Rechnernetze & vert. Systeme

0: Einführung

0.1 – Enstehung des Internets

ARPANET
Entstanden 1969 als Universitätsnetz an der Westküste der USA. Bis 1977 58 beteiligte Knoten (USA, inkl. London).

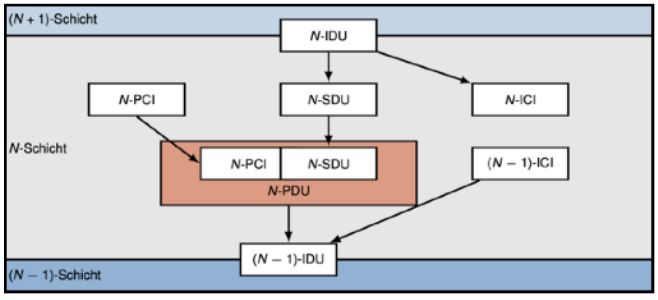
Web 2.0
Das Internet als Plattform.
- Services, keine statische Software mehr
- Partizipation
- Kosteneffiziente Skalierbarkeit
- Schwarmintelligenz

0.2 – Schichtenmodelle

Wozu Schichtenmodelle?
Schichtenmodelle dienen der Unterteilung komplexer Kommunikationsvorgänge. Die höher liegenden Schichten können hierzu auch Dienste der niedrigeren Schichten zugreifen.

ISO/OSI Model
Das ISO/OSI (Open Systems Interconnection Model) unterteilt die Architektur zur Kommunikationsfunktionalität in 7 Schichten:

7. Anwendungsschicht
6. Darstellungsschicht
5. Sitzungsschicht
4. Transportschicht
3. Vermittlungsschicht
2. Sicherungsschicht
1. Physikalische Schicht



Die Horizontale Kommunikation erfolgt physisch ausschließlich über einen Kanal, den die physikalische Schicht bespielt (sonst nur logische Kommunikation).

IDU	Interface Data Unit	N-Schicht erhält IDU von $N+1$-Schicht (SDU & ICI)
SDU	Service Data Unit	Nutzdaten
ICI	Interface Control Information	Kontrollinformationen (z.B. Länge SDU, Adressinfor.)
PDU	Protocal Data Unit	Von N-Schicht erzeugt (SDU & PCI)
PCI	Protocol Control Information	Informationen für die N-Schicht der Gegenseite

Schwächen des ISO/OSI Models
- Trennung der Schichten widerspricht teilweise anderen Interessen (Bsp. Effizienz)
- Manche Protokollmechanismen sind nicht klar zu einer Schicht zuzuordnen

1: Physikalische Schicht

1.1 – Signale, Informationen & deren Bedeutung

Definitionen

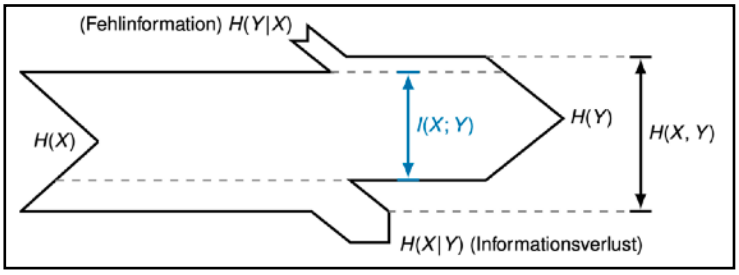
- Singale: zeitabhängige und messbare physikalische Größen
- Symbole: definierte messbare Signaländerung
- Informationsgehalt: wieviel Information wird durch Zeichen/Symbole übertragen
  > Je seltener eine Zeichen auftritt, desto höher der Informationsgehalt.
  > Informationsgehalt einer Zeichenkette ist die Summe des Informationsgehaltes der einzelnen Zeichen.
  > Der Informationsgehalt eines vorhersagbaren Zeichens ist 0.
- Information: besteht in der Unsicherheit, die Veränderung eines Signals vorhersagen zu können. Der Informationsgehalt ist somit abhängig von der Auftrittswahrscheinlichkeit eines Zeichens.
  > $I(x) = -\log_2 p(x) \quad [I] = \text{bit}$
- Entropie: mittlerer Informationsgehalt einer Quelle
  > $H(X) = \Sigma_{x\in\mathcal{X}}\ p(x)\cdot I(x) = -\Sigma_{s\in\mathcal{X}}\ p(x)\cdot\log_2(p(x))$
- Bedingte Entropie: Sind X und Y voneinder abhängig, so ist die bedingte Entropie kleiner als im unabhängigen Fall
  > $H(Y\,|\,X) = \Sigma_{x\in\mathcal{X}}\ p(x)\cdot H(Y\,|\,X=x) = -\Sigma_{x\in\mathcal{X}}\ p(x)\cdot\Sigma_{y\in\mathcal{Y}}\ p(y\,|\,x)\ \log_2\ p(y\,|\,x)$
- Verbundenentropie: Informationsgehalt zweier Nachrichtenquellen X und Y
  > $H(X,Y) = -\Sigma_{x\in\mathcal{X}}\Sigma_{y\in\mathcal{Y}}\ p(x,y)\ \log_2 p(x,y)$

Informationstheoretisches Modell: Gedächtnisloser Kanal

Die Entropie auf der Empfangsseite entspricht

$$H(Y) = H(X) - H(X\,|\,Y) + H(Y\,|\,X)$$

Sendeentropie – Verlust + Fehlinform.



- Transinformation: Information, die von Sender zum Empfänger über einen gedächtnislosen Kanal transportiert wird (eng. Mutual Information)
  > $I(X;Y) = H(X) - H(X\,|\,Y) - H(Y\,|\,X)$

1.2 – Signaldarstellung

Fourierreihe

Periodische Zeitsignale lassen sich als Überlagerung von Sinus- und Konsinusschwingungen unterschiedlicher Frequenzen auffassen.

$$s(t) \approx \frac{a_0}{2} + \sum_{k=1}^{\infty}(a_k\cos(k\omega t) + b_k\sin(k\omega t)) \qquad \text{für } \omega = 2\pi/T \qquad \frac{a_0}{2} \text{ Verschiebung}$$

Das k-te Summenglied bezeichnet man als k-te Harmonische.
Die Fouriertransformation ist nur bei periodischen Signalen möglich.

1.3 – Abtastung, Rekonstruktion und Quantisierung

Abtastung, Rekonstruktion und Quantisierung

Natürliche Signale sind zeit- und wertekontinuierlich und können von Computern (diskret) daher nicht direkt verarbeitet werden. Eine Diskretisierung der Signale ist nötig im

- Zeitbereich (Abtastung)
- Wertbereich (Quantisierung)

Abtastung

Das Signal wird in regelmäßigen zeitlichen Abständen (T_a) abgetastet.

$$\hat{s}(t) = s(t) \sum_{n=-\infty}^{\infty} \delta[t - nT_a] \quad \text{mit } \delta[t - nT_a] = \begin{cases} 1 & t = nT_a \\ 0 & \text{sonst} \end{cases}$$

Man erhält zeitdiskrete aber wertkontinuierliche Signale ($\hat{s}[n]$).

Rekonstruktion

Mittels der Abtastwerte $\hat{s}[n]$ ist es möglich, das ursprüngliche Signal $s(t)$ zu rekonstruieren.

$$s(t) \approx \sum_{n=-\infty}^{\infty} \hat{s}[n] \cdot \sin c\left(\frac{t - nT_a}{T_a}\right)$$

Die Abtastwerte sind dabei die Stützstellen für die trigonometrische Interpolation.

FALTUNG

Abtasttheorem

Ein auf $|f| \leq B$ bandbegrenztes Signal $s(t)$ ist vollständig durch äquidistante Abtastwerte $\hat{s}[n]$ rekonstruierbar, sofern diese nicht weiter als $T_a \leq 1/2B$ auseinander liegen. Die Abtastfrequenz, welche eine vollständige Signalrekonstruktion erlaubt, ist folglich durch

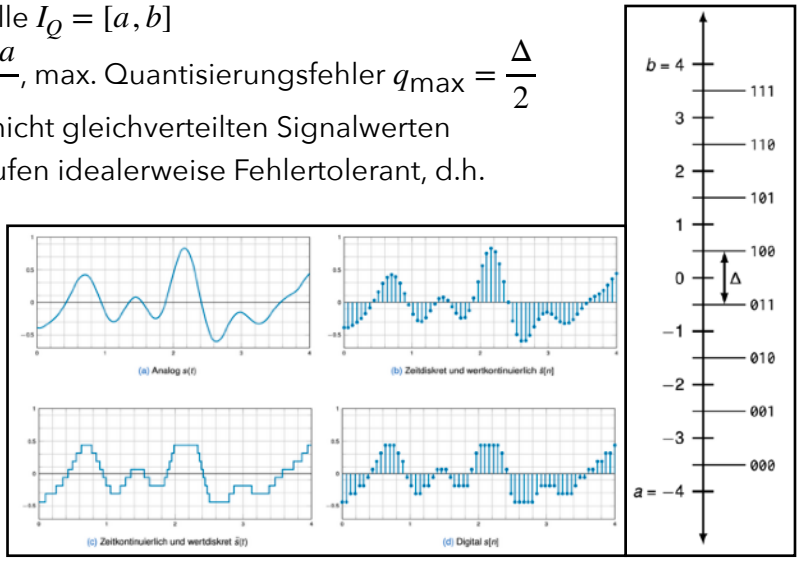
$$f_a > 2B$$

nach unten beschränkt.

Quantisierung

Die Abtastwerte $\hat{s}[n]$ sind kontinuierlich > müssen quantisiert werden.

- Unterscheidung in $M = 2^n$ Signalstufen mit Codewörtern von n bit
- Einteilung in Quantisierungsintervalle $I_Q = [a, b]$
  - > Lineare Quantisierung: $\Delta = \frac{b-a}{M}$, max. Quantisierungsfehler $q_{\max} = \frac{\Delta}{2}$
  - > Nicht lineare Quantisierung bei nicht gleichverteilten Signalwerten
- Zuweisung Codewörter → Signalstufen idealerweise Fehlertolerant, d.h. niedrige Hamming-Distanz zwischen benachbarten Codewörtern.



1.4 – Übertragungskanal

Kanaleinflüsse

Zur Modellierung von Kanaleinflüssen wird ein linearer zeitinvarianter Kanal mit einem Ein- und Ausgang verwendet.

- Berücksichtigt: Dämpfung, Tiefpassfilterung, Verzögerung, Rauschen (Additive White Gausian Noise)
- nicht Berücksichtigt: Interferenzen, Relektionen, Verzerrung, zeitvariante Einflüsse

Kanalkapazität

Vereinfacht wird angenommen, dass durch Tiefpass (niedrige Frequenzen passieren, hohe werden abgeschnitten/gedämpft) für die Bandbreite folgende Grenze gilt:

Frequenzanteil $|f| < B$ passiert Frequenzanteil $|f| < B$ wird gesperrt

Die Kanalkapazität ist durch die Kanalbandbreite B , einer Anzahl M von unterscheidbaren Signalstufen und einem Verhältnis zwischen Nutzsignal und Rauschen.

- Nyquist-Rate: mit Abtasttheorem $f = 2B$
- Hartleys Gesetz: $C_H = 2B \log_2(M)$bit
- Signal to Noise Ratio: $\text{SNR}[db] = 10 \cdot \log_{10}(\frac{\text{Signalleistung}}{\text{Rauschleistung}})$

Shannon Hartley etc. pp.

1.5 – Nachrichtenübertragung

Quellenkodierung

Ziel der Quellenkodierung ist das Entfernen von Redundanzen durch Abbildung von Bitsequenzen auf Codewörter (verlustfreie Datenkompression), z.B. Huffmann-Code.

Kanalkodierung

Die Kanalkodierung füght Redundanz zur Erkennnung und Behebung von Bitfehlern hinzu. In digitalen Systemen können Übertragungsfehler schwerwiegende Konsequenzen haben.

- Blockcodes unterteilen den Datenstrom in Blöcke der Länge k und übersetzen diese in Kanalwörter der Länge $n > k$, wobei die zusätzlichen $n - k$ bit der Fehlererkennung und Rekonstruktion dienen.

  $\frac{k}{n}$ Coderate/Datenrate

Leitungskodierung

Leitungscodes definieren die Abfolge einer bestimmten Art von Grundimpulsen zur Repräsentation von Bits/Gruppen von Bits (Sendeimpuls).

- Eigenschaften: Anzahl Signalstufen (binär, ternär, …), Anzahl kodierter Bits pro Symbol, Schrittgeschwindigkeit (Symbolrate/Baurate)
- Optionale Eigenschaften: Taktrückgewinnung (Sender und Empfänger haben selben Takt), Gleichstromfreiheit (konstanter Mittelwert des Signalpegels auf der Übertragungsleitung)

Leitungskodierung: Grundimpulse

- Rechteckimpuls: einfache Darstellung im Zeitbereicht; abrupte Signalwehcsel schwer umsetzbar, langsam abklingendes Spektrum > hohe Frequenzanteile
- $\cos^2$-Impuls: einfache praktische Umsetzung, schnell abklingendes Spektrum; erschwerte Abtastung bei nicht taktsynchronen Empfängern/Sendern

Leitungskodierung: Leitungscodes

- Non-Return-To-Zero (NRZ): binärer Code, 1 Symbol/bit, keine Taktrückgewinnung, keine Gleichstromfreiheit, relativ breites Spektrum
  > Sendeimpuls $g(t) = \text{rect}(t/T)$
  > Sendesignal $s(t) = \sum_{n=0}^{\infty} d_n \, g(t - nT), d_n = \begin{cases} 1 & b_n = 1 \\ -1 & b_n = 0 \end{cases}$
- Return-To-Zero (RZ): binärer Code, 2 Symbole/bit, einfache Taktrückgewinnung, keine Gleichstromfreiheit, breiteres Spektrum als NRZ
  > Sendeimpuls $g(t) = \text{rect}(\frac{2t + 0{,}5}{T})$
  > Sendesignals $s(t) = \sum_{n=0}^{\infty} d_n \, g(t - nT), d_n = \begin{cases} 1 & b_n = 1 \\ -1 & b_n = 0 \end{cases}$
- Manchester-Code: binärer Code, 2 Symbole/bit, einfache Taktrückgewinnung, Gleichstromfreiheit, beitres und langsam abklingendes Spektrum
  > Grundimpuls $g(t) = \text{rect}(\frac{2t + 0{,}5}{T}) - \text{rect}(\frac{2t - 0{,}5}{T})$
  > Sendesignals $s(t) = \sum_{n=0}^{\infty} d_n \, g(t - nT), d_n = \begin{cases} 1 & b_n = 1 \\ -1 & b_n = 0 \end{cases}$
- Multi-Level-Transmit 3 (MLT 3): tenärer Code, 1 Symbol/bit, keine Taktrückgewinnung, keine Gleichstromfreiheit, schmales Spektrum
  > Grundimpuls $g(t) = \text{rect}(t/T)$
  > Sendesignals $s(t) = \sum_{n=0}^{\infty} d_n \, g(t - nT), d_n = \sin(\frac{\pi}{2} \sum_{k=0}^{n} b_k$

2: Sicherungsschicht

2.1 – Darstellung von Netzwerken als Graphen

Aufgaben der Sicherungsschicht

- Steuerung des Medienzugriffs (Koordination auf Bus [Kollision = Datenverlust])
- Prüfung der Nachrichten auf Fehler (Fehler → neu senden)
- Adressierung innerhalb von Direktverbindungsnetzen

Direktverbindungsnetze

- Alle angeschlossenen Knoten sind direkt erreichbar, es wird an alle kommuniziert und der Empfänger anhand einer Adresse indentifiziert
- Keine Vermittlung, einfache Weiterleitung durch Bridging oder Switching

Gerichtete Graphen

Netztopologien werden häufig als gerichtete Graphen dargestellt: $G = (N, A)$

N Knoten $A = \{(i,j) \mid i,j \in N \wedge i,j \text{ sind gerichtet verbunden}\}$

Ungerichtete Graphen

Ein symmetrisches Netz lässt sich als ungerichteter Graph darstellen: $G = (N, E)$

N Knoten $E = \{\{i,j\} \mid i,j \in N \wedge i,j \text{ sind ungerichtet verbunden}\}$

Pfade, Pfadkosten & Pfadlängen

$P_{st} = \{(s,i),(i,j),\ldots,(k,l),(l,t)\}$ Pfad

$c(P_{st}) = \sum_{(i,j)\in P_{st}}$ Pfadkosten

$I(P_{st}) = |P_{st}|$ Pfadlänge/Hop Count

Netztopologie

Die Topologie beschreibt die Struktur der Verbindungen. Man unterscheidet physikalische Topologie (physikalische Verknüpfung, z.B. durch Kabel) und logischew Topologie (durch Protokolle definiert, z.B. einen Leader im Netzwerk).

- Wichtige Topologien: Punkt-zu-Punkt, Bus, Kette, Stern, Vermaschung, Baum (meist logisch)

Adjazenz- und Distanzmatrix

Netzwerke lassen sich als Matrizen schreiben. A ist für ungerichtete Graphen symmetrisch.

$$A = (a)_{ij} = \begin{cases} 1 & \exists (i,j) \in A \\ 0 & \text{sonst} \end{cases}, \quad \forall i,j \in N, \quad A \in \{0,1\}^{N\times N}$$ Adjazenzmatrix

$$D = (d)_{ij} = \begin{cases} c_{ij} & \exists (i,j) \in A \\ 0 & \text{wenn } i = j, \forall i,j \in N \\ \infty & \text{sonst} \end{cases}, \quad \forall i,j \in N, \quad D \in \mathbb{R}_{0_+}^{N\times N}$$ Distanzmatrix

Man erhält den kürzesten Pfad zwischen je zwei Knoten (apsd) durch Potenzierung bzgl. Des min-plus-Produkts: $D^n = D^{n-1} \oplus D \;:\; \min_{k\in N}\{d_{ik}^{n-1} + d_{kj}\}$. Dabei ist die Pfadlänge (n) durch die Anzahl der Knoten N beschränkt $\Rightarrow n < N$.

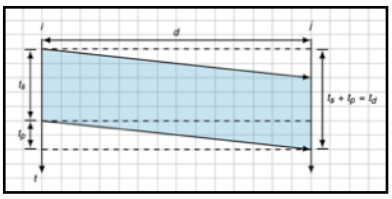
Erzeugung von Baumstrukturen

Ein Baum ist ein zusammenhängender, schleifenfreier Graph.

- Shortest Path Tree (SPT): Wurzelnknoten mit jeweils minimalen Kosten mit jedem anderen Knoten verbunden.
- Minimum Spanning Tree (MST): Verbindet alle Knoten im Netzwerk mit insgesamt minimalen Kosten.

2.2 – Verbindungschar., Mehrfachzugriff, Medienzugriffskontrolle

Verbindungscharakterisierung



Die Verbindung lässt sich anhand von *Übertragungsrate*, *Übertragunsverzögerung*, *Übertragungsrichtung* & *Mehrfachzugriff*

Zur Visualisierung werden Nachrichtenflussdiagramme verwendet.

- Übertragungsrate & Serialisierungszeit: Zeit um Daten auf Übertragungsmedium zu legen

  $t_s = \frac{L}{R}$, r ist Übertragungsrate
- Ausbreitungsgeschwindigkeit: Zeit von einem zum anderen Ende des Übertragungsmediums

  $t_p = \frac{d}{v \cdot c_0}$, $0 < v < 1$ ist relative Ausbreitungsgeschwindigkeit des Mediums
- Bandbreitenverzögerung: Speicherkapazität des Kanals (durch Ausbreitungsverzögerung)

  $C = t_p \cdot r = \frac{d}{v \cdot c_0} r$
- Übertragungsrichtung
  - > Simplex: Unidirektional
  - > Halbduplex: Bidirektional, immer nur ein Sender
  - > Vollduplex: Bidirektional, beide Sender gleichzeitig
- Mehrfachzugriff: Nachrichten mehrerer Teilnehmer koordiniert über eine Leitung übertragen

Übersicht Multiplexverfahren

- Zeitmultiplex: Aufteilung des Kanals in Zeit-Slots für jeden Kommunikationspartner
- Frequenzmultiplex: Aufteilung des Kanals in untersch. Frequenzbänder (spektrale Zerlegung)
- Raummultiplex: Verwendung paralleler Übertragungskanäle
- Codemultiplex: Verwendung orthogonaler Alphabete je Kommunikationspartner

Medienzugriff: Bewertungskriterien

- Durchsatz: Gesamtzahl an Nachrichten pro Zeiteinheit
- Verzögerung
- Fairness: Faire Aufteilung zwischen Teilnehmern für dasselbe Medium
- Implementierungsaufwand

Random Access

- ALOHA: senden mehrere Stationen gleichzeitig, kommt es zur Kollision > neu senden; erfolgreiche Übertragung werden auf anderem Kanal bestätigt

  $P[X_t = k] = \frac{\lambda^k \cdot e^{-\lambda}}{k!}$ Wahrscheinlichkeit, dass k Knoten senden (poisson)

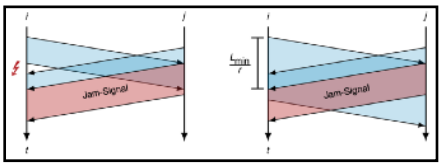
- Slotted ALOHA: wie ALOHA, allerdings kann Senden nur zu Beginn von def. Slots starten

Carrier Sence Multiple Access (CSMA)

Wie ALOHA, aber mit „Listen Before Talk".

- 1-persitentes CSMA: 1) wenn Medium frei, sende; 2) wenn Medium belegt, warte bis Medium frei, dann senden
- p-persistentes CSMA: 1) wenn Medium frei, sende mit Wahrscheinlichkeit p, verzögere mit Wahrscheinlichkeit $(1-p)$ um eine feste Zeit, dann 1)
- nicht-persitentes CSMA: 1) wenn Medium frei, sende; 2) wenn belegt, warte zufällig gewählte Zeitspanne, dann 1)

Carrier Sence Multiple Access/Collision Detection (CSMA/CD)

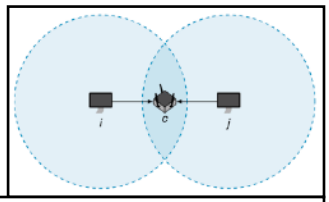


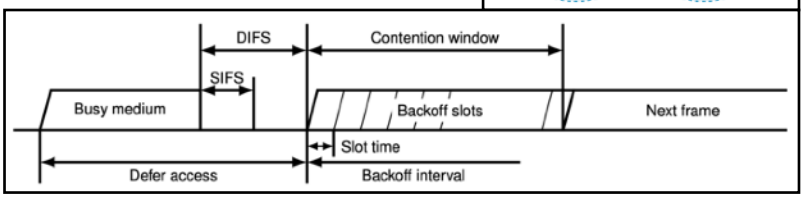
Kollisionserkennung ermöglicht Verzicht auf Sendungsbestätigung. Bei Kollision wird ein JAM-Signal gesendet. Nach Kollision warten beide Seiten zufällig lange und senden erneut (Binary Exponential Backoff: beim k-ten Sendeversuch einer Nachricht wartet der Sender zufällig $n \in [0; \min\{2^{k-1}-1, 1023\}]$ Slotzeiten).

- Voraussetzung: Nachrichten müssen Mindestlänge aufweisen

$$L_{\text{min}} = \frac{2d}{v \cdot c_0} r$$

Carrier Sence Multiple Access/Collision Avoidance (CSMA/CA)





In Funknetzen funktioniert CSMA/CD nicht. Fallbeispiel DCF:

- Feste Zeitintervalle zwischen Rahmen für Bestätigungen
- Zum Senden werden zufällige Backoff slots ausgewählt

Erweiterung durch RTS & CTS:

- Request to Send an Basisstation, Antwort Clear to Send, welches alle potentiellen Sender empfangen - der Requester sendet
- Andere potentielle Sender müssen eine definierte Zeitspanne abwarten

Token Passing

Ein Token zirkuliert im Ring. Wer sendet nimmt den Token & lässt Nachricht zirkulieren.
Empfänger markiert Nachricht und lässt sie weiter zirkulieren. Sender nimmt die Nachricht wieder aus dem Ring und lässt den Token weiter zirkulieren.
Verlorener Token werden von Monitor Station ersetzt.

+) effizient, da keine Wiederholung, garantierte maximale Verzögerung
–) Monitor Station nötig, Fehleranfällig, aufwändige Zusammenschaltung

2.3 – Rahmenbildung, Adressierung & Fehlererkennung

Codetransparenz

Verfahren zur Rahmenbegrenzung & zur klaren Unterscheidung von Steuerzeichen und Nutzlast.

- Längenfeld zu Beginn der Nachricht, Steuerzeichen an Anfang und Ende oder Coderegelverletzung
  > Bit Stuffing: Steuerzeichen in Nutzdaten über 4B5B-Code oder Escape Sequenzen herausfiltern

Adressierung

Adressierung in Direktverbindungsnetzen findet (ohne Routing) mit eindeutigen Schicht 2 Adressen statt. Überlicherweise werden MAC (Media Access Control) Adressen verwendet.

- Netzkarten besitzen ab Werk eine feste MAC Adresse
- Zusätzlich gibt es Broadcast und Multicast Adressen

Fehlererkennung

Trotz Kanalkodierung können Übertragungsfehler auftreten. Es werden Prüfsummen (eng. *checksums*) eingesetzt, um Fehler zu erkennen.

Cyclic Redundancy Check

CRC ist fehlererkennend (Einbut- & Burstfehler) und hat geringer Redundanz.

$$a(x) = \sum_{i=0}^{n-1} a_i x^i \quad a_i \in \mathbb{F}_2,\ \mathbb{F}_2 = \{0,1\} \qquad \text{Datenwort der Länge } n$$

$$F_q[x] = \{a \mid a(x) = \sum_{i=0}^{n-1} a_i x^i,\ a_i \in \mathbb{F}_2\} \qquad \text{Menge aller Datenwörter der Länge } n$$

Mit passend definierter Addition und Multiplikation entsteht ein endlicher Körper $\langle F_q[x], +, \cdot \rangle$. Eine passende Addition zweier Datenwörter entspricht einer XOR-Verknüpfung. Eine passende Multiplikation benötigt ein (irreduzibles: $r(x)$ kann nicht als Produkt zweier $a, b \in \mathbb{F}_q[x]$ dargestellt werden) Reduktionspolynom $r(x)$: $d(x) = ((a(x) \cdot b(x)) \mod r(x)$.

Wie funktioniert CRC?

Annahme: $\deg(r(x)) = n$, $\deg(m(x)) = k$.

1. Hänge n Nullen an $m(x)$ an: $m'(x) \cdot x^n$
2. Bestimme Divisionsrest $c(x) = m'(x) \mod r(x)$ (Prüfsumme)
3. Sende Nachricht $s(x) = m'(x) + c(x)$

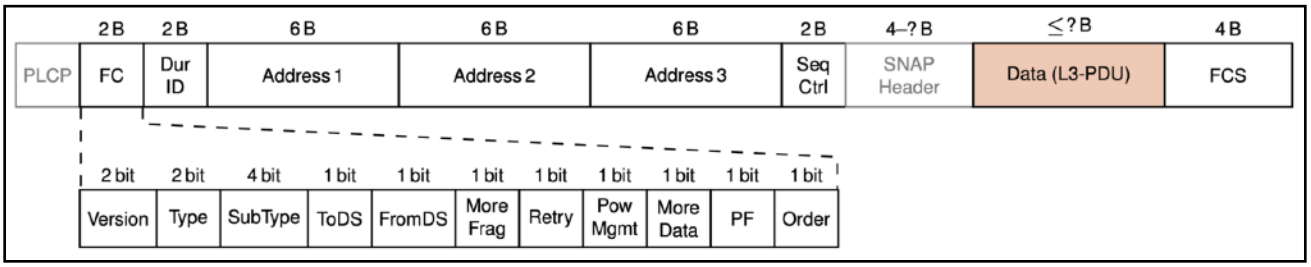
Sendeprozess & Prüfung durch Empfänger

1. Divisionsrest bestimmen: $c'(x) = s'(x) \mod r(x) = (s(x) + e(x)) \mod r(x)$
2. $c'(x) \neq 0 \Rightarrow$ Fehler $\quad c'(x) = 0 \Rightarrow$ wahrscheinlich kein Fehler

Fehlererkennung & Korrektur

- Erkennung: 1-bit Fehler, isolierte (mehr als n Abstand) 2-bit Fehler, manche Burst Fehler ($> n$)
- Korrektur: steht nicht im Vordergrund, bei ausreichender Redundanz aber möglich

Daten-Frame im Infrastructure Mode



- PLCP: Header Physical Layer, Synchronisation von Übertragungsparametern (Datenrate, Modulation, Coderate, etc.
- Frame Control (FC): Typ des Rahmens (Data/Control/Management), weitere Parameter
- Adressen: 1) direkten Empfänger, 2) direkter Absender, 3) Quelle/Empfänger
- Sequence Control: Sequenznummer des Rahmens (zur Ordnung)
- Subnetwork Access Protocol (SNAP): Typ der Daten (L3-PDU)
- Data: Daten variabler Länge
- Frame Check Sequence (FCS)

2.4 – Verbindung Schicht 1 & 2

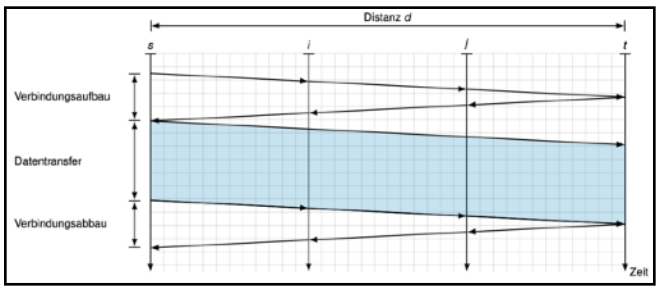
Hubs, Bridges & Switches

- Hub: verbindet einzelne Links zu gemeinsamen Bus, Rahmen erreicht alle Knoten
  > Nur ein gleichzeitiger Sender, gemeinsamer Kollisions-Domäne
  > aktive (Repeater): verstärken Signal; passive: nur Sternverteiler
  > Kaskadieren (5-4-3): 5 Abschnitte, 4 Repeater, 3 Abschnitte mit aktiven Endgerät
- Switch: learning: wie Hub mit Ports; productive: Weiterleitung über Switching-Table
  > Switch mit zwei Ports = Bridge
  > Switch, bzw. Bridge unterbricht Kollisions-Domäne
  > Pro Switchport ein Host = Microsegmentation/vollständiges geswitchtes Netz
- Transparente Kopplung: Stationen bemerken den Switch nicht
- Switching Arten: Store-and-Forward: vollständiges empfangen, FCS Prüfung, (evtl. Pufferung,) Senden; Cut-Through: direktes Weiterleiten, sobald Ausgangsport bestimmt

3: Vermittlungsschicht

3.1 – Vermittlungsarten

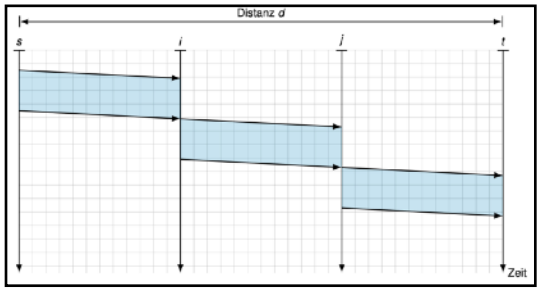
Leitungsvermittlung



1. Verbindungsaufbau: Austausch von Signalisierungsnachrichten (inkl. Routing) zum Aufbau einer dezidierten Verbindung
2. Datenaustausch: Daten versenden, Adressierung weitesgehend unnötig (point-to-point)
3. Verbindungsabbau: Austausch von Signalisierungsnachrichten zum Abbau der Verbindung

+) Konstante Qualität der Verbindung, schnelle Übertragung ohne Vermittlungsentscheidungen
–) Ressourcenverschwendung durch Reservierung zur exklusiven Nutzung
–) Zeitintensiver Verbindungsaufbau, hoher Aufwand beim Schalten physischer Verbindungen

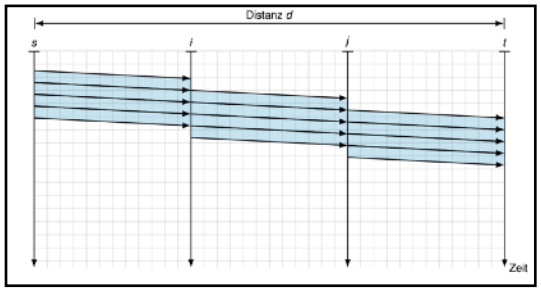
Nachrichtenvermittlung



- Nachrichten mit Header (Adressinformationen) -> PDU (als ganzes übertragen)
- Asynchrone Kommunikation, Empfänger muss zum Sendezeitpunkt nicht empfangsbereit sein
- Wegfallen von festen Pfaden ermöglicht gemeinsame Nutzung von Teilstrecken
- Zeitmultiplex (Time Division Multiplex, TDM)

+) Flexibles Zeitmultiplex, bessere Nutzung Kanalkapazität, keine Verzögerung durch Verbindungsaufbau
–) Pufferung bei hoher Auslastung (Verlust möglich)

Paketvermittlung



- Nachrichten werden in Pakete unterteilt und einzeln übertragen (unabhängig voneinder)
- Jedes Paket mit eigenem Header mit Informationen zu Adresse (& Reassemblierung)
- Multiplexing durch Vermittlung kleiner Pakete, wdh senden nur von Teilen einer größeren Nachricht

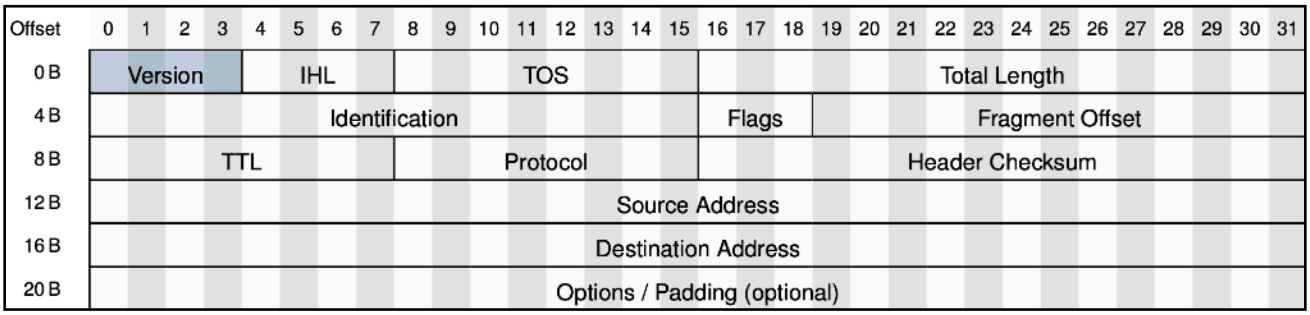
+) Flexibles Zeitmultiplex, Pufferung kleiner Pakete
–) Pufferung bei hoher Auslastung (Verlust möglich)
–) Header pro Paket -> Overhead in Paketen & Zusammensetzung

Einsatzgebiete

- Leitungsvermittlung: analoge Telefonverbindung, Interneteinwahl (letze Meile)
- Nachrichtenvermittlung: nicht auf Schicht 3
- Paketvermittlung: VoIP, digitales Fernsehen/Radio, Peripherieschnittstellen an Computern, etc.

3.2 – Adressierung mit IPv4

IPv4



- Version: verwendete IP-Version (IPv4/IPv6)
- IHL (IP Header Length): Länge des Headers in Vielfachen von 32 bit
- TOS (Type of Service): Klassifizierung & Priorisierung
- Total Length: Gesamtlänge des Paketes (Header + Daten), max 1500B
- Identification: Identifikation zusammenhängender Fragmente
- Flags: (16) reserviert auf 0; (17) 0>Fragmentierung erlaubt, 1>Fragmentierung verboten; (18) 0>keine weiteren Fragmente, 1>weitere Fragmente folgen
- Fragment Offset: absolute Position der Daten als Vielfaches von 8B
- TTL (Time to Live): Zähler für maximale Hops (verhindert Routing Loops)
  > decrement des TTL = decrement der Prüfsumme
- Protocol: Schicht 4 Protokoll zur Prozesszuordnung im Betriebssystem (z.B. UDP, TCP)
- Header Checksum: Prüfsumme für Header (nur fehlererkennend)
- Source Adress: Absenderadresse
- Destination Adress: Zieladresse (Empfänger)
- Options/Padding: weitere Optionen (z.B. Zeitstempfel, Route Recording, etc.) als optinale Fehler + Padding auf 4B

Network-Byte-Order

Die Network-Byte-Order ist Big Endian. Host-Byte-Order ist häufig Little Endian.

```
htons(0x0001) = 0x0100  // Host to Network
ntohs(0x0100) = 0x0001  // Network to Host
```

IPv4-Adressauflösung

Im selben Netz:

- Source sendet ARP-Request zur Auflösung der IP-Adresse zur MAC-Adresse (mit seiner eigenen IP- und MAC-Adresse > Request wird als Broadcast versendet
- Destination antwortet mit ARP-Reply (Unicast)

In unterschiedlichen Netzen:

- Falls die Destination nicht im selben Netz liegt, wird an Router (mittels MAC-Adresse) gesendet
- Der Router akzeptiert & bestimmt ausgehendes Interface
- Router sendet an Zielhost anhand MAC-Adresse (möglicherweise weiterer ARP-Schritt nötig)

MAC-Adressen dienen zur Adressierung innerhlab eines Direktverbindungsnetzes & werden beim Forwarding verändert. IP-Adressen dienen der End-zu-End Adressierung zwischen mehreren Netzen und werden beim Forwarding *nicht* verändert.

- Ergebnisse der Adressauflösung werden im ARP-Cache des Hosts zwischengespeichert.
- ARP-Replies können auch Broadcast versendet werden (unsolicited ARP replies) > speichern.

Ping

- Host wählt zufälligen Identifier (wird für jeden Echo-Request inkrementiert)
- Echo-Request wird von Router wie üblich weitergeleitet
- Zielhost antwortet mit Echo Reply (selber Identifier, Sequenznummer & Daten wie Request)
  > Fehler: ICMP-Nachricht an Host (Internet Control Message Protocol), z.B. durch exceed. TTL

Traceroute

- Host sendet ICMP Echo Request an Zielhost, TTL = 1
- Schrittweises erhöhen des TTL-Feldes um 1

DHCP

IP Adressen innerhalb eines Netzes können statisch von Hand, oder automatisch & zeitlich begrenzt (Lease) durch einen DHCP (Dynamic Host Configuration Protocol) festgelegt werden.

1. Client sendet DHCP-Discover (Layer 2 Broadcast)
2. DHCP-Server antwortet mit DHCP-Offer (bietet IP Adresse an)
3. Client antwortet mit DHCP-Request (Anforderung der Adresse)
4. DHCP-Server antwortet mit DHCP-ACK (Freigabe) oder DHCP-NACK (Untersagung)

- DHCP-Server können weitere Optionen ausliefern: DNS-Resolver, statische Routen (Default Gateway, NTP-Server Zeitsynchronisation, etc.)
- Redundante DHCP-Server mit disjunkten Adressbereichen möglich

Adressklassen

Klasse	1. Oktett	1. Adresse	Letzte Adresse	Netz/Host	Anzahl Netze	Adressen pro Netz
A	0xxxxxxx	0.0.0.0	127.255.255.255	N.H.H.H	$2^7 = 128$	$2^{24} = 16777216$
B	10xxxxxx	128.0.0.0	191.255.255.255	N.N.H.H	$2^{14} = 16384$	$2^{16} = 65536$
C	110xxxxx	192.0.0.0	223.255.255.255	N.N.N.H	$2^{21} = 2097152$	$2^8 = 256$
D	1110xxxx	224.0.0.0	239.255.255.255	*Reserviert für multicast*		
E	1111xxxx	240.0.0.0	255.255.255.255	*Reserviert*		

Adressräume

Adressräume werden durch die IANA verwaltet (zu Beginn ineffizient). Mittlerweile ist der IPv4-Adressraum praktisch aufgebraucht.

Subnetting

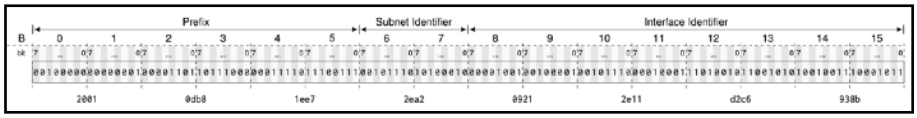
Zusätzlich zur IP-Adresse erhält ein Interface eine (ebenfalls 32 bit lange) Subnetzmaske zur Unterteilung in Netz- (1) & Hostanteil (0). Die Broadcastadresse entspricht der letzen Adresse im jeweiligen Subnetz.

Supernetting

Besondere Adressbereiche

- `0.0.0.0/8`: Hosts in diesem Netzwerk
  > Verwendung z.B. vor Adresszuweisung durch DHCP
- `127.0.0.0/8`: Loopback-Adressen (localhost)
  > Lokaler Rechner
- `10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16`: private Adressbereiche
  > Kein öffentliches Routing, darf innerhalb privater Netze beliebig vergeben werden
- `169.254.0.0/16`: automatic private IP adressing (APIPA)
- `255.255.255.255/32`: Global Broadcast
  > Identifiziert alle Hosts, wird niemals geroutet

3.3 – Adressierung mit IPv6

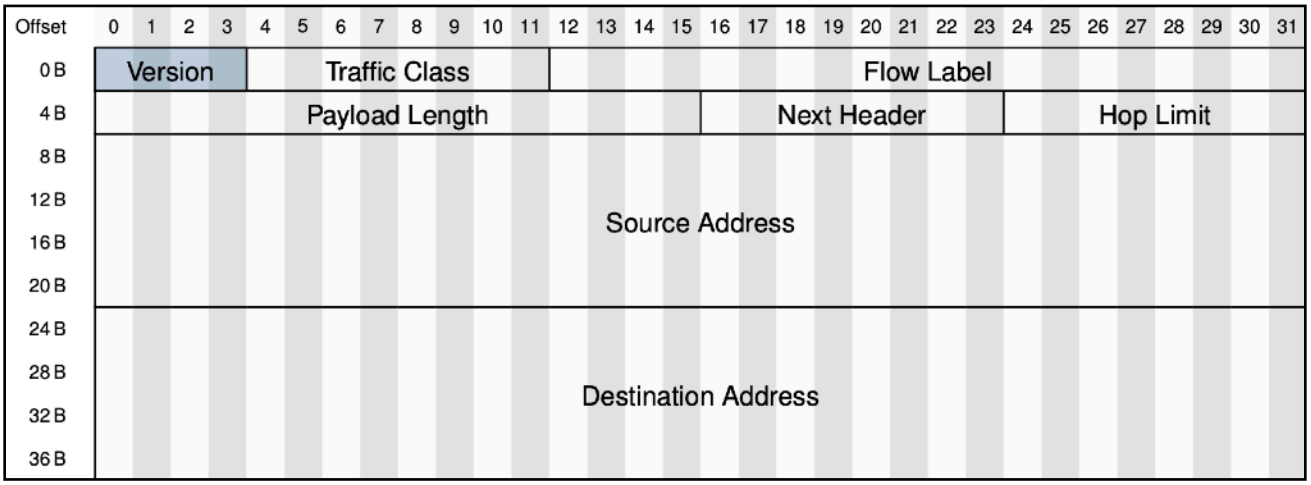


Hintergrund IPv6

Aufgrund sich andeutender Adressknappheit bereits 1995 als Nachfolger von IPv4 vorgeschlagen & 1998 standardisiert.

- Notation: zu je 16 bit gruppiert, getrennt durch Doppelpunkte, in hex. Schreibweise
  > Führende Nullen einzelner Blöcke werden weggelassen
  > Eine Gruppe konsekutiver Blöcke aus Nullen darf mit `::` abgekürzt werden
  > `::` wird für die längste Sequenz von Nullen verwendet (falls ambivalent die Erste)
  > Ein einzelner Null-Block darf nicht mit `::` abgekürzt werden

IPv6



- Version: verwendete IP-Version (IPv4/IPv6)
- Traffic Class: äquivalent zu TOS, Klassifizierung & Priorisierung
- Flow Label: Erkennung zusammenhängender Pakete
- Payload Length: Länge der angehängten Daten (inkl. Extension Header) als Vielfaches von 8B
- Next Header: Typ des nächsten Headers (L4-Header, ICMP oder Extension Header)
- Hop Limit: vgl. TTL, dekrementierung bei Weiterleitung, 0 > ICMPv6 Time Exceeded an Sender
- Source Address: 128 bit IPv6 Quelladresse
- Destination Address: 128 bit IPv6 Zieladresse

IPv6 Extension Header

Extension Header erlauben zusätzliche L3-Informationen anzufügen.

- Fragment Header: Next Header$_8$, Reserved$_8$, Fragment Offset$_{15}$, Reserved$_2$, More Fragments$_1$, Identification$_{32}$

Besondere Adressbereiche

- `::/128`: nicht-spezifizierte Adresse (siehe Hosts in diesem Netzwerk, kein Routing)
- `::1/128`: Loopback-Adressen (localhost)
  > Lokaler Rechner
- `fe80::/10`: Link-Local Adresse
  > Gültig innerhalb eines lokalen Links (kein Routing)
- `fc00::/7`: Unique-Local Unicast-Adressen
  > Für lokale Kommunikation, rounting nur lokal und nicht im globalen Netz
- `ff00::/8`: Multicast-Adressen (werden geroutet)

Multicast

- Unicast: an ein Ziel adressiert, Nicht-Ziele verwerfen oder leiten weiter
- Broadcast: an alle Stationen in Netzwerk adressiert, meist auf lokales Netzwerk begrenzt
- Multicast: an Gruppe von Knoten adressiert, spezielle Protokolle für Multicast über lokales Netzsegment hinaus
- Anycast: an beliebige Stationen einer bestimmten Gruppe
- `ff02::1`: All Nodes (alle Knoten auf lokalem Link)
- `ff02::2`: All Routers (alle Router auf lokalem Link)
- `ff02::1:2`: All DHCP-Agents (alle DHCP-Server auf lokalem Link)
- `ff02::1:ff00:0/104`: Solicited-Node Adress (Neighbour Discovery Protocol zur Adressaufl.)

Mapping Multicast IPv6 → MAC

Die ersten 2 Oktette der MAC-Adresse werden auf 33:33 gesetzt.

- Letztes Bit des ersten Oktetts gesetzt → Multicast
- Vorletztes Bit des ersten Oktetts gesetzt → locally administered

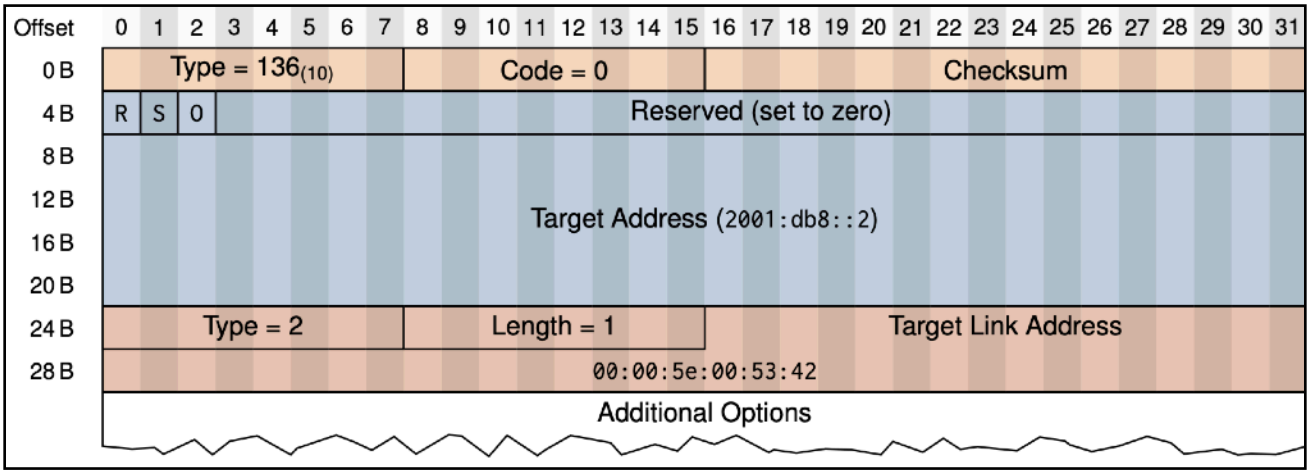
Stateless Address Autoconfiguration

TODO

Internet Control Message Protocol (v6)

- Type: Art der Nachricht, z.B. Echo, Time Exceeded, usw.
- Code: präzisiert Type, z.B. um es sich um Echo-Request oder Echo-Response handelt
- Checksum: fehlerkennende Checksumme über das gesamte ICMPv6-Paket
- Payload: Inhalt (Bsp. folgen)

Neighbour Discovery Protocol



- ICMPv6 Header: Type & Code (`0x88` und `0x00`) + Checksumme
- Neighbour Discovery Body
  > R: 1, falls antwortender Knoten ein Router ist
  > S: 1, falls Advertisement infolge einer Solication gesendet wurde (siehe Multicast)
  > O: 1, falls das Advertisement eine (pot) gecachte Link-Adresse beim Empfänger aktualisieren soll
- Neighbour Discovery Options
  > Type: Typ, z.B. 2 für Target Link Layer Address
  > Length: Gesamtlänge als Vielfaches von 8B
  > Target Link Adress: falls Knoten angefragt wurde, L2-Adresse des angefragten Knotens

Kompatibilität:
IPv4 und IPv6 sind nicht kompatibel, können aber nebeneinander existieren (Dual Stack).

3.3 – Wegwahl (Routing)

Routing Table

Im Routing Table speichert ein Router (oder Host)
- Netzadresse des Ziels
- Länge des Präfixes (vgl. Subnetzmaske)
- Next-Hop (Gateway), `0.0.0.0` = eigenes Netz
- Interface zum Erreichen des Gateways
- Kosten zum Ziel

Destination	NextHop	Costs	Iface
192.168.255.8/30	0.0.0.0	0	eth2
192.168.255.0/29	0.0.0.0	0	eth1
192.168.0.0/24	0.0.0.0	0	eth0
172.16.1.0/24	192.168.255.3	1	eth1
172.16.0.0/23	192.168.255.2	1	eth1
0.0.0.0/0	192.168.255.10	0	eth2

Longest Prefix Matching (statisch)

Rountingtabelle wird von längeren Präfixen (spezifischer) zu kürzeren Präfixen durchsucht
1. R1 berechnet logisches AND aus Zieladresse und Subnetzmaske aus Routing Table
2. Ergebnis mit Destination vergleichen > passt > Gateway & Interface bestimmen
3. ggf. MAC Adresse mit ARP auflösen und Paket mit neuen Ethernet-Header weiterleiten

- `0.0.0.0/0` liefert immer einen Match und leitet an das Default Gateway weiter.
- Routen zu direkt verbundenen Netzen können automatisch erzeugt werden (= Next-Hop)
- Routen zu entfernten Netzen werden händisch (statisch) oder durch Routing-Protokolle (dynamisch) eingetragen

Distanz-Vektor-Protokolle

- Router kennen nur Richtung (Next-Hop) und Entfernung (Kosten) zum Ziel
- Router haben keine Informationen zur Netzwerktopologie, nur kummulierte Kosten

Algorithmus von Bellman-Ford, SSSP, $O(|N| \cdot |E|)$, keine komplexe Datenstruktur notwendig, verteilte (dezentrale) Implementierung ohne Kenntnis der Topologie möglich.
RIP, IGRP, EIGRP, AODV

Link-State-Protokolle

- Router speichern Kosten & Route
  > komplexe Nachbarschaftsbeziehungen & Update-Nachrichten nötig
- Router haben vollständige Topologieinformationen

Dijkstras Algorithmus, ASSP, $O(|E| + |N| \cdot \log_2(|N|))$, Ressourcenintensiv durch kompl. Datenstruktur, vollständige Netzwerktopologie bekannt, asymptotisch bessere Laufzeit, Greedy.
OSPF, IS-IS, HWMP

Routing Information Protocol (dynamisch)

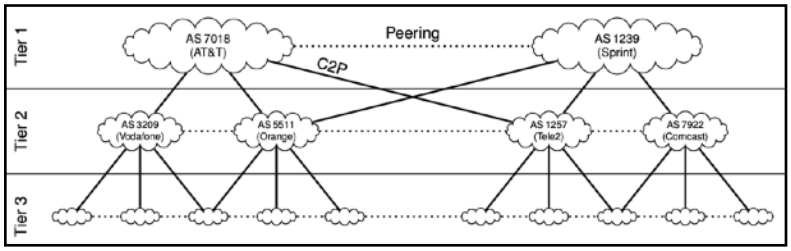
Distanz-Vektor Protokoll, Metrik: Hop Count (mit Limit von 15)

1. In Regelmäßigen Abständen (30s) Routingtabelle an Multicast-Adresse `224.0.0.9`
2. Router akzeptieren Update, inkrementieren Kosten um 1, und verleicht mit eigener Routingtabelle, evtl. Updates
3. Bleiben fünf Updates eines Nachbarn aus, werden die Routen über diesen Nachbarn gelöscht

- Bei gewichteten Kanten Kosten entsprechend der Kosten des Links inkrementieren
- Triggered Updates: falls es zur Änderung kommt, wird direkt an alle weitergeleitet
- Count to Infinity Problem: Verbindung fällt aus, aber andere Router im Netz verbreiten fehlerhafte Route weiter (bis Kosten = 15)
  > Split Horizon: Update wird nicht an ‚Lehrer' gesendet
  > Poison Reverse: Route mit unendlicher Metrik an ‚Lehrer'
  > Path Vector: Update enthält vollständigen Pfad > nur Update, falls man selbst nicht im Pfad enthalten ist (verhindert Rounting Loops & Count to Infinity)

Autonome Systeme

Eine Menge von Netzwerken, die unter einheitlicher administrativer Kotrolle stehen, bezeichnet man als Autonomes System (AS) > Policy Based Routing.
Innerhalb des autonomen Systems werden Interior Gateway Protocols (IGP) [RIP, OSPF, EIGRP, IS-IS] eingesetzt.
Zwischen autonomen Systemen wird ein Exterior Gateway Protocol (EGP) verwendet.



4: Transportschicht

4.1 – Multiplexing

Multiplexing

Auf der Transportsschicht werden Datenströme in Segmente mit Headern unterteilt.

(SrcIPAddr, SrcPort, DstIPAddr, DstPort, Protocol) 5-Tupel Header

Portnummer erlaubt dem OS ein Mapping von Sockets > File-Deskriptor > Anwendung.

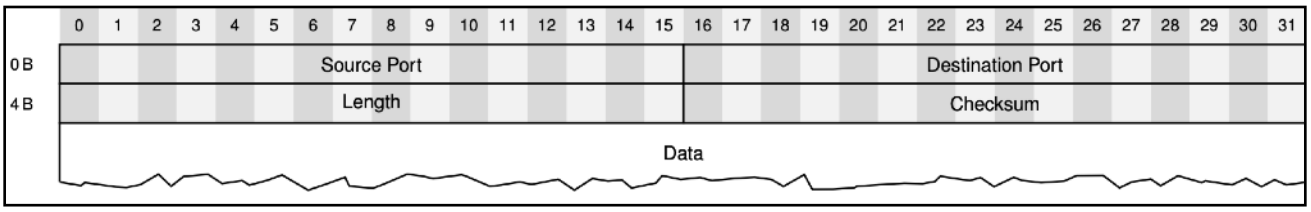
4.2 – Verbindungslose Übertragung

User Datagram Protocol (UDP)

Segmente werden unabhängig voneinander und zustandslos versendet.

- Pakete können verloren gehen
- Reihenfolge kann durch unterschiedliches Routing falsch sein

ungesicherte, verbindungslose, nachrichtenorientiere *Kommunikation*



- Length als Vielfaches von Byte
- Checksum über Header und Daten (nur bei IPv6 zwingend, sonst 0)
  > Berechnung mittels Pseudo Header

+) Geringer Overhead, keine Verzögerungen (Echtzeitanwendungen)
+) Kein Einfluss durch Fluss- und Staukontrollmechanismen
–) Keine Qualitätssicherung, beliebig hohe Fehlerrate
–) Reihenfolge nit definiert
–) Keine Fluss- (pot. Überforderung des Empfängers) und Staukontrollmechanismen (Überlast im Netz möglich)

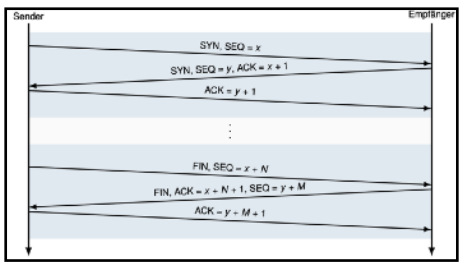
4.3 – Verbindungsorientiete Übertragung

Verbindungsorientiete Übertragung

Linear nummerierte Segmente (Segmentnummern).

+) Bestätigung erfolgreicher Übertragung
+) Identifikation fehlender Segmente & erneutes Anfordern der Segmente
+) Zusammensetzen der Segmente in korrekter Reihenfolge
–) Synchronisation & Zustand nötig

1. Verbindungsaufbau (Handshake)
2. Datenübertragung
3. Verbindungsabbau (Teardown)



Sliding Window-Verfahren

- Bisher hat der Sender jeweils nur ein Segment gesendet > Bestätigung abgewartet.
  > ineffizient, da RTT (Round Trip Time) limitierender Faktor

Sliding Window-Verfahren: definierte Anzahl von Segmenten kann übertragen werden > Bestätigung für alle (kumultative Bestätigung: ACK $= m + 1 \Rightarrow$ Bestätigung $\forall$SEQ $\leq m$)

+) Effizietere Zeitnutzung, Fluss- und Staukontrolle (durch Fenstergröße)

–) Sender & Empfänger benötigen Zustand, endlichen Sequenznummernraum verwalten

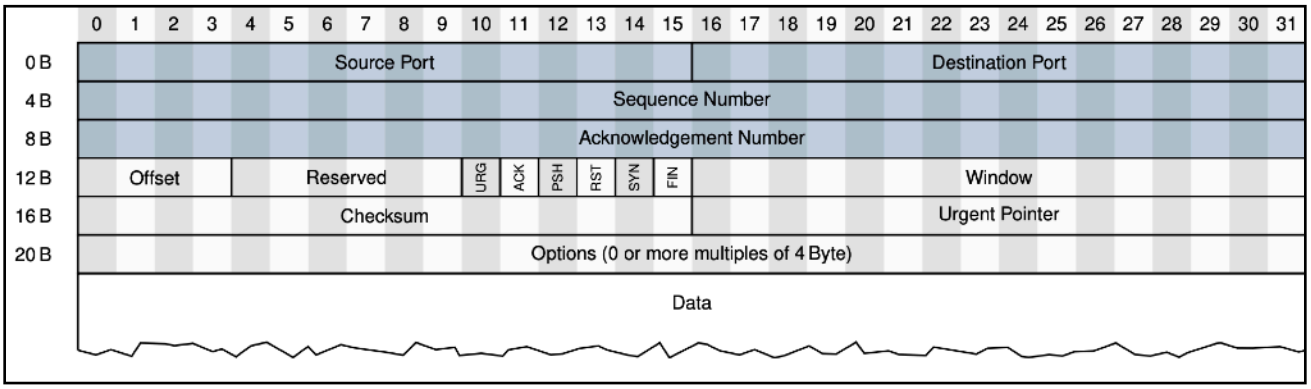
$W_s \subset S$ Sendefenster $W_r \subset S$ Empfangsfenster

Sende und Empfangsfenster verschieben & überlappen sich während des Datenaustauschs.

Bei Segmentverlust:

1. Go-Back-N: akzeptiere nur nächste Segmentnummer, sonst alles verwerfen
   > $w_s \leq N - 1$
2. Selective-Repeat: akzeptiere alles in W_r (gepuffert), frage verlorene Segmente neu an
   > $w_s \leq \lfloor \frac{N}{2} \rfloor$

Transmission Control Protocol (TCP)



- Quell- und Zielport (wie bei UDP)
- Sequenz- und Bestätigungsnummer: für gesicherte Übertragung und Bestätigung einzelner Bytes (Anfrage kann auch Bestätigung beeinhalten)
- Offset: Länge des Headers in Vielfachen von 4B
- Reserved: aktuell ohne Verwendung, default 0
- URG: urgent, Daten werden sofort an höhere Schicht weitergeleitet (falls gesetzt)
- ACK: ackknowledgement, es handelt sich um eine Empfangsbestätigung (falls gesetzt)
- PSH: push, sende- & empfangsseitige Puffer des TCP-Stacks werden umgangen (falls gesetzt)
- RST: reset, Abbruch der TCP-Verbindung ohne Verbindungsabbau (falls gesetzt)
- SYN: synchronization, Aufbau der TCP-Verbindung, initialer Austausch von Seq, (falls gesetzt)
- FIN: finish, Abbau der TCP-Verbindung (falls gesetzt)
- Windows: Größe des aktuellen Empfangsfensters W_r in Byte
- Checksum: Prüfsumme über Header und Daten (bei UDP mit Pseudoheader)
- Urgend Pointer: Ende der Urgent Daten, die bei gesetzten URG direkt an höhere Schicht weitergeleitet werden
- Options: weitere Optionen beliebiger Länge (Window Scaling, selektive Bestätigun, MSS, etc.)

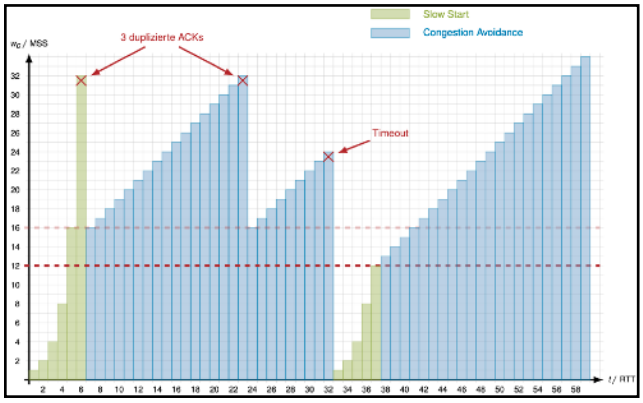
Maximum Segment Size

Die Maximum Segment Size (MSS) gibt die maximale Größe eines TCP-Segments (Nutzdaten ohne Header) an und sollte so gewählt sein, dass keine Fragmentierung nötig wird.

Stau- und Flusskontrolle

- Flusskontrolle: Überlastsituationen beim Empfänger vermeiden
  > Regulierung des Empfangsfensters W_r, gesetzt im Feld Receive Window
- Staukontrolle: Überlastung im Netz vermeiden
  > Regulierung des Staukontrollfensters W_c, Verlustfreiheit > (+), Verlust > (–)

$$w_s = \{w_c, w_r\}$$



1. Slow Start
   > Für jedes bestätigte Segment wird w_c bis zu einem Schwellwert um eine MSS vergrößert
   > Beginne Congestion Avoidance
2. Congestion Avoidance
   > Für jedes bestätigte Segment wird w_c um $1/w_c$ MSS vergrößert

TCP Reno

1. 3 duplizierte Bestätigungen
   > Schwellwert für Stauvermeidung auf $w_c/2$, w_c auf (neuen) Schwellwert setzen
   > Beginne Congestion Avoidance
2. Timeout
   > Schwellwert für Stauvermeidung auf $w_c/2$, $w_c = 1$ MSS
   > Beginne Slow Start

Risiken

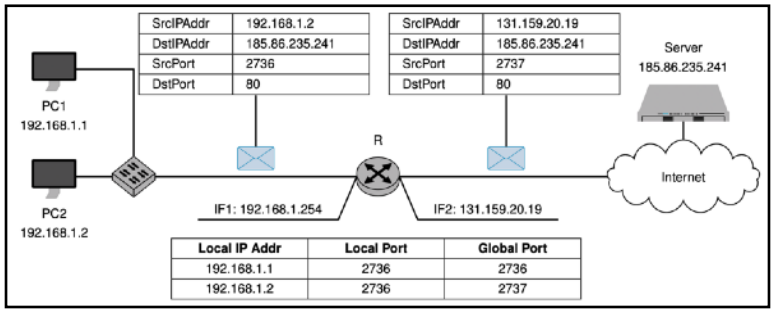
Die Schichten 1–3 müssen eine für TCP ausreichend geringe Paketfehlerrate bereitstellen, da Fehler als Metrik für Netzüberlastung herangezogen werden. Schlechte Übertragungsqualität könnte eine unnötige Drosselung zur Folge haben.

4.4 – Network Address Translation

Network Address Translation (NAT)

NAT ist eine Technik, die es erlaubt, N private IP-Adressen auf M öffentl. IP-Adressen abzubilden.

- $N \leq M$: statische oder dynamische Übersetzung, bei der jede private IP-Adresse mindestens einer öffentlichen IP-Adresse zugeordnet wird
- $N > M$: öffentliche IP-Adresse wird von meheren Computern über Port-Multiplexing genutzt



1. NAT-Tabelle zu Beginn leer
2. PC1 sendet an Server
3. Router legt Eintrag in NAT-Tabelle an & tauscht Absenderadresse
4. Server antwortet an Router
5. Router macht Adress-übersetzung auf Basis der NAT-Tabelle rückgängig

Teilweise muss ein zufälliger Port gewählt werden, da der gewünschte (SrcPort von PC1) Port bereits belegt ist.

NAT und ICMP

NAT verwendet Portnummern des Transportprotokolls - ICMP haben keine TCP/UDP Header.

> Verwende ICMP-ID anstelle von Portnummern

ICMP-TLL-Exceeded-Nachrichten (traceroute) haben keine ID.

> Daten IP-Header in der Antwort durchsuchen und zu ausgehender Nachricht mappen

NAT und IPv6

Es wird eine 1:1 Mapping von Adressen ohne Ports erzeugt, Übersetzung erfolgt zustandslos.

5: Sitzungs-, Darstellungs- und Anwendungsschicht

5.1 – Sitzungsschicht

Sitzungsschicht

Eine Verbindung der Sitzungsschicht ist nicht gleichbedeutend mit einer Verbindung auf der Transportsschicht.

- verbindungsorientiert: Verbindung bleibt über die Dauer einzelner Tramsfers bestehen
  > Verbindungsaufbau, Datentransfer, Verbindungsabbau
- verbindungslos: Daten werden nur weitergereicht, kein Zustand zwischen Kmk.partnern.

Dienste der Sitzungsschicht

- Session: Kommunikation zwischen Kmk.partnern mit definiertem Anfang & Ende.
- HTTP: zustandslos, Cookies erlauben Sessions
- TLS (Transport Layer Security): verbindungsorientiert mit Session-IDs, bietet:
  > Autentifizierung: „ist mein Gegenüber der, für den ich ihn halte“
  > Integritätsschutz: Schutz vor Datenmanipulation
  > Verschlüsselung: Vertraulichkeit, Schutz vor unberechtigtem Mitlesen

5.2 – Darstellungsschicht

Darstellungsschicht

Aufgabe der Darstellungsschicht ist das Ermöglichen einer einheitlichen Interpretation der Daten durch die verschiedenen Kommunikationspartner (Semantik auf Anwendungsschicht).

- Darstellung der Daten (Syntax), Datenstruktur zu Übertragung, Darstellung der Aktionen an Datenstrukturen, Datentransformationen
- Kodierung: Übersetzung von Zeichensätzen, Kompression, Verschlüsselung
- Strukturierte Darstellung: Plattformunabhängigkeit, Übersetung zw. Formaten, Serialisierung

Zeichensätze & Kodierung

- Textzeichen („human readable“)
- Binäre Daten (Sequenz von Bits)

Ein Zeichensatz schafft ein Mapping von Textzeichen auf binäre Daten.

- ASCII: 128 Zeichen inkl. Steuerzeichen
- ISO-8859-15: 256 Zeichen inkl. ASCII + Sonderzeichen aus europäischem Sprachraum
- Unicode: 1114112 Zeichen inkl. ISO-8859-1, regelmäßige Erweiterung

Zur Koderung unterscheided man zwischen

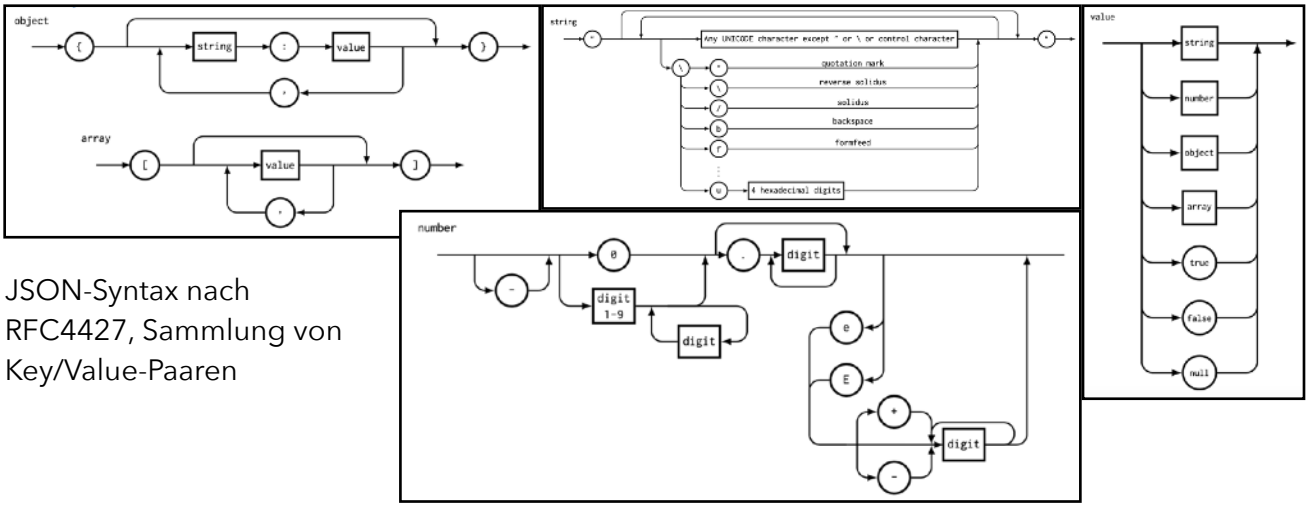
- Fixed-Length Code: alle Codewörter haben selbe Länge (ASCII, ISO-8859)
- Variable-Length Coed: unterschiedliche lange Codewörter (UTF-8)

Unicode-Bereich	Länge	Binäre UTF-8 Kodierung	kod. Bits
U+0000 — U+007F	1B	0*xxxxxxx*	7
U+0080 — U+07FF	2B	110*xxxxx* 10*xxxxxx*	11
U+0800 — U+FFFF	3 B	1110*xxxx* 10*xxxxxx* 10*xxxxxx*	16
U+10000 — U+1FFFFF	4 B	11110*xxx* 10*xxxxxx* 10*xxxxxx* 10*xxxxxx*	21

Strukturierte Darstellung

Damit Anwendungen Daten austauschen können, ist eine einheitliche Syntax nötig.

- (gepackte) structs/serialisierte Speicherbereiche: Daten werden so übertragen, wie sie vorliegen, gleiche Datenstrukturen nötig
- Ad-hoc Datenformate: wird bei Bedarf entworfen
  > problematisch: Dokumentation, Eindeutigkeit, Fehlerfreiheit & Erweiterbarkeit
- Strukturierte Serialisierungsformate: XML oder JSON



JSON-Syntax nach RFC4427, Sammlung von Key/Value-Paaren

Datenkompression

- Verlustfreie Kompression: Wiederherstellung ohne Informationsverlust möglich (z.B. ZIP, PNG)
- Verlustbehaftete Kompression: Wiederherstellung mit Informationsverlust, vairable Kompressionsrate (z.B. MP3, JPEG)

Huffman-Code

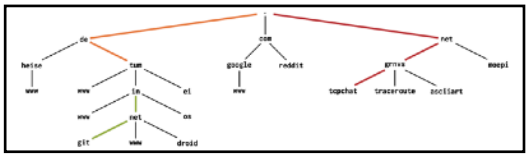
Idee: Zeichen treten unterschiedlich oft auf: kürzere Codewörter für häufige Zeichen.

$$E[I_H(z)] = \sum_{z \in \mathcal{A}} \Pr[X = z] I_H(z) \qquad \text{Huffman-Code}$$

$$E[I_H(z)] = \log_2(|\mathcal{A}|) \qquad \text{Uniformer Code}$$

5.3 – Anwendungsschicht

Domain Name System (DNS)



DNS ermöglicht das adressieren eines Ziels über einen hierarisch aufgebauten Namen statt einer IP-Adresse (Name wird zur Adresse aufgelöst).

1. Domain Namespace: hierarisch aufgebauter Namensraum mit baumartiger Struktur
   > Verteilte Datenbank, unterteilt in Zonen (zusammenhängende Teilbäume,
   > Zonen: zusammenhängende Teilbäume mit gemeinsamer Wurzel
2. Nameserver: Informationen über (kleine Ausschnitte) des Namensraums
   > (meherere) autoritative Nameserver sind für eine Zone verantwortlich, primärer ist Master
   > DNS hat Mechanismen zum Transfer von Zonen zwischen autoritativen Nameservern
3. Resolver: Programme, die durch Anfragen Informationen aus dem Namespace extrahieren

- Label: beliebiger Knoten im Namespace
- Domain Name: Sequenz von Labels
- Resource Records: Informationen, die in einer Zone gespeichert sind
  > SOA (Start of Authority) Records: Wurzel der Zone, für die der Nameserver autoritativ ist
  > NS Records: gibt FQDN (fully qualified domain name) eines Nameservers an, kann auf andere Zonen verweisen
  > A Records: assoziieren einen FQDN mit einer IPv4-Adresse
  > AAAA Records: assoziieren einen FQDN mit einer IPv6-Adresse
  > CNAME Records: Aliase, FQDN verweist auf Canoncical Name
  > MX Records: FQDN eines Mailservers für bestimmte Domain (nicht zwingend in eige. Zone)
  > TXT Records: assoziieren FQDN mit einem String (multiple use cases)
  > PTR Records: assoziieren IPv4/IPv6-Adressen mit einem FQDN (Gegenstück A/AAAA Rec.)
- Resolver: extrahieren Informationen aus dem DNS und liefern Ergebnis an anfragenden Client
  > Resolver fragen schrittweise autoritative Nameserver der jeweiligen Zonen an
  > Ergebnis wird an anfragenden Client zurückgegeben & gecached
  > Auflösung beginnt am Root, siehe Root Hints
- Router arbeitet als Resolver bzw. Leiete an Resolver des Providers weiter
  > recursive queries: DNS-Server erfragen selbst weitere Ebenen des Baums
  > iterative queries: anfragender Client fragt nacheinander weitere Ebenen des Baums an
- Reverse DNS: mittels PTR (Pointer Records) auflösen von IPv4/IPv6-Adressen zu FQDN
  > Bei IPv4 wird ein Namespace durch die vier Oktette in umgekehrter Reihenfolge erzeugt.

Uniform Resource Locator (URL)

Mit DNS kann das Ziel einer Verbindung auf Schicht 3 identifiziert werden - es fehlt allerdings das Protokoll bzw. Eine adressierung von bestimmten Ressourcen.

`<protocol>://[<username>[:<pw>]@]<fqdn>[:<port>][/path][?<query>][#<fragment>]`

- Protocol: Anwendungsprotokoll, z.B. HTTP(S), FTP, SMTP
- Username & Password: eventuell zur Authentifizierung, siehe SSH
- FQDN: Fully Qualified Domain Name zur Auflösung der Schicht 3 Adresse
- Port: Angabe einer abweichenden Portnummer (nicht Standard für das Protokoll)
- Path: Pfad relativ zu Wurzel „/, in Verzeichnisstruktur
- Query: Übergabe von Variablen
- Fragment: Adressierung von einzelnen Fragmenten/Abschnitten eines Dokuments

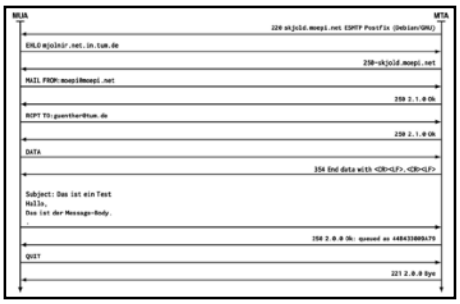
Browser vervollständigen die URLs normalerweise um Standardwerte für Webserver.

HyperText Transfer Protocol (HTTP)

HTTP wird zur Dateiübertragung zwischen Client und Server genutzt und definiert eine Reihe von legalen Anfragen. Jeder HTTP-Command überträgt maximal ein ‚Objekt'. HTTP wird auf dem well-known Port TCP 80 erwartet.

- Request: Method mit gewünschter Aktion; Pfad & Query-Parameter, um die Ressource genauer zu beschreiben; Headerfelder mit FQDN, Zeichensatz & Encoding, Quelle, User Agent
  - > GET: Anfrage zur Übertragung eines Objekts
  - > HEAD: Anfrage zur Übertragung des Headers eines Objekts
  - > PUT: Übertragung eines Objekts vom Client zum Server, eventuell überschreiben
  - > POST: Übertragung eines Objekts vom Client zum Server, eventuell anhängen an bestehendes Objekt
  - > DELETE: Löschen eines Objekts vom Server
- Response: Status-Line mit Code (+ evtl. Fehler); Response Header mit ggf. weiteren Informationen; Body mit eigentlichen Daten (nach Carriage Return Line Feed [CRFL])
  - > 200: OK
  - > 3xx: Redirection
  - > 400: Bad Request
  - > 401: Unauthorized
  - > 403: Forbidden
  - > 404: Not Found
  - > 418: I'm a teapot (Aprilscherz, keine echte Funktion)
  - > 5xx: Server Error
- HTTP-Proxy: Client bleibt hinter Proxy verborgen, Proxy kann Anfragen cachen und bei mehrfachen identischen Anfragen aus Cache antworten.
  - > Transpar. Proxies: Client weiß nichts über Proxy, TLS-Verschlüsselung Überwachung möglich.

Simple Mail Transfer Protocol (SMTP)



Das SMTP ist ein textbasiertes Protokoll zum Versenden von Emails

- Von Mail User Agent (MUA) zu Mail Transfer Agent (MTA)
- Zwischen Mail Transfer Agents (MTAs)

Die Empfängeradresse enthält den FQDN des Ziels.
Zum Empfang von E-Mails werden Post Office Protocol (POP) oder Internet Message Access Protocol (IMAP) verwendet.
MTAs akzeptieren i.d.R. nur Emails für die eigenen Domains.
Authentifizierung zwischen den MTAs ist technisch nicht umsetzbar, es kann aber opportunistisch verschlüsselt werden.

File Transfer Protocol (FTP)

Das File Transfer Protocol ist ein Protokoll zum Transfer von Daten. Im Unterschied zu HTTP nutzt FTP zwei getrennte TCP-Verbindungen:

- Kontrollkanal: Übermittlung von Befehlen & Statuscodes zwischen Client & Server (stateful)
- Datenkanal: Übertragung der eigentlichen Daten

FTP erfordert eine Authentifizierung mittels Benutzername und Passwort und arbeitet mit dem Kontrollkanal auf ‚well-known port' 21.

- Active mode: Client teilt Server eine zufällige Portnummer mit, über die er Daten empfangen möchte (PORT Kommando vs. PASV Kommando)
- Passive mode: Client erhält von Server IP & Port und baut TCP-Verbindung als Datenkanal auf.

FTP mit NAT

FTP active mode und NAT können zu Problemen führen, da sich der Server hier selbst zum Client verbindet, der aber (unwissend) hinter einem NAT versteckt liegt. Die Verbindung kann nicht aufgebaut werden.

Lösungen:

- NAT Implementierung so erweitern, dass sie FTP unterstützt
  > NAT müsste die L7-PDU prüfen und bei Vorhandensein von PORT die private IP-Adresse durch eine öffentlich gültige ersetzen und einen Eintrag in der NAT Tabelle für den entsprechenden Port erstellen
- FTP passive mode
  > Da Server keine Verbindung zum Client aufbaut, gibt es keine NAT Probleme

–