Rechnernetze Zusammenfassung

sgeisler
sgeisler@wh2.tu-dresden.de

chlewe
ch_lewe@wh2.tu-dresden.de

August 2017
Version: 02bae74

1 Allgemeines

$$K_n = (V, E) \quad \Rightarrow \quad |E| = \frac{n \cdot (n-1)}{2}$$

Dienst Schnittstelle zwischen OSI-Schichten, definiert erbrachte Funktionalität einer Schicht.

Protokoll Regeln zur Ablaufsteuerung von Kommunikation in einer Schicht.

1.1 OSI-Schichten

7. Anwendungsschicht (RPC, FTP, E-Mail)
6. Darstellungsschicht (ASCII, MP3, RSA)
5. Sitzungsschicht (Transaktionshandling)

} auch insgesamt als Anwendungsschicht (Layer 5) bezeichnet

4. Transportschicht (TCP, UDP)
3. Vermittlungsschicht (IP)
2. Sicherungsschicht (Ethernet)
1. Bitübertragungsschicht (Umsetzung in elektrische Signale)

1.2 Sequenzdiagramme

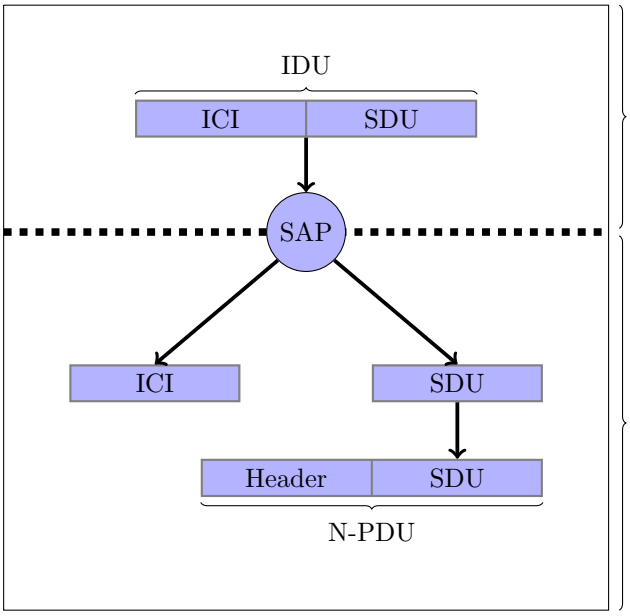
Req: Request, von Client gesendet

Ind: Indication, von Server empfagener Request

Rsp: Response, von Server gesendet

Cnf: Confirmation, von Client empfangene Response

1.3 OSI-Schichtenarchitektur



IDU:
Interface Data Unit

ICI:
Interface Control Information

SDU:
Service Data Unit

PDU:
Protocol Data Unit

SAP:
Service Access Point

2 Bitübertragungsschicht

2.1 IKT-Wiederholung

SNR: Signal-Rausch-Abstand (*S-R-Verhältnis*)
SR: Symbolrate (*Baudrate, Schrittgeschw.*)
S: Symbolstufen (*Signalstufen*)
IG: Informationsgehalt
B: Bandbreite
b**:** Bitrate (*Datenrate, Übertragungsrate*)
f_g**:** Grenzfrequenz
f_a**:** Abtastfrequenz

$$\mathrm{SNR_{dB}} = 10 \cdot \log_{10}(\mathrm{SNR}) \iff \mathrm{SNR} = 10^{\frac{\mathrm{SNR_{dB}}}{10}}$$
$$\mathrm{IG} = \log_2(\mathrm{S})$$
$$\mathrm{SR} = \frac{b}{\log_2(S)}$$
$$b = f_a \cdot \log_2(S)$$

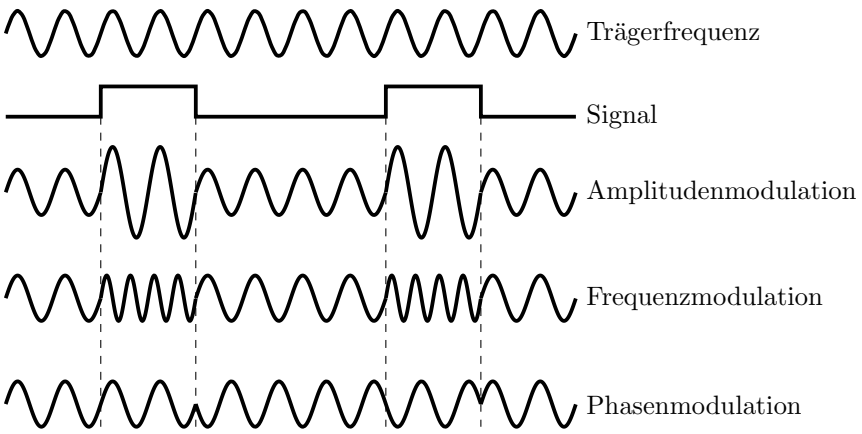
Abtast-Theorem: $f_a > 2 \cdot f_g$

Nyquist-Theorem („*Nyquist 1*“): $b < 2B \cdot \log_2(\mathrm{S})$

Erweiterung durch Shannon („*Nyquist 2*“): $b < B \cdot \log_2(1 + \mathrm{SNR})$

$$b < \min\left(2B \cdot \log_2(S),\ \ B \cdot \log_2(1+SNR)\right) \qquad B > \max\left(\frac{b}{2 \cdot \log_2(S)},\ \ \frac{b}{\log_2(1+\mathrm{SNR})}\right)$$

2.2 Modulationsarten



2.2.1 Amplitudenmodulation

- Kodierung des Signals durch Variation der Amplitude
- Einfache Kodierung/Dekodierung
- Abstandsabhängige Amplitude ⇒ störanfällig

2.2.2 Frequenzmodulation

- Kodierung des Signals durch Variation der Frequenz
- Kompliziertere Kodierung/Dekodierung
- Weniger abstandsabhängig, dafür Dopplereffekt

2.2.3 Phasenmodulation

- Kodierung des Signals durch Phasensprung bei 0, kein Phasensprung bei 1
- Komplizierte Kodierung/Dekodierung
- Weniger Störanfällig als FM und AM

2.3 Leitungskodierung

2.3.1 Non-Return-to-Zero

- Kodiert jede 1 mit hohem Pegel, jede 0 mit niedrigem Pegel
- Netto-Datenrate = Bitrate (2 bit pro Periode)

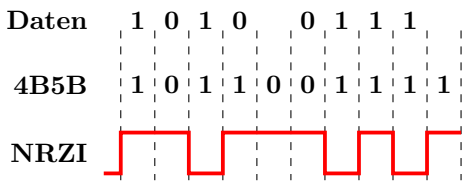
2.3.2 Non-Return-to-Zero Inverted / 4B5B

- NRZI kodiert:

  0: keine Pegeländerung

  1: Pegeländerung

- 4B/5B kodiert je 4 bit (Nibble) in 5 bit, nie mehr als 3 Nullen in Folge



0	1	2	3	4	5	6	7
0000	0001	0010	0011	0100	0101	0110	0111
11110	01001	10100	10101	01010	01011	01110	01111

8	9	A	B	C	D	E	F
1000	1001	1010	1011	1100	1101	1110	1111
10010	10011	10110	10111	11010	11011	11100	11101

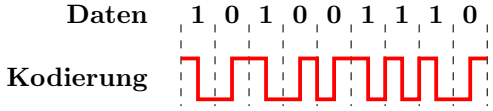
Tabelle 1: 4B5B Code

2.3.3 Einfache Manchesterkodierung

Nach G.E. Thomas (wie in Übung):

0: steigende Flanke

1: fallende Flanke



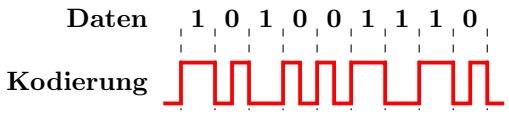
Der IEEE 802.3 Standard beschreibt genau die invertierte Form der hier beschriebenen Manchesterkodierung.

2.3.4 Differentielle Manchesterkodierung

Am Anfang einer jeden Zeiteinheit findet eine Pegeländerung statt. Der Datenstrom wird wie folgt kodiert:

0: eine weitere Pegeländerung findet in der Mitte der Zeiteinheit statt

1: während der Zeiteinheit findet keine weitere Pegeländerung statt



Durch Leitungskodierungen wie z.B. die Manchesterkodierung können Fehler erkannt, Gleichstromanteile vermieden und der Takt rückgewonnen werden.

2.4 Multiplexing

Synchrones Zeitmultiplexing: Jeder Teilnehmer besitzt fest zugeordnete Zeitintervalle in denen er Nachrichten senden kann

Asynchrones Zeitmultiplexing: Die Nachrichten werden getaggt

Frequenzmultiplexing: Die Teilnehmer senden mit verschiedenen Trägerfrequenzen

3 Sicherungsschicht

Die Sicherungsschicht ist für *Medium Access Control* (MAC; Medienzugriff) und *Logical Link Control* (LLC; Flusskontrolle, Rahmenbildung und Fehlerüberprüfung).

3.1 Fehlererkennung und -korrektur

Gerades Paritätsbit: $\sum_i d_i \mod 2 = p$ mit $d_1, \ldots, d_n$ Datenbits und p Paritätsbit

Hamming-Distanz d: Zwichen zwei Bitfolgen a, b: Zahl der Bitflips um aus a b zu machen. Einer Menge an Bitfolgen: minimum aller paarweisen Hamming-Distanzen.

Erkennbare Fehler: $f_e = d_{min} - 1$

Korrigierbare Fehler: $f_k = \lfloor \frac{d_{min}-1}{2} \rfloor$

Kreuzsicherung Je eine Spalte und eine Zeile einer Matrix mit Paritätsbit ermöglichen Fehlererkennung in einer Matrix.

Cycling Redundancy Check (CRC) r = grad g(x), r*0 Bits an Datenpolynom P_D anhängen, $P_D x^r : g(x) =>$ Rest an P_D anhängen = Sendesequenz P_E
Kontrolle: P_E : g(x) mit Rest = 0 => korrekte Übertragung

3.2 Medienzugriff

(*Pure*) ALOHA Man sendet sofort wenn man kann.

Slotted ALOHA Man darf nur zu Beginn eines Zeitslots senden. Ansonsten muss man warten.

CSMA (*Carrier-sense multiple access*)

p-persistent CSMA

1. Warte auf den nächsten Zeitslot
2. Lausche, ob andere gerade senden
3. Falls nicht, dann sende mit der Wahrscheinlichkeit p
4. Gehe zu 1.

non-persistent CSMA

1. Lausche, ob andere gerade senden
2. Falls nicht, dann sende und gehe zu 1.
3. Warte eine zufällige Dauer und gehe zu 1.

CSMA/CD (*Carrier-sense multiple access with collision detection*)

Wie *non-persistent CSMA*, nur dass beim Senden (2.) eine Kollisionserkennung geschieht:

(i) Lausche, ob andere auch senden

(ii) Falls ja, dann sende ab sofort das *Jam-Signal* anstatt dem Frame bis die Mindestframelänge erreicht ist; danach gehe zu 3.

(iii) Gehe zu (i)

Das Jam-Signal sorgt dafür, dass der CRC des Frames bei allen Empfängern fehlschlägt.

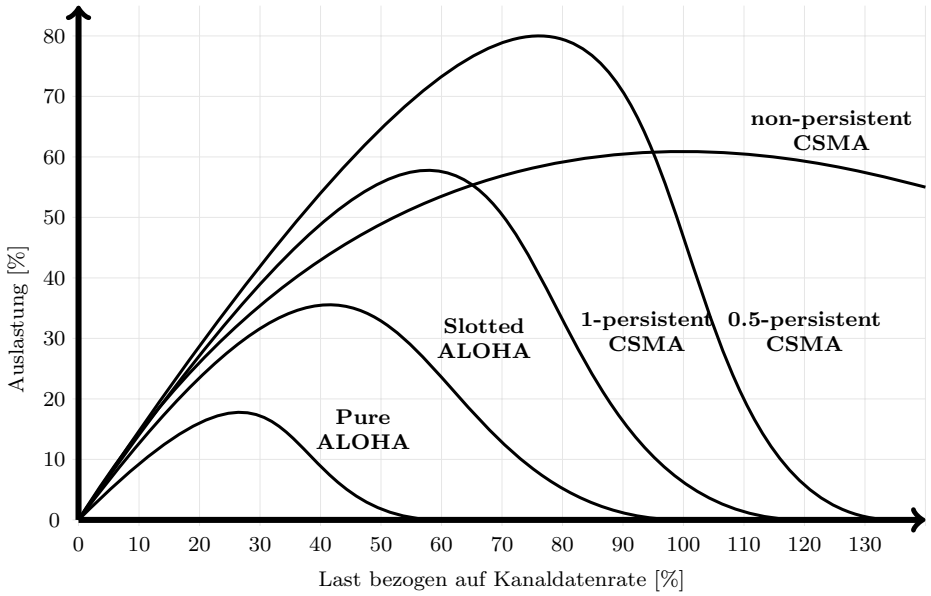


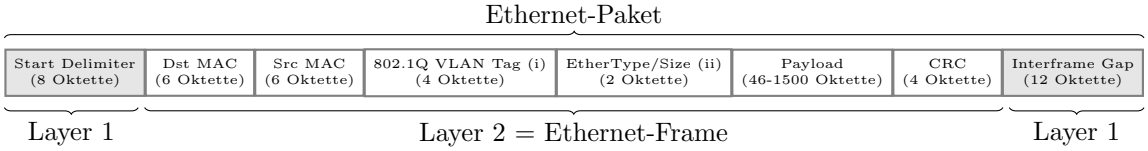
Abbildung 1: Performacevergleich verschiedener MAC-Protokolle

3.3 Ethernet

Standard	Geschwindigkeit	Mindestframelänge	SDL	IFG
Ethernet	$10\frac{\text{Mbit}}{\text{s}}$	64 Oktette	8 Oktette	47 Bit
Fast Ethernet	$100\frac{\text{Mbit}}{\text{s}}$	64 Oktette	8 Oktette	12 Oktette
Gigabit Ethernet	$1\frac{\text{Gbit}}{\text{s}}$	512 Oktette	8 Oktette	8 Oktette
10-Gigabit Ethernet	$10\frac{\text{Gbit}}{\text{s}}$	512 Oktette, kein CSMA/CD	8 Oktette	5 Oktette

SDL: Start Delimiter; IFG: Interframe Gap
Ethernet nutzt CSMA/CD (außer 10-Gigabit Ethernet)

Ethernet-Paket (Layer 1) mit Ethernet-Frame (Layer 2)



(i) VLAN-Tag ist optional

(ii) EtherType/Size ≤ 1500: Länge des Payloads in Bytes
EtherType/Size ≥ 1536: Typ des Frames [1]

[1] Die Lücke zwischen 1500 und 1536 ist kein Tippfehler und hat technische sowie historische Gründe.

Kabel-Standards (10-Mbit-Ethernet)

- Bis zu 500m physische Kabellänge
- Bis zu 4 Repeater (bis zu 2500m logische Kabellänge)

Maximale Framelängen

- Bis zu 1518 Bytes lange Frames (14 Bytes Header, 1500 Bytes Payload, 4 Bytes CRC)
- Jumborahmen: z.B. bis zu 8192 Bytes lange Frames

Transparent Bridging Layer-2-Switches speichern Switching-Tabellen, in denen MACs auf Ports abgebildet werden. Kreise im Netz führen zum Endlos-Switching.

1. Frame kommt an Port i an
2. Falls *Src-MAC* noch nicht in Tabelle ist: Bilde sie auf Port i ab
3. Falls *Dst-MAC* in Tabelle ist: Sende Frame über den eingetragenen Port j; gehe zu 1.
4. Sende Frame über alle Ports außer Port i (*Flooding*); gehe zu 1.

Ethernet Flow Control Wenn der sendende Computer schneller überträgt, als der Empfänger aufnehmen kann, werden Pause-Frames übertragen.

Store-and-forward Switching Die Switch puffert den gesamten Frame, überprüft die Prüfsumme (CRC) und sendet bei Erfolg den Frame weiter. Ansonsten wird er verworfen.

Cut-through Switching Die Switch puffert den Frame bis die Dst-MAC gelesen wurde und sendet ihn anschließend sofort weiter. Fehler werden erst beim Empfänger erkannt, aber die Übertragung ist schneller.

Spanning Tree Protocol Kreise im Netz werden ausfindig gemacht und durch Portabschaltungen aufgelöst.

3.4 Funktechnologien

	WiMAX	WLAN (IEEE 802.11)
Gemeinsamkeiten		OFDM MIMO
Reichweite	weit	nah
Anzahl Nutzer	hoch	gering
Sendeleistung	hoch	gering
Medienzugriffsverfahren	sehr effizient	weniger effizient

OFDM (*Orthogonal frequency-division multiplexing*) Digitaler Datenstrom wird auf mehreren Trägerfrequenzen übertragen.

MIMO (*Multiple-input and multiple-output*) Nutzung von n Sende- und n Empfangsantennen pro Gerät für n-fache Kanalkapazität. Hardwarekosten steigen durch komplizierte Kanalmatrixberechnung.

3.5 Resilient Packet Ring (RPR)

Ein RPR besteht aus Teilnehmern, die über zwei gegenläufige, gerichtete Ringe verbunden sind. Das Netzwerk ist auch bei Ausfall einiger Verbindungen weiterhin funktionsfähig.

Eine Übertragung blockiert den gesamten Pfad. Ist dieser noch nicht frei, muss gewartet werden bis er frei wird. Es sind mehrere Übertragungen gleichzeitig möglich (*spatial reuse*).

3.6 Carrier Ethernet

Um zwischen verschiedenen Standorten (z.B. größere Firmen) mehrere getrennte Layer 2 Netzwerke zu tunneln, muss auch das Tunneln mehrerer VLANs möglich sein. Dies kann per Q-in-Q (äußere und innere VLAN-ID) oder MAC-in-MAC (ein Ethernetframe in ein anderes Ethernetpaket einpacken) geschehen.

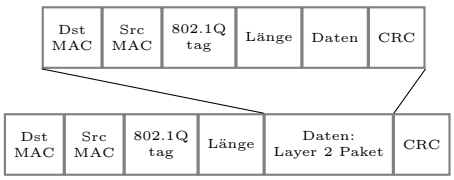


Abbildung 2: 802.1ah MAC-in-MAC

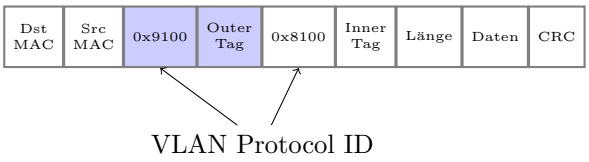


Abbildung 3: QIEEE 802.1ad Q-in-Q

3.7 Multiprotocol Label Switching (MPLS)

Ingress-Router labeln eingehende Pakete anhand ihrer Zieladresse und ihrer Herkunft

Switche routen Pakete anhand ihrer Label und labeln sie um.

Egress-Router entfernen Label von Paketen, sendet sie anhand dieser in externes Netz weiter.

Label bestimmen pro Punkt-zu-Punkt-Verbindung eindeutig die Route.

3.8 SONET/SDH

SONET: Synchronous Optical Network

SDH: Synchronous Digital Hierarchy

Beide Technologien werden zur Bündelung mehrerer Datenströme (z.B. Telefonverbindungen) eingesetzt. Dabei besteht jeder SONET-Basisrahmen aus je einenm Bit per Datenstrom und Overhead (9 Zeilen, 87 Spalten Nutzdaten, 3 Spalten Overhead ⇒ 783 parallele Datenströme).

SONET		SDH	Datenrate ($\frac{\text{Mbit}}{\text{s}}$)		
elektrisch	optisch	optisch	gesamt	SPE	User
STS-1	OC-1	STM-0	51,84	50,112	49,536
STS-3	OC-3	STM-1	155,52	150,336	148,608
-	-	STM-2	311,04	300,672	297,216
STS-9	OC-9	STM-3	466,56	451,008	445,824
STS-12	OC-12	STM-4	622,08	601,344	594,432
STS-18	OC-18	STM-6	933,12	902,016	891,648
STS-24	OC-24	STM-8	1244,16	1202,688	1188,864
STS-36	OC-36	STM-12	1866,24	1804,032	1783,296
STS-48	OC-48	STM-16	2488,32	2405,376	2377,728
STS-96	OC-96	STM-32	4976,64	4810,752	4755,456
STS-192	OC-192	STM-64	9953,28	9621,504	9510,912
STS-256	OC-256	-	13271,04	12828,672	12681,216
STS-384	OC-384	STM-128	19906,56	19243,008	19021,824
STS-768	OC-768	STM-256	39813,12	38486,016	38043,648
STS-1536	OC-1536	STM-512	79626,24	76972,032	76087,296
STS-3072	OC-3072	STM-1024	159252,48	153944,064	152174,592

STS: Synchronous Transport Signal

STM: Synchronous Transfer Mode

OC: Optical Carrier

SPE: Synchronous Payload Envelope

4 Vermittlungsschicht

4.1 Dijkstra

Dijkstra-Algorithmus Algorithmus zur Ermittlung der kürzesten Pfade von einem Startknoten aus in (nicht-negativ) gewichtetem Graphen

Gegeben Graph mit gewichteten Kanten, Startknoten S

1. Knotenvorrat := {S}
2. aktiver Knoten := S
3. aktive Routen := {(S-S; 0)}
4. festgeschriebene Routen := ∅
5. Solange Knotenvorrat nicht leer:
   - Entferne kürzeste Route R aus aktiven Routen und füge R festgeschriebenen Routen hinzu
   - Mache Ziel von R zum aktiven Knoten A und entferne A aus Knotenvorrat
   - Für jeden Knoten K, zu dem keine Route festgeschrieben ist, mit Kante zu A:
     - Füge K zum Knotenvorrat hinzu
     - Füge (R-K; gewicht(R)+gewicht(A-K)) den aktiven Routen hinzu
   - Falls mehrere Pfade mit gleichem Ziel in Routen: entferne längere Pfade

Knotenausfall Setze alle Kantengewichte am ausgefallenen Knoten auf ∞

Ergebnis: Baum mit Wurzel am Startknoten **(Sink Tree)**

4.2 IP

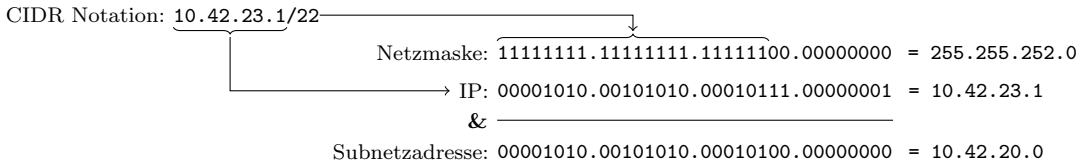
IPv4: 32 bit

Private Adressbereiche:

- `10.0.0.0/8` 1 Class-A Netz
- `172.16.0.0/12` 16 Class-B Netze
- `192.168.0.0/16` 256 Class-C Netze

IPv6: 128 bit

4.2.1 Classless Inter-Domain Routing (CIDR)



4.2.2 IPv4-Paket

Version (4 bit)	IHL (4 bit)	ToS (8 bit)	TL (16 bit)	ID (16 bit)	Flags (3 bit)	FO (13 bit)	TTL (8 bit)	Protocol (8 bit)	HC (16 bit)	Src (32 bit)	Dst (32 bit)	Options/ Padding	Payload

Version: **IPv4**/IPv6

IHL: Header Länge

ToS: Type of Service (Bandbreiten-, Zuverlässigkeitsanforderungen, etc.)

TL: Total Length

ID: Eindeutige ID zu zusammensetzen fragmentierter Pakete

Flags:

Bit 0: reserviert

Bit 1: Don't Fragment

Bit 2: More Fragments

FO: Fragment Offset (bezogen auf 8-Oktett-Blöcke)

TTL: Time to Live (heute Anzahl der übrigen erlaubten Hops. Falls 0, so wird das Paket gedroppt.)

Protocol: Eingeschlossenes Protokoll

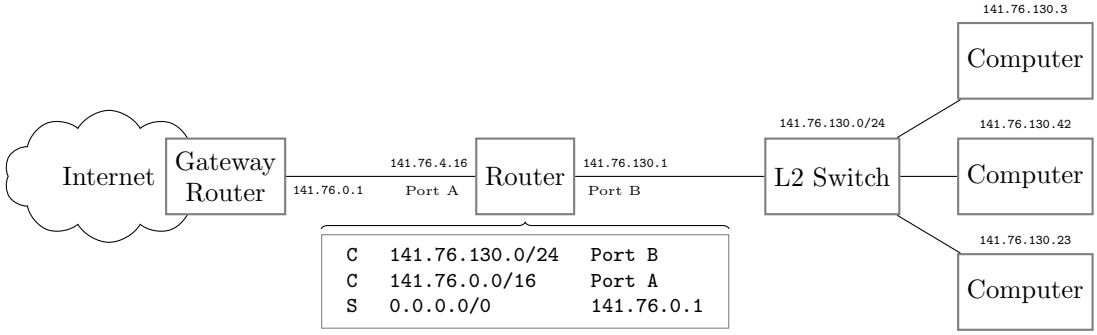
HC: Header Checksum

Src: Herkunftsadresse

Dst: Zieladresse

4.3 Routing

Longest Prefix Match Wenn die Ziel-IP mit mehreren Routing-Einträgen übereinstimmt, so wird der Eintrag mit dem längsten übereinstimmenden Präfix ausgewählt. Das kann erreicht werden, indem man den Eintrag mit der größten Netzmaske wählt.



C*onnected*: Das Subnetz liegt direkt am angegebenen Port an

S*tatic*: Das Subnetz ist über das angegebene Gateway erreichbar

D*ynamic*: Das Gateway wird dynamisch ermittelt (z.B. OSPF)

5 Transportschicht

Port Mechanismus für eine Anwendung-zu-Anwendung-Übertragung.

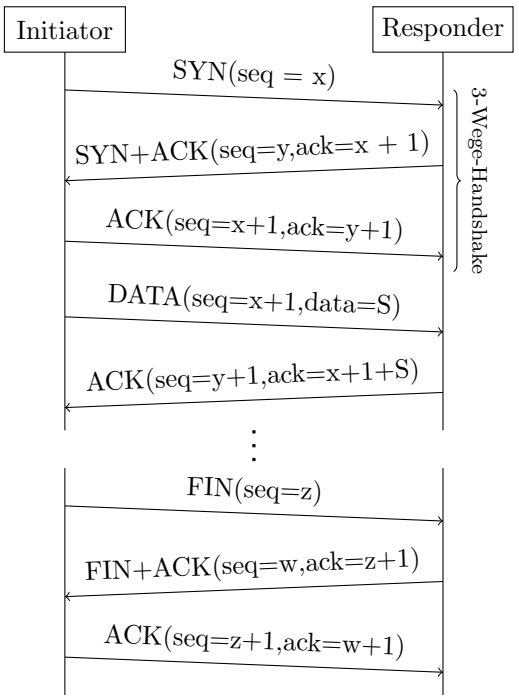
UDP (*User Datagram Protocol*) verbindungsloses Protokoll nach dem *Best Effort*-Prinzip. Implementierung von *Ports*.

Verbindung Kanal, in dem zusammengehörige Pakete verschickt werden. Der Erhalt und die richtige Reihenfolge werden garantiert.

TCP (*Transmission Control Protocol*) verbindungsbehaftetes Protokoll mit Ende-zu-Ende-Sicherung. Implementierung von *Verbindungen*.

Slow Start Algorithmus zur Flusskontrolle bei TCP.

1. Fenstergröße := Segmentgröße
2. Bei Erhalt eines ACKs:
   - Falls Fenstergröße < Schwellenwert: Fenstergröße *= 2
   - Falls Fenstergröße < Empfangsfenser: Fenstergröße += Segmentgröße
   - Gehe zu 2.
3. Bei einem Timeout:
   - Schwellenwert := Fenstergröße / 2
   - Gehe zu 1.



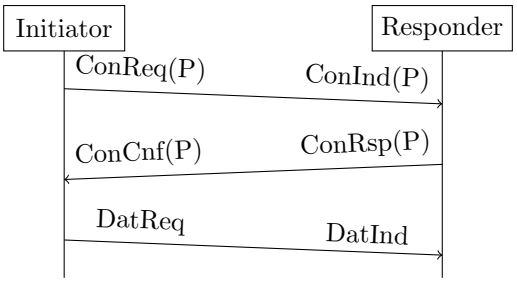


Abbildung 4: Quality-of-Service mit dem Parameter P

Schiebefensterprotokoll Effizienzsteigerung, indem man F viele Pakete hintereinander versendet, ohne auf die Bestätigung des vorherigen Pakets warten zu müssen. (*Stop-and-Wait ist wie ein Schiebefensterprotokoll mit* $F = 1$)

Bei Erhalt eines Pakets mit falscher Sequenznummer (Empfangs- = Sendereihenfolge!):
Go-Back-N:

- Empfänger verwirft das Paket
- Nach einem Timeout muss der Sender alle unbestätigten Pakete erneut senden

Selective Repeat:

- Empfänger speichert das Paket zwischen und sendet ein ACK mit der Sequenznummer des ersten fehlenden Pakets an den Sender
- Nach einem Timeout wird das Paket mit dieser Sequenznummer erneut gesendet

6 Performance

Fairness Alle Teilnehmer haben die gleiche Chance, heranzukommen.

Max-Min-Fairness Jeder bekommt gleich viel[2] aus dem Topf. Hat jemand mehr als er braucht, kommt der Überschuss wieder in den Topf. Dieser Algorithmus wird wiederholt bis alle gesättigt sind oder der Topf leer ist.

Lastberechnung $\text{Last}_{\text{neu}} = a \cdot \text{Last}_{\text{alt}} + (1 - a) \cdot \text{Last}_{\text{gemessen}}$ $a \ldots$ Anpassungsfaktor

Bandwidth Delay Product $\text{BDP} = b \cdot \frac{d}{v}$ $b \ldots$ Bitrate, $d \ldots$ Kabellänge, $v \ldots$ Übertragungsgeschw.

Worst-Case-Paketeffizienz mit $F = 64$

- Fast Ethernet: `8 Byte SDL | 64 Byte Frame | 12 Byte IFG`
- Gigabit: `8 Byte SDL | 64 Byte Frame | 448 Byte Padding | 8 Byte IFG`
- Gigabit mit Bursting: `Padding` und `IFG` tauschen, um Bursting anzuzeigen; dann beliebig viele [3] `8 Byte SDL | 64 Byte Frame | 8 Byte IFG` Blöcke anfügen.
- $\eta_{\text{frame}} = \frac{\text{Framebytes}}{\text{Gesamtbytes}}$

[2] Oder gewichtet: Jeder bekommt $\frac{\text{eigenes Gewicht}}{\text{Summe aller Gewichte}}$ viel
[3] Maximale Framelänge darf nicht überschritten werden: 1518 Bytes (normal) oder 8192 Bytes (Jumbo)

7 Internetdienste

URL (*Uniform Resource Locator*) $\underbrace{\text{http}}_{\text{Protokoll}}\text{://}\underbrace{\text{www}}_{\text{Host}}\text{.}\underbrace{\text{inf}}_{\text{Subdomain}}\text{.}\underbrace{\text{tu-dresden}}_{\text{Domain}}\text{.}\underbrace{\text{de}}_{\text{TLD}}\underbrace{\text{/test/index.html}}_{\text{Resource}}$
(Im weiteren wird die Kombination einer Teilmenge aus Host, Subdomain, Domain und TLD als **Host** *bezeichnet.)*

DNS (*Domain Name System*) System zum Auflösen von **Host** s zu IPs.

- Client sendet Ziel-**Host** an den lokalen DNS-Server
- Lokaler DNS-Server befragt den Root-DNS-Server nach dem **Host**
- **Rekursiver Ansatz:** `lookup(rootServer, host)`
  `lookup(server, host)`:
  - Falls **Host** als IP vorliegt, gebe sie aus
  - Sonst gebe `lookup(server', host)` aus,
    wobei `server'` der zuständige DNS-Server ist
- **Iterativer Ansatz:**
  - Root-Server sendet dem lokalen Server die IP des nächsten zuständigen DNS-Servers
  - Lokaler Server muss diesen dann befragen usw.
  - Der letzte zuständige Server sendet dann die IP des **Host** s
- Lokaler DNS-Server sendet die Ziel-IP an den Client
- *Jeder DNS-Server kann die Antworten übergeordneter DNS-Server cachen*

NAME	TTL	CLASS	TYPE	VALUE
jupiter	86400	IN	A	117.186.1.1
jupiter	86400	IN	AAAA	2001:db8:85a3:8d3::1
zeus	86400	IN	CNAME	jupiter
rn-edu.de.	86400	IN	NS	jupiter
_sip._tcp.rn-edu.de.	86400	IN	SRV	0 0 5060 jupiter

ARP (*Address Resolution Protocol*) Protokoll zum Abbilden von IP-Adressen auf MAC-Adressen: Man sendet einen ARP-Request mit der IP als Broadcast über Ethernet. Der Netzwerkteilnehmer mit der entsprechenden IP antwortet mit seiner MAC.

Base64 Kodierung zur ASCII-Übertragung von Binärdaten (*z.B. bei E-Mail*). 3 Bytes ($3 \cdot 8bit$) werden jeweils zu 4 Base64-Zeichen ($4 \cdot 6bit$) umgewandelt. Unvollständige 3-Byte-Blöcke werden durch Padding trotzdem zu einem 4er-Base64-Block.

SMTP (*Simple Mail Transfer Protocol*) Protokoll zum Senden von E-Mails an einen Server.

POP3 (*Post Office Protocol 3*) Protokoll zum Herunterladen von E-Mails von einem Server (*Server wird entlastet*).

IMAP (*Internet Message Access Protocol*) Protokoll zum selektiven Abrufen von E-Mails von Servern (*E-Mail bleibt auf dem Server*).

SNMP (*Simple Network Management Protocol*) Protokoll zum einheitlichen Konfigurieren und Verwalten von Netzwerkgeräten.

8 Multimediakommunikation

Lauflängenkodierung Wenn sich ein Byte mindestens S mal wiederholt ($S \ldots$ Schwellenwert), wird es wie folgt kodiert: `Byte | Marke | Anzahl - S`. Die anderen Bytes bleiben unverändert.

Huffman-Kodierung

1. Zu kodierende Zeichen (*Knoten*) ihrer Wahrscheinlichkeit nach aufsteigend ordnen
2. Die beiden Knoten mit den niedrigsten Wahrscheinlichkeiten zu einem neuen Knoten verbinden und die Wahrscheinlichkeiten addieren
3. Falls noch keine Wurzel entstanden ist: Gehe zu 2.
4. Beschrifte alle Rechtsabzweigungen mit einer Zahl (0 oder 1) und alle Linksabzweigungen mit der jeweils anderen Zahl
5. Kodierung eines Zeichens: Beschriftungen auf dem Pfad von de Wurzel zum Zeichen

SIP (*Session Initiation Protocol*) Protokoll zum Aufbauen von VoIP-Sessions, meist über SIP-Proxys. Es verwendet das SDP (*Session Description Protocol*), was eigentlich nur eine Beschreibungssprache und *kein* Protokoll ist.

RTP (*Real-time Transport Protocol*) Protokoll zur effizienten Übertragung von Audio- und Videodaten. Flusskontrolle und Quality-of-Service werden vom RTCP (*Real-time Transport Control Protocol*) übernommen.

9 Verteilte Systeme

RPC (*Remote Procedure Call*) Aufruf einer Funktion auf einem entfernten Rechner.

Stub Schnittstelle zwischen den Anwendungen und dem Transportsystem. Sie (de)kodiert die Aufrufe und Ergebnisse.

RPC-Bindevorgang Der RPC-Server wird über Direktadressierung, Broadcast oder einen Verzeichnisdienst ausfindig gemacht. Nach dem Binding können RPCs gesendet werden.

Fehlersemantik	Bei Fehler	Filterung von Duplikaten	Bei Client- oder Serverfehler
maybe	kein Neusenden	keine	kein Neusenden
at-least-once	Neusenden	keine	kein Neusenden
at-most-once	Neusenden	ja	kein Neusenden
exactly-once	Neusenden	ja	Neusenden

ACID-Prinzip

- **A**tomicity: Transaktionen werden ganz oder gar nicht ausgeführt.
- **C**onsistency: Transaktionen führen von konsistenten Zuständen zu konsistenten Zuständen.
- **I**solation: Parallele Transaktionen beeinflussen sich nicht gegenseitig.
- **D**urability: Erfolgreiche Transaktionen werden persistiert.

2PC (*2-Phase Commit Protocol*) Protokoll zur Umsetzung von *ACID* in verteilten Systemen.

1. Voting-Phase:
   - Koordinator sendet `C_BEGIN` an alle Teilnehmer
   - Jeder Teilnehmer führt die Transaktion aus (und speichert sich Undo-Informationen)
   - Koordinator sendet `C_PREPARE` an alle Teilnehmer
   - Jeder Teilnehmer sendet entweder `C_READY` wenn sie der Transaktion zustimmen oder `C_REFUSE` wenn nicht
2. Commit-Phase:
   - Koordinator sendet `C_COMMIT` genau dann wenn alle Teilnehmer zugestimmt haben; sonst sendet er `C_ROLLBACK` (*auch bei einem Timeout*)
   - Jeder Teilnehmer persistieren den Zustand bzw. setzen die Änderungen zurück
   - Jeder Teilnehmer sendet `ACK` zum Koordinator

SOAP (*Simple Object Access Protocol*) Protokoll zum Arbeiten mit Ressourcen auf Webservern. Es nutzt HTTP mit einem XML-Body.

REST (*Representational State Transfer*) Paradigma um mit Ressourcen auf Webservern zu arbeiten mithilfe von zustandslosen Operationen.

10 Mobile Computing

Zellulares Netzwerk Das Frequenzband wird in unterschiedliche Teilfrequenzen aufgeteilt. Nachbarzellen nutzen andere Teilfrequenzen zur Vermeidung von Interferenzen. Das entstehende Muster (*Cluster*) wird wiederholt.

- Je größer die Zelle, desto weniger Kanäle pro Fläche
- Je größer die Zelle, desto größer die benötigte Sendeleistung und Interferenzzone
- $R \ldots$ Radius der Zelle, Interferenzradius $= 5R$
- $D = R \cdot \sqrt{3k}$ $\quad D \ldots$ Abstand Basisstationen gleicher Zellen, $k \ldots$ Anzahl der Zelltypen

DHCP (*Dynamic Host Configuration Protocol*) Protokoll zum automatischen Verteilen von IP-Adressen und Parametern wie DNS-Servern und Gateways.

1. Client sendet `DHCPDISCOVER` als Broadcast und erhält `DHCPOFFER`
2. Client sendet `DHCPREQUEST` an einen der DHCP-Server und erhält `DHCPACK` mit den Zeiten T (*Lease time*) und T1 (*50% T*) T2 (*87.5% T*)
3. Wenn T1 abläuft, sendet der Client `DHCPREQUEST` an den momentanen DHCP-Server, um das Lease der momentanen IP zu verlängern
4. Wenn T2 abläuft, sendet der Client `DHCPREQUEST` als Broadcast, um das Lease der momentanen IP zu verlängern
5. Wenn Client `DHCPACK` erhält (3. oder 4.), werden T, T1 und T2 aktualisiert (gehe zu 3.)
6. Wenn T abläuft oder der Client `DHCPNACK` erhält (3. oder 4.), verliert er die IP und bricht den Netzwerkverkehr ab (gehe 1.)

Mobile IP Protokoll um mobile Internetnutzer über eine statische IP adressieren zu können.

- Jeder *Mobile Host* (MH) ist bei einem *Home Agent* (HA) angemeldet
- Der HA bildet die statische IP (*Home Address*) auf die dynamische *Care-of Address* (COA) des MH ab

Foreign-Agent COA:

- MH meldet sich beim *Foreign Agent* (FA) mit seiner IP und seinem HA an
- FA meldet sich beim HA an mit dem MH und der IP des FA (COA)
- HA kann nun Pakete an die *Home Address* in Pakete vom HA zum FA verpacken und dorthin schicken; der FA packt sie aus und sendet sie zum MH

Co-located COA:

- MH meldet sich beim HA mit seiner IP (COA) an (MH braucht eine eindeutige IP)
- HA kann nun Pakete an die *Home Address* in Pakete vom HA zum MH (COA) verpacken; der MH packt sie selber aus