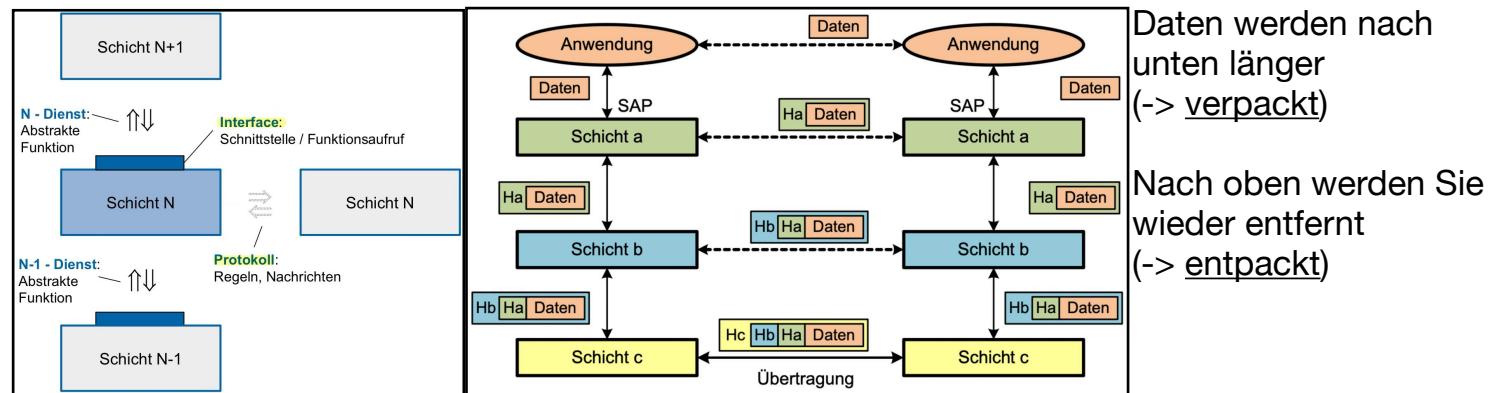


# Kommunikationstechnik - ZHAW FS 2024

## OSI-Referenzmodell - Überblick

### Schichtenmodell:



### Klassifizierung von Diensten:

Verbindungsorientiert	Verbindungslos	Zuverlässig	Unzuverlässig
Erfordert eine feste Verbindung zwischen Sender und Empfänger, bevor Daten übertragen werden (z.B. Anruf, TCP).	Überträgt Daten ohne feste Verbindung, wobei jedes Datenpaket unabhängig gesendet wird (z.B. E-Mail, UDP).	Garantiert die korrekte und vollständige Übertragung von Daten mit Bestätigung und Fehlerkorrektur (z.B. TCP).	Keine Garantie für vollständige zeitgerechte Übertragung, keine Retransmission / Flow Control (z.B. UDP).

### Alle Schichten des OSI-Modells (OSI = Open Systems Interconnection):

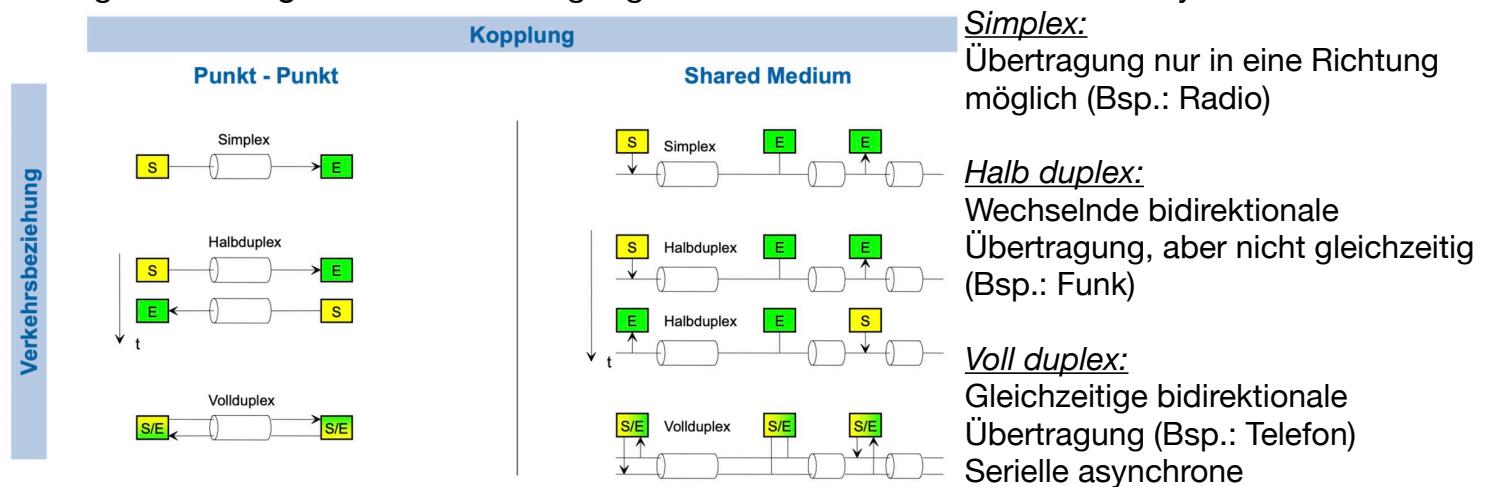
Application Layer Verarbeitungsschicht	7	Bietet Netzwerkdienste direkt für Anwendungen, wie E-Mail, Dateitransfer und Webdienste, und ermöglicht die Interaktion mit der Netzwerkschicht. (-> Anwendungsschichten)
Presentation Layer Darstellungsschicht	6	Übersetzt Datenformate und -syntax zwischen verschiedenen Systemen, bietet Datenkompression und Verschlüsselung. (-> Anwendungsschichten)
Session Layer Kommunikationsschicht	5	Veraltet Sitzungen oder Verbindungen zwischen Anwendungen, einschließlich Auf- und Abbau sowie Synchronisation von Dialogen. (-> Anwendungsschichten)
Transport Layer Transportschicht	4	Gewährleistet die zuverlässige Datenübertragung zwischen Endsystemen, bietet Fehlererkennung und -korrektur sowie Datenflusskontrolle, typischerweise durch Protokolle wie TCP und UDP. (-> Transportschichten)
Network Layer Vermittlungsschicht	3	Bestimmt den besten Weg zur Datenübertragung und leitet Pakete zwischen verschiedenen Netzwerken, dabei verwendet sie IP-Adressen. Netzweite Adressierung und Nachführen der Routing Tabelle (-> Transportschichten)
Data Link Layer Sicherungsschicht	2	Sorgt für die fehlerfreie Übertragung von Datenrahmen über die physische Verbindung und regelt den Zugriff auf das Übertragungsmedium. (-> Transportschichten)
Physical Layer Bitübertragungsschicht	1	Verantwortlich für die Übertragung von Rohdatenbits über ein physisches Medium, einschließlich Hardware wie Kabel, Hubs und Netzwerkkarten. (-> Transportschichten)

### Übertragungsmedien

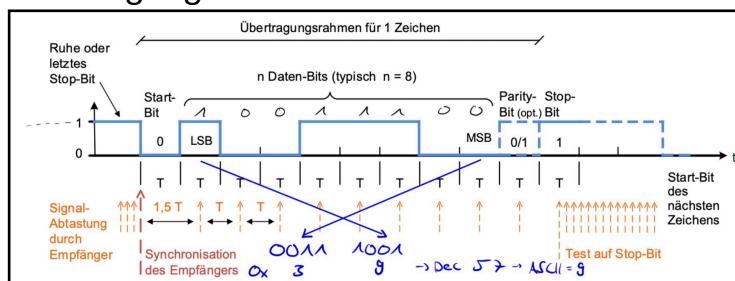
- Liegt unterhalb der Physical Layers und somit **nicht** Teil des OSI-Modells
- Ausbreitungsgeschwindigkeit:  $20 \text{ cm} / \text{ns} = 200'000 \text{ km} / \text{s} \rightarrow \sim 2/3c$
- Signaldämpfung: Leistungsabnahme von Signal  $\rightarrow$  Halbierung  $\approx 3\text{dB}$   $\rightarrow A = 20 \cdot \left( \frac{U_1}{U_2} \right)^2$
- Signal-to-Noise-Ratio (SNR):  $10 * \log(P_{\text{Signal}} / P_{\text{Störung}}) \text{ dB}$
- Noise wird "abgefangen" mit komplementären Signalen, Schirmungen und Verdrillen
- Brechungsgesetz:  $\sin(\beta) \times n_1 = \sin(a) \times n_2$  ( $a, \beta$  = Winkel,  $n_1, n_2$  = Brechungsindex)
- Moden-Dispersion (Verschmierung des Signals, wegen verschiedener Winkeln)  
 $\rightarrow$  Lösung: mehrere Stufen, Gradienten oder sehr dünner Kern im Kabel (= teurer!)

## Physical Layer

- sorgt für die ungesicherte Übertragung eines Bit-Stroms zwischen zwei Systemen



## Übertragung:



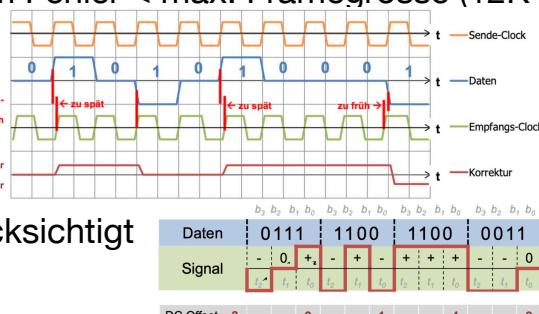
lesen: Rechts nach Links!

1. geschickt = niedrigster Wert

Stop-Bit: nötig, da asynchron!

Empfänger kann zu schnell/langsam sein

-> Summe ppm Fehler < max. Framegrösse (12K Bits)



Synchrone Übertragung ist einfach direkt. Ohne Reverse!

- Takt kann zusammen mit Signal codiert werden!
  - > nur eine Leitung, aber ein Modul mehr
  - > Takt wird zurückgewonnen

PAM3 Kanalcodierung (- + 0): kumulierter DC Offset wird berücksichtigt

-> wenn hoch wird minus eingeleitet

Datenrate, Bandbreite, Baudrate

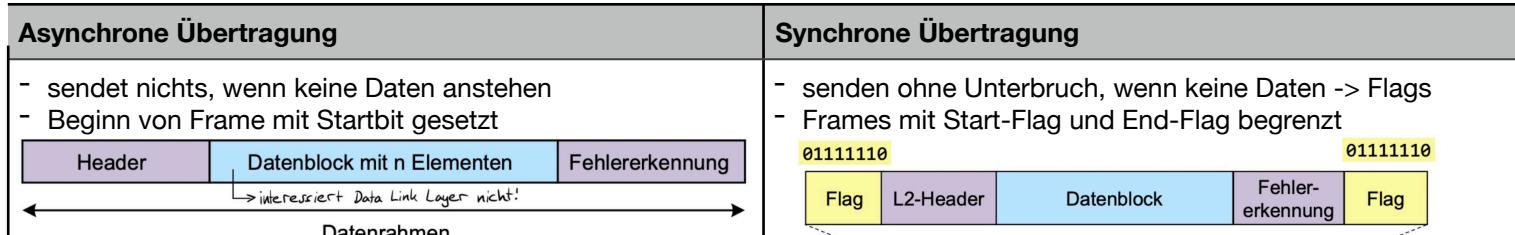
- Datenrate = Bitrate ( $\Rightarrow \text{kBit} = 10^3$  Bit  $\neq 1024$  Bit!)  $\rightarrow \neq$  Baudrate (ausser rein binäres Signal)
- Baudrate = Symbolrate  $\rightarrow$  Ein Symbol kann aus mehreren Bits codiert sein (Bsp.: PAM3)  
Bit/Symbol =  $\log_2(x)$   $\rightarrow$  x = Anzahl unterschiedliche Symbole
- Bandbreite (Hz): Daten/Sekunde  $\rightarrow$  max Baudrate = doppelte Bandbreite ( $f_s = 2B$ )
- Maximale Bitrate:

Hartley:  $R \leq 2B \times \text{Id}(M)$  (M = Anzahl Symbole,  $\text{Id}(M) = \log_2(M)$ )

Shannon:  $C_s = B \times \text{Id}(1 + S/N)$  (S/N = SNR)  $\rightarrow$  genauer da Rauschen berücksichtigt

## Data Link Layer

- sorgt für eine zuverlässige Verbindung durch Framing (Einpacken der Daten), Fehlerkontrolle und Flusssteuerung zwischen direkt verbundenen Systemen



Bit-/Rahmenfehlerwahrscheinlichkeit: (0-1)

-BER: Bit Error Ratio ( $\neq$  Bitfehler pro Zeit)

-FER: Frame Error Ratio (fehlerhafte Frames)

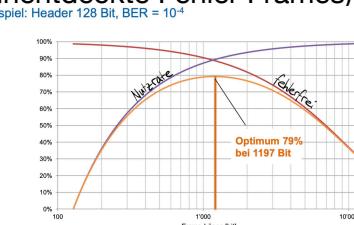
-RER: Residual Error Ratio (unentdeckte Fehler Frames)

$P_{\text{Fehler, Frame}} = N \times p_e$

(=FER,  $p_e = \text{BER}$ )

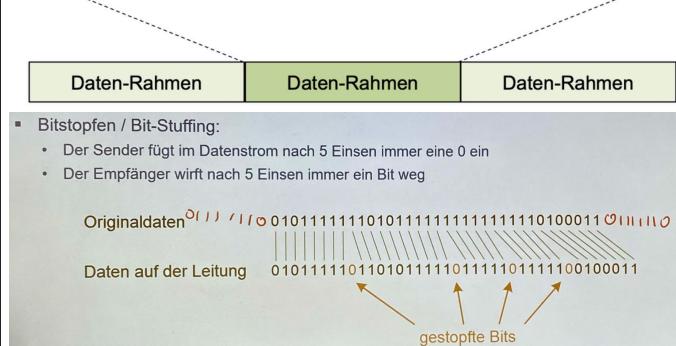
Lange Frames:

Hohe Nutzrate vs.  
Hohe Fehleranfälligkeit->

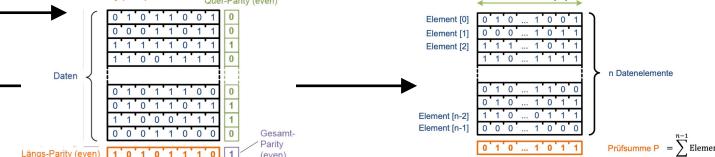
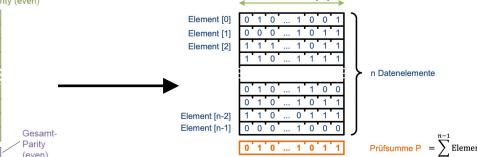


## Synchrone Übertragung

- senden ohne Unterbruch, wenn keine Daten  $\rightarrow$  Flags
- Frames mit Start-Flag und End-Flag begrenzt

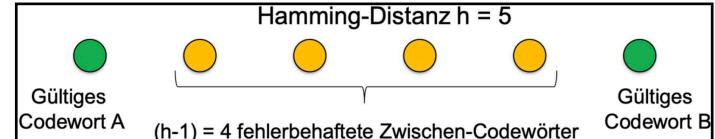


## Fehlererkennung

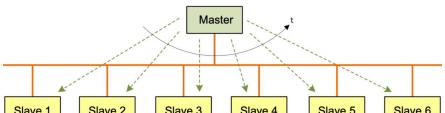
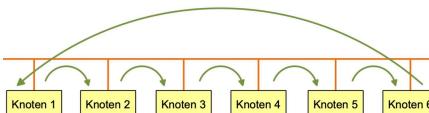
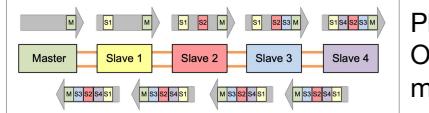
- Parity Bit -> Odd/Even, Anzahl 1er ist un/gerade (1 Fehler kann erkannt werden)
- Längs und Quer Parity: 
- Prüfsumme: 

## Fehlererkorrektur

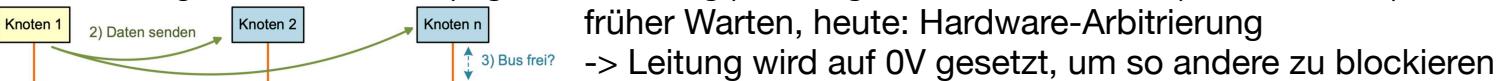
- Backward Error Correction (BEC): Nachricht muss quittiert werden (warten + Rückkanal nötig)
- Forward Error Correction (FEC): statt zu verwerfen  
-> wahrscheinlichste Nachricht abschätzen  
Anzahl korrigierbare Bitfehler:  $k \leq (h-1) / 2$   
Anzahl erkennbare Bitfehler:  $e = h-1$



## Zugriffsmechanismen (gesteuert):

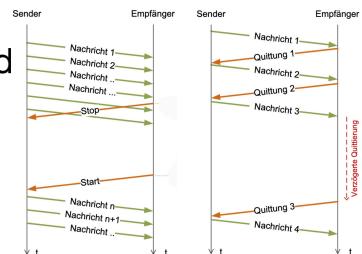
Master-Slave Verfahren	Token Verfahren	Token Verfahren (mit Frame)	Zeitsteuerung
 Konfliktfrei vs. Single Point of Failure	 Deterministisch vs. Aufwändig	 Eine Art Post (effizient)	Plan via Zeit Optimierungen möglich aber erfordert Planung

random Zugriff: kein Master (= gleichberechtigt) -> Fragen vor dem senden (Carrier Sense)



WLAN: geteiltes Medium -> "einfach Losschreien" und zufälliges Warten

RTS/CTS Mechanismus: "Request-to-Send" -> "Clear-to-Send" -> Versand

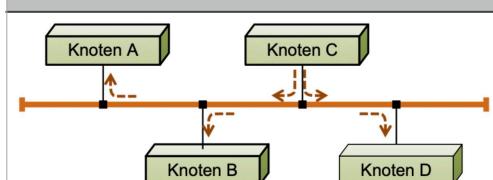
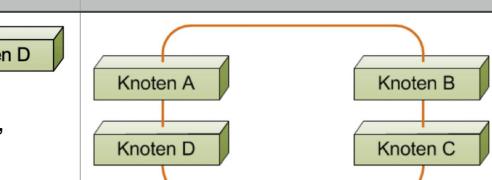


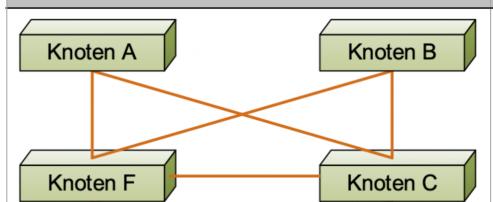
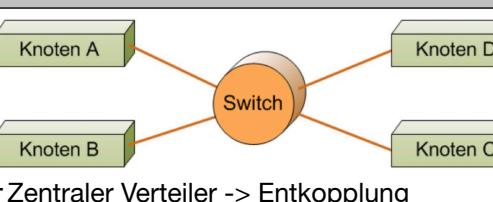
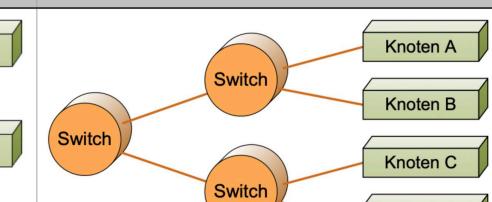
## Flow Control

- Explizit: temporär stoppen (Bild 1)
- Implizit: Sender wartet auf Quittung (neuer Versuch nach Zeit) (Bild 2)

## Ethernet / Local Area Network (LAN)

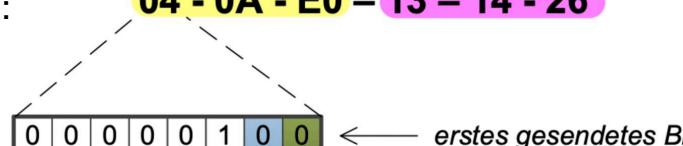
LAN: lokales kleines Netzwerk mit hoher Geschwindigkeit (Topologie):

Bus	Linie	Ring
 - hören Leitung ab keine feste Richtung - Relevanz anhand Adresse erkannt	 - Alle Stationen müssen Daten empfangen, regenerieren, weiterleiten - Ausfall führt zu Segmentierung	 - eventueller "endloser Kreisverkehr", dafür Redundanz

Vermisch (teilweise oder komplett)	Stern	Baum
 - höhere Redundanz -> mehr Aufwand Mehrfaelle Datenlieferung verhindern	 - Zentraler Verteiler -> Entkopplung weniger störungsanfällig	 - Lokale Kommunikation möglich

Übertragungsarten: Unicast:

**04 - 0A - E0 - 13 - 14 - 26**



MAC Adressen:

Multicast:

Broadcast:

-> Hersteller + Laufnummer

Individual/Group Bit (I/G):

0 = individual address (Normalfall),

1 = group address z.B. Broadcast FF-FF-FF-FF-FF-FF

Universally/Locally Bit (U/L):

0 = universally administrated address (Normalfall)

1 = locally administrated address

## Ethernet - Frame Format (Länge: 64-1518 Bytes)

MAC-Overhead (18) = DA+SA+Len/Typ (Header)+FCS (Trailer)

-> zuerst geschickt: Bit-Level: LSB, Byte-Level: MSB

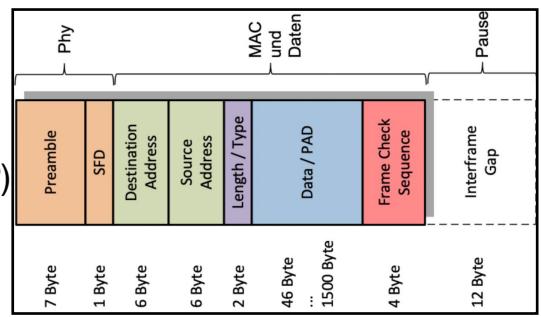
Length/Type:  $\leq 1500$ : Länge(Data ohne PAD),  $\geq 1536$ : Type(z.B. IP)

Data/Padding: wenn unter 46 Bytes -> mit "0" aufgefüllt

Frame Check Sequence (FCS): IEEE CRC-32 Algorithmus

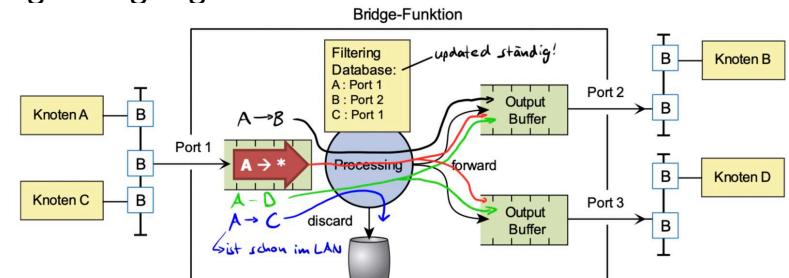
Interframe Gap: Zwangspause (NICHT Teil von Ethernet)

-> Sendedauer: Alles ohne IFG, Dauer der Leitungsbelegung: Alles



## Network Gear

- Repeater: stärkt Signal ohne Überprüfung!  
(arbeitet auf der physical Layer)
- Switch: prüft Checksummen, verbindet Netzwerke (arbeitet auf Data Link Layer) ->
  - > Wenn Zielport nicht bekannt: Flooding
  - > Adresse wird dann gelernt

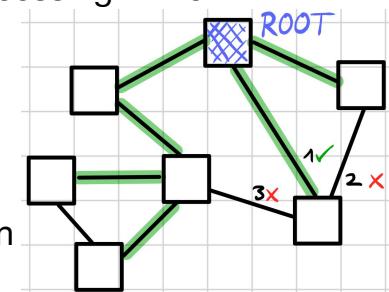


Repeaters sind viel schneller! Switches haben zusätzlich noch eine "processing" time

## Redundanz (Spanning Tree)

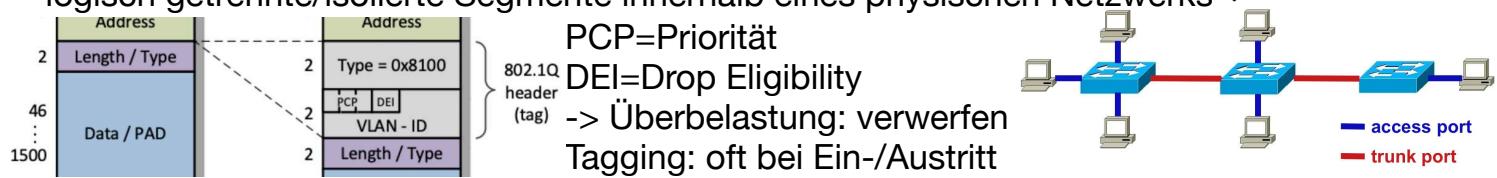
Ziel: loop-freie Topologie -> Algorithmus:

- Jeder nimmt an: "ich bin Root"
- Austausch von ID mit Nachbarn (tiefer ID gewinnt)
  - > Verlierer updated die Root und seine Distanz zu ihr
- der Pfad mit den kleinsten Kosten nehmen, andere werden verworfen



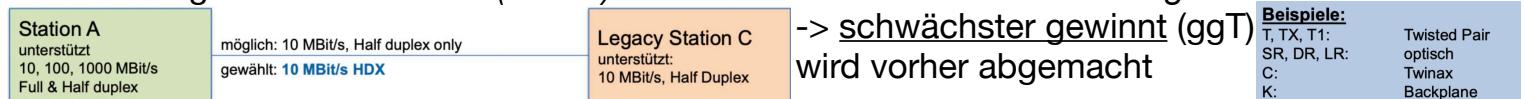
## VLANs und Quality of Service

- logisch getrennte/isolierte Segmente innerhalb eines physischen Netzwerks



Quality of Service (QoS): **Priorisierung** aufgrund der PCP-Werte (0...7) -> Strategie: frei wählbar

Bezeichnungs-Schema: **Bitrate (Mbit/s) BASE oder BROAD - Art/Codierung des Mediums**



	10BASE-T IEEE 802.3i	100BASE-TX IEEE 802.3u	1000BASE-T IEEE 802.3ab	10GBASE-T IEEE 802.3an	Kompatibilität (10)/ 100/1000BASE-T durch: - Gleiche Schnittstelle (Frame/Format) zwischen PHY und MAC - Autonegotiation PHY Codierung ist unterschiedlich - Höhere Datenrate -> höhere Ansprüche an Verarbeitung/ Algorithmen im PHY - Reduktionmassnahmen werden wichtiger
Kabelkategorie (für 100m Link Distanz)	CAT3 B = 16 MHz CAT5 B = 100 MHz	CAT5 B = 100 MHz CAT6 B = 250 MHz	CAT5 B = 100 MHz CAT6 B = 250 MHz	CAT6A B = 500 MHz CAT7/7A B = 600/1000 Mhz	
Line Coding	Manchester 2 Adernpaare simplex	MLT-3 (synchron), 4B5B 2 Adernpaare simplex	PAM-5 (plus scrambling) 4 Adernpaare duplex	PAM-16, 64B/65B, FEC 4 Adernpaare duplex	
Baudrate	10 MBaud	125 MBaud	4 x 125 MBaud	4 x 800 Mbaud	
Link Pulses	NLP (Link Presence Detection)	FLP (Autonegotiation, Autopolarity)	FLP (Autonegotiation, Autopolarity, Next Page)	FLP (Autonegotiation, Autopolarity, NextPage)	
Erfolgreich durch kompatible Elemente					



## Internetprotokolle (Network Layer)

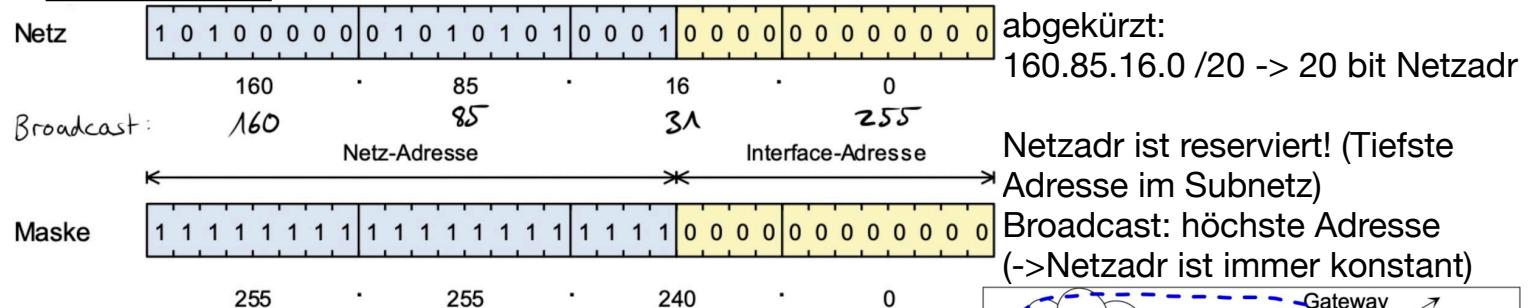
- Forwarding: Weiterleiten der Daten durch Routingtabellen (-> Aufbau durch Routing)
- 4 Grundsätze (Internet): jedes Netzwerk aus unabhängig sein, Kommunikation is "best effort", Verbindung via Black Boxes (Router), keine zentrale Funktionssteuerung.
- Network Layer kümmert sich um IP-Pakete weiterleiten (ohne Korrektur!)

## Router

- verbindet Netze via dem entsprechenden Interface (wirkt für das Netz wie ein normaler Knoten)
- Kann für Segmentierung/Abgrenzung eines grösseren Netzes verwendet werden

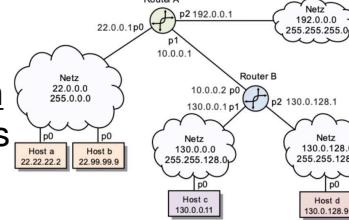
## IPv4

- Aufbau: Netz-Adresse + Interface-Adresse (insgesamt 4 Byte) -> jeder Host hat mind. Eine -> 4 Zahlen (0-255) mit Punkt getrennt (z.B. 128.154.26.246)
- Subnetzmaske bestimmt Grösse von Netz und Interface Adresse:



## Routing und Forwarding

- Routing-Tabelle enthält wie jedes Netz erreicht werden kann -> erster passender wird genommen (default passt immer!)
- Flaches Routing: jeder Weg bekannt -> sehr grosse Tabelle
- Hierarchisches Routing: kennt nur seine internen Interfaces für alles Außenstehende kennt er EINEN Router. ->



Routing-Tabelle Host a		Routing-Tabelle Host c	
22.0.0.0	255.0.0.0	p0 (direkt)	130.0.0.0
default		p0 22.0.0.1	255.255.255.0

Netzadresse	Netzmaske	Port	Gateway
160.85.18.0	255.255.255.240	eth1 (direkt)	
160.85.19.0	255.255.255.0	eth2 (direkt)	
160.85.16.0	255.255.254.0	eth0 (direkt)	
default	0.0.0.0	eth0	160.85.16.1

## Mögliche Werte in der Subnetzmaske:

255 (1111'1111) /24 256 - 2

254 (1111'1110) /23 512 - 2

252 (1111'1100) /22 1'024 - 2

248 (1111'1000) /21 2'048 - 2

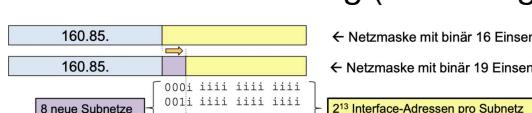
240 (1111'0000) /20 4'096 - 2

224 (1110'0000) /19 8'192 - 2

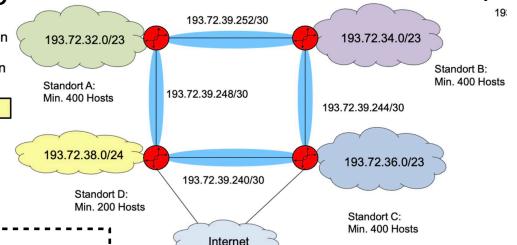
192 (1100'0000) /18 16'384 - 2

128 (1000'0000) /17 32'768 - 2

0 (0000'0000) /16



## Subnetting (Aufteilung eines Netzes in kleinere Netze)

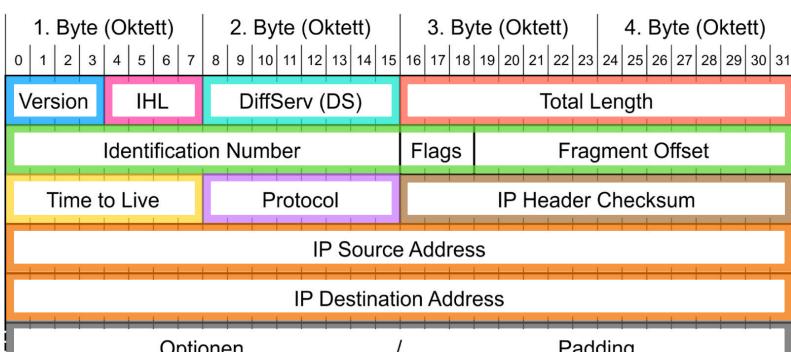


### Spezielle IP-Adressen:

- Loopback: A-Netz 127.0.0.0/8
- Private Adressbereiche

## IPv4-Header Format

- **Version:** Wert 4 oder 6 (IPv4 = 4)
- **Internet Header Length:** Länge des Headers (Vielfachen von 4 -> Zeilen) (max. Wert 15 (4 Bits) = 60 Bytes)
- **Differentiated Services:** Priorisierung
- **Total Length:** Länge des Pakets in Bytes (inkl. Header und Nutzdaten)  
-> zu grosses Paket: fragmentieren
- **Id-Num, Flags, Frag. Offset:** Fragmentierung und Reassemblieren von IP-Paketen
- **Time to Live:** Übrige Lebenszeit für ein Paket. Router dekrementiert Wert beim Weiterleiten (=keine Zirkulation)
- **Protocol:** die Nutzdaten werden an den richtigen Handler übergeben->
- **Header Checksum:** berechnet jeder Router neu (schützt nur Header)
- **Source Address/Destination Address:** IP-Adresse von Sender/Empfänger-Host
- **Options / Padding (variabel):** selten, Padding um Vielfaches von 32 Bit aufzufüllen



Protocol	Name
1	ICMP – Internet Control Message Protocol
6	TCP – Transport Control Protocol
17	UDP – User Datagram Protocol

## Fragmentierung: Identifikations-Nummer, Flags, Fragment Offset

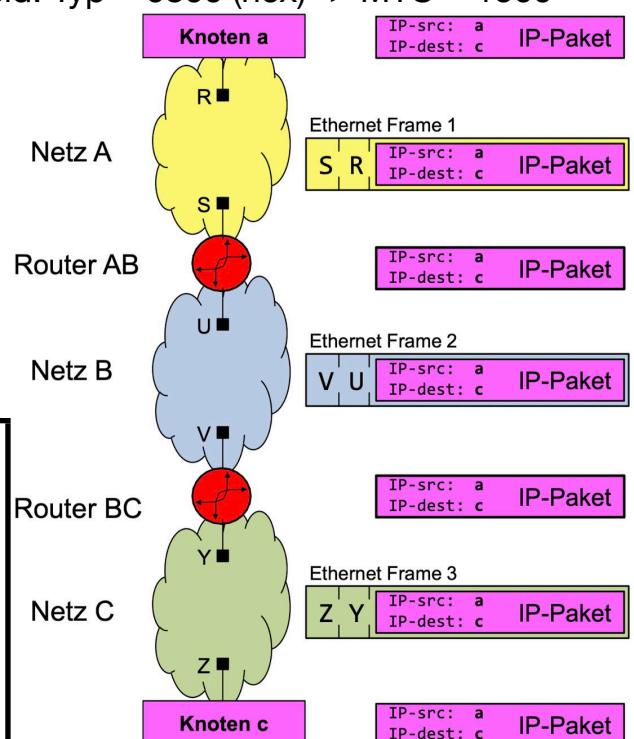
-> Rekonstruktion, auch wenn nicht in richtigen Reihenfolge erhalten

- Fragmentierung wird von Routern vermieden - sondern Sender  
Sender braucht MTU (Maximum Transfer Unit) = maximale Payload
- Pakete werden erst beim Ziel-Host reassembliert -> unterschiedliche Pfade möglich

Feld	Position	Werte	Funktion
	0	0	Reserved, must be Zero
DF	1	0 / 1	May / Don't Fragment
MF	2	0 / 1	Last / More Fragments

## Kapseln und Adressauflösung = Adress Resolution Protocol

- heute meist: IP-Paket direkt im Nutzdatenteil -> eth-Feld: Typ = 0800 (hex) -> MTU = 1500
- IP Source und Dest bleiben konstant!
- A muss aus der IP-Adresse von AB die hardware Adresse S herausfinden -> **Adressauflösung (ARP)**
- Wenn Hardware Adresse nicht bekannt:  
Alle Knoten via Broadcast anfragen. Der gemeinte kennt eigene Hardware-Adresse und meldet diese.  
-> Ein Ethernet-Frame, Type 0806, Dest: FF-... Hard: 0
- Jeder Knoten hat ein ARP-Cache
- Zu Beginn eine rekursive Anfrage um zu sehen, ob niemand sonst die gleiche Adresse hat.



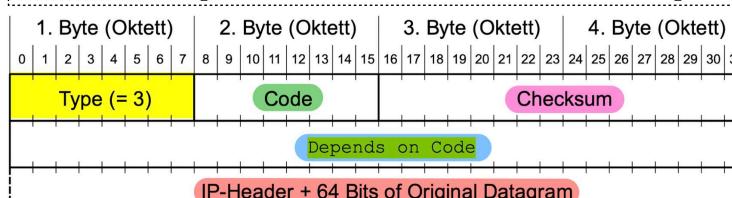
## Internet Control Message Protocol (ICMP)

Übertragung von Fehlermeldungen/Informationen

- nutzdirekt IP -> Keine Garantie, dass sie ankommt
- Meldungen sind informativ gedacht!

### ICMP Meldungstypen

- |                              |                         |
|------------------------------|-------------------------|
| • 0: Echo Reply              | • 11: Time Exceeded     |
| • 3: Destination Unreachable | • 12: Parameter Problem |
| • 5: Redirect                | • 13: Timestamp Request |
| • 8: Echo Request            | • 14: Timestamp Reply   |



Feld	Router -> Absender: Wenn Paket nicht weitergeleitet werden kann:	
	Type	3
Type	Code	0 = net unreachable, 1 = host unreachable, 2 = protocol unreachable, 3 = port unreachable, 4 = fragmentation needed and DF set, 13 = communication administratively prohibited
	Checksum	Prüfsumme über die ICMP Meldung
	IP Header + 64 Bits of Original Datagram	Information für den Empfänger zur Zuordnung der Meldung zu einem gesendeten IP Paket

Host 160.85.31.3 versucht folgendes Paket an

Hist 160.85.29.99 zu senden:

Sender + Destination adress?

4500 0028 8b10 0000 0711 a8a4 a055 1f03  
a055 1d63 8b0d 829d 0014 a348 030a 0000  
7504 1137 407c 0800

Ein Router kennt keinen Weg und sendet diese "Destination Unreachable" Message zurück:

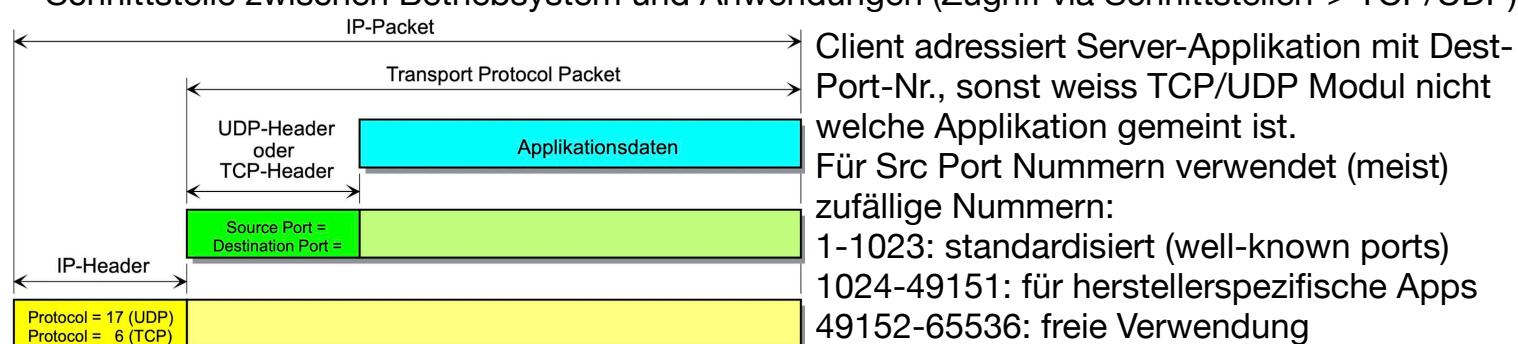
Protokoll: 01 = ICMP, Typ: 03, ICMP-Header

4500 0038 8038 0000 fd01 5bc0 a055 821e  
a055 1f03 0301 4bf7 0000 0000 4500 0028  
8b10 0000 0711 a8a4 a055 1f03 a055 1d63  
8b0d 829d 0014 a348

**Path MTU discovery:** -> kleinste MTU auf dem Weg (PMTU) -> keine Fragmentierung unterwegs  
-> Startet mit eigener MTU und versucht zu senden. Wenn "Dest. Unreachable" mit Code 4 "fragmentation" zurückkommt: PMTU reduzieren auf "Next-Hop MTU" (enthalten im Octet 5...8)

## Transport Layer

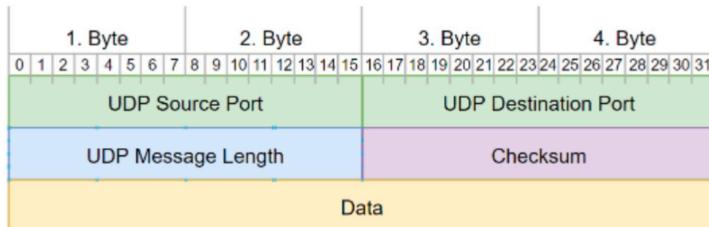
- Schnittstelle zwischen Betriebssystem und Anwendungen (Zugriff via Schnittstellen-> TCP/UDP)



Client adressiert Server-Applikation mit Dest-Port-Nr., sonst weiss TCP/UDP Modul nicht welche Applikation gemeint ist.  
Für Src Port Nummern verwendet (meist) zufällige Nummern:  
1-1023: standardisiert (well-known ports)  
1024-49151: für herstellerspezifische Apps  
49152-65536: freie Verwendung

## UDP - User Datagram Protocol

-> verbindungslose, unzuverlässige Multi- / Demultiplexen der Datagramme zu den Applikationen



### UDP-Header

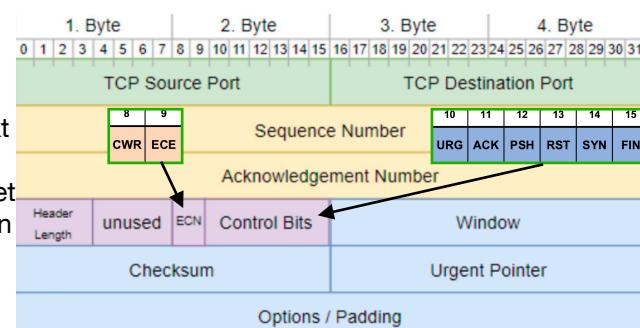
- **Source Port** Sendende Applikation
- **Destination Port** Applikation des Empfängers
- **Message Length** Länge des Datagramms
- **Checksum** Prüfsumme über einen Pseudo-Header, UDP-Header und Daten (kann Null sein)
  - Pseudo-Header: IP Source- und Destination Address, Protocol Feld, Länge des Datagramms
  - \* so können fehlgeleitete Datagramme erkannt werden

## TCP - Transmission Datagram Protocol

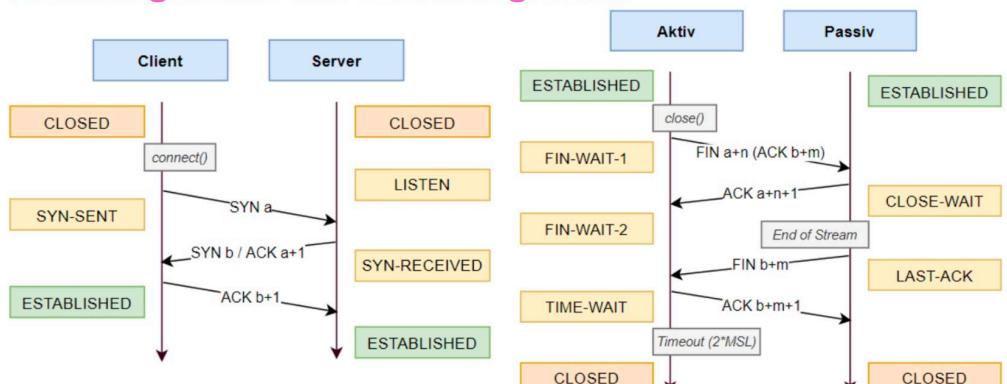
- **Verbindungsaufbau:** Verbindung zwischen Client/Server wird vor Übertragung hergestellt.
- **Zuverlässiger Verbindungsaufbau:** Beide Endpunkte bestätigen die Verbindung aktiv.
- **Hohe Zuverlässigkeit:** Daten werden verlustfrei und in richtiger Reihenfolge übertragen.
- **Voll duplexübertragung:** Gleichzeitige, unabhängige Übertragung in beide Richtungen.
- **Stream-Schnittstelle:** Senden/Empfangen einer unstrukturierten Byte-Folge.
- **Graceful Termination:** Gewährleistung der Zustellung aller Daten beim Verbindungsabbau.
- **Punkt-zu-Punkt Kommunikation:** Direkter Datenaustausch. kein Multi/Broadcast.

### TCP-Header:

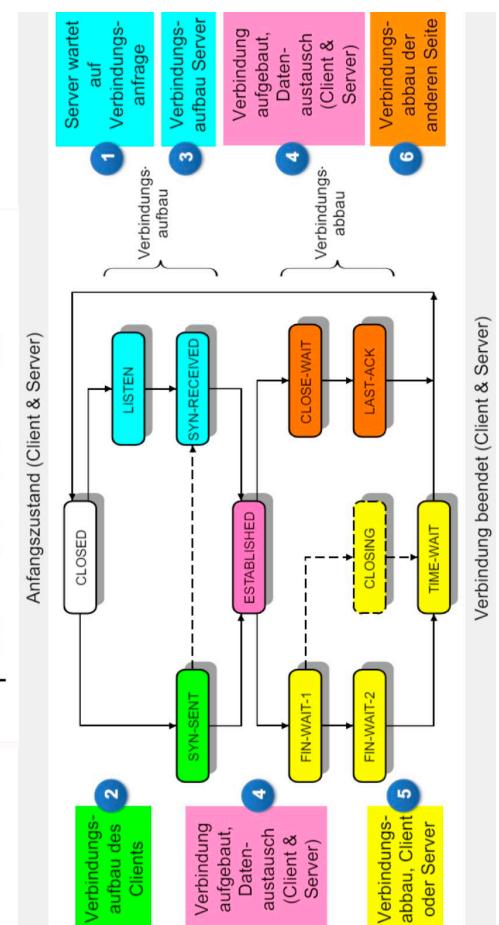
- **Src/Dest. Port:** Beim Aufbau: Dest Port = gewünschter Dienst
- **Seq. Num:** Sicherstellung Reihenfolge + Erkennung fehlender Daten
- **Ack. Num:** nächste erwartete Seq Num - n+1 -> alle bis und mit n korrekt
- **Header Length:** in Double Words (32 Bit Einheiten -> Faktor 4)
- **ECN Flags:** CWR: Überlastfenster verkleinert, ECE: Überlastung gemeldet
- **Control Bits:** Für Verbindungsauflösung, welche Felder Bedeutung haben
  - URG: Urgent-Pointer enthält gültigen Wert
  - ACK: Acknowledgement Number enthält gültigen Wert
  - PSH: Push-> Daten sofort ohne Buffering an die Anwendung senden
  - RST: Reset-> Verbindung rücksetzen / geschlossener Port Signal
  - SYN: Synchronize -> Verbindung aufbauen
  - FIN: Verbindung abbauen
- **Window:** aktuell verfügbare Puffergröße Empfänger (in Bytes)
- **Checksum:** Prüfsumme über Pseudoheader (TCP und Daten)
- **Urgent Pointer:** URG = 1 -> Position (Byte Offset) wichtigen Daten
- **Options:** Häufig Max Segment Size die empfangen werden kann



## Verbindungsauftau und Verbindungsabbau



ACK nr. muss mit der Anzahl der Bits der empfangenen Daten aktualisiert werden.



## Vollständiges Beispiel

## TCP Adaptive Elemente

Verbindungsauftbau:

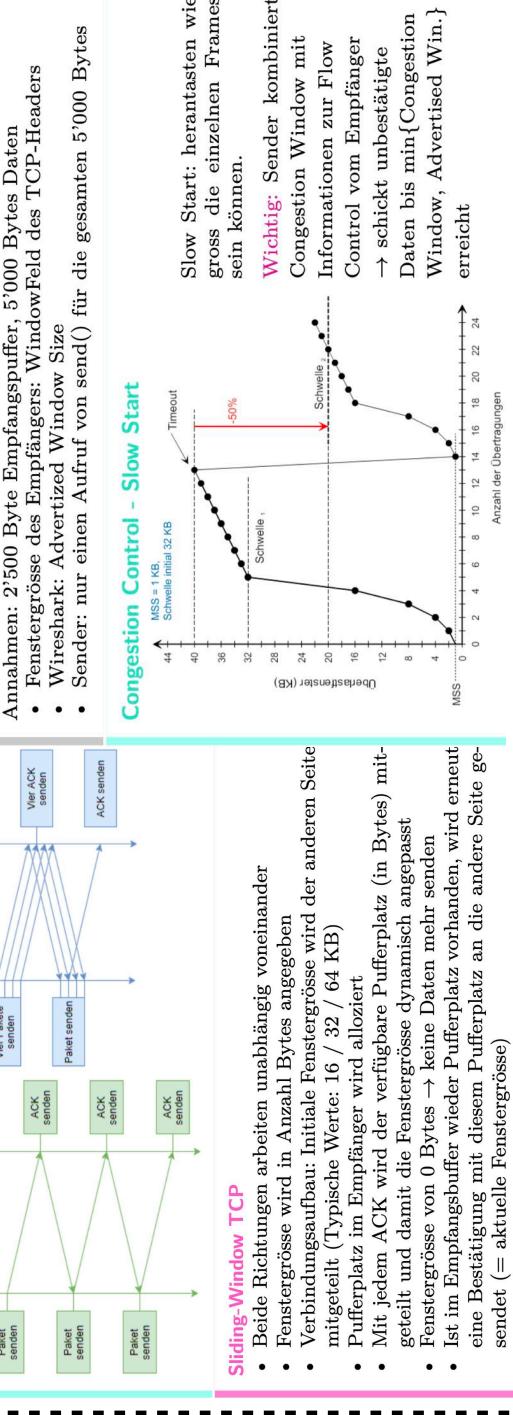
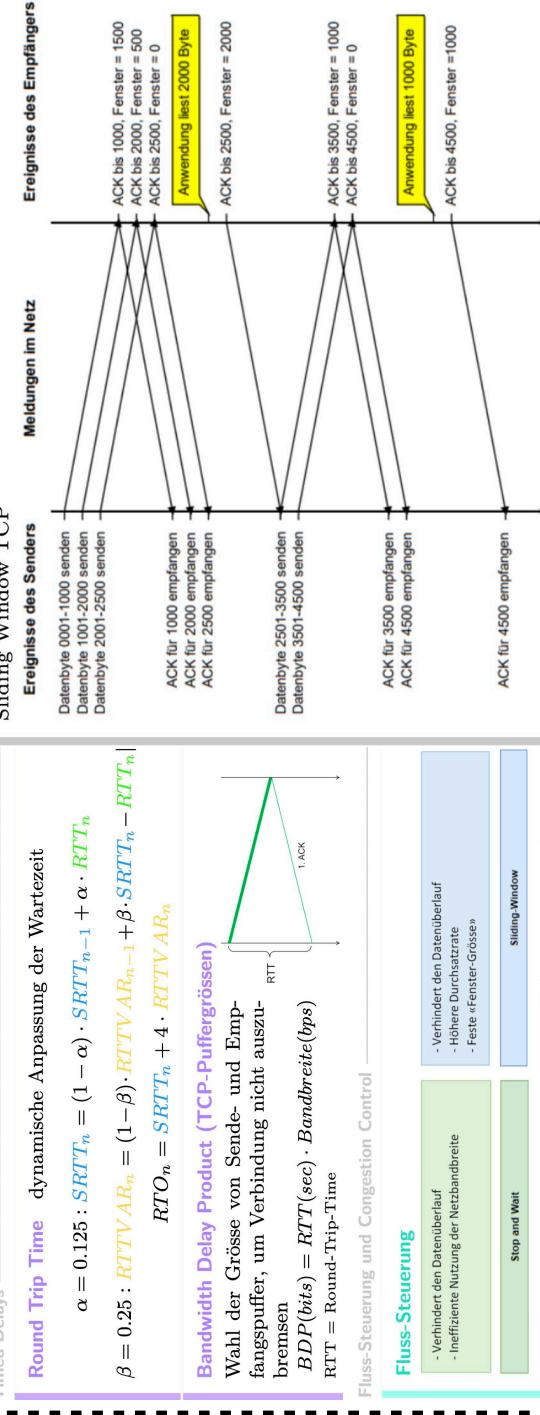
- Server „horcht“ (LISTEN) auf einer bestimmten Port Nummer
- Client sendet Segment mit SYN=1 und zufälliger init. Sequenznummer a (ACK=0, weil ACK nr. ungültig)
- Server bestätigt Sequenznummer mit ACK nr. a+1 und ACK=1,
- wählt zufällige initiale Sequenznummer b, setzt SYN=1
- Client bestätigt b mit ACK nr. b+1
  - Erstes Byte vom Client zum Server hat Sequenznummer a+1
  - Erstes Byte vom Server zum Client hat Sequenznummer b+1



Datenaustausch: TCP-Nachrichten werden bi-direktional ausgetauscht

## Herausforderungen zur Zuverlässigkeit zwischen Ethernet/TCP:

Problem	Schicht 2	Schicht 4	Massnahmen bei TCP
Nachrichtenverlust	$P_{Verlust} = FER$	Varierend	Positives ACK
Telegramm-Reihenfolge	fix	kan variieren	Sequenznummern
Round Trip Time	konstant, us .. s	varibel, ms .. s	Adaptiver Retransmission Timeout
Überlast des Empfängers	kommt vor	nur indirekt beobachtbar	Sliding Window mit dynamischer Fenstergrösse
Überlast des Netzwerks	direkt beobachtbar (Medium)	nur indirekt beobachtbar	Slow Start (Congestion Window)
Neustart von Hosts	direkt beobachtbar	nur indirekt beobachtbar	3 Weg Handshake, Initialisierung Sequenznr.



- Beide Seiten können den Verbindungsabbau einleiten
- Ist eine Richtung geschlossen (FIN, ACK), so können in die andere Richtung immer noch Daten gesendet werden (Half-Closed)
  - In Richtung der "geschlossenen" Verbindung wird nicht mehr kommuniziert (Acknowlegde number mismatch)
  - Falls die zweite Seite die Verbindung auch schliesst, können die 3. und die 4. Nachricht zusammengefasst werden → FIN/ACK



## Application Layer

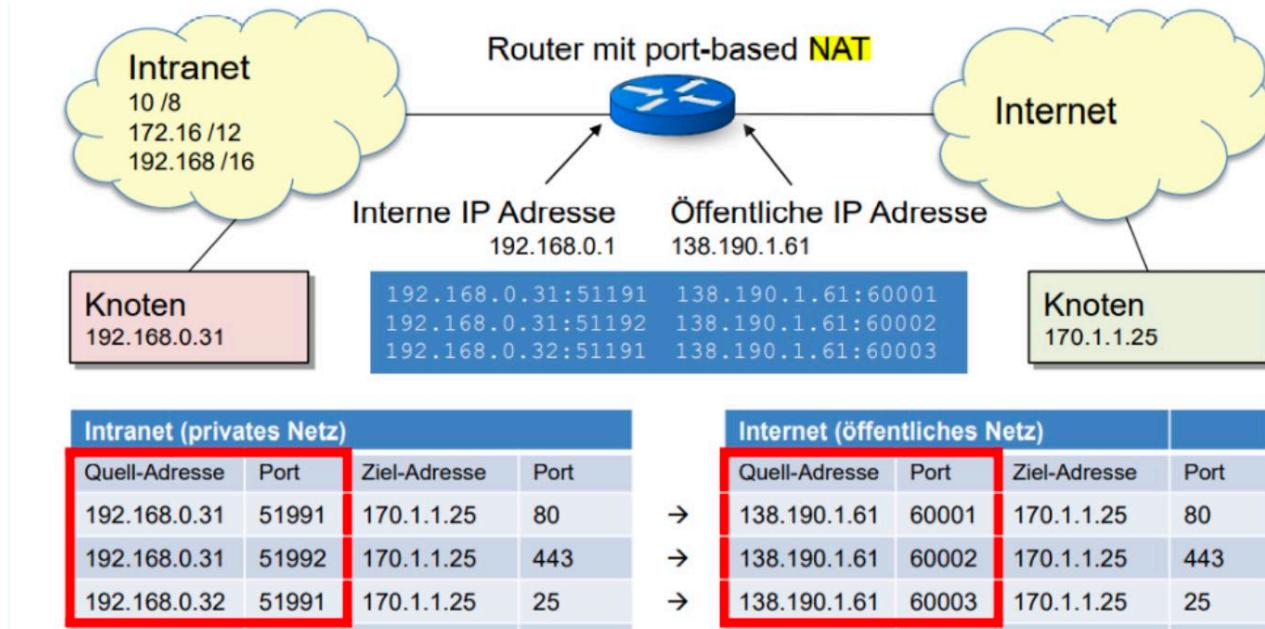
### NAT - Network Address Translation - Port-basiertes NAT = NAPT

- Ersetzt private IP-Adr. durch public IP des Gateways/Routers
- Ersetzt private Port-Nr. des Hosts durch freie zulässige Port-Nr. des Gateways/Routers
- Mapping privater IP-Adr. und Port-Nr. zur öffentlichen Port-Nr. auch statisch möglich, aber nur Port-Nr wird übernommen

Problem mit NAT: Verletzung des OSI-Layer-Konzepts

Um Port im TCP Header zu ändern müssen Daten im IP-Frame verändert werden

-> Netzwerkfunktion greift auf Transport Header zu, IP-Adresse/Portnummer werden so verändert



∀ Hosts im privaten Netz 192.168.0.0/8: Default-Gateway 192.168.0.1

NAT Tabelle

Inside	Outside
192.168.1.21:54321	60000

6. NAT:

Port 80 ist der offizielle Port für HTTP TCP/UDP Verbindungen

