorientiert Eine Verbindung ist eine temporäre Beziehung zweier Kommunikationspartner zum Zwecke des Datenaustausches. Charakterisiert ist eine Beziehung dadurch, dass

- eine Verbindung zuerst aufgebaut werden muss. Dann können die Daten übertragen werden und am Ende wird eine Verbindung wieder abgebaut. D.h. es gibt 3 Phasen: Verbindungsaufbau (engl. connection establishment), Datentransfer (engl. data transfer), Verbindungsabbau (engl. connection release). Der Vorgang des Abbauens einer Verbindung wird oft mit „disconnect" bezeichnet.
- Verbindungsparameter, die die Übertragung der Daten betreffen, definiert und ausgehandelt werden.
- beim Aufbau der Verbindung oft ein Bezeichner (engl. identifier) bereitgestellt wird, der die Verbindung eindeutig identifiziert und dadurch beim Datentransfer der Overhead einer etwaigen wiederholten Adressauflösung und Wegfindung vermieden wird.
- eine Folge von Nachrichten genau so beim Empfänger ankommen wie sie der Sender abgesendet hat.

verbindungslos Dabei wird jede einzelne Nachricht unabhängig von vorangehenden oder nachfolgenden Nachrichten versendet.

- Es gibt nur eine Phase, nämlich den Transfer der Daten. Im Zuge der Übermittlung der Daten werden auch alle – zur Übertragung notwendinger Parameter – mitgegeben.
- Eine Übertragung wird in der Regel nicht garantiert.

Multiplexen Unter Multiplexen (wörtlich: mehrfach) versteht man das Konzept der gemeinsamen Nutzung einer Systemressource durch mehrere Nutzer (siehe Abschnitt 3.3.3 auf Seite 41).

Verbindung Eine Verbindung ist eine über einen bestimmten Zeitraum andauernde Kommunikationsbeziehung, die eine verbindungsorientierte Kommunikation ermöglicht. Es gibt prinzipiell zwei Arten der Verbindungen:

leitungsvermittelt Bei der Leitungsvermittlung (engl. circuit switching) wird eine physikalische Leitung wie z.B. eine Wählleitung (engl. switched circuit) oder eine Standleitung (engl. leased line) geschalten (Raummultiplex, siehe Abschnitt 3.3.3 auf Seite 42) oder eine virtuelle Leitung mittels synchronem Multiplexen (siehe Abschnitt 3.3.3 auf Seite 42) aufgebaut.

paketvermittelt Die Paketvermittlung (engl. packet switching) ist eine Paketübertragung mittels asynchronen Zeitmultiplexverfahren (siehe Abschnitt 3.3.3 auf Seite 42) und bildet heute die Basis für Rechnernetze. Es hat sich hier der Begriff Paket (engl. packet) etabliert, da entweder eine Nachricht als Pakete aufgefasst wird bzw. die Nachrichten in einzelne Pakete aufgeteilt werden. Danach werden die Pakete im asynchronen Zeitmultiplexverfahren übertragen. In Abschnitt 2.4.3 auf Seite 27 werden wir den Begriff Paket noch genauer eingrenzen und im Abschnitt 4.2.2 auf Seite 61 die Paketvermittlung noch genauer behandeln.

Eine paketvermittelte Verbindung wird als virtuelle Verbindung (da eben nicht physikalisch vorhanden) oder auch als logischer Kanal oder kurz Kanal bezeichnet.

2.3. Leistung

In weiterer Folge werden wir die „Übertragungskapazität" eines Kanales definieren.

Bandbreite und Datenrate

Als *Bandbreite* (engl. bandwidth) eines Übertragungskanales verstehen wir die Differenz der oberen Grenzfrequenz und der unteren Grenzfrequenz eines physikalischen Übertragungskanales. D.h. zwischen oberer und unterer Grenzfrequenz ist eine Signalübertragung möglich. Die konventionelle Telefonleitung hat eine untere Grenzfrequenz von 300Hz und eine obere Grenzfrequenz von 3300Hz. Damit übergibt sich eine Bandbreite von 3000Hz.

In Abhängigkeit der Leitungskodierung (siehe Abschnitt 3.3.2 auf Seite 41) ergibt sich eine (Roh-)*Datenrate R* in Bit/s. Aber Achtung: auch hier wird oft von der Bandbreite gesprochen: „Die Bandbreite einer 10MBit Ethernet-Verbindung beträgt 10MBit/s".

Durchsatz

Die gemessene Datenrate eines Übertragungskanales wird als *Durchsatz* (engl. throughput) bezeichnet und wird meistens in Bit/s angegeben. Zum Beispiel kann der eigentliche Durchsatz auf einer 10MBit Ethernet-Verbindung in Abhängigkeit von der Anzahl der gleichzeitigen Sender lediglich 2MBit/s betragen.

Aufpassen muss man außerdem, dass in der Regel der Durchsatz eines logischen Kanales – zum Beispiel über eine Verbindung – gemessen wird. Dadurch kann der eigentliche Durchsatz noch kleiner werden. Das rührt daher, dass eine Verbindung ein Protokoll verwendet, das einen gewissen Overhead erzeugt.

Achtung: es wird oft der Begriff Bandbreite oft synonym zu Durchsatz verwendet!

Latenz

Latenz (engl. delay, latency) ist die Dauer, die eine Nachricht von einem Ende des Kanales zu dem anderen Ende benötigt. Bei dem Kanal kann es sich wiederum um den Übertragungskanal oder um einen logischen Kanal handeln. Auf der Ebene des Übertragungskanales ist

die Latenz durch die Signalausbreitungsgeschwindigkeit bestimmt. Auf der Ebene der logischen Kanäle müssen die Verzögerungen der Zwischenknoten und die durch die Protokolle bedingten Verzögerungen hinzugezählt werden. Die Signalausbreitungsgeschwindigkeit liegt in der Größenordnung der Lichtgeschwindigkeit (also ca. 3.0×10^8 m/s): Bei Kupferkabel ca. 2.3×10^8 m/s und bei Lichtwellenleiter ca 2.0×10^8 m/s. Die Laufzeit t_D des Signals berechnet sich also als $t_D = s/v$ wobei s die Distanz und v die Signalausbreitungsgeschwindigkeit ist. Die tatsächliche Verzögerung ensteht zwischen Senden des ersten Bit und dem Empfangen des letzten Bit einer Nachricht berechnet sich als $t_L = t_D + t_M + t_W$, wobei t_M die Sendedauer der Nachricht und t_W die Summe aller Wartezeiten der Zwischenknoten ist. Die Sendedauer t_M berechnet sich als $t_M = N/R$, wobei N die Datenmenge in Bit und R die Datenrate in Bit/s ist.

Betrachtet man auf dieser Ebene den Durchsatz TP, dann entsteht dieser als $TP = N/(2t_D + 2t_W + N/R)$. Das Doppelte der Laufzeit des Signales – also $2t_D$ – wird auch als Round-Trip-Time (RTT) bezeichnet. D.h. die RTT gibt an wie lange eine Nachricht von dem Sender zum Empfänger und wieder zurück benötigt. Eine etwaige Verarbeitungszeit beim Empfänger ist hiermit noch nicht eingerechnet.

Es ist wieder zu beachten, dass die Begriffe in Abhängigkeit der gewählten Abstraktion eine verschiedene Bedeutung aufweisen.

Verzögerung-Bandbreite-Produkt

Stellen wir uns vor, dass auf einem Übertragungskanal einer direkten Verbindung permanent gesendet wird, dann gibt das Verzögerung-Bandbreite-Produkt (engl. bandwidth-delay product) die Anzahl der Bits an, die in dem Kanal gespeichert sind: Wird ein Bit beim Sender abgeschickt, dann erreicht dieses den Empfänger nach der Zeit t_D. Innerhalb dieser Zeitspanne befinden sich als $t_D \cdot R$ Bits auf der Leitung.

Betrachtet man eine Leitung mit einer Datenrate R von 10MBit/s sowie einer Verzögerung t_D von 50ms, dann sind immer $50 \times 10^{-3}s \times 10 \times 10^6 Bit/s = 500 \times 10^3 Bit$ in der Leitung gespeichert. Es handelt sich also um ca. 61KB.

Aufpassen muss man wie die Größenangaben Kilo, Mega, Giga in der Informatik und im speziellen in der Netzwerktechnik verwendet werden: Prinzipiell kann ein Kilo entweder 1000 oder 1024 also entweder 10^3 oder 2^{10} bedeuten! Warum ist das in der Netzwerktechnik besonders zu beachten? Die Angaben für die Bandbreite wird wie schon besprochen in Hz – also einer physikalischen Größe – angegeben. Daraus resultiert, dass ein kHz also 1000Hz sind. Wird jedoch von KB gesprochen, dann handelt es sich nicht um eine Größe aus der Physik sondern um eine aus der Informatik. Dementsprechend wird unter einem KB also 1024 Byte verstanden.

2.4. OSI Referenzmodell

Die Schichtenbildung spielt bei verteilten Systemen und im speziellen in der Netzwerktechnik eine große Rolle. Die ISO (International Organization for Standardization) ist die wichtigste internationale Normungsorganisation. Sie gibt allgemein anwendbare Normen

heraus wie z.B. für Währungscodes, Ländercodes oder die bekannten SI-Einheiten für physikalische Größen, aber auch wichtige Normen für die Informatik, wie z.B. die Zeichencodes (ISO-8859-15 oder UCS wie der UNICODE bei der ISO genannt wird). Für die Rechnernetze ist jedoch das OSI Referenzmodell der wichtigste Standard. OSI steht für „Open Systems Interconnection" und ist ein Referenzmodell für die Kommunikation zwischen offenen Systemen. Offen bedeutet in diesem Fall, dass sich diese Systeme an gemeinsame Standards halten, sodass eine Zusammenarbeit unabhängig von konkreten Hard- und Softwarekomponenten möglich ist, solange sich diese Komponenten an diese Standards halten.

Das OSI Referenzmodell wurde herangezogen, um konkrete Systeme auf dieser Basis zu entwickeln, die sich jedoch hauptsächlich nur in der Telekommunikationstechnologie etabliert konnten. Das liegt daran, dass es sich um ein sehr umfangreiches Modell handelt, das einen hohen Aufwand bei der Implementierung bedeutet. Außerdem waren diese ersten Systeme nicht besonders effizient, nicht fehlerfrei und auch nicht miteinander kompatibel. Ein anderer, vielleicht der wichtigste, Grund liegt darin, dass das Internet mit der Technologie TCP/IP vorherrschend im Einsatz ist und sich demzufolge durchgesetzt hat.

Heute wird das OSI Referenzmodell hauptsächlich dafür herangezogen die Grundlagen und das Verständnis für Netzprotokolle zu vermitteln. In diesem Sinne wird das OSI Modell hier beschrieben. Im Teil III auf Seite 85 wird gezeigt wie das TCP/IP Modell auf das OSI Modell abgebildet wird und welche die Unterschiede zwischen dem TCP/IP Modell und dem OSI Modell existieren.

2.4.1. Schichtenbildung

Es handelt sich beim OSI Modell um eine Hierarchie von 7 Schichten. Jede dieser Schichten besitzt eine Schnittstelle, die von der darüberliegenden Schicht verwendet werden kann und eine Schnittstelle, die von der darunterliegenden Schicht verwendet werden kann. Diese Hierarchie ist strikt. Das bedeutet, dass eine beliebige Schicht nur genau auf die Funktionen der direkt unterliegenden Schicht zugreifen kann.

Warum eine Hierarchie? Wenn man sich die vielfältigen Aufgaben vor Augen führt, die ein verteiltes System zu erfüllen hat, dann erkennt man, dass es sich bei einer Netzwerksoftware um ein komplexes Gebilde handelt. Wie schon erwähnt, ist es eine Möglichkeit mit Komplexität umzugehen, indem eine Hierarchie von Abstraktionen gebildet wird.

An Aufgaben fallen für die Netzwerksoftware zum Beispiel an:

- Sicherstellung, dass auf einer Punkt-zu-Punkt Übertragungsstrecke die Daten fehlerfrei übertragen werden.
- Bereitstellung von Adressen, um Knoten identifizieren zu können.
- Vermittlung und Wegsuche über mehrere Teilnetze hinweg.
- Sicherstellung der Vollständigkeit und der Reihenfolge der einzelnen Informationsblöcke.
- Vergabe von Namen, die auf Adressen abgebildet werden und für den Benutzer aussagekräftig sind.
- Auf- und Abbau von Verbindungen.

- Auf- und Abbau von Sitzungen.
- Umwandlung von Datenrepräsentationen.
- Bereitstellung von Diensten für Anwendungen wie z.B. Dateiübertragung.

Auf Grund dieser vielfältigen Aufgaben wurde eine Hierarchie von 7 Schichten gewählt, die jeweils eine bestimmte Funktion erfüllen. Diese Struktur mit den 7 Schichten ist in Abbildung 2.1 zu sehen.

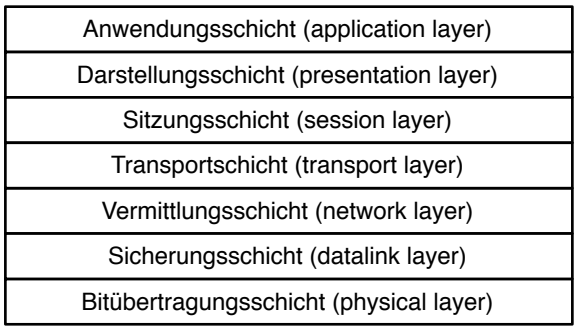


Abbildung 2.1.: OSI Schichtenmodell

Die Funktionen der einzelnen Schichten werden in den nachfolgenden Abschnitten beschrieben.

Zwei Netzwerksprozesse kommunizieren so miteinander, dass jeder Prozess – also jedes laufende Anwendungsprogramm – ausschließlich direkten Kontakt mit der obersten Schicht – der Anwendungsschicht – aufnimmt. Diese Anwendungsschicht verwendet zur weiteren Erfüllung ihrer Funktionen die direkt darunterliegende Schicht. Jede einzelne Schicht verwendet ihrerseits jeweils nur die direkt darunterliegende Schicht. Das wird solange fortgeführt bis die unterste Schicht – die Schicht 1 – erreicht ist. Diese Schicht ist die Bitübertragungsschicht, da auf dieser Ebene die eigentliche Übertragung der Bits in Form von Signale stattfindet. Danach wird beim Kommunikationspartner die Hierarchie der Schichten von unten nach oben durchlaufen. Bei der Antwort wird der gleiche Weg zurückgenommen. Dieser Vorgang ist in Abbildung 2.2 auf der nächsten Seite dargestellt.

2.4.2. Protokollhierarchie

Meistens ist es so, dass die Prozesse auf verschiedenen Knoten im Netz laufen. Damit handelt es sich notwendigerweise auch um verschiedene Ausprägungen der Netzwerksoftware. Eine Schicht einer auf einem Knoten installierten Netzsoftware wird in diesem Zusammenhang als Instanz einer Schicht bezeichnet. In diesem Sinne kommunizieren zwei Instanzen einer Ebene untereinander mittels eines Protokolles. Da die eigentlichen Signale nur über die physikalische Übertragungsstrecke übertragen werden können, findet die Kommunikation – wie oben beschrieben – über den Umweg der darunterliegenden Instanzen statt. Dadurch entsteht eine Protokollhierarchie, auch Protokollstack genannt (siehe Abbildung 2.3 auf Seite 28).

Eine beliebige benannte Schicht wird in der OSI Nomenklatur als (N)-Schicht bezeichnet, wobei N die Werte 1 bis 7 annehmen kann.

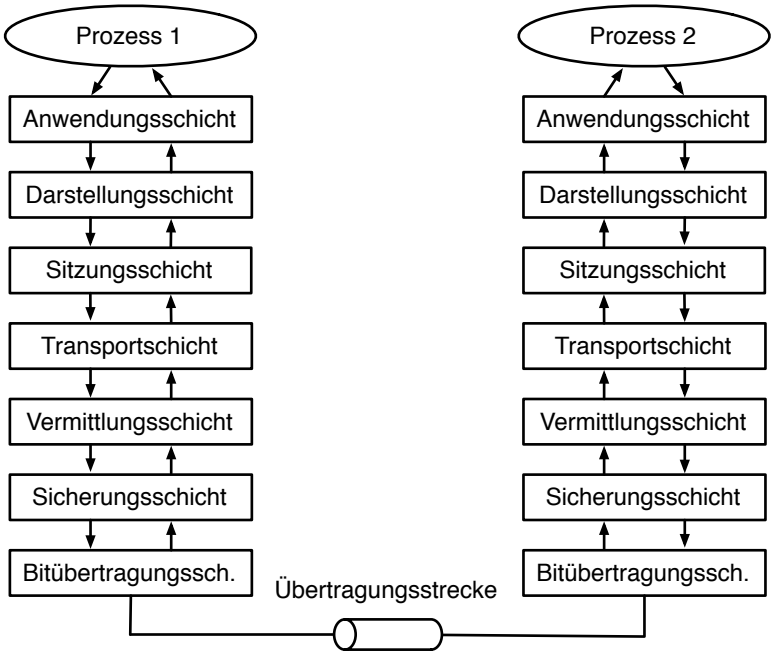


Abbildung 2.2.: Kommunikation zwischen den Schichten im OSI Modell

2.4.3. Nachrichten zwischen Schichten

Die eigentliche Kommunikation wird so durchgeführt, dass jeder Informationsblock, der nach unten an die unterliegende Instanz gegeben wird, in einen Umschlag (engl. envelope) verpackt wird, der zusätzliche Informationen beinhaltet. Bei diesem Umschlag handelt es sich um einen Vorspann (engl. header) und manchmal auch um einen Nachspann (engl. trailer). Beim Prozess des Kommunikationspartners wird dieser Umschlag von der Partnerinstanz entfernt und der so erhaltene Informationsblock an die obenliegende Instanz weitergereicht. Der Rückweg wird wieder analog behandelt (siehe Abbildung 2.4 auf Seite 29).

Die grau hinterlegten Teile der Datenblöcke sind Teile des Umschlags. Bei TCP/IP wird ein Trailer nur auf Schicht 2 verwendet. In Abbildung auf Seite 29 wurde aus diesem Grund nur in der Schicht 2 ein Trailer eingezeichnet.

Diese Informationsblöcke, die zwischen den einzelnen Schichten ausgetauscht werden, sind die eigentlichen Nachrichten, die zwischen je zwei Instanzen einer Schicht im Zuge des gemeinsamen Protokolls ausgetauscht werden. Dabei haben sich teilweise je nach Schicht eigene Bezeichnungen etabliert:

- Auf der Schicht 2 wird von einem Rahmen (engl. frame) gesprochen. Handelt es sich um kurze Rahmen mit fester Größe, dann werden diese Rahmen als Zellen (engl. cell) bezeichnet.
- In der Schicht 3 wird von Datagrammen (engl. datagram) oder auch von Paketen (engl. packet) gesprochen.
- Im Kontext von TCP werden die Nachrichten Segmente (engl. segment) genannt. Bei UDP wird ebenfalls von Datagrammen gesprochen.

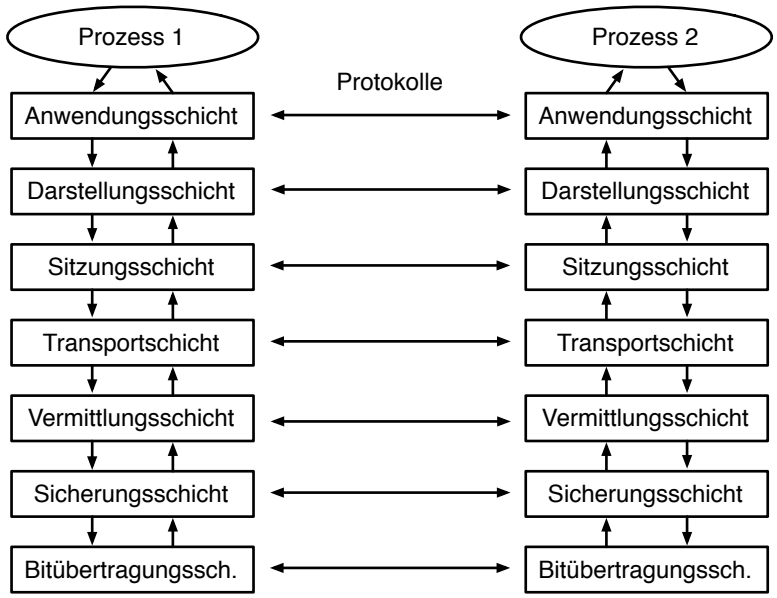


Abbildung 2.3.: Protokollhierarchie der 7 OSI Schichten

- Auf den restlichen Schichten 5 bis 7 bzw. als Überbegriff wird der Begriff Nachricht (engl. message) verwendet.

Nachrichten im Sinne der Schichten 2-4 sind dadurch charakterisiert, dass sie eine festgelegte maximale Größe aufweisen. Dadurch unterscheiden sie sich von den Nachrichten, die direkt zwischen den Anwendungen ausgetauscht werden, die keinerlei Einschränkung bzgl. der Größe unterliegen.

2.4.4. Protokollmechanismen

In den einzelnen Schichten gibt es Funktionen, die bzgl. einer Abstraktion vergleichbar sind. Diese Funktionen werden als Protokollmechanismen bezeichnet. Diejenigen Funktionen, die spezifisch für eine jeweilige Schicht sind, werden dann im Abschnitt 2.4.5 auf Seite 33 beschrieben.

Protokollauswahl

Die Auswahl eines Protokolles (engl. protocol selection) bzw. die Identifizierung (engl. identification) ist eine der ersten Aufgaben beim Kommunikationsbeginn. Öfters ist es so, dass zwischen zwei Kommunikationspartnern oder spezieller in einer Schicht mehrere Protokolle zur Verfügung stehen. Eines dieser Protokolle muss zur Kommunikation ausgewählt werden. Dazu muss es natürlich identifiziert, also benannt werden.

Im Laufe der Lebenszeit eines Systemes ist es sehr wahrscheinlich, dass das Protokoll an die laufenden Änderungen angepasst werden muss (Protokollevolution). D.h. das Protokoll wird in mehreren Versionen vorliegen und es ist die Auswahl einer konkreten Version (engl. protocol version selection) notwendig.

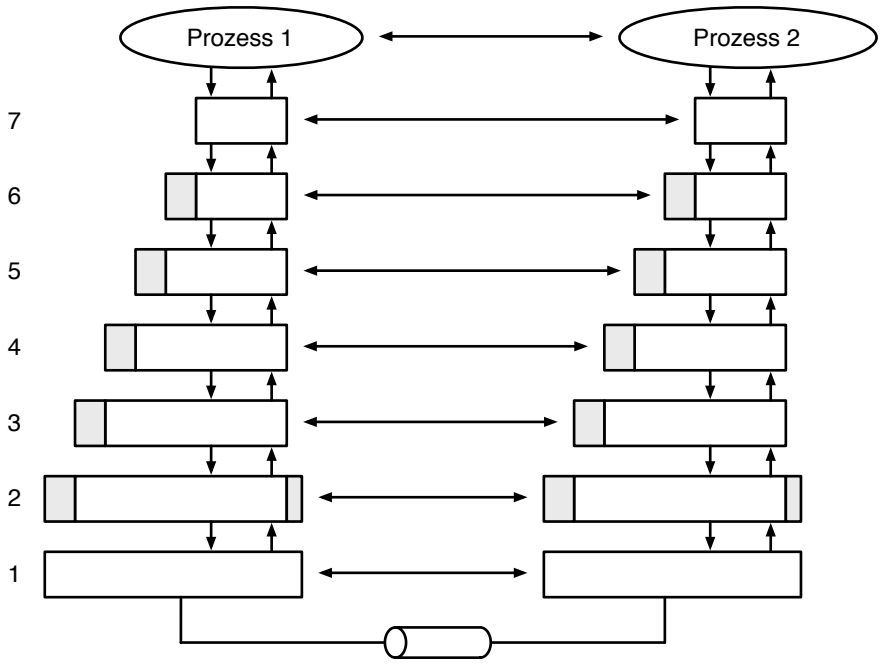


Abbildung 2.4.: Nachrichtenfluss zwischen den 7 OSI Schichten

Diese Auswahl eines geeigneten Protokolles bzw. einer speziellen Version muss zwischen den beiden Kommunikationspartnern ausgehandelt werden (engl. negotiation mechanism).

Verbindungen

Der Aufbau einer Verbindung (engl. connection establishment), der Transfer der Daten und der Abbau einer Verbindung (engl. connection release) sind die grundlegenden Elemente im Lebenszyklus einer Verbindung. Allgemein verbindet eine Verbindung zwei Kommunikationspartner, bei denen es sich wiederum speziell um Instanzen von Schichten handeln kann.

Eine Verbindung in einer (N)-Schicht kann mittels einer Verbindung auf der (N-1)-Schicht oder verbindungslos auf der (N-1)-Schicht realisiert werden. Eine verbindungslose Kommunikation auf einer (N)-Schicht kann verbindungsorientiert oder verbindungslos mittels der (N-1)-Schicht realisiert werden. Es ergeben sich daher 4 verschiedene Möglichkeiten.

Abweichend vom normalen Datentransfer ist es manchmal sinnvoll, Daten vorrangig auszuliefern. Dies wird als Vorrang-Datentransfer (engl. expedited data transfer) bezeichnet.

Zusätzlich zum normalen Abbau einer Verbindung gibt es auch einen speziellen Verbindungsabbruch (engl. abort). Dieser ist dadurch gekennzeichnet, dass ausstehende Übertragungen nicht mehr durchgeführt werden, während bei einem normalen Verbindungsabbau diese ausstehenden Übertragungen sehr wohl noch durchgeführt werden.

Manche Dienste erfordern eine Zurücksetzfunktion (engl. reset), die den Dienst oder im Speziellen die Verbindung in einen vordefinierten Zustand zurücksetzen. Das kann notwendig werden, wenn das Protokoll von einem Partner nicht mehr eingehalten wurde. Dadurch kann es zu einem Verlust oder zu einer Duplizierung von Daten kommen.

D.h. man kann zwischen verbindungsorientierten (engl. connection-oriented) und verbindungslosen (engl. connectionless) Protokollen unterscheiden:

- Verbindungsloses Protokoll
  - effizient
  - wenn Netzwerk relativ zuverlässig ist (wie in einem LAN), d.h. solange Pakete nicht verloren gehen oder beschädigt werden.

    Beispiel: Wenn keine Antwort kommt, dann muss z.B. der Client nochmals schicken. Problem: „Überweise 10.000 Euro von meinem Konto" sollte nicht nochmals gesendet werden. D.h. die Anforderung nochmals zu senden macht nur in bestimmten Fällen Sinn.
- Verbindungsorientiertes Protokoll
  - Overhead, da Einrichten und Abbauen einer Verbindung relativ kostspielig ist.
  - wenn Netzwerk nicht zuverlässig ist (wie in einem WAN).

Prinzipiell kann auch noch zwischen einem *zustandsbehafteten* und einem *zustandslosen* Protokoll unterschieden werden. D.h. hängen im Protokoll die Nachrichten vom Zustand der vorhergehend gesendeten Nachrichten ab oder nicht.

http ist ein typisch zustandsloses Protokoll, das über ein verbindungsorientiertes Transportprotokoll übertragen wird. Man beachte die verschiedenen Protokollhierarchieebenen, die hier eine Rolle spielen.

Eine *Sitzung* (Session) ist charakterisiert, dass es

- eine feste Beziehung zwischen den kommunizierenden Prozessen auf Anwendungsebene mit vereinbarten Eigenschaften (Namen, Ressourcen, Charakteristika,...) gibt.
- einen gemeinsamen Zustand zwischen den kommunzierenden Prozessen während der Session gibt.
- meist auch Mechanismen der Authentifikation und Autorisierung gibt.

D.h. eine Session kann nur mit einem zustandsbehafteten Protokoll aufgebaut werden. Setzt man diese Aussage in den Zusammenhang mit Sessions im Web, dann kann man einwerfen, dass es sich bei http um ein zustandsloses Protokoll handelt. Das ist richtig, deshalb wird in solch einem Fall der Zustand explizit zwischen den Kommunikationspartnern bei jedem Nachrichtenaustausch übertragen. Oft wird jedoch nicht der gesamte Zustand (vgl. Warenkorb) übertragen, sondern nur eine eindeutige Information unter der der Zustand im Server gespeichert ist.

Sicherstellung einer zuverlässige Übertragung

Zur Fehlererkennung (engl. error detection) und Fehlerkorrektur (engl. error correction) gibt es verschiedene Möglichkeiten.

Ein Verlust einer Nachricht kann erkannt werden, wenn nach dem Absenden einer Nachricht innerhalb einer vorgegebenen Zeitspanne (engl. timeout) keine positive Quittierung

(engl. acknowledgement oder kurz ACK) empfangen wird. Ein ACK kann entweder als eigene Nachricht oder als Huckepack-Quittierung (engl. piggy back acknowledgement) gesendet werden.

Zur Erkennung von Fehlern innerhalb einer Nachricht werden Prüfsummen (engl. checksum) herangezogen. Wird in einer Nachricht ein Fehler erkannt, dann gibt es die folgenden Möglichkeiten:

- Die Nachricht wird verworfen.
- Die Nachricht wird verworfen und es wird eine negative Quittierung (engl. negative acknowledgment oder kurz NACK) gesendet.
- Die Nachricht kann automatisch korrigiert werden.

Allgemein: Die positive Quittierung dient als Bestätigung für das Erreichen eines bestimmten Zustandes. Bei Nichterreichen eines bestimmten Zustandes kann analag dazu eben eine negative Quittierung gesendet.

Damit die Reihenfolge der Nachrichten innerhalb eines Protokolle sichergestellt werden kann, werden Sequenznummern verwendet (engl. sequencing). Diese Sequenznummern werden auch bei der Anpassung der Systemleistung (siehe Abschnitt 2.4.4) verwendet.

Anpassung an die Länge

Wie in schon Abschnitt 2.4.3 beschrieben ist es oft so, dass die maximale Größe einer Nachricht je Schicht unterschiedlich sein kann. Dadurch ergibt sich die Notwendigkeit die Größen der Nachrichten, die zwischen einer (N)-Schicht und der (N-1)-Schicht ausgetauscht werden anzupassen.

Den Vorgang des Aufteilens einer Nachricht auf mehrere Nachrichten geringerer Größe zum Zwecke der Anpassung an die geringere maximale Größe der anderen Schicht nennt man Segmentierung (engl. segmenting). Den umgekehrten Vorgang des Zusammensetzens mehrere kleinerer Nachrichten zu einer größen Nachricht nennt man Reassemblierung (engl. reassembling).

Anpassung an die Systemleistung

Der Empfänger von Nachrichten muss vor einer Überlastung durch den Sender geschützt werden. Dies nennt man *Flusskontrolle* (engl. flow control).

Der Schutz eines Netzes vor Überlastung durch die von allen Sendern gesendeten Nachrichten nennt *Überlaststeuerung* (engl. congestion control).

Anpassung der Übertragungsleistung

Will man eine Übertragungsleistung einer Verbindung mehreren Kommunikationspartner zur Verfügung stellen, spricht man von Multiplexen (engl. multiplexing). D.h. allgemein

wird eine (N-1)-Verbindung – eine Verbindung in einer (N-1)-Schicht – für mehrere (N)-Verbindungen nutzbar gemacht. Den analogen Vorgang die (N-1)-Verbindung wieder auf die (N)-Verbindungen aufzuteilen heißt Demultiplexen (engl. demultiplexing).

Wird allerdings eine Übertragungsleistung gefordert, die nicht durch eine Verbindung der unterliegenden Schicht erfüllt werden kann, dann müssen mehrere Verbindungen dieser untenliegenden Schicht verwendet werden. Dazu wird eine (N)-Verbindung auf mehrere (N-1)-Verbindungen aufgeteilt (engl. splitting) und beim Empfänger werden analog dazu die (N-1)-Verbindungen wieder zu der (N)-Verbindung zusammengesetzt (engl. recombining).

Dienstgüte

Die Verhandlung einer bestimmten Dienstgüte (engl. quality of service oder kurz QoS) fasst alle Parameter zusammen, die eine Verbindung oder eine einzelne Nachrichtenübermittlung betreffen. Diese Parameter werden beim Aufbau der Verbindung ausverhandelt (verbindungsorientiert) oder bei jeder einzelnen Nachrichtenübermittlung (verbindungslos). D.h. der Sender fordert Parameter an, die der Empfänger voll oder teilweise akzeptiert.

Beispiele für solche Parameter (nach Attributen aufgeschlüsselt):

Leistung

Durchsatz Unter dem Durchsatz (engl. throughput) wird die zugesicherte Menge an Benutzerdaten pro Zeiteinheit verstanden, die übertragen werden.

Übertragungsverzögerung Die Übertragungsverzögerung (engl. transmission delay, latency) gibt die maximale Zeitdauer zwischen dem Absenden der Anfrage und der Ankunft beim Emfänger an.

Schwankung der Übertragungsverzögerung (engl. jitter) ist die Varianz der Latenzzeit von Datenpaketen. Dieser Effekt ist insbesondere bei Multimedia-Anwendungen (zum Beispiel Audio-Streaming und IP-Telefonie) unerwünscht.

Verbindungaufbauverzögerung (engl. connection establishment delay) beschreibt die maximale Zeit die vergeht bis eine Verbindung aufgebaut ist.

Verbindungsbeendigungsverzögerung (engl. connection release delay) analog zur Verbindungsaufbauverzögerung.

Priorität Die Priorität (engl. priority) des Datentransfers gibt an inwieweit vorrangig Daten übertragen werden können.

Zuverlässigkeit

Vollständigkeit Angabe inwieweit eine Zusicherung gegeben werden kann, dass die Nachrichten zumindest einmal beim Kommunikationspartner ankommen bzw. inwieweit es zu Verlust von Nachrichten kommt (engl. probability of loss).

Fehlerraten Wahrscheinlichkeit der Datenveränderung (engl. probability of corruption).

Eindeutigkeit Vermeidung der Duplizierung von Nachrichten (engl. probability of duplication).

Reihenfolge Einhaltung der Reihenfolge der Nachrichten (engl. probability of out of sequence delivery).

Garantien Angabe inwieweit die Leistungsparameter erbracht werden können. Das meist genutzte Prinzip „best effort" gibt lediglich an, dass die Werte so gut wie möglich eingehalten werden, aber keine Garantie angegeben werden kann. Andere Angaben werden oft mit Wahrscheinlichkeiten beziffert, sofern andere Bedingungen – wie z.B. funktionierende Software und Hardware – erfüllt sind.

Sicherheit Darunter werden alle Maßnahmen angegeben, die die Sicherheit der Kommunikation sicherstellen können (siehe Teil **??**).

2.4.5. Schichten

Schicht 1 – Bitübertragungsschicht

In der Bitübertragungsschicht (engl. physical layer) werden die physikalischen Eigenschaften der Übertragungsstrecke beschrieben. Dies inkludiert das Übertragungsmedium (siehe Abschnitt 3.2 auf Seite 36) wie z.B. ein Kabel, das Übertragungsverfahren (siehe Abschnitt 3.3 auf Seite 39) und den Übergang auf das Übertragungsmedium wie z.B. Steckverbindungen. Im Detail handelt es sich um die elektrischen, mechanischen, optischen oder elektromagnetischen Spezifikationen.

Als Erweiterung zum OSI Modell kann die Bitübertragungsschicht noch weiter unterteilt werden in:

- Media Independent Interface (abgekürzt mit MII), bietet die einheitliche unterschiedlicher Übertragungsmedien.
- Physical Media Dependent (abgekürzt mit PMD) ist für den eigentlichen Zugriff auf das Übertragungsmedium zuständig. Z.B. kann als Übertragungsmedium entweder eine verdrillte Zweitdrahtleitung oder ein Koaxialkabel verwendet werden.

Schicht 2 – Sicherungsschicht

Die Hauptaufgabe der Sicherungsschicht (engl. data link layer) liegt in der fehlerfreien Punkt-zu-Punkt Übertragung ganzer Rahmen zwischen *benachbarten Stationen*. Diese Stationen können entweder direkt miteinander oder über einen Bus verbunden sein.

Im Falle eines Bussystems muss jeder Knoten auch eine eigene Adresse haben, die MAC-Adresse (engl. media access control) genannt wird. Es handelt sich dabei um eine Hardwareadresse, die direkt dem Netzwerkadapter des Knotens zugeordnet ist. Als solches muss diese MAC-Adresse zumindest innerhalb des Bussystems eindeutig sein. Häufig sind diese MAC-Adressen jedoch global eindeutig vergeben (siehe Abschnitt 5 auf Seite 75).

Als Erweiterung zum OSI Modell kann die Sicherungsschicht noch weiter unterteilt werden:

Logical Link Control (abgekürzt mit LLC), die die eigentlichen Funktionen der Schicht 2 abdeckt.

Media Access Control (abgekürzt mit MAC), die den Zugriff auf das Übertragungsmedium steuert.

Schicht 3 – Vermittlungsschicht

In der Vermittlungsschicht oder auch Netzwerksschicht (engl. network layer) genannt werden hauptsächlich die Nachrichten weitervermittelt. Die Übertragung geht über das gesamte Netzwerk hinweg und schließt das Weiterleiten zwischen den Netzknoten mit ein. D.h. eine Nachricht wird von einem (End-)Knoten – unter Umständen über mehrere Zwischenknoten hinweg – zu einem anderen (End-)Knoten übermittelt (Ende-zu-Ende, engl. end-to-end). Ebenfalls auf dieser Ebene sind die Netzadressen angesiedelt, die über das gesamte Netz eindeutig vergeben werden müssen.

Schicht 4 – Transportschicht

Die Transportschicht (engl. transport layer) stellt Verbindungen zwischen den kommunizierenden Netzprozessen als Ende-zu-Ende Verbindungen zur Verfügung. D.h., dass auf dieser Ebene wird einerseits das Konzept der Verbindungen und andererseits werden Mechanismen für eine Adressierung der Netzprozesse als Kommunikationspartner zur Verfügung gestellt. Wie schon erwähnt beinhaltet das Konzept der Verbindung auch, dass Nachrichten vollständig, fehlerfrei, in der richtigen Reihenfolge und genau einmal beim Empfänger eintreffen. Dafür ist ebenfalls die Transportschicht zuständig.

Schicht 5 – Sitzungsschicht

In der Sitzungsschicht (engl. session layer) wird die Steuerung der Kommunikation zwischen den kommunizierenden Netzprozessen gesteuert. Sie unterstützt eine Dialogsteuerung, um zu verfolgen welche Partei gerade spricht und stellt Funktionen für die Synchronisierung bereit. Dazu wird ebenfalls eine Verbindung auf dieser Ebene realisiert, die auch Wiederaufsetzpunkte (engl. check point) einsetzt, um die Kommunikation bei einem Synchronisationsverlust zwischen Sender und Empfänger wieder kontrolliert fortsetzen zu können.

Schicht 6 – Darstellungsschicht

Die Darstellungsschicht (engl. presentation layer) ist für die korrekte Umwandlung unterschiedlicher Datenformate (z.B. ASCII in UTF-8, little-endian in big-endian oder allgemeine Umwandlung von Zahlendarstellungen) zuständig. Ebenfalls zählen Datenkompression und Verschlüsselung zu den Aufgaben dieser Ebene.

Schicht 7 – Anwendungsschicht

Als oberste Schicht im OSI Modell stellt die Anwendungsschicht (engl. application layer) die Schnittstelle zum Netzprozess dar. Die Aufgaben dieser Schicht liegen im Bereitstellen von Diensten, die oft von Netzprozessen benötigt werden, wie z.B. Datenübertragung, E-Mail, oder entferntes Anmelden (engl. remote login).

3. Nachrichtenübertragung

Dieser Abschnitt erläutert die Grundlagen zur Nachrichtenübertragung zwischen *zwei* Kommunikationspartnern, die *direkt* physikalisch über ein Medium oder logisch über einen Kanal miteinander verbunden sind (Punkt-zu-Punkt Verbindung).

In diesem Kapitel werden einerseits die allgemeinen Aspekte einer Nachrichtenübertragung bei einer Punkt-zu-Punkt Verbindung behandelt, aber auch die spezielle Kommunikation über eine physikalische Übertragungsstrecke.

3.1. Kommunikationsmodell

Ein allgemeines Kommunikationsmodell, das dann natürlich auch für eine physikalische Punkt-zu-Punkt Verbindung gilt sieht folgendermaßen aus:

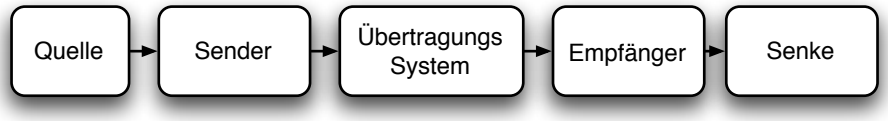


Abbildung 3.1.: Kommunikationsmodell

Quelle (engl. source) Die (Informations-)Quelle erzeugt die zu übertragenden Nachrichten.

Sender (engl. transmitter) Der Sender nimmt von der Quelle die Nachrichten entgegen, kodiert diese und wandelt sie in Signale um. Danach werden diese gemäß den Zugriffsregeln dem Übertragungssystem zum Transport übergeben.

Übertragungssystem (engl. transmission System) Im eigentlichen Übertragungssystem (oder auch Übertragungsstrecke genannt) werden die Signale übertragen. Im einfachsten Fall handelt es sich beim Übertragungssystem um eine elektrische Leitung über die elektrische Signale übertragen werden, im allgemeinen Fall jedoch um ein komplettes Netzwerk (siehe Kapitel 4 auf Seite 49).

Empfänger (engl. receiver) Am anderen Ende wandelt der Empfänger die Signale wieder in Nachrichten um, die entsprechend dekodiert an die Senke weitergegeben werden. Im Prinzip besteht ein Empfänger aus einem Signalumsetzer und einer Fehlersicherungseinrichtung.

Senke (engl. destination) Die (Informations-)Senke empfängt die Nachrichen und verarbeitet diese.

Sowohl Quelle als auch Senke werden aus der Sicht eines verteilten Systems als Netzprozess oder kurz Prozess betrachtet. Unter einem Prozess versteht man die Ausführung von einem Progamm. Zwei solcher Prozesse sind logisch über einen (Kommunikations-) Kanal miteinander verbunden. Die Rollen der Quelle und der Senke können natürlich dynamisch tauschen.

Technisch gesehen lässt sich das Kommunikationsmodell für eine physikalische Verbindung folgendermaßen abbilden:

Abbildung 3.2.: Datenübertragung

Unter der Datenendeinrichtung (DEE oder engl. Data Terminal Equipment, DTE) wird sowohl die Quelle der Information als auch die Senke verstanden. Im Falle der Rechnernetze handelt es sich dabei um die Rechner, die auch als Hosts bezeichnert werden. Unter einer Datenübertragungseinrichtung (DÜE oder engl. Data Communications Equipment, DCE) werden die Geräte verstanden, die die Transmitter bzw. die Receiver darstellen. Beispiele für solche DÜE sind z.B. Modems (MODulator/DEModulator) oder die Netzwerksschnittstellenkarten (Network Interface Card - NIC).

3.2. Übertragungsmedien

Dieser Abschnitt betrachtet jetzt speziell die verschiedenen Arten von Übertragungsmedien, die in Rechnernetzen eingesetzt werden. Prinzipiell erfolgt die Übertragung in einer von zwei Formen:

- leitungsgebunden d.h. entweder über metallische Leiter wie z.B. bei den Ethernet Technologien oder optische Fasern wie z.B. bei FDDI.
- leitungsungebunden d.h. mittels elektromagnetischer Wellen wobei als Übertragungsmedium die Luft dient. Hier haben sich folgende Spezialformen etabliert:
  - ungerichtete Übertragung wie z.B. beim Einsatz von Technologien im terrestrischen Funk wie Wireless LAN, Bluetooth, GSM, UMTS.
  - gerichtete Übertragung wie z.B. beim Richtfunk, der Satellitenkommunikation oder auf Basis optischer Signale wie z.B. Infrarotübertragung.

Je nach Übertragungsmedium werden zusätzlich entweder Steckverbindungen, Antennen oder andere Übertragungsglieder benötigt, um den Übergang des Signals auf das Übertragungsmedium durchzuführen.

3.2.1. Metallische Leiter

Es werden entweder symmetrische Kupferkabel oder Koaxialkabel eingesetzt. Bei Verwendung metallischer Leiter kommen heute nur mehr symmetrische Kupferkabel zum Einsatz.

Symmetrische Kupferkabel

Diese Form der Kupferkabel verwendet ein oder mehrere verdrillte Paare von Kupferadern. Aus diesem Grund werden diese Kabel auch als Twisted Pair (TP) bezeichnet. Der Aufbau der Kabelbezeichnungen ist nach ISO/IEC-11801:2002 folgendermaßen XX/YZZ wobei XX die Angabe der Kabelabschirmung darstellt und entweder

U unshielded, d.h. ohne Abschirmung des gesamten Kabels,

S screened, d.h. Abschirmung des gesamten Kabels mittels Drahtgeflecht,

F foiled, d.h. Abschirmung des gesamten Kabels mittels Folie,

SF screened, foiled, d.h. Abschirmung des gesamten Kabels mittels Drahtgeflecht und Folie

annehmen kann. Y steht für die Abschirmung der einzelnen Aderpaare und kann die Werte

U unshielded, d.h. keine Abschirmung der einzelnen Aderpaare,

S shielded, d.h. Abschirmung der einzelnen Aderpaare mittels Drahtgeflecht,

F foiled, d.h. Abschirmung der einzelnen Aderpaare mittels Folie

annehmen. ZZ ist immer TP (also Twisted Pair). In der Abbildung 3.3 ist ein 4 adriges TP Kupferkabel schematisch dargestellt.

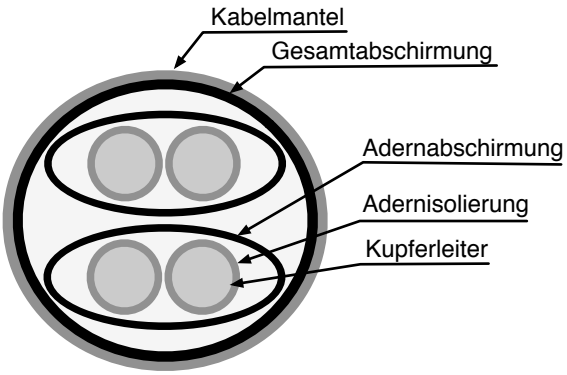


Abbildung 3.3.: TP Kupferkabel mit 4 Adern

In der ISO/IEC Spezifikation 11801 werden außerdem sogenannte Linkklassen definiert, die die Qualität einer gesamten Übertragungsstrecke inkl. Kabel und Steckverbindungen angeben. Es gibt die Linkklassen A bis F, in denen die notwendigen elektrischen Parameter definiert sind, die für den Einsatz von bestimmten Anwendungen – wie z.B. 100MBit Ethernet oder Gigabit-Ethernet – benötigt werden.

Je Linkklasse können verschiedene Kabeltypen eingesetzt werden. Die Kabeltypen werden in sogenannte Kategorien eingeteilt, wobei die Kategorien von 1 bis 7 durchnummeriert werden. Für jeden Kabeltyp sind wieder verschiedene elektrische Parameter festgelegt.

Aus der Kombination von Linkklasse und Kabeltyp ergeben sich überbrückbare Distanzen. Im LAN werden derzeit hauptsächlich Kategorie 5 (Cat-5) Kabel eingesetzt, die eine 100MBit Ethernet Verbindung über 100m erlauben. Mit einem Cat-5 Kabel bestehend aus 4 Doppeladern kann auch eine Gigabit Ethernet Verbindung in beide Richtungen aufgebaut werden. Für 10 Gigabit Ethernet muss allerdings ein Kabeltyp höherer Kategorie verwendet werden.

Zu dem Kabel passend müssen auch die Steckverbindungen ausgeführt sein. Als Steckverbindungen werden heute meist die 8 poligen RJ-45 Stecker verwendet.

Die Vorteile der Twisted-Pair Kabel liegen darin, dass die Verlegung relativ einfach ist, die Kosten gering sind, eine weite Anwendbarkeit besteht und die Verbreitung sehr hoch ist.

Koaxialkabel

Koaxialkabel bestehen gewöhnlich aus einem isolierten Innenleiter (auch Seele genannt), der vom Außenleiter umgeben ist. Dieser Außenleiter ist in einem konstanten Abstand um den Innenleiter angebracht und von diesem durch einen Isolator getrennt. Im Regelfall ist diese Ummantelung ebenfalls nach außen isoliert. Solch ein in Rechnernetzen eingesetztes Koaxialkabel ähnelt dem Fernsehkabel.

Das Koaxialkabel wurde früher in Ethernet Rechnernetzen verwendet. Dazu wurden hauptsächlich 2 Kabeltypen eingesetzt: Eines für die Ethernet Spezifikation 10Base5 und eines für die Ethernet Spezifikation 10Base2.

10Base5 (Thicknet oder Yellow Cable genannt) verwendete ein 10mm dickes Koaxialkabel. Die Verbindungen wurden mittels Anbohren des Kabels (im Abstand von min. 2.5m) und Anklemmen eines Transceivers, der über eine AUI (engl. access unit interface) Schnittstelle die Verbindung zum Ethernet-Controller herstellte.

10Base2 (Thinnet genannt) verwendete ein ca. 6mm dickes Koaxialkabel. Zur Verbindung wurden allerdings BNC (engl. bayonet Neill-Concelman) Stecker verwendet.

Koaxialkabel werden heutzutage nicht mehr oft verwendet.

3.2.2. Optische Leiter

Die optischen Leiter (engl. optical fiber) bestehen aus einem Kern (engl. core) und einem Mantel (engl. cladding). Das Lichtsignal wird in den Kern über eine Photodiode oder eine Laserdiode eingespeist. Da der Brechungsindex des Mantels niedriger ist als der des Kerns kommt es zu einer Reflexion am Übergang von Kern zu Mantel, wodurch sich das Lichtsignal im Kern zickzackförmig ausbreitet.

Es werden im Wesentlichen zwei Typen von Glasfasern unterschieden: dies sind die Multimodefasern und die Single- bzw. Monomodefasern. Multimodefaser haben einen Durchmesser des Kernes von $50\mu m$ oder $62.5\mu m$, während Singlemodefasern einen Kerndurchmesser von ca. 3 bis $9\mu m$ haben.

Auf Grund des relativ hohen Durchmessers der Multimodefasern wird das Licht in mehreren Wellen (Moden) übertragen. Dadurch kann es bei der Multimodeübertragung allerdings zu Signalbeeinflussungen kommen. Daher sind Multimodefasern für sehr lange Übertragungsstrecken bei hoher Bandbreite nicht geeignet. Bei der Singlemodefaser kann sich das Licht auf Grund des kleinen Kerndurchmessers nahezu geradlinig in einer Mode ausbreiten. D.h. es kommt zu keinen Signalbeeinflussungen und zu einer geringeren Dämpfung. Unter Dämpfung versteht man die Umwandlung von Schwingungsenergie in eine andere Energieform, d.h. es kommt zu einer Verminderung der übertragenen Energie im Verlauf einer Übertragungsstrecke. Dadurch lassen sich höhere Entfernungen bei größerer Bandbreite als bei den Multimodefasern überbrücken.

Optische Leiter zeichnen sich – abgesehen von der höheren Bandbreite bei größerer Entfernung – durch geringere Störempfindlichkeit und höhere Abhörsicherheit aus. Allerdings ist die Verlegung schwieriger und die Kosten sind höher.

3.2.3. Funk

Die Übertragung basiert darauf, dass hochfrequente Wellen (800MHz bis mehrere GHz) über das Übertragungsmedium Luft gesendet werden. Besonders in LANs oder auch in PANs (siehe Seite 15) werden häufig Netzwerke mittels Funktechnologie aufgebaut, aber auch die Abdeckung über immer größere geographische Gebiete wie ganze Städte wird aktuell. Der große Vorteil liegt in der Einfachheit und Flexibilität sowie in der leichten Wartbarkeit der Netze. Ebenfalls ein großer Vorteil liegt in der Mobilität. Ein drahtloser Übertragungskanal besitzt die Broadcast-Eigenschaft.

Funkübertragung hat mit vielen Problemen zu kämpfen:

- Es treten Störungen und Interferenzen auf. Die Qualität des Übertragungskanals ändert sich mit der Zeit.
- Das Übertragungsmedium muss mit anderen Kommunikationsteilnehmern geteilt werden. Unter Umständen handelt es sich um „ungewünschte" Sender.
- Es gibt viele Regulierungen (speziell bei der Frequenzvergabe) seitens nationaler Anforderungen. Daher ist die Übertragungskapazität beschränkt, da freie Frequenzbereiche nicht einfach zu finden sind.
- Auf Grund des einfachen Zugriffes auf das Medium Luft treten Sicherheitsprobleme auf.

Gegenüber der leitungsgebundenen Übertragung ergeben sich niedrigere Übertragungsraten im Bereich von 1 bis 54MBit/s. Neuere Entwicklungen gehen bis 540MBit/s.

3.3. Übertragungsverfahren

In diesem Abschnitt wird zuerst etwas näher auf Signale eingegangen, danach wird besprochen wie die Kodierung der Signale durchgeführt werden kann und zum Schluss wie mehrere Signale über eine Übertragungsstrecke mittels Multiplexen übertragen werden können.

3.3.1. Signal

Wie schon in Abschnitt 2.2 auf Seite 21 angesprochen, werden Nachrichten auf pyhsikalischer Ebene in Form von Signalen (elektrisch, optisch, Funk) über ein Übertragungsmedium versendet. Ein Signal ist die physikalische Darstellungsform einer Nachricht und besteht aus einer diskreten oder kontinuierlichen Folge von Werten eines Signalparameters (z.B. ein Spannungswert, Stromwert oder Feldstärke) über die Zeit.

Signalarten

Signale können bezüglich Wertevorrat oder Zeit entweder kontinuierlich oder diskret sein.

zeitkontinuierlich Werte eines Signals können zu jedem beliebigen Zeitwert innerhalb eines Zeitintervalls auftreten.

zeitdiskret Werte eines Signals können nur zu bestimmten Zeitwerten eines Zeitintervalls auftreten. Diese Zeitwerte sind oft innerhalb des Zeitintervalls äquidistant.

wertkontinuierlich Werte des Signals können jeden beliebigen Wert innerhalb des Werteintervalls annehmen.

wertdiskret Werte des Signals können nur bestimmte Werte des Werteintervalls annehmen.

Daraus ergeben sich 4 Möglichkeiten:

- zeitdiskret und wertdiskret: Man spricht von digitalen Signalen. Kommen genau zwei diskrete Werte im Wertebereich des digitalen Signales vor, dann spricht man von einem binären Signal. Oft werden die Begriffe digital und binär synonym verwendet.
- zeitdiskret und wertkontinuierlich.
- zeitkontinuierlich und wertdiskret.
- zeitkontinuierlich und wertkontinuierlich: analoge Signale.

Signalverarbeitung und Signalübertragung

Analoge Signale treten auf, wenn z.B. ein Audio- oder Videosignal vorliegt. Analoge Signale müssen in digitale Signale transformiert werden, damit sie in einem Rechner verarbeitet werden können. Dazu dienen Analog/Digital Wandler. Diese tasten das analoge Signal zu diskreten Zeitpunkten ab und ermitteln zu diesem Zeitpunkt einen diskreten Wert des Signalparameters.

Danach wird das Signal kodiert und kann entweder verarbeitet oder übertragen werden. Eine Übertragung findet entweder im Basisbandbereich oder im Breitbandbereich statt.

Unter dem Begriff Basisband (engl. baseband) versteht man denjenigen Frequenzbereich, in dem sich das zu übertragende Nutzsignal befindet. In der Regel hat ein Basisbandsignal daher einen Frequenzbereich von 0 bis f_{max}Hz. Bei der analogen Telefonie beispielsweise ist das Frequenzband der Bereich von 300 bis 3400 Hz.

Unter Breitband (engl. broadband) wird speziell bei der Datenübertragung ein Verfahren verstanden, bei dem die digitalen Signale – im Unterschied zum Basisbandverfahren – nicht

direkt übertragen werden, sondern auf ein oder mehrere hochfrequente Trägersignale aufmoduliert werden (Modulation). Damit gibt es bei einem Breitbandsignal einen Frequenzbereich von f_{min} bis f_{max}. Breitband wird jedoch oft auch mit einer hohen Bandbreite gleichgesetzt oder in der Bedeutung verwendet, dass es sich um eine schnelle Datenübertragung handelt.

Allgemein ist die Modulation die Änderung von Signalparametern (Amplitude, Frequenz, Phase,...) eines Trägersignals durch ein modulierendes/aufgeprägtes Signal. Bei der Demodulation wird das ursprüngliche Signal wieder zurückgewonnen. Gründe dafür liegen z.B. in der Mehrfachbenutzung des Übertragungssystems (z.B. Frequenzmultiplex, siehe Abschnitt 3.3.3 auf der nächsten Seite) oder einer besseren Übertragung und Filterung hochfrequenter Signale (Basisbandübertragung nur über elektrische Leitungen).

Nachdem das Signal übertragen worden ist, wird es unter Umständen wieder dekodiert, in ein analoges Signal gewandelt und ausgegeben (z.B. Fernseher, Lautsprecher).

Das gesamte Modell der Signalverarbeitung und Signalübertragung lässt sich folgendermaßen darstellen:

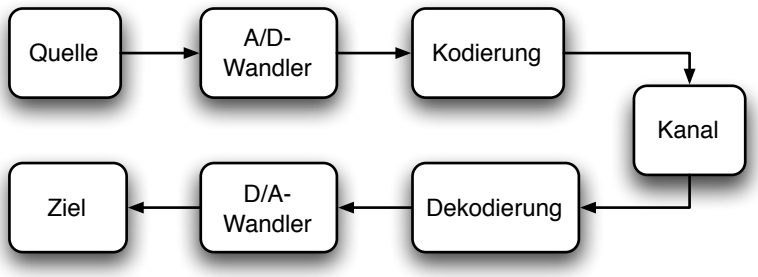


Abbildung 3.4.: Signalverarbeitung und Signalübertragung

3.3.2. Kodierung

siehe Foliensatz!

3.3.3. Multiplexverfahren

Unter Multiplexen (wörtlich: mehrfach) versteht man das Konzept der gemeinsamen Nutzung einer Systemressource durch mehrere Nutzer. Der Grund für Multiplexen liegt einerseits in der wirtschaftlichen Nutzung einer Ressource und andererseits in der Anforderung eine gewisse Form der Nebenläufigkeit zu erzeugen. Das bedeutet, dass mehrere Nutzer die Systemressource gewissermaßen gleichzeitig nutzen. Ein bekanntes Beispiel sind die Betriebssystemprozesse, die vom Scheduler des Betriebssystems in einem Multiplexverfahren dem Prozessor (die Systemressource) zugeteilt werden.

Betrachtet man den Begriff Ressource, der im Sinne der Rechnernetze und der Übertragung von Nachrichten eine Rolle spielt, dann handelt es sich bei der Ressource um konkrete Übertragungskanäle.

Im Kontext der Übertragungstechnik kann folgende Unterteilung der Multiplexverfahren getroffen werden.

Raummultiplex

Das Raummultiplex (engl. space division multiplexing) wird auch Space Division Multiple Access (SDMA) genannt. Die Signale werden dabei über räumlich verschiedene – also mehrfache (→ multiplex) – Übertragungswege übertragen. Sollen n Signale übertragen werden, dann werden n Leitungen verwendet. Das analoge Telefonnetz mit der Leitungsvermittlung kann in diese Kategorie eingeordnet werden.

Synchrones Multiplex

Die hier angeführten Multiplexverfahren werden deshalb als „synchron" bezeichnet, da die einzelnen Signale jeweils die gleichen (oder zumindest konstante) Anteile der Übertragungskapazität zur Verfügung gestellt bekommen.

Frequenzmultiplex Es werden beim Frequenzmultiplexverfahren (engl. frequency division multiplexing, FDM) die einzelnen Signale auf Träger unterschiedlicher Frequenz aufmoduliert. Die (lineare) Summe dieser modulierten Signale wird übertragen.

Codemultiplex Beim Codemultiplex (engl. code division multiplexing, CDM) wird jedes Bit mit einer Symbolfolge, die Code genannt wird, multipliziert. Die Codes sind zu einander orthogonal und jeder Sender erhält einen solchen Code. Die, sich so ergebenen Signale werden gleichzeitig auf dem Übertragungskanal im gleichen Frequenzband übertragen. Solange Empfänger mit dem Sender zeitlich synchronisiert sind, kann der Empfänger das empfangene Signal wieder mit einem einzelnen Code multiplizieren und erhält das einzelne Bit wieder zurück. Es ist ein aufwändiges Verfahren, das z.B. bei UMTS verwendet wird.

Synchrones Zeitmultiplex (engl. synchronous time division multiplexing, STDM oder STM) Es wird ein Rahmen (engl. frame) mit einer festgelegten Dauer definiert. Dieser Frame wird in Zeitschlitze (engl. time slots) unterteilt, die den einzelnen Signalen zugeordnet werden. Solche Frames werden nacheinander über den Übertragungskanal übertragen.

Asynchrones Zeitmultiplex

Das asynchrone Zeitmultiplexverfahren (engl. asynchronous time division multiplexing, ATDM oder ATM) wird auch als statistisches Multiplexen bezeichnet. Im Prinzip funktioniert es wie STM, jedoch erhalten nur diejenigen Signale einen Zeitschlitz, die auch etwas zu übertragen haben. Dieses Verfahren heißt deshalb auch statistisches Multiplexen, weil die verfügbare Übertragungskapazität im statistischen Mittel besser ausgenutzt wird. Damit allerdings ein Zeitschlitz einem Signal zugeordnet werden kann, muss eine Zuordnung im Zeitschlitz mitübertragen werden. Diese Zuordnung wird in der Regel mit einem Header durchgeführt, der am Anfang eines Zeitschlitzes übertragen wird. Dadurch entsteht ein gewisser Overhead.

Das asynchrone Zeitmultiplexverfahren wird außer in der Übertragungstechnik bei der Entwicklung von Kommunikationsprotokollen eingesetzt! Damit ist es das wichtigste Multiplexverfahren, das für den Entwickler von verteilten Systemen eine Rolle spielt.

3.4. Sicherungsverfahren

In diesem Abschnitt werden wir uns zuerst mit der Rahmenbildung beschäftigen, dann mit dem Thema Fehlererkennung und Fehlerkorrektur und anschließend Methoden zur zuverlässigen Übertragung zwischen Punkt-zu-Punkt Verbindungen beschreiben.

3.4.1. Rahmen

Die Bildung von Rahmen (engl. frames) behandelt das Erkennen von Anfang und Ende von Informationseinheiten. Auf der Ebene der physikalischen Übertragung von Bits geht es darum den Anfang und das Ende von Bitfolgen bzw. von Bytefolgen zu erkennen. So eine zusammenhängende Folge von Bits oder Bytes wird als Frame bezeichnet.

Protokolle, die auf einer Folge von Bytes basieren werden Byte-orientierte Protokolle genannt. Analog dazu gibt es die Bit-orientierten Protokolle. Der Unterschied zwischen einem Byte-orientierten Protokoll und einem Bit-orientierten Protokoll liegt darin, dass sich ein Bit-orientiertes Protokoll nicht um Byte-Grenzen kümmern muss.

Folgende zwei grundlegende Methoden werden wir behandeln: die Sentinel-Methode und die Zählmethode.

Sentinel-Methode

Die Sentinel-Methode (sentinel zu Deutsch: Wächter) geht davon aus, dass ein Frame mit je einem speziellen Startzeichen und einem Endezeichen – eben den Sentinel-Zeichen – markiert wird.

Das Protokoll BSC (engl. binary synchronous communication) von IBM ist ein Beispiel für ein solches Byte-orientiertes Protokoll. D.h. der Rahmen besteht aus einer Folge von Bytes, die durch je ein Startbyte (STX, start of text) und ein Endebyte (ETX, end of text) markiert werden. Was passiert allerdings, wenn ein ETX im Datenanteil des Frames enthalten ist? Dann wird diesem ETX Zeichen ein spezielles Zeichen (DLE, data link escape) vorangestellt. Sollte ein DLE Zeichen im Datenanteil vorkommen, dann wird diesem ebenfalls ein DLE Zeichen vorangestellt. Dieses Verfahren nennt man das character stuffing (Zeichen auffüllen). Beim BSC Protokoll gibt es zusätzlich zum Datenanteil noch einen Header, Synchronisationsbytes und eine Prüfsumme (2 Bytes).

Ein weiteres Beispiel für ein Byte-orientiertes Protokoll, das auf der Sentinel-Methode basiert ist PPP (point-to-point Protokoll).

Das bitorientierte Protokoll HDLC (high-level data link control) verwendet ebenfalls die Sentinel-Methode. Es wird als Anfangs- und Endesequenz jeweils die Bitfolge 01111110

verwendet. Auch hier stellt sich die Frage wie zu verfahren ist, wenn eine längere Folge von Einsen im Frame vorkommen. Der Sender fügt nach fünf aufeinanderfolgenden Einsen im Datenanteil eine Null in den Datenstrom ein. Der Empfänger verfährt folgendermaßen, wenn er eine Folge von fünf Einsen erhält: ist das nächste Bit eine 0, dann wird es vom Empfänger entfernt, ist es eine Eins, dann wird das darauffolgende Bit auch noch angesehen. Ist dieses eine Null, dann handelt es sich um die Endesequenz, ist es eine Eins, dann liegt ein Fehler vor. Im Fehlerfall wird der Frame verworfen und auf den Beginn des nächsten Rahmens gewartet.

Zählmethode

Die Zählmethode basiert darauf, dass die Bits bzw. Bytes des Datenanteils gezählt werden und diese Länge vor dem Datenanteil übertragen wird. Damit weiß der Empfänger genau wie lange der zu empfangene Datenanteil ist und muss nicht den Datenanteil auf ein Endezeichen untersuchen. Ein Nachteil dieser Methode ist, dass die Anzahl der übertragenen Bits bzw. Bytes beschränkt ist, da das Längenfeld eine feste Größe haben muss und dieses nicht zu groß gewählt werden sollte, damit der Overhead bei kurzen Frames nicht zu groß ist.

Ein Beispiel für ein Bit-orientiertes Rahmenformat, das auf der Zählmethode basiert ist das Ethernet Protokoll (siehe Kapitel 5 auf Seite 75). Als Beispiel für ein Byte-orientiertes Protokoll kann UDP (siehe Kapitel 9.2 auf Seite 104) dienen.

3.4.2. Fehlererkennung und Fehlerkorrektur

siehe Foliensatz!

3.4.3. Zuverlässige Übertragung

Hinausgehend über eine einfache Fehlererkennung (unter Umständen mit Fehlerbehebung in einzelnen Nachrichten) ist es notwendig, eine zuverlässige Übertragung sicherzustellen. Die wichtigsten Aufgaben, die im Zuge der zuverlässigen Übertragung von Nachrichten anfallen sind:

- Zuverlässige Zustellung von Nachrichten. Dafür gibt es zwei grundlegende Mechanismen: Bestätigungen (engl. acknowledgements) und Zeitablauf (engl. timeout). Eine Bestätigung oder kurz ACK ist eine kurze Nachricht vom Empfänger an den Sender, die die ursprüngliche Nachricht bestätigt. Ein ACK kann außerdem auch im Huckepackverfahren (engl. piggyback) an eine normale Nachricht, die der Empfänger an den Sender senden will, angeschlossen werden. Dadurch wird das Senden einer Nachricht eingespart. Durch den Empfang eines ACK weiß der Sender, dass der Empfänger die Nachricht erhalten hat. Erhält der Sender kein ACK innerhalb einer bestimmten Zeitspanne – eben dem timeout –, dann sendet der Sender die Nachricht noch einmal an den Empfänger.

- Einhaltung der Reihenfolge von Nachrichten. Die Reihenfolge der Nachrichten wird durch Sequenznummern realisiert.
- Flusskontrolle. Darunter versteht man einen Mechanismus mit dem der Empfänger den Sender drosseln kann, sodass der Sender Nachrichten nicht schneller sendet als der Empfänger diese verarbeiten kann. Im Prinzip funktioniert das so, dass der Sender nur senden darf, wenn nicht mehr als eine bestimmte Anzahl von ACKs ausständig sind (siehe Abschnitt 3.4.3 und Abschnitt 3.4.3).

In diesem Abschnitt werden die beiden wichtigsten Verfahren beschrieben, die jedoch auch in anderen Abschnitten erwähnt werden.

Stop-And-Go Algorithmus

Der Stop-And-Go Algorithmus basiert darauf, dass der Sender auf das ACK wartet bevor der Sender die nächste Nachricht absendet. Falls das ACK nicht innerhalb einer bestimmten Zeit (timeout) eintrifft, wird die ursprüngliche Nachricht noch einmal abgesendet.

Dadurch ergeben sich genau 4 Fälle:

1. Das ACK kommt innerhalb der timeout Zeitspanne an.
2. Die ursprüngliche Nachricht geht am Weg zum Empfänger verloren und der Sender sendet nach Ablauf der Zeitspanne die Nachricht erneut.
3. Das ACK geht am Weg vom Empfänger zum Sender verloren und der Sender sendet nach Ablauf der Zeitspanne die Nachricht erneut.
4. Das ACK trifft nach Ablauf der Zeitspanne ein und der Sender hat die ursprüngliche Nachricht noch einmal gesendet.

Die Fälle 3 und 4 sind insoferne interessant als das der Empfänger die Nachricht zwei Mal bekommt! Da der Empfänger nicht wissen kann, dass es sich beim zweiten Mal um eine Kopie der ersten Nachricht bekommt, muss es einen Mechanismus geben, der es ihm ermöglicht dies zu erkennen. Die Lösung sieht so aus, dass jede Nachricht noch einen Header aufweist, die eine 1 Bit lange Sequenznummer bekommt. Erhält der Empfänger mehrmals hintereinander die gleiche Sequenznummer weiß dieser, dass es sich um die gleiche Nachricht handelt.

Der Nachteil dieses Verfahrens ist, dass der Übertragungskanal nicht ausgelastet wird.

Sliding-Window Algorithmus

Bei dem Sliding-Window Algorithmus handelt es sich um eine Erweiterung des Stop-And-Go Algorithmus.

Sender

Der Sender weist jeder Nachricht eine Sequenznummer (engl. sequence number, abgekürzt mit SEQ) zu und verwaltet zwei Variablen:

- Die maximale Anzahl der noch nicht bestätigten Nachrichten, die der Sender senden kann, wird als Größe des Sendefensters (engl. send window size, SWS) bezeichnet.
- Die Sequenznummer des zuletzt empfangenen ACK (engl. last ack received, LAR).

Die Abbildung 3.5 zeigt das.

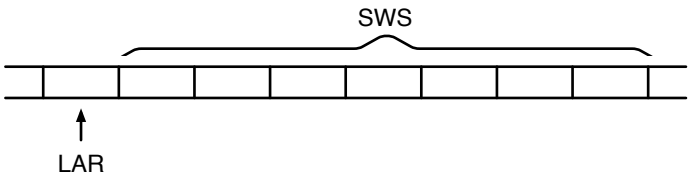


Abbildung 3.5.: Sliding-Window des Senders

Empfänger

Der Empfänger verwaltet drei Variablen:

- Die maximale Anzahl der Nachrichten, die der Empfänger annimmt ohne eine Bestätigung für diese gesendet zu haben, wird als Größe des Empfangsfensters (engl. receive window size, RWS) bezeichnet.
- Die Sequenznummer der zuletzt akzeptierten Nachricht (engl. last ack sent, LAS). D.h. das ist die größte Sequenznummer, für die eine Bestätigung geschickt wurde.
- Die kleinste Sequenznummer, die empfangen wurde und für die noch keine Bestätigung gesendet wurde (engl. first message not acknowledged, FMN).

Die Abbildung 3.6 illustriert dies:

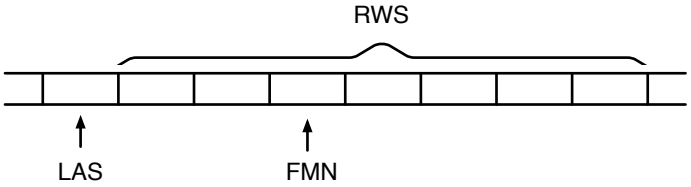


Abbildung 3.6.: Sliding-Window des Empfängers

Algorithmus

Der Sender schickt eine Nachricht mit der Sequenznummer SEQ ab, wenn die Bedingung $SEQ \leq LAR + SWS$ eingehalten ist. Das sagt aus, dass der Sender nicht mehr Nachrichten im voraus absendet als er Bestätigungen erhalten hat. Der Sender verhält sich folgendermaßen:

1. Für jede Nachricht, die der Sender abschickt startet er einen Timer und legt diese Nachricht in einem Zwischenspeicher ab. Wird für diese Nachricht eine Bestätigung empfangen, dann wird der Timer gestoppt. Kommt es zu einem Zeitablauf, dann wird die Nachricht, die zu diesem Timer gehört nochmals abgesendet.
2. Erhält der Sender eine Bestätigungsnachricht für die Sequenznummer SEQ, dann setzt der Sender LAR auf SEQ (wenn SEQ größer als LAR).

Der Empfänger verhält sich folgendermaßen:

1. Erhält der Empfänger eine Nachricht mit der Sequenznummer SEQ und befindet sich diese innerhalb des Empfangsfensters, d.h. LAS $<$ SEQ $\leq$ LAS $+$ RWS, dann wird diese Nachricht akzeptiert und zwischengespeichert, wenn eine Nachricht mit dieser SEQ noch nicht empfangen wurde. Anderenfalls wird diese Nachricht verworfen.
2. Ist die Sequenznummer der gerade zwischengespeicherten Nachricht kleiner als FMN, dann wird FMN zu SEQ gesetzt.
3. Ist die Bedingung FMN = LAS + 1 erfüllt, dann wird die größte Sequenznummer NMA (next message to acknowledge) der empfangenen Nachrichten innerhalb des Sliding-Window gesucht, sodass sich keine nicht empfangenen Nachrichten zwischen NMA und FMN befinden. Es wird LAS auf NMA gesetzt und es wird eine Bestätigungsnachricht mit NMA an den Sender gesendet. Damit bestätigt der Empfänger alle Nachrichten inkl. derjenigen mit der Sequenznummer NMA. Danach wird noch FMN neu gesetzt.

Varianten

Es gibt Varianten zu dem gerade beschriebenen Algorithmus:

- Es besteht die Möglichkeit jede empfangene Nachricht, die sich innerhalb des Empfangsfensters befindet sofort zu bestätigen. Diese Möglichkeit nennt man *selektives* Bestätigen im Gegensatz zur gerade beschriebenen Methode der *kumulativen* Bestätigung.
- Im Gegensatz zu den positiven Bestätigungen kann der Empfänger auch negative Bestätigungen (engl. negative acknowledgments, NAK) senden, die anzeigen, dass eine Nachricht nicht erhalten wurde.

Beiden Varianten ist gemein, dass sie die Komplexität der Software vergrößern und in gewissen Maße auch zu einem erhöhten Nachrichtenverkehr führen. Allerdings kann der Sender schneller auf Fehler reagieren und unter Umständen den verfügbaren Übertragungskanal besser ausnutzen.

Endliche Sequenznummern

Nicht betrachtet wurde bei dem beschriebenen Algorithmus allerdings, dass die Sequenznummern eine maximale Größe haben, die durch die Implementierung festgelegt ist. D.h. die Sequenznummern können nicht unendlich wachsen: sie sind endlich.

Betrachten wir zuerst den Stop-And-Go-Algorithmus mit einer 1 Bit langen Sequenznummer. Damit gibt es 2 verschiedene Sequenznummern, aber nur eine Nachricht darf ausständig sein.

Vergrößert man den Speicherplatz für die Sequenznummern auf 3 Bit, ergibt sich ein Bereich für die Sequenznummern von 0 bis 7 (= SEQ_{max}). Nehmen wir an, dass SWS = RWS = 7 ist und der Sender Nachrichten mit den Sequenznummern 0 bis 6 sendet. Der Empfänger empfängt diese und sendet ACKs, die jedoch verloren gehen. D.h. aus der Sicht des Empfängers sieht es so aus, dass als nächstes die Nachricht mit der Sequenznummer 7 kommen muss und danach wieder Sequenznummern mit 0 beginnend. Beim Sender laufen jedoch die Timer ab, sodass dieser die alten Nachrichten mit den Sequenznummern 0 bis 6 nochmals sendet, die vom Empfänger jedoch als neue Nachrichten interpretiert werden würden.

Die Anzahl der möglichen Sequenznummern muss größer sein als die der maximalen Anzahl der ausstehenden Frames. Es muss gelten, dass: SWS + RWS $<$ SEQ_{max} + 1. Bei SWS = RWS folgt damit, dass SWS $<$ (SEQ_{max} + 1) / 2 sein muss. D.h. analog zum Stop-And-Go Algorithmus, der jeweils zwischen den Sequenznummern 0 und 1 wechselt, wird hier – allerdings kontinuierlich – zwischen den beiden Hälften der Sequenznummern hin- und herbewegt.

4. Netzarchitektur

Der vorhergehende Abschnitt hat die Kommunikation zwischen zwei Kommunikationspartnern behandelt. Ein Netz besteht im allgemeinen jedoch aus mehreren Kommunikationspartnern, die miteinander verbunden sind und miteinander kommunizieren können. Damit wird aus einer Punkt-zu-Punkt Verbindung ein Datennetz.

Ein Datennetz (engl. data network) überträgt digitale Signale zwischen Paaren oder Gruppen von Kommunikationspartnern. Dabei können Kommunikationsbeziehungen wahlfrei zwischen beliebigen Kommunikationspartnern hergestellt werden.

Das Modell der Datenübertragung, wie es im Abschnitt 3.1 besprochen wurde, wird zu einem Datennetz:

Abbildung 4.1.: Datennetz

Eine Punkt-zu-Punkt Verbindung wird von nun an als eine Spezialform eines Datennetzes verstanden. Handelt es sich bei den DEEs um autonome Rechner oder rechnerartige Geräte (wie z.B. Drucker, Speichersysteme,...) dann sprechen wir von nun an von einem Rechnernetz (engl. computer network). Datennetze, Rechnernetze oder eben nur Netze werden, wie im vorhergehenden Diagramm, mit einer „Wolke" als Symbol gekennzeichnet. Die Kommunikationsteilnehmer in einem Rechnernetz werden als Host bezeichnet.

Solch ein Netz kann entweder direkt mehrere Netzteilnehmer miteinander verbinden oder durch einen Zusammenschluss von zwei oder mehreren Netzen entstehen:

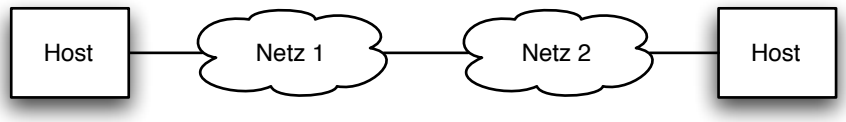


Abbildung 4.2.: Verbundene Rechnernetze

In der vorhergehenden Abbildung wurden zum Zwecke der Übersichtlichkeit vorerst die Netzgeräte außer Acht gelassen, die man für den Zusammenschluss mehrerer Teilnetze zu einem Gesamtnetz benötigt. In Abschnitt 4.1.2 auf Seite 54 werden diese erläutert. Fasst man zwei oder beliebig viele Netze zu einem Netz zusammen, dann bezeichnet man die einzelnen Netze des entstehenden Gesamtnetzes als Teilnetze eben dieses ganzen Netzes.

In diesem Kapitel wird im folgenden der prinzipielle Aufbau und die Funktionsweise eines Netzes erklärt.

4.1. Struktur und Komponenten eines Netzes

Ein Netz besteht aus einem oder mehreren Netzsegmenten. Mehrere Netzsegmente sind durch Kopplungsgeräte (siehe Abschnitt 4.1.2) miteinander verbunden. Netzsegmente sind daher eine Strukturierung für Rechnernetze auf der phyischen Ebene.

Die logische Struktur eines Netzwerkes ist durch die Netztopologie vorgegeben.

4.1.1. Netztopologie

Wir definieren, dass es sich bei einem Netz (engl. network) um eine Menge von Knoten (engl. node) handelt. Je zwei Knoten werden durch eine Teilstrecke oder Verbindung (engl. link) miteinander verbunden. Eine Art von Knoten sind die Hosts, eine andere Art sind die Kopplungselemente, die lediglich Netzsegmente miteinander verbinden. Jetzt betrachten wir, in welcher Art diese Knoten miteinander zu einem Netz verbunden werden können. Eine bestimmte Struktur des Verbindens bezeichnet man als Topologie. Allerdings sollte beachtet werden, dass die physikalische von der logischen Topologie abweichen kann ($\rightarrow$ Abstraktion).

Vermaschtes Netz

Das vermaschte Netz ist eine logische Weiterführung der Punkt-zu-Punkt Verbindung indem bestimmte (beliebige) Hosts direkt miteinander verbunden werden. Ein Beispiel für ein solches vermaschtes Netz ist in Abbildung 4.3 zu sehen.

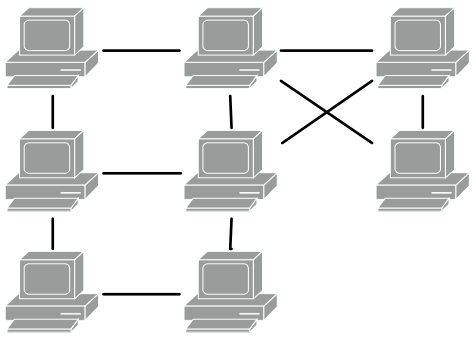


Abbildung 4.3.: Vermaschtes Netz

Wollen nur die Hosts miteinander kommunizieren, die direkt miteinander verbunden sind, ergibt sich eigentlich kein Unterschied zu mehreren Punkt-zu-Punkt Verbindungen. Sollen die anderen Hosts auch miteinander kommunizieren, dann muss ein Kommunikationspfad gefunden werden, der diese Hosts (indirekt) miteinander verbindet und die Hosts auf diesem Kommunikationspfad müssen die Nachrichten weiterreichen. Fällt ein Host oder eine

Verbindungsleitung auf einem Kommunikationspfad aus, dann gibt es je nach Struktur des Netzes gegebenfalls alternative Pfade, die gewählt werden können.

D.h. es gibt prinzipiell keine zugrundeliegende Struktur, die Verbindungen werden nach gewissen Gesichtspunkten wie z.B. Leistung oder Ausfallsicherheit gewählt.

Als Vorteile dieser Struktur können die Ausfallsicherheit und die Leistung hervorgehoben werden, als Nachteile fallen die hohen Verkabelungskosten, Wartungskosten und ein vergleichsweises kompliziertes Finden der Kommunikationspfade an.

Sternnetz

Will man ein Netz einfach strukturieren, dann bietet sich eine Sternstruktur an. Dieses war entwicklungsgeschichtlich die erste Struktur. Es wird ein Knoten ausgezeichnet, der (hauptsächlich) eine Vermittlungsfunktion übernimmt und zu jedem Host im Netzwerk eine Verbindungsleitung führt. Dieser Knoten wird allgemein als Vermittlungsknoten bezeichnet (siehe Abbildung 4.4).

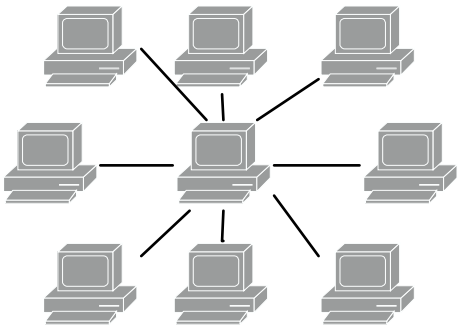


Abbildung 4.4.: Sternnetz

Die Vorteile in einer derartigen Struktur liegen in der einfachen Verkabelung, der einfachen Wartbarkeit und der leichten Erweiterbarkeit. Ein Ausfall eines Hosts hat auf die Kommunikation der anderen Hosts keine Auswirkung. Nachteilig wirkt sich die zentrale Funktion des Vermittlungsknotens aus: Fällt dieser aus, dann kann keine weitere Kommunikation im Netzwerk stattfinden. Bezüglich Leistung und Erweiterbarkeit lässt sich sagen, dass diese lediglich von dem Vermittlungsknoten (und natürlich den Übertragungskanälen) abhängt. Nachteilig wirkt sich auch aus, dass die Verkabelung sehr teuer ist.

Baumnetz

Werden Sternnetze hierarchisch miteinander verbunden, dann entsteht ein Baumnetz (siehe Abbildung 4.5 auf der nächsten Seite). Im englischen Sprachgebrauch werden diese auch als „extended star" bezeichnet. Verbunden werden die einzelnen Vermittlungshosts mittels sogenannter Uplinks.

Die Vorteile dieser Topologie liegen in der leichten Erweiterbarkeit und der Administrierbarkeit (Teilung der Verantwortung).

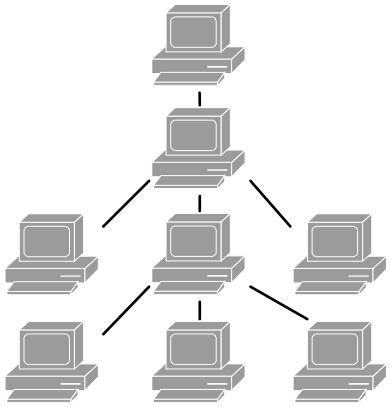


Abbildung 4.5.: Baumnetz

Ringnetz

Es wird jeder Host mit genau zwei anderen Hosts so verbunden, sodass ein geschlossener Ring entsteht (siehe Abbildung 4.6). Der eine verbundene Host wird als Vorgänger definiert und der andere als Nachfolger. D.h. prinzipiell ist eine Richtung in diesem Ring vorgegeben. Der Sender sendet die Nachricht an seinen Nachfolger. Ist die Nachricht beim Empfänger angekommen, dann nimmt dieser die Nachricht vom Netz, andererseits wird die Nachricht wieder an den Nachfolger des aktuellen Knotens weitergereicht. Das wird solange durchgeführt bis der Empfänger erreicht ist.

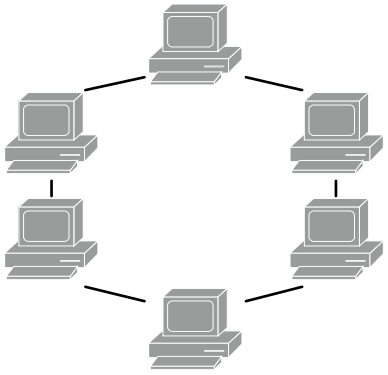


Abbildung 4.6.: Ringnetz

Die Vorteile dieser Topologie liegen darin, dass jeder Knoten eine Signalauffrischung durchführt und dadurch eine große Netzausdehnungen möglich ist, dass kein Flaschenhals wie beim Sternnetz vorliegt und die eigentliche Datenübertragung sehr einfach ist, da die Übertragung nur in einer Richtung stattfindet. An Nachteilen sind zu nennen, dass ein Ausfall eines Knotens oder einer Verbindung den gesamten Ring beeinträchtigt und die Nachrichten alle Knoten am Pfad zum Empfänger durchlaufen müssen.

Busnetz

Ein Busnetz ist grundsätzlich anders: Hier gibt es einen Bus an den alle Knoten direkt angeschlossen sind (siehe Abbildung 4.7). D.h. es muss einen Mechanismus geben, der den Zugriff auf das gemeinsame Medium regelt, wenn mehrere Hosts gleichzeitig zugreifen wollen.

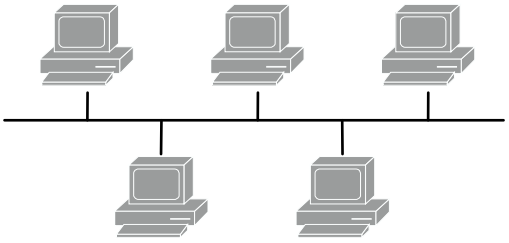


Abbildung 4.7.: Busnetz

Große Vorteile liegen in der Verkabelung, der Erweiterbarkeit und den Kosten. An Nachteilen sind zu nennen:

- Fehler finden ist schwieriger als bei den anderen Topologien.
- Ein Fehler in der Verkabelung bedeutet, dass das gesamte Netzwerk nicht mehr funktioniert.
- Die Leistung des Netzwerks sinkt bei mehreren gleichzeitigen Zugriffen.
- Jeder Host kann den gesamten Netzwerksverkehr lesen. Das kann unter Umständen ein Sicherheitsproblem darstellen.

4.1.2. Netzgeräte

Ein Rechnernetz besteht also aus Knoten und Verbindungen:

- Knoten sind entweder Hosts (d.h. Endgeräte) oder Kopplungselemente (z.B. ein Router), die Teilnetze miteinander verbinden.
- Die eigentlichen Verbindungen werden mittels Kabel, Stecker und Netzzugangsgeräten (z.B. eine Schnittstellenkarte) realisiert.

Alle diese in einem Rechnernetz konkret vorkommenden Geräte werden als Netzgeräte bezeichnet.

Zugriffsverfahren

Ein Netzzugangssgerät speist die Signale in das Übertragungsmedium ein. Da mindestens 2 Kommunikationspartner auf das Übertragungsmedium zugreifen wollen, muss es ein Verfahren geben, das diesen Zugriff regelt. Man unterscheidet:

Reservierungsverfahren Bei den Reservierungsverfahren wird zu Beginn eine Reservierung vereinbart. Die Multiplexverfahren sind solche Verfahren.

Zuteilungsverfahren Der Zugriff auf das Übertragungsmedium wird zugeteilt, d.h. der Zugriff findet geregelt statt. Diese Erteilung des Zugriffes kann entweder zentral (z.B. mittels polling) oder dezentral (z.B. mittels token passing) erfolgen.

Wettbewerbsverfahren Diese funktionieren im Prinzip so, dass jeder Sender versucht auf den Kanal zu senden, wenn dieser frei ist. Der Sender, der einen freien Kanal vorfindet, beginnt zu senden. Vertreter sind CSMA/CD (carrier sense multiple access/collision detection) oder CSMA/CA (carrier sense multiple access/collision avoidance).

Unter dem Begriff *Kollisionsdomäne* (engl. collision domain) wird in einem Computernetz ein Bereich bezeichnet, in dem Kollisionen auftreten können. Sie entstehen, wenn zwei Stationen gleichzeitig versuchen, auf einem einzigen physikalischen Medium (Segment) etwas zu senden. Die Spannungsimpulse werden im Kabel vermischt und die Signale somit zerstört.

Eine *Broadcastdomäne* ist ein logischer Verbund von Computern in einem lokalen Netzwerk, der sich dadurch auszeichnet, dass ein Broadcast alle Domänenteilnehmer erreicht.

Netzzugangsgeräte

Im Prinzip kann man zwei Arten von Netzzugangsgeräten unterscheiden, je nachdem, ob es sich um einen Breitbandzugang oder einen Basisbandzugang handelt.

Modem Ein Modem moduliert und demoduliert ein analoges Signal, um digitale Daten übertragen zu können. D.h. die digitalen Daten werden z.B. auf ein hochfrequentes Trägersignal aufmoduliert. Ursprünglich wurden Modems hauptsächlich verwendet, um Daten über normale Telefonleitungen übertragen zu können. Danach wurde ISDN eingeführt und mittels ISDN Modems (PCM, pulse code modulation) auf das ISDN Netzwerk zugegriffen. Heute gibt es auch Kabelmodems bzw. ADSL Modems.

Der Zugriff von einem Host zu einem Modem kann über eine der verfügbaren Schnittstellen wie z.B. RS-232 oder USB erfolgen.

Schnittstellenkarte Eine Schnittstellenkarte (engl. network card oder network interface card, kurz NIC) oder auch Netzwerkadapter (engl. network adapter) genannt, ist im Sinne der Datenübertragung eine Datenübertragungseinrichtung, die es dem Computer erlaubt über das Netz zu kommunizieren.

Für jede NIC gibt es eine MAC Adresse und eine eigene Netzwerkadresse.

Kopplungselemente

Netze können auf der physikalischen Ebene mittels Kopplungselementen verbunden werden. Die mit solchen Kopplungselementen verbundenen Teilnetze werden als Netzsegmente bezeichnet. Der Anschluss eines Kopplungselementes für ein Netzwerksegment wird auch Port genannt.

Repeater Ein Repeater (dt. Wiederholer) hat die Aufgabe Signale zu empfangen, diese zu regenerieren (verstärken, verbessern) und danach weiterzusenden. Repeater verbinden zwei (oder mehrere) Netzwerksegmente gleichen Typs zu einem größeren Netzsegment.

Die primäre Aufgabe von Repeatern ist, die physikalische Ausdehnung des Netzwerkes zu erhöhen. Repeater arbeiten auf der ISO/OSI Schicht 1.

Ein Netzsegment, das auf diese Weise mittels Repeater aus mehreren Netzsegmenten entstanden ist, verhält sich als Kollisionsdomäne.

Hub Ein Hub (dt. zentraler Knoten, Mittelpunkt, Drehscheibe) ist lediglich ein Repeater, der mehrere Netzsegmente miteinander verbindet (d.h. diese haben mehrere Ports). Hubs werden auch Multiport-Repeater genannt. Dieser Begriff wird häufig im Ethernet verwendet.

Bridge Eine Bridge (dt. Brücke) verbindet zwei Netzwerkssegmente auf der ISO/OSI Schicht 2 zu einem größeren Netz. Anders gesehen teilt eine Brücke ein Netz in zwei Netzsegmente. Haben die Netzsegmente unterschiedliche Zugriffsverfahren, werden diese von der Bridge angepasst.

Eine Bridge gibt nur Rahmen weiter, deren MAC Adresse in dem entsprechenden Netzsegment liegt. Bridges trennen Netzsegmente daher sowohl physisch als auch logisch. Damit endet eine Kollisionsdomäne an einer Bridge.

Switch Ein Switch (dt. Schalter) verbindet mehrere Netzwerksegmente auf der ISO/OSI Schicht 2. Ein Switch arbeitet prinzipiell in einem von zwei Modi: Ein *Store-and-Forward* Switch nimmt immer einen ganzen Frame auf, speichert diesen zwischen, analysiert diesen und gibt diesen am richtigen Zielport wieder aus. Ein *Cut-Through* Switch nutzt die Tatsache, dass sich in einem Ethernet Frame die Zieladresse am Anfang befindet und leitet den Frame zum Zielport weiter, nachdem der lediglich der Frame bis inkl. der Zieladresse gelesen wurde. Der Vorteil eines Store-and-Forward Switch ist, dass dieser nur gültige Frames weiterreicht, während ein Cut-Through Switch eine höhere Performance aufweist dafür aber auch fehlerhafte Frames weiterleitet.

Die Methode wie ein Switch den richtigen Zielport für einen Frame kennen lernt, wird als Backward-Learning bezeichnet: Jedes Mal, wenn ein Switch einen Frame an einem Port erhält, merkt sich der Switch die Quell-MAC (siehe Abschnitt 4.2.1 auf Seite 60) gemeinsam mit dem Port an dem der Frame empfangen wurde. Kennt der Switch die die Ziel-MAC nicht, wird der Frame an allen anderen Ports weitergeleitet. Diesen Vorgang nennt man Fluten (port flooding). Auf diese Weise erreicht der Frame sein Ziel. Sendet der Empfänger einen Frame zurück an den ursprünglichen Sender, dann merken sich die Switches auch dessen MAC, wodurch bei wiederholten Senden zwischen diesen beiden Hosts kein Fluten mehr notwendig ist.

Router Ein Router (dt. Vermittler) verbindet mehrere Netze miteinander und hat die Aufgabe eine empfangene Nachricht an das richtige Netz weiterzuleiten, sodass der eigentliche Empfänger letztendlich die Nachricht erhält. Router müssen dazu den Netzaufbau in gewisser Weise kennen.

An einem Router endet sowohl eine Kollisionsdomäne als auch eine Broadcastdomäne.

Ein Router arbeitet auf der ISO/OSI Schicht 3.

Gateway Das Gateway verbindet ebenfalls mehrere Netze miteinander, unterscheidet sich von einem Router jedoch dadurch, dass ein Gateway Netze mit verschiedenen Anwen-

dungsprotokollen verbindet. D.h. ein Gateway übersetzt ein Anwendungsprotokoll in ein anderes Anwendungsprotokoll. Ein einfaches Beispiel für ein Gateway ist ein http Proxy.

Gateways arbeiten auf der ISO/OSI Schicht 7.

Die folgende Abbildung gibt einen Überblick über die verschiedenen Arten der Kopplungssysteme im Zusammenhang mit dem OSI Modell.

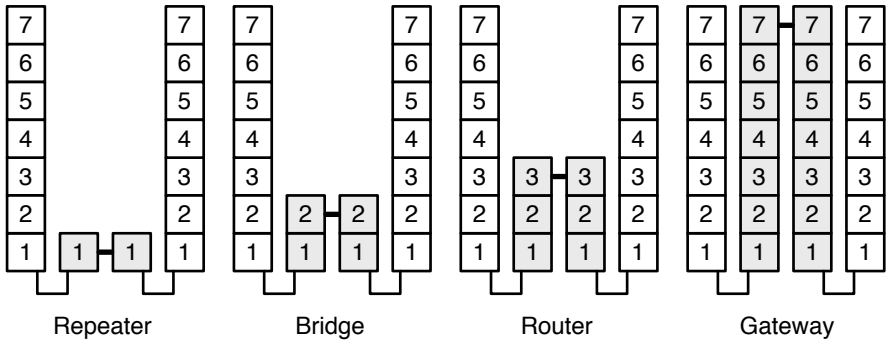


Abbildung 4.8.: Übersicht über die Kopplungssysteme

4.1.3. Netzstrukturen

Netze können – unabhängig von ihren – Topologien beliebig miteinander verbunden werden.

Netzhierarchie

Größere Netze werden hierarchisch in Schichten aufgebaut.

- Auf der untersten Ebene befinden sich die *lokalen Netzwerke*, die Netzgeräte innerhalb einer begrenzten Umgebung miteinander verbinden. Solche Netze können natürlich auch wieder miteinander verbunden sein. Ein typischer Vertreter einer solchen Netztechnologie ist Ethernet.
- Um einen Zugang zu weiter entfernten Hosts zu bekommen, werden die lokalen Netze mittels *Zugangsnetze* (engl. access networks) an ein übergeordnetes Netzwerk verbunden. Beispielsweise kann ein Einzelcomputer mittels ADSL als Zugangsnetz an das Netz des Providers angeschlossen werden. Beispiele für weitere Technologien sind ISDN, DSL Varianten oder Richtfunkstrecken.
  Im Bereich der Zugangsnetze wird oft PPP (Point-to-Point Protocol) verwendet, das eine Punkt-zu-Punkt Verbindung herstellt und verschiedenste Netzprotokolle (wie z.B. IP, IPX, AppleTalk) transportieren kann. Für den Betrieb von PPP über Ethernet gibt es die modifizierte Variante PPPoE (Point-to-Point over Ethernet). Auch das PPTP (siehe Abschnitt 4.1.3 auf Seite 58) wird oft für Zugangsnetze verwendet.

- Über das Zugangsnetz werden die Hosts an ein *Verteilungsnetz* (engl. distribution network) angeschlossen. Dabei handelt es sich z.B. um ein Firmennetz, das mittels eines Backbones mehrere Firmenstandorte verbindet oder um das Netz eines Providers. Das Verteilungsnetz grenzt Zugangsnetze von einem Kern-Netzwerk ab (z.B. dem globalen Internet). In einem Verteilungsnetz werden auch noch Funktionen wie Adressumsetzung (engl. network address translation, NAT), Sicherheitsüberprüfungen oder VLAN Routing implementiert. Als Technologien kommen z.B. FDDI, ATM oder auch Gigabit-Ethernet zum Einsatz.
- Ein Kernnetz (engl. core network) dient der weltweiten Vernetzung. Als Technologien werden Standleitungen, ATM und auch Frame Relay eingesetzt.

Ein Beispiel für eine derartige Netzhierarchie ist in der Abbildung 4.9 zu finden.

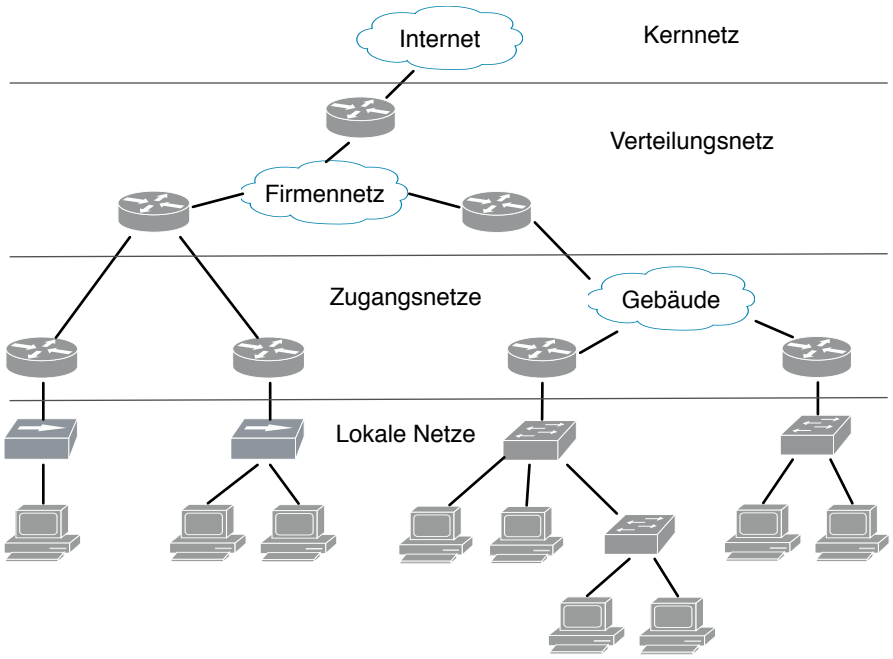


Abbildung 4.9.: Beispiel einer Netzhierarchie

Strukturierte Verkabelung

Eine konkrete Ausprägung einer Netzhierarchie ist die Verkabelung eines Standortes. Die europäische Norm EN 50173-1 ist unter dem Begriff *strukturierte Verkabelung* bekannt. Meist wird hierfür als Topologie eine Baumtopologie verwendet.

Dazu wird die Verkabelung in Primärbereich, Sekundärbereich und Tertiärbereich eingeteilt.

Der Primärbereich deckt die Verbindung der einzelnen Gebäude untereinander ab. Dazu gibt es je Gebäude einen Hauptverteiler, die direkt mit den anderen Gebäuden verbunden sind. Der Sekundärbereich übernimmt die (vertikale) Stockwerksverkabelung und verbindet hiermit die einzelnen Etagenverteiler mittels Stichleitungen mit dem Hauptverteiler. Der Tertiärbereich stellt die horizontale Stockwerksverkabelung dar und verbindet die einzelnen Arbeitsstationen mittels Stichleitungen mit dem Etagenverteiler.

Um flexibel zu sein, werden die Verteiler mittels Patchfelder (engl. patch panel) realisiert. Hierunter versteht man ein Schaltfeld, das alle Leitungen an Buchsen führt. Die Verbindungen werden mittels Patchkabel hergestellt.

VLAN

Ein VLAN (engl. virtual local area network) ist ein virtuelles lokales Netzwerk innerhalb eines LAN. Mit VLANs können mehrere logische Netze innerhalb eines physisches LAN betrieben werden. Sie dienen also dazu eine weitere Struktur in ein physisches Netz zu legen.

VLANs will man im Prinzip aus folgenden Gründen einrichten:

- Flexibilität beim Aufbau der Netze, da die logischen Netze unabhängig zum physischen Aufbau des LAN ist.
- Einsparung von Routern und damit eine vereinfachte Verwaltung sowie eine Verringerung der Latenz.
- Verringerung der Broadcast-Last, denn jedes VLAN bildet eine eigene Broadcast-Domäne.
- Steigerung der Sicherheit, denn es kann nicht auf die Daten eines anderen Netzes zugegriffen werden.

Die standardisierte Grundlage für den Betrieb von VLANs liegt in IEEE 802.1Q begründet (es gibt jedoch auch herstellerspezifische Implementierungen). Die Implementierung von VLANs basiert im Prinzip darauf, dass dem Ethernet-Frame 4 Bytes an zusätzlicher Headerinformation hinzugefügt wird. Von diesen 4 Bytes werden 12 Bit zur Aufnahme einer VLAN ID verwendet. Daraus resultiert eine maximale Anzahl von 4094 VLANs, da die IDs mit lauter Nullen bzw. lauter Einsen reserviert sind und nicht zur Verfügung stehen.

VPN

Ein VPN (engl. virtual private network) ist ein Netz von zwei oder mehreren Rechnern, die über Tunnels (Übertragen der Daten eines Protokolls eingebettet in ein anderes Protokolls) verbunden sind. Ein VPN wird (meist) öffentlich verwendet und ist in der Regel durch kryptographische Verfahren gesichert. VPNs transportieren im Normalfall ISO/OSI Schicht 2 Frames oder Schicht 3 Pakete.

Als Anwendung wird ein VPN oft verwendet, um Außenstellen oder externen Mitarbeitern den Zugriff auf das Unternehmensnetzwerk zu ermöglichen. Wichtig ist, dass der Client ebenfalls richtig konfiguriert ist, sodass die gesamte Kommunikation des Clients nach dem Öffnen des VPN-Zugangs über den VPN-Pfad läuft!

Um ein vollständiges VPN zu bilden müssen drei Komponenten kombiniert werden: Tunnels, Verschlüsselung und Authentifizierung.

Einige „wichtige" Protokolle mit, denen ein VPN aufgebaut werden kann, sind:

PPTP (point-to-point tunneling protocol) ermöglicht das Tunneling von PPP durch ein IP Netzwerk. Es wird oft als Zugangsnetz für die Anbindung von privaten Nutzern an die Internet Service Provider verwendet, obwohl es relativ unsicher ist. Eine Verschlüsselung ist im Prinzip in diesem Standard vorgesehen und muss durch höhere Protokolle realisiert werden (abgesehen von der Microsoft-spezifischen Erweiterung MPPE, die PPP Pakete verschlüsselt). Der „einzige" Vorteil von PPTP ist, dass es in Windows vorinstallliert ist.

L2TP (layer 2 tunneling protocol) ist in etwa von der Funktionalität dem PPTP ähnlich und ebenfalls relativ unsicher. Erst die Kombination mit IPSec ermöglicht auch eine Verschlüsselung.

IPSec IPSec (siehe Abschnitt **??** auf Seite ??) ist eine Gruppe von Protokollen zur Verschlüsselung. Es befindet sich auf der Schicht 3 und kann Protokolle ab (inklusive) Schicht 3 transportieren und sichern. Es wurde ursprünglich für IPv4 entwickelt und als Kernkomponente bzgl. Sicherheit in IPv6 eingegliedert. Es gilt als ausreichend sicher, wenn es richtig konfiguriert wird. Jedes Betriebssystem benötigt seine eigene Implementierung im TCP/IP Stack.

TLS/SSL Mittels TLS/SSL (siehe Abschnitt **??** auf Seite ??) kann ebenfalls ein VPN aufgebaut werden (siehe das Produkt OpenVPN im Abschnitt **??** auf Seite ??).

Mit TLS kann auch ein einfacher Tunnel aufgebaut werden. Die Vor- und Nachteile sind analog zu der Technik mit SSH wie im nächsten Punkt beschrieben. Als Produkt gibt es z.B. Stunnel (siehe Abschnitt **??** auf Seite ??), das es einfach erlaubt verschiedene Protokolle über TLS zu tunneln.

SSH Auch mittels SSH (secure shell, siehe Abschnitt **??** auf Seite ??) kann ein Tunnel aufgebaut werden. Allerdings ist SSH nicht für eine permanente LAN-zu-LAN und Client-zu-LAN Verbindungen geeignet, da mittels SSH nur Socket-Tunnels möglich sind. Allerdings entsteht gerade dadurch ein Vorteil, da die Tunnels je Applikation spezifisch sind. Ein kompromittierter Tunneleingang einer SSH Verbindung betrifft nur die Applikation bzw. den Port am Tunnelende und nicht das gesamte Netz, das mit dem Tunnelende verbunden ist!

4.1.4. Netzsoftware

In einem verteilten System kann man die Software prinzipiell in drei Klassen einteilen:

- Die verteilte Anwendung an sich. D.h. es handelt sich um die Software, die oberhalb der Schicht 7 angesiedelt ist.
- Meistens benötigt die Anwendung noch unterstützende Software, die bestimmte Dienste zur Kommunikation, Synchronisierung, Transaktionsverwaltung,... zur Verfügung stellt. Diese Software ist meistens zwischen der Schicht 7 und der eigentlichen Anwendung (also in der Mitte) angesiedelt. Diese Software wird aus diesem Grund auch Middleware genannt.
- Unbedingt erforderlich ist die eigentliche Netzsoftware, die die gesamte Kommunikation ermöglicht. Wie schon erläutert, handelt es sich meistens um eine Implementierung eines Protokollstacks (z.B. ISO/OSI oder am häufigsten TCP/IP).

4.2. Vermittlung und Weiterleitung

In diesem Abschnitt wird zuerst die Adressierung der Knoten besprochen, dann die Prinzipien der Paketvermittlung behandelt und danach der grundlegende Ablauf des Weiterleitens angeführt als auch die Suche eines Weges zwischen je zwei Knoten behandelt. Alle diese Themen sind der Schicht 3 des OSI Modells zuzuordnen.

4.2.1. Adressierung

Adressen

Jeder Knoten benötigt im Netz bzw. im Netzsegment eine eindeutige Adresse. Solch eine Adresse ist in der Regel ein nummerischer Wert. Adressen sind abhängig von der Schicht in der sie verwendet werden:

- In der Schicht 2 werden diese Adressen als MAC Adressen bezeichnet
- Im Internet sind die Adressen in der Schicht 3 die sogenannten IP Adressen.
- In der Schicht 4 wird im Internet als Adresse die Kombination aus IP Adresse, Port und Protokoll verwendet.

Eine Adresse in der Schicht 2 muss nicht unbedingt global eindeutig sein. Es reicht, dass so eine Adresse nur innerhalb eines Teilnetzes eindeutig ist. So eine Adresse nennt man auch lokale Adresse im Gegensatz zu einer globalen Adresse, die im gesamten Netz eindeutig sein muss (z.B. eine IP Adresse).

Teilweise muss beim Übergang zwischen den Schichten auch zwischen den Adressen gewandelt werden, z.B. muss eine IP Adresse in eine MAC Adresse aufgelöst werden. Dieser Vorgang wird Adressauflösung (engl. address resolution) genannt.

Es können verschiedene Typen von Adressen unterschieden werden:

- Die Individualadresse identifiziert genau einen Knoten.
- Eine Gruppenadresse identifiziert eine Gruppe von Knoten. Diese Knoten können sich auch in verschiedenen Netzen befinden. Es handelt sich um eine Multicast-Adresse.
- Eine Broadcast-Adresse ist eine spezielle Adresse, die alle Knoten in einem Netz identifiziert.

Adressen können entweder flach sein oder hierarchisch aufgebaut sein. Bei einer flachen Adresse gibt es keinen Zusammenhang zwischen der Adresse und der geographischen Lage des Knotens, dem diese Adresse zugewiesen wird. Bei hierarchischen Adressen gibt es solch einen Zusammenhang. Beispiel eines flachen Adressraumes sind die MAC Adressen der Ethernet-Technologie und ein Beispiel hierarchischer Adressen sind die IP-Adressen.

In weiterer Folge betrachten wir die hierarchisch aufgebauten IP Adressen. Jede IP Adresse besteht aus zwei Teilen: einem Netzanteil und einem Hostanteil. Der Netzanteil gibt das Teilnetz an, in dem der Host zu finden ist und der Hostanteil ist die Adresse des Host in diesem Teilnetz. Die Teilnetzangabe ist einer geographischen Lage zugeordnet.

Namen

Im Gegensatz zu den Adressen sind die Namen symbolische Werte, die an der Stelle der (nummerischen) Adressen verwendet werden. Da für die eindeutige Adressierung jedoch die Adresse notwendig ist, werden Namensdienste bzw. Verzeichnisdienste benötigt, die einen Namen auf eine Adresse abbilden.

Der Grund in der Verwendung von Namen liegt entweder in der leichteren Merkbarkeit oder in der gewünschten Einführung einer Indirektion. Eine Indirektion kann z.B. notwendig sein, um Ortstransparenz zu erreichen. Damit ist es möglich einen Server eines Dienstes in ein anderes Netz zu verlegen, d.h. die Adresse zu ändern.

Label

Bei verbindungsorientierten Netzen reicht es, wenn die Kopplungselemente für jede ihrer Verbindungen zu den nächsten Knoten jeweils einen eindeutigen lokalen Bezeichner vergeben. Solch ein Bezeichner wird Label genannt. Eine genauere Beschreibung ist in Abschnitt 4.2.2 auf der nächsten Seite zu finden.

4.2.2. Paketvermittlung

In diesem und den weiteren Abschnitten wird von nun an nur mehr die Paketvermittlung (engl. packet switching) behandelt. Dazu werden die Nutzdaten in Pakete (engl. packet) aufgeteilt und zusammen mit anderen Paketen im asynchronen Zeitmultiplexverfahren über Teilstrecken übertragen. Teilstrecken sind über Knoten miteinander verbunden, die als Router bezeichnet werden.

Der grundlegende Ablauf in einem Router sieht so aus, dass ankommende Pakete zwischengespeichert werden (store), dann der nächste Router ermittelt wird und das Paket an die richtige Ausgabeschnittstelle weitergeleitet (engl. forwarding) wird.

Da das Speichern des Paketes, das Analysieren des Headers (z.B. Berechnen einer Prüfsumme) und unter Umständen das Verändern des Headers (z.B. siehe TTL im Abschnitt 8) relativ lange dauert, hat man Spezialformen der Paketvermittlung geschaffen, die diese Nachteile vermeiden. Einerseits handelt es sich um das Konzept Frame Relay (Pakete mit variabler Länge) und Cell Relay (Paket mit fixer Länge).

Im Prinzip gibt es 3 Möglichkeiten: verbindungsloses Weiterleiten, verbindungsorientiertes Weiterleiten und Source-Routing.

Verbindungsloses Weiterleiten

Das Prinzip ist, dass ein Paket genügend Informationen enthält, damit jedes Kopplungselement weiß wie das Paket zum Ziel weiterzuleiten ist. Dafür ist zumindest die Netzadresse des Ziels notwendig.

Das verbindungslose Weiterleiten weist folgende Merkmale auf:

- Ein Knoten kann jederzeit ein Paket zu jedem beliebigen anderen Knoten schicken.

- Es besteht für den Knoten keine Möglichkeit zu eruieren, ob das Paket am Zielknoten ankommt bzw. ob der Zielknoten überhaupt erreichbar ist.
- Jede Nachricht wird unabhängig von den anderen Paketen gesendet. Obwohl zwei Pakete unter Umständen dasselbe Ziel haben, können die Pakete unterschiedliche Wege im Netz nehmen, nicht in der Reihenfolge des Sendens ankommen oder auch mehrfach das Ziel erreichen.
- Der Ausfall eines Kopplungselementes kann unter Umständen vom Netz kompensiert werden, indem ein Paket einen anderen Weg zum Ziel nimmt.

Verbindungsorientiertes Weiterleiten

Beim verbindungsorientiertem Weiterleiten wird zuerst eine virtuelle Verbindung aufgebaut. Es wird deshalb auch von virtueller Leitungsvermittlung gesprochen. Virtuell deshalb, weil eben keine echte Leitungen geschalten werden.

Im Prinzip gibt es zwei Möglichkeiten:

1. Das Konzept der Verbindung wird nur von den beiden Endsystemen umgesetzt. Die zwischenliegenden Knoten wissen von Verbindungen nichts und übertragen die Pakete mittels verbindungslosem Weiterleiten. Dies ist so in TCP umgesetzt (siehe Teil III). In diesem Sinne handelt es sich eigentlich gar nicht um eine virtuelle Leitungsvermittlung!
2. Das Konzept der Verbindung wird von den beiden Endsystemen *und* den Zwischensystemen umgesetzt. D.h. beim Verbindungsaufbau wird ein Weg zwischen den beiden Endsystemen gesucht und alle auf diesem Weg befindlichen Zwischensysteme tragen sich entsprechende Informationen in ihren Weiterleitungstabellen (siehe Abschnitt 4.2.3 auf der nächsten Seite) ein. Bei den Informationen, die eingetragen werden handelt es sich um sogenannte Labels (dt. Marken), die einen logischen Kanal zwischen zwei Knoten identifizieren. Pro Zwischenstation gibt es zwei Einträge. Ein Eintrag umfasst den Eingangsport und ein Label und ein Eintrag umfasst den Ausgangsport und ein Label. D.h. bei diesen Labels handelt es sich um lokal eindeutige Bezeichner, die zwischen zwei Knoten einen Teil der virtuellen Leitung identifiziert. Solch ein System wird z.B. bei X.25 verwendet. Beim Verbindungsabbau werden alle Einträge in den Zwischensystemen wieder entfernt.

Unter *Signalisierung* versteht man in diesem Zusammenhang den Austausch der Nachrichten, die zum Aufbau, der Überwachung und dem Abbau einer Verbindung notwendig sind. Einerseits gibt es die ältere *In-Band-Signalisierung* (engl. in-band signalling), die für diese Steuernachrichten den gleichen logischen Kanal verwendet wie die Nutzdaten und andererseits die modernere *Außer-Band-Signalisierung* (engl. out-of-band signalling), die für die Steuernachrichten einen getrennten logischen Kanal benutzt.

Source-Routing

Beim Source-Routing wird in jeder Nachricht gespeichert, welchen Weg diese Nachricht durch das Netz nehmen soll. Es handelt sich hierbei natürlich um eine beliebig lange Information, die in den Header der Nachricht eingesetzt werden muss. Dieses Verfahren setzt

allerdings voraus, dass der Sender genügend Information über die Netztopologie zur Verfügung hat, damit er diese Informationen in den Header einfügen kann.

Verwendet wird dieses Verfahren selten, da es nicht gut skaliert. Eine Anwendung ist: der Empfänger will, dass die Nachrichten einen bestimmten Weg nehmen sollen. Auf Grund von Sicherheitsüberlegungen werden solche Pakete von Firewalls meistens blockiert.

4.2.3. Weiterleitung in IP

Wir betrachten hier die Weiterleitung wie es im Prinzip in IP umgesetzt wird. Zum Weiterleiten enthält jeder Router eine Weiterleitungstabelle. Jeder Eintrag dieser Weiterleitungstabelle enthält einerseits eine Netznummer und andererseits eine Adresse eines Routers, der mit dieser Netznummer direkt oder indirekt verbunden ist. Der nächste Schritt in einem Weg zum Ziel wird auch als Hop bezeichnet.

Wie es zum Aufbau solch einer Weiterleitungstabelle kommt, ist die Aufgabe des Routing (siehe Abschnitt 4.2.4).

Die prinzipielle Vorgehensweise in einem Router beim Weiterleiten sieht folgendermaßen aus:

1. Paket zwischenspeichern.
2. Header kontrollieren (Struktur und Prüfsummen).
3. Zieladresse aus Header lesen.
4. Wenn Netznummer der Zieladresse gleich mit der Netznummer eines lokalen Netzes,
   a) dann: Paket an diesem Interface ausliefern, das zu diesem Netz führt.
   b) anderenfalls: Wenn Router für diese Netznummer der Zieladresse in Weiterleitungstabelle vorhanden,
      i. dann: Paket an den zugehörigen Router senden.
      ii. anderenfalls: Paket an den Default-Router senden.

4.2.4. Routing

Das Routing ist das Suchen von Wegen von einem Knoten des Netzwerks zu einem anderen Knoten. D.h. es ist ein Verfahren der Wegsuche und damit etwas ganz Anderes als das Weiterleiten, das davon ausgeht, dass ein Weg schon gefunden ist. Als Ergebnis des Routings ergibt sich eine Weiterleitungstabelle (auch Routingtabelle genannt), die direkt beim Weiterleiten verwendet wird. Achtung: Routing und Weiterleiten wird oft synomym verwendet.

Wie werden nun diese Weiterleitungstabellen erstellt? Dazu wird das Netz als ein Graph betrachtet, dessen Kanten mit den Kosten bezeichnet werden, die dieser Verbindung zugeordnet sind. Diese Kosten geben Aufschluss inwieweit es wünschenswert ist, Verkehr über diese Verbindung zu senden. D.h. je kleiner die Kosten sind, desto besser ist es diese Verbindung zu verwenden. Was diese Kosten wirklich bedeuten, wie man zu diesen Kosten kommt und inweit sich diese Kosten im Laufe der Zeit ändern können, werden wir nicht

im Detail betrachten. Beispielsweise könnte als Kosten die Übertragungsleistung (bzw. die inverse Übertragungsleistung) des Übertragungskanals herangezogen werden.

Wir gehen der Einfachheit halber davon aus, dass der Hin- und der Rückweg gleich gewichtet sind und verwenden daher nur eine ungerichtete Kante zwischen zwei Knoten anstatt zwei gerichteten Kanten.

Das einfachste Maß für die Kosten ist, einfach die Anzahl der Hops zu zählen. Damit sind die Kanten auch gleich gewichtet und damit können ungerichtete Kanten verwendet werden. Diese Art der Kosten wird auch in den kommenden Beispielen verwendet!

Für das eigentliche Erstellen dieser Weiterleitungstabellen gibt es mehrere Ansätze:

Statisches Verfahren Die Weiterleitungstabellen werden einmal erstellt. Damit ist dieses Verfahren aus der Sicht der Administration aufwändig und unflexibel. Problematischer sind außerdem zeitweilige Ausfälle von Verbindungen, da dafür kurzfristig Änderungen in allen Weiterleitungstabellen notwendig wären, die auf dieser Basis jedoch praktisch nicht durchgeführt werden können. Das statische Verfahren wird heutzutage nur noch für provisorische Zwecke sowie für kleine Netze angewendet, in denen sich die Topologie kaum ändert.

Isolierte Verfahren Bei den isolierten Verfahren verwendet jeder Knoten nur die ihm verfügbare, lokale Information für die Routing-Entscheidungen. Dazu zählen so einfache Verfahren wie das Fluten (engl. flooding), das einfach ein Paket an alle Schnittstellen weiterreicht. Ein weiteres Verfahren ist „Hot Potato" (dt. heiße Kartoffel), das ein Paket so schnell wie möglich weitergibt. Dazu wird das Paket an derjenigen Schnittstelle ausgegeben, die gerade die kürzeste Warteschlange hat.

Zentrale Verfahren Es gibt eine zentrale Stelle, die die Informationen der Weiterleitungstabellen an alle anderen Router verteilt. Diese zentrale Stelle muss die gesamte Netzstruktur kennen und muss entsprechend ausfallsicher sein. Unter Umständen kann sich diese zentrale Stelle als Flaschenhals erweisen.

Verteilte Verfahren Die verteilten Verfahren sind dadurch gekennzeichnet, dass jeder Router seine Routing-Entscheidungen selbstständig trifft. In TCP/IP haben sich zwei grundlegende Algorithmen für verteilte Verfahren etabliert: der Distanzvektor-Algorithmus und der Link-State-Algorithmus.

Zentrale und verteilte Verfahren werden im Gegensatz zu statischen Verfahren als adaptive Verfahren bezeichnet, da auf Änderungen der Netztopologie reagiert werden kann.

Als Beispiel für die beiden Algorithmen wird der folgende Graph verwendet:

Distanzvektor-Algorithmus

Beim Distanzvektor-Algorithmus (engl. distance vector) handelt es sich um einen Routing Algorithmus der nach folgendem Prinzip funktioniert: „Teile deinen Nachbarn mit, wie du die Welt siehst".

Dieser Algorithmus basiert darauf, dass jeder Knoten einen Vektor verwaltet, der die Entfernungen (engl. distance) oder allgemeiner die Gewichte zu allen übrigen Knoten enthält.

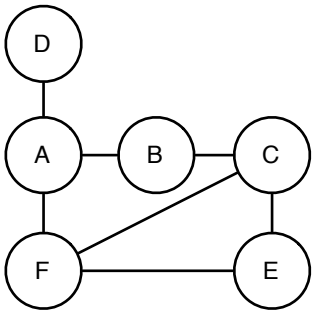


Abbildung 4.10.: Beispielgraph für ein Netzwerk

Diese Entfernungen sind in Wirklichkeit die Kosten. Dieser Vektor wird an die unmittelbaren Nachbarn verteilt.

Im Detail funktioniert der Algorithmus folgendermaßen:

1. Am Anfang wird in jedem Knoten der Vektor folgendermaßen initialisiert.
   - es wird eine 1 eingetragen, wenn es eine direkte Verbindung gibt und diese nicht ausgefallen ist. Der Ausfall eines Nachbarn kann z.B. durch regelmäßige „Hello" Pakete überprüft werden.
   - es wird ∞ eingetragen, wenn es keine direkte Verbindung gibt.
   - es wird eine 0 eingetragen, wenn es die Entfernung zu dem Router selbst angibt.
2. Jeder Knoten sendet seinen Vektor an seine direkten Nachbarn. Es gibt zwei verschiedene Ansätze des Zeitpunktes wann ein Knoten seinen Vektor an seine Nachbarn schickt:
   - Der Knoten schickt periodisch seinen Vektor an seine Nachbarn. Die Größe dieser Zeitdauer liegt im Bereich mehrerer Sekunden bis zu mehreren Minuten.
   - Der Knoten schickt genau dann seinen Vektor an seine Nachbarn, wenn es zu einer Änderung seines eigenen Vektors gekommen ist.
3. Empfängt ein Knoten einen Vektor von einem Nachbarn, dann heißt dies, dass der Knoten alle Knoten dieses Nachbarn über ihn erreichen kann. Deshalb wird zu jeder Distanz des erhaltenen Vektors eine 1 hinzugezählt. Ist der erhaltene Wert kleiner als der Wert des eigenen Vektors, dann wird der neue Wert an Stelle des alten in den eigenen Vektor eingetragen. Weiters wird in der Weiterleitungstabelle vermerkt, dass der Knoten über diesen Nachbarn erreicht werden kann.

Wie sieht das bei unserem Beispiel aus? Die folgende Tabelle zeigt alle Vektoren aller Knoten im Anfangszustand:

Knoten	A	B	C	D	E	F
A	0	1	∞	1	∞	1
B	1	0	1	∞	∞	∞
C	∞	1	0	∞	1	1
D	1	∞	∞	0	∞	∞

E	∞	∞	1	∞	0	1
F	1	∞	1	∞	1	0

Jetzt beginnt das eigentliche Routing. Jeder Knoten sendet seine Vektoren an seine direkten Nachbarn. Beispielsweise könnte dies so aussehen:

1. Knoten B erhält zum Beispiel einen Vektor von Knoten C. Er addiert eine 1 zu den Gewichten der Verbindung zu C und erhält den Vektor $(\infty, 2, 1, \infty, 2, 2)$. Dann vergleicht die Distanzen mit den eigenen Distanzen. Ist eine Distanz kleiner, wird diese in den eigenen Vektor übernommen, anderenfalls kommt es zu keiner Änderung. D.h. der Knoten B hat nach dieser Operation den Vektor (1, 0, 1, ∞, 2, 2) gespeichert.
2. Nehmen wir jetzt an, dass Knoten A den Vektor von Knoten B bekommt. A aktualisiert seinen eigenen Vektor zu (0, 1, 2, 1, 3, 1).
3. Gehen wir weiter davon aus, dass Knoten A jetzt den Vektor von F bekommt. Daher aktualisiert A seinen Vektor zu (0, 1, 2, 1, 2, 1).

Die endgültigen Vektoren werden so aussehen:

Knoten	A	B	C	D	E	F
A	0	1	2	1	2	1
B	1	0	1	2	2	2
C	2	1	0	3	1	1
D	1	2	3	0	3	2
E	2	2	1	3	0	1
F	1	2	1	2	1	0

Selbstverständlich muss sich jeder Knoten auch merken über welchen Knoten er mittels der Distanz den Zielknoten erreichen kann. Je nach Abfolge der Nachrichten könnte die Weiterleitungstabelle von Knoten A im Prinzip so aussehen:

Ziel	nächster Hop
C	B
E	F

Der Router muss natürlich auch die Schnittstellen kennen, über die er die nächsten Hops erreichen kann!

Was passiert jetzt, wenn eine Verbindung ausfällt? Nehmen wir einmal an die Verbindung von B nach C fällt aus. Dann wird B an A mitteilen, dass die Entfernung von B nach C den Wert ∞ ausmachen. Daraufhin wird A seine Entfernung für den Weg nach C auf ∞ setzen, da A das Ziel C über B erreicht hätte. A würde diese Kosten auch an F übermitteln. Da F den Knoten C jedoch direkt erreichen kann, wird F seine Kosten für C nicht ändern. F würde im Gegenzug A mitteilen, dass C ein direkter Nachbar von F ist und A würde die Kosten für C auf 2 verändern und dies auch B mitteilen. B weiß dann, dass C über A in 3 Schritten zu erreichen ist.

Allerdings gibt es ein kleines Problem, das als Count-to-Infinity Problem bezeichnet wird.

Nehmen wir an, dass die Verbindung von A nach D ausfällt. A setzt daher in seinem Vektor den Wert ∞ für die Verbindung nach D ein. Nehmen wir jetzt an, dass danach eine Aktualisierung von F kommt, die die Kosten von 2 für die Verbindung nach D beinhalten. A schlussfolgert daraus, dass es den Knoten D in 3 Schritten erreichen kann und wird dies in einer weiteren Aktualisierungsmeldung wieder an F weitergeben. F wird daraufhin seinen Vektor bezüglich D auf 4 anpassen und diesen wieder aussenden.

Dieser Vorgang bricht erst ab, wenn im Algorithmus die Schranke erreicht wird, die für ∞ eingeführt wurde. Angenommen diese Schranke steht bei 16, dann wird dieser Vorgang bei 16 als „unendlich" abgebrochen werden. Wenn Aktualisierungen alle 20 Sekunden ausgetauscht werden, kann es 15*20 Sekunden, also ca. fünf Minuten dauern, bis der Ausfall eines Routers erkannt und die Verbindung zu ihm als unerreichbar markiert wird. Deshalb gibt es einige Varianten zu diesem Algorithmus.

Eine dieser Verbesserungen nennt sich Split Horizon (soviel wie geteilter Horizont). Dabei sendet ein Knoten keine Aktualisierung an den Knoten zurück von dem er einen Wert erhalten hat. In dem obigen Beispiel würde F keine Änderungsmeldung bzgl. D an A schicken. Allerdings wird A eine Änderungsmeldung an F schicken und dieser dann seinen Vektor bzgl. D ebenfalls auf ∞ setzen.

Betrachten wir jetzt was passiert, wenn die Verbindung von A nach D ausfällt. Wenn jetzt die Aktualisierungen in einer bestimmten Reihenfolge auftreten, kann folgendes passieren: B weiß, dass es D über A nicht mehr erreichen kann, weiß aber von C, dass es D über C in 4 Schritten erreichen kann und aktualisiert seinen Vektor. Gleichzeitig gibt er diese Änderungen an A weiter. A folgert, dass es D in 5 Schritten erreichen kann. A gibt diese Änderung wiederum an F weiter. F folgert, dass es D jetzt in 6 Schritten erreichen kann, usw. D.h. für eine Schleife, die mehr als 2 Knoten enthält funktioniert Split Horizon nicht.

Eine Erweiterung zum Split-Horizon besteht darin, dass der Rechner bei dem die Verbindung ausfällt Meldungen mit negativen Kosten aussendet. Solche „negativen" Meldungen werden weiter verbreitet. Allerdings werden damit keine größeren Schleifen als 3 vermieden.

Die Zeit, die zum Verbreiten konsistenter Routinginformationen benötigt wird, wird als Konvergenzdauer oder kurz Konvergenz (engl. convergence) bezeichnet. Der Distanzvektor Algorithmus hat eine schlechte Konvergenz. Deshalb wurde der Link-State-Algorithmus entwickelt.

Link-State-Algorithmus

Beim Link-State-Algorithmus handelt es sich um einen Routing Algorithmus der nach folgendem Prinzip funktioniert: „Teile der Welt mit, wer deine Nachbarn sind".

Dieser Algorithmus basiert darauf, dass zuerst die Topologie des Netzes ermittelt wird. Dann erst werden die kürzesten Wege bestimmt.

Die Vorgehensweise von jedem Knoten ist folgende:

1. Jeder Knoten bildet ein LSP (link state packet, auch link state announcement, kurz LSA, genannt) mit den Namen seiner Nachbarn und den Gewichten der Verbindungen.

2. Der Knoten verschickt das LSP per Broadcast über alle seine Verbindungen. Es werden jedoch nur die Änderungen übertragen.
3. Der Knoten speichert alle LSP der anderen Knoten. Damit hat jeder Knoten eine vollständige Sicht des Netzes.

Damit dies funktioniert werden zwei grundlegende Mechanismen benötigt:

- Die LSPs müssen zuverlässig alle anderen Knoten erreichen! Man nennt das zuverlässiges Fluten.
- Ein Algorithmus, der die kürzesten Wege in dem Graphen berechnet. Dazu wird der Dijkstra-Algorithmus verwendet.

Zuverlässiges Fluten

Damit zuverlässiges Fluten ermöglicht wird, muss jedes LSP folgende Informationen beinhalten:

- Die Adresse des Knotens, der das LSP erzeugt hat.
- Die Liste der direkten Nachbarn gemeinsam mit den Kosten der Verbindung.
- Eine Sequenznummer für das LSP.
- Eine Lebensdauer oder auch TTL (time to live) genannt für das LSP

Das Fluten funktioniert folgendermaßen:

- Der Austausch der LSPs wird mittels Bestätigungen und Neuübertragungen zuverlässig gemacht.
- Jedes Mal wenn ein Knoten ein neues LSP erzeugt, wird die Sequenznummer erhöht. Solch eine Sequenznummer darf nicht überlaufen. Deshalb muss das Feld dafür relativ groß sein, z.B. 64 Bit. Fällt ein Knoten aus, dann beginnt die Sequenznummer wieder von vorne.
- Das Fluten wird mittels Broadcasts unter Ausnutzung der Sequenznummer und der Lebensdauer realisiert:
  - Empfängt ein Knoten ein LSP und es ist noch kein LSP von diesem Knoten gespeichert, dann wird dieses gespeichert.
  - Ist schon ein LSP von diesem Knoten gespeichert, dann wird die Sequenznummer betrachtet. Ist die neue Sequenznummer größer als die alte Sequenznummer, dann wird das neue LSP gespeichert, anderenfalls wird das neue LSP verworfen.
  - Wurde ein LSP gespeichert, dann wird die Lebensdauer des LSP dekrementiert und dieses LSP an alle Nachbarn außer an den Knoten von dem das LSP empfangen wurde, weitergesendet.
  - Die Lebensdauer wird auch dekrementiert, wenn das LSP im Knoten gespeichert ist.
  - Hat die Lebensdauer den Wert 0 erreicht, wird das LSP auf jeden Fall gelöscht.

Dijkstra-Algorithmus

Jeder Knoten führt den Dijkstra-Algorithmus selbstständig durch. Die Grundlage ist ein Graph, der aus den LSPs konstruiert wurde.

Wir definieren:

- V (engl. vertices) ist die Menge der Knoten, die durchnummeriert von 1 bis size(V) bezeichnet sind.
- w(i,j) ist die Gewichtung der Verbindung vom Knoten i zum Knoten j. w(i,j) = ∞, wenn keine Verbindung zwischen dem Knoten i und dem Knoten j besteht.
- Mit s wird derjenige Knoten bezeichnet, der den Algorithmus ausführt.
- Mit N wird die Menge der Knoten bezeichnet, für die noch kein kürzester Weg zu s gefunden wurde.
- c(n) bezeichnet die Gesamtkosten vom Knoten n zum Knoten s.

Mit diesen Definitionen sieht die Grundstruktur des Dijkstra-Algorithmus folgendermaßen aus:

```
N = V - {s}
for n in N:
    c(n) = w(s,n)        # Kosten für alle Knoten zu s initialisieren
while not empty(N):
    wähle f in N, sodass c(f) minimal ist
    N = N - {f}          # es ist ein neuer Knoten gefunden worden
    for n in N:          # Kosten aktualisieren
        c(n) = min(c(n), c(f) + w(f,n))
```

In dieser Form werden einerseits nur die minimalen Kosten ermittelt und außerdem fällt auf, dass die Wahl von f in N, sodass c(f) minimal wird, jeweils alle Knoten in N betrachten muss. Die Idee ist, die Menge der noch nicht behandelten Knoten weiter zu unterteilen. Es wird die Menge der Randknoten B (engl. border) eingeführt, die alle Knoten beinhaltet, die direkte Nachbarn der Knoten sind, für die schon ein kürzester Weg gefunden wurde:

```
R = {s}                  # Menge der Ergebnisknoten initialisieren
for v in V - {s}:        # alle anderen Knoten
    c(v) = w(s,v)        # Kosten für alle Knoten zu s
    p(v) = None          # Vorgänger gibt es noch keinen
B = neighbours(s)        # mit Nachbarn von s initialisieren
for b in B:              # und Vorgänger von b setzen
    p(b) = s
while not empty(B):
    wähle f in B, sodass c(f) minimal ist
    B = B - {f}          # aus Rand entfernen
    R = R + {f}          # zu den Gefundenen hinzufügen
    for n in neighbours(f):
        if n not in R:
            B = B + {n}  # mit Nachbarn zu f auffüllen
            if c(f) + w(f,n) < c(n): # wenn kürzer
```

```
c(n) = c(f) + w(f,n)
p(n) = f
```

Praktische Implementierungen in Routern interessieren sich nicht für die genauen Wege sondern nur für die nächsten Router. Insoferne wird nicht der Vorgänger des Knotens bestimmt sondern gleich der nächste Router.

Folgendes Beispiel soll das verdeutlichen:

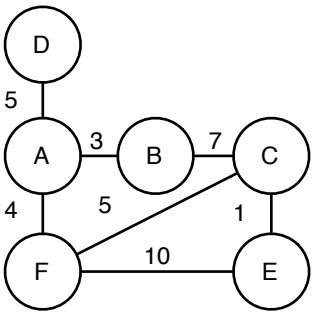


Abbildung 4.11.: Beispielgraph für ein Netzwerk mit Gewichten

Vorausgesetzt, dass der Knoten A der Knoten ist, der den Dijkstra-Algorithmus ausführt werden folgende Schritte abgearbeitet:

1. Die Menge der Ergebnisknoten mit A initialisieren.
2. Randknoten bestimmen.
3. Knoten mit kleinstem Gewicht innerhalb der Randknoten bestimmen, also B und diesen in die Menge der Ergebnisknoten überführen. Neue Menge der Randknoten bestimmen und Gewicht von C auf 10 korrigieren.
4. Knoten mit kleinstem Gewicht innerhalb der Randknoten bestimmen, also F und diesen in die Menge der Ergebnisknoten überführen. Neue Menge der Randknoten bestimmen und Gewicht von C und E korrigieren. Gleichzeitig wird der Vorgänger von C der Knoten F.
5. Knoten mit kleinstem Gewicht der Randknoten bestimmen, also D und diesen in die Menge der Ergebnisknoten überführen. Menge der Randknoten ist nicht größer geworden, da keine Knoten mehr vorhanden sind, die weder in der Menge der Ergebnisknoten noch in der Menge der Randknoten sind.
6. Knoten mit kleinstem Gewicht der Randknoten bestimmen, also C und diesen in die Menge der Ergebnisknoten überführen. Gewicht von C auf 10 korrigieren und als Vorgänger von E den Knoten C eintragen.

Das folgende Abbildung zeigt diese einzelnen Schritte. Die Menge der Ergebnisknoten ist dunkelgrau und die Menge der Randknoten hellgrau gekennzeichnet:

4.3. Ende-zu-Ende Kommunikation

Ab der Schicht 4 im OSI Modell findet eine Ende-zu-Ende Kommunikation statt, während in den Schichten 1 bis 3 jeweils eine Punkt-zu-Punkt Kommunikation stattfindet.

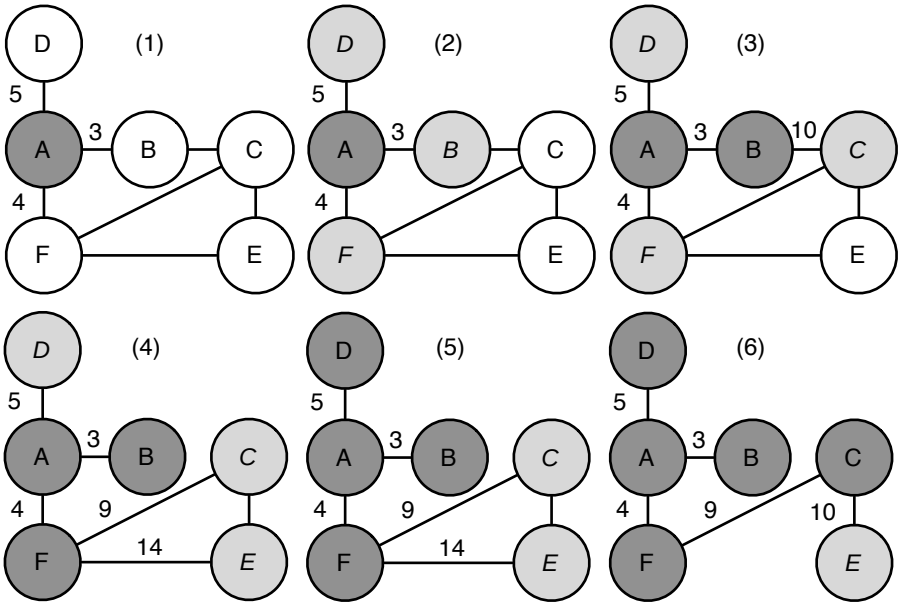


Abbildung 4.12.: Abarbeitung des Dijkstra-Algorithmus

Zu den Aufgaben gehören:

- Bereitstellen der Verbindungslogik.
- Verwalten (mehrerer) Verbindungen.
- Flusskontrolle.
- Verwaltung von Sitzungen (engl. session).
- Kompression und Verschlüsselung.
- Darstellungsformatierung behandelt die Umwandlung der verschiedenen Datenformate, sodass der Informationsgehalt jedoch unverändert bleibt. Dafür gibt es:
  - Spezifikationen der ISO: ASN.1 (abstract syntax notation one) mittels der Datentypen beschrieben werden können und BER (basic encoding rules), die eine konkrete Transfersyntax beschreibt.
  - Standards des W3C (world wide web consortium): XML, XPath, XPointer, XLink, XSLT, XML Schema

4.4. Überlastkontrolle

Obwohl die Verbindungen auf der Transportebene eine Flusskontrolle bieten und damit den Empfänger vor Überlastung schützen, kann es vorkommen, dass die Router auf dem Weg überlastet werden. Zusätzlich gibt es noch die verbindungslosen Nachrichtenübertragungen, die ebenfalls eine Last für die Router darstellen.

Folgende grundsätzliche Möglichkeiten zum Umgang mit Überlast gibt es:

- Überdimensionierung des Netzes.

- Überlast wird aus dem Netz entfernen. Dazu muss die Netzwerksoftware wissen welche Pakete entfernt werden sollen.
- Neue Kommunikationsbeziehungen werden nur zugelassen, wenn genügend Kapazitäten frei sind (analog zum Telefonnetz).
- Quelle wird gedrosselt (TCP hat ein Verfahren, dass darauf basiert)

Teil II.

Technologien

In diesem Teil werden die wichtigsten Technologien für LAN, WAN und Zugangsnetze vorgestellt und im Überblick beschrieben.

5. Ethernet

Ethernet wurde von der IEEE im Standard IEEE 802 und von der ISO als ISO-Standard 8802 als weltweiter Standard übernommen. Innerhalb dieses Standards gibt es viele Teilspezifikationen, die selbst wiederum gegliedert sind und auch andere Technologien wie Token-Ring, Token-Bus oder drahtlose LANs enthält.

802.2 Logical Link Control (LLC) ist die Übertragungssteuerung der Sicherungsschicht, die auf dem MAC Protokoll aufsetzt und damit eine Unabhängigkeit von der unterliegenden Übertragungstechnologie sicherstellt.

802.3 Medium Access Control (MAC) – also die Zugriffssteuerung – und die Spezifikation des Physical Layer (PHY) des CSMA/CD basierenden Ethernet. Wichtige Teile sind:

- **802.3** 10Base-5 Ethernet basierend auf Koaxialkabel (Thick Coax). 10 gibt die Übertragungsgeschwindigkeit in MBit/s an, Base bedeutet Basisband und 5 ist die maximale Länge eines Segmentes in 100m (also 500m).
- **802.3a** 10Base-2 Ethernet basierend auf Koaxialkabel (Thin Coax). Die maximale Länge eines Segmentes beträgt hier 185m, also aufgerundet sind das 200m. Daher kommt die Ziffer 2 in der Namensbezeichnung.
- **802.3u** 100Base-TX Ethernet basierend auf Twisted-Pair Kabel und 100Base-FX basierend auf Glasfaser. Diese Art wird auch als Fast Ethernet bezeichnet.
- **802.3ab** 1000Base-T Ethernet über Cat-5 Kabel mit einer Distanz bis 100m. Damit man Cat-5 Kabel verwenden kann, wurden Änderungen am ursprünglichen Ethernet vorgenommen. Dazu zählt die Verwendung von 4 Aderpaaren anstatt von 2 wie bei Fast Ethernet.
- **802.3z** 1000Base-LX bzw. 1000Base-SX Ethernet über Glasfaser mit Distanzen von 440m bis zu 5km.
- **802.3ae** 10GBase-xx Ethernet. Die Ziele von 10GEthernet sind, dass es im LAN, MAN und WAN gleichermaßen einsetzbar ist und ebenfalls (wie 1000Base-T) eine kostengünstige Technologie ist. Dafür wurde festgelegt, dass nur mehr Punkt-zu-Punkt Verbindungen im Vollduplex-Betrieb verwendet werden und als Übertragungsmedium Glasfasern verwendet werden. D.h., es müssen je Vollduplex-Verbindung zwei Glasfasern verwendet werden. Da keine Kollisionen mehr auftreten können, ist das CSMA/CD Prinzip nicht mehr notwendig und große Link-Längen werden möglich.

  xx steht für verschiedene Varianten, die in Abhängigkeit der der Wellenlänge und der Glasfaser Distanzen von 300m bis 40km ermöglichen.

802.11 Wireless LAN (WLAN). Beschreibt die Zugriffssteuerung und den physische Schicht für drahtlose Netze.

802.15 Wireless PAN. Definiert die drahtlose Kommunikation für den nutzernahen Bereich über kurze Distanzen. Datenraten 1MBit/s bis 20MBit/s.

5.1. Prinzip

5.1.1. CSMA/CD

Bei den genannten Ethernet Varianten 10Base-5, 10Base-2 sowie dem Fast Ethernet handelt es sich um eine klassische Bustopologie mit einem Zugriffsverfahren, das auf einem Wettbewerbsverfahren basiert. Das bedeutet, dass alle Stationen einen gemeinsamen Übertragungskanal benützen und alle Stationen gleichberechtigt sind. Die Station, die als erstes zum Senden beginnt, darf den Übertragungskanal alleine nutzen. D.h. die Stationen stehen in einem Wettbewerb zueinander. Bei dem implementierten Wettbewerbsverfahren handelt es sich um CSMA/CD (carrier sense multiple access/collision detect).

Jede Station, die nicht senden will hört permanent das Übertragungsmedium ab. Erkennt es einen Rahmen (engl. frame), der an sie gerichtet ist, wird dieser gelesen ansonsten wird der Rahmen ignoriert. Will eine Station einen oder mehrere Rahmen senden, muss ein Algorithmus abgearbeitet werden, da es vorkommen kann, dass zwei (oder mehrere) Stationen gleichzeitig (engl. multiple access) zum Senden beginnen:

1. Eine sendebereite Station hört das Übertragungsmedium ab (engl. carrier sense). Ist das Übertragungsmedium besetzt, wird gewartet bis dieses frei ist und zusätzlich werden noch zusätzliche 9.6μs gewartet (engl. interframe gap).
2. Ist das Übertragungsmedium frei, beginnt die Station zu senden.
3. Trotz Überwachung des Übertragungsmediums, kann es vorkommen, dass zwei Stationen zur gleichen Zeit senden. Das kann vorkommen, da beide Stationen ein freies Übertragungsmedium vorgefunden haben und deshalb zur gleichen Zeit zu senden begonnen haben. Da die Stationen jedoch weiter am Übertragungskanal hören, hören sie nicht ihr eigenes Signal, sondern das überlagerte Signal. Dieses überlagerte Signal kennzeichnet eine Kollision. Bei Erkennen einer Kollision (engl. collision detect) wird von den Stationen folgendes durchgeführt:

   a) Die Stationen brechen das Senden ab.

   b) Die Stationen senden ein spezielles Stausignal (engl. jam signal), damit eindeutig eine Kollision erkannt wird.

   c) Danach wird eine variable Zeit gewartet (Backoff-Zeit), die mit jedem fehlgeschlagenen Sendeversuch größer werden kann.

Damit dieser Algorithmus in dieser Form funktioniert, müssen – in Abhängigkeit von der Sendegeschwindigkeit und der Netzausdehnung – bestimmte Anforderungen an die Länge des Stausignals bzw. an die Wartezeiten gestellt werden.

Das Flussdiagramm in Abbildung 5.1 auf der nächsten Seite zeigt den Algorithmus in einer Übersicht.

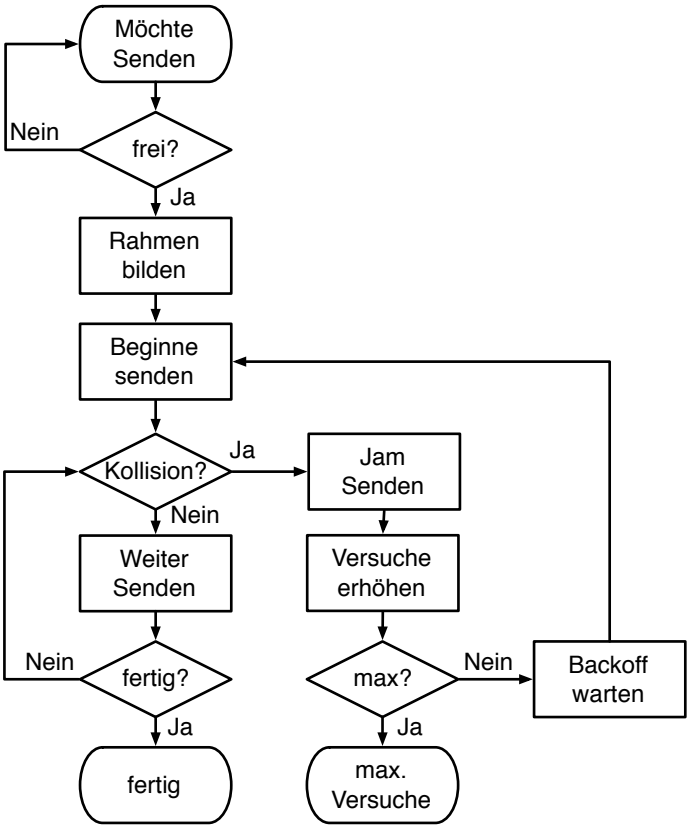


Abbildung 5.1.: CSMA/CD

5.1.2. Adressierung und Rahmenaufbau

Wie im vorigen Abschnitt erläutert, sendet eine Station Frames über den Bus. Damit eine Station einen Frame an eine andere Station senden kann, wird die Adresse dieser Station benötigt.

MAC Adressen

MAC Adressen sind für alle IEEE 802 Standards einheitlich definiert und sind weltweit prinzipiell eindeutig. Diese MAC Adresse wird vom Hersteller in einen nichtflüchtigen Speicher der Schnittstellenkarte (engl. network interface card, NIC) geschrieben.

Die MAC Adressen sind insgesamt 48 Bits lang setzen sich aus 3 Teilen zusammen:

1. Adresstyp bestehend aus 2 Bits (höchstwertigste Bits, also Bit 47 und Bit 46), die angeben, ob es sich um eine Individualadresse oder eine Gruppenadresse (Multicast-Adresse) bzw. um eine durch die IEEE verwaltete Adresse oder eine lokal verwaltete Adresse (private Adresse) handelt.
2. die nächsten 22 Bits kennzeichnen den Hersteller der NIC und werden weltweit eindeutig vergeben.
3. einem Teil, der von dem Hersteller der NIC selbst und eindeutig vergeben wird.

Die Adresse mit dem Wert FF-FF-FF-FF-FF-FF ist die Broadcast-Adresse.

Generell liest eine NIC nur die Frames, die an die eigene Adresse gesendet worden sind. Allerdings kann die NIC in einen anderen Modus geschalten werden, sodass alle Frames gelesen werden. Dieser Modus wird promiscuos mode (dt. ausschweifend) genannt.

Rahmenaufbau

Für den Aufbau eines Rahmen gibt es historisch gesehen zwei verschiedene Arten, wobei heutzutage nur mehr die Variante von IEEE802.3 im Einsatz ist. Die IEEE 802.3 Variante sieht folgendermaßen aus:

1. Präambel (7 Bytes): eine alternierenden Bitfolge (also 1010...), die der Synchronisation der Stationen dient.
2. SFD (start frame delimiter, 1 Byte): alternierende Bitfolge 10101011, die den Start des Frame kennzeichnet.
3. Zieladresse (6 Bytes)
4. Quelladresse (6 Bytes)
5. Längenfeld (2 Bytes)
6. Datenblock (0 bis 1500 Bytes)
7. Pad (0 bis 46 Bytes): beinhaltet der Datenblock weniger als 46 Bytes, dann müssen eine entsprechende Anzahl von Pad Bytes übertragen werden, sodass der Rahmen auf eine minimale Größe von 64 Bytes kommt.
8. FCS (engl. frame check sequence, 4 Bytes): ein CRC, der ebenfalls vom Empfänger berechnet wird, sodass fehlerhafte Frames einfach erkannt und danach verworfen werden.

5.2. Netzaufbau

Wie schon beschrieben handelt es sich bei Ethernet um eine Bus-basierte Technologie. Bzgl. der Verkabelung gibt es allerdings zwei Möglichkeiten. Entweder die Verkabelung wird ebenfalls als Bustopologie ausgeführt (mittels Koaxialkabel) oder sternartig wie bei Twisted-Pair Kabeln bzw. bei Glasfaserkabeln.

5.2.1. Bustopologie mit Koaxialkabel

Früher eingesetzte Ethernet Technologien sind 10Base-5 bzw. 10Base-2, die auf Koaxialkabeln basieren (siehe 3.2.1 auf Seite 38). Ein Segment besteht aus einem langen Kabel, das an beliebigen Stellen entweder angebohrt (10Base-5) bzw. mittels BNC Steckverbindungen (10Base-2) den Anschluss von Arbeitstationen ermöglicht. Mehrere Segmente können mittels Repeater oder Brücken zusammengeschlossen werden.

Auf Grund der technischen Spezifikationen mussten folgende Bedingungen eingehalten werden, die unter der 5-4-3 Regel bekannt waren: Zwischen je zwei Endgeräten dürfen maximal 5 Segmente und 4 Repeater liegen. Nur an 3 Segmenten ist der Anschluss von Arbeitsstationen erlaubt.

5.2.2. Sterntopologie mit Twisted-Pair Kabel bzw. Glasfaser

Heute wird hauptsächlich die Sterntopologie verwendet. Es werden die Arbeitstationen sternförmig mit Kopplungselementen (siehe 4.1.2 auf Seite 54) verbunden. Bei den Kopplungsgeräten handelt es sich hauptsächlich um Hubs, Bridges und Switches. In dieser Konfiguration wird die Ethernet Technologie im Zuge der strukturierten Verkabelung eingesetzt (siehe 4.1.3 auf Seite 57).

5.3. Spanning-Tree-Algorithmus

Der Backward-Learning Algorithmus (siehe 4.1.2 auf Seite 54) hat allerdings ein Problem sobald Schleifen in der Netzstruktur vorhanden sind. Schleifen entstehen, wenn redundante Pfade vorhanden sind. Solche redundante Pfade werden gerne verwendet, um die Verfügbarkeit des Netzes zu erhöhen.

Das Problem lässt sich folgendermaßen beschreiben: Durch das Fluten in einer Schleife werden Frames im Kreis gesendet und es ensteht eine unnötige Last. Ist der Zielhost im Netz vorhanden, dann wird dieser eine Antwort schicken und die Switches können sich im Zuge des Backward-Learning auch die Ziel-Adresse merken, wodurch das Fluten ausbleibt und die Frames nicht mehr im Kreis gesendet werden. Ist der Zielhost allerdings nicht im Netz verfügbar ist, dann kreisen die Frames in der Schleife. Genauso ist problematisch wie fehlende Zielhosts ist, wenn Broadcast- oder Multicast-Frames verschickt werden, da kreisende Frames auch nicht durch Antworten der Zielhosts entfernt werden.

Eine einfache Möglichkeit, die von Switches implementiert wird ist, dass diese automatisch einen Port abschalten, wenn die Auslastung auf diesen Port zu hoch (z.B. größer als 90%) wird. Die Argumentation ist, dass ein solch hohes Verkehrsaufkommen im normalen Betrieb nicht auftritt und deshalb vermutlich kreisende Frames die Ursache sind.

Die bessere Lösung, die redundante Verbindungen erst ermöglicht ist die Implementierung des Spanning Tree Protocol (STP, IEEE802.1). Das Ziel dieses Verfahrens ist, eine schleifenlose Topologie aufzubauen (einen Baum), die alle Switches einbindet und die redundanten Verbindungen deaktiviert. Stellt man sich das LAN als einen Graphen vor, der Zyklen enthält, dann wird durch den STP ein zyklenloser Graph generiert, der alle Zielhosts enthält. Eine Lastverteilung ist in diesem Fall nicht mehr möglich.

Jeder Switch hat einen eindeutigen Identifier und tauscht mit den anderen Switches im Zuge des STP Konfigurationsnachrichten aus. Der prinzipielle Ablauf von STP sieht folgendermaßen aus:

1. Es wird der Switch mit dem kleinsten Identifier als Wurzel des Baumes festgelegt. Eine Wurzel leitet Frames prinzipiell an alle ihre Ports weiter (d.h. keiner der Ports ist deaktiviert).

2. Dann berechnet jeder Switch den kürzesten Weg zur Wurzel und merkt sich den Port, der zur Wurzel führt.
3. Alle an ein Segment angeschlossene Switches wählen einen designierten Switch, der alle Nachrichten aus diesem Segment an die Wurzel weiterleitet. Es wird derjenige Switch gewählt, der den kleinsten Weg zur Wurzel hat. Gibt es mehrere Switches mit dem kleinsten Weg, dann wird wieder derjenige ausgewählt, der von diesen den kleinsten Identifier besitzt. D.h. derjenige Switch, der nicht als designierter Switch festgelegt wurde, deaktiviert seinen Port Richtung Wurzel.
4. Ist der Prozess abgeschlossen, dann sendet nur mehr die Wurzel Konfigurationsnachrichten aus, die von allen Switches weitergeleitet werden. Enthält ein Switch innerhalb einer gewissen Zeitspanne keine Konfigurationsnachricht, wird angenommen, dass ein Switch ausgefallen ist und der Prozess beginnt wieder von vorne.

In Abbildung 5.2 ist ein Beispiel eines Netzes als Graph zu sehen, in dem die deaktivierten Ports strichliert markiert sind. Als Wurzel wurde S1 bestimmt.

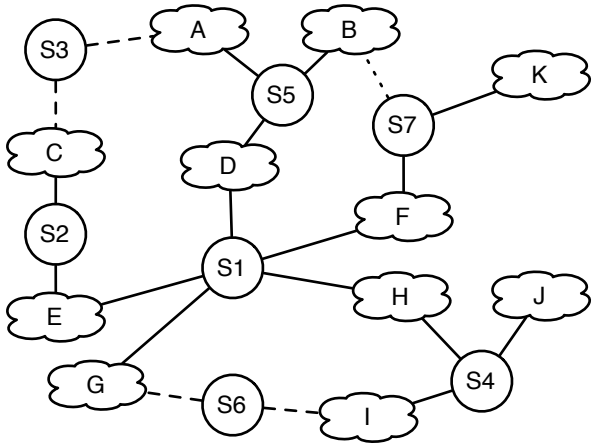


Abbildung 5.2.: Beispiel eines Netzes mit durch STP deaktivierten Ports

6. WLAN

Heute wird unter WLAN (wireless LAN) meist die in IEEE 802.11 beschriebene Technologie verstanden. Diese ist für den Aufbau drahtloser Netze innerhalb eines begrenzten geographischen Bereich (Haushalte, Bürogebäude, Firmengelände) ausgelegt.

6.1. Prinzip

In IEEE 802.11 sind zwei Übertragungstechniken mittels Funk im Spreizspektrumverfahren bei dem das Signal auf ein breiteres Frequenzband ausgedehnt (gespreizt) wird, um die Auswirkungen von Störungen durch andere Geräte zu minimieren:

- Frequenzsprungverfahren (engl. frequence hopping): es wird das Signal auf ein ständig die Frequenz wechselndes Trägersignal moduliert. Die Auswahl der Frequenz finden pseudozufällig mittels eines Pseudozufallszahlen-Generators statt. Sowohl Sender als Empfänger initialisieren ihren Pseudozufallszahlen-Generator gleich.
- Direct Sequence: Dieses Spreizspektrumverfahren basiert darauf, dass jeder Teilnehmer einen Code (wieder eine Pseudozufallfolge) erhält mit der er seinen eigentlichen Bitstrom verknüpft. Damit wird das Signal auf eine relativ große Bandbreite gespreizt. Alle Sender belegen die selbe Bandbreite. Der Empfänger verwendet wieder die selbe Pseudozufallsfolge und kann dadurch das Signal wieder rekonstruieren. In IEEE 802.11 wird eine 11 Bit lange Folge für jedes Bit in dem ursprünglichen Signal verwendet.

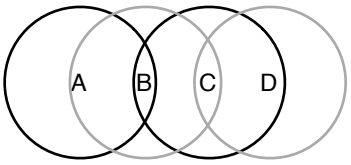
Die dritte Übertragungstechnik basiert auf Infrarotübertragung und ist auf ca. 10m Reichweite begrenzt. Sender und Empfänger müssen nicht direkt aufeinander ausgerichtet sein.

Mit den Funk-basierten Verfahren lassen sich Übertragungsraten im Bereich von 11MBit/s bis 54MBit/s erzielen. Der neue Standard IEEE 802.11n, der derzeit als Draft vorliegt erlaubt Datenraten bis 540MBit/s (eine endgültige Version des Standard wird für 2008 erwartet).

6.2. Kollisionsvermeidung

Das Verfahren von IEEE802.11 ist ähnlich dem CSMA/CD Verfahren von Ethernet, jedoch kann keine Kollisionserkennung durchgeführt werden. Lediglich eine Kollisionsvermeidung lässt sich erzielen. Das liegt daran, dass nicht jeder Knoten in der Reichweite der anderen befindet. Zwei Probleme treten auf:

Verborgene Knoten (engl. hidden node) Es kann prinzipiell vorkommen, dass es zu Kollisionen kommt, diese jedoch nicht erkannt werden (siehe nachfolgende Abbildung; die Umgebung von B und D ist der Übersichtlichkeit halber grau eingezeichnet). Angenommen A und C möchten mit B kommunizieren und senden je einen Frame. A und C können die Kollision bei B nicht erkennen! D.h. A und C sind für einander verborgene Knoten.



Exponierte Knoten (engl. exposed node) Angenommen B sendet an A. Knoten C weiß von der Kommunikation, allerdings wäre es für C falsch anzunehmen, dass C an niemanden übertragen darf, nur weil C die Kommunikation an A bemerkt. C könnte ohne Probleme an D senden.

In IEEE 802.11 sind zwei prinzipiell verschiedene Methoden definiert wie diese Probleme gelöst werden:

RTS/CTS funktioniert folgendermaßen: Sender und Empfänger tauschen zusätzliche Nachrichten aus, um festzulegen wer senden darf. Bevor ein Sender senden will sendet dieser ein RTS (request to send) Frame an den Empfänger. Der Empfänger antwortet mit einem CTS (clear to send) Frame. CTS reicht das Längenfeld an den Sender zurück. Sieht ein anderer Knoten den CTS Frame, weiß er, dass er nahe beim Empfänger liegt und daher solange nicht übertragen kann, bis ein Frame mit der spezifizierten Länge übertragen wurde. Ein Knoten, der zwar den RTS aber nicht den CTS sieht befindet sich nicht nahe genug am Empfänger, um diesen zu stören, sodass er nach Belieben übertragen kann. Außerdem sendet der Empfänger nach dem Erhalt der Nachricht ein ACK an den Sender. Alle Stationen müssen auf dieses ACK warten bis sie übertragen können. Zwei RTS Signale können trotzdem am Übertragungskanal kollidieren. Da solch eine Kollision nicht erkannt werden kann, nimmt der Sender an, dass eine Kollision aufgetreten ist, wenn innerhalb einer gewissen Zeitspanne kein CTS erkannt wird und sendet nach einer gewissen Zeitdauer wieder (wie bei Ethernet).

PCF (point coordination function) Der Zugangspunkt (siehe Abschnitt 6.3) ist der einzige, der spontan senden darf und fragt alle Stationen permanent ab (polling), ob sie etwas senden wollen. Es handelt sich also um eine zentrale Zuteilung. Dadurch werden Kollisionen vermieden.

6.3. Netzaufbau

IEEE 802.11 Netze können entweder als Infrastrukturnetze oder als Ad-hoc-Netze aufgebaut werden. Ein Infrastrukturnetz liegt vor, wenn die Teilnehmer über feste Zugangspunkte (engl. access point, AP) kommunizieren. Ad-hoc-Netze benötigen keine Infrastruktur, d.h.

die Teilnehmer kommunizieren direkt miteinander und steuern den Ablauf der Kommunikation.

Bei einem Infrastrukturnetz mit mehreren APs sind diese mittels eines Verteilnetzes miteinander verbunden. Jeder Knoten meldet sich bei einem AP an, indem er einen Probe-Frame sendet (aktives Scanning), der von jedem AP beantwortet, der in der Umgebung des Knoten liegt. Der Knoten wählt sich darauf hin einen der AP aus. Alternativ dazu gibt es die Möglichkeit, dass der AP von sich aus in regelmäßigen Abständen sogenannte Beacon-Frames (dt. Bake) aussendet (passives Scanning) und seine Präsenz als auch die von ihm unterstützten Übertragungsraten bekannt gibt.

6.4. Alternative Technologien

6.4.1. Bluetooth

Bluetooth (IEEE 802.15.1) ist eine andere funkbasierte Technologie mit Schwerpunkt der Vernetzung von Geräten über eine kurze Distanz (je nach Sendeleistung von 10m bis 100m). Es wurde als solches für den Einsatz als PAN konzipiert und wird deshalb auch zur Vernetzung von Handys, PDAs, Computer oder Peripheriegeräten verwendet. Für diese Anwendungen unterstützt es Daten und Sprache und auch eine Verschlüsselung.

Um für diese Einsatzgebiete verwendbar zu sein, wurde es von Haus aus als kostengünstige Technik entworfen und verwendet auch das weltweit verfügbare ISM (Industrial, Scientific and Medical Band) im 2.4GHz Band. Es ermöglicht Datenraten bis 700KBit/s im Download und gleichzeitig 57KBit/s im Upload. In der erweiterten Variante werden Datenraten bis 2.1MBit/s unterstützt.

6.4.2. IrDA

Die IrDA-Schnittstelle (Infrared Data Association) basiert auf einer gerichteten Sichtverbindung mit infrarotem Licht und hat die Zielstellung Kabelverbindungen zu ersetzen. Die Verbindung ist auf eine Entfernung von 2m begrenzt und ermöglicht halbduplex eine Datenrate von 115KBit/s. In der Version 1.1 der IrDA Spezifikation ist eine Datenrate von 4MBit/s möglich.

Teil III.

TCP/IP

Dieser Teil beschreibt den „defacto" Standard TCP/IP in solch einem detaillierten Niveau wie für das Verständnis von Rechnernetzen notwendig.

7. Überblick und Struktur von TCP/IP

TCP/IP ist bezüglich der Funktionen ebenfalls in Schichten aufgeteilt, auch wenn die Schnittstellen zwischen den Schichten (im Gegensatz zum ISO/OSI Modell) nicht definiert sind. Es wird oft eine Abbildung der Schichten vom TCP/IP Stack zum OSI Modell vorgenommen. Auf der untersten Ebene gibt es wieder die Bitübertragungsschicht, darüber die Sicherungsschicht, darüber die Vermittlungsschicht und zum Schluss die Transportschicht. Darüber hinaus gibt es nur noch eine Anwendungsschicht und die eigentlichen Anwendungen.

- D.h. der TCP/IP Protokollstack lässt sich in diesem Sinne als ein Schichtenmodell von 4 Schichten beschreiben, die jeweils 4 Schichten im ISO/OSI Referenzmodell zugeordnet werden können:
- Als unterste Schicht dient die Verbindungsschicht (engl. auch network interface layer genannt), die die Bitübertragungsschicht *und* die Sicherungsschicht des ISO/OSI Modells umfasst.
- Darauf baut die Internetschicht (engl. internet layer) auf, die der Vermittlungsschicht im ISO/OSI Modell entspricht. In dieser Schicht gibt es in TCP/IP die Protokolle Internet Protocol (IP), Internet Control Message Protocol (ICMP), das Address Resolution Protocol (ARP) und das Reverse Address Resolution Protocol (RARP).
- Die nächste Schicht ist die Transportschicht (engl. transport layer), die wiederum direkt auf die Transportschicht im ISO/OSI Modell abgebildet werden kann. Im TCP/IP Modell gibt es zwei verschiedene Transportprotokolle: Transport Control Protocol (TCP) und User Datagram Protokoll (UDP).
- Als nächste Schicht folgt im TCP/IP Modell direkt die Anwendungsschicht, die hiermit auch die Funktionen der Sitzungsschicht und der Darstellungsschicht beinhalten muss. Für diese beiden Schichten des ISO/OSI Modells gibt es in TCP/IP kein Gegenstück.

Die Abbildung 7.1 auf der nächsten Seite stellt die Schichten und die Querverbindungen dar. Es ist daraus auch zu erkennen, dass das TCP/IP Modell eigentlich kein striktes Schichtenmodell implementiert.

Man sieht, dass in der Verbindungsschicht nicht nur IP (der eigentliche Kern von TCP/IP) angesiedelt ist, sondern noch weitere Protokolle, die nicht direkt zu IP gehören, jedoch ein integraler Bestandteil dieser Schicht sind. Dies ist einerseits ICMP, das hauptsächlich zur Benachrichtigung von Fehlern in der Internetschicht dient und die Protokolle ARP und RARP, die zur Umsetzung von IP Adressen in Hardwareadressen (wie z.B. Ethernet-MAC-Adressen) bzw. umgekehrt dienen.

Der Hauptzweck der Internetschicht liegt in der globalen Adressierung der Hosts und der Übertragung der Pakete zwischen den Hosts.

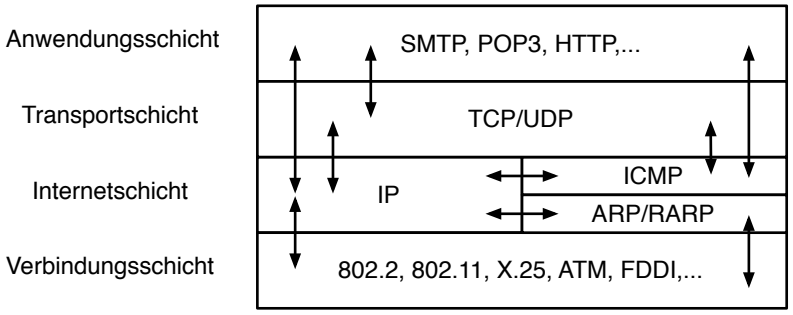


Abbildung 7.1.: TCP/IP Schichtenmodell

In der Transportschicht gibt es zwei Protokolle, nämlich das verbindungsorientierte, zuverlässige TCP und das verbindungslose, unzuverlässige UDP.

Die Anwendungsschicht ist in den eigentlichen TCP/IP Standards nicht beschrieben und muss, wie schon erwähnt, ebenfalls die Funktionen der Sitzungsschicht und der Darstellungsschicht übernehmen. D.h. obwohl der TCP/IP Protokollstack vom Prinzip her ebenfalls eine Anwendungsschicht vorsieht, es jedoch keinen Standard gibt, ist jede Anwendung aus diesem Grund dafür selbst zuständig die benötigten Funktionen der Sitzungsschicht, der Darstellungsschicht und der Anwendungsschicht zu implementieren.

Die Protokolle IP, TCP, UDP, ICMP und ARP werden in weiterer Folge noch genauer beschrieben.

8. Internet Protokoll

Das Internet Protokoll (IP) ist der Kern des TCP/IP Stacks. Derzeit ist die Version 4 (IPv4) beschrieben im RFC 791 aus dem Jahr 1981 flächendeckend im Einsatz. Weitere relevante RFCs sind 950, 919 and 922 und RFC 2474.

Bei IP handelt es sich um ein verbindungsloses Protokoll auf der Schicht 3, dessen Hauptfunktionen in der Bereitstellung der Adressierung, der Weiterleitung und der Abstraktion der unterliegenden physikalischen Schicht liegt.

Es handelt sich bei IP um ein nicht zuverlässiges Protokoll, das nach dem best-effort Prinzip arbeitet. Das bedeutet, dass versucht wird die Pakete so gut wie möglich zuzustellen. Trotzdem können Pakete verloren gehen, nicht in der richtigen Reihenfolge ankommen oder auch mehrfach ankommen. IP geht davon aus, dass die höheren Schichten sich um diese Punkte kümmern.

8.1. IP Adressen

Die IP Adresse dient dazu, Hosts auf der Vermittlungsschicht eindeutig ansprechen zu können und die Wegewahl von IP-Paketen zwischen Sender und Empfänger zu unterstützen.

In IPv4 besteht jede Adresse aus genau 32 Bits, die meist in 4 Bytes angeschrieben werden. Die Bytes werden in der Darstellung durch je einen Punkt getrennt und in dezimaler Zahlendarstellung angegeben.

IPv4 unterteilt die Adressen in Unicast-, Multicast- und Broadcastadressen. Zusätzlich bietet IPv4 insgesamt 3 Arten von Adressierungsvarianten: Standardadressen, Subnetzadressen und das sogenannte Classless Inter-Domain Routing (CIDR).

Prinzipiell besteht jede Adresse aus einem Netzanteil und einem Hostanteil. Der Netzanteil identifiziert das Netz und der der Hostanteil gibt den Host innerhalb des Netzes an. Der Grund für diese Einteilung liegt in der Notwendigkeit der Weiterleitung!

8.1.1. Standardadressen

Bei den Standardadressen gibt es 4 Hauptadressklassen und eine Klasse, die für zukünftige Anwendungen vorgesehen ist:

Klasse A Diese Klasse ist für eine hohe Anzahl an Hosts vorgesehen. Der Netzanteil ist jeweils das erste Byte einer Adresse, wobei das erste Bit einer solchen Adresse ist immer 0 ist. Daraus ergibt sich ein theoretischer Adressbereich von 0.0.0.0

bis 127.255.255.255. Die Netze 0.0.0.0 und 127.0.0.0 sind jedoch reserviert, sodass es 126 verschiedene Klasse A Netze gibt. Je Klasse A Netz gibt es maximal 16.777.214 ($2^{24} - 2$) Hosts. Der Grund warum 2 abgezogen werden muss und warum die Netze 0.0.0.0 und 127.0.0.0 ebenfalls nicht zur Verfügung stehen wird im Abschnitt 8.1.2 auf der nächsten Seite und im Abschnitt 8.1.3 auf Seite 92 beschrieben. Der effektive, nutzbare Bereich geht also von 1.0.0.1 bis 126.255.255.254.

Klasse B Klasse B Netze sind für eine mittlere Anzahl an Hosts vorgesehen. Der Netzanteil sind die beiden ersten Bytes, wobei die ersten beiden Bits immer 10 sind. Daraus ergibt sich ein theoretischer Adressbereich von 128.0.0.0 bis 191.255.255.255. Insgesamt sind so 16384 (2^{14}) verschiedene Klasse B Netze möglich. Je Klasse C Netz gibt es maximal 65534 ($2^{16} - 2$). Der effektive Bereich ergibt sich zu 128.0.0.1 bis 191.255.255.254.

Klasse C Klasse C Netze sind für eine kleine Anzahl an Hosts vorgesehen. Der Netzanteil sind die ersten drei Bytes, wobei die ersten drei Bits immer 110 sind. Daraus ergibt sich ein theoretischer Adressbereich von 192.0.0.0 bis 223.255.255.255. Insgesamt sind so 2097152 (2^{21}) verschiedene Klasse C Netze möglich. Je Klasse C Netz gibt es maximal 254 ($2^8 - 2$) Hosts. Der effektive Bereich ergibt sich zu 192.0.0.1 bis 223.255.255.254.

Klasse D Klasse D Netzadressen sind für Multicast-Anwendungen reserviert. Die ersten vier Bits sind immer 1110 und die restlichen 28 Bits geben die Multicast-Gruppen-Id an. Der theoretische Adressbereich geht von 224.0.0.0 bis 239.255.255.255. Von diesen sind jedoch einige vordefiniert, wie z.B. wieder 224.0.0.0 als reservierte Basisadresse, 224.0.0.1 für alle Hosts in dem Subnetz und weitere.

Klasse E Diese Klasse ist für zukünftige Anwendungen rerserviert. Solche Adressen beginnen mit 11110.

Die Gründe für den Aufbau der Adressen sind, dass

- durch diese Einteilung die Einträge in den Routern minimiert werden können. Daher kommt es auch, dass den einzelnen geographischen Regionen auf der Erde verschiedene IP Adressbereiche zugeordnet sind. Europa sind z.B. für die Klasse C Netze die Adressbereiche 194.0.0.0 bis 195.255.255.255 zugeordnet.
- Router lediglich durch Überprüfung der ersten 5 Bits die entsprechende Klasse eruieren können. Außerdem ist der Zugriff auf den Netzanteil bzw. den Hostanteil sehr einfach, da diese Teile immer an Bytegrenzen ausgerichtet sind.
- die Einteilung der Klassen auf diese Art und Weise so vorgenommen wurde, dass sehr große Organisationen Klasse A Netze zugeteilt bekommen und sehr kleine Organisationen Klasse C Netze. Mittleren Organisationen, d.h. alle die mehr als 254 Hosts benötigen oder die erwarten, dass sie mehr als 254 IP Adressen benötigen, wurden die Klasse B Netze zugedacht, wodurch in den 80er Jahre fast nur Klasse B Netze verteilt wurden. Das verursachte Ende der 80er Jahre die Voraussage: „Mitte der 90er Jahre gehen die Klasse B Netze aus". Daher wurde die Zuteilung der Netze neu geregelt. Es wurden Regeln aufgestellt, die genau festlegen wann ein Klasse A Netz vergeben

wird oder alternativ mehrere aufeinanderfolgende Klasse B Netze bzw. ein Klasse B Netz oder alternativ mehrere aufeinanderfolgende Klasse C Netze. Abgesehen von diesen organisatorischen Maßnahmen sind auch noch technische Maßnahmen eingeführt worden, die später noch beschrieben werden.

8.1.2. Spezielle IP Adressen

Netzanteil und Hostanteil können spezielle Muster aufweisen. Lauter 0en und lauter 1en werden besonders interpretiert:

- Sind alle Bits im Hostanteil 0, dann wird dies als „dieser Host" interpretiert.
- Sind alle Bits im Netzanteil 0, dann wird dies als „dieses Netz" interpretiert.
- Sind alle Bits im Hostanteil 1, dann wird dies als „alle Hosts" interpretiert.
- Sind alle Bits im Netzanteil 1, dann wird dies „alle Netze" interpretiert.

Daraus ergeben sich 6 verschiedene sinnvolle Fälle:

Netzanteil	Hostanteil	Bedeutung
Netz Id	Host Id	Normale Bedeutung. Ein Beispiel ist die Klasse A Adresse 10.2.9.14.
Netz Id	alle 0	Prinzipiell handelt es sich um diesen Host (wenn z.B. der Host sein IP Adresse noch nicht kennt. Außerdem wird so auch das Netz bezeichnet. Ein Beispiel ist das Klasse A Netz 10.0.0.0.
alle 0	Host Id	Mit dieser Bezeichnung wird ein spezieller Host im aktuellen Netz bezeichnet. So eine Adresse kann verwendet werden, wenn ein Host noch nicht die Netzadresse kennt oder wenn die Netzinformation nicht relevant ist, dann sendet er ein Paket mit der Hostadresse und dem Netzanteil bestehend aus lauter 0en. Der angesprochene Host auf diesem Netz wird mit einem Paket mit ausgefülltem Netzanteil antworten. Ein Beispiel einer solchen Klasse A Adresse könnte sein: 0.2.9.14.
alle 0	alle 0	Diese Adresse bezeichnet den eigenen Host. Meist wird diese verwendet, wenn die eigene Adresse noch nicht bekannt ist (z.B. bei DHCP). Allerdings wird diese Adresse auch bei multihomed Hosts (wie z.B. Router) verwendet, um eine beliebige Adresse zu spezifizieren. Die einzige Adresse gemäß diesem Muster ist 0.0.0.0.
Netz Id	alle 1	Es werden alle Hosts in dem angegebenen Netz angesprochen. D.h. es handelt sich um eine Broadcast-Adressen. Ein Beispiel für eine Klasse A Broadcast-Adresse: 10.255.255.255.

alle 1	alle 1	Die Bedeutung ist: „alle Hosts von allen Netzen". D.h. diese Adresse gibt einen globalen Broadcast an. Da es jedoch praktisch unmöglich ist an alle Hosts im Internet zu senden, ist die Bedeutung dieser Adresse eines Broadcasts an das direkt angeschlossene Netz! Die einzige Adresse gemäß diesem Muster ist 255.255.255.255. Semantisch äquivalent wäre noch die Adresse mit einem Netzanteil mit 0en und einem Hostanteil mit 1en. Das wird jedoch nicht verwendet, die obige Lösung außerdem den Vorteil hat, nicht zwischen Netz- und Hostanteil unterscheiden zu müssen.

Eine Kombination hat keine sinnvolle Bedeutung und wird deshalb nicht verwendet: Im Netzanteil alles 1en und im Hostanteil eine Host Id. Das würde bedeuten: alle Hosts einer bestimmten Host Id in allen Netzen. Das ist wieder praktisch unmöglich und wird daher nicht verwendet.

8.1.3. Reservierte Adressen

Verschiedenen Adressen wurden, abgesehen von den klassen-basierten Standardadressen eine bestimmte Bedeutung zugewiesen und reserviert:

- 127.0.0.0 ist ein spezielles Netz, das für den lokalen IP Verkehr vorgesehen ist und als Loopback Netz bezeichnet wird. Adressen in diesem Netz sind nicht mit einem physikalischen Interface (wie z.B. einem Ethernet-Interface) verbunden. Meistens wird die Adresse 127.0.0.1 einer virtuellen Schnittstelle, dem sogenannten „loopback interface" zugeordnet. Jedes Paket, das an diese Schnittstelle gesendet wird, wird zurückgeliefert als ob es von einem anderen System kommen würde. Damit eignet sich diese Schnittstelle sehr gut für Tests oder lokale Anwendungen, die keinen Anschluss an das Internet benötigen.
- Private Adressen: Benötigt man Adressen für ein lokales Netz, die nicht global verfügbar sein müssen, dann kann man Adressen aus bestimmten Adressbereichen verwenden, die als private Adressen reserviert worden sind. Pakete mit solchen Adressen werden von Routern im Internet *nicht* weitergeleitet. Damit sind solche Adressen ein Mechanismus, um die Situation der Adressknappheit im Internet zu verbessern. Folgende Adressbereiche wurden als privat festgelegt:

Klasse	von	bis	Bemerkung
A	10.0.0.0	10.255.255.255	1 Klasse A Netz
B	172.16.0.0	172.31.255.255	16 aufeinanderfolgende Klasse B Netze.
C	192.168.0.0	192.168.255.255	256 aufeinanderfolgende Klasse C Netze.

Will man so ein privates Netz (Intranet), das mit solchen Adressen betrieben wird an das Internet anschließen, dann kann man dies durch

  - ein anwendungsspezifisches Gateway erreichen (siehe Abschnitt 4.1.2 auf Seite 54) erreichen.
  - ein anwendungsneutralen Gateway wie z.B. SOCKS erreichen.
  - NAT (siehe Abschnitt 11 auf Seite 117) erzielen.

- Zur automatischen Zuweisung einer privaten Adresse im Zusammenhang mit DHCP steht ein spezielles Klasse B Netz zur Verfügung: 169.254.0.0 bis 169.254.255.255. Eine Erläuterung dazu findet sich im Abschnitt 10.1 auf Seite 115.
- Sonstige reservierte Adressbereiche, sind z.B.:

  - Klasse A Netz: 0.0.0.0 bis 0.255.255.255 (siehe Abschnitt 8.1.1 auf Seite 89)
  - Klasse B Netz: 128.0.0.0 bis 128.0.255.255
  - Klasse C Netz: 192.0.0.0 bis 192.0.0.255

  An sich muss man sich um diese reservierten Klasse B und Klasse C Netze nicht kümmern, da diese Adressen einfach nicht vergeben werden. Weiters gibt es noch einige andere Netze, die ebenfalls reserviert sind.

8.1.4. Subnetting

Subnetting bezeichnet die Bildung von Teilnetzen und ist aus mehreren Gründen notwendig:

- Aus organisatorischen Gründen, wie z.B. abteilungsweise Gliederung der Teilnetze.
- Aus geographischen Gründen, wenn eine große Distanz zwischen den Hosts eine Einführung von Teilnetzen nahelegt bzw. erfordert.
- Wenn ein neuer Typ von physischem Netz installiert wird.
- Wenn durch Hinzufügen weiterer Hosts eine Teilung des Netzes notwendig wird.

Nur durch Verwendung klassenbasierter Standardadressen mussten immer entsprechende Adressbereiche angefordert werden. Dadurch entstehen gerade in diesem Zusammenhang (meist bei größeren Organisationen) etliche Probleme:

- Die Routertabellen wachsen explosionsartig an. Große Routertabellen bewirken jedoch einerseits einen vergrößerten Wartungsaufwand bei den Routern als auch eine verminderte Leistung der Router.
- Jedes Mal, wenn mehr Adressen in einem Netz benötigt werden, muss ein neuer Adressbereich angefordert werden, obwohl noch Adressen in den schon vergebenen Netzen zur Verfügung wären.
- Jede Änderung der internen Netzstruktur hatte Auswirkungen auf die Adressen!

Um diese Probleme zu lösen, wurde das Konzept des Subnetting eingeführt. Das grundlegende Prinzip ist, dass Subnetting lokal durchgeführt wird und das gesamte Netz von außerhalb der Organisation als ein einziges Netz gesehen wird. Das wird dadurch erreicht, dass das zweistufige Konzept mit Netz-Id und Host-Id auf ein dreistufiges Konzept mit Netz-Id, Subnetz-Id und Host-Id erweitert wird. Diese Erweiterung wird dadurch erreicht, dass der ursprüngliche Hostanteil geteilt wird. Lokal ist subnetting deshalb, da die Subnetzmaske von außerhalb nicht gesehen wird.

Damit können die oben genannten Probleme gelöst werden. Konkret ergeben sich damit folgende Vorteile:

- Die Routertabellen vergrößern sich nicht.
- Es müssen viel seltener neue Adressen angefordert werden.
- Die Flexibilität erhöht sich, da Änderungen an der Netzstruktur keine Änderungen der Adressen mit sich ziehen.
- Die Netze können besser auf die physikalischen Gegebenheiten abgestimmt werden.
- Die interne Netzstruktur ist von außen nicht sichtbar. Das ist aus sicherheitstechnischen Überlegungenen wünschenswert.

Bei der Subnetzmaske handelt es sich um eine 32 Bit Maske, wobei jede 1 in der Subnetzmaske den Netzanteil angibt, während eine 0 in der Subnetzmaske den Hostanteil angibt. Für die ursprüngliche Netz-Id stehen immer eine 1en. Aus praktischen Gründen handelt es sich immer um zusammenhängende Bereiche (einfachere Implementierung im Router), auch wenn es vom Standard nicht unbedingt gefordert wird.

Damit kann man die sogenannten Default-Subnetzmasken für die klassenbasierten Adressbereiche angeben:

- Klasse A Netze haben die Default-Subnetzmaske 255.0.0.0
- Klasse B Netze haben die Default-Subnetzmaske 255.255.0.0
- Klasse C Netze haben die Default-Subnetzmaske 255.255.255.0

Diese Default-Subnetzmasken bewirken natürlich eigentlich kein Subnetting. Allerdings bewirkt die Subnetzmaske 255.255.255.0 z.B. bei einem Klasse B Netz (also eine Subnetz-ID mit der Länge von 8 Bits) sehr wohl Subnetting: 192.168.0.0/255.255.255.0 bezeichnet den Adressbereich für Hosts von 192.168.0.1 bis 192.168.0.254. D.h., mit dem Klasse B Netz 192.168.0.0 und der Subnetmaske 255.255.255.0 ergeben sich 256 Teilnetze mit je 254 Hosts.

Das Beispiel in der Abbildung 8.1 auf der nächsten Seite zeigt die Aufteilung des Klasse B Netzes 172.16.0.0 mit 65534 Hosts mittels einer 5 Bit langen Subnetz-Id in 32 Subnetze mit je 2046 Hosts.

Die Subnetz-Id wird meistens ebenfalls in der dezimalen Form durch Punkte getrennt angegeben. Für das Beipiel in Abbildung 8.1 lautet die Subnetzmaske 11111111.11111111.11111000.00000000. Das dezimale Äquivalent und damit die Form in der die Subnetzmaske angeführt wird lautet daher 255.255.248.0. Das erste Subnetz wird dann folgendermaßen angegeben: 172.168.0.0/255.255.248.0. Unter der gültigen Voraussetzung, dass nur zusammenhängende Bereiche als Subnetzmaske zugelassen sind, kann man

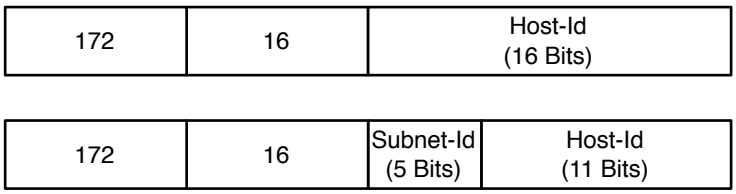


Abbildung 8.1.: Beispiel einer Subnetzbildung eines Klasse B Netzes

die Subnetzmaske auch durch Angabe der Anzahl der 1en vom linken Rand angeben: 172.168.0.0/21. Das nächste Subnetz hat dann folgende Adresse: 172.168.8.0/255.255.248.0!

Die bisher behandelte Form von Subnetting wird auch als „static subnetting" bezeichnet, da alle Teilnetze immer die gleiche Größe haben. Dies ist jedoch oft ungünstig! Betrachten wir folgendes Beispiel: Eine Organisation bekommt das Klasse C Netzwerk 193.170.149.0 zugeordnet. Nehmen wir weiters an, dass aus organisatorischen und technischen Gründen der Bedarf an folgenden Netzen besteht: 4 Netze mit je 10 Hosts, 1 Netz mit 50 Hosts und ein Netz mit 100 Hosts. In Summe ergibt das 100 Hosts. Insgesamt stehen bei einem Klasse C Netz 254 IP Adressen für Hosts zur Verfügung. Allerdings ergibt sich folgendes Problem: es werden insgesamt 6 Teilnetze benötigt. Also muss die Subnet-ID die Länge 3 haben. Das wiederum bedeutet, dass insgesamt 5 Bits für die Hosts je Teilnetz zur Verfügung stehen. Das ergibt eine maximale Anzahl an Hosts von 30 je Teilnetz. Man sieht, dass eine Einteilung in Teilnetze nicht möglich ist. Die Lösung dafür liegt in VLSM (siehe Abschnitt 8.1.5).

Damit Subnetze eingeführt werden können, müssen die Router innerhalb der Organisation beim Weiterleiten der Pakete (siehe Abschnitt 8.1.6) und auch die Routingprotokolle (wenn nicht statisches Routing verwendet wird, siehe Abschnitt 10.3 auf Seite 115) angepasst werden.

8.1.5. VLSM

Um unterschiedlich große Teilnetze einrichten zu können, wurde VLSM (variable length subnet masking) eingeführt. Darunter versteht man das Prinzip, dass Subnetze weiter unterteilt werden. Jedes Subnetz hat seine eigene Subnetzmaske.

Gehen wir wieder von dem Beispiel aus dem vorhergehenden Abschnitt aus: Eine Organisation bekommt das Klasse C Netz 193.170.149.0 zugeordnet und benötigt 4 Netze je 10 Hosts, 1 Netz mit 50 Hosts und ein Netz mit 100 Hosts. Die resultierende Aufteilung in entsprechende Subnetze ist in Abbildung 8.2 auf der nächsten Seite zu sehen. In dieser Abbildung ist jeweils die Subnetznummer und die Subnetzmaske angegeben.

8.1.6. Weiterleiten

Weiterleiten in IP funktioniert im Prinzip wie in Abschnitt 4.2.3 auf Seite 63 beschrieben. In IP kommt es durch die Verwendung von Subnetzmasken es zu folgendem prinzipiellen Ablauf im Router:

1. Wenn es sich um das Zielsystem handelt, dann stopp. D.h. der Router ist das Ziel.

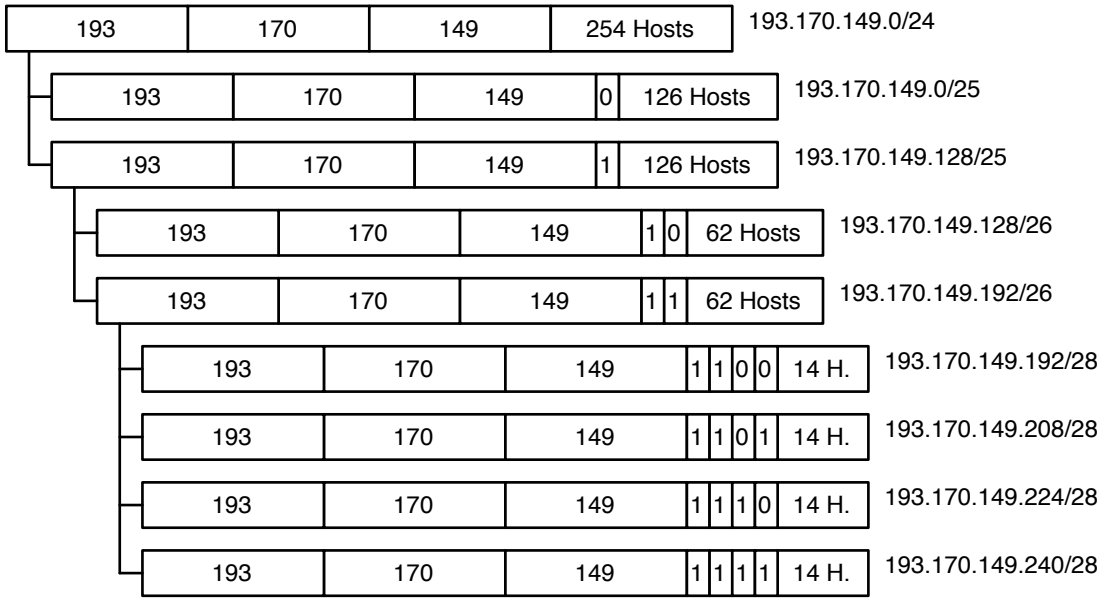


Abbildung 8.2.: Beispiel für Anwendung von VLSM

2. Für jeden Eintrag (Subnetznummer, Subnetmaske, Nächster Hop) der Weiterleitungstabelle
   a) D1 = Zieladresse & Subnetmaske
   b) Wenn D1 == Subnetznummer
      i. dann: Wenn Nächster Hop ist ein Interface
         A. dann: Paket an diesem Interface ausliefern
         B. anderenfalls: Paket an dem Interface ausliefern, das zu diesem Router führt
3. Wenn kein Router gefunden, dann an den Default-Router weitersenden. Ist kein Defaultrouter definiert, dann ICMP Fehlernachricht „network unreachable" schicken (siehe Abschnitt 8.4 auf Seite 100).

Es handelt sich hier um den prinzipiellen Ablauf und nicht um eine Beschreibung einer konkreten Implementierung, da es nicht sehr sinnvoll ist, jedes Mal eine lineare Suche in der Weiterleitungstabelle durchzuführen bzw. die bitweise UND Verknüpfung immer neu zu berechnen, auch wenn sich die Subnetmaske nicht geändert hat.

Der Default Router, an den alle IP Pakete weitergeleitet werden, für die keine Route in der Weiterleitungstabelle eingetragen ist wird oft auch Default Gateway genannt, obwohl es sich im strengen Sinne um kein Gateway handelt.

8.1.7. CIDR

Der Erschöpfung der IP Adressen wird durch Subnetting nicht weiter vorgebeugt, da eine Organisation, die z.B. 2 Adressen benötigt, ein Klasse C Netzwerk zugewiesen bekommt.

D.h. die „Effizienz" beträgt 2/254*100% also 0.78%! Nehmen wir als weiteres Beispiel an, dass eine Organisation 256 Hosts an das Internet anschließen muss. Es wird also ein Klasse B Netzwerk benötigt. D.h. die „Effizienz" beträgt 256/65534*100% also lediglich 0.39%!

Benötigt eine Organisation mehr als 254 Adressen aber nicht nahezu 65534 Adressen, dann bekommt sie wahrscheinlich anstatt eines Klasse B Netzes mehrere Klasse C Netze zugewiesen, da freie B Netze nur in beschränkten Umfang vorhanden sind bzw. die Effizienz der Adressvergabe verbessert wird. Damit wachsen jedoch automatisch die Routertabellen im Internet an, wodurch es zu Performanceinbußen kommt. Ein weiteres Problem ist, dass Routertabellen nur eine beschränkte Größe haben.

Trotz der Vorteile des Subnetting, private Netze effizient zu partitionieren, bleibt die ineffiziente Adressvergabe weiters bestehen. Gelöst werden diese Probleme mit CIDR (engl. classless inter-domain routing). CIDR wird auch supernetting genannt, da es sich im Prinzip, um eine analoge Lösung wie das subnetting handelt, allerdings auf Internet-Ebene.

CIDR versucht die Anzahl der Routingeinträge in den Routern des Internet zu minimieren. Das wird so gemacht, dass eine Organisation mehrere *aufeinanderfolgende* z.B. Klasse C Adressbereiche zugewiesen bekommt. Diese aufeinanderfolgenden Routen werden zu einer Route mit Hilfe einer Netzmaske (meist angegeben durch Suffixdarstellung) aggregiert.

Nehmen wir an, dass eine Organisation einen Bedarf an 16 Klasse C Netzen hat und diese Organisation deshalb einen zusammenhängenden Bereich von 192.4.16.0/24 bis 192.4.31.0/24 zugewiesen bekommt. Man erkennt, dass die oberen 20 Bits aller dieser Adressen gleich ist, nämlich 11000000 00000100 0001. Das bedeutet, dass dadurch eine 20 Bit lange effektive Netznummer erzeugt worden ist. D.h. das resultierende Netz lässt sich als 192.4.16.0/20 angeben. Man sieht, dass die Angabe in gleicher Weise wie beim Subnetting vorgenommen wird. Das Ergebnis ist, dass nur mehr ein Routereintrag für diese Organisation vorgenommen werden muss.

Dieses Verfahren lässt sich auch über mehrere Organisationen kaskadieren: Werden die Adressbereiche mehrerer Organisationen ebenfalls aufeinanderfolgend vergeben, können diese Adressbereiche ebenfalls zu einem Eintrag aggregiert werden! Damit sinkt die Anzahl der Routereinträge weiter.

Damit wird mit CIDR sowohl die Effizienz der Vergabe verbessert als auch die Anzahl der Routereinträge minimiert.

Ähnlich wie bei Subnetting müssen die Routingprotokolle (siehe Abschnitt 10.3 auf Seite 115) ebenfalls mit dieser Form der Adressangabe umgehen können.

8.2. Aufbau eines Datagram

Der Aufbau eines IPv4 Datagram ist in Abbildung 8.3 auf der nächsten Seite zu sehen.

Version Enthält entweder den Wert 4 für IPv4 oder den Wert 6 für IPv6. An Hand dieser 4 Bits kann jede Implementierung des TCP/IP Protokollstacks sofort erkennen, um welches IP Paket es sich handelt. Das ist deshalb wichtig, da der Aufbau eines IPv6 Datagrams ansonsten verschieden ist!

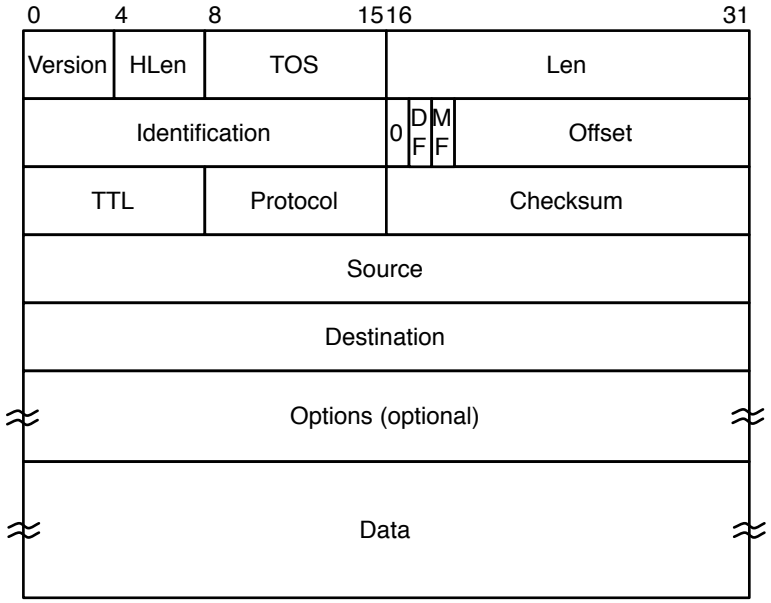


Abbildung 8.3.: IP Datagram

HLen Gibt die Anzahl der 32 Bitworte an, die ein Header inklusive Optionen lang ist. Der Optionenbereich ist variabel, daher ist das Feld HLen notwendig.

TOS Type-of-service Feld wurde schon für mehrere Funktionen eingesetzt. Seit 1998 wird es lt. RFC 2474 für DS (differential services) verwendet. DS ist ein bestimmtes Quality of Service (QoS) Verfahren zur Priorisierung von IP Paketen.

Len Gibt die Gesamtlänge eines IP Datagrams in Byte an. Da hierfür 16 Bit zur Verfügung stehen, ist die maximale Größe eine IP Datagrams auf 65535 Bytes beschränkt.

Identification Das Feld Identification wird im Zuge der Fragmentierung verwendet und stellt eine eindeutige Identifikation zur Verfügung (siehe Abschnitt 8.3 auf der nächsten Seite).

Flags Zwischen Identifcation und Offset sind 3 Flags untergebracht:

- 0 Das erste Bit muss immer ein 0 Bit sein.
- DF „Do not fragment" wird ebenfalls im Zuge der Fragmentierung verwendet (siehe Abschnitt 8.3 auf der nächsten Seite).
- MF „More fragments" wird ebenfalls im Zuge der Fragmentierung verwendet (siehe Abschnitt 8.3 auf der nächsten Seite).

Offset Offset wird ebenfalls im Zuge der Fragmentierung verwendet (siehe Abschnitt 8.3 auf der nächsten Seite).

TTL „Time-to-live" gibt die Lebensdauer des Paketes an und wird zur Verhinderung endlos kreisender Pakete eingesetzt. Der sendende Host setzt TTL auf einen Anfangswert und jede Station, die das Datagram weiterreicht vermindert den Zähler um eins. Hat TTL den Wert 0 erreicht, dann wird das Paket verworfen. Üblich sind die Anfangswerte von 64 und 128.

Protocol Dieses Feld enthält eine eindeutige Zahl, die das verwendete Transportprotokoll angibt. Diese Nummern werden von der IANA (Internet Assigned Numbers Authority) verwaltet. Beispiele: 1 (ICMP), 6 (TCP), 17 (UDP).

Checksum Das ist eine Prüfsumme über den gesamten Header. Die Idee ist, dass die Korrektheit der Daten von einem übergeordneten Protokoll überprüft werden muss und dadurch die Berechnung der Prüfsumme schnell durchgeführt werden kann, sodass die Verzögerung bei der Weiterleitung in einem Router gering gehalten werden kann. Das Verfahren der Berechnung der Checksumme ist in Abschnitt **??** auf Seite ?? beschrieben.

Source Die IP-Adresse des Senders.

Destination Die IP-Adresse des Empfängers.

Options Das Options-Feld ist dafür gedacht zusätzliche variable Informationen mitzuübertragen, für die es nicht notwendig ist einen festen Platz in der Header-Struktur vorzusehen. Es handelt sich hierbei um Informationen zum Routing, Security, Zeitstempel,... Eventuell muss mit Nullen aufgefüllt werden, damit der Header an einer 32 Bitgrenze endet.

8.3. Fragmentierung

Die Anpassung der Paketlänge an eine kleinere maximale Paketgröße der untenliegenden Schicht, also die Segmentierung (siehe Abschnitt 2.4.4 auf Seite 31), wird in IP Fragmentierung genannt. Wie im Abschnitt 5.1.2 auf Seite 78 beschrieben besteht ein Datenblock in einem Ethernet Frame aus maximal 1500 Bytes während ein Paket in einem FDDI Netzwerk maximal 4500 Bytes beinhalten kann. Wird ein IP Paket über mehrere Netze mit unterschiedlichen Technologien versendet, besteht entweder die Möglichkeit sicherzustellen, dass die IP Pakete maximal die kleinste Größe aller Technologien annehmen, oder die Pakete jeweils an die verschiedenen Technologien durch Segmentierung und Reassemblierung anzupassen.

IPv4 unterstützt den Ansatz mittels Segmentierung und Reassemblierung. Das zentrale Konzept ist, dass es bei jeder Technologie eine maximale Übertragungseinheit (engl. max transmission unit oder MTU) gibt, die die maximale Paketgröße angibt, die über ein Netz übertragen werden kann. Im Fall von Ethernet beträgt dieser Wert 1500. In IP müssen die Router bzw. die physische Netztechnologien eine minimale MTU von zumindest 576 Bytes unterstützen.

Das Header-Feld DF zeigt an, dass dieses IP Paket nicht fragmentiert werden darf. Kommt ein IP Paket zu einen Router oder einem Netz, das eine Fragmentierung dieses Paketes erfordern würde, dann wird eine ICMP Nachricht (siehe 8.4 auf der nächsten Seite) vom Typ 3 mit Code 5 (Fragmentation needed but the Do Not Fragment bit was set) gesendet.

Das Zusammenspiel der Identification, des Flags MF und des Offsets ist in folgenden Beispielen zu sehen, wenn wir von einer Länge der zu übertragenden Daten von 1400 ausgehen:

- nicht fragmentiertes Paket mit: Identification = x; MF = 0; Offset = 0; Data (1400 Bytes)
- fragmentierte Pakete bei einer MTU mit 532 Bytes mit:

1. Identification = x; MF = 1; Offset = 0; Data (512 Bytes)
2. Identification = x; MF = 1; Offset = 64; Data (512 Bytes)
3. Identification = x; MF = 0; Offset = 128; Data (376 Bytes)

Das Prinzip der Fragmentierung ist folgendes:

- Das Identifikationsfeld bezeichnet ein IP Datagram, wird vom Sender frei gewählt und ist für alle Fragmente gleich.
- Das Flag MF ist gesetzt, wenn weitere Fragmente folgen.
- Der Offset gibt den Offset des Fragmentes relativ zum gesamten Datenblock in jeweils 8 Byte an (512 / 8 = 64). Die Einheit von 8 Bytes liegt darin begründet, das nur 13 Bits für den Offset zur Verfügung stehen und die maximale Größe eine IP Datagrams 65535 Bytes ist.
- Das Datenfeld beinhaltet 512 Bytes, da 20 Bytes für den IP Header anfallen.
- Die Fragmente werden vom Empfänger im Zuge der Reassemblierung wieder zusammengesetzt. Der Empfänger kann dies deshalb durchführen, da er alle zusammengehörigen Fragmente an der gleichen Identifikation erkennt. Geht eines der Fragmente verloren, dann wird das gesamte IP Datagram samt allen bisher empfangenen Fragmente verworfen. D.h. IP versucht nicht, fehlende Fragmente wieder zu beschaffen. Die Reassemblierung erfolgt am empfangenden Host und nicht an einem Router.

8.4. ICMP

ICMP (engl. internet control message protocol) ist ein Hilfsprotokoll, dessen Funktion primär die Unterstützung des verbindungslosen und nicht fehlergesicherten IP mit zusätzlichen Status-, Steuer- und Fehlerinformationsmeldungen ist. Neben IP können auch höher geschichtete Protokoll wie TCP, Routingprotokolle oder auch Anwendungen die Information von ICMP Nachrichten auswerten.

Die bekannteste Anwendung von ICMP ist die Verwendung der ICMP Echo Request bzw. der ICMP Echo Reply Nachricht, die in der ping Anwendung eingesetzt wird.

Jede ICMP Nachricht besteht aus einem Typ und einem Code, einer Prüfsumme und einem Inhalt, der von Typ und Code abhängig ist. Jede dieser Nachrichten ist entweder eine Query Nachricht oder eine Fehler Nachricht.

Beispielhaft sind die wichtigsten ICMP Nachrichtentypen angeführt:

- Echo Request (Typ 8, Code 0) und Echo Reply (Typ 0, Code 0). Query-Nachricht.
- Ziel nicht erreichbar (Typ 3, Code 0-15). Fehler-Nachricht. Dieser Nachrichtentyp inkludiert die Fehlernachrichten für „Netzwerk nicht erreichbar", „Host nicht erreichbar", „Protokoll nicht erreichbar", „Port nicht erreichbar", „Fragmentierung erforderlich, aber DF Bit gesetzt", „Source Route fehlerhaft", „Zielnetzwerk unbekannt",...
- Quelle unterdrücken (engl. source quench) (Typ 4, Code 0). Fehler-Nachricht. Dieser Nachrichtentyp stellt die einzige Möglichkeit von IP dar, einem Sender mitzuteilen,

dass er seine Datenrate reduzieren soll. Der Empfänger sendet so lange solche Nachrichten an den Sender solange dieser die Datenrate reduzieren soll. Empfängt der Sender keine weiteren ICMP Nachrichten, dann kann er seine Datenrate wieder erhöhen. D.h. in dieser Nachricht gibt es keine explizite Möglichkeit die Datenrate anzugeben bzw. anzugeben wann wieder eine höhere Datenrate erlaubt ist. Es ist im Standard explizit festgehalten, dass im Falle eines Verwerfens eines IP-Datagrams ein ICMP source quench Nachricht gesendet werden kann (aber nicht muss). Auf jeden Fall muss der Empfang eines solchen ICMP Paketes an die Transportschicht weitergegeben werden. Im Falle von TCP sollte TCP die Senderate reduzieren und im Falle von UDP sollte die Anwendung darauf entsprechend reagieren.

- Redirect (Typ 5, Code 0-3). Fehler-Nachricht. Dient der Signalisierung eines Routers an einen Sender, dass ein besserer Router für die eingegangenen Pakete existiert. Diese stellen ein Sicherheitsrisiko, da gefälschte ICMP Pakete den Sender veranlassen seine Paket an einen anderen Router zu senden.
- Router-Advertisement (Typ 9, Code 0) und Router-Anforderung (Typ 10, Code 0). Dienen dazu, um Router in einem lokalen Netzwerk zu erkennen (Sicherheitsrisiko).

8.5. ARP

Mittels Weiterleiten kommt ein IP Datagram zum richtigen Netz. Die offene Frage ist jetzt, wie das Paket zum richtigen Host kommt. Jedes Gerät ist mit seiner MAC Adresse eindeutig zu adressieren. Es ist notwendig die MAC Adresse zu kennen, um einen Host in einem Segment zu adressieren.

ARP (engl. address resolution protocol) ist ein Protokoll, das die MAC Adresse eines Hosts ermittelt. Dazu macht es sich die Eigenschaft zu Nutze, dass Ethernet und andere LAN Technologien ebenfalls über eine Broadcast Eigenschaft verfügen. Jeder Host merkt sich alle ihm schon bekannten Zuordnungen von IP Adresse zu MAC Adresse in einem Cache (ARP Cache oder ARP Tabelle genannt). Soll ein Paket einem Host in einem Netz zugestellt werden, dann wird zuerst dieser Cache konsultiert. Ist eine Zuordnung dort vorhanden, dann wird sie verwendet, um das Paket zuzustellen. Ist keine Zuordnung für diese IP Adresse im Cache vorhanden, dann wird ein Broadcast Frame an alle Hosts in diesem Segment gesendet, der die fragliche IP Adresse enthält. Der gesuchte Host antwortet mit seiner IP Adresse, worauf dem ARP Cache die neue Zuordnung hinzugefügt werden kann.

Diejenigen Stationen, die ebenfalls den Broadcast empfangen, können diese Zuordnung (Sender IP, Sender MAC) ebenfalls lernen. Ist so eine Zuordnung schon im ARP Cache enthalten wird sie lediglich aufgefrischt und der Timer zurückgesetzt. Läuft der Timer ab, dann wird die Zuordnung aus dem Cache gelöscht.

ARP wurde nicht unter dem Gesichtspunkt der Sicherheit entworfen und ist deshalb besonders anfällig: Unter ARP-Spoofing versteht man das gezielte Versenden gefälschter ARP Pakete, die dazu benutzt werden, den Datenverkehr zwischen zwei Hosts in einem Netz abzuhören oder zu manipulieren.

RARP (engl. reverse address resolution protocol) ist ein Protokoll, das die inverese Funktionalität aufweist und die IP Adressse zu einer MAC Adresse finden soll. Es wird aller-

dings nicht mehr eingesetzt, da es durch BOOTP und in weiterer Folge von DHCP abgelöst worden ist. Die Gründe liegen darin, dass man mittels RARP nur die IP Adresse, aber keine zusätzlichen Informationen wie z.B. die Netzmaske oder den Default-Router ermitteln kann.

9. Transportprotokolle TCP und UDP

9.1. Adressierung

Die Adresserierung in IP wird über die IP Adressen vorgenommen, die global eindeutig sein müssen. Auf der Transportebene wird eine weitere Form der Adressierung benötigt, da es sich um eine Ende-zu-Ende Kommunikation auf Prozessebene handelt. Das bedeutet, dass über die Transportschicht direkt zwei Prozesse miteinander verbunden sind und miteinander kommunizieren.

In diesem Sinne muss es eine Möglichkeit geben, einen Prozess auf einem Host zu identifizieren. Das Konzept, das TCP/IP auf dieser Ebene eingeführt hat sind die Ports. Es handelt sich dabei um einen nummerischen Bezeichner, der Werte im Bereich von 0 bis 65535 annehmen kann.

In diesem Zusammenhang führt TCP das Konzept eines Kommunikationsendpunktes ein, der als „socket" bezeichnet wird und sich aus der IP Adresse und dem Port zusammensetzt. Zwei Sockets, nämlich der Sender-Socket und der Empfänger-Socket identifizieren gemeinsam mit der Angabe des verwendeten Transportprotokolls eindeutig eine Kommunikation in Internet! Ein lokaler Socket kann gleichzeitig in mehreren Kommunikationsverbindungen verwendet werden.

Für jede Nachricht, die von einem Senderprozess (also eine laufende Anwendung) an einen Empfängerprozess (ebenfalls eine laufende Anwendung) gesendet wird, gibt es einen Sender-Socket und einen Empfänger-Socket. In der Regel ist es so, dass ein Prozess an einem Empfänger-Socket lauscht und Nachrichten an diesem empfangen kann. Der Senderprozess schickt eine Nachricht mittels eines Sender-Socket unter Angabe des Transportprotokolls über den Empfänger-Socket an den Empfängerprozess. Dieser Empfangsprozess erbringt in der Regel einen Dienst und wird auf die eingegangenen Nachrichten mittels Antworten reagieren.

Dieses Senden der Nachrichten verwendet das unterliegende IP. Der Sendeprozess übergibt die Nachricht an die Transportschicht. Die Transportschicht packt die Nachricht in ein TCP oder UDP Paket und reicht dieses an IP weiter. Beim Empfänger werden die Pakete wieder entpackt und an den richtigen Prozess weitergegeben. Dies wird, analog zu den Signalen, als Multiplexing bzw. Demultiplexing bezeichnet. Die zentrale Information, die zum Multiplexen bzw. Demultiplexen verwendet wird sind die Ports.

Portnummern von 0 bis 1023 werden als sogenannte „well known ports" bezeichnet und von der IANA verwaltet und für Anwendungen reserviert. Unter Unix werden diese Ports insoferne speziell behandelt als das ein passives Öffnen (siehe 9.3.2 auf Seite 109) nur Prozessen mit der Benutzernummer 0 erlaubt wird. Bestimmte Ports sind bestimmten Diensten zugeordnet, wichtige Dienste sind:

Port#	Protokoll	Schlüsselwort	Beschreibung
20	tcp	ftp-data	file transfer protocol (Datenkanal)
21	tcp	ftp	file transfer protocol (Steuerkanal)
22	tcp	ssh	secure socket shell
25	tcp	smtp	simple mail transfer protocol
53	tcp+udp	domain	domain name system
67	udp	dhcp	dynamic host configuration protocol (server)
68	udp	dhcp	dynamic host configuration protocol (server)
80	tcp	http	hyper text transfer protocol
110	tcp	pop	post office protocol
113	tcp	ident	ident protocol
119	tcp	nntp	network news transfer protocol
123	tcp	ntp	network time protocol
143	tcp	imap	internet mail access protocol
161	udp	snmp	simple network management protocol
389	tcp	ldap	lightweight directory access protocol
443	tcp	https	http over SSL
514	tcp	syslog	syslog

9.2. UDP

UDP (engl. user datagram protocol) ist das einfachere Transportprotokoll, das TCP/IP zur Verfügung stellt. Es wurde 1980 im RFC 768 publiziert und seit dem nicht verändert.

Das Entwurfsziel war ein verbindungsloses, unzuverlässiges Ende-zu-Ende Transportprotokoll zur Verfügung zu stellen, das einfach ist und eine schnelle Kommunikation erlaubt. Es ist charakterisiert durch folgende Eigenschaften:

- Es ist unzuverlässig: Es gibt keine Garantie ob Daten angekommen sind, es wird nicht erkannt, ob Daten verloren gegangen sind und es gibt keine Garantie, dass die Daten in der Reihenfolge ankommen, in der sie abgesendet worden sind.
- Es gibt keinen Mechanismus zur Flusskontrolle.
- Es erfolgt eine einfache Überprüfung auf die Korrektheit der übertragenen Daten

Man sieht, dass UDP an sich nur zwei zusätzliche Eigenschaften gegenüber IP zur Verfügung stellt: Multiplexing/Demultiplexing und Überprüfung auf Korrektheit der Daten.

UDP eignet sich gut für Audio- und Videostreams und außerdem wird es auch bei einigen wichtigen Protokollen, wie DNS, SNMP und DHCP eingesetzt.

9.2.1. Aufbau eines Datagrams

Der Aufbau eines UDP Datagram ist in Abbildung 9.1 auf der nächsten Seite zu sehen.

Source Port 0–65535

Destination Port 0–65535

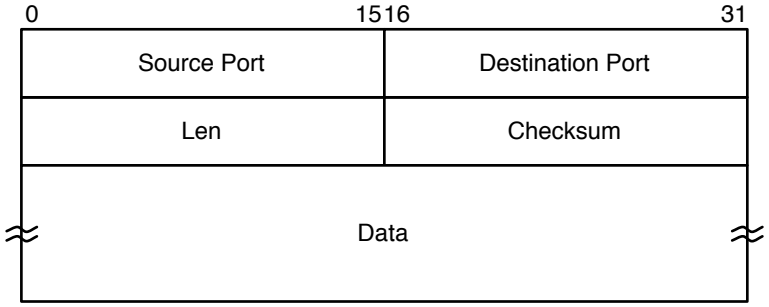


Abbildung 9.1.: Aufbau eine UDP Datagrams

Len Länge des vollständigen UDP Datagrams inklusive Header und Datenfeld in Bytes. D.h. der minimale Wert ist 8. Man beachte, dass es sich hierbei um eine redundante Information handelt, da im Feld Len von IP ebenfalls die Länge des Datenbereichs angegeben ist.

Checksum Prüfsumme. Der Algorithmus funktioniert wie derjenige zur Berechnung der Prüfsumme in IP, jedoch wird als Grundlage der Header des UDP Datagrams, das Datenfeld des UDP Datagrams und ein sogenannter Pseudoheader herangezogen, der sich folgendermaßen zusammensetzt: IP Quelladresse (32 Bits), IP Zieladresse (32 Bits), der Wert 0 (8 Bits) das Feld Protocol vom IP Header (8 Bits) und das Feld Len vom UDP Header (16 Bits). Der Pseudoheader wird nicht übertragen: er wird nur zur Berechnung der Prüfsumme herangezogen!
Man sieht, dass im Gegensatz zu IP auch das Datenfeld bei der Berechnung der Prüfsumme herangezogen wird.

Data Datenfeld

Dass UDP die Prüfsumme ebenfalls über das Datenfeld berechnet ist einerseits ein Vorteil, aber auch andererseits unter Umständen ein Nachteil, wenn der Overhead durch die Berechnung (z.B. bei Datenraten größer als 1 GBit/s) zu groß ausfällt.

9.3. TCP

TCP (engl. transmission control protocol) ist das komplexere Transportprotokoll, das TCP/IP zur Verfügung stellt. Es wurde 1981 im RFC 793 publiziert und seit dem nicht verändert. Erweiterungen sind in zusätzlichen RFCs spezifiziert.

Das Entwurfsziel war ein verbindungsorientiertes, zuverlässiges Ende-zu-Ende Transportprotokoll zur Verfügung zu stellen, das einfach ist und eine schnelle Kommunikation erlaubt. Es ist charakterisiert durch folgende Eigenschaften:

- Es ist zuverlässig: verloren gegangene oder falsche Daten werden wieder neu übermittelt, doppelt ankommende Daten werden verworfen und es wird sichergestellt, dass die Daten in der Reihenfolge des Absendens dem Empfängerprozess weitergereicht werden.
- Verbindungen werden unterstützt und verwaltet: Verbindungsauf- und Abbau.

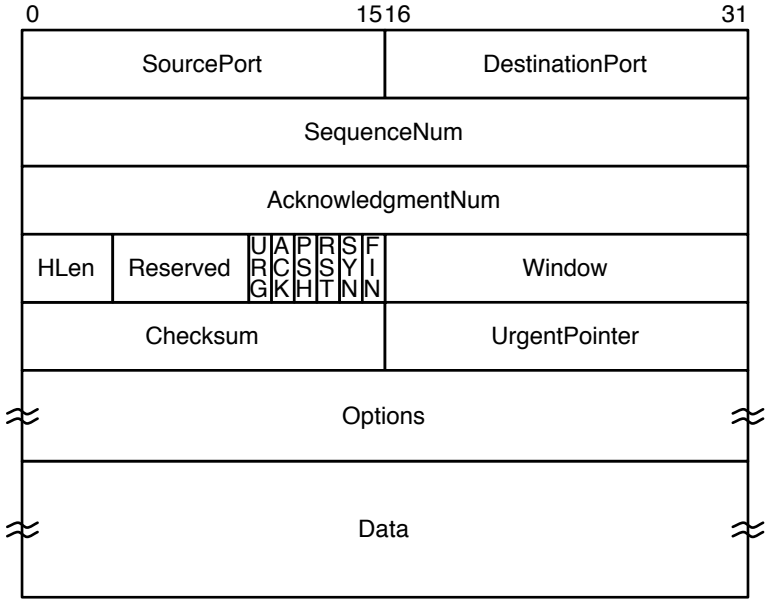


Abbildung 9.2.: Aufbau eine TCP Segmentes

- Die Datenübertragung ist vollduplex: es werden gleichzeitig zwei Datenströme (einer in jeder Richtung) unterstützt.
- Zur Datenübertragung wird eine Abstraktion über Streams zur Verfügung gestellt. D.h. die Anwendungsschicht kann einen Byte-Strom zu TCP senden bzw. einen Byte-Strom von TCP lesen. Dazu stellt TCP eine Art von Segmentierung zur Verfügung (siehe Abschnitt 9.3.1 und 9.3.3 auf Seite 111).
- Es gibt einen Mechanismus zur Flusskontrolle. Flusskontrolle hindert den Sender daran, die Kapazität des Empfängers zu überschreiten. Es handelt sich also um einen Mechanismus auf der Ende-zu-Ende Ebene und betrifft die beiden kommunizierenden Hosts.
- Es gibt eine Überlastkontrolle. Überlastkontrolle bietet einen Schutz vor Überlastung des Netzes. Es handelt sich um einen Mechanismus der einerseits den Sender und andererseits das Netz betrifft.
- Es erfolgt eine einfache Überprüfung auf Korrektheit der übertragenen Daten.
- Es ist als robustes Protokoll entworfen und folgt dem Grundsatz: konservativ in dem was gesendet wird und tolerant bzgl. der empfangenen Segmente.

9.3.1. Aufbau eines Segmentes

Bei TCP werden die einzelnen Pakete als Segmente genannt. Der Aufbau eines TCP Segmentes ist in Abbildung 9.2 zu sehen.

SourcePort 0–65535

DestinationPort 0–65535

SequenceNum Die Sequenznummer wird für den Window Sliding Mechanismus verwendet. Da TCP ein byte-orientiertes Protokoll ist, hat jedes Byte eine Sequenznummer. Dieses Feld enthält die Sequenznummer des ersten Byte im Datenfeld.

AcknowledgmentNum Wird dem Sender ein Segment gesendet, das das ACK Bit gesetzt hat, dann enthält dieses Feld die Bestätigungsnummer. Sie gibt die Nummer der nächsten erwarteten Sequenznummer an. Alle Bytes mit einer kleineren Bestätigungsnummer werden hiermit bestätigt.

HLen 4 Bit: Länge des Headers in 32 Bitworten bis zum Beginn des Datenfeldes.

Reserved 6 Bit langes Bitfeld, das lauter 0en enthalten muss.

URG Das Bit URG (urgent) zeigt der empfangenen Anwendung an, dass Daten priorisiert behandelt werden sollen. Die dringenden Daten befinden sich am Anfang des Datenbereiches und reichen bis zu der Stelle, die durch das Feld UrgentPtr angezeigt werden, das auf das letzte Oktett der dringlichen Daten zeigt. URG wird selten verwendet, z.B. bei telnet oder rlogin. Diese Kennzeichnung ist deshalb notwendig, da TCP den Inhalt der Daten nicht kennt; für TCP handelt es sich um einen Strom von Daten, die auf Grund der Stream-Eigenschaft mittels des FIFO Prinzips übertragen werden.
Überträgt man z.B. eine große Datei und will diesen Transfer der Daten abbrechen, dann ist es nicht sinnvoll ein Abbruch-Kommando am Ende des Datei hinten anzufügen. Es muss eine Möglichkeit geben das Abbruch-Kommando vorzureihen. TCP stellt eine Möglichkeit zur Verfügung solche dringenden Daten zu versenden. Es werden diese Daten am Anfang des Datenbereiches eines Segmentes platziert, das URG Flag gesetzt und der UrgentPtr richtig gesetzt. Beim Empfänger werden diese Daten vorgereiht und dem Prozess vorrangig und als „urgent" markiert ausgeliefert. Die restlichen Daten des Datenbereiches werden normal im Datenstream ausgeliefert.

ACK Mittels ACK (acknowledgement) bestätigt TCP den Erhalt von Daten (siehe Acknowledgment Number).

PSH Ist PSH (push) gesetzt, dann soll der TCP/IP Stack dieses Segment ohne weiteres Zwischenspeichern an die Anwendung weiterreichen. Dafür bietet TCP eine push Funktion an, die von der Anwendungsschicht aufgerufen werden kann. D.h. die normale Pufferung der Bytes bis sich genügend Bytes angesammelt haben, wird ausgeschalten, das PSH Flag gesetzt und alle Daten werden sofort abgesendet. Auf der Empfängerseite wird erkannt, dass das PSH Flag gesetzt ist und die Daten werden sofort an die Anwendung weitergereicht. D.h. auch hier wird nicht gewartet bis „genügend" Daten eingetroffen sind.

RST Dieses Bit RST (reset) wird verwendet, um eine sofortige Auflösung der TCP Verbindung anzuzeigen und sollte nicht der normale Vorgang zum Beenden einer Verbindung sein. Es wird z.B. verwendet, wenn der Empfänger ein unerwartetes Segment erhalten hat, wodurch ein Fehlerzustand eingetreten ist und dadurch die Verbindung abgebrochen werden muss.

SYN Beim Verbindungsaufbau wird SYN (synchronize) verwendet (siehe 9.3.2 auf Seite 109).

FIN Beim Verbindungsabbau wird FIN (finish) verwendet (siehe 9.3.2 auf der nächsten Seite).

Window Dieses Feld gibt die Fenstergröße an. Gemeinsam mit SequenceNum, AcknowledgeNum wird der Sliding Window Mechanismus in TCP realisiert (siehe Abschnitt 9.3.3 auf Seite 112).

Checksum Prüfsumme. Der Algorithmus funktioniert wie derjenige zur Berechnung der Prüfsumme in IP, jedoch wird als Grundlage der Header des TCP Segmentes, das Datenfeld des TCP Segmentes und ein sogenannter Pseudoheader herangezogen, der sich folgendermaßen zusammensetzt: IP Quelladresse (32 Bits), IP Zieladresse (32 Bits), der leere Checksum-Wert 0 (8 Bits) das Feld Protocol vom IP Header (8 Bits) und das Feld Len vom TCP Header (16 Bits). Der Pseudoheader wird nicht übertragen: er wird nur zur Berechnung der Prüfsumme herangezogen!
Man sieht wiederum, dass im Gegensatz zu IP auch das Datenfeld bei der Berechnung der Prüfsumme herangezogen wird.

UrgentPtr Bei dem Feld UrgentPtr handelt es sich um einen Zeiger auf das letzte Oktett der dringlichen Daten und wird nur interpretiert, wenn das URG Flag gesetzt ist.

Options Das Feld Options kann mehrere Optionen beinhalten. Jede Option besteht aus einem Feld „Art der Option" (option-kind) und eventuell einem Feld „Länge der Option" (option-length) gefolgt von einem Feld „Daten der Option" (option-data). Einige Optionen haben keine Daten und damit auch kein Längenfeld. In RFC 793 sind nur 3 Optionen definiert:

- „End of Option List" (option-kind=0) kennzeichnet das Ende der Optionen und wird nur verwendet, wenn das Ende der Optionen nicht mit dem Ende des TCP Headers übereinstimmt.
- „No-Operation" (option-kind=1) kann verwendet werden, um Optionen von einander zu trennen und wird eingesetzt, wenn der Sender Optionen an Wortgrenzen beginnen lassen will.
- „Maximum Segment Size (MSS)" (option-kind=2, option-length=4, option-data enthält MSS). Bzgl. MSS siehe Abschnitt 9.3.3 auf Seite 111. Diese Option kann nur in einem Segment vorhanden sein, das das SYN Bit gesetzt hat.

Nach dem Ende der Optionen muss der verbleibende Platz zur nächsten 32 Bit Grenze mit Nullen aufgefüllt (engl. padding) werden. Weitere Optionen wurden in getrennten RFCs definiert.

Data Datenfeld. Betrachtet man die maximale MTU bei Ethernet-Netzen sieht man folgendes: der minimale TCP Header ist 20 Bytes und damit entsteht gemeinsam mit dem IP Header eine minimale Headerlänge von 40 Bytes. Da die maximale Größe des Datenbereiches eines Ethernet-Frames bei 1500 Bytes liegt, kommt man auf eine max. Datengröße bei TCP auf 1460 Bytes.

9.3.2. Verbindungsauf- und Abbau

Verbindungsaufbau

Der Verbindungsaufbau wird in TCP mittels eines 3-Wege-Handshake (engl. three-way handshake) vorgenommen und wird durch die Synchronisierung zweier Sequenznummern (eine am Server, eine am Client) erreicht:

1. Der Client sendet ein Segment zum Server mit:
   SYN = 1; SequenceNum = x
   Das bedeutet, dass der Client eine Verbindung aufbauen will und mit dem Server bzgl. seiner SequenceNum synchronisieren will. Der Client hat als anfänglichen Wert dafür einen beliebigen Wert x gewählt. Dieser Vorgang wird als aktives Öffnen bezeichnet (d.h. der Host beginnt den Verbindungsaufbau).
2. Der Server antwortet mit einem Seqment an den Client:
   SYN = 1; ACK = 1; SequenceNum = y; AcknowledgeNum = $x + 1$
   Damit wird angezeigt, dass der Server die Verbindung prinzipiell akzeptiert, mit dem Client seine SequenceNum y synchronisieren will und gleichzeitig die SequenceNum x des Clients bestätigt indem er als AcknowledgeNum $x + 1$ zurücksendet. Das bedeutet, dass der Server als nächste SequenceNum vom Client $x + 1$ erwartet.
   Damit der Server auf diese Weise antwortet, muss er zuerst ein sogenanntes passives Öffnen (d.h. der Server wartet auf einen Verbindungsaufbau) durchgeführt haben.
3. Als Abschluss schickt der Client an den Server ein Segment mit:
   ACK = 1, AcknowledgeNum = $y + 1$
   Hiermit wird der Verbindungsaufbau abgeschlossen, da der Client die Sequenznummer des Servers akzeptiert und damit mitteilt, dass er vom Server beim nächsten Seqment die Sequenznummer $y + 1$ erwartet.

Mittels einer um eins erhöhten AcknowledgeNum wird eigentlich angezeigt, dass alle vorangegangenen Sequenznummern bestätigt werden. Beim Senden des ersten Datensegmentes hat demzufolge das erste Datenbyte die Sequenznummer $x + 1$.

Nicht ersichtlich ist, dass entsprechende Timer gesetzt werden, die bei Ablauf vor der Bestätigung bewirken, dass das jeweilige Segment neu gesendet wird.

Zustandsübergangsdiagramm

In Abbildung 9.3 auf der nächsten Seite finden sich alle Zustände, die beim Aufbau (alles oberhalb des Zustandes ESTABLISHED) und beim Abbau einer Verbindung (alles unterhalb des Zustandes ESTABLISHED) auftreten. Der Vorgang der eigentlichen Datenübertragung ist im Zustand ESTABLISHED verborgen.

Das Diagramm verwendet die übliche Notation für Zustandsübergangsdiagramme, wobei alle in Großbuchstaben verfassten Bezeichner entweder empfangene Segmente (vor dem /) bzw. gesendete Segmente (nach dem /) angeben. Die in Kleinbuchstaben angegebenen Bezeichner „close", „passive open" und „active open" geben Funktionen an, die der Anwendungsprozess aufruft und der Bezeichner „timeout" kennzeichnet einen Zeitablauf.

Drei Punkte gibt es zu beachten:

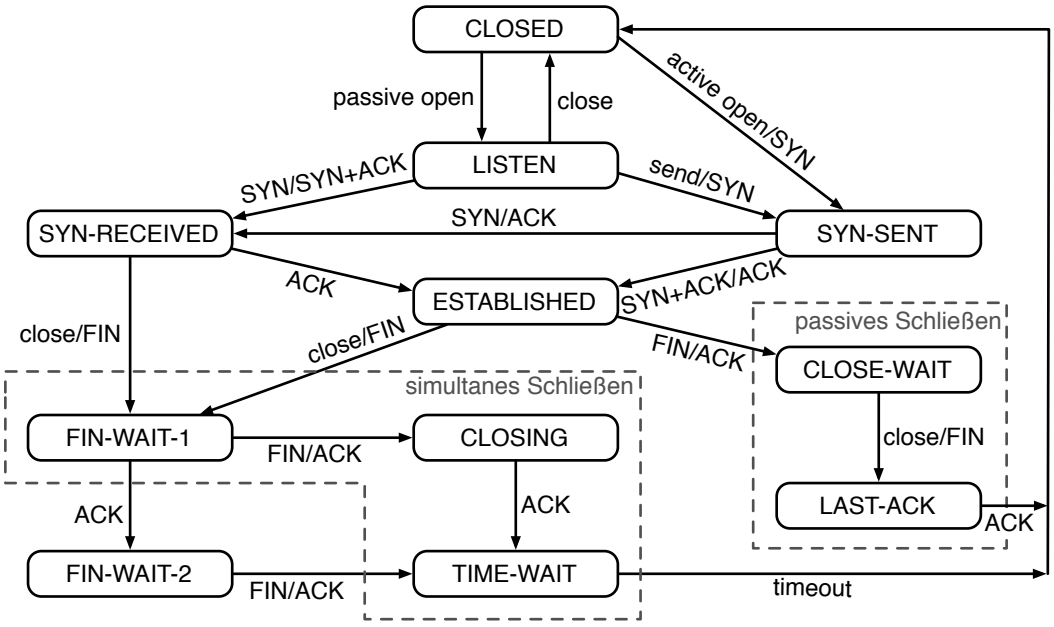


Abbildung 9.3.: Zustandsübergangsdiagramm

1. Auch wenn das ACK vom Client an den Server verloren geht, funktioniert die Verbindung noch korrekt. Das liegt daran, dass der Client sich schon im Zustand ESTABLISHED befindet und der Server zwar noch im Zustand SYN_RECEIVED ist, aber beim nächsten empfangenen Segment in den Zustand ESTABLISHED wechselt. Dieser Wechsel in ESTABLISHED kann deshalb durchgeführt werden, da das Flag ACK in jedem Datensegment gesetzt ist und das Feld Acknowledgment den korrekten Wert enthält. Das ist wichtiger Punkt in TCP: Jedes Segment meldet, welche Sequenznummer als nächstes erwartet wird, auch wenn sich dabei die gleiche Sequenznummer wiederholt, die in einem der vorherigen Segmente enthalten war.
2. Der Übergang von LISTEN nach SYN_SENT ist im TCP Standard vorgesehen, aber wird in der Regel von keiner TCP Implementierung dem Anwenerprozess als Funktion zur Verfügung gestellt: In den Zustand LISTEN kommt man vom Zustand CLOSED mittels passiven Öffnen, das durch einen Funktionsaufruf am Server erfolgt. Normalerweise kann danach nicht direkt gesendet werden.
3. Viele Zustandsübergänge sind nicht eingezeichnet. Kommt innerhalb eines gesetzten Timeouts kein Acknowledgment an, dann wird das Segment neu gesendet. Diese Neuübertragungen sind nicht eingezeichnet. Nach mehreren Versuchen gibt TCP auf und wechselt in den Zustand CLOSED.

Verbindungsabbau

Beim Abbau einer Verbindung ist es wichtig, dass die Anwendungsprozesse auf beiden Seiten die Verbindung schließen. Schließt nur eine Seite die Verbindung bedeutet das, dass diese Seite keine Daten mehr senden darf, aber sehr wohl noch bereit für den Empfang von Segmenten sein muss. Aus diesem Grund ist das Zustandsübergangsdiagramm für den Abbau einer Verbindung komplizierter, weil die Möglichkeit beachtet werden muss, dass beide

Seiten die close Funktion aufrufen. Ferner besteht die Möglichkeit, dass zuerst die eine Seite und später die andere Seite close aufruft. Daraus resultieren drei Möglichkeiten:

- Ein Anwenderprozess initiert den Verbindungsabbau mittels „close".
- Der entfernte Kommunikationspartner initiert den Verbindungsabbau mittels Senden eines „FIN" (passives Schließen).
- Beide Anwenderprozesse wollen gleichzeitig die Verbindung mittels „close" beenden (simultanes Schließen).

Da beide Partner die Verbindung abbauen müssen, kommt es zum Austausch von insgesamt 4 Segmenten. Allerdings kann das ACK Segment und das FIN Seqment, die der Host sendet, der passives Schließen durchführt in ein Segment zusammengefasst werden. Damit werden – analog zum Verbindungsaufbau – ebenfalls nur 3 Segmente versendet.

Das Timeout vom Zustand TIME-WAIT ist notwendig, dass sicher ist, dass das ACK angekommen ist bzw. dass es zu keiner Überschneidung mit einer folgenden Verbindung kommt.

9.3.3. Datenübertragung

Segmentierung

Aus Sicht der Anwendung stellt TCP als Schnittstelle zwei Byte-Streams zur Verfügung. Ein Byte-Stream wird zum Lesen und einer zum Schreiben verwendet. Das heißt die Anwendung kann einzelne Bytes mittels dieser Streams schreiben bzw. lesen.

Andererseits werden in der Internet-Schicht IP Pakete übertragen und es wäre sehr ineffizient jedes Byte in einem eigenen IP Paket zu versenden. Das Verhältnis von Header Daten zu Nutzdaten wäre sehr ungünstig. Aus diesem Grund speichert TCP die Bytes in einem Puffer zwischen und erzeugt daraus eine Folge von Segmenten. Diese Segmente werden mittels IP Paketen übertragen und auf der Empfängerseite wieder in TCP Segmente zurückgewandelt. Auf der Empfangsseite werden diese Segmente wieder in einem Puffer gespeichert und der Anwendung als Byte-Stream zur Verfügung gestellt.

Daraus folgt, dass große Segmente im Grunde die effizientere Wahl darstellen. Es gibt allerdings zwei Gründe, die gegen eine groß gewählte Segmentgröße sprechen: Einerseits kommt es dann zu langen Wartezeiten, bis Seqmente versendet werden und andererseits kommt es wahrscheinlich zur Fragmentierung, da die Segmente bei dem Transport über Netztechnologien, die eine MTU aufweisen, die kleiner ist als das in ein IP Datagram verpackte Segment.

Das Versenden eines Segmentes hängt von drei Kriterien ab:

- Die maximale Segmentgröße (engl. maximum segment size, MSS) ist erreicht. MSS definiert die Anzahl der Bytes, die als Nutzdaten in einem TCP Segment gesendet werden können. Da die Header von IP und TCP zusammen mind. 40 Bytes groß sind, muss die MSS um mind. 40 Bytes kleiner sein als die MTU. Im Falle von Ethernet beträgt die MSS also 1460 Bytes. Wird z.B. allerdings PPP über Ethernet verwendet, vermindert sich um weitere 8 Byte auf 1452 Bytes.
- Die Anwendung wünscht einen sofortigen Versand (push).

- Der Timer, der beim Einspeisen eines Bytes in den Sendepuffer gestartet wird läuft ab. D.h. es wurde eine gewissse Zeit von der Anwendung keine neuen Daten in den Stream geschrieben.

Sliding-Window

Das Prinzip Sliding-Window Algorithmus wurde schon detailiert im Abschnitt 3.4.3 auf Seite 45 beschrieben. Der in TCP implementierte Algorithmus unterscheidet sich in folgenden Punkten von dem schon beschriebenen Algorithmus:

1. Die Fenstergröße in TCP hat keine feste Größe, sondern wird dem Sender mittels des „Window" Feldes im TCP Header bekanntgegeben. Das bedeutet, dass der Empfänger das „Window" Feld im TCP Header auf Grundlage der freien Puffergröße auswählt (siehe auch nächsten Punkt).
2. In der TCP Variante wird auch noch der Sendepuffer und der Empfangspuffer beachtet (siehe 9.3.3 auf der vorherigen Seite).

Mit jeder Bestätigung eines Segmentes sendet der Empfänger auch die neue Windowgröße mit. Kann der Empfänger keine weiteren Daten annehmen, wird in der Bestätigungsantwort der Wert 0 für „Window" gesetzt. Damit weiß der Sender, dass der Empfänger derzeit keine weiteren Daten annehmen kann. Damit stellt sich allerdings die Frage, wie der Sender weiß wann er wieder mit dem Senden fortfahren kann. Das Problem wird folgendermaßen gelöst: Der Sender sendet in regelmäßigen Abständen Segmente mit der Datenlänge 1 mit dem Wissen, dass diese Daten wahrscheinlich verworfen werden. Kann der Empfänger wieder Daten annehmen, dann wird er dieses 1 Byte lange Datum annehmen und mit einer Bestätigungsnachricht mit neu gesetzter „Window" Größe reagieren.

In diesem Zusammenhang ist es wichtig darauf hinzuweisen, dass ein ACK nicht bedeutet, dass der Anwenderprozess die Daten entgegengenommen hat!

Flusskontrolle

Wie schon beschrieben ist es die Aufgabe der Flusskontrolle den Empfänger vor Überlauf zu schützen. Kommen mehr Segmente beim Empfänger an als dieser in seinem Puffer zwischenspeichern kann, dann werden diese Segmente vom Empfänger verworfen. In TCP wird die Flusskontrolle mittels des Sliding-Window Algorithmus (siehe Abschnitt 9.3.3) realisiert.

Das bedeutet, dass TCP ein Verfahren mit positiven Quittungen implementiert, die Summenquittungen darstellen.

Überlastkontrolle

Unter Überlastkontrolle versteht man die Mechanismen zur Vermeidung von Überlastung des Netzes (engl. congestion avoidance). Im Gegensatz zur Flusskontrolle, dessen Aufgabe es ist, den Empfänger vor Überlauf zu schützen, werden mittels Mechanismen der Überlastkontrolle die Router am Weg zwischen Sender und Empfänger geschützt.

Eine Überlastsituation kann im Netz leicht entstehen, obwohl u.U. keine Überlast in einem Empfänger auftritt. Senden viele Sender über einen Router, dann kann dieser leicht überlastet werden. Diese Situation verschlimmert sich noch weiter dadurch, dass in TCP verlorengegangene Segmente eine erhöhte Netzaktivität hervorrufen, da es zu sinnlosen Wiederholungen kommen kann, wenn die Bestätigungen den Sender erst nach Ablauf des Timeouts erreichen.

Da die Überlastkontrolle in TCP jedoch beim Sender angesiedelt ist, benötigt dieser Informationen über die Netzauslastung, um seine Sendegeschwindigkeit an die aktuelle Situation anpassen zu können. Das ist jedoch das Schwierige an der Überlastkontrolle, da der Sender diese Information nicht direkt vom Netz erfragen kann, sondern diese indirekt ermitteln muss.

Folgende grundlegende Mechanismen zur Überlastkontrolle sind direkt in TCP vorhanden, wobei es auch etliche fortgeschrittene Verfahren gibt:

Nagle-Algorithmus

Betrachten wir nochmals den Aspekt wann ein Segment zu senden ist, diesmal jedoch unter dem Gesichtspunkt der Überlastkontrolle: Gehen wir davon aus, dass der Sender MSS Daten-Bytes senden will und das Fenster um zumindest diese Anzahl an Bytes offen ist. In diesem Fall wird der Sender senden, wenn einer der Fälle eintritt, die im Abschnitt 9.3.3 auf Seite 111 beschrieben sind. Was passiert jedoch, wenn das Fenster nur zur Hälfte offen ist? Soll kein Segment gesendet werden oder soll ein Segment mit der Hälfte der Bytes gesendet werden?

Der Standard definiert diesbezüglich kein Vorgehen. In den ersten TCP Implementierungen wurde die Entscheidung getroffen, die Hälfte der Datenbytes zu senden. Es stellte sich jedoch heraus, dass genau dieses Vorgehen zu einer Überlastung des Netzes führen kann. Es trat das sogenannte „Silly-Window-Syndrom" auf: Gehen wir davon aus, dass der Puffer des Empfängers voll ist und der Empfänger deshalb keine weiteren Daten annehmen kann. Gehen wir weiters davon aus, dass die Anwendung am Empfänger ein einziges Byte aus dem Puffer ausliest. D.h. es kann eine Bestätigung zum Sender gesendet werden, die eine Fenstergröße von 1 enthält. Worauf der Sender wieder ein Segment mit der Datenlänge 1 senden kann. Wenn der Empfänger wieder genau ein Byte ausliest, dann wird wieder eine Bestätigung mit der Fenstergröße 1 zurückgesendet. D.h. das Netz wird mit vielen kleinen Segmenten überschwemmt, wodurch eine potentielle Überlastsituation entsteht.

Das Silly-Window-Syndrom stellt nur dann ein Problem dar, wenn entweder der Sender ein kleines Segment überträgt oder der Empfänger das Fenster nur ein wenig öffnet.

Wie kann man mit dieser Situation umgehen? Man kann einfach eine gewisse Zeit warten bis verfügbare Daten gesendet werden. Die Frage ist wie lange eine vernünftige Zeitdauer bemessen ist. Wird zu lange gewartet, dann werden interaktive Anwendungen wie telnet oder ssh beeinträchtigt. Wird hingegen zu kurz gewartet, dann entsteht wieder das Silly-Window-Syndrom.

TCP implementiert den sogenannten Nagle-Algorithmus, der nach John Nagle benannt ist. Die zugrundeliegende Idee ist, dass der Sender irgendwann ein ACK empfangen wird, solgange TCP Daten unterwegs sind. Dieses ACK kann wie ein Timerablauf behandelt werden, wobei dadurch die Übertragung von mehr Daten angestoßen wird.

Wenn die Anwendung sendebereite Daten produziert, dann:

1. Wenn sowohl die verfügbaren Daten als auch das Fenster $\geq$ MSS, dann sende Daten.
2. Anderenfalls: Wenn unbestätigte Daten anstehen,
   a) dann puffere die neuen Daten bis ein ACK kommt.
   b) Anderenfalls: Sende alle neuen Daten jetzt.

Um interaktive Anwendungen vom Nagle-Algorithmus nicht zu beeinträchtigen, besteht die Möglichkeit diesen auszuschalten. Dies geschieht über das Socket-API mittels der Option TCP_NODELAY.

Slow-Start

Am Anfang einer Übertragung ist noch keine Information über die Auslastung eines Netzes bekannt. Die Idee ist, die Senderate langsam zu steigern bis entweder der Fall eines Segmentverlustes oder ein ACK-Timeout eintritt.

Fast-Retransmit

Die Idee von Fast-Retransmit ist, nach einem Paketverlust schneller auf eine Stausituation zu reagieren. Dazu informiert der Empfänger den Sender, wenn Pakete außer der Reihe ankommen und somit ein Paketverlust vorliegt. Das wird dadurch erreicht, dass der Empfänger das zuletzt korrekt empfangene Paket für jedes außer der Reihe empfangene Paket jedes Mal neu bestätigt. Diese mehrfach versendeten Bestätigungen werden „Dup-Acks" genannt. Hat der Sender drei solcher Dup-Acks erhalten schließt er daraus, dass ein Paket verloren gegangen ist und sendet es auch vor Ablauf des Timeout wieder an den Empfänger. Außerdem schließt der Sender, dass Folgepakete sehr wohl angekommen sind und erhöht das Sendefenster um die Anzahl der empfangenen Dup-Acks.

10. Protokolle der Anwendungsschicht

Hier werden die wichtigsten Protokolle besprochen, die der Anwendungsschicht zuzurechnen sind und trotzdem für den Betrieb eines TCP/IP Netzes notwendig sind.

10.1. DHCP

siehe Foliensatz!

10.2. DNS

siehe Foliensatz!

10.3. Routingprotokolle

In Abhängigkeit, ob ein Router ein Teil eines autonomen Systems (ein oder mehrere Netze, die von einer Organisation verwaltet werden) ist oder an der Grenze eines autonomen Systems eingesetzt wird, verwendet man verschiedene Arten von Routingprotokollen:

Interior Gateway Protocol (IGP) Es handelt sich um einen Typ von Routing-Protokollen, die ausschließlich innerhalb eine autonomen Systems zum Einsatz kommt. Beispiele sind: RIP und OSPF. RIP (routing information protocol) ist das am häufigsten eingesetzte Protokoll und verwendet den Distanzvektor-Algorithmus (siehe Abschnitt 4.2.4 auf Seite 64), während OSPF (open shortest path first) den Link-State-Algorithmus (siehe Abschnitt 4.2.4 auf Seite 67) verwendet.

RIP wird nur in kleineren Organisationen eingesetzt. Dies liegt einerseits an den schon im Abschnitt 4.2.4 erläuterten Nachteilen des eingesetzten Verfahrens, aber auch daran, dass z.B. immer nur die Hops als Gewichte herangezogen werden. Damit würde eine ISDN Verbindung, die über einen Hop geht einer hoch performanten Standleitung mit 2 Hops der Vorzug gegeben werden. Außerdem kann die Anzahl der Hops 15 ja nicht überschreiten.

OSPF führt eine Wertigkeit der Links ein und kann damit einer Standleitung den Vorzug gegenüber einer ISDN Leitung geben. Allgemein findet es eine optimale Route auf Grund mehrerer Kriterien und unterteilt außerdem noch in sogenannte Areas, um effiziente eine Router-Tabellen-Übertragung zu erreichen. OSPF ist ein bedeutend komplexeres Protokoll.

Exterior Gateway Protocol (EGP) Diese werden zwischen Routern autonomer Systeme eingesetzt. Beispiele sind: BGP, EGP (nicht mehr in Verwendung).

11. NAT

NAT (Network Address Translation) ist in Rechnernetzen der Sammelbegriff für Verfahren, um automatisiert und transparent Adressen in IP Paketen durch andere Adressen zu ersetzen. Diese Adressumsetzung wird in der Regel zwischen einem öffentlichen und einem internen Netz durchgeführt.

Der Haupteinsatz von NAT besteht darin Netze mit privaten IP Adressen an das öffentliche Internet anzubinden und damit auch öffentliche IP Adressen zu sparen. In gewisser Weise kann es auch als eine Sicherheitsmaßnahme eingesetzt werden, auch wenn dies nicht die primäre treibende Absicht ist.

Das Prinzip ist, dass am Router, der das interne vom öffentlichen Netz trennt eine Umsetzungstabelle enthalten ist, die eine Abbildung der privaten IP Adressen auf die öffentlichen IP Adressen enthält. Unter Umständen enthält diese Umsetzungstabelle weitere Informationen, um die Abbildung eindeutig zu machen.

Um komplexere Protokolle (wie ftp oder VoIP) über einen NAT-Router betreiben zu können, muss der NAT-Router allerdings als ein Gateway fungieren.

NAT hat trotz des komplexeren Systemaufbaus sicher dazu beigetragen, dass die Anzahl der freien IPv4 Adressen viel langsamer abgenommen hat.

Es gibt zwei grundlegende Formen von NAT, die in den beiden folgenden Abschnitten kurz besprochen werden.

11.1. Source-NAT

Beim Source-NAT (SNAT) wird bei einer Anfrage eines Clients im lokalen Netz an einen Server im Internet die private Client-Adresse auf eine verfügbare externe Adresse geändert. Es gibt prinzipiell drei Möglichkeiten:

Statisches SNAT Hierbei wird jeder privaten Adresse genau eine externe Adresse zugeordnet. D.h. es gibt eine 1:1 Abbildung von lokalen zu extern verfügbaren Adressen. Dies kann notwendig sein, wenn der interne Rechner als Server im Internet verfügbar sein soll oder wenn man die Sicherheit erhöhen will (Struktur des internen Netzes wird nicht preisgegeben).

Dynamisches SNAT Sind mehr lokale Rechner vorhanden als externe Adressen zur Verfügung stehen, muss die Zuordnung von lokaler zu externer Adresse dynamisch vorgenommen werden. Die Zuordnungstabelle im NAT-Router kann noch mit zusätzlichen Informationen wie z.B. Portangaben versehen werden, sodass auch gleichzeitig mehr interne Rechner mit Servern im Internet kommunizieren können als externe Adressen zur Verfügung stehen.

Masquerading Es handelt sich um eine Sonderform bei der alle internen Adressen auf *eine* externe Adresse abgebildet werden.

11.2. Destination-NAT

Das Destination-NAT (DNAT) ist im Grunde die Umkehrung des SNAT: Es wird die Ziel-Adresse abgebildet. Ein externer Client sendet eine Anfrage an eine extern verfügbare Adresse unseres Netzes (z.B. an die Adresse des NAT-Routers). Der NAT-Router bildet die Zieladresse z.B. auf die interne Adresse des DMZ Servers ab und leitet daher diese Anfrage an den DMZ Server weiter. Hier gibt es prinzipiell zwei Möglichkeiten:

- Der NAT-Router leitet alle Anfragen bzgl. einer extern verfügbaren Adresse an den DMZ Server weiter.
- Der NAT-Router leitet Anfragen bzgl. einer extern verfügbaren Adresse an die externe Adresse weiter, wenn zusätzliche Kriterien zutreffen. Das wichtigste Kriterium ist die Portangabe. D.h. nur wenn die Portangabe z.B. 80 ist wird an den internen DMZ Server weitergeleitet.

12. IPv6

Die wesentlichen funktionellen Unterschiede zu IPv4 sind:

- Adressraum: Einer der Beweggründe IPv6 zu entwickeln, war die geringe Anzahl an freien Adressbereichen. IPv6 hat 128 Bit lange Adressen. D.h. es gibt ca. $3.4 \cdot 10^{38}$ mögliche Adressen. Die Notation wird nicht mehr byteweise in Dezimalschreibweise durch Punkte getrennt, sondern wortweise (16 Bit) in Hexadezimalschreibweise durch Doppelpunkte getrennt angegeben. Z.B.: 2002:89d0:e02e::89d0:e02e.
- Durch die hohe Anzahl an verschiedenen Adressen besteht die Möglichkeit diese geographisch-hierarchisch zu vergeben. Dadurch können die Anzahl der Routereinträge weiter gesendet werden.
- Es wurde eine lokale Adressvergabe eingeführt, sodass Adressen automatisch vergeben werden. Dazu kann sich ein Host eine IPv6 Adresse direkt aus einer MAC Adresse geben und mit angrenzenden Hosts und Routern diesbezüglich abstimmen.
- Es gibt im IP Header keine Checksumme mehr. D.h. alle Überprüfungen müssen durch höhere Protokolle erledigt werden. Das liegt daran, dass viele Router die Checksum überhaupt nicht überprüft haben, sondern einfach um eins erhöht (wg. dem Header TTL) haben.

  Der Basisheader wurde generell kürzer gestaltet. Er enthält nur mehr 7 anstatt 13 Felder. Es kann allerdings auch mehrere Header geben. Diese Änderungen ermöglichen Routern Paket schneller zu verarbeiten.
- Sicherheitsfunktionen wurden direkt eingebaut und basieren auf IPSec. IPSec kann zwar auch mit IPv4 verwendet werden, ist aber in IPv6 direkt integriert.
- Die Fragmentierung der Datenpakete in Routern wurde entfernt. Müsste ein Paket fragmentiert werden, dann wird eine Fehlermeldung an den Sender zurückgeschickt, der daraufhin die MSS anpassen muss und erneut sendet.
- QoS wurde ebenfalls integriert. Ziel ist die verbesserte Übertragung von Audio und Videodaten sowie die Übertragung von Daten in Echtzeit.
- Weiters gibt es zusätzlich zu Broadcast und Multicast auch Anycast-Adressen.

Teil IV.

Dienste und Anwendungen

Der Teil beschreibt wichtige Dienste und Anwendungen für das Internet.

13. Entfernter Zugriff

13.1. FTP

Datenzugriff mittels ftp (RFC 959) ist ein relativ altes und noch immer sehr weit verbreitetes Protokoll.

Anders als http benützt ftp *mehr* als eine Verbindung zwischen Client und Server: Zunächst baut der Client eine Verbindung zum Port 21 (Control Port) des Servers auf. Diese Verbindung dient zur Authentifizierung und Befehlsübertragung. Zur eigentlichen Datenübertragung wird bei Bedarf eine eigene Verbindung aufgebaut.

Dazu gibt es *2 Betriebsarten*:

Active Mode Der Server baut von seinem Port 20 (Data Port) eine Verbindung zu einem vom Client gewählten Port (über 1023) auf.

Passive Mode Der Client baut eine Verbindung zu einem vom Server gewählten Port auf (meistens beide größer 1023).

Dieses Protokoll wird als relativ unsicher angesehen, da sowohl die Passwortübertragung als auch die Datenübertragung unverschlüsselt stattfindet. Technisch gesehen ist der Befehlskanal wie beim telnet Protokoll aufgebaut. Aus diesem Grund sollte man Daten nur mit scp (secure copy protocol) oder der Weiterentwicklung sftp (siehe 13.3.1 auf der nächsten Seite) arbeiten.

13.2. Netzwerkdateisysteme

Als Alternative zu einer Übertragung mit ftp, scp oder sftp kann für den Datenzugriff ein Netzwerkdateisystem wie NFS (Network File System) oder CIFS (Common Internet File System) verwendet werden.

NFS

NFS wurde von SUN entwickelt und stellt ein Netzwerkdateisystem für Unix dar. NFS hat heutzutage keine dominante Bedeutung mehr. Viele wichtige Punkte wie die Authentifizierung der Benutzer (und nicht nur der Rechner wie in NFSv3) oder der Wegfall der Unix-Lastigkeit wurden erst in NFSv4 eingeführt. Diese Version hat sich jedoch nicht durchsetzen können.

CIFS

CIFS wurde 1996 von Microsoft bei der IETF eingereicht und stellt eine Erweiterung von SMB (Server Message Blocks) dar. SMB wurde von IBM mit dem Ziel entwickelt für DOS ein Netzwerkdateisystem zur Verfügung zu haben. Microsoft hat SMB weiterentwickelt und in ihre Produktlinie integriert. CIFS bietet entfernten Zugriff auf Dateien, Drucker oder andere Geräten als auch die Möglichkeit zur IPC.

13.3. Entfernte Ausführung

13.3.1. SSH

siehe Foliensatz!

13.3.2. X-Window-Protokoll

Mit dem X-Window-Protokoll kann man graphische Ausgaben zum Client (dieser wird als Server bezeichnet) und Eingaben zum Server (das ist der eigentliche Client) schicken.

Ähnlich: Windows Terminal Server

14. E-Mail

siehe Foliensatz *E-Mail*