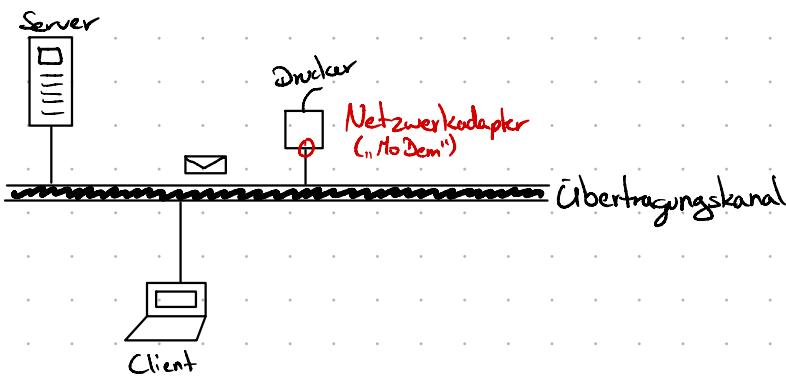


Ein Kommunikationsnetzwerk (Netzwerk) besteht aus einer Menge von End-Systemen (Host), die über einen speicherlosen Übertragungskanal verbunden sind und darüber Nachrichten austauschen.



## End-System

- (1) Anwendung: E-Mail, WhatsApp
- (2) End-Geräte: PCs, Server, Laptops, IoT
- (3) Übertragungsschnittstelle: Netzwerkadapter, Modem

## Übertragungskanal

- (1) Medium: Kupfer-Kabel, Glasfaser, Luft
- (2) Vermittlungseinrichtungen: WLAN-Access Point, Ethernet-Switches, IP-Router  
(Netzwerkgeräte)

## Klassifikation von Netzwerken

LAN = Local Area Network (Campus, Firmengebäude,...)

WAN = Wide Area Network (Weitverkehrsverbindung, ihre Internetanbindung zu ihrem ISP)

PAN = Personal Area Network (Bluetooth,...)

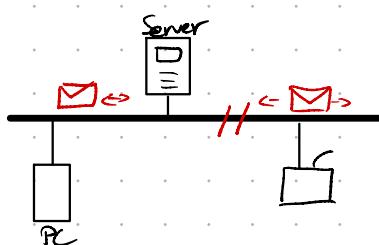
Verwendet man Funktechnik, dann ergänzt man den Namen um ein „w“ für „Wireless“

WAN: LTE, 5G, Satellitenverbindung

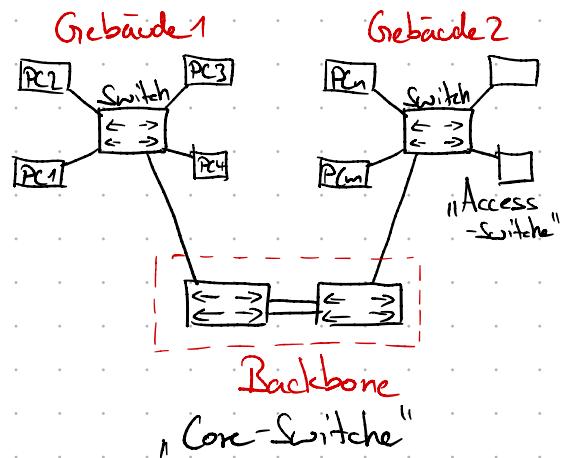
WLAN: Hotspot

WPAN: Bluetooth, ZigBee

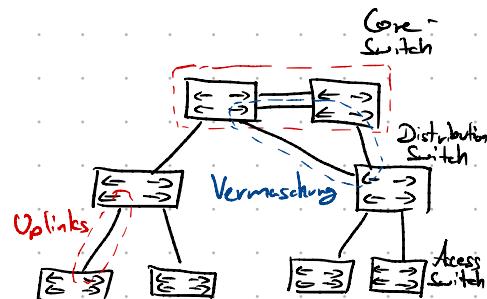
## Bus Topologie



## Sterntopologie



## Baumtopologie



Vermischung bedingt ein Protokoll um das Kreisen von Nachrichten zu verhindern.

STP: Spanning Tree Protocol

## Latency und Round Trip Time (RTT)

Die Paketumlaufzeit RTT gibt die Zeit an, die ein kleines Paket benötigt um in einem Netzwerk von der Quelle zum Ziel und zurück zu reisen.

$$RTT = 2 \cdot \text{Latenz}$$

Ein Netzwerkprotokoll steuert das Senden, Übertragen und Empfangen von Nachrichten über ein Kommunikationsnetzwerk.

Damit 2 Systeme sich verstehen müssen sie standardisierte Protokolle verwenden.

## Begriff des IT-Dienstes

- **Server-Dienst:** Serveranwendung stellt einen Dienst bereit.
- **Netzwerkdienste:** Netzwerkprotokolle stellen Dienstleistungen zur Verfügung.  
(TCP: zuverlässige Datenübertragung)
- **Internetdienst:** Ein im Internet bereitgestellter Serverdienst.  
(Cloud-Speicher, Videoconferencing)

- **Serviceleistung:** Bereitstellung einer IT-Dienstleistung im Sinne von Dienstqualität (Verfügbarkeit: 99,99%, Bandbreite, Latenz,...)

## Leistungsvermittlung:

Einteilung der Ressourcen in diskrete Einheiten, die einem Nutzen fest zugewiesen werden.

- Solang die Verbindung steht sind die Ressourcen exklusiv reserviert.  
→ Nachteil: Ressourcen bleiben ungenutzt während einer Sendepause

## Ressourcenaufteilung:

- Frequenzmultiplexing (FDM)

→ Verbindung erhält einen permanenten, festen Anteil der Bandbreite

- Zeitmultiplexing (TDM)

→ Senderzeit wird in feste Zeitintervalle aufgeteilt

## Wartezeit in Puffern

Warteschlangentheorie für die mittlere Wartezeit im Puffer eines Switch oder Routers:

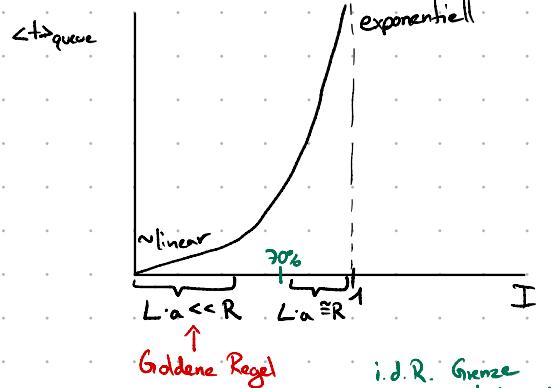
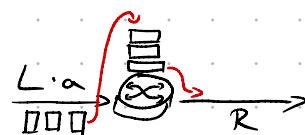
$R$ : Übertragungsrate bit/s

$L$ : Paketgröße bit

$a$ : Paketankunftsrate Pakete/s

$$\leftrightarrow_{\text{queue}} = \frac{L}{R} \cdot \frac{I}{1-I} ; I = \frac{L \cdot a}{R}$$

↑  
Verkehrswert



i.d.R. Grenze  
ab hier Ausbau des Systems notwendig

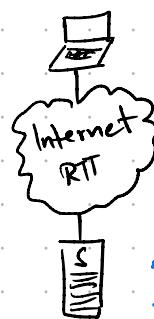
Netzwerk Anwendungen bestehen aus Programmen, die auf verschiedenen Hosts und über ein Netzwerk miteinander kommunizieren.

Host: hat einen eindeutigen IP Knoten

Client - Server - Architektur

P2P - Architektur

Hybrides - Szenario: Client & Peer 2 Peer



Web-Browser  
• dynamische IP-Adresse  
• sporadisch angeschlossen  
• keinen Dienst  
↳ dynamische Port-Nummer

• feste IP-Adresse  
• immer eingeschaltet  
• Dienst bestellen über  
eindeutige Portnummer

Ein Prozess ist ein Programm, das auf einem Host läuft

a) innerhalb Host: IPC Internet Protocol

$$RTT = 2 \cdot t_{\text{latenz}}$$

Satellit meist 36000km  
Starlink 340-13000km

$$t_{\text{latenz}} = \frac{2 \cdot d}{c}$$

Abschallzeit  
Ausbreitungsgeschw.

Übertragungsrate R  
Paketgröße L

### Leistungsermittlung

↳ Frequenzmultiplexing  $\equiv$  FDM  
↳ Zeitmultiplexing  $\equiv$  TDM

$$P(n) = \binom{N}{n} \cdot p^n \cdot (1-p)^{N-n}; \text{Binomialverteilung}$$

$P(n)$ : Wahrscheinlichkeit das  $n$  Benutzer gleichzeitig aktiv sind

$N$ : Gesamtzahl Benutzer,

$n$ : gleichzeitig aktive Benutzer

$p$ : Wahrscheinlichkeit das ein Benutzer sendet

Mittlere Anzahl (max. gleichzeitig aktive Benutzer)  
 $\langle n \rangle = p \cdot N$  mit  $p = \frac{t_{\text{aktiv}}}{t_{\text{aktiv+passiv}}} \ll 1$

### Store-and-Forward-Verzögerung

$$t = (n+1) \frac{L}{R}$$

Hopcount

### Ausbreitungsverzögerung

$$t = \frac{d}{c}$$

Leistungslänge  
Ausbreitungsgeschw.

mittlere Wartezeit  
 $t_{\text{queue}} = \frac{L}{R} \cdot \frac{(L-a)/R}{(L-a)/R}$  für  $L-a < R$   $\frac{\text{durchschnittliche Paketlänge}}{\text{Verkehrswert}}$   
↳ Goldene Regel:  $L-a < R \cdot 70\%$

Gesamtverzögerung  
 $t = (t_{\text{verb}}) + t_{\text{queue}} + \frac{L}{R} + t_{\text{latenz}}$

Durchsatz:  $L/a$

Anwendungs geschicht

Darstellungs geschicht

↓ Transport geschicht

Transport schicht

Vermittlungs geschicht

Sicherungs geschicht

↓ Bitübertragungs geschicht

überträgt Webseiten usw. Client-Server  
**HTTP, SMTP, DNS**

Daten aufbereitung (Kodierung, Kompression)

Synchronisation

Daten transfer (zuverlässig)  
**TCP, UDP**

Weiterleitung über Netzwerk (Weg)  
**ID**

Ethernet / WLAN

Überträgt Bit auf Kabel  
**Kabel / Funkwelle**

Jede Schicht erweitert um Header  $\hat{=}$  Kapselung  
↳ Protokoll overhead

HTTP	80/tcp
HTTPS	443/tcp
SMTP	25/tcp
FTP	20
DNS	53/tcp/udp
DHCP	67/68/udp
SNMP	161/162/tcp

Well known 0-1023  
Registered 1024-49151  
Dynamic/Private 49152-65535

## TCP

- Zuverlässiger Transport
- Verbindungsorientiert (Handshake)
- Flusskontrolle (Empfänger)
- Überlastkontrolle (Netzwerk)
- Zeit / Bandbreitengarantie
- 20 Byte Header

## UDP

- Unzuverlässiger Transport
- ↳ Schneller als TCP (Video/Telefon)
- nicht enthalten in UDP
- 8 Byte Header

## Nicht persistentes HTTP/1.0

- max 1 Obj. pro TCP
- für 1 Datei:  $2 \cdot RTT + t_{Proc} + t_{Übertragung}$

$$t_{Übertrag} = \frac{L}{R} \cdot (n+1)$$

$\uparrow$   
Hops  $\delta$

## Persistentes HTTP/1.1

- 1 Mal TCP am Anfang
- ↪ Dann Datei nach Datei
- RTT für TCP + für jede Datei:  $RTT + t_{Proc} + t_{Übertragung}$

## Persistentes HTTP/1.1 mit Pipelining

- 1 Mal TCP am Anfang
- ↪ Objekte gebündelt
- RTT für TCP + RTT für HTML +  $t_{Proc} + t_{Übertragung}$  + RTT für alle Obj. + für jedes Obj.:  $t_{Proc} + t_{Übertragung}$

## Persistentes HTTP/2.0 mit Multiplexing

- 1 Mal TCP am Anfang
- RTT für TCP + RTT für HTML +  $t_{Proc} + t_{Übertragung}$  (enthält ev. Server Push CSS)
- ↪ + RTT für alle Objekte +  $t_{Proc\ max} + t_{Übertrag\ max}$  (Übertragung parallel → nur größtes Paket zu beachten)

## Proxyserver

- Cachet Webseiten → schneller aufrufbar intern + geringe Datenverbindung zu Internet kompensierbar (s. Goldene Regel)
- GET: Proxyserver überprüft regelmäßig ob seine gespeicherte Seite aktuell ist (modified timestamp)
  - ↪ Aktuell: keine Daten über Internet; Nicht Aktuell: Neue Daten werden zurück geschickt

## DNS

Host → lokaler DNS-Server

lokal → DNS-Root-Server (Top Level) → lokal	Iterativ
lokal → TLD-Server (Second-Level/Autoritativer) → lokal	
lokal → Autoritativer DNS-Server (IP-Adresse) → lokal	
lokal → Host	

=> 4 DNS Anfragen = 3 RTT +  $t_{Übertragung}$  lokaler Server

rekursiv

Host → lokal → Root → TLD	
→ Autoritativer → TLD → Root	
→ lokal → Host	

⇒ hohe Belastung von Root/TLD  
 ⇒ Vorteil: Caching (meist TTL 900s)

## Header

TCP/UDP Header: Quell- und Zielport  
 IP Header: Quell- und Ziel IP

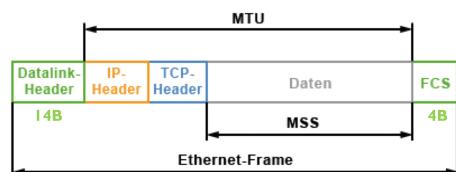
⇒ Pro TCP-Verbindung eigener Socket  $\delta$  (Auf Server wie Client Seite) Zunächst Three-Way Handshake mit Welcoming Socket des Servers

⇒ Prüfsummenfeld im UDP-Header: Header + Data + IP-Adressen → Einer Komplement => Check: 1er  
 ↪ 16-Bit → Übertrag dazu addieren

Source port	dest port
sequence number	
acknowledgement	
1	1
C E A I R L S W F receive window	
checksum	

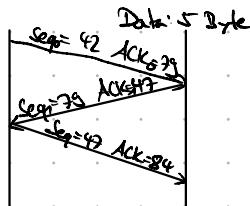
Aufbau:  
AS  
 01  
 11  
 10 + Daten

Schließen:  
 F  
 1  
 → ACK  
 (S. 31 Skript 4)

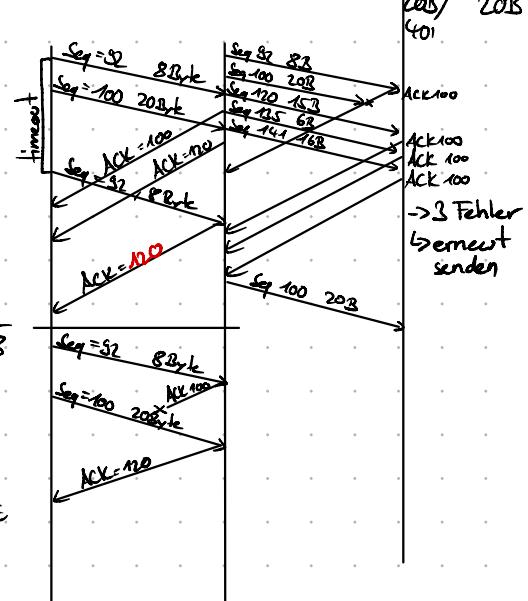


## SEQ / ACK

- $ACK_1 = SEQ_0 + DATA$
- $SEQ_1 = ACK_0$
- $\rightarrow$  kein ACK  $\Rightarrow$  erneut senden nach timeout



$$MSS = MTU - IP - TCP$$



## TCP Rundlaufzeit / Timer

$$RTT(t_i) = (1-\alpha) \cdot RTT_{\text{Est.}}(t_{i-1}) + \alpha \cdot RTT_{\text{Sample}}(t_i) \quad \alpha = \frac{1}{8}$$

$$\Delta RTT_{\text{Est.}}(t_i) = (1-\beta) \cdot \Delta RTT_{\text{Est.}}(t_{i-1}) + \beta \cdot |RTT_{\text{Sample}}(t_i) - RTT_{\text{Est.}}(t_i)| \quad \beta = \frac{1}{4}$$

$$RTO(t_i) = RTT_{\text{Est.}}(t_i) + 4 \cdot \Delta RTT_{\text{Est.}}(t_i)$$

$1s \leq RTO \leq 60s$  : Timeout = verdoppeln  $\Rightarrow$  Überlastkontrolle

## Rcv Window

- $\rightarrow$  Rückmeldung durch Empfänger (TCP-Header) Last Byte Send - LB ACK  $\leq$  Rcv Window
- $\rightarrow$  Rcv Window = 0  $\Rightarrow$  "TCP Zero Window Probe": 1 Byte Daten senden

$$\text{Durchsatz: } D_i = \frac{\text{Rcv Window}_i}{RTT_i + (n+1) \frac{RTT_i}{R}} \approx \frac{\text{Rcv Window}_i}{RTT_i}$$

## Überlastkontrolle

Ende zu Ende Congestion Control: Beobachtung Datenverlust / Verzögerungszeit z.B. bei TCP  
Netzwerkgestützt: Router Melden an Empfänger via CE-Bit in IP-Header  
 $\hookrightarrow$  Empfänger  $\rightarrow$  Sender über ECE-Bit in TCP Header

- Ablauf RTT  $\rightarrow$  viele Pakete verloren
- 3 ACK gleich  $\rightarrow$  ein Segment verloren

$$\text{Eff Window} = \min(\text{ConWin} \cdot \text{MSS}, \text{Rcv Window})$$

$$\text{Mittlere Datenrate } \langle D \rangle \approx \frac{\text{Eff Window}}{RTT} \approx \frac{\text{ConWin} \cdot \text{MSS}}{RTT}$$

Slowstart: Exponentiell beginnen mit 1...2...4... bis ssthresh, dann linear 8...9...10...

TCP-Tahoe: Fehler  $\rightarrow$  ssthresh =  $\frac{\text{ConWin}_{\text{max}}}{2}$ ; Con Win = 1 - SlowStart

TCP-Reno: Timeout  $\rightarrow$  Con Win = 1 - SlowStart  
3 ACK  $\rightarrow$  Con Win =  $\frac{\text{ConWin}_{\text{max}}}{2}$  - linear  $\rightarrow$  ssthresh =  $\frac{\text{ConWin}_{\text{max}}}{2}$

$$\text{mittlerer Durchsatz: } \langle D \rangle \approx 0,75 \cdot \frac{\text{ConWin}_{\text{max}} \cdot \text{MSS}}{RTT}$$

# Prager

RS45

	TS68-A (EU)	TS68-B (Intern.)
1	grün weiß	[ orange weiß
2	grün	orange
3	orange weiß	[ grün weiß
4	blau	[ blau
5	blau weiß	[ blau weiß
6	orange	grün
7	brown weiß	[ brown weiß
8	brown	brown

Transmit Data +	RX
Transmit Data -	RX
Receive Data +	TX
Leer (1) -	DC+
Leer (1) +	DC+
Receive Data -	TX
Leer (4) +	DC-
Leer (4) -	DC-

10Gb	2
1Gb	4
100Mb	15
16Mb	62
10Mb	100
4Mb	250

bis Kat. 6 RS45  
Kat. 7 GG45

## VLAN

↪ kleine Broadcasts + leichte Administration

→ Layer 3 Switche → Aufteilen in logische Switches (Ports zu VLAN zugeordnet)

→ Switching schneller als Routing

⇒ Johannes was wichtig?

## Spanning Tree (Layer 2)

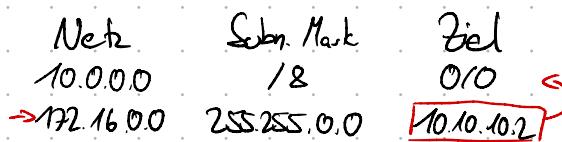
- geringste Priorität / geringst. Mac ist Root | (Priorität: 16 Sprunge (4096) v. 0-61.440; Zwischenwerk: VLW-ID)
- R (root), D (designated), A (alternating/blacking)
- → Vergleich zwei Switche: 1. geringste Abstand zu Root 2. geringste Prio 3. geringste Mac ⇒ Gewinner schaltet ①, verlierer ②/③
- BPDU (root Bridge, eigene ID, Kosten)
- Disabled → Blocking → Listening → Learning → Forwarding
- Lastenteilung durch unterschiedl. root-Bridge pro VLAN

## First Hop Redundancy

- Virtueller Router mit eigener IP+Mac → leitet Pakete dann an aktiven Router weiter → Ausfallsicherheit
- Virtueller Router IP als Gateway eintragen, Nicht! die IP der tatsächlichen Router

## Statisches Routing

- Gateway of last resort 0.0.0.0/0
- RIP: versenden der ges. Routingtabelle an Andere
- ↪ Merken sich Netze + benötigte Hops
- ↪ Hop-Count 0-15 → ab 16 nicht erreichbar ⇒ Verworfen



## Shortest Path Tree (Router) Dijkstra

- Tabelle: Knoten, besucht, Distanz, Vorgänger → Start bei 0
- ↪ alle Connections von Start aus eintragen → nächst niedr. Distanz Router aussuchen → Conn. zu umliegenden berechnen
- ↪ billiger? → Update der Tabelle → überprüfter Router besucht setzen

## OSPF

- Max Router pro Netz ca. 25 → Area Border Router (ABR) → Router kennen nur eigenes Netz + ABR für den Rest
- Alle Netze sind mit "Area 0" verbunden (Backbone)

## Subnetting

- 1. Adresse = Host ; letzte = Broadcast ⇒ Anzahl benötigter IPs = Anzahl Geräte + 2