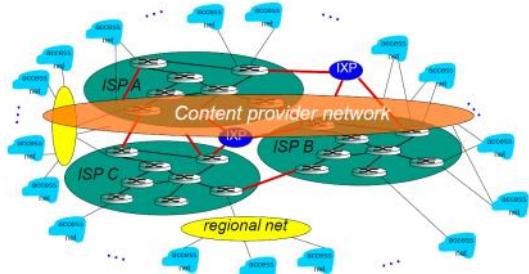


## ① Einführung

- Aufteilung in Rand (Clients und Server) und Kern (ISPs) des Internets
- zwischen ISPs (Anbietern) liegen IXPs (Internet Exchange Points)
- (oder auch nicht)
- Dazu können Content provider networks von großen Anbietern gelegt werden



## ② Anwendungsschicht

- oberste Schicht, enthält Anwendungsprotokolle
- kümmert sich nicht um Datentransport
- erhält Verzögerung von darunter liegenden Schichten

o Abhängig von

$$\text{I} \quad \text{Ausbreitungsverzögerung} \quad t_s = \frac{x}{v} \quad \text{mit} \quad \begin{array}{l} \text{Datenverkehrsgeschw. } v \approx \frac{2}{3} \text{ Lichtgeschw.} \\ \text{Datenrate des Mediums } r_d \text{ (z.B. in bit/s)} \end{array}$$

II Füllstände der Puffer

- o Übertragungsdauer = Sendezent + Ausbreitungsverzögerung  
ggf. weitere Wartezeit durch Puffewechseln und Verarbeitung (Feldgrößenbegrenzung.)
- Internet-Protokollstack:

Application
Transport
Netzwerk
Data Link
Physical

z.B.

SMTP, HTTP
TCP, UDP
IP
Ethernet, 802.11
Bits auf Medium

- Prozess: Programm in Anwendungsschicht auf Endsystem. Nachricht: zw. Prozessen auf versch. Endsystemen
- Socket-Interface: API vom OS für Netzwerk. Nachrichten werden zw. Sockets transferiert.

- o (Dc-)Multiplexen mit Portnummern. Auf Vermittlungsschicht werden Sockets durch IPs identifiziert (Endsystem-id), auf Transportschicht durch Port (Prozess-id)

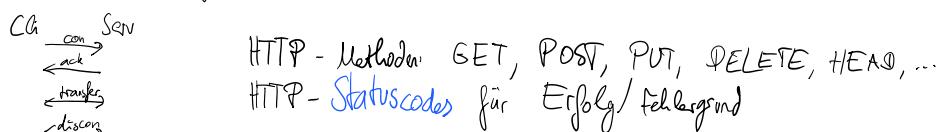
- Peer-to-peer: Clients kommunizieren direkt miteinander

## 2.1 WEB UND HTTP

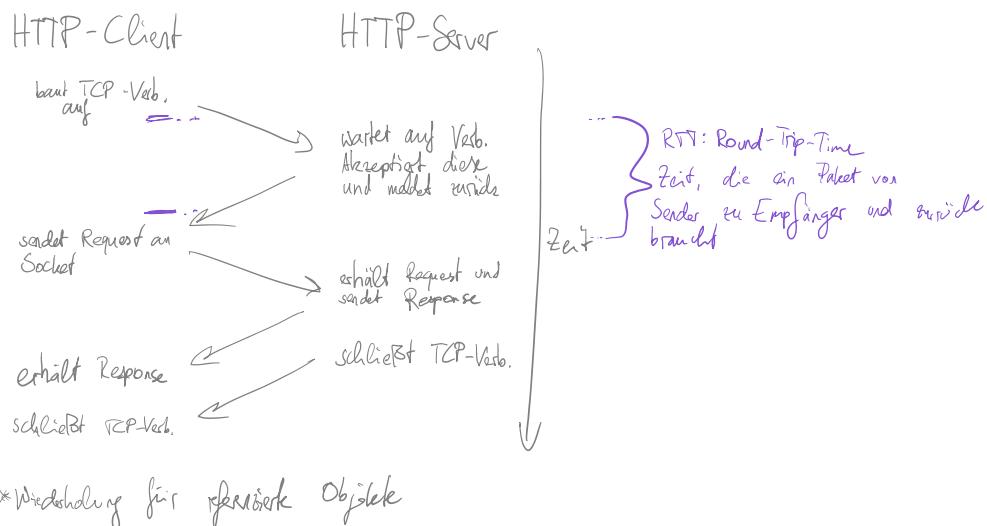
- URL



- HTTP: Transferprotokoll, ASCII, Client-/Server-modell, Nachrichtentypen: Request/Response, zustandslos, TCP für Kommunikation



- o non-persistent HTTP: pro TCP-Verbindung max. ein gerendertes Objekt
- o persistent HTTP: mehrere Objekte pro TCP-Verbindung
- o als Weg-Zeit-Diagramm:



- o HTTP-Antwortzeit:  $2 \cdot RTT + t_s$  mit  $t_s =$  Reklendezzeit  
(persistent: 1x RTT pro referenziertem Objekt)

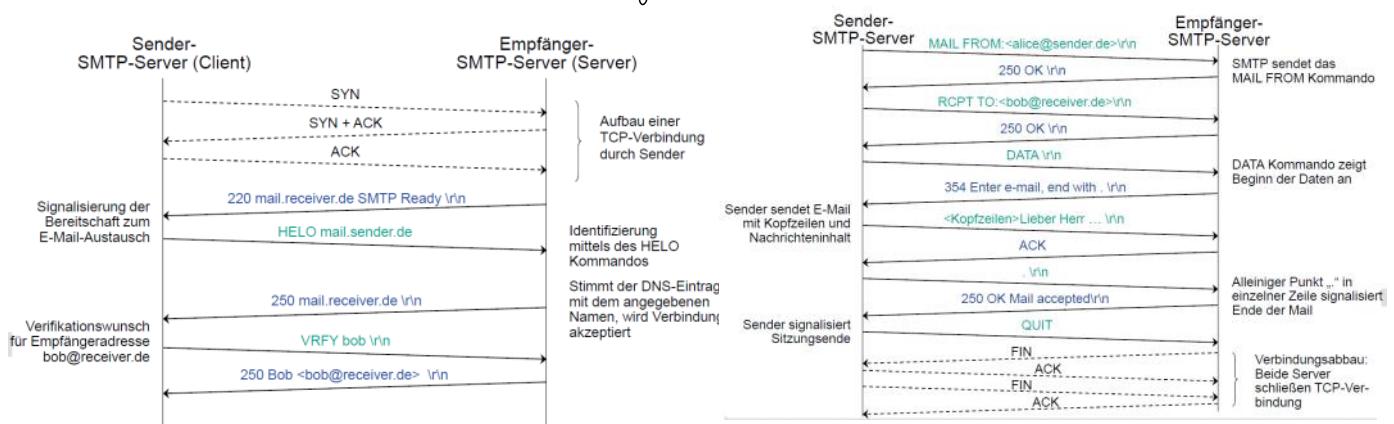
- Cookies: ermöglichen Nutzerunterschieden

- E-Mails

- o Grundlegende Komponenten (kommunizieren mit SMTP (Simple Mail Transfer Protocol))
  - User Agent (UA): Outlook, ...

## II Mail Server (Mail Transfer Agent MTA, Mail Delivery Agent MDA, Mailboxen)

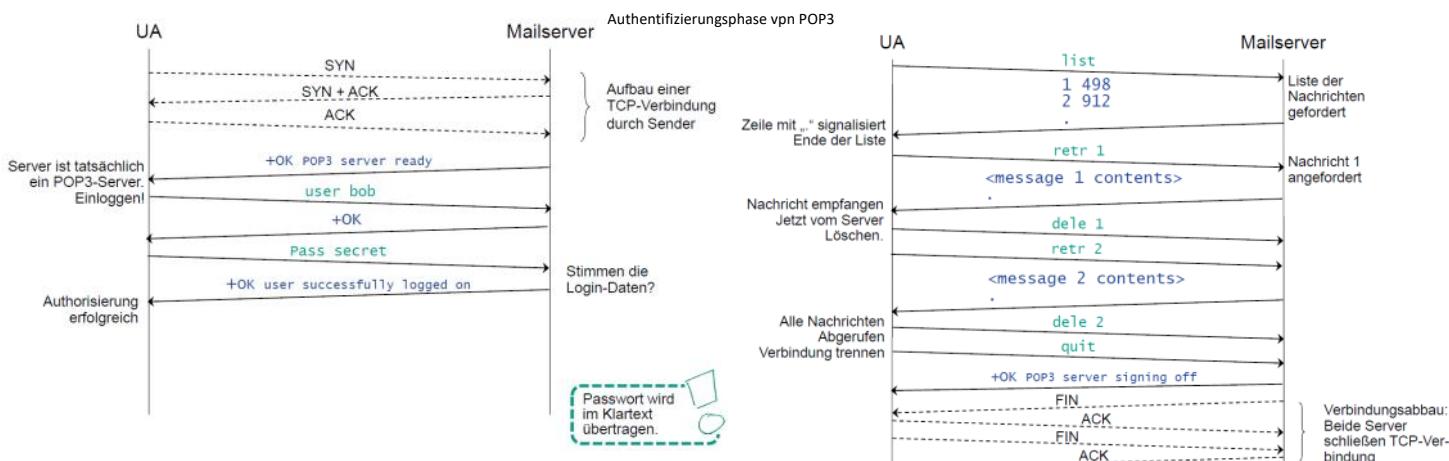
- SMTP: Handshake, Nachrichtübermittlung, Abschluss.



- MIME-type: Multipurpose Internet Mail Extensions

Codierungen innerhalb ASCII: =?charset?encoding?encoded-text?=  
z.B.=?US-ASCII?Q?...?=

- POP3: damit Client Nachrichten von Mailbox abholen kann



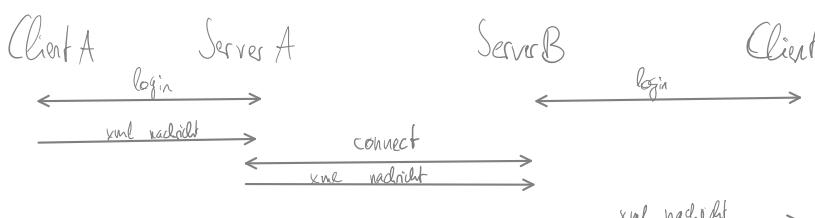
- IMAP: damit Client Nachrichten auf Mailbox verwalten kann

- Webmail: Benutzerfreundlicher

- Geforderte Schutzziele: Nachricht-Vertraulichkeit, Nachricht-Integrität, Sender-Authentizität, Empfänger-Authentizität

### - WhatsApp

- basiert auf XMPP. XMPP-Nachrichten sind in XML



XMP ist im Gegensatz zu WhatsApp zentralisiert

- DNS: Verwendung von Namen statt IP-Adressen
  - Verteiltes System für bessere Skalierung
  - kein Server kennt alle Namen → Abfrager, lokale DNS für ISPs, große Firmen, ... vs. autoritärer DNS
  - Client-/Servermodell über UDP
    - DNS-Aufgabe: → lokaler DNS → 1. eigene Datenbank falls autoritativ
    - 2. Cache
    - 3. anderer DNS

□ rekursive vs. iterative Aufgabe. Meist Client fragt lokalen DNS rekursiv, dieser arbeitet dann iterativ

- DNS-Hierarchie: Root-Server → TLD-Server → Autoritärer DNS → Lokaler DNS (wählt auch aus lokalem DNS)
- Wörter Domäne: Host-, Mail-Alias, Lastverteilung
- DNS-Einträge: Resource-Records (RR)

- HTTP Streaming für z.B. Videosreams

- Dynamic, Adaptive Streaming over HTTP (DASH)  
Videodatei in Chunks (je versch. Qualitäten möglich), Manifest mit Infos

## - Content Distribution

- Mega-Server: skaliert nicht
- Privates CDN / Third-Party CDN
  - Edge Deep: Viele kleine Cluster
  - Edge Home: Wenige große Cluster in wichtigen IXPs

für DASH benötigt Manifest vom Server herunter und Videochunks von Edge-Deep-CDN

Funktionsweise „DNS-Manipulation“ (2-114 / 2-115)

## ③ Transportschicht

- Transportprotokoll läuft auf Endsystemen
  - Sendende Seite segmentiert Nachricht, empfangende Seite reassembliert Segmente
    - Vermittlungsschicht
    - Anwendungsschicht
  - UDP: User Datagram Protocol (verbindungslos, unzuverlässig)
  - TCP: Transmission Control Protocol (verbindungsorientiert, zuverlässig)
- Zuverlässig: Übermittelte Daten sind vollständig, korrekt, in korrekter Reihenfolge, ohne Duplikate, keine Phantom-Pakete

- Segmente



- (De-)Multiplexer: Ende-zu-Ende auf Vermittlungsschicht, dann (de-)multiplexes auf Transportschicht mit Nutzer-zu-Nutzer (vgl. Nutzer: $\Leftrightarrow$  Prozess)

- o Adressierung auf Transportschicht: Ports (16bit, 0..65535)
- o Adressierung auf Vermittlungsschicht: IP (IPv4: 32bit, IPv6: 128bit)

IP + Port = internet-weite Anwendung identifizierung



- UDP: Einfach, geringer Transportkopf-Overhead (8byte: Ports, Länge, Prüfsumme)  
 (Unregelmäßiges Senden, Best Effort, kein Verbindungsauflauf, kein Verbindungsstand (skalierbar besser für Server))

- o Fehlertolerante aber Geschwindigkeitskritische Anwendungen: VoIP, Multimedia
- o Schnelle Reaktion, wenig/einzelne Pakete: DNS



- Prinzipien zuverlässiger Datenübertragung

- o Bitfehler: Einzelbitfehler oder Bündelfehler (aufeinanderfolgende Bits)

$$\text{II Bitfehlerrate} = \frac{\text{Summe gestörter Bits}}{\text{Summe übertragener Bits}}$$

Störung auch von Datenrate abhängig (Störung mit festen Zeitabständen)

- o Paketfehler: Paketverlust, Paketduplicierung, Phantompaket, Reihenfolgevertausch

- o Drogen: Redundanz

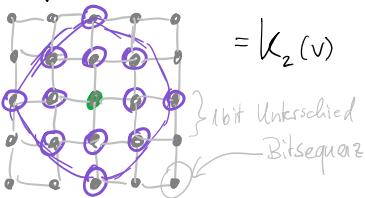
**Bitfehler**  
 II Fehlererkennung/Error Detecting Code EDC: Daten mit mehr Redundanz, z.B.  
 längere Code-Wörter für Daten

**Paketfehler**  
 II Fehlerkorrektur/Forward Error Correction FEC: falls Fehler erkannt, versuche Korrekture mit Redundanz

**Paketfehler**  
 D) Autoreaktivierung / Autoreaktivierung mit ARP: Empfänger teilt Sender

**FR** Wiederholungsaufforderung / Automatic repeat request ARQ: Empfänger teilt Sender korrigierte Daten mit

- o Paritätsbits: Nutzdaten  $x = x_1 x_2 \dots x_n$ , Paritätsbit  $p = x_1 + x_2 + \dots + x_n \bmod 2$ , Codewort:  $z = xp$
- o Hamming-Distanz  $d_{i,j} = \# \text{Bitpositionen, in denen sich Codewörter } c_i \text{ und } c_j \text{ unterscheiden}$   
 $d_{\min} = \min \{d(c_i, c_j) \mid c_i, c_j \in C, c_i \neq c_j\}$  für Code  $C$  aus Codewörtern  
 für  $d_{\min}$  können bis zu  $d_{\min}-1$  Bitfehler erkannt werden 3-43
- o Hamming-Kugel:  $K_r(v) = \{w \in C \mid d(v, w) \leq r\}$  für Codewort  $v \in C$  und Hamming-Distanz  $r$



- o  $r$ -Fehlererkennende Codes:  $\Leftrightarrow$  Codewort  $v$  liegt nicht in Hamming-Kugel eines anderen gültigen Code-Wortes.  $r$ -Fehlererkennend  $\Leftrightarrow d_{\min} > r$ . ( $d_{\min} \geq r+1$ )
- o  $r$ -Fehlerkorrigierende Codes:  $\Leftrightarrow$  Hammingkugel von  $v$  ist echt disjunkt zu Hamming-Kugeln anderer gültigen Code-Wörter.  $r$ -Fehlerkorrigierend  $\Leftrightarrow d_{\min} \geq 2r+1$  ( $d_{\min} \geq 2r+1$ )

- o  $(n,m)$  Hamming-Code:  $n-m$  bit Nutzdaten,  $m$  bit Parität

Bitpos	1	2	3	4	5	6	$\dots$	
P1	$x_1^+$	$x_2^+$	$x_4^+$	$x_5^+$	$x_7^+$	$x_9^+$	$x_{11}^+$	$\dots$
P2	$x_1^+$	$x_3^+ x_4^+$		$x_6^+ x_7^+$		$x_{10}^+ x_{11}^+$		$\dots$
P3		$x_2^+ x_3^+ x_4^+$			$x_8^+ x_9^+ x_{10}^+$		$\dots$	$\bmod 2$
P4							$\dots$	$\bmod 2$
P5							$\dots$	$\bmod 2$
⋮								

und, falls korrekt,  $P_k + \underset{P_k}{\underset{\text{Formel für}}{P_k}} \bmod 2 = P_k + P_k \bmod 2 = 0$

oder, falls inkorrekt, Bitfehler an Position  $P_k P_{k-1} \dots P_2 P_1$

- o Kontrollmatrix  $H$  eines linearen Codes  $V$ :  $w \in V \Leftrightarrow w \cdot H^T = (0 \dots 0)$

$w = H^T$  heißt Syndrom. 3-57, kann irgendwie nicht sein

– Internet-Prüfsumme

- o Aufaddieren aller übertragenen 16bit-Wörter, wird im Paket mit-übertragen (nicht Anzahl der Wörter, sondern Inhalt im Einen-Komplement addieren, d.h. jeweils Übertrag mit-aufaddieren)

– Fehlerkontrolle bei Paketfehlern

- o Erkennung

$\rightarrow S \dots R \dots 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1$

- o Erkennung

II Sequenznummern (Sequence numbers): Pakete werden durchnummierat

II Zeitgeber (Timer): Nach zeitlicher Obergrenze ohne Quittung nimmt Sender Paketverlust an und startet Sendewiederholung

- o Behebung

II Quittungen (Acknowledgment): Empfänger bestätigt Pakethalt bei Sender:

pos. Quittung (ACK) vs neg. Quittung (NACK, Daten nicht korrekt erhalten)

selektive Quittung (SACK): Quittung bezieht sich auf einzelnes Paket

kumulative Quittung: Quittung bezieht sich auf mehrere Pakete

II Sendewiederholung (Retransmission) bzw. Automatic Repeat Request: ARQ

- Stop-and-Wait: Sender schickt immer nur einzelne Pakete und wartet immer auf Quittungen  
Dabei können durch Timeouts Pakete doppelt gesendet werden, dagegen Sequenznummer (hier reicht dafür 1bit)

- Leistungsbewertung

- o Durchsatz / Datarate in bit/s

- o Verzögerung / Latenz

II Verarbeitungsverzögerung: z.B. Pfifsummenberechnung  $t_v$

II Warteschlangenverzögerung: von Netzwerkauslastung abhängig

II Senderseite: abhängig von Durchsatz  $t_s = \frac{\text{Paketlänge}}{\text{Datarate}}$

II Ausbreitungsverzögerung:  $t_a = \frac{\text{Mediumlänge}}{\text{Ausbreitungsgeschw.}}$

$$t_{\text{Ges}} = t_s^{\text{Daten}} + t_a + t_v^{\text{Daten}} + t_s^{\text{Quittungen}} + t_a + t_v^{\text{Quittungen}} \quad \left. \right\} \text{für Stop-and-Wait}$$

$$\text{vereinfacht: } t_{\text{Ges}} = t_s^{\text{Daten}} + 2t_a$$

- o Auslastung des Mediums:  $U = \frac{1}{1+2\frac{t_a}{t_s}} = \frac{1}{1+2\alpha} \quad \text{für } \alpha = \frac{t_a}{t_s} = \frac{\frac{M}{v}}{\frac{X}{r}} = \frac{M \cdot r}{X}$

Auslastung mit Übertragungsfällen:  $n=1$  konsekutive Sendewiederholungen,  $U = \frac{t_s}{n(t_s + 2t_a)}$  } Stop-and-Wait

Auslastung mit Fehlerwahrsch.  $p$ :  $U = \frac{1-p}{1+2\alpha} = \frac{1-p}{1+2\frac{t_a}{t_s}}$

- Go-Back-N ARQ: Sender sendet mehrere Pakete, bevor Quittung verlangt wird, max # an nicht-quittierten Paketen, bei Fehler werden alle Pakete seit letzter gültiger Quittung verworfen und Sender wiederholt unquittierte Pakete

$$\text{Nachrichtenanzahl: } n = \begin{cases} 1 & \text{für } U \geq 1+2\alpha \\ \dots & \dots \end{cases} \quad (\text{Sender kann ohne Pause senden})$$

werk und Sender wiederholt ungültige Pakete

$$\text{Leistungsbewertung: } U = \begin{cases} 1 & \text{für } W \geq 1+2a \\ \frac{W}{1+2a} & \text{für } W < 1+2a \end{cases}$$

(Sender kann ohne Pause senden)  
(Sender kann nach Fensterbrauch nicht weiter senden)

mit  $W = \text{Fenster}$ , begrenzt max. # gesendete, nicht gültige Pakete (weil Puffer)

Leistungsbewertung mit Überlagerungsfaktor: (mit Paketfehlerrate  $p$ )

$$U = \begin{cases} \frac{1-p}{1+2ap} & \text{für } W \geq 1+2a \\ \frac{W(1-p)}{(1+2a)(1-p + Wp)} & \text{für } W < 1+2a \end{cases}$$

- Selective Repeat ARQ: höhere Auslastung als Stop-and-Wait, wichtiges Datenkomma als Go-Back N

Sender wie Go-Back-N, Empfänger quittiert selektiv (nur gültige Pakete wiederholen)

alternativ: Selective Reject (gültige Pakete werden negativ quittiert und direkt neu gesendet statt erst nach Timeout)

$$\text{Leistungsbewertung: } U = \begin{cases} 1-p & \text{für } W \geq 1+2a \\ \frac{W(1-p)}{1+2a} & \text{für } W < 1+2a \end{cases}$$

- Vorwärtsfehlerkorrektur: für Paket  $p$ : wird redundantes  $\oplus p$  geschickt. Bei einem falschen von vier Paket kann Empfänger Daten wiederherstellen

≡

- Flusskontrolle: Sender kann Empfänger überlastet werden

o Closed Loop: Sender adaptiert seinen Datstrom

□ Half-and-Halte: Empfänger kann „Halt“ und „Weiter“ senden. Nur bei Vollduplex.

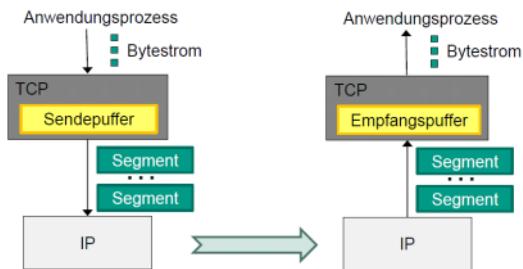
□ Stop-and-Wait: Zurückhalten des Quittung, um Warten zu erwürgen. Verzögerungsgefahr mit Paketfehlern

□ Kreditbasierte Flusskontrolle („Sliding Window“): Sender kann bis zu  $n$  Pakete senden ohne Quittung, wobei  $n = \text{Puffergroße des Empfängers}$  („Sackcredit“)

▷ Sender-Invariante:  $\text{Last Frame Send} - \text{Last ACK received} + 1 \leq \text{Send Window Size}$

▷ Empfänger-Invariante:  $\text{Last Frame Acceptable} - \text{Next Frame Expected} + 1 \leq \text{Receiver Window Size}$   
Sequenznummern  
Paketmengen

- Transmission Control Protocol (TCP)



TCP erhält Bytestrom von Anwendung und übergibt TCP-Segmente an IP

- TCP sammelt Bytes aus Strom. Wann gesammelte Daten verpacken und schicken?

□ MSS: Maximum Segment Size

□ Push: Sofort schicken wenn Daten kommen

□ Zeitgeber

- Eigenschaften

□ Phasen: Verbindungsauflauf, Datentransfer, Verbindungsabbau

□ Fehlertypen: Sequenznummern, Prüfsumme, Quittierung, ggf. Sendewiederholung

□ Flusskontrolle: Empfänger nicht überlasten (Ziel)  
Empfänger reserviert Puffer pro Verbindung.

$$\text{Invariante: } \frac{\text{Last Byte Received}}{\text{Receive Buffer}} - \frac{\text{Last Byte Read}}{\text{Receive Buffer}} \leq \frac{\text{Receive Buffer}}{\text{Receive Buffer}}$$

$$\text{Empfangsfenster: } \frac{\text{Receive Window}}{\text{Receive Buffer}} = \frac{\text{Receive Buffer}}{\text{Receive Buffer}} - \left( \frac{\text{Last Byte Received}}{\text{Receive Buffer}} - \frac{\text{Last Byte Read}}{\text{Receive Buffer}} \right)$$

$$\text{Invariante: } \frac{\text{Last Byte Sent}}{\text{Receive Window}} - \frac{\text{Last Byte Acked}}{\text{Receive Window}} \leq \frac{\text{Receive Window}}{\text{Receive Window}}$$

□ Staukontrolle: Netze nicht überlasten (Ziel)

$$\triangleright \text{Staukontrollfenster (CWnd): } \frac{\text{Last Byte Sent}}{\text{CWnd}} - \frac{\text{Last Byte Acked}}{\text{CWnd}} \leq \min \{ \text{CWnd}, \text{RcvWindow} \}$$

▷ Schwellwert (SSTresh)

nur Zeitgeber, Auslieferende Quittung  $\Rightarrow$  verm. Stau  
Dann: Reduktion von CWnd, danach langsam wieder erhöhen

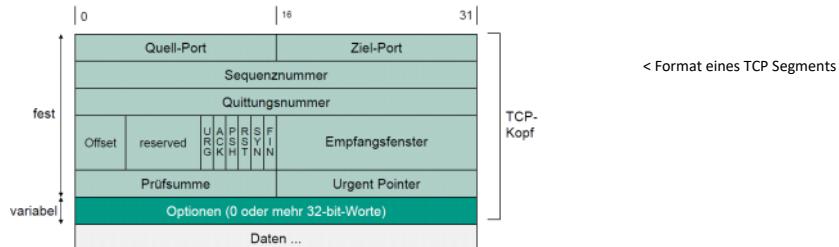
Aufang:  $CWnd := \text{Max Segment Size}$

$\triangleright CWnd \leq SSTresh$ , Quittungen rechtzeitig empfangen  
Exponentielles Skalieren:  $CWnd += 1$

$\triangleright CWnd > SSTresh$ , Quittungen rechtzeitig empfangen  
Lineares Skalieren:  $CWnd += \frac{1}{CWnd}$

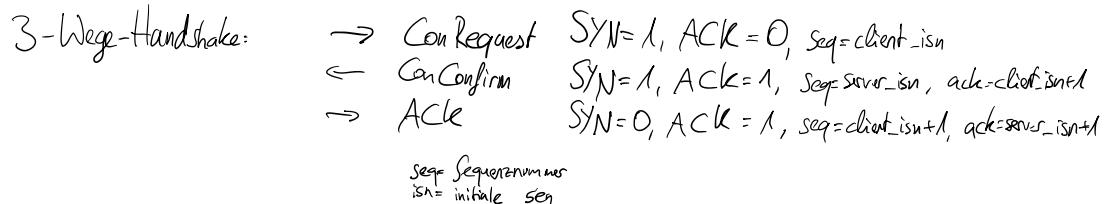
▷ Quittung nicht empfangen (Timeout)  
 $SSThresh := \max \{ Flightsize / 2, 2 \cdot MSS \}$ , CWnd := MSS  
 Flightsize = gesendete unquittierte Daten. MSS = Max Segment Size

- Sequenznummern pro Byte (holt Paketen) rückläufige initiale Sequenznummer
- Quittungen: positiv, kumulativ, enthält Sequenznummer des nächsten zu erwartenden Bytes



- Quell-Port und Ziel-Port
  - Identifizieren Endpunkte der Verbindung
- Sequenznummer
  - Gemessen in Byte (nicht pro Segment)
- Quittung
  - Die nächste vom Empfänger erwartete Sequenznummer
- Offset
  - Anzahl der 32-Bit-Wörter im TCP-Kopf
- URG
  - Wird auf 1 gesetzt, falls der Urgent Pointer verwendet wird
    - ... in der Regel nicht benutzt
- SYN
  - Wird beim Verbindungsaufbau verwendet, um Connection Request (TConReq) bzw. Connection Confirmation (TConCnf) anzuzeigen
- ACK
  - Unterscheidet bei gesetztem SYN-Bit eine TConReq-PDU von einer TConCnf-PDU
  - Signalisiert die Gültigkeit des Quittungs-Feldes
- FIN
  - Gibt an, dass der Sender keine Daten mehr senden möchte
- RST
  - Wird benutzt, um eine Verbindung zurückzusetzen
- PSH
  - Signalliert, dass übergebene Daten sofort weitergeleitet werden sollen
  - Gilt sowohl für den Sender als auch für den Empfänger
  - Wird in der Regel nicht benutzt
- Empfangsfenster
  - Dient zur Flusskontrolle
- Prüfsumme
  - Enthalt Prüfsumme über TCP-Kopf, Daten und Pseudoheader (wie bei UDP)
- Urgent-Zeiger
  - Relativer Zeiger auf wichtige Daten
- Das Optionen-Feld
  - kann Optionen variabler Länge aufnehmen ( $n * 32$  Bit)

- Verbindungsverwaltung: Handshake am Anfang

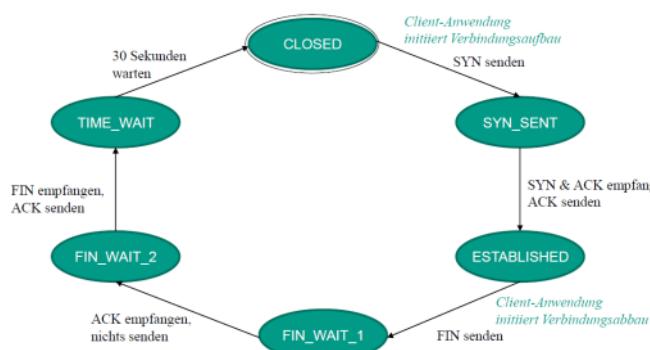


▷ festlegen initiale Sequenznummern, Größe von Puffer bekanntgeben, ..

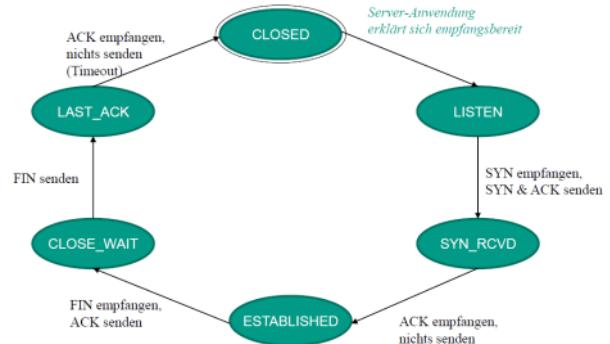
Verbindungsabbau:  $\xrightarrow{\text{TDsReq}}$ ,  $\xleftarrow{\text{TDisCf}}$  oder

4-Wege-Handshake: einmal Verbindungsabbau pro Richtung

Vereinfachter Zustandsautomat: TCP-Client



Vereinfachter Zustandsautomat: TCP-Server



(4)

## Vermittlungsschicht

- Vermittlungstechniken

- Leitungsvermittlung: <sup>Beispiel: Telefonnetz</sup> Verbindungsorientierte Kommunikation, Verbindung hält konstanten Kanal mit konstanter Bandbreite (ggf. ungenutzte Ressourcen). Keine Übertragungsverzögerung durch Pufferfüllstandschwankungen, Reihenfolge bleibt erhalten.

Vermittlung in Zwischensystemen braucht keine Verbindungsinformationen, dafür Zustandshaltung.

### ☒ Starres Multiplexen

▷ Frequenzmultiplex

▷ Zeitmultiplex

- Speichervermittlung → Paketvermittlung: Weiterleitung durch Kontrollinformationen in Paketen. Mögliche Reihenfolgefalsche wegen wechselnden Wegen. Zwischensysteme haben Puffer, dadurch Paketverlust möglich.

### ☒ Statistisches Multiplexen

#### □ Varianten

▷ Datagramme - Verbindungslos

Paket (Datagramm) als isolierte Einheiten. Keine Handshake, unterschiedliche Wege, keine Verbindungslos in Zwischensystemen.

Weiterleitungstabellen in Zwischensystemen (mappiert Endsystem  $\rightarrow$  ausgehende Leitung)

▷ Virtuelle Verbindung - Verbindungsorientiert

Fester Übertragungsweg zw. zwei Endsystemen, daher Reihenfolgefrei. Paketvermittlung durch Kennung, Zieladresse nur beim Verbindungsauflauf notwendig.

Verbindungsauflauf: Festlegung d. Kennungen auf Zwischensystemen  
Dateübertragung

Verbindungsabbau: Vermittlungsinformationen auf Zwischensystemen werden gelöscht

- Speichervermittlung → Nachrichtenvermittlung: Nachricht soll als ganzer verschickt werden. zw. Zwischensystemen wird Paketvermittlung verwendet, d.h. in Zwischensystemen: Reassemblierung und Segmentation der Nachrichten

Anwendung: Delay-Tolerant Networks

#### - Prinzipien der Vermittlungsschicht

- Protokolle sind in allen End- und Zwischensystemen

o Aufgaben:

□ Weiterleitung: Datenebene: Leite Daten an Routereingang an korrekten Ausgang weiter.

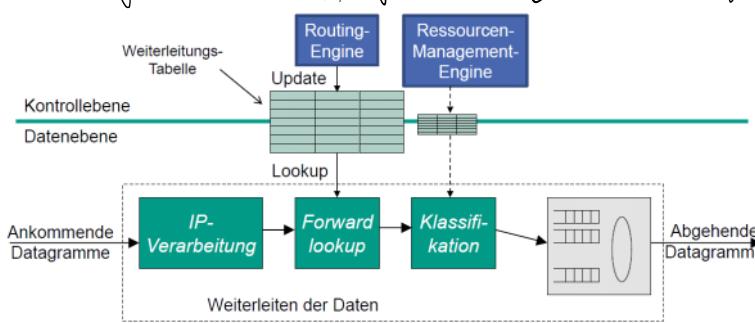
▷ Datenebene: Funktionen lokal im Router

□ Wegewahl: Kontrollebene: Routingalgorithmus, Routingprotokoll

▷ Kontrollebene: Ganzes Netz, bestimmt komplett Route eines Datagramms.  
Traditionaler Routingalgorithmus in Routern, Software-defined networking in (entfernten) logisch zentralen Servern.

- IP: Einiges (für Interoperabilität, wenige unterschiedliche Interfaces) und einfaches (für viele verbundene Netze, wenige Fehler) Protokoll in Verbindungsschicht

Verbindungslos, unzuverlässig, für Weiterleitung von IP-Datagrammen.



Datenpakete werden potentiell in jedem Zwischensystem fragmentiert, nur im Endsystem reassembliert (Anpassen an maximale Paketlänge versch. Netze, Maximum Transfer Unit, MTU).  
Vorgehensweise Flagfelder des IP-Kopfes

Bit 0: für 0 reserviert

Bit 1: 0 = darf fragmentiert werden  
1 = darf nicht fragmentiert werden

Bit 2: 0 = letztes Fragment  
1 = es folgen weitere Fragmente

Fragment-Offset: Stelle, an der empfangenes Fragment eingesetzt werden muss (Basisheit: 8 Bytes)

- o Weiterleitung von IP-Datagrammen: direkt an Endsystem, falls in derselben Netzwerk, sonst an Default-Routes. Grundlage: Weiterleitungstabelle ?
- o Empfangen eines IP-Datagramms: Korrekte Kopflänge? IP-Versionsnummer? Korrekte Datagrammlänge? Prüfsumme? Lebensdauer? Protokoll-Identifikation? Quell- oder Zieladresse?  
Falls Fehler: ICMP (Internet Control Message Protocol)
- o IP-Adressen: Subnetzteil (High bits) und Endsystemteil (Low Bits)
  - Format: a.b.c.d/x, mit x = # Bits des Subnetzteils

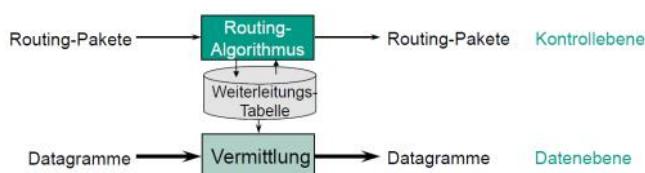
- ▷ Dynamische Allokation via DHCP
- ▷ Provider erhält Adressblöcke von ICANN, Netz bekommt Subnetz-Teil von Provider
- ▷ DHCP: Endgeräte fordern IP-Adresse bei Verbindungsauflauf, Zeitliche Begrenzung der vorgebenen Adressen.  
Funktionsweise:  $\rightarrow$  DHCP discover,  $\leftarrow$  DHCP offer,  $\rightarrow$  DHCP request,  $\leftarrow$  DHCP ack  
es ist Nachricht als Broadcast an alle im Netz
- Internet Control Message Protocol (ICMP): Austausch von schwerwiegenden Problemen (nicht einzelne Datagramm-Verluste)
  - ▷ Statusabfragen
    - ▷ Echo und Echoantwort: Überprüfung der Aktivität von Komm.-Systemen
    - ▷ Zeitstempel und Zeitstempelantwort: Bestimmung von Umwegzeit (Round Trip Time)
  - ▷ Format: Typ, Code (=genaue Beschreibung), Checksum (des ICMP Datagramms), Info (abhängig von Typ)
  - ▷ Übertragung: Im Datenteil von IP Datagrammen mit Protocol-Feld=1

## - Routing

- Modelliere Netz als Graph: Router = Knoten, Übertragungssabschnitt = Kante, Verzögerung (G.Ä) = Kantenkosten  
Nutz Routing-Algorithmen für kürzesten Pfad
- Routing - Verfahren
  - Dynamik: Nicht adaptiv: Routen ändern sich selten (seltener als Verkehrsänderungen)
  - adaptiv: Routen ändern sich abhängig von Verkehr / Netktopologie  
teils da Systeme veraltete Netzinfos haben

Kontroll- vs. Datenebene: Datenebene: Datentransfer auf Vermittlungsschicht

Kontrollebene: Steuert Datenvermittlung



## Routing-Vorfahren

- **Statistisches Routing:** Weinkettentabelle mit gewichteten ausgehenden Systemen, dann Zufallsentscheidung mit Gewichtung
- **Zentralisiertes Routing:** adaptiv. zentrales **Routing Control Center (RCC)**  
jedes System berichtet periodisch an RCC, RCC berechnet dann kürzeste Wege
  - + RCC trifft perfekte Entscheidungen + Systeme brauchen keine Routingbedingungen
  - langsam für große Netze - Fehleranfälligkeit
  - Inkonsistenzen durch RCC-Distanz - hohe Belastung des RCC's
- **Isoliertes Routing:** jedes System entscheidet komplett selbst
  - ▷ Flut: einfach, nicht adaptiv. Eingehendes Datagramm wird auf allen anderen Ausgängen weitergeschickt. Duplikate entfernen durch Sequenznummer. Hop-Zähler := max. Weglänge, Hop-- pro Route, verweigerte Datagramm bei Hop=0.
    - Selektives Fluten: nicht alle Ausgänge nutzen
    - Random Walk: ein zufälliger Ausgang
  - ▷ Hot-Potato: Wähle Ausgang mit kleinstem Waitesch-länge.
- **Verteiltes adaptives Routing:** jedes System hat Routing-Tabelle, Routing-Infos werden zwischen Nachbarn ausgetauscht. Routing-Tabelle enthält Schätzungen Austausch periodisch oder bei signifikanten Änderungen.

## Routing-Algorithmen:

- **Distanz-Vektor-Algorithmen:** Verwendet Distanz als Metrik. Router kennen Distanzen zu allen anderen Routern aus ausgetauschten Nachbarschaften. Problem: nutzt Distanz, nicht Dauer
- **Link-State-Algorithmen:** Verst. mögl. Metriken. Jeder Router kennt ganze Netztopologie. Besser für große Netze.
- **Distanz-Vektor-Routing:** Verteilt, iterativ  
Distanz-Vektor-Tabelle:
 

	Machbar 1	Machbar 2	...
1	1	1	...
2	1	1	...
...	...	...	...

X will Daten an Y über direkten Nachbarn Z weiterleiten ( $X \rightarrow Z \rightarrow \dots \rightarrow Y$ ):

$$D^X(Y, Z) = c(X, Z) + \min_w \{ D^Z(Y, w) \}$$

Restring:

0	1	1	1	1	1	1	1	1	1
---	---	---	---	---	---	---	---	---	---

Distanz-Vektor-Algorithmus: Bellman-Ford, siehe 4-89

+/-: einfacher zu implementieren

o Link-State-Routing: Am Anfang kennen Systeme nur direkte Nachbarn

Link-State-Broadcast: Teile Identität und Kosten durch z.B. Fluten. Topologie wird durch Broadcasts anderer Systeme erkannt.

Kürzeste Wege-Berechnung: Dijkistra, siehe 4-102

+/-: konvergiert schnell, robust

- Software-Defined Networking (SDN): Eigenschaften:

E1: Separation von Kontroll- und Datenbahn

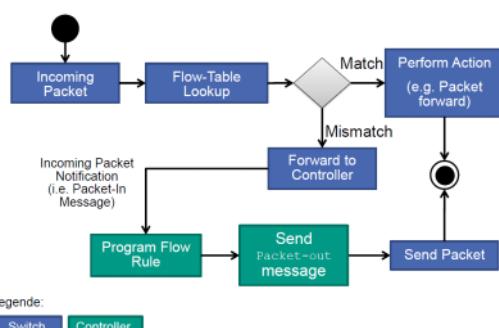
E2: Flow-basierte Weiterleitung von Paketen

E3: Logik ausgelagert an externen Controller

E4: Programmierbarkeit des Netzwerks

Traditionelles IP-Routing	vs.	SDN
<b>Pro</b>		<b>Pro</b>
+ Aufwachbarkeit	- Proprietär	+ Logisch zentrale Sicht
+ schnelle Reaktion	- Unflexibel	+ Mehr Software-Funkt.
+ bewährt	- Teuer	+ Trennung der Ebenen
		- Shalierbarkeit
		- Fehleranfälliger
		- langsame Controllerkomm.

Reactive Flow-Programmierung

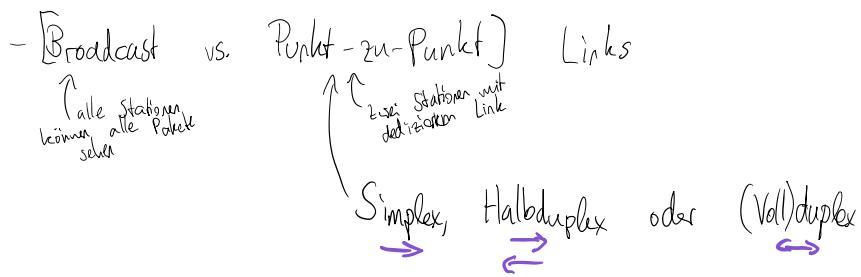


## ⑤ Sicherungsschicht

In jedem Knoten ( $\in \{\text{Endsystem, Router, Switch}\}$ ) implementiert, meist auf Netzadapter via PCI-Bus  
Sender kapselt Datagramm in Rahmen mit Metadaten, Empfänger überprüft & extrahiert für Vermittlungsschicht

- Rahmen: Pakete auf Transportschicht, darin sind IP-Datagramme

- Aufgaben der Sicherungsschicht: Datenstromstrukturierung (Framing), Medienzugangskontrolle, Adressierung durch MAC-Adr., entl. Fehlerrückkopplung



- Fehlererkennung

Ahnlich zu Vermittlungsschicht.

Sender fügt Rahmen eine Sicherungssequenz (Frame Check Sequence) dazu

- o CRC: Cyclic Redundancy Check: Symbole eines Codeworts werden als Polynomkoeffizienten interpretiert:

$$\begin{array}{r} 0101 \\ | \quad \diagdown \\ 0x^3 + 1 \cdot x^2 + 0 \cdot x + 1 = x^2 + 1 \end{array}$$

Verwende gleiches Generatopolynom  $G(x) = 1 \cdot \{0, 1\}^{r-1}$  für Empf. u. Sender

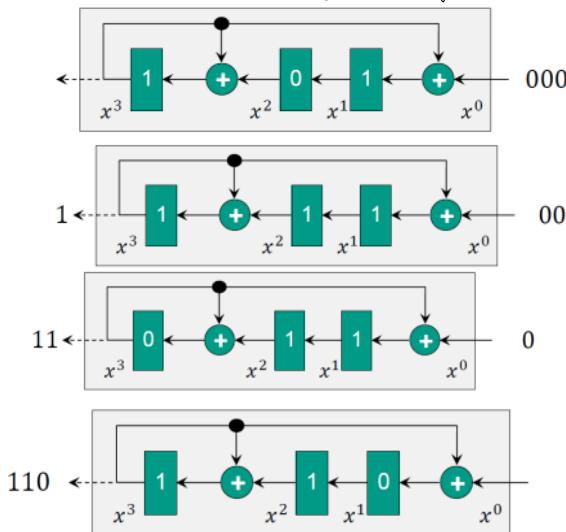
Prüfsummenberechnung: Rest von  $(x^r M(x)) / G(x)$ .

Prüfsumme hinter Ende des Rahmens  
mit r = Generatopolynomgrad = Generatopolynomlänge - 1

Divisionsrechnung mittels Bitweisen XOR (aber mit Übertrag)

Erkennt alle Einzelbitfehler, fast alle Doppelbitfehler, jede ungerade Anzahl an Bitfehlern, alle Bursts mit bis zu r Bitfehlern

Hardwareimplementierung: Rücksiegelkoppte Schieberegister



- Multiplexverfahren

## o Multidimensionen: Raum (r), Zeit (t), Frequenz (f), Code (c)

□ Raum: gedachte Antennen auf aufgeteilter Raum, dedizierte Ressourcen etc.

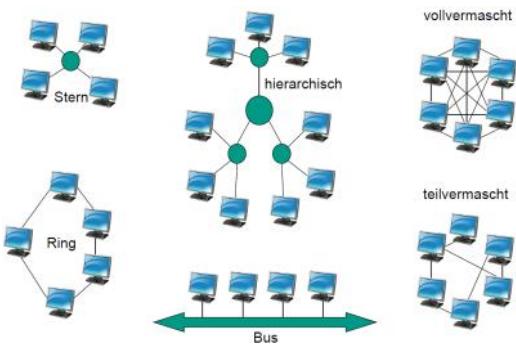
□ Code: Signal wird von Sender mit für ihn eindeutiger Pseudozufallszahl verknüpft, Empfänger restauriert daraus. Viele Möglichkeiten und großer Coderaum

Im folgenden Zeitmultiplex

### - Medienzugriffsvorfahren

Unterschiedliche Medien (Kabel, "Äste", ...) entweder fest Medienzuordnung oder konkurrierende Nutzung (kontrollierter Zugriff zentral oder dezentral, oder zufälliger Zugriff)

## o Netztopologien



## o Zufälliger Zugriff

□ Aloha, Verbesserung: Slotted-Aloha

für zufällige unabhängige und seltene Sendungen. Letzteres erfordert Sync der Stationen

□ CSMA: Listen before Talk, vorher prüfen ob Medium frei

CSMA/CD: Listen while Talk, während Senden Kollisionserkennung und ggf. Abbrechen

Bei Ethernet Mindestlänge für Rahmen für Kollisionsdetektion

## o Kontrollierter Zugriff:

□ Zentraler Knoten (Polling): Sequentielle Senderechtezuweisung

□ Dezentral: Weitergabe des Senderechts (Token Passing)

Tokenring: Systeme sind Punkt-zu-Punkt zu Ring verbunden. Daten gehen durch den Ring bis zum Sender, dieser nimmt Daten vom Ring raus und gibt Token weiter

Realisierung: Stern mit Ringverteiler in der Mitte, ausfallende Systeme werden mit Relais überspannt, damit Ring geschlossen bleibt

Protokoll: Token Holding Time (um mehrere Rahmen zu senden), Rückmeldung an Sender vs. piggy-backing, Prioritätsmechanismus, Monitor (ausgewähltes Endsystem überwacht Token-Ring)

Wenn niemand Senden will, kreist Token bis Sendewunsch

Tokenbus: Ethernet + Tokenring. Physische Busverbindung mit logischem Tokenring. Garantierte Antwortzeiten. Token braucht exakte Zieladresse, nur Nachfolgestation darf Token annehmen

## - Lokale Netze

- Mac-Adressen: Adressen der Sicherungsschicht  
(Vermittlungsschicht: IP, Transportschicht: Ports)  
Theoretisch weltweit eindeutig: von Hersteller durchnummiert, aber konfigurierbar
- ARP - Address Resolution Protocol: ermittelt MAC zu bekannte IP  
In jedem System ist ein ARP-Cache mit Zeila.: (IP, MAC, Max. Lebenszeit)  
Adressauflösung mittels Broadcasting lokaler Netz  
(Broadcast: ARP-Request, ↪ Reply mit MAC, Cacheeintrag) meist 20min  
Falls separate Subnetze: Sender sendet ARP-Request und Datagramm (mit MAC vom Router) an Router, dieser leitet weiter

### ■ Kopf des ARP-Pakets

Netzwerk-Typ		Protokoll-Typ
HLEN	PLEN	Betriebs-Code
MAC-Adresse des Senders		
MAC-Adresse des Senders	IP-Adresse des Senders	
IP-Adresse des Senders	MAC-Adresse des Empfängers	
MAC-Adresse des Empfängers		
IP-Adresse des Empfängers		

↔ 32 Bit

Netzwerk-Typ:  
1 = Ethernet;  
6 = IEEE 802.2  
Protokoll-Typ:  
2048 = IP  
HLEN:  
2 = 16-Bit MAC-Adresse  
6 = 48-Bit MAC-Adresse  
PLEN:  
4 = 32-Bit IP-Adresse  
Betriebs-Code:  
1 = Request;  
2 = Reply

HLEN: Header Address Length  
PLEN: Protocol Address Length

### ○ Ethernet

□ Medienzuweisung: Zeitmultiplex, variabel, zufälliger Zugriff, CSMA/CD

□ Netztopologie: Stern, führer Bus

□ Drahtgebunden, z.B. Koaxial

□ Varianten:

▷ <Datenrate in Mbit/s><Baseband oder Broadband><Medium>

▷ 10Base5: Segmentkopplung über Repeater, max 100 Systeme pro Segment, Bus, robust, teuer, unflexibel  
10mm Koaxial

▷ 10Base2: 185meter-Segmente, 30 Systeme pro Segment, Bus, Koaxial  
5mm Koaxial

Ethernet			
	10Base5	10Base2	10Base-T
Medium	Koaxialkabel		Twisted Pair
Kodierung		Manchester	
Topologie	Bus		Stern

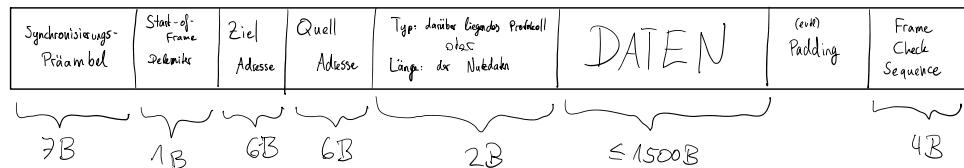
  

Fast Ethernet				
	100Base-T	100Base-T4	100Base-Tx	100Base-Fx
Medium	Twisted Pair		Glasfaser	
Kodierung	Manchester	8B/6T NRZ	4B/5B NRZI & MLT-3	4B/5B NRZI
Topologie		Stern		

Gigabit Ethernet and beyond				
	1000Base-SX	1000Base-T	10GBase-SR	10GBase-T
Medium	Glasfaser	Twisted Pair	Glasfaser	Twisted Pair
Kodierung	8B/10B NRZ	PAM-5 & Trellis	66B/68B	PAM-16 & DSQ128
Topologie		Stern		

□ Ethernet Rahmen:



□ Exponentieller Backoff: Bei Kollision zufällige Wartezeit  $\in \{0, 1, 2, \dots, 2^{i-1}\}$  bitslalte bei der  $i$ -ten Kollision (ab  $i=16$  wird Systemfehler angenommen)  
Zeitschlitz Dauer = minimale Rahmenlänge

□ Ethernet-Switch: Zusammenkopplung mehrerer Ethernet-Netze auf Transportschicht  
Trennung von Inter- und Intranetverkehr  $\Rightarrow$  höhere Netzkapazität



Anschluss der Systeme Punkt-zu-Punkt Halfduplex (mit CSMA/CD)  
oder Voll duplex (keine Kollisionsbehandlung notwendig)

Aufgaben:

- ▷ Etablierung einer Netztopologie ohne Schleifen (<sup>Spanning</sup><sub>tree</sub> <sup>Algorithmus</sup>)
- ▷ Etablierung von Umgang zw. Endsystemen (<sup>selbstlernende</sup><sub>Switches</sub>):  
Switch empfängt Rahmen und flutet weiter, mustert sich dann  
(MAC, Interface, Lebenszeit)

□ Kollisionsdomäne: Nutzzeitraum, der von Kollision betroffen ist. Bei Bus-Topologie:  
ganzes Broadcastmedium (Segment)

Netz wird in versch. Kollisionsdomänen aufgeteilt durch

▷ Brücke bei Bus-Topologien

▷ Switches bei Stern-Topologien

▷ Switch vs. Router  
 auf Sichtungsschicht      auf Vermittlungsschicht  
 Store-and-forward

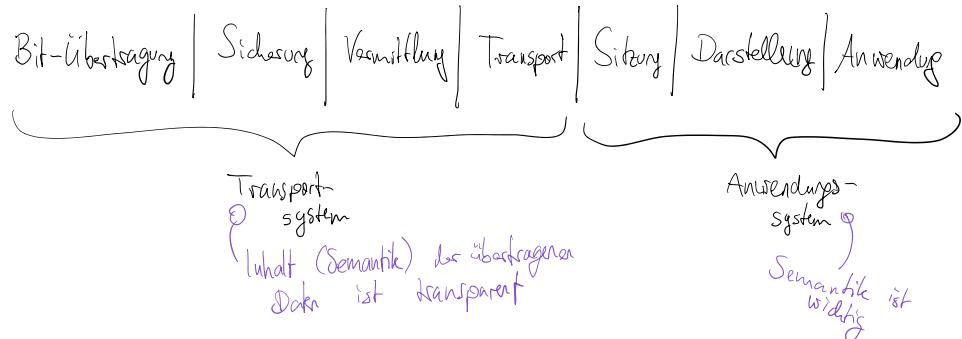


- VLAN: Virtuelles LAN, auf Ethernet-Ebene wird Datenverkehr logisch getrennt
  - ⇒ Sicherheit, Flexibilität, Performance
  - ▷ Interface-basiert
  - Switchanschlüsse sind als Gruppen von virtuellen Switches konfiguriert
  - ⇒ Verkehrsisolierung, Dynamische Zuweisung, Weiterleitung über Router im Switch
  - ▷ Mehrere Switches
- Verewendung spezieller VLAN-Trunks statt Ethernet-Rahmen (mit VLAN-Id Feld ...)

## 6 Netzarchitektur

- Geschichtete Architekturen: versch. Schichten implementieren jeweils einen Dienst, bauen auf Diensten tiefer Schichten auf
- Wie viele Schichten?
  - ▷ Eher wenige + kleine Dienste + geringe Interaktion zw. Diensten
  - ▷ Schichten nach Aufgabe trennen
- OSI - Referenzmodell: offen, aber keine Protokolle definiert (aber Problemstrukturierung)

□ Schichten:



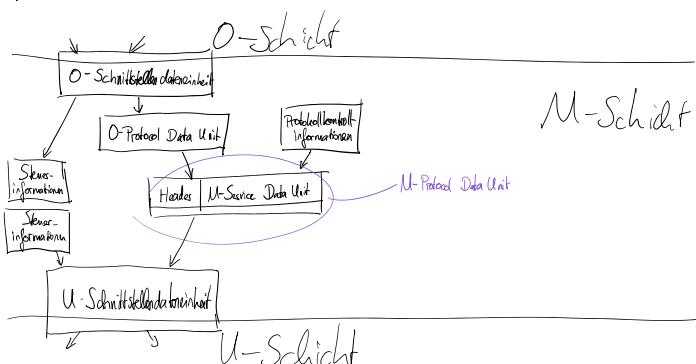
- ▷ Bitübertragungsschicht: Keine Pufferung, bietet keinen zuverlässigen Service, evtl. Übertragungsstörungen, Ziel: feste Übertragungsqualität
- ▷ Sicherungsschicht: Kommunikation zw. physischen Nachbarn, Fehlerbehebung von Bit-Schicht, Rahmen-Gliederung, Pufferung

- ▷ Vermittlungsschicht: Verknüpft Übertragungsabschnitte zu End-zu-End-Strecke, Wegwahl, Gerätadressierung, Multiplexing
- ▷ Transportschicht: Datenaustausch zwischen Anwendungen, + Fehlerbehebung, Pufferung, Transportrouten-Adressierung, Multiplexing
- ▷ Sitzungsschicht: Liefert Nichtunterbrechbarkeit, Ablaufsteuerung, Bereitstellung von Sitzungen
- ▷ Darstellungsschicht: Übersetzung in/out lokaler Syntax, einheitliche Datendarstellung
- ▷ Anwendungsschicht: Austausch anwendungsabhängiger Daten
- Kapselung der Daten: Information wird durch alle Schichten durchgereicht, dabei in jeder Schicht neu gekapselt  
Pro Kapselung: + Kontrollinformationen (Kopf/Header am Anfang, Anhang/Trailer am Ende)
  - Internet-Referenzmodell: nicht formal spezifiziert
    - „Rechner-Netzanschluss“ als altern. Name
    - Schichten: Bit-Übertragung/Sicherung, Vermittlung, Transport, Anwendung
    - Paketnahmen pro Schicht: Rahmen, Datagramm, Segment, Nachricht

## - Protokolle und Dienste

- Protokoll: Regeln und Formate zum Datenaustausch, innerhalb Schicht ( $\leftrightarrow$ ), je Gerät und Anwendung eine Protokollinstanz
- Dienst: innerhalb Schicht durch Zusammenwirken von Protokollinstanzen, über ganzes Kommunikationssystem hinweg
- Horizontale Kommunikation:  $\square \leftrightarrow \square$  zw. Kommunikationspartnern innerhalb einer Schicht
- Vertikale Kommunikation:  $\square \uparrow \downarrow \square$  zw. Schichten in einem System

## Zusammenspiel zw. Schichten



- Dienst: Bündelung zusammengehörender Funktionen, höchstens gekapselt in Dienstfunktionen

Dienstprimitiv: Einzdvorgänge einer Dienstfunktion Dienstarchitektur: 

↳ Grundtypen: Request, Indication, Response, Confirmation

↳ Beschreibung: Name (z.B. Vermittlung), Name d. Dienstleistung (z.B. Disconnect), Parameter (z.B. HTTP Get[Req](URL))

Dienstzugangspunkte stellen Schnittstellen dar, nämlich:

□ Dienstprimitiven (Zeitpunkt, Ort, Typ, Parameter) + zeitl. Ablauffestlegung

Unbestätigter vs. Bestätigter Dienst: Response/Confirmation oder nicht

Beispielablauf: Websiekaufzug

8. Ab 6-47 bis 6-53

## ⑦ Netz Sicherheit

- Angreifersmodell: Dolev-Yao-Angriff: überall im Netz, kann alle Kommunikationen abhören, eigene Pakete verschicken und fremde modifizieren, kein Zugriff auf Endsysteme
- Passive Angriffe: Abhören
- Aktive Angriffe: Pakete duplizieren und wieder einspielen, einfügen, löschen, modifizieren
- Schutzziele: CIA
  - Confidentiality (Vertraulichkeit): keine unautorisierte Informationsgewinnung
  - Integrity (Integrität)
    - starke Integrität: Daten können nicht unauthorisiert manipuliert werden
    - schwache Integrität: Daten können nicht unbemerkt unauthorisiert manipuliert werden
  - Availability (Verfügbarkeit)
    - oder
    - Authentizität: Angegebene Quelle ist korrekt. Dabei kann man Echtheit von Subjekten und Echtheit von Daten überprüfen
- Verschlüsselung
  - Kryptografie: Klartext  $\leftrightarrow$  Chiffertext
  - Symmetrische Kryptografie

▷ Blockchiffren: Datenblöcke fester Länge. Blöcke unabhängig voneinander verschlüsselt

▷ Stromchiffren: Bit-/Zeichenweises Verschlüsseln, Seed erzeugt Schlüsselstrom

$$\text{Verschlüsselung: } \text{Enc}_{\text{Schlüssel}_i}(\text{kla}_{\text{text}_i}) = \text{ciph}_{\text{text}_i} \equiv \text{kla}_{\text{text}_i} + \text{Schlüssel}_i \pmod{2}$$

$$\text{Entschlüsselung: } \text{Dec}_{\text{Schlüssel}_i}(\text{ciph}_{\text{text}_i}) = \text{kla}_{\text{text}_i} \equiv \text{ciph}_{\text{text}_i} + \text{Schlüssel}_i \pmod{2}$$

□ Asymmetrische Kryptographie: verwendet  $f(x)$  mit  $f^{-1}$  schwer berechenbar

z.B.: Integro-Faktorisierung, Diskreter Logarithmus, Elliptische Kurven

▷ RSA: basiert auf Integro-Faktorisierung

Modulare Mathematik:  $[(a \bmod n) \otimes (b \bmod n)] \bmod n = (a \oplus b) \bmod n$   
für  $\otimes \in \{+, -, *\}$

$$\Rightarrow (a \bmod n)^d \bmod n = a^d \bmod n$$

Arbeitet sehr langsam  $\Rightarrow$  nur für Schlüsselaustausch von symmetrischen Verfahren

(nicht verwechseln mit  $|m| < |n|$ !)

Nachricht  $m \in \{0, 1\}^k$  mit  $m < n$

$$\text{Verschlüsselung: } \text{Enc}_{K_E}(m) = c = m^e \bmod n$$

$$\text{Entschlüsselung: } \text{Dec}_{K_D}(c) = m = c^d \bmod n$$

Schlüsselerzeugung: Wähle zwei große Primzahlen  $p, q$

$$n := p \cdot q, \quad z := (p-1)(q-1)$$

Wähle  $e$  mit  $1 < e < z$  und  $e$  kongruent zu  $z$

Finde  $d$ , dass  $e \cdot d - 1$  durch  $z$  teilbar ist

$$\Rightarrow \text{Öffentliches Schlüssel: } K_E = (n, e)$$

$$\text{Privater Schlüssel: } K_D = (d)$$

□ Symmetrisch  
effizient, geringe Komplexität

vs.  
Assymetrisch

einfache Schlüsselverteilung  
 einfache digitale Unterschriften

- Integritätsüberprüfung: Man in the Middle Schaden verhindern

Eigenschaften von Hashfunktionen: Einwegfunktion, Schwache/Stärke Kollisionsresistanz, beliebige Nachrichtenlänge, feste Hashwertlänge, effizient

- Message Authentication Code (MAC): Nachricht wird gesucht, Hash wird an Nachricht angehängt, Überprüfung nur mit symmetrischen Schlüssel möglich  
⇒ Nachrichtenintegrität + Nachrichtenauthentizität
- Digitale Signatur: Unterschrift mittels asymmetrischer Kryptografie  
Dokument  $m$  wird mit privatem Schlüssel  $K_D$  signiert:  $SIG := \text{Sign}_{K_D}(m)$ ,  
Empfänger prüft mit öffentlichem Schlüssel:  $\text{Verif}_{K_E}(SIG) = m'$ , prüft  $m = m'$   
Umsetzung mit RSA:  $\text{Sign}_{K_D} := \text{Enc}_{K_D}$ ,  $\text{Verif}_{K_E} := \text{Dec}_{K_E}$ , also jeweils privater und öffentlicher Schlüssel gebraucht

- Authentifizieren des Kommunikationspartners  
Problem: Signatur sichert nur Nachrichtenintegrität, nicht Authentizität  
⇒ Digitale Zertifikate (von Dritter, CA: Certification Authority)  
ID-Zertifikat: bindet öffentl. Schlüssel an Namen / Identität

### - Beispiel Internet-Sicherheit

- PGP: Pretty Good Privacy
  - Wählbare Bestandteile: Prüfung von Vertraulichkeit und Integrität der Nachricht  
(Wahl eines symmetrischen Verfahrens)
  - und Prüfung von Authentizität von {Empfänger, Absender}  
(Wahl eines asymmetrischen Verfahrens)

### □ Beispiel Nachrichtensend

$$4 \text{ Schlüssel: } K_E^A, K_E^B, K_D^A, K_D^B \quad \begin{array}{ll} A = \text{Sender} & E = \text{Encode (privat)} \\ B = \text{Empfänger} & D = \text{Decode (öffentl.)} \end{array}$$

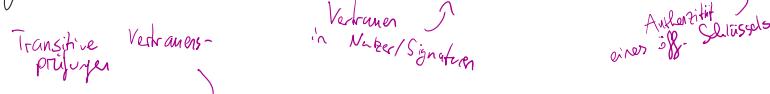
Nachrichtenversand:

- ▷ A verschlüsselt  $m$  symmetrisch mit zufälligem Schlüssel  $K_R$
- ▷ A verschlüsselt  $K_R$  asymmetrisch mit  $K_E^B$   
 $A \rightarrow B$  senden
- ▷ B entschlüsselt  $K_R$  asymmetrisch mit  $K_D^B$
- ▷ B entschlüsselt  $m$  symmetrisch mit  $K_E$

- Web of Trust

Statt zentraler vertrauenswürdigen Schlüsselserver: Dezentralisiert, transitive Schlüsselpfeilung

Kategorisiert Kontakte nach „Grad des Vertrauens“, „Grad der Authentizität“



Nutzer bauen Vertrauen zu Kontaktten auf, daraus wird Schlüsselaufhärigkeit (Key Legitimacy L) berechnet

$$L = \frac{\text{Signaturen geringen Vertrauens}}{x} + \frac{\text{Signaturen vollen Vertrauens}}{y}, \quad x, y \in \mathbb{Z}$$

Bereitung:  
 $L = 0 \Rightarrow$  nicht authentisch  
 $0 < L < 1 \Rightarrow$  teilweise authentisch  
 $1 \leq L \leq 1 \Rightarrow$  voll authentisch

Fazit: Anarchisch & dezentralisiert, öff. Schlüssel Pfeilung über Signaturketten von vertrauenswürdigen Nutzern. Aber: Insolubildung, Authentizität so stark wie schwächstes Glied, ...

## - Infrastruktursicherheit

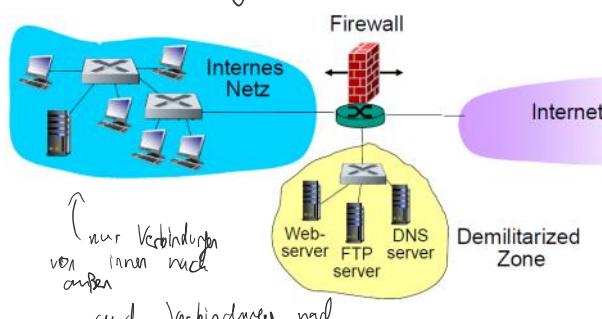
- o Firewall: Verhindert Eindringen unvertraulicher Pakete in Netzwerke, Paketfilterung, (passive) Zugriffskontrolle
- o Zustandslose Paketfilterung: Filtert basierend auf IP, TCP/UDP Ports, Transportprotokoll, ... (and Access - Control List ist ACL)
- o Zustandsbasierte Paketfilterung: Prüft z.B., ob TCP-Segmente zueinander passen
- o ACL: zustandslose Prüfung, Beispiel:

Aktion	Quell-IP	Ziel-IP	Protokoll	Quell-Port	Ziel-Port	Flags
Erlauben	222.22.16	Außerhalb 222.22.16	TCP	> 1023	80	any
Erlauben	Außerhalb 222.22.16	222.22.16	TCP	80	> 1023	ACK
Erlauben	222.22.16	Außerhalb 222.22.16	UDP	> 1023	53	---
Erlauben	Außerhalb 222.22.16	222.22.16	UDP	53	> 1023	---
Verwerfen	Alle	Alle	Alle	Alle	Alle	Alle

ggf. Erweiterung um Zustandsprüfung (z.B. Timeout)

Firewalls brechen End2End Beziehung auf (Application Layer Gateway).

Weitere mögl. Anwendung:



and Verbindungen nach  
inneren

- Grenzen:
  - ▷ Überlastungsrisiko (Single Point of Failure)
  - ▷ Appl. Layer Gateway Filterung aufwendig
  - ▷ Nicht 100% sicher, z.B. Angriffe durch legitime Nutzer

- Intrusion Detection and Prevention

- Intrusion Detection System IDS: Nutzt verschiedene Sensoren im Netz, um Korrelationen im System zu entdecken. Erkennt bekannte Angriffe.
- Intrusion Prevention System IPS: Wie IDS, aber Reaktion bei Angriff: z.B. Nutzer disconnecten, kritische Systeme abschalten, Firewall auf Notfallbetrieb, ...

Techniken zur Angreifserkennung:

- ▷ Deep Packet Inspection: Wie Antivirussystem auf Paketinhalt
- ▷ Anomaly Detection: versucht auch nicht bekannte Angriffe aufzudecken, kontinuierliches Monitoring, Alarm bei Abweichen von Normalverhalten, Machine Learning

Fazit: Sicherheit  
Sicherheit ist  
kein Zustand  
Prozess!